G2-SNMP Bridge

User's Guide

Version 4.0 Rev. 2

G2-SNMP Bridge User's Guide, Version 4.0 Rev. 2

June 2020

Ignite Technologies, Inc.
401 Congress Ave., Suite 2650
Austin, TX 78701 USA
Telephone: +1-800-248-0027
Email: success@ignitetech.com

Part Number: DOC102-400

Contents

Preface v

About this Guide v

Audience v

Conventions vi

Related Documentation vii

Customer Support Services x

Chapter 1 **Overview** 1

Introduction 1

Version Information 2

Summary of Features 2
- Standard SNMP (V2c) Functionality 2
- Bridge Configurability through GSI Interface Initialization Strings 3
- DNS Host Name Translation 3
- Filtering of Incoming Traps 3
- General Logging of SNMP Transactions 4
- Run-time Modification of Bridge Parameters 5
- Graceful Logging of Traps 5
- Transparent Type Conversion for Large Integer Values 5
- Bridge Shutdown Capability 5
- Conversion of Non-printing Character Strings 6

Acquiring Data from the G2-SNMP Bridge 6

Chapter 2 **Installation and Startup** 9

Introduction 9

System Configuration Requirements 9
- Supported Platforms 9
- Software Requirements 10
- Ensuring Correct Configuration 10

Installing the Bridge Software 11

Chapter 2 **Installation and Startup** *(continued)*

Running the Java-Based G2-SNMP Bridge **11**
- Running the Bridge **11**
- Using Command-Line Options **11**
- Starting a Bridge Process from within a G2 Application **12**
- Connecting G2 to the Bridge Process **12**
- Running a Bridge with Multiple Connections to G2 **14**
- Running Multiple Copies of the Bridge **14**
- Exiting the Bridge **14**

Chapter 3 **Command-Line Options** **15**

Introduction **15**

Chapter 4 **Bridge API Reference** **17**

Introduction **17**

Remote Procedure Calls **18**
- oxsj-g2snmp_blocking_transaction **19**
- oxsj-g2snmp_nonblocking_transaction **25**
- oxsj-g2snmp_modify_comm_params **30**
- oxsj-g2snmp_use_snmp_defaults **32**
- oxsj-g2snmp_use_snmp_comm_params **33**
- oxsj-g2snmp_add_filtered_trap **34**
- oxsj-g2snmp_delete_filtered_trap **37**
- oxsj-g2snmp_shutdown **40**

Receiver Procedures **41**
- g2snmp_receive_nonblocking **42**
- oxsj-g2snmp_receive_message **45**
- oxsj-g2snmp_receive_trap **46**

Chapter 5 **Bridge Messages** **49**

Introduction **49**

Error Messages **50**

Warning Messages **51**

G2 Messages **53**

Type Conversion Messages **55**

State Messages **56**

Appendix A **Known Bugs and Limitations** **57**

Index **59**

Preface

Describes this guide and the conventions that it uses.

About this Guide **v**

Audience **v**

Conventions **vi**

Related Documentation **vii**

Customer Support Services **x**

About this Guide

This guide:

- Provides reference information about the Java-based G2-SNMP (Simple Network Management Protocol) Bridge.
- Explains how to install and start the Java-based G2-SNMP Bridge.
- Contains specific instructions for using the Java-based G2-SNMP Bridge as a stand-alone application.
- Includes an appendix of known bugs and limitations.

Audience

This guide assumes that you are a bridge developer or test engineer. Because the Java-based bridge uses terms and concepts of SNMP and the G2 programming environment, a previous knowledge of SNMP and G2 is helpful but not essential. For descriptions of advanced G2 topics, please refer to the *G2 Reference Manual*.

Conventions

This guide uses the following typographic conventions and conventions for defining system procedures.

Typographic

Convention Examples	Description
g2-window, g2-window-1, ws-top-level, sys-mod	User-defined and system-defined G2 class names, instance names, workspace names, and module names
history-keeping-spec, temperature	User-defined and system-defined G2 attribute names
true, 1.234, ok, “Burlington, MA”	G2 attribute values and values specified or viewed through dialogs
Main Menu > Start KB Workspace > New Object create subworkspace Start Procedure	G2 menu choices and button labels
conclude that the x of y ...	Text of G2 procedures, methods, functions, formulas, and expressions
new-argument	User-specified values in syntax descriptions
<u>*text-string*</u>	Return values of G2 procedures and methods in syntax descriptions
File Name, OK, Apply, Cancel, General, Edit Scroll Area	GUIDE and native dialog fields, button labels, tabs, and titles
File > Save Properties	GMS and native menu choices
workspace	Glossary terms

Convention Examples	Description
`c:\Program Files\Gensym\`	Windows pathnames
`/usr/gensym/g2/kbs`	UNIX pathnames
`spreadsh.kb`	File names
`g2 -kb top.kb`	Operating system commands
`public void main()` `gsi_start`	Java, C and all other external code

Note Syntax conventions are fully described in the *G2 Reference Manual*.

Procedure Signatures

A procedure signature is a complete syntactic summary of a procedure or method. A procedure signature shows values supplied by the user in *italics*, and the value (if any) returned by the procedure underlined. Each value is followed by its type:

g2-clone-and-transfer-objects
(*list*: class item-list, *to-workspace*: class kb-workspace,
delta-x: integer, *delta-y*: integer)
–> *transferred-items*: g2-list

Related Documentation

G2 Core Technology

- *G2 Bundle Release Notes*
- *Getting Started with G2 Tutorials*
- *G2 Reference Manual*
- *G2 Language Reference Card*
- *G2 Developer's Guide*
- *G2 System Procedures Reference Manual*

- *G2 System Procedures Reference Card*
- *G2 Class Reference Manual*
- *Telewindows User's Guide*
- *G2 Gateway Bridge Developer's Guide*

G2 Utilities

- *G2 ProTools User's Guide*
- *G2 Foundation Resources User's Guide*
- *G2 Menu System User's Guide*
- *G2 XL Spreadsheet User's Guide*
- *G2 Dynamic Displays User's Guide*
- *G2 Developer's Interface User's Guide*
- *G2 OnLine Documentation Developer's Guide*
- *G2 OnLine Documentation User's Guide*
- *G2 GUIDE User's Guide*
- *G2 GUIDE/UIL Procedures Reference Manual*

G2 Developers' Utilities

- *Business Process Management System User's Guide*
- *Business Rules Management System User's Guide*
- *G2 Reporting Engine User's Guide*
- *G2 Web User's Guide*
- *G2 Event and Data Processing User's Guide*
- *G2 Run-Time Library User's Guide*
- *G2 Event Manager User's Guide*
- *G2 Dialog Utility User's Guide*
- *G2 Data Source Manager User's Guide*
- *G2 Data Point Manager User's Guide*
- *G2 Engineering Unit Conversion User's Guide*
- *G2 Error Handling Foundation User's Guide*
- *G2 Relation Browser User's Guide*

Bridges and External Systems

- *G2 ActiveXLink User's Guide*
- *G2 CORBALink User's Guide*
- *G2 Database Bridge User's Guide*
- *G2-ODBC Bridge Release Notes*
- *G2-Oracle Bridge Release Notes*
- *G2-Sybase Bridge Release Notes*
- *G2 JMail Bridge User's Guide*
- *G2 Java Socket Manager User's Guide*
- *G2 JMSLink User's Guide*
- *G2-OPC Client Bridge User's Guide*
- *G2 PI Bridge User's Guide*
- *G2-SNMP Bridge User's Guide*
- *G2 CORBALink User's Guide*
- *G2 WebLink User's Guide*

G2 JavaLink

- *G2 JavaLink User's Guide*
- *G2 DownloadInterfaces User's Guide*
- *G2 Bean Builder User's Guide*

G2 Diagnostic Assistant

- *GDA User's Guide*
- *GDA Reference Manual*
- *GDA API Reference*

Customer Support Services

You can obtain help with this or any Gensym product from Gensym Customer Support. Help is available online, by telephone and by email.

To obtain customer support online:

➔ Access Ignite Support Portal at `https://support.ignitetech.com`.

You will be asked to log in to an existing account or create a new account if necessary. Ignite Support Portal allows you to:

- Register your question with Customer Support by creating an Issue.
- Query, link to, and review existing issues.
- Share issues with other users in your group.
- Query for Bugs, Suggestions, and Resolutions.

To obtain customer support by telephone or email:

➔ Use the following numbers and addresses:

United States Toll-Free	+1-855-453-8174
United States Toll	+1-512-861-2859
Email	`support@ignitetech.com`

Overview

Describes the features of the Java-based G2-SNMP Bridge.

Introduction 1

Version Information 2

Summary of Features 2

Acquiring Data from the G2-SNMP Bridge 6

Introduction

The Java-based G2-SNMP (Simple Network Management Protocol) Bridge enables a user application to communicate with devices that support SNMP. This guide describes the functionality of the bridge and explains how to install and run it.

The Java-based G2-SNMP Bridge provides a set of functions to perform SNMP v2c transactions (SET, GET, GETNEXT, GETBULK, and INFORM) and to send and receive SNMP traps.

The bridge uses the G2 JavaLink/Gateway. JavaLink is a toolkit for creating Java-based bridges between G2 and external systems.

The Java-based G2-SNMP Bridge is designed to interact with the Operations Expert SNMP (OXS) application. The user's application will in turn interface with the bridge through the OXS application.

Version Information

The G2-SNMP Bridge supports SNMP version 2c.

Summary of Features

The Java-based G2-SNMP Bridge provides a user application with the following capabilities:

- Standard SNMP (V2c) Functionality: GET, SET, GETNEXT, GETBULK, INFORM requests and SEND/RECEIVE traps
- Bridge Configurability through GSI Interface Initialization Strings
- DNS Host Name Translation
- Filtering of Incoming Traps
- General Logging of SNMP Transactions
- Run-time Modification of Bridge Parameters
- Graceful Logging of Traps
- Transparent Type Conversion for Large Integer Values
- Bridge Shutdown Capability
- Conversion of Non-printing Character Strings

Standard SNMP (V2c) Functionality

The Java-based G2-SNMP Bridge supports all standard functions of SNMP version 2c, i.e., GET, SET, GETNEXT, GETBULK, INFORM requests and SEND/RECEIVE traps. GET, SET, GETNEXT, and GETBULK transactions are implemented in both synchronous (blocking) and asynchronous (non-blocking) modes.

To perform transaction and send traps, the bridge provides the following two RPCs, which are callable from a connected G2 application:

```
OXSJ-G2SNMP_BLOCKING_TRANSACTION
```

```
OXSJ-G2SNMP_NONBLOCKING_TRANSACTION
```

To return transaction results, pass incoming traps, and report error information to a connected G2 application, the bridge in its turn calls the following receiver procedures, which should be provided by the G2 application:

```
G2SNMP_RECEIVE_NONBLOCKING
OXSJ-G2SNMP_RECEIVE_TRAP
OXSJ-G2SNMP_RECEIVE_MESSAGE
```

For a detailed description of these and other procedures provided by the Java-based G2-SNMP Bridge, please refer to the "Bridge API Reference" chapter of this document.

Bridge Configurability through GSI Interface Initialization Strings

The user's application is able to configure certain parameters of the bridge via the initialization string of the GSI interface through which the G2 application is connected to the bridge. Most importantly, the interface initialization string is used to specify the mode of operation that the bridge will run in. For a list of options supported in interface initialization strings, see "Connecting G2 to the Bridge Process" on page 12.

DNS Host Name Translation

The bridge performs DNS name translation for network addresses contained in incoming traps/events. All IP addresses contained in any incoming SNMP packet (either a trap or a response) in attributes like `agent-hostname` or bound variables are looked up in DNS and supplied with a resolved hostname when possible.

Filtering of Incoming Traps

The Java-based G2-SNMP Bridge can filter traps that your application does not want or need to receive. The bridge performs trap filtering based on trap attributes (enterprise OID, generic ID, and specific ID). The bridge keeps a list of trap types that should be discarded by the bridge.

The list of traps filtered by the bridge is initially imported from an external filter definition file. The name of the filter definition file is communicated to the bridge via the -`ppdfilename` command line option. You should use this option to specify the full pathname (optionally) and filename of the filter definition file. When you connect to the bridge in mode 1 (to start the trap daemon), the bridge parses this filter definition file and adds any trap definitions found in it to the list of actively filtered traps.

Filtered traps are specified in the filter definition file as `IGNORE` or `LOGONLY`. The traps are identified by the three-element identification vector containing enterprise OID, generic trap ID, and specific trap ID. Every trap received by the bridge that matches one of the specified three-element identification vectors is ignored in the bridge, without any interaction with a connected G2 application. In compliance with SNMP specifications, the matching algorithm does not take into account specific trap IDs unless the generic ID is equal to 6.

While the Java-based G2-SNMP Bridge is running, your application can turn filtering on or off for any trap using the following remote procedure calls:

RPC	Description
`OXSJ-G2SNMP_ADD_FILTERED_TRAP`	This remote procedure call adds the trap to the filtered trap list. The bridge will not pass the trap to G2.
`OXSJ-G2SNMP_DELETE_FILTERED_TRAP`	This remote procedure call removes the trap from the filtered trap list. The bridge will now pass the trap to G2.

Note that the Java-based G2-SNMP Bridge does not require a PPD file for these RPCs to work correctly. For a detailed description of these and other procedures provided by the Java-based G2-SNMP Bridge, please refer to the "Bridge API Reference" chapter of this document.

General Logging of SNMP Transactions

The Java-based G2-SNMP Bridge provides the capability of logging every SNMP transaction performed as well as every trap received. Logged information is written to an external log file (in ASCII text). For each transaction or trap, the following information is logged:

- Timestamp
- Transaction type (GET, SET, GETNEXT, GETBULK, INFORM, SENDTRAP, or TRAP)
- Destination host name/agent address
- Transaction status – failed/sent *(*for transactions only*)*
- Enterprise OID, generic ID, specific ID *(*for incoming SNMP v1 traps only*)*
- Filter transaction details *(*for trap filter transactions only*)*

Run-time Modification of Bridge Parameters

The SNMP bridge possesses two adjustable parameters: one parameter specifies the number of retries for SNMP GET/SET requests (default value: 3) and one specifies the initial time interval between retries for SNMP GET/SET requests (default value: 0.8 sec.).

The bridge currently provides the following RPCs to allow changing its parameters from G2:

```
OXSJ-G2SNMP_MODIFY_COMM_PARAMS

OXSJ-G2SNMP_USE_SNMP_COMM_PARAMS

OXSJ-G2SNMP_USE_SNMP_DEFAULTS
```

For a detailed description of these and other procedures provided by the Java-based G2-SNMP Bridge, please refer to the Bridge API Reference chapter in this document.

Graceful Logging of Traps

The Java-based G2-SNMP Bridge provides the capability of handling incoming traps gracefully, i.e. without losing any traps during “burst periods”. The bridge keeps the traps in a special buffer (partially in RAM, partially in an external file). The bridge always buffers incoming traps in the file buffer whenever the internal trap queue becomes full. Regardless of the current trap arrival rate, the bridge renders all traps to the connected G2 application at a maximum rate of 20 traps per second.

Transparent Type Conversion for Large Integer Values

The bridge handles correctly the special case where the data type of an incoming trap field or an SNMP request parameter is an integer, with a value greater than the maximum integer value in G2. In this case the bridge converts the value to a float instead of an integer. All integer arguments of the bridge’s RPCs and receiver procedures that can contain large integers are declared as `quantity` in the bridge’s API.

Bridge Shutdown Capability

This version of the Java-based SNMP bridge supports shutdown of the JavaLink-based portion of the bridge from G2. The bridge API includes the OXSJ-G2SNMP_SHUTDOWN remote procedure that causes the JavaLink portion of the bridge to exit. For a detailed description of this RPC, please refer to the “Bridge API Reference” chapter in this document.

Conversion of Non-printing Character Strings

This version of the Java-based SNMP bridge includes improved handling of non-printing characters embedded within bound variables of type octet string (in traps and response PDUs). Previously, the bridge would return the octet string as it is, resulting in non-printing characters being passed to G2. This release of the bridge now converts any octet string containing one or more non-printing characters into a string in hexadecimal notation.

The bridge will consider printable characters to be all hex values from 0x20 to 0x7E (or the space character to '~' [tilde] in ASCII). If there is at least one non-printing character in an octet string, then all octets in the string will be represented byte-by-byte in the format:

```
'0xnn nn nn...'
```

where:

- `'nn'` are any two hexadecimal digits, 0x0-0xF.
- The leading `'0x'` prepended to the first byte signals that a hexadecimal representation of the octet string follows.

If all the values in the octet string are printable, the octet string will be represented in the normal ASCII format.

Acquiring Data from the G2-SNMP Bridge

The Java-based G2-SNMP Bridge can send information to OXS in either of two ways:

Mode 1: The bridge runs a trap daemon listening for traps and forwards them to OXS by making remote procedure calls (RPCs) to the trap receiver procedure (`OXSJ-G2SNMP_RECEIVE_TRAP`). Calls to the trap receiver procedure are blocking — that is, the bridge's calling thread has to wait while the call to the trap receiver procedure is being processed in G2.

Mode 2: OXS performs SNMP requests and sends traps through the bridge by making remote procedure calls to procedures in the bridge. The bridge returns data to OXS only when OXS solicits the data. Calls to remote procedures in the bridge may be blocking — that is, the called procedure does not return until the transaction is complete — or non-blocking.

Every separate bridge process may be used in mode 2 (performing SNMP transactions) by several G2 processes and/or by several GSI interface objects within one G2 process. A separate instance of SNMP transaction handler will be automatically started for each GSI interface connected in this mode.

A single bridge process may not be used by more than one GSI interface object connected in mode 1 (receiving traps), because there is no way to start several SNMP trap daemons on the same machine.

A typical user application requires that two GSI interfaces connect to the running bridge, one to receive traps and forward them to OXS and the other to perform SNMP transactions at the request of OXS.

If you are connecting to the bridge to receive traps (mode 1), any other trap daemon (such as the HP OpenView trap daemon) must be turned off on the machine running the bridge.

Installation and Startup

Provides instructions on installing and running the Java-based G2-SNMP Bridge.

Introduction 9

System Configuration Requirements 9

Installing the Bridge Software 11

Running the Java-Based G2-SNMP Bridge 11

Introduction

This chapter describes the system configuration requirements, installation, and running the G2-SNMP Bridge.

System Configuration Requirements

Supported Platforms

Since the Java-based G2-SNMP bridge is written using 100% pure Java, its platform requirements are the same as for G2 JavaLink.

The Java-based G2-SNMP bridge has been tested in the following configurations:

- Intel Pentium PC, Microsoft Windows XP Pro, JDK 1.3
- Sun SparcStation, Solaris 2.6, JDK 1.3

Software Requirements

The Java-based G2-SNMP bridge requires that the following software components be installed and functioning properly:

- G2 5.1 Rev. 6 or later.
- Sun’s Java Development Kit 1.3 or later (JDK 1.1.7B or later is required for Sparc Solaris-based systems).
- JavaLink 1.1 Rev. 0 or later.
- AdventNet SNMPv2c Version 3.2 or later (a version with SNMP v2c support is required).

Note For NT installations, JRE 1.3, Javalink 1.1r0, and the AdventNet libraries will automatically be installed on the disk.

This release will only work with G2 5.1 Rev. 6 (or later) and Sun's JDK 1.3, which should be correctly installed on your machine before using this software. Please refer to the JDK and G2 documentation for installation details. Please also note that Sparc Solaris-based systems require JDK 1.2 or later for JavaLink to work correctly.

This release of the bridge requires JavaLink 1.1 Rev. 0. The required files for Javalink are supplied with the installation of this bridge. If you will be installing a separate, full copy of Javalink 1.1r0, then please refer to the JavaLink `readme` file for installation and configuration details.

This release of the bridge requires AdventNet’s SNMPv2c Version 3.2 or later. The required AdventNet libraries have been supplied with this release of the Java-base G2-SNMP bridge. Note that AdventNet libraries come in several flavors (with SNMP v1, v2c, and v3 support). For the Java-based G2-SNMP bridge to operate correctly, you should install a package that supports SNMP v2c. This release of the bridge has been thoroughly tested with AdventNet 3.2.

Ensuring Correct Configuration

The installation process creates three directories: bin, classes, and G2snmp. The bin directory contains the batch file to start the Java SNMP bridge. This batch file contains all the set/configuration in order to make the bridge run. For NT systems JRE has been included in this release. For Solaris systems the JAVA_ HOME environment variable must point to you installation of JDK 1.3.

If you have separately installed Javalink 1.1r0 and/or a separate installation of AdventNet, please refer to those products installation procedures for the correct installation and configuration.

Installing the Bridge Software

Starting with Integrity release 3.3, a new installation procedure is followed. In order to install the Java-based G2-SNMP bridge, you must obtain a software license key that enables the installer to unlock the Java-based G2-SNMP bridge. This key may or may not unlock other products under the Integrity family of products. Please refer to the installation steps outlined on the CD cover for installation.

When the installation process is complete, the bridge is ready to run.

No additional authorization is required.

Running the Java-Based G2-SNMP Bridge

Running the Bridge

To start the bridge using the default parameters:

1 Change to the `bin` directory.

2 Type the following at the command prompt:

```
jsnmp
```

Upon successful startup, the bridge should display a welcoming message and the port it started listening on. Note that the bridge requires write access to the current directory to create a trap buffer and to write any log files.

Using Command-Line Options

When the bridge process is started with no command line options, the following default parameters are assumed:

- The bridge starts listening for G2 connection requests on port 22041.
- A log file is created in the current directory. Its name is constructed using the default `straps` prefix.
- The maximum number of lines in any single log file is assumed to be 50,000.
- No PPD file will be read when the trap-receive process is started.
- Debug mode is off (the bridge does not display any debugging messages).
- Silent mode is off (the bridge may display messages that reflect certain changes in its state.)

You can change some of the bridge's global parameters by using its command line options. For example, to make the bridge start listening on a port other than the default one, run it with the `-listenerport` option specified:

```
jsnmp -listenerport 22222
```

The command line options supported by the bridge are covered in detail in the "Command Line Options" chapter of this document.

Starting a Bridge Process from within a G2 Application

You can start a bridge process from within a G2 application by calling the following system procedures defined in the standard module `sys-mod.kb`:

```
id = call g2-spawn-process-to-run-command-line
("jsnmp <optional parameters>")

id = call g2-spawn-remote-process-to-run-command-line
("jsnmp <optional parameters>", <window>, <timeout>)
```

The first procedure starts a bridge process on the host running G2. The second procedure starts a bridge process on a host running the Telewindows process specified by the `window` argument. The return value is an operating system-specific process identifier. For a detailed description of these system procedures, please refer to the *G2 System Procedures Reference Manual*.

The command line specified can run a bridge process on every supported platform (provided that the bridge and all required software were correctly installed on that particular host).

Starting a bridge process does not establish a connection between the bridge process and G2. To establish connections, you must use corresponding GSI interface objects that contain the information needed to configure the connections. In particular, the network information in the Gsi-connection-configuration attribute must apply to the bridge process that you started.

To kill a connected bridge process from within a G2 application, you can use the `OXSJ-G2SNMP_SHUTDOWN` procedure defined in the Bridge API chapter. This procedure correctly closes any existing connections to the bridge and frees all resources in use by the bridge.

Connecting G2 to the Bridge Process

In order for a G2 application to be able to perform the SNMP transactions (send/receive traps, perform GET, SET, GETNEXT, GETBULK, and INFORM requests), you must configure the G2 application to talk to the bridge process. The following steps outline what you must do for G2 and the SNMP bridge to communicate.

- Create the GSI Interface Objects:

  Trap Receive Process
  Transactions Process

- Fill out the attributes of the GSI Interface Objects.
- Connect G2 to the GSI bridge processes via the GSI Interface objects.

When connecting to the Java-based G2-SNMP bridge, you should provide an initialization string for the GSI interface object. The interface initialization string, in particular, determines the type of bridge to be started and connected to G2 via that interface.

The following flags are supported in this string:

Flag	Description
`-d`	Debug ON. No `-d` flag specified indicates that debug is OFF (default).
`-t #`	The timeout period for SNMP retries, in tenths of a second specified as an integer (default value = 8 for 0.8 seconds).
`-p #`	Mode of operation that the Java-based G2-SNMP bridge will run in for that particular connection. Possible values are 1 and 2: 1 – receive traps. A trap daemon is automatically started in the same JVM when a GSI interface is connected to the bridge in this mode. If an SNMP trap daemon is already started on the machine running the bridge, the connection fails. The bridge process requires root privileges to start a trap daemon. 2 – perform SNMP v2c transactions (default mode). A separate instance of SNMP transaction handler is automatically created in the same JVM when a GSI interface is connected to the bridge in this mode.

The `oxsjdemo.kb` contains examples of two GSI Interface objects for connecting to the Java-based SNMP bridge.

Running a Bridge with Multiple Connections to G2

You can run a Java-based G2-SNMP bridge with connections to several different G2 knowledge bases. You can also run a bridge with several different connections to the same G2 knowledge base.

Each connection between a G2 knowledge base and a bridge is configured by a separate GSI interface object. When you start the G2 application, G2 establishes a connection to the bridge for each interface object that is fully defined and enabled.

You can use different connections to perform different kinds of actions. For example, you can perform SNMP transactions (sending outgoing SNMP requests and receiving responses) through one connection and listen for SNMP traps through another connection. You cannot perform both kinds of actions through the same connection, so if you want to perform transactions and receive traps in a single G2 knowledge base, you should create at least two connections to the bridge.

You can create as many "transaction" connections to a single bridge process as you wish. However, you cannot create more than one connection to a bridge process that listens for traps.

Running Multiple Copies of the Bridge

You can run multiple copies of the Java-based G2-SNMP bridge on the same machine. When starting the bridge on a machine already running a bridge process, make sure that you use the `-listenerport` command line option to specify a port that is different from the port used by the existing bridge. Otherwise, bridge startup will fail.

Note that even if you start several bridge processes on a single machine, only one of them will be able to listen for traps. It happens because only one process can use a specific TCP port for listening, and the currently implemented trap daemon can only listen on the default SNMP trap port (port 162).

Exiting the Bridge

Apart from pressing Ctrl-C in the bridge's console window or using a system-specific command to terminate the bridge process, the execution of the bridge can be interrupted by invoking its remote shutdown procedure from a connected G2. This procedure, named `OXSJ-G2SNMP_SHUTDOWN`, can be invoked across any GSI interface connected to the bridge (no matter what type of interface it is — "transaction" or "trap receive"). Invoking this RPC causes the bridge process to exit immediately. All connections between the bridge and any G2s connected to it are closed, regardless of their state.

For more information on the `OXSJ-G2SNMP_SHUTDOWN` RPC, please refer to the "Bridge API Reference" chapter of this document.

Command-Line Options

This section describes the command-line options for the Java-based G2-SNMP Bridge.

Introduction

The Java-based G2-SNMP bridge supports the following command-line options:

Option	Description
`-listenerport <port-number>`	Specifies the port to start listening on. The default port number is 22041.
`-log <filename>`	Specifies the prefix for the log file name. If this parameter is omitted (or `filename` is missing), a log file with a default prefix is created. Complete log file names consist of the specified prefix and a unique system-generated suffix (based on the current date). Note that for the log file to be successfully created the specified directory should allow write access for the bridge process.
`-nolog`	Specifies that no log file should be created.
`-maxloglines <number>`	Specifies the maximum number of lines to be written to a log file. When this limit is exceeded, the bridge closes the old log and creates a new log.

Option	Description
`-ppdfilename <filename>`	Specifies the PPD file containing trap filter patterns
`-debug`	Debug mode. When specified, allows the bridge to write debugging information to the console output. Should not normally be used.
`-silent`	Silent mode. Prevents the bridge from writing any information to the console output (except for debugging information if `-debug` is specified).
`-help`	Displays a short help message describing these command line options.

Bridge API Reference

This section describes the API for the Java-based G2-SNMP Bridge

Introduction 17

Remote Procedure Calls 18
- oxsj-g2snmp_blocking_transaction 19
- oxsj-g2snmp_nonblocking_transaction 25
- oxsj-g2snmp_modify_comm_params 30
- oxsj-g2snmp_use_snmp_defaults 32
- oxsj-g2snmp_use_snmp_comm_params 33
- oxsj-g2snmp_add_filtered_trap 34
- oxsj-g2snmp_delete_filtered_trap 37
- oxsj-g2snmp_shutdown 40

Receiver Procedures 41
- g2snmp_receive_nonblocking 42
- oxsj-g2snmp_receive_message 45
- oxsj-g2snmp_receive_trap 46

Introduction

This chapter provides a listing of the Java-based G2-SNMP bridge remote procedure calls accessible to the developer.

Remote Procedure Calls

The following is a list of the remote procedure calls from a G2 application to the Java-based G2-SNMP bridge. These RPCs are:

- Blocking and non-blocking transaction requests.
- Configuration of SNMP bridge parameters.
- Addition and deletion of trap filters.

oxsj-g2snmp_blocking_transaction on page 19
oxsj-g2snmp_nonblocking_transaction on page 25
oxsj-g2snmp_modify_comm_params on page 30
oxsj-g2snmp_use_snmp_defaults on page 32
oxsj-g2snmp_use_snmp_comm_params on page 33
oxsj-g2snmp_add_filtered_trap on page 34
oxsj-g2snmp_delete_filtered_trap on page 37
oxsj-g2snmp_shutdown on page 40

oxsj-g2snmp_blocking_transaction

Enables a G2 knowledge base to perform a blocking SNMP transaction.

```
oxsj-g2snmp_blocking_transaction (
{transaction-tagname} text,
{snmp-version} integer,
{request-code} integer,
{node-name} text,
{community-name} text,
{getbulk-parameters} structure (
        non-repeaters: integer,
        max-repetitions: integer
    ),
{variables}sequence (
        structure (
            variable-oid: text,
            variable-asn1-type: integer,
            variable-value: value
        )
        [,...]
    )
) = (
{return-values}structure (
        error-code: integer,
        error-string: text,
        node-name: text,
        variables:sequence (
            structure (
                error-code: integer,
                error-string: text,
                variable-oid: text,
                variable-asn1-type: integer,
                variable-value: value
            )
            [,...]
        )
    ))
```

Argument	Description
transaction-tagname	A handle for matching an incoming response with the outgoing request. Not currently used.
snmp-version	Version of the outgoing SNMP request (0 for SNMP v1, 1 for SNMP v2).
request-code	The type of blocking transaction requested. Possible values are: 160 (GET request), 161 (GETNEXT request), 163 (SET request), 165 (GETBULK request)
node-name	The name or dotted IP address of the node to which the transaction is directed.

Argument	Description
community-string	A string specifying the administrative relationship for the transaction.
getbulk-parameters	A structure specifying GETBULK request parameters with the following attributes: *non-repeaters* - specifies the number of elements from the beginning of the *variables* sequence for which a single instance should be returned; *max-repetitions* - specifies the maximum number of instances to return for the remaining elements in the *variables* sequence. This structure is ignored for non-GETBULK requests.

Argument	Description
variables	A sequence of structures containing variable bindings. Each structure in this sequence specifies a single variable and should have the following attributes: *variable-oid* - the object identifier (OID) in dotted notation for the variable involved in the transaction; *variable-ASN1-type* - the ANS.1 type of the variable involved in the transaction. The *variable-ASN1-type* attribute should be set to one of the following integer constants: 2 (INTEGER), 4 (STRING), 5 (NULL OBJECT), 6 (OBJECT ID), 64 (IP ADDRESS), 65 (COUNTER), 66 (GAUGE or UNSIGNED32), 67 (TIMETICKS), 68 (OPAQUE), 70 (COUNTER64) *variable-value* - the new value of the variable in a SET transaction. For GET, GETNEXT, or GETBULK requests, the attributes *variable-ANS1-type* and *variable-value* may be omitted.

Return Value	Description
return-values	A structure with the following attributes: *error-code* - returns a value of 0 if the transaction completes successfully else the SNMP integer error code. *error-string* - returns a value of “No Errors” if the transaction completes successfully else the text translation of the error specified by *error-code*. *node-name* - returns the node name to which the transaction was directed. *variables* - returns a sequence of structures, each structure describing a single bound variable and containing the following attributes: *error-code* - an error code for the bound variable (valid only for SNMPv2c requests) *error-string* - the text translation of the error (valid only for SNMPv2c requests) *variable-oid*, *variable-ASN1-type*, *variable-value* - contain the OID, the type, and the value of the returned bound variable (see above).

Description

Use this RPC to perform a blocking SNMP transaction. This RPC does not return until the transaction is complete (i.e. until the bridge receives a response to this request or times out).

Only the request types listed above are allowed. Any other request codes cause the RPC to signal an error by calling the OXSJ-G2SNMP_RECEIVE_MESSAGE receiver procedure.

A valid version of the SNMP request should be specified. If it is other than 0 or 1, then an error is generated. In case of request code and version value mismatch (for example, if you specify snmp-version=0 for a GETBULK request), the version parameter is ignored and the request is sent with the correct version number.

Variable OIDs should be valid object identifiers in dotted notation; otherwise, an error is generated. It may or may not contain the leading dot. In the latter case the leading dot is added automatically in the bridge.

The actual variable value should match the variable’s ASN.1 type specified. Otherwise, an error is generated. Although the value of any SNMP data type may be represented by a text value, the procedure allows passing values of the following G2 types: text, symbol, integer, and float. The bridge will attempt to cast the specified value to the specified type when possible; for example, if you specify an integer value for a variable of type STRING, it will be automatically converted to a string.

The current implementation of the bridge doesn’t allow selecting a TCP port number for a specific request. All outgoing blocking requests are sent to port 161, where an SNMP agent is supposed to listen to them.

This RPC is defined in the “transaction” portion of the bridge only. It cannot be invoked across a GSI interface connected to a “trap-receiver” bridge.

oxsj-g2snmp_nonblocking_transaction

Enables a G2 knowledge base to perform a non-blocking SNMP transaction.

```
oxsj-g2snmp_nonblocking_transaction (
{transaction-tagname} text,
{snmp-version} integer,
{request-code} integer,
{node-name} text,
{community-name} text,
{getbulk-parameters} structure (
        non-repeaters: integer,
        max-repetitions: integer
    ),
{trap-parameters} structure (
        enterprise-id: text,
        agent-IP-address: text,
        generic-trap-id: integer,
        specific-trap-id: quantity,
        agent-run-time: quantity
    ),
{variables} sequence (
        structure (
            variable-oid: text,
            variable-asn1-type: integer,
            variable-value: value
        )
        [,...]
    )
) = (
{result-id} text
)
```

Argument	Description
transaction-tagname	A handle for matching an incoming response with the outgoing request. Not currently used.
snmp-version	Version of the outgoing SNMP request (0 for SNMP v1, 1 for SNMP v2).

Argument	Description
request-code	The type of blocking transaction requested. Possible values are: 160 (GET request), 161 (GETNEXT request), 163 (SET request), 165 (GETBULK request) 166 (INFORM request) 167 (SENDV2TRAP request)
node-name	The name or dotted IP address of the node to which the transaction is directed.
community-string	A string specifying the administrative relationship for the transaction.
getbulk-parameters	A structure specifying GETBULK request parameters with the following attributes: *non-repeaters* - specifies the number of elements from the beginning of the *variables* sequence for which a single instance should be returned; *max-repetitions* - specifies the maximum number of instances to return for the remaining elements in the *variables* sequence. This structure is ignored for non-GETBULK requests.

Argument	Description
trap-parameters	A structure specifying trap parameters with the following attributes: *enterprise-id* - defines a unique vendor-specific device. It is the standard way of identifying an organization or company. The *enterprise-id* is included in the event header defined in the SNMP protocol that is passed as part of every SNMP event. *agent-IP-address* - the name or dotted IP address of the agent node to which the transaction is directed; *generic-trap-id* - generic and specific trap IDs refer to two fields in an SNMP trap definition which, along with the enterprise ID, identify a trap event uniquely. The *generic-trap-id* field can contain values 0 to 6. Values 0 to 5 refer to generic events, such as a warm start or a cold start. A value of 6 indicates that this is an enterprise-specific trap and that the *specific-trap-id* value is meaningful (it is assumed to be zero for generic traps); *specific-trap-id* - specifies an enterprise-specific trap event (if *generic-trap-id* is set to 6). Assumed to be 0 if *generic-trap-id* specifies a generic event (values 0 to 5); *agent-run-time* - the amount of time elapsed between the last initialization of the agent and the generation of the trap;

Argument	Description
variables	A sequence of structures containing variable bindings. Each structure in this sequence specifies a single variable and should have the following attributes: *variable-oid* - the object identifier (OID) in dotted notation for the variable involved in the transaction; *variable-ASN1-type* - the ANS.1 type of the variable involved in the transaction. The *variable-ASN1-type* attribute should be set to one of the following integer constants: 2 (INTEGER), 4 (STRING), 5 (NULL OBJECT), 6 (OBJECT ID), 64 (IP ADDRESS), 65 (COUNTER), 66 (GAUGE or UNSIGNED32), 67 (TIMETICKS), 68 (OPAQUE), 70 (COUNTER64) *variable-value* - the new value of the variable in a SET transaction or the reported value of the variable in a SENDTRAP, SENDV2TRAP, or INFORM transaction. For GET, GETNEXT, or GETBULK requests, the attributes *variable-ANS1-type* and *variable-value* may be omitted.

Return Value	Description
result-id	Returns a transaction identifier. It is a text representation of a large integer value.

Description

Use this RPC to perform a non-blocking SNMP transaction. This RPC returns immediately after the request has been sent, without waiting for any responses to arrive.

Only the request types listed above are allowed. Any other request code cause the RPC to signal an error by calling the `OXSJ-G2SNMP_RECEIVE_MESSAGE` receiver procedure.

SNMP version arguments and variable values are handled in the same way as in the `OXSJ-G2SNMP_BLOCKING_TRANSACTION` RPC.

Trap parameters are taken into account for SNMP v1 trap requests (SENDTRAP) only. Otherwise, the *trap-parameters* structure is ignored (may be empty, i.e. `structure ()`).

GETBULK parameters are taken into account for GETBULK requests only. Otherwise, the *getbulk-parameters* structure is ignored (may be empty, i.e. `structure ()`).

The current implementation of the bridge doesn't allow selecting a TCP port number for a specific request. All outgoing non-blocking requests (except for trap sending requests) are sent to port 161, where an SNMP agent is supposed to listen to them. SENDTRAP, SENDV2TRAP, and INFORM requests are sent to port 162.

For SENDV2TRAP and INFORM requests, the *trap-parameters* argument is not meaningful and ignored by the bridge. It is the caller's responsibility to provide correct values for the `sysUpTime` (.1.3.6.1.2.1.1.3.0) and `snmpTrapOid` (.1.3.6.1.6.3.1.1.4.1.0) variable bindings required for SNMPv2c TRAP/INFORM requests.

The unique transaction identifier returned by this RPC (*result-id*) can be used to identify the non-blocking request when a response is returned via the `G2SNMP_RECEIVE_NONBLOCKING` receiver procedure (the response to a particular non-blocking request will have the same *result-id* value as the originating request). When sending a trap, the result-id value has no special meaning and may be ignored.

This RPC is defined in the "transaction" portion of the bridge only. It cannot be invoked across a GSI interface connected to a "trap-receiver" bridge.

oxsj-g2snmp_modify_comm_params

Enables a G2 knowledge base to modify the timeout interval and the number of retries for sending SNMP requests.

```
oxsj-g2snmp_modify_comm_params (
{time-out-interval} integer,
{retry-count} integer,
) = (
{return-values} structure (
        error-code: integer,
        error-string: text
    )
)
```

Argument	Description
time-out-interval	The initial length of time, in tenths of a second, before the bridge attempts to try sending the request again. For subsequent retries, the bridge increases the interval exponentially. For example, if *time-out-interval* = 2 and *retry-count* = 3 and the first request is not answered, the bridge will repeat the request three times with the delays of 200, 400, and 800ms.
retry-count	The number of times the bridge makes a request after the initial request. For example, if *retry-count* = 3 and the bridge sends an initial request that is not answered, it sends three additional identical request before returning an error. The *error-string* is "Error ([transaction type]): request timed out to [node-name]."

Return Value	Description
return-values	A structure with the following attributes: *error-code* - returns a value of 0 if the transaction is successful else the bridge integer error code. *error-string* - returns a text value of "[*time-out-interval*]-[*retry-count*]" if the transaction is successful else the text translation of the bridge error code.

Description

Use this RPC to modify the timeout interval and the number of retries for sending SNMP requests.

The *time-out-interval* and *retry-count* arguments cannot be zero or negative, otherwise an error is signaled via the `G2SNMP_RECEIVE-MESSAGE` receiver procedure.

In the current version of the bridge, the parameters specified by a call to this procedure do not take effect until a call to the `OXSJ-G2SNMP_USE_SNMP_COMM_PARAMS` procedure is made.

This RPC is defined in the “transaction” portion of the bridge only. It cannot be invoked across a GSI interface connected to a “trap-receive” bridge.

This RPC changes the timeout and the number of retries for the current connection only. All other transaction handlers that may be started in the same bridge are unaffected.

oxsj-g2snmp_use_snmp_defaults

Enables a G2 knowledge base to configure the bridge to use the default values for the request timeout interval and the number of retries.

```
oxsj-g2snmp_use_snmp_defaults (
) = (
{return-values} structure (
        error-code: integer,
        error-string: text
    )
)
```

Return Value	Description
return-values	A structure with the following attributes: *error-code* - returns a value of 0. *error-string* - returns a text value of GSI_SUCCESS.

Description

Use this RPC to configure the bridge to use the default values for the request timeout interval (length of time before the bridge attempts to try sending the request again in tenths of a second) and the number of retries (after the initial request) for re-sending the request. These parameters apply to all subsequent outgoing SNMP requests. The default parameter values are the following:

Request timeout = 0.8 seconds (8 in tenths of a second)

Number of retries = 3

This RPC is defined in the “transaction” portion of the bridge only. It cannot be invoked across a GSI interface connected to a “trap-receiver” bridge.

This RPC changes the timeout and the number of retries for the current connection only. All other transaction handlers that may be started in the same bridge are unaffected.

oxsj-g2snmp_use_snmp_comm_params

Enables a G2 knowledge base to configure the bridge to use the user-specified values for the request timeout interval and the number of retries.

```
oxsj-g2snmp_use_snmp_comm_params (
) = (
{return-values} structure (
          error-code: integer,
          error-string: text
     )
)
```

Return Value	Description
return-values	A structure with the following attributes: *error-code* - returns a value of 0. *error-string* - returns a text value of GSI_SUCCESS.

Description

Use this RPC to configure the bridge to use the user-specified values for the request timeout interval (length of time before the bridge attempts to try sending the request again in tenths of a second) and the number of retries (after the initial request) for re-sending the request.

This RPC is defined in the “transaction” portion of the bridge only. It cannot be invoked across a GSI interface connected to a “trap-receiver” bridge.

This RPC changes the timeout and the number of retries for the current connection only. All other transaction handlers that may be started in the same bridge are unaffected.

oxsj-g2snmp_add_filtered_trap

Enables a G2 knowledge base to add a trap defined by an enterprise ID, generic trap ID, and specific trap ID to the list of traps that the bridge filters.

```
oxsj-g2snmp_add_filtered_trap (
{parameters}structure (
        enterprise-id: text,
        generic-trap-id: integer,
        specific-trap-id: quantity,
        agent-IP-address: text,
        agent-hostname: text
    )
) = (
)
```

Argument	Description
parameters	A structure specifying a trap to be filtered. This structure should contain the following attributes: *enterprise-id* - defines a unique vendor-specific device. It is the standard way of identifying an organization or company. The *enterprise-id* is included in the event header defined in the SNMP protocol that is passed as a part of every SNMP event. It is a component of an SNMP Object Identifier (OID) in dotted notation. *generic-trap-id* - generic and specific trap IDs refer to two fields in an SNMP trap definition which, along with the enterprise ID, identify a trap event uniquely. The *generic-trap-id* field can contain values 0 to 6. Values 0 to 5 refer to generic events, such as a warm start or a cold start. A value of 6 indicates that this is an enterprise-specific trap and that the *specific-trap-id* value is meaningful (it is assumed to be zero for generic traps). *specific-trap-id* - specifies an enterprise-specific trap event (if *generic-trap-id* is set to 6). Assumed to be 0 if *generic-trap-id* specifies a generic event (values 0 to 5). *agent-IP-address* - specify none or do not specify at all. Not yet implemented. *agent-hostname* - specify none or do not specify at all. Not yet implemented.

Description

Use this RPC to add a trap defined by the specified enterprise ID, generic trap ID, and specific trap ID to the list of traps that the bridge filters. If the specified trap is already in the list, it is not duplicated; the existing entry is used. Filtered traps are deleted at the bridge and not passed to G2.

For example, to ignore all traps with an enterprise ID of “1.3.6.1.4.1.597.2.1”, a generic Id of 6 (meaning this is an enterprise-specific trap), and a specific ID of 7, invoke OXSJ-G2SNMP_ADD_FILTERED_TRAP() with these parameters. At the time the remote procedure is invoked, the bridge begins deleting any traps matching this enterprise/generic/specific signature; thus, they are not returned to G2.

The enterprise-id parameter has to be a valid SNMP object identifier; otherwise, an error is generated. The valid range for generic-trap-id is [0-6]; for specific-trap-id, it is [0-2147483647].

In the current version of the bridge, SNMPv2c traps are not filtered. All received SNMPv2c traps are forwarded to G2.

This RPC is defined in the “trap-receiver” portion of the bridge only. It cannot be invoked across a GSI interface connected to a “transaction” bridge.

oxsj-g2snmp_delete_filtered_trap

Enables a G2 knowledge base to disable filtering a trap defined by an enterprise ID, generic trap ID, and specific trap ID and remove it from the list of traps that will be filtered by the bridge.

```
oxsj-g2snmp_delete_filtered_trap (
{parameters} structure (
        enterprise-id: text,
        generic-trap-id: integer,
        specific-trap-id: quantity,
        agent-IP-address: text,
        agent-hostname: text
    )
) = (
)
```

Argument	Description
parameters	A structure specifying a trap to stop filtering. This structure should contain the following attributes: *enterprise-id* - defines a unique vendor-specific device. It is the standard way of identifying an organization or company. The *enterprise-id* is included in the event header defined in the SNMP protocol that is passed as a part of every SNMP event. It is a component of an SNMP Object Identifier (OID) in dotted notation. *generic-trap-id* - generic and specific trap IDs refer to two fields in an SNMP trap definition which, along with the enterprise ID, identify a trap event uniquely. The *generic-trap-id* field can contain values 0 to 6. Values 0 to 5 refer to generic events, such as a warm start or a cold start. A value of 6 indicates that this is an enterprise-specific trap and that the *specific-trap-id* value is meaningful (it is assumed to be zero for generic traps). *specific-trap-id* - specifies an enterprise-specific trap event (if *generic-trap-id* is set to 6). Assumed to be 0 if *generic-trap-id* specifies a generic event (values 0 to 5). *agent-IP-address* - specify none or do not specify at all. Not yet implemented. *agent-hostname* - specify none or do not specify at all. Not yet implemented.

Description

Use this RPC to disable filtering a trap defined by the specified enterprise ID, generic trap ID, and specific trap ID and remove it from the list of traps that will be filtered by the bridge

For example, to begin receiving a previously filtered trap with an enterprise ID of “1.3.6.1.4.1.597.2.1”, a generic Id of 6 (meaning this is an enterprise-specific trap), and a specific ID of 7, invoke OXSJ-G2SNMP_DELETE_FILTERED_TRAP() with these parameters. At the time the remote procedure is invoked, the bridge stops deleting traps matching this enterprise/generic/specific signature, and will return them to G2.

This RPC can also remove a trap definition that was added by parsing a PPD file at the bridge startup.

This RPC is defined in the “trap-receiver” portion of the bridge only. It cannot be invoked across a GSI interface connected to a “transaction” bridge.

oxsj-g2snmp_shutdown

This RPC provides a clean mechanism to shut down the Java-based portion of the bridge. This RPC does not take or return any arguments.

```
oxsj-g2snmp_shutdown (
) = (
)
```

Description

Invoking this RPC causes the bridge process to exit immediately with an exit code of 0. All connections between the bridge and any G2s connected to it are closed, regardless of their state. All resources in use by the bridge are freed.

It is recommended that this RPC be invoked from G2 by using a “start” statement, not “call”. Otherwise, an error will be signaled to G2, indicating that the GSI connection has broken during the call to this procedure.

Unlike all other RPCs, this particular RPC is well understood by both “trap-receive” and “transaction” bridges. It can be invoked across any GSI interface connected to a Java-based G2-SNMP bridge.

Receiver Procedures

These are G2 procedures that the Java-based G2-SNMP bridge calls to return data to G2. The bridge can call these procedures to return:

- Data requested by non-blocking requests;
- Error messages;
- Traps received by the bridge through the SNMP trap receiver process.

g2snmp_receive_nonblocking on page 42
oxsj-g2snmp_receive_message on page 45
oxsj-g2snmp_receive_trap on page 46

g2snmp_receive_nonblocking

Receives the parameters of a response to a non-blocking request.

```
g2snmp_receive_nonblocking (
{parameters} structure (
        error-code: integer,
        error-string: text,
        result-id: text,
        node-name: text,
        variables: sequence (
            structure (
                error-code: integer,
                error-string: text,
                variable-oid:t ext,
                variable-ASN1-type: integer,
                variable-value: value
            )
            [,...]
        )
    )
) = (
)
```

Argument	Description
parameters	A structure specifying the parameters of the non-blocking response. This structure contains the following attributes. *error-code* - the value of 0 if the transaction completes successfully else the SNMP integer error code; *error-string* - the value of "No Errors" if the transaction completes successfully else the SNMP error description; *result-id* - a transaction identifier. It is a text representation of a large integer value. Can be used to identify the non-blocking request that originated this response; *node-name* - the node-name from which the variables are requested. This is returned as either the hostname or the IP address.; *variables* - a sequence of structures, each structure describing a single bound variable and containing the following attributes: *error-code* - an error code for the bound variable (valid only for SNMPv2c requests; *error-string* - the text translation of the error (valid only for SNMPv2c requests) *variable-oid* - the object identifier of the received variable; *variable-ASN1-type* - the ASN.1 type of the received variable *variable-value* - the value of the received variable.

Description

The bridge calls this RPC when a response to a non-blocking request arrives. Use the result-id parameter to identify the non-blocking request that originated this response.

The values returned from the SNMP agent are converted to G2 values using the following conversion logic:

SNMP Data Type	G2 Data Type
INTEGER, TIMETICKS, COUNTER, COUNTER64, UNSIGNED32	If the SNMP value is greater than the maximum G2 integer value (2^29-1) or less than the minimum (-2^29+1), then the value returned to G2 will be of type FLOAT, else INTEGER
STRING, OPAQUE	TEXT. Strings containing one or more non-printing characters are converted into strings in hexadecimal notation.
IPADDRESS, OBJID	TEXT

oxsj-g2snmp_receive_message

Receives a message in G2 from the Java-based G2-SNMP bridge.

```
oxsj-g2snmp_receive_message (
{tag} text,
{node-name} text,
{error-string} text,
{error-code} integer
)
```

Argument	Description
tag	General text to be displayed as a preface to the message.
node-name	The *node-name* from which the message was sent.
error-string	Text description of the message status
error-code	Message status.

Description

This procedure is called by the bridge to send a message to G2 (usually, to indicate that an error has occurred). For a list of possible error messages returned via this RPC, please refer to the section named "*G2 Messages*" in the "Bridge Messages" chapter of this document.

oxsj-g2snmp_receive_trap

Receives a trap in G2 from the Java-based G2-SNMP bridge.

```
oxsj-g2snmp_receive_trap (
{error-code} integer,
{error-string} text,
{result-id} text,
{snmp-version} integer,
{enterprise-id} text,
{agent-ip-addr} text,
{agent-hostname} text,
{generic-id} integer,
{specific-id} quantity,
{agent-run-time} quantity,
{variables} sequence (
        structure (
            error-code: integer,
            error-string: text,
            variable-oid: text,
            variable-ASN1-type: integer,
            variable-value: value
        )
    ),
{community-string} text
)
```

Argument	Description
error-code	The value 0 if the transaction completes successfully else the SNMP integer error code.
error-string	The value "No Errors" if the transaction completes successfully else the SNMP error description
result-id	A transaction identifier. It is a text representation of a large integer value.
snmp-version	The SNMP version of the received trap.
enterprise-id	Defines a unique vendor-specific device. it is the standard way of identifying an organization or company. The enterprise-id is included in the event header defined in the SNMP protocol that is passed as part of every SNMP event. It is a component of an SNMP Object Identifier (OID) in dot notation.

Argument	Description
agent-IP-address	The dot-notation IP address of the agent sending the trap.
agent-hostname	The resolved name of the host on which the agent sending the trap resides.
generic-trap-id	Generic and specific trap IDs refer to two fields in an SNMP trap definition which, along with the enterprise ID, identify a trap event uniquely. The *generic-trap-id* field can contain values 0 to 6. Values 0 to 5 refer to generic events, such as a warm start or a cold start. A value of 6 indicates that this is an enterprise-specific trap and that the *specific-trap-id* value is meaningful (it is assumed to be zero for generic traps).
specific-trap-id	An enterprise-specific trap event (if *generic-trap-id* is set to 6). Assumed to be 0 if *generic-trap-id* specifies a generic event (0 to 5).
agent-run-time	The amount of time elapsed between the last initialization of the agent and the generation of the trap.
variables	A sequence of structures, each structure describing a single bound variable and containing the following attributes: *error-code* - an error code for the bound variable (valid only for SNMPv2c requests); *error-string* - the text translation of the error (valid only for SNMPv2c requests); *variable-oid* - the object identifier of the received variable; *variable-ASN1-type* - the ASN.1 type of the received variable; *variable-value* - the value of the received variable
community-string	The community string of the agent sending the trap.

Description

The bridge calls this procedure when a trap arrives to provide a G2 application with the received trap's data. This receiver procedure is not called for traps matching one of the trap definitions in the list of actively filtered traps.

This RPC uses the same type conversion logic as the `G2SNMP_RECEIVE_NONBLOCKING` RPC.

The *result-id* argument has no meaning here and is always set to 0.

At execution time a logical-parameter, called `OXSJ-CREATE-TRAP-MIB-RECEIVER`, is checked. If this value is TRUE, a search will be conducted searching for a matching object definition. If will first attempt to match "trap-[*enterprise-name*]-[*generic-trap-id*]-[*specific-trap-id*]". It will then start using XX in place of *specific-tap-id*, *generic-trap-id*, and finally *enterprise-name*, resulting in "trap-XX-XX-XX". If a definition if found, a mib-receiver object is created and populated. If no object-definition is found and `MIB-CREATE-NONEXISTENT-TRAPS` is greater than 0, a new trap class will be created in the form of "trap-[*enterprise-name*]-[*generic-trap-id*]-[*specific-trap-id*]".

Finally, the varbind is populated into the mib-receiver. If the `OXSJ-CREATE-MIB-RECEIVER` is FALSE and there exists a procedure named by the `OXSJ-PROCESS-TRAP-STRUCTURE-PROC`, this procedure will be executed passing the following structure:

error-code:
error-text:
result-id:
snmp-version:
enterprise-id:
agent-ip-address:
agent-hostname:
generic-trap-id:
specific-trap-id:
agent-run-time:
variables:
community-string:

The values of the fields of this structure are populated with the information passed to the `OXSJ-G2SNMP_RECEIVE_TRAP` procedure.

Bridge Messages

This section describes the messages for the Java-based G2-SNMP Bridge.

Introduction **49**

Error Messages **50**

Warning Messages **51**

G2 Messages **53**

Type Conversion Messages **55**

State Messages **56**

Introduction

Depending on the severity of errors, the Java-based G2-SNMP bridge may report errors in either of the following ways:

- Printing an error or warning message on the bridge’s console. These messages are prefixed with an “Error:” or “Warning:” tag.
- Sending an error message to the G2 process that caused the error (via the `OXSJ-G2SNMP_RECEIVE_MESSAGE` RPC).

To indicate certain changes in its state, the Java-based G2-SNMP bridge may also print state messages on the console.

Error Messages

The table below shows the error messages that may be issued by the Java-based G2-SNMP bridge to the console during execution.

Error Message	Meaning
Error: incorrect -log parameter value	The value specified in the `-log` command line option was invalid. The bridge exits with an exit code of 1.
Error: incorrect port number	The value specified in the `-listenerport` command line option was invalid. The bridge exits with an exit code of 1.
Error: TCP port <port> initialization failed	Javalink was unable to set up a listening session on the port specified. The bridge exits with an exit code of 1.
Error: G2 Gateway initialization failed	Javalink was already initialized. The bridge exits with an exit code of 1.
Error: SNMP trap receiving daemon failed to initialize	The bridge was unable to create an instance of the TRAP RECEIVE daemon. the bridge continues to run normally. A possible cause is that the bridge failed to set up a listening session on port 162, which is used to receive traps. Note that the bridge requires root privileges for this operation.
Error: SNMP transaction handler failed to initialize	The bridge was unable to create an instance of the GET/SET bridge (an SNMP session could not be established). The bridge continues to run normally.
Error: Non-positive timeout specified. Using default value	In a call to the `OXSJ-G2SNMP_MODIFY_COMM_PARAMS` RPC, a zero or negative value was specified in the timeout parameter. No SNMP parameters were changed. the bridge continues to run normally.
Error: Non-positive number of retries specified. Using default value	In a call to the `OXSJ-G2SNMP_MODIFY_COMM_PARAMS` RPC, a zero or negative value was specified in the retries parameter. No SNMP parameters were changed. The bridge continues to run normally.

Warning Messages

The table below shows the warning messages that may be issued by the Java-based G2-SNMP bridge to the console during execution.

Warning Message	Meaning
Warning: Getting G2 connection information failed	The bridge was unable to obtain connection data from G2.
Warning: G2 access (from <host>:<port> was denied because of unknown -p parameter value	the value specified in the `-p` option of the GSI interface's initialization string was invalid.
Warning: Log file writing failed	An IO error occurred when attempting to write to the current log file.
Warning (trap buffer): failed to create SNMP variable	Invalid MIB variable type encountered when restoring a trap PDU from the file buffer.
Warning (trap filter): No PPD file named specified. Starting with empty filter	No PPD file name was specified in the `-ppdfilename` command line option.
Warning (trap filter): PPD file not found. Starting with empty filter	The file specified in the `-ppdfilename` command line option was not found.
Warning (trap filter): Less than 5 elements in line <N> ignored	Line <N> in the PPD file has invalid format. The rest of the PPD file was ignored.
Warning (trap filter): <typemessage>. Line <N> ignored	Line <N> was ignored because of invalid trap parameters.

Warning Message	Meaning
Warning (trap filter): I/O exception parsing PPD file. Line <N>	An IO error occurred when attempting to read the PPD file specified.
Warning: Can't forward trap to G2	An error occurred when attempting to send trap information to G2.
Warning (trap queue): can't add PDU	An IO error occurred when attempting to store a trap PDU into the file buffer.
Warning (trap queue): can't read trap buffer	An IO error occurred when attempting to restore a trap PDU from the file buffer.
Warning (trap queue): can't clear trap buffer	An IO error occurred when attempting to clear the file-based trap buffer.
Warning (trap queue): can't create trap buffer	An IO error occurred when attempting to create the file-based trap buffer.
Warning (trap queue): buffer overflow detected, PDU rejected	In non-buffered mode, the internal trap queue's capacity was exceeded. A number of subsequent traps are going to be lost. (This message may only appear in non-buffered mode, meaning that the file-based trap buffer could not be created for some reason).
Warning (trap queue): <N> PDU(s) lost due to buffer overflow, continuing normally	In non-buffered mode, the internal trap queue is now able to receive traps normally. The message also indicates the number of traps lost due to buffer overflow. (This message may only appear in non-buffered mode, meaning that the file-based trap buffer could not be created for some reason.)

G2 Messages

To indicate that a serious error occurred during the processing of an SNMP transaction, the Java-based G2-SNMP bridge may report error information to G2 via the `OXSJ-G2SNMP_RECEIVE_MESSAGE` RPC. The table below lists the messages that can be issued by the bridge via that RPC.

Tag	Message	Meaning
Error modifying comm params	Non-positive timeout specified	In a call to the `OXSJ-G2SNMP_ MODIFY_COMM_PARAMS` RPC, a zero or negative value was specified in the timeout parameter. No SNMP parameters were changed.
Error modifying comm params	Non-positive number of retries specified	In a call to the `OXSJ-G2SNMP_ MODIFY_COMM_PARAMS` RPC, a zero or negative value was specified in the timeout parameter. No SNMP parameters were changed.
Error modifying comm params	Error creating return structure	The `MODIFY_COMM_PARAMS` RPC succeeded, but failed to create the return structure.
Error using snmp defaults	Error creating return structure	The `USE_SNMP_DEFAULTS` RPC succeeded, but failed to create the return structure.
Error (nonblocking transaction)	Wrong SNMP version parameter (<N>	An attempt was made to preform a non-blocking transaction with the wrong SNMP version (must be 0 or 1 for SNMP v1 or v2c, respectively).
Error (nonblocking transaction)	<type conversion message>	An attempt was made to perform a non-blocking transaction with incorrect parameters (via the `OXSJ-G2SNMP_NONBLOCKING_ TRANSACTION` RPC). Type conversion messages are listed below in the next section.

Tag	Message	Meaning
Error (blocking transaction)	wrong SNMP version parameter <N>	An attempt was made to preform a non-blocking transaction with the wrong SNMP version (must be 0 or 1 for SNMP v1 or v2c, respectively).
Error (blocking transaction)	cannot perform blocking request for the request code <code>	An attempt was made to perform a blocking SENDTRAP or INFORM transaction. These transaction types can only be non-blocking.
Error adding filtered trap	<type conversion message>	An attempt was made to add a filtered trap with incorrect parameters (via the `OXSJ-G2SNMP_ADD_FILTERED_TRAP` RPC).
Error deleting filtered trap	<type conversion message>	An attempt was made to delete a filtered trap with incorrect parameters (via the `OXSJ-G2SNMP_DELETE_FILTERED_TRAP` RPC).

Type Conversion Messages

These messages may appear in extended error information provided by the `OXSJ-G2SNMP_RECEIVE_MESSAGE` RPC. Errors indicated by these messages are usually caused by an incorrect parameter being passed to either the blocking or the non-blocking RPC.

Type Conversion Message	Meaning
Empty VARIABLE-VALUE or VARIABLE-ASN1-TYPE is not allowed for the request of type <code>	The value or type of a bound variable was not specified. For SET and SENDTRAP requests, the entire triad of (VARIABLE-OID, VARIABLE-ASN1-TYPE, VARIABLE-VALUE) must be specified for each bound variable.
Unknown host <hostname>	The specified hostname cannot be resolved.
Cannot create variable <oid> of type <type> with value <value>	An attempt was made to create a bound variable with an invalid value for the given type, or the type was invalid.
<name> does not exist	The attribute <name> was not found in the structure passed to the RPC.
Invalid type conversion for the attribute <name>	An item or value of the wrong type was specified for the attribute <name>.
Invalid type <Java type> for the attribute <name>	The value specified for the attribute <name> was not a Double, Integer, String, or Symbol value
Invalid OID argument: <oid>	The object identifier (of a bound variable or the enterprise attribute) was invalid.
Value <value> is out of interval [0-6] for generic-id	An attempt was made to create a trap with an invalid value of `generic-id`.
Value <value> is out of interval [0-2147483647] for specific-id	An attempt was made to create a trap with an invalid value of `specific-id`.
Value <value> is out of interval [0-4294967295] for agent-uptime	An attempt was made to create a trap with an invalid value of `agent-uptime`.

State Messages

To report information about certain changes in its state, the Java-based G2-SNMP bridge may issue state messages to the console. The table below lists the messages that can be issued by the bridge for that purpose. Note that the output of state messages may be suppressed by specifying the -silent command line option.

State Message	Meaning
Java-based G2-SNMP Bridge Version X.X Rev. X	Start-up message
Listening on port <N>	The bridge successfully established a listening session on the specified port and is waiting for G2 connections.
G2 access (from <host>:<port> was granted to receive traps	A G2 connection was established to a RECEIVE TRAP bridge.
G2 access (from <host>:<port> was granted to perform transactions	A G2 connection was established to a GET/SET bridge.
G2 access (from <host>:<port> was closed	A G2 connection was closed.
A connected G2 asked to shut down	A G2 process connected to the bridge requested bridge shutdown. All connections will be closed and the bridge will exit with exit code of 0.
Trap queue: buffered mode, cache size: <N>	The file-based trap buffer was created successfully. The limit of the internal trap queue is <N>, which means that up to <N> traps will be sorted in memory, and any traps arriving after this limit was exceeded will be temporarily stored in the file-based buffer.
Trap queue: non-buffered mode, maximum allowable traps in queue: <N>	The file-based trap buffer was not created due to some error. The limit of the internal trap queue is <N>. Any traps arriving after this limit was exceeded will be rejected.

Known Bugs and Limitations

This section describes the known bugs and limitations of the Java-based G2-SNMP Bridge.

The following are known bugs and limitations:

- A lengthy delay occurs when the bridge receives a trap with a non-resolvable agent address.
- Empty or null IP addresses in SNMP PDUs are resolved to the 'localhost' address.
- When sending a blocking GET request to an existing inaccessible host (for example, a host with a disabled SNMP agent), a timeout message should appear in G2; instead, the AdventNet library sometimes throws "SnmpException: sync send request failed".
- Times written to the log file may be in GMT, depending on the current time zone.
- To send a v2c trap, the sender should explicitly set up the trap's bound variables according to SNMP v2c specs. The bridge API does not create these parameters from the corresponding fields of the `trap-parameters` structure.
- Since Javalink may establish listening sessions on a port other than specified (issue HQ-2418271), the bridge creates a temporary server socket (and closes it) to check that the specified port is free before setting up a Javalink listening session. This check, although successful in most cases, does not guarantee that this port will be available at the moment of creating a session. It means that under certain rare conditions the bridge may establish a listening session on a port other than specified in the `-listenerport` command line option.

- The bridge guarantees that no incoming PDUs will be lost at the bridge level if the file-based trap buffer was created successfully. Nevertheless, loss of PDUs may occur at the network level (depending on network or processor load), since there is no guarantee of delivery for UDP datagrams. The bridge does not guarantee that PDUs will not be lost in the AdventNet library either.
- Due to the limitations of AdventNet, the bridge cannot receive response PDUs larger than 8KB. For example, traps with about 500 integer variables will be rejected (with nologging).
- There is a special form of text values recognized by AdventNet: when constructing an OCTET STRING bound variable (for SET requests, for example), values like `'xx:xx:xx'` (apostrophes required) are treated as a sequence of hexadecimal character codes, and the resulting string will consist of characters thus encoded. This can be checked by setting a string value of `'41:42:43'` into an OCTET STRING MIB variable; the actual value will be `ABC`. This encoding does not seem to be SNMP standard and is not documented in the AdventNet documentation.
- The bridge does not support setting text values that contain non-ASCII characters into OCTET STRING variables. You can use the AdventNet hexadecimal notation (see above) for that purpose.
- the `G2SNMP_SHUTODWN` RPC should be invoked by a 'start' action, not 'call'. It does not return anything to the G2 process that started it. All connections between the bridge and any connected G2s are closed, regardless of their state.
- Loss of precision is possible for incoming MIB variables of type COUNTER64 since they are converted into a G2 float.
- to actually apply the values provided in a call to the `OXSJ-G2SNMP_MODIFY_COMM_PARAMS` RPC, you should invoke the `OXSJ-G2SNMP_USE_SNMP_COMM_PARAMS` RPC. This behavior is intentional.

Index

A
add a trap 34
authorization 11

B
blocking and non-blocking transaction requests 18
blocking SNMP transaction 23
Bridge API Reference 17
Bridge Messages 49
bridge messages 49
bridge process exit 40
bridge process, killing 12
bridge process, starting 12
bridge, exiting 14

C
Command Line Options 15
command line options 11
configuration 18
Connecting G2 to the Bridge Process 12
conversion logic 43
converted values 43
customer support services x

D
debug 16
Debug mode 16
disable filtering a trap 37

E
enterprise ID 34
Error messages 41
error messages 50
errors 49

G
G2 10
G2 messages 53
G2 Reference Manual v
G2SNMP_RECEIVE_NONBLOCKING 42, 48
G2SNMP_RECEIVE-MESSAGE receiver procedure 31
GETBULK parameters 29
GETBULK request 23

H
help 16
help message 16

I
incorrect parameter 55
Intel Pentium PC 9

J
JDK 1.3 9

K
Known Bugs and Limitations 57

L
log file 11
log file, size 15

M
maxloglines 15
message 45
message to G2 45
multiple connections to G2 14
multiple copies of the bridge 14

N

non-blocking request 42
non-blocking requests 41
non-blocking SNMP transaction 29

O

object identifiers 23
OXSJ-G2SNMP_ADD_FILTERED_TRAP 34
OXSJ-G2SNMP_BLOCKING_
TRANSACTION 19, 29
OXSJ-G2SNMP_DELETE_FILTERED_
TRAP 37
OXSJ-G2SNMP_MODIFY_COMM_
PARAMS 30
OXSJ-G2SNMP_NONBLOCKING_
TRANSACTION 25
OXSJ-G2SNMP_RECEIVE_MESSAGE 29, 45,
53
OXSJ-G2SNMP_RECEIVE_TRAP 46
OXSJ-G2SNMP_SHUTDOWN 40
OXSJ-G2SNMP_USE_SNMP_COMM_
PARAMS 31, 33

P

ppdfilename 16

R

receiver procedures 41
remote procedure calls 17
request timeout interval 32
retries 32

S

silent 16
silent command line option. 56
Silent mode 16
specific trap ID 34
starting the bridge 11
state messages 56
Sun Java Development Kit 1.3 10
Sun SparcStation 9

T

TCP port number 24, 29
trap data 48
trap filter patterns 16
trap filters 18
trap receiver process 41
type conversion messages 55

W

Warning 49
warning messages 51
Windows XP 9