Table Of Contents

INTRODUCTION........................................................................................................ 7

QUICK TOUR ............................................................................................................ 9
   About Simulation Toolkit.......................................................................................... 10
   Simulation Toolkit Experience............................................................................... 16
   What is New in 6.0..................................................................................................... 19
   Simulation Toolkit Features.................................................................................... 20
      SNMP Agent Simulator Features......................................................................... 21
      TL1 Agent Simulator Features............................................................................... 24
      Network Simulator Features.................................................................................. 26
   Utility Tools ............................................................................................................ 32
   Contacting AdventNet.............................................................................................. 33

INSTALLATION GUIDE........................................................................................... 35
   System Requirements ............................................................................................. 36
   Installation and Startup Options............................................................................ 37
      Installing Simulation Toolkit .................................................................................. 38
      Starting Simulation Toolkit ................................................................................... 39
   Using License Manager ......................................................................................... 41
   Using Launcher......................................................................................................... 46
   Installing Service Packs......................................................................................... 48
   Uninstalling the Service Pack................................................................................ 51
   Uninstalling Simulation Toolkit............................................................................. 53
   Packages Bundled..................................................................................................... 54
      Package Structure ................................................................................................ 55
      Third Party Software ............................................................................................ 58
      MySQL .................................................................................................................. 59

RELEASE NOTES................................................................................................... 60
   Release Features ...................................................................................................... 61
   Known Issues ......................................................................................................... 73
   Limitations.............................................................................................................. 76

SNMP AGENT SIMULATOR GUIDE....................................................................... 77
   Getting Started....................................................................................................... 78
   Simulating Default Values..................................................................................... 80
   Configuring Values ................................................................................................. 82
      Recording from Real Agents................................................................................. 83
      Configuring MIB Values....................................................................................... 88
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuring a Device Added in the Device Tree ..................................</td>
<td>232</td>
</tr>
<tr>
<td>Deleting Devices in Device Tree .....................................................</td>
<td>235</td>
</tr>
<tr>
<td>Creating and Configuring a Network ...............................................</td>
<td>236</td>
</tr>
<tr>
<td>Creating a New Network ....................................................................</td>
<td>237</td>
</tr>
<tr>
<td>Template Devices and Bulk Configurations .......................................</td>
<td>242</td>
</tr>
<tr>
<td>Network View, List View and Topology View .....................................</td>
<td>248</td>
</tr>
<tr>
<td>Network Directory Structure .........................................................</td>
<td>251</td>
</tr>
<tr>
<td>Configuring Network Devices .........................................................</td>
<td>253</td>
</tr>
<tr>
<td>Configuring Router/Switch Properties .............................................</td>
<td>255</td>
</tr>
<tr>
<td>Configuring Behavior in SNMP Devices .............................................</td>
<td>257</td>
</tr>
<tr>
<td>Configuring Behavior in TL1 Devices ..............................................</td>
<td>278</td>
</tr>
<tr>
<td>Configuring SNMPv3 Security Settings .............................................</td>
<td>287</td>
</tr>
<tr>
<td>Configuring Run-Time Settings ......................................................</td>
<td>297</td>
</tr>
<tr>
<td>Adding Devices to the Network ......................................................</td>
<td>299</td>
</tr>
<tr>
<td>Bulk Addition of Scripts ..................................................................</td>
<td>302</td>
</tr>
<tr>
<td>Bulk Modification of IP address, Port and SNMP OIDs ..........................</td>
<td>304</td>
</tr>
<tr>
<td>Customizing the SNMP PDU in Network Simulator ..................................</td>
<td>308</td>
</tr>
<tr>
<td>Configuring IP addresses ..................................................................</td>
<td>311</td>
</tr>
<tr>
<td>Topology Editor ..............................................................................</td>
<td>315</td>
</tr>
<tr>
<td>Editing Operation of Devices .......................................................</td>
<td>317</td>
</tr>
<tr>
<td>Find a Device in the Network .........................................................</td>
<td>318</td>
</tr>
<tr>
<td>Viewing Device, Template and Network Info .......................................</td>
<td>319</td>
</tr>
<tr>
<td>Modifying Device Information .......................................................</td>
<td>321</td>
</tr>
<tr>
<td>Deleting Devices in the Network ....................................................</td>
<td>322</td>
</tr>
<tr>
<td>Deleting the Network .......................................................................</td>
<td>323</td>
</tr>
<tr>
<td>Saving the Network ..........................................................................</td>
<td>324</td>
</tr>
<tr>
<td>Starting and Stopping the Network ..................................................</td>
<td>325</td>
</tr>
<tr>
<td>Starting and Stopping Network Devices/Agents ..................................</td>
<td>327</td>
</tr>
<tr>
<td>Loading and Unloading a Network ....................................................</td>
<td>329</td>
</tr>
<tr>
<td>Porting Your Network ........................................................................</td>
<td>330</td>
</tr>
<tr>
<td>Automate the Network .......................................................................</td>
<td>332</td>
</tr>
<tr>
<td>Starting Multiple Network from Command Line ....................................</td>
<td>334</td>
</tr>
<tr>
<td><strong>IOS SIMULATOR GUIDE</strong> ...................................................................</td>
<td>336</td>
</tr>
<tr>
<td>Getting Started ..............................................................................</td>
<td>337</td>
</tr>
<tr>
<td>IOS Commands Supported ....................................................................</td>
<td>340</td>
</tr>
<tr>
<td>IOS commands supported in PIX Firewall .........................................</td>
<td>349</td>
</tr>
<tr>
<td>CLI Script API methods .....................................................................</td>
<td>351</td>
</tr>
<tr>
<td>Adding IOS Commands .......................................................................</td>
<td>356</td>
</tr>
<tr>
<td>Editing Scripts ..............................................................................</td>
<td>357</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td><strong>TFTP, FTP AND TELNET SUPPORT</strong></td>
<td>358</td>
</tr>
<tr>
<td>TFTP protocol support</td>
<td>359</td>
</tr>
<tr>
<td>FTP Protocol Support</td>
<td>363</td>
</tr>
<tr>
<td>Telnet Protocol Support</td>
<td>365</td>
</tr>
<tr>
<td><strong>MANAGING AGENTS AND NETWORK VIA RMI</strong></td>
<td>369</td>
</tr>
<tr>
<td>Overview</td>
<td>370</td>
</tr>
<tr>
<td>Managing the Agent / Network via RMI</td>
<td>372</td>
</tr>
<tr>
<td>RMI Tutorials</td>
<td>375</td>
</tr>
<tr>
<td><strong>RECORER GUIDE</strong></td>
<td>378</td>
</tr>
<tr>
<td>Network Recorder</td>
<td>378</td>
</tr>
<tr>
<td>Record and Replay Traps</td>
<td>382</td>
</tr>
<tr>
<td><strong>SIMULATING LARGE NETWORKS</strong></td>
<td>384</td>
</tr>
<tr>
<td><strong>PERFORMANCE GUIDE</strong></td>
<td>387</td>
</tr>
<tr>
<td><strong>SNMP TRAP STORMER</strong></td>
<td>389</td>
</tr>
<tr>
<td><strong>SNMP PROXY AGENT SIMULATION</strong></td>
<td>393</td>
</tr>
<tr>
<td><strong>TL1 GATEWAY NETWORK SIMULATION</strong></td>
<td>395</td>
</tr>
<tr>
<td><strong>MIGRATION GUIDE</strong></td>
<td>398</td>
</tr>
<tr>
<td>Migrating the Simulated SNMP Agent</td>
<td>399</td>
</tr>
<tr>
<td>Migrating the Simulated Network</td>
<td>400</td>
</tr>
<tr>
<td>Migrating the Simulated TL1 Agent</td>
<td>401</td>
</tr>
<tr>
<td><strong>TROUBLESHOOTING GUIDE</strong></td>
<td>402</td>
</tr>
<tr>
<td>TroubleShooting - General</td>
<td>403</td>
</tr>
<tr>
<td>TroubleShooting - SNMP Agent Simulator</td>
<td>407</td>
</tr>
<tr>
<td>TroubleShooting - Network Simulator</td>
<td>409</td>
</tr>
<tr>
<td><strong>USING TEST &amp; EDITOR TOOLS</strong></td>
<td>412</td>
</tr>
<tr>
<td>MIB Browser</td>
<td>413</td>
</tr>
<tr>
<td>Overview</td>
<td>414</td>
</tr>
<tr>
<td>Getting Started</td>
<td>416</td>
</tr>
<tr>
<td>Configuration</td>
<td>417</td>
</tr>
<tr>
<td>Setting Common Parameters</td>
<td>418</td>
</tr>
<tr>
<td>Setting MIB Parameters</td>
<td>422</td>
</tr>
<tr>
<td>MIB Operations</td>
<td>423</td>
</tr>
<tr>
<td>Loading MIBs</td>
<td>424</td>
</tr>
<tr>
<td>Unloading MIBs</td>
<td>429</td>
</tr>
<tr>
<td>Parsing MIBs</td>
<td>430</td>
</tr>
<tr>
<td>SNMP Operations</td>
<td>432</td>
</tr>
<tr>
<td>SNMP GET</td>
<td>433</td>
</tr>
<tr>
<td>SNMP GETNEXT</td>
<td>434</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>SNMP GETBULK</td>
<td>435</td>
</tr>
<tr>
<td>SNMP SET</td>
<td>436</td>
</tr>
<tr>
<td>Trap Handling</td>
<td>438</td>
</tr>
<tr>
<td>Viewing Traps</td>
<td>439</td>
</tr>
<tr>
<td>Parsing Traps</td>
<td>442</td>
</tr>
<tr>
<td>Creating Parser Files</td>
<td>448</td>
</tr>
<tr>
<td>Editing Parser Files</td>
<td>450</td>
</tr>
<tr>
<td>Table Handling</td>
<td>451</td>
</tr>
<tr>
<td>Retrieving Table Data</td>
<td>452</td>
</tr>
<tr>
<td>Adding Rows</td>
<td>455</td>
</tr>
<tr>
<td>Deleting Rows</td>
<td>457</td>
</tr>
<tr>
<td>Graphs</td>
<td>458</td>
</tr>
<tr>
<td>Line Graph</td>
<td>459</td>
</tr>
<tr>
<td>Bar Graph</td>
<td>462</td>
</tr>
<tr>
<td>Debugging and Decoding</td>
<td>464</td>
</tr>
<tr>
<td>Error Messages</td>
<td>467</td>
</tr>
<tr>
<td>FAQs</td>
<td>471</td>
</tr>
<tr>
<td>TL1 Message Builder</td>
<td>473</td>
</tr>
<tr>
<td>Getting Started</td>
<td>474</td>
</tr>
<tr>
<td>Startup Options</td>
<td>475</td>
</tr>
<tr>
<td>Main Screen</td>
<td>476</td>
</tr>
<tr>
<td>Actions</td>
<td>478</td>
</tr>
<tr>
<td>Loading Files</td>
<td>479</td>
</tr>
<tr>
<td>General Options</td>
<td>480</td>
</tr>
<tr>
<td>Working with Command Set</td>
<td>482</td>
</tr>
<tr>
<td>Input Message Definitions</td>
<td>483</td>
</tr>
<tr>
<td>Creating Input Message Definitions</td>
<td>483</td>
</tr>
<tr>
<td>Example</td>
<td>484</td>
</tr>
<tr>
<td>Response Message Definitions</td>
<td>487</td>
</tr>
<tr>
<td>Creating Response Message Definitions</td>
<td>487</td>
</tr>
<tr>
<td>Example</td>
<td>488</td>
</tr>
<tr>
<td>Autonomous Message Definitions</td>
<td>490</td>
</tr>
<tr>
<td>Creating Autonomous Message Definitions</td>
<td>490</td>
</tr>
<tr>
<td>Example</td>
<td>491</td>
</tr>
<tr>
<td>Editing Alarm Codes</td>
<td>493</td>
</tr>
<tr>
<td>Using Templates</td>
<td>494</td>
</tr>
<tr>
<td>Using Resource Bundles</td>
<td>496</td>
</tr>
<tr>
<td>Using Command Set Glossary</td>
<td>498</td>
</tr>
<tr>
<td>Working with Patterns</td>
<td>500</td>
</tr>
</tbody>
</table>
Introduction

AdventNet Simulation Toolkit is a comprehensive and versatile set of intuitive tools that simplify development, testing, and demonstration of network management products without requiring real devices. It provides unparalleled benefits and capabilities to simulate a network with SNMP, TL1, TFTP, FTP, Telnet and IOS manageable devices on a single PC, in addition to simulating standalone SNMP and TL1 agents. The Topology editor enables establishing inter-connections across routers, switches and other network devices and visualize the topological relationship between the devices.

AdventNet Simulation Toolkit 6 offers a completely new data model that enables creation of large networks in seconds and offers enhanced network performance. Since the simulated environment is fully controllable, complete testing of the management application can be done with 100% accuracy. Every tool is independent and you can use them for your specific requirements.

The various sections that are covered in this product documentation is given below. Please choose the section, based on your requirement, to learn more.

- The Quick Tour section gives an overview about Simulation Toolkit, the product experience, and the valuable features available with the product. Further, it provides the contact information and various other products of AdventNet.
- The Installation Guide gives you an overview of the system requirements and how to install the product. It also provides information on licensing, installing patches, the directory structure, the third-party software that are bundled with the product.
- The Release Notes section explains features available with the current release of the product. It also gives a list of Known issues and Limitations with the current release of the product.
- The SNMP Agent Simulator Guide explains in detail about the usage of the features supported in simulating a standalone SNMP agent.
- The TL1 Agent Simulator Guide explains in detail about the usage of the features supported in simulating a standalone TL1 agent.
- The Network Simulator Guide explains in detail about the usage of the features supported in simulating a network with SNMP and TL1 devices.
- The IOS Simulator Guide explains in detail about the Cisco IOS software simulation supported in the Network simulator.
- The TFTP, FTP and Telnet section explains the TFTP client and server protocol, FTP client protocol and Telnet support implemented in Simulation Toolkit, to enable transfer of files between the manager and the agents.
- The Managing Agents and Network via RMI section explains the RMI support in SNMP Agent.
Simulator, TL1 Agent Simulator and Network Simulator, to manage the simulated agents via RMI, from a remote client program. The RMI client APIs defines the methods in the simulator which can be accessed via RMI.

- **The SNMP Trap Recorder** explains the usage of the Trap Recorder tool to record SNMP v1 and SNMP v2c traps. The **Network Recorder** explains the usage of the Network Recorder tool to record real networks.

- The topic **Simulating Large Networks** explores the capabilities of Simulation Toolkit to simulate a large network with ease at the shortest possible time.

- **The Performance Guide** gives the performance statistics of the Network Simulator.

- **SNMP Trap Stormer** explains the usage of this exclusive tool to configure and send SNMP v1/v2c traps.

- **SNMP Proxy Agent Simulation** explains the Proxy feature to simulate such scenarios, where there are multiple SNMP agents with different community strings.

- **TL1 Gateway Network Simulation** explains the Gateway NE feature to simulate multiple NEs connected via a GNE and also simulate the GNE.

- **Migration Guide** is designed to aid users of AdventNet Simulation Toolkit 5 in migrating to the latest AdventNet Simulation Toolkit 6 release version.

- **TroubleShooting Guide** helps you in identifying and correcting some common problems that you might encounter while working with the different tools available in the product.

- **Using Test and Editor Tools** section explains about the functionality of each test tool and editor tool that is bundled with this product.

- **FAQs** section lets you familiarize yourself with the frequently asked questions.

Send your valuable suggestions and feedback to simulator-support@adventnet.com
Quick Tour

About Simulation Toolkit provides an overview on the need for the product, what can be done with the product, the target market and the benefits and capabilities which make Simulation Toolkit, the most comprehensive tool in the field of Simulation.

Simulation Toolkit Experience briefly talks about how the toolkit can be used for simulating a network, in addition to simulating a Standalone SNMP and TL1 Agent This section also explains about how to work with the various tools available with this product.

Simulation Toolkit Features provides a list of all the features available with the product. The features are listed in separate topics for SNMP, TL1, and Network Simulators.

What Is New in This Release gives you brief explanation about the new features and enhancements available in the latest release of AdventNet Simulation Toolkit.

Contacting AdventNet furnishes complete details about contacting AdventNet, its Sales team, and availing the free technical support provided by AdventNet.

AdventNet's Other Products provides a comprehensive range of network management products that can address all the network and systems management needs. This document briefs the various products of AdventNet Simulation Toolkit.
About Simulation Toolkit

- Introduction: What Is Simulation Toolkit?
- Why Simulation Toolkit?
- Simulation Toolkit Uses
- Simulation Toolkit Benefits
- Simulation Toolkit Capabilities
- Simulation Toolkit Product Suite

Introduction: What Is Simulation Toolkit?

AdventNet Simulation Tool comprises agent simulator and network simulator for testing, training, and demonstration of network management applications. The network simulator enables network simulation of SNMP (v1, v2c, v3), TL1, TFTP, FTP, Telnet, and IOS devices, on a single PC. The IOS Simulator integrated with the Network Simulator, adds the capability to respond to Cisco IOS commands over Telnet.

Simulation Toolkit provides Network Recorder and Trap Recorder to record and replay real SNMP networks and traps, allows bulk addition of devices with unique IP address and port, enables run-time configuration of devices both on an individual and collective basis, allows advanced modeling of agent/network behavior and trap generation. The agent management through RMI provides solution for automated testing.

Ability to simulate 50000+ agents simultaneously for scalability testing, dynamic configuration of IP address, bulk modification of device values, utilities to control agents from command line, fault management testing using Trap Stormer and easy-to-use GUI enables full-fledged simulation of large networks.
Why Simulation Toolkit?

Testing a network management application in realistic and peak-load environments with multi-vendor network devices is critical to its commercial success. Sales demos are essential to demonstrate the capabilities of the software in large, complex, network scenarios, or at customer premises. Users need to evaluate the tool exhaustively before buying. However, in real life, the testing, demonstration and evaluation of NMS applications involve the following hurdles.

Challenges in Building and Testing Management Applications

- Procurement of complex multi-vendor network devices for testing is prohibitively expensive.
- Complete and reliable testing is not possible due to the various types of agents in the devices and their different versions.
- Critical test scenarios, such as device failures and agent crashes are difficult to reproduce with live devices.
- Trained people with expertise on various devices and technologies are required to evaluate the network management tools.
- Networks are complicated and extremely expensive, and most vendors cannot afford to set up a demo network in every sales office.

AdventNet Simulation Toolkit, a software suite which comprises SNMP and TL1 agent and network simulator tools is designed to overcome the above hurdles.

AdventNet Simulation Toolkit offers,

- An integrated network design studio to create a virtual multi-vendor network environment with SNMP and TL1 devices, on a single PC.
- Simulation of any type of device by adding vendor specific MIBs/TCS and simulating values for the same.
- Creation of realistic and flexible networking scenarios for effective testing, training and evaluation.
- A portable simulation environment, enabling easy demonstration of management applications.

Simulation Toolkit Uses

Portable Product Demonstrations

Simulation Toolkit helps to create different network scenarios instantly. The network configurations are portable across all OS. This enables marketing and sales to set up powerful "live" demonstrations of their network management products at trade shows or at customer premises, easily and effectively. Reduced demo set up time and absence of bulky equipments results in reduced cost of sales.

Evaluation before Purchase

Users can easily yet exhaustively evaluate new management products prior to purchase and customize them after purchase. The Jython script APIs bundled with the product can be used to evaluate the capability of the management products.

Quick and Efficient Support

With Simulation Toolkit, the support staff do not have to visit the customer premises for problem reproduction. Its recording tool can record the customer's network environment and reproduce the problem quickly to provide fast and efficient support.
Virtual Multi-Vendor Network Environment on a Single PC

Developers and testing departments can test management applications by simulating large, complex network environments with SNMP and TL1 devices. Full control over the values returned by the SNMP and TL1 agents and the ability to generate SNMP traps and TL1 autonomous messages on demand, enable thorough testing of applications. This ensures greater product quality.

Efficient Training

Simulations can be used to train network operators, administrators, and technicians. Training groups can create realistic and flexible networking scenarios by setting up user defined Jython script. The IOS Simulator will serve as an effective self-study tool for learning about Cisco device management by interacting with Routers and Switches.

Simulation Toolkit Benefits

Reduces Product Development Costs

Simulation Toolkit has a positive impact in reducing the cost of developing, testing, and delivering management applications. It reduces the investment in equipment, support infrastructure, and testing. By facilitating easy demonstrations, it reduces the cost of selling and enhances profitability of the business.

Reduces "Time-to-Market"

Management application developers can develop their applications in parallel with agent development through simulation of device prototypes. This enables quick implementation of the management products, thus significantly shortening the "time-to-mar-ket" and the associated costs.

Efficient Network Management Staff

Customized simulations can be used to train network operators, administrators, and technicians. Efficient training results in well trained network management staff with better understanding of the network management products and protocols.

Delivers High Quality Management Applications

Simulation of complex network scenarios enables complete testing of network management applications. This enhances product quality and results in a reliable and rugged management applications.

Enables Easy and Successful Demonstrations

Portable device configurations ease the task of setting up customer demos to show the capabilities of the management application. With options to enable different permutations and combinations of network configurations rapidly, the Simulation Toolkit helps in such presales activities at trade shows and customer locations.
Simulation Toolkit Capabilities

Simulating Real Networks

The **Network Recorder** records real time network devices simultaneously. The recorded network can be replayed instantly in the Network Designer. The recorded configuration can also suitably be modified to create test configuration variations.

The **Trap Recorder** records SNMPv1 and SNMPv2c traps by listening for traps at the specified port. The output of the trap recorder will give you the traps in the order they were received, with time stamps and the variables included in the trap PDU. The recorded traps can be stored in XML files and replayed in the SNMP agent simulator and network simulator as request-based, threshold-based or time-based traps.

Simulating Dynamic Network Behavior

Simulation Toolkit can simulate addition of new devices (including bulk additions), start and shutdown of devices, change of read-only and read-write variables, generation of traps and autonomous messages, addition and deletion of rows dynamically for provisioning. These simulations can be enabled dynamically without disrupting the simulation process and thus helps to create a real-life scenario with high scalability. The toolkit enables querying of the same simulated device by multiple network management applications.

Dynamic Virtual IP Configuration

Simulation Toolkit provides the facility to dynamically configure IP addresses when starting each agent instance in the network. This facility is supported in Windows NT, 2000, XP, 2003 and Linux / Solaris OS.

Jython Scripting Support

Simulation Toolkit provides an intuitive GUI to model and define inter-relationship among MIB variables, among TL1 commands, and across devices in the network. The toolkit offers powerful built-in SNMP and TL1 script APIs to automate a complete network environment. The intuitive UI and powerful script APIs, combined with the flexible scripting capabilities of Jython, offers a highly versatile environment in simulating advanced network behavior and trap generation. This enables complete control of the virtual network environment.

Automated Network Environment

Simulation Toolkit provides an automated network environment. The Network Simulator’s Automation Wizard enables you to configure the actions that needs to be performed on the network devices, from the UI and run the automated network from command line.

Network Topology

Simulation Toolkit supports the topology wizard to visualize interconnections between the devices in the network. You can connect new devices or disconnect existing devices in the topology.

Fault Management Testing Using Trap Stormer

Simulation Toolkit offers the Trap Stormer, an exclusive tool to storm SNMPv1/v2c traps at burst and normal modes. Trap storms can be generated to validate the effective event handling of the manager application.
TFTP, FTP and Telnet Support

Simulation Toolkit supports TFTP client and server protocol and FTP client protocol, to facilitate transfer of files between the client and server using TFTP or FTP. This facility can be used for purposes like statistics collection, software download, upload and download of configuration files etc. You can also create your own SNMP devices with Telnet support and execute telnet commands on the simulated devices.

Cisco IOS Software Simulation

Simulation Toolkit supports Cisco IOS Software Simulation as part of its existing Network Simulation suite. It allows you to simulate Telnet Enabled Cisco routers and Cisco Switches. CLI support is provided to access the Cisco IOS Software

Controlled Environment via RMI

RMI (Remote Method Invocation) is implemented in Simulation Toolkit to control the simulated SNMP and TL1 agent behavior from a remote client program. RMI's unique capabilities to load and execute user-defined tasks at runtime, helps in test case automation of manager applications.

Pack and Play

Any network created in Simulation Toolkit can be packaged and installed in any other system where Simulation Toolkit has been installed. Similarly, new device types configured in the simulator can be packaged and installed in any other system. This is useful to give demonstration of management applications at customer premises or during trade shows without the need for configuring and carrying of devices.

Platform-Independent Network Environment

Simulation Toolkit creates the complete virtual network environment that can run on any industry-standard Intel-based PC or Sun Sparc with a variety of network interfaces. It can even run on the same machine that runs the management application. Large scale enterprise networks can run simultaneously on a single PC.

Scalability

With Simulation Toolkit, the number of agents that can be run simultaneously on a single PC are:

10000 in Windows, 50000+ in Linux, 4096 in Solaris. This enables simulation of large enterprise networks. Simulation Toolkit's extensibility and scalability, provide the opportunity to create network environments that would be impractical to create in a physical lab environment.

Simulation Toolkit Product Suite

The main components which make up the AdventNet Simulation Toolkit and provides a complete simulation environment are:

SNMP Agent Simulator

To simulate an SNMPv1/ v2c/v3 agent (or device) with user configured data or recorded agent values. Supports Jython-based scripting to model agent behavior and express inter-relationship among MIB variables.
TL1 Agent Simulator

To simulate a TL1 network element with user configured data. Supports Jython-based scripting to model agent behavior and express inter-relationships between TCS commands.

Network Simulator

To design and simulate an entire network with SNMP, TL1, TFTP, FTP, Telnet and IOS manageable devices, on one Intel-based PC. Supports Jython-based scripting to model real time network behavior and express inter-relationships among MIB variables and across agents in the network.

SNMP Trap Stormer

To configure and send traps, to test the reliability of your management application for receiving any number of traps at the specified time interval.

Network Recorder

To record real SNMP networks. The recorded network can be instantly replayed in the network simulator.

Trap Recorder

To record real traps. The recorded traps can be replayed in the trap configuration dialog of the SNMP Agent Simulator and Network designer tools.

Manager Tools

- SNMP MIB Browser: SNMP Manager application.
- TL1 Craft Interface: TL1 Manager application

Editor Tools

- SNMPv3Config Tool: SNMP v3 configuration tool to configure v3 users for the SNMP v3 agents.
- TL1 Message Builder: TL1 Editor Tool to create and edit TCS files.
Simulation Toolkit Experience

- Overview
- Simulating an SNMP Agent
- Simulating a TL1 Agent
- Simulating a Network

Overview

AdventNet Simulation Toolkit enables you to simulate your device rapidly with ease and in a short period of time. To accomplish this task, the toolkit provides a complete range of easy-to-use visual tools that makes simulation simple right from specifying the input for simulation until packaging and installing the simulated agent.

The components of Simulation Toolkit, work together to shield the complexity of the underlying syntax and structures specific to each protocol. This enables enterprises, device vendors, management software developers, service providers and end users to concentrate on what matters - creating high quality simulation scenarios.

The choice of simulations provided by Simulation Toolkit in this release are Standalone SNMP Agent Simulation, Standalone TL1 Agent Simulation, Network Simulation with SNMP, TL1, TFTP, FTP, Telnet and IOS manageable devices. The usage of the tools available with the toolkit depends on the type of simulation you would like to develop.

The diagram given below provides you an overview of the Simulation Toolkit product:

Given below is a brief description of each simulation type, how and where the tools are put to use in the development and testing of each of these simulation choices supported by the Toolkit.
Simulating an SNMP Agent

The SNMP Agent Simulator is used to simulate a Standalone SNMP Agent and is purely SNMP specific. It supports SNMPv1/v2c/v3 versions. Now, let us walk through the steps involved in Simulating a standalone SNMP Agent

Specifying the Input to the SNMP Agent Simulator

The first step in simulating an SNMP agent, is to load the MIB file or an existing Configuration file in the SNMP Agent Simulator. When you have completed loading the MIBs and simulating default values, a basic SNMP simulation is created.

Configuring Values

Any variation on this basic simulation can be created for use with the SNMP Agent Simulator. You can configure scalar and table values, record real agents, populate the SNMP table, configure traps and informs, configure real time behavior of devices using Jython scripts, simulate error conditions and scenarios and so on. In case of SNMPv3, the v3 config tool is used to configure v3 users.

Testing the Simulated Agent in MIB Browser

The MIB Browser bundled with the toolkit, can be used as SNMP Manager to test the simulated SNMP agent behavior.

Refer to the section, Simulating an SNMP Agent, to learn more.

Simulating a TL1 Agent

The TL1 Agent Simulator is used to simulate a Standalone TL1 Agent which confirms to the TL1 protocol. Now, let us walk through the steps involved in Simulating a standalone TL1 Agent

Specifying the Input to the TL1 Agent Simulator

The first step in simulating a TL1 agent, is to load the TCS file or an existing Configuration file in the TL1 Agent Simulator. A basic TL1 agent simulation is created, based on the parameter names and syntax defined in the TCS file.

You can define your own TCS file using TL1 Message Builder, bundled with the product.

Configuring Values

Any variation on this basic simulation can be created for use with the TL1 Agent Simulator. You can configure the parameters defined in the TCS file for Input Command Messages, acknowledgments, Output Response Message and Autonomous Messages. It also supports to configure scripts to define agent behavior, simulate delayed response, error response, acknowledgment for specific responses, and configure echo requests.

Testing the Simulated Agent in TL1 Craft Interface

The TL1 Craft Interface bundled with the toolkit, can be used as TL1 Manager, to test the simulated TL1 agent behavior.

Refer to the section, Simulating a TL1 Agent, to learn more.
Simulating a Network

The Network Designer integrated with Simulation Toolkit helps you to simulate a network with SNMP v1/v2/v3, TL1, TFTP, FTP, Telnet and IOS manageable devices. The **Cisco IOS Software Simulation** allows you to simulate Cisco IOS commands in Cisco routers and switches. CLI support is provided to access the Cisco IOS Software. Each simulated device can support its own MIBs, TCS File, data and IP Address.

Now, let us walk through the steps involved in Simulating a Network.

**Creating the Network in Network Designer**

The Network Designer tool allows you to **create a new network** in two ways:

- Create a blank new network and drag and drop devices into the network
- Create a network by adding multiple devices using network design wizard.

You can also record real SNMP networks using **Network Recorder** and load it directly in the Network Designer.

Each device in the network must have a unique IP address and port number combination.

**Configuring Network Devices**

Each device in the network can be simulated with different agent values, traps and behavior. In case of a TL1 device, you can simulate the device by configuring values for the parameters defined in the TCS file. You can also configure request based and timer based autonomous messages.

**Testing the Manager Application**

The simulated network can be used to test your manager applications

Refer to the section, **Simulating a Network**, to learn more.
What is New in 6.0

With the goal of constantly building upon and improving the management simulation experience, AdventNet Simulation Toolkit 6 offers a completely new data model that enables creation of large networks in seconds and offers enhanced network performance. In addition, it offers new tools, utilities and numerous enhancements over the earlier versions. This section provides a brief description of the key features added to the various modules of Simulation Toolkit 6.

Network Simulator

- **New Template based model of device creation and data storage**
  Simulation Toolkit 6 is based on a completely new data model. The new template based model of device creation and device configuration offers the following benefits:
  - Enables creation of large networks in seconds.
  - Occupies very less storage space of MySQL data for the network devices.
  - Enables template based configuration and modification of devices. (i.e) if 50 devices are created using Template device "A", any bulk modifications can be achieved by modifying this template device.
  - Offers enhanced network performance.

- **GUI changes enhances the user experience with the product**
  - Intuitive GUI to enable template based device configuration of table rows, traps, scripts, MIB(s) loading/ unloading, and protocol specific parameters, in addition to bulk modification feature.
  - Multiple device selection for Start / Stop agent.
  - Option to view the network devices as network device images or in the form of a list.
  - Option to view all the inform response in intuitive table GUI.
  - Separate wizard to specify 'unique oid values' while creating the network.
  - Option to enable/disable the display of device library in the network designer.
  - Simplified network creation with automatic generation of IP address, the next time you create a network.
  - Simplified GUI to record real agent values.
  - Option to view the Template tree or the Device Library.
  - Multiple device selection for deletion, when not required.

- **Complete SNMP v2, v3 Inform Support**
- **Support to package and install individual /multiple devices in the device tree.**
- **Option to execute timer scripts after specified time after agent startup.**
- **New Cisco IOS Commands added - EIGRP router mode and Vlan mode.**
- **Command Line utility to create SNMP WALK file.**

SNMP Agent Simulator

- **Complete SNMP v2, v3 Inform Support**
- **Option to execute timer scripts after specified time after agent startup.**

General

- **New SNMPv3 configuration tool** with intuitive GUI, to simplify v3 configurations.
- **Migration tools** to migrate the SNMP agent, TL1 agent and network created in release 5 version to the current version.
Simulation Toolkit Features

- SNMP Agent Simulator Features
- TL1 Agent Simulator Features
- Network Simulator Features
- Other Utilities
## SNMP Agent Simulator Features

The following features are supported in SNMP Agent Simulator

<table>
<thead>
<tr>
<th>General</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNMP Versions</td>
</tr>
<tr>
<td>• SNMP v1</td>
</tr>
<tr>
<td>• SNMP v2c</td>
</tr>
<tr>
<td>• SNMP v3</td>
</tr>
<tr>
<td>Storage Option</td>
</tr>
<tr>
<td>• MySQL Database</td>
</tr>
</tbody>
</table>

### SNMP Agent Configuration

(The SNMP Agent configurations listed below are supported both in SNMP Agent Simulator and Network Simulator. The ** mark denotes configurations not supported in Network Simulator.)

<table>
<thead>
<tr>
<th>Simulation Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Constant.</td>
</tr>
<tr>
<td>• Random.</td>
</tr>
<tr>
<td>• Linear</td>
</tr>
<tr>
<td>• Exponential</td>
</tr>
<tr>
<td>• Wave</td>
</tr>
<tr>
<td>• Sine Wave</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scalar/Table Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Configure scalar/table values from the Network Designer UI</td>
</tr>
<tr>
<td>• SYSUPTIME macro is supported to provide the sysUpTime value of the agent.</td>
</tr>
<tr>
<td>• Add and Modify a single row.</td>
</tr>
<tr>
<td>• Add Multiple rows in a single shot.</td>
</tr>
<tr>
<td>• Modify a column value for all the rows and selected row.</td>
</tr>
<tr>
<td>• Find option for finding Instance and the row numbers.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Trap and Inform Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Configurable v1/v2c/v3 trap generation.</td>
</tr>
<tr>
<td>• Configurable request-based, timer-based and threshold-based trap generation.</td>
</tr>
<tr>
<td>• Configurable static and dynamic varbinds. SysUpTime and CurrentTime macros supported in vabind configuration.</td>
</tr>
<tr>
<td>• Option to configure agent IP address and manager details of traps, globally.</td>
</tr>
<tr>
<td>• Send traps to multiple managers and ports.</td>
</tr>
<tr>
<td>• Generation of default traps defined in the MIB **</td>
</tr>
<tr>
<td>• Configurable SNMPv2c and v3 Inform generation.</td>
</tr>
<tr>
<td>• Configurable request-based, timer-based and threshold-based inform generation</td>
</tr>
<tr>
<td>Feature</td>
</tr>
<tr>
<td>----------------------------------------------</td>
</tr>
</tbody>
</table>
| **Behavior Simulation**                      | • Automatic generation of request-based, threshold-based, timer-based scripts in Jython.  
• Execute timer based scripts at agent startup or when agent stops or at specified time periodically or after specified time after agent startup, periodically.  
• Powerful build-in script APIs which can be used to format SNMP responses and trap messages. |
| **Error Simulation**                         | • Simulate SNMP v1/v2/v3 errors conditions.  
• Jython scripts can also be used to configure error conditions and scenarios. |
| **Record Real Agents**                       | • Record and replay real agent values and traps to create simulations of actual devices.  
• Once data are captured from a device, any variation on this basic simulation can be created for use with the SNMP Agent Simulator. |
| **Customize the Request and Response SNMP PDU** | • Customization of the request SNMP PDU  
• Customization of response SNMP PDU |
| **Run Time Settings**                        | • SNMP agent version and port  
• SNMPv3 Engine ID  
• Manage using RMI  
• Read/Write community  
• Debug Message Options: Disabled, Fatal, Error, Warning, Info, Debug, Trace  
• Automatic Save of runtime settings. |

**Operations**

<table>
<thead>
<tr>
<th>Operations</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Loading and Unloading**                      | • MIBs  
• Configuration File |
| **Save Configuration**                         | • To save the agent configuration in the database |
| **Start the simulated SNMP agent**             | • From UI  
• From Command line |
| **Stop the simulated SNMP agent**              | • From UI  
• From Command line  
• From Remote |
### RMI Support

- RMI (Remote Method Invocation) is implemented in the SNMP agent simulator to control the simulated agent behavior from a remote client program.
- The RMI client APIs, defines the methods in the simulator, which can be accessed via RMI from the client program.

### Value Added Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find a node in the MIB</td>
<td>--</td>
</tr>
</tbody>
</table>
| RMI Support                      | - RMI (Remote Method Invocation) is implemented in the SNMP agent simulator to control the simulated agent behavior from a remote client program.  
  - The RMI client APIs, defines the methods in the simulator, which can be accessed via RMI from the client program. |
| Package and Install SNMP Agent   | Option to package the SNMP agent configuration and install it in any system where Simulation Toolkit is installed. |
| Initialize Services              | Option to enable/disable the execution of Trap/Inform messages and behavior scripts at agent runtime. |
| Command Line SNMP Agent Simulator| To start the simulated SNMP Agent from command line.                        |
| Proxy Agent Simulation           | Simulate multiple SNMP agents with different community strings.              |
| Re-simulation                    | Simulate the value of the newly added groups in the MIB without modifying the existing configuration. |
# TL1 Agent Simulator Features

The following features are supported in TL1 Agent simulator:

<table>
<thead>
<tr>
<th>General</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Option</td>
</tr>
<tr>
<td>• MySQL Database</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TL1 Agent Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>(The TL1 Agent configurations listed below are supported both in TL1 Agent Simulator and Network Simulator. The ** mark denotes configurations not supported in Network Simulator)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Command Message Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Multiple Access Identifier</td>
</tr>
<tr>
<td>• Multiple Message Payload Block</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Acknowledgment Message Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• In Progress</td>
</tr>
<tr>
<td>• PrintOut Follows</td>
</tr>
<tr>
<td>• Repeat Later</td>
</tr>
<tr>
<td>• No acknowledgment</td>
</tr>
<tr>
<td>• All Right</td>
</tr>
<tr>
<td>• No Good</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output Response Message Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Completion Codes : Compld, Deny</td>
</tr>
<tr>
<td>• Multiple Valid and Error Response lines in Text Block</td>
</tr>
<tr>
<td>• Response Line Type: Quote, UnQuote</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Autonomous Message Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Alarm Codes</td>
</tr>
<tr>
<td>• Multiple Autonomous Response Lines in Text Block</td>
</tr>
<tr>
<td>• Response Line Type: Quote, UnQuote</td>
</tr>
<tr>
<td>• Configurable Request-based, Timer-based Autonomous message generation for a command code - AID - MPB combinations.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Echo Request Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>NE responds with the request, instead of the actual output response message.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Behavior Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Configure Request-based scripts on a Command Code - AccessId - MessagePayload combination.</td>
</tr>
<tr>
<td>• Powerful Built-in Script APIs</td>
</tr>
</tbody>
</table>
| **Simulate Random Variations in Response** | • Delayed Response  
• Error Response  
• Acknowledgment Message |
| --- | --- |
| **TL1 Security** | • User Related Security is implemented as per the specifications in GR 815 and TR 835.  
• User Related Security authorizes the establishment of a session (i.e., login) and its continuation until logoff. |

### Operations

<table>
<thead>
<tr>
<th>Save Configuration</th>
<th>• To save the agent configuration in the database</th>
</tr>
</thead>
</table>
| Start simulated TL1 agent | • From UI  
• From Command line |
| Initialize Services | • Option to enable/disable the execution of Autonomous messages at agent runtime.  
• Option to enable/disable the execution of Behavior scripts at agent runtime. |
| **RMI Support** | RMI (Remote Method Invocation) is implemented in the TL1 agent simulator, to control the simulated agent behavior from a remote client program. The TL1 RMI client API, defines the methods in the simulator, which can be accessed via RMI. |

### Utilities

| Gateway NE Simulation | • Simulates NEs connected via a GNE  
• Simulates the GNE.  
• The GNE will also route the Autonomous Messages from the NE back to the managers connected. |
| --- | --- |
| Resimulation | • Resimulate values without modifying the existing configuration,  
  o if new Command Code(s) and Autonomous codes are added to the TCS file  
  o if existing Command codes and Autonomous codes are deleted |
| **Built-in and Self Monitoring Commands** | • The most common input commands and self-monitoring commands is already implemented in the TL1 agent.  
• The commands are defined in `tl1builtincommand.tcs` file. This file can be loaded in the TL1CraftInterface tool to query the TL1 agent. |
| **Command Line TL1 Agent Simulator** | To start the simulated TL1 agent from command line. |
## Network Simulator Features

The following features are supported in Network Simulator:

<table>
<thead>
<tr>
<th>General</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supported Protocols</td>
</tr>
<tr>
<td>• SNMPv1, SNMP v2c, SNMP v3</td>
</tr>
<tr>
<td>• TL1</td>
</tr>
<tr>
<td>• TFTP client and server</td>
</tr>
<tr>
<td>• FTP client.</td>
</tr>
<tr>
<td>• Telnet</td>
</tr>
<tr>
<td>• Cisco IOS Software Simulation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Integrated Network Designer Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Components of Network Designer</td>
</tr>
<tr>
<td>o Device tree with pre-configured devices like Cisco routers/switches, Printer, Hosts, Cisco Pix Firewall, TL1 Acme</td>
</tr>
<tr>
<td>o Template tree view after device creation.</td>
</tr>
<tr>
<td>o Network area displaying the network devices with unique device name, ip address, port number and protocols supported.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number of Devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>• 10000 devices in Windows OS</td>
</tr>
<tr>
<td>• 50000+ devices in Linux (changes to be done in linux conf files)</td>
</tr>
<tr>
<td>• 4096 devices in Solaris</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Storage Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>• MySLQ Database</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Device Tree Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Adding, deleting group nodes</td>
</tr>
<tr>
<td>• Adding, deleting devices</td>
</tr>
<tr>
<td>• Adding SNMP devices using MIB/Configuration file</td>
</tr>
<tr>
<td>• Adding SNMP devices with SNMP WALK file.</td>
</tr>
<tr>
<td>• Adding Router/Switch/PixFirewall devices with CLI support.</td>
</tr>
<tr>
<td>• Adding TL1 devices using TCS/Configuration file</td>
</tr>
<tr>
<td>• Configuring network devices</td>
</tr>
</tbody>
</table>
### Creating a Network
- By drag and drop of devices from device tree.
- By adding bulk devices of same type using network design wizard
- By adding bulk devices of different device types, at random
- Adding new devices individually or in bulk, to existing network.
- For SNMP devices, the IP address assigned to the device is added to the `ipaddrTable`, and the unique name of the device is updated as the `sysName` in the system group of RFC1213-MIB.

### Unique Value Configurations
Option to configure unique OID values while creating a network. The following options are provided:
- By increment the given values
- By assigning ipaddress/unique name macro
- By assigning random values.
- By assigning constant values.

### Virtual IP Configuration
- Supports dynamic virtual IP configuration for the simulated devices, in Windows NT, 2000, XP, 2003 and Linux/Solaris OS.

### Network Device Configuration

#### SNMP Agent Configurations
- Refer to SNMP Agent Simulator Features : SNMP Agent Configurations section, for configuring SNMP network devices
- Configure Behavior Across Agents in the Network

#### TL1 Agent Configurations
- Refer to TL1 Agent Simulator Features : TL1 Agent Configurations section, for configuring TL1 network devices.

#### IOS Configurations
- Start/Stop CLI Agent (Cisco IOS Simulation) individually.
- Set user name and passwords to log into simulations.
- Supports User, Privileged, Configuration, Interface configuration, Router and Vlan modes..
- Script Editor to view and edit IOS command behavior scripts.
- Command customization with extensive script APIs.
- Command Configurator to define new IOS commands.
- Commands Implemented:
  - Set passwords, ip addresses, clock rates, hostname etc.
  - Enable/disable cdp service, configure cdp holdtime and timer.
  - Display cdp and flash information.
  - Configure access-list, display access-list configurations.
  - Configure rmon alarms and events, remove alarms/events and display active alarms and events.
  - Enable/disable SNMP trap service, configure recipients of the SNMP notification operation.
<table>
<thead>
<tr>
<th>Telnet Configurations</th>
<th>Telnet Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>o Configure SNMP community, contact, location, packetsize etc.</td>
<td>• Commands Implemented</td>
</tr>
<tr>
<td>o Specify the MTU value for the network interface node</td>
<td>o ls</td>
</tr>
<tr>
<td>o Enable RIP (Routing Information Protocol), configure RIP for a specific router.</td>
<td>o date</td>
</tr>
<tr>
<td>o Configure Vlan for Cisco3750 and Cisco6500 Switches.</td>
<td>o free</td>
</tr>
<tr>
<td>o Enable BGP routing process</td>
<td>o echo</td>
</tr>
<tr>
<td>o Enable OSPF routing process</td>
<td>o cp &lt;sourcefile&gt; &lt;destination&gt;</td>
</tr>
<tr>
<td>o Enable EIGRP routing processs</td>
<td>o mv &lt;sourcefile&gt; &lt;destination&gt;</td>
</tr>
<tr>
<td>o Configure ARP</td>
<td>o mkdir &lt;directory&gt;</td>
</tr>
<tr>
<td>o Show Startup config and Running Config.</td>
<td>o cd &lt;directory&gt;</td>
</tr>
<tr>
<td>o Ping devices.</td>
<td>o rm &lt;filename&gt;</td>
</tr>
<tr>
<td>Telnet Script Editor to edit existing telnet script files.</td>
<td>o hostname</td>
</tr>
<tr>
<td>Telnet Script Configuration tool to configure new telnet commands.</td>
<td>o pwd</td>
</tr>
<tr>
<td></td>
<td>o exit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TFTP/FTP Configurations</th>
<th>TFTP/FTP Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>• TFTP client and server support for all the devices in the network.</td>
<td>• TFTP client and server support for all the devices in the network.</td>
</tr>
<tr>
<td>• FTP client support for all the devices in the network.</td>
<td>• FTP client support for all the devices in the network.</td>
</tr>
<tr>
<td>• Execution of file transfer from GUI or at runtime using scripts or through RMI.</td>
<td>• Execution of file transfer from GUI or at runtime using scripts or through RMI.</td>
</tr>
<tr>
<td>• Script execution on file upload/download.</td>
<td>• Script execution on file upload/download.</td>
</tr>
<tr>
<td>• Start and Stop of TFTP server from the UI or at runtime through scripts or using RMI.</td>
<td>• Start and Stop of TFTP server from the UI or at runtime through scripts or using RMI.</td>
</tr>
<tr>
<td>• Configuration of common root directory for all or a specified range of devices.</td>
<td>• Configuration of common root directory for all or a specified range of devices.</td>
</tr>
<tr>
<td>• Copying of a common file to all TFTP root directories.</td>
<td>• Copying of a common file to all TFTP root directories.</td>
</tr>
<tr>
<td>• Stop the TFTP server from the UI or at runtime through scripts or using RMI.</td>
<td>• Stop the TFTP server from the UI or at runtime through scripts or using RMI.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network Configurations</th>
<th>Network Configurations</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Modification of IP address of all/selected devices in the network</td>
<td>• Modification of IP address of all/selected devices in the network</td>
</tr>
<tr>
<td>• Modification of Port number of all the devices in the network</td>
<td>• Modification of Port number of all the devices in the network</td>
</tr>
<tr>
<td>• Modification of OID values in all or selected devices present in the network</td>
<td>• Modification of OID values in all or selected devices present in the network</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bulk Modification</th>
<th>Bulk Modification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Modification of IP address of all/selected devices in the network</td>
</tr>
<tr>
<td></td>
<td>• Modification of Port number of all the devices in the network</td>
</tr>
<tr>
<td></td>
<td>• Modification of OID values in all or selected devices present in the network</td>
</tr>
</tbody>
</table>
| Template based bulk modifications | • Modification of SNMP and TL1 device values, Traps, Informs and Behavior Scripts. The modification will be reflected on all the template devices in the network.  
• Modification of protocol specific properties. The modification will be reflected on all the template devices in the network. |
| Bulk Script Addition for SNMP Devices | Configure the same script file to the specified range of IP address in the network. |
| Topology Editor | Topology Editor enables you to visualize interconnections between the devices in the network. You can connect new devices or disconnect existing devices in the topology. |
| Logging Levels | Configurable debug levels: OFF, Non-Critical, Critical and debug |
| Character Encoding Format | Configurable character encoding format. When the network is started, any string provided in the simulator, will be encoded as per the encoding format provided |
| Initialization of Trap/Autonomous and Behavior Service | • Option to enable/disable the execution of Trap/Autonomous/Inform messages at agent runtime.  
• Option to enable/disable the execution of SNMP and TL1 behavior scripts at agent runtime. |
| RMI Support | • RMI (Remote Method Invocation) is implemented in the network simulator to control the simulated SNMP and TL1 agents in the network from a remote client program.  
• The RMI client APIs, defines the methods in the network simulator, which can be accessed via RMI from the client program. |
| Network Operations |  |
| Network View | • View the properties of individual devices in the network from the Network tree or from the Network area.  
• View the network information like the total number of agents in the network, the agents started, the idle agents and information about each device type. |
| Modify & Delete Network devices | • Modify Network Device Properties  
• Modify Network Device Configurations  
• Delete devices in the network |
| Save Network | • Saving currently open network  
• Saving all open networks  
• Saving a network in the specified directory |
| Start Network          | • Start single/multiple network from UI  
|                       | • Start single/multiple network from command line |
| Stop Network           | • Stop Network from UI  
|                       | • Stop Network from Command Line |
| Start/Stop individual agents in the Network | • Option to start/stop individual agents in the network with all or specific protocols, from UI  
|                       | • Option to start/stop multiple agents in the network.  
|                       | • Option to start/stop individual agents in the network at runtime using scripts or through RMI. |
| Load and Unload        | • Load multiple networks. The network tree will display only the network that is currently loaded in the Network Designer.  
|                       | • Unload a Network |
| Delete Network         | • Delete a Network and its configurations. |
| Edit Operations        | • Copy and paste devices in the network area.  
|                       | • Find a device in the network by specifying the ipaddress/device unique name. |
| Operations permitted on a running network | • Adding a device to the network and starting the device.  
|                       | • Copy - Paste a device in the network  
|                       | • Configuration of Agent Values.  
|                       | • Recording from a Real SNMP Agent  
|                       | • Behavior Scripts Configurations  
|                       | • Trap and Autonomous Message Configurations  
|                       | • Configuring Echo requests for TL1 devices |
| Others                 |                       |
| Automated Network Simulation | • Automation Wizard to automate a network from the UI and run the automated network from command line. |
| Porting the Network/Devices in device tree | • Package the network that you have created with the desired configurations  
|                       | • Install the network in any other system having Simulation Toolkit installed.  
<p>|                       | • Package and install individual /multiple devices in the device tree. |
| SNMP Trap Viewer       | The Trap viewer is integrated with the Network Designer tool to view incoming traps |</p>
<table>
<thead>
<tr>
<th>Inform Response Receiver</th>
<th>The Inform Response Receiver is integrated with the Network Designer tool to view incoming inform responses.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Walk File Creator</td>
<td>Command Line utility to create SNMP WALK file.</td>
</tr>
<tr>
<td>Cable Modem Simulation</td>
<td>Cable modem device simulation is supported. The cable modem device supports SNMP and TFTP.</td>
</tr>
</tbody>
</table>
## Utility Tools

<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trap Recorder</td>
<td>The Network Recorder enables users to record real SNMP traps</td>
</tr>
<tr>
<td></td>
<td>- GUI tool to record SNMPv1 and v2c traps.</td>
</tr>
<tr>
<td></td>
<td>- The recorder traps can be stored in XML files and replayed in the</td>
</tr>
<tr>
<td></td>
<td>SNMP agent simulator and network simulator as request-based,</td>
</tr>
<tr>
<td></td>
<td>threshold-based or time-based traps.</td>
</tr>
<tr>
<td>Network Recorder</td>
<td>The Network Recorder enables users to record a real network.</td>
</tr>
<tr>
<td></td>
<td>- Any number of target SNMP devices can be recorded simultaneously by</td>
</tr>
<tr>
<td></td>
<td>providing the starting and ending IP address.</td>
</tr>
<tr>
<td></td>
<td>- The recorded network can be saved and replayed instantly in the</td>
</tr>
<tr>
<td></td>
<td>Network Designer.</td>
</tr>
<tr>
<td></td>
<td>- Any variation on the recorded data can be created for use with the</td>
</tr>
<tr>
<td></td>
<td>Network Designer tool.</td>
</tr>
<tr>
<td>SNMP Trap Stormer</td>
<td>An exclusive tool to send SNMP v1/v2c traps at burst and normal mode.</td>
</tr>
<tr>
<td>SNMPv3 Configuration Tool</td>
<td>To configure SNMPv3 users in USM Tables</td>
</tr>
<tr>
<td></td>
<td>To configure SNMPv3 users in VACM Tables</td>
</tr>
<tr>
<td>MIB Browser</td>
<td>SNMP Manager tool</td>
</tr>
<tr>
<td>TL1 Message Builder</td>
<td>To create and modify TL1 Command Set (TCS) definitions in the XML format</td>
</tr>
<tr>
<td>TL1 Craft Interface</td>
<td>TL1 Manager tool</td>
</tr>
</tbody>
</table>
Contacting AdventNet

AdventNet is eager to hear your comments, feedback, and suggestions for improving our products. We give you the contact information of our Corporate Headquarters, Sales, and Technical support teams. During the evaluation phase, AdventNet provides support "at no cost". This support includes email access to our product specialists for problem resolution, clarifications in documentation, and technical guidance. Feel free to send in your queries. We guarantee to respond as quickly as possible to make your evaluation a success.

Corporate Headquarters

<table>
<thead>
<tr>
<th></th>
<th>AdventNet Headquarters</th>
<th>AdventNet Development Center</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Site</td>
<td>AdventNet, Inc.</td>
<td>AdventNet Development Center (India) Private Limited</td>
</tr>
<tr>
<td></td>
<td>5200 Franklin Dr, Suite 115</td>
<td>Sree Narayana Complex</td>
</tr>
<tr>
<td></td>
<td>Pleasanton, CA 94588, USA</td>
<td>11 Sarathy Nagar</td>
</tr>
<tr>
<td></td>
<td>Phone: +1-888-720-9500</td>
<td>Velachery, Chennai 600 042</td>
</tr>
<tr>
<td></td>
<td>Fax: +1-925-924-9600</td>
<td>INDIA</td>
</tr>
<tr>
<td></td>
<td>Email : <a href="mailto:info@adventnet.com">info@adventnet.com</a></td>
<td>Phone: +91-44-22431115 (10 lines)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Fax: +91-44-22435327</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Email : <a href="mailto:info@adventnet.com">info@adventnet.com</a></td>
</tr>
</tbody>
</table>

Sales

For sales and product licensing inquiries, please use the contact information available in the following URL:
http://www.adventnet.com/contact.html

You can also send e-mail to sales@adventnet.com or call AdventNet headquarters at the following numbers

Telephone : +1-925-924-9500
Fax : +1-925-924-9600

Note: Please request for Sales, when you call us at the above mentioned phone number.

Support

Support and maintenance are AdventNet's key competencies. To ensure that you get an immediate solution to the problem,

Search our Knowledge Base: There is a possibility of your support issue being resolved already and the solution readily available to you in the knowledge base. We recommend you to browse through the knowledge base prior to sending an issue to us.

User Forum: We have an active forum that immediately addresses issues reported. We recommend the use of forums for healthy communication and knowledge sharing for registered and non-registered users.

Technical support: For obtaining technical support, please send email to: simulator-support@adventnet.com. For those with support contracts, AdventNet provides priority support through e-mail with a reply usually within 24 hours. Please provide the following information while sending support mails:
• Release version of the product (Ex. AdventNet Simulation Toolkit, Release 6)
• Build Number
• Operating System and version
• Stack traces, if any
• Debug messages or hex dumps, if any
• CLASSPATH environment variable, if applicable

and any other information. This helps us provide faster responses to your query.

Feedback

We welcome your feedback that will help us improve the product and provide future enhancements. Send your feedback to: simulator-support@adventnet.com.
Installation Guide

The Installation Guide explains the system requirements for installing and working with AdventNet Simulation Toolkit in different operating systems such as Windows, Linux, and Solaris. It describes the installation procedures to be followed when installing AdventNet Simulation Toolkit on different platforms. Next, it details the starting and shutting down procedures in the product. It also describes the package structure in the product along with the third party software bundled.

- System Requirements
- Installation and Startup Options
- Packages Bundled
System Requirements

The operating systems supported by Simulation Toolkit and the optimal system requirements at the time of installation of the product is given below:

Supported Platforms

- Windows NT
- Windows 2000
- Windows XP
- Windows XP with Hyper Threading enabled.
- Windows 2003
- Linux 6.2 and above
- Linux AS, ES, WS and Debian
- Solaris 5.6 & above
- Fedora - 3 core

System Requirements

The following are the optimal system requirements for AdventNet Simulation Toolkit at the time of installation.

<table>
<thead>
<tr>
<th>Windows (NT2000/XP/2003)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
</tr>
<tr>
<td>Memory</td>
</tr>
<tr>
<td>Disk Space</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Solaris (Sun OS 5.6 and above)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
</tr>
<tr>
<td>Memory</td>
</tr>
<tr>
<td>Disk Space</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Linux 6.2 and above, Linux AS, WS, ES and Debian</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
</tr>
<tr>
<td>Memory</td>
</tr>
<tr>
<td>Disk Space</td>
</tr>
</tbody>
</table>
Installation and Startup Options

- Installing Simulation Toolkit
- Starting Simulation Toolkit
- Using License Manager
- Using Launcher
- Installing Service Packs
- Uninstalling the Service Pack
- Uninstalling Simulation Toolkit
Installing Simulation Toolkit

This section discusses the installation procedure of AdventNet Simulation Toolkit in various operating systems such as Windows, Linux, and Solaris. The product is available for free download and evaluation from the website url: http://www.adventnet.com/products/simulator/download.html. You must be a “Admin” user to install Simulation Toolkit.

Installation of AdventNet Simulation Toolkit on Windows

The AdventNet Simulation Toolkit is available in a self-extracting EXE format. Download the EXE file AdventNet_Simulation_Toolkit_6_Windows.exe and install AdventNet Simulation Toolkit by executing the EXE file.

Installation of AdventNet Simulation Toolkit on Linux / Solaris

The AdventNet Simulation Toolkit is available as a bin file for Linux and Solaris. Download the bin file AdventNet_Simulation_Toolkit_6_Linux.bin (or) AdventNet_Simulation_Toolkit_6_Solaris.bin, and install Simulation Toolkit by executing the bin file.

Check the executable permission of the bin file. If the bin file is not executable, then execute the command

```
chmod a+x AdventNet_Simulation_Toolkit_6_Linux.bin (or)
chmod a+x AdventNet_Simulation_Toolkit_6_Solaris.bin
```

Then, execute the bin file as

```
./AdventNet_Simulation_Toolkit_6_Linux.bin or
./AdventNet_Simulation_Toolkit_6_Solaris.bin
```

If the execution throws an error such as “there may not be enough temporary space available in the temp folder”, then execute the file with the argument as

```
./AdventNet_Simulation_Toolkit_6_Linux.bin -is:tempdir $DIRNAME
```

or

```
./AdventNet_Simulation_Toolkit_6_Solaris.bin -is:tempdir $DIRNAME
```

where $DIRNAME is the absolute path of any existing directory.
Starting Simulation Toolkit

Starting the application in Windows

The installation process creates the *AdventNet Simulation Toolkit 6* in Start -> Programs Menu.

- The **Simulation Toolkit Launcher** can be started by choosing the Start -> Programs -> AdventNet Simulation Toolkit 6 -> AdventNet Simulation Toolkit icon. It helps to start all other Simulation Toolkit applications bundled with this product. Refer to **Using Launcher** section, to learn more.
- The user can also start the tools as separate applications by invoking the relevant item from the 'Program' group.
  - To start **SNMP Agent Simulator**, invoke Start -> Programs -> AdventNet Simulation Toolkit 6 -> Snmp Agent Simulator
  - To start **TL1 Agent Simulator**, invoke Start ->Programs -> AdventNet Simulation Toolkit 6 -> TL1 Agent Simulator
  - To start **Network Designer**, invoke Start ->Programs -> AdventNet Simulation Toolkit 6 -> Network Designer
  - To start **Network Recorder**, invoke Start ->Programs -> AdventNet Simulation Toolkit 6 -> Network Recorder
  - To start **SNMP Trap Recorder**, invoke Start ->Programs -> AdventNet Simulation Toolkit 6 -> SNMP Trap Recorder
  - To start **SNMP Trap Stormer**, invoke Start ->Programs -> AdventNet Simulation Toolkit 6 -> SNMP Trap Stormer
  - To start **SNMP MIB Browser**, invoke Start -> Programs -> AdventNet Simulation Toolkit 6 -> Snmp MIB Browser
  - To start **TL1 Message Builder**, invoke Start -> Programs -> AdventNet Simulation Toolkit 6 -> TL1 Message Builder
  - To start **TL1 Craft Interface**, invoke Start -> Programs -> AdventNet Simulation Toolkit 6 -> TL1 Craft Interface
  - To start **SNMP v3 configuration tool**, invoke Start -> Programs -> AdventNet Simulation Toolkit 6 -> Snmp v3 Configurator

Starting the application will invoke the License Agreement dialog. If you agree and proceed to the next screen, Licensee Details dialog will be displayed. Refer to, **Using License Manager**, to learn about the licensing procedure.

Starting the Application in Solaris / Linux

- The **Simulation Toolkit Launcher** can be started by invoking the shell script *Simulation Toolkit.sh* in the installed directory, it helps to start all other Simulation Toolkit applications bundled with the product. Refer to **Using Launcher** section, to learn more.
- The user can also start the tools as separate applications by invoking the relevant shell script from the `<Simulator_Home>/bin` directory.
  - To start SNMP Agent Simulator as separate application, invoke the shell script *SNMPAgentSimulator.sh* under `<Simulator_Home>/bin` directory.
  - To start TL1 Agent Simulator as separate application, invoke the shell script *TL1AgentSimulator.sh* under `<Simulator_Home>/bin` directory.
  - To start Network Designer as separate application, invoke the shell script *NetworkDesigner.sh* under `<Simulator_Home>/bin` directory.
To start Network Recorder as separate application, invoke the shell script `NetworkRecorder.sh` under `<Simulator_Home>\bin` directory.

To start SNMP Trap Recorder as separate application, invoke the shell script `SNMPTrapRecorder.sh` under `<Simulator_Home>\bin` directory.

To start SNMP Trap Stormer as separate application, invoke the shell script `SNMPTrapStormer.sh` under `<Simulator_Home>\bin` directory.

To start SNMP MIB Browser as separate application, invoke the shell script `SNMPMibBrowser.sh` under `<Simulator_Home>\bin` directory.

To start TL1 Craft Interface as separate application, invoke the shell script `TL1CraftInterface.sh` under `<Simulator_Home>\bin` directory.

To start TL1 Message Builder as separate application, invoke the shell script `TL1MessageBuilder.sh` under `<Simulator_Home>\bin` directory.

To start SNMP v3 configurator as separate application, invoke the shell script `SnmpV3Config.sh` under `<Simulator Home>\bin` directory.

Starting the application will invoke the License Agreement dialog. If you agree and proceed to the next screen, Licensee Details dialog will be displayed. Refer to, Using License Manager, to learn about the licensing procedure.
Using License Manager

- Overview
- Procedure for Licensing
- Simulation Toolkit Licensing Scheme
- Obtaining Registered License
- Upgrading Your License Policy From Trial to Registered User

Overview

AdventNet provides an Evaluation (trial) period for 30 days for AdventNet Simulation Toolkit, which can be extended if required. No license file is required for evaluation. AdventNet will provide a registered user file after you purchase the product. To get the Registered User file you must contact our sales team at sales@adventnet.com

Procedure for Licensing

After installing the product, you must provide the required licensing information for starting the product. When you start the Application Launcher or any other application, the License Agreement will be invoked as shown in the screen shot below:

![License Agreement Screen Shot](image)
• If you accept the License Agreement, click Next to proceed to the next screen.
• If you are an evaluation user, specify the User Type as Evaluation User (no license file is required for evaluation) and click Finish to start the product, as shown in screen shot below:

![Screen shot of AdventNet Simulation Toolkit 6.0 user interface]

• If you are a Registered User (if you have got a registered license file) select the Registered User option and click Next, to specify the license file.
• You must browse and provide the license file in the below screen and click Open button.
• Click **Next**, to view the user names to whom the license file is authorised. Select the User Name and click **Finish** as show in screen-shot below. The licensing procedure will be completed and the product will get started.
Simulation Toolkit Licensing Scheme

AdventNet Simulation Toolkit follows a unique trial licensing scheme.

Trial License scheme

You can evaluate AdventNet Simulation Toolkit. The evaluation version has the following restrictions:

<table>
<thead>
<tr>
<th>Tools</th>
<th>Evaluation Restriction</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Network Simulator</strong></td>
<td>• SNMP&lt;br&gt;  o Restricted to 25 SNMP devices&lt;br&gt;  o Response for 2 SNMP group nodes per device. (first two nodes that get the request)&lt;br&gt;  o Allows configuration of 3 traps per device.&lt;br&gt;• TL1&lt;br&gt;  o Restricted to 25 TL1 devices&lt;br&gt;  o Response for 5 Command Codes per device. (first five Command codes that get the request.)&lt;br&gt;  o Allows configuration of 3 input based Autonomous messages and 3 timer based autonomous messages per device.&lt;br&gt;  o TFTP&lt;br&gt;  o Restricted to 25 TFTP devices.&lt;br&gt;  o CLI&lt;br&gt;  o Restricted to 25 Cisco IOS devices.&lt;br&gt;  o Telnet&lt;br&gt;  o Restricted to 25 Telnet devices</td>
</tr>
<tr>
<td><strong>SNMP Agent Simulator</strong></td>
<td>• Response only for the first two group nodes requested.&lt;br&gt;  • Allows configuration of 3 traps and 3 Informs</td>
</tr>
<tr>
<td><strong>TL1 Agent Simulator</strong></td>
<td>• Response for 5 Command Codes. (first five Command codes in the tree)&lt;br&gt;  • Allows configuration of 3 input based Autonomous messages and 3 timer based autonomous messages.</td>
</tr>
</tbody>
</table>

Obtaining Registered License

To obtain a Registered License contact sales@adventnet.com

Upgrading Your License Policy From Trial to Registered User

If you are using Simulation Toolkit with an Evaluation License but later on obtained a Permanent/Registered license, you can upgrade this license and make the evaluation version as registered version. To upgrade your license from trial user to registered user, do the following:

- Start the launcher by running the SimulationToolkit.bat/sh file present under the <Simulator Home> directory.
• From the Help menu, select the Register menu Item. The License Manager Screen will popup.
• Use the above procedure to upgrade from trial user to registered user.

Please contact AdventNet for any clarifications about Licensing of the product.
Using Launcher

The AdventNet Simulation Toolkit launcher provides a single UI from which the various tools can be invoked. Each tool can be started by double clicking on its icon or selecting the **Options -> Run** menu from the toolbar. The application can also be run by right-clicking and selecting the **Run** option. The Simulation Toolkit launcher can be toggled to the toolbar view by clicking the **View --> Tool Bar** menu from the toolbar. The launcher can be toggled back to the normal view by selecting the "normal view" icon.

The Launcher UI is organized as two panels - the top and the bottom panel. The top panel shows all the applications that are part of the AdventNet Simulation Toolkit package. All the applications can be started by double-clicking or by selecting the run option from the menu bar.

The bottom panel shows the currently running applications in the form of tabs. The applications can be terminated from the launcher itself by selecting **Options -> Terminate** menu. The bottom panel also displays the error and exception messages of the respective applications.

Selecting the **Options -> Settings** menu pops up the Java Environment settings dialog box which can be used to set the various path and classpath variables needed to use the applications. Any new classpath settings, modification of the existing environment variables can also be done. The General settings is used to set the global settings which will be used throughout the AdventNet Simulation Toolkit package.
The following environment settings can be set using the settings option in the launcher:

Browser path - path for the web browser. The web browser is used for loading the applets and for viewing the help files. To set the path, click on the browse button and select the corresponding executable for the browser. (eg. c:\Program Files\communicator\netscape.exe or usr/bin/netscape). Please note that the complete file name has to be given, not just the path where the executable lies.

JAVA HOME - sets the JAVA_HOME environment variable. By default it uses the JRE bundled with the product. The user can set it for any Java version (i.e JDK 1.4 and above). If the field is left blank it will take the default option (JRE).

JAVA Classpath - the classpath settings necessary for running the application is set here. By default the necessary classpaths are already set. Additional classpath can be added to the existing list. Please note that the classpath settings takes precedence in the order in which they are listed.

The Applications folder in the settings panel displays all the applications that are part of the AdventNet Simulation Toolkit package. Any property specific to the application can be set by selecting the corresponding application. The settings tab corresponding to each application displays the class path settings specific to the application. Any new classpath settings can be added using this option.

To Quit from the Launcher application select Options -> Exit.

To view any error messages generated when the applications are started, select Options -> Log. The LogInfo dialog box is opened.

Select a particular log file name to view the error messages.
Installing Service Packs

- Overview
- Installing the Service Pack
- Viewing the Details of the Service Pack

Overview

AdventNet periodically provides Service Packs which provide new features (requested by the customers), fixes for certain bugs and document updates in the form of HTML files. Service Packs can be downloaded from the web site, and updated into Simulation Toolkit using the Update Manager tool.

Installing the Service Pack Using Update Manager

1. Start the Launcher by invoking `SimulationToolkit.bat/sh` from `<SimulatorHome>` directory.
2. Choose **Update Manager** from the **Options** menu in the Launcher, to start Update Manager.
3. When you start the Update Manager, all the Service Packs that are already installed are listed and it provides the options to Install, Uninstall and view the details of any Service Pack.

   **Warning**: Ensure that no application is running when applying the Service Pack. This prevents any files used by the application from being over-written.

4. Click **Browse** button and select the Service Pack file (.ppm) to be installed.
5. When the Service Pack is selected, the Update Manager enables the **Install** and **Readme** buttons as shown in the screen below.
6. You can go through the Readme file of the Service Pack by clicking the **Readme** button and install the Service Pack by clicking the **Install** button. The service pack will be installed as shown in the screen-shot below:

![AdventNet Update Manager](image)

**Note:**
1. On clicking the "Install" button, the tool checks whether there is enough space for the installation of the service pack. If there is not enough space, the tool informs the user about the lack of space. You must clear the space and then proceed with the installation.

2. After installing the service pack, exit from the update manager and launcher and restart the launcher
Viewing the Details of a Service Pack

The installed service packs are listed in the main window of the Update manager.

- The details can be viewed by selecting the service pack version and clicking the Details button. The following screen will be shown.

  ![Service Pack Details Screen]

  - A dialog pops-up displaying the General details about the patch in a tab. It contains the patch name, description, installed size and installation time.
  - The installation log messages are shown below these details.
  - The other tab displays the README related to this Service pack.
  - The details can also be viewed by double clicking the required service pack version.
Uninstalling the Service Pack

1. Start the Launcher by invoking `SimulationToolkit.bat/sh` from `<SimulatorHome>` directory.
2. Choose **Update Manager** from the **Options** menu in the Launcher, to start Update Manager.
3. When you start the Update Manager, all the Service Packs that are already installed are listed and it provides the options to Install, Uninstall and view the details of any Service Pack.
4. Select the service pack, which needs to be uninstalled, from the list. Click **Uninstall** button to proceed with the uninstallation.
5. This will bring up a dialog which shows the service pack to be uninstalled. It also lists all the related service packs that will be uninstalled. Click **Finish** to uninstall the service pack.
Uninstalling Simulation Toolkit

In Windows

For uninstalling AdventNet Simulation Toolkit,
- Execute the `uninstaller.exe` file present under `<Simulator_Home>_uninst` folder (or)
- Select Start -> Programs -> AdventNet Simulation Toolkit 6 -> Uninstall

In Linux/Solaris

For uninstalling AdventNet Simulation Toolkit, execute the `uninstaller.bin` file present under `<Simulator_Home>_uninst` folder.
Packages Bundled

- Package Structure
- Third Party Software
- Installing MySQL
Package Structure

The AdventNet Simulation Toolkit package is installed under *AdventNet/Simulator5* directory. The directory structure and the files under *AdventNet/Simulator6_0* are as follows:

<table>
<thead>
<tr>
<th>Directory</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bin</strong></td>
<td>Contains shell scripts/batch files to run the UI tool as a separate application</td>
</tr>
<tr>
<td></td>
<td>- NetworkDesigner.bat/sh - To start the Network Designer Tool</td>
</tr>
<tr>
<td></td>
<td>- NetworkRecorder.bat/sh - To start the Network Recorder Tool</td>
</tr>
<tr>
<td></td>
<td>- SnmpAgentSimulator.bat/sh - To start the SNMP Agent Simulator.</td>
</tr>
<tr>
<td></td>
<td>- SnmpMibName.bat/sh - To start the MIB Browser tool</td>
</tr>
<tr>
<td></td>
<td>- SnmpTrapStormer.bat/sh - To start the SNMP Trap Stormer tool</td>
</tr>
<tr>
<td></td>
<td>- SnmpTrapRecorder.bat/sh - To start the SNMP Trap Recorder tool</td>
</tr>
<tr>
<td></td>
<td>- SnmpV3Config.bat/sh - To configure the SNMPv3 agent with USM and VACM entries.</td>
</tr>
<tr>
<td></td>
<td>- TL1AgentSimulator.bat/sh - To start the TL1 Agent Simulator.</td>
</tr>
<tr>
<td></td>
<td>- TL1CraftInterface.bat/sh - To start the TL1Browser.</td>
</tr>
<tr>
<td></td>
<td>- TL1MessageBuilder.bat/sh - To start TL1 Message Builder.</td>
</tr>
<tr>
<td></td>
<td>- StartMySQL.bat/sh - To explicitly start MySQL in Windows/Linux/Solaris</td>
</tr>
<tr>
<td></td>
<td>- StopMySQL.bat/sh - To explicitly stop MySQL in Windows/Linux/Solaris</td>
</tr>
<tr>
<td></td>
<td>- SetEnv.bat/sh - To set the classpath.</td>
</tr>
<tr>
<td></td>
<td>- readme_ui_tools.html - The README for all the UI tools in HTML format.</td>
</tr>
<tr>
<td></td>
<td>- readme_ui_tools.txt - The README for all the UI tools in txt format</td>
</tr>
<tr>
<td><strong>cmdline</strong></td>
<td>- this folder contains shell scripts/batch files to run the command line tools.</td>
</tr>
<tr>
<td></td>
<td>- NetworkRecorder.bat/sh - To start the command line network recorder tool.</td>
</tr>
<tr>
<td></td>
<td>- StartSnmpProxyAgent.bat/sh - To start multiple SNMP agent simulation in command line.</td>
</tr>
<tr>
<td></td>
<td>- StartTLIGNE.bat/sh - To start multiple TL1 agent simulation in command line.</td>
</tr>
<tr>
<td></td>
<td>- StartNetwork.bat/sh - To start the simulated network in command line.</td>
</tr>
</tbody>
</table>
### Directory

<table>
<thead>
<tr>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>• StartAutomatedNetwork.bat/sh - Utility to start the automated network.</td>
</tr>
<tr>
<td>• StartSnmpAgent.bat/sh - To start the simulated SNMP agent in command line.</td>
</tr>
<tr>
<td>• StormTraps.bat/sh - To start the command line trap stormer.</td>
</tr>
<tr>
<td>• StartTL1Agent.bat/sh - To start the simulated TL1 agent in command line.</td>
</tr>
<tr>
<td>• StopSnmpAgent.bat/sh - Command Line Agent Remote Shutdown.</td>
</tr>
<tr>
<td>• StartRMIRegistry.bat/sh - To start the RMI registry.</td>
</tr>
<tr>
<td>• CreateVIP.bat/sh - Utility to create virtual IP address.</td>
</tr>
<tr>
<td>• DeleteVIP.bat/sh - Utility to delete virtual IP address</td>
</tr>
<tr>
<td>• WalkFileConfigCreator.sh/.bat - Utility to create an SNMP configuration from command line by specifying a WALK file as input</td>
</tr>
<tr>
<td>• WalkFileCreator.sh/bat - Utility to create an SNMP Walk File by specifying hostname, port number and community of the agent.</td>
</tr>
<tr>
<td>• readme_cmdline_tools.html - The README for all the command line tools in HTML format.</td>
</tr>
<tr>
<td>• readme_cmdline_tools.txt - The README for all the command line tools in txt format.</td>
</tr>
</tbody>
</table>

• **migration** - contains migration tools to aid users of AdventNet Simulation Toolkit 5 in migrating to the latest AdventNet Simulation Toolkit 6 release version

  • NetworkMigrator - to migrate the network configuration
  • SnmpMigrator - to migrate the SNMP Agent configuration
  • TL1Migrator - to migrate the TL1 Agent configuration

### conf

<table>
<thead>
<tr>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>• launcher_conf.txt - Contains the launcher configuration.</td>
</tr>
<tr>
<td>• database_params.conf - The details of the database server, port, user name, password and server path.</td>
</tr>
<tr>
<td>• DeviceOid.prop - This file contains the sysObjectId variable and the name of the device. During network recording, the network device is assigned a name, based on the sysObjectId value specified in this file for the specific device.</td>
</tr>
<tr>
<td>• migrationdb_params.conf - The details of the old database server, old port, old username, old password and the old server path is stored in this conf file.</td>
</tr>
<tr>
<td>• RMIParameters.conf - When the simulator is started with the RMI Option both in the commandline and the UI Mode, this file is read and the simulator reference is registered to the RMI Registry.</td>
</tr>
<tr>
<td>Directory</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td><strong>settings</strong></td>
</tr>
<tr>
<td><strong>USMUserTable.xml</strong></td>
</tr>
<tr>
<td><strong>VacmContextTable.xml</strong></td>
</tr>
<tr>
<td><strong>VacmAccessTable.xml</strong></td>
</tr>
<tr>
<td><strong>VacmSecurityToGroupTable.xml</strong></td>
</tr>
<tr>
<td><strong>VacmViewTreeFamilyTable.xml</strong></td>
</tr>
<tr>
<td><strong>csh</strong></td>
</tr>
<tr>
<td><strong>conf</strong></td>
</tr>
<tr>
<td><strong>db</strong></td>
</tr>
<tr>
<td><strong>images</strong></td>
</tr>
<tr>
<td><strong>library</strong></td>
</tr>
<tr>
<td><strong>conf</strong></td>
</tr>
<tr>
<td><strong>db</strong></td>
</tr>
<tr>
<td><strong>images</strong></td>
</tr>
<tr>
<td><strong>library</strong></td>
</tr>
<tr>
<td><strong>dtds</strong></td>
</tr>
<tr>
<td><strong>examples</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td><strong>Examples</strong></td>
</tr>
<tr>
<td><strong>help</strong></td>
</tr>
<tr>
<td><strong>images</strong></td>
</tr>
<tr>
<td><strong>jars</strong></td>
</tr>
<tr>
<td><strong>jre</strong></td>
</tr>
<tr>
<td><strong>jython</strong></td>
</tr>
<tr>
<td><strong>lib</strong></td>
</tr>
<tr>
<td><strong>logs</strong></td>
</tr>
<tr>
<td><strong>mibs</strong></td>
</tr>
<tr>
<td><strong>mysql</strong></td>
</tr>
<tr>
<td><strong>tcs</strong></td>
</tr>
<tr>
<td><strong>tutorials</strong></td>
</tr>
<tr>
<td><strong>COPYRIGHT</strong></td>
</tr>
<tr>
<td><strong>LICENSE_AGREEMENT</strong></td>
</tr>
<tr>
<td><strong>README.html</strong></td>
</tr>
<tr>
<td><strong>SimulationToolkit.bat</strong></td>
</tr>
</tbody>
</table>

**devices**

- conf - device properties
- db - device configurations of the devices in the device tree
- images - device images in device tree (23 * 23 px) and network area (36 * 72 px)
- library - devices in the device library
Third Party Software

The following third-party tools are bundled with the AdventNet Simulation Toolkit:

<table>
<thead>
<tr>
<th>Package Name</th>
<th>Version</th>
<th>Web Site</th>
</tr>
</thead>
<tbody>
<tr>
<td>jython.jar</td>
<td>1.1</td>
<td><a href="http://www.jython.org">http://www.jython.org</a></td>
</tr>
<tr>
<td>jre</td>
<td>1.4.2</td>
<td><a href="http://java.sun.com/j2se/1.4/runtime.html">http://java.sun.com/j2se/1.4/runtime.html</a></td>
</tr>
</tbody>
</table>
MySQL

- How MySQL is Bundled
- How Simulation Toolkit Gets Connected to MySQL
- Utility to Start/Stop MySQL Server Separately

How MySQL is Bundled

The MySQL version 4.0.20 is bundled along with AdventNet Simulation Toolkit.

In Windows/Linux/Solaris OS

The MySQL is bundled in a ready to use pattern. MySQL is present in the 
<Simulator_Home>/mysql directory. The mysql will be started automatically when the tools are started.

How Simulation Toolkit Gets Connected to MySQL

You have to specify the port at which the Simulation Toolkit should connect to MySQL. If your MySQL is already running at the specified port, then Simulation Toolkit will connect to that MySQL without prompting. This is applicable for all OS.

Note: The MySQL bundled with windows always tends to get connected to any MySQL that is running. Make sure that it connects to the MySQL that is bundled with the product.

When MySQL is installed as Service in Windows

If you have your MySQL already installed as a service, then the Simulation Toolkit will always connect to service. To connect to the MySQL bundled with the product follow the below stated steps

- Stop the Service.
- Open the my.ini file present under c:\WINNT and change the data directory to the data directory present under the <simulator home> \ <mysql>.
- Restart the Service.

Utility to Start / Stop MySQL Server Separately

Simulation Toolkit provides a separate batch/shell script utility to start/stop the MySQL server in Windows / Linux / Solaris. This utility is available in <SimulatorHome>/bin directory.

To start the MySQL server: StartMySQL.bat/sh
To stop the MySQL server: StopMySQL.bat/sh
Release Notes

This section gives details about the features available with the current release of the product, the known Issues and Limitations available with the product at the time of release.

- Release Features
- Known Issues
- Limitations
Release Features

- 6.0 Release Features
- Issues Fixed and Features Added in 5.1 SP2
- Issues Fixed and Features Added in 5.0 SP1 (5.1)
- 5.0 Release Features
- Issues Fixed and Features Added in 4.1 SP1
- 4.1 Release Features
- 4.0 Release Features
- Issues Fixed and Features Added in 3.0 SP2
- Issues Fixed and Features Added in 3.0 SP1
- 3.0 Release Features

6.0 Release Features

Network Simulator

- Template based model of device creation and data storage
  Simulation Toolkit 6 is based on a completely new data model. The new template based model of device creation and device configuration offers the following benefits:
  - Enables creation of large networks in seconds.
  - Occupies very less storage space of MySQL data for the network devices.
  - Enables template based configuration and modification of devices. (i.e) if 50 devices are created using Template device "A", any bulk modifications can be achieved by modifying this template device.
  - Offers enhanced network performance.
- GUI changes enhances the user experience with the product
  - Intuitive GUI to enable template based device configuration of table rows, traps, scripts, MIB(s) loading / unloading, and protocol specific parameters, in addition to bulk modification feature.
  - Multiple device selection for Start / Stop agent.
  - Option to view the network devices as network device images or in the form of a list.
  - Option to view all the inform response in intuitive table GUI.
  - Separate wizard to specify 'unique oid values' while creating the network.
  - Option to enable/disable the display of device library in the network designer.
  - Simplified network creation with automatic generation of IP address, the next time you create a network.
  - Option to view the Template tree or the Device Library, after network creation.
  - Multiple device selection for deletion when not required.
- New Cisco IOS Commands added - EIGRP router mode and Vlan mode.
- Complete SNMP v2, v3 Inform Support
- Support to package and install individual /multiple devices in the device tree.
- Option to **execute timer scripts after specified time** after agent startup.
- Command Line utility to **create SNMP WALK file**.

**SNMP Agent Simulator**
- Complete SNMP v2, v3 Inform Support.
- Package and install SNMP agent configurations.
- Option to **execute timer scripts after specified time** after agent startup.

**General**
- New **SNMPv3 configuration tool** with intuitive GUI, to simplify v3 configurations.
- **Migration tools** to migrate the SNMP agent, TL1 agent and network created in release 5 version to the current version 6.

**Issues Fixed and Features Added in 5.1 SP2**

**Network Simulator**

**Features added**
- Support for SNMPv2c and SNMPv3 informs in Network Designer.
- Automatic update of unique MAC address for all devices in the network.
- File input option to specify the IPAddress during device creation.

**Issues Fixed**
- Write community now provides read/write access.
- Issue in converting nodes of type octet string to hexadecimal format is fixed.
- Improved logging while creating configuration from Walk file.
- Issue in Walk file configuration creator is fixed.
- Issue in execution of scripts with Before Processing and After Processing conditions is fixed.
- Issue with row deletion and addition in buffer, is fixed.
- Mac Address is added without the length of the string.
- CPU usage while querying the agent is minimised.
- MAC Address instances treated as String, when added at runtime, is fixed.

**SNMP Agent Simulator**

**Issues Fixed**
- Write community now provides read/write access.
- Issue in specifying Trap OID for V3 traps is fixed.
- Issue in resetting the Agent sysUpTime value is fixed.

**Network Recorder**

**Issues Fixed**
- Issue with MIB loading is fixed
- Issue with row deletion and addition in buffer is fixed.
- Mac Address is added without the length of the string
Trap Stormer

Issue Fixed

- Issue with restricting 255 in First, Second and Third Octets of the IPAddress, when the IPAddress is added in the agent settings dialog, is fixed.

General

Features Added

- Macro returning current time for varbinds is added.

Issues Fixed

- Issues with script buffering is fixed.
- Exceptions received when sending request-based traps is fixed.
- Issue in resolving index with type MAC address is fixed.
- Issue with executing Timer scripts after agent restart is fixed.

Issues Fixed and Features Added in 5.0 SP1 (5.1)

Network Simulator

Features added

- Topology wizard is added to visualize the device connections.
- Enhanced Network Performance with new buffer design.
- New Cisco IOS commands added for BGP, OSPF, ARP, Startup config and Running config.
- Telnet Script Editor to edit existing telnet script files.
- Telnet Script Configuration tool to configure new telnet commands.
- Load and Save options provided for the Random Network Creation Wizard.
- SYSUPTIME macro is supported to provide the sysUpTime value of the agent.
- Cable Modem Simulation with SNMP support.
- Device creation time is optimized.

Issues Fixed

- Issues in RMI script API methods is fixed.
- Byte value 0 and 255 can be included in the IP address of Network Recorder.
- The issue with SNMP configuration getting corrupted is fixed by synchronizing the method.
- The issue in adding and deleting a device, when called simultaneously, is fixed by synchronizing the methods.
- Telnet Prompt changed as per standards.
- Issue in CLI Browser (running in Windows) when connected to Telnet server is fixed.
- While deleting a network, the issue with network database not getting deleted is fixed.
- Issue in sending Date and Time as varbind in V2 trap is fixed.
SNMP Agent Simulator

Features Added

- Option to configure scalar varbinds without instance (.0) is added.
- SYSUPTIME macro support is added.
- Option to configure the Engine ID in hexa format is provided.

Issues Fixed

- Issue with processing Set Request is fixed.
- Issue with creation of device using default templates is fixed.
- The issue in specifying trap OID for V3 traps fixed
- Issue in resetting the Agent sysUpTime value is fixed.

5.0 Release Features

Network Simulator

- Cisco IOS Software Simulation allows you to simulate telnet enabled Cisco routers and Cisco switches. CLI support is provided to access the IOS software. The IOS simulator fully supports SNMPv1, v2c, v3 and TFTP.
- Cisco IOS commands implemented in User, Privileged, Configuration, Interface configuration, Router and Vlan modes.
- IOS Script Editor and IOS Script Configuration Tools to edit existing script files and configure new IOS commands to the Cisco router/switch devices.
- Extensive CLI script API methods to customize the Cisco IOS commands.
- Telnet support to access the simulated devices in the network from the remote through Telnet protocol and execute Telnet commands.
- Pre-configured devices like Cisco routers and switches, HP and Lexmark printers, Hosts and Cisco Pix Firewall have been added to the device library. These devices are available in the device tree of the network simulator and can be directly used to create the network of your choice.
- Utility to create a WALK file and simulation of a device with the SNMP WALK file.
- Random Device Addition wizard to add multiple devices of different types at one shot. The devices will be added by incrementing the specified IP address.
- Option to view the devices in the network tree with their unique device name or with ip address and port number as per the preferences specified.
- Device specific parameters like system and interface details captured in a single UI to enable easy configuration.
- Support to configure multiple interfaces for each device and multiple ip address for each interface. The SNMP agent can be started at multiple IP addresses configured for each interface in the device.
- View the network information like the total number of agents in the network, the agents started, the idle agents and information about each device type.
- Bulk Modification of network device properties to modify the parameters specific to SNMP, CLI, TFTP and Telnet.
- Automatic configuration of device IP address to the ipaddrTable while adding SNMP devices to the network.
- Automatic configuration of sysName to the system group while adding SNMP devices to the network.
• Automated Network Simulation Wizard to automate a network from the UI and run the automated network from command line.

• Interface for customization of request/response SNMP PDUs is now supported for SNMP devices in network simulator. The interface also provides access to the SNMPScriptInterface object, for working with behavior API methods before processing the request/response PDU.

• The Network Recorder now supports recording SNMPv3 networks in addition to SNMP v1 and v2 versions.

• SNMP Trap Stormer is enhanced to configure and send traps with multiple "Agent IP address" for each trap.

• Command line tool to start multiple networks from command line.

• Migration Tools to migrate configuration created in AdventNet Simulation Toolkit 4 to the latest AdventNet Simulation Toolkit 5 release version.

• An additional option provided in command line network simulator to provide enhanced network performance and improved response time of the simulated agents.

• The Network Simulator provides the facility to dynamically configure IP addresses in Windows NT, 2000, XP when starting each agent instance in the network.

• Supports the latest MySQL version 4.0.20. The same is bundled with the product for all OS.

Issues Fixed and Features Added in 4.1 SP1

Network Simulator

Features Added

• Dynamic configuration of virtual IP address in Windows NT, 2000, XP is supported in addition to Linux/Solaris OS.

• Performance of the network simulator is increased while doing asynchronous requests.

• The following new script API methods are added in SNMPScriptInterface in \<SimulatorHome>/examples/javadocs directory, to be used within the JPython script files in Network Simulator. The methods are also defined in the RMI client API, NetSimSnmpInterface in \<SimulatorHome>/examples/javadocs directory, that can be accessed via RMI.

  • addSnmpAgent : To add an SNMP agent to the existing network.
  • deleteSnmpAgent : To delete an SNMP agent from the network.
  • isSnmpAgentRunning : To find if the SNMP agent is running at the specified IP address and port number.
  • getNoOfSnmpAgentsRunning : To get the number of SNMP agents running in the network.
  • getAllDeviceNames : To get the name of all the devices in the network.

Issues Fixed

• Multi-Threading issues in Windows 2000 Server is fixed.

• Issue in caching of data configured in the Performance Tuner dialog, while starting the network, is fixed.
SNMP Agent Simulator

Features Added
- The configuration of empty context name is allowed for v3 traps in the "Trap Configuration" dialog and while configuring scripts.
- The index resolving for nodes of Type "NetworkAddress" is solved. (In some cases, the instance string will be without the number representing the kind of address being encoded, for example '.1')

TL1 Agent Simulator

Features Added
- The following new script API methods are added in TL1ScriptInterface in <SimulatorHome>/examples/javadocs directory, to be used within the JPython script files in TL1 Agent Simulator.
  - getCurrentCommandCode : To get the current command code.
  - sendResponseMessage : To send different response messages for the same command code.

4.1 Release Features

Network Simulator
- Dynamic configuration of IP Address for the simulated agents.
- Bulk addition of devices with single IP Address and different port
- Bulk modification of devices properties like IP address and Port number.
- Agent management through RMI interface. The RMI client API defines the methods in the simulator, which can be accessed via RMI to manage the simulated SNMP and TL1 agents in the network.
- TFTP server and client implementation to enable transfer of files between the manager and agents in the network. The file transfer can be done from the Network designer UI or using Jython script methods or through RMI client methods.
- FTP client implementation to enable transfer of files between the manager and agents in the network. The file transfer can be done from the Network designer UI or using Jython script methods or through RMI client methods.
- Sending traps with the configured “Agent IP address"
- Global control of manager details for sending traps.
- Sending agent Uptime as varbind in Traps.
- Agent value manipulation for varbinds in Timer Traps

SNMP Agent Simulator
- Interface for customization of request and response SNMP PDUs.

TL1 Agent Simulator
- The TL1 agent can be managed through RMI. The RMI client API defines the methods in the simulator, which can be accessed via RMI to manage the simulated TL1 agent.

SNMP Trap Stormer
- Persistance in XML is supported.
New Tools

Network Recorder

The Network Recorder is a GUI and command line tool to create simulations of real SNMP networks. Any number of target SNMP devices can be recorded simultaneously by providing the starting and ending IP address. The recorded network can be saved in Database or XML format and replayed instantly in the Network Designer.

SNMP Trap Recorder

The SNMP Trap Recorder is a GUI tool to record SNMPv1/v2c traps from real agents. The recorder traps can be stored in XML files and replayed in the trap configuration dialog of the SNMP agent simulator and Network simulator as request-based, threshold-based or time-based traps.

4.0 Release Features

This section provides a brief description of the Key features added to the various modules of Simulation Toolkit 4.0.0

Network Designer

- Bulk addition of scripts to define the same behavior across all devices in the network.
- Option to start the network with SNMP v3 support. The agent will respond to the v3 requests in addition to v1/v2c requests.
- Start and Stop individual agents in the network.
- Simulate behavior for TL1 devices. Powerful script API methods to perform TL1 operations from a JPython script.
- Registration of selected OIDs for improved network performance.
- Registration of Command codes for new TCS files added to an existing simulated TL1 agent.
- Option to enable/disable the execution of Trap/Autonomous messages and behavior scripts configured in the UI.
- Operations like adding new devices to the network, starting and stopping individual agents in the network, Copy - Paste a device in the network, and simulating new MIBs, can be performed on the running network.
- Sample IP Router Simulation and Demo Network bundled with the product.
- Enhanced Network Designer GUI

SNMP Agent Simulator

- SNMP Proxy Agent Simulation.
- Customization of response SNMP PDU to send garbled/invalid responses.
- Support for periodic execution of Timer-based traps.
- Global Settings for agent IP address and manager details in traps.
- Sending Traps to multiple managers with multiple ports.
- Send traps with sysUpTime of the agent as varbind value.
- Send Timer-based traps with updated agent values in varbinds.
- Enhanced SNMP Script API Methods
- Option to enable/disable the execution of Trap, Inform, and Behavior scripts configured in the UI.
- Command line agent remote shutdown.

### SNMP Trap Stormer
- An exclusive tool to configure and send any number of v1/v2c traps.

### TL1 Agent Simulator
- Gateway Network Simulation.
- Configuration of multiple valid and error responses.
- Configuration of Autonomous messages for Command code - AID - MPB combination:
  - Powerful script API methods to perform TL1 operations from a JPython script.
  - Support for User Related Security as per the specifications in GR 815 and TR 835.
- Simulation of Error Response at random.
- Simulation of Delayed Responses at random.
- Option to enable/disable the execution of Autonomous messages and Behavior scripts configured in the UI.
- Re-simulation of agent values for newly added/deleted command code(s) and autonomous codes.
- Built-in and Self-monitoring commands implemented in the TL1 agent.
- TL1 Agent Simulator is developed over the latest AdventNet TL1 4.0 API.

### General
- Provides migration tools to migrate the simulations created in 3.0 release (simulated SNMP agent and simulated network with SNMP devices) to the latest 4.0 release version.
- Provides enhanced performance, especially for large scalability testing.

### Issues Fixed and Features Added in 3.0 SP2

#### Network Design Tool

**Features Added**
- Change in the representation of the Protocols supported by the device - removal of SNMP and TL1 nodes in the tree and addition of SNMP and TL1 buttons. The selection of the button will load the configuration.
- More than one network can be started at runtime. (i.e) After starting one network, it is possible to start another network also.
- You can load a new network and perform edit operations on the network, after starting Network Simulation. Edit operations cannot be performed on the running network alone.
- Logging support for the network simulation.
- Find MIB Node integrated in Network Design Tool.

**Issues Fixed**
- While sending traps using "Send" button, the traps were sent with the system IP address instead of the configured "From IP Address". This issue has been fixed.
- Problem with loading MIBs during bulk modification, when there is a mismatch between MIB name and MIB module name, has been fixed.
Network Simulator

Features added

- Provision to specify the character encoding format. For example, if you configure a script which sends a trap including a Japanese word and specify the encoding format as MS932 in the Encoding.txt file, the simulator can send a trap including an MS932 Japanese word.

Issues Fixed

- When a SNMP request was made on a variable with Counter64 type, the "response too long" error was received. This issue has been fixed.
- When SNMP GET/SET commands were sent simultaneously to multiple nodes running on different virtual IPs, the configured scripts were not getting executed properly due to overlapping of database name, table name or XML files. This issue has been fixed.
- While configuring script API methods across agents, in XML mode, the issue with some of the API methods not working properly is now fixed.
- If scalar nodes in the MIB were defined in a random order with tables in between, the problem with the registration of scalar OIDs is fixed.
- While sending periodical traps, the issue with elapsed traps being sent at Agent start up, is now fixed.
- The manager name and port values given in command line is given higher priority over the configured manager name and port in the script API sendTrap methods.
- Proper conversion has been provided while responding to hexa-decimal values.
- The Network Simulator crashes while sending trap with varbinds having the combination of both custom value and agent value. This issue has been fixed.

SNMP Agent Simulator

Features added

- Provides a more powerful and easy to use Trap User Interface.
- Option to configure "From IP Address" for traps generated.
- Resimulation of newly added scalar/table groups in the MIB.
- Bulk modification of selected rows in a table.

Issues Fixed

- While doing a SET from the script on a table with more than one external index, the script does not work. This issue has been fixed.
- While doing a SET from the Manager on a table with more than one external index, the SET was not implemented. This issue has been fixed.
- While executing a script with condition to execute as "row status = 6", the script was not getting executed because the request was executed before processing the script. This issue has been fixed by processing the script before executing the request.
- The SNMP GETNEXT responses were sometimes incorrect for tables with more than one index. This issue has been fixed.
- When a SET request was made on a MIB variable having 'write-only' access, the value could not be SET. Now, it is possible to write MIB variables with Max-Access = write-only.
- When simulating SNMP v2 Traps defined in the MIB, the SNMP Trap OID was taken as the Notification group OID instead of the Notification node OID. Now, it allows to configure Notification node OIDs.
• While configuring the action "Send Trap" with trap version Snmp v2c, in the Behavior simulation dialog, the SNMP Trap OID was taken as the Notification group OID instead of the Notification node OID. This issue has been fixed.

• An exception was thrown while executing the following API methods in XML mode. This issue is now fixed. getAllTableValues( ), deleteRowWithValue( ), insertRow( ).

• If Timer based scripts were configured to be executed when the agent starts, the scripts were not executed. This issue has been fixed.

• If Timer based scripts were configured to be executed while loading the configuration, the scripts were not executed. This issue has been fixed.

• The issue with setting Agent Id for Timer based scripts has been fixed.

• If timer based traps were configured for a variable, the traps were executed when a request was made on the variable for which the timer trap was configured. The issue is now fixed and timer traps will not be executed on request.

• While configuring trap varbinds, modifying the Agent value to Custom value does not reflect the changes made in the trap. This issue has been fixed.

• Problem with loading MIBs when there is a mismatch between MIB name and MIB module name, is now fixed.

General

Features added
• Support for Windows ME.

• Script API method getConfigDirectory( ) has been added for Request Based scripts and Timer Based scripts. This method will return the loaded configuration directory if configuration file is loaded and latest instance directory if MIBs are loaded.

Bugs Fixed and Features Added in 3.0 SP1

Some of the bugs identified and fixed after the 3.0 release of AdventNet Simulation Toolkit and features added have been listed under different modules:

Network Design Tool

Features added:
• Change in the representation of the Protocol supported by the device - removal of the SNMP and TL1 nodes in the tree and addition of SNMP and TL1 buttons. The selection of the button will load the configuration.

• More than one network can be started at runtime. (i.e) After starting one network, it is possible to start another network also.

• Restrictions to edit operations, new network, load network after starting the network is removed. Edit operations cannot be performed in the running network alone.

• Logging support for the network simulation.

• Find MIB Node integrated in Network Design Tool.

Bugs Fixed:
• Issue in "From IP Address" while sending traps using "Send" button.

SNMP Agent Simulator

Features added:
• Provides a more powerful and easy to use Trap User Interface.

• Option to configure "From IP Address" for traps generated.
• Resimulation of newly added scalar/table groups in the MIB.
• Bulk modification of selected rows in a table.

**Bugs Fixed**
• Fix for the issue while doing a SET from the script on a table with more than one external index.
• Fix for the issue while doing a SET from the Manager on a table with more than one external index.
• Fix for the issue while executing a script to set a table instance with row status = 6.
• Fix for the issue with GETNEXT responses of the simulator for tables with more than one index.

**General**
• Support for Windows ME.

**3.0 Release Features**

**Network Design Tool**
Integrated Network Design Tool with dynamically configurable SNMP and TL1 agents.
• Full TL1 support in Network Design Tool.
• Storage options in Database and XML mode.
• Improved wizard to add multiple devices at one shot.
• Configure SNMP agents for individual devices : SNMP configurable operations like recording from real agents, configuring traps, simulating behavior.
• Bulk modification of IP address and port number to improve productivity.
• Simulate behavior of devices in the Network.
• Integrated Trap viewer to receive traps.
• Configure TL1 agents for individual devices like Configuration of Autonomous message in TL1.
• Feature to package a network in a zip file and install it across platforms.
• Search a particular device in the network.
• Network UI organized as pages to view number devices as user wants.
• Creating SNMP and TL1 batch files while saving to start the network directly from command line.
• Performance tuning wizard to increase the response rate when simulating large networks.

**Network Simulator**
• Simulate multiple SNMP and TL1 agents in the network
• XML mode supported in Network Simulator
• Behavior of the devices in a network made possible through scripts
  Eg: You can update value of one agent from another

**SNMP Agent Simulator**
Intuitive UI to generate jython scripts used to simulate behavior in agents
• Add Multiple rows to tables in a single shot
• Bulk modification of columnar nodes
• Script API with new methods
TL1 Agent Simulator

- Support for Multiple Target Identifiers
- Multiple User name and Password support
- Access Identifier and Message Payload block support
- Support for the acknowledgment messages.
- Support for COMPLD / DENY Completion Codes
- Support for Severity Code in Autonomous Messages
- Multiple Text Block support for single Autonomous code
- Sending Request-Based Autonomous Messages
- Support Timer-Based Autonomous Messages
- Support for the Quote /UnQuote type of lines in Response and Autonomous messages
- Support for the echoing of the Input messages
- Save the Configuration from XML to Database
Known Issues

We, at AdventNet take all efforts to ensure that our customers do not waste their time and resources unnecessarily. This section lists out the Known Issues available with the product at the time of the release.

- Network Simulator
- Topology Editor
- SNMP Agent Simulator
- TL1 Agent Simulator
- Trap Stormer
- Trap Recorder
- Harmless Exceptions

Network Simulator

- When the network is successfully started, the message "Network started successfully" is displayed. Sending queries to the simulator before this will cause the simulator to crash.
- In Solaris OS, the system will be restarted when the network is stopped.
- In network creation, while specifying the unique values, if you choose RANDOM simtype, more than one value must be specified. Specifying only one value, will throw exception and the network creation will be affected.
- If a WALK file with mismatched datatype is used to create a device, the device will not be created properly.
- The simulator allows to set both the following types of value for atTable : 1.1.127.0.0.1 and also 1.127.0.0.1
- If Start Network operation fails to start the network devices for some reason, doing the Start Network operation again, will not list the network name in the "Select the Networks" dialog. As a work around, you can do Stop Network operation and then do the Start Network operation.
- During bulk modification of ipaddress for a specified range of devices, if a device with new ipaddress is already present in the network, this will be ignored by the simulator and the same ip will be assigned. This will result in two devices having the same ip.
- Modification of timer traps in template devices will create duplicate varbinds.
- Sending traps using the "Send" button, will not send the traps with Global IP address.
- While configuring a script file, if a script with the same name already exists in the configuration directory, this old script is updated without any intimation or warning message.
- UpInterface() and DownInterface() script APIs will not work.
- User codes added inside the generated script will be lost if the script is once again loaded and generated from the Behavior Simulation Wizard.
- The script configured for External index node will not be executed.
- While connecting to a CLI agent, no characters should be entered before the login prompt appears.
- The "no" command in vlan mode may cause the network simulator to crash.
- Specifying invalid values to the IOS commands, may cause the Network Simulator to crash.
• When a query is made on a TL1 command code with completion code configured as 'DENY', the error codes and error messages displayed in the response are different for each query.

• Deleting an accessId and message payload present in the device tree, from the network device, will not delete the same. The response will be generated.

• It is not possible to connect to the TL1 agent using Telnet from Windows machine.

• Unquoted message is not supported in Valid response.

• Error response supports only Quoted lines.

• If script files are configured to be executed after the upload/download of files in TFTP server, the script is executed even when the file is not transferred.

• Performing a FTP download from Windows to Linux OS by specifying a non-existing file, will hang the FTP client dialog.

• When a device with Telnet support is copied and pasted, the pasted device will not have telnet support.

• The response time for the telnet device increases when the number of telnet devices started is greater than 250.

Topology Editor

• If a switch (s1), which has one or more devices connected to it, is connected to a switch/router, the devices connected to s1 will not be displayed. To display the devices, the router to which s1 is connected must be clicked again in the router tree on the left-side.

• If a switch (s2) is connected to switch (s1), and s1 is connected to a router, disconnecting s1 from the router, will not display s1 and s2 in the remaining device panel, as s1 and s2 are still inter-connected.

SNMP Agent Simulator

• In the evaluation version, the first two group nodes will respond based on your query. If there are a lot of OIDs between the first group node and second group nodes, a MIB walk may time-out. This is because of the evaluation restriction on two group nodes.

• When configuring Simulation type for a counter64 data type the value will rollback to the starting value when the value reaches 2^63 - 1. Where according to the original value, it should accommodate 2^64 - 1.

• Recording values from the real agent or modification of scalar values will not invoke the Save dialog. The configuration must be saved manually.

• In Default Trap Generation there is a problem with getting varbinds in case of v2 MIBS.

• Modifying the instance value in a table in the SNMP agent simulator will not delete the instance in DB and it can be queried from manager.

• The script API methods deleteRowWithValue, getAllTableValues will not work properly for a table with external indexes.

• The Default Traps generation will have problems when loading multiple MIBs.

• The script API method GetAllTableIndices() will return only the internal indices for a table having both internal and external indices.

• Command line agent simulator is not supported in SNMP v3.

TL1 Agent Simulator

• If the product is using JDK1.3 or JDK1.3.1 in Linux OS, starting the TL1 agent in TCP port, will not free the port even if the agent is stopped.

• If the Command Code have both Valid and Error Responses in the TCS, the Simulator simulates only the Valid Response lines by default.
- If a new access identifier / message payload is added after configuring an autonomous message for a command code, then the autonomous message configured will not be reflected in the run-time.

- While configuring unquoted response lines, the parameter values must not have any white spaces.

- If the echo codes is configured for a command code, then while adding an access identifier or message payload block through scripts, the echo codes will not be reflected for the newly added access identifier / message payload.

- Changing the Target ID after starting the agent, will not be reflected in the Timer Based Autonomous Messages. The new Target Id will be reflected in the Timer Based Autonomous Messages only after querying a command code,

- The script API methods `getLine()` and `deleteLine()` for the Response block will get/delete the line irrespective of whether it is quoted or unquoted.

- When a query is made on a TL1 command code with completion code configured as 'DENY', the error codes and error messages displayed in the response are different for each query.

**Trap Stormer**
- The command line trap stormer will not exit by itself after completion of sending of the traps.

**Trap Recorder**
- In SNMP Trap Recorder, even if all the traps in the dialog are deleted, clicking on the **Save** button will ask for a file name. If the file name is specified, the deleted traps will be saved in the file.

**Harmless Exceptions**

**During Runtime for JDK version after 1.2.2**

Exception occurred during event dispatching:
java.lang.NullPointerException
at javax.swing.plaf.basic.BasicTableUI.paintCell(Unknown Source)
at javax.swing.plaf.basic.BasicTableUI.paintCells(Unknown Source)
at javax.swing.plaf.basic.BasicTableUI.paint(Unknown Source)
at javax.swing.plaf.ComponentUI.update(Unknown Source)
at javax.swing.JComponent.paintComponent(Unknown Source)
at javax.swing.JComponent.paint(Unknown Source)
at javax.swing.JComponent.paintChildren(Unknown Source)

**Note:** It is not advisable to change the JDK version of the product.

**Exception while adding a row in the Table**

java.lang.NullPointerException
at javax.swing.plaf.basic.BasicTableUI.paintCell(Unknown Source)
at javax.swing.plaf.basic.BasicTableUI.paintCells(Unknown Source)
at javax.swing.plaf.basic.BasicTableUI.paint(Unknown Source)
at javax.swing.plaf.ComponentUI.update(Unknown Source)
at javax.swing.JComponent.paintComponent(Unknown Source)
at javax.swing.JComponent.paint(Unknown Source)
Limitations

This section lists out the Limitations available with the product at the time of the release.

SNMP Agent Simulator

- The SNMP Agent Simulator does not support a configuration in which two nodes have the same name but different OIDs.
- SNMP V3 Traps cannot be sent using the Trap Stormer.
- Specifying negative values for Total traps in the Generate Traps feature, will generate infinite number of traps.
- The values for DisplayString type should not have pipe (|) symbols.
- Opaque data type is not supported.

TL1 Agent Simulator

- Simulation of same Command/Autonomous Codes from different TCS file(s) is not supported.
- Autonomous codes without textblock not supported.
- Simulation of TCS file which has same Response Name with different Parameters is not supported.
- The TL1 Agent Simulator supports only TCP protocol.
- If the isMPBExists() script is configured for a command code, then the message payload in the script should be name defined or position defined as it is defined in the TCS file. Else incorrect response is displayed.
- For deleteLineWithValue() script if the line to be deleted contains different delimiters, then the line will not be deleted.

Network Simulator

- Adding a instance with length > 255 will create problems in the mysql database.
- Template based bulk modifications cannot be performed on a TL1 device.
- Inform feature is not supported in Solaris OS.
- The process of adding bulk devices in the network, cannot be stopped intermediately (when creating the network through Network Design Wizard).
- If the isMPBExists() script is configured for a command code, then the message payload in the script should be name defined or position defined as it is defined in the TCS file. Else incorrect response is displayed.
- For deleteLineWithValue() script if the line to be deleted contains different delimiters, then the line will not be deleted.
- Opaque data type is not supported.
- The maximum number of TFTP servers that can be started is 750.
SNMP Agent Simulator Guide

The SNMP Agent Simulator, bundled with Simulation Toolkit, is a platform-independent JAVA-based tool, which is used to simulate SNMPv1, SNMPv2C and SNMPv3 manageable devices. You can run this tool on Windows / Linux / Solaris and simulate one agent at a time. The SNMP Agent Simulator takes the MIB file as input and simulates instance variables for the MIBs. These instance variables are saved into a file and the agent responds from this file. You can create default variables from the MIB or learn variables from an existing agent.

The use of Jython-based scripting allows for advanced modeling of agent behavior, trap generation, creation of error scenarios, and expression of inter-relationships between MIB variables.

This section covers the following topics:

- Getting Started
- Simulating Default Values
- Configuring Values
- Resimulating Values
- Saving Agent Configuration
- Starting and Stopping the Agent
- Testing the Agent
- Package and Install SNMP Agent Configuration
- Stopping the Remote Agent
- Simulating a V3 Agent
- Starting the SNMP Agent from Command Line
Getting Started

- Starting SNMP Agent Simulator
- User Interface

Starting SNMP Agent Simulator

The SNMP Agent Simulator can be started in the following ways:

- From the Simulation Toolkit Launcher by invoking `<Simulator_Home>/SimulationToolkit.bat/sh` file
- By invoking `SnmpAgentSimulator.bat` from `<Simulator_Home>/bin` directory, for Windows OS.
- By invoking `SnmpAgentSimulator.sh` from `<Simulator_Home>/bin` directory, for Linux/Solaris OS.

The simulator will store the simulated data in MySQL database. The MySQL database is bundled with the product and is part of the package. Refer to MySQL for more information. By default, the following details will be stored in `database_params.conf` file in `<SimulatorHome>/conf` directory.

<table>
<thead>
<tr>
<th>User Name</th>
<th>root</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>(No password is configured. Leave the field blank)</td>
</tr>
<tr>
<td>Server Name</td>
<td>localhost</td>
</tr>
<tr>
<td>Server Port</td>
<td>2306</td>
</tr>
<tr>
<td>MySQL Home</td>
<td>./mysql/bin shows the MySQL bin in the product directory.</td>
</tr>
</tbody>
</table>

In Linux OS, it is not necessary to specify the MySQL Home path.
User Interface

A general introduction on the UI of the SNMP Agent Simulator will help you to get familiar with the tool.

The image above depicts the primary window of AdventNet SNMP Agent Simulator. It consists of a menu bar, toolbar, left frame, and a right frame.

The menu bar displays a list of commands to perform various operations.

The toolbar displays buttons with images that act as shortcuts to the menu options.

The left frame holds the MIB tree. A MIB tree is a structure that displays all the loaded MIBs. The MIB tree enables to traverse the tree, view the loaded MIBs, and the definition of each node.

The right frame contains text fields to specify the basic parameters of the selected MIB node such as OID, access, syntax, node description, simulation type, scalar and table value and so on. It also contains a text area to display the results.
Simulating Default Values

Overview

The first step in simulating an SNMP agent, is to load the MIB file and simulate default values for the MIB. MIBs are specifications containing definitions of management information so that network systems can be remotely monitored, configured and controlled. The rules for writing MIBs are defined in a collection of documents called the Structure of Management Information (SMI).

In this topic, you will learn to create a basic simulation by loading MIB(s) in SNMP Agent Simulator and simulating default values for the MIB. If you already have a saved configuration of the simulated SNMP agent, refer to Loading Agent Configuration.

Loading MIB

You can load any MIB which follows the SMI standards to simulate a new SNMP agent with default values. To load a MIB file(s),

- Choose **File -> Load MIB File** option from the menu bar. The **Choose MIB file** dialog appears displaying all the MIB files.
- Select the MIB file(s). To load more than one file, select the files that you want to load with the Ctrl and Shift key depressed.
- Click on **Load MIBs** to load the MIB file in simulator. If the MIB is loaded successfully, you will find the MIB added to the MIB tree. This completes the MIB loading operation. If the MIB file cannot be loaded, the message "Error in loading MIBs" will be displayed and the MIB will not be added in the MIB tree.

When you have completed loading the MIBs, the next step is to simulate default values for the MIB nodes.

Simulating Default Values

- On successful loading of the MIB(s), the simulator comes up with a dialog prompting if you would like to simulate default values for the loaded MIB(s). To simulate values, Click Yes. (or)
- You can also choose **Operations -> Simulate values** from the menu bar or Simulate default values icon from the toolbar, to simulate default values for the loaded MIB.
- The simulator prompts another dialog asking where you like to store the simulated values.
- You can provide any desired table name. Example: sim_table1
- The simulated values will be stored in the table.

Now, you have created a basic simulation of an SNMP agent with default values. A default value is assigned to every scalar node depending on the syntax of the node. In case of SNMP Tables, a single row will be simulated by default for every SNMP table present in the loaded MIB.
Starting and Testing the Agent

To test the simulated agent,

- Start the Agent. Refer to the topic, Starting and Stopping the Agent.
- Query the agent from the Mib Browser or any SNMP Manager application. Refer to the topic, Testing the Agent.
- The simulated agent will respond with default values.

Any variation on this basic simulation can be created for use with the SNMP Agent Simulator. Refer to section Configuring Values, to learn more.

Loading Agent Configuration

The option Load Agent Configuration can be used, if you already have a saved configuration of the SNMP agent and you want to make changes to the existing configuration file.

To load an existing configuration file in the simulator,

- Choose File -> Load Agent Configuration from the menu bar or click on the Load Configuration icon. This will bring up the configuration loading dialog. Browse to select the configuration file. Example: The configuration file would be saved as <yourfilename_main.xml>
- Click Load to have the configuration loaded.
Configuring Values

When a MIB is loaded in the SNMP Agent Simulator and default values are simulated, the MIB data is updated in Database format, based on the chosen mode of operation. Any variation on this basic simulation can be created for use with the SNMP Agent Simulator.

The SNMP Agent simulator provides an intuitive UI to record values from a real agent, configure scalar and table values, populate the SNMP table, configure traps and informs, configure real time behavior of devices using Jython scripts, simulate error conditions and scenarios and so on.

Refer the links given below to learn about the configurations in detail.

- Recording from Real Agents
- Configuring MIB Values
- Populating the SNMP Table
- Configuring Traps
- Configuring Informs
- Configuring Behavior
- Configuring Error Conditions
- Configuring Agent Settings
- Customizing the SNMP PDU
Recording from Real Agents

Overview

The Recording wizard allows to easily create simulations of actual devices. Once data is captured from a device, any variation on this basic simulation can be created for use with the SNMP Agent Simulator and Network Designer. The wizard is easy-to-use and user-friendly.

You can save the recorded configuration and create multiple copies of the same to simulate a network in the Network Designer. The recorded configuration can also be suitably modified to re-create problems or test configuration variations.

Recording from Real Agents

Select **Configure -> Record from a Real Agent** option from the menu bar or click on the record icon from the toolbar. The Recording Wizard is displayed, wherein you have to specify the following details.

- **Agent Name/IP**: You can specify either the machine name or IP address of the machine. For example, agent-test or 192.168.215.1
- **Port Number**: The port on which the real agent is running. The default port is set to 161.
- **Community**: Specify community if defined or else the default community is taken to be public.
- **SNMP Version**: Specify the SNMP agent version. The support versions are V1, V2c and V3.
  - If the real agent version is **SNMP v3**, specify the v3 user details like user name, context name, security level, auth / priv protocol values and password in the `v3recording.conf` file available in `<SimulatorHome>/conf` directory.
  - **Time out (in secs)**: If the real agent is not reachable, how long should it try to maintain the connection.
  - **Retries**: If the real agent takes time to respond, how many times should the simulator try again.
  - **Record all data**: Select this option to record all the OIDs in the real agent. If this option is not selected, the starting oid and ending oid option will be enabled.
  - **Starting OID**: Browse to specify where you want to start recording from by specifying the starting OID. For example: 1.3.6
  - **Ending OID**: You can browse to specify the Ending OID. However, it is optional. Learning from the real agent happens until end of MIB when the ending OID is not specified.
- **Generate Debug message while recording**: You can also generate debug messages to trace the proceeding of the recording from the real agents. The log file will be generated in the name of the host specified in the ./logs directory.
• Click **Record** to record from the real agent. The progress bar shows the rate at which information is read from the real agent and is updated in the simulator. When the recording is successfully completed, the following message is displayed. Click **OK** to exit from the dialog.

• If the simulator encounters any errors while updating the recorded data, the following message is displayed after the recording process is completed. Click **Yes** to view the log messages.

• If the real agent is not running at the specified port, the message *Recording Terminated because of Errors* will be displayed in the Error dialog as shown in the screen-shot below:
• In case you want to stop the recording of the real agent, click Stop to abort process.

On recording the real agent configuration, the configuration must be saved.

Log Messages

The errors encountered by the simulator, while updating the recorded values are captured in a recordError.log file, in <Simulator Home>/logs directory. The following messages are captured in the log file as shown in the screen shot below:

• Real Agent Details
• Empty values obtained
• Data Type mismatch between the Real agent and the MIB.
• Index Mismatch
• The scalar and table groups actually recorded from the real agent.
Real Agent Details

The log file captures the details about the real agent being recorded, such as Agent Name, Agent Port, Agent Community, Agent Version, Timeout value and number of retries.

Empty Values

The log file displays the OID of the nodes which contain empty values in the real agent being recorded.
For example, if RFC1213 MIB is recorded and the value of the node `sysLocation` in `System group` does not contain any value, the following error message is displayed: *Empty value obtained for:.1.3.6.1.2.1.1.6.0*

**Data Type Mismatch**

If there is a data type mismatch between the real agent and the MIB loaded in the simulator, the simulator will record it as empty value. The log file captures any data type mismatches between the real agent and the MIB loaded in the simulator.

For example: The syntax of the scalar node `sysObjectID` with OID `.1.3.6.1.2.1.2.1.2.0` in the real agent is *"Object Identifier"*. In the MIB loaded in the simulator, the same OID has the syntax as *"Integer"*. This error is captured and displayed in the log file.

**Index Mismatch**

If the simulator is not able to resolve the index in a particular table while updating the recorded value, an error message will be logged in the log file. The corresponding table will not be updated with the recorded data.

Example: Could not get simulated objects for the instance `.6.1.192.168.1.224` in the table with oid `.1.3.6.1.2.1.3.1.1`

**Scalar and Table Groups Recorded**

The real agent may or may not have all the scalar and table groups available in the MIB. Similarly, the MIB, may or may not have all the scalar and table groups available in the real agent. The log file captures this information and displays the scalar and table groups actually recorded from the real agent.
Configuring MIB Values

- Overview
- Configuring Values for Scalars and Tables
- Configuring SysUpTime
- Simulation Types Supported
- Configuring Simulation Type for Scalar Nodes
- Configuring Simulation Type for Tables

Overview

When a MIB is loaded in the SNMP Agent Simulator and default values are simulated, a basic simulation is created. The default value is assigned to every scalar node depending on the syntax of the node. In case of SNMP Tables, a single row will be simulated by default for every SNMP table present in the loaded MIB.

In this topic, you will learn to configure values and simulation types for each type of scalars and tables present in the MIB.

Configuring Values for Scalars and Tables

Configuring values for the MIB with default simulation types is very simple as explained below.

- To configure values for a scalar node,
  - Click on the scalar node. The default value of the node and other properties will be displayed on the right-side top panel. The description of the selected node will be displayed, by selecting the Description button, available at the right-side top panel.
  - Provide any value to the node. Press Enter Key to save the value of the node. Traversing to another node will also save the value of the node previously changed.
  - To configure values for a table,
    - Select the table node, for example adiskTable, aaplicationTable. The properties of the selected table node will be displayed on the right-side top panel. The description of the selected table column will be displayed, by selecting the Description button, available at the right-side top panel. The Configure Table button will be enabled on the right-side. Click this button to display all the existing rows in the table.
    - To configure row values, select a particular row and click on Modify Row option. Now, you will be able to configure values for each column of the row, by modifying the existing default values. Press OK to save the values.

Configuring sysUpTime

- For scalar/table nodes of type TimeTicks, you can configure the sysuptime macro as the node value, to get the agent uptime. For example: configuring $sysuptime as the value for the sysUpTime node in the system group, will return the agent uptime in the time format "0 hours, 0 minutes, 00 seconds "
  - Press Enter Key to save the value of the node. Traversing to another node will also save the value of the node previously changed.
Supported Simulation Types

The SNMP Agent Simulator provides the following simulation types for configuring the value of a node. The value simulated for a node will depend upon its simulation type. If no simulation type is specified then the simulator will assume a default value consistent with the syntax of the node. You can associate the following simulation type to the MIB nodes. The MIB nodes will take the values of the simulation type.

The following table provides details about the Simulation Types supported for each Syntax type.

<table>
<thead>
<tr>
<th>Syntax Type</th>
<th>Simulation Types Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>OCTET STRING</td>
<td>CONST, RANDOM</td>
</tr>
<tr>
<td>OBJECT ID</td>
<td></td>
</tr>
<tr>
<td>IPADDRESS</td>
<td></td>
</tr>
<tr>
<td>Counter and Counter64</td>
<td>CONST, RANDOM, Exponential, Wave, Linear, Sine Wave</td>
</tr>
<tr>
<td>Gauge</td>
<td></td>
</tr>
<tr>
<td>Integer</td>
<td></td>
</tr>
</tbody>
</table>

**Constant**: The agent will always respond with the given constant value.

**RANDOM**: The agent will respond with a randomly selected value from the given set of values. For e.g. if the values given for the DisplayString sysLocation are I floor, II floor, III floor and IV floor, then the agent for every GET will randomly select one of the four values. Multiple ranges can be given.

Random can be classified into three types

- Simple Range
- Delta varied ranges
- Enumerated range

<table>
<thead>
<tr>
<th>Syntax Type</th>
<th>Simulation Types Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type I: Simple Range</td>
<td>DISPLAY STRING, OBJECT ID, IP ADDRESS</td>
</tr>
<tr>
<td>Type II: Delta varied ranges</td>
<td>COUNTER &amp; COUNTER64</td>
</tr>
<tr>
<td>Type III: Enumerated range</td>
<td>GAUGE, INTEGER, COUNTER</td>
</tr>
</tbody>
</table>

**LINEAR**: The Linear type is generated using the formula \[y = m * x + c\] The required input values being m and c. For each request the value will be incremented linearly.

**WAVE**: The wave type is generated using the formula \[y = a \cdot \cos (w \cdot t)\] The required input values are a and w. The response if plotted will take the form of a wave

**SINE WAVE**: The sine wave would generate using the formula \[a \cdot \sin (w \cdot t)\] The required input values would be a and w.

**EXPONENTIAL** The exponential type is generated using the formula \[y = a \cdot e^{b \cdot x}\] The required input values are a and b. For each request, the value will be incremented exponentially.
Configuring Simulation Types forScalar Node

The SNMP Agent Simulator will assign a value to every scalar node present in the loaded MIB(s). This value will depend upon the syntax of the node. The simulation type will be by default CONST (except in the case of nodes of type COUNTER or COUNTER64 where the default is LINEAR). The MIB nodes will take the values of the configured simulation type Refer table for details on simulation types supported for a particular syntax type.

Configure values for Simulation Type Constant

You can associate the simulation type CONST to a MIB node, the value that you specify will be associated with the MIB node.

1. Choose a Scalar Node from the MIB loaded. When the scalar node is selected, the main screen displays the properties of the Node.
   - Object ID: This field displays the corresponding OID of the selected Node.
   - Max-Access: The field shows the Maximum Access allowed on the MIB node.
   - Syntax: This specifies the data type and the type of value it can accept
   - Description: This gives a small description about the node

2. The simulation types available are listed in a drop down list. Select CONST from the list. You can provide any value to this MIB Node like simulatortest.
   Example: You can associate the MIB node sysDescr with simulatortest. Whenever the MIB node is queried it will constantly respond with the configured value simulatortest.

Configure values for Simulation Type Linear

You can associate the simulation type Linear to the MIB node. The value that you specify will be associated with the MIB node.

1. Choose a Scalar Node from the MIB loaded. When the scalar node is selected, the main screen displays the properties of the Node.
   - Object ID: This field displays the corresponding OID of the selected Node.
   - Max-Access: The field shows the Maximum Access allowed on the MIB node.
   - Syntax: This specifies the data type and the type of value it can accept
   - Description: This gives a small description about the node

2. The simulation types available are listed in a drop down list. Select Linear from the list.

3. The linear type simulation is based on the formula \[ y = m \times x + c \]

4. You will have to provide the values for \( m \) and \( c \).

5. The response when plotted would be a straight line.

Configure values for Simulation Type Wave

You can associate the simulation type Wave to the MIB node, the value that you specify will be associated with the MIB node.

1. Choose a Scalar Node from the MIB loaded. When the scalar node is selected, the main screen displays the properties of the Node.
   - Object ID: This field displays the corresponding OID of the selected Node.
   - Max-Access: The field shows the Maximum Access allowed on the MIB node.
   - Syntax: This specifies the data type and the type of value it can accept
   - Description: This gives a small description about the node

2. The simulation types available are listed in a drop down list. Select Wave from the list.

3. The wave type simulation is based on the formula \[ y = a \times \cos ( w \times t ) \]

4. You will have to provide the values for \( a \) and \( w \).

5. The response when plotted would be in the form of a wave.
Configure values for Simulation Type Sine wave

You can associate the simulation type Sine Wave to the MIB node, the value that you specify will associated with the MIB node.

1. Choose a Scalar Node from the MIB loaded. When the scalar node is selected, the main screen displays the properties of the Node.
   - Object ID: This field displays the corresponding OID of the selected Node.
   - Max-Access: The field shows the Maximum Access allowed on the MIB node.
   - Syntax: This specifies the data type and the type of value it can accept
   - Description: This gives a small description about the node Select the a MIB node from the tree.

2. The simulation types available are listed in a drop down list. select Sine wave from the list.

3. The Sine wave type simulation is based on the formula \[ a \cdot \sin ( w \cdot t ) \]

4. You will have to provide the values for \( a \) and \( w \).

5. The response when plotted would be in the form of a Sin wave.

Configure values for Simulation Type Exponential

You can associate the simulation type Exponential to the MIB node, the value that you specify will associated with the MIB node.

1. Choose a Scalar Node from the MIB loaded. When the scalar node is selected, the main screen displays the properties of the Node.
   - Object ID: This field displays the corresponding OID of the selected Node.
   - Max-Access: The field shows the Maximum Access allowed on the MIB node.
   - Syntax: This specifies the data type and the type of value it can accept
   - Description: This gives a small description about the node Select the a MIB node from the tree.

2. The simulation types available are listed in a drop down list. select Exponential from the list.

3. The Exponential type simulation is based on the formula \[ y = a \cdot e ^ { b \cdot x } \]

4. You will have to provide the values for \( a \) and \( b \).

5. The response when plotted would be in the form of a Exponential curve.

Configure values for Simulation Type Random

Whenever the simulation type is selected as RANDOM the "Range" button below the "Value" text field is enabled. The dialog that is displayed on clicking this button will depend on the type of the selected node.

Type I : Simple Range

1. Select the a MIB node from the tree of any one of the syntax types DISPLAY STRING, OBJECT ID, IP ADDRESS.

2. The simulation types available are listed in a drop down list. select Random from the list.

3. Click Range to specify the Random simulation type

4. In this type you have two fields value and values specified. Provide a string eg. If the MIB node is sysLocation, you can provide a value " I floor ", in this manner you can specify as many values as you want.

5. When queried it will respond with any one value from the list.

Example: If the values for sysLocation are specified as Floor I, Floor II and Floor III, The values in the list {Floor I ; Floor II ; Floor III} will be associated with the node randomly.
Type II : Delta varied ranges

1. Select a MIB node from the tree with syntax type COUNTER or COUNTER64.
2. The simulation types available are listed in a drop down list. Select Random from the list.
3. Click Range to specify the Random simulation type.
4. Specify a starting value say 10 and ending value say 100.
5. In this case you have to provide a delta variable. A Delta Variable is a random number generated between zero and the delta value, and this value is used as difference value in any one of the ranges that is again selected at random.
6. Click Add to add to Ranges specified list.
7. You can add as many ranges as you want. The MIB node would be randomly associated with a random value within the ranges specified.

Example:

If the Delta value is 5 and the Ranges Specified are: 10-20; 21-30, then every time a random number say 3,1,2 is generated between 0 and 5 that is used as difference value in the ranges, the output values would be 3,4,6 on Range 1 and 24,25,27 on range 2 and the ranges are again selected at random.

Type III Enumerated Range

1. Select the a MIB node from the tree of any one of the syntax types GAUGE, INTEGER, COUNTER
2. The simulation types available are listed in a drop down list. Select Random from the list.
3. Click Range to specify the Random Simulation Type.
4. Specify a starting value say 10 and ending value say 100.
5. Click Add to add to Ranges specified list.
6. You can add as many ranges as you want, the MIB node would be randomly associated with a random value within the ranges specified.

Note: You can also specify a starting value alone.

Configuring Simulation Type for Tables

To configure simulation types for Tables,

- Choose the Table node from the displayed MIB nodes in the MIB tree. The Configure Table button gets enabled denoting that you can perform the table operations.
- Select the Configure Table Button. All the existing rows in the table will be displayed.
- Selecting a row will display all the column names and values of that row. You can configure simulation type and value for each column of a row in the Table., add new row, Add multiple rows, modify multiple rows and delete rows from the Table.

Refer the topic, "Populating the SNMP Table" to know more about the table operations.
Populating the SNMP table

- **Overview**
- **Adding Rows**
  - To add a single row with default and configured values
  - To add multiple rows with default and configured values
- **Simulating Holes in SNMP Tables**
- **Modifying Rows**
  - To modify a single row
  - To modify a column value for all rows
  - To modify a column value for selected rows
- **Deleting row(s)**
- **Finding a Row**

**Overview**

In the previous topic, you learnt to configure values and simulation types for Scalar and Table nodes in the MIB. In this topic, you will learn about the operations that can be performed on a table. The table operation will be useful when you have large number of rows in the table. You can populate the table, view existing rows in the table, modify existing values in the table, delete rows in the table and display the *Number of rows* present in the table.

**Adding Rows**

1. Select the Table Node in the MIB tree.
2. Click on the *Configure Table* button. The Table Configuration dialog opens listing the default rows available in the table. A maximum of 15 rows will be displayed in the table at a time. You can use the navigate button to navigate the table.

To add single row

1. To add a single row with default values, click *Add Row* button. Now, the "Add/Modify Entry" dialog displays the new row values. Click OK to add the row to the table.
2. To add a single row with configured and default values, click on *Add Row* button. Now, the "Add/Modify Entry" dialog displays the new row values. Configure values for all or any column. Click OK to add the row to the table.

**Note:**

1. If the index field is of type "Integer", the index column will be automatically incremented by 1, with other column values as default. (or)
2. If the index field is of type "string", the index value must be manually modified with a new unique value. Now, click OK to add the row to the table.
3. For nodes of type *TimeTicks*, you can configure the sysuptime macro (*$sysuptime*) as the node value, to get the agent uptime in the time format "0 hours, 0 minutes, 00 seconds"
To add multiple rows

1. To **add multiple rows with default values**, click **Add Multiple Rows** button. The **Add Multiple Rows** dialog will be displayed. Specify the total number of rows to be added and the starting index value. Click **OK** to add the rows to the table.

2. To **add multiple rows with configured and default values**, click **Add Multiple Rows** option. The **Add Multiple Rows** dialog will be displayed. Specify the total number of rows to be added and the starting index value. Click **OK** to add the rows to the table.
   - To configure values for a specific row, refer to section, **Modify a Row**.
   - To configure a column value for all rows in the table, refer to section **Modifying all Rows**.
   - To configure a column value for selected rows in the table, refer to section **Modifying Selected Rows**.

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• If the index column is of type &quot;Integer&quot;, an integer value must be specified as the starting index. The index values of the other rows will be incremented, based on this value.</td>
</tr>
<tr>
<td>• If the index column is of type &quot;Object Identifier&quot;, an OID value must be specified as the starting index. The index values of other rows will be incremented based on the given OID. Click <strong>OK</strong> to add the rows to the table.</td>
</tr>
<tr>
<td>• If the index column is of type &quot;display string&quot;, the starting index value must be a string. For example: &quot;public&quot;. The index values of the new rows will be created as public0, public1, public2 etc. Click <strong>OK</strong> to add the rows to the table.</td>
</tr>
<tr>
<td>• For nodes of type <strong>TimeTicks</strong>, you can configure the sysuptime macro ($sysuptime) as the node value, to get the agent uptime in the time format &quot;0 hours, 0 minutes, 00 seconds&quot;</td>
</tr>
</tbody>
</table>

**Simulating Holes in SNMP Tables**

The SNMP Agent Simulator, supports to simulate holes in SNMP Tables. This enables you to add empty column values while populating the SNMP tables. While querying the simulated agent values, the simulator handles the empty values as given below:

- While doing a GETNEXT/SNMP WALK, the simulator will skip the node which has an empty value.
- While doing a GET request, the simulator will throw a relevant error message.

**Modifying Rows**

**To modify a single row**

You can also modify the Simulation type and value in the column nodes. To modify an existing row.

1. Select the row you want to modify and click **Modify Row** button.
2. All the columnar nodes' Simulation type and value come up in the dialog. You can make the desired changes.
3. Click **Ok** to modify the row.

**To modify a column value for all rows**

You can modify the Simulation type and value of a column for all the rows in the table. At a time, only one column can be modified in all/selected rows
1. Click **Modify Multiple Rows**, a Modify Multiple rows dialog appears with all the columnar nodes present in the row.
2. Select a column that you want to modify. Example. ifDescr.
3. To modify all the rows of the table, select **Modify All Rows**.
4. Now you can change the value of this node to the desired value. Example : You can choose the Simulation type as Random and specify values like value1, value2 and Value3. These value will be associated at random with the MIB node.
5. This change will be immediately reflected in all the rows in the table

A general procedure in table operation is that, when you want to simulate an agent with large number of rows, you can add say 1000 rows in a single shot and modify few column values in the added rows to suit your requirement.

### To modify a column value for selected rows

You can modify the Simulation type and value of a column in multiple rows. At a time, only one column can be modified in all/selected rows

1. To modify only selected rows in the table, select the rows to be modified in the table.
2. Click **Modify Multiple Rows**, a Modify Multiple rows dialog appears with all the columnar nodes present in the row.
3. Select a column that you want to modify. Example, ifDescr.
4. Select **Modify Selected Rows** option.
5. Now you can change the value of this node to the desired value. Example : You can choose the Simulation type as Random and specify values like value1, value2 and Value3. These value will be associated at random with the MIB node.
6. This change will be immediately reflected in all the selected rows in the table

A general procedure in table operation is that, when you want to simulate an agent with large number of rows, you can add say 1000 rows in a single shot and modify few column values in the added rows to suit your requirement.

**Note:**

1. If the table has more than 15 rows, you can select only rows within the displayed 15 rows.
2. For nodes of type **TimeTicks**, you can configure the sysuptime macro (\$sysuptime) as the node value, to get the agent uptime in the time format "0 hours, 0 minutes, 00 seconds "

### Deleting row(s)

When you no longer require the row(s) you can delete them. To delete rows.

1. Select the row to be deleted. You can also select multiple rows ( use shift up/down arrow keys )
2. Click **Delete Row** button. This will delete all the selected row(s)

### Finding a row

You can find a row in the table, either by providing the row number or the Instance value.

1. In the **Table Configuration** dialog, select the **Find Row** button. The **Find Row** dialog is displayed.
2. Select **Find by Instance** or **Find by Row Number** and specify the corresponding instance value or row number.
3. Press **OK**.

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finding a row by instance is applicable only for tables having numeric indices, (i.e.) index of type Integer, Counter etc.</td>
</tr>
<tr>
<td>Finding a row by row number is applicable for all Data types.</td>
</tr>
</tbody>
</table>
Configuring Traps

Overview

In addition to retrieving data from the managed resource and sending response to them, agents also have the ability to send unsolicited messages to managers when they detect some significant event. An unsolicited message of this sort is called Trap.

You can configure the simulated agents in the network so that the agent will send traps when it satisfies the specified condition. In real world, a trap will be sent based on a real event. When simulating traps the trap needs an event. This event will act as the trigger to fire traps. In this aspect every trap that is configured is associated with a MIB node and takes the OID of the node as the associated variable. You can configure the event that should fire the trap (events : Get / GetNext / Set on trap variable). Whenever the event happens on the associated trap variable, the configured trap will be fired.

You can generate traps on the following conditions:
- Default traps
- Request based traps
- Threshold based traps
- Timer based traps

Enabling/Disabling Trap Service

The SNMP Agent Simulator/ Network Designer provides option to enable/disable the execution of traps configured in the SNMP agent, at agent runtime. If trap service is disabled, the agent will not
check for trap configurations before sending the response. Hence, you can use this option to improve the performance of the SNMP agent, if traps are not required to be executed.

By default, the service is enabled and the traps configured (if any), will be executed. To disable the service,

- Select **Operations -> Initialize Services** in SNMP Agent Simulator or select **Run-> Initialize Service** in Network Designer.
- The **Initialize Service** dialog is displayed.
- Deselect the option **Enable SNMP Trap Service**.
- Start the Agent. Now, the traps configured in the simulator, will not be executed.

**Note:**
1. The options in the **Initialize Services** dialog will be disabled when the SNMP agent is started.
2. If Trap Service is deselected, the traps configured in behavior scripts are also not executed.

### Configuring Traps

You can configure Request based, Threshold based and Timer based traps from the SNMP Agent Simulator or Network Simulator. To configure the same,

1. Choose the option **Configure --> Traps** from the menu bar or click on the Configure Traps icon in the Tool bar. The **Trap Configurations** dialog is displayed. This dialog will list the default traps defined in the MIB.
2. To configure a new trap, click **Add**. The **Configure Trap** dialog opens up with three icons displayed on the left side for Request-based , Threshold-based and Timer-based traps. Select the required icon.
3. Configure the **General Parameters** that are common for Request based, Threshold based and Timer based Traps.
4. To configure Request Based Traps, specify **Request Based Parameters**.
5. To configure Threshold Based Traps, specify **Threshold Based Parameters**.
6. To configure Timer Based Traps, refer to **Timer Based Parameters**.

### General Parameters

- **Associated Variable**: The trap variable or associated variable is a unique identifier, the SNMP operation on which will trigger the trap. Browse and select the required scalar/columnar node from the MIB tree and click OK. The OID of the selected node will be displayed as the trap variable.

- **Trap Version**: Select the Trap Version from the list displayed. If the trap version is v1, the trap type must be specified as either Generic type or Specific type. The default trap version is SNMPv1.

If the trap version is v3, the v3 agent must be started and the user name and context name of the v3 user must be specified. Please refer the topic **Simulating a v3 Agent**, to know more about starting the v3 agent, the default user names and password, and configuring new user names and password.

- **Generic Type**: If the trap type value is zero through five, then the trap is one of the generic SNMP traps defined by the snmp group of MIB-II and the Specific trap type will be zero. If the trap type value is six, then the trap is an enterprise-specific trap that is defined in a private MIB.
coldStart (0),
warmStart (1),
linkDown (2),
linkUp (3),
authenticationFailure (4),
egpNeighborLoss (5),
enterpriseSpecific (6).

- **Specific Type**: Specify an integer value for the enterprise-specific trap that is defined in a private MIB.
- **Enterprise OID**: The OID specified in the Enterprises field indicates exactly which management enterprise defines the trap.
- **Manager Name**: Specify the manager name. If the Manager is running on the same machine it will contain the value localhost. If the Manager is running on a different machine specify the manager name or IP address of that machine.
- **Manager Port**: The default manage port is 162. If the manager is running on a different port, specify the port number.
- **Community**: Specify the community name of the manager.
- **Variable Bindings**: It enables the user to add more varbinds to the trap other than the trap variables/objects. It is optional. To know more about adding Variable Bindings, refer the topic "Adding Varbinds".

### Request Based Parameters

Request based traps are traps that are generated when a request (Get and Set / Set only) is made by the Manager on the simulated agent. To configure Request based parameters, enter the following details in the check boxes:

- **Request Type**: The simulated trap needs an operation (Get or Set) on a MIB node. This will be the trigger based on which the trap will be generated when any request is made on the selected OID on which the operation will be performed. Choose the Request type from the following options:
  - Get & Set
  - Set only.
- **Total Traps**: Specify the number of traps that you want to send. Example: 100. It is optional. If total traps is not specified, the traps will be sent infinite number of times.
- **Time Interval**: You can also specify the time interval between the traps that are to be sent. The time interval option is available in Minutes, seconds and Milliseconds. Eg: 1000 Milliseconds. It is optional.

- **Click Ok** to add the request based trap.

### Threshold Based Parameters

Threshold based traps are traps that are generated when the value in the selected OID satisfies the specified threshold condition and the threshold value. To configure threshold parameters, enter the following details in the check boxes:

- **Threshold Condition and Value**: Specify the threshold condition and value based on which the trap will be sent. Threshold Condition: The threshold conditions listed depend upon the type of the selected trap variable. For a node of type Octet String, Object ID and IP Address, the conditions displayed are:
  - Equal To
  - Not Equal To
For types other than Octet String, Object ID and IP Address, the following threshold conditions are listed:

- Equal To
- Not Equal To
- Greater than
- Less than
- Custom Range: For Custom range, a trap will be sent if the value of the associated trap variable is not within the lower and upper threshold value. For example, if the lower threshold value is 10 and upper threshold value is 20, a trap is sent if the associated trap variable value is less than or equal to 10 (or) greater than or equal to 20.

Threshold Value: Specify the threshold value. For custom range, the lower threshold value and higher threshold value must be specified.

- **Total Traps**: Specify the number of trap that you want to send. Eg: 100. It is optional. If you leave the number of traps field blank, the traps will be generated infinitely.
- **Time Interval**: You can also specify the time interval between the traps that are to be sent. The time interval option is available in minutes, seconds and milliseconds. Eg: 1000 Milliseconds. It is optional.

Click **Ok** to add the threshold based trap.

### Timer Based Parameters

Timer based traps are traps that are generated at the configured time. In timer based traps the OID uniquely identifies the configured traps. To configure timer parameters, enter the following details in the check boxes:

- **Timer Details**:
  - **Staring Time**: Configure the date and time at which the simulator will start sending traps.
  - **Periodically send**: Check this option, to send timer traps on a periodical basis. Specify the duration as once in number of hour(s)/day(s). Example: You have configured Total Traps as 10 with a timer interval of 2 seconds and choose to periodically send it once in every 5 days. This configuration will generate 10 traps with a time interval of 2 secondsonce in every 5 days.
  - **Total Traps**: Specify the total number of trap that you want to send. Eg: 100.
  - **Time Interval**: You can also specify the time interval between the traps that are to be sent. The time interval option is available in Minutes, seconds and Milliseconds. Eg: 1000 Milliseconds.

Click **Ok** to add the timer based trap.

### Note:

1. By default, the traps will be sent with the IP Address of the system from which the traps are generated. To configure the "from IP Address" of the traps, refer to Configuring Global Trap Settings.
2. If the total traps and timer interval field is blank, a single trap is sent. If the Timer interval is specified as "1" and the total traps field is blank, total number of traps/Informs is taken as Infinity.
3. At Network Simulator startup, the timer traps with elapsed time will be executed.
4. Selecting the device right-click option SNMP -> Stop timer Traps will stop the execution of timer traps in Network Simulator. To stop the execution of all timer traps in all devices, right-click on the blank network area and select Stop All timer traps.
Adding Variable Bindings

Variable Bindings or Varbinds is an additional information that is sent to the manager along with the trap or inform. It can be value of the selected MIB nodes. This value is sent to the manager along with the trap or Inform. By default, SysUpTime and SNMP Trap OID is sent as varbinds for v2 Traps.

The Varbinds dialog can be invoked from the add/modify trap dialog and also from the trap configuration dialog.

To configure varbinds,

- Click **Configure** button in the Trap Configuration wizard. The VarBinds dialog pops up with the MIB tree displayed on the right-side.
- Select a varbind (only leaf nodes) in the MIB tree and click **Add**. The Configure VarBinds dialog pops up displaying the OID and instance of the selected varbind.
- You can specify the varbind value with any one of the following options:
  - **Agent Value**: Agent value refers to the value of the node configured in the simulated agent. You can also configure to send traps with updated agent values.
  - **Agent Macro**:
    - **SysUpTime**: Selecting this macro will return the sysUpTime value of the agent as the varbind value.
    - **CurrentTime**: Selecting this macro will return the current time as the varbind value.
  - **Custom Value**: If custom value is selected, the value must be specified in the appropriate format based on the selected node.
- Click **OK** to add the varbind(s).

To modify Varbinds,

- Select the added varbind and click **Modify**.
- Make the necessary changes in the Configure Varbinds dialog.
- Click **OK** to add the varbind with modified values.

To Delete Varbinds

- Select the varbind from the list and click **Delete** to remove the varbind.

Configuring Global Trap Settings

The SNMP Agent Simulator provides option to configure the agent IP address and manager details of traps, globally. It also provides option to send all timer traps with updated agent values in varbinds.

**Agent IP Address of Traps**

By default, traps will be sent with the IP Address or host name of the system from which the traps are generated. The simulator provides you the option to configure the IP address or host name, with which the traps are to be sent. This property will not be stored in the simulator. Every time when the simulator is invoked, the IP address/host name has to be specified. All traps will be sent from the simulated agent with the specified address.

To configure "Agent IP Address" of traps globally,

- Select **Settings -> Trap Settings** option in the menu bar. The Trap Settings dialog pops up.
- Specify the **Agent Address**. The default agent IP address is "localhost". If the specified address is not available in the network, traps will be sent with the default address.
- Click **OK** button.
Manager Details

By default, a trap will be sent to the manager name, port and community specified for each trap in the Trap Configuration dialog. If you want all the traps to be sent to the same set of managers, you can specify the manager details in the Trap Settings dialog. Adding manager details to this dialog, will send the traps only to the managers configured in this dialog.

To configure Manager Details globally,

- Select Settings -> Trap Settings option in the menu bar. The Trap Settings dialog pops up.
- Configure the Manager details like manager name, port and community, by selecting the Add button.
- Click OK button, to add the entry to the list.
- You can also delete a manager entry, by selecting the Delete button.

Sending Timer traps with Updated Agent Value in Varbinds

By default, if timer traps are configured with agent value in varbinds, the simulator will not check for the updated agent value, every time the trap is sent. Hence, all the timer traps will be sent with the agent value which is read at the startup time.

Selecting the option Sending timer traps with updated Agent values, in Settings -> Trap Settings dialog, will send the timer trap every time with the updated agent value. Since, the simulator checks for the agent value each time the trap is sent, performance will be reduced to some extent depending on the number of times the timer trap is sent.

Loading the Recorded Traps

You can record SNMP v1/v2c traps with the Trap Recorder. The recorded traps can be stored in XML files and loaded in the trap configuration dialog of the SNMP Agent Simulator and Network Designer as request-based, threshold-based and timer-based traps.

Refer the topic, Trap Recorder -> Replaying Traps in Simulator

Sending Traps

The traps configured in the Trap Configuration dialog, can be sent in the following ways.

Using Send Button

By using the Send button in the Trap Configuration dialog, you can select and send any trap irrespective of the condition configured. The selected traps will be sent to the specified manager at the port specified.

Note: The traps can be sent only when the agent is started.
Sending Request Based Traps

A request based trap will be generated when a request is made on the configured node. For example: A request based trap is configured with the following values:

- Associated variable is sysUpTime of the RFC1213-MIB with OID .1.3.6.1.2.1.1.3
- Trap Version is Snmp V1
- Generic Type is coldStart(0)
- Request Type is Get and Set.
- Total traps configured is 100
- Time Interval is 2000 milliseconds.

When a Get/Set request is made from the Manager, on the variable sysUpTime, 100 SNMP v1 traps with Generic Type coldStart, will be generated with a time interval of 2 seconds between the traps. The traps will be sent to the managers running at the specified ports.

Sending Threshold Based Traps

A Threshold Based Trap will be generated when the selected OID satisfies the specified threshold condition and the threshold value. For example: A threshold based trap is configured with the following values:

- Associated Variable is ifAdminStatus of the IfTable of RFC1213 MIB with OID .1.3.6.1.2.1.2.2.1.7
- Trap Version is Snmp V1
- Generic Type is linkDown(2)
- Threshold Condition is Equal To
- Threshold Value is 2
- Total Traps configured is 10
- Time Interval is 5000 milliseconds

If the value of the OID .1.3.6.1.2.1.2.2.1.7 equals to 2, 10 SNMP v1 traps with Generic Type linkDown, will be generated with a time interval of 5 seconds between the traps. The traps will be sent to the managers running at the specified ports.

Sending Timer Based Trap

Timer based traps are generated at the configured time. For example: A Timer Based trap is generated with the following values:

- Associated Variable is sysName of the RFC1213-MIB with OID .1.3.6.1.2.1.1.5. In timer traps, the OID is specified to uniquely identify the trap.
- Trap Version is Snmp V1
- Generic Type is coldStart(0)
- Starting Time is 20 - Mar 2003 ; 17 : 00 : 00
- Periodically Send Once in 2 Days
- Total Traps configured is 10
- Time Interval is 5000

When the specified date and time is reached (20th Mar, 2003 at 17 hrs), 10 SNMP v1 traps with Generic Type coldStart, will be generated with a time interval of 5 seconds between the traps. The traps will be periodically send once in two days, to the managers running at the specified ports.

Note:
1. The generated traps can be viewed from the Trap Viewer of the Mib Browser.
2. Selecting the device right-click option SNMP -> Stop timer Traps will stop the execution of timer traps in Network Simulator. To stop the execution of timer traps in all devices, right-click on the blank network area and select Stop All timer traps option.
Modifying and Deleting Traps

You can modify an existing trap to change the configured values.

- Choose the Configure --> Traps from the menu bar or click on the Trap icon in the tool bar. The Trap Configurations dialog is brought up, displaying the traps configured.
- Select the trap that you want to modify and Click Modify. The Configure Trap dialog opens up.
- Make the required modifications and click OK.

You can delete existing traps from the Trap Configuration table.

- Choose the Configure --> Traps from the menu bar or click on the Trap icon in the tool bar. The Trap Configurations dialog is brought up, displaying the traps configured.
- Select the trap that you want to delete and Click Delete. The selected traps will be deleted.

Generating Default Traps

Default Traps are traps and notifications defined for a particular variable in the MIB. You can view and send default traps defined in the loaded MIB file using the Default Trap dialog. It will send the traps with the default values.

Note: Default traps can be sent only when the agent is started. This feature is not supported in Network Designer.

To generate default traps in SNMP Agent Simulator,

- Choose Operations --> Generate Traps from the Menu bar.
- The Default Traps Dialog opens up. The tree on the left displays all the MIB nodes along with the Traps and Notification nodes.
- When you select the defined Trap node, the following information is displayed on the right-side screen:
  - Trap Name : The name of the default trap will be displayed.
  - Varbinds : The varbinds associated with the default traps will be displayed.
  - Description : The Trap description as it appears in the MIB.
- Manager Details : You can specify the manager details, where the trap should be sent. You can specify the following details like Manager Name, Manager Port, Manager Community.
  - Example :
    - Manager Name : 192.168.215.1 or manager-simtest
    - Manager Community : public
  - You can also specify Total traps you want to send with a specified Timer interval. Example :
    - When you specify the Total Traps as 25 and Time Interval as 2000 milliseconds, then 25 traps will be sent with a time interval of 2 seconds between each trap.
    - Start the Agent at the specified port.
- Click Send to send the default traps.

Viewing Traps

The generated traps can be viewed from the Trap Viewer. The Trap viewer must be started at the specified port before sending the traps. The Trap Viewer can listen to one or more port at a time and the traps can be sent from any host.
Trap Viewer is not integrated in the SNMP Agent Simulator. To view the traps, use the Trap viewer integrated with SNMP MIB Browser. Refer to Mib Browser -> Trap Handling -> Viewing Traps section, to learn more about the Trap viewer.

The Trap Viewer is integrated in Network Designer to view the Traps received from network elements.

To open the Trap Viewer

- Click on the Trap Viewer icon on the toolbar (or)
- Choose **Configure --> Trap Viewer** menu item from the menu bar.
Configuring Informs

- Overview
- Enabling/Disabling Inform Service
- Configuring Informs
  - General Parameters
  - Request-based Parameters
  - Threshold-based Parameters
  - Timer-based Parameters
  - Adding Varbinds
- Sending Informs
- Modifying and Deleting Informs
- Viewing Inform Response

Overview

SNMP Notifications can be sent as Traps or Inform requests. Traps are unreliable because the Manager does not send any acknowledgment on receiving the Trap. So, the Agent cannot determine if the Trap was received by the Manager. However, an SNMP Manager that receives an Inform request acknowledges the message with an SNMP response PDU. If the Manager does not receive an inform request, it does not send a response. If the Agent does not receive a response at all, the inform request can be sent again.

You can generate SNMPv2c and SNMPv3 Informs, on the following conditions:

- Request-based Informs
- Threshold-based informs
- Timer based informs

Enabling/Disabling Inform Service

The SNMP Agent Simulator and Network Simulator provides option to enable/disable the execution of Informs configured in the simulator, at agent runtime. If inform service is disabled, the agent will not check for inform configurations before sending the response. Hence, you can use this option to improve the performance of the SNMP agent, if informs are not required to be executed. By default, the service is enabled and the informs configured (if any), will be executed.

To disable the service in SNMP Agent Simulator,

- Select Operations -> Initialize Services. The Initialize Service dialog is displayed.
- Deselect the option Enable SNMP Inform Service.
- Start the Agent. Now, the informs configured in the simulator, will not be executed.

To disable the service in Network Simulator,

- Select Run -> Initialize Services. The Initialize Service dialog is displayed.
- Deselect the option Enable SNMP Inform Service.
- Start the Agent. Now, the informs configured in the network simulator, will not be executed.
### Configuring Informs

Request-based informs are informs that are generated when a request (Get and Set / Set only) is made by the Manager on the simulated agent. Thus, Request-based informs will be triggered if there is a GET / SET operation from the manager on the specified OID. The Inform variable is a unique identifier based on which the configured inform will be identified.

You can configure request-based, timer-based and threshold-based informs from the SNMP Agent Simulator and Network Simulator. To configure the same,

1. Choose the option **Configure --> Informs** from the menu bar or click on the **Configure Informs** icon in the tool bar. The **Configure Inform Requests** dialog is displayed. In **Network Simulator**, you can also right-click on a device and select **SNMP --> Informs**.
2. To configure a new inform request, click **Add**. The **Add/Modify Inform Request** dialog is displayed.
3. Configure the **General Parameters** that are common for Request-based, Response based and Timer based Informs
4. To configure Request-based Informs, specify **Request-based Parameters**.
5. To configure Threshold-based Informs, specify **Threshold-based Parameters**.
6. To configure Timer Based Informs, specify **Timer-based Parameters**.

#### General Parameters

- **Variable**: The trap variable or associated variable is a unique identifier. When an SNMP operation is performed on this variable, it will trigger the Inform. Browse and select the required scalar/columnar node from the MIB tree and click OK. The OID of the selected node will be displayed as the Inform variable.
- **Version**: SNMP v2c and v3 Informs are supported. If the inform version is v3, the v3 agent must be started and the user name and context name of the v3 user must be specified. Refer the topic, **Simulating a V3 agent**, to know more about starting the v3 agent, the default user names and password, and configuring new user names and password.
- **Manager Name**: Specify the manager name. If the Manager is running on the same machine it will contain the value localhost. If the Manager is running on a different machine specify the manager name or IP address of that machine.
- **Manager Port**: The default manage port is 162. If the manager is running on a different port, specify the port number.
- **Community**: Specify the community name of the manager.
- **Varbinds**: There are two OIDs that are default varbinds in Inform. They are sysUpTime and snmpTrapOID. In addition to this, you can add extra varbinds. It is optional. To know more about adding Variable Bindings, refer the topic **Adding Varbinds**.
Request Based Parameters

Request based informs are informs that are generated when a request (Get and Set / Set only) is made by the Manager on the simulated agent. To configure Request based parameters, enter the following details in the check boxes:

- **Generate for**: The simulated inform needs an operation (Get or Set) on a MIB node. This will be the trigger based on which the inform will be generated when any request is made on the selected OID on which the operation will be performed. Choose the Request type from the following options:
  - Generate inform on Get & Set
  - Generate inform on Set only.

- **Time Interval**: You can also specify the time interval between the informs that are to be sent. The time interval option is available in milliseconds. Eg: 1000 Milliseconds. It is optional.

- **Number of times inform should be sent**: Specify the number of informs that you want to send. Eg: 100. It is optional. If you leave this field blank, the informs will be generated infinitely.

- Click **OK** to add the Request based inform.

Configuring Threshold-based Informs

Threshold-based informs are generated when the value in the selected OID satisfies the specified threshold condition and the threshold value. Thus, when the threshold condition is satisfied the inform will be sent. The Inform variable is a unique identifier based on which the configured inform will be identified.

- **Generate for**: Select *Generate Inform based on Threshold*. Specify the threshold condition and value based on which the inform will be sent.
  - Threshold Condition: The threshold conditions listed depend upon the type of the selected inform variable. For a node of type Octet String, Object ID and IP Address, the conditions displayed are:
    - Equal To
    - Not Equal To
  - For types other than Octet String, Object ID and IP Address, the following threshold conditions are listed:
    - Equal To
    - Not Equal To
    - Greater than
    - Less than
    - Custom Range: For Custom range, a trap will be sent if the value of the associated trap variable is not within the lower and upper threshold value. For example, if the lower threshold value is 10 and upper threshold value is 20, a trap is sent if the associated trap variable value is less than or equal to 10 (or) greater than or equal to 20.

- **Threshold Value**: Specify the threshold value. For custom range, the lower threshold value and higher threshold value must be specified.

- **Time Interval**: You can also specify the time interval between the informs that are to be sent. The time interval option is available in milliseconds. Eg: 1000 Milliseconds. It is optional.
• **Number of times inform should be sent**: Specify the number of informs that you want to send. Eg: 100. It is optional. If you leave this field blank, the informs will be generated infinitely.

• Click **OK** to add the Threshold-based inform.

### Configuring Timer-based Informs

Timer-based informs are generated at the configured time. In timer-based informs, the OID uniquely identifies the configured informs. Thus, when the timer condition is satisfied the inform will be sent. The Inform variable is a unique identifier based on which the configured inform will be identified.

• **Generate for**: Choose **Generate Informs based on timer**. Specify the timer configurations.
  
  o **Starting Time**: The simulator will start sending Informs at the configured time. You can use the spinner control to set the time.
  
  o **Time Interval**: You can also specify the time interval between the Informs that are to be sent. The time interval option is available in milliseconds. Eg: 1000 Milliseconds.
  
  o **Number of Informs**: Specify the number of Inform that you want to send. Eg: 100

• Click **OK** to add the Inform.

### Adding Varbinds

Variable Bindings or Varbinds is an additional information that is sent to the manager along with the trap or inform. It can be value of the selected MIB nodes. This value is sent to the manager along with the trap or Inform. The Varbinds dialog can be invoked from the add/modify trap dialog and also from the trap configuration dialog.

The following options are provided to configure varbinds for inform messages,

• **sysUptime**: Configure the required sysUptime value to be sent with the inform response.

• **snmpTrapOid**: Configure the SNMP trap oid to be sent with the inform response.

• To add additional varbinds, click on the **Varbinds** button. The Varbinds dialog pops up.

• Select a varbind (only leaf nodes) in the MIB tree and click **Add**. The **Configure Varbinds** dialog pops up with the OID and instance of the selected varbind.

• You can specify the varbind value with any one of the following options:
  
  o **Agent Value**: Agent value refers to the value of the node configured in the simulated agent.
  
  o **Agent Macro**: The following agent macros are supported:
    
    o **SysUpTime**: Selecting this macro will return the sysUpTime value of the agent as the varbind value.
    
    o **CurrentTime**: Selecting this macro will return the current time as the varbind value.
    
  o **Custom Value**: If custom value is selected, the value must be specified in the appropriate format based on the selected node.

• Click **OK** to add the varbind(s).
To modify varbinds,

- Select the added varbind and click **Modify**.
- Make the necessary changes in the Configure Varbinds dialog appears.
- Click Ok to add the varbind with modified values.

To delete varbinds,

- Select the varbind from the list and click **Delete** to remove the varbind.

### Sending Informs

The informs configured in the *Configure Inform Request* dialog, can be sent in the following ways.

#### Using Send Button

By using the **Send button** in the dialog, you can select and send any inform irrespective of the condition configured. The selected informs will be sent to the specified manager at the port specified.

**Note**: The informs can be sent only when the agent is **started**.

#### Sending Request Based Informs

Request Based Informs are generated when a request is made on the configured node. For example: A request based inform is configured with the following values with other options as default.

- Associated variable is sysUpTime of the RFC1213-MIB with OID **.1.3.6.1.2.1.1.3**
- Trap Version is **Snmp V2c**
- Request Type is **Generate Inform on Get and Set**.
- Time Interval is **2000 milliseconds**.
- Number of times inform should be sent is **5**

When a *Get/Set* request is made from the Manager, on the variable *sysUpTime*, 5 SNMP v2c informs with default varbinds, will be generated with a time interval of **2 seconds** between the informs. The informs will be sent to the managers running at the specified ports. The inform responses can be viewed from the *Inform Response* tab of the *Configure Inform Requests* dialog.

#### Sending Threshold Based Informs

Threshold-based Informs are generated when the selected OID satisfies the specified threshold condition and the threshold value. For example: A Threshold-based inform is configured with the following values with other options as default.

- Variable is ifAdminStatus of the IfTable of RFC1213 MIB with OID **.1.3.6.1.2.1.2.2.1.7**
- Version is **Snmp V2c**
- Threshold Condition is **Equal To**
- Threshold Value is **2**
- Time Interval is **5000 milliseconds**
- Number of times inform should be sent is **10**

If the value of the OID **.1.3.6.1.2.1.2.2.1.7** equals to **2**, 10 SNMP v2c informs with default varbinds , will be generated with a time interval of 5 seconds between the informs. The informs will be sent to the managers running at the specified ports. The inform responses can be viewed from the *Inform Response* tab of the *Configure Inform Requests* dialog.
Sending Timer Based Informs

Timer-based informs are generated at the configured time. For example: A timer-based inform is generated with the following values, with other options as default.

Variable is sysName of the RFC1213-MIB with OID .1.3.6.1.2.1.1.5. In timer informs, the OID is specified to uniquely identify the inform.

Version is Snmp V2c
Starting Time is 20 - Mar 2003 ; 17 : 00 : 00
Time Interval is 5000
Number of times inform should be sent is 10

When the specified date and time is reached (20 th Mar, 2003 at 17 hrs), 10 SNMP v2c informs with default varbinds, will be generated with a time interval of 5 seconds between the informs. The inform will be send to the managers running at the specified ports. The inform responses can be viewed from the Inform Response tab of the Configure Inform Requests dialog.

Note:
1. The generated inform messages can be viewed from the Inform Response tab of the Inform Response dialog or from the Tools -> Inform Response option.
2. To stop the execution of all timer based informs in all devices, right-click on the blank network area and select Stop All timer informs option.

Viewing Inform Response

When informs are sent, the SNMP Manager that receives the Inform request acknowledges the message with an SNMP response PDU.

To view the inform response from SNMP Agent Simulator,

- Choose the option Configure --> Informs from the menu bar or click on the Configure Informs icon in the Tool bar. The Configure Inform Requests dialog is displayed.
- Select the Inform Response tab. The response received from the SNMP Manager, will be listed in the table.
- To clear all the responses, click Clear.
- To close the dialog, click Close.

To view the inform responses from the Network Simulator,

To view the inform response of a specific device:

- Right-click on a device and select SNMP -> Informs or choose the option Configure --> Informs from the menu bar. The Configure Inform Requests dialog is displayed.
- Select the Inform Response tab. The response received from the SNMP Manager for the selected device, will be listed in the table.
- To clear all the responses, click Clear.
- To close the dialog, click Close.
To view the inform response from all the devices:

- Select Tools -> Inform Response. The Inform Response dialog displays the inform responses for all the devices which sent the inform messages.
- To delete a specific entry, select the entry and click Delete.
- To close the dialog, click OK.

Modifying and Deleting Informs

You can modify an existing inform to change the configured values.

- Click on the Inform icon in the Tool bar or choose Configure --> Inform from the menu.
- The Inform Configurations dialog is brought up, displaying the Informs configured.
- Select the Inform that you want to modify from the displayed list.
- Click Modify to make all the necessary modifications.
- Click OK

You can delete existing informs from the Configure Inform Requests dialog.

- Choose the Configure --> Informs from the menu bar or click on the inform icon in the tool bar. The Inform Configurations dialog is brought up, displaying the informs configured.
- Select the inform that you want to delete and Click Delete. The selected informs will be deleted.
Configuring Behavior

- Overview
- Enabling/Disabling Behavior Service
- Configuring Behavior
- Modifying and Deleting an Already Configured Script
- Modifying and Deleting Condition for Action
- Modifying and Deleting Action to Be Executed.
- Scripts in Examples Directory
- Sample Scripts

Overview

The most important part of the simulation is to set up the simulated environment. You can set up the desired environment by defining the behavior of the device, so that it will behaves like a real device. The behavior of devices is done through Jython scripts. For example, device values can be changed, rows can be dynamically added or deleted, Traps and Informs can be configured to be sent etc.

AdventNet Simulation Toolkit provides a ScriptInterface, which defines methods to perform simple operations, such as update, insert, and delete from a JPython script. The javadocs of the script API methods is provided in SNMP_TL1_SimScrIntf.html under the <simulator home>/examples/javadocs directory.

Enabling/Disabling Behavior Service

The SNMP Agent Simulator provides option to enable/disable the execution of behavior scripts configured in the simulator, at agent runtime. If behavior service is disabled, the agent will not check for script configurations before sending the response. Hence, you can use this option to improve the performance of the SNMP agent, if behavior scripts are not required to be executed.

By default, the service is enabled and the behavior scripts configured (if any), will be executed. To disable the service,
- Select Operations -> Initialize Services. The Initialize Service dialog is displayed.
- Deselect the option Enable SNMP Behavior Service.
- Start the Agent. Now, the behavior scripts configured in the simulator, will not be executed.

Note: The options in the Initialize Services dialog will be disabled when the SNMP agent is started.

Configuring Behavior

The SNMP Agent Simulator provides an intuitive UI to configure Behavior scripts. To invoke the Behavior dialog, do the following:
- Select the Device in the Device Tree for which you would like to configure Behavior.
- Click on the Behavior Simulation icon or select Tools -> Behavior option. This will display the Configure Behavior dialog.
The Configure Behavior dialog will list the configured scripts files and the defined conditions and actions to execute the script files. You can configure new Behavior, modify or delete an existing Behavior configuration, add new conditions and actions and modify existing conditions and actions, from this dialog.

**Steps in Configuring Behavior**

**Step 1: Configuring the Conditions to Execute the Script**: Here, you will define when the script should be executed. Scripts can be executed based on three conditions. Scripts can be executed based on a request, based on a threshold condition and value, or timer based.

Step 2 and Step 3 is used for generating the script file from the UI, with the necessary actions to be executed and the conditions to execute the actions (if any). The conditions and actions are defined using the Script API methods.

### Note:

Defining Step 2 and Step 3 is not required in the following cases

- If you have defined your own script file with necessary conditions and actions, you can configure this script file in step 1 and can Finish 'Behavior Configuration'.
- If your scope for defining the conditions and actions is beyond the current scope of our API, you have to write your own script. Configure this script file in step 1 and Finish 'Behavior Configuration'.
- If you want to configure the script file available in the .examples directory, you can specify this file in Step 1 and Finish 'Behavior Configuration'.

In all the above cases, the conditions and actions defined in the script file will be executed only when the condition to execute the script (specified in step 1) is satisfied.

**Step 2: Configuring the Conditions to Execute the Actions**: In this step, you can define on what conditions the actions are to be executed. You can define any number of conditions.

**Step 3: Configuring the Actions to be Executed**: In this step, you can define the actions to be executed, when the condition to execute the script and the condition to execute the action (if any) are satisfied.

**Step 1: Configuring the Conditions to Execute the script**

To configure conditions to execute the script, select the Add button in the Behavior Configuration dialog. Select any of the following conditions and configure the necessary details on the right-side of the dialog.

- Request-based Script
- Threshold-based Script
- Timer-based Script

**Request-Based Script**

1. Choose the Request Based icon that is displayed on the left of the wizard.
2. Select the MIB node and instance based on which behavior will be configured.
3. The name of the selected MIB node will be displayed as the script filename. You can also browse and select the required file, if it already exists.
4. The option Copy script file to local directory will be enabled only if you browse and select the script file from the <Simulator_Home>/examples/scripting/snmp directory. Selecting this option will copy the script file into your configuration directory.
5. To execute the script, one or more of the following conditions must be satisfied:
   - Before processing Get
   - After processing Get
   - Before processing GetNext
   - After processing GetNext
   - Before processing Set
   - After processing Set

6. Click Next in the wizard to proceed to Step 2 or click Finish to finish Behavior configuration and exit the dialog.

**Threshold-Based Script**

1. Choose the Threshold Based icon that is displayed on the left of the wizard.
2. Threshold based scripts can be configured only for leaf nodes. Select the leaf node and instance based on which behavior will be configured.
3. The name of the selected node will be displayed as the script filename. You can also browse and select the required file, if it already exists.
4. If you want the script to be copied on to the device directory, you can check the option Copy script file to local directory.
5. To execute a script, you can specify any one of the following threshold conditions and value. When the condition and value is satisfied, the script will be executed.
   1. Equal to
      - Not equal to
      - Less than
      - Greater than
      - Custom range: For Custom range, the script will be executed if the value of the MIB node is not within the lower and upper threshold value. For example, if the lower threshold value is 10 and upper threshold value is 20, the script is executed if the variable value is less than or equal to 10 (or) greater than or equal to 20.
6. Specify the threshold value. For custom range, the lower threshold value and higher threshold value must be specified.
7. Click Next in the wizard to proceed to Step 2 or click Finish to finish Behavior configuration and exit the dialog.

**Timer-Based Script**

1. Choose the Timer Based icon that is displayed on the left of the wizard.
2. The name of the script file will be displayed, based on the time specified to execute the script. You can also browse and select the required file, if it already exists.
3. If you want the script to be copied on to the device directory, you can check the option Copy script file to local directory.
4. You can specify when you want the scripts to get executed. The following options are provided:
   - **Execute scripts when agent starts**: The configured script would get executed as soon as agent starts.
   - **Execute scripts when agent stops**: The configured script would get executed when the agent stops.
- **Execute scripts at specified time:** You can specify the date and time of script execution. Choose the date and time using the spinner buttons provided. The scripts will get executed at the specified time.
- **Execute scripts after specified time after agent startup:** You can specify the time in hours, minutes and seconds after which the script will get executed after the agent startup time.
- **Periodically execute (once in):** If scripts are to be executed at specified time or after specified time, periodically, you can check this option. Specify the duration as once in number of hour(s)/day(s).

6. Specify a time interval between each script execution. Eg. Timer interval 1000 millisecs.
7. The number of times you want the script to execute at the specified time. Eg: Number of times 5.
8. Click **Next** in the wizard to proceed to Step 2 or click **Finish** to finish Behavior configuration and exit the dialog.

Example: You have configured **Number of Times** as 10 with a **Timer Interval** of 2 seconds and have chosen to **periodically send** it once in every 5 days. This configuration will execute the script 10 times with a time interval of 2 seconds, once in 5 days.

**Step 2 : Configuring the Conditions to Execute the Action**

In this step, you can define on what conditions you want the actions defined in the scripts to be executed. You can define any number of conditions.

1. Click **Add** button that brings up the **Configure Condition to be Checked** dialog. This dialog displays all the methods available for the selected node.
2. Select the desired method, for example, getCurrentInstance. The methods that start with getCurrent are applicable only to the selected agent. When you select these methods, the condition that you are specifying is for the agent. When you select other methods, such as getNodeValue, this is valid for all the network elements. You will see the select OID enabled and network elements listed in the drop-down box.
3. Select the agent from the network and specify the OID and instance, if you wish to configure behavior across the network.
4. Select the operator from the drop-down box. The list of operators are listed as follows:
   - Equal to ==
   - Not equal to !=
   - Lesser than <
   - Greater than >
   - Lesser than or equal to <=
   - Greater than or equal to >=
5. After choosing the operator, you can specify the value to which the expression should be equated to.
6. Click **OK**. You will see that the conditions that you have specified get added to the Conditions to execute actions list.
7. Click **Add** to define another condition, (or) click **Next** to proceed to **Step 3**, or click **Finish** to finish Behavior configuration and exit the dialog.
8. As already mentioned, you can specify as many conditions as you want. Hence you have to specify whether you want to
   - Match all conditions added in the list
   - Match any of the conditions added in the list
Step 3 : Configuring the Actions to be Executed

In this step, you can configure the actions to be performed when the condition to execute the script and the condition to execute the action (if any) are satisfied. Click Add to bring up the Configure Actions to be Performed dialog. Select a method from the drop-down list. Methods are listed below:

- Update value
- GetNode value
- Insert row into table
- Delete a row from table.
- Delete all rows from table
- Send trap
- Send Inform
- Generate random number
- Delay action
- Terminate operation

**Update Value**

1. Select the *update value* method.
2. When simulating behavior of devices across a network, the agents in the network are listed in the drop-down list. When you select the agent, the value will be updated in the specified agent.
3. Select the OID and instance on which it should be updated. When selecting the OID and the instance, the details of the nodes are displayed.
4. The syntax of the node is displayed along with the simulation type. You can choose the simulation type the node should have, and specify values for the simulation type.
5. Click OK to add the action to the list.

**Example**

If you select the OID .1.3.6.1.2.1.1.6, the syntax would be Display String. The options that you have with this node are constant and random. If you choose the random option, you will see the range button being enabled. Specify value and click Add so that the value will be associated at random.

**Get Node Value**

1. Select the *GetNode value* from the drop-down list.
2. When simulating behavior of devices across a network, the agents in the network will be listed in the drop-down list. You will be able to select agent from the network.
3. Select the OID and the instance.
4. Now you have to specify a variable in the field that is displayed at the right corner.
5. The value of the selected OID will get updated in the specified variable.
6. Click OK for the action to be added to the list.
Insert Row To a Table:

1. Select *Insert Row to a table* from the drop down list.
2. When simulating behavior of devices across a network, the agents in the network will be listed in the drop-down list. You will be able to select agent from the network.
3. Select a Table node and instance to insert row into the table.
4. Click *Add* to add a row.; You will see dialog coming up with details that contain:
   - Column name
   - Simulation Type
   - Value
5. You can make the desired changes to the row entries.
6. Click OK to insert the selected row when the script is executed.

Delete a Row From Table:

1. Select *delete row from the table* from the drop down list.
2. When simulating behavior of devices across a network, the agents in the network will be listed in the drop down list. You will be able to select agent from the network.
3. Specify the table OID and instance, that you want to delete from the table.
4. Click Ok to delete table

Delete All Rows From Table:

1. Select *delete all rows from the table* from the drop-down list.
2. When simulating behavior of devices across a network, the agents in the network will be listed in the drop-down list. You will be able to select agent from the network.
3. Specify the table OID and instance. All rows from the specified table node will be deleted.
4. Click OK.

Send Trap

There are three trap versions available: Sending a V1 trap, Sending a V2 trap and Sending a V3 trap.

Sending V1 Trap

1. Select trap type as snmp V1.
2. Manager name: Specify the manager name. by default it is given as localhost.
3. Manager port: The port at which the manager is running. by default it is 162.
4. The community string is set to public by default.
5. You can choose from the generic types available:
   - Cold start
   - Warm start
   - Link down
   - Link up
• Authentication failure
• EgpNeighborloss
• Enterprise specific : On selecting the enterprise specific type, you should also specify specific type for the trap.

6. Specify the enterprise OID for this trap if the Generic type is 
   Enterprise specific

Sending a V2c Trap
1. Select trap type as snmp V2c.
2. Manager name : Specify the manager name. by default it is given as localhost.
3. Manager port : The port at which the manager is running. by default it is 162.
4. The community string is set to public by default.
5. Specify the SNMP Trap OID and click OK.

Sending a V3 Trap
1. Select trap type as SNMP V3
2. Manager name : Specify the manager name. by default it is given as localhost.
3. Manager port : The port at which the manager is running. by default it is 162.
4. The community string is set to public by default.
5. You to provide a User name and Context Name for the snmp v3 trap. By default the values are provided User name has the value No Auth
   priv and context name has the value No Auth.
6. You can specify the trap OID.
7. Click ok.

Note: When you configure a Timer based SNMP V3 trap, the trap will be sent regardless of the manager's presence. You need the User name and the context name only to receive the traps. But when the SNMP v3 trap is configured as request based, the trap will not be sent unless the request is generated from the manager with the user name and context name.

Send Inform
1. Select send inform from the drop-down list.
2. Manager name : Specify the manager name. By default, it is given as localhost.
3. Manager port : The port at which the manager is running. By default, it is 162.
4. The community string is set to public by default.
5. You have to specify the SysupTime and Enterprise OID. Example: SysupTime can take value of 1101101 and Enterprise OID can have take a value like 1.3.6.1.2162.2.1.1
Generate Random Numbers

To generate random numbers, specify minimum value and a maximum value so that random numbers will be generated within the specified range.

Minimum value can be 100 and maximum value can 200; random numbers will be generated within this range.

Delay action :

You can introduce a delay in the action. Specify the value in milliseconds and the script execution will be delayed by the specified time.

Terminate operation

This action can be configured before a GET/GETNEXT/SET. It will terminate the ongoing operation and send the corresponding error message specified. The error message can be specified from the drop down box. You can choose any one of the following error types.
- tooBig
- noSuchObject
- noSuchInstance
- endOfMibView
- badvalue
- readOnly
- genErr
- noAccess

Modifying and Deleting Script

To modify a script
- Select the script to be modified, from the list displayed in the Configure Behavior dialog.
- Click Modify, to display the existing configuration. Make the necessary changes and click Finish.

To delete a script
1. Select the script to be deleted, from the list displayed in the Configure Behavior dialog.
2. Click Delete, to delete the script file. The conditions and actions configured for the script will also be deleted.

Modifying and Deleting Condition for Action

If you require to change an existing condition.
- Select the condition from the Condition to execute action list and click Modify to change the defined condition. The Configure Condition to be Checked dialog is brought up.
- Make the required changes and click OK for the changes to take effect.

You can delete a condition if it is no longer required.
- Select the condition from the Condition to execute action list.
- Click Delete to remove the condition from the list.
Modifying and Deleting Action to Be Executed

To modify an action

- Select the action from the Actions to be Performed list and click Modify to change the defined action. The Configure Actions to be Performed dialog is brought up.
- Make the required changes and click OK for the changes to take effect.

To delete an action

- Select the action from the Actions to be Performed list.
- Click Delete to delete the defined action.

Scripts in Examples Directory

1. DeleteAllRows.py

The script file uses the scriptAPI method deleteAllRows to delete all the entries in the ifTable. It takes the tableEntryOID as input.

Format:

```python
scriptinterface.deleteAllRows(String tableEntryOid)
```

2. DeleteRow.py

This script file uses the scriptAPI method deleteRow to delete the first entry in the ifTable. It takes the tableEntryOID and the instance of the entry to be deleted, as input.

Format:

```python
scriptinterface.deleteRow(String tableEntryOid, String instance)
```

where

- tableEntryOid is the Object Id of the table entry node
- instance is the instance of the entry to be deleted

3. DeleteRowWithValue.py

This script uses the scriptAPI method deleteRowWithValue to delete a specific entry from a table when the value of the specified column matches the specified value. It takes the tableEntryOID, the column name and value as input.

Format:

```python
scriptinterface.deleteRowWithValue(String tableEntryOid, String columnName String value)
```

where

- tableEntryOid is the Object Id of the table entry node
- columnName is the name of the column node in the table
- value is the value of the specified column
4. GetAllTableValues.py

This script file uses the scriptAPI method `getAllTableValues` to return the values of all the nodes for all the rows present in Table. It takes the `tableEntryOID` as input.

Format:

```java
scriptinterface.getAllTableValues(String tableEntryOid)
```

where

tableEntryOid is the Object Id of the table entry node

5. GetCurrentInstanceAndValue.py

This script file uses the scriptAPI method `getCurrentInstance()` and `getCurrentValue()` to return the instance and value of an incoming SET request. It should be configured for the condition "Before Processing Set" on a columnar node. This script gets the instance of the OID in the incoming SET request and its value and saves it in a text file named `instance.txt`

Format:

```java
scriptinterface.getCurrentInstance()
scriptinterface.getCurrentValue()
```

6. GetInstances.py

This script file uses the scriptAPI method `getInstances()` to get all the instances in a table. It takes the `tableOID` as input and will return the instances in a Vector.

Format:

```java
scriptinterface.getInstances(String tableOid)
```

where

tableOid is the Object Id of the table node

7. GetNextInstance.py

This script file uses the scriptAPI method `getNextInstance()` to get the instance next to the one specified. It takes the Table OID and the instance of a table as input. This will return the value as a String

Format:

```java
scriptinterface.getNextInstance(String tableOid, String instance)
```

where

tableOid is the Object Id of the table node
instance is the instance of a row present in the table
8. GetNodeValue.py

This script file uses the method `getNodeValue()` to get the current value of a scalar or a table node. It takes the OID of the MIB node and the instance as input. This method will return the value as a string.

Format:
```
scriptinterface.getNodeValue(String objectId, String instance)
```

where
- `objectId` is the object identifier (without the instance) of the MIB Node
- `instance` is the instance of the node whose value should be obtained, for scalar nodes this should be ".0"

9. GetRowCount.py

This script file uses the method `getRowCount()` to get the total number of rows in a table. It takes the Table OID as input and returns an integer value.

Format:
```
scriptinterface.getRowCount(String tableOid)
```

where
- `tableOid` is the Object Id of the table node

10. InsertRow.py

This script file uses the `insertRow()` method to insert an entry in the Table. It takes the Table OID, the Object Id of the columns in the table, the values of each column and its simulation type as input.

Format:
```
scriptinterface.insertRow(String tableOid, String[] objectId, String[] values, String[] simTypes)
```

where
- `tableOid` is the object id of the table node in which the entry has to be added
- `objectId` is an array of the object Id of the columns in the table
- `values` is an array of values for the columns in the table
- `simTypes` is an array of the simulation types for the columns

11. MaskNode.py

This script file uses the method `maskNode()` to mask a MIB node from the Manager.

Format:
```
scriptinterface.maskNode(String objectId, String instance, String maskValue)
```
12. UpdateColumnarValue.py and UpdateScalarValue.py

This script file uses the method `updateValue()` to update the value of a scalar node and columnar node. It takes the OID and instance of the MIB node for which the value should be updated, the value to be assigned and the simulation type.

Format

```
scriptinterface.updateValue(String objectId, String instance, String value, String simType)
```

where

- `objectId` is the object identifier (without the instance) of the mib node whose value has to be updated
- `instance` is the instance for which the value should be updated, for scalar nodes this should be ".0"
- `value` is the value to be assigned to the node
- `simType` is the type of simulation required

13. SendInform.py

This script file uses the method `sendInform()` to send an inform request with no extra varbinds. It takes the manager name, manager port, community, `sysUpTime` and SNMP Trap OID as input.

Format

```
sendInform(String mgrName, String mgrPort, String community, String sysUpTime, String snmpTrapOid)
```

where

- `mgrName` - the name of the manager to which the inform has to be sent
- `mgrPort` - the port at which the manager is listening for incoming inform requests
- `community` - the snmp community
- `sysUpTime` - the System UpTime
- `snmpTrapOid` - the value of the snmpTrapOid

14. SendInform_varbinds.py

This script file uses the scriptAPI method `sendInform()` to send an inform request with extra varbinds. The varbind for the inform is specified with the following information:

- `varbindOid` is OID of the inform varbind
- `varbindTypes` is the type of the varbind
- `varbindValues` is the value of the varbinds.

Format

```
sendInform(String mgrName, String mgrPort, String community, String sysUpTime, String snmpTrapOid, String[] varbindOid, String[] varbindTypes, String[] varbindValues)
```
15. SendV1Trap.py

This script file uses the scriptAPI method `sendV1Trap()` to send a v1 trap without varbinds. It takes the manager name, manager port, community, enterprise OID, generic type and specific type as input.

Format

```
sendV1Trap(String mgrName, int mgrPort, String community,
           String enterpriseOid, int genericType, int specificType)
```

where

- `mgrName` - the name of the manager to which the trap has to be sent
- `mgrPort` - the port at which the manager is listening for incoming traps
- `community` - the snmp community
- `enterpriseOid` - the Enterprise OID
- `genericType` - the Generic Type
- `specificType` - the Specific Type

16. SendV1Trap_varbinds.py

This script file uses the scriptAPI method `sendv1Trap()` to send a v1 trap with varbinds. It takes the manager name, manager port, community, enterprise OID, generic type, specific type, varbind OID, type and value as input.

Format

```
sendV1Trap(String mgrName, int mgrPort, String community,
           String enterpriseOid, int genericType, int specificType,
           String[] varbindOid, String[] varbindTypes, String[] varbindValues)
```

where

- `mgrName` - the name of the manager to which the trap has to be sent
- `mgrPort` - the port at which the manager is listening for incoming traps
- `community` - the snmp community
- `enterpriseOid` - the Enterprise OID
- `genericType` - the Generic Type
- `specificType` - the Specific Type
- `varbindOid` - oid of the varbind
- `varbindTypes` - the type of the varbind
- `varbindValues` - the value

17. SendV2Trap.py

This script file uses the `sendv2Trap()` method to send a v2 trap with no extra varbinds.

```
sendV2Trap(String mgrName, int mgrPort, String community, String trapOID)
```

18. SendV2Trap_varbinds.py

This script file uses the `sendv2Trap()` method to send a v2 trap with extra varbinds. The varbinds for the trap is specified with the following information:
varbindOid - oid of the varbind
varbindTypes - the type of the varbind
varbindValues - the value

```java
sendV2Trap(String mgrName, int mgrPort, String community,
           String trapOID, String[] varbindOid, String[]
           varbindTypes, String[] varbindValues)
```

19. SendV3Trap.py

This script file uses the `sendv3Trap()` method to send a v3 trap with no extra varbinds.

```java
sendV3Trap(String mgrName, int mgrPort, String community,
           String trapOID, String userName, String contextName)
```

20. SendV3Trap_varbinds.py

This script file uses the `sendV3Trap()` method to send a v3 trap with extra varbinds. The varbinds for the trap is specified with the following information:

- `varbindOid` is oid of the varbind
- `varbindTypes` is the type of the varbind
- `varbindValues` is the value

```java
sendV3Trap(String mgrName, int mgrPort, String community,
           String trapOID, String userName, String contextName,
           String[] varbindOid, String[] varbindTypes, String[] varbindValues)
```

21. TerminateOperation.py

This script file uses the scriptAPI method `terminateOperation()` for terminating the current operation i.e. a Get, GetNext or Set operation. The general format of the terminateOperation method is:

```java
scriptinterface.terminateOperation(int errorNumber)
```

where,

- `errorNumber` is the error to be thrown on terminating the operation.

24. GetTotalColumns.py

This script file invokes the method `getTotalColumns()` and returns the total number of columns in a table. It takes the table entry oid as input.

Format

```java
public int getTotalColumns(String tableEntryOid);
```

25. GetAllColumnByNames.py

This script file invokes the method `getAllColumnByNames()` and returns the name of all the columns in the table. It takes the table entry oid as input.

Format

```java
public String[] getAllColumnByNames(String tableEntryOid);
```
26. getAllColumnByOids.py

This script file uses the method `getAllColumnByOids()` and returns the OID value of all the columns in the table. It takes the table entry oid as input.

Format:
```
public String[] getAllColumnByOids(String tableEntryOid);
```

27. GetAllIndices.py

This script file uses the method `getAllIndices()` to return the string array having name of all the indices defined in a table. It takes the table entry oid as input.

Format:
```
public String[] getAllIndices(String tableEntryOid);
```

28. SendTrap.py

This script file uses the scriptAPI method `sendTrap()` to send traps configured in the Trap Configuration dialog. It takes the associated variable as input. The parameter 'associatedVariable' is the name with which the trap is configured in 'TrapConfiguration' dialog.

The manager details and trap details will be taken as such configured in the 'TrapConfiguration' dialog.

Format:
```
public void sendTrap(String associatedVariable);
```

29. SendTrapWithEnterpriseOid.py

The script file uses the method `sendTrap()` to send traps configured in the Trap Configuration dialog. It takes the associated variable as input. The parameter 'associatedVariable' is the name with which the trap is configured in 'TrapConfiguration' dialog. The 'enterpriseOid' is the one which is going to be sent in the trap.

The manager details and trap details will be taken as such configured in the 'TrapConfiguration' dialog. The enterpriseOid will be the one specified in the second argument irrespective of the one configured in 'TrapConfiguration' dialog.

Format:
```
public void sendTrap(String associatedVariable, String enterpriseOid);
```

30. GetConfigDirectory.py

This script file uses the scriptAPI method `getConfigDirectory()` to get the current working directory name. It returns the directory name as a string.

Format:
```
scriptinterface.getConfigDirectory();
```
31. SendV1TrapWhileAddingRow.py

This Script file will send the V1 Trap while adding the row in the tables. This Script file using the following Script API Methods:

- getRowCount()
- getCurrentInstance()
- sendV1Trap()

32. SendV1TrapWithVarbindsWhileAddingRow.py

This Script file will send the V1 Trap with Varbinds while adding the row in the tables. This Script file uses the following Script API Methods:

- getRowCount()
- getCurrentInstance()
- sendV1Trap()

33. StartSnmpAgent.py

This script file uses the scriptAPI method `startAgent()` to start an SNMP agent. This method is useful in the runtime to start any SNMP agent in the Network from any other agent. This file will work only for the **Network Simulator** and not for the SNMP Agent Simulator.

**Format:**

```
scriptinterface.startAgent(java.lang.String networkName, 
java.lang.String ipAddress, java.lang.String snmpPort,java.lang.String tl1Port);
```

34. StopSnmpAgent.py

This script file uses the scriptAPI method `stopAgent()` to stop an SNMP agent. This method is useful in the runtime to stop any SNMP agent in the Network from any other agent. This file will work only for the **Network Simulator** and not for the SNMP Agent Simulator.

**Format:**

```
scriptinterface.stopAgent(java.lang.String networkName, 
java.lang.String ipAddress, java.lang.String snmpPort,java.lang.String tl1Port) ;
```

**Sample Scripts Defined**

**Script with Condition for Execution Alone**

In this sample, the script `DeleteAllRows.py` will be executed, after a get request is made on the MIB node "ifDescr" in the ifTable of the RFC 1213 MIB, with instance as .1

1. Select the Add button in the Behavior Configuration dialog.
2. Choose the Request based icon that is displayed on the left of the wizard.
3. Select the MIB node "ifDescr" in the ifTable of the RFC 1213 MIB. Specify the instance as .1
4. The name of the selected MIB node will be displayed as the script filename. Browse and select the script file "DeleteAllRows.py" in the .examples directory.
5. Select the option "After processing Get" to execute the script.
6. Click Finish to end Behavior configuration and exit from the dialog.
Script with Condition for Execution, Condition for Action, Action to Be Executed

In this sample, the script file, ifOutErrors.py will be executed, when a set request is made on the node ifOutErrors. The condition for action is "if current value of ifOutErrors is greater than 10", the action to be executed is "send a v2trap". Hence, the script will send a trap, when a set is made on ifOutErrors with value greater than 10. The steps given below, defines the above Behavior:

1. Select the Add button in the Behavior Configuration dialog.
2. Choose the Request based icon that is displayed on the left of the wizard.
3. Select the MIB node "ifOutErrors" in the ifTable of the RFC 1213 MIB. Specify the instance as .1
4. The name of the selected MIB node will be displayed as the script filename "ifOutErrors.py".
5. Specify the condition to execute the script, by selecting the option "Before processing Set"
6. Click Next button, to specify the condition for action.
7. Click Add and select the method getCurrentValue, with operator as > and value as 10.
8. Click OK to add the condition to list. Now, you will find the condition getCurrentValue() >10, added to the list.
9. Click Next button, to specify the action to be executed.
10. Click Add and select the method Send Trap, specify the version as Snmp v2c, with other options like Manager, Port and Community as default.
11. Click OK to add the action to the list. Now, you will find the action Send a V2c trap to localhost and port 162, added to the list.
12. Click Finish to end Behavior configuration. You will find the script ifOutErrors.py, added to the list with the defined conditions and actions, in the main dialog. Click Close to exit from the dialog.

The script file ifOutErrors.py is given below:

```python
# The Actions
def actionfunc0():
    scriptinterface.sendV2Trap("localhost",162,"public",".1.3.6.1.2.1.11")

# The end of the Actions

# The Condition
if(scriptinterface.getCurrentValue() > "10"):
    actionfunc0()

# The end of Conditions
```

Script with Condition for Execution and Actions to Be Executed

In this sample, the script file, timer.py will be executed, when the agent is started. The following actions are configured to be executed at agent startup : Get the node value sysUpTime and store it in a variable., send v1Trap. The steps given below, defines the above Behavior:

1. Select the Add button in the Behavior Configuration dialog.
2. Choose the Timer based icon that is displayed on the left of the wizard.
3. Specify the script filename as `timer.py`.
4. Select the option `Execute Script while Agent Starts`.
5. Specify the `Time Interval` and `Number of Times` the script must be executed.
6. Click `Next`. The dialog to define the condition for action will be displayed. In this sample, we are not defining any condition for action. Hence click `Next`, to go the next dialog.
7. Click `Add` to add the action to the list. Select `GetNodeValue`, browse and select the OID as `sysUpTime` in System group of RFC 1213 MIB, specify the variable name as `agentUpTime`. The value of the node will be stored in this variable.
8. Click OK. You will find the action `GetValue of Node of oid .1.3.6.1.2.1.1.3 with Instance .0`, added to the list.
9. Click Add again, to add the second action. Select the method `Send Trap`, with other options like `version`, `Manager`, `Port` and `Community` as default.
10. Click OK. You will find the second action `Send a V1 trap to localhost and port 162`, added to the list.
11. Click Finish to end Behavior configuration. You will find the script `timer.py`, added to the list with the defined actions, the main dialog. Click Close to exit from the dialog.

The script file `timer.py` is given below:

```python
# The Actions

def actionfunc0():
    agentUpTime = scriptinterface.getNodeValue
        (".1.3.6.1.2.1.1.3" , ".0")
    scriptinterface.sendV1Trap("localhost",162,
        "public",".1.3.6.1.2.1.11",1,0)

# The end of the Actions

# The Condition
actionfunc0()

# The end of Conditions
```

### Script with Different Sets of Conditions for Action and Actions to Be Executed

In this sample, the script file, `ifAdminStatus.py` will be executed, when a get request is made on the node `ifAdminStatus` in `ifTable` of the RFC 1213 MIB. Two sets of conditions and actions are defined as given below:

- If node value of `ifAdminStatus` with instance ".1" is not equal to 1, Send a V1 trap to localhost at port 162
- If total number of rows in `ifTable` is less than 5, Insert a row in the `ifTable`.

The steps given below, defines the above Behavior:

1. Select the `Add` button in the Behavior Configuration dialog.
2. Choose the Request based icon that is displayed on the left of the wizard.
3. Select the MIB node "ifAdminStatus" in the `ifTable` of the RFC 1213 MIB. Specify the instance as ".1"
4. The name of the selected MIB node will be displayed as the script filename "ifAdminStatus.py".
5. Specify the condition to execute the script, by selecting the option "After processing Get"
6. Click Next button, to specify the condition for action.
7. Click Add and select the method *getNodeValue*, to specify the OID, browse and select the *ifTable*, with operator as ! = and value as 1.
8. Click OK to add the condition to list. Now, you will find the condition *

\[ \text{getNodeValue}(\text{"1.3.6.1.2.1.2.2.1.7"}, \text{"1"}) \neq 1 \]

added to the list.
9. Click Next button, to specify the action to be executed.
10. Click Add and select the method *Send Trap*, specify the version as *Snmp v1*, with other options like Manager, Port and Community as default.
11. Click OK to add the action to the list. Now, you will find the action *Send a V1 trap to localhost and port 162*, added to the list.
12. Click Finish. You will find the script *ifAdminStatus.py* added to the list, with the defined set of condition and actions, in the main dialog.
13. To add the second set of condition and action to the list, click Add button .
14. To add the second condition, click Add and select the method *getRowCount*, to specify the OID, browse and select the *ifTable*, with operator as < and value as 5.
15. Click OK to add the condition to list. Now, you will find the condition

\[ \text{getRowCount}(\text{"1.3.6.1.2.1.2.2.2"}) < 5 \]

*Insert a Row in the ifTable*, added to the list.
16. Click Next button, to specify the action to be executed.
17. Click Add and select the method *InsertRow*, to specify the OID, browse and select the *ifTable*. To specify the values of the new row, click Add . The new row with default values will be displayed. Modify the *ifIndex* value as '2' and click OK.
18. Click OK to add the action to the list. Now, you will find the action *Insert a Row in the ifTable*, added to the list.
19. Click Finish to end Behavior configuration. Click Close to exit from the dialog.

The script file *ifAdminStatus.py* is given below:

```python
# Imports required for the scripts
# to create an array the following lines should be in the
# The end of imports

# Row values of Tables

objectid1 = array(["1.3.6.1.2.1.2.1.1", "1.3.6.1.2.1.2.1.2", "1.3.6.1.2.1.2.1.3", "1.3.6.1.2.1.2.1.4", "1.3.6.1.2.1.2.1.5", "1.3.6.1.2.1.2.1.6", "1.3.6.1.2.1.2.1.7", "1.3.6.1.2.1.2.1.8", "1.3.6.1.2.1.2.1.9", "1.3.6.1.2.1.2.1.10", "1.3.6.1.2.1.2.1.11", "1.3.6.1.2.1.2.1.12", "1.3.6.1.2.1.2.1.13", "1.3.6.1.2.1.2.1.14", "1.3.6.1.2.1.2.1.15", "1.3.6.1.2.1.2.1.16", "1.3.6.1.2.1.2.1.17", "1.3.6.1.2.1.2.1.18", "1.3.6.1.2.1.2.1.19", "1.3.6.1.2.1.2.1.20", "1.3.6.1.2.1.2.1.21", "1.3.6.1.2.1.2.1.22"], String)
nodetypes1 = array(["CONST", "CONST", "CONST", "CONST", "CONST", "CONST", "CONST", "CONST", "CONST"], String)
```

"LINEAR","LINEAR","LINEAR","LINEAR","LINEAR","LINEAR","LINEAR","CONST","CONST"],[String)
values1 = array(["2","Test Interface","32","1","1", "ifPhysAddress","1","1","101010", "1|1|0","1|1|0","1|1|0","1|1|0", "1|1|0","1|1|0","1|1|0","1|1|0","1|1|0","1|1|0", "1",".1.3.6.1.2.1.2.2.1.22"],String)

# The end of the varbinds

# The Actions

def actionfunc0():
    scriptinterface.sendV1Trap("localhost",162,"public", ".1.3.6.1.2.1.11",0,0)

def actionfunc1():
    scriptinterface.insertRow(".1.3.6.1.2.1.2.2", objectid1,values1,nodetypes1)
    # The end of the Actions

# The Condition
if(scriptinterface.getNodeValue(".1.3.6.1.2.1.2.2.1.7" , ".1") != "1"):
    actionfunc0()

if(scriptinterface.getRowCount(".1.3.6.1.2.1.2.2") < 5):
    actionfunc1()
    # The end of Conditions
Configuring Error Conditions

Overview

The SNMP Agent Simulator allows you to configure an SNMP agent to send invalid responses. The intention is to test the SNMP manager's robustness in the face of receiving bad packets. The feature *Error Simulation* allows you to simulate error conditions and scenarios by which you can send errors for the requests received. The errors can be thrown for all the three type of requests: GET, GetNext, and Set.

All the error conditions will be reflected only at agent run time and will not be saved in the simulator. You can define the type of error that an OID should respond with, when queried.

Configuring Error Conditions

To simulate errors, do the following:

1. Choose **Configure -> Error Simulation** in the menu bar. The Simulate error dialog appears with the MIB tree loaded.
2. Select the OID from the MIB tree. In case of tables, the instance can be selected from "Select the Instance" dialog, which pops up when the **Add** button is clicked.
3. Choose the error type that you want to specify from the options. The error types are
   - tooBig
   - noSuchObject
   - noSuchInstance
   - endOfMibView
   - badvalue
   - readOnly
   - genErr
   - noAccess

4. You have to specify the condition on which you want to simulate this error. You can choose one or all the conditions. The condition are as follows:
   - Get Request
   - Get Next Request
   - Set Request

5. Click **Add** to add the OID, error type, and condition to the list.
6. To modify any of the parameters in the list, select an entry from the list, modify the parameters, and click **Modify**. The entry will be modified.
7. To delete an entry, select the entry form the list and click **Delete**.
Note:
1. While doing `GETNEXT / SNMPWALK` the nodes for which the error is configured will be skipped.
2. You can specify only one type of error at an instance.

Examples

Assume that the Simulated agent with RFC1213 MIB is started. You can configure the following sample error conditions in the SNMP Agent Simulator and test the same from the MIB Browser.

1. In the error simulation dialog, select the node "SysDescr" under system group, choose the error type as "readOnly", specify the error condition as "Set Request" and click Add. When a SET request is made from the Mib Browser on the SysDescr node, the agent will respond with "readOnly" error.

2. In the error simulation dialog, select the Table node "ifDescr" in the ifTable, choose error type as "noSuchInstance", specify the error condition as "Get Request", and click Add. Specify an instance from the list of instances. When a GET request is made from the Mib Browser for the specified instance, the agent will respond with "noSuchInstance" error.
Configuring Agent Settings

- Overview
- Agent Settings
- Port Number
- Engine ID
- Version
- Debug Level
- Encoding
- Manage Using RMI
- Log Inform Response
- Reverse Order of Script Execution
- Read Community
- Write Community

Overview

Before starting the simulated agent, you can configure the run time Agent Settings such as port, version, community, etc. If the settings are not configured, the simulated agent will start with the default settings. To configure the agent settings, Select **Settings -> Agent Settings** option in the menu bar or click on the Agent Settings icon in the toolbar.

Agent Settings

Port

The simulated SNMP agent, starts at default port number 8001. You can specify any alternative port. If you are unable to start the agent at port 161, refer to the topic, **TroubleShooting - General**.

Engine ID

Engine ID is used in SNMPv3. It is the unique identifier of the SNMP Engine in the domain. You can specify Engine ID or by default it will take the combination of the port and IPAddress.

Version

The SNMP Version can be specified as either V1, V2c, or V3. By default, the SNMP version is V2c.

Debug Level

The following Logging options are available in SNMP Agent Simulator. The logs will be generated in the agent.log file inside the `<Simulator_Home>/logs` directory, based on the selected logging level.

- **Disabled (0)**: This level indicates that the debugging mode is disabled. Hence, log messages do not get stored in the text file.
• **Fatal (1)**: This level indicates that the application has encountered a severe error. These errors need to be attended with care. When this level is chosen, critical messages such as details on Agent crash, Authentication errors, Agent already open in another session, etc., get stored in the log file.

• **Error (2)**: This level indicates that an error has occurred and the application cannot handle the request as designed. This may be automatically corrected during execution or the application should be capable of handling those situations when a similar request comes again. Selecting this level stores all the Error and Exception messages in the log file along with the fatal messages.

• **Warning (3)**: This level indicates that the application has met with an expected error. All Warning messages get stored in the log file when debug level is WARN.

• **Info (4)**: This level does not signify the occurrence of any error but logs all important events. Any information that might be required for reference gets stored in the log file under this info level.

• **Debug (5)**: This level indicates that the application moves into the debug mode. It will be used for generating debugging information and should not be used in general cases.

• **Trace (6)**: This level indicates that the application logs every activity from entering a method to exiting it. Hence, selecting this level may affect performance. The Trace level should be used only to solve any complex problem.

**Encoding**

The agent simulator supports English, Japanese and Chinese encoding types. The default encoding type is "English". The Encoding option enables you to select the encoding type to be supported by the simulator. When the agent simulator is started, any string will be encoded as per the encoding format selected.

**Manage using RMI**

The SNMP Agent Simulator provides option to manage the SNMP agent from the remote, using Remote Method Invocation (RMI). Selecting the option 'Manage using RMI' will register the simulator with the RMI registry.

Refer to the topic Managing Agents via RMI, to learn more.

**Log Inform Response**

When a Inform is sent from the SNMP Agent, an SNMP Manager that receives the Inform request acknowledges the message with an SNMP response PDU. Selecting this option will capture all the inform response received from the manager, in a log file. The log file informresponse will be created under <Simulator_Home>/logs directory.

**Reverse Order Of Script Execution**

A script may be configured at the table instance level, table column level and table entry level. By default, the instance level script will be executed first, the column level script will be executed next and the table entry level script will be executed last.

If the option Reverse Order of Script execution is selected, the script will be executed in the reverse order. (i.e) the script configured at the table entry level will be executed first, the column level script will be executed next, and then the instance level script will be executed.
Read Community

The community to be used while processing the GET request. The default Read Community is public. You can define more than one community name. Click Add to add your own read community string.

Write Community

The community to be used while processing the SET request. The default Write Community is public. You can define more than one community name. Click Add to add your own write community string.
Customizing SNMP PDU in SNMP Agent Simulator

- Overview
- Customizing the request SNMP PDU
- Customizing the response SNMP PDU

Overview

When a request is made by the SNMP Manager, the snmp agent simulator creates the response SNMP PDU in a memory buffer and writes the PDU in the socket layer, where the IP and UDP headers get added before sending the packet to the SNMP Manager. The "SNMP PDU Scrambler" supports customizing the request/response SNMP PDU by exposing it to the user for customization.

The PDU Scrambler Interface enables you to,

- Access to the request PDU for customization, before sending it to the simulator for further processing.
- Update the request PDU with error status and error index values, if the incoming PDU is valid.
- Update the request PDU with error status and error index values, if the incoming PDU is valid and terminate the operation. In this case the simulator will skip the normal SNMP processing and will send the updated PDU as response PDU.
- Access the SNMPScriptInterface object, for working with behavior API methods before processing the request/response PDU. The api implementation here is equivalent to configuring jython scripts to be executed before/after processing request condition.
- Access to the response PDU for customization, before sending it to the socket layer.
- Update the response PDU with proper error status and error index values, if invalid and send a error response to the manager.

For example, a request SNMP PDU can be customized to validate that it contains multiple varbinds before the agent performs SNMP operation in OIDs. The response SNMP PDU can be customized, for example, to send garbled/invalid responses to test the SNMP managers robustness in the event of receiving bad packets.

The level of customization varies depending on your requirement. For example, you can change the initial ASN.1 length value to a smaller or bigger value, change the length of a varbind OID to be very large, to see if the SNMP manager can read the full packet and give an error or to test other problem involved in dealing with un-initialized memory.

Customizing the request SNMP PDU

The method processRequestPDU( SnmpPDU reqPDU ) in <Simulator_Home>/examples/pduscrambler/SimulatorPDUscrambler.java is used to customize the request PDU. This method will be called with the SnmpPDU object when a SNMP GET, GET NEXT, SET request is made, before sending the packet to the simulator for processing the request. You can add your code inside this method to do the required PDU processing. After the processing, this method returns the modified or scrambled PDU to the simulator.
The prototype of this method is given below:

```java
import com.adventnet.snmp.snmp2.SnmpPDU;
import com.adventnet.simulator.scripting.SimScrIntf;

public class SimulatorPduScrambler implements SimPDUScramInterface {
    /*Describe processRequestPDU method here.*/

    public SnmpPDU processRequestPDU(SnmpPDU reqPdu) throws com.adventnet.snmp.snmp2.SnmpException {
        /*
        * add User Code here.
        **/
        /**
        * Here user can use the scriptIntf obj ref for working with
        behavior API
        * The api implementation here is equivalent to configuring jython
        scripts
        * to be executed before processing request condition
        */
        // doProcessWithScriptAPI();
        /**
        * if you need to terminate the operation you have to throw the
        error. In this case * simulator will skip the normal SNMP
        processing and send updated PDU as response * * PDU.
        */
        updateErrStatAndTerminate(reqPdu);
        return reqPdu;
    } // End of the method processRequestPDU()
```

After adding the desired code, compile the `SimulatorPDUScrambler.java` file. The compiled class file must be placed in `<Simulator_Home>/conf` directory.

Refer to SNMP PDU Javadocs for customizing the response PDU. Refer to Script Interface Javadocs to access the script API methods.

### Customizing the response SNMP PDU

The method `processResponsePDU(SnmpPDU resPDU)` in `<Simulator_Home>/examples/pduscrambler/SimulatorPDUScrambler.java` is used to customize the response PDU. This method will be called with the SnmpPDU object when a SNMP GET, GET NEXT, SET request is made, before sending the packet to the socket layer. You can add your code inside this method to do the required PDU processing. After the processing, this method returns the modified or scrambled PDU to the simulator.

The prototype of this method is given below:

```java
import com.adventnet.snmp.snmp2.SnmpPDU;
public class SimulatorPduScrambler implements SimPDUScramInterface {
    /*Describe processResponsePDU method here.*/
```
public SnmpPDU processResponsePDU(SnmpPDU respPdu) {

    /**
     * add User Code here.
     */

    /**
     * Here user can use the scriptIntf obj ref for working with
     * behavior API
     * The api implementation here is equivalent to configuring jython
     * scripts
     * to be executed after processing request condition
     */
    // doProcessWithScriptAPI();

    /* Here user can set the ErrorStatus and ErrorIndex vlaue in the
    PDU, for example */
    // setErrorIndexAndErrorStatus((byte)1,byte(1),respPdu);

    return respPdu;
} // End of the method processResponsePDU()

After adding the desired code, compile the SimulatorPDUScrambler.java file. The compiled class file must be placed in <Simulator_Home>/conf directory.

Refer to SNMP PDU Javadocs for customizing the response PDU.
Refer to SNMPScriptInterface.html to access the script API methods.
Re-simulating Values

- Overview
- Enabling the Option
- Example

Overview

The option **Operations -> Re-simulate Values** is used to simulate the value of the newly added groups in the MIB without modifying the existing configuration. Values will be re-simulated only for newly added Scalar/Table groups and not for new nodes added inside the existing groups in the MIB.

To re-simulate values, the existing configuration file must be loaded in the SNMP Agent Simulator only after making the necessary modifications.

Enabling the Option

To re-simulate values, follow the steps given below:

- Load the existing configuration file of the modified RFC1213 MIB in SNMP Agent Simulator.
- Choose **Operations -> Resimulate Values**.
- The values of the newly added scalar/table groups will be simulated and added to the existing configuration file.

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The option to Re-simulate values is not supported for SNMP devices in <strong>Network Designer</strong>.</td>
</tr>
<tr>
<td>2. The new scalar/table nodes must be added only at the end of the MIB file to avoid OID conflict with already existing groups.</td>
</tr>
</tbody>
</table>

Example

You have a simulated configuration of the RFC 1213-MIB, wherein you have added large number of data to the configuration. The configuration is saved as **sample_main.xml** in **./snmpagents/sample** directory.

Now, you have defined a new table group say "xxxTable" to this MIB, available in **./snmpagents/sample** directory and you want to simulate this table alone and add it to the existing configuration file.

You can do this by loading the sample_main.xml file in the SNMP Agent Simulator and selecting the option **Operations -> Resimulate Values**. The xxxTable values will be added to the existing configuration.
Saving SNMP Agent Configuration

When you have simulated values for MIB nodes and completed configuring the agent, you can save the agent configuration for later use. To save the agent configuration,

- Choose the menu option File -> Save option.
- The database name will be `simulatordb` and is non-editable. You can specify the table name and the configuration file name. The configuration will be stored as `<filename>_main.xml` along with all the other configuration details in the specified table name.
Starting and Stopping the Agent

- **Overview**
- **Starting the Agent**
- **Stopping the Agent**

**Overview**

In the topic, *Simulating Default Values*, you have learnt that the agent is simulated as soon as you load the MIB file(s) and simulate default data for the loaded MIB. Any variation on this basic simulation is created in the SNMP Agent Simulator by *Configuring Values*.

When you have finished configuring the agent with the required values, behavior scripts and traps, you can test the simulated agent for the required behavior, after starting the agent. The agent can be started at the specified port from the SNMP Agent Simulator or from Command line.

**Starting the Agent**

**a. From UI**

To start the simulated agent from the UI, do the following:

1. Click on the start icon or choose **Operations --&gt; Start Agent** from the menu bar.
2. You will get a message in the text area "Agent started at port 8001". The port number might vary based on the port you choose to run the simulator.

**Note:** For evaluation copy, the agent will respond only for the first two groups queried.

**b. From Command Line**

To start the simulated agent from command Line, refer to the topic *Starting the SNMP Agent from Command Line*.

**c. By executing Jython scripts**

You can start the SNMP Agent by executing the SNMP script interface method **startAgent()** inside the Jython script file. The javadocs of the methods defined in the script interface can be accessed from `<SimulatorHome>/examples/javadocs` directory.

**Stopping the Agent**

**a. From UI**

- Select **Operations -&gt; Stop Agent** from the menu bar or click on the stop icon of the toolbar.
- The message *Agent Stopped* will be printed in the text area.

**b. By executing Jython scripts**

You can stop the SNMP Agent by executing the SNMP script interface method **stopAgent()** inside the Jython script file. The javadocs of the methods defined in the script interface can be accessed from `<SimulatorHome>/examples/javadocs` directory.
c. From Command Line

You can stop SNMP agents at your will from command line, by executing the `StopSnmpAgent.bat/sh` file in `<SimulatorHome>/bin/cmdline` directory. For example, to stop the SNMP agent running at port 8001 and ip address 190.191.1.2, specify the following command.

```
StopSnmpAgent -h 190.191.1.2 -p 8001 -c public
```

Refer to the topic [Stopping the Remote Agent](#) for more information.
Testing the Agent using MIB Browser

The simulated SNMP agent can be tested using the MIB Browser bundled with the product. Before testing, the agent must be started at the specified port. To test the simulated SNMP Agent, do the following:

- Start the MIB Browser from the Launcher or by executing the SnmpMibBrowser.bat/ SnmpMibBrowser.sh from <Simulator home>/bin directory.
- Specify the Port number, Host name and Community of the agent in the Mib Browser main screen dialog.
- Load the MIB which is configured in the simulated agent. For example, if you have simulated the values of agent-sample-mib, load agent-sample-mib.txt using the File -> Load MIB option.
- Expand the AGENT-SAMPLE-MIB tree. Select the enterprises node and do a query by selecting Operations -> GET or clicking on the GET icon.
- The simulated agent running at port 8001 (default port) will respond with the simulated values.

Refer to section Using Test and Editor Tools -> MIB Browser, to learn more about the SNMP operations that can be performed using the MIB Browser.
Porting SNMP Agent Configuration

- Overview
- Packaging SNMP Agents
- Installing SNMP Agents

Overview

The SNMP Agent Simulator allows you to package the SNMP agent you have created with the desired configurations and install it in any other system having Simulation Toolkit. For example, you have created the SNMP agent in Linux or Solaris system with all the desired configuration. This configuration can be packaged and sent across for demo or trade shows or kept as a backup.

Packaging SNMP Agents

To package your network,

- Choose File --> Package SNMP Configuration from the menu. The Package SNMP Agent dialog opens up.
- Click Add and select the SNMP agent main file. (i.e) the configuration will be stored as <filename>_main.xml.
- Click Browse to select the location of where this zip file (packaged SNMP agent) should be stored and provide a name for the zip file.
- Click OK to package the SNMP agent.

Installing SNMP Agents

You can install the SNMP agent that you packaged, in another system having this product installed.

- Select File --> Package SNMP Configuration menu item from the menu bar. This brings the Install SNMP Agents dialog.
- Browse and select the package name. (i.e) the packaged zip file and click Open.
- Click OK to install the packaged zip file.
Stopping the Remote Agent

- Overview
- Stopping the Remote Agent

Overview

The Simulated SNMP Agent which is started from the SNMP Agent Simulator UI or from Command line can be stopped from a remote system. The command line utility StopSnmpAgent.bat/sh file is used to achieve this. To stop the remote agent, you must know the remote agent host name, port number and write community.

Stopping the Remote Agent

To stop the remote SNMP Agent, do the following:

- Run the <Simulator Home>/bin/cmdline/StopSnmpAgent.bat/sh file from command Line with the following options:
  - Host Name: The name of the host where the remote agent is running
  - Port Number: The port number of the remote agent.
  - Community: The write community of the remote agent

Usage:

```
StopSnmpAgent -h [hostName] -p [port] -c [write community]
```
Simulating a V3 Agent

- Overview
- Security Levels in SNMPv3
- Default Users of SNMPv3 Agent
- Starting a v3 Agent
- Testing the SNMPv3 Agent with noAuth and Auth User
- Privacy Settings for Priv User
- Testing the SNMPv3 Agent with Priv Users
- Adding New User to the USM Table
- Adding New User to the VACM Tables

Overview

The version 3 of SNMP (SNMP V3) is used to provide a secured environment in managing the systems and networks. To simulate an SNMPv3 agent using Agent Simulator is a simple task. On just specifying the version as V3 while starting the agent, a v3 agent gets initialized. This section deals with the Security levels supported by the v3 agent, the steps involved in simulating and testing the v3 agent, and adding new users to the USM and VACM Tables.

Security Levels in SNMPv3

The SNMPv3 Agent supports the following set of security levels as defined in the USM MIB (RFC 2574):

- **noAuthnoPriv** - Communication without authentication and privacy.
- **authNoPriv** - Communication with authentication and without privacy. The protocols used for Authentication are MD5 and SHA (Secure Hash Algorithm).
- **authPriv** - Communication with authentication and privacy. The protocols used for Authentication are MD5 and SHA ; and for Privacy, DES (Data Encryption Standard) protocol is used. For Privacy Support, you have to install some third-party privacy packages. Details about installation is dealt with in the topic "Privacy Support".

The SNMPv3 agent also supports the VACM MIB as a default Access Control Model. It consists of four tables. The context name, group name and read/write access for a user are configured in these tables.

Default Users of SNMPv3 Agent

By default, the SNMPv3 Agent provides support for three level of users, namely:

- **noAuthUser** - Users with security level noAuthnoPriv and context name as noAuth.
- **authUser** - Users with security level authNoPriv and context name as auth.
- **privUser** - Users with security level authPriv and context name as priv.

The details about the users get stored in the XML files under the `<Simulator Home/conf>` directory.
Starting a SNMPv3 Agent

The steps involved in simulating a v3 agent using Agent Simulator are are given below:
1. Start the SNMP Agent Simulator.
2. Load the MIBs for which the agent should be simulated. For example, load AGENT-SAMPLE-MIB.
3. Click on the Settings -> Agent Settings option in the menu bar.
4. In the dialog that is displayed change the version of the agent to V3 and click on the 'OK' button.
5. Click on the menu item Operations -> Start Agent to start the agent.
6. The SNMPv3 agent gets started with the default user configurations present in XML files under the <Simulator Home/conf/snmpv3> directory.

Testing the SNMPv3 Agent with noAuth and Auth Users

Assuming that the simulated SNMPv3 agent is started, let us see how it can be tested using the default entries available in the USM and VACM Tables.

User Name "noAuthUser", Security level "noAuth,noPriv"

The default entry for noAuthUser in USM Table will be as follows. The default Context Name of this user as defined in the VACM Context Table is noAuth. The subtree OID for which read/write access is provided for this user in the VACM View Table is .1.3.6

<table>
<thead>
<tr>
<th>User Name</th>
<th>Security Level</th>
<th>Auth Protocol</th>
<th>Priv Protocol</th>
<th>Auth Password</th>
<th>Priv Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>noAuthUser</td>
<td>noAuth,noPriv</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

To test the agent,
1. Make sure the SNMPv3 Agent is started
2. Start the MibBrowser application from <Simulator_Home>/bin directory.
3. Load the MIB which is implemented in the Agent. For example, load AGENT-SAMPLE-MIB.
4. Click MIB Browser Settings icon in the toolbar. The MibBrowser Settings dialog opens.
5. Choose the SNMP version as v3 in the "General" tab.
6. Select "v3Settings" tab to add the noAuthUser entry in the table. The following details need to be filled:
   - Target Host : localhost (by default)
   - Target Port : 8001
   - User Name : noAuthUser
   - Security Level : noAuth,noPriv
7. Click Add Entry. The entry gets listed in v3 Settings table
8. Select the entry and click OK to close the MIB Browser Settings wizard.
9. Move on to the MIB Browser Main UI.
10. Give noAuth in the Context Name field of MibBrowser.
12. You find the agent responding to the request.
User Name "authUser", Security level "AuthnoPriv" with MD5 Auth protocol

The default entry for authUser in USM Table will be as follows. The default Context Name of this user as defined in the VACM Context Table is auth. The sub-tree OID for which read/write access is provided for this user in the VACM View Table is .1.3.6 included and 1.3.6.1.4.1.2162.4.1.1 excluded.

<table>
<thead>
<tr>
<th>User Name</th>
<th>Security Level</th>
<th>Auth Protocol</th>
<th>Priv Protocol</th>
<th>Auth Password</th>
<th>Priv Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>authUser</td>
<td>Auth,noPriv</td>
<td>MD5</td>
<td>-</td>
<td>authUser</td>
<td>-</td>
</tr>
</tbody>
</table>

To test the agent,

1. Make sure the SNMPv3 Agent is started.
2. Start the MibBrowser application from <Simulator_Home>/bin directory.
3. Load the MIB which is implemented in the Agent. For example, load AGENT-SAMPLE-MIB.
   Click MIB Browser Settings icon in the toolbar. The MibBrowser Settings dialog opens.
4. Choose the SNMP version as v3 in the "General" tab. Select "v3Settings" tab to add the noAuthUser entry in the table. The following details need to be filled:
   - Target Host : agent host name
   - Target Port : 8001.
   - User Name : authUser
   - Security Level : Auth,noPriv
   - Auth Protocol : MD5
   - Auth Password : authUser
5. Click Add Entry. The entry gets listed in v3 Settings table
6. Select the entry and click OK to close the MIB Browser Settings wizard.
7. Move on to the MIB Browser Main UI.

Privacy Settings for Priv User

For privacy support, any one of the following Encryption packages can be used:

- JCE
- Cryptix

**Note:** JDK 1.4 bundles JCE privacy packages by default. The Simulation toolkit package bundles jre i.4. Hence, if JDK 1.4 is used, then JCE privacy jars are not required to be in the classpath.
If you are using lower JDK versions then you have to include the corresponding privacy packages in the classpath.

**To make use of JCE classes**

- Download JCE classes 1.2 or 1.2.1 from the following URL: http://java.sun.com/products/jce/
  - In case JCE 1.2 classes are downloaded, you get the following jar : jce12-rc1-dom.jar
In case JCE 1.2.1 classes are downloaded, you get the following four jars:
- jce1_2_1.jar
- local_policy.jar
- sunjce_provider.jar
- US_export_policy.jar

- Make sure the jars are placed under <Simulator_Home> directory.
- Also make sure the jars are included in the setenv.bat/sh file CLASSPATH. The setenv.bat file is available in <Simulator_Home>/bin directory. Please note that the jars are required to be in the CLASSPATH settings of run.bat/sh file, that are used for running the Simulated Agent.
- Edit the java.security file present in the jre/lib/security folder under the JDK installed in your machine. And add the following piece of line below:
  - security.provider.1=sun.security.provider.Sun
  - security.provider.2=com.sun.crypto.provider.SunJCE

- Save the java.security file.
- The USMUtils.class required for encrypting v3 requests and responses is available in AdventNetSnmp.jar in <Simulator_Home>/jars directory.
- Now, the v3 Agent is ready for supporting Privacy.
- If you run MibBrowser from the batch/script file then you have to add JCE jar in the Classpath. Similarly if you start MibBrowser from Launcher then you have to set JCE jar in the class path variable of MibBrowser. For this, select MibBrowser icon in the launcher, right-click and select settings, and add JCE jar file in the classpath.

To make use of Cryptix classes

Changes to be made in the Agent side:

- Download Cryptix classes 3.1 or 3.2 from the following URL: http://www.cryptix.org/
- Make sure the jars are included in the setenv.bat file CLASSPATH (available in <Simulator_Home>/bin directory) in the beginning. This is applicable to all JDK versions. Please note that the jars are required to be in the CLASSPATH settings of run.bat/sh file, that are used for running the Simulated Agent.
- The USMUtils.class required for encrypting v3 requests and responses is available in AdventNetSnmp.jar (<Simulator_Home>/jars directory).
- Edit the java.security file present in the jre/lib/security folder under the JDK installed in your machine. And add the following piece of line below:
  - security.provider.1=sun.security.provider.Sun
  - security.provider.3=cryptix.provider.Cryptix
- Now, the v3 Agent is ready for supporting Privacy.

Changes to be made in the Manager Side:

- If you run MibBrowser from the batch/script file then you have to add Cryptix zip file in the Classpath. Similarly if you start MibBrowser from Launcher then you have to set Cryptix zip in the class path variable of MibBrowser. For this, select MibBrowser icon in the launcher, do right click, select settings and add Cryptix zip file in the classpath.

Export Restrictions

Encryption packages are bound by Export restrictions.

- If JCE 1.2 or its implementations are used in developing application and applets, they cannot be used outside US and Canada.
- JCE 1.2.1 does not have any export restrictions and it can be used in applications, which can be distributed throughout the world.
- The latest JDK version (JDK 1.4) comes integrated with the JCE 1.2.1.
- Cryptix package does not have any such export restrictions.
Testing the v3 Agent with Priv Users

Assuming that the privacy settings are made and the simulated SNMPv3 agent is started, let us see how it can be tested using the default entries available in the USM and VACM Tables.

User Name "privUser", Security Level "AuthPriv" with MD5 Priv Protocol

<table>
<thead>
<tr>
<th>User Name</th>
<th>Security Level</th>
<th>Auth Protocol</th>
<th>Priv Protocol</th>
<th>Auth Password</th>
<th>Priv Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>privUser</td>
<td>Auth,Priv</td>
<td>MD5</td>
<td>-</td>
<td>authUser</td>
<td>privUser</td>
</tr>
</tbody>
</table>

Note: For "privUserMD5" and "privUserSHA" you need either cryptix package for jdk1.x or JCE for jdk1.2.x to run the MibBrowser. Please refer the topic "Privacy Settings for Priv User" for more information.

The default entry for privUser in USM Table will be as follows. The default Context Name of this user as defined in the VACM Context Table is priv. The sub-tree OID for which read/write access is provided for this user in the VACM View Table is .1.3.6 included and 1.3.6.1.4.1.2162.4.1.6 excluded.

To test the agent, do the following:

- Make sure the SNMPv3 Agent is started
- Start the MibBrowser application from <Simulator_Home>/bin directory.
- Load the MIB which is implemented in the Agent. For example, load AGENT-SAMPLE-MIB.
- Click MIB Browser Settings icon in the toolbar. The MibBrowser Settings dialog opens.
- Choose the SNMP version as v3 in the "General" tab.
- Select "v3Settings" tab to add the privUser entry in the table. The following details need to be filled:
  - Target Host: agent host name
  - Target Port: 8001 (agent port number)
  - User Name: privUser
  - Security Level: Auth,Priv
  - Auth Protocol - MD5
  - Auth Password - authUser
  - Priv Password - privUser
- Click Add Entry. The entry gets listed in v3 Settings table.
- Select the entry and click OK to close the MIB Browser Settings wizard.
- Move on to the MIB Browser Main UI.
- Give priv in the Context Name field of MibBrowser. Select ".iso.org.internet.private.enterprise.AdventNet" of agent-sample-mib.txt and do a get next. You find the agent responding to the request.

Adding New User to the USM Table

To add new users to the usmUserTable of the SNMP-USER-BASED-SM-MIB, AdventNet provides a tool called v3Config Tool. To invoke the tool, select SNMPv3config.bat/sh file from <Simulator_Home>/bin directory. By default, the USMUser tab will be selected and the default entries displayed as shown in the screen-shot below:
New user entries must be added to the USM and VACM tables before starting and accessing the v3 agent with the required new user names.

**Note:** To access the v3 agent, the new users must also be configured in the VACM Tables. Please refer the section "Adding new users to the VACM Table" for more details.

To add a new user to the USM Table,

1. Choose **USMUser Tab** and click on **Add** from menu bar. Now, the **Properties dialog** pops up. The following are the information in this dialog.

   - Fill in the "UserName" (can be any name) and "Security Level" in the space provided using the options provided in the combo box as shown in the image.
   - In case, the "Security Level" given is AuthnoPriv, then the Auth Password has to be mentioned. If it is AuthPriv then both Auth Password and Priv Password have to be stated. Here again, the password can be of the user's choice.
   - Auth Protocol can be MD5 or SHA though by default it is MD5.
   - Priv Protocol should be CBC-DES.

**Sample entry would be :**

- UserName : NewUser
- Security Level : AuthnoPriv
- AuthProtocol : MD5
- AuthPassword : newuser
2. On filling the entries, click on **OK** button. This adds the new user entry.

**Adding New User to the VACM Tables**

To provide view based access control, the new user entry has to be added to the VACM Tables of the SNMP-VIEW-BASED-ACM-MIB. The **V3Config Tool** can be used to configure the VACM entries. The steps involved in adding a new user to the VACM Table using Table view is given below:

- Adding new user to the VACM Context Table
- Adding new user to the VACM GroupTable
- Adding new user to the VACM Access Table
- Adding new user to the VACM View Tree Family Table

**Adding New User to the VACM Context Table**

The VACM Context table lists the context names defined for the users. To view the default context names select the **VacmContext tab**. The default context names will be listed as shown in the image below:
1. To add a new context name to the VACM context table, select the **Add** button.
2. Enter the context name in the space provided. It can be of any name. The context name is used as a reference when accessing the v3 agent.
3. Click **OK** to add the new context name to the table.

The next step is to configure the VACM Group Table.

**Adding New User to the Vacm Group Table**

The VACM group Table has a set of security to group mappings. If the received context name is valid then the group name is obtained from this table by giving user (security) name and security model as an input. Currently the application supports only the USM model and by default it is specified.

- Model will always have to be USM.
- Security Name can be either ‘authUser’ or ‘noAuthUser’ or ‘privUser’.
- Group Name can be of any user defined name.

To view the default group names or add new groups in the table,

1. Select the **VacmGroup** Tab. The default groups provided for the default users configured in the USM table are listed as shown in the image below:

![Vacm Group Table](image)

2. To add a new group to this table, click **Add** and enter the following details in the space provided:
   - **Security Model**: The Security Model supported by the snmpv3 agent. Currently, only USM is supported.
   - **Security Name**: The name of the new user as configured in the USM Table.
   - **Vacm Group Name**: The group name to which the user belongs. You can specify any name as group name.

3. Click **OK** to add the entry to the table.

The next step is to configure the VACM Access Table.
Adding New User to the Vacm Access Table

The VACM Access table lists the access rights and restrictions of the various groups. By giving group name, context name, security model, and security level of the v3 user, the read, write, notify we can provide view name based on the received request type. It is configurable through SNMP.

The group’s access rights are given for,

- read - for retrieval operations - get, getnext and getbulk.
- write - for write operations - set
- notify - for notification operations - trap

To view the default views or add new views in the table,

1. Select the VacmAccess Tab. The default views provided for the default users configured in the USM table will be listed.
2. To add a new entry to this table, click Add and enter the following information in the space provided:
   - **Vacm Group Name**: The group to which the user belongs as defined in the VACM group table.
   - **Context Prefix**: The context name of the user as defined in the VACM context table.
   - **Security Model**: The Security Model supported by the snmpv3 agent - USM
   - **Security Level**: The Security Level of the user
   - **Context Match**: The Context Match "Exact"
   - **Read View Name**: The read view name provided for the v3 user.
   - **Write View Name**: The write view name provided for the v3 user.
   - **Notify View Name**: The notify view name provided for the v3 user.
3. Click OK, to add the entry to the table.

The next step is to configure the VACM View Tree Family Table.

Adding New User to the Vacm View Tree Family Table

This table assigns the Sub tree OID that can be accessed by a particular view. To view the default views or add new views in the table,

1. Select the VacmViewTree Tab displayed on the left side on the tool. The default views and sub tree oid provided for the default users configured in the USM table is listed as shown in the image below:

2. To add a new entry to this table, enter the following information in the space provided.
   - **Type**: Select the type as either "included" or "excluded". The type "excluded" denotes that the specified view cannot be accessed by the user.
   - **View Name**: The view name can be any name. Separate view names can be defined for read, write and notify access.
   - **SubTree**: The subtree is the subtree oid in the MIB for which read, write, notify access is allowed for the user.
   - **Mask**: The "mask" field is used to control the elements of the OID subtree that should be considered as relevant when determining the view in which an OID is in. Normally, the OID is included on whole, so you'll need a mask with as many bits set as there are in the OID elements. The default value is "ff". To know more, click on Mask.

3. Click on OK to add the entry to the table.
**Mask:**

The "mask" field is used to control the elements of the OID sub-tree that should be considered as relevant when determining the view in which an OID is in. Normally, the OID is included on whole, so you'll need a mask with as many bits set as there are in the OID elements.

Thus, ".1" (the whole dod tree) has one element, so the mask has one bit set - i.e. '80' (in hex). ".iso.org.dod.internet.mgmt.mib-2" has six elements, so six bits set ('fc'). If there are more than eight elements, you specify the longer masks as single octet values, separated by dots (e.g. 'ff.c0' for 10 bits)

The mask value defines how the incoming OID should be matched with the SubTree value. For example, if the subtree value is .1.3.6 and the incoming OID is .1.3.6.1.2.1.1.0, and the Mask is ff(H), (ie) 1111 1111. Then the incoming OID should exactly match .1.3.6 as the first three values are 1 (1 stands for exact match). If the Mask value is 1011 1111, then it is enough if the first and third value i.e 1,6 in the SubTree value match with the incoming OID as 0 does not expect exact match.

After adding the new user entry to the USM and VACM Tables, click on **Save** to update the changes in the SNMPv3 xml configuration files. The v3 agent can now be accessed by the new user.

To test the v3 agent with the new user, follow the steps specified in Testing the SNMPv3 agent with default users by specifying the new user name, security level, password and context name.
Starting the SNMP Agent from Command Line

- Overview
- Starting the Simulated SNMP Agent
- Examples

Overview

The command line SNMP Agent Simulator is purely NON-GUI and uses only character user interface for obtaining various inputs for the simulator. It can be invoked by executing the StartSnmpAgent.sh/.bat in <Simulator_Home>/bin/cmdline directory. Starting the agent from command line is more faster as the non-GUI agent saves on memory.

Starting the SNMP Agent

To start the command line simulator, execute the StartSnmpAgent.sh /.bat file with the following options.

Configuration File Options : One of the below given options is mandatory

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-m &lt;MibFile&gt;</td>
<td>The name of the MIB file to be loaded with absolute path. Multiple mib files separated by comma (,) can be given. If this option is specified, the agent will respond with simulated values in the MIB file.</td>
</tr>
<tr>
<td>-c &lt;ConfigFile&gt;</td>
<td>The name of the configuration file to be loaded with absolute path.</td>
</tr>
</tbody>
</table>

MySQL Options :

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>- tn&lt;tablename&gt;</td>
<td>Name of the table to be created in the database if the simulator is started with specifying the -m &lt;MibFile&gt; option. This option is not applicable if the simulator is started with specifying the -c &lt;ConfigFile&gt; option.</td>
</tr>
</tbody>
</table>

Note:
The details of the Database server name, Port number, User name, Password and the Server path is stored in database_params.conf file in <Simulator_Home>/conf directory. You can make the required database settings in this file.

Initialize Services Option : Optional

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-T &lt;trapstatus&gt; [enabled/disabled]</td>
<td>To enable/disable the execution of traps configured in the simulator. By default, the trap service will be enabled. If the service is disabled, the agent will not check for trap configurations.</td>
</tr>
</tbody>
</table>
Options to manage the Simulated SNMP Agent remotely using RMI: Optional

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-rmi</td>
<td>To manage SNMP Agent Simulator remotely.</td>
</tr>
</tbody>
</table>

Examples

1. To start the simulator with loading MIbs:
   
   StartSnmpAgent.sh -m mibs\RFC1213.MIB -p 8005 -tn <tableName>
   
2. To start the simulator by loading configuration:
   
   StartSnmpAgent.sh -c snmpagents\snmpconfig_main.xml -p 8005
TL1 Agent Simulator Guide

The TL1 Agent Simulator integrated with Simulation Toolkit, provides an intuitive UI to simulate TL1 manageable devices. It takes the TCS file as input and creates default values for all the command code and the Autonomous code in the loaded file. The default values are generated, based on the parameter names and syntax defined in the TCS file. These values are stored in a configuration file.

The TL1 Agent Simulator also supports to simulate delayed response, error response, acknowledgment for specific responses, and configure echo requests. The use of Jython-based scripting enables simulation of real time behavior of devices.

The following topics are covered in this section:

- Getting Started
- Simulating Default Values
- Configuring Values
- Re-simulating Values
- Saving TL1 Agent Configuration
- Starting and Stopping the Agent
- Starting the TL1 Agent from Command Line
- Testing the TL1 Agent
- Using Built-in Commands
- Using Self-Monitoring Commands
Getting Started

- Starting TL1 Agent Simulator
- User Interface

Starting TL1 Agent Simulator

The TL1 Agent Simulator can be started in the following ways:

- From the Simulation Toolkit Launcher by invoking `<Simulator Home>/SimulationToolkit.bat/sh` file
- By invoking TL1AgentSimulator.bat from `<Simulator Home>/bin` directory, for Windows OS.
- By invoking TL1AgentSimulator.sh from `<Simulator Home>/bin` directory, for Linux/SolarisOS.

The TL1 Agent Simulator will store the simulated data in MySQL database. The MySQL database is bundled with the product and is part of the package. Refer to MySQL for more information. The following details have to be provided for the database settings dialog:

<table>
<thead>
<tr>
<th>User Name</th>
<th>root</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>(No password is configured. Leave the field blank)</td>
</tr>
<tr>
<td>Server Name</td>
<td>localhost</td>
</tr>
<tr>
<td>Server Port</td>
<td>3306</td>
</tr>
<tr>
<td>MySQL Home</td>
<td>c:\products\SimulationToolkit\AdventNet\Simulator\mysql\bin shows the MySQL bin in the product directory.</td>
</tr>
</tbody>
</table>

In Linux OS, it is not necessary to specify the MySQL Home path.
User Interface

A general introduction on the UI of the TL1 Agent Simulator will help you to get familiar with the tool.

The TL1 Agent Simulator UI, consists of two panels : The left panel consist of the TL1 Message tree, where the TCS file or the Configuration file is loaded and unloaded. The right-side panel displays the parameters that can be configured for the selected command.

Selecting the Autonomous code in the TCS file will display the parameters that can be configured for an Autonomous message, as shown in the screen-shot below :
Simulating Default Values

- Overview
- Loading TCS File
- Loading Configuration File

Overview

The first step in simulating a TL1 agent is to load the TCS file. The TL1 Command Set (TCS) file contains information about the TL1 message in the form of TL1 Command Sets. The rules for defining a TL1 message are defined in GR-831 standard. These command sets are loaded to the simulator and displayed in the form of a tree.

In this topic, you will learn to create a basic simulation by loading TCS File(s) in TL1 Agent Simulator and simulating default values for all the command code and the Autonomous code in the loaded file. If you already have a saved configuration of the simulated TL1 agent, refer to Loading Agent Configuration.

Loading TCS File

You can load a TCS file(s) to simulate a new TL1 agent with default values. To load a TCS File,

1. Click File -> Load TCS option in the menu bar or click the Load TCS File icon from the toolbar. The Choose TCS File dialog prompts up with all the TCS files in the <Simulator_Home>/tcs directory.

2. Select the File name you want to load and click the Open button. The file will be loaded in the TL1 tree and the simulator will create default values for all the command code and the Autonomous code defined in the loaded TCS file.

3. The simulator prompts another dialog asking where you like to store the simulated values. You can provide any desired table name, example: sim_table1 and click OK. The simulated values will be stored in the table.

Now, you have created a basic simulation of a TL1 agent with default values. Any variation on this basic simulation can be created for use with the TL1 Agent Simulator. Refer to section Configuring Values, to learn more. You can also save this basic simulation configuration. As soon as you save, the configuration file is loaded.

Loading Multiple TCS File

The TL1 Agent Simulator supports to load more than one TCS file in the TL1 tree. You can load the files one by one. The command codes for the loaded files should be unique.

Note:
You can define your own TCS file using TL1 Message Builder, bundled with the product. The TL1 Message Builder is used to create and modify TL1 Command Set definitions in the XML format.

Loading Configuration File

The option Load Configuration can be used, if you already have a saved configuration of the TL1 agent and you want to make changes to the existing configuration file. To load a saved Configuration File,
1. Click **File --> Load Configuration** or click the *Load Configuration* icon from the toolbar. The **Choose Configuration File** dialog comes up with the .prp file present in the directory.

2. Select the File name you want to load and click the **Open** button.

3. The file will be loaded in the TL1 tree.
Configuring Values

When a TCS is loaded in the TL1 Agent Simulator, a default value will be assigned to each command code and Autonomous code defined in the TCS file. The default values are generated based on the parameter names and syntax defined in the TCS file.

The TL1 Agent simulator provides an intuitive UI to configure the parameters defined in the TCS file for Input Command Messages, acknowledgments, Output Response Message, and Autonomous Messages. It also supports to configure scripts to define agent Behavior, simulate delayed response, error response, acknowledgment for specific responses, and configure echo requests.

The topics covered in this section explains the format of different types of TL1 messages and using the TL1 Agent simulator UI to configure the parameters defined in the TCS file.

Refer to the links given below to learn about the configurations in detail.

- Input Command Message
- Acknowledgments
- Output Response Message
- Autonomous Message
- Echo Message
- Configuring Behavior
- Simulating Random Variations in Response
- Configuring Security
Input Command Message

- Overview
- Input Command Message Format
  - Command Code
  - Target Identifier (TID)
  - Access Identifier (AID)
  - Correlation Tag (CTAG)
  - General Block
  - Message Payload Block (MPB)
- Configuring Values for Input Command Message
  - Target Identifier (TID)
  - Access Identifier (AID)
  - Message Payload Block (MPB)
- Deleting AID and MPB Values

Overview

An Input command message is a message from an OS or other source (i.e., manager) to a Network Element (NE) (i.e., agent). The message requests the NE to perform some action. The TLI Agent Simulator supports to configure parameters defined in the TCS file for the Target Identifier, Access Identifier and Message Payload blocks.

This topic explains the different blocks which form the Input Command Message and the steps involved in configuring the TID, AID, and MPB parameter values. Users who are familiar with the TL1 message structure can skip the relevant contents in the topic.

Message Format

The general structure of a TL1 Input message is of the form:

```
<command_code>:[<targetidentifier>]:[<accessidentifier(s)>]:<correlationtag>:
[<general block>]:[<message_payload_block>];
```

Command Code

It determines the action to be taken at the NE as a result of receiving the input message. Each command must begin with a command code consisting of a mandatory verb followed by up to two optional identifiers, each separated by hyphen.

```
<command_code> ::= <verb>[-<modifier>[-<modifiers>]]
```

Verb identifies the action to be taken at the NE as a result of receiving a TL1 message from an OS. The command code modifiers are optional depending upon the specific command and application domain. The first modifier identifies where the action is to be applied in the NE. The second modifier may be used to further define the identity of the object upon which the action is to be taken.
For example, the command DLT-CRS-T0 will disconnect (DLT) one or more cross-connected (CRS) DS0 object entities (T0). The modifier CRS is further defined to identify T0 object entities.

**Target Identifier Block**

An input message associated with the management of a particular object may be directly addressed to an NE or it may be routed to or through one or more intermediate NEs. The Target Identification code (TID) parameter block provides the capability within the TL1 message format to perform network routing tasks. The presence of the TID in all input commands is a requirement, but its value may be null (represented by two successive colons).

The TID block contains a single position-defined parameter that identifies the end-target NE to which the input command is directed. The value of TID may be any valid simple or compound TL1 identifier or text string limited to 20 characters.

**Access Identifier Block**

The Access IDentifier (AID) block normally contains one or more parameters that uniquely identifies the entity within the target NE to be acted upon by the input message. Typical examples of entities addressed by the AID parameter values are

- Circuits and common equipment modules having hierarchical relationships.
- Record entities within the context of an administrative view of the NE database.
- Test session and Test Access Point aliases.

**Correlation Tag (CTAG)**

It contains one position defined parameter to serve as a means of correlating an input command with its associated output response. The OS assigns an arbitrary non-zero CTAG value and it is the responsibility of the NE to copy this value into the output response associated with that input command. The value of CTAG must either be a TL1 identifier or a non-zero decimal number consisting of not more than six characters.

**General Block (GB)**

It includes support parameters whose values affect the way in which the input command is to be executed in the NE. The presence of GB in all input commands is a requirement but its value may be null (represented by two successive colons). The form of the General Block is

```
::[<delay activation parameters>],[<contingency flag>],[<indirect data retrieval identifier>]:
```

Delay Activation is a function whereby an input message may be stored in a message pending buffer at the NE for final execution at some later time. To simulate delayed response, refer to Simulating Random Variations in Response -> Simulate Delayed Response. The Delay Activation function is provided by a set of parameters within the GB of the form:

```
[ON]=<order no.>,[DATE]=<date>,[TIME]=<time>
```

Contingency Flag parameter is a boolean data type within the GB that indicates, when set true, a dependent relationship among the several records specified in the AID data block of a multi-unit command. If CF is false, partial installation of the records in a multi-unit message
may be completed with a report sent to the OS listing the records that were not successfully installed in a database.

Indirect Data Retrieval Indicator is a functional capability by which information may be retrieved from more than one linked administrative view by a single RTRV command.

**Message Payload Block (MPB)**

This block indicates the subject matter of the message. This section may consists of zero or more data blocks in the form (::{ Px}()^{*}); where Px represents a data item. Each data block is delimited by colons (:) and the last terminated by a semicolon (;).

Each data block may have an unlimited number of data items, each delimited by commas (,). The data items within a data block may be either name-defined ( <keyword>=<value> ) or position-defined where values are specified and the keyword is implied by its position in the data block.

The *semi-colon* character (;) terminates a TL1 input message. This is required for all TL1 message types.

**Configuring Values for Input Command Message**

**Configuring Target Identifier (TID)**

To configure TID,

- Click on the **Agent Settings** icon or Select **Edit -> Agent Settings**.
- Specify the NE Name to which the input command is directed and click **OK**.

**Configuring Access Identifier (AID)**

The AID Block parameter values can be configured only if AID block is defined in the TCS file, for the input message.

To configure Access Identifier Parameters, do the following:

1. Select the required command code from the loaded command set file (as .tcs file or .prp configuration file format).
2. If the command code allows configuration of new Access Identifiers, the **Add** button in the AID field is enabled.
3. Click **Add** to configure the AID value. The **Configure Access ID** dialog is displayed with the Param Name and Param Type.
4. Specify the AID value for the Param Name and click **OK**.

**Configuring Message Payload Block (MPB)**

The Message Payload block parameter values can be configured in the simulator, only if this block is defined in the TCS file, for the input message. To configure Message Payload Block, do the following:

1. Select the required command code from the loaded command set file (as .tcs file or .prp configuration file format).
2. Select the Access Identifier (AID), if any, configured for the selected command code.
3. If the command code allows configuration of Message Payload block, the **Add** button in the Message Payload field is enabled.
4. Click **Add** to configure the Message Payload values. The **Configure Message Payload Block** dialog is displayed with the Param Name and Param Type.
5. Specify the message payload value for the Param Name and click **OK**.

### Deleting AID and MPB Values

To Delete an AID value, do the following:

1. Select the required command code from the loaded command set file (as .tcs file or .prp configuration file format).
2. If the command code allows configuration of AID values, the AID field on the right-hand side gets enabled.
3. Select the AID to be deleted from the list. Click **Delete** to delete the Access Identifier.

To Delete a Message Payload value, do the following:

1. Select the required command code from the loaded command set file (as .tcs file or .prp configuration file format).
2. Select the message payload block value to be deleted from the list and click **Delete**.
### Acknowledgments

- **Overview**
- **Message Format**
  - Acknowledgment Code
  - Configuring Acknowledgments

#### Overview

An acknowledgment is a short reply from the NE indicating that an input command message is being acted upon or has been immediately rejected. The essential purpose of an acknowledgment is to reassure a human user that a command that takes a long time to execute has been received by the NE.

An acknowledgment is later followed by an output response message to the command. However, in some circumstances, an acknowledgment is the only output message triggered by a command. An acknowledgment should be used if an output response message to the command cannot be transmitted within two seconds of its receipt.

#### Message Format

The format of an acknowledgment is as follows:

```
<acknowledgment code> ^ <ctag> <cr> <lf>
```

- **acknowledgment code**: Identifies the reason for the acknowledgment.
- **CTAG**: Identifies the associated input command.
- **(<) character**: The acknowledgment terminator.

#### Acknowledgment Code

The valid values for acknowledgment code and their meanings are given below.

1. **IP - In Progress** and **PF - Printout Follows**
   
   The request has been initiated. These acknowledgments should produce output messages that give a termination report or a termination report and results of the command. They are used often for test and access messages requiring a long execution time. These acknowledgments are often followed by either Completed or Denied response messages.

2. **OK - All Right**
   
   The command was received and the requested action was initiated and completed. In addition, OK is used in response to a command that has been cancelled by inclusion of the CAN (Cancel) character.

3. **NA - No acknowledgment**
   
   This command is sent if initiation or execution of the requested command is not possible. Under abnormal conditions, NA is sent when command has been accepted but control of the processing has been lost. It can also be used to respond in
circumstances if the command is garbled during transmission. If the CTAG value of
the command could not be determined, then Zero(0) should be used as a ctag value
for the acknowledgment.

4. **NG - No Good ( Not Used )**

The command is valid but the requested action conflicts with current system or
equipment status. For inadequate system resources use RL instead of NG. This
acknowledgment code is seldom used because specific error code in output
response messages can be employed to signify the same information. It can be used
if desired.

5. **RL - Repeat Later**

The requested action cannot be executed due to unavailable system resources
due to system overload, excessive queue lengths, busy programs, etc. The
command can be sent at later time for any response.

6. **No - No acknowledgment message is configured.**

**Configuring Acknowledgments**

To configure acknowledgment message for a command,

- Select the required command code from the loaded command set file ( as .tcs file or .prp
  configuration file format ).

- Select the Access ID and Message Payload value from the list, for which acknowledgment
  must be configured.

- Select the required acknowledgment Message from the values listed in the combo box.

**Note:** If Access ID and Message Payload are defined for a command, the
acknowledgment values in the combo-box will be enabled only after selecting the
Access ID and Message Payload value from the list.
Output Response Message

- Overview
- Output Response Message Format
  - Header
  - Completion Code
  - Text Block (Response Line)
- Configuring Values for Output Response Message
  - Configuring Completion Code
  - Configuring Response Line
    - Valid Responses
    - Error Responses
- Modifying and Deleting Responses
- Sending Response Messages using Jython Scripts

Overview

A TL1 output response message is the response to a TL1 input command message.

Output Response Message Format

The general structure of a TL1 output response message is of the form given below. It consists of an optional text block and all the other blocks are essential.

```
<header> <response identification> [<text block>] <termination>
```

Header

The Header represents information common to all output response.

```
<cr><lf><lf>^^^<sid>^<year>-<month>-<day>^<hour>:<minute>:<second>
```

It contains System identifier `<sid>`, date, and time stamps. `<sid>` is restricted to 20 characters maximum and identifies the NE generating the message. The syntax of `<sid>` is any TL1 identifier or text string. The `<year><month><day>` construct represent the day in which the output response is generated. The `<hour><minute> <second>` construct represent the time at which the output response is generated.

Completion Code (Response Identification)

The form of the response identification is :

```
<cr><lf>M^^<ctag>^<completion code>
```

This construct consists of three components, namely the character M, a correlation Tag, and a completion code. The character M signifies that the message is the response to the input command. The ctag must be the same as that of the input message in order to associate the
response with the proper input command. The various completion codes are described below.

- **COMPLD**: Total successful execution of the input command.
- **DENY**: Total denial of the input command.
- **PRTL**: Partial successful execution of the input command.
- **DELAY**: Successful lining of the input command submitted for delayed activation.
- **RTRY**: Output response of a input retrieve command that retrieves extensive information from the NE and uses more time for processing.

The completion codes supported by TL1 Agent Simulator are **Compld** and **Deny**.

**Text Block (Response Line)**

The optional [<text block>] is used to represent information specific to the particular autonomous message. The format of the text block is as follows:

```
((<cr><lf>^^^<unquoted line>) | (<cr><lf>^^^<quoted line>)
|(<cr><lf>^^^<comment>))
```

It consists of three components, namely unquoted line, quoted line, and comment. Both quoted and unquoted lines consists of text that is parsable, while comment is not. The most popular usage of unquoted line is for representing error code in some response messages. The quoted line consists of parsable text and shall be preceded and followed by double quotes. The comment line is used to allow free format text. It should be preceded by ( /* ) and followed by ( */ ).

The TL1 Agent Simulator supports only quoted and unquoted response lines.

The semi-colon character (;) terminates a TL1 output message. The form of the terminator is :<cr><lf>( ; | > ). This is required for all TL1 message types.

**Configuring Values for Output Response Message**

The TL1 Agent Simulator supports to configure the parameter values for the response message defined the TCS file.

**Configuring Completion Code**

The completion code supported by TL1 Agent Simulator are **Compld** and **Deny**. To configure completion code, do the following:

1. Select the required command code from the loaded command set file (as .tcs file or .prp configuration file format).
2. Select the Access ID and Message Payload value from the list, for which Completion code must be configured.
3. Select the required completion code from the values listed in the combo box. The default completion code is **compld**.

**Note:**

1. If Access ID and Message Payload are defined for a command, the Completion Code values in the combo box will be enabled only after selecting the Access ID and Message Payload value from the list.
2. The completion code supported by TL1 Agent Simulator are **Compld** and **Deny**.
Configuring Response Line

A response line can be configured in the TL1 Agent Simulator, only if the Response message is defined in the TCS file for the TL1 command. The response message may be a valid response message or error response message or both.

To configure a **Valid Response**, do the following:

1. Select the required command code from the loaded command set file (as .tcs file or .prp configuration file format).
2. Select the **Access ID** and **Message Payload** value from the list, for which response text must be configured.
3. If a valid response message is defined in the tcs file, the **Configure Valid Response** text field is enabled.
4. Click the text field to display the **Configure Valid Responses** dialog with default response line.
5. Click **Add** to configure valid responses for the selected **CC-AID-MPB** combination.
6. The **ConfigureParamValues** dialog is displayed. The **Response Line type** displays the textblock type, as defined in the tcs file. The textblock type may be quoted or unquoted. The Parameter name and Parameter type defined for the response message is listed with default values.
7. Enter the required parameter values and click **OK**, to add the response to the list.
8. Click **OK** to exit from the **Configure Valid Responses** dialog.

To configure an **Error Response**, do the following:

1. Select the required command code from the loaded command set file (as .tcs file or .prp configuration file format).
2. Select the **Access ID** and **Message Payload** value from the list, for which response text must be configured.
3. If an error response message is defined in the tcs file, the **Configure Error Response** text field is enabled.
4. Click the text field to display the **Configure Error Responses** dialog.
5. Click **Add** to configure error responses for the selected **CC-AID-MPB** combination.
6. The **ConfigureParamValues** dialog is displayed. The **Response Line type** displays the textblock type, as defined in the tcs file. The textblock type may be quoted or unquoted. The Parameter name and Parameter type defined for the response message is listed with default values.
7. Enter the required parameter values and click **OK**, to add the response to the list.
8. Click **OK** to exit from the **Configure Error Responses** dialog.

**Note:**

- Only quoted and unquoted response lines are supported.
- Multiple Response lines can be configured.
- If Completion Code is Compld, a valid response is sent. If Completion Code is Deny, an Error Response is sent.
Modifying and Deleting Responses

The TL1 Agent Simulator, supports to modify the response parameter values configured in the simulator or delete an existing response line.

To modify a response line, do the following:

1. Select the response line to be modified, in the Configure Valid/Error Responses dialog.
2. Click Modify. The ConfigureParamValues dialog is displayed. Enter the required parameter values and click OK, to add the modified response to the list.

To delete a response line, do the following:

1. Select the response line to be deleted, in the Configure Valid/Error Responses dialog.
2. Click Delete. The response line will be deleted from the list.

Sending Response Messages using Jython Scripts

The script api method sendResponseMessage() in SNMP_TL1_SimScrlntf.html in <simulator home>/examples/javadocs directory, enables you to send different response messages for the same command code. Refer to the topic, Configuring Behavior to learn more about configuring scripts.
Overview

An Autonomous message is a message that is sent from the NE to the appropriate OS without having an explicit input message associated with it. Typical scenarios where Autonomous messages are used include

- Reporting of alarmed or non-alarmed trouble events.
- Reporting of scheduled diagnostic tests in the NE.
- Reporting of Performance Monitoring data.
- Reporting of a change in the NE's database.
- Periodic reporting of selected NE conditions.

This topic explains the different blocks which form the Autonomous Message and the steps involved in configuring the autonomous message in TL1 Agent Simulator.

Autonomous Message Format

The general structure of a TL1 Autonomous message is given below. Here the text block is the optional field and all other fields are essential.

```
<header> <auto id> [ <text block> ] <terminator>
```

Header

The Header represents the information common to all output responses and autonomous messages. It contains System identifier <sid>, date, and time stamps.

```
<cr><lf><lf>^^^<sid>^<year>-<month>-<day>^<hour>:<minute>:<second>
```

It contains System identifier <sid>, date, and time stamps.<sid> is restricted to 20 characters maximum and identifies the NE generating the message. The syntax of <sid> is any TL1 identifier or text string. The <year><month><day>
construct represents the day in which the output response is generated. The <hour><minute> <second> construct represents the time at which the output response is generated.

AutoID

The AutoID indicates the severity and the nature of the Autonomous message. The <Auto id> entry for an autonomous message is of the form:

```
<cr><lf> <almcde>^<atag>^<verb>[^<modifier>[^<modifier>]]
```

<almcde> is the alarm code. It can be any of the following based on the severity of the autonomous message. Valid values in decreasing order of severity are

- *C - Critical Alarm
- ** - Major Alarm
- * - Minor Alarm
- A - Non-Alarm Message

<atag> is the Autonomously Generated Correlation Tag. It is assigned by the NE. Must be sequential and must be included in all autonomously generated messages. It allows an OS to correlate spontaneous outputs triggered by a common problem and also to identify whether the OS has failed to receive any output.

<verb>[^<modifier>[^<modifier>]] entry identifies the nature of the spontaneous output. The first identifier is a required entry and indicates the message verb. The autonomous message can have two optional modifiers separated by space character.

Text Block

The optional [<text block>] is used to represent information specific to the particular autonomous message. The format of the text block is as follows:

```
(<cr><lf>^^^<unquoted line>)(<cr><lf>^^^<quoted line>)(<cr><lf>^^^<comment>)
```

It consists of three components, namely unquoted line, quoted line, and comment. Both quoted and unquoted lines consists of text that is parsable, while comment is not.

The semi-colon character (;) terminates a TL1 Autonomous message. The terminator block has the form <cr><lf> ( ; | >) . This is required for all TL1 message types.

Enable/Disable TL1 Autonomous Message Service

The TL1 Agent Simulator/ Network Simulator provides option to enable/disable the execution of autonomous messages configured in the TL1 agent, at agent runtime. If autonomous message service is disabled, the agent will not check for autonomous message configurations before sending the response. Hence, you can use this option to improve the performance of the TL1 agent, if autonomous messages are not required to be executed.

By default, the service is enabled and the autonomous messages configured (if any), will be executed. To disable the service,

- Select Operations -> Initialize Services in TL1 Agent Simulator or select Run-> Initialize Service in Network Designer.
- The Initialize Service dialog is displayed.
- Deselect "Enable TL1 Autonomous Message Service".
• Start the Agent. Now, the autonomous messages configured in the simulator will not be executed.

**Note:** The options in the *Initialize Services* dialog will be disabled when the TL1 agent is started.

### Configuring Values for Autonomous Code

The Autonomous message comprises of the Alarm code and Text Block combination. The TL1 Agent Simulator supports to configure any number of Alarm code and Text block combination for a single Autonomous code.

To configure the TextBlock for the Autocode,

1. Select the required Autonomous code from the loaded TCS file (as .tcs file or .prp configuration file format). The dialog on the right side provides options to select the required Alarm Code and configure Text Blocks for the selected Autonomous command.

**Note:** You can configure any number of Alarm code and Text block combination for a single command. The Alarm Description list box is used to identify the Alarm code and Text block combination.

2. Click the **Add** button of the Alarm Description list box. Enter the value for Alarm Description and click **OK**, to add to the list. You can add any number of alarm description to the list.

3. Select a **Alarm Description** from the list, for which the Alarm Code, Text Block combination must be defined.

4. Select the **Alarm Code** from the values listed in the combo box.

5. To configure Text block, click **Add** button of the Configure Text Block. The **ConfigureParamValues** dialog is displayed. The Response Line type displays the textblock type, as defined in the tcs file. The text block type may be quoted, unquoted or commented. The Parameter name and Parameter type defined for the Autonomous message is listed with default values. Enter the required parameter values and click **OK**, to add the text to the list. You can configure any number of text blocks.

**Example**

1. Select the Autonomous code, **REPT ALM EQPT** defined in Acme-MSU.tcs file.

2. Click the **Add** button of the Alarm Description list box. Enter the value for Alarm Description as **A1** and click **OK**, to add to the list. You can add any number of Alarm descriptors to the list.

3. Select the Alarm Description A1 from the list, for which the Alarm Code, Text Block combination must be defined.

4. Select the Alarm Code as **Major** from the values listed in the combo box.

5. To configure Text block, click **Add** button of the Configure Text Block. The **ConfigureParamValues** dialog is displayed. Enter the following parameter values:
   - i. aid : SLOT-7
   - ii. ntfncnde : MJ
   - iii. conditiontype : HITEMP

6. Click **OK** to add the above values to the text block. Now, the screen will look like this:
Configuring Request-Based Autonomous Messages

The TL1 Agent Simulator supports to generate Request Based Autonomous Messages, by configuring Autonomous code and Alarm Descriptor for the selected Input Command Code, Command Code - AID or Command Code - AID - MPB combinations. When a request is made for the Command, Command Code - AID or Command Code - AID - MPB, on which the message is configured, the NE sends the configured Autonomous message to the OS concerned.

To configure Request based Autonomous messages,

1. Select **Configure -> Request Based Autonomous Message** from the menu bar or the icon from the toolbar. The **Request Based Autonomous Message** dialog is displayed.
2. Click **Add** button to configure a new Request Based Autonomous Message.
3. You have the option to configure Request Based Autonomous Message at the following levels.
   - To configure request based autonomous message **for a command code**, select the required command code.
   - To configure request based autonomous message **for a Command code - AccessID** combination,
     - Select the required command code. The Access ID configured for the selected command is displayed on the right-frame.
     - Select the required Access Id.
To configure request based autonomous message for a Command code - Access ID - Message Payload combination,

- Select the required command code. The Access ID configured for the selected command is displayed on the right-frame.
- Select the required Access ID. Selecting the AccessID, will display the Message Payload configured.
- Select the required Message Payload.

4. Browse and select the required Autonomous Code from the Autonomous Code dialog. Selecting the autonomous code will display the Alarm Descriptors configured for the code. Select the required Alarm Descriptor and click OK.

5. Click OK to add the selection to the list.

6. Click Close to exit from the dialog.

Example

1. Configure a Request based autonomous message for the Input Command RTRV-EQPT with autonomous code REPT ALM EQPT and Alarm Descriptor A1, as shown in the screen-shot below and click OK to add the entry to the list.
2. Perform a Retrieve operation from TL1 Craft Interface or from any other TL1 manager applications for Command Code `RTRV-EQPT` defined in Acme-MSU.tcs. The NE responds to the request and sends the configured Autonomous message, as shown in the screen shot below:

```
Show Autonomous and Response message in the same window

Message sent: RTRV-Eqpt::1:::
AdVTLSim 2003-03-04 18:18:25
M 1 COMPLD
  *CARD ID:1,TYPE,WKMOD:IS,IDLE*

AdVTLSim 2003-03-04 18:18:25
**1.0 REPT*ALM*EQPT
  *admncncde,nfncde*
  *SL0T:8,MJ,HITEMP*
```

Configuring Timer-Based Autonomous Messages

The TL1 Agent Simulator supports to generate Timer-Based Autonomous Messages by configuring Autonomous code and Alarm descriptor for a specific date and time. These messages are generated at the configured time.

- Select **Configure -> Timer Based Autonomous Message** from the menu bar or the icon from the toolbar. The **Timer Based Autonomous Message** dialog is displayed.
- Click **Add** button to configure Autonomous code, alarm descriptor, and timer details.
- The **Configure Timer Based Autonomous Message** dialog opens with the Autonomous Code defined in the TCS file, displayed on the top portion of the dialog. Selecting an Autonomous code, will display the Alarm descriptors configured for the selected Autonomous code.
- Select the Alarm Descriptor for the Autonomous code.
- Enter the Timer Details.

**Timer Details**

- **Staring Time**: Configure the date and time at which the simulator will start sending Autonomous messages.
- **Periodically send**: Check this option to send Autonomous message on a periodical basis. Specify the duration of the message to be sent as once in number of hour(s) / day(s).
- **Total Messages**: Specify the total number of messages that you want to send. Eg: 100.
- **Time Interval**: You can also specify the time interval between the messages that are to be sent. The time interval option is available in Milliseconds. Example: 1000 Milliseconds.

- Click **Ok** to add the configuration to the list.
- Click **Close** to exit from the dialog.

**Example**

1. The Autonomous code, `REPT ALM EQPT` is defined in Acme-MSU.tcs file. In the TL1 Simulator, the Autonomous code `REPT ALM EQPT` is selected and the following configuration is made: Alarm Code : **Critical**, TextBlock : Default Text Block
2. The above autonomous message with Autonomous code `REPT ALM EQPT`, Alarm Code `Critical`, and the Text Block must be sent at a specified time. This is defined using `Configure -> Timer Based Autonomous Message` option, as shown in the screen shot below.

![Configure Timer Based Autonomous Messages](image)

You have configured Total messages as 3 with a timer interval of 5000 milliseconds and choose to periodically send it once in 5 days.

3. The above configuration will generate 3 messages with a time interval of 5 seconds each, once in 5 days, starting from the Date and Time specified. The timer based autonomous message can be viewed from the TL1 craft interface as shown in the screen shot below:

```
Iatha 2003-03-26 12:51:30
*C 1.0 REPT*ALM*EQPT
  *a1:*ftpnode,*ftpnode*
```

**Deleting Request/Timer Autonomous Messages**

The TL1 agent simulator provides option to delete the configured Request-based/Timer-based Autonomous messages.

To delete Request-based Autonomous messages,
- Select the configuration to be deleted from the Request Based Autonomous Messages dialog.
• Click **Delete**, to delete the configuration.

To delete Timer-based Autonomous messages,

The TL1 agent simulator provides option to delete the configured timer based autonomous messages. To delete request based autonomous messages,

• Select the configuration to be deleted from the **Timer Based Autonomous Messages** dialog.

• Click **Delete**, to delete the configuration
Echo Message

- **Overview**
- **Configuring Echo Message**

**Overview**

When a request is made by an OS or any other source to an NE, the NE sends an output response message for the received input command message. If Echo Message is configured for a command, the NE responds with the request, instead of the actual output response message.

**Configuring Echo Message**

To configure echo messages, do the following:

- Click the Echo Message icon or choose Configure -> *Echo Message* option from the menu bar. This brings up the Configure Echo Codes dialog. The dialog displays all the command code in the tcs tree.

- Select the command code for which you would like to configure echo message. Move it to the *Selected Command Codes* list box using the button with >> label.

- The request for the selected command code will be echoed by the simulated TL1 agent.

**Example:**

Load a TCS file and configure Echo message for the command code ACT-IMG. Start the agent. Start the TL1 Browser and send the request to the command code ACT-IMG. You will see the request being echoed.

To delete the Echo codes configured, do the following:

- Select the command code which has been added to the *Selected Command Codes* list box.

- Click on the button with << label. The selected command code will be deleted from the list.
Configuring Behavior

Overview

The most important part of simulation is to set up the simulated environment. You can set up the desired environment by defining the behavior of the TL1 Devices, so that they will behave like a real device. The behavior of devices is done through Jython scripts.

The TL1 Agent Simulator provides a TL1ScriptInterface, which defines methods to perform various TL1 operations from a JPython script. For example, you can add or delete Access ID, Response Lines and Message Payload for a specific command code, send different response messages for the same command code, get current Access ID, Message payload values, set acknowledgment and completion code, etc.

The javadocs for the Script API methods is provided under the `<SimulatorHome>/examples/javadocs` directory. The name of the Script API file is `SNMP_TL1_SimScrIntf.html`.

Enabling/Disabling TL1 Behavior Service

The TL1 Agent Simulator provides option to enable/disable the execution of behavior scripts configured for the TL1 agents, at agent runtime. If behavior service is disabled, the agent will not check for script configurations before sending the response. Hence, you can use this option to improve the performance of the TL1 agent, if behavior is not required to be executed.

By default, the service is enabled and the behavior scripts configured (if any), will be executed.

To disable the service,

- Select Operations -> Initialize Service in TL1 Agent Simulator.
- The Initialize Service dialog is displayed.
- Deselect "Enable TL1 Behavior service".
- Start the Agent. Now, the behavior scripts configured in the simulator, will not be executed.

Note: The option Operations -> Initialize Service will be disabled when the TL1 agent is started.

Configuring Behavior

The TL1 agent simulator, supports to configure request based scripts to simulate real time behavior of agents. Scripts can be configured on a Command Code, Command Code - AccessID or Command Code - AccessId - MessagePayload combination.
The configured scripts will be copied to the configuration directory. It will be executed at run time, when a request is made for the command, before and/or after processing the request.

Follow the steps given below, to configure scripts:
1. Select **Configure -> Behavior Simulation**. The Script Table dialog is invoked. This dialog displays the scripts that are already configured.
2. Click **Add**, to configure a new script. The Configure Script dialog is invoked. This dialog displays the TCS tree on the left-frame. Expand the TCS tree.
3. You have the option to configure the script file at the following levels.
   - To configure script for a command code, select the required command code.
   - To configure script for a Command code - AccessId combination,
     - Select the required command code. The Access ID configured for the selected command is displayed on the right-frame.
     - Select the required Access Id.
   - To configure script for a Command code - Access ID - Message Payload combination,
     - Select the required command code. The Access ID configured for the selected command is displayed on the right-frame.
     - Select the required Access Id. Selecting the AccessID, will display the Message Payload configured.
     - Select the required Message Payload.
4. Specify the condition to execute the script. The following options are available:
   - Before processing request
   - After processing request
   - Before and after processing request
5. Browse and select the script file to be configured. The sample scripts are available under `<simulator home>/examples/scripting/tl1` directory. Click **Open** to configure the script file.
6. Click **OK** to add the script configuration in the Script Table.
7. Click **Close** to exit the dialog.

**Scripts in Examples Directory**

The following sample scripts are defined in the `<simulator home>/examples/scripting/tl1` directory. A brief overview on each of the script file is provided below.

1. **AddAccessID.py**

   The script file AddAccessID.py uses the method `addAccessID()` to add a new AID value to the Command Code, at agent run time. This method takes the Command code and AccessId as its parameters.

   Format:
   ```python
tlscriptinterface.addAccessId("ED-FAC","ST1-9-1");
```

   When this script is executed, the AID value ST1-9-1 is added for the command code ED-FAC. You can replace the sample command code and AID values with your own configuration.

2. **AddLines.py**

   The script file AddLines.py uses the method `addLines()` to Add Response Lines to the Command Code at agent run time. This method takes the following parameters as input:
   - `commandCode` - is the CommandCode present in the agent.
   - `accessId` - is the accessId corresponding to the commandCode.
   - `msgPayload` - is the MessagePayload corresponding to the accessId.
   - `isValid` - is a boolean representing whether the line is to be added for a valid response/error response.
   - `lineType` - represents whether the line is quoted/unquoted.
   - `paramNames` - is the list of parameter names available for line to be added.
   - `paramValues` - is the list of values for the parameter names.
Format:

tl1scriptinterface.addLines ("RTRV-EQPT","ALL", ",", boolean.booleanValue(), 1, ParamNames,ParamValues);

3. AddMsgPayload.py

The script file AddMsgPayload.py uses the method addMsgPayLoad() to add a new Message Payload for a commandCode, accessId combination. This method will also add the accessId if it is not present. It takes the following parameters as input:

- commandCode - is the CommandCode present in the agent.
- accessId - is the accessId to be added.
- msgPayLoad - is the MessagePayload to be added.

Format:

tl1scriptinterface.addMsgPayLoad("ED-EQPT" , "ST1-4" , "CSU");

4. AddAutoLines.py

This script file uses the method addLines() to add the Autonomous Lines to the corresponding Alarm Description for a Autonomous Code at agent RunTime. It takes the following parameters as input:

- AutoCode - The Autonomous Code
- AlarmDescription - The Alarm Description for the Auto code.
- LineType - 0 : for Quoted, 1 : for Unquoted
- String[] Parameter Names - the string array of the Parameter Names present for the Autonomous Code
- String[] Parameter Values - the string array of the Parameter Values corresponding to the Parameter Names

Format:

tl1scriptinterface.addLines (String AutoCode,String
AlarmDescription, int LineType, String[] Parameter Names,
String[] Parameter Values);

4. DeleteAccessID.py

The script file DeleteAccessID.py uses the method deleteAccessID() to delete an accessId for a Command Code at agent run time. This method takes the following parameters as input:

- commandCode - is the CommandCode present in the agent.
- accessId - is the accessId to be deleted.

Format:

tl1scriptinterface.deleteAccessId("ED-FAC", "ST1-9-1");

When this script is executed, the AID value ST1-9-1 is deleted for the command code ED-FAC. You can replace the sample command code and AID values with your own configuration.
5. DeleteLines.py

The script file DeleteLines.py uses the method `delLines()` to delete an existing Response Line configured for a Command Code, at agent run time. This method takes the following parameters as input:

- `commandCode` - is the CommandCode present in the agent.
- `accessId` - is the accessId corresponding to the commandCode.
- `msgPayload` - is the MessagePayload corresponding to the accessId.
- `responseType` - is a boolean representing whether the line is to be deleted for a valid response/error response.
- `lineType` - represents whether the line is quoted/unquoted.
- `paramNames` - is the list of parameter names available for the line to be deleted
- `paramValues` - is the list of values for the parameter names.

Format:
```
tl1scriptinterface.delLines("RTRV-EQPT","ALL", ",", 
    boolean.booleanValue(), 1, ParamNames,ParamValues);
```

You can replace the sample parameter values in the script file with your own configuration.

6. DeleteLineWithValue.py

This script file DeleteLineWithValue.py uses the method `deleteLine()` to delete the response line for the specified Param Name and Value configured for a command code, at agent runtime.

Format:
```
tl1scriptinterface.deleteLine(String commandCode, String 
    accessID, String msgPayLoad, boolean isValid, int lineType, 
    String uniquePName, String uniquePValue);
```

7. DeleteAutoLines.py

This script file DeleteAutoLines.py uses the method `delLines()` to delete the Autonomous Lines of the corresponding alarm description configured for a Autonomous Code, at agent runtime.

Format:
```
tl1scriptinterface.delLines(String AutoCode , String 
    AlarmDescription, int LineType, String[] Parameter Names, 
    String[] Parameter Values);
```

8. DeleteAutoLineWithValue.py

This script file DeleteAutoLineWithValue.py uses the method `deleteLine()` to delete the Autonomous Response Line of the corresponding Alarm Description with the specified Param Name and Value, at agent runtime.

Format:
```
tl1scriptinterface.deleteLine(String autoCode, String 
    alarmDescr, int lineType, String uniquePName, String 
    uniquePValue);
```
9. DeleteMsgPayLoad.py

The script file DeleteMsgPayLoad.py uses the method `delMsgPayLoad()` to delete a MessagePayLoad for a command code, accesssid combination. This method takes the following parameters:

- `commandCode` - is the CommandCode present in the agent.
- `accessid` - is the accessId corresponding to the MessagePayLoad.
- `msgPayLoad` - is the MessagePayLoad to be deleted.

Format:

```python
tl1scriptinterface.delMsgPayLoad("ED-EQPT", "ST1-4", "CSU");
```

10. GetCurrentAID.py

The script file GetCurrentAID.py uses the method `getCurrentAID()`, to GET the accessid that comes in the request. If the accesssid is not given in the request, this method returns an empty string.

Format:

```python
AccessID = tl1scriptinterface.getCurrentAID();
```

11. GetCurrentMPB.py

The script file GetCurrentMPB.py uses the method `getCurrentMPB()` to get the msgPayLoad that comes in the request. If the msgPayLoad is not given in the request, this method returns an empty string.

Format:

```python
MsgPayload = tl1scriptinterface.getCurrentMPB();
```

12. GetAllAIDs.py

The script file GetAllAIDs.py uses the method `getAllAids()` to Get all the Access Identifier to the Command Code at agent runtime. This method takes the Command code as the input parameter. It returns the value as a Vector compound.

Format:

```python
tl1scriptinterface.getAllAids("RTRV-EQPT");
```

13. GetAllMPBs.py

The script file GetAllMPBs.py uses the method `getAllMpbs()` to get all the Message Payloads for the specified command code and access identifier at agent runtime. This method takes the following parameters as input. It returns the value as a Vector compound.

- `commandCode` - is the CommandCode present in the agent.
- `accesssid` - is the accesssid corresponding to the commandCode.

Format:

```python
tl1scriptinterface.getAllMpbs("ED-IP", "ETHER");
```
14. GetNumberOfAIDs.py

The script file GetNumberOfAIDs.py uses the method `getNumberOfAids()` to get the number of access identifiers to the command code at agent runtime. This method takes the following parameters as input. It returns an integer value as the count of the AIDs.

- commandCode - the Command Code for which the number of Access Identifiers should be taken.

Format:
```java
tl1scriptinterface.getNumberOfAids("RTRV-EQPT");
```

15. GetNumberOfMPBs.py

The script file GetNumberOfMPBs.py uses the method `getNumberOfMpbs()` to get the number of Message Payloads to the Command Code and the AID at agent runtime. This method takes the following parameters as input. It returns an integer value as the count of the MPBs.

- commandCode - the Command Code for which the number of Access Identifiers should be taken.
- accessId - the Access Identifier for which the number of Message Payloads should be taken

Format:
```java
tl1scriptinterface.getNumberOfMpbs("RTRV-EQPT");
```

16. GetLine.py

This script file uses the method `getLine()` to get the Response Lines to the Command Code at agent runtime.

```java
tl1scriptinterface.getLine(String commandCode, String accessID, String msgPayLoad, boolean isValid, String uniquePName, String uniquePValue)
```

17. GetAgentPort.py

This script file uses the method `getPort()` to get the port number of the agent from which the agent is started.

Format:
```java
tl1scriptinterface.getPort();
```

18. GetIPAddress.py

This script file uses the method `getIPAddress()` to get the IP Address of the agent. This will be very useful for the Network Simulation of TL1 Devices

Format:
```java
tl1scriptinterface.getIPAddress();
```
19. getScriptCondition()

This script file returns the condition (before/after get) at which the script is executing.

Format:
```
tl1scriptinterface.getScriptCondition();
```

20. IsAIDExists.py

The script file IsAIDExists.py uses the method `isAidExists()` to check whether the given Access Identifier is present in the AID list for the Command Code. This method takes the following parameters as input. It returns a Boolean value as '0' for False and '1' for True.

- `commandCode` - the name of the Command Code
- `accessId` - the value of the Access Identifier to be checked in the specified Command code.

Format:
```
tl1scriptinterface.isAidExists("RTRV-EQPT", "ST1-1");
```

21. IsMPBExists.py

The script file IsMPBExists.py uses the method `isMpbExists()` to check whether the given Message Payload is present in the MPB list for the Command Code and its AID combination. This method takes the following parameters as input. It returns a Boolean value as '0' for False and '1' for True.

- `commandCode` - the name of the Command Code
- `accessId` - the value of the Access Identifier
- `msgpayload` - the Message Payload to be checked for the specified Command code and AID

Format:
```
tl1scriptinterface.isMpbExists("ED-IP", "ETHER", "165.162.1.20,255.255.255.0");
```

22. SetAckCode.py

The script file SetAckCode.py uses the method `setAckMessage()` to SET the acknowledgment Message for a commandCode, accessId, msgPayLoad combination.

**Note**: The script API method `setAckMessage()` cannot be configured to be executed with the 'Before processing request' condition.

This method takes the following parameters as input:

- `commandCode` - is the CommandCode present in the agent.
- `accessId` - is the accessId corresponding to the commandCode.
- `msgPayLoad` - is the MessagePayLoad corresponding to the accessId.
- `ackNo` - is the acknowledgment message to be set.

Format:
```
tl1scriptinterface.setAckMessage(String commandCode , String accessId , String msgPayLoad , int ackNO);
```
24. SetComplCode.py

The script file SetComplCode.py uses the method `setComplCode()` to SET the Completion Code(COMPLD, DENY) for a commandCode, accessId, msgPayload combination. This method takes the following parameters:

- `commandCode` - is the CommandCode present in the agent.
- `accessId` - is the accessId corresponding to the commandCode.
- `msgPayload` - is the MessagePayload corresponding to the accessId.
- `complCode` - is the completion code to be set (COMPLD/DENY).

Format:
```
tllscriptinterface.setComplCode("ACT-USER", "ROOT", "Password", 1);
```

25. SendAutonomousMessage.py

The script file SendAutonomousMessage.py uses the method `sendAutonomousMessage()` to send the Autonomous Message at agent runtime. This method takes the following parameters as input.

- `autoCode` - the name of the Autonomous Code. The modifiers should be separated by ^
- `almDescr` - the value of the Alarm Description for which the Autonomous Message should be sent.

Format :
```
tllscriptinterface.sendAutonomousMessage("REPT^ALM^EQPT","TextBlock");
```

26. SendAutonomouswithNumberofTimes.py

The script file SendAutonomouswithNumberofTimes.py uses the method `sendAutonomousMessage()` to send the Autonomous Message with specified number of times, at agent runtime. This method takes the following parameters:

- `autoCode` - the name of the Autonomous Code. The modifiers should be separated by ^
- `almDescr` - the value of the Alarm Description for which the Autonomous Message should be sent
- `numberOfTimes` - the number of times the Autonomous message must be sent
- `interval` - the time interval between the consecutive Autonomous messages.

Format :
```
tllscriptinterface.sendAutonomousMessage("REPT^ALM^EQPT","TextBlock", 10, 1000);
```

27. SendDenyMessage.py

This script file uses the method `sendDenyMessage()` to send the deny message at runtime

Format :
```
tllscriptinterface.sendDenyMessage(String errorCode, String explanation);
```
28. StopTL1Agent.py

This script file uses the method `stopAgent()` to stop the TL1 agent in the network.

Format:

```python
tl1scriptinterface.stopAgent(String ipAddress, String snmpPort, String tl1Port);
```

29. StartTL1Agent.py

This script file uses the method `startAgent()` to start the TL1 agent in the network.

Format:

```python
tl1scriptinterface.startAgent(String ipAddress, String snmpPort, String tl1Port);
```

30. UpdateParamValue.py

This script file is to update the parameter value of a Response Line configured for a command code at runtime.

uniquePName & uniquePValue indicates the parameter name and value that will be unique across all the lines in the response.

Format:

```python
tl1scriptinterface.updateParamValue(String commandCode, String accessID, String msgPayload, boolean isValid, int lineType, String uniquePName, String uniquePValue, String pNameToModify, String modifyValue);
```

31. UpdateAutoParamValue.py

This script file is to update the parameter value of a Response Line of the Autonomous code at runtime

Format:

```python
tl1scriptinterface.updateParamValue(String autoCode, String alarmDescr, int lineType, String uniquePName, String uniquePValue, String pNameToModify, String modifyValue);
```
Configuring Agent Settings

- Overview
- Agent Settings
  - NE Name
  - Port Number
  - Manage using RMI
  - Simulate Delayed Response
  - Simulate Acknowledgment Message
  - Simulate Error Response

Overview

The TL1 Agent Simulator supports to simulate random variations in response time for TL1 commands, as well as the action of the acknowledge response. You can also configure to occasionally send a DENY error message instead of a correct response.

Agent Settings

NE Name

Specify the name of the network element. The default NE name is AdvTL1Sim.

Port

The simulated TL1 agent, starts at default port number 9099. You can specify any alternative port.

Manage using RMI

The TL1 Agent Simulator provides option to manage the TL1 agent from the remote, using Remote Method Invocation (RMI). Selecting the option 'Manage using RMI' will register the TL1 simulator with the RMI registry.

Refer to the topic Managing Agents via RMI, to learn more.

Simulate Delayed Response

Delayed Response is a function whereby an input message may be stored in a message pending buffer at the NE for final execution at some later time. The TL1 Agent Simulator supports to simulate delayed response, once in every 'n' number of responses. The time for delay will be generated in random between the specified minimum and maximum delay time.

To simulate delayed response do the following:
- Select Edit -> Agent Settings from the menu bar. The Configure Agent Settings dialog is displayed with the General Tab enabled.
- Select the check box for Simulate Delayed Response.
• Specify when the delayed response must be sent. For example, if every 5th response has to be delayed, configure "Send delayed response once in every 5 responses"
• Specify the Maximum delay Time in milliseconds
• Specify the Minimum delay Time in milliseconds.

**Example:** Delayed response is configured to be sent once in every 5 responses with maximum delay time as 10000 milliseconds and minimum delay time as 5000 milliseconds. In this case, NE will sent a delayed response, once in every 5 responses, from the time of agent startup. The response will be delayed by a minimum of 5 seconds and a maximum of 10 seconds.

**Simulate Acknowledgment Message**

An acknowledgment is a short reply from the NE indicating that an input command message is being acted upon or has been immediately rejected. Please refer to acknowledgments for more information.

The TL1 Agent Simulator supports to send acknowledgment message once in every 'n' number of responses. If an acknowledgment is already configured for a command, it will not be sent and only the acknowledgment specified in the Edit -> Agent Settings dialog will be sent.

To simulate acknowledgment message do the following:
• Select Edit -> Agent Settings from the menu bar. The Configure Agent Settings dialog is displayed with the General Tab enabled.
• Select the check box for Simulate acknowledgment Message.
• Specify when the configured acknowledgment must be sent. For example, if the configured acknowledgment has to be sent once in every 4 responses, configure "Send acknowledge message once in every 4 responses"
• Select the acknowledgment message from the list.
**Example:** acknowledgment Message "Repeat Later" is configured to be sent once in every 4 responses. In this case, NE will send the acknowledgment RL, once in every 4 responses, from the time of agent startup.

**Simulate Error Response**

The TL1 Agent Simulator allows you to configure a TL1 agent to send error responses. The intention is to test the TL1 manager's robustness in the face of receiving bad packets. The TL1 Agent Simulator supports to simulate error response once in every 'n' number of responses. To simulate error response do the following:
• Select Edit -> Agent Settings from the menu bar. The Configure Agent Settings dialog is displayed with the General Tab enabled.
• Select the check box for Simulate Error Response.
• Specify when an error response must be sent. For example, if you want to sent an error response once in every 10 responses, from the time of agent startup, specify the error response number as 10.

**Note:** The random response configurations overwrites the real Command Code response configuration.
Configuring Security in TL1 Agent

Overview

Any manager application can access the TL1 NE/NS and perform any operation on any resource which is not desirable. Confidential data may be accessed by anyone and the NE/NS can be easily misused. Without authentication on the users accessing a particular resource or using any particular command etc., the NE/NS is at risk of intrusion. Security enables authentication, that helps an administrator to prevent unauthorized access to any particular resource.

To prevent the possibility of a deliberate or accidental attack (intrusion) on a TL1 NE, User Security View is implemented in the TL1 Agent Simulator as per the specifications in GR 815 and TR 835.

User Related Security

User Related Security authorizes the establishment of a session (i.e., login) and its continuation until logoff. Before granting a session, the TL1 Agent will validate and authenticate the session requester. In addition, the NE/NS also ensures that the communication path between the NE/NS and the session requester is trusted so that no intruder can enter the channel.

User-Related Security

User-Related Security View contains system access control parameters such as user name, password, password aging, privilege, etc. NE uses this data to authenticate a user when he tries to establish a session with the NE.

The user details and various other privileges associated with the user are entered in the User related Security View.

Enabling User-Related Security

Follow the steps given below to enable User Security in TL1 Agent Simulator.

1. In the TL1 Agent Simulator, select Configure -> Security Configuration. The Configure User Security dialog appears.
2. Select the option Enable Security. Now, User Security is enabled.

User Related Security Parameters

The following are the user detail parameters that are available in the User Related Security View.

- **User Name (UID):** This parameter contains the Name of the user. User Name can be a maximum of 10 Alpha Numeric characters.
- **Password (PID):** This parameter contains the password for the corresponding user name. password will be present in encrypted format. MD5 Algorithm is used for encryption.
- **User Status (STATE):** This parameter determines if the particular user name is functioning i.e. in service (IS) or not functioning i.e. out of service (OS).
• **Password Aging Time (PAGE):** This is the expiry duration for the password. When an user logs in after the page period, he/she will be intimated with the details of the PCNN and PCND. The user has to change the password before the PCND or the PCNN period. The password aging time is in days. Default value is 60. Range is between 0 to 999 days.

• **Early warning on imminent password aging (PCND):** This is the time period in number of days after the PAGE, which is allowed for the user to change his password. The unit of PCND is in days. Default value is 7. Range is between 0 to 999 days.

• **Early warning on imminent password aging (PCNN):** This is the number of times that a user can log in after PAGE limit was reached and before which the user has to change his password. Default value 3. Range is between 0 to 999 times.

• **Time of Last Log in (LastLoginTimeStamp):** This is the last recorded Login time of a user. This time will be displayed every time a user logs in.

• **Password Obsolescence Interval (POINT):** This is the minimum time interval that is required if the User wishes to use a password that was already used. Unit of POINT is in days. Please note that POINT has not been implemented in this release. Default value 180. Range is between 0 to 999 days.

• **User ID Aging Interval (UOUT):** This parameter specifies the aging or expiry interval of a particular User Id. At the end of this interval, the UID is disabled if during this interval it has never been used to setup a session. Default value is 90. Range is between 0 to 999 days.

• **Channel Identifier List (CID):** This is the list of channels (like TCP UDP etc) through which a particular user has access. Default value is TCP.

• **User Access Privilege (UAP):** This parameter contains information about the access privileges. UAP is alphanumeric. By default the TL1 Simulator takes the UAP value of "priv1".

• **List of Ports (LSTOI):** This is the list of objects or ports (or directory numbers) that a particular user is authorized to access. The user needs to handle this parameter.

### Adding User-Related Security Parameter Values

Follow the steps given below to add the values for User Security View parameters. In this chapter we will be taking an example of adding a new user with user name "User1" and password "pswd" to the User Security View.

1. In the TL1 Agent Simulator, select **Configure -> Security Configuration.** The Configure User Security dialog appears.

2. Select the option **Enable Security.** Now, User Security is enabled.

3. Click the **Add** button below the table to add a new entry to the User Security Table. The "Security Settings" dialog box will appear, where you will have to enter the new user details.

   Please note that by default PCND is enabled in the TL1 Agent simulator. This is achieved by giving negative value to PCNN. However the user can enable PCNN just by giving any positive value less than 999.

4. After adding the new entry details click the **OK** button. The new entry for the User Security View will be added to the table.

5. Now **Start** the TL1 Agent.

6. The newly added user (User1) can now authenticate into the TL1 Agent as explained in the **Logging in and Logging out** section. The User Id to be passed in the AID block will be User1 and the Password will be pswd.

---

**Note:** The values of the parameters of an existing user can be modified from this configuration table or an existing user entry can be deleted from this table.
The new user entries will be stored in the **UserSecurity.txt** file in the encrypted format, in the configuration directory. The entries are also stored in XML format in the **SecurityFile.xml**, in the configuration directory.

**Logging In and Logging Out**

Logging in and Logging out can otherwise be called as Authentication. Follow the steps given below to login and logout of the TL1 Agent. Here we will be logging in as the administrator whose default user name and password are "root" and "public" respectively.

1. Create a Simulated TL1 Agent for the TCS AcmeMSU.tcs
2. Start the Agent at a specified port, say 9099.
4. Load the TCS file implemented in the simulated agent and tl1security.tcs in the TL1CraftInterface using the File -> Load option in the menu bar.
5. Choose Session -> Open Session option in the menu bar
6. Enter the Host Name and Port Number where the agent is running in the dialog box that appears.
7. Click Open Session to connect to the TL1 Agent running at port 9099.
8. Expand the tl1security.tcs tree and select ACT-USER command.
9. Authenticate into the TL1 Agent by using the Input message "ACT-USER::root:1::public;". After typing the message, choose Session -> Send Message in the Menu Bar.

The Agent will send the following response message which can be viewed in the Response Message Block.

** 1.0 REPT EVT SESSION
    "root:NO"
    /*NOTICE:This is a private computer system. Unauthorised access or use may lead to prosecution*/
    ;
    AdvTL1Sim 2003-10-22 16:02:55
    M 3 COMPLD
    "root:,0"
    ;

This means that you have successfully authenticated into the TL1 Agent.

10. You can **logout** of the TL1 Agent anytime using the **CANC-USER** command shown below:

    CANC-USER::root:2::;}
Re-simulating Values

- Overview
- Enabling the Option
- Example

Overview

The TL1 Agent simulator provides the option to re-simulate values, if new Command Code(s) and Autonomous codes are added to the TCS file or if existing Command codes and Autonomous codes are deleted, without modifying the existing configuration.

To re-simulate values, the existing configuration file must be loaded in the TL1 Agent Simulator only after making the necessary modification in the TCS file.

Enabling the Option

To re-simulate values, follow the steps given below.

- Load the existing configuration file of the modified Acme-MSU TCS, in TL1 Agent Simulator.
- Choose Operations -> Resimulate
- The values of the new command code will be simulated and added to the existing configuration file.

Note: The option to Re-simulate values is not supported for TL1 devices in Network Designer.

Example

You have configured large number of data to the command codes and autonomous codes defined in the Acme-MSU TCS file. The configuration is saved as sample.prp in /tl1agents/sample directory.

Now, you have added a new Command Code to this TCS file available in /tl1agents/sample directory and you want to simulate this command code alone and add it to the existing configuration file.

You can do this by loading the sample.prp file in the TL1 Agent Simulator and selecting the option Operations -> Resimulate. The values of the newly added command codes will be added to the existing configuration.
Saving TL1 Agent Configuration

- Overview
- Saving the Configuration

Overview

When you have simulated values for the TL1 commands and completed configuring the agent, you can save the agent configuration for later use.

Saving in Configuration

To save the agent configuration,

- Choose the menu option File -> Save Configuration
- The database name will be simulatordb and is non-editable. You can specify the table name and the configuration file name. The configuration will be stored as <filename>.prp along with all the other configuration details in the specified table name.
- When you save the agent configuration, two files are created: runTL1Sim.bat / sh. This file is used to start the simulated agent from command line, with the configuration file as input.

Example: If you save the configuration with a filename sample, a directory will be created by the name sample and the main file will be stored as sample.prp in the specified directory and all the configuration information will be stored in the specified table name. The main file to be loaded will be stored in the name sample.prp.
Starting and Stopping the Agent

- Overview
- Starting the Agent
- Stopping the Agent

Overview

In the topic, Simulating Default Values, you have learnt that the agent is simulated with default values as soon as you load the TCS file(s) for all the command code and the Autonomous code defined in the TCS file. Any variation on this basic simulation is created in the TL1 Agent Simulator by Configuring Values.

When you have finished configuring the agent with the required values, behavior scripts and autonomous messages, you can test the simulated agent for the required behavior, after starting the agent. The agent can be started at the specified port from the TL1 Agent Simulator or from Command line.

Starting the Agent

From UI

To start the simulated TL1 agent from the UI, do the following:

1. Click on the start icon or choose Operations --> Start from the menu bar.
2. You will get a message in the text area "Agent started running at port 9099". The port number might vary based on the port number you specify in the Settings --> Agent Settings dialog.

**Note:** For evaluation copy, the agent will respond only for the first five Command codes in the tree.

From Command Line

To start the simulated agent form command Line, refer to the topic Starting the TL1 Agent from Command Line.

Stopping the Agent

To stop the simulated agent do the following:

- Select Operations -> Stop from the menu bar or click on the stop icon of the toolbar.
- The message Agent Stopped running at port 9099 will be printed in the text area.
Starting the TL1 Agent from Command Line

- Overview
- Using Command Line TL1 Agent Simulator
- Examples

Overview

The command line TL1 Agent Simulator is purely NON-GUI and uses only character user interface for obtaining various inputs for the simulator. It can be invoked by executing the `StartTL1Agent.bat/.sh` in `<Simulator Home>/bin/cmdline` directory. Starting the agent from command line is more faster as the non-GUI agent saves on memory.

Using Command Line TL1 Agent Simulator

Execute the shell/bat file `StartTL1Agent.sh /StartTL1Agent.bat` present under the `<Simulator_Home>/bin/cmdline` directory with the following options.

Configuration File Options : Any one of the options are mandatory.

<table>
<thead>
<tr>
<th>Mandatory Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-cs &lt;Command Set File&gt;</td>
<td>TCS file to be loaded with absolute path.</td>
</tr>
<tr>
<td>-c &lt;Configuration File&gt;</td>
<td>Configuration file to be loaded with absolute path.</td>
</tr>
</tbody>
</table>

MySql options :

<table>
<thead>
<tr>
<th>Database Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-dbt &lt;tablename&gt;</td>
<td>Name of the Database Table to be created</td>
</tr>
</tbody>
</table>

**Note:** The details of the Database server name, Port number, User name, Password and the Server path is stored in `database_params.conf` file in `<Simulator_Home>/conf` directory. You can make the required database settings in this file.

Initialize Services Options : Optional. The services are enabled by default.

<table>
<thead>
<tr>
<th>Mandatory Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-at &lt;Autonomous&gt;[enabled/disabled]</td>
<td>To enable/disable the execution of Autonomous messages configured in the simulator. By default, the autonomous service will be enabled. If the service is disabled, the agent will not check for autonomous configurations.</td>
</tr>
<tr>
<td>-bt &lt;Behavior&gt;[enabled/disabled]</td>
<td>To enable/disable the execution of behavior scripts configured in the simulator. By default, the behavior service will be enabled. If the service is disabled, the agent will not check for script configurations.</td>
</tr>
</tbody>
</table>
Miscellaneous Options: Optional

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-t &lt;Target Identifier&gt;</td>
<td>Network Element Names. The names should be comma separated.</td>
</tr>
<tr>
<td>-p &lt;Server Port&gt;</td>
<td>Port at which the Simulator starts</td>
</tr>
<tr>
<td>-a &lt;Authentication&gt;</td>
<td>Authentication Status true or false</td>
</tr>
<tr>
<td>-rmi</td>
<td>Manage TL1 Agent Simulator remotely</td>
</tr>
</tbody>
</table>

Examples

(1) Loading tcs file

StartTL1Agent.bat -cs tcs\Acme-MSU.tcs -p 9002

(2) Loading config file

StartTL1Agent.bat -c info\test\tl1_conf\agent.prp -p 9002

Note: Here the config file path is given absolute to <Simulator_Home> directory.
Testing the TL1 Agent

- Overview
- Testing the Agent Using TL1CraftInterface

Overview

TL1CraftInterface is a manager-side graphical interface tool that can be used to test the real-time TL1 agents or any TL1 simulated environments. This section explains the steps involved in testing the TL1 Agent and the functionality available in the graphical interface and the ways to interact with the TL1 agents.

The TL1CraftInterface tool can be used to query a TL1 device and get responses. You can also load any TL1 message definitions conforming to the GR-831 standards created using the TL1 Message Builder for faster access. This allows the user to view and operate on available data through a TL1 agent on a managed device, thereby managing the device.

Testing the Agent Using TL1CraftInterface

Follow the steps given below to test the TL1 Agent created as per the previous chapter. It is assumed that the simulated TL1 agent is created and running on port 9099.

1. Start the TL1 Craft Interface Application using the TL1CraftInterface.bat or TL1CraftInterface.sh file from the <Simulator_Home>/bin directory. It can also be started from the launcher.
2. Load the TCS file which is configured in the simulated agent using the File -> Load option in the menu bar.
3. Choose Session -> Open Session option in the Menu Bar.
4. Enter the Host Name and Port Number of the Agent in the dialog box that appears. Default values are "localhost" and "9099" respectively.
5. Click Open Session. Now you will be connected to the Agent.
6. Load the TCS for which the simulated agent was created. If security is enabled, load the tl1security.tcs to authenticate into the TL1 agent.
7. Expand the tl1security.tcs tree and select ACT-USER command.
8. Authenticate into the TL1 Agent by sending the Input command "ACT-USER::root:1::public;" by choosing Session -> Send Message in the menu bar. Here root is the default Administrator and public is the default password.
9. The Agent will respond by sending the following Autonomous Message. This can be viewed in the Autonomous Message Block. This means that you have successfully authenticated into the TL1 Agent.

   Autonomous Message Received From :localhost
   <CR> <LF><LF> latha 2003-08-11 12:22:25<CR> <LF> ** 1 REPT EVT SESSION<CR> <LF> "latha:NO"<CR> <LF> /*NOTICE:This is a private computer system. <LF> Unauthorized access or use may lead to prosecution"/<CR> <LF>;
10. A Response Message will also be sent for the ACT-USER command. You can view the Response Message in the Response Message Block of the TL1CraftInterface.
11. After authenticating into the TL1 Agent, you can send messages. Expand the TCS tree of Acme-MSU.tcs. Choose the command code RD-MEM. Send the input message RTRV-EQPT:::1::; by choosing Session -> Send Message option in the menu bar.
Now, the Agent will send the following response message.
Message sent: RTRV-EQPT:::7:::;
AdvTL1Sim 2003-09-25 16:49:29
M 7 COMPLD
"CARD ID:1,WK:IS,IDLE"

This means that your simple TL1 Agent works!

Refer to the topic Using Test & Editor Tools -> TL1 Craft Interface, to learn more.
Using Built-in Commands

Overview

Built-in Commands are provisioning input commands common to all OS/NE interfaces. They are commonly used input commands that have already been implemented in the TL1 Agent. Some General Built-in Commands as well as Built-in Commands for Delayed Activation have already been implemented in the TL1 Agent and are available in the `tl1builtincommand.tcs` file. This file can be loaded in the TL1CraftInterface tool to query any running agent.

Built-in Commands in TL1 Agent

The TL1 Agent supports six general Built-in Commands and three Built-in Commands for Delayed Activation. You can use these commands to query the TL1 Agent from the TL1 Craft Interface.

The TL1 Agent supports the following general Built-in Commands.

- Inhibiting Autonomous Messages
- Allowing Autonomous Messages
- Clearing Autonomous Messages
- Setting Source Identification.
- Getting List Of Built-in commands
- Getting all registered Commands

The TL1 Agent supports the following Built-in Commands for Delayed Activation feature.

- Activate Delay Activation
- Cancel Delay Activation
- Retrieve Delay Activation

Using General Built-in Commands

Following are the General Built-in commands that are available in the TL1 Agent. These commands are available in the `tl1builtincommands.tcs` file.

**Note:**

Please note that since the commands in the `tl1builtincommands.tcs` file has already been implemented, it should not be loaded in the TL1 Agent Simulator for simulating values.

Before testing the examples for Built-in commands, make sure you do the following:

1. Run a simulated TL1 Agent at a specified Port.
2. Connect to the Agent at the specified port using the TL1CraftInterface.
3. Load the TCS for which the Agent was created, `tl1security.tcs`, and `tl1builtincommands.tcs` in the TL1CraftInterface.

4. Expand the `tl1security.tcs` tree and select ACT-USER command.

5. Authenticate into the TL1 Agent by sending the Input command `ACT-USER::root:1::public;`.

   Here, the user is the "root" (the administrator) and the password is "public". For more information on Authentication, refer to "Configuring Security".

After executing the above command successfully, you can start using the general Built-in commands that are available in the `tl1builtincommands.tcs` as explained.

1. Command To Inhibit Autonomous Message

   The syntax for this Built-in Command is as follows:
   
   `INH-AUTMSG:::<ctag>::;`

   When this Built in Command is executed, the agent stops sending Autonomous messages. All the Autonomous Messages generated after executing this command will be stored in the memory and will not be displayed.

   E.g.: `INH-AUTMSG:::4::;`

2. Command To Allow Autonomous Messages

   The syntax for this built-in command is as follows:
   
   `ALW-AUTMSG:::<ctag>::;`

   This Command generally follows Inhibit Autonomous Message Command. After execution of this command, the agent will start sending Autonomous Messages that were initially stopped using the Inhibit Autonomous Message Command.

   E.g.: `ALW-AUTMSG:::5::;`

3. Command To Clear Autonomous Messages

   The syntax for this built-in command is as follows:
   
   `CLR-ALARM:::<ctag>::;`

   This Command is used to clear all the Autonomous messages that are collected in the memory as a result of execution of Inhibit Autonomous Message Command. The execution of this command usually follows Inhibit Command.

   For example, you may initially want to stop receiving Autonomous Messages, for which you may execute the Inhibit Autonomous Message Command. Then after a while, you may want to view only the Autonomous Messages from a given time, i.e., you may not need the previously generated Autonomous Messages. Under such circumstances, you can execute the Clear Autonomous Message command followed by Allow Autonomous Message Command.

   E.g.: `CLR-ALARM:::6::;`

4. Command To Set the Source Identification String

   The syntax for this built-in command is as follows:
   
   `SET-SID:::<ctag>::<INPUT SOURCE ID STRING>;>्`
This Built-in Command is used for setting the source identification string. The default value "machine name" can be replaced by a string of your choice using this string. Source identifier resets output and autonomous message header. It identifies the NE from where output and autonomous messages are coming.

E.g.: SET-SID:::7::System;

The above example sets "System" as the source identification string.

5. Command To Get the List of Built-in Commands

The syntax for this built-in command is as follows:
GET-CMD-BLTIN:::<ctag>::;

This Command lists all the available Built-in Commands.
Example Request:
GET-CMD-BLTIN:::1::;

Example Response:

```
Message sent : GET-CMD-BLTIN:::1::;
AdVT1Sim 2003-09-25 17:10:50
M 1 COMPLD
"ACT-DA"
"CANC-DA"
"RTRV-DA"
"INH-AUTMSG"
"ALW-AUTMSG"
"CLR-ALARM"
"SET-SID"
"GET-CMD-BLTIN"
"GET-CMD-REG"
;
```

6. Command To Get the List of All Registered Commands.

The syntax for this built-in command is as follows:
GET-CMD-REG:::<ctag>::;

This Command Lists all the Commands that are registered with the agent. This includes the registered Built-in Commands as well as the Commands that are specific to the TCS file.

Using Delayed Activation Built-in Commands

The TL1 Agent supports the following Built-in Commands for Delayed Activation feature:
- Activate Delay Activation
- Cancel Delay Activation
- Retrieve Delay Activation

These commands are available in the `tl1builtincommands.tcs` file. Before testing the examples for Built-in commands, make sure you do the following:

1. Run the simulated TL1 Agent at a specified Port.
2. Connect to the Agent using the TL1CraftInterface.
3. Load the TCS for which the Agent was created, tl1security.tcs, and tl1builtincommands.tcs in the TL1CraftInterface.
4. Expand the tl1security.tcs tree and select ACT-USER command.
5. Authenticate into the TL1 Agent by sending the Input command "ACT-USER::root:1::public;". Here, the user is the "root" (the administrator) and the password is "public". Please refer to Testing the TL1 Agent for more information on Authentication.

After executing the above command successfully, you can start using the Built-in commands that are available in the tl1builtincommands.tcs as explained. Please note that Delay Activation of messages is session specific. Retrieving, Activating, and Canceling of Delayed Activation of Messages are also session specific.

Please note that Delay Activation of messages are session specific. Retrieving, Activating and Canceling of Delayed Activation of Messages are also session specific.

1. Command To Activate Delayed Activation

The syntax for this command is as follows:

ACT-DA::[ALL|ORDNUM]:<ctag>::;

This command executes the input message, which is stored in a message pending buffer for manual activation. This can also be called as Delayed Manual Activation.

By specifying ALL in the AID part of the command, all the pending messages in the buffer will be executed and corresponding responses will be sent immediately.

E.g.: ACT-DA::all:26::;

The above example activates all commands that were delayed activation and sends the processed response.

When ORDNUM, i.e., order number is specified in the AID part of the command, the input message for the corresponding order number alone will be executed and response will be sent.

E.g.: ACT-DA::3:26::;

The above example activates the command with Order Number as 3 that was delayed activation and sends the processed response.

2. Command To Cancel Delayed Activation

The syntax for this command is as follows:

CANC-DA::[ALL|ORDNUM]:<ctag>::;

Using this Command, the Delayed activation of the input message that was initially stored in a Message Pending buffer for execution at a later time can be canceled. This input command deletes the input message from the message pending buffer.

By specifying ALL in the AID part of the command, all the pending messages in the buffer are deleted.

E.g.: CANC-DA::all:26::;

The above example cancels the execution of all commands that were delayed activation.

When ORDNUM, i.e., order number is specified in the AID part of the command, the input message for the corresponding order number alone will be deleted from the message pending buffer.

E.g.: CANC-DA::3:26::;

The above example cancels the execution of the command with Order Number as 3 that was delayed activation.
3. Command To Retrieve Delayed Activation

The syntax for this command is as follows:

RTRV-DA::[ALL|ORDNUM]:<ctag>::TTYPE=TIME/MANUAL,TTIMEGE=yy-mm-dd-hh-mm,TTIMELE=yy-mm-dd-hh-mm;

Execution of this command lists the pending input messages in the buffer for Delayed Activation.
The AID part of the command is compulsory.

By specifying ALL in the AID part of the command, all the pending messages for Delayed Activation in the buffer get retrieved and listed. The MPB list is keyword defined.
E.g., Request Sent:
RTRV-DA::all::;

E.g., Response:

```
Response Message Received From :localhost
<CR>
<LF><LF> Source 02-01-08 17:14:07<CR>
<LF>M 16 COMPLD<CR>
<LF> "ORDERNUMBER=2:GB=18-0-0:OR_MESG=RTRV-PM"<CR>
<LF> "GB=2-1-8,18-0-0:OR_MESG=ALW-LPBK"<CR>
<LF> "ORDERNUMBER=3:GB=2-1-9:OR_MESG=RD-MEM"<CR>
<LF> "ORDERNUMBER=4:GB=2-1-9,18-30-0:OR_MESG=OPR-SYNCNSW"<CR>
<LF>;
```

The above examples retrieve the all the commands that were Delayed activation.

When ORDNUM, i.e., order number is specified in the AID part of the command, the input message for the corresponding order number alone gets retrieved from the message pending buffer and listed.
E.g., Request Sent
RTRV-DA::1:23::;

E.g., Response

```
Response Message Received From :localhost
<CR>
<LF><LF> Source 02-01-09 22:51:49<CR>
<LF>M 23 COMPLD<CR>
<LF> "ORDERNUMBER=1:OR_MESG=RD-MEM"<CR>
<LF>;
```

The above examples retrieve the Delay activation command with OrderNumber 1.

TTYPE, TTIMEGE, and TTIMELE are the Message Payload block parameters for which values can be passed. These are optional inputs. One, two, or all the three MPB parameters can be passed as input along with any one of the AIDs. This helps in retrieving the required input message quickly.

**TTYPE** is the target type, which can be Manual Delayed Activation input message or Automatic Delayed Activation input message. TTYPE can be MANUAL, or TIME, or BOTH.
TTIMEGE is the equal to or greater than time for which pending input message is to be retrieved and listed.

TTIMELE is the equal to or lesser than time for which pending input message is to be retrieved and listed.

TTIMEGE and TTIMELE are applicable only when TTYPE is specified as TIME.

E.g., Request

RTRV-DA::all:25::TTYPE=TIME, TTIMEGE=02-01-08, TTIMELE=02-01-09;

E.g., Response

<table>
<thead>
<tr>
<th>Response Message Received From :localhost</th>
</tr>
</thead>
<tbody>
<tr>
<td>06-01-07 17:21:58</td>
</tr>
<tr>
<td>M 25 COMPLD</td>
</tr>
<tr>
<td>&quot;ORDERNUMBER=2:GB=2-1-8,18-0-0:OR_MESG=ALW-</td>
</tr>
<tr>
<td>&quot;ORDERNUMBER=3:GB=2-1-9:OR_MESG=RD-MEM&quot;&lt;CR&gt;</td>
</tr>
<tr>
<td>&lt;LF&gt;;</td>
</tr>
</tbody>
</table>
Using Self-Monitoring Commands

- Overview
- Parameters That Can Be Monitored
- Using Self-Monitoring Commands

Overview

Self-Monitoring enables the TL1 Agent to monitor the Agent details, such as the number of requests received, number of responses and autonomous messages sent, etc. The Agent details are automatically updated and monitored by the TL1 Agent and their values can be retrieved, whenever you require, by sending the appropriate input messages.

Parameters That Can Be Monitored

The Self Monitoring feature allows the TL1 Agent to monitor the following agent parameter details:

- Input Request Received
- Autonomous Messages Sent
- Acknowledgment Messages Sent
- Delayed Activation Messages
- Output Responses

These parameters are automatically updated and monitored by the TL1 Agent and their values can be retrieved, whenever you require, by sending the appropriate input command messages.

Using Self-Monitoring Commands

Following are the commands that are available for Self-Monitoring of the TL1 Agent. These commands are available in the `tl1builtincommands.tcs` file.

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Please note that since the commands in the <code>tl1builtincommands.tcs</code> file has already been implemented, it should not be loaded in the TL1 Agent Simulator for simulating values.</td>
</tr>
</tbody>
</table>

Before testing the examples for Self-Monitoring commands, make sure you do the following:

1. Run the simulated TL1 Agent at a specified Port.
2. Connect to the Agent using the TL1CraftInterface.
3. Load the TCS for which the Agent was created, `tl1security.tcs`, and `tl1builtincommands.tcs` in the TL1CraftInterface.
4. Expand the `tl1security.tcs` tree and select ACT-USER command.
5. Authenticate into the TL1 Agent by sending the Input command "ACT-USER::root:1::public;". Here, the user is the "root" (the administrator) and the password is "public". Please refer to Testing the TL1 Agent for more information on Authentication.

After executing the above command successfully, you can start using the Self-Monitoring commands that are available in the `tl1builtincommands.tcs` as explained in the following sections.
1. Command for Getting Count of Input Requests Received

This Self-Monitoring command gets the count of the number of Input Request messages that were received by the Agent from the Clients.

The Syntax of the Input Command is as follows:

GET-SM-MSGIN::<CTAG>::;

Example Request: GET-SM-MSGIN:::11::;

The Response Message for the above input command appears as shown below:

```plaintext
Message sent : GET-SM-MSGIN:::11::;
AdvTL1Sim 2003-09-25 17:19:28
M 11 COMPLD
"COUNT=27"
```

2. Command for Getting the Count of Autonomous Messages Sent

This Self-Monitoring command gets the count of the number of Autonomous Messages sent by the Agent. The Syntax of the Input Command is as follows:

GET-SM-ALRMSENT::<CTAG>::;

Example Request:

GET-SM-ALRMSENT:::12::;

The Response Message for the above input command appears as shown below:

```plaintext
Message sent : GET-SM-ALRMSENT:::12::;
AdvTL1Sim 2003-09-25 17:21:01
M 12 COMPLD
"COUNT=1"
```

3. Acknowledgment Messages Sent

This Self-Monitoring command gets the count of the number of Acknowledgment Messages sent by the Agent for the request messages received from the Clients. The Syntax of the Input Command is as follows:

GET-SM-MSGACK::<CTAG>::;

Example Request: GET-SM-MSGACK:::13::;

The Response Message for the above input command appears as shown below:

```plaintext
Message sent : GET-SM-MSGACK:::13::;
AdvTL1Sim 2003-09-25 17:21:45
M 13 COMPLD
"COUNT=2"
```
4. Delayed Activation Messages

This Self-Monitoring command gets the number of pending input request messages (i.e.,
delayed activation messages) which require processing to send responses at a later stage.
In other words, this Self-Monitoring command gets number of Delay activation messages
that are to be processed and responded to the Client at a later stage.

The Syntax of the Input Command is as follows:

GET-SM-MSGDA::<CTAG>::;

Example Request: GET-SM-MSGDA:::14::;

The Response Message for the above input command appears as shown below:

Message sent : GET-SM-MSGDA:::14::;
AdvTL1Sim 2003-09-25 17:23:33
M  14 COMPLD
"COUNT=7"

5. Output Responses

This Self-Monitoring command gets the number of Output Response Messages that have
been sent by the Agent. The Syntax of the Input Command for getting the value of this
parameter is as follows:

GET-SM-MSGOUT::<CommandcompletionCode>:<CTAG>::;

Example Request: GET-SM-MSGOUT::COMPLD:15::;

The Response Message for the above input command appearas as shown below:

Message sent : GET-SM-MSGOUT::COMPLD:15::;
AdvTL1Sim 2003-09-25 17:24:11
M  15 COMPLD
"COUNT=13"

The above input command gets and displays the number of Output Response Messages that
were processed and sent successfully. The following is the syntax for the Input Command for
getting and displaying the number of Output Messages for the input messages that were
partially completed:

GET-SM-MSGOUT::PRTL:<CTAG>::;
Example Request: GET-SM-MSGOUT::PRTL:17::;

The Response Message for the above input command appears as shown below:

Message sent : GET-SM-MSGOUT::PRTL:17::;
AdvTL1Sim 2003-09-25 17:24:54
M  17 COMPLD
"COUNT=8"

The following is the syntax for the Input Command for getting and displaying the number of
Input Messages that were denied successful execution:

GET-SM-MSGOUT::DENY:<CTAG>::;
Example Request: GET-SM-MSGOUT::DENY:17::;

The Response Message for the above input command appears as shown below:

```
Message sent : GET-SM-MSGOUT::DENY:17::;
 AdvTL1Sim 2003-09-25 17:25:31
 M  17 COMPLD
    "COUNT=17"
```

The following is the syntax for the Input Command for getting and displaying the number of Output Messages that were sent for the input retrieve commands that were sent to retrieve from the NE:

GET-SM-MSGOUT::RTRV:<CTAG>::;

Example Request: GET-SM-MSGOUT::RTRV:20::;

The Response Message for the above input command appears as shown below:

```
Message sent : GET-SM-MSGOUT::RTRV:20::;
 AdvTL1Sim 2003-09-25 17:26:21
 M  20 COMPLD
    "COUNT=5"
```

The following is the syntax for the Input Command for getting and displaying the number of Output Messages that were delayed processing and queued up in the processing queue:

GET-SM-MSGOUT::DELAY:<CTAG>::;

Example Request: GET-SM-MSGOUT::DELAY:11::;

The Response Message for the above input command appears as shown below:

```
Message sent : GET-SM-MSGOUT::DELAY:13::;
 AdvTL1Sim 2003-09-25 17:27:11
 M  13 COMPLD
    "COUNT=12"
```
Network Simulator Guide

The **Network Simulator**, integrated with Simulation Toolkit, helps you to simulate a network with SNMP, TL1, CLI, TFTP, FTP, Telnet and IOS devices. It provides an intuitive UI to simulate a network in Windows, Linux, and Solaris.

The Network Simulator provides a rich set of utilities which help in setting up a simulated network in a very short time. The use of Jython-based scripting allows advanced modeling of agent behavior, trap generation, creation of error scenarios, and expression of inter-relationships between MIB variables and across agents. This enables simulation of real-time behavior of devices.

Each simulated device can support its own MIBs, TCS File, data, and IP Address. The devices support TFTP client and server protocol and FTP client protocol, to enable transfer of files between the manager and the agent. You can add your device to the network tree and then drag and drop the device to the blank network to simulate a network of devices.

The following topics are covered in this section:

- Getting Started
- Device Library
- Device Tree Configuration
- Creating and Configuring a Network
- Editing Operation of Devices
- Finding a Device in the Network
- Viewing Device and Network Info
- Modifying Device Information
- Deleting Devices in the Network
- Deleting the Network
- Saving the Network
- Starting and Stopping the Network
- Starting and Stopping Agents in the Network
- Loading and Unloading a Network
- Porting your Network
- Automated Network Simulation
- Starting Multiple Network from Command Line
Getting Started

- Starting Network Designer
- User Interface

Starting Network Designer

The Network Designer can be started in the following ways:

- From the Simulation Toolkit Launcher by invoking `<Simulator Home>/SimulationToolkit.bat/sh` file.
- By invoking `NetworkDesigner.bat` from `<Simulator_Home>/bin` directory, for Windows OS.
- By invoking `NetworkDesigner.sh` from `<Simulator_Home>/bin` directory, for Linux OS.

The Network Designer stores the simulated data in MySQL database. The MySQL database is bundled along with the product and is part of the package. By default, the following details will be stored in `database_params.conf` file in `<SimulatorHome>/conf` directory. When the simulator is started for the first time, the database parameters are read from the `database_params.conf` file and updated in `NDTStartUp.prop` file in `<SimulatorHome>/conf/settings` directory. The next time when the simulator is started, the database settings are read from `NDTStartUp.prop` file.

Refer to MySQL for more information. By default, the following details will be stored in `database_params.conf` file in `<SimulatorHome>/conf` directory.

<table>
<thead>
<tr>
<th>User Name</th>
<th>root</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password</td>
<td>(No password is configured. Leave the field blank)</td>
</tr>
<tr>
<td>Server Name</td>
<td>localhost</td>
</tr>
<tr>
<td>Server Port</td>
<td>2306</td>
</tr>
<tr>
<td>MySQL Home</td>
<td>./mysql/bin</td>
</tr>
<tr>
<td></td>
<td>(shows the MySQL bin in the producdirectory.)</td>
</tr>
</tbody>
</table>

In Linux OS, it is not necessary to specify the MySQL Home path.

User Interface

A general introduction of the UI of the Network Designer will help you to get familiar with the tool.

a. Main UI

This is the main UI of the network designer, when it is started. On the left side, is the device library, which is a palette of simulated devices. It is organized as group nodes and devices. You can add your own devices to the tree. The right side provides the network options to create, load and delete network and the options to package and install networks.
b. Network View with Device Library

When you create a new network, the devices are displayed in the network area, on the left-side as shown in the screen-shot below. By default, the View -> Network View option is selected and the network devices are displayed as device images, as shown in the screen-shot below.
c. Network View with Template tree

By default, the device library will be displayed on the left-side tree. Selecting the option View -> Tree View -> Show Template Tree, will display the template tree on the left-side. The template tree will display only those devices that were used to create the network and the number of devices that were created of that type. For example, in the screen-shot shown below, Cisco3600_1 (3) is displayed in the template tree. This means that the sample2 network has three router devices created from the template device Cisco3600.

![Template Tree Screenshot](image)

d. List view

Selecting the menu option View -> List view, displays the network as a list, as shown in the screen-shot below.

![List View Screenshot](image)
e. Topology View

Selecting the option, View - > Topology View, displays the network router and switch devices in the topology editor. You can establish inter-connections across routers, switches and other network devices and visualize the topological relationship between the devices. You can connect new devices or disconnect existing devices in the topology. For more information on creating the network topology, refer Topology Editor.
Device Tree Configuration

A network comprises of different types of devices such as printers, routers, systems and switches. Before creating a network, you have to add the required type of devices to the device tree in Network Designer or use the template devices available in the device tree. For example, to create a 10000 node network with 10 different kinds of devices, you have to first add your ten different kinds of devices in the device tree. You can then create the required network using the Network Design Wizard of the Network Designer Tool.

The following topics are covered in this section:

- Adding Devices in Device Tree
- Device Creation using SNMP WALK File
- Configuring a Device Added in Device Tree
- Deleting Devices in Device Tree
# Devices in Device Library

The Device library consists of pre-configured devices that can be used to create the network. By default, some of the most commonly used devices are available in the device tree of the network simulator. You can view the same when you start the network simulator, as shown in the screen-shot below:

![Diagram of Device Library](image)

The other devices are available in the device library. The details about the devices that are available in the device tree and device library are given below:

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Supported Protocols</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host (Linux and Solaris)</td>
<td>SNMP, TFTP, Telnet</td>
<td>Available in the Device Tree</td>
</tr>
<tr>
<td>Host (Windows)</td>
<td>SNMP</td>
<td>Available in the Device Tree</td>
</tr>
<tr>
<td>Printer - HP 4500</td>
<td>SNMP, TFTP</td>
<td>Available in the Device Tree</td>
</tr>
<tr>
<td>Cisco Routers: Cisco3600, Cisco7600</td>
<td>SNMP, TFTP, IOS</td>
<td>Available in the Device Tree</td>
</tr>
<tr>
<td>Cisco Routers: Cisco1700, Cisco12000</td>
<td>SNMP, TFTP, IOS</td>
<td>Available in device library</td>
</tr>
<tr>
<td>Cisco3750 Switch</td>
<td>SNMP, TFTP, IOS</td>
<td>Available in the Device Tree</td>
</tr>
<tr>
<td>Cisco Switches: Cisco2950Switch, Cisco6500Switch</td>
<td>SNMP, TFTP, IOS</td>
<td>Available in device library</td>
</tr>
<tr>
<td>Cisco PIX Firewall</td>
<td>SNMP, TFTP, IOS</td>
<td>Available in the Device Tree</td>
</tr>
<tr>
<td>TL1 Acme</td>
<td>TL1, TFTP</td>
<td>Available in the Device Tree</td>
</tr>
</tbody>
</table>
Adding Devices in the Device Tree

- Adding a Group Node
- Adding SNMP Devices by specifying MIB / Configuration file / SNMP WALK File
- Adding Router / Switch / Printer / Firewall devices from the Template device
- Adding TL1 devices by specifying TCS / Configuration file
- Adding Device Images

Adding a Group Node

When you add a large number of devices to the device tree, it is necessary to organize them so that they do not clutter the device tree. For example, if you are going to simulate printers, you can name the group node as printer and add all the related devices under the Group Node printer.

To add a Group to the devices tree, do the following:

1. Right-click on Devices Node in the component tree and choose the Menu item Add a Group. The Add a Group dialog box is displayed.
2. Specify the name of the Group in the Specify Name combo-box. You can provide any desired group name. Example: Printer
3. Click OK to add this as a Group in the device tree.

Adding SNMP Devices by specifying MIB / Configuration file / SNMP WALK File

1. Right-click a group node and select Add Device. The DeviceAdditionWizard is displayed, wherein you have to specify the following details:
   - Device name: You can provide any device name.
   - Device Image: It is preferred that you choose a device image to differentiate between the devices in a network. A collection of images are available in <Simulator_Home>/devices/images directory. You can also provide your own image for the new device. Refer to Adding Device Images.
   - Device Type: Select the device type as SNMP Device from the list displayed.
2. You can simulate the SNMP device by specifying the MIB file or the configuration file or the SNMP WALK file.
   - **MIB File**
     - To create a device using a MIB/Configuration file, select the MIB/Config tab.
     - Select MIB Files option and click Load button. This brings up the MIBs selection dialog listing all the MIBs.
     - Select a MIB or MIBs and click the Open button. This lists the MIB file you have selected. To select multiple MIBs, press the Shift key and select the MIB file name that you want.
   - **Configuration File**
     - To create a device using a MIB/Configuration file, select the MIB/Config tab.
     - Select the Pre-Configuration Files option and click Load to select the configuration file. The main configuration file created using SNMP Agent Simulator (filename_main.xml) must be selected.
     - After selecting the configuration file, click Open button. This loads the configuration file you have selected.
SNMP WALK File
- To create a device using SNMP WALK file, select the MIB WALK file tab. Refer to the Help Doc section, Device Creation using SNMP WALK File for more information.

3. To enable TFTP service, select TFTP and configure the TFTP parameters.
   - Default root directory: The default TFTP root directory is .networks/<networkname>/tftp/<templatedevicename> directory.
   - Root directory : Configure the TFTP root directory as per your requirement.
   - Upload script/Download script: You can configure Jython script files to be executed when a file is uploaded to the server or downloaded from the server. This facility can be used to inform the manager about a file transfer, by way of sending traps etc. To configure scripts, browse and select the script (.py file).

4. To enable Telnet Service, select Telnet. Configure the user name and password to login to Telnet. The default user name is root and password is public. The telnet commands supported by the SunSolaris device will be supported.

5. Click Create to create and add the device in the device tree.

Note:
1. The SNMP device will support SNMP, TFTP (if TFTP service is enabled) and Telnet  (if Telnet is enabled).
2. The configuration will be copied in <Simulator_Home>/devices/db/device name/ directory. The main file will be in the name ./snmp/<device name>_main.xml.
3. The device database files will be created with the specified device name in <Simulator_Home>/mysql/data/device db directory.

Adding a Router, Switch, Printer, Firewall Device
To add an router, switch, printer or firewall device, do the following :

1. Right-click a group node and select Add Device. The DeviceAdditionWizard is displayed, wherein you have to specify the following details:
   - Device name : You can provide any device name. The device name cannot have a space in between.
   - Device Image : It is preferred that you choose a device image to differentiate between the devices in a network. A collection of images are available in <Simulator_Home>/devices/images directory. You can also provide your own image for the new device. Refer to Adding Device Images.
   - Device Type : Select the device type as Router/Switch/Printer/Firewall from the list displayed.
   - Select the template device from the list of predefined configurations. The template router/switch and firewall devices supports SNMP and IOS commands. The template printer device supports only SNMP.

2. To enable TFTP Service, select TFTP and configure the TFTP parameters.
   - Default root directory: The default TFTP root directory is .networks/<networkname>/tftp/<templatedevicename> directory.
   - Root directory : Configure the TFTP root directory as per your requirement.
   - Upload script/Download script: You can configure Jython script files to be executed when a file is uploaded to the server or downloaded from the server. This facility can be used to inform the manager about a file transfer, by way of sending traps etc. To configure scripts, browse and select the script (.py file).

3. Click Create to create and add the device in the device tree.
Note:

1. The router/switch device will support SNMP, CLI and TFTP (if TFTP service is enabled).
2. The Printer device will support SNMP, TFTP (if TFTP service is enabled) and Telnet (if Telnet is enabled).
3. The configuration will be copied in `<SimulatorHome>/devices/db/device name/directory`. The main file will be in the name `.snmp/<device name>_main.xml`.
4. The device database files will be created with the specified device name in `<Simulator_Home>/mysql/data/devicedb` directory.

Adding TL1 devices by specifying TCS / Configuration file

1. Right-click a group node and select Add Device. The DeviceAdditionWizard is displayed, wherein you have to specify the following details:
   - Device name: You can provide any device name.
   - Device Image: It is preferred that you choose a device image to differentiate between the devices in a network. A collection of images are available in `<Simulator_Home>/devices/images` directory. You can also provide your own image for the new device. Refer to Adding Device Images.
   - Device Type: Select the device type as **TL1 Device** from the list displayed.

2. You can simulate the TL1 device by specifying the TCS file or the configuration file.
   - **TCS File**
     - Select TCS Files option and click Load button. This brings up the TCS selection dialog listing all the TCS files.
     - Select a TCS file and click the Open button.
   - **Pre Configuration File**
     - Select the Pre-Configuration Files option and click Load to select the configuration file of type `<filename>.prp`. The configuration file created using TL1 Agent Simulator or Network Simulator can be used.
     - After selecting the configuration file, click Open button. This lists the configuration file you have selected.

3. To enable TFTP Service, select **TFTP** and configure the TFTP parameters.
   - Default root directory: The default TFTP root directory is `<SimulatorHome>/devices/db/<devicename>/tftp` directory.
   - Root directory: Configure the TFTP root directory as per your requirement.
   - Upload script/Download script: You can configure Jython script files to be executed when a file is uploaded to the server or downloaded from the server. This facility can be used to inform the manager about a file transfer, by way of sending traps etc. To configure scripts, browse and select the script (.py file).

4. Click Create to create and add the TL1 device in the device tree.

Note:

1. The TL1 device will support TL1 and TFPT (if TFTP service is enabled).
2. The configuration main file will be created with the specified device name in `<Simulator_Home>/devices/db/device name/directory`. The main file will be in the name `<device name>_main.xml` for SNMP and `<device name>.prp` for TL1.
3. The device database files will be created with the specified device name in `<Simulator_Home>/mysql/data/devicedb` directory.
Adding Device Images

The rules to add a new device image is given below:

- The image to be displayed in the device tree must be of size 23 * 23 px. Any name can be specified for this image.

- The image to be displayed in the network area must be of size 36 * 72 px. The name of this image must end with <name>_32.png. If this image is not provided, the image with size 23*23 px is taken.

- The image must be in .png format.
SNMP Device Creation using SNMP WALK File

- Overview
- Creating the WALK File From Command Line
- SNMP Device Creation From UI using WALK file
- SNMP Device Creation From Command Line using WALK file

Overview

If you cannot record a device (e.g., if you don’t have one available, or it is under development), then you can create a basic simulation of a device by specifying the SNMP WALK file as input to the simulator. The walk file must be in the format as supported by the simulator.

A new device can be added in the device tree of the Network Designer, by specifying the 'walk file' through the GUI or from command line with other necessary arguments. The simulator also provides a command line utility to create a WALK file.

Creating the WALK File

The command line utility `WalkFileCreator.bat/sh` enables you to create a walk file by specifying the hostname, community, port number of the agent and the name of the walk file. To use this utility,

1. Run the `WalkFileCreator.bat/sh` file in `<SimulatorHome>/bin/cmdline` directory. The following mandatory options must be specified.

   - `-h <hostname>`: The hostname of the system where the agent is running.
   - `-c <community>`: The community name of the agent.
   - `-p <port>`: The port number where the agent is running.
   - `-f <walk file name>`: The walk file to be created.

Example: `WalkFileCreator -h 201.202.203.29 -c public -p 8001 -f walkfile1`

SNMP Device Creation from UI using WALK File

The Network Simulator provides option to create a device from a WALK file. The WALK file must be in the format supported by the simulator.

1. Right-click a group node and select Add Device. The DeviceAdditionWizard is displayed, wherein you have to specify the following details:
   - Device name: You can provide any device name.
   - Device Image: It is preferred that you choose a device image to differentiate between the devices in a network. A collection of images are available in `<Simulator_Home>/devices/images` directory. You can also provide your own image for the new device. Refer to Adding Device Images.
   - Device Type: Select the device type as SNMP Device from the list displayed.
2. To simulate the SNMP device using SNMP WALK file, select **MIB WALK File** tab.
   - **WALK File**: Select the WALK file from the required directory.
   - **MIBs directory**: Browse and select the MIBs directory where the MIBs related to the device are placed. If the MIBs directory is not specified, then the "<simulator home>/mibs" directory will be taken by default.
   - **WALK File format**: Specify the format of the walk file. The formats currently supported will be listed here.
     - NumericOid=Value, for example .1.3.6.1.2.1.1.1.0=sysdesc
     - StringOid=Value, for example .iso.org.dod.internet.mgmt.mib-2.system.sysDescr=sysdesc
     - NodeName=Value, for example sysDescr.0=Hardware
     - NodeName:->Value, for example sysDescr.0:-->Hardware
     - NodeName Type Value, for example .1.3.6.1.2.1.1.1.0 DisplayString Hardware
     - NodeName Value, for example sysDescr.0 Hardware
     - NumericOIDValue, for example .1.3.6.1.2.1.1.1.0 sysdesc

   **Note**: The format "NumericOIDValue" is supported if a UCD WALK is made with option -O tenq meaning,
   - q: Quick print for easier parsing.
   - n: Print oids numerically.
   - e: Print enums numerically.
   - t: Print timeticks unparsed as numeric integers.)

4. Click **Create** to add the device to the device tree.

**Device Creation from Command Line using Walk File**

The **WalkFileConfigCreator.bat/sh** is a utility to create an SNMP configuration from command line by specifying a WALK file as input. The WALK file must be in the format supported by the simulator. The configuration can then be used in the Network Designer UI to create a device in the device tree.

1. Run the **WalkFileConfigCreator.bat/sh** file in `<SimulatorHome>/bin/cmdline` directory. The following mandatory options must be specified.
   - `w <Walk File>`: The name of the Walk file.
   - `c <ConfigFile>`: The name of the configuration file to be created. The configuration will be created inside `<SimulatorHome>/snmpagents` directory
   - `m <mibs directory>`: The directory name, where all the related MIBs are present. If the MIBs directory is not specified, then the "<simulator home>/mibs" directory will be taken by default.
   - `f <walk file format(1 to 6)>` : Specify the supported walk file format number. The supported formats and their corresponding numbers are given below.

<table>
<thead>
<tr>
<th>Walk file format</th>
<th>Walk file format number</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1.3.6.1.2.1=value</td>
<td>1</td>
</tr>
<tr>
<td>.iso.org.dod=value</td>
<td>2</td>
</tr>
<tr>
<td>sysName=value</td>
<td>3</td>
</tr>
<tr>
<td>sysName.0:--&gt;value</td>
<td>4</td>
</tr>
</tbody>
</table>
### Walk file format

<table>
<thead>
<tr>
<th>Walk file format</th>
<th>Walk file format number</th>
</tr>
</thead>
<tbody>
<tr>
<td>.1.3.6 DisplayString value</td>
<td>5</td>
</tr>
<tr>
<td>sysName value</td>
<td>6</td>
</tr>
<tr>
<td>.1.3.6.1.2.1 value</td>
<td>7 (This format is supported if a UCD WALK is made with option <code>-O tenq</code>)</td>
</tr>
</tbody>
</table>

- `tn <TableName>`: The name of the Table to be created.
- `dn <DatabaseName>`: The name of the Database to be created. The default database name is simulatordb.

#### Example

```bash
sh WalkFileConfigCreator.sh -w advent2/test/walk_str_oid_value -m /advent2/ADVENTNET_PRODUCTS//Simulator5/mibs/mibs_dir/ -c host4_agent -f 1
```

2. After creating the SNMP Agent configuration, you can add a device in the device tree of the Network Designer by specifying the configuration files.

#### Note:

While creating the device using Walk File, the format has to be uniform throughout the file. If the walk file is created with mixed formats as given below, it will hinder the creation of the device.

```plaintext
snmpOutGetResponses.0-->3169
snmpOutTraps.0-->442
snmpEnableAuthenTraps.0-->enabled(1)
.1.3.6.1.2.1.16.1.1.1.1.1-->1
.1.3.6.1.2.1.16.1.1.1.2.1-->iso.org.dod.internet.mgmt.mib-2.16.1.1.2
.1.3.6.1.2.1.16.1.1.1.3.1-->8
```
Configuring a Device Added in the Device Tree

- Overview
- Configuring an SNMP Device
- Configuring a TL1 Device

Overview

When a device is added to the device tree, the device is simulated with default values in the MIB/TCS file or values in the configuration file. You can also create variations of this basic simulation by configuring the devices in the device tree.

An SNMP device can be simulated by recording values from real SNMP agents, configuring values for scalar and table nodes in the MIB, configuring traps and configuring real-time behavior of devices. A TL1 device can be simulated, by configuring values for the Input Command Messages, Output Response Messages, Acknowledgment message, Autonomous messages, Echo messages and real-time behavior of devices.

Note: Configuring a device in the device tree will not reflect on the configurations of the network devices, that was already created.

Configuring an SNMP Device

Recording from Real Agent

To configure values of a real agent in your MIB, you can record values from the real agent and save this configuration. This configuration file, which is a simulation of a real agent, can be used in the network.

To enable the recording option,

1. Select the menu bar option Configure -> Record Real Agent Values or right-click on the device and select Record Values. The Record from Real Agent dialog is displayed.

2. Refer to the topic, Simulating an SNMP Agent -> Recording from Real Agents, to learn about the steps involved in recording.

Configuring SNMP Agent Values

You can configure scalar and table values for the MIB, configure simulation types, and perform table operations. To configure values,

1. Right-click on the device and select SNMP Agent Values or select Configure -> SNMP Agent Values. The Configure SNMP Agent Values dialog is displayed with the MIBs loaded in the MIB tree. You can load a new MIB using the Load MIBs button or unload an existing MIB using the Unload MIBs button. You can use the Simulate option to simulate the loaded MIBs.

2. Refer to the topic, Simulating an SNMP Agent -> Configuring Values for the MIB to learn about the steps involved in configuring MIB values and simulation types.

3. Refer to the topic, Simulating an SNMP Agent -> Populating the SNMP Table, to learn about the table operations supported.
Configuring Traps

You can configure the SNMP agents in the network so that the agent will send traps when it satisfies the specified condition. The following traps can be configured.

- Request-based traps
- Threshold-based traps
- Timer-based traps

To configure traps,
1. Right-click on the device and select **SNMP Traps** or select **Configure -> SNMP Traps**. The Trap Configuration dialog is invoked.
2. Refer to the topic, *Simulating an SNMP Agent -> Simulating Traps*, to learn about the steps involved in trap configuration.

Configuring Informs

You can configure the SNMP agents in the network so that the agent will send informs when it satisfies the specified condition. The following informs can be configured.

- Request-based informs
- Threshold-based informs
- Timer-based informs

To configure informs,
1. Right-click on the device and select **SNMP -> Informs** or select **Configure -> SNMP Informs**. The Inform Configuration dialog is invoked.
2. Refer to the topic, *Simulating an SNMP Agent -> Simulating Informs*, to learn about the steps involved in inform configuration.

Configuring Behavior Scripts

You can configure Real-Time Behavior of SNMP devices in the network, using Jython scripts. To configure behavior scripts,
1. Right-click on the device and select **SNMP Behavior** or select **Configure -> SNMP Behavior**. The Configure Behavior dialog is invoked.
2. Refer to the topic, *Simulating a Network -> Configuring Behavior in SNMP Devices*, to learn about the steps involved in behavior configuration.

Configuring a TL1 Device

When a TCS is loaded in the TL1 Agent Simulator, a default value will be assigned for each command code and Autonomous code defined in the TCS file. The default values are generated, based on the parameter names and syntax defined in the TCS file.

Configuring TL1 Agent Values

You can configure the parameter values of Input Command Messages, Acknowledgment messages, Output Response Messages and Autonomous Messages defined in the TCS file. To configure TL1 agent values,
1. Right-click on the device and select **TL1 Agent Values** or select **Configure -> TL1 Agent Values**. The Configure TL1 Agent Values dialog is displayed.
2. Refer to the following topics, to learn about the steps involved in configuring TL1 agent values.
   - *Simulating a TL1 Agent -> Configuring Values -> Input Command Message*
Configuring Request-based Autonomous Message

On configuring the Autonomous messages for Autonomous code you can configure to send the messages as Request-based or Timer-based Autonomous messages. To configure Request-based Autonomous messages,

1. Right-click on the device and select the option Request Autonomous or select Configure -> Request Autonomous. The Request Based Autonomous Message dialog is displayed.

2. Refer to the topic, Simulating a TL1 Agent -> Configuring Values -> Autonomous Message to learn about the steps involved in configuring Request-based Autonomous messages.

Configuring Timer-based Autonomous Message

On configuring the Autonomous messages for Autonomous code you can configure to send the messages as Request-based or Timer-based Autonomous messages. To configure Timer-based Autonomous messages,

1. Right-click on the device and select the option Timer Autonomous or select Configure - > Timer Autonomous. The Timer Based Autonomous Message dialog is displayed.

2. Refer to the topic, Simulating a TL1 Agent -> Configuring Values -> Autonomous Message to learn about the steps involved in configuring Timer-based Autonomous messages.

Configuring Echo Request

The TL1 simulator supports to configure echo messages. If Echo Message is configured for a command, the NE responds with the request, instead of the actual output response message. To configure Echo Messages,

1. Right-click on the device and select Echo Request or select Configure -> Echo Request. The configure Echo request dialog is displayed.

2. Refer to the topic, Simulating a TL1 Agent -> Configuring Values -> Echo Message to learn about the steps involved in configuring Echo Messages.

Configuring Behavior

You can configure Real-Time Behavior of TL1 devices in the network, using Jython scripts. To configure behavior scripts,

1. Right-click on the device and select TL1 Behavior or select Configure -> TL1 Behavior. The Configure Behavior dialog is invoked.

2. Refer to the topic, Simulating a TL1 Agent -> Configuring Values - >Configuring Behavior, to learn about the steps involved in behavior configuration.
Deleting Devices in Device Tree

- Deleting a Device From the Device Tree
- Deleting a Group Node From the Device Tree

Deleting a Device from the Device Tree
To delete a device added in the device tree
- Right-click on a device in the device tree and choose the menu item Delete Device. The Confirm Deletion dialog is displayed asking for the confirmation.
- Click Yes to delete the selected device from the device tree.

Deleting a Group Node from the Device Tree
To delete a group node added in the device tree
- Right-click on a group node in the device tree and choose the menu item Delete Group. The Confirm Deletion dialog is displayed asking for the confirmation.
- Click Yes to delete the group node and all the child nodes added to the group node.
Creating and Configuring a Network

When you have added devices to the device tree, you can create a new network and add devices to the network. This section deals with the following topics:

- Creating a New Network
- Template Devices and Bulk Configurations
- Network View, List View and Topology View
- Network Directory Structure
- Configuring Network Devices
- Configuring Router/Switch Devices
- Configuring Behavior in SNMP Devices
- Configuring Behavior in TL1 Devices
- Configuring SNMPv3 Security Settings
- Configuring Runtime Settings
- Adding Devices to the Network
- Bulk Addition of Scripts
- Bulk Modification of IP, Port & SNMP OIDs
- Customizing the SNMP PDU
- Configuring IP Addresses
- Topology Editor
Creating a New Network

- Overview
- Creating a Network with Same Device Type
- Creating a Network with Multiple Device Types
- Specifying Unique OID Values

Overview

The Network Designer allows you to create a network in two ways. You can create a network by adding multiple devices of the same type or create a network by adding multiple devices of different types.

Creating a Network with Same Device Type

The Network Designer provides an option to create a network with multiple devices of same type. The devices will be added by incrementing the IP address. Thus, large networks can be created easily and in a very short time. The Network Design Wizard can be used to add multiple devices of the same type.

1. Select Network -> New option from the menu bar or New Network icon from the toolbar. The Create a New Network dialog appears.
2. Click on the first icon to create multiple devices of the same type.
3. Specify the network name and click OK button. The network name must not contain hyphen or blank space.
4. The new network will be created in <Simulator_Home>/networks/<network name> directory.
5. The Network Design wizard appears as shown in the screen-shot given below. You have to specify the network device details in this wizard as explained below:
- **Device Type**: Browse to select the device type from the displayed tree. The devices in the network will be created with the specified device type.

- **Unique Name**: This field is not editable. The device name will be displayed as its unique name. The files and folders which contain the device configuration will be generated with this unique name. The unique name will be updated as the `sysName` in the `system group` of RFC1213-MIB for SNMP devices.

- **Increment IP Address**: Select this option to create the devices by incrementing the specified IP address. The port number will be the same for all the devices. For example, if you want to create 500 devices by incrementing the IP address 190.191.1.1, the simulator will create the 500 devices with IP addresses ranging from 190.191.1.1 to 190.191.1.254 and 190.191.2.1 to 190.191.2.246.

- **IP Address**: If the option to increment the IP address is selected, specify the starting IP address of the network. If you have already created a network, for example, with IP address 200.201.202.1 to 200.201.202.10, the next IP address (i.e., 200.201.202.11) will be displayed while creating a new network. The IP address assigned to the devices will be updated in `ipAddrTable` of RFC1213-MIB for SNMP devices.

- **IP Address list from file**: Select this option to create the devices with the IP addresses specified in a text file. The text file will contain the IP addresses in the following format:

```
200.201.202.3
200.201.202.4
....
```

- **Select a file**: If the IP address is to be read from a text file, browse and select the file. For example: `ipaddress.txt`. The IP address assigned to the devices will be updated in `ipAddrTable` of RFC1213-MIB for SNMP devices.

- **Total Devices**: Specify the total number of devices to be created by incrementing the IP address or port number.

- **SNMP Port**: If the device supports SNMP, the default SNMP port 8001 is displayed. Specify the SNMP port at which the device will get started.

- **TFTP Port**: If the device supports TFTP, the default TFTP port 6969 is displayed. Specify the TFTP port at which the device will get started.

- **TL1 Port**: If the device supports TL1, the default TL1 port 8004 is displayed. Specify the TL1 port at which the device will get started.

- **CLI Port**: If the device supports CLI, the default CLI port 2323 is displayed. Specify the CLI port at which the device will get started.

- **Telnet Port**: If the device supports Telnet, the default Telnet port 2424 is displayed. Specify the Telnet port at which the device will get started.

6. Specify the **unique OID values** to be configured in the network devices.

7. Click **Create** to create the specified number of devices in the network. The **Creating Devices dialog** will pop up with the following message: "Creating devices for the network :: <Network name>". You will find the new network added to the network area. Now, the menu option, **View -> Tree View -> Show Template Tree**, is enabled. Selecting this option displays the Template Tree on the left-side. For more information refer the next topic, **Template Devices and Bulk Configurations**.

The Network Design wizard can also be invoked by selecting **Tools -> Network Design Wizard**, to add multiple devices of the same type to an existing network.
Creating a Network with Multiple Device Types

You can add multiple devices of different types using the Random Device Addition wizard. The devices will be added by incrementing the specified IP address. Thus, large networks with different device types can be created easily and in a very short time.

To add multiple devices of different types,

1. Select Network -> New option from the menu bar or New Network icon from the toolbar. The Create a New Network dialog appears.
2. Click on the second image to create a random network using the wizard.
3. Specify the network name and click OK button. The network name must not contain hyphen or blank space.
4. The new network will be created in <Simulator_Home>/networks/<network name> directory.
5. The "Random Device Creation" wizard appears, wherein you have to specify the details about the devices to be created, as shown in the screen-shot below. You can also load an existing configuration in this wizard, using the Load option.

<table>
<thead>
<tr>
<th>Added devices</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>SunSolaris</td>
<td>2</td>
</tr>
<tr>
<td>Windows 2000</td>
<td>3</td>
</tr>
<tr>
<td>Cisco 3600</td>
<td>1</td>
</tr>
<tr>
<td>HP 4500 Series</td>
<td>1</td>
</tr>
</tbody>
</table>

6. Click Add to select the device type and the number of devices in each type.
7. Specify the unique OID values to be configured in the network devices.
8. Specify the following options:
   o **SNMP Port**: If the device supports SNMP, specify the SNMP port at which the device will get started.
   o **TFTP Port**: If the device supports TFTP, specify the TFTP port at which the device will get started.
o **TL1 Port**: If the device supports TL1, specify the TL1 port at which the device will get started.

o **CLI Port**: If the device supports CL1, specify the CLI port at which the device will get started.

o **Telnet Port**: If the device supports Telnet, specify the Telnet port at which the device will get started.

o **Starting IP Address**: Specify the Starting IP Address. The devices will be added by incrementing the specified IP address. If you have already created a network, for example, with ipaddress 200.201.202.1 to 200.201.202.10, the next ipaddress (i.e) 200.201.202.11 will be displayed, while creating a new network. The IP address assigned to the devices will be updated in `ipAddrTable` of `RFC1213-MIB` for SNMP devices.

o **Total devices**: Specify the total number of devices to be created in the above order of the device type and the number of devices of each type.

9. You can save the above details in a file and reload it later. To save the above configuration, select **Save** option. Specify the **Filename**, to save the configuration. The configuration will be saved in the specified filename in XML format, in `<SimulatorHome>/randomnets` directory.

10. Click **OK** to create the specified number of devices in the network. You will find the new devices added to the Network area. Now, the menu option, View -> Tree View -> Show Template Tree, is enabled. Selecting this option displays the Template Tree on the left-side. For more information refer the next topic, Template Devices and Bulk Configurations.

### To load an existing random device configuration,

1. Select the **Load** option.

2. The list of existing random device configuration files will be displayed.

3. Select the configuration to be loaded, and click **OK**. The configuration will be loaded in the Random Device Creation wizard.

The random device addition wizard can also be invoked by selecting Tools -> Random Device Addition, to add multiple devices of different types to an existing network.

---

**Note:**

When the network is created,

1. The Unique Name of the device will be updated as the `sysName` in the system group of `RFC1213-MIB` for SNMP devices.

2. The specified IP address will be updated in `ipAddrTable` of `RFC1213-MIB` for SNMP devices.

3. A unique **mac address** will be assigned to each device and updated in `ifTable -> ifPhysAddress`.

4. The configured unique values will be assigned to the selected OIDs for all the devices in the network. For more information, refer Configuring Unique Values.

5. The menu option, View -> Tree View -> Show Template Tree, is enabled. Selecting this option displays the Template Tree on the left-side. For more information refer, Template Devices.

6. You can also **drag and drop** devices to a network created with the Network Design Wizard.

7. The Network Simulator dynamically configures IP addresses when starting each agent instance in the network. Refer to the topic Configuring IP Addresses for more information.

8. No two devices in the same network will have the same IP address and port number combination. The SNMP, TL1, TFTP and CLI port numbers should not clash.
Configuring Unique Values

While creating a new network, the network simulator provides the option to configure unique OID values for the simulated devices. To specify unique OID values,

- Select the option "Specify unique OIDs" in the Network Design Wizard or the Random Device Addition Wizard. The unique OID dialog is displayed. This dialog displays the MIB tree on the left-side with the sysName selected as the unique OID. By default, the sysName OID will be assigned with the unique name macro. (i.e) the unique name of the device will be updated as the sysName in the system group of RFC1213-MIB for SNMP devices.

- Select the required OID and click Add to configure the unique value. The "Enter the values" dialog will be displayed as shown in the screen-shot given below:

- For table OIDs, select the instance which has to be assigned with unique values.

- The following types of unique value simulation is supported;
  - **Increment Values**: If increment values is selected, the initial value to be incremented must be specified. This initial value will be incremented for the selected OID, for all the devices in the network.
  
  - **Macro**: The macros supported are IPaddress and Unique name.
    - If the ipaddress macro is selected, the ipaddress of the simulated devices will be assigned to the selected OID.
    - If the unique name macro is selected, the unique name of the simulated devices will be assigned to the selected OID.
  
  - **Random**: If Random is selected, the random values must be specified. The devices in the network will be assigned with random values for the selected OID.
  
  - **Constant**: If Constant is selected, the constant value must be specified. The devices in the network will be assigned with this constant value for the selected OID, for all the devices in the network.
Template Devices and Bulk Configurations

- Template Tree View
- Template Device Information
- Bulk Configurations using Template Devices
  - Configuring SNMP agent values
  - Configuring Traps messages
  - Configuring SNMP Informs
  - Configuring SNMP Parameters
  - Configuring TFTP Parameters
  - Configuring CLI Parameters
  - Configuring Telnet Parameters

Template Tree View

On creating a network, the View -> Tree View -> Show Template Tree View will be enabled. By default, the device library will be listed on the left-side tree. The template tree will list the template devices that were used to create the network and the number of devices of each type. For example, `Cisco3600_1(3)` means that the loaded network consists of 3 devices created using the device Cisco3600. If we add another set of devices of the same type to this network, for example, 4 devices of type Cisco3600, the template tree will display it as `Cisco3600_2 (4)`. Refer the screen-shots of the Template tree view, given below:
Template Device Information

The template device right-click menu option **Template Info** provides details about the devices created using the template. The ip address of the devices and the protocol specific properties will be displayed as shown in the screen-shot given below. You can modify the protocol specific properties. This will be reflected in all the selected devices. By default, all the devices are selected.

**Note:** When a template device is double-clicked, the devices created of this type are highlighted in the network area.
Bulk configurations using template devices

Configuration changes made to the template devices will be reflected on all the devices in the network which was created using this template. The screen-shot given below shows the right-click menu options that are available on the SNMP template devices. Each of these options is explained below:

![SNMP template device options]

**Configuring SNMP Agent Values**

You can configure scalar and table values, configure simulation types, and perform table operations on the template device. This will be reflected in all the devices in the network created using this template. To configure values,

Right-click on the template device and select SNMP -> Agent Values. The Configure SNMP Agent Values dialog is displayed with the MIBs loaded in the MIB tree. You can load a new MIB using the Load MIBs button or unload an existing MIB using the Unload MIBs button.

Refer to the topic, Simulating an SNMP Agent -> Configuring Values for the MIB to learn about the steps involved in configuring MIB values and simulation types.

Refer to the topic, Simulating an SNMP Agent -> Populating the SNMP Table, to learn about the table operations supported.

**Configuring Traps**

You can configure SNMP traps on the template device. This will be reflected in all the devices in the network created using this template. The following traps can be configured.

- Request-based traps
- Threshold-based traps
- Timer-based traps
To configure traps,
1. Right-click on the template device and select **SNMP -> Trap**. The **Trap Configuration** dialog is invoked.
2. Refer to the topic, *Simulating an SNMP Agent -> Simulating Traps*, to learn about the steps involved in trap configuration.

### Configuring Informs

You can configure SNMP informs on the template device. This will be reflected in all the devices in the network created using this template. The following inform messages can be configured.

- Request-based informs
- Threshold-based informs
- Timer-based informs

To configure informs,
1. Right-click on the template device and select **SNMP -> Informs**. The **Inform Configuration** dialog is invoked.
2. Refer to the topic, *Simulating an SNMP Agent -> Simulating Informs*, to learn about the steps involved in inform configuration.

### Configuring Behavior Scripts

You can configure Real-Time Behavior of SNMP devices using Jython scripts. This will be reflected in all the devices in the network created using this template. To configure behavior scripts,
1. Right-click on the template device and select **SNMP -> Behavior**. The **Configure Behavior** dialog is invoked.
2. Refer to the topic, *Simulating a Network -> Configuring Behavior in SNMP Devices*, to learn about the steps involved in behavior configuration.

### Configuring SNMP Parameters

This option allows you to modify the SNMP read and write communities. This will be reflected in all the devices in the network created using this template.

1. Right-click on the template device and select **Template Info**. The **Template Info** dialog is displayed.
2. Select the SNMP Tab and specify the new **Read Community** and **Write Community** to be assigned to the devices.
3. Click **OK** to update the changes.

### Configuring TFTP Parameters

This option allows you to modify the TFTP properties for the template based devices in the network. You can modify the TFTP root directory and configure a script to be executed when a file is uploaded/downloaded.

1. Right-click on the template device and select **Template Info**. The **Template Info** dialog is displayed.
2. Select the TFTP Tab. The following TFTP parameters can be configured:

   **a. Execute Script on File Upload/Download**
   - To execute a script when a file is downloaded from the TFTP server in the simulator, browse and select the script (.py file) in the **Execute script on Download** combo box. This script file will be copied in
To execute a script when a file is uploaded to the TFTP server in the simulator, browse and select the script (.py file) in the **Execute script on Upload** combo-box. This script file will be copied in \./networks/\<networkname>/tftp/\<templatedevicename> directory in the name TftpUploadScript.py.

### b. Root Directory
- Configure the TFTP root directory for the template device. This will set the specified directory as the root directory for the template based devices in the network. All TFTP file transfers will be with respect to this root directory.

3. Click **OK** to apply the changes to the specified range of TFTP servers.

### Configuring CLI command scripts

If the template device supports CLI, the CLI -> Command script right-click menu option will be displayed, in addition to SNMP. This option enables you to view and edit the script files configured for the IOS commands, as per your requirement. It gives a list of all the IOS commands supported for the specific device. Selecting a command will display the script file configured for the command.

1. Right-click on the template device and select **CLI -> Command script**. The CLI command editor is displayed.
2. Refer to the topic, **IOS Simulator Guide -> Editing Scripts**, to learn more about editing scripts.

### Configuring CLI Parameters

This option allows you to modify the user name and password to login to the IOS simulation.

1. Right-click on the template device and select **Template Info**. The Template Info dialog is displayed.
2. Select the CLI Tab. Specify the new **User name** and **Password** to login to the IOS simulation.
3. Specify the new **Enable Mode Password** to get into exec mode.
4. Click **OK** to update the changes on the template based devices in the network.

### Configuring Telnet command scripts

If the template device supports Telnet, the Telnet -> Command script right-click menu option will be displayed, in addition to SNMP. This option enables you to view and edit the script files configured for the telnet commands, as per your requirement. It displays a list of all the telnet commands supported for the specific device. Selecting a command will display the script file configured for the command.

1. Right-click on the template device and select **Telnet -> Command script**. The command editor is displayed.
2. Refer to the topic, **Editing Scripts**, to learn more about editing telnet scripts.

### Configuring Telnet Parameters

This option allows you to modify the user name and password to login to the telnet devices.

1. Right-click on the template device and select **Template Info**. The Template Info dialog is displayed.
2. Select the Telnet Tab. Specify the new **User name** and **Password** to login to Telnet.
3. Click **OK** to update the changes on the template based devices in the network.
Note:

1. Configuration of SNMP Agent values, traps, informs and behavior scripts made to the template devices will be reflected on all the devices in the network which was created using this template, except on the following devices:
   - The above changes will not be reflected on the device which is already configured from the network area.
   - The above changes will not be reflected on the device which is de-selected in the Template info -> Devices Tab dialog.

2. Configuration of SNMP, TFTP, CLI and Telnet parameters made to the template devices will be reflected on all the devices in the network which was created using this template.

3. Template based bulk configurations are not supported on TL1 devices.
Network View, List View and Topology View

- Network Device View
- Network List View
- Network Topology View

Network Devices View

When the network is created, the network view will be displayed as shown in the screen-shot given below. This view is the default view with the network devices displayed on the right-side network area and the device library listed on the left-side tree. Choose the View menu option to change the network view and View -> Tree view option to change the tree view.

The top panel of the network area displays the following information about the Network:

- Name of the Network: The name of the network selected in the network tree is displayed here.
- Number of devices displayed in a frame: You can customize the number of devices displayed in a frame, by selecting the required number from the list. For example, if you have 100 devices and you select 40 from the list, the 1st frame will display the first 40 devices, the 2nd frame will display the next 40 devices, the 3rd frame will display the next 20 devices. The default number of devices displayed in a frame is 25.
- Toggle buttons: To toggle through the frames, you can use the toggle buttons.
- Frame Number: The frame number displays the number of the frame currently displayed. When you select a device from the device tree, the frame number to which the device belongs, is displayed.
Network List View

Selecting the View -> List view option, will display the network device information in the form of a list as shown in the screen-shot given below. The name of the network devices, its ip address, port number of each protocol supported by the device and the read and write community will be displayed.
Network Topology View

Selecting the menu option View -> Topology view, will display the topology panel with the router devices in the network, displayed on the left-side tree and the connection between the routers, displayed on the right. The Topology editor enables you to establish inter-connections across routers, switches and other network devices and visualize the topological relationship between the devices. To learn more, refer Topology Editor.

To view the device and network information refer to the topic, Viewing Device and Network info.
Network Directory Structure

When you create a new network, the following folders will be created:

1. In `<SimulatorHome>/networks` directory:
   - `<NetworkName>`
     - `<mibs>`: Contains all the MIB files implemented by the SNMP agents in the network.
     - `<tcs>`: Contains all the TCS files implemented by the TL1 agents in the network.
     - `<snmp>`
       Contains a folder for each SNMP template device used to create the network. The folder name will be `<TemplateDeviceName>`.
       For example, if you add 5 devices of type "SunSolaris" to the network, the device type will be added to the template tree with name SunSolaris_1 and a folder with name `SunSolaris_1` will be created. If the same device type "SunSolaris" is used to create another set of devices, the device type will be added to the template tree with name SunSolaris_2 and a new folder with name SunSolaris_2 will be created.
       - `<TemplateDeviceName>`: The behavior script files (.py files) will be generated in this folder.
         - `<UniqueDeviceName>`: If any behavior script files are newly added/modified for a specific device of this type, in the network, a folder with the unique device name will be created and the added script files will be generated here.
     - `<cli>`
       Contains a folder for each template device that supports CLI, in the network. The folder name will be `<TemplateDeviceName>`.
       - `<TemplateDeviceName>`: All the script files (.py files) related to IOS commands will be generated in this folder.
         - `<UniqueDeviceName>`: If any IOS script files are newly added/modified for a specific device of this type, in the network, a folder with the unique device name will be created and the added script files will be generated here.
     - `<tftp>`
       Contains a folder for each template device that supports TFTP, in the network. The folder name will be `<TemplateDeviceName>`.
       - `<TemplateDeviceName>`: The tftp download and upload script files (.py files) will be generated in this folder.
         - `<UniqueDeviceName>`: If any upload/download script file is newly modified for a specific device of this type, in the network, a folder with the unique device name will be created and the added script files will be generated here.
     - `<telnet>`
       Contains a folder for each template device that supports Telnet, in the network. The folder name will be `<TemplateDeviceName>`.
       - `<TemplateDeviceName>`: The telnet script files (.py files) will be generated in this folder.
         - `<UniqueDeviceName>`: If any telnet script files are newly added/modified for a specific device of this type, in the network, a folder with the unique device name will be created and the added script files will be generated here.
o \texttt{<tl1>}

Contains a folder for each TL1 template device used to create the network. The folder name will be \texttt{<TemplateDeviceName>}. For example, if you add 5 devices of type "AcmeMSU" to the network, the device type will be added to the template tree with name AcmeMSU\_1 and a folder with name \texttt{AcmeMSU\_1} will be created. If the same device type "AcmeMSU" is used to create another set of devices, the device type will be added to the template tree with name AcmeMSU\_2 and a new folder with name AcmeMSU\_2 will be created.

- \texttt{<TemplateDeviceName>}: The template device folder will contain the script files, if any, configured for the TL1 device.
  - \texttt{<UniqueDeviceName>}: If any behavior script files are newly added/modified for a specific device of this type, in the network, a folder with the unique device name will be created and the added script files will be generated here.

2. In \texttt{<SimulatorHome>/mysql/data} directory:

- \texttt{<NetworkName>\_<1>}

  The network device data will be held in mysql database tables. The database tables will be created only for the template devices used to create the network. For example, if a 20 node network is created using SunSolaris and Cisco3600 devices, one set of database tables will be created for the SunSolaris\_1 device and one set of database tables will be created for Cisco3600\_1 device.

  If the configuration of a specific device in the network changes, a new entry is added to its template device table with the agentid of this device. This reduces the storage space to a great extent and enhances network performance. The database tables will start with the template device name followed by the table name. For example, sunsolaris\_1\_config.
Configuring Network Devices

- Overview
- Configuring an SNMP Device
- Configuring a TL1 Device

Overview

Individual SNMP and TL1 devices in the network can be simulated with different agent values, traps and autonomous messages and behavior scripts. These will not be reflected on the devices in the template tree or the device tree.

Configuring an SNMP Device

You can configure MIB values, load and simulate new MIBs, record values from real agents, configure traps and behavior scripts on a SNMP device in the network. Configuring a SNMP device in the network is similar to configuring an SNMP device in the Device Tree.

Recording from Real Agent

To configure values of a real agent in your MIB, you can record values from the real agent and save this configuration. This configuration file, which is a simulation of a real agent, can be used in the network.

To enable the recording option,

1. Select the menu bar option Configure -> Record Real Agent Values or right-click on the device and select Record Values. The Record from Real Agent dialog is displayed.
2. Refer to the topic, Simulating an SNMP Agent -> Recording from Real Agents, to learn about the steps involved in recording.

Configuring SNMP Agent Values

You can configure scalar and table values for the MIB, configure simulation types, and perform table operations. To configure values,

Right-click on the device and select SNMP Agent Values or select Configure -> SNMP Agent Values. The Configure SNMP Agent Values dialog is displayed with the MIBs loaded in the MIB tree. You can load a new MIB using the Load MIBs button or unload an existing MIB using the Unload MIBs button. You can use the Simulate option to simulate the loaded MIBs.

Refer to the topic, Simulating an SNMP Agent -> Configuring Values for the MIB, to learn about the steps involved in configuring MIB values and simulation types.

Refer to the topic, Simulating an SNMP Agent -> Populating the SNMP Table, to learn about the table operations supported.
Configuring Traps

You can configure the SNMP agents in the network so that the agent will send traps when it satisfies the specified condition. The following traps can be configured.

- Request-based traps
- Threshold-based traps
- Timer-based traps

To configure traps,
1. Right-click on the device and select SNMP Traps or select Configure -> SNMP Traps. The Trap Configuration dialog is invoked.
2. Refer to the topic, Simulating an SNMP Agent -> Simulating Traps, to learn about the steps involved in trap configuration.

Configuring Informs

You can configure the SNMP agents in the network so that the agent will send informs when it satisfies the specified condition. The following informs can be configured.

- Request-based informs
- Threshold-based informs
- Timer-based informs

To configure informs,
1. Right-click on the device and select SNMP -> Informs or select Configure -> SNMP Informs. The Inform Configuration dialog is invoked.
2. Refer to the topic, Simulating an SNMP Agent -> Simulating Informs, to learn about the steps involved in inform configuration.

Configuring Behavior Scripts

You can configure Real-Time Behavior of SNMP devices in the network, using Jython scripts. To configure behavior scripts,
1. Right-click on the device and select SNMP Behavior or select Configure -> SNMP Behavior. The Configure Behavior dialog is invoked.
2. Refer to the topic, Simulating a Network -> Configuring Behavior in SNMP Devices, to learn about the steps involved in behavior configuration.

Configuring a TL1 Device

You can configure values for the parameters defined in the TCS file, load and simulate new TCS files, configure scripts to define agent behavior, simulate delayed response, send error response and acknowledgment for specific responses, and configure echo requests.

Configuring a TL1 device in the network is similar to configuring a TL1 device in the Device Tree. Refer to the section Configuring a Device Added in Device Tree -> Configuring a TL1 Device, to learn more.
Configuring Router/Switch Properties

- Overview
- Configuring Router Properties
- Configuring Switch Properties

Overview

A network management application discovers a device as a router or switch by polling the appropriate OIDs configured in the device. The OIDs polled varies with the type of the device. The Network Simulator captures these device specific parameters in a separate user interface and displays them in an easily understandable format. This enables the user to configure these values easily even without having to understand the MIB syntaxes.

Configuring Router Properties

1. To configure router properties, right-click on a router device and select the option Router Properties. The Router/Properties dialog is displayed, wherein you can configure the following node values of the RFC 1213 MIB.

   System Details: The following node values of the system group will be displayed.
   - System Description: RFC1213-MIB -> system -> sysDescription
   - System ObjectID: RFC1213-MIB -> system -> sysOID

   Interface Details:
   - The following interface details will be displayed for the instances available in the ifTable of the RFC 1213 MIB. You can add a new entry to the table, modify existing values or delete an entry from the table.
     - Interface Index: RFC1213-MIB -> interfaces -> ifTable -> ifIndex
     - Interface Name: RFC1213-MIB -> interfaces -> ifTable -> ifDescr
     - Physical Address: RFC1213-MIB -> interfaces -> ifTable -> ifPhysAddress
     - Interface Status: RFC1213-MIB -> interfaces -> ifTable -> ifAdminStatus

   - The IP address assigned to each interface will be displayed for the instances available in the ipAddrTable of the RFC 1213 MIB. You can add a new entry to the table, modify existing values or delete an entry from the table.
     - IP Address: RFC1213-MIB -> ip -> ipAddrTable -> ipAdEntIfIndex
     - IP Address: RFC1213-MIB -> ip -> ipAddrTable -> ipAdEntAddr

   - The ip addresses of the devices connected to each interface will be displayed for the instances available in the ipNetToMediaTable of the RFC 1213 MIB. You can add a new entry to the table, modify existing values or delete an entry from the table.
     - Device IP Address: RFC1213-MIB -> ip -> ipNetToMediaTable -> ipNetToMedialfIndex
     - Device IP Address: RFC1213-MIB -> ip -> ipNetToMediaTable -> ipNetToMediaPhysAddress

2. Select Update the update the values in the RFC 1213 MIB or select Cancel to ignore the changes made and exit from the dialog.
Configuring Switch Properties

1. To configure switch properties, right-click on a switch device and select the option **Switch Properties**. The **Switch Properties dialog** is displayed, wherein you can modify the following node values of the RFC 1213 MIB and Bridge MIB.

   **System Details** : The following node values of the **system** group in **RFC1213 MIB** will be displayed.
   - system - > sysDescription
   - system - > sysOID

   **Port Details** : The following node values of the **dot1dBase** group, **dot1dBasePortTable** and **dot1dStpPortTable** in **BRIDGE-MIB** will be displayed. You can add a new entry to the table, modify existing values or delete an entry from the table.
   - Number of ports - > dot1dBaseNumPorts
   - Port Index - > dot1dBasePortTable - > dot1dBasePort
   - Port Current State - > dot1dStpPortTable - > dot1dStpPortState

2. Select **Update** to update the values in the agent or select **Cancel** to ignore the changes made and exit from the dialog.
Configuring Behavior in SNMP Devices

- Overview
- Enabling / Disabling Behavior Service
- Configuring Behavior scripts
- Configuring Behavior Across Agents
- Modifying and Deleting an Already Configured Script
- Modifying and Deleting Condition for Action
- Modifying and Deleting Action to be Executed.
- Stopping Timer Script Execution
- Scripts in Examples Directory
- Sample Scripts

Overview

The most important part of the simulation is to setup the simulated environment. You can setup the desired environment by defining the behavior of devices in the network, so that it will behave like a real device. The behavior of devices, is done through Jython scripts. For example, device values can be changed, rows can be dynamically added or deleted, Traps and Informs can be configured to be sent, SNMP agents can be added/deleted from the existing network etc. You can also define behavior in such a way that behavior of one device can affect another device.

AdventNet Simulation Toolkit provides a ScriptInterface, which defines methods to perform simple operations like update, insert and delete from a JPython script. The Script API is provided under the `<simulator home>/examples/javadocs` directory. The name of the Script API file is `SNMP_TL1_SimScrIntf.html`.

If behavior is configured for a device in the device tree, this behavior will be reflected in all the devices dragged and dropped.

Enabling/Disabling Behavior Service

The Network Simulator provides option to enable/disable the execution of behavior scripts configured for the SNMP devices in the network, at agent runtime. If behavior service is disabled, the agent will not check for script configurations before sending the response. Hence, you can use this option to improve the performance of the network, if behavior scripts are not required to be executed.

By default, the service is enabled and the behavior scripts configured (if any), will be executed. To disable the service,

- Select Run -> Initialize Services. The Initialize Service dialog is displayed.
- Deselect the option Enable SNMP Behavior Service.
- Start the Agent. Now, the behavior scripts configured in the simulator, will not be executed.

Note: The options in the Initialize Services dialog will be disabled when the SNMP agent is started.
Configuring Behavior

The Network Designer provides an intuitive UI to configure Behavior scripts. To invoke the Behavior dialog,

- Select the Device in the Device Tree for which you would like to configure Behavior.
- Click on the icon in the toolbar or select Configure ->SNMP Behavior option from the menu bar or right-click and select SNMP Behavior option from the pop up menu. This will display the Configure Behavior dialog.

The Configure Behavior dialog will list the configured scripts files, the defined conditions and actions to execute the script files. You can configure new Behavior, modify or delete an existing Behavior configuration, add new conditions and actions, modify existing conditions and actions, from this dialog.

Steps in Configuring Behavior

**Step I : Configuring the Conditions to Execute the Script** : Here, you will define when the script should be executed. Scripts can be executed based on three conditions. Scripts can be executed based on a request, based on a threshold condition and value, or timer based.

Step 2 and Step 3 is used for generating the script file from the UI, with the necessary actions to be executed and the conditions to execute the actions (if any). The conditions and actions are defined using the Script API methods.

<table>
<thead>
<tr>
<th>Note: Defining Step 2 and Step 3 is not required in the following cases :</th>
</tr>
</thead>
<tbody>
<tr>
<td>• If you have defined your own script file, with necessary conditions and actions, you can configure this script file in step 1 and can Finish 'Behavior Configuration'.</td>
</tr>
<tr>
<td>• If your scope for defining the conditions and actions is beyond the current scope of our API, you have to write your own script. Configure this script file in step 1 and Finish 'Behavior Configuration'.</td>
</tr>
<tr>
<td>• If you want to configure the script file available in the Examples directory, you can specify this file in Step 1 and Finish 'Behavior Configuration'.</td>
</tr>
</tbody>
</table>

In all the above cases, the conditions and actions defined in the script file will be executed only when the condition to execute the script (specified in step 1) is satisfied.

**Step 2 : Configuring the Conditions to Execute the Actions** : In this step you can define on what conditions the actions are to be executed. You can define any number of conditions.

**Step 3 : Configuring the Actions to be Executed** : In this step, you can define the actions to be executed, when the condition to executed the script and the condition to execute the action (if any) are satisfied.

**Step 1: Configuring the Conditions to Execute the script**

To configure conditions to execute the script, select the Add button in the Behavior Configuration dialog. The Define Your Script dialog is invoked. Select any of the following conditions and configure the necessary details on the right-side of the dialog.

- Request based Script
- Threshold based script
- Timer based script
Request Based Script

1. Choose the Request based icon that is displayed on the left of the wizard.
2. Select the MIB node and instance based on which behavior will be configured.
3. The name of the selected MIB node will be displayed as the script filename. You can also specify your own script file name or browse and select the required file, if it already exists.
4. If you want the script to be copied on to the device directory, you can check the option Copy script file to local directory.
5. The script requires a trigger, in which particular instance it should be executed. It can also be executed in more than one instance. You have the following options:
   - Before processing Get
   - After processing Get
   - Before processing GetNext
   - After processing GetNext
   - Before processing Set
   - After processing Set
6. Click Next in the wizard to proceed to Step 2 or click Finish to finish Behavior configuration and exit the dialog.

Threshold Based Script

1. Choose the Threshold Based icon that is displayed on the left of the wizard.
2. Threshold based scripts can be configured only for leaf nodes. Select the leaf node and instance based on which behavior will be configured.
3. The name of the selected node will be displayed as the script filename. You can also browse and select the required file, if it already exists.
4. If you want the script to copied on to the device directory, you can check the option Copy script file to local directory.
5. The script requires a trigger, in which particular instance it should be executed. It can also be executed in more than one instance. You can specify any one of the following threshold conditions and value. When the condition and value is satisfied, the script will be executed.
   - Equal to
     1. Not equal to
     2. Less than
     3. Greater than
     4. Custom range : For Custom range, the script will be executed if the value of the MIB node is not within the lower and upper threshold value. For example, if the lower threshold value is 10 and upper threshold value is 20, the script is executed if the variable value is less than or equal to 10 (or) greater than or equal to 20.
2. Specify the threshold value. For custom range, the lower threshold value and higher threshold value must be specified.

3. Click **Next** in the wizard to proceed to **Step 2** or click **Finish** to finish Behavior configuration and exit the dialog.

**Timer Based Script**

1. Choose the Timer Based icon that is displayed on the left of the wizard.

2. The name of the script file will be displayed, based on the time specified to execute the script. You can also browse and select the required file, if it already exists.

3. If you want the script to copied on to the device directory, you can check the option Copy script file to local directory.

4. You can specify when you want the scripts to get executed.
   - **Execute scripts when agent starts**: The configured script would get executed as soon as agent starts.
   - **Execute scripts when agent stops**: The configured script would get executed when the agent stops.
   - **Execute scripts at specified time**: You can specify the date and time of script execution. Choose the date and time using the spinner buttons provided. The scripts will get executed at the specified time.
   - **Execute scripts after specified time after agent startup**: You can specify the time in hours, minutes and seconds after which the script will get executed after the agent startup time.
     - **Periodically execute (once in)**: If scripts are to be executed at specified time or after specified time, periodically, you can check this option. Specify the duration as once in number of hour(s)/day(s).

5. Specify a time interval between each script execution. Eg. Timer interval 1000 millisecs.

6. The number of times you want the script to execute at the specified time. Eg: Number of times 5.

7. Click **Next** in the wizard to proceed to **Step 2** or click **Finish** to finish Behavior configuration and exit the dialog.

Example: You have configured to periodically execute the script once in every 5 days, the **Number of Times** the script must be executed is 10 with a timer interval of 2 seconds This configuration will execute the script once in every 5 days.

**Note:**

Selecting the device right-click option **SNMP -> Stop Timer Scripts** will stop the execution of timer scripts in Network Simulator. To stop the execution of **timer scripts** in all devices, right-click on the blank network area and select **Stop All timer scripts**.
Step 2 : Configuring the Conditions to Execute the Action

In this step you can define on what conditions you want the actions defined in the scripts, to be executed. You can define any number of conditions.

1. Click Add button that brings up the Configure Condition to be Checked dialog. This dialog displays all the methods available for the selected node.
2. Select the desired method, for example, getCurrentInstance.

   **Note:** A few methods that start with "getCurrent" are applicable only for the agent implemented in the selected device. (i.e) the condition you are specifying is for the selected agent. When you select other methods like getNodeValue, this is valid for all the network elements. You will find the text field Select Agent enabled and network elements listed in the drop down box.

1. If you wish to configure behavior across the network, select the agent from the network. Specify the OID and instance. The MIBs of the selected agent will only be displayed in the MIB tree.
2. Select the operator from the drop down box. The list of operators are listed as follows.

   - Not equal to !=
   - Lesser than <
   - Greater than >
   - Lesser than or equal to <=
   - Greater than or equal to >=

5. After choosing the operator you can specify the value to which the expression should be equated to.
6. Click ok, You will see that the conditions that you have specified gets added to the Conditions to execute actions list.
7. Click Add to define another condition (or) click Next to proceed to Step 3 or click Finish to finish Behavior configuration and exit the dialog.
8. As already mentioned, you can specify as many conditions as you want. Hence you have to specify whether you want to
   - Match all conditions added in the list
   - Match any of the conditions added in the list

Step 3 : Configuring the Actions to be Executed

In this step, you can configure the actions to be performed when the condition to execute the script and the condition to execute the action (if any) is satisfied.

1. Click Add, to brings up the Configure Actions to be Performed dialog.
2. Select a method from the drop down list. Methods are listed below

   - Update value
   - GetNode value
   - Insert row into table
   - Delete a row from table.
   - Delete all rows from table
   - Send trap
   - Send Inform
   - Generate random number
   - Delay action
   - Terminate operation
Update Value:

1. Select the **Update Value** method.
2. When simulating behavior of devices across a network, the agents in
   the network are listed in the drop down list. When you select the
   agent the value will be updated in the specified agent.
3. To specify the OID and instance on which it should be updated, click
   on the **Select OID** button. Only the MIBs of the selected agent will be
   displayed. When selecting the OID and the instance, the details of
   the nodes are displayed.
4. The syntax of the node is displayed along with the simulation type.
   You can choose the simulation type the node should have, and
   specify values for the simulation type.
5. Click **OK** to add the action to the list.

   **Example:**
   
   If you select the OID .1.3.6.1.2.1.1.6 the syntax would be Display
   String. The options that you have with this node is constant and
   random. If you choose the random option, you will see the range
   button being enabled. Specify value and click Add so that the value
   will be associated at random.

Get Node Value:

This method is used to get the current value of a MIB node.

1. Choose the method **GetNode Value** from the drop down list.
2. When simulating behavior of devices across a network, the agents in
   the network will be listed in the drop down list. You will be able to
   select an agent from the network.
3. To specify the OID, click on the **Select OID** button. Only the MIBs of
   the selected agent will be displayed. Select the OID and the instance.
4. Now you have to specify a variable in the field that is displayed at the
   right corner.
5. The value of the selected OID will get updated in the specified
   variable.
6. Click **OK** for the action to be added to the list.

Insert Row to a Table:

This method is used to insert an entry in a table.

1. Choose the method **Insert Row to Table** from the drop down list.
2. When simulating behavior of devices across a network, the agents in
   the network will be listed in the drop down list. You will be able to
   select agent from the network.
3. To specify the OID, click on the **Select OID** button. Only the MIBs of
   the selected agent will be displayed. Select the OID and the instance.
4. To insert row into the table.
5. Click **Add** to add a row. You will see dialog coming up with details
   that contain
   - Column name
   - Simulation type
   - value
5. You can make the desired changes to the row entries.
6. Click OK. You have chosen to insert the selected row when the script executes and condition that you have specified earlier is satisfied.

Delete a Row From Table.

This method is used to delete a specific row from the table.

1. Select the method Delete a Row from Table from the drop down list.
2. When simulating behavior of devices across a network, the agents in the network will be listed in the drop down list. You will be able to select agent from the network.
3. To specify the OID, click on the Select OID button. Only the MIBs of the selected agent will be displayed. Specify the table OID and instance that you want to delete from the table.
4. Click OK to delete table

Delete All Rows From Table

This method is used to delete all the rows from a given table.

1. Select the method Delete All Rows from Table from the drop down list.
2. When simulating behavior of devices across a network, the agents in the network will be listed in the drop down list. You will be able to select agent from the network.
3. To specify the OID, click on the Select OID button. Only the MIBs of the selected agent will be displayed. Specify the table OID, all rows from the specified table node will be deleted.
4. Click OK.

Send Trap

There are three trap versions available : Sending a V1 trap, Sending a V2 trap , Sending a V3 trap.

Trap Version : Select the Trap Version from the list displayed. If the trap version is v1, the trap type must be specified as either Generic type or Specific type. The default trap version is SNMPv1.

If the trap version is v3, the v3 agent must be started and the user name and context name of the v3 user must be specified. Please refer the topic "Simulating a v3 Agent" to know more about starting the v3 agent, the default user names and password, and configuring new user names and password.

Generic Type : If the trap type value is zero through five, then the trap is one of the generic SNMP traps defined by the snmp group of MIB-II and the Specific trap type will be zero. If the trap type value is six, then the trap is an enterprise-specific trap that is defined in a private MIB.

coldStart (0),
warmStart (1),
linkDown (2),
linkUp (3),
authenticationFailure (4),
egpNeighborLoss (5),
enterpriseSpecific (6).
Specific Type: Specify an integer value for the enterprise-specific trap that is defined in a private MIB.

Enterprise OID: The OID specified in the Enterprises field indicates exactly which management enterprise defines the trap.

Manager Name: Specify the manager name. If the Manager is running on the same machine it will contain the value localhost. If the Manager is running on a different machine specify the manager name or IP address of that machine.

Manager Port: The default manage port is 162. If the manager is running on a different port, specify the port number.

Community: Specify the community name of the manager.

Variable Bindings: It enables the user to add more varbinds to the trap other than the trap variables/objects. It is optional.

Note: When you configure a Timer based SNMP V3 trap, the trap will be sent regardless of the manager's presence. You need the User name and the context name only to receive the traps. But when the SNMP v3 trap is configured as request based, the trap will not be sent unless the request is generated from the manager with the user name and context name.

Send Inform

1. Select the method Send Inform from the drop down list.
2. Manager name: Specify the manager name. By default it is given as localhost.
3. Manager port: Specify the port at which the manager is running. By default it is 162.
4. The community string is set to public by default.
5. You have to specify the SysUpTime and Enterprise OID.

Example: SysUpTime can take value of 1101101 and Enterprise OID can have take a value like 1.3.6.1.2162.2.1.1
6. To add varbinds to the inform, click on the Configure button. It is optional.

Generate Random Numbers

To generated random numbers, specify minimum value and a maximum value so that random numbers will be generated within the specified range.

Minimum value can be 100 and maximum value can 200, random numbers will be generated within this range.

Delay Action:

You can introduce a delay in the action. Specify the value in milliseconds and the script execution will be delayed by the specified time.

Terminate Operation

This action can be configured before a GET/GETNEXT/SET. It will terminate the ongoing operation and send the corresponding error message specified. The error
message can be specified from the drop down box. You can choose any one of the following error types.

- tooBig
- noSuchObject
- noSuchInstance
- endOfMibView
- badvalue
- readOnly
- genErr
- noAccess

3. You can add any number of conditions Click **Add** to define another condition or click **Finish** to finish Behavior configuration and exit the dialog.

**Simulating Behavior Across Agents**

You can simulate behavior across agents in the network. For example, you have a network with five devices (device 1 ... device 5). You can simulate behavior in such a way that if a specific condition is satisfied define condition for execution in device1, define the condition for action in device2 and the action can be defined to take place on device5.

When you select an agent, the MIBs of the selected agent will only be displayed.

**Modifying and Deleting Script**

To modify a script,

1. Select the script to be modified, from the list displayed in the **Configure Behavior** dialog.

2. Click **Modify**, to display the existing configuration. Make the necessary changes and click **Finish**.

To delete a script,

1. Select the script to be deleted, from the list displayed in the **Configure Behavior** dialog.

2. Click **Delete**, to delete the script file. The conditions and actions configured for the script will also be deleted.

**Modifying & Deleting Condition for Action**

To modify a condition for action,

If you require to change an existing condition.

1. Select the condition from the **Condition to execute action** list and click **Modify** to change the defined condition.

2. The **Configure Condition to be Checked** dialog is brought up, you can make the required changes and click OK for the changes to take effect.

To delete a condition,

You can delete a condition if it is no longer required.

1. Select the condition from the **Condition to execute action** list.

2. Click **Delete** to remove the condition from the list.
Modifying and Deleting Action to be Executed

To modify an action,

- Select the action from the Actions to be Performed list and click Modify to change the defined action.
- The Configure Actions to be Performed dialog is brought up, you can make the required changes and click OK for the changes to take effect.

To delete an action,

1. Select the action from the Actions to be Performed list
2. Click Delete to delete the defined action.

Stopping Timer Script Execution

Selecting the device right-click option SNMP -> Stop Timer Scripts will stop the execution of timer scripts in Network Simulator. To stop the execution of timer scripts in all devices, right-click on the blank network area and select Stop All timer scripts.

Scripts in Examples Directory

The example scripts for SNMP devices are available in <SimulatorHome>/examples/scripting/snmp directory. Some of the example scripts are explained below:

1. DeleteAllRows.py

The script file uses the scriptAPI method deleteAllRows to delete all the entries in the ifTable. It takes the tableEntryOID as input.

Format :

```
scriptinterface.deleteAllRows(String tableEntryOid)
```

2. DeleteRow.py

This script file uses the scriptAPI method deleteRow to delete the first entry in the ifTable. It takes the tableEntryOID and the instance of the entry to be deleted, as input.

Format :

```
scriptinterface.deleteRow(String tableEntryOid,String instance)
```

where

- tableEntryOid is the Object Id of the table entry node
- instance is the instance of the entry to be deleted

3. DeleteRowWithValue.py

This script uses the scriptAPI method deleteRowWithValue to delete a specific entry from a table when the value of the specified column matches the specified value. It takes the tableEntryOID, the column name and value as input.

Format :

```
scriptinterface.deleteRowWithValue(String tableEntryOid,String columnName String value)
```
where

tableEntryOid is the Object Id of the table entry node
columnName is the name of the column node in the table
value is the value of the specified column

4. GetAllTableValues.py

This script file uses the scriptAPI method `getAllTableValues` to return the values of all the nodes for all the rows present in Table. It takes the tableEntryOID as input.

Format:

```python
scriptinterface.getAllTableValues(String tableEntryOid)
```

where

tableEntryOid is the Object Id of the table entry node

5. GetCurrentInstanceAndValue.py

This script file uses the scriptAPI method `getCurrentInstance()` and `getCurrentValue()` to return the instance and value of an incoming SET request. It should be configured for the condition "Before Processing Set" on a columnar node. This script gets the instance of the OID in the incoming SET request and its value and saves it in a text file named instance.txt

Format:

```python
scriptinterface.getCurrentInstance()
scriptinterface.getCurrentValue()
```

6. GetInstances.py

This script file uses the scriptAPI method `getInstances()` to get all the instances in a table. It takes the tableOID as input and will return the instances in a Vector.

Format:

```python
scriptinterface.getInstances(String tableOid)
```

where

tableOid is the Object Id of the table node

7. GetNextInstance.py

This script file uses the scriptAPI method `getNextInstance()` to get the instance next to the one specified. It takes the Table OID and the instance of a table as input. This will return the value as a String

Format:

```python
scriptinterface.getNextInstance(String tableOid, String instance)
```

where

tableOid is the Object Id of the table node
instance is the instance of a row present in the table
8. GetNodeValue.py

This script file uses the method `getNodeValue()` to get the current value of a scalar or a table node. It takes the OID of the MIB node and the instance as input. This method will return the value as a String.

Format:

```java
scriptinterface.getNodeValue(String objectId, String instance)
```

where

- `objectId` is the object identifier (without the instance) of the MIB Node.
- `instance` is the instance of the node whose value should be obtained, for scalar nodes this should be ".0"

9. GetRowCount.py

This script file uses the method `getRowCount()` to get the total number of rows in a table. It takes the Table OID as input and returns an integer value.

Format:

```java
scriptinterface.getRowCount(String tableOid)
```

where

- `tableOid` is the Object Id of the table node.

10. InsertRow.py

This script file uses the `insertRow()` method to insert an entry in the Table. It takes the Table OID, the Object Id of the columns in the table, the values of each column and its simulation type as input.

Format:

```java
scriptinterface.insertRow(String tableOid, String[] objectId, String[] values, String[] simTypes)
```

where

- `tableOid` is the object id of the table node in which the entry has to be added.
- `objectId` is an array of the object Id of the columns in the table.
- `values` is an array of values for the columns in the table.
- `simTypes` is an array of the simulation types for the columns.

11. MaskNode.py

This script file uses the method `maskNode()` to mask a MIB node from the Manager.

Format:

```java
scriptinterface.maskNode(String objectId, String instance, String maskValue).
```
12. UpdateColumnarValue.py and UpdateScalarValue.py

This script file uses the method `updateValue()` to update the value of a scalar node and columnar node. It takes the OID and instance of the MIB node for which the value should be updated, the value to be assigned and the simulation type.

Format

```java
scriptinterface.updateValue(String objectId, String instance, String value, String simType)
```

where

- `objectId` is the object identifier (without the instance) of the mib node whose value has to be updated.
- `instance` is the instance for which the value should be updated, for scalar nodes this should be ".0".
- `value` is the value to be assigned to the node.
- `simType` is the type of simulation required.

13. SendInform.py

This script file uses the method `sendInform()` to send an inform request with no extra varbinds. It takes the manager name, manager port, community, sysUpTime and SNMP Trap OID as input.

Format

```java
scriptinterface.sendInform(String mgrName, String mgrPort, String community, String sysUpTime, String snmpTrapOid)
```

where

- `mgrName` - the name of the manager to which the inform has to be sent.
- `mgrPort` - the port at which the manager is listening for incoming inform requests.
- `community` - the snmp community.
- `sysUpTime` - the System UpTime.
- `snmpTrapOid` - the value of the snmpTrapOid.

14. SendInform_varbinds.py

This script file uses the scriptAPI method `sendInform()` to send an inform request with extra varbinds. The varbind for the inform is specified with the following information:

- `varbindOid` is OID of the inform varbind.
- `varbindTypes` is the type of the varbind.
- `varbindValues` is the value of the varbinds.

Format

```java
scriptinterface.sendInform(String mgrName, String mgrPort, String community, String sysUpTime, String snmpTrapOid, String[] varbindOid, String[] varbindTypes, String[] varbindValues)
```
15. SendV1Trap.py

This script file uses the scriptAPI method `sendV1Trap()` to send a v1 trap without varbinds. It takes the manager name, manager port, community, enterprise OID, generic type and specific type as input.

Format

```python
scriptinterface.sendV1Trap(String mgrName, int mgrPort, String community, String enterpriseOid, int genericType, int specificType)
```

where

- `mgrName` - the name of the manager to which the trap has to be sent
- `mgrPort` - the port at which the manager is listening for incoming traps
- `community` - the snmp community
- `enterpriseOid` - the Enterprise OID
- `genericType` - the Generic Type
- `specificType` - the Specific Type

16. SendV1Trap_varbinds.py

This script file uses the scriptAPI method `sendV1Trap()` to send a v1 trap with varbinds. It takes the manager name, manager port, community, enterprise OID, generic type, specific type, varbind OID, type and value as input.

Format

```python
scriptinterface.sendV1Trap(String mgrName, int mgrPort, String community, String enterpriseOid, int genericType, int specificType, String[] varbindOid, String[] varbindTypes, String[] varbindValues)
```

where

- `mgrName` - the name of the manager to which the trap has to be sent
- `mgrPort` - the port at which the manager is listening for incoming traps
- `community` - the snmp community
- `enterpriseOid` - the Enterprise OID
- `genericType` - the Generic Type
- `specificType` - the Specific Type
- `varbindOid` - oid of the varbind
- `varbindTypes` - the type of the varbind
- `varbindValues` - the value

17. SendV2Trap.py

This script file uses the `sendv2Trap()` method to send a v2 trap with no extra varbinds.

```python
scriptinterface.sendV2Trap(String mgrName, int mgrPort, String community, String trapOID)
```

18. SendV2Trap_varbinds.py

This script file uses the `sendv2Trap()` method to send a v2 trap with extra varbinds. The varbinds for the trap is specified with the following information:
varbindOid - oid of the varbind
varbindTypes - the type of the varbind
varbindValues - the value

```java
scriptinterface.sendV2Trap(String mgrName, int mgrPort, String community, String trapOID, String[] varbindOid, String[] varbindTypes, String[] varbindValues)
```

19. SendV3Trap.py

This script file uses the `sendv3Trap()` method to send a v3 trap with no extra varbinds.

```java
scriptinterface.sendV3Trap(String mgrName, int mgrPort, String community, String trapOID, String userName, String contextName)
```

20. SendV3Trap_varbinds.py

This script file uses the `sendV3Trap()` method to send a v3 trap with extra varbinds. The varbinds for the trap is specified with the following information:

- `varbindOid` is oid of the varbind
- `varbindTypes` is the type of the varbind
- `varbindValues` is the value

```java
scriptinterface.sendV3Trap(String mgrName, int mgrPort, String community, String trapOID, String userName, String contextName, String[] varbindOid, String[] varbindTypes, String[] varbindValues)
```

21. TerminateOperation.py

This script file uses the `scriptAPI` method `terminateOperation()` for terminating the current operation i.e. a Get, GetNext or Set operation. The general format of the `terminateOperation` method is:

```java
scriptinterface.terminateOperation(int errorNumber)
```

where, `errorNumber` is the error to be thrown on terminating the operation.

24. GetTotalColumns.py

This script file invokes the method `getTotalColumns()` and returns the total number of columns in a table. It takes the table entry oid as input.

Format

```java
scriptinterface.getTotalColumns(String tableEntryOid);
```

25. GetAllColumnByNames.py

This script file invokes the method `getAllColumnByNames()` and returns the name of all the columns in the table. It takes the table entry oid as input.

Format

```java
scriptinterface.getAllColumnByNames(String tableEntryOid);
```
26. getAllColumnByOids.py

This script file uses the method `getAllColumnByOids()` and returns the OID value of all the columns in the table. It takes the table entry oid as input.

Format :
```
scriptinterface.getAllColumnByOids(String tableEntryOid);
```

27. GetAllIndices.py

This script file uses the method `getAllIndices()` to return the string array having name of all the indices defined in a table. It takes the table entry oid as input.

Format :
```
scriptinterface.getAllIndices(String tableEntryOid);
```

28. SendTrap.py

This script file uses the scriptAPI method `sendTrap()` to send traps configured in the Trap Configuration dialog. It takes the associated variable as input. The parameter ‘associatedVariable’ is the name with which the trap is configured in 'TrapConfiguration' dialog.

The manager details and trap details will be taken as such configured in the 'TrapConfiguration' dialog.

Format :
```
scriptinterface.sendTrap(String associatedVariable);
```

29. SendTrapWithEnterpriseOid.py

The script file uses the method `sendTrap()` to send traps configured in the Trap Configuration dialog. It takes the associated variable as input. The parameter ‘associatedVariable’ is the name with which the trap is configured in 'TrapConfiguration' dialog. The 'enterpriseOid' is the one which is going to be sent in the trap.

The manager details and trap details will be taken as such configured in the 'TrapConfiguration' dialog. The enterpriceOid will be the one specified in the second argument irrespective of the one configured in 'TrapConfiguration' dialog.

Format :
```
scriptinterface.sendTrap(String associatedVariable, String enterpriseOid);
```

30. GetConfigDirectory.py

This script file uses the scriptAPI method `getConfigDirectory()` to get the current working directory name. It returns the directory name as a string.

Format :
```
scriptinterface.getConfigDirectory();
```
31. SendV1TrapWhileAddingRow.py

This Script file will send the V1 Trap while adding the row in the tables. This Script file using the following Script API Methods:

- getRowCount()
- getCurrentInstance()
- sendV1Trap()

32. SendV1TrapWithVarbindsWhileAddingRow.py

This Script file will send the V1 Trap with Varbinds while adding the row in the tables. This Script file uses the following Script API Methods:

- getRowCount()
- getCurrentInstance()
- sendV1Trap()

33. StartSnmpAgent.py

This script file uses the scriptAPI method startAgent() to start an SNMP agent. This method is useful in the runtime to start any SNMP agent in the Network from any other agent. This file will work only for the Network Simulator and not for the SNMP Agent Simulator.

Format:

```
scriptinterface.startAgent(java.lang.String networkName,
java.lang.String ipAddress, java.lang.String
snmpPort,java.lang.String tl1Port);
```

34. StopSnmpAgent.py

This script file uses the scriptAPI method stopAgent() to stop an SNMP agent. This method is useful in the runtime to stop any SNMP agent in the Network from any other agent. This file will work only for the Network Simulator and not for the SNMP Agent Simulator.

Format:

```
scriptinterface.stopAgent(java.lang.String networkName,
java.lang.String ipAddress, java.lang.String
snmpPort,java.lang.String tl1Port) ;
```

Sample Scripts Defined

Given below are some sample behavior scenarios. The steps to define the behavior scenario and the output script file are given.

**Script with condition for execution alone**

In this sample, the script DeleteAllRows.py will be executed, after a get request is made on the MIB node "ifDescr" in the ifTable of the RFC 1213 MIB, with instance as .1

1. Select the Add button in the Behavior Configuration dialog.
2. Choose the Request based icon that is displayed on the left of the wizard.
3. Select the MIB node "ifDescr" in the ifTable of the RFC 1213 MIB. Specify the instance as .1
4. The name of the selected MIB node will be displayed as the script filename. Browse and select the script file "DeleteAllRows.py" in the ./examples directory.

5. The script requires a trigger, in which particular instance it should be executed. Select the option "After processing Get"

6. Click Finish to end Behavior configuration and exit from the dialog.

**Script with condition for execution, condition for action, action to be executed**

In this sample, the script file, `ifOutErrors.py` will be executed, when a set request is made on the node `ifOutErrors`. The condition for action is "if current value of ifOutErrors is greater than 10", the action to be executed is "send a v2trap". Hence, the script will send a trap, when a set is made on ifOutErrors with value greater than 10. The steps given below, defines the above Behavior:

1. Select the Add button in the Behavior Configuration dialog.
2. Choose the Request based icon that is displayed on the left of the wizard.
3. Select the MIB node "ifOutErrors" in the ifTable of the RFC 1213 MIB. Specify the instance as .1
4. The name of the selected MIB node will be displayed as the script filename "ifOutErrors.py".
5. Specify the condition to execute the script, by selecting the option "Before processing Set"
6. Click Next button, to specify the condition for action.
7. Click Add and select the method `getCurrentValue`, with operator as > and value as 10.
8. Click OK to add the condition to list. Now, you will find the condition `getCurrentValue() >10`, added to the list.
9. Click Next button, to specify the action to be executed.
10. Click Add and select the method `Send Trap`, specify the version as `Snmp v2c`, with other options like `Manager`, `Port` and `Community` as default.
11. Click OK to add the action to the list. Now, you will find the action `Send a V2c trap to localhost and port 162`, added to the list.
12. Click Finish to end Behavior configuration. You will find the script `ifOutErrors.py`, added to the list with the defined conditions and actions, in the main dialog. Click Close to exit from the dialog.

The script file `ifOutErrors.py` is given below:

```python
# The Actions
def actionfunc0():
    scriptinterface.sendV2Trap("localhost",162,"public",".1.3.6.1.2.1.11")

# The end of the Actions

# The Condition
if(scriptinterface.getCurrentValue() > "10"): actionfunc0()

# The end of Conditions
```
Script with condition for execution and actions to be executed

In this sample, the script file, timer.py will be executed, when the agent is started. The following actions are configured to be executed at agent startup : Get the node value sysUpTime and store it in a variable., send v1Trap. The steps given below, defines the above Behavior :

1. Select the Add button in the Behavior Configuration dialog.
2. Choose the Timer based icon that is displayed on the left of the wizard.
3. Specify the script filename as timer.py
4. Select the option Execute Script while Agent Starts.
5. Specify the Time Interval and Number of Times the script must be executed.
6. Click Next. The dialog to define the condition for action will be displayed. In this sample, we are not defining any condition for action. Hence click Next, to go the next dialog.
7. Click Add to add the action to the list. Select GetNodeValue, browse and select the OID as sysUpTime in System group of RFC 1213 MIB, specify the variable name as agentUpTime. The value of the node will be stored in this variable.
8. Click OK. You will find the action GetValue of Node of oid .1.3.6.1.2.1.1.3 with Instance .0, added to the list.
9. Click Add again, to add the second action. Select the method Send Trap, with other options like version, Manager, Port and Community as default.
10. Click OK. You will find the second action Send a V1 trap to localhost and port 162, added to the list.
11. Click Finish to end Behavior configuration. You will find the script timer.py, added to the list with the defined actions, the main dialog. Click Close to exit from the dialog.

The script file timer.py is given below:

```python
# The Actions

def actionfunc0():
    agentUpTime = scriptinterface.getNodeValue(".1.3.6.1.2.1.1.3", ".0")
    scriptinterface.sendV1Trap("localhost",162,"public",".1.3.6.1.2.1.11",1,0)

# The end of the Actions

# The Condition
actionfunc0()

# The end of Conditions
```

Script with different sets of conditions for action and actions to be executed

In this sample, the script file, ifAdminStatus.py will be executed, when a get request is made on the node ifAdminStatus in ifTable of the RFC 1213 MIB. Two sets of conditions and actions are defined as given below:

- If node value of IfAdminStatus with instance ".1" is not equal to 1, Send a V1 trap to localhost at port 162
- If total number of rows in ifTable is less than 5, Insert a row in the ifTable.

The steps given below, defines the above Behavior :

1. Select the Add button in the Behavior Configuration dialog.
2. Choose the Request based icon that is displayed on the left of the wizard.
3. Select the MIB node "ifAdminStatus" in the ifTable of the RFC 1213 MIB. Specify the instance as .1
4. The name of the selected MIB node will be displayed as the script filename "ifAdminStatus.py".
5. Specify the condition to execute the script, by selecting the option "After processing Get"
6. Click Next button, to specify the condition for action.
7. Click Add and select the method getNodeValue, to specify the OID, browse and select the ifTable, with operator as != and value as 1.
8. Click OK to add the condition to list. Now, you will find the condition getNodeValue(".1.3.6.1.2.1.2.2.1.7", ".1") != 1 , added to the list.
9. Click Next button, to specify the action to be executed.
10. Click Add and select the method Send Trap, specify the version as Snmp v1, with other options like Manager, Port and Community as default.
11. Click OK to add the action to the list. Now, you will find the action Send a V1 trap to localhost and port 162 added to the list.
12. Click Finish. You will find the script ifAdminStatus.py, added to the list with the first set of condition and actions defined conditions and actions, in the main dialog.
13. To add the second set of condition and action to the list, click Add button .
14. To add the second condition, click Add and select the method getRowCount, to specify the OID, browse and select the ifTable, with operator as < and value as 5.
15. Click OK to add the condition to list. Now, you will find the condition getRowCount(".1.3.6.1.2.1.2.2.2") < 5 Insert a Row in the ifTable , added to the list.
16. Click Next button, to specify the action to be executed.
17. Click Add and select the method InsertRow, to specify the OID, browse and select the ifTable. To specify the values of the new row, click Add . The new row with default values will be displayed. Modify the ifIndex value as '2' and click OK.
18. Click OK to add the action to the list. Now, you will find the action Insert a Row in the ifTable, added to the list.
19. Click Finish to end Behavior configuration. Click Close to exit from the dialog.

The script file ifAdminStatus.py is given below:

```python
# Imports required for the scripts
# to create an array the following lines should be in the
script file
from java.lang import String
from jarray import array
# The end of imports

# Row values of Tables
objectid1 = array([".1.3.6.1.2.1.2.2.1.1", ".1.3.6.1.2.1.2.2.1.2", ".1.3.6.1.2.1.2.2.1.3", ".1.3.6.1.2.1.2.2.1.4", ".1.3.6.1.2.1.2.2.1.5", ".1.3.6.1.2.1.2.2.1.6", ".1.3.6.1.2.1.2.2.1.7", ".1.3.6.1.2.1.2.2.1.8", ".1.3.6.1.2.1.2.2.1.9", ".1.3.6.1.2.1.2.2.1.10", ".1.3.6.1.2.1.2.2.1.11", ".1.3.6.1.2.1.2.2.1.12", ".1.3.6.1.2.1.2.2.1.13", ".1.3.6.1.2.1.2.2.1.14", ".1.3.6.1.2.1.2.2.1.15", ".1.3.6.1.2.1.2.2.1.16", ".1.3.6.1.2.1.2.2.1.17", "]
```
".1.3.6.1.2.1.2.1.18",".1.3.6.1.2.1.2.1.19",
".1.3.6.1.2.1.2.1.20",".1.3.6.1.2.1.2.1.21",
".1.3.6.1.2.1.2.1.22"],String)
nodetypes1 = array(["CONST","CONST","CONST",
"CONST","CONST","CONST","CONST","CONST",
"LINEAR","LINEAR","LINEAR","LINEAR","LINEAR",
"LINEAR","LINEAR","LINEAR","LINEAR","LINEAR",
"LINEAR","CONST","CONST"],String)
values1 = array(["2","Test
Interface","32","1","1","ifPhysAddress","1",
"1","101010","1|1|0","1|1|0","1|1|0","1|1|0",
"1|1|0","1|1|0","1|1|0","1|1|0","1|1|0",
"1|1|0","1|1|0","1",".1.3.6.1.2.1.2.2.1.22"],String)
# The end of the varbinds

###########################################
##################################
# The Actions
def actionfunc0():
    scriptinterface.sendV1Trap("localhost",162,
"public",".1.3.6.1.2.1.11",0,0)

def actionfunc1():
    scriptinterface.insertRow(".1.3.6.1.2.1.2.2",
objectid1,values1,nodetypes1)
# The end of the Actions

# The Condition
if(scriptinterface.getNodeValue(".1.3.6.1.2.1.2.2.1.7",
".1") != "1"):
    actionfunc0()
if(scriptinterface.getRowCount(".1.3.6.1.2.1.2.2") < 5):
    actionfunc1()
# The end of Conditions

Script with condition for execution in device1, condition for action in
device2, action to be executed in device3

In this sample, the network has three devices, with IP address 0.0.0.1, 0.0.0.2, 0.0.0.3 running at port 8001. All three devices, implement the same agent. The script file, device.py is configured on device 1 with IP address 0.0.0.1. This script file will be executed, when a set request is made on the node sysName. If sysName="simulation" in device 1, the node value sysName in device 2 and device 3 is updated with the same value.

The script file device.py is given below:

# The Actions
def actionfunc0():
    scriptinterface.updateValue("0.0.0.2","8001",".1.3.6.1.2.1.1.5",
",0","simulation","CONST")
    scriptinterface.updateValue("0.0.0.3","8001",".1.3.6.1.2.1.1.5",
",0","simulation","CONST")
# The end of the Actions

# The Condition
if(scriptinterface.getNodeValue(".1.3.6.1.2.1.2.1.7","
",1") !="1"):
    actionfunc0()
if(scriptinterface.getRowCount(".1.3.6.1.2.1.2.2") < 5):
    actionfunc1()
# The end of Conditions
Configuring Behavior in TL1 Devices

- Overview
- Enabling/Disabling TL1 Behavior Service
- Configuring Behavior
- Scripts in Examples Directory

Overview

The most important part of simulation is to set up the simulated environment. You can set up the desired environment by defining the behavior of the TL1 Devices, so that they will behave like a real device. The behavior of devices is done through Jython scripts.

The TL1 Agent Simulator provides a script interface, which defines methods to perform various TL1 operations from a JPython script. For example, you can add or delete Access ID, Response Lines and Message Payload to a specific command code, get current Access ID, Message payload values, set acknowledgment and completion code, etc.

The javadocs for the Script API methods is provided under the `<SimulatorHome>/examples/javadocs` directory. The name of the Script API file is `SNMP_TL1_SimScrIntf.html`

Enabling / Disabling TL1 Behavior Service

The Network Simulator provides option to enable/disable the execution of behavior scripts configured for the TL1 agents, at agent runtime. If behavior service is disabled, the agent will not check for script configurations before sending the response. Hence, you can use this option to improve the performance of the TL1 agent, if behavior is not required to be executed.

By default, the service is enabled and the behavior scripts configured (if any), will be executed.

To disable the service,
- Select Run-> Initialize Service in Network Designer.
- The Initialize Service dialog is displayed.
- Deselect "Enable TL1 Behavior service".
- Start the Agent. Now, the behavior scripts configured in the simulator, will not be executed.

**Note**: The option Operations -> Initialize Service will be disabled when the TL1 agent is started.

Configuring Behavior

The TL1 agent simulator, supports to configure request based scripts to simulate real time behavior of agents. Scripts can be configured on a Command Code, Command Code - AccessID or Command Code - AccessId - MessagePayload combination.

The configured scripts will be copied to the configuration directory. It will be executed at run time, when a request is made for the command, before and/or after processing the request.
Follow the steps given below, to configure scripts:

1. Select **Configure -> Behavior Simulation**. The Script Table dialog is invoked. This dialog displays the scripts that are already configured.
2. Click **Add**, to configure a new script. The Configure Script dialog is invoked. This dialog displays the TCS tree on the left-frame. Expand the TCS tree.
3. You have the option to configure the script file at the following levels.
   - To configure script for a command code, select the required command code.
   - To configure script for a Command code - AccessID combination,
     - Select the required command code. The Access ID configured for the selected command is displayed on the right-frame.
     - Select the required Access Id.
   - To configure script for a Command code - Access ID - Message Payload combination,
     - Select the required command code. The Access ID configured for the selected command is displayed on the right-frame.
     - Select the required Access Id. Selecting the AccessID, will display the Message Payload configured.
     - Select the required Message Payload.
4. Specify the condition to execute the script. The following options are available:
   - Before processing request
   - After processing request
   - Before and after processing request
5. Browse and select the script file to be configured. The example scripts are available under `<simulator home>/examples/scripting/tl1` directory. Click **Open** to configure the script file.
6. Click **OK** to add the script configuration in the Script Table.
7. Click **Close** to exit the dialog.

**Scripts in Examples Directory**

The following sample scripts are defined in the `<simulator home>/examples/scripting/tl1` directory. A brief overview on each of the script file is provided below. The javadocs for the Script API methods is provided under the `<SimulatorHome>/examples/javadocs` directory. The name of the Script API file is `SNMP_TL1_SimScrIntf.html`

1. **AddAccessID.py**

   The script file `AddAccessID.py` uses the method `addAccessID()` to add a new AID value to the Command Code, at agent run time. This method takes the Command code and AccessId as its parameters.

   Format:

   ```
   scriptinterface.addAccessId("ED-FAC","ST1-9-1");
   ```

   When this script is executed, the AID value `ST1-9-1` is added for the command code `ED-FAC`. You can replace the sample command code and AID values with your own configuration.

2. **AddLines.py**

   The script file `AddLines.py` uses the method `addLines()` to Add Response Lines to the Command Code at agent run time. This method takes the following parameters as input:
   - `commandCode` - is the CommandCode present in the agent.
   - `accessId` - is the accessId corresponding to the commandCode.
   - `msgPayload` - is the MessagePayload corresponding to the accessId.
   - `isValid` - is a boolean representing whether the line is to be added for a valid response/error response.
• lineType - represents whether the line is quoted/unquoted.
• paramNames - is the list of parameter names available for line to be added.
• paramValues - is the list of values for the parameter names.

Format:
```
scriptinterface.addLines ("RTRV-EQPT","ALL", ",", boolean.booleanValue(), 1, ParamNames,ParamValues);
```

3. AddMsgPayload.py

The script file AddMsgPayload.py uses the method `addMsgPayload()` to add a new Message PayLoad for a commandCode, accessId combination. This method will also add the accessId if it is not present. It takes the following parameters as input:

• commandCode - is the CommandCode present in the agent.
• accessId - is the accessId to be added.
• msgPayLoad - is the MessagePayLoad to be added.

Format:
```
scriptinterface.addMsgPayLoad("ED-EQPT", "ST1-4", "CSU");
```

4. AddAutoLines.py

This script file uses the method `addLines()` to add the Autonomous Lines to the corresponding Alarm Description for a Autonomous Code at agent RunTime. It takes the following parameters as input:

• AutoCode - The Autonomous Code
• AlarmDescription - The Alarm Description for the Auto code.
• LineType - 0 : for Quoted, 1 : for Unquoted
• String[] Parameter Names - the string array of the Parameter Names present for the Autonomous Code
• String[] Parameter Values - the string array of the Parameter Values corresponding to the Parameter Names

Format:
```
scriptinterface.addLines(String AutoCode,String AlarmDescription, int LineType, String[] Parameter Names, String[] Parameter Values);
```

4. DeleteAccessID.py

The script file DeleteAccessID.py uses the method `deleteAccessID()` to delete an accessId for a Command Code at agent run time. This method takes the following parameters as input:

• commandCode - is the CommandCode present in the agent.
• accessId - is the accessId to be deleted.

Format:
```
scriptinterface.deleteAccessId("ED-FAC", "ST1-9-1");
```

When this script is executed, the AID value `ST1-9-1` is deleted for the command code `ED-FAC`. You can replace the sample command code and AID values with your own configuration.
5. DeleteLines.py

The script file DeleteLines.py uses the method `delLines()` to delete an existing Response Line configured for a Command Code, at agent run time. This method takes the following parameters as input:

- `commandCode` - is the CommandCode present in the agent.
- `accessId` - is the accessId corresponding to the commandCode.
- `msgPayload` - is the MessagePayload corresponding to the accessId.
- `responseType` - is a boolean representing whether the line is to be deleted for a valid response/error response.
- `lineType` - represents whether the line is quoted/unquoted.
- `paramNames` - is the list of parameter names available for the line to be deleted
- `paramValues` - is the list of values for the parameter names.

Format:
```
scriptinterface.delLines("RTRV-EQPT","ALL", "", boolean.booleanValue(), 1, ParamNames,ParamValues);
```

You can replace the sample parameter values in the script file with your own configuration.

6. DeleteLineWithValue.py

This script file DeleteLineWithValue.py uses the method `deleteLine()` to delete the response line for the specified Param Name and Value configured for a command code, at agent runtime.

Format:
```
scriptinterface.deleteLine(String commandCode, String accessID, String msgPayload, boolean isValid, int lineType, String uniquePName, String uniquePValue);
```

7. DeleteAutoLines.py

This script file DeleteAutoLines.py uses the method `delLines()` to delete the Autonomous Lines of the corresponding alarm description configured for a Autonomous Code, at agent runtime.

Format:
```
scriptinterface.delLines(String AutoCode , String AlarmDescription, int LineType, String[] Parameter Names, String[] Parameter Values);
```

8. DeleteAutoLineWithValue.py

This script file DeleteAutoLineWithValue.py uses the method `deleteLine()` to delete the Autonomous Response Line of the corresponding Alarm Description with the specified Param Name and Value, at agent runtime.

Format:
```
scriptinterface.deleteLine(String autoCode, String alarmDescr, int lineType, String uniquePName, String uniquePValue);
```
9. DeleteMsgPayLoad.py

The script file DeleteMsgPayLoad.py uses the method `delMsgPayLoad()` to delete a MessagePayLoad for a command code, accesssid combination. This method takes the following parameters:

- `commandCode` - is the CommandCode present in the agent.
- `accessid` - is the accessId corresponding to the MessagePayLoad.
- `msgPayLoad` - is the MessagePayLoad to be deleted.

Format:

```python
scriptinterface.delMsgPayLoad("ED-EQPT", "ST1-4", "CSU");
```

10. GetCurrentAID.py

The script file GetCurrentAID.py uses the method `getCurrentAID()` to GET the accessId that comes in the request. If the accessId is not given in the request, this method returns an empty string.

Format:

```python
AccessID = scriptinterface.getCurrentAID();
```

11. GetCurrentMPB.py

The script file GetCurrentMPB.py uses the method `getCurrentMPB()` to get the msgPayLoad that comes in the request. If the msgPayLoad is not given in the request, this method returns an empty string.

Format:

```python
MsgPayload = scriptinterface.getCurrentMPB();
```

12. GetAllAIDs.py

The script file GetAllAIDs.py uses the method `getAllAids()` to Get all the Access Identifier to the Command Code at agent runtime. This method takes the Command code as the input parameter. It returns the value as a Vector compound.

Format:

```python
scriptinterface.getAllAids("RTRV-EQPT");
```

13. GetAllMPBs.py

The script file GetAllMPBs.py uses the method `getAllMpbs()` to get all the Message Payloads for the specified command code and access identifier at agent runtime. This method takes the following parameters as input. It returns the value as a Vector compound.

- `commandCode` - is the CommandCode present in the agent.
- `accessId` - is the accessId corresponding to the commandCode.

Format:

```python
scriptinterface.getAllMpbs("ED-IP", "ETHER");
```
14. GetNumberOfAIDs.py

The script file GetNumberOfAIDs.py uses the method `getNumberOfAids()` to get the number of access identifiers to the command code at agent runtime. This method takes the following parameters as input. It returns an integer value as the count of the AIDs.

- `commandCode` - the Command Code for which the number of Access Identifiers should be taken.

Format:

```python
scriptinterface.getNumberOfAids("RTRV-EQPT");
```

15. GetNumberOfMPBs.py

The script file GetNumberOfMPBs.py uses the method `getNumberOfMpbs()` to get the number of Message Payloads to the Command Code and the AID at agent runtime. This method takes the following parameters as input. It returns an integer value as the count of the MPBs.

- `commandCode` - the Command Code for which the number of Access Identifiers should be taken.
- `accessId` - the Access Identifier for which the number of Message Payloads should be taken.

Format:

```python
scriptinterface.getNumberOfMpbs("RTRV-EQPT");
```

16. GetLine.py

This script file uses the method `getLine()` to get the Response Lines to the Command Code at agent runtime.

```python
scriptinterface.getLine(String commandCode, String accessID, String msgPayload, boolean isValid, String uniquePName, String uniquePValue)
```

17. GetAgentPort.py

This script file uses the method `getPort()` to get the port number of the agent from which the agent is started.

Format:

```python
scriptinterface.getPort();
```

18. GetIPAddress.py

This script file uses the method `getIPAddress()` to get the IP Address of the agent. This will be very useful for the Network Simulation of TL1 Devices.

Format:

```python
scriptinterface.getIPAddress();
```
19. `getScriptCondition()`

This script file returns the condition (before/after get) at which the script is executing.

Format:

```java
scriptinterface.getScriptCondition();
```

20. `IsAIDExists.py`

The script file `IsAIDExists.py` uses the method `isAidExists()` to check whether the given Access Identifier is present in the AID list for the Command Code. This method takes the following parameters as input. It returns a Boolean value as '0' for False and '1' for True.

- `commandCode` - the name of the Command Code
- `accessId` - the value of the Access Identifier to be checked in the specified Command code.

Format:

```python
scriptinterface.isAidExists("RTRV-EQPT", "ST1-1");
```

21. `IsMPBExists.py`

The script file `IsMPBExists.py` uses the method `isMpbExists()` to check whether the given Message Payload is present in the MPB list for the Command Code and its AID combination. This method takes the following parameters as input. It returns a Boolean value as '0' for False and '1' for True.

- `commandCode` - the name of the Command Code
- `accessId` - the value of the Access Identifier
- `msgpayload` - the Message Payload to be checked for the specified Command code and AID

Format:

```python
scriptinterface.isMpbExists("ED-IP", "ETHER", "165.162.1.20,255.255.255.0");
```

22. `SetAckCode.py`

The script file `SetAckCode.py` uses the method `setAckMessage()` to SET the acknowledgment Message for a commandCode, accessId, msgPayLoad combination.

**Note**: The script API method `setAckMessage()` cannot be configured to be executed with the 'Before processing request' condition.

This method takes the following parameters as input:

- `commandCode` - is the CommandCode present in the agent.
- `accessId` - is the accessId corresponding to the commandCode.
- `msgPayLoad` - is the MessagePayload corresponding to the accessId.
- `ackNo` - is the acknowledgment message to be set.

Format:

```java
scriptinterface.setAckMessage(String commandCode , String accessId , String msgPayLoad , int ackNO);
```

24. `SetComplCode.py`

The script file `SetComplCode.py` uses the method `setComplCode()` to SET the Completion Code(COMPLD,DENY) for a commandCode, accessId, msgPayLoad combination. This method takes the following parameters:
• commandCode - is the CommandCode present in the agent.
• accessId - is the accessId corresponding to the commandCode.
• msgPayLoad - is the MessagePayLoad corresponding to the accessId.
• complCode - is the completion code to be set (COMPLD/DENY).

Format:
```
scriptinterface.setComplCode("ACT-USER" , "ROOT" , "Password" , 1);
```

25. SendAutonomousMessage.py
The script file SendAutonomousMessage.py uses the method `sendAutonomousMessage()` to send the Autonomous Message at agent runtime. This method takes the following parameters as input.
• autoCode - the name of the Autonomous Code. The modifiers should be separated by ^
• almDescr - the value of the Alarm Description for which the Autonomous Message should be sent.

Format:
```
scriptinterface.sendAutonomousMessage("REPT^ALM^EQPT","TextBlock");
```

26. SendAutonomouswithNumberofTimes.py
The script file SendAutonomouswithNumberofTimes.py uses the method `sendAutonomousMessage()` to send the Autonomous Message with specified number of times, at agent runtime. This method takes the following parameters:
• autoCode - the name of the Autonomous Code. The modifiers should be separated by ^
• almDescr - the value of the Alarm Description for which the Autonomous Message should be sent
• numberOfTimes - the number of times the Autonomous message must be sent
• interval - the time interval between the consecutive Autonomous messages.

Format:
```
scriptinterface.sendAutonomousMessage("REPT^ALM^EQPT", "TextBlock", 10, 1000);
```

27. SendDenyMessage.py
This script file uses the method `sendDenyMessage()` to send the deny message at runtime

Format:
```
scriptinterface.sendDenyMessage(String errorCode, String explanation);
```

28. StopTL1Agent.py
This script file uses the method `stopAgent()` to stop the TL1 agent in the network.

Format:
```
scriptinterface.stopAgent(String ipAddress, String snmpPort, String tl1Port);
```
29. StartTL1Agent.py

This script file uses the method startAgent( ) to start the TL1 agent in the network.

Format :

```
scriptinterface.startAgent(String ipAddress, String snmpPort, String tl1Port);
```

30. UpdateParamValue.py

This script file is to update the parameter value of a Response Line configured for a command code at runtime.

uniquePName & uniquePValue indicates the parameter name and value that will be unique across all the lines in the response.

Format :

```
scriptinterface.updateParamValue(String commandCode, String accessID, String msgPayLoad, boolean isValid, int lineType, String uniquePName, String uniquePValue, String pNameToModify, String modifyValue);
```

31. UpdateAutoParamValue.py

This script file is to update the parameter value of a Response Line of the Autonomous code at runtime

Format :

```
scriptinterface.updateParamValue(String autoCode, String alarmDescr, int lineType, String uniquePName, String uniquePValue, String pNameToModify, String modifyValue);
```
Configuring SNMPv3 Security Settings

- Overview
- Security Levels in SNMPv3
- Default Users of SNMPv3 Agent
- Testing the SNMPv3 Agent with noAuth and Auth User
- Testing the SNMPv3 Agent with Priv Users
- Adding New User to the USM Table
- Adding New User to the VACM Tables

**Overview**

The version 3 of SNMP (SNMP v3) is used to provide a secured environment in managing the systems and networks. The Network Simulator supports SNMP v3 and provides option to start the network with SNMP v3 support in the **Settings -> Runtime Settings** dialog. In this case, the agent will respond to the v3 requests in addition to v1/v2c requests.

This section deals with the Security levels supported by the v3 agent, the steps involved in simulating and testing the v3 agent, and adding new users to the USM and VACM Tables.

**Security Levels in SNMPv3**

The SNMPv3 Agent supports the following set of security levels as defined in the **USM MIB (RFC 2574)**:

- **noAuthnoPriv** - Communication without authentication and privacy.
- **authNoPriv** - Communication with authentication and without privacy. The protocols used for Authentication are MD5 and SHA (Secure Hash Algorithm).
- **authPriv** - Communication with authentication and privacy. The protocols used for Authentication are MD5 and SHA; and for Privacy, DES (Data Encryption Standard) protocol is used. For Privacy Support, you have to install some third-party privacy packages. Details about installation is dealt with in the topic "Privacy Support".

The SNMPv3 agent also supports the **VACM MIB** as a default Access Control Model. It consists of four tables. The context name, group name and read/write access for a user are configured in these tables.

**Default Users of SNMPv3 Agent**

By default, the SNMPv3 Agent provides support for three level of users, namely:

- **noAuthUser** - Users with security level noAuthnoPriv and context name as noAuth.
- **authUser** - Users with security level authNoPriv and context name as auth.
- **privUser** - Users with security level authPriv and context name as priv.

The details about the users get stored in the **XML files** under the `<Simulator Home/conf>` directory.
Testing the SNMPv3 Agent with noAuth and Auth Users

Assuming that the simulated network is started with SNMPv3 support, let us see how it can be tested using the default entries available in the USM and VACM Tables.

**User Name "noAuthUser", Security level "noAuth,noPriv"**

The default entry for noAuthUser in USM Table will be as follows. The default Context Name of this user as defined in the VACM Context Table is noAuth. The subtree OID for which read/write access is provided for this user in the VACM View Table is .1.3.6

<table>
<thead>
<tr>
<th>Context name</th>
<th>User Name</th>
<th>Security Level</th>
<th>Auth Protocol</th>
<th>Priv Protocol</th>
<th>Auth Password</th>
<th>Priv Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>noAuth</td>
<td>noAuthUser</td>
<td>noAuth,noPriv</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

To test the agent,

1. Make sure the network is started with SNMPv3 support. Start the MibBrowser application from `<Simulator_Home>/bin` directory.
2. Load the MIB which is implemented in the Agent. For example, load AGENT-SAMPLE-MIB.
3. Click MIB Browser Settings icon in the toolbar. The MibBrowser Settings dialog opens.
4. Choose the SNMP version as v3 in the "General" tab.
5. Select "v3Settings" tab to add the noAuthUser entry in the table. The following details need to be filled:
   - Target Host: localhost (by default)
   - Target Port: 8001.
   - User Name: noAuthUser
   - Security Level: noAuth,noPriv
6. Click Add Entry. The entry gets listed in v3 Settings table
7. Select the entry and click OK to close the MIB Browser Settings wizard.
8. Move on to the MIB Browser Main UI.
9. Give noAuth in the Context Name field of MibBrowser.
11. You find the agent in the network, responding to the SNMPv3 request.

**User Name "authUser", Security level "Auth,noPriv" with MD5 Auth protocol**

The default entry for authUser in USM Table will be as follows. The default Context Name of this user as defined in the VACM Context Table is auth. The sub-tree OID for which read/write access is provided for this user in the VACM View Table is .1.3.6 included and 1.3.6.1.4.1.2162.4.1.1 excluded.

<table>
<thead>
<tr>
<th>Context name</th>
<th>User Name</th>
<th>Security Level</th>
<th>Auth Protocol</th>
<th>Priv Protocol</th>
<th>Auth Password</th>
<th>Priv Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>auth</td>
<td>authUser</td>
<td>Auth,noPriv</td>
<td>MD5</td>
<td>-</td>
<td>authUser</td>
<td>-</td>
</tr>
</tbody>
</table>
To test the agent,

1. Make sure the network is started with SNMPv3 support.
2. Start the MibBrowser application from `<Simulator_Home>/bin` directory.
3. Load the MIB which is implemented in the Agent. For example, load AGENT-SAMPLE-MIB.
   Click MIB Browser Settings icon in the toolbar. The MibBrowser Settings dialog opens.
4. Choose the SNMP version as v3 in the "General" tab. Select "v3Settings" tab to add the noAuthUser entry in the table. The following details need to be filled:
   - Target Host : agent host name
   - Target Port : 8001.
   - User Name : authUser
   - Security Level : Auth,noPriv
   - Auth Protocol : MD5
   - Auth Password : authUser
5. Click Add Entry. The entry gets listed in v3 Settings table.
6. Select the entry and click OK to close the MIB Browser Settings wizard.
7. Move on to the MIB Browser Main UI.

**Testing the v3 Agent with Priv Users**

**User Name "privUser" , Security Level "AuthPriv" with MD5 Priv Protocol**

The default entry for privUser in USM Table will be as follows. The default Context Name of this user as defined in the VACM Context Table is `priv` . The sub-tree OID for which read/write access is provided for this user in the VACM View Table is .1.3.6 included and 1.3.6.1.4.1.2162.4.1.6 excluded.

<table>
<thead>
<tr>
<th>Context name</th>
<th>User Name</th>
<th>Security Level</th>
<th>Auth Protocol</th>
<th>Priv Protocol</th>
<th>Auth Password</th>
<th>Priv Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>priv</td>
<td>privUser</td>
<td>Auth,Priv</td>
<td>MD5</td>
<td>SHA</td>
<td>authUser</td>
<td>privUser</td>
</tr>
</tbody>
</table>

To test the agent, do the following:

- Make sure the network is started with SNMPv3 support.
- Start the MibBrowser application from `<Simulator_Home>/bin` directory.
- Load the MIB which is implemented in the Agent. For example, load AGENT-SAMPLE-MIB.
- Click MIB Browser Settings icon in the toolbar. The MibBrowser Settings dialog opens.
- Choose the SNMP version as v3 in the "General" tab.
- Select "v3Settings" tab to add the privUser entry in the table. The following details need to be filled:
  - Target Host : agent host name
  - Target Port : 8001 (agent port number)
  - User Name : privUser
  - Security Level : Auth,Priv
Adding New User to the USM Table

To add new users to the usmUserTable of the SNMP-USER-BASED-SM-MIB, AdventNet provides a tool called **v3Config Tool**.

To invoke the tool, select `SNMPv3config.bat/sh` file from `<SimulatorHome>/bin` directory or from the Network Designer, by selecting **Settings -> SNMPv3 Security Settings**. By default, the USMUser tab will be selected and the default entries in the table will be displayed as shown in the screen-shot below:

New user entries must be added to the USM and VACM tables before starting and accessing the v3 agent with the required new user names.
Note: To access the v3 agent, the new users must also be configured in the VACM Tables. Please refer the section "Adding new users to the VACM Table" for more details.

To add a new user to the USM Table,

1. Choose USMUser Tab and click on Add from menu bar. Now, the Properties dialog pops up. The following are the information in this dialog.

   - Fill in the "UserName" (can be any name) and "Security Level" in the space provided using the options provided in the combo box as shown in the image.
   - In case, the "Security Level" given is AuthnoPriv, then the Auth Password has to be mentioned. If it is AuthPriv then both Auth Password and Priv Password have to be stated. Here again, the password can be of the user's choice.
   - Auth Protocol can be MD5 or SHA though by default it is MD5.
   - Priv Protocol should be CBC-DES.

   Sample entry would be:
   
   UserName: NewUser
   Security Level: AuthnoPriv
   AuthProtocol: MD5
   AuthPassword: newuser

2. On filling the entries, click on OK button. This adds the new user entry.

Adding New User to the VACM Tables

To provide view based access control, the new user entry has to be added to the VACM Tables of the SNMP-VIEW-BASED-ACM-MIB. The V3Config Tool can be used to configure the VACM entries. The steps involved in adding a new user to the VACM Table using Table view is given below:

   - Adding new user to the VACM Context Table
   - Adding new user to the VACM GroupTable
   - Adding new user to the VACM Access Table
   - Adding new user to the VACM View Tree Family Table
**Adding New User to the Vacm Context Table**

The VACM Context table lists the context names defined for the users. To view the default context names select the **VacmContext tab**. The default context names will be listed as shown in the image below:

1. To add a new context name to the VACM context table, select the **Add** button.
2. Enter the context name in the space provided. It can be of any name. The context name is used as a reference when accessing the v3 agent.
3. Click **OK** to add the new context name to the table.

The next step is to configure the **VACM Group Table**.

**Adding New User to the Vacm Group Table**

The VACM group Table has a set of security to group mappings. If the received context name is valid then the group name is obtained from this table by giving user (security) name and security model as an input. Currently the application supports only the USM model and by default it is specified.

- Model will always have to be USM.
- Security Name can be either 'authUser' or 'noAuthUser' or 'privUser'.
- Group Name can be of any user defined name.
To view the default group names or add new groups in the table,

1. Select the **VacmGroup** Tab. The default groups provided for the default users configured in the USM table are listed as shown in the image below:

<table>
<thead>
<tr>
<th>Security Model</th>
<th>Security Name</th>
<th>Vacm Group Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>noAuthUser</td>
<td>ncAuthGrp</td>
</tr>
<tr>
<td>2</td>
<td>authUser</td>
<td>authGrp</td>
</tr>
<tr>
<td>3</td>
<td>privUser</td>
<td>privGrp</td>
</tr>
</tbody>
</table>

2. To add a new group to this table, click **Add** and enter the following details in the space provided:

   **Security Model**: The Security Model supported by the snmpv3 agent. Currently, only USM is supported.

   **Security Name**: The name of the new user as configured in the USM Table.

   **Vacm Group Name**: The group name to which the user belongs. You can specify any name as group name.

3. Click **OK** to add the entry to the table.

The next step is to configure the **VACM Access Table**.

**Adding New User to the Vacm Access Table**

The VACM Access table lists the access rights and restrictions of the various groups. By giving group name, context name, security model, and security level of the v3 user, the read, write, notify we can provide view name based on the received request type. It is configurable through SNMP.

The group’s access rights are given for,

- read - for retrieval operations - get, getnext and getbulk.
- write - for write operations - set
- notify - for notification operations - trap
To view the default views or add new views in the table,

1. Select the **VacmAccess** Tab. The default views provided for the default users configured in the USM table will be listed.

2. To add a new entry to this table, click **Add** and enter the following information in the space provided:
   - **Vacm Group Name**: The group to which the user belongs as defined in the VACM group table.
   - **Context Prefix**: The context name of the user as defined in the VACM context table.
   - **Security Model**: The Security Model supported by the snmpv3 agent - USM
   - **Security Level**: The Security Level of the user
   - **Context Match**: The Context Match "Exact"
   - **Read View Name**: The read view name provided for the v3 user.
   - **Write View Name**: The write view name provided for the v3 user.
   - **Notify View Name**: The notify view name provided for the v3 user.

3. Click **OK**, to add the entry to the table.

The next step is to configure the **VACM View Tree Family Table**.
Adding New User to the Vacm View Tree Family Table

This table assigns the Sub tree OID that can be accessed by a particular view. To view the default views or add new views in the table,

1. Select the VacmViewTree Tab displayed on the left side on the tool. The default views and sub tree oid provided for the default users configured in the USM table is listed as shown in the image below:

2. To add a new entry to this table, enter the following information in the space provided.
   - **Type**: Select the type as either "included" or "excluded". The type "excluded" denotes that the specified view cannot be accessed by the user.
   - **View Name**: The view name can be any name. Separate view names can be defined for read, write and notify access.
   - **SubTree**: The subtree is the subtree oid in the MIB for which read, write, notify access is allowed for the user.
   - **Mask**: The "mask" field is used to control the elements of the OID subtree that should be considered as relevant when determining the view in which an OID is in. Normally, the OID is included on whole, so you'll need a mask with as many bits set as there are in the OID elements. The default value is "ff". To know more, click on Mask.

3. Click on OK to add the entry to the table.

**Mask**:

The "mask" field is used to control the elements of the OID sub-tree that should be considered as relevant when determining the view in which an OID is in. Normally, the OID is...
included on whole, so you'll need a mask with as many bits set as there are in the OID elements.

Thus, ".1" (the whole dod tree) has one element, so the mask has one bit set - i.e. '80' (in hex). ".iso.org.dod.internet.mgmt.mib-2" has six elements, so six bits set ('fc'). If there are more than eight elements, you specify the longer masks as single octet values, separated by dots (e.g. 'ff.c0' for 10 bits)

The mask value defines how the incoming OID should be matched with the SubTree value. For example, if the subtree value is .1.3.6 and the incoming OID is .1.3.6.1.2.1.1.0, and the Mask is ff(H),(ie) 1111 1111. Then the incoming OID should exactly match .1.3.6. as the first three values are 1 (1 stands for exact match). If the Mask value is 1011 1111, then it is enough if the first and third value i.e 1,6 in the SubTree value match with the incoming OID as 0 does not expect exact match.

After adding the new user entry to the USM and VACM Tables, click on Save to update the changes in the SNMPv3 xml configuration files. The v3 agent can now be accessed by the new user.

To test the v3 agent with the new user, follow the steps specified in Testing the SNMPv3 agent with default users by specifying the new user name, security level, password and context name.
Configuring Run-Time Settings

Overview

Before starting the network, configure the run-time properties that have to be used by the Network Simulation. The options selected in the Runtime Settings dialog, will be automatically saved.

1. Choose Settings -> Runtime Settings in the menu bar.
2. The Network Properties dialog opens, showing the following run-time options that can be configured for SNMP devices.

Run-Time Settings

Logging Level

When the simulated agent is running, it will generate debug information to know about the agent processing. This information will be generated in your log file under <SimulatorHome>/logs directory. The following four levels of debug options are provided:

1. OFF: This level indicates that the debugging mode is disabled. Hence, log messages do not get stored in the log file.
2. NON-CRITICAL: Selecting this option will generate messages about the request made, the response received, trap details, etc., in your log file. Any information that might be required for reference is stored in the log file under this level and may not be useful for debugging errors.
3. CRITICAL: Selecting this option will generate only critical error messages, such as details on Agent crash, Authentication errors, Agent already open in another session, etc., in your log file.
4. DEBUG: Selecting this level will log every activity from entering a method to exiting, including critical and non-critical messages. Hence, selecting this level may affect performance. The debug level should be used only to solve any complex problem.

Log File Name

Specify the name of the log file to be generated under <SimulatorHome>/logs directory. The Log file should be specified only when the debug level is selected. This field is disabled if the debug level is OFF.
Start Network in SNMPv3 version

Network Simulator supports SNMPv3. The details about the default v3 users is stored in XML file format in `<Simulator_Home>/conf` directory. You can add new users to this file using the v3 Configuration tool.

This tool can be invoked from the Network Designer, by selecting Settings -> SNMPv3 Security Settings. Refer to the topic, Configuring SNMPv3 Security Settings to learn more about the default v3 users and the steps to add new users to the v3 agent.

Character Encoding

The network simulator now supports various character encoding other than the UTF-8 encoding format previously supported. The required encoding format must be specified in the Character Encoding option, before starting the network.

For example, if you configure a Jython script which sends a trap including a Japanese word and specify the encoding format as MS932, the simulator can send a trap including an MS932 Japanese word.

The following link contains a list of character encoding formats http://www.iana.org/assignments/character-sets. Network Simulator supports all the encoding formats that are supported in Java language. If Network Simulator does not support the specified format, the default encoding specified in the system is implemented.

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The options selected in the Runtime Settings dialog, will be automatically saved.</td>
</tr>
<tr>
<td>2. After network startup, any changes in run-time settings will not be considered. Network has to be restarted.</td>
</tr>
</tbody>
</table>

Manage using RMI

The Network Simulator provides option to manage the SNMP agent from the remote, using Remote Method Invocation (RMI). Selecting the option Manage using RMI will register the network simulator with the RMI registry.

Refer to the topic Managing Agents via RMI, to learn more.

Scramble the PDU

The Network Simulator supports customization of SNMP PDU by providing access to the request PDU before sending it to the simulator and the response PDU in the buffer, before sending it to the socket layer. The PDU scrambler interface also provides access to the SNMPScriptInterface object, using which the user can access all the behavior API methods. The SNMP Script API is provided under the `<simulator_home>/examples/javadocs` directory.

Refer to the topic SNMP PDU Scrambler, to learn more.
Adding Devices to the Network

- Overview
- Drag and Drop a Device to the Network
- Add Multiple Devices of Same Type
- Add Multiple Devices of Different Types

Overview

You can add devices to an existing network either by drag and drop of devices from the device tree or by adding multiple devices of same type or different types using wizard. Each device in the network will have a unique name, unique ipaddress and port number and unique mac address. The unique device name will be updated as the sysName, unique ipaddress assigned to the devices will be updated in ipAddrTable and the unique mac address of the device will be updated in the ifTable.

Drag and Drop a Device to the Network

You can drag and drop devices from the device tree, into a blank network or to an existing network.

To drag and drop devices into the network

1. Select a device from the Device Tree and drag and drop it into the network area. The default device properties like device type, device name, ip address and port number will be displayed in the General Tab.
   - Device Type: The selected device in the device tree will be added to the network.
   - Device Name: This field is not editable. The device name will be displayed as its unique name. The files and folders which contain the device configuration will be generated with this unique name. The unique name will be updated as the sysName in the system group of RFC1213-MIB for SNMP devices.
   - IP Address: The devices in the network are uniquely identified by the IP address and Port number combination. Specify the device IP address. The IP address assigned to the devices will be updated in ipAddrTable of RFC1213-MIB for SNMP devices.
   - Port Number: The port number will be enabled for the protocols supported by the device. For example, if the device implements SNMP and TFTP, the default port numbers of these protocols will be displayed. Specify the port number at which the device will get started for the protocols supported.

2. If the device supports SNMP, the SNMP Tab is enabled. Selecting this tab displays the following information.
   - Read Community: The default SNMP read community is public. You can specify any other read community name for the device.
   - Write Community: The default SNMP write community is public. You can specify any other write community name for the device.

3. If the device supports CLI, the CLI Tab is enabled. Selecting this tab displays the following information.
   - User name: Specify the new user name to login to the IOS simulation. The default user name is user.
   - Password: Specify the new password to login to the IOS simulation. The default password is password.
4. If the device supports TFTP, the **TFTP Tab** is enabled. Selecting this tab displays the following information.

1. **Execute script on download**: You can configure Jython script files to be executed when a file is downloaded from the TFTP server. This facility can be used to inform the manager about a file transfer, by way of sending traps etc. To configure scripts, browse and select the script (.py file). This script file will be copied in 
   .networks/<networkname>/tftp/<templatedevicename> directory in the name
   TftpDownloadScript.py. The script will be executed whenever a file is downloaded from the TFTP server in the simulator.

2. **Execute script on upload**: You can configure Jython script files to be executed whenever a file is uploaded to the TFTP server. This facility can be used to inform the manager about a file transfer, by way of sending traps etc. To configure scripts, browse and select the script (.py file). This script file will be copied in
   .networks/<networkname>/tftp/<templatedevicename> directory in the name
   TftpUploadScript.py. The script will be executed whenever a file is uploaded to the TFTP server in the simulator.

3. **Root directory**: Configure the TFTP root directory of the device. The TFTP file transfers will be with respect to this root directory.

5. If the device supports Telnet, the **Telnet Tab** is enabled. Selecting this tab displays the following information.

1. **User name**: Specify the new user name to login to Telnet. The default user name is root.

2. **Password**: Specify the new password to login to Telnet. The default password is public.

6. Click **Add** to add the device to the network area. The network area displays the device icon with its IP address and the protocol supported in the devices.

### Note:

1. The Unique Name of the device will be updated as the **sysName** in the system group of **RFC1213-MIB** for SNMP devices.

2. The specified IP address will be updated in **ipAddrTable** of **RFC1213-MIB** for SNMP devices.

3. A unique **mac address** will be assigned to each device and updated in **ifTable->iPhysAddress**.

4. No two devices in the same network will have the same IP address and port number combination. The SNMP, TL1, TFTP, CLI and Telnet port numbers should not clash.

5. The Network Simulator dynamically configures IP addresses when starting each agent instance in the network. Refer to the topic **Configuring IP Addresses** for more information.

### Adding Multiple Devices of Same Type

You can add multiple devices of the same type using the Network Design Wizard. This wizard can be invoked either while creating a new network or to add multiple devices to an existing network.

To add multiple devices of same type, select the option **Tools -> Network Design Wizard** from the menu bar. This will invoke the Network Design wizard. For more information, refer to the topic **Creating a Network with Same Device Type**.
Adding Multiple Devices of Different Type

You can add multiple devices of different type using the Network Design Wizard. This wizard can be invoked either while creating a new network or to add multiple devices to an existing network. To add multiple devices to an existing network, select the option **Tools -> Network Design Wizard** from the menu bar. This will invoke the Network Design wizard. For more information, refer to the topic **Creating a Network with Multiple Device Types**.
Bulk Addition of Scripts

- Overview
- Configuring Bulk Addition of Scripts

Overview

In the topic "Configuring Network Behavior", you learned to set up the desired network environment by defining the behavior of SNMP and TL1 devices in the network. This is done by configuring Jyphon scripts for each device in the network. In some cases, it is necessary to define the same behavior across all the devices in the network. This can be achieved through Bulk Addition of Scripts. This feature enables you to configure the same script file to the specified range of IP address in the network.

Note: This feature is supported only for SNMP Devices.

Configuring Bulk Addition of Scripts

The Network Designer provides an intuitive UI to configure bulk scripts. To invoke the dialog and configure bulk scripts, follow the steps given below:

- Select the option **Tools -> Bulk Script Addition** from the menu bar or from the toolbar icon.
- The **Bulk Addition of Scripts** dialog is displayed. Specify the following options in the dialog:
  - From IP Address: The starting IP Address of the device for which the script must be configured. Example: 192.168.1.10
  - To IP Address: The ending IP Address of the device in the network, till which the script is configured. Example: 192.168.1.20.
  - Script File Name: Browse and select the script file name to be configured in the network devices.
- Condition to Execute the Script: Select the condition based on which the script is executed. The script can be executed when a request is made on a selected variable or at the specified time.
  - Request Based: If the script is request based, browse and select the MIB node based on which the script is executed. Specify the instance of the node, if the selected node is a columnar node. Specify the request based condition. The following conditions are supported:
    - Before Processing Get
    - Before Processing GetNext
    - Before Processing Set
    - After Processing Get
    - After Processing GetNext
    - After Processing Set
  - Timer Based: If the script is timer based, specify the following details: You can specify when you want the scripts to get executed. The following options are specified.
  - Execute scripts when agent starts: The configured script would get executed as soon as agent starts.
- **Execute scripts when agent stops**: The configured script would get executed when the agent stops.

- **Execute scripts at specified time**: You can specify the date and time of script execution. Choose the date and time using the spinner buttons provided. The scripts will get executed at the specified time.

- **Execute scripts after specified time after agent startup**: You can specify the time in hours, minutes and seconds after which the script will get executed after the agent startup time.

- **Periodically execute (once in)**: If scripts are to be executed at specified time or after specified time, periodically, you can check this option. Specify the duration as once in number of hour(s)/day(s).
  - Specify a **Time interval** between each script execution. For example, Timer interval 1000 milliseconds
  - The **number of times** you want the script to execute at the specified time. For example, Number of times 5.
  - Click **Apply**, to add the script.

**Example for Request-Based Bulk Addition of Scripts**

There are 20 devices in network_1 with IP address 192.168.1.1 to 192.168.1.20. The script file Sendv1Trap.py is configured for devices with ip address starting from 192.168.1.10 to 192.168.1.20. The condition to execute the script is "After processing a Get request on node sysUpTime of the RFC1213-Mib.

When a Get request is made on the node sysUpTime of RFC1213-MIB, the script file **Sendv1Trap.py** will be executed on the devices with IP address from 192.168.1.10 to 192.168.1.20.

**Example for Timer-Based Bulk Addition of Scripts**

There are 20 devices in network_1 with IP address 192.168.1.1 to 192.168.1.20. The script file Sendv1Trap.py is configured for devices with IP address starting from 192.168.1.10 to 192.168.1.20. The script is configured to be executed at the specified time. The option to execute the script on a periodical basis is selected with duration as once in 5 days.

The time interval is 5000 milliseconds and the number of times is 10.

Based on the above configuration, the script **Sendv1trap.py** will be executed at the specified time on the devices with ip address from 192.168.1.10 to 192.168.1.20. This configuration will generate 10 traps with a time interval of 5 seconds once in 5 days.
Bulk Modification of IP address, Port and SNMP OIDs

- Overview
- Bulk Modification of IP Address
- Bulk Modification of Port Number
- Bulk Modification of SNMP Values

Overview

The Bulk Modification feature is a very important feature that can improve your productivity. In the topic, Template Devices and Configurations, we learnt that template based bulk modifications can be made on the network devices.

The Bulk Modification option in the Tools menu enables you to do ipaddress based bulk modifications on the network devices. Using this option, you can modify the IP address, Port number and SNMP agent values corresponding to SNMP OIDs across all or selected agents present in the network.

Bulk Modification of IP Address

The IP address of all the devices or a specified range of devices in the network can be modified, either by incrementing the given IP address or by updating the same IP address to all the devices.

To modify the IP address of all the devices,

1. Select Tools -> Bulk Modification option from the menu bar. The Bulk modification dialog is displayed.
2. Select the option IP addresses.
3. Select the range of ipaddress to be modified.
4. Specify the Starting IP Address to be assigned for the first device.
5. To increment the IP address, select the option Increment the IP address and click Modify. All the devices in the network will be incrementally assigned an IP address starting from the ip address that you have specified.
6. To assign the ipaddress from a file, select the option IP address list from file and specify the text file.

Example

- Assume you have added 10 devices using the Network Design Tool wizard with IP address: 190.191.1.1 to 190.191.1.10.
- You wish to change the IP address of the devices in the network. Using the Bulk modification tool if you specify the starting IP address as 190.191.40.21, all the device IP address will be changed incrementally.
- Now the device IP address in the selected network will be 190.191.40.21 to 190.191.40.30.

Note: The IP address of the devices cannot be modified if the network is running.
Bulk Modification of Port number

The port number of all the devices in the network can be modified, either by incrementing the specified port number or updating the same port number to all the devices.

To modify the port number of all the devices, follow the steps given below.

1. Select Tools -> Bulk Modification option from the menu bar. The Bulk modification screen comes up with two options.
2. Select the option Ports.
3. Select the Protocol whose port number is to be modified and specify the Port number to be assigned for the first device.
4. To increment the port number, select the option Increment the Port number and click Modify. All the devices in the network will be incrementally assigned a port number starting from the port number that you have specified.
5. To update the same port number to all devices, select the option Same port number and click Modify. All the devices in the network will be assigned the same port number that you have specified.

Note:
1. The port number of the devices cannot be modified if the network is running.

Example

- Assume you have added 10 devices using the Network Design Tool wizard with SNMP port number 8001 to 8010.
- You wish to change the SNMP port number of the devices in the network, starting with 9001. Using the Bulk modification tool if you select SNMP port specify the starting port number as 9001, all the device SNMP port numbers will be changed incrementally.
- Now the device SNMP port number in the selected network will be 9001 to 9010.

Bulk Modification of SNMP OIDs

To change the value of a particular MIB node (interested value) across all or selected agents present in the network.

1. Select Tools -> Bulk Modification option from the menu bar. The screen comes up with two options.
2. Choose SNMP OIDs.
3. Click Add to display the Bulk Modification dialog. A screen appears to make the changes.
4. Select Starting IP address and Ending IP address from the drop-down box. This is the range where the modification is to be applied.
5. Click Browse button to select the scalar or table column OID that you want to change, from the MIBs displayed. This dialog will display all the MIBs implemented in the network devices. When you have selected the OID, click OK to exit from the dialog.
6. If the selected OID is an index column, you have to specify the instance value of the index column. For example to change the 3rd instance you would specify .3
7. You can modify the MIB node values with the same value for all the devices or by incrementing the specified value or by using $ prefixed macros.
   o Modifying SNMP OIDs with the specific value
   o Select Value to specify a new value in the Value text field. When you specify the value, it will be substituted across all the devices.
   o Select increment if you want the specified value to be incremented for the specified range of ip address / port number.
For nodes of type "string", the specified value will be incremented by appending numbers starting with 1.

For nodes of type ifPhysAddress, the value must be 12 characters using the digits "1 to 9" and characters a to f and must be specified within single quotes followed by h, for example '1486b894f4e3h'.

For nodes of type "Object Identifier", the value specified must be in the OID format, for example '.1.3.6.1.4.1.2162.4.1'.

Modifying SNMP OIDs using $ prefixed macros

Select Macro and choose among the $ prefixed macros listed in the available options.

- $IP: The $IP assigns the agent IP Address for the corresponding OID value, across all or selected agents present in the network.
- $Unique Name: The $Unique name assigns the device unique name for the corresponding OID value, across all or selected agents present in the network.
- $Read Community: The $Read Community assigns the device read community name for the corresponding OID value, across all or selected agents present in the network.
- $Write Community: The $Write Community assigns the device write community name for the corresponding OID value, across all or selected agents present in the network.
- $Community: The $Community assigns the device Read:Write community name for the corresponding OID value, across all or selected agents present in the network.
- $Port: The $Port assigns the agent port number for the corresponding OID value, across all or selected agents present in the network.
- $SYSUPTIME: The $SYSUPTIME assigns the agent sysuptime value for the corresponding OID value, across all or selected agents present in the network.
- $CURRENTTIME: The $CURRENTTIME assigns the agent current time for the corresponding OID value, across all or selected agents present in the network.

You can also prefix or suffix the $ variable with your own values such as 23$No, xxx$Community, $No123 etc.

9. Click OK to add the configuration to the list. When an entry is added to the list, the Modify and Delete options will be enabled.

1. To modify an entry in the Bulk Modification list, select the entry and click Modify. Make the necessary changes and click OK.

2. To delete an entry in the Bulk Modification list, select the entry and click Delete. The entry will be deleted from the list.

10. Click Modify to make the necessary modifications to the OID values.

**Note:** If more than one entry is added to the list, the modifications will be implemented one by one. That is, the first entry will be implemented first, the second entry will be implemented next, and so on.

**Example**

- Assume have added 10 devices using the Network Design Tool wizard with starting IP address: 192.168.1.1 to 192.168.1.10.
- Now you wish to change the value of the sysName to simulatortest in all the devices in the network.
• After completing steps 1 to 7, in step 8, you can specify the value simulatortest.
• Click OK to add the entry and click Finish to change the OID value in all the selected devices.

| Note: Template based modification of device values, traps, informs, scripts and protocol specific parameters can be made from the Template devices in the Template tree. |
Customizing the SNMP PDU in Network Simulator

- Overview
- Enabling the PDU Scrambler feature
- Customizing the request SNMP PDU
- Customizing the response SNMP PDU

Overview

When a request is made by the SNMP Manager, the network simulator creates the response SNMP PDU in a memory buffer and writes the PDU in the socket layer, where the IP and UDP headers get added before sending the packet to the SNMP Manager. The "SNMP PDU Scrambler" supports customizing the request/response SNMP PDU by exposing it to the user for customization.

The PDU Scrambler Interface enables you to,

- Access to the request PDU for customization, before sending it to the simulator for further processing.
- Update the request PDU with error status and error index values, if the incoming PDU is valid.
- Access the SNMPScriptInterface object, for working with behavior API methods before processing the request/response PDU. The api implementation here is equivalent to configuring jython scripts to be executed before/after processing request condition.
- Access to the response PDU for customization, before sending it to the socket layer.
- Update the response PDU with proper error status and error index values, if invalid and send a error response to the manager.

For example, a request SNMP PDU can be customized to send garbled/invalid responses to test the SNMP managers robustness in the event of receiving bad packets.

Enabling the PDU Scrambler Feature

The PDU Scrambler feature will be enabled if the option Scramble the PDU is selected in the Settings -> Runtime Settings dialog. If the network is started from commandline, the -ps <enable/disable> is used to enable or disable the PDU scrambler feature.

Customizing the request SNMP PDU

The method processRequestPDU(SimulatorSnmpPdu snmpPdu) in <Simulator_Home>/examples/pduscrambler/NetsimPduScrambler.java is used to customize the request PDU.

If the option Scramble the PDU is selected, this method will be called with the SnmpPDU object when a SNMP GET, GET NEXT, SET request is made, before sending the packet to the simulator for processing the request. You can add your code inside this method to do the required PDU processing. After the processing, this method returns the modified or scrambled PDU to the simulator.
The prototype of this method is given below:

```java
import com.adventnet.simulator.scripting.SimScrIntf;
import com.adventnet.simulator.netsim.snmp.SimulatorSnmpPdu

public class NetsimPduScrambler {
    /* Describe processRequestPDU method here */
    public void processRequestPDU(SimulatorSnmpPdu snmpPdu) {
        /* add User Code here */
        /**
         * Here user can use the scriptIntf obj ref for working with
         * behavior API
         * The api implementation here is equivalent to configuring jython
         * scripts
         * to be executed before processing request condition
         */
        /* doProcessWithScriptAPI();
        setErrorStatAndIndex(1,1,snmpPdu); */
    }
}
```

After adding the desired code, compile the `NetsimPduScrambler.java` file. The compiled class file must be placed in `<Simulator_Home>/conf` directory.

Refer to `Simulator SNMP PDU Javadocs` for customizing the response PDU. Refer to `Script Interface Javadocs` to access the script API methods.

### Customizing the response SNMP PDU

The method `processResponsePDU(SimulatorSnmpPdu snmpPdu)` in `<Simulator_Home>/examples/pduscrambler/SimulatorPDUScrambler.java` is used to customize the response PDU.

If the option Scramble the PDU is selected, this method will be called with the SnmpPDU object when a SNMP GET, GET NEXT, SET request is made, before sending the packet to the socket layer. You can add your code inside this method to do the required PDU processing. After the processing, this method returns the modified or scrambled PDU to the simulator.

The prototype of this method is given below:

```java
import com.adventnet.simulator.scripting.SimScrIntf;
import com.adventnet.simulator.netsim.snmp.SimulatorSnmpPdu

public class NetsimPduScrambler {
    /* Describe <code>processResponsePDU</code> method here
    *
    * @param snmpPdu <code>SimulatorSnmpPdu</code> value.
    * The response PDU created by the Simulator sent for scrambling
    */
    public void processResponsePDU(SimulatorSnmpPdu snmpPdu) {
        /*
        * add User Code here.
        */
    }
}
```
Here user can use the `scriptIntf obj ref` for working with behavior API.
* The api implementation here is equivalent to configuring jython scripts.
* to be executed after processing request condition

```java
/*
doProcessWithScriptAPI();
if(snmpPdu.getIPAddress().equalsIgnoreCase("192.168.1.129")
&&
   snmpPdu.getPort() == 8001) {
    setErrorStatAndIndex(1,1,snmpPdu);
}
*/
```

After adding the desired code, compile the `NetsimPduScrambler.java` file. The compiled class file must be placed in `<Simulator_Home>/conf` directory.

Refer to Simulator SNMP PDU Javadocs for customizing the response PDU.
Refer to Script Interface Javadocs to access the SNMP and TL1 script API methods.
Configuring IP addresses

- Overview
- Creating and Deleting Virtual IPs from a batch/shell script
- Manual Configuration of Virtual IP Address

Overview

When you create and start the network, the simulator will dynamically configure the specified IP address of the devices in your system. For example, if you create a network with IP address 200.201.202.1 to 200.201.202.10, the IP address will be dynamically configured in the system when each agent instance is started. This facility is supported in Windows NT, 2000, XP, Linux and Solaris OS, when the system is started with Admin user access.

NOTE: Do not assign addresses that are already used by existing devices in your network. The IP addresses you give to the simulated devices must be unique within the network on which the simulation is running.

Creating and Deleting Virtual IPs from a batch/shell script

To create virtual IPs from a batch/shell script:

- Run CreateVIP.bat/sh file from <SimulatorHome>/bin/cmdline directory.
- Specify the starting IP address and the number of devices for which the virtual IPs must be created and click OK. The specified number of virtual ips will be created.

Syntax: CreateVIP.bat <Starting IP Address> <No. of IP Addresses to be created>
Example: CreateVIP.bat 192.168.1.1 100

To delete virtual IPs from a batch/shell script:

- Run DeleteVIP.bat/sh file from <SimulatorHome>/bin/cmdline directory.
- Specify the starting IP address and the number of virtual IPs to be deleted and click OK.
  The specified number of virtual ips will be deleted from the system.

Syntax: DeleteVIP.bat <Starting IP Address> <No. of IP Addresses to be deleted>
Example:DeleteVIP.bat 192.168.1.1 100

Manual Configuration of Virtual IP address

The links given below will guide you to configure the virtual ip address, manually. Select the appropriate links based on the OS used.

- Configuring Virtual IP Address in Windows NT
- Configuring Virtual IP Address in Windows XP/2000/ ME/2003
- Configuring Virtual IP Address in Bulk under Windows 2000
- Configuring Virtual IP Address in Windows in DHCP mode
- Configuring Virtual IP Address in Linux
- Configuring Virtual IP Address in Solaris
Configuring Virtual IP Address in Windows NT

The procedure to configure multiple virtual IP addresses in Windows NT is given below. This procedure can be performed only by an user with admin privilege.

1. Click on the Start Menu on the Taskbar and choose Control Panel.
2. Double-click Network among the components displayed in the Control Panel. This opens up a Network dialog with 5 tabs.
3. Choose the Protocols tab and click on TCP / IP Protocol from listed protocols.
4. Click on Properties button when TCP / IP Protocol is selected.
5. This brings up the Microsoft TCP / IP properties dialog; choose IP Address tab.
6. Click the Advanced button; this will bring the Advanced IP addressing with the Adapter Name shown above the configured IP addresses in your system.
7. Click Add to display the TCP/IP Address dialog. Specify IP Address and the corresponding Subnet Mask.
8. Re-start the system for the changes to take effect.

Configuring Virtual IP Address in Windows XP/2000/ ME/2003

The procedure to configure virtual IP addresses in bulk under Windows XP/2000/ME is given below. This procedure can be performed only by an user with admin privilege.

1. Click on the Start Menu and choose Settings.
2. Among the listed items select Network and Dial-up connections.
3. Select Local area connection from the items under Network and Dial-up connections; this opens with the general information on Local area connection status.
4. Click on the Properties button. The Local area connection properties dialog opens listing all the protocols.
5. Choose Internet Protocol ( TCP / IP ) and click on the Properties button.
6. The Internet Protocol TCP / IP Properties dialog is invoked. Click on the Advanced button at the bottom of the dialog. The Advanced TCP / IP settings dialog displays all the configured IP addresses in your system.
7. Click Add button next to the displayed IP addresses to add an IP address with a corresponding sub-net mask.
8. You will have to re-start the system for the changes to take effect.

Configuring Virtual IP Address in bulk under Windows 2000

The procedure to configure virtual IP addresses in bulk under Windows 2000 is given below. This procedure can be performed only by an user with admin privilege.

1. Select Start -> Run and type regedt32 in the run dialog ( or you can type it on the DOS prompt).
2. This will bring up the Registry Editor - HKEY_LOCAL_MACHINE on Local Machine screen.
3. Choose the HKEY_LOCAL_MACHINE on the tree displayed on the left hand side.
4. Choose the path : HKEY_LOCAL_MACHINE -> System -> CurrentControlSet -> Services -> NetBT -> Parameters -> Interfaces. Here, you will find a series of Tcpip_{labels} under Interfaces, an example is shown below:
   Tcpip_{2B34C0C3_FCA8_486C_8559_989429901786}
5. Select all the Tcpip_{labels} listed to check if the right hand side data content which has Netbios Options:REG_DWORD :0x1 and make a note of this Tcpip_{ label }.
6. Search for this label under HKEY_LOCAL_MACHINE -> System -> CurrentControlSet -> Services -> Tcpip -> Parameters -> Interfaces. Here, you will find the same Tcpip_{label} (that you had noted down in point 5). When you select this, the right side data content will display IP address and properties.
7. Double click on the IP address field. This opens a Multi-String editor where the virtual IP addresses you want have to be keyed in. You can cut and paste to minimise the time for large number of IPs.
8. When you have finished entering the IP addresses, select the subnetmask field (255.255.255.0) in the right hand side. This opens a Multi-String editor where the subnetmask you want have to be keyed in. Enter the same number of subnetmasks(255.255.255.0) as the number of IPs.
9. You will have to re-start the system for the changes to take effect.
10. Now you should be able to ping these virtual IP addresses. Also, make sure that there are no conflicts in the network (i.e. the virtual IP you have configured is not already being used by some other user).

Configuring Virtual IP Address in Windows in DHCP mode

The procedure to configure virtual IP addresses in windows with DHCP mode is the same as Configuring Virtual IP Address in bulk under Windows 2000.

Configuring Virtual IP Address in Linux

1. Invoke the linuxconf tool. (type linuxconf in the shell prompt and press Enter). For starting this tool you have to be logged in as Super user.
2. In the displayed menu, select the menu item: IP Aliases for virtual hosts and press Enter. This will display the two interfaces eth0 and lo. (Ethernet interfaces in Linux are called by such names as eth0 and eth1)
3. Select eth0 and press Enter. This will show all the IP aliases configured for the selected interface.
4. In the field IP alias or range, enter the desired IP address or the range with a corresponding Netmask.

Example
IP alias or range : 177.177.177.1-25
Netmask : 255.255.255.0

Note:
1. These commands have to be executed with the super user privilege.
2. You will have to re-start the system for the changes to take effect.
3. When you are configuring a large number of IP address, it might take a long time to boot up.

Using the following command, we can configure Virtual IPs in the Linux box at run-time without re-booting the machine or re-starting the network.

ifconfig eth0:1 172.19.1.2 netmask 255.255.255.0 up
ifconfig eth0:3 172.19.1.3 netmask 255.255.255.0 up

The interface number must be unique for each interface (for example eth0:1, eth0:2 etc). The same interfaces can be made down by using the following command.

ifconfig eth0:1 172.19.1.2 netmask 255.255.255.0 down
ifconfig eth0:3 172.19.1.3 netmask 255.255.255.0 down

Note:
1. These commands have to be executed with the super user privilege.
2. It can be used in versions up to 7.2
3. The configurations will not be available when you re-start your system.

Configuring Virtual IP Address in Solaris
Follow the three simple steps that will help you to configure virtual IP address:

1. Edit the /etc/hosts file and add additional IP addresses, along with the associated hostname. For example: To add 192.168.0.1 (web1) and 192.168.0.2 (web2) to the system, enter in the following manner.
   
   192.168.0.1 web1
   192.168.0.2 web2

   The system needs to be configured for the newly added IP addresses.

2. Create the startup files so that the system will automatically add the new IP addresses upon bootup.
   
   a. Create new files in the /etc directory. The file names should follow the following convention according to the new ip address added.

   hostname.hme0:1
   hostname.hme0:2

   b. The contents of these files will be a single line with the respective hostname entered. For example:

<table>
<thead>
<tr>
<th>File Name</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostname.hme0:1</td>
<td>web1</td>
</tr>
<tr>
<td>hostname.hme0:2</td>
<td>web2</td>
</tr>
</tbody>
</table>

   The file "hostname.hme0:1" will contain "web1" and "hostname.hme0:2" will contain "web2".

   If you look in the "/etc" directory, you will find a file named "hostname.hme0" (which is the real interface, the :n represents the virtual interface). In this file, you will find the name of your system. If you look in the "/etc/hosts" file, you will find your IP address.

   Now you have configured virtual IP address for two new interfaces that you have added (.hme0:1 and .hme0:2).

3. To activate this virtual IP address, you have to provide the following commands:

   # ifconfig hme0:1 plumb
   # ifconfig hme0:2 plumb

   The system startup in Solaris searches the "/etc" directory for files named "hostname.*". When it finds one or more of these files, it reads the hostname from them, looks up the hostname in the "/etc/hosts" file for a match, and if found, configures the interface with the IP address of the match.

   Please **re-boot your system** for the virtual IP address to be configured in your system.

   (OR)

   To configure the interface **without re-starting the system**, use the "ifconfig" command. To add the two IP addresses, type:

   # ifconfig hme0:1 192.168.0.1 up
   # ifconfig hme0:2 192.168.0.2 up

   After having issued these two commands, the system would recognize the new IP addresses immediately, without re-booting the system.
Topology Editor

- Overview
- User Interface
- Establishing Router to Router Connections
- Establishing Router to Device Connections
- MIB Data Used

Overview

The Network Simulator provides the Topology editor to establish inter-connections across routers, switches and other network devices and visualize the topological relationship between the devices. You can connect new devices or disconnect existing devices in the topology.

User Interface

The "Topology view" button is displayed in the top panel of the network area and is enabled when at least one network is loaded in the network designer. Selecting the button, displays the topology panel with the router devices in the network, displayed on the left-side tree and the connection between the routers, displayed on the right.

Selecting a specific router in the left-side tree, displays the devices connected to the router. A panel containing a list of devices in the network, that are not connected to this router, is displayed on the right side. A combo box to filter the list of devices based on the ip address is also present.

You can toggle between the "Network view" and "topology view". When a new device is added to the network or any changes are made to the network, select the Refresh button available at the top right-corner of the Topology view, to refresh the devices.

Establishing Router to Router Connections

The Topology Editor enables you to create connections between the router devices in the network or disconnect existing connections between the routers.

To create connections between routers,
- Select the "Routers" node in the left-side tree. The existing router topology will be displayed on the right-side.
- To connect a router to another router, right-click on the router and select "Connect to". The ip address of the other routers in the network will be listed.
- Select the ip address of the router to which the connection must be established and click OK. The connection between the two routers will be created.
- To disconnect the connection, right-click on the router device and select Disconnect.

Establishing Router to Device Connections

To connect a new device to the router,
- Select a router from the left-side tree. The topology of the devices connected to the router are displayed with the interconnection lines displayed between the devices.
- To connect a new device, drag and drop the device from the right-side frame, to the router/switch.
• When dragged and dropped on a particular router, the interface details of the router are displayed. If connection is made to a switch, the ports available are displayed in addition to the list of interfaces.
• Select the interface/port number to make the connection and click OK. The device is connected to the router/switch. When a device is connected, the IpNetToMediaTable (OID: ".1.3.6.1.2.1.4.22") is updated. If all the interfaces are occupied, the device is not connected.
• If the device is connected to an ethernet interface, the link is displayed in blue color.
• If the device is connected to a serial interface, the link is displayed in Green color
• If the device is connected to a switch, the link is displayed in red color with the port number displayed in the middle.

**Note:** When a Router/Host is connected to switch port, the MAC Address of the Routers(Interface) / Host will be updated in the dot1dTpFdbTable of the switch.

To disconnect a device,
• Right-click on a device in the topology and select Disconnect. When a device is disconnected, the IpNetToMediaTable (OID: ".1.3.6.1.2.1.4.22") is updated.

**Note:** When a switch is disconnected, the devices connected to the switch will not be disconnected.

**MIB Data used**

The MIB data used in the Topology Editor is given below:
• The ipaddress of the devices connected to a router/switch is updated in the RFC1213-MIB -> IpNetToMediaTable with OID ".1.3.6.1.2.1.4.22".
• The MAC address of the Routers/Host device connected to a switch, will be updated in the dot1dTpFdbTable of the switch.
• The interface details of the routers are displayed from the RFC1213-MIB -> IfTable with OID ".1.3.6.1.2.1.2.1.2.1.2"
• The port details of the switch device are displayed from Bridge MIB -> dot1dBasePortTable with OID ".1.3.6.1.2.1.17.1.4".
Editing Operation of Devices

The most common of operations that make daily work easy are also available in Network Design tool. The Copy and Paste options are available for all the network components.

Copy

The devices that are to be duplicated can be copied and pasted inside the same network.

To copy a device, select the device, right-click and select the copy option or choose Edit -> Copy option from the menu bar.

Paste

This option is enabled only after a copy operation. The device that is copied can be pasted only in the same network.

To paste a device

- Click on the paste icon or select the Edit -> Paste menu item in the menu bar. While using the menu bar or toolbar option to paste a device, the position to paste has to be indicated with the mouse click.
- The Device Properties dialog pops up. A device that is duplicated needs the Port number or IP address to be changed because two devices with same IP address and Port number cannot run in a network. Hence, provide a unique IP address and port combination.
Find a Device in the Network

The **Search Device** option in the **Edit** menu, will search for a device in the network, with the given IP address or with the specified device name. Only one device will be found at a time. Use the **Find Next** button to find the other devices.

To find a device in the network,

- Click on the **Find** icon or choose **Edit --> Search Device** option in the menu bar. You can also use the Ctrl + F shortcut keys to bring up the Find dialog.

- You can find a device in a network by,
  - **IPAddress** :
    
    A device in the network is uniquely identified by its IP address. Specify the IP address to search. For example, 192.168.224.1
  
  - **Device Name** :
    
    You can also specify the unique device name to search a specific device. For example, SunSolaris5.
Viewing Device, Template and Network Info

- **Device Info**

  You can view the properties of individual devices in the network from the Network area in both Network View and List View.

  To view the device properties:
  - Right-click on a device in the Network area.
  - Select **Device Info** option.
  - The Properties dialog will display the following properties of the device:
    - The MIBs implemented by the device
    - The name of the device
    - The device IP address
    - The SNMP, TL1, TFTP, CLI and Telnet port numbers, based on the protocols supported by the device
    - The Read and Write community of the SNMP agent
    - The TFTP properties

- **Template Info**

  You can view the properties of template devices listed in the Template tree. To view the template tree, select the menu option **View -> Tree View -> Show Template Tree**.

  To view the Template device info,
  - Right-click on a Template device in the Template tree.
  - Select **Template Info** option.
  - The TemplateInfo dialog will display the following properties of the device:
    - The devices in the network area with this template.
    - The protocols supported by the device and the properties of each protocol.

  For more information on template devices and template device configurations, refer the topic **Template Devices and Bulk Configurations**

- **Network Info**

  You can view the network information like the total number of agents in the network, the agents started, the idle agents and information about each device type. To view the Network details,
  - Right-click on a blank network area and select **Network Info** from the right-click options.
  - The **Network Info dialog** displays the following information:
    - Total number of agents in the network and their breakup based on the protocol supported.
- Number of agents started and their breakup based on the protocol supported.
- Number of agents idle (not started) and their breakup based on the protocol supported.
- Number of devices of each type and the protocols supported. The protocol supported is denoted by a 'x' mark.
Modifying Device Information

- Overview
- Modifying General Device Properties
- Modifying Network Device Configurations

Overview

The devices once added to the network might need some modifications to be made. These changes can be made easily.

Modifying General Device Properties

To modify the IP address, port number, and community of the devices in the network area,

1. Double-click on the device in the network area or right-click on the device and choose Device Info. This will bring up the Device Properties dialog.
2. Make the desired changes to IP address, Port number, or community.
3. Click OK to save changes.

Modifying Network Device Configurations

You can modify the SNMP and TL1 configuration of the devices in the Network. Refer to the topic Configuring Devices in the Network, to learn more about the configurations.
Deleting Devices in the Network

- Deleting Individual Devices
- Deleting Multiple Devices

You can delete a specific device or multiple devices from the Network area.

Deleting Individual Devices

To delete a specific device in the Network, from the Network area,

1. Right-click on the device that you want to delete from the Network area.
2. Choose the option Delete. The device and device configurations will be deleted.

Note: You can also delete a device using the Delete button in your system keyboard.

Deleting Multiple Devices

To delete multiple devices from the network area,

1. Use Ctrl + arrow key to select the devices to be deleted. The selected devices will be highlighted.
2. Press Ctrl key and right-click on any of the selected device.
3. Select the right-click menu option Delete Devices, to delete the selected devices. The Confirm Deletion dialog is displayed. Click Yes to delete the devices.
Deleting the Network

You can delete a network from the Network designer UI. Deleting a Network deletes all the files and directories created while saving the network. All the database tables created for the network is deleted.

Before deleting a network, the network must be unloaded from the network area. To delete a network, follow the steps given below:

- Unload the network loaded by selecting Network -> Close option.
- If all the loaded networks are unloaded then you will see the 'Main Screen' which displays the link to delete the network
- Select Delete Network option from the Intro screen of the Network Designer.
- Browse and select the <networkname> in the ./networks directory.
- Click Delete. You will be prompted with a Confirm Deletion dialog. Click Yes to delete the network files and directories of the specified network.
Saving the Network

- Overview
- Saving the Currently Open Network
- Saving all Open Networks

Overview

When a device is added to the device tree or when a network is created it is automatically saved. Similarly, any template based device configurations or load and unload of mibs are automatically saved by the simulator. All the network configurations will be saved in the path in which the new network was created. To learn about the Network directory structure, refer the topic, Network Directory Structure.

The Save option is invoked only when device configurations, trap/inform configurations and behavior configurations are made on individual devices in the network area.

Saving the Currently Open Network

To save the currently open network,

1. Click on the save icon or select Network -> Save.
2. The network will be saved with the specified network name under <SimulatorHome>/networks directory.

Saving all Open Networks

If you have more than one network open, and you wish to save all the networks,

1. Click on the Save All or choose the Network -> Save All option.
2. It prompts to save all the open networks with the specified network name under <SimulatorHome>/networks directory.
Starting and Stopping the Network

- Starting the Network from UI
- Starting the Network from Command Line
- Operations Permitted on a Running Network
- Stopping the Network

The Network that is designed through the Network Designer can be started from the UI or started from the command line after creating the configuration file in the Network Designer tool. The Network Simulator provides the facility to dynamically configure IP addresses when starting each agent instance in the network. This facility is supported in Windows NT, 2000, XP and Linux/Solaris OS, when the system is started with Admin user access. Refer to Configuring Virtual IP Address, for more information.

Starting the Network(s) from the UI

To starting the network(s) from the UI

1. Click icon or choose menu bar Run -> Start Network.
2. If only one network is loaded in the network designer, all the devices in the network will be started. If more than one network is loaded, the Select the Networks dialog opens, showing all the Currently open networks.
3. Select the networks that you want to start and move it to Start simulator with networks list using the >> button. To deselect a network from this list, use the << button.
4. Click OK to start the network. All the devices in the network will be started at the respective ports and ip address. For example, if a device supports SNMP, TFTP and CLI, the SNMP agent will start at the specified SNMP port, the TFTP agent will start at the specified TFTP port and the CLI agent will start at the specified TFTP port.

Starting the Network From Command Line

To start the network from Command Line, refer to the topic, Starting Multiple Network from Command Line.

Note: You can start any other network, when a network is currently running.

Operations on a Running Network

All operations cannot be performed on a running network. The following table displays the operations that can be performed and operations that cannot be performed on the running network:

<table>
<thead>
<tr>
<th>Operations Permitted</th>
<th>Operations Not Permitted</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Adding a device to the network and starting the device.</td>
<td>• Deleting a device from the network</td>
</tr>
<tr>
<td>• Copy - Paste a device in the network</td>
<td>• Changing properties of a device</td>
</tr>
<tr>
<td>• Configuration of Agent Values.</td>
<td>• Loading/Unloading MIBs/TCS</td>
</tr>
<tr>
<td>• Recording from a Real SNMP Agent</td>
<td></td>
</tr>
</tbody>
</table>
Operations Permitted | Operations Not Permitted
---|---
• Behavior Scripts Configurations
• Trap and Autonomous Message Configurations
• Configuring Echo requests for TL1 devices

**Stopping the Network**

To stop the network, click the stop icon or choose Run -> Stop Network. To stop the network in command line use Ctrl+C. All the devices in the network will be stopped at their respective port numbers.

**Note:** When more than one network is running, the Stop Network option will stop all the networks.
Starting and Stopping Network Devices/Agents

Overview

The Network Simulator provides option to start and stop individual agents in the network from the UI or from a script file at runtime. You can also change the interface status of the device.

The Network Simulator provides the facility to dynamically configure IP addresses when starting each agent instance in the network. This facility is supported in Windows NT, 2000, XP and Linux/Solaris OS, when the system is started with Admin user access. Refer to Configuring Virtual IP Address, for more information.

Starting Agents in the Network - From UI

The Network Designer provides options to start individual/multiple/all agents in the network with specific protocols or all the supported protocols.

1. To start a device/agent with all the protocols supported by the device:
   - Right-click on a device in the network tree or network area
   - Select **Start Device**. This will start the device at the specified port numbers. For example, if a device supports SNMP, TFTP and CLI, the SNMP agent will start at the specified SNMP port, the TFTP agent will start at the specified TFTP port and the CLI agent will start at the specified TFTP port.

2. To start a specific protocol in the device:
   - Right-click on a device in the network tree or network area
   - Select the **protocol** name supported in the device and click **Start**. For example, if your select SNMP -> Start, the SNMP agent will start at the specified SNMP port number.

3. To start all the agents in the network will all the protocols supported:
   - Right-click on the blank network area
   - Select **Start All Agents**

4. To start all devices in the network with the specified protocol:
   - Right-click on the blank network area
   - To start all SNMP agents, select **Start All SNMP Agents**
   - To start all TFTP agents, select **Start All TFTP Agents**
   - To start all CLI agents, select **Start All CLI Agents**
   - To start all TL1 agents, select **Start All TL1 Agents**
   - To start all Telnet agents, select **Start All Telnet Agents**

5. To start multiple devices in the network:
   - Use Ctrl + arrow key to select the required devices. The selected devices will be high-lighted.
   - Press Ctrl key and right-click on any of the selected device.
   - Select the right-click menu option **Start Devices**, to start the selected devices.
Start Agents From Script file

To start individual/multiple SNMP and TL1 agent in the network from the script file, the method `startAgent()` in the script interface can be used in the script file.

The Script API methods are provided under the `<SimulatorHome>/examples/javadocs` directory. For an SNMP device, the agent will be started at the specified SNMP port. For a TL1 device, the agent will be started at the specified TL1 port. If the device implements both SNMP and TL1, the agent will be started at the specified SNMP and TL1 ports.

Stopping Agents From UI

The Network Designer provides options to stop individual/multiple/all agents in the network with specific protocols or all the supported protocols.

1. To stop a device with all the protocols supported by the device:
   - Right-click on a device in the network tree or network area
   - Select Stop Device. This will stop the device at the specified port numbers. For example, if a device is started at SNMP, TFTP and CLI ports, the Stop device option will stop the agents running at SNMP port, TFTP port and CLI port.

2. To stop a specific protocol in the device:
   - Right-click on a device in the network tree or network area.
   - Select the protocol name supported in the device and click Stop. For example, if the SNMP agent is started, and you select SNMP -> Stop, the SNMP agent will stop running at the specified SNMP port number.

3. To stop all the agents in the Network
   - Right-click on the blank network area
   - Select Stop All Agents

4. To stop all devices in the network with the specified protocol
   - Right-click on the blank network area
   - To stop all SNMP agents, select Stop All SNMP Agents
   - To stop all TFTP agents, select Stop All TFTP Agents
   - To stop all CLI agents, select Stop All CLI Agents
   - To stop all TL1 agents, select Stop All TL1 Agents
   - To stop all Telnet agents, select Stop All Telnet Agents

5. To stop multiple devices in the network
   - Use Ctrl + arrow key to select the required devices. The selected devices will be high-lighted.
   - Press Ctrl key and right-click on any of the selected device.
   - Select the right-click menu option Stop Devices, to stop the selected devices.

Stopping Agents/Devices From Script file

To stop individual SNMP and TL1 agent in the network from the script file, the method `stopAgent()` in the script interface can be used in the script file.

The Script API methods are provided under the `<simulatorhome>/examples/javadocs` directory.
Loading and Unloading a Network

Loading a Network

When you create a new network in the Network Designer tool or record a real network in the Network Recorder tool, it will be saved in <SimulatorHome>/networks directory with the name of the network.

To load a network,
- Select Network -> Open option from the menu bar.
- The Load Network dialog appears. Browse and select the network name.
- Click Open to Load the Network, along with its configuration.

Note: Only the network that is currently loaded in the Network Designer will be loaded in the network area. All the loaded networks, will be displayed in the Select a Network combo-box provided in the top panel of the network area on the left side.

To load a network that is already added in the Select a Network combo-box, you have to just select the required network from the list of networks displayed here.

Unloading a Network

Unloading a network using the Network -> Close option, will unload the existing network and add it to the Select a Network combo-box provided in the top panel of the network area on the left side.
Porting Your Network

Overview

The Network Designer allows you to package the network that you have created with the desired configurations and install it in any other system having Simulation Toolkit. For example, you have created the network in Linux or Solaris system with all the desired configuration. This network and its configuration can be packaged and sent across for demo or trade shows or kept as a backup.

You can also package the required devices added to the device tree and install it in any other system having Simulation Toolkit. In this topic, you will learn about the options available in Network Designer to package and install your network and devices.

Packaging Your Network

To package your network

- Choose Tools --> Package from the menu. The Package Network dialog opens up.
- Select Package Network option.
- Click Add and select the networks that you have created earlier. You can choose all the network you want to package one by one.
- Click Browse to select the location of where this zip file (packaged network) should be stored and provide a name for the zip file.
- Click OK to package the network.

Packaging the Devices in the Device Tree

To package your devices in the device tree,

- Choose Tools --> Package from the menu. The Package Network dialog opens up.
- Select Package Devices option.
- Click Add to display the list of devices added to the device tree. You can choose the required devices you want to package, one by one.
- Click Browse to select the location of where this zip file (packaged devices) should be stored and provide a name for the zip file.
- Click OK to package the devices.

Installing the Network/Devices

You can install the network or the devices that you packaged, in another system having this product installed.

- Select Tools -> Install menu item from the menu bar. This brings the Install Networks dialog.
- Choose the option of whether you want to just install the package or Install and load the package.
- Browse to select the packaged zip file.
• Click **OK** to complete the operation.

If you had chosen the **Install Package** option, it will install the network /devices. The installed devices will be displayed in the device tree under the first group.

If you had chosen **Install and Load Package**, the network(s) will be installed and loaded in the Network Design Tool.
Automate the Network

- Overview
- Actions that can be Automated
- Steps to Automate the Network
- Running the Automated Network

Overview

The Network Automation Wizard gives you the ability to automate the most common tasks performed by the network devices. You can configure tasks to be performed on any device in the network and set triggers to instigate each task. Based on the tasks configured, the simulator automatically generates scripts that schedule and trigger the activities. The automated network can be started from command line.

This feature eliminates repetitive jobs and is useful in test case automation. Scripts are automatically generated and hence no knowledge on scripting is required to automate the network.

Actions that can be Automated

Start Agent

The "Start Agent" action can be configured on a device to be executed periodically on a daily or hourly basis. The agent will be started at the specified time or the current time when the network automation is started and will repeatedly get executed on a hourly/daily basis.

Stop Agent

The "Stop Agent" action can be configured on a device to be executed periodically on a daily or hourly basis. The agent will be stopped at the specified time or at the current time when the network automation is started and will repeatedly get executed on a hourly/daily basis.

Send Traps

The "Send Trap" action can be configured on a device to be executed periodically on a daily or hourly basis. Only the traps that are configured in the Trap Configuration wizard can be automated. In case of multiple traps, the traps are identified based on the Associated Variable.

The traps will be sent at the specified time or at the current time when the network automation is started and will repeatedly get executed on a hourly/daily basis.

Steps to Automate the Network

The steps to automate the given network:


2. Click Add to add the details of the actions to be performed. The Automation Wizard is displayed wherein you specify the actions to be performed, the time-interval and the device for which the configuration is made.
   - Select the required Action to be executed from the Action list box.
Select the **Repeat parameter** as either to repeat the actions hourly or daily and specify the time at which the action has to be performed.

Select the **device name** from the list. All the devices in the network are displayed here.

If the **Send Trap** action is selected, the associated variable of all the traps configured in the Trap Configuration dialog is listed. Select the required **Associated Variable**.

Click **OK** to add the entry to the list. The text file in the name `<networkname>.txt` will be generated in the `<SimulatorHome>/networks/<network name>` directory. This file contains the configuration details. The same text file can be reloaded by the automation dialog and hence the actions can be changed as required.

3. To modify an entry in the list, select the entry to be modified and click **Modify**.

4. To delete an entry from the list, select the entry to be deleted and click **Delete**.

**Running the Automated Network**

To run the automated network simulator execute `StartAutomatedNetwork.bat/sh` file in `<SimulatorHome>/bin/cmdline` directory, with the following options. All the options are mandatory

- `-net <network name>`: The Network name in `<SimulatorHome>/networks/` directory, that has to be started.
- `-txt <Auto Text file>`: The text file in `<SimulatorHome>/networks/<network name>` which defines the way in which the network has to function
- `-time <specified/current>`:
  - **specified** - If the actions are to be executed according to the timings specified in the text file,
  - **current** - Considers the present time as the time for executing the first action and executes all the actions continuously.

**Note:**

1. When the automated network is started, the text file is parsed and two script files are generated in `<SimulatorHome>/networks/<network name>` directory. These script files will be executed by the automated network.
   - `<networkname>_day.py`: This script file will be executed by the automated network for the daily activities.
   - `<networkname>_hour.py`: This script file will be executed by the automated network for hourly activities.

2. Both the script files will be executed in separate threads.

3. In case of traps, only the traps configured in the UI can be automated.
Starting Multiple Network from Command Line

- **Overview**
- **Command Line Options**
- **Starting Network Simulation in Windows**
- **Starting Network Simulation in Linux and Solaris**

**Overview**

The command line Network Simulator is purely NON-GUI and uses only character user interface for obtaining various inputs for the simulator. It can be invoked by executing the `StartNetwork.bat/sh` in `<Simulator Home>/bin/cmdline` directory. Using this utility, you can **start a single network** or **multiple networks**, from command line by specifying the name of the network created with the Network Designer tool.

**Command Line Options**

**Command Line option**: Mandatory

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-net &lt;network name&gt;</td>
<td>Specify the name of the network. To start multiple networks, specify the name of each network, separated by a comma.</td>
</tr>
</tbody>
</table>

**Protocol Option**: Optional

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-proto &lt;snmp, cli, tl1, tftp, telnet&gt;</td>
<td>To start a specific protocol present in all the devices. The protocol to be supported can be specified, separated by a comma. For example, if SNMP and Telnet are supported, specify the option as follows: -proto snmp, telnet</td>
</tr>
</tbody>
</table>

**Debug Options**: Optional.

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-m &lt;debugLevel&gt;</td>
<td>The option to generate debug messages. Debug Level can be 1 or 2 or 3 or 4</td>
</tr>
<tr>
<td>-l &lt;logFile&gt;</td>
<td>Log file name</td>
</tr>
</tbody>
</table>

**Initialize Services Options**: Optional. The services are enabled by default.

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-trap or -ts &lt;enable/disable&gt;</td>
<td>Enable/Disable Trap Sending. If the service is disabled, the agent will not check for trap configurations.</td>
</tr>
<tr>
<td>-snmpbehavior or -bs &lt;enable/disable&gt;</td>
<td>Enable/Disable Script Execution. If the service is disabled, the agent will not check for script configurations.</td>
</tr>
<tr>
<td>-scramblepdu or -ps &lt;enable/disable&gt;</td>
<td>Enable/Disable to scramble the PDU</td>
</tr>
<tr>
<td>-autonomous or -at &lt;enable/disable&gt;</td>
<td>Enable/Disable Autonomous Sending. If the service is disabled, the agent will not check for autonomous configurations.</td>
</tr>
<tr>
<td>-tl1behavior or -tb &lt;enable/disable&gt;</td>
<td>Enable/Disable TL1 Script Execution. If the service is disabled, the agent will not check for script configurations.</td>
</tr>
</tbody>
</table>
Miscellaneous Options : Optional

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanations</th>
</tr>
</thead>
<tbody>
<tr>
<td>-v3</td>
<td>Starts the network with v3 agents.</td>
</tr>
<tr>
<td>-encoding</td>
<td>Starts the network with the encoding format specified. If the specified</td>
</tr>
<tr>
<td></td>
<td>encoding format is not supported, then default encoding format is selected.</td>
</tr>
<tr>
<td>-rmi</td>
<td>To manage the network simulator remotely using RMI.</td>
</tr>
</tbody>
</table>

Global Manager Settings to send traps : Optional

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-tms &lt;Trap Manager Settings&gt;</td>
<td>To specify the global manager details, if you want all the traps to be sent to the same set of Managers, specified in this option. It should be in the following format - mgr1:port1:community1,mgr2:port2,community</td>
</tr>
</tbody>
</table>

Starting Network Simulation in Windows

To start Network Simulation in windows, type the following command from the `<SimulatorHome>/bin/cmdline` directory.

```
StartNetwork.bat -net .\networks\samplenetwork -m <specify debug level as 1 or 2> -l <to log the messages>
```

Starting Network Simulation in Linux and Solaris

To start Network Simulation in Windows OS, type the following command from the `<SimulatorHome>/bin/commandline` directory:

```
$ Sh StartNetwork.sh -net ./networks/samplenetwork -m <specify debug level as 1 or 2> -l <to log the messages>
```

Note:

1. The details of the Database server name, Port number, User name, Password and the Server path is stored in `database_params.conf` file in `<SimulatorHome>/conf` directory. You can make the required database settings in this file.

2. To start multiple networks from command line, the name of each network must be specified with the -net option, separated by a comma. For example: StartNetwork -net test1, test2

3. If there is an IP clash while starting multiple networks, an error message will be displayed and the networks will not get started.
IOS Simulator Guide

The IOS Simulator integrated with the Network Simulator, allows you to simulate Cisco routers and Cisco switches supporting Cisco IOS Software. It offers a realistic simulation for any management application that uses IOS. The IOS Simulator fully supports SNMPv1, v2c, v3 and TFTP. The simulated Cisco routers and switches can be accessed using CLI.

Enterprises as well as management application developers can simulate a complete Cisco network environment. The IOS simulator supports the following modes - user, privilegedEXEC, global configuration, Interface configuration mode, Router mode (BGP, OSPF, RIP, EIGRP), Vlan mode and config-if mode.

You can login and logout of the IOS simulator, get into and out of different modes, configure cdp, access-list, rmon alarms and events, set passwords, ip addresses, clock rates, hostname, enable/disable SNMP trap service, configure SNMP community, contact, location etc., specify MTU value for the network interface node, configure BGP, OSPF, RIP and vlan for a specific router, ping devices, display startup and running config and many other functions.

The following topics are covered in this section:

- Getting Started
- IOS Commands Supported
- CLI Script API Methods
- Adding IOS Commands
- Editing Scripts
Getting Started

- Creating the IOS Simulation
- Logging into IOS Devices

Creating the IOS Simulation

The following devices in the device tree supports IOS commands and CLI protocol support.

- Router : Cisco 3600, Cisco 7600
- Switch : Cisco 3750

The following Router/Switch devices will be available in the device library PPM that can be downloaded from the website. To download and add the following devices to the device tree, refer the topic, Adding Devices from PPM.

- Routers : Cisco1700, Cisco12000
- Switches : Cisco2950Switch, Cisco6500Switch

1. Create a new network using the template devices in the device tree. To create a new network, refer to Network Simulator Guide -> Creating a New Network.

2. Right-click on the router/switch device in the network area and select the option CLI -> Start Agent. This will automatically start the CLI agent with the given IPAddress and at default port 2323.

Logging into IOS Devices

1. To establish a telnet connection with the agent, go to command prompt and type: `c:\> telnet <ip address> <port number>`. This prompts you to enter the user name as shown below:

```
-----------------------------------------
AdventNet IOS Simulator
1996-2006 AdventNet SimulationToolKit
AdventNet Inc.All Rights Reserved.
http://www.adventnet.com
-----------------------------------------
Login :

Login :user
Password :********
```

2. To log in to IOS device, the default user name is user and password is password. Specify the default user name and password or the configured user name and password using deviceInfo dialog or BulkModification dialog.

```
-----------------------------------------
AdventNet IOS Simulator
1996-2006 AdventNet SimulationToolKit
AdventNet Inc.All Rights Reserved.
http://www.adventnet.com
-----------------------------------------
Login :
Password :user
Password :*******
Cisco1700>
```
3. To set up Passwords for your Router you need to first enter what is called: Privileged Exec Mode. To enter Privileged Exec Mode type in the word enable at the prompt.

```
Cisco17000> enable
```

4. Specify the password. The default password is test. This changes the prompt from Cisco1700> to Cisco1700#, as shown below:

```
Login :user
Password :********
Cisco1700> enable
Password :****
Cisco1700#
```

The # means that you have entered Privileged Exec Mode, also called "Enable Mode" for short. If you want to go back to being a plain User, just type disable. Type ? to list the commands that can be executed in this mode.

5. To get from Enable Mode to Configuration Mode, type the word configure terminal as shown below:

```
Cisco17000#configure terminal
```

6. The prompt will appear as shown below, which means that you are in the Router (Configure) mode. Now and only now can you start the process of configuring the router. Configure is a Global Command. Type ? to list the commands that can be executed in this mode.

```
Cisco17000(config)#
```

For example, you can configure your own hostname by specifying the following command. Note that the prompt changes to Router(config)#

```
Cisco17000(config)#hostname Router
Router(config)#
```

7. To get from Configuration Mode to configif mode (interface mode), type the word interface as shown below, specify the interface name and number.

```
Router(config)#interface ethernet 0
```
8. The prompt will appear as shown below, which means that you are in the Router (config-if) mode. Now you can start the process of configuring the router interface commands. Type \texttt{?} to list the commands that can be executed in this mode, as shown below. Type \texttt{exit} to go back to the previous mode.

```
Router(config-if)#?
Configure Interface Commands :
cdp To enable CDP
exit To Exit from this mode
help
ip To configure IP MTU value
no To disable CDP
```

Router(config-if)#
IOS Commands Supported

- **IOS Commands**
  - User Mode
  - Enable Mode
  - Config Mode
  - Vlan Mode
  - Configif Mode
  - Router Mode - RIP
  - Router Mode - EIGRP
  - Router Mode - BGP
  - Router Mode - OSPF

**List of supported IOS commands:**

<table>
<thead>
<tr>
<th>User Mode</th>
<th>Command Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable</td>
<td>To change the mode from User to Enable mode.</td>
</tr>
<tr>
<td>help</td>
<td>Displays the help description in the Modes.</td>
</tr>
<tr>
<td>exit</td>
<td>To log out from the client.</td>
</tr>
<tr>
<td>logout</td>
<td>To log out from the client.</td>
</tr>
<tr>
<td>show clock</td>
<td>To show the clock time of the CLI agent in the network.</td>
</tr>
<tr>
<td>show history</td>
<td>To display the History of the connected terminal.</td>
</tr>
<tr>
<td>show snmp</td>
<td>To show the SNMP status of the agent in the network.</td>
</tr>
<tr>
<td>show version</td>
<td>To show the IOS version and its details.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Enable Mode</th>
<th>Command Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clock set &lt;time&gt; &lt;month&gt; &lt;day&gt;</td>
<td>Sets the Clock with Date and Time in the Router</td>
</tr>
<tr>
<td>clock set &lt;month&gt; &lt;day&gt;</td>
<td>Sets the Clock with Date and Time in the Router</td>
</tr>
<tr>
<td>configure terminal</td>
<td>Change the mode to Configuration Mode</td>
</tr>
<tr>
<td>disable</td>
<td>Change the mode to user mode from Enable mode.</td>
</tr>
<tr>
<td>exit</td>
<td>To logout from the CLI client.</td>
</tr>
<tr>
<td>help</td>
<td>Displays help description in the Mode.</td>
</tr>
<tr>
<td>show access-list &lt;number&gt;</td>
<td>Displays the access-list configurations</td>
</tr>
<tr>
<td><strong>User Mode</strong></td>
<td><strong>Command Description</strong></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>show cdp</td>
<td>Displays the CDP service in the network node</td>
</tr>
<tr>
<td>show cdp entry all</td>
<td>Displays information about all neighboring devices discovered using CDP.</td>
</tr>
<tr>
<td>show cdp entry &lt;name&gt;</td>
<td>Displays information about a specific neighboring device discovered using CDP.</td>
</tr>
<tr>
<td>show cdp interface</td>
<td>Displays information about the interfaces on which Cisco Discovery Protocol (CDP) is enabled</td>
</tr>
<tr>
<td>show cdp neighbors</td>
<td>Displays detailed information about all neighboring devices discovered using CDP.</td>
</tr>
<tr>
<td>show clock</td>
<td>Display clock time of the CLIAgent in the network.</td>
</tr>
<tr>
<td>show flash</td>
<td>Displays the details of the CISCO Flash File Table</td>
</tr>
<tr>
<td>show flash all</td>
<td>Displays complete information about Flash memory, including information about the individual ROM devices in Flash memory and the names and sizes of all system image files stored in Flash memory</td>
</tr>
<tr>
<td>show flash chips</td>
<td>Displays information per partition and per chip, including which bank the chip is in, plus its code, size and name</td>
</tr>
<tr>
<td>show flash detailed</td>
<td>Displays detailed file directory information per partition, including file length, address, name, Flash memory checksum, computer checksum, bytes used, bytes available, total bytes, and bytes of system Flash memory.</td>
</tr>
<tr>
<td>show flash error</td>
<td>Displays write or erase failures in the form of number of retries</td>
</tr>
<tr>
<td>show history</td>
<td>Displays the History of the connected terminal.</td>
</tr>
<tr>
<td>show ip alias</td>
<td>Displays the IP Alias configured in the CLIAgent in the network.</td>
</tr>
<tr>
<td>show rmon alarms</td>
<td>Display all the active alarms in the rmon alarm table</td>
</tr>
<tr>
<td>show rmon events</td>
<td>Display all the event in the rmon event table</td>
</tr>
<tr>
<td>show snmp</td>
<td>Displays the SNMP status of the CLIagent in the network.</td>
</tr>
<tr>
<td>show version</td>
<td>Displays the IOS Version and its details of the CLIagent.</td>
</tr>
<tr>
<td>ping  &lt;word&gt;</td>
<td>To ping the device</td>
</tr>
<tr>
<td>show ip route static</td>
<td>To show the IP address route for the Interface in the device.</td>
</tr>
<tr>
<td>show ip interface brief</td>
<td>To show the brief of the interfaces in the device.</td>
</tr>
<tr>
<td>show vlan &lt;Line&gt;</td>
<td>To show the details of VLAN configured in the device.</td>
</tr>
<tr>
<td>User Mode</td>
<td>Command Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>show protocols</td>
<td>Displays the active network routing protocols.</td>
</tr>
<tr>
<td>show arp</td>
<td>Displays the ARP table.</td>
</tr>
<tr>
<td>show ip access-lists</td>
<td>List of IP access lists.</td>
</tr>
<tr>
<td>show ip bgp neighbors</td>
<td>Displays detailed information on TCP and BGP neighbor connections.</td>
</tr>
<tr>
<td>show ip bgp neighbors &lt;IP Address&gt;</td>
<td>Displays detailed information on TCP and BGP connections for the specified neighbor ipaddress.</td>
</tr>
<tr>
<td>clear ip bgp &lt;IP Address&gt;</td>
<td>Clear BGP connection.</td>
</tr>
<tr>
<td>clear ip bgp &lt;AS Number&gt;</td>
<td>Clear BGP connection based on the AS number of the peer.</td>
</tr>
<tr>
<td>show ip ospf</td>
<td>Display general information about OSPF routing processes</td>
</tr>
<tr>
<td>show ip ospf &lt;processid&gt;</td>
<td>Display general information about OSPF routing process for the specified processid</td>
</tr>
<tr>
<td>show startup-config</td>
<td>Displays the contents of startup configuration</td>
</tr>
<tr>
<td>show running-config</td>
<td>Displays the current operating configuration</td>
</tr>
<tr>
<td>Vlan database</td>
<td>This command is to enter into the Vlan mode without specifying any Vlan number. The vlan_database_en.py script file is used.</td>
</tr>
<tr>
<td>show vlan name [WORD]</td>
<td>To show the details of the specified Vlan name. The script file used for this command is show_vlan_name_en.py.</td>
</tr>
<tr>
<td>show vlan [vlan-id]</td>
<td>To show the details of the specified Vlan id. The script file used for this command is show_vlan_number_en.py.</td>
</tr>
<tr>
<td>show vlan</td>
<td>To show the details of all the configured Vlans. The script file used for this command is show_vlan_en.py.</td>
</tr>
<tr>
<td>Show vlan brief</td>
<td>This file is to show the details of VLan Interface configured. The script file used for this command is show_vlan_brief_en.py</td>
</tr>
<tr>
<td>show vtp status</td>
<td>To show the VTP status for the Vlan in the device. The script file used for this command is show_vtp_status_en.py.</td>
</tr>
<tr>
<td>show vtp counters</td>
<td>To show the VTP Counters for the Vlan in the device. The script file used for this command is show_vtp_counters_en.py.</td>
</tr>
<tr>
<td>show ip interface brief</td>
<td>This command's output needs to be changed with Vlan configuration. show_ip_ifbrief_en.py</td>
</tr>
<tr>
<td>show startup config</td>
<td>This command's output needs to be changed with Vlan configuration. show_startup_config_en.py</td>
</tr>
<tr>
<td>show running config</td>
<td>This command's output needs to be changed with Vlan configuration. show_running_config_en.py</td>
</tr>
<tr>
<td>User Mode</td>
<td>Command Description</td>
</tr>
<tr>
<td>------------------------------</td>
<td>--------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>show ip eigrp interface</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py.</td>
</tr>
<tr>
<td>show ip eigrp neighbours</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py.</td>
</tr>
<tr>
<td>show ip eigrp traffic</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py.</td>
</tr>
<tr>
<td>show ip protocols</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py.</td>
</tr>
<tr>
<td>show running-config</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py.</td>
</tr>
<tr>
<td>show startup-config</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Config Mode</th>
<th>Command Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>access-list &lt;number&gt; &lt;deny/permit&gt; &lt;any&gt;</td>
<td>Creates the standard access list entry</td>
</tr>
<tr>
<td>access-list &lt;number&gt; &lt;deny/permit&gt; host &lt;hostname&gt;</td>
<td>Creates the standard access list entry</td>
</tr>
<tr>
<td>access-list &lt;number&gt; &lt;deny/permit&gt; &lt;hostname&gt;/&lt;ip address&gt;</td>
<td>Creates the standard access list entry</td>
</tr>
<tr>
<td>cdp holdtime &lt;time in sec.&gt;</td>
<td>Configure the HoldTime for the CDP service in the network Interface node.</td>
</tr>
<tr>
<td>cdp run</td>
<td>Run the CDP service in the network node.</td>
</tr>
<tr>
<td>cdp timer &lt;time in sec&gt;</td>
<td>To configure the Timer value to the CDP service in the network node.</td>
</tr>
<tr>
<td>end</td>
<td>Change the mode from Configuration to Enable mode.</td>
</tr>
<tr>
<td>exit</td>
<td>Change the mode from Configuration to Enable mode.</td>
</tr>
<tr>
<td>help</td>
<td>Displays help description in the Mode.</td>
</tr>
<tr>
<td>hostname &lt;word&gt;</td>
<td>Updates the given value as the host name.</td>
</tr>
<tr>
<td>interface ethernet &lt;0-0&gt;</td>
<td>Change the mode from Configuration to Config Interface mode.</td>
</tr>
<tr>
<td>no &lt;access-list&gt; number</td>
<td>To remove a specific access-list</td>
</tr>
<tr>
<td>no cdp holdtime</td>
<td>Setting cdpGlobalHoldTime to CDP default, if the cdpGlobalRun value is 1/True(default value: 180 seconds.</td>
</tr>
<tr>
<td>no cdp run</td>
<td>Stops the CDP service in the network node.</td>
</tr>
<tr>
<td>User Mode</td>
<td>Command Description</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>--------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>no cdp timer</td>
<td>The node cdpGlobalMessageInterval is set to the default time interval of 60 seconds.</td>
</tr>
<tr>
<td>no snmp-server enable traps</td>
<td>Disables all the supported trap types</td>
</tr>
<tr>
<td>no snmp-server enable trap &lt;trap type&gt;</td>
<td>Disables the trap specified in the command</td>
</tr>
<tr>
<td>no snmp-server host &lt;ip address&gt;</td>
<td>Removes a specific recipient of the SNMP notification operation</td>
</tr>
<tr>
<td>no rmon event &lt;number&gt;</td>
<td>Removes an event in the RMON event table</td>
</tr>
<tr>
<td>no rmon alarm &lt;number&gt;</td>
<td>Removes an alarm in the RMON alarm table</td>
</tr>
<tr>
<td>rmon alarm number variable interval {delta</td>
<td>absolute} rising-threshold value [event- number] falling-threshold value [event-number] [owner string]</td>
</tr>
<tr>
<td>rmon event number description &lt;string&gt; owner &lt;string&gt; log trap &lt;community&gt;</td>
<td>Adds an event in the RMON event table that is associated with an RMON event number.</td>
</tr>
<tr>
<td>rmon event number description &lt;string&gt; owner &lt;string&gt; trap &lt;community&gt; log</td>
<td>Adds an event in the RMON event table that is associated with an RMON event number, use the #rmon event in global configuration command</td>
</tr>
<tr>
<td>snmp-server community &lt;community name&gt;</td>
<td>Sets up the community access string to permit access to the Simple Network Management Protocol(SNMP)</td>
</tr>
<tr>
<td>snmp-server contact &lt;line&gt;</td>
<td>Sets the system contact (sysContact) string.</td>
</tr>
<tr>
<td>snmp-server enable traps</td>
<td>Enables all the supported trap types</td>
</tr>
<tr>
<td>snmp-server enable trap &lt;trap type&gt;</td>
<td>Enables the trap specified in the command by inserting the data in the database</td>
</tr>
<tr>
<td>snmp-server location &lt;line&gt;</td>
<td>Sets up the system location string that describes the system location information.</td>
</tr>
<tr>
<td>snmp-server packetsize &lt;byte-count&gt;</td>
<td>Integer byte count from 484 to 8192. The default is 1500 bytes.</td>
</tr>
<tr>
<td>snmp-server host [host-addr] [community- string]</td>
<td>To specify the recipients of the SNMP notification operation</td>
</tr>
<tr>
<td>enable password &lt;Line&gt;</td>
<td>To change the Password of the User Enable mode in the Router.</td>
</tr>
<tr>
<td>ip route &lt;IP Address&gt; &lt;netmask&gt; &lt;gateway&gt;</td>
<td>To specify the secondary IP Address for the Interface in the router.</td>
</tr>
<tr>
<td>router rip</td>
<td>To change the Mode to Router mode</td>
</tr>
<tr>
<td>vlan &lt;vlan id&gt;</td>
<td>To set the Vlan to the device</td>
</tr>
<tr>
<td>router bgp &lt;number&gt;</td>
<td>Enable bgp routing process</td>
</tr>
<tr>
<td>arp &lt;IP Address&gt; &lt;hardware address&gt; arpa</td>
<td>Configure IP address of ARP entry</td>
</tr>
</tbody>
</table>
### User Mode

<table>
<thead>
<tr>
<th>Command Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>no arp &lt;IP Address&gt;</td>
</tr>
<tr>
<td>router ospf &lt;processid&gt;</td>
</tr>
<tr>
<td>no router ospf &lt;processid&gt;</td>
</tr>
<tr>
<td>router eigrp</td>
</tr>
<tr>
<td>To enable EIGRP routing. This command is added into the 'basedata' database with ParamName as 'eigrp_as'. The script file used for this command is <code>router_eigrp_autonomous_system_ct.py</code></td>
</tr>
</tbody>
</table>

### Vlan Mode

<table>
<thead>
<tr>
<th>Command Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name &lt;vlan name&gt;</td>
</tr>
<tr>
<td>mtu &lt;mtu size of vlan&gt;</td>
</tr>
<tr>
<td>vlan vlan-id name vlan-name</td>
</tr>
<tr>
<td>This command is used to enter the Vlan name with specifying the Vlan id in Config mode. The <code>vlan_name_vid_ct.py</code> script file is used.</td>
</tr>
<tr>
<td>vlan vlan-id name vlan-name mtu mtu-size</td>
</tr>
<tr>
<td>This command is used to enter the Vlan name and MTU value with specifying the Vlan id in Config mode. The <code>vlan_name_mtu_vid.py</code> script file is used.</td>
</tr>
<tr>
<td>vlan vlan-id mtu mtu-size name vlan-name</td>
</tr>
<tr>
<td>This command is used to enter the MTU value and Vlan name with specifying the Vlan id in Config mode. The <code>vlan_mtu_name_vid.py</code> script file is used.</td>
</tr>
<tr>
<td>vlan vlan-id mtu mtu-size</td>
</tr>
<tr>
<td>This command is used to enter the MTU size with specifying the Vlan id in Config mode. The <code>vlan_mtu_vid_ct.py</code> script file is used.</td>
</tr>
<tr>
<td>vlan [vlan-id]</td>
</tr>
<tr>
<td>show current [vlan-id]</td>
</tr>
<tr>
<td>To show the details of the specified Vlan id. The script file used for this command is <code>show_current_vid_vlan.py</code>.</td>
</tr>
<tr>
<td>show current</td>
</tr>
<tr>
<td>To show the details of all the configured Vlans. The script file used for this command is <code>show_current_vlan.py</code>.</td>
</tr>
<tr>
<td>Show proposed</td>
</tr>
<tr>
<td>To show the details of all the configured Vlans. The script file used for this command is <code>show_proposed_vlan.py</code>.</td>
</tr>
<tr>
<td>Show proposed [vlan-id]</td>
</tr>
<tr>
<td>To show the details of all the configured Vlans. The script file used for this command is <code>show_proposed_vid_vlan.py</code>.</td>
</tr>
<tr>
<td>exit</td>
</tr>
<tr>
<td>To exit from the Vlan mode to Config mode. The script file used for this command is <code>vlan_exit.py</code>.</td>
</tr>
<tr>
<td>apply</td>
</tr>
<tr>
<td>Just print Apply Completed. Write the new script file for this. <code>vlan_apply.py</code></td>
</tr>
<tr>
<td>no vlan vlan-id</td>
</tr>
<tr>
<td>To remove the configured Vlan from the device. The script file used for this command is <code>no_vlan_vid_vlan.py</code></td>
</tr>
</tbody>
</table>
### User Mode

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>no vlan [vlan id] name</td>
<td>To remove the Vlan name for the Vlan configuration from the device. The script file used for this command is <code>no_vlan_name_vlan.py</code>.</td>
</tr>
<tr>
<td>no vlan mtu</td>
<td>To remove the Vlan MTU value for the Vlan configuration from the device. The script file used for this command is <code>no_vlan_mtu_vlan.py</code>.</td>
</tr>
<tr>
<td>vtp server</td>
<td>To change the VTP mode of Vlan configuration. The script file used for this command is <code>set_vtp_mode_vlan.py</code>.</td>
</tr>
<tr>
<td>vtp client</td>
<td>To change the VTP mode of Vlan configuration. The script file used for this command is <code>set_vtp_mode_vlan.py</code>.</td>
</tr>
<tr>
<td>vtp transparent</td>
<td>To change the VTP mode of Vlan configuration. The script file used for this command is <code>set_vtp_mode_vlan.py</code>.</td>
</tr>
<tr>
<td>no vtp client</td>
<td>To disable the VTP Client mode of Vlan configuration. The mode has been changed into Transparent. The script file used for this command is <code>no_vtp_client_vlan.py</code>.</td>
</tr>
<tr>
<td>no vtp transparent</td>
<td>To disable the VTP Transparent mode of Vlan configuration. The mode has been changed into Server. The script file used for this command is <code>no_vtp_transparent_vlan.py</code>.</td>
</tr>
</tbody>
</table>

### Configif Mode

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cdp enable</td>
<td>Enables the CDP service in the network node.</td>
</tr>
<tr>
<td>ip address &lt;A.B.C.D&gt;</td>
<td>Set the ipaddress of an interface</td>
</tr>
<tr>
<td>ip mtu &lt;value&gt;</td>
<td>To specify the MTU value for the network interface node to which the CLI client is connected.</td>
</tr>
<tr>
<td>ip ospf transmit-delay &lt;seconds&gt;</td>
<td>Configure Link state transmit delay for OSPF</td>
</tr>
<tr>
<td>ip ospf retransmit-interval &lt;seconds&gt;</td>
<td>Configure time between retransmitting lost link state advertisements for OSPF</td>
</tr>
<tr>
<td>ip ospf priority &lt;number&gt;</td>
<td>Configure IP OSPF router priority</td>
</tr>
<tr>
<td>help</td>
<td>To display the help description</td>
</tr>
<tr>
<td>exit</td>
<td>Change the mode from config-if to Configuration mode.</td>
</tr>
<tr>
<td>no cdp enable</td>
<td>Disables the CDP service in the network node for the CLI command.</td>
</tr>
</tbody>
</table>

### Router Mode - RIP

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network &lt;network id&gt;</td>
<td>To enable RIP for the specified network id.</td>
</tr>
</tbody>
</table>
### User Mode

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>neighbour &lt;network id&gt;</td>
<td>To add the Route for the network in the Router</td>
</tr>
<tr>
<td>no neighbor &lt;network id&gt;</td>
<td>To remove the Route for the network in the Router</td>
</tr>
<tr>
<td>no network &lt;network id&gt;</td>
<td>To disable RIP for the specified network id</td>
</tr>
<tr>
<td>help</td>
<td>To display the help description</td>
</tr>
<tr>
<td>exit</td>
<td>Exit from routing protocol configuration mode</td>
</tr>
</tbody>
</table>

### Router Mode - BGP

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network &lt;IP Address&gt;</td>
<td>Specify a network to announce via BGP</td>
</tr>
<tr>
<td>neighbor &lt;IP Address&gt; remote-as &lt;1-65535&gt;</td>
<td>Specify a BGP neighbor</td>
</tr>
<tr>
<td>neighbor &lt;IP Address&gt; version &lt;2-4&gt;</td>
<td>Set the BGP version to match a neighbor</td>
</tr>
<tr>
<td>no network &lt;IP Address&gt;</td>
<td>Disable BGP routing on an IP network</td>
</tr>
<tr>
<td>no neighbor &lt;IP Address&gt;</td>
<td>Disable BGP neighbor entry</td>
</tr>
<tr>
<td>help</td>
<td>To display the help description</td>
</tr>
<tr>
<td>exit</td>
<td>Exit from routing protocol configuration mode</td>
</tr>
</tbody>
</table>

### Router Mode - EIGRP

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network &lt;network-number&gt;</td>
<td>To add the Network address to the EIGRP routing mode. The script file used for this command is eigrp_network_router.py</td>
</tr>
<tr>
<td>neighbor &lt;neighbor address&gt;</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py</td>
</tr>
<tr>
<td>variance &lt;variance number&gt;</td>
<td>To add the Variance value to the EIGRP routing mode. The script file used for this command is eigrp_variance_router.py</td>
</tr>
<tr>
<td>timers active-time</td>
<td>To configure the active timer to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py</td>
</tr>
<tr>
<td>timers active-time &lt;timer in minutes&gt;</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py</td>
</tr>
<tr>
<td>timers active-time disable</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py</td>
</tr>
<tr>
<td>help</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is eigrp_neighbor_router.py</td>
</tr>
</tbody>
</table>
### User Mode

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>exit</td>
<td>To add the Neighbor address to the EIGRP routing mode. The script file used for this command is <code>eigrp_neighbor_router.py</code></td>
</tr>
</tbody>
</table>

### Router Mode - OSPF

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network&lt;IP Address&gt; areaid &lt;number&gt;</td>
<td>Set the ospf areaid</td>
</tr>
<tr>
<td>help</td>
<td>To display the help description</td>
</tr>
<tr>
<td>exit</td>
<td>Exit from routing protocol configuration mode</td>
</tr>
</tbody>
</table>

**Note:**

1. Vlan, VTP support is provided for the following switch devices:
   - Cisco3750 and Cisco6500 Switches
2. EIGRP support is provided for the following Router devices:
   - Cisco3600 and Cisco1700 Routers.
3. Refer to [device library ppm](#), to download and install the devices in the device library PPM.
### IOS commands supported in PIX Firewall

This document gives a brief overview of the IOS commands supported in Cisco PIX Firewall device.

#### Cisco PIX Firewall Commands Supported in CLI

<table>
<thead>
<tr>
<th>Command Mode</th>
<th>Command</th>
<th>Command Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User Mode</strong></td>
<td>enable</td>
<td>To change the mode from User to Enable mode.</td>
</tr>
<tr>
<td></td>
<td>help</td>
<td>Displays the help description in the Modes.</td>
</tr>
<tr>
<td></td>
<td>exit</td>
<td>To log out from the CLI client.</td>
</tr>
<tr>
<td></td>
<td>logout</td>
<td>To log out from the CLI client.</td>
</tr>
<tr>
<td></td>
<td>show clock</td>
<td>To show the clock time of the CLI agent in the network.</td>
</tr>
<tr>
<td></td>
<td>show history</td>
<td>To display the History of the connected terminal.</td>
</tr>
<tr>
<td></td>
<td>show snmp</td>
<td>To show the SNMP status of the device in the network.</td>
</tr>
<tr>
<td></td>
<td>show version</td>
<td>To show the version of the device.</td>
</tr>
<tr>
<td><strong>Enable Mode</strong></td>
<td>clock set &lt;time&gt; &lt;day&gt; &lt;month&gt;</td>
<td>Sets the Clock with Date and Time in the firewall device</td>
</tr>
<tr>
<td></td>
<td>configure terminal</td>
<td>Change the mode to Configuration Mode</td>
</tr>
<tr>
<td></td>
<td>disable</td>
<td>Change the mode to user mode from Enable mode.</td>
</tr>
<tr>
<td></td>
<td>ping</td>
<td>Send echo messages</td>
</tr>
<tr>
<td></td>
<td>show access-list &lt;number&gt;</td>
<td>Displays the access-list configurations</td>
</tr>
<tr>
<td></td>
<td>show blocks</td>
<td>To show the system buffer usage</td>
</tr>
<tr>
<td></td>
<td>show conn</td>
<td>To show the number of connections in use</td>
</tr>
<tr>
<td></td>
<td>show failover</td>
<td>To show the failover status</td>
</tr>
<tr>
<td></td>
<td>show memory</td>
<td>To show the free memory available</td>
</tr>
<tr>
<td></td>
<td>show nameif</td>
<td>To show the current names and security levels of each interface</td>
</tr>
<tr>
<td></td>
<td>show clock</td>
<td>Display clock time of the CLI agent in the network.</td>
</tr>
<tr>
<td></td>
<td>show history</td>
<td>Displays the History of the connected terminal.</td>
</tr>
<tr>
<td></td>
<td>show version</td>
<td>Displays the IOS Version and its details of the CLI agent.</td>
</tr>
<tr>
<td><strong>Config Mode</strong></td>
<td>access-group &lt;acl_ID&gt; in interface &lt;interface_name&gt;</td>
<td>Binds the access list to the inside interface.</td>
</tr>
<tr>
<td></td>
<td>access-list &lt;number&gt; &lt;deny/permit&gt; host &lt;hostname&gt;</td>
<td>Creates the standard access list entry</td>
</tr>
<tr>
<td></td>
<td>access-list &lt;number&gt; &lt;deny/permit&gt; &lt;host name&gt;/&lt;ip address&gt;</td>
<td>Creates the standard access list entry</td>
</tr>
<tr>
<td></td>
<td>exit</td>
<td>Change the mode from Configuration to Enable mode.</td>
</tr>
<tr>
<td></td>
<td>hostname &lt;word&gt;</td>
<td>Updates the given value as the host name.</td>
</tr>
<tr>
<td></td>
<td>snmp-server enable traps</td>
<td>Enables all the supported trap types</td>
</tr>
<tr>
<td></td>
<td>snmp-server host [host-</td>
<td>To specify the recipients of the SNMP notification</td>
</tr>
<tr>
<td>Command Mode</td>
<td>Command</td>
<td>Command Description</td>
</tr>
<tr>
<td>--------------</td>
<td>---------</td>
<td>---------------------</td>
</tr>
<tr>
<td>addr [community-string]</td>
<td>operatio</td>
<td>Set the inside/outside default route to the router attached to the Internet.</td>
</tr>
<tr>
<td>route &lt;if_name&gt; &lt;ip_address&gt; &lt;netmask&gt; &lt;gateway_ip&gt; [metric]</td>
<td>To remove a specific access-list</td>
<td></td>
</tr>
<tr>
<td>no &lt;access-list&gt; number</td>
<td>disables all the supported trap types</td>
<td></td>
</tr>
<tr>
<td>no snmp-server enable traps</td>
<td>Removes a specific recipient of the SNMP notification operation</td>
<td></td>
</tr>
</tbody>
</table>
CLI Script API methods

Overview

The CLI script API methods are added to the ScriptInterface to perform operations such as getScriptFile, add parameters, get parameter value, delete parameters, etc. from a JPython script. The SNMP script API methods can also be used within the Jython scripts to implement the IOS commands.

The CLI and SNMP script API methods are provided under `<simulator home>/examples/javadocs` directory. The name of the Script API file is CLI_SimScrIntf.html

CLI Script API methods

a. getScriptFile method

This method returns the script file for the given command.

```java
public void getScriptFile(String command, String modeis)
```

where

command - IOS command
modeis   - The mode in which the IOS Command is present

b. updateScriptFile method

This method updates the script file for a particular command.

```java
public void updateScriptFile(String command, String scriptfile, String modeis)
```

where,

command - IOS command
scriptfile - The new script file to be replaced for the command
modeis   - The mode in which the command is present.

c. getCurrentCommand method

This method returns the command which is now being executed.

```java
public String getCurrentCommand()
```
d. getCurrentLevel method

This method returns the level (Integer values for mode) at which the command is executed.

```java
public int getCurrentLevel()
```

**Note**: The modes supported are represented by Integer values. User mode - 5, Enable mode - 6, Config mode - 7, Config interface mode - 9

e. getMode method

This method returns the mode in which the command is executed.

```java
public String getMode()
```

f. changeMode method

This method changes the mode of the client to the specified one, when the mode is enable and greater.

```java
public void changeMode(int mode)
```

where,

mode - Integer value of the mode.

g. getCmdWord method

This method returns the respective command word for the specified index of the currently executing command.

```java
public String getCmdWord(int index)
```

h. updateHost method

This method is used to update the hostname of the client with the given name.

```java
public void updateHost(String name)
```

where

name - refers to the new Host name

i. getInterfaceStatus method

This method gives what is the interface being configured.

```java
public String getInterfaceStatus()
```

j. addParameters method

This method enables you to add the given paramValue with the ParamKey in the base table.

```java
public void addParameters(String paramKey, String paramValue)
```

where
paramkey    -  The key for the paramvalue.
paramvalue  -  The value to be stored.

k. getParamValue method
This method enables you to get the paramValue for the given ParamKey.

```java
public String getParamValue(String paramKey)
```

where,
paramkey    -  The key for which a value is stored.

l. deleteParameters method
This method enables you to delete the row for the given param Key.

```java
public void deleteParameters(String paramKey);
```

where,
paramkey    -  The key for which a value is stored.

m. getParamListValues method
This method enables you to get the list of values for the paramkey (which is just the starting of the actual paramKey)

```java
public Vector getParamListValues(String paramKey);
```

where,
paramkey    -  The key for which a value is stored.

n. executeScript method
This method enables you to execute another script from a given script file.

```java
public String executeScript(String scriptFile)
```

where,
scriptfile - The script file name along with the path from Simulator home directory

o. getParamKeyList method
To get a list of paramKeys.

```java
public Vector getParamKeyList(String paramKey)
```

where,
paramkey    -  The key in the basedata table.
p. `setConnectionTime` method

This method is to set the time for the client session to be alive even when it is idle.

```java
public void setConnectionTime (int time )
```

where ,

time - time value in seconds.

q. `showHistory` method

This method is to display the commands executed previously. The number of commands displayed will depend upon the history size. The default history size is ten.

```java
public void showHistory()
```

h. `updateIPAddress` method

```java
public void updateIPAddress(String ipAddress)
```

This method is used to update the IP Address of the device for which the request has come.

i. `setEnablePassword` method

This method sets the Enable mode password for the device for which the request has come.

```java
public void setEnablePassword(String password)
```

j. `getEnablePassword` method

```java
public String getEnablePassword()
```

This method gets the password for the enable mode for which the request has come.

k. `isInterfaceUp` method

```java
public boolean isInterfaceUp(String ipAddress)
```

This method checks whether a particular IP Address is up or not.

l. `getRoundTripTime` method

```java
public int getRoundTripTime()
```

This method gives the average time taken by `isInterfaceUp` method to check whether a particular IP Address is up or not and hence has to be called after `isInterfaceUp` method.

m. `getMaxRoundTripTime` method

```java
public int getMaxRoundTripTime();
```

This method gives the maximum time taken by `isInterfaceUp` method to check whether a particular IP Address is up or not and hence has to be called after `isInterfaceUp` method.
n. getMinRoundTripTime method

```
public int getMinRoundTripTime();
```

This method gives the minimum time taken by isInterfaceUp method to check whether a particular IPAddress is up or not and hence has to be called after isInterfaceUp method.

o. setPassword method

```
public void setPassword(String password);
```

This method sets the password for the device for which the request has come

p. getPassword method

```
public String getPassword();
```

This method gets the password for the device for which the request has come
Adding IOS Commands

- **Overview**
- **Adding IOS Commands**

**Overview**

The Network Simulator provides the option to define your own IOS Commands and add the required script files to a device in the device tree. New IOS commands cannot be added to individual devices in the network area or to devices in the template tree.

The script files can be defined using the SNMP and CLI script API methods provided in the scriptinterface. The CLI and SNMP script API methods are provided under `<simulator home>/examples/javadocs` directory. The name of the Script API file is `CLI_SimScrIntf.html`.

**Adding IOS Commands**

1. Right-click on the CLI device in the device tree and select the option **IOS Command Configurator**. The IOS Script Configuration dialog is displayed, wherein you have to specify the device details and command details.

**Specifying Device Details**

- The name of the selected device, database name and the configuration name is displayed and is not configurable.
- Specify any of the following options
  - **Add command to device only**: Adds the IOS command only to the selected device type in the device tree.
  - **Add command to selected network**: Adds the IOS commands to the devices of same type, in the network selected in the network tree.

**Specifying Command Details**

- **Mode**: Select the command mode in which the IOS command must be added, from the list displayed.
- **Command**: Specify the exact command to be added. For example: `route <WORD>, host <Hostname or A.B.C.D> <WORD>, snmp-server community <WORD>`
- **Command Description**: Specify the description of the command
- **Script File**: Browse and select the script file to be configured for the command.

2. Click **Add** to add the command to the device in the device tree.
Editing Scripts

- Overview
  - Using Script Editor

Overview

IOS Simulator comes along with a Script Editor which allows you to view and edit the script file configured for the commands. It gives a list of all the IOS commands supported for the specific device. Selecting a command will display the script file configured for the command. You can view and edit the script file as per your requirement.

Using Script Editor

Right-click on the Router/Switch device in the network which supports CLI protocol and select CLI -> Commands Script. The CLI commands Script Editor is invoked. It displays the IOS commands and the scripts implemented for the commands on the left-side tree structure as shown in the image below. The commands are organized based on the modes in which it is supported.

The icon shows that a script file is configured for the command. For example: In Enable mode, in the command, show cdp entry all, the script file show_cdp_entry_all_en.py will be implemented when the complete command is entered in the CLI console.

The following options are provided in the script editor to edit and save the changes or revert back.

Revert : To revert back the changes made in the current file.
Revert All : To revert back the changes made in all the files.
Save : To save the changes made in the current file.
Save All : To save the changes made in all the files.
TFTP, FTP and Telnet Support

AdventNet Simulation Toolkit supports TFTP client and server protocol and FTP client protocol, to enable transfer of files between the manager and the agent. You can also create your own SNMP devices with Telnet support and execute telnet commands on the simulated devices. This facility is supported in Network Simulator.

The following topics are covered in this section

- TFTP Protocol Support
- FTP Protocol Support
- Telnet Protocol Support
TFTP protocol support

• Overview
• Transferring Files from the Network Designer UI
  • Configuring TFTP server root directory
  • Configuring scripts during file upload and download
  • Starting TFTP server
  • Transferring files between the client and server
  • Stopping TFTP Server
• Transferring files from Jython Scripts and RMI
  • Starting the TFTP server
  • Transferring files between the client and server
  • Stopping the TFTP server
  • Checking if TFTP server is running

Overview
AdventNet Simulation Toolkit has implemented the TFTP client and server protocol, an optional facility that enables transfer of files between the TFTP client and the TFTP server. This facility enables the manager to configure an agent by changing the agent configuration files. The agent can use this facility to report any issues to the manager.

The file transfer can be done from the Network designer UI or using Jython scripts or through RMI at network startup or during runtime. The mode of transfer may be ascii or binary. The files transferred should have the ‘public’ access. TFTP uses UDP to transfer packets across the network.

Transferring Files From the Network Designer UI

The Network designer provides options to configure and start the TFTP server and transfer files between the client and the server.

Configuring TFTP server root directory

You can configure root directory for individual devices in the network. By default, a TFTP server will have the root directory below ./networks/<networkname>/tftp/<templatename> for all the devices.

To configure root directory for individual devices,

• Right-click on a device in the network either from the network area.
• Select the option Device Info -> TFTP
• The default root directory will be displayed as ./networks/<networkname>/tftp/<templatename>. This can be configured as per your requirement.
• Click OK to apply the changes.
To configure root directory in bulk based on the template devices, use the template device right-click menu option. Refer to Template Devices and Configurations - Telnet Parameters for more information.

Configuring Scripts to be Executed on File Upload or Download

You can configure Jython script files to be executed when a file is uploaded to the server or downloaded from the server. This facility can be used to inform the manager about a file transfer, by way of sending traps etc. You can configure different script files for individual servers or same script file for all the servers.

To configure scripts on individual devices,
- Right-click on a device in the network either from the network tree or network area.
- Select the option Device Info -> TFTP
- To execute a script when a file is downloaded from the server, browse and select the script (.py file) in the Execute script on download combo box. This script file will be copied in .networks/<networkname>/tftp/<templatedevicename>/<devicename> directory in the name TftpDownloadScript.py. The script will be executed whenever a file is downloaded from the TFTP server in the simulator.
- To execute a script when a file is uploaded to the server, browse and select the script (.py file) in the Execute script on upload combo-box. This script file will be copied in .networks/<networkname>/tftp/<templatedevicename>/<devicename> directory in the name TftpUploadScript.py. The script will be executed whenever a file is uploaded to the TFTP server in the simulator.
- Click OK to apply the changes.

To configure upload/download scripts in bulk based on the template devices, use the template device right-click menu option. Refer to Template Devices and Configurations - Telnet Parameters for more information.

Starting the TFTP server

To start the TFTP server in individual devices,
- Right-click on a device in the network either from the network tree or network area.
- Select the option TFTP -> Start. This will start the server at the port specified in the TFTP Server Properties dialog. By default, the TFTP server starts at port 6969 with the root directory as .networks/<networkname>/tftp/<templatedevicename>.

Transferring Files Between the Client and the Server

- Right-click on a device in the network either from the network area.
- Select the option TFTP Client. This will display the TFTP client dialog. Specify the following details in this dialog:
- **Server IP**: Specify the TFTP server IP address (e.g., 200.201.202.1)
- **Port**: The TFTP client talks to the standard TFTP port 6969 on the TFTP server by default. If the server is configured to use a non-standard port, you can specify it in this option.
- **Timeout**: Specify the time to wait for a response between successive retransmits in milliseconds. The default is 5000 milliseconds.
- **Retries**: Specify the number of retransmits. The default number of retries is 0.
- **Mode**: By default, the transfer mode is binary (octet). If you want ascii data to be transferred, then select the mode as netascii.
**Command**: Select the command as **Download file** to download a file from the server to the client. Select the command as **Upload File** to upload a file from the client to the server.

- If the command is to download a file from the server, specify the source and target as follows:
  - **Source**: Specify the file name to be retrieved from the server with the relative path. The path specified must not be outside the root directory of the server, as specified in the TFTP server properties.
  - **Target**: Specify the file name with path, to save the downloaded data in the client.
- If the command is to upload a file to the server, specify the source and target as follows:
  - **Source**: Specify the file name with path, to be retrieved from the client.
  - **Target**: Specify the file name with path, to upload the data in the server. The path specified must not be outside the root directory of the server as specified in the TFTP server properties.
- Click **Transfer** to download or upload the files. The timeout value will be 5000 milliseconds by default and the retries will be 0 by default. If the file is transferred the message **File Transferred** will be displayed.
- Click **Close** to exit the dialog.

### Stopping the TFTP server

To stop the TFTP server in individual devices,

- Right-click on a device in the network either from the network tree or network area.
- Select the option **TFTP -> Stop**. This will stop the TFTP server running at the specified port.

### Transferring files from Jython Scripts and RMI

Simulation Toolkit has implemented methods in RMI and Jython script interface to start the TFTP server, transfer files between the client and server and stop the TFTP server, at agent runtime. These methods are defined in the following script APIs and RMI client APIs in `<SimulatorHome>/examples/javadocs` directory.

- **SNMPScriptInterface**: Defines the methods that can be used to perform operations on a SNMP device, from a Jython script file.
- **TL1ScriptInterface**: Defines the methods that can be used to perform operations on a TL1 device, from a Jython script file.
- **NetSimSnmpInterface**: Defines the methods that can be used to perform operations on a SNMP agent via RMI.
- **NetSimTL1Interface**: Defines the methods that can be used to perform operations on a TL1 agent via RMI.

### Starting the TFTP server

This method can be used to start the TFTP server in the Network from any other agent, at runtime.

```java
startTftpServer(String ipAddress, int port, String rootDir);
```

where,
- ipAddress - the ip address of the server
- port - the port number where the server is started
- rootDir - the root directory of the server.

**Note**: The files can be downloaded from or uploaded to the root directory or within its subdirectories.
Transferring Files between the Client and the Server

This method can be used to upload the 'source' file to the TFTP server running at the specified ip address and port number, to the 'target' file in ascii or binary mode using the 'tftp/ftp' protocol.

\[
\text{uploadFile(String ipAddress, int port, String source, String target, int mode, String protocol);}\
\]

where,

ipAddress - The ip address at which the TFTP server is running
port - The port in the host at which the TFTP server is running
source - The source file name with path, to be retrieved from the client.
target - The file name with path, to upload the data in the server.
mode - The mode of transfer, 0 for netascii mode and 1 for octets mode
protocol - The protocol used to transfer the files, ftp/tftp

This method can be used to download the 'source' file from the TFTP server running at the specified ip address and port number, to the 'target' file in ascii or binary mode using the 'tftp/ftp' protocol.

\[
\text{downloadFile(String ipAddress, int port, String source, String target, int mode, String protocol);}\
\]

where,

ipAddress - The ip address at which the TFTP server is running
port - The port in the host at which the TFTP server is running
source - The file name with path, to be retrieved from the server.
target - The file name with path, to download the data in the client.
mode - The mode of transfer, 0 for netascii mode and 1 for octets mode
protocol - The protocol used to transfer the files, ftp/tftp

Stopping the TFTP server

This method can be used to stop the TFTP server from any other agent in the network, at runtime.

\[
\text{stopTftpServer(String ipAddress, int port);}\
\]

where,

ipAddress - ip address of the tftp server
port - the port number where the server is started

Checking if TFTP server is running at the specified port

This method can be used at runtime to check if the TFTP server is running at the specified ip address and port number.

\[
\text{isTftpRunning(String ipAddress, int port);}\
\]

Refer to section Managing Agents via RMI, to learn more about managing the agent via RMI.

Refer to section Configuring Network Behavior, to learn more about configuring behavior scripts.
FTP Protocol Support

- Overview
- Transferring files from the Network designer UI using FTP
- Transferring files from Jython Scripts and RMI using FTP

Overview

AdventNet Simulation Toolkit has implemented the FTP client protocol, an optional facility that enables transfer of files between the FTP client and the FTP server. This facility enables the manager to configure an agent by changing the agent configuration files. The agent can use this facility to report any issues to the manager.

The file transfer can be done from the Network designer UI or using Jython scripts or through RMI at network startup or during runtime. The mode of transfer may be ascii or binary. The files transferred should have the ‘public’ access. FTP uses UDP to transfer packets across the network.

Transferring files from the Network designer UI

To transfer files between the client and the server
- Right-click on a device in the network either from the network tree or network area.
- Select the option FTP Client. This will display the FTP client dialog. Specify the following details in this dialog:
  - Server IP: Specify the FTP Server IP address (e.g., 200.201.202.1)
  - Port: The FTP client talks to the standard FTP port 21 on the FTP server by default. If the server is configured to use a non-standard port, you can specify it in this option.
  - User Name and Password: Specify the username and password to connect to the FTP server. The default username and password is guest.
  - Mode: By default, the transfer mode is binary. If you want ascii data to be transferred, then select the mode as ascii.
  - Command: Select the command as Download file to download a file from the server to the client. Select the command as Upload File to upload a file from the client to the server.
  - If the command is to download a file from the FTP server, specify the source and target as follows:
    - Source: Specify the file name to be retrieved from the FTP server with the path.
    - Target: Specify the file name with path, to save the downloaded data in the client.
  - If the command is to upload a file to the FTP server, specify the source and target as follows:
    - Source: Specify the file name with path, to be retrieved from the FTP client.
    - Target: Specify the file name with path, to upload the data in the server.
- Click Transfer to download or upload the files. The maximum timeout value will be 5000 milliseconds by default and the maximum retries will be 0 by default. Timeout and retries are not configurable. If the file is transferred the message File Transferred will be displayed.
- Click Close to exit the dialog.
Transferring files from Jython Scripts and RMI

Simulation Toolkit has implemented methods in RMI and Jython script interface to transfer files between the client and server using FTP, at agent runtime. These methods are defined in the following script APIs and RMI client APIs in <SimulatorHome>/examples/javadocs directory.

- **SNMPScriptInterface**: Defines the methods that can be used to perform operations on a SNMP device, from a Jython script file.
- **TL1ScriptInterface**: Defines the methods that can be used to perform operations on a TL1 device, from a Jython script file.
- **NetSimSnmpInterface**: Defines the methods that can be used to perform operations on a SNMP agent via RMI.
- **NetSimTL1Interface**: Defines the methods that can be used to perform operations on a TL1 agent via RMI.

To transfer files between the client and the server

This method will **upload** the 'source' file in the client to the FTP server running at the specified ip address and port number, in ascii or binary mode using the 'FTP/ftp' protocol. It returns a boolean whether the file is uploaded or not.

```java
uploadFile(String ipAddress, int port, # String source, String target, int mode, String protocol, # String user, String password);
```

where,

- ipAddress - The host at which the FTP server is running
- port - The port in the host at which the FTP server is running
- source - The source file to be uploaded
- target - The destination file name
- mode - The mode of transfer either '0 for ascii' or '1 for binary'
- protocol - The protocol to be used for uploading file
- user - The user name for the protocol connection
- password - The password of the user for the protocol connection

This method can be used to **download** the 'source' file from the FTP server running at the specified ip address and port number, to the client 'target' file in ascii or binary mode using the 'FTP/ftp' protocol.

```java
downloadFile (String ipAddress, int port, # String source, String target, int mode, String protocol), # String user, String password); ;
```

where,

- ipAddress - The host at which the FTP server is running
- port - The port in the host at which the FTP server is running
- source - The source file to be downloaded
- target - The destination file name
- mode - The mode of transfer either '0 for ascii' or '1 for binary'
- protocol - The protocol to be used for uploading file
- user - The user name for the protocol connection
- password - The password of the user for the protocol connection

Refer to section **Managing Agents via RMI**, to learn more about managing the agent via RMI. Refer to section **Configuring Network Behavior**, to learn more about configuring behavior scripts.
Telnet Protocol Support

- Overview
- Starting Telnet
- Commands Supported
- Telnet Script API Methods
- Adding Telnet Commands
- Editing Telnet Scripts

Overview

The Network Simulator supports Telnet protocol for the Linux/Solaris devices in the device tree of the network simulator. You can also create your own SNMP devices with Telnet support. It enables execution of telnet commands on the simulated devices, from command line. To communicate with the simulated devices through telnet, you must log in to telnet by entering a valid user name and password.

Starting Telnet


(or)

Define your own SNMP devices in the device tree. Create a new network using the template devices. To create a new network, refer to Network Simulator Guide -> Creating a New Network.

The Network Simulator supports dynamic configuration of virtual ip addresses in Windows 2000/NT/XP and Linux/Solaris OS.

2. Right-click on the device in the network area and select the option Telnet -> Start Agent. This will automatically start the agent with the given IPAddress and at default port 2424.

3. To establish a telnet connection with the agent, go to command prompt and type: c:\> telnet <ip address> <port number>. Type the user name and password as root and public respectively.

4. The prompt displays the name of the device started. For example, if the device is a solaris device, the prompt will be displayed as follows:

[SunSolaris4@SunSolaris4 SunSolaris4]$ help

5. Type Help to list the commands that can be executed in this mode.

[SunSolaris4@SunSolaris4 SunSolaris4]$ help
exit
echo
date
help
is
pwd
rm
free
cp
Commands Supported

The commands supported in Linux and Solaris devices are given below. The script files (.py) executed for these commands are available in <SimulatorHome>/network/<networkname>/<devicename>/telnet directory.

<table>
<thead>
<tr>
<th>Command Description</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Displays a list of files and sub directories in a directory.</td>
<td>ls</td>
</tr>
<tr>
<td>Displays the system date</td>
<td>date</td>
</tr>
<tr>
<td>Displays the system available memory</td>
<td>free</td>
</tr>
<tr>
<td>Displays messages</td>
<td>echo</td>
</tr>
<tr>
<td>Copies one or more files from the source location to another location</td>
<td>cp &lt;sourcefile&gt; &lt;destination&gt;</td>
</tr>
<tr>
<td>Moves one or more files from one directory to another directory</td>
<td>mv &lt;sourcefile&gt; &lt;destination&gt;</td>
</tr>
<tr>
<td>Creates a directory</td>
<td>mkdir &lt;directory&gt;</td>
</tr>
<tr>
<td>To change the telnet working directory to specified one</td>
<td>cd &lt;directory&gt;</td>
</tr>
<tr>
<td>Deletes a file</td>
<td>rm &lt;filename&gt;</td>
</tr>
<tr>
<td>To display the hostname of the device.</td>
<td>hostname</td>
</tr>
<tr>
<td>To display the current working directory</td>
<td>pwd</td>
</tr>
<tr>
<td>To close the telnet connection</td>
<td>exit</td>
</tr>
</tbody>
</table>

Telnet Script API Methods

The Telnet script API methods are added to the ScriptInterface to perform operations such as execute directory specific commands, execute global commands, execute display commands etc. from a JPython script. The script API methods are used within the Jython scripts to implement the telnet commands.

The Telnet script API methods are provided under <simulator home>/examples/javadocs directory. The name of the script API file is TELNET_SimScrIntf.html

1. getTelnetRootDir()

This method returns the current working directory of the telnet client.

```java
public String getTelnetRootDir();
```

2. execDisplayCommand(String cmd)

This method is used in the execution of commands that result in some output display. The output will be specific to the directory from where the command is being executed.

For example, the output of the command "ls" results in the display of files and sub directories in a specific directory.

```java
public String execDisplayCommand(String cmd);
```
3. `executeGlobalCmd(String cmd)`

This method is used in the execution of commands that result in some output display. The output will remain globally similar, irrespective of the directory from where it is executed.

For example, the output of the "echo" command will be the same, irrespective of the directory from where it is executed.

```java
public String executeGlobalCmd(String cmd);
```

4. `executeTelnetCmd(String cmd)`

This method is used in the execution of commands that result in some action being executed in the system without any output display. The action executed will be within the specified directory.

For example, the output of the command "mkdir" creates a directory, within the specified directory.

```java
public void executeTelnetCmd(String cmd);
```

5. `execGlobalTelnetCmd(String cmd)`

This method is used in the execution of commands that result in some action being executed in the system without any output display. The action will be globally executed and not confined to any specific directory.

For example, the output of the "cd <directory>" command changes the telnet working directory to specified one.

```java
public void execGlobalTelnetCmd(String cmd);
```

6. `changeDirectory(String dir)`

To method is used to change the current working directory.

```java
public boolean changeDirectory(String dir);
```

7. `getDeviceName()`

This method gets the device name for which the request has come.

```java
public String getDeviceName();
```

8. `getTelnetCommands(int dev)`

This method returns all the commands supported by the device. For linux/solaris device, integer 1 must be specified and for Windows device integer 2 must be specified.

```java
public String getTelnetCommands(int dev);
```

**Note:** Supports only those system commands that are supported in Java.
Adding Telnet Commands

The Network Simulator provides the option to define your own telnet commands and add the required script files to a specific device or to all the devices in the network. The script files can be defined using the Telnet script API methods.

1. Right-click on the Telnet device in the device tree and select the option Add Telnet Commands. The Telnet Script Configuration dialog is displayed, wherein you have to specify the device details and command details.

2. Specify the device details:
   - The name of the selected device, database name and the configuration name is displayed and is not configurable.
   - Specify any of the following options
     - **Add command to device only**: Adds the telnet command only to the selected device type in the device tree.
     - **Add command to selected network**: Adds the telnet command to the devices of same type, in the network selected in the network tree.

3. Specify the command details:
   - **Command**: Specify the exact command to be added.
   - **Command Description**: Specify the description of the command
   - **Script File**: Browse and select the script file to be configured for the command.

4. Click Add to add the command to the specified devices.

Editing Telnet Scripts

The Telnet script editor allows you to view and edit the script file configured for the telnet commands. It gives a list of all the telnet commands supported by the device. Selecting a command will display the script file configured for the command. You can view and edit the script file as per your requirement.

Right-click on the device in the network which supports Telnet protocol and select Telnet Commands Script. The telnet commands Script Editor is invoked. It displays the commands and the scripts implemented for the commands.

The icon shows that a script file is configured for the command. For example, for the command cd, the script file cd_cmd.py will be implemented when the complete command cd <WORD> is entered in the telnet console, where <WORD> is the directory name.

The following options are provided in the script editor to edit and save the changes or revert back.

- **Revert**: To revert back the changes made in the current file.
- **Revert All**: To revert back the changes made in all the files.
- **Save**: To save the changes made in the current file.
- **Save All**: To save the changes made in all the files.
Managing Agents and Network via RMI

Simulation Toolkit provides RMI support in SNMP Agent Simulator, TL1 Agent Simulator and Network Simulator, to control the simulated agents and network via RMI, from a remote client program. The RMI client APIs defines the methods in the simulator which can be accessed via RMI.

This section covers the following topics:

- Overview
- Managing the Agent/Network via RMI
- Tutorials
Overview

- RMI : An Overview
- RMI implementation in Simulation Toolkit
- Advantages of using RMI in simulation

RMI : An Overview

Remote Method Invocation (RMI) enables the programmer to create distributed applications, in which the methods of remote Java objects can be invoked from other Java virtual machines, possibly on different hosts. It allows communication between programs written in Java.

An RMI client is a program that invokes one or more methods on a remote object in another virtual machine. For an RMI client to contact a remote RMI server, the client must first hold a reference to the server. The reference is obtained, either by looking up the remote object in the bootstrap naming service provided by RMI or by receiving the reference as an argument or a return value.

Every remote reference contains a server hostname and port number that allow clients to locate the VM that is serving a particular remote object. Once an RMI client has a remote reference, the client will use the hostname and port provided in the reference to open a socket connection to the remote server.

To summarize, RMI provides the mechanism by which the server and the client communicate and pass information back and forth.

RMI implementation in Simulation Toolkit

Simulation Toolkit has implemented RMI to control the simulated SNMP / TL1 agents and agents in the network via RMI. The RMI client API present under `<SimulatorHome>/examples/javadocs` directory, defines the methods in the simulator, which can be accessed via RMI.

To access the agent/network via RMI, you have to write the RMI Client program, invoking the methods provided in the RMI client APIs. The methods in RMI Client API can be invoked by setting the `AdventNetSimulatorClient.jar` present under the `<Simulator_Home>/tutorials/rmi` directory.

Some of the functionalities that can be achieved by using the methods in RMI Client API is given below:

- Ability to programmatically update values for a specific SNMP/TL1 device within the simulated network
- Ability to programmatically add/delete an SNMP agent in the existing network, get the number of SNMP agents running in the network and their device names.
- Ability to programmatically start/stop individual SNMP/TL1 devices in the network.
- Ability to programmatically send SNMP v1/v2/v3 traps and TL1 autonomous messages, on the fly.

Refer to the javadocs for more information.
Advantages of using RMI in simulation

1. RMI's unique capabilities to load and execute user-defined tasks at runtime helps in test case automation of manager applications. For example, the automated test environment can change the value of a given OID on a given SNMP agent to simulate different responses that is required for the test cases, trigger traps at any time, etc.

2. Results in enhanced performance as number of scripts need not be configured in the agent to simulate agent behavior. RMI provides means for controlling the simulated agent and network behavior.
Managing the Agent / Network via RMI

- Overview
- Starting the RMI Registry
- Registering the Object with the RMI Registry
- Starting the RMI Client

Overview

Running the RMI system and managing the agent via RMI, involves starting the RMI registry, starting the RMI server (agent / network which is accessible via RMI) and starting the RMI Client.

The RMI client can access the RMI server, only if the remote object is created and references to them are accessible. Simulation Toolkit provides option to make the simulated SNMP/TL1 agent and the network devices accessible through RMI.

In this topic, we will learn to start the RMI registry, make the agent accessible through RMI, start the RMI server, and start the client program which invokes the client API methods to access the RMI server.

Starting the RMI Registry

For obtaining references to remote objects, RMI provides a simple remote object registry interface, implemented by RMI's rmiregistry, that provides methods for storing and retrieving remote object references. A server registers its remote objects with the RMI registry.

To start the RMI registry :

1. Go to <Simulator_Home>/bin/cmdline directory.
2. Execute the StartRMIRegistry.bat file for Windows OS and StartRMIRegistry.sh file for Linux OS.
3. This command starts the RMI registry in the default port 1099.

Registering the Object with the RMI Registry

From UI

Making the SNMP agent accessible through RMI and starting the agent :

1. Start the SNMP Agent Simulator by invoking SnmpAgentSimulator.bat file for Windows OS and SnmpAgentSimulator.sh file for Linux OS from <Simulator_Home>/bin directory.
2. Load all the desired MIBs and simulate default values.
3. Choose the menu option Settings --> Agent Settings . This will invoke the Agent Settings dialog.
4. Select the check box Manage using RMI.
5. To start the SNMP agent which is made accessible via RMI, select Operations --> Start Agent option in the menu bar of the SNMP Agent Simulator. Now, when the SNMP agent is started it will be accessible via RMI.
Making **TL1 agent** accessible through RMI and starting the agent:

1. Start the TL1 Agent Simulator by invoking `TL1AgentSimulator.bat` file for Windows OS and `TL1AgentSimulator.sh` file for Linux OS from `<Simulator_Home>/bin` directory.
2. Load all the desired TCS and simulate default values.
3. Choose the menu option **Settings --> Agent Settings**. This will invoke the **Agent Settings** dialog.
4. Select the check box **Manage using RMI**.
5. To start the TL1 agent which is made accessible via RMI, select **Operations --> Start** option in the menu bar of the TL1 Agent Simulator. Now when the TL1 Agent is started it will be accessible via RMI.

Making the **network devices** accessible through RMI and starting the network:

1. Start the Network Designer by invoking `NetworkDesigner.bat` file for Windows OS and `NetworkDesigner.sh` file for Linux OS from `<Simulator_Home>/bin` directory.
2. Load all the desired MIBs and simulate default values.
3. Choose the menu option **Settings --> Runtime Settings**. This will invoke the **Agent Settings** dialog.
4. Select the check box **Manage using RMI**.
5. To start the network which is accessible via RMI, select **Run --> Start Network** option in the menu bar of the Network Designer. Now when the network is started, all the devices in the network will be accessible via RMI.

**From Command line**

- To start the simulated **SNMP agent** from command line and manage via RMI, type the following command from the `<Simulator_Home>/bin/cmdline` directory.

  ```bash
  StartSnmpAgent.sh -m RFC1213.MIB -p 8005 -rmi
  ```

  Refer to the topic, Starting the SNMP Agent from Command Line, to learn more about the command line options provided.

- To start the simulated **TL1 agent** from command line and manage via RMI, type the following command from `<Simulator_Home>/bin/cmdline` directory.

  ```bash
  StartTL1Agent.sh -cs .\tcs\Acme-MSU.tcs -p 9002 -rmi
  ```

  Refer to the topic, Starting the TL1 Agent from Command Line, to learn more about the command line options provided.

- To start the simulated **network** from command line and manage via RMI, type the following command from `<Simulator_Home>/bin/cmdline` directory.

  ```bash
  StartNetwork -snmp < SNMP text file with path > -tl1 < TL1 text file with path> -rmi
  ```

  Refer to the topic, Starting the Network from Command Line, to learn more about the command line options provided.

**Starting the RMI Client**

1. Write the RMI Client program, invoking the methods in RMI client API. The javadocs of the RMI client api methods that can be accessed via RMI is available in `SimScrRmiIntf.html` in `<Simulator_Home>/examples/javadocs` directory.

2. You can invoke methods in RMI Client API by setting the `AdventNetSimulatorClient.jar` present under the `<Simulator_Home>/tutorials/rmi/client/jars` directory. Procedure to write the RMI client program is given below:
• Create the instance of the Client API Class.
• Invoke the method `getObjectRef(host, port, objectName)`. This will return a boolean to denote that the object is fetched successfully from the RMI Server or the object is not fetched successfully.

3. Start the RMI Client program. If the method `getObjectRef(host, port, objectName)` returns a boolean, then you can use the rest of the methods available with the RMI client API and manage the simulated agent / network devices.
RMI Tutorials

- Introduction
- Application Overview
- Working with the Tutorial
  - Starting the Agent
  - Starting the RMI Client

Introduction

This tutorial will guide you through designing the RMI client program and accessing the simulated agent by providing working, illustrative examples. It describes how the simulated SNMP and TL1 agents can be accessed and controlled from a remote system using the RMI client APIs bundled with Simulation Toolkit.

Application Overview

The RMI tutorial is available in `<SimulatorHome>/tutorials/rmi` directory. The client folder contains sample RMI client program which can be run on any remote system. The simulator folder contains the script files to start the simulated agents that can be managed via RMI.

The RMI client program invokes some of the RMI client API methods in the simulator to control the simulated agents running in another system.

Working with the Tutorial

In this section, the steps to start the simulated agent and the RMI client programs are provided. This would help you run the application by yourself and view the results.

Starting the Agent

The `<SimulatorHome>/tutorials/rmi/simulator/bin` directory contains the batch/shell files to start the simulated SNMP and TL1 agents with RMI support.

**Starting the Standalone SNMP Agent with RMI support**

Execute the following command from `<SimulatorHome>/tutorials/rmi/simulator/bin` directory.

```
StartAgentSimulator.bat/sh - snmp - c <configuration filename> -tn <database table name>
```

(or)

```
StartAgentSimulator.bat/sh - snmp - m <MIB filename> -tn <database table name>
```

The above command will start the simulated SNMP agent at port 8001, that can be managed via RMI. It will also start the RMI registry at port number 1099 at localhost.
Starting the Standalone TL1 Agent with RMI support

1. Execute the following command from <SimulatorHome>/tutorials/rmi/simulator/bin directory.

StartAgentSimulator.bat/sh - tl1 - c <configuration filename> -tn <database table name>

(or)

StartAgentSimulator.bat/sh - tl1 - t <TCS filename> -tn <database table name>

The above commands will start the simulated TL1 agent at port 9099, that can be managed via RMI. It will also start the RMI registry at port number 1099 at localhost.

Starting the Simulated Network with RMI support

Batch/Shell files are not provided to start the network with RMI support. This can be done from the Network Designer UI:

1. Go to <Simulator_Home>/bin/cmdline directory.
2. Execute the **StartRMIRegistry.bat** file for Windows OS and **StartRMIRegistry.sh** file for Linux OS. This command starts the RMI registry at the default port 1099.
3. Create the network with the required number of devices in the Network Designer.
4. Select the option **Manage using RMI** in the **Settings -> Runtime Settings** dialog.
5. Start the network. Now when the network is started, all the devices in the network will be accessible via RMI.

Starting the RMI Client

The RMI client folder available in <SimulatorHome>/tutorials/rmi directory contains the RMI client programs and RMI client APIs.

The following files and directories are available under the client folder:

- **bin**: The batch/shell scripts to execute the RMI client programs.
- **conf**: The **RMIParameters.conf** file which contains the RMI parameters for the registration of the simulator object to the RMI registry.
- **source**: **SimulatorRMIClient.java** program gets the simulator object reference and executes the RMI API methods.
- **jar**: The AdventNetSimulatorClient.jar file contains the class files of the above specified java files and the RMI client API implementation.

**Note**: You can create a zip of the client folder, extract and run it in any other system separately.

To run the client program,

1. Edit the **RMIParameters.conf** file available in the **conf** folder, with the host name and port number of the system, where the simulated agent is running.
2. Run the **executeScript.bat/sh** command from <SimulatorHome>/tutorials/rmi/client/bin directory. The SimulatorRMIClient gets the simulator object reference and executes the sample RMI API methods based on the reference obtained.
Note:

1. Refer to the SimulatorRMIClient.java file in <SimulatorHome>/tutorials/rmi/source directory, to learn how to write your own RMI client programs.

2. The javadocs of the RMI client api methods that can be accessed via RMI is available in SimScrRmiIntf.html in <Simulator_Home>/examples/javadocs directory.

3. If any changes are made in the source files, compile the source files and keep the class files in the classpath.
Recorder Guide

Network Recorder

- Overview
- Recording a Network from UI
- Recording a Network from Command Line
- Directory Structure of the Recorded Network
- Loading the network in simulator
- Error Messages

Overview

The network recorder allows to easily create simulations of actual devices on your network. It enables users to automatically simulate the behavior of actual devices on their network, by capturing a "snapshot" of the device MIB in actual operation. Any number of target devices can be recorded simultaneously by providing the starting and ending IP address.

The recorded network can be saved and replayed instantly in the Network Designer. Any variation on the recorded data can be created for use with the Network Designer tool to recreate problems or to test configuration variations. Network recorder is supported both in UI and command line.

Recording a Network from UI

1. Start the Network Recorder by invoking NetworkRecorder.bat/sh from <SimulatorHome>/bin directory.
2. Enter the following details in the UI.
   - **IP address**: Specify the starting and ending IP address of the devices to be recorded. All the SNMP devices in the specified range running at the specified port number will be recorded.
   - **MIBs directory**: Specify the MIBs directory which contains all the MIBs implemented by the devices with the specified IP address. You can browse and select the required directory.
   - **Port number**: The network recorder talks to the standard SNMP port 161 on the target device. If the device is configured to use a non-standard port, you can specify it in this option. If the device with the specified IP address does not exist or is not running at the specified port number, the device will not be recorded.
   - **Community name**: Specify the community name to be assigned to the recorded devices. The default community name is public.
   - **Network name**: Specify the name of the network to be recorded. The network will be recorded in <Simulator_Home>/networks/<network name> directory.
   - **Timeout**: The minimum time to wait for a reply from the agent (in seconds).
   - **Retries**: Specify the number of retries to be made while discovering the devices.
   - **Version**: Specify the SNMP version of the devices in the network. The support versions are V1, V2c and V3. The selected version will be applicable for all the devices in the network.
     - If the selected version is v3, specify the v3 user details like user name, context name, security level, auth / priv protocol values and password in the
v3recording.conf file available in <SimulatorHome>/conf directory. The specified v3 details will be applicable for all the devices in the network.

- The recorder connects to the MySQL bundled with the product with the default settings as specified in the database_params.conf file in <SimulatorHome>/conf directory. The details of the database server, port, user name, password and server path is stored in this conf file.

3. Click Record to record the SNMP devices running at the specified port with the specified IP address. When recording is completed, the following message is displayed.

The log file snmp_network_recorder_<num>.log in <SimulatorHome>/logs directory captures the log messages like details of the devices not recorded and the possible reasons. For example: “The device is not a snmp device or may be out of service 192.168.1.41.

Note: The DeviceOid.prop file in <SimulatorHome>/conf folder contains the sysObjectld variable and the name of the device. During network recording, the network device is assigned a name, based on the sysObjectld value specified in this file for the specific device.
Recording a Network from Command Line

Start the command line network recorder by executing NetworkRecorder.bat/sh from <SimulatorHome>/bin/cmdline directory, with the following options. The recorder connects to the MySQL bundled with the product with the default settings as specified in the database_params.conf file in <SimulatorHome>/conf directory. The details of the database server, port, username, password and server path is stored in this conf file.

**Mandatory Options:**

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-start &lt;starting IP address&gt;</td>
<td>The starting IP address of the network to be recorded</td>
</tr>
<tr>
<td>-end &lt;ending IP address&gt;</td>
<td>The ending IP address of the network to be recorded</td>
</tr>
<tr>
<td>-p &lt;port number&gt;</td>
<td>The port number at which the real devices of the network are running. The default port is 161</td>
</tr>
<tr>
<td>-dir &lt;MIBs dir&gt;</td>
<td>The absolute path of the directory containing all the MIBs implemented by the devices in the network.</td>
</tr>
<tr>
<td>-name &lt;network name&gt;</td>
<td>The name of the network to be recorded. The network will be recorded in &lt;Simulator_Home&gt;/networks/&lt;network name&gt; directory.</td>
</tr>
</tbody>
</table>

**Optional Options:**

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-c &lt;community&gt;</td>
<td>The community of the SNMP device</td>
</tr>
<tr>
<td>-timeOut</td>
<td>The minimum time to wait for a reply from the agent (in microseconds).</td>
</tr>
<tr>
<td>-v v1</td>
<td>v2c</td>
</tr>
<tr>
<td>-retries</td>
<td>The number of retries to be made while discovering the device in the network.</td>
</tr>
</tbody>
</table>

**Example**

NetworkRecorder.bat -start 192.168.1.40 -end 192.168.1.50 -p 161 -dir C:\AdventNet\simulator4\mibs -name sampleNetwork -c public -retries 3

**Directory Structure of the Recorded network**

When you record a network, the following folders will be created in <Simulator_Home>/networks directory, with the following structure:

- **<Network Name>** : Consists of the main file with filename <network name>.net.
  - **<mibs>** : Contains all the MIB files implemented by the SNMP agents in the network.
  - **<device name>** : For each device recorded in the network, a folder is created with the device name.
    - **<snmp>** : If the device implements an SNMP agent, the main file `<device name>_main.xml` will be generated in this folder.

The database tables will be generated in <Simulator_Home>/mysql/data/<network name>_1 directory. If the data exceeds 500 tables, a new database will be created for the same network in <Simulator_Home>/mysql/data/<network name>_2 directory.
Loading the Network in Simulator

The recorded network can be loaded in the network designer and any variation on the recorded data can be created in the network designer tool to recreate problems or to test configuration variations.

To load the recorded network,

- Start the Network Designer by invoking `NetworkDesigner.bat/sh` from `<Simulator_Home>/bin` directory.
- To load the recorded network, select `Network -> Load` option from the menu bar. The `Load Network` dialog appears. Browse and select the `<network name>.net` file from `<Simulator_Home>/networks` directory.
- Click `Open` to Load the Network, along with its configuration.

Error Messages

The log messages are captured in `snmp_network_recorder_<num>.log` and error messages are captured in the `snmp_network_recorder_<num>.err` in `<SimulatorHome>/logs` directory. Some of the error messages are given below:

1. **Problem in loading the MIBs in the directory provided**
   This message is displayed if the MIB file contains the .cmi and .cds extensions or if the required MIB file does not exist.

2. **The database server is not started with specified values**
   This message is displayed if the database server is not started at localhost with user name as root and empty password at port 3306. The database path is `<SimulatorHome>/mysql/bin` directory.

3. **No device discovered in the network**
   This message is displayed if the recorder is unable to discover any device specified in the starting and ending ip address range, at the specified port number.
Record and Replay Traps

- Overview
- Recording Traps in Trap Recorder
- Storing the Recorded Traps in XML files
- Replaying Traps in Simulator

Overview

The trap recorder records SNMP v1 and SNMP v2c traps by listening for traps at the specified port. All the traps sent to the port will be recorded and displayed. The output of the Trap Recorder will give you the traps in the order they were received, with time stamps and the variables included in the trap PDU.

The recorded traps can be stored in XML files and replayed in the simulator as request-based, threshold-based or time-based traps.

Recording Traps in Trap Recorder

1. Start the Trap Recorder by invoking `SnmpTrapRecorder.bat/sh` from `<SimulatorHome>/bin` directory.
2. Enter the following details in the UI.
   - Specify the port number from where the traps will be recorded
   - Specify the community of the agent from which you want to record traps.
3. Click Add to add the port number to the trap list.
4. Click Start. The Trap Recorder starts listening for traps and you can view the recorded traps.

Storing the Recorded Traps in XML Files

To store the recorded traps in XML files,

1. Select Save option in the Trap Recorder dialog. This prompts a dialog to enter the file name.
2. Enter the filename and click OK button. The file will be saved in XML format in `<SimulatorHome>/rectraps` directory.

Note:

1. Traps duplicated will be recorded and displayed in the Trap recorder dialog but will not be saved in the XML file.
2. The recorder saves the entries to a new file each time the Save option is selected. Hence, you cannot save entries to a file which already exists.
3. SNMPv3 traps are not recorded.

Replaying Traps in Simulator

The recorded traps which are stored in XML files can be loaded in the trap configuration dialog of the SNMP Agent Simulator and Network Designer as request-based, threshold-based and timer-based traps.
To replay the recorded traps in SNMP agent simulator / Network designer,

1. Start the SNMP Agent Simulator / Network Designer

2. Invoke the Trap Configurations dialog by selecting Configure -> Traps option from the menu bar.

3. Select the trap type as request-based, threshold-based or timer-based from the icons displayed on the right-side. Click Add to configure traps The Trap Configuration wizard is displayed.

4. For request-based and threshold-based traps, browse and select the required scalar/columnar node from the MIB tree and click OK. The OID of the selected node will be displayed as the trap variable.

5. Click Load to load the recorded trap configuration in this dialog.

6. The Recorded Traps dialog displays all the recorded traps file. Selecting a file will display all the recorded trap entries in the file. Select a trap entry and click OK.

7. The recorded trap configuration will be displayed in the Trap Configurations dialog. (i.e) the trap version, generic type, specific type, enterprise OID and variable bindings will be updated.

8. Specify the other configurations and click OK to add the entry to the list.

Refer to Configuring Traps, to know more about configuring traps.
Simulating Large Networks

- Overview
- System Requirements
- Changing the System Settings in Linux
- Creating and Adding Devices to the Network
- Setting Optimum Heap size
- Starting the Network
- Packaging and Installing the Network
- Performance of the Network

Overview

AdventNet Simulation Toolkit currently simulates 50,000+ device agents in one Linux box and 10,000 device agents in Windows. The aim of this topic is to explore the capabilities of Simulation Toolkit to simulate a large network with ease at the shortest possible time.

Linux-based systems can provide significant simulation speedup and performance improvements over windows. Scripts to configure Virtual IPs in bulk in Linux OS, is also provided with the toolkit. MySQL provides better performance in Linux than in Windows. Hence, to simulate large networks, we prefer and also recommend you to use Linux OS.

In this topic, you will learn how to use the toolkit to create a large network with 50000 SNMP manageable devices in one Linux box. The scaling tips that can be followed while configuring a very large network is covered in the topic.

System Requirements

Configuration of the system where Simulation Toolkit is running : The optimal system requirements are given below:

- Hard Disk : 40 GB
- RAM : 512 MB (The actual requirement depends on the amount of data queried)
- OS : Linux 7.2

Configuration of the system where MySQL and Manager application is running : This is the machine where the simulated data of the devices are stored.

- Hard Disk : 40 GB
- RAM : 512 MB
- OS : Linux 7.2

Changing the System Settings in Linux

When running on Linux 2.2 kernels, each process is limited by default to 1024 file descriptors. A file descriptor is an object that a program uses to read or write an open file, or open network socket.

For simulating 50000 nodes, increase the maximum file descriptors per process as per the steps given below: You should log in as root to edit the configuration files. The steps given below, will guide through the configuration process.
1. In `/etc/security/limits.conf`, add the lines:

   * soft nofile 102400
   * hard nofile 409600

   **Note:** Hard limit on the number of files is the maximum number of file descriptors per process.

   Conditions:
   1. The Soft limit should be less than hard limit (we used soft limit = hardlimit / 4 for testing).
   2. Entry "soft nofile 102400" says that soft limit on the number of open files is 102400.
   3. Entry "hard nofile 409600" says that hard limit on the number of open files (maximum) is 409600.

2. In `/etc/pam.d/login`, add the following line:

   session required /lib/security/pam_limits.so

3. Add the following line in the `/etc/sysctl.conf` file:

   # Increase system-wide file descriptor limit.
   fs.file-max = 102400
   fs.inode-max = 409600

   **Note:**
   - `fs.file-max` = maximum limit on the number of file descriptors.
   - `fs.inode-max` = maximum number of inodes.
   - Condition `fs.inode-max = 4 * fs.file-max`.

4. Add the following lines in `/root/.bashrc`

   ulimit -n 409600

   **Note:** In this file the above value should be specified inside the IF condition.

5. Reboot the machine for this settings to take effect.

**Creating and Adding Devices to the Network**

A network with 50000+ devices can be created using the Network Designer tool.

- Start Network Designer by invoking `NetworkDesigner.sh` from `<SimulatorHome>/bin` directory, for Linux OS.
- You can either use the pre-configured devices in the device tree or add your own devices to the device tree. Refer to section Adding Devices in Device Tree, to create your own devices.
- Add devices in bulk using Network Design Wizard or the Random Device Addition wizard in the Tools menu. Refer Creating a new network for more information.
- The devices will be assigned a unique ipaddress and port number combination.
- The virtual ipaddress assigned to the devices are dynamically configured in your system, when starting each agent instance in the network.
- The Unique Name of the device will be updated as the `sysName` in the system group of RFC1213-MIB for SNMP device.
- The specified IP address will be updated in `ipAddrTable` of `RFC1213-MIB` for SNMP devices.
- A unique `mac address` will be assigned to each device and updated in `ifTable` -> `ifPhysAddress`. 
Setting Optimum Heap size

For enhanced network performance, you can set an optimum heap size of the JVM based on the number of OIDs registered. By default, the network simulator sets the default heap size of the simulator as 200M. For a 50000 node network you can set the heap size as 1600M.

If the network is started from Network Designer UI, the heap size can be set as follows:
- Edit NetworkDesigner.bat/sh file in <SimulatorHome>/bin directory. By default, the heap size of the simulator as 200M. Edit this file to set the heap size as 1600M.

For example, to set the heap size as 1600M, replace "java -mx200M" with "java -mx1600M".

If the network is started from command line, the heap size can be set as follows:
Open the StartNetwork.sh present under the <simulator home>/bin/cmdline directory and edit the -mx value as 1600M. By default it will be 200M.

Starting the Network

The Network that is designed through the Network Designer can be started from the UI or started from the command line.

Starting the network From UI :
Advantages of Starting the network from UI :
1. Configure traps, behavior scripts and traps at runtime.
2. Option to Stop and Start individual agents in the network.
3. View the properties of a network and individual devices in the network.
4. Delete a device from the network.

To start the network from UI select Run -> Start Network option.

Starting the network From Command Line :
To start the network from command line :
$ Sh StartNetwork -net samplenetwork.net -m <specify debug level as 1 or 2> -l <to log the messages>

Note: The virtual ip address assigned to the devices are dynamically configured in your system, when starting each agent instance in the network.

Packaging and Installing the Network

The Network Designer also provides option to package and install the network you have created in any other system having Simulation Toolkit. Refer to the section "Porting your network" to learn more about the package and install options.

The network ported to another system can be started right away after doing the following changes :
1. Configuring the Virtual IP in the system.
2. Modifying the Virtual IP in the network using the Bulk Modification option.

Performance of the Network

Refer to the Performance Guide, to learn about the performance statistics of the Network Simulator.
Performance Guide

- Overview
- Performance Numbers for Request Handling in 250 Node Network
- Performance Numbers for Request Handling in 10000 Node Network
- Performance Numbers for Trap Handling in SNMP Trap Stormer
- Performance Numbers for Trap Handling in Network Simulator

Overview

This section gives the performance statistics of the Network Simulator in 6.0. The performance test were conducted separately for a 250 node network and 100000 node network. The environment in which the tests were performed, the hardware configurations and the performance numbers for request handling and trap handling are provided in this topic.

Performance for Request Handling in 250 Node Network

Test Environment:

The performance test was conducted on Windows and Linux machines.

System Requirement:

Processor: 2.4 GHz
RAM: 512 MB
Hard Disk: 20 GB

Performance Numbers:

<table>
<thead>
<tr>
<th>Number of nodes queried Asynchronously</th>
<th>Number of OIDs queried Asynchronously</th>
<th>Number of response processed in Windows OS</th>
<th>Number of response processed in Linux OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 nodes</td>
<td>Single OID</td>
<td>400 PDU /sec</td>
<td>833 PDU / Sec</td>
</tr>
<tr>
<td>100 nodes</td>
<td>Multi-varbind request with 6 OIDs (2 Table node and 4 Scalar nodes)</td>
<td>217 PDU /sec</td>
<td>444 PDU / Sec</td>
</tr>
</tbody>
</table>

Performance for Request Handling in 10000 Node Network

Test Environment:

The performance test was conducted on Windows and Linux machines.

System Requirement:

Processor: 2.4 GHz
RAM: 512 MB
Hard Disk: 20 GB
Performance Numbers:

<table>
<thead>
<tr>
<th>Number of nodes queried Asynchronously</th>
<th>Number of OIDs queried Asynchronously</th>
<th>Number of response processed in Windows OS</th>
<th>Number of response processed in Linux OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 nodes</td>
<td>Single OID</td>
<td>110 PDU /sec</td>
<td>400 PDU / Sec</td>
</tr>
<tr>
<td>100 nodes</td>
<td>Multi-varbind request with 6 OIDs (2 Table node and 4 Scalar nodes)</td>
<td>90 PDU /sec</td>
<td>294 PDU / Sec</td>
</tr>
</tbody>
</table>

Performance Numbers for Trap Handling in SNMP Trap Stormer

From Trap Stormer:

The Trap Stormer is used to test the performance of trap handling in Simulation Toolkit. The performance of the trap is calculated using the Trap Stormer in normal mode.

<table>
<thead>
<tr>
<th>Trap Version</th>
<th>Varbinds Configured</th>
<th>Rate of Traps in Windows OS in Normal mode</th>
<th>Rate of Traps in Linux OS in Normal mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1 Traps</td>
<td>with 2 Varbinds</td>
<td>900 / sec</td>
<td>935 / sec</td>
</tr>
<tr>
<td>V2C Traps</td>
<td>with 3 Varbinds + default Varbinds</td>
<td>750 / sec</td>
<td>870 / sec</td>
</tr>
</tbody>
</table>

Performance Numbers for Trap Handling in Network Simulator

The performance of traps sent from Network Simulator is given below:

<table>
<thead>
<tr>
<th>Trap Version</th>
<th>Varbinds Configured</th>
<th>Rate of Traps in Windows OS</th>
<th>Rate of Traps in Linux OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>V1 Traps</td>
<td>No Varbinds configured</td>
<td>550 / sec</td>
<td>670 / sec</td>
</tr>
<tr>
<td>V1 Traps</td>
<td>with 2 Varbinds</td>
<td>455 / sec</td>
<td>580 / sec</td>
</tr>
<tr>
<td>V2C Traps</td>
<td>with default Varbinds (2 Varbinds)</td>
<td>410 / sec</td>
<td>610 / sec</td>
</tr>
<tr>
<td>V2C Traps</td>
<td>with 3 Varbinds + default Varbinds</td>
<td>330 / sec</td>
<td>450 / sec</td>
</tr>
</tbody>
</table>
SNMP Trap Stormer

- Overview
- Starting Trap Stormer
- Configuring Traps in Trap Stormer
- Configuring Agent IP Address
- Sending and Viewing Traps
- Saving and Loading the Configuration
- Command Line SNMP Trap Stormer

Overview

The SNMP Trap Stormer is a simple, intuitive, easy-to-use tool, exclusively designed to configure and send SNMP v1/v2c traps. Using this tool you can configure to send any number of traps at the specified time-interval or at the burst mode, without specifying the time-interval.

This tool is capable of sending a maximum of 1800 v1/v2 traps per second in the burst mode with 4 varbinds and 400 traps per second with 2 varbinds in the normal mode, in Linux OS with 2.4 GHz.

You can use this tool to test the reliability of your management application for receiving any number of traps at the specified time interval or at the burst mode.

Starting Trap Stormer

To start the Trap Stormer

- Go to `<Simulator_Home>/bin` directory.
- Invoke the `SnmpTrapStormer.bat` for Windows OS and `SnmpTrapStormer.sh` for Linux/Solaris OS.

Configuring Traps in SNMP Trap Stormer

1. Select the Table -> Add option on the menu bar. The Trap Configuration dialog is displayed.
2. Specify the following Manager Details :
   - Manager Name : Specify the manager name. If the Manager is running on the same machine it will contain the value localhost. If the Manager is running on a different machine specify the manager name or IP address of that machine.
   - Manager Port : The default manage port is 162. If the manager is running on a different port, specify the port number.
   - Community : Specify the community name of the manager.
3. Specify the following Trap Details :
   - Trap Version : Select the Trap Version from the list displayed. If the trap version is v1, the trap type must be specified as either Generic type or Specific type. The default trap version is SNMPv1.
   - Generic Type : If the trap type value is zero through five, then the trap is one of the generic SNMP traps defined by the snmp group of MIB-II and the Specific trap type will be zero. If the trap type value is six, then the trap is an enterprise-specific trap that is defined in a private MIB.
O coldStart (0),
O warmStart (1),
O linkDown (2),
O linkUp (3),
O authenticationFailure (4),
O egpNeighborLoss (5),
O enterpriseSpecific (6).

- Specific Type: Specify an integer value for the enterprise-specific trap that is defined in a private MIB.
- Enterprise OID: The OID specified in the Enterprises field indicates exactly which management enterprise defines the trap.

4. Configure Varbinds for the trap: This option will be enabled only if MIBs are loaded in the trap stormer. To load MIBs, select File -> Load MIB option. The loaded mibs will be displayed in the UI for reference. Variable Bindings or Varbinds are additional information that is sent to the manager along with the trap. It can be value of the selected MIB nodes. This value is sent to the manager along with the trap or Inform. To configure variable bindings,

- Click Configure button. The Configure Varbinds dialog pops up with the MIB tree displayed on the right-side.
- Select a varbind (only leaf nodes) in the MIB tree and click Add. The Configure varbinds dialog pops up displaying the OID and instance of the selected varbind.
- Select the varbind value as either agent value or custom value. If custom value is selected, the value must be specified in the appropriate format based on the datatype of the selected node.
- Click OK to add the varbind(s).

5. Select the option Send All in One Burst, to send the configured traps, the specified number of times, at one burst. If this option is selected the Time interval option will be disabled.

6. Specify the Number of Times you want to send the configured trap. For example, 100.

7. Specify the Time Interval between the traps that are to be sent. The time interval option is available in Milliseconds. For example, 1000 Milliseconds.

8. To send traps with the configured agent ip address, select the Device Type. Refer to Configuring Agent IP address.

9. To modify the trap configuration, select the trap entry and select the option Table -> Modify, to modify the entry. Alternatively, you can also double-click on the trap to modify the same.

10. To delete an existing trap configuration, select the trap entry and select the option Table -> Delete.

Configuring Agent IP Address

By default, traps will be sent with the IP Address or host name of the system from which the traps are generated.

The Trap Stormer provides option to configure "Device Types" and multiple "Agent IP addresses" for each device type. When traps are configured, each trap can be assigned to a specific device type. Traps will be sent with all the ip address configured for the device type.
• Select **Table -> Agent Details** option.
• Select **Add Device**, specify a new device type and click OK
• Specify the starting **IP address** and the **number of IPs** to be created for the device type. Click **Add**. The specified number of IPs will be created by incrementing the starting ip address.

**Sending and Viewing Traps**

To send the configured traps,
1. Select the trap entry/entries to be sent, from the SNMP Trap Stormer dialog.
2. Click **Send** button or choose **Run -> Send** option from the menu bar. This will start sending the configured traps for the specified number of times at the specified time interval. To send multiple traps, select multiple traps entries from the table and select Send.
3. To stop sending traps, click **Run -> Stop** option from the menu bar.
4. To stop sending all the traps, select **Run -> Stop All** option from the menu bar.

The traps can be viewed from the Manager application at the specified host name and Trap port number. For example, to view traps from the Mib Browser application bundled with Simulation Toolkit, **the Trap Viewer must be started before sending the traps.**

To start the Trap Viewer
1. Select **View -> Trap Viewer** option from the menu bar or select the Trap viewer icon on the toolbar. This will invoke the trap viewer.
2. Specify the Port number and Community. Add the Port number to the TrapList by selecting the **Add** button.
3. Select the **Start** button. Now, the Trap viewer is started and listening for traps at the specified port.

**Saving and Loading the Configuration**

The configured traps can be saved in XML files and later loaded in the trap stormer tool.

To save the traps,
1. Select **File -> Save** option.
2. Specify the required file name and click **OK**.
3. The configuration will be saved in the specified file in XML format in `<SimulatorHome>/trapstormer` directory.

To load the file in the Trap Stormer,
1. Select **File -> Load** option.
2. Browse and select the required file from `<SimulatorHome>/trapstormer` directory.

**Command Line SNMP Trap Stormer**

The command line SNMP Trap Stormer is a non GUI tool that can be used to configure and send SNMP v1/v2 traps from command line. To start the command line Trap Stormer, execute the shell/bat file **StormTraps.sh** / **StormTraps.bat** present under the `<Simulator_Home>/bin/cmdline` directory with the following options.

**Options:**

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>-b</td>
<td>To send traps at burst Mode &lt;true / false&gt;</td>
</tr>
<tr>
<td>-v</td>
<td>Version of the Trap &lt; v1 or v2 &gt;</td>
</tr>
<tr>
<td>-n</td>
<td>Number of traps that has to be sent.</td>
</tr>
</tbody>
</table>
### Options and Explanations

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-mn</code></td>
<td>The list of manager names to whom the traps must be sent in the following format: <code>mgr1,mgr2,mgr3</code>. (Even if the managers are same they must be specified separately for each port. For example, <code>-mn a, a, a -mp p1, p2, p3</code>)</td>
</tr>
<tr>
<td><code>-mp</code></td>
<td>The port number of the managers specified in the <code>-mn</code> option. It should be in the following format: <code>-p1, p2, p3</code> (Even if the managers have the same port number, they must be specified for each manager separately)</td>
</tr>
<tr>
<td><code>-mc</code></td>
<td>The Community of the manager.</td>
</tr>
<tr>
<td><code>-var</code></td>
<td>Varbinds Property File name with absolute path. The property file must be in the following format: <code>&lt;varbind name&gt; : &lt;oid&gt;, &lt;instance&gt;, &lt;data type&gt;, &lt;varbind value&gt;</code> The data type must be specified by the following values given below: STRING - 4 OBJID - 6 INTEGER - 2 NULLOBJ - 5 IPADDRESS - 64 NETWORKADDRESS - 64 COUNTER - 65 GAUGE - 66 UNSIGNED32 - 66 TIMETICKS - 67 OPAQUE - 68 COUNTER64 - 70 UINT32 - 66</td>
</tr>
<tr>
<td><code>-f</code></td>
<td>The from IP Address of the Trap.</td>
</tr>
</tbody>
</table>

#### Option to be specified for SNMPv1 Traps:

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-g</code></td>
<td>Generic Type of the trap (0 ... 6)</td>
</tr>
<tr>
<td><code>-s</code></td>
<td>Specific Type of the trap (if generic type is 6)</td>
</tr>
<tr>
<td><code>-e</code></td>
<td>Enterprise OID (if generic type is 6)</td>
</tr>
</tbody>
</table>

#### Option to be specified for SNMPv2 Traps:

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-t</code></td>
<td>SNMP Trap OID</td>
</tr>
</tbody>
</table>

#### Option to be specified if traps are to be sent at non-burst mode without `-b` option:

<table>
<thead>
<tr>
<th>Options</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-i</code></td>
<td>Time Interval between Traps. The time interval option must be specified in <strong>Milliseconds</strong>. For example, 1000 Milliseconds.</td>
</tr>
</tbody>
</table>
SNMP Proxy Agent Simulation

- Overview
- Creating the Agent Configurations
- Creating the Mapping Conf File
- Starting the Proxy Agent from Command line
- Testing Proxy Agent Simulation

Overview

In real-time environment, a device having IP Address and SNMP Agent may be acting as a proxy for "non-SNMP" devices managing its own MIB. These non-SNMP devices will be addressed by the proxy agent using community string. The SNMP Agent Simulator supports to simulate such scenarios, where there are multiple SNMP agents with same IP Address, but with different community strings. This feature is supported in command line.

Creating the Agent Configuration

The Proxy Agent takes the SNMP agent configuration file as input. Hence, each SNMP agent must have the configuration file with `<filename>_main.xml` extension, which contains the simulated agent configurations. The configuration files must be created using the AdventNet SNMP Agent Simulator.

Creating the Mapping Conf File

The mapping information required to access the simulated SNMP agents based on community name must be provided in a text file format. The Proxy Agent uses this information to route the messages to the appropriate agents. The format of the text file is given below:

**Format:**

```
community name=configuration file name with path|agent id
```

where,

- **community name** - Specify the agent community name. This will be used by the Proxy agent to forward the SNMP requests.
- **configuration file name with path** - the configuration file `<filename>_main.xml>` which contains the simulated agent configurations of the agents. This file must be created using SNMP Agent Simulator or Network Simulator
- **agent id** - a unique reference number

Sample conf file for Windows OS

```
xxx=c:/simulator/multiple_agent/conf1_main.xml|1
yyy=c:/simulator/multiple_agent/conf2_main.xml|2
zzz=c:/simulator/multiple_agent/conf3_main.xml|3
abc=c:/simulator/multiple_agent/conf3_main.xml|4
```
Sample conf file for Linux and Solaris OS

<table>
<thead>
<tr>
<th>AAA</th>
<th>BBB</th>
</tr>
</thead>
<tbody>
<tr>
<td>aaa=/home/simulator/multiple_agent/confi4_main.xml</td>
<td>bbb=/advent2/sim_3_1/community_agents/config5_main.xml</td>
</tr>
</tbody>
</table>

Starting the SNMP Proxy Agent from Command Line

After creating the configuration file, start the command line SNMP Proxy Agent by executing the `StartSnmpProxyAgent.bat/.sh` found inside the `<Simulator_Home>/bin/cmdline` directory, with the following options:

- `-pc <conf file name>`: The conf file which contains the mapping information. This option is mandatory.
- `-p <port number>`: The port number where the Proxy Agent must be started.
- `-ip <agent IP address>`: The IP address of the Proxy Agent.

Example

```
StartSnmpProxyAgent.bat -pc conf.txt -p 8005 -ip 200.201.202.1
```

Testing the Proxy Agent Simulation

1. Create the SNMP Agent Simulator configurations (minimum two set of configuration) with different community names. The configuration must be created in SNMP Agent Simulator or Network Simulator.
2. Create the conf file as specified in Creating the conf file. It must contain the details of the configuration file, created in Step 1.
3. Start the SNMP Proxy Agent with the required options. This will start the multiple SNMP agents with the specified configurations.
4. Query the SNMP Proxy Agent from the manager with the required community name and port number. The proxy agent will forward the request to the appropriate SNMP agent and provide the response.
TL1 Gateway Network Simulation

- Overview
- Creating the Agent Configurations
- Creating the Property File
- Starting the Gateway Network Simulation
- Testing Gateway Network Simulation

Overview

Most telecommunications and network systems, such as SONET/SDH/IDLC, etc. have many NEs connected with each other. If an input message is to be sent from a manager to a particular NE, it might have to pass through other NEs connected in the network. Hence it becomes necessary to make one or more of the NEs in a network as Gateway NEs (GNE) which will provide connection to other NEs.

A GNE will route the input message from a manager to the NE connected to it. TL1 Messages coming from the OSS will be routed by the GNE to the appropriate NEs by using the TID (Target Identifier). The GNE will also route the Autonomous Messages from the NE back to the managers connected.

AdventNet Simulation Toolkit supports the Gateway Network Simulation in TL1. This feature is supported in command line.

Creating the Agent Configurations

The Gateway Network Simulation takes the TL1 agent configuration file as input. Hence, each NE in the network must have the configuration file with .prp extension, which contains the simulated agent configurations. The Gateway NE may or may not implement a configuration file. The configuration files must be created using the AdventNet TL1 Agent Simulator.

Creating the Property File

The mapping information required to access the simulated TL1 agent based on Target ID must be provided in a text file format. The Gateway Network simulation uses this information to route the messages to the appropriate NE in the network. The format of the Text file is given below:

```
<Target IDs>=<Configuration file name with correct path>
```

where,

- **Target ID** - Specify any string as Target Identifier (TID). This string will be used by the TL1 Manager and the Gateway NE to route the TL1 messages.

- **Configuration file name** - The configuration file with .prp extension, which contains the simulated agent configurations of the NEs in the network. This file must be created using TL1 Agent Simulator or Network Simulator.

The entries in a sample property file is given below:

```
Agent1=C:/AdventNet/Simulator4/tl1agents/Agent/sample1.prp
Agent2=C:/AdventNet/Simulator4/tl1agents/Agent/sample2.prp
```
### Starting the Gateway Network Simulation

1. The Gateway Network Simulation can be started in command line by executing the `StartTL1GNE.bat/.sh` found inside the `<Simulator_Home>/bin/cmdline` directory, with the following options.

   **Mandatory Options**

   - `-m <property file>`: The multiple agent property file which contains details of the different configuration files with different TID settings
   - `-p <port number>`: The Port Number where the Gateway Network Simulation must be started.

   **Optional Options**

   These options must be specified if the Gateway NE implements a configuration file:

   - `-GTID <TID name>`: Target ID of the Gateway NE.
   - `-GCONF <conf file name>`: The Configuration file used by the Gateway NE.

   **Example**:

   1. Starting the Gateway Network Simulation without the Gateway NE configuration:

      ```
      StartTL1GNE.bat -p 8005 -m conf.txt
      ```

   2. Starting the Gateway Network Simulation with the Gateway NE configuration

      ```
      StartTL1GNE.bat -p 8005 -m conf.txt -GTID xxx -GCONF sample.prp
      ```

2. On successful startup of the Gateway Network simulation, the following message is displayed in command line:

   TL1 Server started on <port>
   Agents started successfully

3. Now, when a request is made by the TL1 manager with the required Target ID, the Gateway Network Simulator will forward the request to the appropriate NE and provide the response. If no target ID is specified, the Gateway NE will respond to the request if it is started with the `-GTID` and `-GCONF` options.

### Testing Gateway Network Simulation

1. Create the TL1 Agent Simulator configurations (minimum two set of configuration) with different Target IDs. The configuration must be created in TL1 Agent Simulator or Network Simulator.

2. Create the property file as specified in Creating the Property File. It must contain the details of the configuration file, created in Step 1.

3. Start the Gateway Network Simulator with the required options. This will start the multiple TL1 agents with the specified configurations.

4. Query the TL1 agent from the manager with their Target IDs and Port number.
5. The GNE will forward the request to the appropriate NE and provide the response. You can also test the agent to send Autonomous messages and execute scripts, if configured for the queried Command code.

6. Query the TL1 agent from the manager without the TargetID. The Gateway NE will respond to the request.

**Note:** The Target ID used to query the TL1 agent must be the Target ID specified in the property file.
Migration Guide

The Migration Guide is designed to aid users of AdventNet Simulation Toolkit 5 in migrating to the latest AdventNet Simulation Toolkit 6 release version. It provides the detailed procedural steps to guide the users during the course of migration.

Simulation Toolkit 6 comes along with a set of command line migration tools to provide a hassle free migration experience. Migration tools are provided to migrate the Simulated SNMP Agent, Simulated TL1 Agent and Simulated Network.

Choose the required option from the links provided below and migrate to the latest version:

- Migrating the Simulated SNMP Agent
- Migrating the Simulated TL1 Agent
- Migrating the Simulated Network
Migrating the Simulated SNMP Agent

The `SnmpMigrator.bat/sh` is used to migrate the SNMP Agent configuration from version 5 to the latest 6 version.

How to Migrate

1. Run the `SnmpMigrator.bat/sh` file in `<Simulator_Home>/bin/migration` directory from command line with the following options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-old &lt;old SNMP configuration file&gt;</td>
<td>The old SNMP configuration file name with absolute path.</td>
</tr>
<tr>
<td>-new &lt;new SNMP configuration folder name&gt;</td>
<td>The new SNMP configuration folder name with absolute path.</td>
</tr>
</tbody>
</table>

2. The database parameters to be specified for old and new MySQL connections are User name, Password, Server name, Port and Server path. These details must be provided in `migrationdb_params.conf` file in `<Simulator_Home>/conf` directory of the version 5 and version 6 product respectively.

3. Now, the configuration file will be migrated to the latest version 6 format in the specified folder with the specified path. You can use this configuration file in Simulation Toolkit 6 version.

Example

To migrate the configuration created with the same MySQL server settings or different MySQL server settings

In Windows:

```
SnmpMigrator.bat  -old  c:\simulator5\snmpagents\sample\sample1_main.xml   -new
c:\simulator6\snmpagents\sample
```

In Linux:

```
SnmpMigrator.sh  -old /simulator5/snmpagents/sample/sample1_main.xml -new
/simulator6/snmpagents/sample
```

Note:
1. The old SNMP configuration file name must end with `_main.xml`, for example `sample1_main.xml`
2. In the -new option, specify the folder name which must contain the new SNMP configuration.
3. The Absolute path of the configuration file directory must be provided.
Migrating the Simulated Network

The NetworkMigrator.bat/sh is used to migrate the network configuration from version 5 to the latest version 6.

How to Migrate

1. Run the NetworkMigrator.bat/sh file in <Simulator_Home>/bin/migration directory from command line with the following options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-old &lt;old network file name&gt;</td>
<td>The old network file name with absolute path.</td>
</tr>
<tr>
<td>-new &lt;new network name&gt;</td>
<td>The name of the new network with absolute path.</td>
</tr>
</tbody>
</table>

2. The database parameters to be specified for old and new MySQL connections are User name, Password, Server name, Port and Server path. These details must be provided in migrationdb_params.conf file in <Simulator_Home>/conf directory of the version 5 and version 6 product respectively.

3. Now, the network will be migrated to the latest format in the specified folder with the specified path. You can use this network in Simulation Toolkit 6 version.

Example

To migrate the configuration created in Database/XML mode with the same MySQL server settings or different MySQL server settings

**In Windows:**

```
NetworkMigrator.bat -old c:\temp\network1.net -new c:\simulator6\networks\network1
```

**In Linux/Solaris:**

```
NetworkMigrator.sh -old /home/network1.net -new /home/network1
```

**Note:**

1. The old network file name must end with .net, for example Demo.net
2. Specify the network name in the -new option.
3. The Absolute path of the network file directory must be provided.
Migrating the Simulated TL1 Agent

The TL1Migrator.bat/sh is used to migrate the TL1 Agent configuration from version 5 to the latest 6 version.

How to Migrate

1. Run the TL1Migrator.bat/sh file in <Simulator_Home>/bin/migration directory from command line with the following options:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-old &lt;old TL1 config&gt;</td>
<td>The old TL1 configuration file name with absolute path.</td>
</tr>
<tr>
<td>-new &lt;new TL1 config&gt;</td>
<td>The new TL1 configuration file name with absolute path.</td>
</tr>
</tbody>
</table>

2. If the old configuration is in Database mode, the Database Settings to get connected with the old configuration mode must be provided in migrationdb_params.conf file in <Simulator_Home>/conf directory. It contains details of the Database server name, Port number, User name, Password and the Server path of the old configuration.

3. Now, the configuration file will be migrated to the latest format in the specified filename with the specified path. You can use this configuration file in Simulation Toolkit 6 version.

Example

To migrate the configuration created in Database/XML mode with the same MySQL server settings or different MySQL server settings

In Windows:

TL1Migrator.bat -old c:\simulator5\TL1agents\sample\sample1.prp -new c:\simulator6\tl1agents\sample\sample1.prp

In Linux:

TL1Migrator.sh -old /simulator5/TL1agents/sample/sample1.prp -new /simulator6/TL1agents/sample/sample1.prp

Note:

1. The old and new TL1 configuration file name must end with .prp extension, for example sample1.prp
2. Absolute path of the configuration file directory must be provided.
TroubleShooting Guide

The Troubleshooting guide gives you tips on how to overcome certain errors that are likely to occur while using the different tools integrated with Simulation Toolkit. The errors are grouped module wise for easy reference.

- General
- SNMP Agent Simulator
- Network Simulator
TroubleShooting - General

General errors that are likely to occur while using the different tools integrated with Simulation Toolkit are classified under different categories. Please follow the links to know about them:

- Troubleshooting Tips - while starting the Toolkit
- Troubleshooting Tips - while saving the agent configuration
- Troubleshooting Tips - while starting the SNMP Agent at port 161

While starting the Toolkit

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Tool does not start and appear to be hanged</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause of the Problem</td>
<td>The database connection could not be obtained</td>
</tr>
</tbody>
</table>

How to correct it

Possible solutions:

1. Check the port number '2306' (by default, simulation toolkit will start the mysql server at 2306) is free. If the port number is already occupied, kindly edit the file `NDTStartup.prop` inside the `<SimulatorHome>/conf/settings` directory and update the port number with some other, which is free and then restart the product.

2. Check if the file `my.cnf` is present inside the directory `/etc/` and if so, rename the file and then restart the product.

3. If you are still not succeeded in starting the mysql server, then you can start the server separately using the shell file `StartMySql.sh` from within `<SimulatorHome>/bin` directory and then start the product (`NetworkDesigner.sh`).

Note: Make sure, you are starting the product as 'root' user with 'admin' previlleges.

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Out of environment space</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause of the Problem</td>
<td>This is a problem with the windows' environment variable settings</td>
</tr>
</tbody>
</table>

How to correct it

If Out of environment space error is thrown, do the following steps to rectify the error:

- Open the Command Prompt, right click on the Title bar of the prompt and select the properties menu item. A window titled MS-DOS Prompt Properties will be opened.
- Select the 'Memory' option in the menu.
- Select the value '4096' in the 'Initial environment' field
- Click on the Apply button and restart the Command Window.
- Now Start the product.
<table>
<thead>
<tr>
<th>Error Message</th>
<th>An unexpected exception has been detected in native code outside the VM.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause of the Problem</td>
<td>This is an issue of jdk1.4 due to the incompatibility of the Java’s dll with some of the hardwares.</td>
</tr>
<tr>
<td>How to correct it</td>
<td>Please use the jdk1.3 in your machine. For more details on the issue, you can refer to the following URL. <a href="http://developer.java.sun.com/developer/bugParade/bugs/4482430.html">http://developer.java.sun.com/developer/bugParade/bugs/4482430.html</a> You have to change the java home in setenv.bat file in &lt;simulator home&gt;/bin directory and the SimulationToolkit.bat in the &lt;simulator home&gt; directory.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Error Message</th>
<th>NoClassDefFoundError</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause of the Problem</td>
<td>This problem generally occurs in Windows 98 &amp; Windows ME. When the classpath variable is set from the bat file the out of environment space error will be thrown. In this case the classpath will not be set and the No Class Found Error will be thrown.</td>
</tr>
</tbody>
</table>
| How to correct it      | • Edit the SnmpAgentSimulator.bat file under the <Simulator_Home>/bin directory.  
• Remove "@echo off" present in the SnmpAgentSimulator.bat file and save the file.  
• Edit the setenv.bat file under the <Simulator_Home>/bin directory.  
• Remove "@echo off" present in the setenv.bat file and save the file.  
• Open a command prompt and move to <Simulator Home>/bin directory. Now execute the SnmpAgentSimulator.bat in this command prompt. |

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Exception while starting the SNMP Agent Simulator in Simulation Toolkit:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause of the Problem</td>
<td>This problem occurs if SNMP Agent Simulator is started as Normal user</td>
</tr>
<tr>
<td>How to correct it</td>
<td>All the tools in Simulation Toolkit must be used only as Admin user.</td>
</tr>
</tbody>
</table>
Error Message: Exception with error code 515 while starting SNMP MIB Browser in Simulation Toolkit:

Exception: java.io.FileNotFoundException: ./conf/settings/mibbrowserset.conf (Permission denied)

Cause of the Problem: This problem occurs if SNMP MIB Browser is started as Normal user.

How to correct it: All the tools in Simulation Toolkit must be used only as Admin user.

While saving the agent configuration

Error Message: "Invalid FileName! Please enter a filename without white spaces"

Cause of the Problem: This is a restriction with Simulation toolkit. The Simulator does not allow to save the Agent Configuration in a directory which has space in its path.

How to correct it: Save the agent configuration in a different directory, which does not have white spaces.

Error Message: "Error in parsing configuration file..."

Cause of the Problem: The error occurs due to the space present in the Configuration file path of the SNMP Agent Simulator.

How to correct it: Save the agent configuration in a different directory, which does not have white spaces.

While starting the agent configuration

Error Message: Unable to start the SNMP Agent at port 161

Cause of the Problem: This error will occur if the port (161/162) your are trying to start the Agent is used by another UDP service.

How to correct it:

For Windows OS:

To check the UDP Port, type the following command in the command prompt:

```
netstat -anp udp
```

If the port is occupied by any other service, then you have to stop that service and start the simulated agent.

To start the agent at port 161 in Windows OS, follow the steps given below:

1. Select Start -> Settings -> Control Panel -> Administrative Tools -> Services. You will find all the services listed in the Services dialog.

2. Select the SNMP Service. The SNMP Service Properties dialog will pop up. Here, you will find the SNMP service status as Started.
3. Select the **Stop** button, to stop the SNMP service and click **OK**.

**Note**: In the SNMP Service Properties dialog, if the Startup Type is mentioned as Automatic, you will need to perform the above steps, each time the system is started. Hence, you can make the Startup Type as Manual.

Now, you will be able to start the simulated agent at port 161.

**For Linux/Solaris OS** :

If you are using Linux/Solaris machine then check the port by using the command `netstat -anu` in the console.

If the port is free, then you can start the simulated agent in 161/162 only if you have **Root** privilege. So kindly login as root and start the agent.

If the port is occupied by any other service, then you have to stop that service and start the simulated agent. To start the agent at port 161 in Linux OS, follow the steps given below:

You have to have root user privileges to stop the **snmpd daemon**. So, login as root user.

Use the command `/etc/rc.d/init.d/snmpd stop` for stopping the snmpd in linux. Now, you will be able to start the simulated agents at port 161.
TroubleShooting - SNMP Agent Simulator

- TroubleShooting Tips - SNMPv3
- TroubleShooting Tips - Traps
- TroubleShooting Tips - MIBs

SNMPv3

Errors that are likely to occur while using SNMPv3 version:

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Cause of the Problem</th>
<th>How to correct it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unable to encode PDU</td>
<td>This error will occur if the security provider has not been added to the java.security file.(assuming that you are using JCE).</td>
<td>The security provider has to be added by manually editing the java.security file under jdk1.2/jre/lib/security. Some sample entries are: security.provider.1=sun.security.provider.Sun security.provider.2=com.sun.crypto.provider.SunJCE</td>
</tr>
<tr>
<td>Discovery Failed</td>
<td>a. You will get this error if you try to add a user with &quot;Auth,Priv&quot; or &quot;Auth, noPriv &quot; or &quot;noAuthnoPriv&quot; security level in the v3settings tab of the Mibbrowser without starting the agent. b. The Port number specified may be wrong. c. The IP Address may be wrong.</td>
<td>Check if the agent is started at the correct port number as specified in the v3 settings tab. If not, start the agent and add the new user entry.</td>
</tr>
<tr>
<td>Time sync failed for user &lt;username&gt;</td>
<td>You will get this error in the following cases: a. If you provide the wrong user name/password while adding the user in the Manager to test the v3 agent. (i.e) the user name provided does not exist in the USM user table.</td>
<td>Hence, enter the correct UserName and password as provided in the USM user table.</td>
</tr>
<tr>
<td>A authorization failed error occurred</td>
<td>You will get this error in the text area of the Mib Browser while trying to do a get/set for the v3 agent. This error occurs in the following cases: a. If the v3 user is not provided access in the vacmviewtreefamilytable for the object being queried. For example, the access provided for the user authUserSHA in the vacmviewtreefamilytable is .1.3.6.1.6.3.16. If this user tries to do a get on the v3 agent with Object ID .1.3.6.1.4.1.2162.4.1.1, the above error is displayed. b. If the context name provided in the Mib Browser is not the</td>
<td></td>
</tr>
<tr>
<td>Error Message</td>
<td>End of Mib view error</td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------</td>
<td></td>
</tr>
<tr>
<td>Cause of the Problem</td>
<td>You will get this error in the text area of the Mib Browser while trying to do a get-next for the v3 agent. This error occurs in the following cases:</td>
<td></td>
</tr>
<tr>
<td>a. If the context name provided in the Mib Browser is not the same as that configured in the vacmContextTable for that user. For example, the context name provided for the user authUserSHA in the vacmcontexttable is &quot;auth&quot; and if this user tries to access the v3 agent by providing the context name as &quot;noauth&quot; in the Mib Browser, the above error is displayed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>How to correct it</td>
<td>Find out the cause of your problem from the above listed reasons and do the necessary changes</td>
<td></td>
</tr>
</tbody>
</table>

**Traps**

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Please select valid Trap/Notification Node from the tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause of the Problem</td>
<td>The above error will be displayed, while configuring default traps defined in the MIB, using Generate Traps option. It occurs, if a node other than the Trap /Notification node is selected as the Trap node.</td>
</tr>
<tr>
<td>How to correct it</td>
<td>While specifying the Trap Name in the Generate Traps dialog, you have to select only the Trap/Notification nodes defined in the loaded MIB. For example, if RFC-1213 MIB is loaded, you have to select the Trap node as egpNeighborLoss. On selecting the trap node, you will find the Trap Name, Varbinds and Description displayed on the right-side dialog. You can then proceed to configure the Manager entries and Sent traps</td>
</tr>
</tbody>
</table>

**MIBs**

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Invalid value for node &quot;XXXXX&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cause of the Problem</td>
<td>One of the possible cause:</td>
</tr>
<tr>
<td>Simulation Toolkit does not support BITS and OPAQUE datatypes. The MIB file might have these TCs defined.</td>
<td></td>
</tr>
<tr>
<td>How to correct it</td>
<td>Change the type ‘Opaque’ and ‘BITS’ as ‘Integer32’ in the required MIBs and then reload the MIBs</td>
</tr>
</tbody>
</table>
## TroubleShooting - Network Simulator

Errors that are likely to occur while working with Network Simulator are listed below:

### 1.

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Cause of the Problem</th>
<th>How to correct it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unable to start SNMP Agent at Port : 8001 and at IpAddress : 121.111.50.2</td>
<td>The above error will occur in Network Simulator in any one of the following cases:</td>
<td>You have to configure a unique virtual IP address so that the device can be associated with the IP address, check if the port is occupied, check if the product installation directory contains white spaces.</td>
</tr>
<tr>
<td>Unable to start SNMP Agent at Port : 8001 and at IpAddress : 121.111.50.1</td>
<td>• The port may be occupied by some other application.</td>
<td></td>
</tr>
<tr>
<td>Unable to start Network Simulator !!!</td>
<td>• There is a IP clash in the network. (i.e) the virtual IP configured in the simulator already exists in the network</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Product is installed in C:\Program Files . Actually this is a limitation of our product. Installation directory name should not have spaces.</td>
<td></td>
</tr>
</tbody>
</table>

### 2.

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Cause of the Problem</th>
<th>How to correct it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem with MySQL Server startup in Linux OS</td>
<td>In linux when the file./etc/my.cnf is present there will be problem in mysql server startup.</td>
<td>The specified file should be moved or renamed.</td>
</tr>
</tbody>
</table>

### 3.

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Cause of the Problem</th>
<th>How to correct it</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database does not exist / Table does not exist</td>
<td>The above error will occur if the database is corrupted.</td>
<td>---</td>
</tr>
</tbody>
</table>

### 4.

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Cause of the Problem</th>
<th>How to correct it</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Could not get registration info. The database XXX cannot be found&quot;</td>
<td>The above error will occur, if you have got connected to a wrong MySQL. That is, you would have got connected to a MySQL which does not have the simulated data</td>
<td>Make sure you connect to the MySQL where your simulated data resides.</td>
</tr>
<tr>
<td></td>
<td>Make sure your simulated data resides there and that you are connecting only to that MySQL.</td>
<td>If you have your MySQL running in a separate machine, make sure your simulated data resides there and that you are connecting only to that MySQL.</td>
</tr>
</tbody>
</table>
5. **Error Message**: Device configuration corrupted  
**Cause of the Problem**: This could happen when you have accidentally deleted any of the system configuration files.  
**How to correct it**: You will have to delete the corrupted file and recreate the device..

6. **Error Message**: The virtual interfaces added in the server machine (where simulator is running) is not accessible from the Manager machine (where the manager application is running).  
**Cause of the Problem**: This could happen if you are running the simulated network and the manager application in different machines.  
**How to correct it**: To access the simulated network from the manager application running in a different system, execute the following command in the system where the manager application is running:

- **In Linux/Solaris**:  
  The following command has to be executed from the console as superuser.  
  **Command Syntax**:  
  `/sbin/route add -net <network address> netmask 255.255.255.0 gw <manager machines IP Address>`  
  For example:  
  `/sbin/route add -net 190.191.192.0 netmask 255.255.255.0 gw 192.168.1.41`

- **In Windows**:  
  The following command has to be executed from the console as superuser.  
  **Command Syntax**:  
  `route add <network address> mask 255.255.255.0 <manager machines IP Address>`  
  For example:  
  `route add 190.191.192.0 mask 255.255.255.0 192.168.1.41`

7. **Error Message**: "Too many open files in system"  
**Cause of the Problem**: The error log displays "Too many open files in system" error if the number of file descriptors set is not sufficient to start the required number of devices in the network. The command "ulimit -n" will display the file descriptors set on the system.  
**How to correct it**: Set the required number of file descriptors as specified in the topic "Simulating Large Networks -> Changing the system settings in Linux", restart the system for the settings to take affect and then start the network.
Error Message : Unable to download file through FTP/TFTP

| Cause of the Problem | 1. Check if the ipaddress and port of the host (where TFTP Server is running), is correct.  
|                      | 2. Check if the sourcefile to be downloaded exists in the host.  
|                      | 3. Check if any Firewall has been configured who has blocked the FTP request. |
| How to correct it    | Specify the correct values, if firewall is configured, disable the firewall and then send the request. |
Using Test & Editor Tools

AdventNet Simulation Toolkit is packaged along with Test and Editor tools to provide the user with the complete Simulation experience.

Test Tools serve as Manager Applications to test the simulated SNMP and TL1 Agents.

Editor Tools are used to edit some of the input files of the simulator.

- **Test Tools**
  - **MIB Browser**: To test the simulated SNMPv1/v2/v3 agent.
  - **TL1 Craft Interface**: To test the simulated TL1 agent.

- **Editor Tool**
  - **TL1 Message Builder**: To create and edit TCS files used as input in TL1 Agent Simulator.
  - **SNMPv3 Config Tool**: To configure user entries in USM and VACM tables for the SNMPv3 agent.
MIB Browser

- Overview
- Getting Started
- Configuration
- MIB Operations
- SNMP Operations
- Trap Handling
- Table Handling
- Graphs
- Debugging and Decoding
- Error Messages
- FAQs

AdventNet MibBrowser is a complete SNMP MIB Browser that enables loading, browsing, and searching MIBs, walking the MIB tree, and performing all other SNMP-related functions. MibBrowser also enables viewing and operating the data available through an SNMP agent in a managed device.

This section gives an overview of AdventNet MibBrowser and highlights its features. It also gives an illustrative overview on the user interface and an extensive focus on the various SNMP operations that can be done using MibBrowser.
Overview

AdventNet MibBrowser is a complete SNMP MibBrowser that enables loading, browsing, and searching MIBs, walking the MIB tree, and performing all other SNMP-related functions. MibBrowser also enables viewing and operating the data available through an SNMP agent in a managed device.

Features of AdventNet MibBrowser

1. Enables saving of MibBrowser settings.
2. Provides the capability to load and view MIB modules in a MIB tree.
3. Helps in traversing the MIB tree to view the definitions of each node for a particular object defined in the MIB.
4. Enables to perform the basic SNMP operations, such as GET, GETNEXT, GETBULK, and SET.
5. Supports multi-varbind requests.
6. Enables real-time plotting of SNMP data in a graph. Line graph and bar graph are the two types of graphs that are currently supported.
7. Provides a user-friendly view of SNMP table data. The table data can be viewed in a separate window called SNMP Table Panel.
8. Enables to view the incoming traps using Trap Viewer. It also allows parsing of traps.
9. Enables loading of MIBs at startup.

How Does AdventNet MibBrowser Look Like?

The image above depicts the primary window of AdventNet MibBrowser. It consists of a menu bar, toolbar, left frame, and a right frame.
The menu bar displays a list of commands to perform various operations.

The toolbar displays buttons with images that act as shortcuts to the menu options.

The left frame holds the MIB tree. A MIB tree is a structure that displays all the loaded MiBs. The MIB tree enables to traverse the tree, view the loaded MiBs, and the definition of each node.

The right frame contains text fields to specify the basic parameters, such as host, community, and so on. It also contains a text area to display the results.

There are three ways in which the primary window of the MibBrowser can be displayed. They are the Result Display, MIB Description, and Multi-Varbind. To change the display, click View --> Display and select the desired view. By default, the MIB Description display is visible in the MibBrowser.
Getting Started

The MibBrowser can be started in any of the following ways:

1. Running the Script or Batch file

Run the batch or shell script `SnmpMibBrowser` from the `<Simulator_Home>/bin` directory. This script executes the `setenv` file to set the appropriate classpath variables.

<table>
<thead>
<tr>
<th>Windows</th>
<th>Linux</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>SnmpMibBrowser</code></td>
<td><code>sh SnmpMibBrowser.sh</code></td>
</tr>
</tbody>
</table>

2. Using the Launcher

The Mib Browser tool can be started in the Launcher in one of the following ways.
- Select Browsers
- Right-click on the Mib Browser icon and select Run (or)
- Select Run from the Options menu after selecting the Mib Browser.

**Note:** For Windows Installation, you can also start the tool from the Start Menu -> Programs -> AdventNet Simulation Toolkit 6 -> Mib Browser

**Windows:**

You can invoke MibBrowser using Simulation Toolki Launcher. The Launcher can be invoked by selecting Start -> Programs -> AdventNet Simulation toolkit 4 -> AdventNet Simulation Toolkit.

Alternatively, you can invoke Launcher by running the `SimulationToolkit.bat` file in the `<Simulator_Home>` folder. In the Launcher window, double-click the MibBrowser icon to run MibBrowser.

**Unix:** Run the `SnmpMibBrowser.sh` file in the `<Simulator_Home>/bin` directory to directly invoke MibBrowser.

**Note:** While invoking MibBrowser from Launcher, the security provider jars should be placed before AdventNetSnmp.jar in the CLASSPATH for v3 security level. To do this, appropriate changes need to be done in the MibBrowser section of the `launcher_conf.txt` file.
Configuration

- Setting Common Parameters
- Setting MIB Parameters

AdventNet MibBrowser can be used to view and operate on the data available through an SNMP agent. MibBrowser can be configured in tune with performing the SNMP operations.

To configure the MibBrowser, click the MibBrowser Settings button or select Edit-->Settings from the menu. You can also use the shortcut key combination Alt + S.

The MibBrowser Settings dialog box is displayed. It has the following two tabs.

- General
- Mib Settings
Setting Common Parameters

The parameters, such as host, port, and community, can be set in the MibBrowser's main window. Applications use the host name or the IP address of the device to communicate with the agent of the device in a particular port number. This remote port number is the UDP port 161. By default, all the SNMP request messages are received in this port.

SNMP mandates that the SNMP agents should accept request messages only if the community string in the message matches its community name. Therefore, the management application should always communicate with the agents along with the associated community name. The default SNMP community names are "public" for read-only (GET) operations and "private" for read-write (SET) operations. The management applications should have provision to include the community names in their request messages. Community strings are used to authenticate SNMP PDUs. Since SNMP packets are usually sent using UDP packets, there is no connection established as in the case of TCP/IP packets. Therefore, when a UDP packet is sent to the agent, the agent validates the packet. It accepts and sends a response if the community string of the PDU is equal to that set on the agent, or else drops the packet. The agent does not change the community name after communicating. Applications typically communicate with the SNMP agents by specifying the community name of the agent.

The default community string is "public" and the default Write Community string is null. When Write Community is null, community itself is used for SET operations. Therefore, applications should explicitly set the Write Community, before they can use it for SET operations.

The other parameters are set in the General tab of the MibBrowser Settings frame. The general settings include the basic protocol options related to SNMP, the display options, and a section for encoding field. The image depicted below shows the General tab of the MibBrowser Settings dialog box.
The various protocol-related options to SNMP are listed in the table below.

<table>
<thead>
<tr>
<th>Options</th>
<th>Default Values</th>
<th>Other Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNMP Version</td>
<td>v1</td>
<td>v2c or v3</td>
</tr>
<tr>
<td>Timeout</td>
<td>5 sec</td>
<td>any user-defined value</td>
</tr>
<tr>
<td>Retries</td>
<td>0</td>
<td>any user-defined value</td>
</tr>
<tr>
<td>Encoding</td>
<td>ISO8859_1</td>
<td>any encoding scheme that supports text format</td>
</tr>
<tr>
<td>Max Repetitions</td>
<td>50</td>
<td>any user-defined value</td>
</tr>
<tr>
<td>Non Repeaters</td>
<td>0</td>
<td>any user-defined value</td>
</tr>
</tbody>
</table>

**Note:**
- Timeout is the time interval that an application waits for a response message from an agent before timing out. Retries is the number of times a request is sent when a timeout occurs. If the retry value is 0, the request is re-transmitted on timeout.
- The Max-repetitions and Non-repeaters options are enabled only when the SNMP version is set to either v2c or v3. This is because, the GETBULK operation is available only in v2c and v3. A GETBULK request is performed by giving an OID along with two other parameters, Max Repetitions value and Non Repeaters value.
- Encoding, in general, means, modifying information into the required transmission format. Computers around the world store information using a variety of encoding schemes. AdventNet MibBrowser support the ISO8859_1, which means ISO 8859_1, Latin alphabet No.1. There are various other encoding schemes that support various text formats. You can use the encoding scheme that best suits your requirement while performing SNMP operations. To view the encoding scheme that is supported by Java Development Kit, see: [http://java.sun.com/products/jdk/1.1/docs/guide/intl/encoding.doc.html](http://java.sun.com/products/jdk/1.1/docs/guide/intl/encoding.doc.html).

The Validate Broadcast Address check box enables you to check the validity of the broadcast address provided. You need to provide the Netmask address to validate the broadcast address. A Netmask is a string of 0's and 1's that hides the network part of the IP address and allows only the host ID to remain.

In the v3 Options section, Context Name and the ContextID are to be provided as additional parameters for an SNMPv3 request. An SNMP context name is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context. An SNMP entity potentially has access to many contexts. In other words, if a management information has been defined under certain context by an SNMPv3 entity, any management application can access that information by giving that context name. The ContextID uniquely identifies an SNMP entity that may recognize an instance of a context with a particular context name within an administrative domain.

The next section is the v3Settings section. The following are the security-related parameters for accessing the SNMPv3 agents. You can add, modify, and delete users by clicking the Add, Modify, Delete buttons.
## Options

<table>
<thead>
<tr>
<th>Options</th>
<th>Default Values</th>
<th>Other Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>User name</td>
<td>null</td>
<td>any user-defined value</td>
</tr>
<tr>
<td>Security level</td>
<td>noAuth noPriv</td>
<td>Auth noPriv and Auth Priv</td>
</tr>
<tr>
<td>Authentication Protocol</td>
<td>MD5 (if authentication is chosen in security level)</td>
<td>SHA</td>
</tr>
<tr>
<td>Privacy Protocol</td>
<td>CBC-DES (if privacy is chosen in security level)</td>
<td>not available</td>
</tr>
<tr>
<td>Authentication password</td>
<td>any user defined value</td>
<td>-</td>
</tr>
<tr>
<td>Privacy password</td>
<td>any user defined value</td>
<td>-</td>
</tr>
<tr>
<td>Target host</td>
<td>localhost</td>
<td>any host with SNMPV3 agent or proxy agent</td>
</tr>
<tr>
<td>Target port</td>
<td>161</td>
<td>any user-defined port</td>
</tr>
</tbody>
</table>

If the security level is "NoAuthNoPriv", no additional parameters are required. If the security level is "AuthNoPriv", the parameters AuthProtocol and AuthPassword are set. If the user security level is "AuthPriv", the privacy password needs to be set in addition to the other parameters.

The v3Settings section has an option for storing the v3 table entries. The v3 table entries can be stored in:

- A serialized file
- A database

### Storing Table Entries in a Serialized File

To enable serialization of v3 table entries, select the Save v3 Settings to File option. If this option is selected, the user information is stored in the serialized files namely UserEntry.ser and EngineEntry.ser. When MibBrowser is invoked the next time, the serialized files are de-serialized and the v3 table is updated. The advantage of storing table entries in serialized files is that the operation is faster.

### Storing Table Entries in a Database

To store v3 table entries in a database, select the Save v3 Settings to Database option. To use this, the database connection has to be established. Clicking the "Database Settings" button displays the Database Parameters dialog box. The image of the dialog box that appears is given below.
Enter the necessary database parameters in this dialog box, and click the OK button. If the database connection is established successfully, all the user information entered is saved in the database. When the MibBrowser is invoked the next time, v3 details are restored and the v3 table is updated.

The advantages of storing v3 table entries in a database are:

- Scalability - Any number of entries can be maintained in the database.
- Accessibility - All the authenticated users of the database can access the entries.

The last section is the field entry section in which the corresponding fields in the v3 table are displayed for data entry. The various buttons available in the field entry section are Add, Modify, and Delete.

To add an entry, enter the required parameters in the respective fields and click the Add Entry button. Based on the parameters and the security level, Discovery and Time Synchronization are done and USM Table is updated and listed in the v3 table.

To modify an entry, select the entry in the v3 table, modify the required fields, and click the Modify button. Time Synchronization is done and the USM table is updated and listed in the v3 table.

**Note:** The NoAuthNoPriv entry cannot be modified. Only the password fields in the AuthNoPriv entry and AuthPriv entry can be modified.

To delete an entry in the v3 table, select the entry in the v3 table and click the Delete button. The entry is removed from the USM table.

**Note:**
- The settings are saved only on exiting the MibBrowser application and not every time the settings are modified.
- The Restore Defaults button is used to reset the default parameters.
Setting MIB Parameters

The Mib Settings tab is used to load MIBs in MibBrowser. The image below displays the Mib Settings tab.

The first section gives the MIB loading options. Refer Loading MIBs and Unloading MIBs for more details. The next section displays the various parsing levels. Refer Parsing MIBs for more information.
MIB Operations

- Loading MIBs
- Unloading MIBs
- Parsing MIBs

The basic MIB operations are loading, unloading, and parsing MIB files. A MIB file can be loaded directly, or from compiled files or from a database.
Loading MIBs

Follow the steps given below to load a MIB file.

1. Click the Load MIB Module button or select File→Load MIB from the menu. Alternatively, you can use the shortcut combination Ctrl + O. This displays the Load a Mib File dialog box as shown below.

![Load MIB File Dialog Box](image)

2. In the Load a Mib File dialog box, select the MIB file from the mibs folder.
There are other options for loading the MIB file. Click the Recent tab to load the recently loaded MIB files. The Recent section of the Load a MIB File dialog box is shown below.

Select the check box next to the MIB that is to be loaded and click Open to load the MIB file.

**Note:** The Load All MIBs option in the File menu loads all the recently loaded MIBs.

Let us have a detailed overview on the various options available in the MIB Settings section of the Load a MIB File dialog box.

**MIB Loading Options**

The various options available for loading MIBs are:

- Load MIBs directly
- Load MIBs from compiled files
- Load MIBs from a database
The following image depicts the Mib Settings tab in the Load MIB dialog box.

### Load MIBs Directly

The MIB file is usually read and parsed into MIB modules and displayed in the MIB tree. In this case, when you load a MIB file, it is parsed and then loaded. This is time consuming because parsing is done every time a MIB file is loaded.

### Load MIBs from Compiled Files

The next option is loading of the MIB files as compiled files. The main advantage here is that the loading time is reduced. This ultimately leads to the improvement in performance. To store the MIB information in a formatted structure, the following two new file types are made available.

- **cmi** - This file type contains MIB information, such as MibNode, MibModule, naming hierarchy, etc.
- **cds** - This file type contains the description and reference of the MIB nodes.

When the option Load MIBs from Compiled File is selected, the MibBrowser loads the MIB from the cmi and cds files. If these files are not present, MibBrowser parses the MIB file, writes the output in the cmi and cds files, and loads the MIB file. For example, if you load the RFC1213-MIB, the MIB file is parsed and stored in the compiled MIB files as RFC1213-MIB.cmi and RFC1213-MIB.cds provided RFC1213-MIB is the module name of the RFC1213-MIB file.
When this MIB file is loaded again, the MIB is loaded from the cmi and cds files and no parsing is done. The advantage of using this option is, we need not parse the MIB each time we load, thus optimizing the load time and improving the performance. While loading the compiled MIBs, you need to load only the cmi file. The cmi file has a reference to the cds files. Therefore, the cds file need not be loaded directly.

**Note:** Any changes made to the MIB file after it has been loaded as a compiled MIB file are not reflected when it is loaded again. You have to remove the existing cmi and cds files and load the MIB again to view the latest changes. To overcome this, select the option "Overwrite existing Compiled MIB Files". If this option is set to true, the cmi and cds files are created each time the MIB is loaded. However, enabling this option is recommended only if you have changed the contents of the MIB file. Otherwise, this serves as a redundant option and increases the load time of the MIBs.

To convert the normal MIB files to the cmi and cds format, the MibToCMI utility can be used. This class is available in the `<classes/com/adventnet/snmp/utils>` directory. To use the utility, set the CLASSPATH to the classes directory and give the following command.

```java com.adventnet.snmp.utils.MibToCMI directory name or file, another directory or file```

The conversion can be done for individual files or for an entire directory of MIB files. If the utility is used across directories, the existing cmi and cds files should first be deleted from the directory.

**Loading MIBs from a Database**

The third option is loading of the MIB files from a database. The MIB files can be stored in any RDBMS such as MySQL or Oracle. Applications can load these MIB files directly from the database. This feature is particularly useful when the MIB files to be loaded are more in number.

The MibBrowser uses JDBC (Java Database Connectivity) for the database support. Applications should use a valid JDBC driver of the respective databases to enable the database support.

Selecting the option Load MIBs from Database enables the text fields in the JDBCParams section. On initializing the necessary database parameters in this section, the database support can be provided for loading MIBs. The required JDBCParams are as follows.

- **Drivername** - name of the database driver.
- **URL** - URL pointing to the database filename.
- **UserName** - name of the user.
- **Password** - password for the user.

After selecting this loading option, select the required MIB file to be loaded from the Open tab of the Load a MIB File dialog box. If the selected MIB file is already present in the database, the MIB file is read, parsed, and loaded from the database. If the MIB file does not exist in the database, MibBrowser parses the MIB file, writes the output to the database, and then loads the MIB file.

**Note:** The recently loaded MIBs can be loaded automatically at startup by selecting the option, 'Load recently loaded Mibs at startup'.

**Instructions for Using MySql Database for Loading MIBs**

Having tested the loading of MIB files from the MySQL database, we understand that the following database parameters have to be configured in the application.

- **driver name** - org.gjt.mm.mysql.Driver
- **url** - JDBC: mysql://<machine name>/<database name>
• username - a valid user name
• password - password for the above user

The jar file mysql_comp.jar has to be included in the CLASSPATH. If the jar is not in the CLASSPATH, the following exception is thrown.

```
java.lang.ClassNotFoundException:org.gjt.mm.mysql.Driver
```

**Instructions for Using Oracle Database for Loading MIBs**

The loading of MIB files from the Oracle database has also been tested. The following database parameters are to be configured in the application.

- driver name - org.jdbc.driver.OracleDriver
- url - jdbc:oracle:thin:@<machine name>:1521:<database name>
- username - a valid user name
- password - password for the user

The oracle driver has to be included in the CLASSPATH. If the jar is not present in the CLASSPATH, the following exception is thrown.

```
java.lang.ClassNotFoundException:oracle.jdbc.driver.OracleDriver
```

For other databases, use the equivalent parameters.

A few MIB files are provided in the mibs directory, i.e., RFC1213-MIB, RFC1271-RMON, RFC1155-SMI, RMON2-MIB, TOKEN-RING-RMON-MIB, and RFC1315-FRAME. It may be convenient to copy your MIB module files that are to be loaded into the "mibs" directory.

**Note:**
Apparently, MIBs are always parsed before loading when the following operations are performed.

- Loading MIBs directly
- Loading MIB from Compiled File for the first time
- Selecting the "Overwrite existing compiled MIB files" option
- Loading MIB from database for the first time
Unloading MIBs

The next basic MIB operation is unloading. To unload the MIB, select the node of the MIB tree and click the Unload MIB Module button or select File-->UnLoad MIB from the menu. Alternatively, you can press the Delete key.

Performing any of the above prompts you for a confirmation. The following image depicts the confirmation dialog box.

![Confirm Unload dialog box]

Selecting 'Yes' unloads the MIB module. If no module is selected in the MIB tree, all the loaded MIB modules are unloaded.

The Load All MIBs option in the menu bar would load all the previously loaded MIBs and Unload All MIBs would unload all the loaded MIBs in the MIB tree.
Parsing MIBs

MibBrowser enables you to parse the given MIB file and check for the macro constructs. It allows different levels of parsing and the parsing is done as per the standard definition of the macros.

The parsing levels can be set in the MibBrowser Settings dialog box. The following table describes the different levels of parsing that can be set and their corresponding checks.

<table>
<thead>
<tr>
<th>S. No</th>
<th>Level of Parsing</th>
<th>Checks</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Lenient</td>
<td>No Default checks</td>
<td>This level accepts all types of MIB files. For example, it allows both SMIv1 and v2.</td>
</tr>
<tr>
<td>2</td>
<td>Normal</td>
<td></td>
<td>This level is the default level conforming to the obsolete standards, such as RFC 1902, RFC 1903, etc. Most MIBs follow the obsolete standard.</td>
</tr>
<tr>
<td>3</td>
<td>Serious</td>
<td>Most checks throw exceptions on first misbehavior.</td>
<td>This level strictly follows the current standard. It accepts the constructs with interoperability and implementation problems.</td>
</tr>
<tr>
<td>4</td>
<td>Critical</td>
<td>All possible checks throw exceptions on first misbehavior</td>
<td>This level completely follows the SMIv1 and v2 standards. However, it does not accept the backward compatibility constructs, constructs with interoperability and implementation problems, etc.</td>
</tr>
</tbody>
</table>

Applications, while loading MIB files, perform the following operations.

- Parsing and validating the syntax of the MIB module
- Constructing the MIB module into the tree structure

While performing the parsing and validation of the MIB files, if the MIB modules fail to conform to the SMI standards, the loading will not be not done. However, the application requirements might mandate the loading of the non-standard files. On the other hand, some applications might require a stricter check on the compliance to the standards.

The parsing and validating the syntax of the MIB file can be made configurable to suit the application requirements. MibBrowser handles this by providing parsing levels which facilitate to select the level of parsing required by the applications.

In addition to the above four parsing levels, MibBrowser supports another level, which is user-defined. In case of user-defined level, you can define your own parsing level with the required checks at run time.

**Note:** It is recommended to use the higher parsing level (SERIOUS, CRITICAL) for validating the MIB file and not for loading the MIB file in the application. It affects the performance of the application while loading the MIB files, because it takes considerable amount of time and resources, such as memory, CPU usage, etc.
User-Defined Parsing Level

In addition to the four parsing levels, you also have another level, which is user-defined to define your own parsing levels at run time. To add a user-defined parsing level, select the User-Defined option and click the Add button in the User-Defined Levels section. This displays the Customized Level dialog box. Double-click the Levels folder to display a list of all the checks as shown below.

By default, all the checks are included. Provide a name for the level in the Level Name text field. To add or remove checks from the level, select or deselect the checks and click OK. Note that if you select (or deselect) a parent check, all its child checks are also selected (or deselected). Click OK to add this level to the user-defined level list.

The level of the parser has to be set in the MibParser before loading a MIB. This level, once set, is used for subsequent MIBs loaded. If the level needs to be modified for the next set of MIBs loaded, it has to be set again in the MIB Parser. In the Mib Settings tab of the MibBrowser Settings dialog box, select the required Parsing level, and click Apply.

The MIB file can contain one or more MIB modules. MibBrowser loads all the dependency files to resolve the MIB module. If the dependency file is not present, the IMPORTS failed error is thrown.

The parsing level can be set for the dependency file by selecting the Import File option and choosing Parsing Levels.
SNMP Operations

- SNMP GET
- SNMP GETNEXT
- SNMP GETBULK
- SNMP SET

MibBrowser allows the user to perform the typical SNMP operations. The operations are categorized as:

- Retrieving Data - GET, GETNEXT, GETBULK
- Altering Variables - SET
- Receiving Unsolicited Messages - Traps

To perform any basic operation as categorized above, it is essential to specify the Object ID, the instance, host name, and the community string. Changes can also be made to the parameters in the MibBrowser Settings dialog box. To get a vivid outlook on the MibBrowser settings, refer the Configuration topic.

Specifying the Index

To specify an object to an SNMP agent, both the Object ID (which defines the type of object) and the instance (the specific object of the given type) need to be provided. From the MIB you can get the Object ID, to which an instance needs to be added to completely identify the object of interest.

For non-tabular or scalar objects, the instance is 0. For example sysDescr is a scalar object under the system group in RFC1213-MIB and it should be specified as sysDescr.0 in Object ID field of MibBrowser. In MibBrowser, this need not be specified if the MIB is loaded, i.e. RFC1213-MIB. MibBrowser adds it to the selected node while performing the GET operation.

For tabular objects, the instance is defined in the MIB as index, and it is a sequence of one or more objects. For example, ifInOctets of ifTable defined in RFC1213-mib, the index is ifIndex (INTEGER) and may be specified as ifInOctets.1 provided that there exists a row with index 1 in the querying agent.

Another example tcpConnState of tcpConnTable under tcp group of RFC1213-MIB. The indexes of the tcpConnTable are tcpConnLocalAddress (IpAddress), tcpConnLocalPort (INTEGER), tcpConnRemAddress (IpAddress), tcpConnRemPort (INTEGER) and may be specified as tcpConnState.179.74.15.126.1192.225.226.126.197.80 provided that there exists a row with Index 179.74.15.126.1192.225.226.126.197.80 in the querying agent where

- 179.74.15.126 represents the value of the first index tcpConnLocalAddress (IpAddress),
- 1192 represents the value of the second index tcpConnLocalPort (INTEGER),
- 225.226.126.197 represents the value of the third index tcpConnRemAddress (IpAddress)
- 80 represents the value of the fourth index tcpConnRemPort (INTEGER).

To get the information about the MIB Node in the MIB Tree, click the Description icon or choose View -->Description menu item or use a shortcut of Alt + R. This gives a list of all the vital characteristics of the node such as the Node, OID, Path, MibModule, Syntax, Status, etc.
SNMP GET

The GET operation is performed to get one or more values from the managed objects. Follow the steps to perform this operation.

1. Load the MIB file. The loading of MIBs and the options available has been dealt with in detail in the Loading and Unloading MIBs section.
2. Select the desired node in the MIB tree.
3. Click the "Get SNMP Variable" icon on the toolbar or select Operations--->Get from the menu bar. Alternatively, you can use the shortcut key combination Ctrl + G.

This operation gets all objects under the selected MIB object, or the specific object if the MIB node and instance are specified.

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>• If the selected node in the MIB tree has child nodes or columnar nodes, GET operation is performed as an SNMP walk. i.e. the values for all the nodes under that sub-tree are retrieved.</td>
</tr>
<tr>
<td>• In case, if the MIB is not loaded in the MibBrowser, the exact numbered OID with full instance (for example 1.3.0 for sysUpTime) should be specified and the GET operation should be performed.</td>
</tr>
</tbody>
</table>

Performing a Multi-Varbind Request

To perform the Multi-Varbind request, follow the steps given below.

1. Make the Multi-Varbind display visible in the MibBrowser. This can be made visible by selecting View-->Display-->Multi-Varbind from the menu. You can also use the General Settings tab in the MibBrowser Settings panel.
2. Select the leaf node and append the instance by clicking the Add button. It will add the OID given in the Object Identifier field and the value given in the SetValue field both separated with a colon to the list. If value is not given in the SetValue field, NULL value is appended. Similarly, you can add as many number of OIDs and values as required.

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make sure you select the Multi-varcheck box before doing an SNMP operation for multiple varbind SNMP request. Otherwise, it does a request for the OID in the Object Identifier field.</td>
</tr>
</tbody>
</table>

3. To delete the varbind(s) from the list, select the varbind(s) from the list and click the Delete button.
4. To edit the varbind, select a varbind and click the Edit button. It shows an OID and the value of the varbind in the text fields to edit. Edit it and click the OK button to modify the OID and value or click the Cancel button to restore the old values.

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>When the request is sent under Multi-varbind mode, it goes as a single PDU irrespective of the number of OIDs added.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Note:</th>
</tr>
</thead>
<tbody>
<tr>
<td>If you want to communicate with a v3 agent, you have to select the v3 option in the MibBrowser Settings dialog box and also make sure that the v3 parameters are set. Then perform the above steps for each of the operations</td>
</tr>
</tbody>
</table>

Finding MIB Nodes

You can find a specific node in the MIB tree using the Find dialog box. This is invoked by selecting Edit-->Find Node from the menu. Use the Find in All option in the Find dialog box to find a node in all the loaded MIB modules.
SNMP GETNEXT

This operation is similar to the SNMP GET operation, but retrieves the value of the next OID in the tree. This operation is used for traversing the MIB tree. To perform this operation follow the steps 1 and 2 as in SNMP GET. Then proceed with the following step 3.

3. Click the "Get Next SNMP Variable" button on the toolbar or select Operations-->GetNext from the menu bar. Alternatively, you can use the shortcut key combination Ctrl + N.

This operation will get the next object after the specified object, or the specific object instance, if a MIB node is specified. The instance may or may not be specified. You can also perform a multi-varbind GETNEXT request.
SNMP GETBULK

To retrieve voluminous data from a large table, the GETBULK operation is performed. A GETBULK request is performed by giving an OID along with two other parameters, namely a Max-Repetitions value and a Non-Repeaters value. The GETBULK operation is performed only on SNMPv2c and SNMPv3.

To perform this operation, follow the steps 1 and 2 as in SNMP GET and continue with the following step 3 through step 5.

3. Configure the MibBrowser to either SNMPv2c or SNMPv3 as desired. This can be done using either the Edit-->Settings option in the menu bar or the MibBrowser Settings icon on the toolbar. You can also use the shortcut key combination Alt + S.

4. Under the same MibBrowser Settings panel, the Max-Repetitions field and the Non-Repeaters field are enabled. The Max-Repetitions value specifies the number of lexicographic successors to be returned for the remaining variables in the variable-bindings list. The default value in this field is 50. The Non-Repeaters value specifies the number of variables in the variable-bindings list for which a single lexicographic successor is to be returned. The default value in this field is 0. Specify the values for these two parameters if you need to and then proceed with the operation.

5. Click the "Get Bulk SNMP data" button or icon on the toolbar or choose Operation-->GetBulk from the menu bar. Alternatively, you can use the shortcut key combination Ctrl + B.

This will get a sequence of Next Objects immediately after the specified object. The number of Object instances returned is equal to the Max-Repetitions field.
SNMP SET

Most network devices have a default value maintained by the agent. Sometimes applications modify the data for one or more MIB variables, thereby using the SNMP SET operation. The following steps will guide you to understand how you can perform the SET operation.

1. Load the MIB file. The loading of MIBs and the options available have been dealt with in detail in the MIB Operations section.
2. Select the desired node in the MIB Tree to which value has to be set. The SET operation can be performed only on the node that has read-write access.
3. Set the value in the Set Value field.
4. Click the "Set SNMP Variable" icon on the toolbar or select the Operations-->Set from the menu bar. You can also use the shortcut key combination Ctrl + W.

Performing a Multi-Variable SET Operation

To perform the multiple variable SNMP SET request, follow the steps given below.

1. Make the Multi-Varbind display visible in the MibBrowser. This can be made visible by selecting View-->Display-->Multi-Varbind from the menu. You can also use the General Settings tab in the MibBrowser Settings panel.
2. Select the nodes in the MIB tree for which the SET operation is to be performed and specify the value to be set for it in the SetValue field.
3. Append the instance by clicking the Add button. This adds the OID given in the Object Identifier field and the value given in the SetValue field, both separated by a colon, to the list. Similarly, you can add as many number of OIDs and values as required.

   Note: Ensure that the OID and the value that are given in the text fields are correct before you add them to the list.

4. Enable the "Multi-Var" check box present at the bottom and click the Set SNMP Variable button on the toolbar. The SET operation is performed on all the nodes added in the multi-varbind list.
5. To delete the varbind(s) from the list, select the varbind(s) from the list and click the Delete button.
6. To edit the varbind, select a varbind and click the Edit button. It will show an OID and the value of the varbind in the text fields to edit. Edit it and click the OK button to modify the OID and value or click the Cancel button to restore the old values.

   Note:
   - When the request is sent under Multi-varbind mode, it goes as a single PDU and not as broken PDUs (irrespective of number of OIDs added).
   - If you want to communicate with a v3 agent, you have to choose Version3, from the list box in the MibBrowser Settings dialog box and also make sure that the v3 parameters are set. Then perform the above steps for each of the operation.

To perform a SET operation for Octet String Type in hex format enter the bytes in hex format with each bytes separated by a colon and the entire string within single quotes. For example to give 0xff0a3212 enter 'ff:0a:32:12' in the Set Value field.

   Note: You can find a specific node in the MIB tree using the Find dialog box. This is invoked by selecting Edit-->Find Node from the menu. If you want to find a node in all the loaded MIB modules, select Find in All option.
The various values that have to be specified in the Set Value field with respect to the SYNTAX of the Object are given below.

<table>
<thead>
<tr>
<th>Base Datatypes/TCs</th>
<th>How to Set the Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value</td>
<td>Value in HexaDecimal</td>
</tr>
<tr>
<td>INTEGER/Integer32</td>
<td>100</td>
<td>'64'h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unsigned32</td>
<td>100</td>
<td>'64'h</td>
</tr>
<tr>
<td>OCTET STRING</td>
<td>adventnet</td>
<td></td>
</tr>
<tr>
<td>OBJECT IDENTIFIER</td>
<td>1.3.6.1.2.1.1.1.0 or 1.1.0</td>
<td></td>
</tr>
<tr>
<td>NULL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Counter/Counter32</td>
<td>100</td>
<td>'64'h</td>
</tr>
<tr>
<td>Counter64</td>
<td>100</td>
<td>'64'h</td>
</tr>
<tr>
<td>Gauge/Gauge32</td>
<td>100</td>
<td>'64'h</td>
</tr>
<tr>
<td>BITS STRING</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>13 5 / one three five / one 3 five</td>
<td>'54'h/'54'H or 54h/54H '50'h/'50'H or 50h/50H '5'h/50'H</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TIMETICKS</td>
<td>100</td>
<td>'64'h</td>
</tr>
<tr>
<td>IpAddress</td>
<td>192.168.1.220 / hostName</td>
<td></td>
</tr>
<tr>
<td>NetworkAddress</td>
<td>192.168.1.220 / hostName</td>
<td></td>
</tr>
<tr>
<td>OPAQUE</td>
<td></td>
<td>'64'h</td>
</tr>
<tr>
<td>TAddress</td>
<td>192.168.1.120 / 161</td>
<td></td>
</tr>
<tr>
<td>MacAddress</td>
<td></td>
<td>f1:f2:f3:f4:f5:f6</td>
</tr>
</tbody>
</table>
Trap Handling

- Viewing Traps
- Parsing Traps
- Creating Parser Files
- Editing Parser Files

The agent, when faced by some problem or error in the transmission of message, responds to the manager by sending unsolicited messages called traps.

Traps are unsolicited messages sent from an SNMP agent to one or more SNMP Management applications. It is an asynchronous notification sent by the agent to the manager about some event occurrence in the device.

In order to receive and view the incoming traps to the specified port, MibBrowser has Trap Viewer.

Trap Viewer can be used for viewing and parsing traps. You can also create parser files using the Trap Parser editor and edit them.
Viewing Traps

Trap Viewer is a graphical tool to view the traps received from one or more SNMP agents. Trap Viewer can listen to one or more port at a time and the traps can be sent from any host. Trap Viewer can handle inform requests as well. Inform requests can be sent from a manager to another manager. Trap Viewer can receive and display inform messages. By default, the inform requests are sent to the port 162 of the manager station.

To open Trap Viewer, click the Trap Viewer icon on the toolbar or select View->Trap Viewer from the menu. You can also use the shortcut key combination Alt + P.

The image depicted below is a screen shot of Trap Viewer.

<table>
<thead>
<tr>
<th>Class</th>
<th>Type</th>
<th>Source</th>
<th>Date</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear</td>
<td>v1 Trap</td>
<td>127.0.0.1</td>
<td>Tue Aug 27 11:37:3</td>
<td></td>
</tr>
<tr>
<td>Clear</td>
<td>v1 Trap</td>
<td>127.0.0.1</td>
<td>Tue Aug 27 11:38:2</td>
<td></td>
</tr>
</tbody>
</table>

Follow these steps to know more about the available options.
1. The value in the Port text field is set to 162 by default. Enter the desired port in this field.
2. The default value of the Community field is public. Set the community of the incoming traps as desired.
3. Click the Add button to add the Port and Community list to the Trap list. This is visible in the TrapList list box.
4. The Port and Community list can be deleted by clicking the Del button.
5. Click the Load button to load the trap parser file.
6. In order to receive the traps now, click the Start button. Trap Viewer begins to receive traps from the specified port and community.
7. The traps when received are listed in the **trap table**. The trap table has the following five columns.
   - The **Class** column defines the severity of the trap.
   - The **Type** column defines the type of the trap or the inform request.
   - The **Source** column represents the IP address of the source from where the traps were sent.
   - The **Date** column shows the date and time when the trap was received.
   - The **Message** column has the VarBind list of the trap, if any.

8. The status of the trap is displayed in the status pane at the bottom of the dialog box. Moreover, the Trap count and the inform count is displayed in the status pane.

9. To log the received traps, select the **Enable Logging** check box. All the incoming traps are logged to a file. The default name of the log file is **trap.log**.

10. The details of the traps can be viewed by clicking the **Trap Details** button. You can also right-click the trap in the trap table and select 'View Trap Details'. The image below is the screenshot of the Trap details.

![Trap Details](image)

**Note:** Note that v3 traps are received only if the corresponding entry is present in the v3 table. Refer **Setting Common Parameters** for more information on adding v3 entries.
The various details available in the Trap Details frame are listed in the table below.

<table>
<thead>
<tr>
<th>Trap Details</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeStamp</td>
<td>This field shows the value stored in the MIB-II sysUpTime variable converted into hours, minutes and seconds. It is a 32-bit unsigned value indicating the number of centiseconds that have elapsed since the (re)start of the SNMP agent and the sending of the trap.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>This field shows the OID of the management enterprise that defines the trap message. The value is represented as an OBJECT IDENTIFIER value and has a variable length.</td>
</tr>
<tr>
<td>Generic Type</td>
<td>This field shows the value based on the type of trap. The value is categorized and numbered 0 to 6. They are 0-coldStart, 1-warmStart, 2-linkDown, 3-linkUp, 4-authenticationFailure, and 5-egpNeighborLoss. The trap type value 6 is identified as an enterprise-specific value.</td>
</tr>
<tr>
<td>Specific Type</td>
<td>This field can have values from 0 to 2147483647. The specific trap type indicates the specific trap as defined in an enterprise-specific MIB. If the Generic Type value is 6, this field shows a value greater than 0. If the Generic Type value is a value other than 6, then the field shows a value 0.</td>
</tr>
<tr>
<td>Message</td>
<td>This is a text field. By default, this field always contains the Varbinds in the Trap PDU. This can be replaced with the text.</td>
</tr>
<tr>
<td>Severity</td>
<td>This field shows the severity or the intensity of the trap. It could be 0-All, 1-Critical, 2-Major, 3-Minor, 4-warning, 5-Clear, and 6-Info.</td>
</tr>
<tr>
<td>Entity</td>
<td>This field contains the source IP address from which the Trap was sent.</td>
</tr>
<tr>
<td>RemotePort</td>
<td>This field reveals the port from which the Trap was sent by the originator.</td>
</tr>
<tr>
<td>Community</td>
<td>This field contains the Community string.</td>
</tr>
<tr>
<td>Node</td>
<td>Source</td>
</tr>
<tr>
<td>TimeReceived</td>
<td>This field contains the date and time when the trap was received.</td>
</tr>
<tr>
<td>HelpURL</td>
<td>The URL shown here gives more details of the received trap. By default, the URL is <code>&lt;generic-type value&gt; - &lt;specific-type value&gt;.html</code></td>
</tr>
</tbody>
</table>

9. You can stop listening to the port by clicking the **Stop** button.
10. When you need to delete a trap, select the trap and click the **Delete Trap** button. You can also right-click the trap in the trap table and select 'Delete the Selected Rows'.
11. Another option in Trap Viewer is the ParserEditor. Trap Viewer can filter the incoming traps according to certain criterion called the Parser Criteria. The configuration of the criterion is made possible by using the ParserEditor. A detailed overview has been given in the Parsing Traps section.
Parsing Traps

The Trap Parser editor is a tool to create a Trap Parser file. Trap Viewer parses the file created using the Trap Parser editor, with the help of the TrapParserBean, to match each incoming traps with certain criterion. Trap Parsers are required to translate or parse traps into understandable information because traps typically contain cryptic information which is not easily understandable to the users.

The Trap Parser editor:
- configures Trap Parser files.
- parses the traps into an understandable format.

Invoking the Trap Parser

To invoke the Trap Parser editor, click the Parser Editor button in Trap Viewer UI. You can create a new Parser file from an existing Parser file or a MIB file. You also have an option to create a parser file from scratch using the Create Custom Parser File option. The image below depicts the first screen of the Trap Parser editor.
Load Trap Parsers and MIB Files

The next screen of the Trap Parser editor has the options for loading Trap Parser files. There is also an option for loading MIB files that contain trap definitions. The image below shows the loading and saving options of the Trap Parser editor.
Match Criteria

Each incoming trap has to essentially possess the Parser match criteria to be shown in the trap table. In the General tab of the next screen, you can enter any number of Parser match criteria into a single parser file with a different parser name. Trap Viewer looks for a match criterion sequentially. Once a criterion is matched for a trap, further checking of match criteria is skipped, and an event is fired to display the corresponding trap entry in the trap table.

While listening to traps, only one parser file can be loaded by Trap Viewer. A parser file can have any number of match criteria. Trap Viewer checks all the criteria in a Trap Parser file sequentially until one criterion matches. The image below depicts the General tab of the Trap Parser editor.

The various match criteria are tabulated below. These are mandatory parameters.

<table>
<thead>
<tr>
<th>Match Criteria</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generic Type</td>
<td>Each trap has a generic type number. The Generic types are 0-coldStart, 1-warmStart, 2-linkDown, 3-linkUp, 4-authenticationFailure, 5-egpNeighborLoss, and 6-enterpriseSpecific. The number is to be specified for the Trap Parser. The trap is parsed only when the criteria matches. The only exception is that, a trap is parsed even when the generic field is left blank or a negative value is entered in the field.</td>
</tr>
<tr>
<td>Specific Type</td>
<td>This field can have an integer value from 0 to 2147483647. If this field is to be matched, the Generic Type must always be enterprise specific.</td>
</tr>
<tr>
<td>Enterprise OID</td>
<td>This is the SNMP enterprise identifier in the trap, which is used for unique identification of traps for a particular application. If you specify the OID in this field, the parser is applied only if the trap enterprise field begins with the enterprise field that you have specified. The only exception is that even when the enterprise field is left blank, the trap is allowed to be parsed.</td>
</tr>
</tbody>
</table>

If you had loaded a MIB file or a parser file in the previous screen, the corresponding trap parsers are listed in the Parser List section. You can add, modify, or delete these parsers to create your parser file.
OID and Value

In the OID tab, you can specify the multiple values in the OID and Value text fields to match. The Trap PDU should match all the OID:Value pairs to conform to this criterion. This criterion extends the matching criterion. The following image depicts the OID tab of the Trap Parser editor.
Agent and Port

In the Agent tab, the trap is sent by an agent, specified in the Agent and Port text fields. If the value of Port is 0, the source can send the trap from any port. The incoming trap must match any of the criteria in the Agent and Port text fields. The following image shows the Agent tab of the Trap Parser editor.
OutPut Event Parameters

For each matching criterion, a trap name is given. The fields in the event details are configured in the Output Event section. Once the trap is matched by the criteria, the trap is added to the Trap Table. The Output Event parameters are shown as the Trap Details, which gives more specific information on the trap.

By default, some of the fields of the Output Event parameters are filled by a variable called parser variables usually starting with a "$". These variables substitute specific characteristics of the parser in the Trap Details. The following image shows the various Output Event parameters.

![Graphical representation of trap parser]

Each parser variable and its corresponding characteristics are tabulated below.

<table>
<thead>
<tr>
<th>Variables</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Community</td>
<td>This token is replaced by the community string of the received trap.</td>
</tr>
<tr>
<td>$Source</td>
<td>This token is replaced by the source name/address of the received trap.</td>
</tr>
<tr>
<td>$Enterprise</td>
<td>This token is replaced by the enterprise ID of the received trap.</td>
</tr>
<tr>
<td>$Agent</td>
<td>This token is replaced by the agent address of the received trap.</td>
</tr>
<tr>
<td>$SpecificType</td>
<td>This token is replaced by the specific type of the received trap.</td>
</tr>
<tr>
<td>$GenericType</td>
<td>This token is replaced by the generic type of the received trap.</td>
</tr>
<tr>
<td>$Uptime</td>
<td>This token is replaced by the up time value in the received trap.</td>
</tr>
<tr>
<td>$*</td>
<td>This token is replaced by all the variable bindings of the received trap,</td>
</tr>
<tr>
<td></td>
<td>including the OID and values of each variable binding.</td>
</tr>
<tr>
<td>$#</td>
<td>This token is replaced by all the SNMP variable values in the variable bindings of the received trap.</td>
</tr>
<tr>
<td>$N</td>
<td>This token is replaced by the (N-1)th SNMP variable value in the variable bindings of the received trap.</td>
</tr>
<tr>
<td>@*</td>
<td>This token is replaced by all the OID values in the variable bindings of the received trap.</td>
</tr>
<tr>
<td>@N</td>
<td>This token is replaced by the (N-1)th OID value in the variable bindings of the received trap.</td>
</tr>
</tbody>
</table>
Creating Parser Files

A parser file can be created using the Trap Parser editor. Let us now create a Parser file using the Trap Parser editor. Refer the Parsing Traps section for invoking Trap Parser and understanding the various parameters available under the Match Criteria.

1. Click the ParserEditor button in the Trap Viewer to invoke Trap Parser.

2. In General tab, enter the values for the Match Criteria parameters as follows.
   - Generic Type: linkUp
   - Specific Type: 0 (because it is a Generic Trap)
   - Enterprise OID: .1.3.6.1.2.11

3. Select the OID tab and enter the following value.
   - OID/Value (VarBinds): .1.3.6.2.1.1.5.0 xyz

4. Select the Output Event tab. The values depicted for the parameters by default are listed below.
   - Severity: "-"
   - HelpURL: "$GenericType-$SpecificType.html"
   - Message: "$"

   Change the values to:
   - Severity: Major
   - Message: This is a Message

5. Click the Severity Color button to change the color in which the trap is visible in the trap table.
6. Enter the trap name in the Trap Parser Name field in the Match Criteria section and click the Add button to add the Trap Parser to the Parser list.

7. Add more Match Criteria for the incoming traps and click the Save button to save the current parser criteria. Save the file with the name "test". The Parser filename is displayed in the Parser File text field.

8. Close the Trap Parser editor. When you need to modify the added Trap Parser, make the changes in the Match Criteria and OutPut Event Parameters, and click the Modify button. It is essential to save the parser file again to effect the changes.

The created Parser file can also be edited manually. Refer Editing Parser File for more details.

**Filtering Traps**

Let us have an overview on filtering of the traps. Follow the steps to perform trap filtering.

1. Click the Load button in the Trap Parser editor to load the Trap Parser file.

2. Now, send a trap from the applications directory of the AdventNet SNMP API package with the following command.

   ```java
   java sendtrap localhost .1.3.6.1.2.11 localhost 3 0 1000
   ```

On receiving the trap, the trap is parsed and checked if it matches the criteria specified in the 'test' parser file. This trap is received because it conforms to the match criteria.

**Adding Trap Definition from Any MIB file to a Parser File**

To add a trap definition from any MIB file to a parser file, select the MIB file by clicking the Load button. The parameters are displayed automatically with their respective definitions in the Match Criteria and Output Event parameters sections.
Editing Parser Files

The Parser file created by Trap Parser editor can also be edited using any text editor. When the Parser file is opened in any text editor, the contents of the Parser file would look like this.

```plaintext
customMatchDefn=
color=-39424
severity=2
helpDefn=$GenericType-$SpecificType.html
GT=1
ST=0
agentAddressDefn=
enterprise = .1.3.6.1.2.1.1
```

The content of the parser file reveals the various vital parameters of Trap Parser. The table below lists the various parameters and the ways to configure them.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>customMatchDefn</td>
<td>This has the OID:Value pair as the match criteria. Example: .1.3.6.1.2.1.1.0:abcd;.1.3.6.1.2.1.1.2.0:acme &quot;&quot; is required before &quot;:&quot; for every OID value pair and each pair is separated by a &quot;,&quot;</td>
</tr>
<tr>
<td>color</td>
<td>This depicts the color of the trap entry in the trap table. The value of the color ranges from -1 to -16777216</td>
</tr>
<tr>
<td>enterprise</td>
<td>This is the enterprise OID of the trap.</td>
</tr>
<tr>
<td>severity</td>
<td>This field shows the severity or the intensity of the trap. This severity determines how a fault is affected by this event. The type of this field is an integer ranging from 0 to 6. 0 denotes All, 1 denotes Critical, 2 denotes Major, 3 denotes Minor, 4 denotes Warning, 5 denotes Clear, 6 denotes Info. By default, 5 is assigned.</td>
</tr>
<tr>
<td>helpDefn</td>
<td>The URL of the document associated with the trap is specified here. By default, it is &lt;generic-type value&gt;-&lt;specific-type value&gt;.html</td>
</tr>
<tr>
<td>GT</td>
<td>This depicts the Generic Type trap and the value ranges from 0 to 6. 0 denotes coldStart, 1 denotes warmStart, 2 denotes linkDown, 3 denotes linkUp, 4 denotes authenticationFailure, 5 denotes egpNeighborLoss, 6 denotes enterprise-specific.</td>
</tr>
<tr>
<td>ST</td>
<td>This is the Specific Type trap. The value can range from 0 to 2147483647.</td>
</tr>
<tr>
<td>agentAddressDefn</td>
<td>This will contain the agent:port pair, as the match criteria. Example: 127.0.0.1:0;192.168.1.2:0 &quot;&quot; is required before &quot;:&quot; for every agent port pair and each pair is separated by a &quot;,&quot;</td>
</tr>
<tr>
<td>textDefn</td>
<td>This is the message text for this event in the ListTraps and logs.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the parser is given here.</td>
</tr>
</tbody>
</table>
Table Handling

Handling

- Retrieving Table Data
- Adding Rows
- Deleting Rows

AdventNet MibBrowser enables you to view the SNMP Table data in a separate window called the SNMP Table Panel. The table has a very user-friendly profile. The SNMP Table panel has various options using which you can add and delete rows, view graphs and use index editor.
Retrieving Table Data

The data in the table can be retrieved with ease. The following steps gives you an insight into how to open the Data Table panel and work on it using the various options.

1. Specify the proper agent host name or IP address in the Host field of MibBrowser.
2. Load the MIB file in MibBrowser. To know more on Loading the MIB file, refer the Loading and Unloading MIBs section.
3. Specify a valid OID or select the OID by traversing through the Mib Tree. The OID should be a valid table OID.
4. Click the View SNMP data table button on the toolbar or choose View-->Snmp Table from the menu. You can also use the shortcut key combination Alt+T.
5. This would invoke the SNMP Table of the specified table OID. The figure below depicts the SNMP Table panel.

Click the Start button at the bottom of the SNMP Table panel. The retrieval of data begins and the columnar objects are obtained and displayed in the table.

7. Click the StartPolling button to start the polling of the table. The polling interval is based on the Polling Interval value set using the Settings option.
8. Click the StopPolling button to stop the polling.
9. When you don’t use the polling option, click the Refresh button to refresh the table.
10. The SNMP Table panel might retrieve more data so that the frame depicting the columnar objects would run to pages. When you need to navigate through the pages (rows), use the Next and Prev button.
11. The Page option at the bottom of the panel is used to specify how the table retrieval needs to be done. If the option is origin, the table is retrieved from the origin. If the option is index, the user can set an index value from which the table can be retrieved.
12. The host name as specified in MibBrowser will be displayed in the Host field. The page number and the number of rows in that page are displayed next to it.

13. It is possible to configure the SNMP Table panel. Click the 'Settings' button. This displays a dialog box with various options to configure the Table panel. The image below depicts the SNMP Table Settings panel.

![SNMP Table Settings](image)

The various options available are listed below.

- **Polling Interval** - This specifies the time interval between each retrieval of data. The default value is 5 secs.
- **Page Size (Rows)** - The number of rows to be retrieved is set here. The default value is 50.
- **No Of Column View** - This specifies the number of columns to be displayed in the SNMP Table panel. The default is 5.
- **Split Columns** - When the size of the PDU exceeds the limit, the agent sends the error message "Too Big PDU Error". This field serves as an option for splitting the PDU. By default, the PDU is split into half. If the number of varbinds is set by the user, then multiple request with PDU containing number of varbinds set is sent.
- **No of Columns/Request** - This field is enabled on selecting the Split Columns check box. You can specify the number for splitting of columns.
- **Port** - This field specifies the port to which the request is made.
- **Table** - The table OID is specified here.
- **Max-Repetitions** - This value specifies the number of lexicographic successors to be returned for the remaining variables in the variable-bindings list. The default value is 5. This is enabled only when the Retrieval Mode option is set to GetBulk.
- **SnmpVersion** - This gives the option to switch to any of the three versions of SNMP- v1, v2c, and v3.
- **Graph Type** - This gives the option to switch between Line Graph and Bar Graph.
- **Retrieval Mode** - By default, GETNEXT is enabled. The GETBULK option is enabled only on selecting v2c or v3.

14. If you need to view the graph, click the Graph button. The section on Graphs gives you more details on the MibBrowser Graph component.

15. You can also **add** and **delete** rows in the table and a detailed explanation has been given in the next two sections.

16. In a table, if one of the index columns is an external index, i.e., the index value is shared by some other table, then the table is called an Augmented table. Augmented table comes into picture when there is a one-to-one dependency between rows of two tables. This situation might arise when a particular MIB imports another MIB and shares a single table. For example, ifXTable defined in IF-MIB is an augmented table, which has an external index ifIndex augmented from ifTable. Clicking the 'Augmented Table' button shows the columns of the table which augments the index from the original table.

17. Click the Index Editor button to edit the index and view the rows from corresponding index.

18. To make changes to a particular column values, right-click the columns. The various options available are:
   - **view column node details** - This option gives the MibNode Information of the selected column.
   - **edit the header name for selected column** - This invokes a dialog box in which you can change the header for the selected column. The image below is a screen shot of the dialog box.

   ![Header Name Change Editor](image)

   - **view graph for selected cell(s)** - This option invokes the graph. It is possible to view the graphical representation for more than one cell by multiple selection of desired cells.
   - **add a new row to the table** - This option is the same as the Add option available in the Table panel. A detailed overview on adding a row is given in the Adding a row section.
   - **delete the selected rows from** - This option is the same as the Delete option available in the Table panel. A detailed overview on deleting a row is given in the Deleting a row section.
   - **view the not-accessible index** - A request cannot be sent to an index that is not accessible. In the table, the not-accessible index is not visible. Selecting this option would enable you to view the not-accessible index values.
Adding Rows

To add a new row to an SNMP table from the manager, the table should be an SMIv1 table with entryStatus defined or an SMIv2 table with rowStatus defined.

SMIv1 Tables with entryStatus Column

The entryStatus column is used to manage the creation and deletion of conceptual rows in SMIv1 tables. This represents the status of a table entry. The status column can have the following:

- 'valid(1)' - indicates that the row exists and is available for use.
- 'createRequest(2)' - supplied by the manager wishing to create a row.
- 'underCreation(3)' - indicates that the row is created.
- 'invalid(4)' - supplied by the manager wishing to invalidate the corresponding entry.

If a manager wishes to add a row, the status column should be set to createRequest(2). Immediately after the creation, the agent sets this object to underCreation(3). The entry remains in the underCreation(3) state until it is configured. Then its value is set to valid(1). If the status remains underCreation(3) for an abnormally long period, the agent sets the status to invalid(4).

SMIv2 Tables with rowStatus Column

In SMIv2 tables, the rowStatus column is used to manage the creation and deletion of conceptual rows. This column has six defined values as follows.

- active(1) - indicates that the conceptual row with all columns is available for use by the managed device.
- notInService(2) - indicates that the conceptual row exists in the agent, but is unavailable for use by the managed device.
- notReady(3) - indicates that the conceptual row exists in the agent, one or more required columns in the row are not instantiated.
- createAndGo(4) - supplied by a manager wishing to create a new instance of a conceptual row and make it available for use.
- createAndWait(5) - supplied by a manager wishing to create a new instance of a conceptual row but not making it available for use.
- destroy(6) - supplied by a manager wishing to delete all the instances associated with an existing conceptual row.

An existing conceptual row can be in any one of the three states, ‘notReady’, ‘notInService’, or ‘active’. If the manager wishes to add a row in a single shot with values for all the columns, the status column should be given as ‘createAndGo(4)’. After the creation of a row, its value is set to active(1). If a row has to be created with values for only some columns, the status column should be ‘createAndWait(5)’. Also, this row with partially filled columns has the status ‘notReady(3)’. The entry remains in this state until the values for all the columns are set. After all the values are set, the agent changes this value to active(1).

For SMIv2 tables, a new row can be added in three ways.

- CreateAndWait
- CreateAndGo using Multiple-Variable Set
- CreateAndGo using SNMP table UI
CreateAndWait

In this method, you should SET the RowStatus with value "CreateAndWait" and SET each column one by one. Finally, SET the RowStatus to "Active".

Each SET method can be performed either using MibBrowser or through command line application.

CreateAndGo Using Multiple-Variable Set

Instead of using multiple SET methods to create a new row, you can use Multiple-variable SET method to create a new row in a single SET method. For this, all the columns with their value should be given and RowStatus should be CreateAndGo.

Multiple-variable SET can be done using the command line application.

CreateAndGo Using SNMP Table

A new row for a table can also be created using SNMP table User Interface. The following image displays the SNMP Table dialog box. Right-click any of the columns to display a context-sensitive menu.

Select "add a new row to the table" to invoke the following dialog box. The user has to fill in all the information and click the OK button.
Deleting Rows

To delete a row from an SNMP table from the manager, the table should be an SMIv1 table with entryStatus defined or an SMIv2 table with RowStatus defined.

Using the SNMP table User Interface, you can delete a row from the table. Refer the following image for more information.
Graphs

- Line Graph
- Bar Graph

One of the vital features of the AdventNet MibBrowser is the Graphs. The graphs depict the real-time plotting of the SNMP data. Currently, two types of graphs are supported - line graph and bar graph. The SNMP data to be polled should be of integer or unsigned integer data type. Typically, the values that are plotted will be of type Counter, Gauge, or Timeticks.
Line Graph

The Line graph depicts the real-time plotting of the SNMP data. Follow the steps below to invoke a line graph.

1. Specify the proper agent host name or IP address in the Host field of MibBrowser.
2. Load the MIB file in MibBrowser. To know more on loading MIBs and other features associated with it, refer to the Loading MIBs section.
3. Specify a valid variable. The variable must be an integer or unsigned integer (Counter, Gauge, Timeticks). This variable can be entered directly in the variable field or it can be chosen by browsing through the MIB tree.
4. Click the 'View real-time graph' button in the toolbar or select View-->Line Graph from the menu. You can also use the shortcut key combination Alt+L. By default, the line graph is invoked.
5. The updated Line graph shows the results of periodically polling the agent for the specified OID.

The image below depicts a Line Graph invoked from MibBrowser.
When you move the cursor over the graph, a box containing the current time and values is displayed. The various options available in the Line graph are tabulated below.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polling Interval</td>
<td>This specifies the polling interval time. The default value is 5 secs. You can change the interval time as desired.</td>
</tr>
<tr>
<td>Average over Interval</td>
<td>By default, the graph shows the actual values of a variable for different hosts. In other words, the values of the specified OID are plotted for different hosts for the given polling interval. Selecting this option, the average of the values at a given polling interval are taken for plotting the graph.</td>
</tr>
<tr>
<td>X-axis Scale</td>
<td>This specifies the X-axis scale. The minimum value is 300 secs and this is the default value. Changing this would alter the X-axis scale of the table. This option is enabled only on clicking 'Show Polled values' option.</td>
</tr>
<tr>
<td>Show Absolute Time</td>
<td>By default, the time is depicted in the graphs only as seconds. Selecting this option would give you the time in hours:secs.</td>
</tr>
<tr>
<td>Max Poll Duration</td>
<td>This option is used to view all the polled values in a particular time period. It is enabled only on clicking 'Show polled values' option. The default value is 3600 secs.</td>
</tr>
<tr>
<td>Show Polled Values</td>
<td>This option is used to display all the polled values in a particular period of time. By default, this option is disabled. Only on selecting this option, the Max Poll Duration option is enabled.</td>
</tr>
<tr>
<td>Log FileName</td>
<td>The filename for the log file can be set here. By default, the log filename is graph.txt. If &quot;Log Polled Values&quot; is selected, all the polled values are logged in this file.</td>
</tr>
<tr>
<td>Log Polled Values</td>
<td>Selecting this option would log the polled values. This would enable the option Log FileName.</td>
</tr>
<tr>
<td>Show Absolute Counters</td>
<td>By default, the graph plots only the difference between the two Counter values. On selecting this, the plotting of the absolute value is performed.</td>
</tr>
<tr>
<td>Disable Error Dialog</td>
<td>Selecting this option would stop displaying the error messages that pop up when a request times out.</td>
</tr>
<tr>
<td>Restart</td>
<td>The Restart button is used to restart the polling.</td>
</tr>
<tr>
<td>Stop</td>
<td>The Stop button is used to stop the polling.</td>
</tr>
<tr>
<td>Close</td>
<td>The Close button is used to close the graph window.</td>
</tr>
</tbody>
</table>

Another way of invoking a graph is through the Table options. For a detailed overview on Table Handling and invoking a graph from the table window, refer to the Table Handling section.
MibBrowser can plot multiple graphs showing values for different variables from different hosts. The image below shows the plotting of multiple graphs and with the option 'Show Absolute Counters' enabled.
Bar Graph

The Bar graph depicts the real-time plotting of the SNMP data. Follow the steps below to invoke a bar graph from the MibBrowser, which is similar to invoking a Line graph.

1. Specify the proper agent host name or IP address in the host field of the MibBrowser.
2. Load the MIB file in MibBrowser. To know more on loading MIBs and other features associated with it, refer the MIB Operations section.
3. Specify a valid variable. The variable must be an integer or unsigned integer (Counter, Gauge, TimeTicks). This variable can be entered directly in the variable field or it can be chosen by browsing through the MIB tree.
4. Select the View-->Bar Graph from the menu bar. You can also use the shortcut key combination Alt + B.
5. The updated Bar graph shows the results of periodically polling the agent for the specified OID.

Note: The Bar graph does not have an option for plotting multiple variables in the graph. Therefore, in case of a columnar OID, you need to append the instance of the index in order to enable the plotting of Bar graph. For example, to plot a bar graph for the values of the first row of the columnar OID ifOperStatus (ifTable), you need to first select the node ifOperStatus. Then, in the Object ID text field, append ".1" with the OID and select View-->Bar Graph from the menu. This plots the value of the first row of the column ifOperStatus.

The image below depicts a Bar graph invoked from MibBrowser.
The various options available in the Bar Graph are tabulated below.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polling Interval</td>
<td>This specifies the polling interval time. The default value is 5 secs. You can change the interval time as desired.</td>
</tr>
<tr>
<td>Average over Interval</td>
<td>By default, the graph shows the actual values of a variable for different hosts. In other words, the values of the specified OID are plotted for different hosts for the given polling interval. Selecting this option, the average of the values at a given polling interval is taken for plotting the graph.</td>
</tr>
<tr>
<td>Range</td>
<td>Altering the range would change the X-axis scale of the table. The default value is 300 secs. The other options are 600 secs and 1000 secs.</td>
</tr>
<tr>
<td>Show Absolute Time</td>
<td>By default, the time is depicted in the graphs only as seconds. Selecting this option would give you the time in hours:mins:secs.</td>
</tr>
<tr>
<td>Restart</td>
<td>The Restart button is used to restart the polling.</td>
</tr>
<tr>
<td>Stop</td>
<td>The Stop button is used to stop the polling.</td>
</tr>
<tr>
<td>Close</td>
<td>The Close button is used to close the Bar graph window.</td>
</tr>
</tbody>
</table>
Debugging and Decoding

The Mib Browser application provides facility to view the debug output of the SNMP operations. The Debug window is used to show the PDU that is sent from the manager and the response PDU that is got from the agent.

Invoking the Debug Window

1. Click the Debug icon in the toolbar or select View-->Debug from the menu bar. You can also use the shortcut key combination Alt + D. This would invoke the Debug window.

2. As long as this Debug window is opened, debugging is turned on and the debugging output is displayed. When this window is closed, debugging is turned off.

The image below depicts a Debug window.

3. The three icons in the debug window provide the following function.
   - Save MibBrowser Debug Results - Saves the debug information to a file.
   - Print MibBrowser Debug Results - Prints the debug information to a file.
   - Snmp Decoder - Switches over to the Decoder window.

The Clear button clears the debug information and the Close button closes the Debug window.
Invoking the Decoder Window

1. To switch over from the Debug window to the Decoder window, click the Decoder icon. The Decoder icon toggles to a Debug button.

2. The three icons in the Debug window provide the following functions.
   - Save MibBrowser Decoder Results - Saves the debug information to a file.
   - Print MibBrowser Decoder Results - Prints the debug information to a file.
   - Debug - Switches over to the Debug window.

   **Note:** The Save and Print options are available only when the MibBrowser is invoked as an application and not as an applet because of security restrictions.

Performing the Decoding Operation

The Snmp Decoder is used to decode the SNMP debug messages. The figure below depicts the SNMP Decoder.

To decode the debug information, select the PDU in the debug window and click the Snmp Decoder icon. This will switch over to the decoder window. Now, the selected debug information is available in the "Hex PDU" text area. Click the Decode button to decode the information. The decoded message is displayed in the bottom panel of the Decoder window.
If the debug message was stored in a file, the decoding can be done by loading the file. This can be done by clicking the Browse button in the Debug frame. You can also enter the URL in the File URL text field and press the 'Enter' key. However, saving and loading of debug information files is done only in applications and not in applets.

**Warning:** You can select the entire PDU debug message displayed in the Debug window with all the strings and click the Decoder button to display the selected information in the Hex PDU text area. You can also save the debug message in the Debug window and load it in the Decoder window. The decoder will decode all the PDU dumps leaving the informative strings. The limitation in this is that the two continuous PDUs should have a string delimiter as a new line in between them. A sample PDU is depicted below.

```
30 06 04 06 70 75 62 6c 69 63 02 01 04 02 01
00 02 01 00 30 0e 30 0c 06 08 06 01 02 01 01 05 00 05 00
Packet from: 192.168.1.215 : 161
DATA:
30 02 01 00 04 06 70 75 62 6c 69 63 a2 21 02 01 04 02 01
00 02 01 00 30 16 30 08 2b 06 01 02 01 01 05 00 04 08
4b 41 4e 4e 41 4e 4b 41
```

The highlighted strings will be the delimiter between the two continuous PDU dumps. In case there is no string delimiter as above, only the first PDU is decoded. You can also save the decoded information using the Save button. The Save button in the Debug view saves the debug information.
## Error Messages

The following is the list of MibBrowser related Error Messages and the different scenarios at which they will be displayed:

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Error Message</th>
<th>When it is displayed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No ObjectID Specified.</td>
<td>If OID is not specified before making a request.</td>
</tr>
<tr>
<td>2</td>
<td>No Host Specified.</td>
<td>If HostName is not specified before making a query.</td>
</tr>
<tr>
<td>3</td>
<td>Host Name Should Be Entered.</td>
<td>If SnmpTable is started without specifying the host name.</td>
</tr>
<tr>
<td>4</td>
<td>Invalid OID Format</td>
<td>If the OID specified is not a valid one Before making any query Invalid OID Format exception is thrown. It should be either Integer type or String type. It will be thrown during instantiating Line/Bar Graph, SnmpTable and for operations, such as GET, GETNEXT, GETBULK, SET, etc.</td>
</tr>
<tr>
<td>5</td>
<td>Invalid Table OID</td>
<td>If SnmpTable is started, after specifying an invalid OID in the OID Textfield.</td>
</tr>
<tr>
<td>6</td>
<td>Table OID should be specified</td>
<td>SnmpTable started without specifying any Table OID</td>
</tr>
<tr>
<td>7</td>
<td>Error Loading MIB: xyz &lt;br&gt;java.io.FileNotFoundException: Could not Open stream for home/../AdventNet/SNMPv3/mibs/xyz.parser</td>
<td>If invalid file is loaded in the LoadMibDialog TextField for loading MIBSFile. But in the description TextField of MibBrowser the message thrown is, “Loading Mibs: xyz”</td>
</tr>
</tbody>
</table>
| 8     | RequestFailed: Error: Request TimedOut To LocalHost                           | The error message is thrown when doing the operations such as GET, GETNEXT, and SET and for creating SnmpTable, LineGraph or BarGraph window.  
- If the agent does not implement the OID that is queried.  
- If the agent queried is not present in the network.  
- If the port no. set is not valid for the agent queried.  
- If the Community&Write Community specified are not right.  
- If the OID queried doesn't contain any data.  
- If for v3 agents Security parameters are not set. |
| 9     | RequestFailed: Get Response PDU received from 192.168.1.001 <br>Error Indication In Response: There is no such variable name in this mib. ErrIndex: | The error message is thrown when doing set operation  
- If, the OID is not instrumented by the agent for which it is setting value.  
- If the agent queried is not present in the network.  
- If the WriteCommunity specified is not right one. |
<table>
<thead>
<tr>
<th>S.No.</th>
<th>Error Message</th>
<th>When it is displayed</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.</td>
<td>If the Syntax of the value you are setting is not as that required by the OID.</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>The same Exception is also thrown when plotting Graph for a leafNode is not instrumented by the agent for which it is setting value. And also when querying for data in an SnmpTable</td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td>If data is not available.</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>RequestFailed: Get Response PDU received from 192.168.1.001 Error Indication In Response: A not writable error occurred. ErrIndex: 1</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Error Sending SET Request: com.adventnet.snmp.beans.DataException: Error: OID not a leaf node. The Error Message is thrown during set operation, if the OID selected is not a leaf node.</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Error sending set request: com.adventnet.snmp.beans: DataException: Error: Creating Variable If setting value for a columnnode of a table does not have rowstatus.</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>LineGraphBean Error: can not plot string value Root..... The exception is thrown when plotting Graph for a leafNode, 1. If value of OID selected is not of Integer/TimeTicks/ type.</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>LineGraphBean Error: can not plot these values .1.3.6.1.2.1..................: value ..................................: value While plotting Line/Bar Graph, if the OID/LeafNode selected has syntax PhysAddress, NetworkAddress, IP Address, OBJECT IDENTIFIER,</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Error: com.adventnet.snmp.beans:DataException: InvalidTable OID:(oid chosen) If SnmpTable is started, with OID which is not a Table OID.</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>ErrorSendingPDU: Failed to Authenticate the Security Parameters, for user SnmpEngineEntry not found for address( host name ) port(portNo.) The exception is thrown while creating SnmpTable, 1. If the host name specified, is of different version than v3 which is set in settings table.</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>LineGraphBeanError: cannot plot string value xyz. If the OID/LeafNode chosen for plotting Graph is of String Type.</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Discovery failed for address (host name ) port (portno.) If wrong port number is set in the MibSettings panel.</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Time Sync Failed for user (user name) If wrong username/user password/priv password/ TargetHost/SecurityLevel is set in the MibSettings panel</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Error in Getting DataBase Connection:Please check the jdbc parameters: com.adventnet.snmp.beans.MibException: java.lang.ClassNotFoundException: If DriverName/URL/User Name/Password has been set wrong when loading MIBs from database</td>
<td></td>
</tr>
<tr>
<td>S.No.</td>
<td>Error Message</td>
<td>When it is displayed</td>
</tr>
<tr>
<td>-------</td>
<td>-------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>21</td>
<td>Error in Getting DataBase Connection:Please check the jdbc parameters: java.lang.ClassNotFoundException: (DriverName set)</td>
<td>If the mysql.jar class is not present in the classes directory.</td>
</tr>
<tr>
<td>22</td>
<td>Please enter the UserName</td>
<td>If the Username is not set for the v3 User.</td>
</tr>
<tr>
<td>23</td>
<td>Sent request to hostName:port no. Request Failed :SNMPv3 Error in Response. usmStatsUnknownUserNames(.1.3.6.1.2.1.1 ...) Counter value = 2HostName</td>
<td>If Security parameters are set after setting the version v3 for a v1/v2 agent and request is made.</td>
</tr>
<tr>
<td>24</td>
<td>Enter the FileName of MibModule</td>
<td>While loading the MIBs file if OK button is clicked without selecting any file in &quot;LoadMibDialog&quot;.</td>
</tr>
<tr>
<td>25</td>
<td>Error Loading MIB:(filename) java io FileNotFoundException: Couldn't open stream for filename.cmi</td>
<td>If any file chosen from outside the MIBs directory.</td>
</tr>
<tr>
<td>26</td>
<td>Error Loading MIB:(filename with full path from home dir).cds</td>
<td>When a .cds file is loaded.</td>
</tr>
<tr>
<td>27</td>
<td>Error Sending PDU: Failed to Authenticate the Security Parameters for user authUser USMUserEntry not found for this user. TimeSynchronizationFailure could have occurred.</td>
<td>If ContextName/ContextEngineID is not set before making query for SnmpTable.(Database Mode)</td>
</tr>
<tr>
<td>28</td>
<td>Error in (get/getNext/getBulk)request to hostName:port no. Failed to Authenticate the Security Parameters for user authUser USMUserEntry not found for Address host name : 161</td>
<td>If ContextName/ContextEngineID is not set before making get/getNext/getBulk request</td>
</tr>
<tr>
<td>29</td>
<td>Error in get request from &lt; host name &gt;: 161 Unable to encode PDU.</td>
<td>get Request for a v3 User(Database mode) AuthProtocol: MD5 Context Name, ContextEngineID are not specified. Same exception is thrown for getnext, getbulk &amp; set operation also.(for AuthProtocol MD5&amp;SHA)</td>
</tr>
<tr>
<td>30</td>
<td>Error in get request from &lt; host name &gt;: 161 Failed to authenticate the security parameters for user privuser authKey length has to be 20.</td>
<td>get Request for a v3 User(Database mode) AuthProtocol: SHA Context Name, ContextEngineID are not specified.</td>
</tr>
<tr>
<td>31</td>
<td>can not plot the chosen variable: iso.org.dod.internet.mgmt.mib2.interfaces.ifTable</td>
<td>If IfTable OID is chosen for plotting Graph. The Error will be thrown if the OID chosen is not a leaf OID.</td>
</tr>
<tr>
<td>32</td>
<td>sent get request to tonyjpaul:161 Request Failed: SNMPv3 Error in Respownse: usm Stats Not InTimeWindows(.1.3.6.1.6.3.1.5.1.1.2.0) CounterValue =13 tonyjpaul</td>
<td>During getRequest for a v3 user Context Name, ContextEngineID not specified. Same Error Message is Thrown for other operations like getNext, getBulk, set</td>
</tr>
<tr>
<td>33</td>
<td>sent getbulk request to localhost:161 Request Failed: Get Response PDU received from 127.0.0.1. Error Indication in response : This is a end of</td>
<td>If getBulk Operation is done for &quot;org&quot;.(setting version v2/v3 for a v1 host) If the OID/LeafNode selected is the last node of the mib.</td>
</tr>
<tr>
<td>S.No.</td>
<td>Error Message</td>
<td>When it is displayed</td>
</tr>
<tr>
<td>------</td>
<td>--------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>34</td>
<td>MIB View. ObjectID: .1.3.6.1.2.1.9.1.4.9 NULLOBJECT:NULL</td>
<td>The same Exception will be thrown for operations like getNext, getBulk also.</td>
</tr>
<tr>
<td>35</td>
<td>Error Sending set Request: com.adventnet.snmp.beans.DataException: Error: Mib node unavailable for OID.</td>
<td>Setting value for an OID which is not having any leafnode. Now it's thrown if we are choosing any OID from &quot;enterprises&quot;.</td>
</tr>
<tr>
<td>36</td>
<td>sent get request to &lt; host name &gt;:161 Request Failed: Get Response PDU received from 192.168.1.182 Error Indication in response: There is no such instance in this MIB. ObjectID: .1.3.6.1.2.1.9.1.4.9 NULLOBJECT:NULL</td>
<td>Get Operation on sysServices node for a v3 agent. This Error is thrown if the leafnode sysServices is not implemented by the agent.</td>
</tr>
<tr>
<td>37</td>
<td>sent get request to &lt;host name &gt;:161 Request Failed: Get Response PDU received from 192.168.1.182 Error Indication in response: A no creation error occurred. Errindex:1</td>
<td>During set operation on ipRouteDest which has read-write access &amp; syntax of datatype:IP Address for a v3 user. As values can't be set if the column is not of Row-Status type.</td>
</tr>
<tr>
<td>38</td>
<td>Error in getting Database Connection. Please check the jdbc Parameters: com.adventnet.snmp.mibs.MibException: java.sql.SQLException: No suitable Driver.</td>
<td>Error is thrown when connecting to the database for an agent of any version, if the DriverName set is not correct.</td>
</tr>
<tr>
<td>39</td>
<td>Error in Getting Database Connection:Please check the jdbc Parameters: java.sql.SQLException: No suitable Driver.</td>
<td>This error is thrown when the URL set is not a valid URL for connecting to database.(mysql-&gt;msql)</td>
</tr>
<tr>
<td>40</td>
<td>Error in Getting Database Connection:Please check the jdbc Parameters: java.sql.SQLException: Can not Connect to MYSQL sever on smplinux:3306. Is there a mysql server is running in the machine/port you are trying to connect to? (java.net.UnknownHostException)</td>
<td>This error is thrown when the URL set is not a valid URL for connecting to database.(smplinux-&gt;smplinux)</td>
</tr>
<tr>
<td>41</td>
<td>Error in Getting Database Connection:Please check the jdbc Parameters: java.sql.SQLException: General error: Unknown database &quot;&lt;tst&gt;&quot;</td>
<td>This error is thrown when the URL set is not a valid URL for connecting to database.(test-&gt;tst)</td>
</tr>
<tr>
<td>42</td>
<td>sent set request to &lt;host name &gt;:161 Request Failed: SNMPv3 Error in Response : usmStatsNotInTimeWindows(.1.3.6.1.6.3.15.1.1.2.0)Counter value = 75 &lt;host name &gt;</td>
<td>Error thrown if the request is made after a certain interval of time..</td>
</tr>
<tr>
<td>43</td>
<td>sent get request to &lt;host name &gt;:161 Request Failed: Get Response PDU received from 192.168.1.182 Error Indication in response: There is no such object in this MIB. ObjectID: .1.3.6.1.2.1.8.1.0 NULLOBJECT:NULL</td>
<td>This OID is not instrumented for the agent you are querying for or no data is available in this OID.</td>
</tr>
</tbody>
</table>
FAQs

1. How do I give Hex strings for SET values in MibBrowser?
   
The SnmpString class accepts Hex strings in a certain format. Any string that starts and ends with a
   single quote(‘) is interpreted as an Hex string. The individual bytes should be separated using a
   colon(:). For example, if you need to enter 0x2a304cab, it should be supplied as ‘2a:30:4c:ab’.

2. How do I give Hex values in the ContextEngineID or in the ContextName text fields?
   
The Hex values should start with a 0x or 0X. Therefore, if you set a value for contextID or
   contextName, it should be 0xHHHHHH.

3. What are the units for timeout and retry values?
   
Both the timeout and retry values should be given in seconds. If you give the timeout value in
   milliseconds, it takes much time to get timed out. For example, a value of 1000 waits for 1000 seconds.

4. How do I set values for the table variables?
   
To create a new row in a table:
   - Define a column with SYNTAX RowStatus, and the definition for the table should have
     RowStatus object defined.
   - Select the Table node from the tree and the Table button from the toolbar to display the
     corresponding table.
   - Right-click on the table header where the name of the column is displayed. It displays a menu
     with the following options.
     - View Graph for Selected Rows
     - Add a New Row to the Table
     - Delete the Selected Rows from the Table
   - Select Add a New Row to the Table. It displays a window for entering the values of the table.
   - The value for the column with RowStatus syntax should be 4 for creating a new row.
   - Click OK after entering all the values.

If RowStatus is not present in the table definition, you can only modify the existing row by double-
   clicking the corresponding cell in the table.

5. I do not get the "NO HOST Specified" error. What should I do?
   
Before making any request, the host name or the IP address of the machine in which the agent is
   running should be specified in the "Host" text field of MibBrowser.
6. How can I load multiple MIB files in MibBrowser?

To load multiple MIBs, files should be separated by a blank space and be given within double quotes. For example,

```
java MibBrowserApplication -m "mibs/RFC1213-MIB mibs/RFC1271-MIB mibs/RFC1155-MIB" -h localhost -c public.
```

If you use MibBrowser.sh, edit the file accordingly.

7. When I ask for 10 rows in an SNMP table, the GETBULK returns only 6 rows and the last attribute of the sixth row is null. The sixth row seems to be truncated. What should I do?

The number of rows you get back may be limited by the PDU size permitted by your agent, manager, or transport.
TL1 Message Builder

- Overview
- Startup Options
- Main Screen
- Actions
- Loading Files
- General Options
- Working with command Set
- Working with Data Set
- Configuring the Message Builder
- Files Used by TL1 Message Builder

TL1 Message Builder is an user-friendly GUI tool to build (create/modify) XML-based TL1 message definitions and pre-built TL1 commands. These message definitions play a crucial role in automating various TL1 operations. The TL1 message definitions (metadata) are defined in a command set (.tcs) file and pre-built TL1 commands are defined in a data set (.dat) file.

This section explains the functionality available in the graphical interface and steps to create TL1 message definitions and pre-built commands.
Getting Started

TL1 Message Builder

TL1 Message Builder is an user-friendly GUI tool to build (create/modify) XML-based TL1 message definitions and pre-built TL1 commands. These message definitions play a crucial role in automating various TL1 operations. The TL1 message definitions (metadata) are defined in a command set (.tcs) file and pre-built TL1 commands are defined in a data set (.dat) file.

This section explains the functionality available in the graphical interface and steps to create TL1 message definitions and pre-built commands.

Key Features

Following are the key features offered by the TL1 Message Builder:

- **Modularized Message Definitions**: The command set and data set can be divided into different modules with each module containing message definitions specific to a device.
- **Grouping Message Definitions**: Allows developers to clearly view message definitions, either category-based or message-based.
- **Create Message Definitions from BNF**: Allows developers to create Input, Response and Autonomous message definitions based on the Bellcore and command-line specifications.
- **AID Modeling**: Allows developers to define AID patterns for components in an NE, Classify them using Patterns and Pattern Sets, and map them to the message definitions.
- **Template-driven Parameter and Block definitions**: Enables normalized parameter and block definitions at the module level, and associate them with multiple message definitions.
- **Enumeration and Range definitions**: Enables normalized enumeration and range value definitions at the module level, and associate them with multiple message definitions.
- **Command Glossary**: Contains a collection of descriptions of complex TL1 terms used in the Command Set. It is often used to describe about parameter values defined for the messages.
- **Create pre-built TL1 Commands**: Allows developers to create device commands based on the definitions used in the Command Set.
Startup Options

The **TL1 Message Builder** tool can be started in one of the following ways:

1. **Running the Script or Batch File**

   Run the batch or shell script **TL1MessageBuilder** from the `<Simulator_Home>/bin` directory. This script executes the `setenv` file to set the appropriate classpath variables.

   - **Windows**: `TL1MessageBuilder`
   - **Linux**: `sh TL1MessageBuilder.sh`

2. **Using the Launcher**

   The Message Builder tool can be started in the Launcher in one of the following ways.
   - Double-click on the TL1 Message Builder icon (or)
   - Right-click on the icon and select Run (or)
   - Select **Run** from the Options menu after selecting the **TL1 Message Builder**.

   **Note**: For Windows Installation, you can also start the tool from the **Start Menu -> Programs -> AdventNet Simulation Toolkit 6 -> TL1 Message Builder.**
Main Screen

The main screen of the TL1 Command Set Builder consists of two panels:

- **Left Panel**: Container for TL1MessageTree
- **Right Panel**: Details about the selected node in the TL1MessageTree (in left panel)

The Tree Panel has the capability to load multiple Command Set and Data Set files. Developers can easily traverse the message tree based on messages. The tree also allows one to view messages based on category.

The Details Panel enables developers to edit the details of the selected node (Request, Response, Notification, etc.). Each node in the message tree has its own details panel where parameters corresponding to that particular node object will be displayed.

The normal view of the TL1 Message Builder with a node selected in the Command Set is shown below:

The Data Set Builder tool enables users to construct pre-built TL1 commands. Unlike the Command Set, the Data Set tree displays only Input Message nodes. Each Input Message node can have multiple Data nodes (each representing a complete input message). The details panel of Data nodes allows the user to construct commands easily by filling the message templates.
The normal view of the TL1 Message Builder with a node selected in the Data Set is shown below:
Actions

Following are the actions that can be performed from the toolbar of the TL1 Message Builder.

<table>
<thead>
<tr>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a new Command set or Data Set file.</td>
</tr>
<tr>
<td>Load a Command Set or Data Set file.</td>
</tr>
<tr>
<td>Unload a Command Set or Data Set file.</td>
</tr>
<tr>
<td>Save selected Command Set or Data Set file.</td>
</tr>
<tr>
<td>Save all loaded files.</td>
</tr>
<tr>
<td>Search a particular node in the Message Tree.</td>
</tr>
<tr>
<td>View a Command Set or Data Set file.</td>
</tr>
<tr>
<td>Delete a node from Command Set or Data Set file.</td>
</tr>
<tr>
<td>Modify tool settings.</td>
</tr>
<tr>
<td>Open help documentation in a new Web browser.</td>
</tr>
<tr>
<td>Show Context-sensitive help of a selected component.</td>
</tr>
<tr>
<td>Close TL1 Message Builder application.</td>
</tr>
</tbody>
</table>

Shortcut Key Table

Following are the shortcut keys to perform various actions available in the TL1 Message Builder.

<table>
<thead>
<tr>
<th>Shortcut Key</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl+Alt+C</td>
<td>Create a new command set file</td>
</tr>
<tr>
<td>Ctrl+Alt+D</td>
<td>Create a new data set file</td>
</tr>
<tr>
<td>Ctrl+J</td>
<td>Load an existing command set or data set file</td>
</tr>
<tr>
<td>Ctrl+U</td>
<td>Unload a command set or data set file</td>
</tr>
<tr>
<td>Ctrl+S</td>
<td>Save command set or data set file</td>
</tr>
<tr>
<td>Ctrl+F</td>
<td>Find a node in the message tree</td>
</tr>
<tr>
<td>Ctrl-O</td>
<td>Modify tool settings</td>
</tr>
<tr>
<td>F2</td>
<td>Validate a command set file</td>
</tr>
<tr>
<td>F1</td>
<td>Open the help documentation</td>
</tr>
<tr>
<td>Alt+F4</td>
<td>Close the application</td>
</tr>
</tbody>
</table>
Loading Files

TL1 Message Builder provides support for loading multiple Command Set and Data Set files. Follow the steps given below to load and unload command set and data set files from the Message Builder.

To Load a Command Set File,

1. Select the Command Set Builder node in the TL1 Message Tree.
2. Select load from the File menu or from the toolbar. This opens a Load Dialog listing the command set and data set files under default directory.
3. Select a command set (.tcs) file to load and click the load button. The command set file gets loaded to the Message Tree.

To Load a Data Set File,

1. Select the Data Set Builder in the TL1 Message Tree.
2. Select load from the File menu or from the toolbar. This opens a Load Dialog listing the command set and data set files under default directory.
3. Select a data set (.dat) file to load and click the load button.
4. A TCS File chooser dialog opens. Select one of the loaded command set, based on which the data set will be built. Click the OK button to accept the association. Click the Cancel button to cancel the action.
5. On successful association, the data set file gets loaded to the TL1 Message Tree.

To Unload a File,

1. Select Unload from the File menu or from the toolbar. This opens up an Unload Dialog listing all the loaded files.
2. Select the file you wish to unload and click the Unload button.
3. Use the Select All check box to unload all the loaded files.
4. Click the Cancel button to cancel the operation
General Options

Following are some of the general options available in the TL1 Message Builder tool.

- Find option
- Look and Feel
- Generalize and Categorize option
- Expand and Collapse option
- Context-sensitive Help
- Conversion Tool
- Command Validator

Find Option

This option is invoked by selecting Find from the Edit menu or from the toolbar. It searches a particular child node under a selected node in the TL1 Message Tree easily.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find What</td>
<td>Uses this text in the search.</td>
</tr>
<tr>
<td>Find Next</td>
<td>Searches for the next node in the TL1 Message Tree.</td>
</tr>
<tr>
<td>Close</td>
<td>Ends the search operation and closes the dialog.</td>
</tr>
<tr>
<td>Match Case</td>
<td>Searches for case-sensitive occurrences of a node text. This limits the search and locates the text that matches the uppercase and lowercase characters you enter.</td>
</tr>
<tr>
<td>Match Word</td>
<td>Search for occurrences of a text as whole word.</td>
</tr>
</tbody>
</table>

Look and Feel

This option changes the look and feel of the Message Builder. Three options are available, Metal, Motif, and Native. Any one of these options can be selected based on the platform. This menu can be hidden from the menu bar by disabling it in the Tools -> Options Window. The default UI Look and Feel used by the tool is Metal.

Generalize and Categorize Message Tree

This option enables the user to view the messages in the Message Tree based on category. The user can choose to categorize the messages or generalize them, where all the messages are grouped under single node.
By default, all messages are listed command-wise (generalize form). To display the commands categorized, right-click on the Request node and select **Categorize**. Select **Generalize** to display them command-wise.

**Expand and Collapse Option**

This option is used to expand and collapse all the nodes in the TL1 Message Tree. To perform this function, select the option by doing a right-click on the TL1 Message Tree - root node.

**Context-sensitive Help**

This option describes the functionality of a particular component in the Message Builder. Select the context-sensitive button available in that dialog and place it on the component you wish to know.

**Conversion Tool**

This utility converts Input Commands in a data set (.dat) file to a textual form. The resultant text file contains a list of ready-to-use TL1 Commands which can be directly loaded and used in TL1 Craft Interface. The Data to Text Conversion tool can be invoked either by doing a right-click on the data set file node or from Tools menu.

**Command Validator**

This utility checks the syntax of a given command set file and generates an output report containing errors and warnings. To use this utility, select **Validate Command Set** from the Tools menu.
Working with Command Set

The TL1 Command Set file contains syntax of TL1 commands used between the manager and agent for management purposes. It contains the full description of each command, i.e. input, response, and autonomous message definitions.

TL1 Message Builder provides a user interface through which the Command Set files can be easily created or modified. The Command Set file is used to define TL1 Messages (Input/Response/Autonomous) in terms of message structure (syntax), parsing information, message categorization, grouping and associations, and various other related information such as description, alias, etc. A file can have many modules which represent a complete set of TL1 commands for a particular type of TL1 device. It is in a sense an equivalent of a MIB.

This book contains the following topics:

- Creating Input Message Definitions
- Example
- Creating Response Message Definitions
- Example
- Creating Autonomous Message Definitions
- Example
- Editing Alarm Codes
- Using Templates
- Using Resource Bundles
- Using Command Set Glossary
- Working with Patterns
- Viewing a Command Set File
Input Message Definitions

Creating Input Message Definitions

TL1 Message Builder helps you to create or edit TL1 Message Definitions based on the GR-831 format. This topic describes the process of creating each aspect of a complete TL1 Definition.

To create any TL1 Message Definition, you must create a Command Set file (.tcs). The file will be loaded to the TL1 Message Tree in the left panel. To create TL1 Message Definitions, you must first create a new module. A new module can be created by selecting Add New Module by right-clicking on the Command Set file. An input dialog is displayed where you must specify the module name. Click the OK button and the new module is added to the Message Tree as shown in the figure. The new module will have the following child nodes by default: Request, Response, Notification, Templates, Bundles, and Glossary.

Follow the steps given here to add a new Input Message Definition to the Command Set:

1. Right-click on the Request node element and select the Add New menu. A Create New Input Message dialog is displayed where you must specify the command name and verb (The modifiers can be specified in the Command Code block). Alternatively, you can also create input commands by simply specifying the BNF of that command. Click the OK button to commit the changes. The Input Message name is displayed as a node in the Request. This node will have the command as a child.
2. Now select the Input Message node. The right panel will display fields pertaining to each block in the Input Message, such as Command Code, Target Identifier, Access Identifier, General Block, Message Payload Block, etc.
3. Specify the category, description, prefix, suffix, and response name for each message.
4. Select each block (For example, Access Identifier, Message Payload Block etc.) and define the parameters for each of the input blocks.
5. Click the Update button to commit the changes made in each block.
6. Click the Save button to save the changes to the file.
7. Click the View File option from the File menu to view the changes made to the Command Set file.

Creation of a complete TL1 Input Message Definition is dealt with in detail in the next topic with an example.
Example

This topic leads you through a series of steps to create Input Message definitions using the TL1 Message Builder tool. To start with, assume you want to create an RTRV-PM-T1 command in the performance module.

Step 1

Create a new Command Set file (say, CISCO-SONET-ADM.tcs) and load it using the Load button from the File menu or toolbar. The file will be loaded into the TL1 Message Tree in the left panel and a new node with the filename will be added to the root node of the tree.

Step 2

Create a new module to define the input message syntax. To create a new module, right-click on the command set file and select the Add New Module from the drop-down menu. An Input dialog is displayed where you need to specify the module name. In this example, specify the module name as Performance. Click the OK button and the new module gets added to the Message Tree as shown in the figure. The new module will have Request, Response, Notification, Templates, Bundles, Glossary as child nodes by default.

Step 3

Now, right-click the Request node element and select Add New from the drop-down menu. A New Input Message dialog is displayed where you must specify the input message name and command verb. In this case, the input message name is Retrieve Performance and command verb is RTRV. Click the OK button to commit the changes. The input message name is displayed as a node in the Request. This node will have all the input commands.

Alternatively, Input Message can also be created by simply specifying the BNF of that command in the BNF editor. To specify the BNF of that command, enable the BNF String option in the Input Message Dialog and specify the command syntax as shown below.

Click the OK button to create the input command.
Step 4

Select the input message RTRV. The right panel will display fields pertaining to each block in the Input Message as shown in the figure. You can modify the input message name, category, description, and other fields as needed. The prefix and suffix are additional fields that can be used if you would like to append any text before and after the message. You can also create the response format for a particular request under Response element and map them to the request by selecting from the Response list.

![Image of TL1 Message Builder]

Step 5

Now, you can add definitions for each of the Input blocks. The table given below lists the parameter definitions for each of the Input Message blocks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Code</td>
<td>RTRV-PM-T1</td>
</tr>
<tr>
<td>Target Identifier</td>
<td>TID</td>
</tr>
<tr>
<td>Access Identifier</td>
<td>AID</td>
</tr>
<tr>
<td>Correlation Tag</td>
<td>Ctag</td>
</tr>
<tr>
<td>General Block</td>
<td>Null</td>
</tr>
<tr>
<td>Message Payload Block</td>
<td>Mon Type</td>
</tr>
<tr>
<td></td>
<td>Mon Level</td>
</tr>
<tr>
<td></td>
<td>Location</td>
</tr>
<tr>
<td></td>
<td>Empty</td>
</tr>
<tr>
<td></td>
<td>Tmper</td>
</tr>
<tr>
<td></td>
<td>Date</td>
</tr>
<tr>
<td></td>
<td>Time</td>
</tr>
</tbody>
</table>
In this case, the command code has two modifiers (PM and T1). Click the **Command Code** button in the right panel. The panel adjacent to it will display the Modifier One and Modifier Two blocks. Click the **Add** button corresponding to the first modifier block. A **Modifier pop up** dialog is displayed where you must specify the modifier name (**PM**). The separator is `'-' by default. If you would like to specify a filename which corresponds to a list of first level modifiers, then select the **rr** option and the file containing modifiers. Click the **OK** button to commit the changes. You can edit or remove the modifiers using the **Edit** and **Remove** buttons respectively.

Create the second modifier by clicking the **Add** button corresponding to the second modifier block. In the case of second modifier, you can set the prefix to `'-' or '##'. In this case, the prefix is `'-'.

Similarly create **Target Identifier**, **Access Identifier**, **Correlation Tag**, **General Block**, and **Message Payload Block**. Each of these blocks will have a Parameter Block table and a Parameter table that lists the names of each parameter in the command. The parameter block specifies the name of the block. For example, it can be AID, MPB, etc. The syntax details of each parameter should be given in the parameter table. Use the **Add**, **Edit**, and **Remove** buttons to add, modify and remove parameter blocks and parameters.

The parameter block is restricted to one in the case of **Target Identifier**, **Access Identifier**, **Correlation Tag**, and **General Block**. But the Message Payload Block can have any number of parameter blocks within it. In this case, the Message Payload Block also contains only one parameter block with many parameters. The parameter blocks are separated by colons while the individual parameters within the block are separated by commas.

**Step 6**

Click the **Update** button to update the corresponding block. Use the **Save** button to write the modified properties to the file.
Response Message Definitions

Creating Response Message Definitions

Now you would be familiar on how to create modules and define Input Messages. This topic explains you how to create Response Message definitions. Defining response messages in the Command Set help to automate most of the TL1 operations. The Response Message definition, like the Input Message, has multiple parameter blocks with parameter definitions.

All response messages must be defined under Response element. There can be one or more response messages inside the Response element. Follow the steps given here to define a new response message in the Command Set:

1. To create a response message, click the Add New item from the Edit menu or right-click the response node. A Create New Response Message dialog is displayed where you must specify the response name. Click the OK button to commit the changes. The response name will be displayed as a child in the response node.
2. Select the response name in the Message Tree. The right panel will display information specific to a TL1 response message, such as valid and error message blocks for quoted, unquoted, and comment lines.
3. Select each block and define the parameters that each block must contain. The response for a specific request can be a valid message or an error message.
4. Click the Update button to commit the changes made in each block.
5. Click the Save button to save the changes to the file.
6. Follow the above steps in sequence to define multiple set of response messages.
7. Click the View File option from the File menu to view the changes made to the command set file.

Creation of a complete TL1 Response Message Definition is explained in the next topic using a sample Response Message.
Example

This section will guide you through a series of steps to create Response Message definitions. The tutorial on creating Input Message definitions would have helped you to create an `RTRV-PM-T1` command. This tutorial will help you to create a response for the `RTRV-PM-T1` request.

**Step 1**

Click the **Add New** item from the edit menu or right-click on the Response node. A **Create New Response Message** dialog is displayed where you must specify the response name. In this case, the response name is `RESP-RTRV-PM`. Click the **OK** button to commit the changes. The response name will be displayed as a child in the response node.

**Step 2**

Select the response name `RESP-RTRV-PM` from the TL1 Message Tree, and the details are shown in the right panel. The response format for the `RTRV-PM` command is given here:

```
SID DATE TIME
M CTAG COMPLD
"<AID>, [<AIDTYPE>]:<MONTYPE>, <MONVAL>, [<VLDTY>], [<LOCN>], [<DIRN>],
[<TMPER>], [<DATE>], [<TIME>]"
```

In this case, the text block is quoted and contains two parameter blocks. Each parameter block contains parameters.

Now you must define the parameter blocks and the parameters in the response details column. Since the response block contains an quoted line, select **Valid Quoted Response** button and create parameter blocks. To create a parameter block, click **Add** button corresponding to the parameter block. A **Parameter Block** dialog is displayed where you must specify the block names. The block names are used to identify a particular block in the response. For example, you can have the block names as block1 and block2. Since both the blocks are mandatory, the depth is 0.

**Step 3**

Now you must define the parameters for the parameter block. To add parameters to a block, select the block and click the **Add** button corresponding to the parameter. The **Parameter** dialog is displayed where you must specify the parameter details, such as name, type, mode etc. Click **OK** to commit the changes. In the same way, you can add all the parameters to the block. You can also specify the error response for the request in the same manner.
Step 4

Click the **Update** button to commit the changes and **Save** button to save the changes to the file.
Autonomous Message Definitions

Creating Autonomous Message Definitions

This topic explains how to create Autonomous Message definitions. Follow the steps given here to add a new Autonomous Message definition to the Command Set:

1. Right-click on the Notification node element and select the Add New menu. A Create New Autonomous Message dialog is displayed where you must specify the Autonomous Message name and output code verb (The modifiers can be specified in the output code block). Click the OK button to commit the changes. The Autonomous Message name is displayed as a node in the Notification. This node will have the output code as a child.
2. Now select the Autonomous Message node. The right panel displays the fields pertaining to each block in the Autonomous Message, such as output code and quoted, unquoted and commented message blocks. The panel also has provision to edit category, default alarm and description fields for each of the Autonomous Messages.
3. Create parameter blocks and define the parameters that each block must contain.
4. Once the parameters are specified, click the Update button to commit the changes made to each block.
5. Click the Save button to save the changes to the file.
6. Click the View File option from the File menu to view the changes made to the Command Set file.

The next topic illustrates the creation of Autonomous Message definitions with an example.
Example

This section guides you through a series of steps to create Autonomous Message definitions using a test example **REPT ALM ENV**.

```
SID DATE TIME
** ATAG REPT ALM ENV
  "<AID>:<NTFCNCD>;<ALMTYPE>,;[<ALMMSG>]
```

**Step 1**

Right-click on the notification node element in the Message Tree and select the **Add New** menu. A **Create New Autonomous Message** dialog is displayed where you must specify the Autonomous Message name and output code verb. In this case, the Autonomous Message name is **REPORT** and the output verb is **REPT**. Click the **OK** button to commit the changes. The Autonomous Message name is displayed as a node in the notification. This node will have output code as a child.

**Step 2**

Select the autonomous message node or the output code from the TL1 Message Tree. The right panel will display the autonomous message details. Now, define the **category**, **default alarm**, and **description** fields for this autonomous message.

**Step 3**

The right panel also displays blocks, such as output code and quoted and unquoted message block. You can modify the output code for the notification by selecting the **Auto Code** button. Click the **Add** button corresponding to the first modifier block. A **Create New Modifier** dialog is displayed where you must specify the modifier name (**ALM**).

To specify a file name that corresponds to a list of first-level modifiers, enable the **rr** option and select the file using the **...** option. Click **OK** button to commit the changes. You can edit or remove the
modifiers using the **Edit** and **Remove** buttons respectively. In the same way, you can create a second modifier by clicking the **Add** button corresponding to the second modifier block.

**Step 4**

Now define the text block parameters. Since the text block is quoted, select the **Quoted tab** to specify the parameter blocks. To create a parameter block, click the **Add** button from the parameter block panel. A **Parameter Block** dialog is displayed where you must specify the block names. The block names are used to identify a particular block in the text block. For example, you can have the block names as block1 and block2. Since both the blocks are mandatory, the depth is 0.

Specify the parameters for the parameter block. To add parameters to a block, select the block and click the **Add** button from the parameter panel. The **Parameter** dialog is displayed where you must specify the details with respect to that parameter such as name, type, mode, etc. Click **OK** to commit the changes. Likewise, you can add rest of the parameters to the parameter block.

If the autonomous message contains an unquoted text block, select the **Unquoted tab** and specify the parameter blocks and parameters.

**Step 5**

Click the **Update** button to commit the changes and **Save** button to save the changes to the file.
Editing Alarm Codes

Alarm codes are a part of TL1 Autonomous Message and denotes the severity of the message. The Message Builder allows you to specify the alarm codes as part of the message definition in the command set file. You can add new alarm codes or modify the existing alarm codes through the Alarm Code Editor. To open this editor, click the **Alarm Code Editor** button from the Tools menu.

Select the file and the module for which the alarm codes are to be updated. Click the **Add** button to add a new alarm code. Select the alarm code and click the **Remove** button to remove the alarm code from the table. You may even choose the default alarm codes by clicking the **Default** button. After modifying the alarm code table, select the **Update** button to update the changes in the command set file. Click the **Close** button to close this window.

The default values for alarm code is shown in the table.

<table>
<thead>
<tr>
<th>Code</th>
<th>Severity</th>
</tr>
</thead>
<tbody>
<tr>
<td>*C</td>
<td>Critical</td>
</tr>
<tr>
<td>**</td>
<td>Major</td>
</tr>
<tr>
<td>^</td>
<td>Minor</td>
</tr>
<tr>
<td>A^</td>
<td>Non Alarm</td>
</tr>
</tbody>
</table>
Using Templates

Parameter Block

Parameter Block element is the basic grouping unit to construct a TL1 message. A set of parameter blocks forms a parameter group and each parameter block in a parameter group is separated by a colon (:). Each parameter block can have a set of parameters associated with it.

Following properties can be defined at the Parameter Block level:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Represents a unique name of the parameter block.</td>
</tr>
<tr>
<td>type</td>
<td>Represents the block type. It can be either list, range, listrange, or compound.</td>
</tr>
<tr>
<td>depth</td>
<td>Represents the degree of option (including the separator before the parameter block).</td>
</tr>
<tr>
<td>description</td>
<td>Description text about the parameter block.</td>
</tr>
</tbody>
</table>

The Command Set API provides support for creating templates for parameter blocks. By creating templates for parameter blocks, the user can reuse the same parameter block properties at multiple message instances.

How to Create a Parameter Block Template

1. Load a Command Set file and select **Templates -> ParameterBlock** node from the TL1 Message Tree.
2. Right-click on the **ParameterBlock** node and select **Add ParameterBlockTemplate** item from the drop-down menu.
3. A **New Parameter Block Template** dialog opens up. Provide a name for the template and click the **OK** button. The template gets added to the Message Tree.
4. Select the template. The right panel displays the parameter block table and parameter table.
5. Add parameter blocks and their associated parameter properties to the table.
6. Click the **Update** button to update the changes.
7. Click the **Save** button to save the changes to the file.

Now the parameter block template is created. You can use this template in any of the message definitions by simply selecting the **Template Name** under parameter block and the appropriate parameter block defined in this template.

Parameter

Parameter element is a single TL1 Param representing a particular entity in the TL1 device. The parameter can be either name-defined or position-defined. If the parameter is name defined, it will be of the form **Name = Value**. If the parameter is position-defined, it will be of the form **Value**. Each parameter in a parameter block is separated by a comma (,).

Following properties can be defined at the Parameter level:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Represents a unique name of the parameter block.</td>
</tr>
<tr>
<td>type</td>
<td>Determines the parameter type. It can be position-defined or name-defined.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>mode</td>
<td>Determines whether the parameter is mandatory or optional in that Parameter Block.</td>
</tr>
<tr>
<td>depth</td>
<td>Represents the degree of option (including the separator before the parameter).</td>
</tr>
<tr>
<td>alias</td>
<td>Alternative name for that parameter.</td>
</tr>
<tr>
<td>description</td>
<td>Description text about that parameter.</td>
</tr>
<tr>
<td>datatype</td>
<td>Identify whether the parameter is a string, character, integer, or float type.</td>
</tr>
<tr>
<td>patternsetname</td>
<td>The Pattern Set can contain number of patterns. A pattern defines a way in which a parameter can be represented. This field is optional and if specified, will be used to validate that parameter with corresponding pattern set defined.</td>
</tr>
<tr>
<td>wordlist</td>
<td>The word list in this glossary contains description about this parameter, or their allowed values. This field is also optional.</td>
</tr>
<tr>
<td>range</td>
<td>Specifies the minimum and maximum values that a parameter can hold.</td>
</tr>
<tr>
<td>enum</td>
<td>Array of values that a parameter can hold.</td>
</tr>
</tbody>
</table>

The Command Set API also provides support for creating templates for parameters. By creating templates for the parameters, the user can reuse the same parameter properties at multiple parameter block instances.

**How to Create a Parameter Template**

1. Load a Command Set file and select **Templates -> Parameter** node from the TL1 Message Tree.
2. Right-click on the **Parameter** node and select **Add ParameterTemplate** item from the drop-down menu.
3. A **New Parameter Template** dialog opens up. Provide a name for the template and click the **OK** button. The template gets added to the Message Tree.
4. Select the template. The right panel displays the parameter table.
5. Add parameters and their properties to the table.
6. Click the **Update** button to update the changes.
7. Click the **Save** button to save the changes to the file.

You can now use this template in any of the parameter block definitions by simply selecting the **Template Name** under parameter and the appropriate parameter defined in this template.
Using Resource Bundles

Enumeration Bundle

The Enumeration Bundle provides a way to define a set of possible values that a TL1 Parameter can hold. You can define the enumeration values for a TL1 Parameter using the User-defined option under Enumeration in the Parameter Dialog. Also, the tool facilitates building enumeration definitions at the global level so that it can be re-used across multiple TL1 parameters.

To Create an Enumeration Bundle,

1. Load a Command Set file and select Bundles -> Enumeration node from the TL1 Message Tree. The right panel displays the Enum Bundle table.
2. Click the Add button in the right panel. The Add Enumeration Entries dialog pops up.
3. Specify the Enum name in the Name field. Enter the possible allowed values for the Enum parameter to the Enum Values table.
4. Click the OK button. The Enum parameter gets added to the Enum Bundle table.
5. Continue to add more Enum parameters to the Enum Bundle table.
6. Click the Update button to update the changes.
7. Click the Save button to save the changes to the file.

Once the Enumeration Bundle is created, you can map them to the appropriate parameter by simply selecting the Enum Name from the Enumeration combo in the Parameter Dialog.

Range Bundle

The Range Bundle specifies the minimum and maximum values that a TL1 Parameter can accept. You can define the range values for any parameter using the User-defined option under Range in the Parameter Dialog. Also, the tool facilitates building range definitions at the global level so that it can be re-used across multiple TL1 parameters.

To Create a Range Bundle,

1. Load a Command Set file and select Bundles -> Range node from the TL1 Message Tree. The right panel displays the Range Bundle table.
2. Click the Add button. The Add Range Entries dialog pops up.
3. Specify the **Range Name**. Enter the lower and upper limits for the range parameter in the **From** and **To** fields. Click the **OK** button. The Range parameter gets added to the **Range Bundle** table.

4. Continue to add more range parameters to the **Range Bundle** table.

5. Click the **Update** button to update the changes.

6. Click the **Save** button to save the changes to the file.

Once the Range Bundle is created, you can map them to the appropriate parameter by simply selecting the **Range Name** from the Range combo in the Parameter Dialog.
Using Command Set Glossary

The Command Set Glossary is a single-point store where definitions about each parameter, enumeration or range entity can be presented in a logical way. The Word Lists defined in the Glossary must be associated properly with the appropriate parameter, enumeration or range value to get the complete information about that entity.

To Create a Command Set Glossary,

1. Load a Command Set file and select Glossary node from the TL1 Message Tree.
2. Right-click on the Glossary node and select Add Word List button. The Create New Word List dialog pops up.
3. Specify the Word List name and click the OK button. The Word List gets added under Glossary node in the TL1 Message Tree.
4. To add words to the Word List, select the Word List. The right panel shows an empty word table.
5. Click the Add button in the right panel and Add New Word dialog pops up. Specify Name, Alias, and Description entries for each word.
6. Click the Update button to update the changes.
7. Continue to add new word lists to the Glossary node.
8. Click the Save button to save the changes to the file.

Once the Command Set Glossary list is created, you can map them to the appropriate parameter, enumeration or range entity by simply selecting the Word List name from the Word List combo in the Parameter Dialog.
Working with Patterns

Patterns allow developers to define the format in which the parameters appear in real-time. This feature is mostly used by developers to model the Access Identifiers which denote the actual components in an NE. These AID patterns are contained within a pattern set and a pattern set can contain multiple patterns.

Since AID modelling is done based on Input Messages (and not vice versa), the Input Messages have to be associated with the Pattern Set. Once the association is made, data is created strictly based on these patterns.

The pattern contains the following entries:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pattern Name</td>
<td>A name that identifies a pattern. Typically it represents a component in an NE.</td>
</tr>
<tr>
<td>Category</td>
<td>Category to which the particular pattern belongs to. Useful when displaying the AID pattern hierarchy in message tree.</td>
</tr>
<tr>
<td>Pattern</td>
<td>A string representing an AID pattern.</td>
</tr>
<tr>
<td>Condition</td>
<td>A condition string in a specified format shown below. This is used as an alternative for pattern.</td>
</tr>
<tr>
<td>Interface</td>
<td>An implementation of PatternMatcher interface available in the com.adventnet.tl1.commandset package.</td>
</tr>
</tbody>
</table>

Following are the keywords to be used to build a condition expression.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contains::&lt;str&gt;</td>
<td>Check if the value string contains &lt;str&gt;.</td>
</tr>
<tr>
<td>!contains::&lt;str&gt;</td>
<td>Check if the value string does not contain &lt;str&gt;.</td>
</tr>
<tr>
<td>equals::&lt;str&gt;</td>
<td>Check if the value string matches &lt;str&gt;.</td>
</tr>
<tr>
<td>!equals::&lt;str&gt;</td>
<td>Check if the value string does not match &lt;str&gt;.</td>
</tr>
<tr>
<td>indexof[start]*[&lt;end&gt;]:&lt;str&gt;</td>
<td>Check if the value string matches with &lt;str&gt; starting from the position specified in &lt;start&gt;.</td>
</tr>
<tr>
<td>!indexof[start]*[&lt;end&gt;]:&lt;str&gt;</td>
<td>Check if the value string does not match with &lt;str&gt; starting from the position specified in &lt;start&gt;.</td>
</tr>
<tr>
<td>startswith::&lt;str&gt;</td>
<td>Check if the value string starts with &lt;str&gt;.</td>
</tr>
<tr>
<td>!startswith::&lt;str&gt;</td>
<td>Check if the value string does not start with &lt;str&gt;.</td>
</tr>
<tr>
<td>endswith::&lt;str&gt;</td>
<td>Check if the value string ends with &lt;str&gt;.</td>
</tr>
<tr>
<td>!endswith::&lt;str&gt;</td>
<td>Check if the value string does not end with &lt;str&gt;.</td>
</tr>
</tbody>
</table>

Example of TL1 patterns:

1. (startswith::ST1)
2. ((endswith::ST1)$$(contains[0*3]:-))
3. ((startswith::OC)||(endswith::5))
4. (equals::ALL)
5. (indexof[0*10]:OC)
6. (((startswith::ST1)&&((endswith::1))||(equals::ST1-1))
7. ((startswith::ST1)||(startswith::ST1))((endswith::5))

To Create a Pattern Set,

1. Load a Command Set file. Traverse the message tree and select Templates -> PatternSet. Right-click on the Pattern Set node and select Add Pattern Set.
2. A **Create New Pattern Set** dialog pops up. Specify the pattern set name and click the **OK** button. The pattern set name is added to the message tree under Pattern Set.

3. Now, select the pattern set name. The right-panel displays an empty table. Right-click and select **Add Pattern**. A **Create New Pattern** dialog is invoked.

4. Specify the pattern name, Category, Pattern, Condition, and Interface fields for that pattern. Click the **OK** button. A row is created and the pattern details are displayed.

5. Click the **Save** button to save the pattern details to the command set file.

The pattern details are specified in the right panel as shown in the image. A Pattern Set can contain any number of patterns. If multiple patterns are defined, the parameter value string is validated for each pattern defined in that pattern set.

Once pattern sets are created, they must be associated with appropriate parameters for validating against pattern set.
Viewing a Command Set File

The TL1 Message Builder tool provides you the ease of building a command set hiding the complexities of the XML structure by providing an easy-to-use graphical interface. The information provided in the command set would help you to create TL1 message definitions specific to a TL1 device or a group of TL1 devices. This topic will explain how to view those files from the tool in the XML format.

You have to load a Command Set file in order to view it. In case you have loaded many Command Set files, select the file to view and click the File button from the View menu. This will open a new window that displays the Command Set file as shown in the figure. If no file is loaded or the file is not selected, an error dialog is displayed.

Click the Close button to close the window.
Working with Data Set

The TL1 Command Set defines the complete syntax of a TL1 command and also the response for a particular command. The TL1 Data Set specifies the values that each parameter in the Command Set can hold. There can be different set of values for a same Input Message.

TL1 Message Builder provides a User Interface through which the Data Set files can be easily created or modified. The structure of the TL1 Data Set is almost similar to the TL1 Command Set. The only difference is that the Data Set does not have Responses and Notifications elements defined in it. The Input Message in the Data Set can have one or more data. Each of these Data Set values is specified within the unique tag name called Data.

The data created for the message definitions are provided as API objects which can be used by applications to send commands to the device.

This book contains the following topics:

- Creating a TL1 Data Set
- Viewing Data Set Files
- Example
Creating a TL1 Data Set

Now you will be familiar with creating Command Set files and the purpose of using it. The Command Set files contain definitions of TL1 messages. In other words, they specify the parameters that each TL1 message (Input, Response, Autonomous) can contain. But the real essence of communication lies in the data that is actually exchanged. The Command Set files do not deal anything with the data. The Data Set files are used for this purpose. The Data Set file holds the data about each input message defined in the Command Set file. The Data Set editor is used for creating or editing the Data Set files.

What Are Data Set Names?

As you know, there can be more than one value for each of the Input Messages defined in the Command Set. That is, you may have one or more Access identifiers or Payload Blocks for a single Input Message. But, how to create multiple data set values remains a question. Here is where Data Set name comes into picture. You can have many names for the single Input Message. Each name will have unique data set values.

Creating Data Set Name and Values

Follow the steps given here to add a new Data Set name and value to the TL1 Data Set:

1. Firstly, load the Command Set file in the TL1 Message Tree for which data has to be created.
2. Now, click the New button from the File menu and create a new data set file from the File Chooser dialog. The Data Set file gets loaded into the TL1 Message Tree. This file will have only the input messages defined in the Command Set. You cannot create values for the response or notification messages since the values for these messages are created by the Agent during run time.
3. Traverse the TL1 Message Tree and select the Input Message for which data names are to be created. Right-click on the Input Message and select Add Data. You have to specify a unique data name for the Input Message. Click the OK button to commit the changes. This name will become a child to the corresponding Input Message in the TL1 Message Tree.
4. Select a data set name from the Message Tree. The right panel displays the parameters defined under each block in the Data Set file, such as Target Identifier block, Access Identifier Block, General Block, and Message Payload Block. All these blocks will have a table with the corresponding parameter names from the Command Set in the left-side column. The right-side column must be filled in with the actual values.
5. Click the Update button to commit the changes made in each block.
6. Click the Save button to save the changes to the file.

You can repeat this process to create multiple Data Set names. But each Data Set name must be unique in itself.

The next topic explains the creation of Data Set files using sample Data Set values.
Viewing Data Set Files

This topic explains how to view Data Set files using the tool in the XML format. To view a Data Set file, you need to load it. As you know, the Data Set file contains the data related to the Command Set. So, you must first load the Command Set file and then the Data Set file to be associated with the Command Set. Click the File button from the View menu to see the source.

In case you have loaded many Data Set files, select the file to view and click the File button from the View menu. This will open a new window that displays the Data Set file as shown in the figure. If no file is loaded or selected, an error dialog will be displayed.

Click the Close button to close the window.
Example

This topic takes you through a sequence of steps to create data set values for the parameters defined in the Command Set.

Let us say that the file `CISCO-SONET-ADM.tcs` contains `RTRV-PM-T1` command. The definitions specified in the Command Set file are shown in the left column while the values to be entered in the Data Set are shown in the right column of the table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Code</td>
<td>RTRV-PM-T1</td>
</tr>
<tr>
<td>Target ID</td>
<td>TID</td>
</tr>
<tr>
<td>Access ID</td>
<td>FAC-2-1</td>
</tr>
<tr>
<td>Correlation Tag</td>
<td>123</td>
</tr>
<tr>
<td>General Block</td>
<td>Null</td>
</tr>
<tr>
<td>Mon Type</td>
<td>CVL</td>
</tr>
<tr>
<td>Mon Level</td>
<td>0-UP</td>
</tr>
<tr>
<td>Location</td>
<td>NEND</td>
</tr>
<tr>
<td>Empty</td>
<td>Null</td>
</tr>
<tr>
<td>Tmper</td>
<td>15-MIN</td>
</tr>
<tr>
<td>Date</td>
<td>04-18</td>
</tr>
<tr>
<td>Time</td>
<td>12-45</td>
</tr>
</tbody>
</table>

Step 1

To begin with, let us assume that the Command Set file `CISCO-SONET-ADM.tcs` is loaded. Follow the steps given below to load a new data set file:

1. Click **New -> Data Set** button from the File menu or toolbar. A load dialog gets displayed.
2. Create or select a new data file and click the **Create New** button. A Tcs File Chooser dialog is displayed which shows the Command Set files already loaded. Select `CISCO-SONET-ADM.tcs` from the list.
3. Click the **OK** button to load the new Data Set file.

Step 2

Now the Data Set file is loaded to the TL1 Message Tree. Traverse the Message Tree and select the `RTRV-PM-T1` command. Right-click on the selected command and click **Add Data**. A pop-up dialog is displayed where you must enter a data name (For example, Performance Data). The data name holds a data for that particular command. There can be multiple data names for a given Input Message. Each data name must be unique. Click the **OK** button to commit the changes. Now, the data name is displayed as a child for the Input Message `RTRV-PM-T1`.

Step 3

Select **Performance Data** from the Message Tree. Now, the right-side panel displays the data name, command code, and various blocks corresponding to the Input Message. You can edit the data name at any time.
Step 4

Now select each of the blocks in the Input Message and add the values as given in the table mentioned above. Once a particular block is completed, click the **Update** button to commit the changes. Also, there can be multiple blocks inside the message payload block (in this case, there is only one block). Each block name is shown in the Parameter Block combo box. Select each Parameter Block from the combo and add values to it.

![Screenshot of the interface showing the data set creation process](image)

Step 5

Click the **Save** button to save the changes to the file.

Step 6

In this way, you can create multiple data names for a same Input Message or all the Input Messages given in the Command Set.

Step 7

Click the **File** option from the View menu to view the Data Set file.
Configuring TL1 Message Builder

- Command
- Resources

The TL1 Message Builder tool provides options to change the default values, such as help or resource file location, enable or disable tree nodes etc. These values are stored to tl1messagebuilder.config file present under <Simulator_Home>/conf/ directory. file and used as default values while starting the tool. The options dialog can be opened by selecting the Options button from the Tools menu.

Command Option

The Command option can be used to select the default values for the delimiter (Separator between modifiers in command code), prefix, and suffix of TL1 Input Message.

Also, the Access Identifier block in the Command Set file supports parameter value complex. The parameter value complex has information units which can be combined using punctuation characters to form more complicated parameter values expressing compound values, groups of values, ranges of values, etc. The default parameter block types can be configured here.

Resources Option

Resources option will let you modify the default location of the files used by the Message Builder, such as Help Documentation and Resource files. You can modify the existing location by selecting the ... button. It also allows you to enable or disable the Tree Nodes in the Message Tree. Click the OK or Apply button to save the changes.
Files Used by TL1 Message Builder

Following files are used by the TL1 Message Builder for performing various functions.

1) Command Set file (.tcs)

The Command Set file contains TL1 command definitions in XML format. Apart from command definitions, it also contains response message and autonomous message details specific to each TL1 device. You can refer the reference command set files bundled in the product for SONET, DSLAM devices etc.

2) Data Set file (.dat)

The Data Set file contains values for each command defined in the Command Set in XML format. You can refer the reference data set files bundled in the product for SONET, DSLAM devices etc.

3) Configuration file (tl1messagebuilder.config)

The configuration file is used to store the default settings of the Message Builder such as, Command, User Interface options. You can use the Options Window in the Message Builder to modify the default settings.

4) Properties file (tl1messagebuilder_i18n.properties)

This file can be used as a reference for creating your own locale-specific properties file for internationalizing the Message Builder tool. The locale-specific properties can be edited using the I18N Editor tool.

5) Context-sensitive help file (tl1messagebuilder_csh.xml)

This file contains short description about each component used in the Message Builder UI. When the context-sensitive icon is placed on a particular component, the description about that component is displayed in a small window.
TL1 Craft Interface

- Overview
- Startup Options
- Main Screen
- Loading Files
- Actions
- General Options
- Establishing Connection
- Managing Multiple Connections
- Sending and Receiving Messages
- Sending and Receiving Bulk Messages
- Configuring TL1 Craft Interface
- Viewing Log Messages
- Files used by TL1 Craft Interface

**TL1 Craft Interface** is an user-friendly GUI tool that enables craft operators to easily manage the TL1 infrastructure. It helps lab technicians and field engineers to test, monitor, administer, and provision multiple TL1 agents. While the normal Telnet interface is too simple, this tool offers a sophisticated and productive environment.

This section explains the functionality available in the graphical interface and ways to interact with TL1 agents. The TL1 Craft Interface tool is built over AdventNet TL1 API and can manage TL1 devices from multiple equipment vendors.
Getting Started

Overview

TL1 Craft Interface is an user-friendly GUI tool that enables craft operators to easily manage the TL1 infrastructure. It helps lab technicians and field engineers to test, monitor, administer, and provision multiple TL1 agents. While the normal Telnet interface is too simple, this tool offers a sophisticated and productive environment.

This section explains the functionality available in the graphical interface and ways to interact with TL1 agents. The TL1 Craft Interface tool is built over AdventNet TL1 API and can manage TL1 devices from multiple equipment vendors.

Key Features

Following are the key features offered by the TL1 Craft Interface:

- **Connectivity with multiple TL1 devices**: Easily manage and monitor multiple TL1 devices simultaneously.
- **Load pre-built TL1 commands**: This will simplify the tedious task of dealing with various device-specific TL1 messages.
- **Load pre-built message templates**: Helps administrators to easily provision/configure a particular element in an NE such as, Equipment, Facility, Cross-connect etc. The message templates are XML-based definitions (Command Set) and are built using TL1 Message Builder.
- **Automatic Message Rationalization**: Allows craft operators to easily understand messages received from multiple TL1 devices. This is done by upgrading the received messages based on the TL1 message definitions.
- **Bulk Message Support**: Allows field engineers to run routine diagnostic tests. These tests can be carried out on-demand to isolate problems on a particular element within an NE.
- **Multiple Transport Protocols**: Allows craft operators to communicate with TL1 devices via, TCP, Telnet, or Serial transport. The tool also provides support for plug-in custom transport protocols.
- **Session Logging**: Allows administrators to generate and browse historic information of TL1 devices. Offers transaction-based logging as well as raw data logging.
Startup Options

The TL1 Craft Interface application can be started in one of the following ways:

1. Running the Script or Batch file

Run the batch or shell script TL1CraftInterface from the <Simulator_Home>/bin directory. This script executes the setenv file to set the appropriate classpath variables.

<table>
<thead>
<tr>
<th>Windows:</th>
<th>TL1CraftInterface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux:</td>
<td>sh TL1CraftInterface.sh</td>
</tr>
</tbody>
</table>

2. Using the Launcher

The Craft Interface tool can be started in the Launcher in one of the following ways.
- Double-click on the TL1 Craft Interface icon (or)
- Right-click on the icon and select Run (or)
- Select Run from the Options menu after selecting the TL1 Craft Interface.

Note: For Windows Installation, you can also start the tool from the Start Menu -> Programs -> AdventNet Simulation Toolkit 6 -> TL1 Craft Interface.
Main Screen

The main screen of the TL1 Craft Interface consists of two panels:

- **Left Panel**
- **Right Panel**

The Left Panel holds the Message Tree and is used for loading TL1 commands. Multiple files can be loaded to the Message Tree. The commands loaded in the tree can be selected and sent to the device directly. The Message Tree allows you to load command set, data set, and text files.

The Right Panel is split into two parts. The upper half holds the Input Message details. It contains a text field to show details about the selected TL1 Input Message. It also holds a table that individually displays the parameters specified in each block in the TL1 command. Select a particular block to view the parameters defined in the block. You can also edit the parameters under each block, which gets reflected in the Input Message field.

The lower half holds the text area which displays messages received from the TL1 device. This text area can be combined or split. In the combined fashion, all the messages are received in the single text area. In split fashion, responses and notifications are received separately. You can change the mode using the **Show Autonomous and Response Message in Same Window** Check box.
Loading Files

TL1 Craft Interface enables loading of pre-built TL1 commands and simplifies the tedious task of dealing with various device-specific TL1 messages. Multiple files can be loaded to the TL1 Message Tree and can be selected to be sent to the device. You can load command set, data set, or text files in the TL1 Message Tree. These files can be created using the TL1 Message Builder tool.

This section explains how to load and unload files from the Craft Interface.

To Load a File

1. Click the Load button from the File menu. Select the file type, i.e. Command Set, Data Set, or Text.
2. A Load dialog opens up. Select the file to load.
3. Click the Load button to load the file in the TL1 Message Tree.
4. Click the Cancel button to cancel the load operation.

Note: If you would like to load command set and data set files, you must first move to the tree mode. To load text files, you must move to the list mode.

To Unload a File

1. Click the Unload button from the File menu or toolbar. This opens up a Unload dialog that lists all the loaded files.
2. Select the file you wish to unload and click the Unload button.
3. Use the Select All check box to unload all the loaded files.
4. Click the Cancel button to cancel the operation.
### Toolbar Icons

Following are the actions that can be performed from the toolbar in the TL1 Craft Interface.

<table>
<thead>
<tr>
<th>Action</th>
<th>Shortcut Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load command set, data set, or text files.</td>
<td>Ctrl+Alt+C</td>
</tr>
<tr>
<td>Unload command set, data set, or text file.</td>
<td>Ctrl+Alt+D</td>
</tr>
<tr>
<td>Find a particular node in the TL1 Message Tree.</td>
<td>Ctrl+Alt+T</td>
</tr>
<tr>
<td>Shift the mode from tree to list.</td>
<td>Ctrl+Alt+U</td>
</tr>
<tr>
<td>Shift the mode from list to tree.</td>
<td>Ctrl+F</td>
</tr>
<tr>
<td>Open a connection with the TL1 device.</td>
<td>Ctrl+O</td>
</tr>
<tr>
<td>Close the connection with the TL1 device.</td>
<td>Ctrl+W</td>
</tr>
<tr>
<td>View the TL1 session properties.</td>
<td>Ctrl+M</td>
</tr>
<tr>
<td>Send messages to the TL1 device.</td>
<td>Ctrl+Alt+B</td>
</tr>
<tr>
<td>Switch to Bulk mode to send multiple messages at a time.</td>
<td>Ctrl+Alt+B</td>
</tr>
<tr>
<td>Switch to Normal mode to send one message at a time.</td>
<td></td>
</tr>
<tr>
<td>View default settings of the TL1 Craft Interface.</td>
<td></td>
</tr>
<tr>
<td>Clear messages from the text area.</td>
<td></td>
</tr>
<tr>
<td>Open Help Documentation in a Web Browser.</td>
<td></td>
</tr>
<tr>
<td>Show Context-sensitive help of a selected component.</td>
<td></td>
</tr>
<tr>
<td>Close Craft Interface Application.</td>
<td></td>
</tr>
</tbody>
</table>

### Shortcut Key Table

Following are the shortcut keys to perform various actions available in the TL1 Craft Interface.

<table>
<thead>
<tr>
<th>Shortcut Key</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl+Alt+C</td>
<td>Load a Command Set file in the Message Tree</td>
</tr>
<tr>
<td>Ctrl+Alt+D</td>
<td>Load a Data Set file in the Message Tree</td>
</tr>
<tr>
<td>Ctrl+Alt+T</td>
<td>Load a Text file to the List</td>
</tr>
<tr>
<td>Ctrl+U</td>
<td>Unload files from the Message Tree or List</td>
</tr>
<tr>
<td>Ctrl+F</td>
<td>Find a particular node in the TL1 Message Tree</td>
</tr>
<tr>
<td>Ctrl+O</td>
<td>Open connection with the TL1 device</td>
</tr>
<tr>
<td>Ctrl+W</td>
<td>Close the TL1 connection</td>
</tr>
<tr>
<td>Ctrl+M</td>
<td>Send messages to the TL1 device</td>
</tr>
<tr>
<td>Ctrl+Alt+B</td>
<td>Switch to Bulk Mode</td>
</tr>
<tr>
<td>Shortcut Key</td>
<td>Action</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>F3</td>
<td>Move to Tree/List mode</td>
</tr>
<tr>
<td>F4</td>
<td>Open Session Properties window</td>
</tr>
<tr>
<td>F9</td>
<td>Open Log Message Viewer window</td>
</tr>
<tr>
<td>Alt+F4</td>
<td>Closes the Craft Interface tool</td>
</tr>
<tr>
<td>F1</td>
<td>Opens Help Documentation in a Web browser.</td>
</tr>
</tbody>
</table>
General Options

Following are some of the general options available in the TL1 Craft Interface.

- Find option
- Look and Feel
- Generalize and Categorize option
- Expand and Collapse option
- Glossary View in Command Set
- Show Messages in the same window
- Clear Messages
- Conversion Tool
- Context-sensitive Help

Find Option

This option is invoked by selecting **Find** from the Edit menu or from the toolbar icon. It searches a particular child node under a selected node in the TL1 Message Tree easily.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find What</td>
<td>Uses this text in the search.</td>
</tr>
<tr>
<td>Find Next</td>
<td>Searches for the next text in the TL1 Message Tree.</td>
</tr>
<tr>
<td>Close</td>
<td>Ends the search operation and closes the dialog.</td>
</tr>
<tr>
<td>Match case</td>
<td>Searches for case-sensitive occurrences of a text. This limits the search and locates the text that matches the uppercase and lowercase characters you enter.</td>
</tr>
<tr>
<td>Match whole word</td>
<td>Search for occurrences of a text as whole words.</td>
</tr>
</tbody>
</table>

**Note**: The find option is supported only in the Tree mode.

Look and Feel

This option changes the look and feel of the Craft Interface. Three options are available, **Metal**, **Motif**, and **Native**. Any one of these options can be selected based on the platform. This menu can be hidden from the menu bar by disabling it in the **Tools -> Options Window**. The default UI Look and Feel used by the tool is Metal.

Generalize and Categorize Message Tree

This option is used to display the commands in the Message Tree either command-based or category-based. By default, all messages are listed command-wise. To display the commands category-wise, right-click on the Request node and select the **Categorize** button. Select the **Generalize** button to display them command-wise.
Expand and Collapse Option

This option is used to expand and collapse all the nodes in the TL1 Message Tree. To perform this function, select the option by doing a right-click on the root node of TL1 Message Tree.

Glossary View in Command Set

Glossary option in Command Set allows you to view description of a parameter, enumeration or range values. To use the glossary option, there must be an entry in the word list of that command set file. Also this entry must be associated with the parameter.

Assuming there is word list entry in the glossary for a set of parameters. Now, load the command set file and select the node in the command set which has a word list defined. The right panel displays a table as shown below. Click the "..." button present in the glossary column. The Glossary List panel appears which contains description about that parameter.

Show Messages in the Same Window

This option allows both Autonomous and Response messages to be shown in the same window. You can enable it by selecting it from the right-side panel of the main frame. By default, there will be two text areas; one for showing the response and acknowledgement messages and the other for showing autonomous messages. Selecting this option allows all the messages to be displayed in the same text area.

Clear Messages

This option can be used to clear all the messages displayed in the text area. It can be selected by clicking the Clear Text from the Edit menu. You can also clear the messages through the text area pop up menu.

Conversion Tool

This tool is used to convert the input commands specified in the data set (.dat) file to a textual form. The resultant text file will have the complete list of input commands which can be loaded in Craft Interface as a list. This tool can be opened by selecting the Data To Text Conversion button from the Tools menu.
Context-sensitive Help

This option is used to describe the functionality of a particular component in the Craft Interface. To use this option, select the context-sensitive button available in that dialog and place it on the component you wish to know.
Working with TL1 Craft Interface

Establishing a Connection with the TL1 Device

- Overview
- Transport Properties
  - TCP
  - Telnet
  - Serial
  - Plugging your own Transport Protocol
- Session Properties
- Log Properties
- How to Terminate the Connection

Overview

Establishing a connection with a TL1 device is the first step to manage any TL1 device. Once a connection is successfully established, you can send messages to the device and get responses. The TL1 Craft Interface is designed to handle multiple device connections at the same time. This is done by means of a Session Manager which manages all the TL1 connections. The tool is built over common Transport Provider Framework and supports multiple transport protocols such as TCP, Telnet, and Serial. You can also plug in your own transport protocol in the Craft Interface and send TL1 messages.

Note: The TL1 Agent Simulator supports only TCP protocol.

To establish a connection with the TL1 device, click the Open Session button from the Session menu or from the toolbar. The Open Session dialog is displayed as shown in the figure. Select the transport protocol such as TCP, Telnet, or Serial to communicate with the TL1 device and specify the connection properties. For example, you must specify the host and port in the case of TCP.
Transport Properties

By default, the TL1 Craft Interface supports transport protocols such as TCP, Telnet, and Serial. But you can also plug in your own transport protocol in the craft interface to work with. The transport-related options specific to each protocol is discussed here.

a) TCP

In this case, you need to specify two mandatory parameters, Host and Port of the TL1 device where the agent is running. The default values for these parameters are given in the table.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Name</td>
<td>localhost</td>
</tr>
<tr>
<td>Port Number</td>
<td>9099</td>
</tr>
</tbody>
</table>

b) Telnet

In the case of Telnet, there are various options such as Host, Port, Command Prompt, Login Prompt, User Name, Password Prompt, Password, and Time out. The mandatory parameters are Host, Port, and Command prompt. The default values for these parameters are given in the table.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Name</td>
<td>Null</td>
</tr>
<tr>
<td>Port</td>
<td>23</td>
</tr>
<tr>
<td>Command Prompt</td>
<td>$</td>
</tr>
<tr>
<td>Login Prompt</td>
<td>login:</td>
</tr>
<tr>
<td>User Name</td>
<td>Null</td>
</tr>
<tr>
<td>Password Prompt</td>
<td>Password:</td>
</tr>
<tr>
<td>Password</td>
<td>Null</td>
</tr>
<tr>
<td>Login Timeout</td>
<td>5000</td>
</tr>
</tbody>
</table>

c) Serial

In the case of Serial (RS232), the options are Port, Baud rate, Flow control, Data bits, Stop bits, and Parity. These parameters should be properly specified based on the settings configured on the TL1 device. The default values for these parameters are given in the table.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>COM1</td>
</tr>
<tr>
<td>Baud Rate</td>
<td>9600</td>
</tr>
<tr>
<td>Flow Control</td>
<td>None</td>
</tr>
<tr>
<td>Stop bits</td>
<td>1</td>
</tr>
<tr>
<td>Data bits</td>
<td>8</td>
</tr>
<tr>
<td>Parity</td>
<td>None</td>
</tr>
</tbody>
</table>
d) Plugging your own Transport Protocol

Apart from the Transport Protocols supported, users can plug in their own transport protocol. Follow the steps given below to add your transport protocol for TL1 communication.

1. Write your own transport protocol implementation that implements the TL1TransportProvider interface and set it in the CLASSPATH.
2. Select the user-defined option and type the full class path of your protocol implementation in the Transport Provider field. In the case of TCP, it is com.adventnet.tl1.transport.tcp.TcpTransportImpl.
3. Add all connection-related properties required for the transport protocol in the table. You can refer the default protocol implementations available in the <Simulator_Home>/default_impl directory.
4. Select the mode of communication. It can be either Normal or Craft.
5. Click the Open Session button to open a connection using the specified transport protocol.

Session Properties

The table below lists the common options which can be used across all the transport protocols.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode</td>
<td>This can be either Normal or Craft. In Normal mode, messages are buffered in a queue, then parsed and sent to the application. In Craft mode, messages are given to the application directly without any processing on the session.</td>
</tr>
<tr>
<td>Connection Handler</td>
<td>Represents the interface ConnectionHandler. Users can do some special tasks while opening and closing the connection (such as login and logout).</td>
</tr>
<tr>
<td>Message Parser</td>
<td>Represents the interface TL1Parser. This interface parses the TL1 messages and generate message objects. By default, com.adventnet.tl1.parser.TL1MessageParser is implemented. You can also plug in your own parser implementation.</td>
</tr>
<tr>
<td>Message Formatter</td>
<td>Represents the interface MessageFormatter. This interface allows you to filter or format specific TL1 messages at runtime. Users can plug-in their own filter / formatter by implementing this interface. By default, it is null.</td>
</tr>
<tr>
<td>Message Rationalizer</td>
<td>Represents the interface MessageRationalizer. This interface allows you to upgrade messages based on the XML message definitions (command set) or filter specific TL1 messages. Users can plug in their own rationalizer by implementing this interface. By default, it's value is null.</td>
</tr>
<tr>
<td>Partial Message Accumulation</td>
<td>Option to provide or accumulate partial messages received from the device. If set to true, all partial messages are accumulated and given only after receiving a response with proper terminator (;).</td>
</tr>
<tr>
<td>Prefix and Suffix</td>
<td>Append special strings at the start and end of each TL1 message. This is optional and may not be applicable for all devices. The default prefix is \r and the suffix is null.</td>
</tr>
</tbody>
</table>
Log Properties

The Log Properties is split into Raw Log and Information Log.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File Name</td>
<td>Represents the name of log file.</td>
</tr>
<tr>
<td>Pattern</td>
<td>Pattern indicates the order in which the log file name should exist.</td>
</tr>
<tr>
<td></td>
<td>%n represents the log file name.</td>
</tr>
<tr>
<td></td>
<td>%d represents the date at which the log file is generated.</td>
</tr>
<tr>
<td></td>
<td>%t represents the time at which the log file is generated.</td>
</tr>
<tr>
<td></td>
<td>%c represents the log file count generated for that particular connection.</td>
</tr>
<tr>
<td></td>
<td>The date and time are optional in pattern.</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Option to set the Timestamp in each log message. The time will be given in hh-mm-ss format.</td>
</tr>
<tr>
<td>Append File</td>
<td>Appends the log information to the same log file that was previously used by this connection.</td>
</tr>
<tr>
<td>Rollover Size</td>
<td>Maximum size of a log file after which a new log file will be created. The size should be given in bytes. You can give a value of 0 if you do not want log files to be created based on size.</td>
</tr>
<tr>
<td>Rollover Count</td>
<td>Maximum log file count for a particular device connection.</td>
</tr>
<tr>
<td>Rollover Time</td>
<td>Time that must be specified after which a new log file will be created. The time should be given in the hh-mm-ss format.</td>
</tr>
<tr>
<td>Rollover Interval</td>
<td>Interval that must be specified after which a new log file will be created. The interval should be given in milliseconds.</td>
</tr>
</tbody>
</table>

How to Terminate the Connection

To close an established connection, follow the steps given below.

1. Select the Close Session button from the Session menu or through the toolbar. The Close dialog is displayed. This dialog displays all the established sessions.
2. Select the sessions you wish to close and click the Close Selected Sessions button.
3. Use the Select All check box to close all the established sessions.
4. Click the Cancel button to cancel the close operation.
Managing Multiple Connections

One unique feature in the TL1 Craft Interface is that, it can interact with multiple TL1 devices simultaneously. This helps field engineers to manage multiple TL1 devices at the same time. The Craft Interface connections are managed by the Session Manager. These connections and their associated parameters can be viewed through the Session Properties window. The Session Properties window can be invoked by selecting the Properties button from the Session menu or through the toolbar.

The User Interface of the Session Properties window is split into two panels. The left panel displays all the established sessions and the right panel is to show the properties pertaining to each connection. You can select a particular session and update their properties.

The image below shows the Session Properties window with default parameters.

Note: The Session Properties window can be viewed only if there is at least one established connection.
Sending and Receiving Messages

After successfully establishing a connection with the TL1 device, you can query by sending commands. The commands can be sent from the Craft Interface in one of the following ways:

- **Using List**
- **Using Command Set**
- **Using Data Set**

**Using List**

This is the crude and easy way of sending commands to the device and get responses. In the list mode, you can either type a command manually in the text field shown in the right panel or load commands from a text file. Click the **Send** button to send the commands. Alternatively, you can also double-click on the selected command in the list or use the **Enter** button to send the command. The responses will be displayed in the text area of the right panel.

In this mode, commands can be sent one at a time, or multiple commands can be selected to be sent as bulk. The figure below shows how to perform the query operation one at a time.

To send one command at a time:

1. Select the command from the list.
2. Edit the command (if you wish) in the right panel.
3. Click the **Send** button to send the command.

The responses can be viewed in the right panel.

The bulk mode of sending commands is discussed in the next section.
Using Command Set

You can also send commands to the device by loading the command set file. Follow the steps mentioned here to send commands using the command set file:

1. Select the command from the message tree.
2. Edit the command (if you wish) in the right panel. The Craft Interface splits the command into parameter blocks and each block can be edited separately. If default values for the parameter is specified in the command set file, they are displayed in the combo box. You can simply select the value from the combo box or edit these values (if necessary) in the input message field.
3. Click the **Send** button to send the command. The responses can be viewed in the right panel.

The figure below shows one such command sent by selecting it from the TL1 Message Tree.

![Command Send Example](image)

Using Data Set

The data set way of sending commands to the device is same as command set. The only difference is that, the message cannot be edited field. Follow the steps mentioned here to send commands from the data set:

1. Select a input message data from the message tree.
2. Edit the command (if you wish) from the input message field in right panel.
3. Click the **Send** button to send the command. The responses can be viewed in the right panel.
The figure below shows one such command sent by selecting it from the TL1 Message Tree.
Sending and Receiving Bulk Messages

In TL1 Craft Interface, one or more messages can be selected to be sent at a time. This section explains how to perform this function from the Craft Interface. At present, the bulk mode is supported only in Data Set and Text file.

To send multiple commands at the same time:
1. Load a Data Set or Text file.
2. Select commands from the data set or list with Ctrl button pressed. All selected commands will be displayed in the right panel under List of all Messages.
3. Select the commands in the order they have to be sent and press the >> button. These commands will be put in the send list.
4. Use the << button to remove selected commands from the send list.
5. Choose the mode of transmission (Synchronous/Asynchronous).
6. Click the Start button or Send Bulk Messages button to send the commands. The responses can be viewed in the right panel.
7. Use the Pause button to pause the send operation. You can click the Resume button to resume the operation again.
8. Use the Stop button to stop the send operation.
Configuring TL1 Craft Interface

- Session
- Message
- User Interface
- Log File

When a TL1 Craft Interface shows up, it always gets loaded with the default properties configured. These default properties can be changed based on your requirements. The Options Window allows you to do this. You can open this window by selecting the Options button from the Tools menu.

The Options Window is split into five categories: Session, Message, User Interface, and Log File. Each option has its own default properties. The default properties are stored into `tl1craftinterface.config` file present under `<Simulator_Home>/conf` directory.

After editing the default properties, click the OK or Apply button to save the changes. Click the Cancel button to discard the changes.

**Session Option**

The Session option will let you modify the parameters pertaining to the TL1 Session, such as Transport, Mode, Keep Alive, Connection Handler, etc.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode</td>
<td><strong>Normal</strong> or <strong>Craft</strong>. In Normal mode, messages are processed, queued up and sent to the application. But in Craft mode, messages are given without any processing.</td>
</tr>
<tr>
<td>Transport</td>
<td>Represents the connection parameters for each transport (i.e. TCP, Telnet, and Serial).</td>
</tr>
<tr>
<td>Connection Handler</td>
<td>Represents the interface <strong>ConnectionHandler</strong>. Users can perform any device-specific tasks by plugging in their own custom class that implements this interface. This interface is invoked while opening and closing the connection.</td>
</tr>
<tr>
<td>Keep Alive</td>
<td>This option if enabled helps you to maintain connection with the device by sending messages periodically. You can set the TL1 message and the time interval.</td>
</tr>
</tbody>
</table>

**Message Option**

The Message option will let you modify the default properties related to TL1 Message, such as Parser, Formatter, Rationalizer, Prefix and Suffix of each message, Timeout and Retries, etc. The default properties are shown in the image.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parser</td>
<td>Represents the interface <code>TL1Parser</code>. This interface parses the TL1 messages and generates message objects. By default, <code>com.adventnet.tl1.parser.TL1MessageParser</code> is implemented. You can also plug in your own parser implementation.</td>
</tr>
<tr>
<td>Formatter</td>
<td>Represents the interface <code>MessageFormatter</code>. This interface allows users to format specific TL1 messages. Users can plug in their own formatter by implementing this interface. By default, it's value is <code>null</code>.</td>
</tr>
<tr>
<td>Rationalizer</td>
<td>Represents the interface <code>MessageRationalizer</code>. This interface allows users to upgrade messages based on the XML message definitions (command set) or filter specific TL1 messages. Users can plug in their own rationalizer by implementing this interface. By default, it's value is <code>null</code>.</td>
</tr>
<tr>
<td>Prefix and Suffix</td>
<td>Append special strings at the start and end of each TL1 message. This is optional and may not be applicable for all devices. The default prefix is <code>\r</code> and the suffix is <code>null</code>.</td>
</tr>
<tr>
<td>Timeout</td>
<td>Represents the timeout value of each request. By default, the timeout is <code>5000</code>.</td>
</tr>
<tr>
<td>Retries</td>
<td>Represents the retries value that session has to do for each request. By default, the retries is <code>0</code>.</td>
</tr>
<tr>
<td>Partial Message Accumulation</td>
<td>Option to provide or accumulate partial messages received from the device. If set to true, all partial messages are accumulated and given only after receiving a response with proper terminator (<code>;</code>).</td>
</tr>
</tbody>
</table>

**Note:** The special characters, `\r`, `\n`, `\f`, `\b`, `\t` are supported for prefix and suffix fields. Other characters are interpreted as specified.

You can change the default properties by editing the appropriate fields.
User Interface Option

The User Interface option will let you modify the default properties of the Craft Interface, such as color, look and feel, etc. You can change the default properties by selecting the appropriate fields. You can also modify the default location of the files used by the Craft Interface, such as Help Documentation and Data files. You can modify the existing location by selecting the ... button.
Log File option

The Log File option will let you modify the default log configuration, such as File Name pattern, Time Stamp enable, Rollover details, etc.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Pattern</strong></td>
<td>Pattern indicates the order in which the log file name must exist.</td>
</tr>
<tr>
<td></td>
<td>%n represents the log file name.</td>
</tr>
<tr>
<td></td>
<td>%d represents the date at which the log file is generated.</td>
</tr>
<tr>
<td></td>
<td>%t represents the time at which the log file is generated.</td>
</tr>
<tr>
<td></td>
<td>%c represents the log file count generated for that particular connection.</td>
</tr>
<tr>
<td></td>
<td>The date and time are optional in pattern.</td>
</tr>
<tr>
<td><strong>Time Stamp</strong></td>
<td>Represents the time at which the log record is created.</td>
</tr>
<tr>
<td><strong>Append</strong></td>
<td>Appends the log information to the same log file that was previously used</td>
</tr>
<tr>
<td></td>
<td>by this connection.</td>
</tr>
<tr>
<td><strong>Rollover Size</strong></td>
<td>Maximum size of a log file after which a new log file will be created.</td>
</tr>
<tr>
<td></td>
<td>You can give a value of 0 if you do not want log files to be created based on size. The size should be given in bytes.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Rollover Time</td>
<td>Time that must be specified after which a new log file will be created. The</td>
</tr>
<tr>
<td></td>
<td>time should be given in the hh-mm-ss format.</td>
</tr>
<tr>
<td>Rollover Count</td>
<td>Maximum number of log files to be created for a specific device connection.</td>
</tr>
<tr>
<td>Rollover Interval</td>
<td>Interval that must be specified after which a new log file will be created.</td>
</tr>
<tr>
<td></td>
<td>The interval should be given in milliseconds.</td>
</tr>
</tbody>
</table>
Viewing Log Messages

TL1 Craft Interface provides an option to view the log files generated for each device connection from the Log Message Viewer. The log files are created in two formats, Information log and Raw log. The information log contains textual description of each log record and the raw log contains the actual messages exchanged through the session. The Log Message Viewer has the facility to view the log files of each TL1 Session.

The Log Message Viewer can be invoked by selecting the Logs button from the Session menu. The User Interface of Log Message Viewer is split into two panels. The left panel has the list of sessions and the right panel contains the log message details.

**Note:** The Log Message window can be viewed only if there is at least one established connection.

Raw Log

To open a raw log,
- Select the session name from the left panel.
- Click the Raw Log Messages tab.

Now you can view all the messages exchanged through this session.
Information Log

To open an information log,

- Select the session name from the left panel.
- Click the Information Log Messages tab. This panel contains a Log Name table.
- Select the Log Name and the details regarding each log record will be displayed in the Log details pane beneath the Log Name table.

Note: Although multiple log files are created for each individual connection based on the Rollover criteria, you can only view the log file that is currently in use.
Files Used by TL1 Craft Interface

Following files are used by the TL1 Craft Interface for performing various functions.

1) Command Set file (.tcs)

The Command Set file contains TL1 command definitions in XML format. Apart from command definitions, it also contains response message and autonomous message details specific to each TL1 device. You can use the TL1 Message Builder tool to create these message definitions. Please refer to TL1 Message Builder section for more details on the structure of the Command Set file.

2) Data Set file (.dat)

The Data Set file contains values for each command defined in the Command Set in XML format. You can use the TL1 Message Builder tool to create data set values. Please refer to the TL1 Message Builder section for more details on the structure of the Data Set file.

3) Text file (.txt)

The text file contains a list of input messages which can be loaded in the list. The input messages must be defined in separate lines.

4) Configuration file (tl1craftinterface.config)

The configuration file is used to store the default settings of the Craft Interface. You can use the Options Window in the Craft Interface to modify the default settings. The default settings include Session, Message, GUI, and Log options. Please refer to the Configuring the TL1 Craft Interface section for more details.

5) Properties file (tl1craftinterface_i18n.properties)

This file can be used as a reference for creating your own locale-specific properties file for internationalizing the Craft Interface tool. The locale-specific properties can be edited using the I18N Editor tool.

6) Context-sensitive help file (tl1craftinterface_csh.xml)

This file contains short description about each component used in the Craft Interface UI. When the context-sensitive icon is placed on a particular component, the description about that component is displayed in a small window.
FAQs

This section contains common questions and doubts raised while using the different modules of Simulation Toolkit. The FAQs are grouped module wise for easy reference.

- General
- SNMP Agent Simulator
- TL1 Agent Simulator
- Network Simulator
- IOS Simulator
FAQs - General

• Database
• Others

Database
1. What is the user name and password for MySQL database?
2. I tried to invoke the mysql server (bundled with simulator toolkit) in Windows OS, via the mysqld.exe command but wasn't successful. Was I doing something wrong?
3. Is it possible to run the MySQL server on a separate machine and provide the MySQL server name to the simulator running in another system? How do I setup the MySQL server on the separate machine?
4. How do I turn off the automatic startup of MySQL server on my simulator machine?
5. I am running MySQL in Windows as a service. How can I start MySQL bundled with the product?
6. When I want to save any configuration, it is not possible to change the displayed Database Name. This results in storage of all files (*.MYD, *.MYI, *.frm) for several configurations in the same directory. Why?

1. What is the user name and password for MySQL database?

MySQL database that is bundled along with product has root as the User Name and there is no password set, the field goes blank. The user also has the option of setting the password later.

2. I tried to invoke the mysql server (bundled with simulator toolkit) in Windows OS, via the mysqld.exe command but wasn't successful. Was I doing something wrong?

• When you start the Network Designer/SNMP Agent Simulator/TL1 Agent Simulator tool in Windows OS, it will automatically start the MYSQL daemon if there is no MYSQL running already. By default, simulation toolkit will start the mysql server at 2306.
• To verify if there is any other MYSQL running, please use the Windows Task Manager to check for the mysqld process. If there is any MYSQL already running before you start Simulation tools, then you can kill that mysql process. If it says , access denied, then the MYSQL is running as a service and you can stop that MYSQL service through Control Panel ->Services.
• Also there is a file called my.ini under the C:\WINNT In case the Mysql runs as service, the mysql database will look for the my.ini file in the C:\winnt/windows installation directory for the start up parameters of the mysql.
• If you want to use the mysql bundled with the Simulation Toolkit you have to delete the my.ini file after uninstalling the mysql service
• Now you can start the Network Design Tool/ SNMP Agent Simulator/TL1 Agent Simulator and it will start the MYSQL.
• If you see that MYSQL still does not start, then you can manually start the MySQL from DOS prompt with the following command in the Windows 2000 machine:
• Please execute this command from the \bin\mysqld --basedir . --standalone
• This will start the mysqld at the port 2306 with default username root. There is no password. Now, you can start the NDT or SNMPAgentSimulator.

3. Is it possible to run the MySQL server on a separate machine and provide the MySQL server name to the simulator running in another system? How do I setup the MySQL server on the separate machine?

Yes, it is possible to run the MySQL server on a separate machine and the MySQL server name can be provided to the simulator. MySQL version 4.0.20 is bundled along with AdventNet Simulation Toolkit. The default user name is root with no password.
1. To setup the MySQL server in Windows OS:
   1. Create a zip of the <SimulatorHome>/mysql directory.
   2. Extract it into the system where MySQL is to be installed. For example, C:\mysql. You'll have to use a program such as WinZip to do this.
   3. Now you're ready to start the server. For all versions, you can start the server. To do this, C:\mysql> bin\mysqld -basedir --standalone
   4. Make sure there is no other mysql running on the same machine.

2. To setup the MySQL server on a separate machine (Linux OS):
   1. Create a zip of the <SimulatorHome>/mysql directory.
   2. Extract the zip into the system where MySQL is installed. For example, /advent/stk5_mysql.
   3. Now you're ready to start the server. For all versions, you can start the server. To do this, $STK_HOME/mysql/bin/mysqld -u root -P 2306 --basedir=$STK_HOME/mysql
   4. Make sure there is no other mysql running on the same machine.

3. To connect to MySQL running in a separate machine:

   To start the network and connect to the MySQL running in a separate machine, follow the steps given below.

   The Network Designer stores the simulated data in MySQL database. The MySQL database is bundled along with the product and is part of the package. By default, the following details will be stored in database_params.conf file in <SimulatorHome>/conf directory:

   - User Name: root
   - Password: No password is configured. Leave the field blank
   - Server Name: Specify the machine name where your mysql is running
   - Server Port: 2306
   - MySQL Home: ./mysql/bin (Specify the mysql home. In the case of Linux, the MySQL Home need not be considered.)

   When the simulator is started for the first time, the database parameters is read from the database_params.conf file and updated in NDTStartUp.prop file in <SimulatorHome>/conf/settings directory. The next time when the simulator is started, the database settings are read from NDTStartUp.prop file. Do the required changes to this file and then start the Network Designer.

4. How do I turn off the automatic startup of MySQL server on my simulator machine?

   In case of Windows/Linux/Solaris, the MySQL server bundled with the product will be started automatically when the product is started and will be stopped when you exit from the product. To stop the MySQL server explicitly, you can use the StopMySQL.bat/sh file available in <SimulatorHome>/bin directory.

5. I am running MySQL in Windows as a service. How can I start MySQL bundled with the product?

   If you have your MySQL already installed as a service, then the Simulation Toolkit will always connect to service. To connect to the MySQL bundled with the product follow the steps given below:

   2. Open the my.ini file present under c:\WINNT and change the data directory to the data directory present under the <simulator home> \ <mysql>.
   3. Restart the Service. This will use the MySQL bundled with the product.
6. When I want to save any configuration, it is not possible to change the displayed Database Name. This results in storage of all files (*.MYD , *.MYI , *.frm) for several configurations in the same directory. Why?

As per our implementation we use the simulatordb for all the configuration done in the SNMP Agent Simulator and the TL1 Agent Simulator. The device configurations added to the device tree in Network Designer, will be stored in the devicedb. To have more control over our simulated configurations, the field database is made non-editable by default.

Others

1. Will Simulation Toolkit work with any management applications?
2. Can a simulated agent configuration be copied from one platform to another?
3. Can I manage one or more simulated agents with any NMS with Performance, Configuration, Fault Management?
4. How many agents can be run simultaneously on one computer?
5. Can I control the Simulator via RMI?
6. How many NMS systems can refer to simulator simultaneously and what is the performance (SNMP requests/sec) in this case?
7. Is it possible to create two different agents with two different community names, but with the same IP address?
8. How to start the simulator, if Simulation Toolkit is installed in a system with multiple NIC (Network Interface cards)?
9. How easy it is to define 10000 agents from 10 different kinds of devices?
10. Do you know if its possible to start multiple simulators on one box and assign each to a virtual IP? Can I use SNMP Agent Simulator for this purpose?
11. How do I change the JAVA_HOME variable settings if I install a different JRE version?

1. Will Simulation Toolkit work with any management applications?

Yes, Simulation Toolkit will work with any management application which supports SNMP and TL1 protocol.

2. Can a simulated network configuration be copied from one platform to another?

Yes, you can use the Package and Install option in the Network Designer, to port the network configurations to the desired location. Refer to the topic, Porting your Network to learn more.

3. Can I manage one or more simulated agents with any NMS with Performance, Configuration, Fault Management?

You will be able to use it to demonstrate Fault Management, Configuration and Performance Management.

Fault Management: The trap configuration in SNMP and the Autonomous message configuration for the TL1 can be fired to the manager depending on various criteria like request based, threshold based, etc., and this will assist you in using the network simulator for the Fault management of NMS.

Configuration: The network simulator will allow the manager application to change its mib variables. Since this is the prerequisite of the Configuration management, The simulator can be used for configuration. You can use the simulator for the discovery, performance management, Status Polling from the manager Application.
4. How many agents can be run simultaneously on one computer?

You will be able to simulate a maximum of 10000 agents in Windows, 50000+ agents in Linux and 4096 in Solaris.

5. Can I control the simulator via RMI?

RMI (Remote Method Invocation) is implemented in the network simulator and the agent simulators, to control the simulated agent and network behavior from a remote client program.

The SNMP and TL1 RMI client APIs, defines the methods in the simulator, which can be accessed via RMI from the client program. The methods can be used to programmatically update values for a specific SNMP/TL1 device within the simulated network, start/stop individual SNMP/TL1 devices in the network, send SNMP v1/v2/v3 traps and TL1 autonomous messages, etc.,

6. How many NMS systems can refer to the simulator simultaneously?

There is no limitation on the number of NMS systems querying the simulator simultaneously.

7. Is it possible to simulate two different agents with two different community names, but with the same IP address?

Yes, the SNMP Proxy Agent Simulation can be used to start two different agents with two different community names but same IP address. Refer to the topic SNMP Proxy Agent Simulation, to learn more.

8. How to start the simulator, if Simulation Toolkit is installed in a system with multiple NIC (Network Interface cards)?

If your system has multiple interface cards installed, the following configurations must be made before starting the simulator.

For Windows

1. The order in which the Interface must be used is specified in the **Network and Dial-up Connections** folder. Select this folder.
2. Go to **Advanced Settings** option in **Advanced** menu and change the order of priority.

For Linux / Solaris

1. Use the command `/sbin/ifconfig` to check the valid interface name configured in your system.
2. Edit CreateVIP.sh file in `<SimulatorHome>/bin/cmdline directory with this interface name. For example, if the interface name is eth0, edit the file as follows: `/sbin/ifconfig eth0:$count $ip netmask 255.255.255.0 up`
3. Now, run the CreateVIP.sh file to create the virtual IP addresses manually before starting the network.

9. How easy it is to define 10000 agents from 10 different kinds of devices?

In the Network Design Tool, you have to first add your ten different kind of devices to the device tree. Then you can add the devices in bulk into the network using the **Random Device Addition** wizard.

10. Do you know if its possible to start multiple simulators on one box and assign each to a virtual IP? Can I use SNMP Agent Simulator for this purpose?

SNMP Agent Simulator is used to simulate only one SNMP Agent. For simulating a network (multiple SNMP agents) you can use Network Designer Tool bundled with AdventNet Simulation ToolKit.
Network Designer Tool is an easy to use GUI tool to simulate a network with multiple SNMP and TL1 agents. Each SNMP agent will be assigned a separate virtual IP address and port number combination.

You can refer to the section "Network Simulator Guide" in our Help Documentation to know more about simulating multiple SNMP agents.

11. How do I change the JAVA_HOME variable settings if I install a different JRE version?

By default, the JAVA_HOME variable is set to the default jre version (jre 1.4.2) bundled with the product. To modify the JAVA_HOME variable settings, do the following:

If you are starting the tools from the launcher by invoking `<Simulator Home>/SimulationToolkit.bat` file:
- Modify the JAVA_HOME variable in SimulationToolkit.bat file.
- Select the Options -> Settings menu in the launcher. This pops up the Settings dialog. Set the JAVA_HOME variable in this dialog. This is used to set the global settings which will be used throughout the AdventNet Simulation Toolkit package.

If you are starting the tool from the `<Simulator Home>/bin` directory by invoking the .bat/sh file:
- Modify the JAVA_HOME variable in `SetEnv.bat`/`SetEnv.sh` file present inside the `<Simulator Home>/bin` directory.

**Note:** The JDK version used by the product needs to be changed only if you face any installation problems with the default version. Otherwise, it is recommended to use the default jre version bundled with the product.
FAQs - SNMP Agent Simulator

Traps

1. Why do you associate traps with a variable?

In the case of Request and Threshold based Traps, the traps will be triggered based on the SNMP request on a OID. For this reason, we associate the trap to a OID on which the SNMP request will be done to trigger the trap.

2. Is there any way to "copy" Trap simulation entries? For all trap simulations I'm working on, they include 15 varbind entries and I was hoping there is a way to copy those such that I don't need to enter them for each trap?

Copying the same varbinds for different traps, is currently not supported.

3. How does the trap configuration table get populated?

The MIBs will have the trap variables defined in it. The trap variable will contain one or more varbinding OIDs associated with it. When the MIB is simulated, Request Based traps will be generated keeping each varbinding OID as associated variable. If the trap variable has three varbinding oid, three traps will be simulated for that trap variable in the trap table.

4. Is it possible to send traps by setting "fake" IP address for the traps?

By default, the traps will be sent with the IP Address of the system from which the traps are generated. The SNMP Agent Simulator provides you the option to configure the IP address or host name (fake IP Address), with which the traps are to be sent. To simulate a 'fake' IP Address, all you need to do is configure the 'fake' IP Address using Settings --> Global Properties Menu option. Specify the 'fake' IP Address in "From IP Address".

For more information, refer the help documentation section : Simulating an SNMP Agent --> Configuring Traps.
5. I want to create a lot of identical agents with the same set of traps. Is there any way to facilitate this work?

The steps below will help you to create same set of traps for all agents:

- Add the device that you want to simulate, in the device tree.
- Configure Traps in that device.
- Choose the Network Design Wizard option from the "Tools Menu".
- In the screen that pops up, choose the device type as the device that you simulated now and Click Next.
- In the next screen, specify the starting and ending IP Address.
- Click Finish.

Now you will have the set of nodes in the network with same set of traps configur.

6. How do I send the current value of sysUpTime with a trap?

While configuring trap varbinds, you can configure the varbind value as sysUpTime. Refer to "Configuring Traps -> Adding Variable bindings".

7. Is there any way of setting up more than one agent to send traps at any one time?

You can send traps from multiple agents at the same time by configuring behavior scripts. The javadocs of the script APIs are available in <SimulatorHome>/examples/javadocs directory. The sendTrap() can be used to send traps from multiple agents with the specified ipaddress and port number.

**Behavior Simulation**

1. I have defined the script file within the behavior configuration. How can I check if the script is started?
2. Do I need to define a condition and an action for my script? Is it obligatory?
3. My objective is to have the values increase in a certain rate for every pre defined interval. How do I do this?
4. Is it possible to import JPython scripts into another JPython script and use the "Interface Class ScriptInterface"?
5. Can I use the java.lang.Runtime class to execute the system command in the Jython scripts with Simulator?
6. Agent Simulator can execute Jython scripts when it receive the request. Can it execute the tftp command in the Jython scripts when it receive the SNMP Set request?
7. Do you provide any java interface in the simulator where I can plugin my java code?
8. Is it possible to invoke the script API methods independently, i.e not from the behavior Simulation?

1. I have defined the script file within the behavior configuration. How can I check if the script is started?

All the debug prints are redirected to the snmpagentsimulator.out and err files. When the script executes, you can confirm it with the following debug print in the snmpagentsimulator.out file in the /logs directory.

"Debug Code at Py Wrap executeScripts 77
The script is executed using the py Object with id 0"

You can also put print statements inside the Script files.
2. Do I need to define a condition and an action for my script? Is it obligatory?

The conditions and actions are defined to have a clear picture of what each and every function of the jython script is intended for. This is followed in the simulator generated script files. It is not mandatory to follow this procedure in the user defined scripts. If there are no conditions and actions, then no action will take place, can be added.

3. My objective is to have the values increase in a certain rate for every pre-defined interval. How do I do this?

You can use scripting to achieve your objective. That is, you can configure a timer-based script which will execute with some predefined time interval (say for example, every 10 secs). Now inside the script you can use the following script API to achieve your objective: getNodeValue(), updateValue().

The getNodeValue() will get the corresponding node's value and using updateValue(), you can update a new value to that node. Now this script will be executed for every 10 seconds. Hence you will be able to increase your value at a certain rate for every predefined interval.

For more information on Scripting, please follow the below link in the help documentation: Simulating an SNMP Agent --> Simulating Behavior in devices.

4. Is it possible to import JPython scripts into another JPython script and use the "Interface Class ScriptInterface"?

It is possible to import JPython scripts into another JPython script. For example, assume you have written a JPython script, namely Example1.py, with some methods namely method1(), method2().

Now if you want to call these methods from some other JPython script namely, Example2.py, then you have to import the methods of the Example1.py file, in Example2.py file. The following line will import the methods used in the Example1.py.

```
from Example1 import method1, method2
```

To import all the methods in the Example1.py, from Example1 import *. But it is not possible to execute such scripts from the Simulator.

5. Can I use the java.lang.Runtime class to execute the system command in the Jython scripts with Simulator?

Yes, we can use Runtime class to execute System commands in Jython scripts. The following code snippet first imports the Runtime Object and then executes the command.

```
from java.lang import Runtime
runtime=Runtime.getRuntime()
runtime.exec(""")
```

6. Agent Simulator can execute Jython scripts when it receive the request. Can it execute the tftp command in the Jython scripts when it receive the SNMP Set request?

Any type of java API including the Runtime class can be imported and executed inside the jython scripts. If your requirement is to execute a command without expecting the execution status of the command, then you can use the Runtime to execute the Operating systems Tftp functions. If you have java based Tftp implementations then you can use those APIs to execute Tftp commands.

7. Do you provide any java interface in the simulator where I can plugin my java code?

The jython script integrated with the Simulation Toolkit has the capability to import any java class file into the script. So any java file can be interfaced to the Simulator through the jython scripts.
8. Is it possible to invoke the script API methods independently, i.e not from the behavior Simulation?

All the java objects used in the processing of the script API Methods are created inside the Simulators. So it is not possible to invoke the script API without the Simulator Running.

1. In the Network Design Tool, if I set up a network with three virtual devices, does all of them has to be controlled by only one JPython script? Or can I use different scripts to control each virtual device?

Network Design Tool has both the features. A single script can be configured for all the devices or separate scripts can be configured for each device. Refer to the section Configuring Devices in the Network and Bulk Addition of Scripts, to learn more.

**Table Configuration**

1. How do I configure values in the Table from the Agent Simulator UI?

2. I am trying to populate some SNMP tables with sample data. I realize that the simulator will create some default data for tables, but it only creates one row. Can rows be created using Agent Simulator in tables that do not have 'row status' or are 'non-accessible'?

1. How do I configure values in the MIB Table from the Agent Simulator UI?

If you want to configure values in a table, please follow the below procedure:

1) Select the Table Module (for example, IfTable).
2) On the right side, you will be able to see a button "Configure Table".
3) Clicking on that will help you in simulating values for all the nodes in that table.

For more information on populating/simulating values in tables, refer to Populating the SNMP table.

3. I am trying to populate some SNMP tables with sample data. I realize that the simulator will create some default data for tables, but it only creates one row. Can rows be created using Agent Simulator in tables that do not have 'row status' or are 'non-accessible'?

You can add rows in the Agent Simulator for tables with or without "row status" option. As specified in the documentation, adding, modifying or deleting rows in the table can be done using the "Configure Table" button. This button will be enabled only when you have simulated values for the MIB and when the Table Node is selected in the MIB.

**SNMPv3**

1. What are the security levels of SNMPv3?

2. How to simulate a V3 agent?

3. What are the third party packages available that can be used to download and test the v3agent with privacy support?

4. I have simulated and started a SNMPv3 agent. Now, I want to add a new user and provide them views for the MIBs that I have in the agent. What do I do?

5. I created a new V3agent.ser file. But I was unable to see all the users configured?

6. What should I set in the "Engine Id" of the AgentSimulator and "ContextEngineID" of the MibBrowser for communicating with the v3 agent?

1. What are the security levels of SNMPv3?

SNMPv3 provides the following three levels of security:

- No Authentication and No privacy (noAuthNoPriv)
• Authentication but no Privacy (authNoPriv)
• Authentication and Privacy (authPriv)

A request can be made on behalf of a user with any one of the above security levels.

2. How to simulate a V3 agent?

To simulate a V3 agent please refer the section Simulating a v3 Agent

Trouble shooting:
1. Unable to Encode PDU: You will get this error if you had not set the privacy jars in the classpath. Please check all the privacy jar settings.
2. Time Sync Error: You will get this error if you have made any mistakes in the entries. Once the connection is refused. You have to stop and start the agent and also delete the entry from the MIB Browser and add it again.
3. General Failure: This is a common error. When you forget to provide the Context name or provide a wrong entry.

3. What are the third party packages available that can be used to download and test the v3agent with privacy support?

Please refer the topic, Privacy Support to know about the third party packages that can be used to provide privacy support.

4. I have simulated and started a SNMP v3 agent. Now, I want to add a new user and provide them views for the MIBs that I have in the agent. What do I do?

To add a new user to the v3 agent, you have to configure the new user information in the USM and VACM tables. The SNMPv3 Config tool bundled with the product is used to add new entries to the USM and VACM Tables. Please refer the section Simulating a v3 Agent to know more.

5. I created a new V3agent.ser file. But I was unable to see all the users configured?

When the SNMPV3 agent is started it reads the snmpv3.ser file. All the changes and modifications should be saved to this file. Only then you will be able to view the changes.

6. What should I set in the "Engine Id" of the AgentSimulator and "ContextEngineID" of the MibBrowser for communicating with the v3 agent?

Engine ID is nothing but a unique ID for the v3 agent. When you are adding a user to the configuration table in MibBrowser then the first request will be sent for discovery packet i.e if the agent is running in the specified port then the agent response by filling the engine ID. Then there after the communication between the browser and the v3 agent will take place with reference to the EngineID.

So there is no relation between the Engine ID specified in the Agent Simulator and the Context Engine ID specified in the MibBrowser.

General
1. I have all the configurations in SNMP Agent Simulator as database format in my local machine, how do I transfer those configuration to the other machines so that the agent simulator on those machines can also load my configuration?
2. Can I to simulate an agent running in a router with many ports? If I can, how do I do it?
3. What is the maximum SNMP packet size that the simulator supports?
4. Our devices have a lot of counters which is represented in MIB as read-only parameters. How can I simulate the multiple changes of different counters in simulated agent?
5. **Is it possible to configure an SNMP agent to send garbled/invalid responses?** Again, the intent is to test the SNMP manager's robustness in the face of receiving bad packets.

6. **Is it possible to configure one or more of the SNMP agents so that they sometimes fail to respond to queries?** (I want to test my SNMP manager's retry mechanism; so being able to predictably force retries to occur by having the agent drop every other packet, for instance, would be most useful).

7. **Is it possible to simulate error conditions using SNMP Agent Simulator?**

1. **I have all the configurations in SNMP Agent Simulator as database format in my local machine, how do I transfer those configuration to the other machines so that the agent simulator on those machine can also load my configuration?**

   All the tables corresponding to the configuration will be stored under data directory present inside "\mysql\" directory. You can zip all the files corresponding to the configurations (the directory name will be the database name you had given while saving the configuration) and transfer those files to the remote host wherever you want to share your configurations.

2. **Can I to simulate an agent running in a router with many ports? If I can, how do I do it?**

   Yes, you can simulate an agent running in a Router. Load the RFC1213 mib and the corresponding Router Mibs in the Agent Simulator. The simulator simulates the default values for all the nodes in the loaded mibs.

   For the Router agent, the following values has to be configured in the loaded RFC1213 mib:

   - The interfaces table (ifTable) should have at least two rows. By default the 'ifTable' will have only one row. So you have to add rows in the 'ifTable'.
   - The 'ipForwarding' scalar node under the 'ip' group should have value as 1.

   Start the Agent Simulator. Now the simulator will act as a router agent. You can query this agent from your manager application.

3. **What is the maximum SNMP packet size that the simulator supports?**

   The maximum SNMP packet size supported in SNMP Agent Simulator is 64 KB and in Network Designer is 20000 bytes.

4. **Our devices have a lot of counters which is represented in MIB as read-only parameters. How can I simulate the multiple changes of different counters in simulated agent?**

   You can configure simulation types for any type of node. The simulation type that are supported are: Constant, Linear, Wave, Sine Wave, Exponential, Random. For example, if you configure random as the simulation type for your counter node, then the response that you get for that node will be random.

   For more information on configuring simulation types, refer to Configuring MIB Values. Other than the simulation type, we also support scripting. Scripting will be very useful in cases where you want to make the agents behave according to your known needs. For more information on scripting, refer to Configuring behavior.

5. **Is it possible to configure an SNMP agent to send garbled/invalid responses?** Again, the intent is to test the SNMP manager's robustness in the face of receiving bad packets.

   Yes, it is possible to configure an SNMP agent to send garbled/invalid responses. The SNMP Agent simulator supports customizing the SNMP request/response PDU by accessing and modifying the PDU content in the buffer, before sending it to the socket layer. By customizing the SNMP PDU to send garbled/invalid responses, you can test the SNMP managers robustness in the event of receiving bad packets.

   Refer to **Customizing the SNMP PDU**, to know more.
6. Is it possible to configure one or more of the SNMP agents so that they sometimes fail to respond to queries? (I want to test my SNMP manager's retry mechanism; so being able to predictably force retries to occur by having the agent drop every other packet, for instance, would be most useful).

We do not have a direct implementation of dropping request packets received from the manager. But this scenario can be very well achieved by creating a delay in response using the behavior simulation.

For example, if we want to drop alternative packet from the manager, we can delay every alternative request from the manager more than the retry time or the time out time so that the manager sends the second packet before it receives the response for the request.

7. Is it possible to simulate error conditions using SNMP Agent Simulator?

We are supporting 'Error Simulation', by which you can send errors for the requests received. The errors can be thrown for all the three type of requests Get, GetNext and Set. Refer to, Configuring Error Conditions to learn more.
FAQs - Network Simulator

- **General**
- **Configuring Virtual IP Address**

For FAQs on Traps, Behavior, SNMPv3, Table Configuration refer [SNMP Agent Simulator](#).

### General

1. In the Network Design Tool, if I set up a network with three virtual devices, does each need to have the same configuration?

2. How can I add devices to the existing network. I think the Network Design Tool wizard adds devices only to the new network?

3. I created one Nework (network01) and added 10 Cisco Router SnmpNodes. I set virtual ipAddresses (192.168.12.1 ~ 192.168.12.10) to SnmpNodes. I added the ipAddresses in the Netowrk properties of Window 2000 and started the Network Simulation. The WebNMS discovers just one device. Could you check this problem?

4. I have simulated an agent with Read community = 'public' and Write community = 'private'. When I opened my NMS application with community 'private', which is defined as Read/Write, the simulated agents were not discovered. When I opened my NMS application with community 'public', which is defined as Read Only, all agents were discovered successfully. In addition the 'Community' = 'private' in MIB Browser does not allow to Get the parameters' values.

5. I have the real Agent with our own configuration, is it possible to simulate this agent and start it as multiple instances?

6. I am unable to access the virtual interfaces added in the server machine (where simulator is running) from the Manager machine (where the manager application is running). What could be the problem?

7. Is there a topology editor to create the connections between the devices easily?

8. Will the product work with multiple network cards?

9. I cannot start agents in simulator. I get "Unable to start agent" error when starting an agent instance. Why?

10. I have started agents in the simulator, but I cannot ping them from my management station. Why?

11. How do I add a MIB to an existing simulation?

---

1. In the Network Design Tool, if I set up a network with three virtual devices, does each need to have the same configuration?

   Each device in the network can have its own configuration with unique IP address and port number.

2. How can I add devices to the existing network. I think the Network Design wizard adds devices only to the new network?

   If you click on the new network and choose the Network Design wizard all the devices will be created in a new network.

   1. To add devices to an existing network, select Network -> Open and browse and select the .net file to open the network.

   2. Click on the Tools -> Network Design wizard or Tools -> Random Device Addition Wizard and configure the devices to be added. You can also drag and drop devices from the device tree to the network area.
3. I created one Network (network01) and added 10 Cisco Router SnmpNodes. I set virtual addresses (192.168.12.1 ~ 192.168.12.10) to SnmpNodes. The WebNMS discovers just one device. Could you check this problem?

During the discovery process, NMS will be querying the ipAddrTable to retrieve the IPAddress of the simulated agents. While adding SNMP devices to the network, the virtual IP address assigned to a device is automatically added to the ipaddrTable.

For Windows OS, the following step must be performed, in addition to changing the ipAddrTable.

Edit the "HOSTS" file on the Windows PC. It will exist in C:\WINNT\SYSTEM32\DRIVERS\ETC directory. Append the IP addresses to host names. Each entry should be kept on an individual line. The IP address should be placed in the first column followed by the corresponding host name. The host name can be the ip address. Any Other Name could also be given.

Format : <IP_Simulated>  <HOST_NAME>
Example : 192.168.12.1 192.168.12.1

4. I have simulated an agent with Read community = 'public' and Write community = 'private'. When I opened my NMS application with community 'private', which is defined as Read/Write, the simulated agents were not discovered. When I opened my NMS application with community 'public', which is defined as Read Only, all agents were discovered successfully. In addition the 'Community' = 'private' in MIB Browser does not allow to Get the parameters’ values.

The Read community = "public" indicates that for any get/get-next request, the "community field" in the MibBrowser should be "public". The Write Community = "private" indicates that for any set request the "community field" in the MibBrowser should be "private".

Similarly, in your NMS application, while doing a discovery the "community" that you have to specify is the Read community (here public). This is because, the discovery is nothing but a get/get-next operation on some node. Only while doing a Set from your NMS to the agents, the "community" that you have to specify is the Write Community (here private).

5. I have the real Agent with our own configuration, Is it possible to simulate this agent and start it as multiple instances?

Yes, you can use the Record Real Agents feature to record the agent and create multiple instances of this agent using the Network Designer tool.

6. I am unable to access the virtual interfaces added in the server machine (where simulator is running) from the Manager machine (where the manager application is running). What could be the problem?

If the simulator and manager application are running in separate machines, you have to use the route add command in the system where the manager application is running, to access the virtual interfaces. Refer to TroubleShooting Guide -> Network Simulator, for more information.

7. Is there a topology editor to create the connections between the devices easily?

Yes, the topology editor enables you to visualize interconnections between the devices in the network. You can connect new devices or disconnect existing devices in the topology. Refer to Topology Editor for more information.

8. Will the product work with multiple network cards?

The product does not support Multiple Network card environment. As virtual IPs are created dynamically it is not possible to distribute the creation of IPs to multiple cards.
9. I cannot start agents in simulator. I get "Unable to start agent" error when starting an agent instance. Why?

The above error will occur in Network Simulator in any one of the following cases:

- The port may be occupied by some other application. If you are unable to start the SNMP agents at port 161, refer to TroubleShooting Guide.
- There is a IP clash in the network. (i.e) the virtual IP configured in the simulator already exists in the network. Please ensure that the virtual ip address specified is unique.
- Product is installed in C:\Program Files. Actually this is a limitation of our product. Installation directory name should not have spaces.

10. I have started agents in the simulator, but I cannot ping them from my management station. Why?

If the simulator and the manager application is running in separate machines on the same network, the virtual ipaddress configured in the simulator (machine where the simulation application is running) will not be accessible from the Manager (machine where the manager application is running).

To access the simulated network, execute the `route add` command in the system where the manager application is running.

11. How do I add a MIB to an existing simulation?

You can add a MIB to the template devices in the device tree by using the SNMP -> Agent values right-click menu option. This will be reflected on all the template based devices in the network area. If a MIB is loaded to the template device when the agents are running, the agent will not return the newly added MIB values when the corresponding OIDs are queried. The network has to be restarted to register the OIDs.

You can also add a MIB to individual devices in the network area by using the SNMP -> Agent values right-click menu option or select Configure -> SNMP Agent Values. Select the "Load MIBs" option to load and simulate new MIBs.

Configuring Virtual IP Address

1. Is dynamic configuration of virtual IP supported?
2. I would like to start out by using the Demo Network. I am running Windows 2000 with DHCP configured. So far I can't seem to figure out how to add the virtual IP's - this capability is grayed out, presumably because of DHCP. Is there a way around this?

1. Is dynamic configuration of virtual IP supported?

Yes, dynamic configuration of virtual ip is supported in Windows NT, 2000, XP, 2003 and Linux/Solaris OS. Refer to the topic, Configuring Virtual IP Address, for more information.

2. I would like to start out by using the Demo Network. I am running Windows 2000 with DHCP configured. So far I can't seem to figure out how to add the virtual IP's - this capability is grayed out, presumably because of DHCP. Is there a way around this?

In a system with DHCP configuration virtual IP Addresses cannot be configured. In the case of the DHCP enabled machines you can have only one IP Address for the machine. An arbitrary IP Address is assigned to the machine when it boots up. Everytime the system is rebooted/restarted, a different IP Address is assigned to the system. For finding out the current IP Address of the machine, execute the command `ipconfig` in the command prompt. The work around for this is to simulate devices using different ports. For the DemoNetwork to work properly, change all the IP Addresses to the machine IP and provide different ports for the agents.
FAQs - TL1 Agent Simulator

1. How many TL1 NEs can the simulator handle?
   As of now, the simulator can handle 255 TL1 Agents only. Since this is a TCP Connection, we will be increasing this number to a limit of 5000+ Agents in the forthcoming release.

2. Can I (randomly) issue DENY responses?
   Yes, the TL1 agent simulator supports to simulate DENY responses at random. Refer to Simulating Random Variations in Response, for more information.

3. What is Delayed Activation? Does the simulator support to send delayed response?
   Delay Activation is one of the features available in TL1. Delay Activation is a function whereby an input message may be stored in a Message Pending buffer at the NE for final execution at some later time, either automatically or by a subsequent message from the OS. The TL1 agent simulator supports to simulate delayed responses. Refer to Simulating Random Variations in Response, for more information.

4. What is an autonomous message?
   An Autonomous message is a message that is sent from the NE to the appropriate OS without having an explicit input message associated with it. Typical scenarios where autonomous messages are used include:
   - Reporting of alarmed or non-alarmed trouble events.
   - Reporting of scheduled diagnostic tests in the NE.
   - Reporting of Performance Monitoring data.
   - Reporting of a change in the NE’s database.
   - Periodic reporting of selected NE conditions.

5. Is there any restriction on the maximum number of TL1 sessions that can be established with the simulated TL1 agent?
   There is no restriction on the maximum number of TL1 sessions that can be established with the simulated TL1 agent.

6. Does the TL1 agent provide security as per TR 835 and GR815 standards?
   User Security View is implemented in the TL1 Agent Simulator as per the specifications in GR 815 and TR 835.

7. Can I define behavior scripts for Command code - AID - MPB combinations?
   Yes, you can define TL1 behavior scripts for Command code - AID - MPB combinations. The TL1 Script Interface provides powerful script APIs to define agent behavior.
8. Does the TL1 Agent Simulator support Gateway NE simulation?

Yes, the TL1 Agent Simulator simulates NEs connected via a GNE and also simulates the GNE.
FAQs - IOS Simulator

1. What are the devices that support Cisco IOS Simulation?

The following series of Cisco Routers and Switches added to the network simulator's device tree, supports Cisco IOS Simulation.

- Cisco Router 1700
- Cisco Router 2600
- Cisco Router 3600
- Cisco Router 7400
- Cisco Router 7600
- Cisco Router 12000
- Cisco Switch 4500
- Cisco Switch 2950
- Cisco Switch 3750
- Cisco Switch 6500
- PIX Firewall

2. Does the IOS Simulator support all modes supported by Cisco IOS?

The IOS simulator supports the following modes - user, privilegedEXEC, global configuration, Interface configuration, Router (BGP, OSPF, RIP) mode and Vlan modes.

3. What are the commands supported by the IOS simulator?

Refer to the Help topic, IOS Simulator Guide -> IOS Commands supported, to learn about the complete set of commands supported by the IOS simulator.

4. What are the protocols supported by an IOS enabled device?

The IOS enabled devices support SNMP, CLI, TFTP and FTP protocols. The IOS Software Simulation can be accessed using CLI.

5. How can I use the IOS simulator to configure the router?

- Create a network with the IOS enabled devices. (Cisco Routers/ Switches)
- Start the device at CLI port and ipaddress. The default port is 2323.
- Establish a Telnet connection with the device running at the specified IP and Port.
- Login to the IOS Simulator, and start working with the commands to configure the router.

Refer to the Help topic, IOS Simulator Guide -> Getting Started, to learn more.
6. Are there any pre-configured Cisco devices available with the product?

Yes, the Cisco Router/Switch devices available in the device tree are pre-configured devices that can be used directly in the network.

7. What are the operations that can be performed on an IOS enabled device?

You can login and logout of the IOS simulator, get into and out of different modes, configure cdp, access-list, rmon alarms and events, set passwords, ip addresses, clock rates, hostname, enable/disable SNMP trap service, configure SNMP community, contact, location etc., specify MTU value for the network interface node, configure BGP, OSPF, RIP and vlan for a specific router, ping devices, display startup and running config and many other functions.

Refer to the Help topic, IOS Simulator Guide -> IOS Commands supported, to learn about the complete set of commands supported.

8. Is it possible for me to customize commands supported by the IOS device?

Yes, it is possible to customize the commands supported by the IOS device. The IOS simulator comes along with a Script Editor to facilitate viewing and editing of script files configured for the IOS commands. Refer to the Help topic, IOS Simulator Guide -> Script Editor, to learn about the complete set of commands supported.

9. Is it possible for me to add new commands to the existing IOS Device?

Yes, you can add new commands to the existing IOS devices. Refer to the Help topic, IOS Simulator Guide -> Adding IOS Commands, to learn about adding new commands.

10. Do you have any relation of SNMP OIDs with the IOS Commands? If yes, how can I do that?

If the IOS Command supports a MIB Object, the SNMP script API methods defined in the SNMP Script Interface can be used to implement the IOS command. For example, Cisco-CDP-MIB defines Mib object to enable/disable cdp. Hence, the command **cdp enable** uses the SNMP script API methods to implement the command.

Refer to the script file **cdp_enable_ci.py** in `<SimulatorHome>/devices/db/Cisco1700/cli` for reference. The SNMP and CLI script api methods are available in the javadocs inside `<SimulatorHome>/examples/javadocs` directory.

11. How can I send a trap on execution of an IOS Command?

You can send a trap by calling the SNMP Script interface method **sendTrap()** in the script file which executes the IOS command.