

STOREADM-14.6
**Isode M-Store X.400
Administration Guide**

isode

isode and Isode are trade and service marks of Isode Limited.

All products and services mentioned in this document are identified by the trademarks or service marks of their respective companies or organizations, and Isode Limited disclaims any responsibility for specifying which marks are owned by which companies or organizations.

Isode software is © copyright Isode Limited 2002-2010, All rights reserved.

Isode software is a compilation of software of which Isode Limited is either the copyright holder or licensee.

Acquisition and use of this software and related materials for any purpose requires a written licence agreement from Isode Limited, or a written licence from an organization licensed by Isode Limited to grant such a licence.

This manual is © copyright Isode Limited 2010.

Contents

Preface	v
Chapter 1 Introduction	1
1.1 Role of a Message Store	1
1.2 Services Used by the Message Store.....	3
Chapter 2 Management	1
2.1 Message Store Server.....	1
2.2 Message Store Console.....	2
2.2.1 Monitoring Tab	3
2.3 What to Run Each Night	6
2.3.1 Backup the mailboxes	6
2.3.2 Remove old messages from the mailboxes.....	7
2.3.3 Save the old logs	7
2.4 Troubleshooting	8
2.4.1 Restoring mailboxes from a backup	8
Chapter 3 Configuration	9
3.1 Message Store Tailoring	9
3.1.1 Tailoring Options.....	9
3.2 Message Store Logging	11
3.2.1 Getting Started.....	11
3.2.2 How Logging Works	11
3.2.3 File and TTY Streams.....	14
3.2.4 System Streams	17
3.2.5 MPP Streams.....	18
3.2.6 Events Tab.....	18
3.3 Message Store Audit Logging.....	19
3.3.1 BIND Operation	19
3.3.2 SUBMIT-MESSAGE Operation	20
3.3.3 SUBMIT-MESSAGE-RESULT operation	20
3.3.4 SUBMIT-PROBE Operation.....	20
3.3.5 SUBMIT-PROBE-RESULT operation	20
3.3.6 DELIVER-MESSAGE operation	20
3.3.7 DELIVER-REPORT operation	20
3.3.8 CANCEL-DEFERRED-DELIVERY Operation	21
3.3.9 FETCH Operation.....	21

- 3.3.10 DELETE Operation 21
- 3.3.11 Other Operations 21
- 3.4 User Agent Configuration 21
- 3.5 Use of the X.500 Directory..... 21
- 3.6 Other Configuration 21

Preface

Software Version

This guide is published in support of Isode M-Store X.400 R14.6. It may also be pertinent to later releases - please consult the release notes for further details.

Readership

This Guide is intended for administrators who need to set up or manage Message Stores.

It is assumed that readers are already familiar with the contents of *GENSERV: Administrator's Guide: General Services*.

Revision History

- 3.0 reissued adding section on MHS-DS configuration and **ICAM**. Updated various other sections and diagrams.
- 10.2 rebranded and updated.
- 10.6 further updates
- 11.0 Updated to add information on new logging configuration and new modes in xmsconsole.
- 14.1 Updated for new Store Database implementation.
- 14.4 Updated in line with new release.
- 14.6 Updated to replace xmsconsole with mconsole.

Related Publications

Related topics are discussed in the volumes of the Isode documentation set listed below.

Volume	Title
SWADM-14.6	M-Switch Administration Guide

Typographical Conventions

The text of this manual uses different typefaces to identify different types of objects, such as file names and input to the system. The typeface conventions are shown in the table below.

Object	Identified by	Example
File and directory names	<i>italic</i>	<i>isoentities</i>
Program and macro names	bold	mkpasswd
Input to the system	Courier	cd newdir
Required user input	<i>italic Courier</i>	<i>filename</i>
Cross references	<i>italic bold</i>	see <i>Section 5.3.2</i>
Additional information to note, or a warning that the system could be damaged by certain action		Note Warning

In addition, text which refers to only one platform, UNIX or Windows, has markers in the margin indicating the differentiated text. An example of this can be seen in the next section.

Special Directory Names

Where directory names are given in the text, they are place holders for the names of actual directories where particular files are stored.

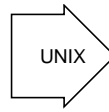
The actual directory names used depend on how the software is built and installed. All of these directories can be changed by configuration.

Directory name	Place holder for...
<i>(ETCDIR)</i>	The directory for system-specific configuration files.
<i>(SHAREDIR)</i>	The directory for configuration files that may be shared between systems.
<i>(BINDIR)</i>	The directory for programs run by users.
<i>(SBINDIR)</i>	The directory for programs run by the system administrator.
<i>(EXECDIR)</i>	The directory for programs run by other programs; for example, M-Switch channel programs.
<i>(LIBDIR)</i>	The directory for libraries.
<i>(LOGDIR)</i>	The directory for log files.
<i>(CONFPDUSPOOLDIR)</i>	The directory used for holding large PDUs on disk.
<i>(QUEDIR)</i>	The directory for the M-Switch queue.
<i>(DSADIR)</i>	The Directory Server's configuration directory.

Certain configuration files are searched for first in *(ETCDIR)* and then *(SHAREDIR)*, so local copies can override the shared information.

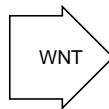
For M-Switch, the tables and other configuration files are held in a subdirectory of (*ETCDIR*) called *switch*.

The actual directories vary, depending on whether the platform is UNIX or Windows.



On UNIX, if a prefix is explicitly set in a source build, the second column applies. The third column applies to both a source release built without a specific prefix, and a binary release.

Placeholder	Source with prefix	Source without prefix/Binary
(<i>ETCDIR</i>)	<prefix>/etc	/etc/isode
(<i>SHAREDIR</i>)	<prefix>/share	/opt/isode/share
(<i>BINDIR</i>)	<prefix>/bin	/opt/isode/bin
(<i>SBINDIR</i>)	<prefix>/sbin	/opt/isode/sbin
(<i>EXECDIR</i>)	<prefix>/libexec	/opt/isode/libexec
(<i>LIBDIR</i>)	<prefix>/lib	/opt/isode/lib
(<i>LOGDIR</i>)	<prefix>/var/log	/var/isode/log
(<i>CONFPDUSPOOLDIR</i>)	<prefix>/var/tmp	/var/isode/tmp
(<i>QUEDIR</i>)	<prefix>/var/switch	/var/isode/switch
(<i>DSADIR</i>)	\$(<i>LOGDIR</i>)/../dsa-db	/var/isode/dsa-db



On Windows, the default values depend on the installation; however, if the default choices are made the values in the following table apply:

Placeholder	Source	Binary
(<i>ETCDIR</i>)	G:\isode\etc	C:\isode\etc
(<i>SHAREDIR</i>)	G:\isode\share	C:\Program Files\isode\share
(<i>BINDIR</i>)	G:\isode\bin	C:\Program Files\isode\bin
(<i>SBINDIR</i>)	G:\isode\bin	C:\Program Files\isode\bin
(<i>EXECDIR</i>)	G:\isode\bin	C:\Program Files\isode\bin
(<i>LIBDIR</i>)	G:\isode\bin	C:\Program Files\isode\bin
(<i>LOGDIR</i>)	G:\isode\log	C:\isode\var\log
(<i>CONFPDUSPOOLDIR</i>)	G:\isode\tmp	C:\isode\var\tmp
(<i>QUEDIR</i>)	G:\isode\switch	C:\isode\var\switch
(<i>DSADIR</i>)	\$(<i>LOGDIR</i>)\..\dsa-db	C:\isode\var\dsa-db

Support Queries and Bug Reporting

`customer-service@isode.com` for all account-related inquiries and issues. If customers are unsure of which list to use then they should send to this list. The list is monitored daily, and all messages will be responded to.

`license@isode.com` for all licensing related issues.

`support@isode.com` for all technical inquiries and problem reports, including documentation issues from customers with support contracts. Customers should include relevant contact details in initial calls to speed processing. Messages which are continuations of an existing call should include the call ID in the subject line. Customers without support contracts should not use this address.

`sales@isode.com` for all sales inquiries and similar communication.

Bug reports on software releases are welcomed. These may be sent by any means, but electronic mail to the support address listed above is preferred. Please send proposed fixes with the reports if possible. Any reports will be acknowledged, but further action is not guaranteed. Any changes resulting from bug reports may be included in future releases.

Isode sends release announcements and other information to the Isode News email list, which can be subscribed to from the address
<http://www.isode.com/company/news-signup.php>

Chapter 1 Introduction

1.1 Role of a Message Store

The primary role of a Message Store is to accept the delivery of messages from a Message Transfer Agent (MTA) on behalf of a user, and to retain them for subsequent retrieval by the user's User Agent (UA).

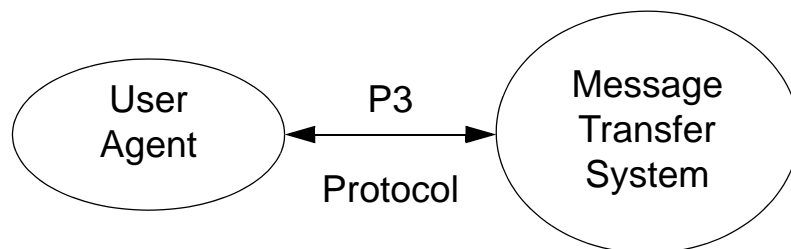
A secondary function is to provide indirect message-submission and message-administration services to the UA. This enables the Message Store to provide additional functionality to that available through direct submission to an MTA, such as forwarding messages residing in the Message Store.

A Message Store (MS) is, therefore:

- a provider of messaging services for User Agents (UAs), and
- a user of messaging services provided by Message Transfer Agents (MTAs).

In a messaging system which does not use a Message Store, outbound messages go directly and immediately from the User Agent (UA) to the Message Transfer Agent (MTA) and inbound messages go directly and immediately from the MTA to the UA, as shown in Figure 1, "Message Transfer without a Message Store," on page 1.

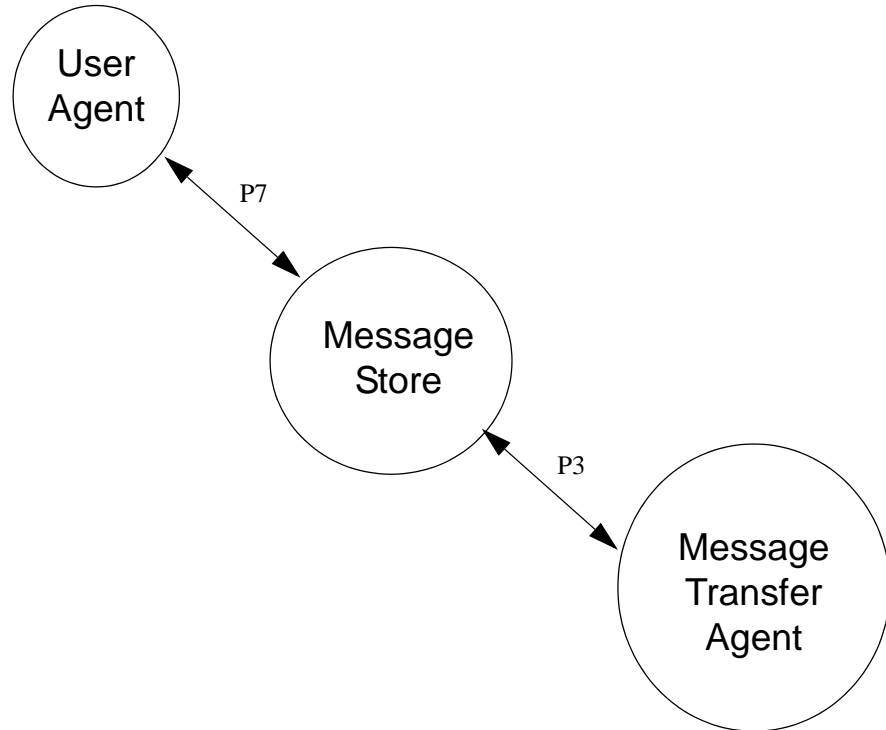
Figure 1: Message Transfer without a Message Store



In the basic CCITT model, both inbound and outbound messages are transferred using the P3 protocol. However, the actual protocol used is dependent on the message handling system that is in place and whether or not the MTA and the UA are co-resident.

When a Message Store is in use, messages between a UA and the MTA do not have to be handled immediately; they can be stored for later collection or transmission. The basic relationships for a Message Handling System which includes a Message Store are shown in Figure 2, "Message Transmission and Retrieval using a Message Store," on page 2.

Figure 2: Message Transmission and Retrieval using a Message Store



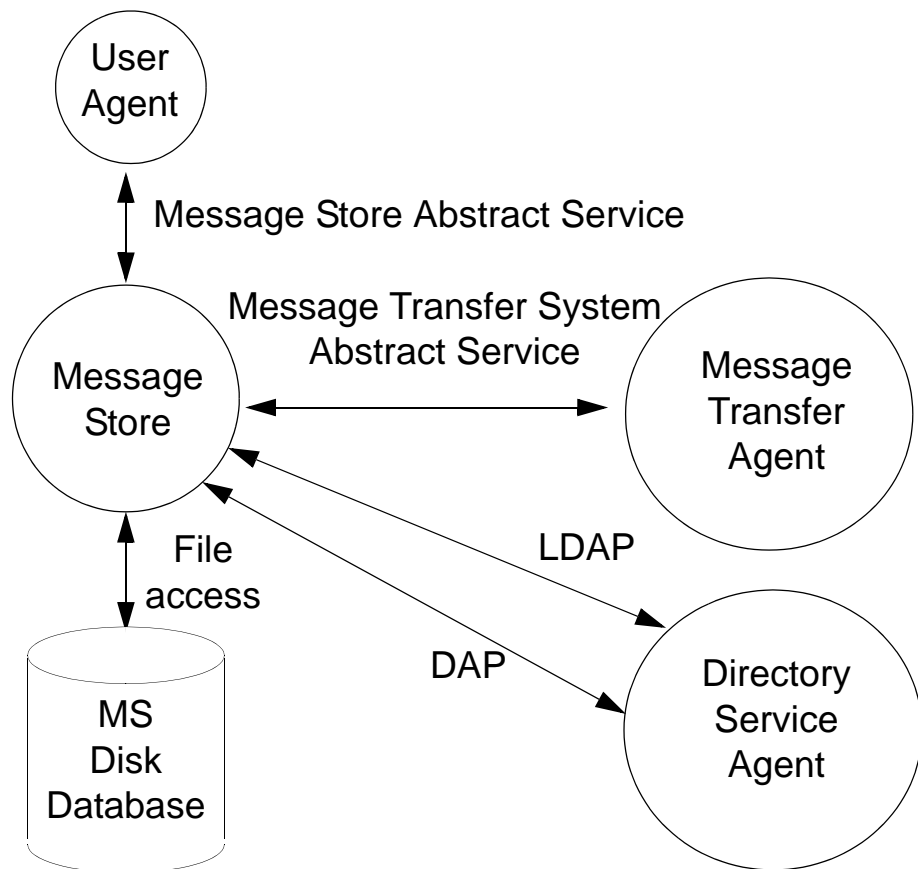
In the CCITT model for implementation of a Message Store, only one protocol is used for communication between the UA and the Message Store - the P7 protocol.

A Message Store has a one-to-one relationship with a UA; that is, a Message Store must be set up for each UA. There is no restriction on the number of Message Stores that can co-exist.

Messages to and from a UA are stored in an associated database by the Message Store until they are ready for collection or transmission. There is one Message Store database per UA. Each database forms an integral part of the Message Store.

The relationships between the Message Store implementation, its databases and the MTA and UAs are shown in Figure 1.1, "Message Transmission and Retrieval," on page 2. Refer to *Section 1.2* on page 3 for more information on the services used.

Figure 1.1: Message Transmission and Retrieval



1.2 Services Used by the Message Store

Four services are used by the Message Store:

- Message Store Abstract Service for communication with User Agents;
- Lightweight Directory Access Protocol (LDAP) for communication with the Directory for User Agent authentication.
- Directory Access Protocol (DAP) for communication with the Directory for message index access.
- Message Transfer System Abstract Service for communication with the MTA.

The first three enable the Message Store to provide services to User Agents and the fourth is used by the Message Store in the provision of those services.

In addition, the Message Store uses direct file access when manipulating messages stored on disk.

Chapter 2 Management

2.1 Message Store Server

The Message Store consists of a single server process (**isode.pumice**) which allows P7 User Agents (UAs) to access the Message Store (performing the List, Summarize, Fetch, Delete and Register-MS operations) and to (indirectly) submit messages and probes to a Message Transfer Agent (MTA). This server process is also used for remote (P3) delivery of messages and reports to the Message Store by the Message Transfer Agent. The server process also supports a completely separate management protocol (using sockets-based communication, rather than the OSI stack), which allows remote management and monitoring of the Message Store.

Each mailbox supported by the Message Store contains 'inbox' and 'outbox' folders, which are used to store delivered and (optionally) submitted messages. Each message is stored as a single binary file which includes both the envelope and content of the message.

The Message Store uses the X.500 Directory to maintain an index entry for each message within the Store database. The index entry is used to maintain the ms-entry-status attribute for each message, indicating whether it is a new message or has been listed or processed (fetched), and to provide fast and efficient searching and filtering. The Directory is also used to store various per-mailbox message counts and other items of management information across Message Store restarts. The Directory Access Protocol (DAP) is used for all of this functionality.

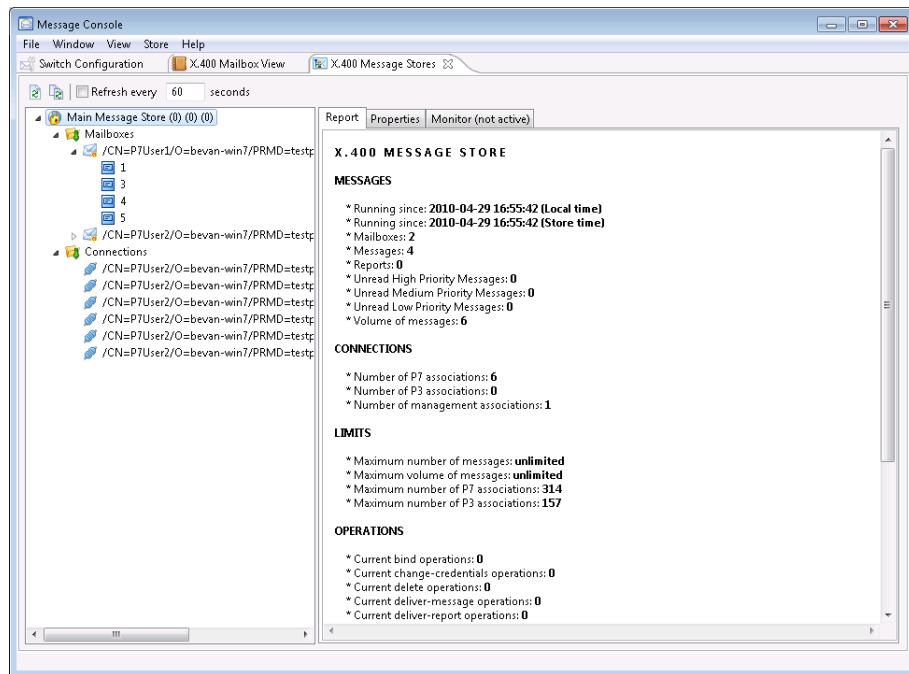
In addition, the Message Store uses the Directory for User authentication, accessing information which is set up using MConsole. Refer to the M-Switch Administration Guide for information on MConsole. This authentication information is accessed using the Lightweight Directory Access Protocol (LDAP).

Refer to *Figure 1.1* on page 2, for a schematic of the relationships.

2.2 Message Store Console

The **isode.pumice** server process provides a management and monitoring port in addition to the P7 protocol access described above. A graphical monitoring and management tool for the Message Store which accesses this port is provided as the **X.400 Message Stores** view within the **mconsole** tool. An example of this view is shown below.

Figure 2.1: Example X.400 Message Stores View



When connected to a Message Store in “unauthenticated” mode, only monitoring functions are available. These allow information about the Store as a whole, mailboxes, messages and active connections to be displayed. If an “authenticated” connection is made, specifying the Store Manager id and password values configured for the Message Store via the **Switch Configuration** view in **MConsole**, management functions become available as well. These allow:

- The Message Store to be stopped, aborted or restarted.
- Specific operations to be blocked or unblocked.
- Active connections to be disconnected.
- Mailboxes to be deleted, backed-up or restored from backup.
- Messages to be deleted, moved between mailboxes or have their status changed.

Multiple Message Stores may be monitored and managed by a single instance of **MConsole**.

2.2.1 Monitoring Tab

A **Monitor** tab is provided by **MConsole's X.400 Message Stores** view. When **monitor** mode is enabled for a Message Store (via the **Start Monitoring** menu option on the Store object), the Message Store process **MConsole** whenever a connection is opened or closed, and whenever a message is delivered, read or deleted. This allows a realtime display of the current connections and status of mailboxes to be maintained.

Figure 2.2: Monitor Tab

Store	O/R Address	Unread High	Unread Medium	Unread Low	Age of Oldest
bevan-win7	/CN=P7User1/O=bevan-win7/PRMD=test	0	0	0	
bevan-win7	/CN=P7User2/O=bevan-win7/PRMD=test	0	0	0	

Store	Connection ID	Type	Direction	O/R Address	Last Operation
bevan-win7	34072	p7	inbound	/CN=P7User2/O=bevan-win7/PRMD=test	bind
bevan-win7	34112	p7	inbound	/CN=P7User2/O=bevan-win7/PRMD=test	bind
bevan-win7	34008	p7	inbound	/CN=P7User2/O=bevan-win7/PRMD=test	bind
bevan-win7	33972	p7	inbound	/CN=P7User2/O=bevan-win7/PRMD=test	bind
bevan-win7	34000	p7	inbound	/CN=P7User2/O=bevan-win7/PRMD=test	bind
bevan-win7	33700	p7	inbound	/CN=P7User2/O=bevan-win7/PRMD=test	bind

2.2.1.1 Registering Autoactions

Autoactions are a feature of the service provided by an X.400 Message Store to clients through the P7 protocol. Isode release 11.2 and onwards added a range of management features around autoactions to enable Isode customers to make effective use of this feature.

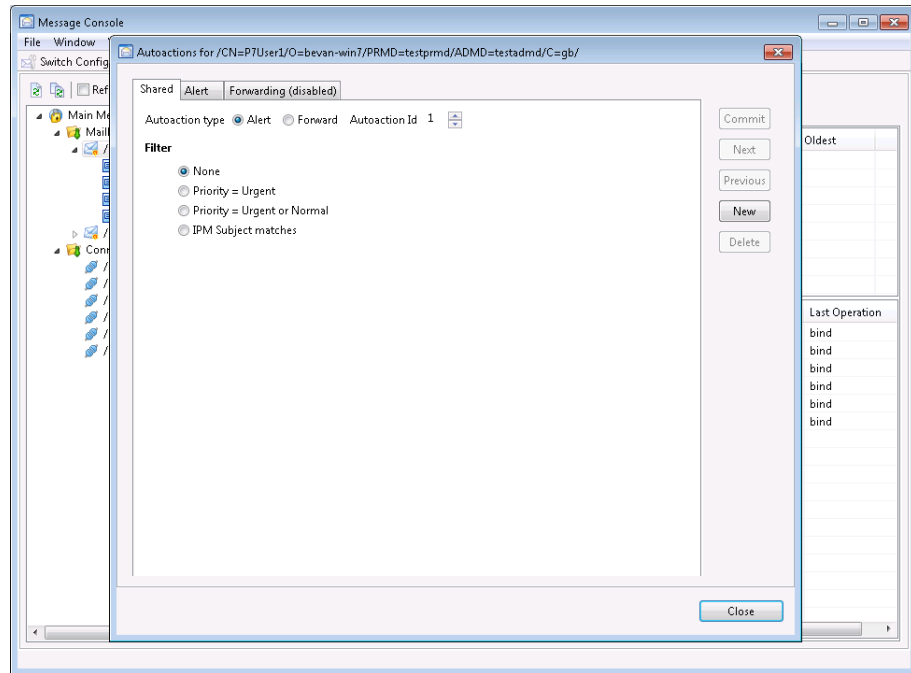
There are two autoactions that can be configured by the **X.400 Message Stores** view within **MConsole**: **AutoForwarding** and **AutoAlert**.

AutoForwarding

The **AutoForward** action forwards a delivered message to one or more recipients specified in the autoaction, and marks the message as **AutoForwarded** (an X.400 feature). It may be set to include a cover note. There is an option in **AutoForward** to delete the message after it is **AutoForwarded**. **AutoForwarding** is useful to ensure that messages get processed when a user is away for a period. It can also be helpful to distribute messages to multiple recipients (essentially an alternate approach to using distribution lists), which allows more user control. This will be desirable for some AMHS users, as this model reflects the way that a lot of AFTN systems operate.

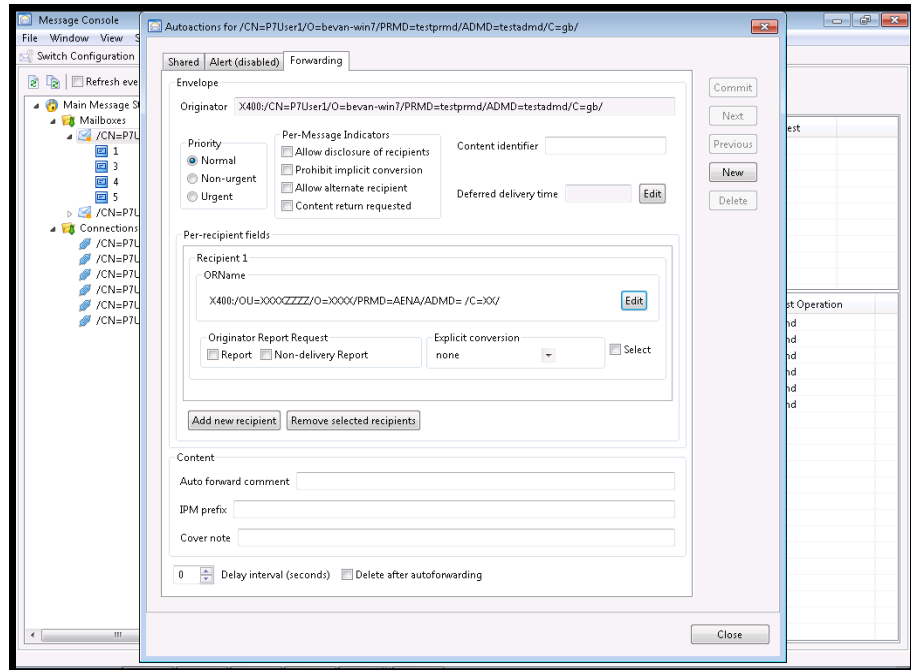
To set it in **MConsole**, connect to a Message Store in authenticated mode, select the mailbox user that you want to edit, right click on it and select *Configure autoactions*. A window similar to the one illustrated in **Figure 2.3** will appear.

Figure 2.3: Configuring Autoactions



Click on the *New* button and then change the radiobutton to *Autoforwarding*. The *AutoForwarding* tab will now be activated. If you selected, you will see the window illustrated in **Figure 2.4**

Figure 2.4: Configuring AutoForwarding



Then click on *Add new recipient* and enter the O/R address of the user which will receive the forwarded messages in the Address field.

There are many options that you can set for the new message that is generated, like setting its Priority, Content Identifier and Per-Message Indicators.

Don't forget to click on *Commit* to save the changes that you have made.

Delayed auto-forward

A variant of AutoForward, which was defined for the US Defense Messaging System, is to perform the AutoForward in the event that the message has not been fetched from the message store after a defined period (i.e., has not been read by the intended recipient). This mechanism can be used to help ensure that messages get read promptly. The filter mechanism can be useful here, for example to have a shorter AutoForward delay for urgent messages.

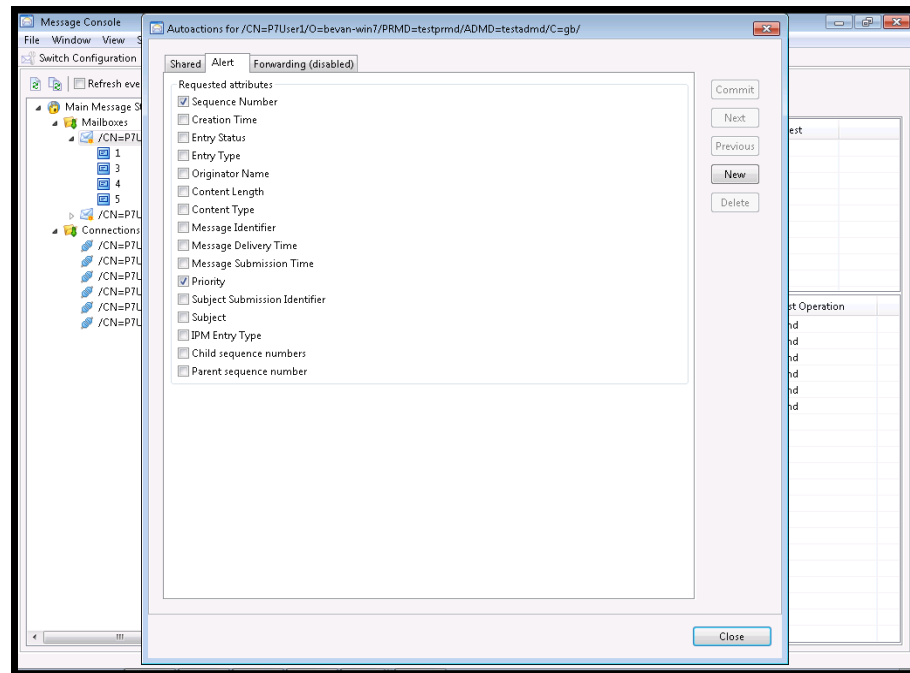
AutoAlerts

Most P7 operations are client initiated, and normal P7 operation is for the client to poll the server at intervals to check for new messages. The AutoAlert autoaction causes an alert operation to be sent from the server to the client, which provides information on a newly arrived message. Use of AutoAlert allows a client to get new messages more quickly, and improve efficiency by reducing the frequency of polling the store.

AutoAlert autoactions are normally registered by the User Agent, but they can also be registered manually using **MConsole**.

To register the AutoAlert autoaction, click on the *New* button and then select the AutoAlert tab, you will see the window illustrated in **Figure 2.5**

Figure 2.5: Configuring AutoAlert



You can choose which attributes to request in the alert. The ability to configure the attributes requested in an alert is currently not supported by the X.400 Client API's RegisterMS interface, so until this is available, you can configure the attribute set using **MConsole** if required.

Don't forget to click on *Commit* to save the changes that you have made.

2.3 What to Run Each Night

Here are some procedures things you should perform regularly, ideally each night.

2.3.1 Backup the mailboxes

You should regularly back up the mailboxes to another disk, or even to another system.

The script `(SBINDIR)/mstore_backup.tcl` does the backup of one or all the mailboxes, making sure that there are no conflicts even if a message is about to be delivered to or deleted from a mailbox. This script needs to contact the Message Store, so the service needs to be running. If the Message Store is not running, then you can copy the mailboxes simply with `cp -pr` or a similar command.

`(SBINDIR)mstore_backup.tcl`

```
Usage: mstore_backup.tcl -user <user> -password <password> -
destination <dest>
    [-all | -mailbox <mbx>] [-host <host>] [-port <port>]
    [-verbose]
    <host> defaults to localhost, <port> to 3002
-host <host> The host name of the machine where the Message Store is running
-port <port> The port number which the Message Store is using for management
connections, it defaults to 3002.
-user <user> The user name to use to connect to the Message Store management.
Usually pp or msadmin.
-password <password> The password to use to authenticate the above user.
-verbose Output more information about the mailboxes that are being backed up.
-destination <dest> The destination directory under which all the mailboxes
will be backed up. It must exist.
-all Choose to back up all the mailboxes.
-mailbox <mbx> Choose to back up only the mailbox associated with a specific
O/R address. You can specify the -mailbox flag multiple times on the command
line, to select several mailboxes for backup
```

2.3.2 Remove old messages from the mailboxes

Another important task that should to be performed regularly is removing old messages from the Message Store mailboxes. This not only prevents filling up the disk, but it also allows the store to perform delivery more efficiently.

The script (*SBINDIR*)/*mstore_tidy.tcl* can be run to delete messages that are older than a certain age (in days). This script needs to contact the Message Store, so the service needs to be running.

```
Usage: mstore_tidy.tcl [-host <host>] [-port <port>] [-age
<age-in-days>] -user <user> -password <password> [-verbose]
-host <host> The host name of the machine where the Message Store is running.
Defaults to localhost.
-port <port> The port number which the Message Store is using for management
connections, it defaults to 3002.
-user <user> The user name to use to connect to the Message Store management.
Usually pp or msadmin.
-password <password> The password to use to authenticate the above user.
-verbose Output more information about the mailboxes that are being backed up.
-age <days> The maximum age in days of messages to leave in the mailbox. All
older messages will be deleted. Defaults to 7 days.
```

2.3.3 Save the old logs

Saving logs depends very much on the individual policies for each site. If you are concerned about keeping the information, you should backup the logs each night. In

any case, after a sensible period (e.g. a week) logs should be removed from the (*LOGDIR*) directory.

The xms-audit log output can be processed to show statistics.

2.4 Troubleshooting

2.4.1 Restoring mailboxes from a backup

The script (*SBINDIR*)/*mstore_backup.tcl* restores one or all the mailboxes, making sure that there are no conflicts even if a message is about to be delivered to or deleted from a mailbox.

This script needs to contact the Message Store, so the service needs to be running.

(*SBINDIR*)/*mstore_restore.tcl*

Usage: `./mstore_restore.tcl -user <user> -password <password> [-host <host>]`

`[-message <msg> | -source <src>] [-port <port>]`

`[-all | -mailbox <mbx>] [-verbose]`

`-host <host>` The host name of the machine where the Message Store is running. Defaults to localhost.

`-port <port>` The port number which the Message Store is using for management connections, it defaults to 3002.

`-user <user>` The user name to use to connect to the Message Store management. Usually *pp* or *msadmin*.

`-password <password>` The password to use to authenticate the above user.

`-verbose` Output more information about the mailboxes that are being restored.

`-source <src>` The source directory under which all the mailboxes are backed up.

`-message <msg>` The specific message that has to be restored.

`-all` Choose to restore all the mailboxes

`-mailbox <mbx>` Choose to restore only the mailbox associated with a specific O/R address. You can specify the `-mailbox` flag multiple times on the command line, to select several mailboxes for restoration.

Chapter 3 Configuration

3.1 Message Store Tailoring

A tailoring file, *pumicetailor.xml*, located in (*ETCDIR*), provides the configuration information that the **isode.pumice** process need to be able to access the Directory. An example file is shown in *Figure 3.1, Example pumicetailor.xml File*, on page 10. In releases prior to R14.6, this file was called *pumicetailor*, and was structured as key:value pairs. The R14.6 **isode.pumice** will still read this old style of configuration file if found.

Configuration of various operational aspects of the Message Store is held in the Directory, and can be modified using **MConsole**. This is described in the M-Switch Administration Guide. **MConsole** can also be used to create *pumicetailor.xml* files which contain the correct connection information for individual Message Stores.

Section 3.1.1, below, describes each component of the file.

3.1.1 Tailoring Options

The tailoring options fall into two groups: those which are always applicable and those which apply to standard Isode tailoring variables.

3.1.1.1 General Tailoring

bind_name: This holds the Directory Name with which the **isode.pumice** process will bind to the Directory.

bind_password: OPTIONAL. The password which the **isode.pumice** process will use when binding to the Directory. An anonymous bind will be attempted if no value is supplied.

p7server_name: This holds the Shared Message Store's own Directory Name. This will normally be the same as the value for **bind_name**.

3.1.1.2 Isode Tailoring

Isode base library general variable tailoring is available from the Message Store's tailoring file as follows:

isode: allows any Isode tailoring variable to be set. Overrides any value configured in the *isotailor* file. For example, the tailoring line:
"isode: etcpath /var/etc" would set the Isode tailoring variable **etcpath** to value `"/var/etc"`.

Three specific Isode tailoring variables must always be set:

```
isode:mhsds_x500_access ldap
```

This configures the Message Store to use LDAP access for its User Authentication lookup. This setting should not be changed, or the Store will not function correctly.

```
isode:mhsds_ldap_port 19389
```

This configures the port on which the Message Store will attempt to contact the LDAP server. You will need to modify this if your LDAP server is listening on a different port.

```
isode:mhsds_ldap_host localhost
```

This configures the hostname on which to contact the LDAP server. You will need to modify this if your LDAP server is on a different system from your Message Store.

Figure 3.1: Example pumicetailor.xml File

```
<pumicetailor>
<servpass:info service="isode.pumice"/>
<bind_name>&lt;cn=Main Message Store,cn=Messaging
Configuration,o=Isode Limited,c=GB&gt;</bind_name>
<bind_password servpass:encrypt="true">secret</bind_password>
<p7server_name>&lt;cn=Main Message Store,cn=Messaging
Configuration,o=Isode Limited,c=GB&gt;</p7server_name>
<isode name="mhsds_x500_access">ldap</isode>
<isode name="mhsds_ldap_host">localhost</isode>
<isode name="mhsds_ldap_port">19389</isode>
</pumicetailor>
```

3.2 Message Store Logging

This section gives an overview of the general structure of the Isode logging subsystem, and a description of the GUI which is provided to enable configuration of the Message Store's use of this subsystem.

3.2.1 Getting Started

If you wish to modify the default logging settings for the Message Store, you should do the following:

Copy the file *xmslogging.xml* from (*SHAREDIR*) into (*ETCDIR*).

On Windows, a shortcut to the **Log Configuration Tool** will have been set up in the **Isode** folder on your Start menu.

On Unix, you should run `/opt/isode/sbin/logconfig`.

Once the GUI is running, open *xmslogging.xml* from (*ETCDIR*). You will see a display of a number of predefined logging streams used by the Message Store, which can be modified as required. For full details of the options available, see *File and TTY Streams* on page 14.

3.2.2 How Logging Works

3.2.2.1 Record types

Isode server programs (like the **isode.pumice** process) write two types of log records during normal execution - Audit records and Event records.

Audit records are used to record "auditable events" - message submission, for example. They do not have a severity level associated with them, and have a well-defined format, so that they can be easily parsed. Audit records normally consist of an event-type indicator, followed by a list of key=value pairs.

Event records are used to record errors, normal program operation, or to provide debugging information. They are associated with a particular severity level, and contain freeform text with substituted data items. The freeform text is contained in a separate dynamically-loaded library (on Windows) or a message catalog (on Unix), which makes it possible to replace the standard set of English messages with equivalent text in other languages simply by substituting a suitable message file..

No output mechanism is directly associated with log records. When an event or audit record is generated by an application, then whether or not it is logged, where it is logged to, and what the output of the log looks like, depends on what *output streams* have been configured.

3.2.2.2 Output Streams

An output stream is a description of how a particular set of event and audit records should be recorded or displayed. Multiple output streams may be configured for an

application, and whenever an event or audit record is generated, the logging subsystem checks to see which, if any, of the available output streams is eligible to process it.

As well as defining which records are eligible to be logged, the configuration of an output stream also determines the format of the messages that are produced by the stream.

This means that a single event or audit record may be processed by one or more separate streams (or by no stream at all), and that, in the case of multiple streams, the messages output by the streams may be of differing formats, containing more or less detail. For example, it would be possible to configure one output stream to generate a brief message about all "warning" level events, and another to generate a detailed message about a specific "warning" event which is of particular interest.

Four stream types are currently available: the *file* type, where the records are output to a file, the *system* type, where the records are passed to the system event log (syslog on Unix-type systems and the Application Event Log on Windows), the *tty* type, which is identical to *file* type, except that the records are written to either stdout or stderr, and the *mpp* type, which sends records to the Isode Server Watch Daemon using either a Unix named pipe or via TCP.

3.2.2.3 Configuration Storage and Loading

Information about output stream configuration is stored as XML. All Isode applications will load the XML contained in the file *logtailor.xml*, located in (*ETCDIR*) or (*SHAREDIR*), if it exists, at startup. This filename and location can be overridden if required by defining the environment variable **LOGTAILOR** to be an alternative filename or filepath.

An application may then load a private stream configuration. In the case of the Message Store, this is contained in the *xmslogging.xml* file. A default version of this file is located in (*SHAREDIR*) - if you wish to make changes, copy this file into (*ETCDIR*) and modify this version. If the configuration file exists in both (*ETCDIR*) and (*SHAREDIR*), the version in (*ETCDIR*) will be used.

Format of Messages in Output Streams

When a given audit or event is generated, then for each output stream that is configured to process records of that type, the settings for the output stream determine the format of the message that is output. In the case of *file* and *tty* streams, the stream may be configured to contain any combination (including none) of the following fields:

- **date and time**
The format of date and time is configurable on a per-stream basis
- **program name**
The name of the program generating the message. Any "isode" prefix will have been removed, and the program name will be truncated to 8 characters
- **process id**
- **thread id**
This field may be useful to distinguish separate threads in the same process
- **username**
The username of the process which generated the record. This field is only meaningful on Unix systems. If the username cannot be established, then a numeric UID is logged.

- **severity**

Audit records have no associated severity, but event records always have a severity, which, if displayed, is represented using one of the following single letters, as follows:

 - I** - Info
 - N** - Notice
 - S** - Success
 - D** - Detail
 - W** - Warning
 - E** - Error
 - F** - Fatal
 - C** - Critical
 - L** - AuthOK
 - A** - Authfail
 - X** - Debug
 - P** - PDU

- **facility code**

The name of the facility which generated the message. Audit records are not associated with a particular facility.
- **message identifier**

An identifier representing the event. Audit records do not have a message identifier
- **text**

The formatted text describing this event. Audit records do not have a text field
- **supplementary audit record parameters**

For certain types of audit records, extra information may be associated with the record, and if the stream is suitably configured, this will be included as a sequence of "key:value" pairs on the end of the message.

3.2.3 File and TTY Streams

For *file* and *tty* streams, the following parameters can be configured:

3.2.3.1 File Tab

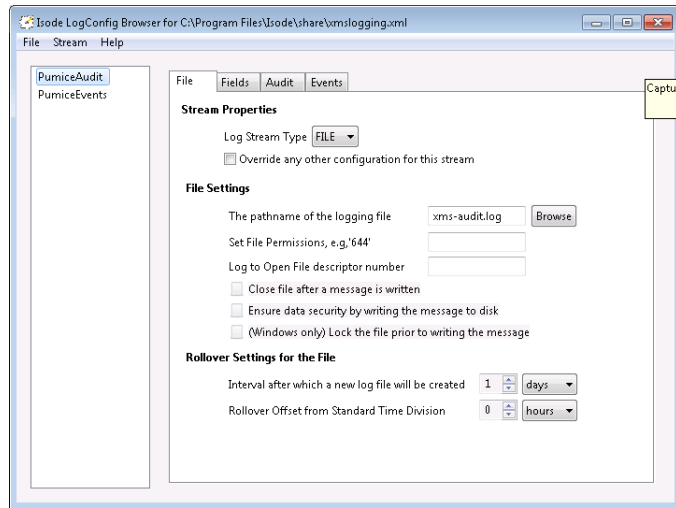


Figure 3.2: File Tab

This tab allows details of the file to which the stream writes to be specified.

- **Log stream type:** This selects whether the stream outputs to a log file or TTY (**stderr** or **stdout**). If TTY is selected, only the “Standard System Streams” field described below is displayed.
- **Override:** If set, any other configuration for this stream, including which events are logged, is overridden. This will probably not be of any relevance to Message Store logging.
- **Standard System Streams:** selects whether the logging output will go to **stderr** or **stdout**.
- **Pathname:** The pathname of the logging file, if this is a *file* stream. Note that if you have rollover configured (see below) the actual filename will in part be derived from the rollover interval.
- **File Permissions:** By default, log files are created so that any process can write to them. If different process owners are writing to the file, then this is necessary. However, the file mode, in octal, can be configured here. To set the value so that only the creator can write to it, but others can read, for example, the value set should be '644'.
- **Log to Open File:** This enables you to log to an open file descriptor. The integer value of the file descriptor is set in this field.
- **Close file after a message is written:** The file is opened for each message and then closed after the message is written. This can be used to ensure that the data is secure but there is a significant performance penalty.
- **Ensure data security:** The operating system is asked to ensure that the message is written to disk. This can be used to ensure that the data is secure, and there is some performance penalty.

- **Lock file prior to writing:** (Windows only) The file is locked prior to writing the message, and then unlocked afterwards. This ensures that when multiple processes are logging to the same file that the messages are not mixed. It is only used on Windows, and the default is to lock.
- **Rollover settings:** You can configure file logging so that a new logfile will be created at regular intervals. This may be useful, for example, in the event that you wish to purge old logfiles selectively. You configure rollover by specifying:

rollover interval: This control specifies how frequently a new file should be created. Note that the name of the generated logfile will include the date and time at which it was created (for example "mta-event.2005-06-13-00-00.log").

rollover offset: Normally the time interval for rollover coincides with a standard time division. E.g. midnight for intervals of a day or more. The offset enables you to move this start time.

3.2.3.2 Fields Tab

This tab allows control of the format of the common elements of each record logged.

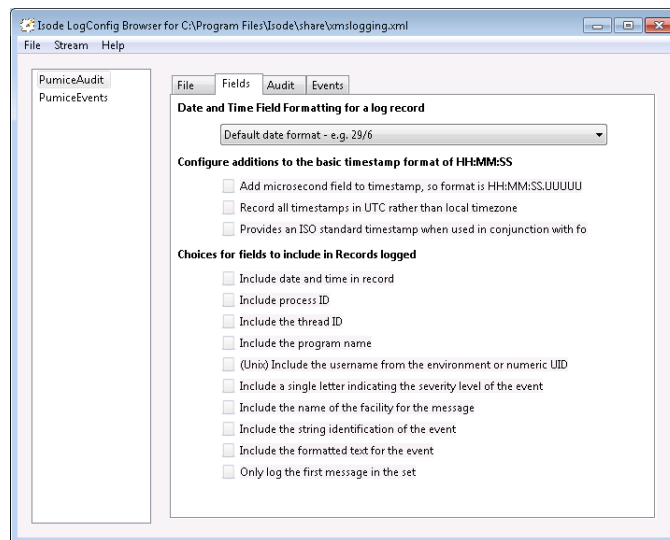


Figure 3.3: Fields Tab

- **Date options:** Configure how the date element of the time-and-date field at the start of each log record is formatted. Available choices are:
 - default:** This gives a date format which is the same as in pre-R11 releases - e.g. " 2/ 8 14:35:06".
 - none:** No time or date field
 - month:** Day of month and month number, i.e. "DD MM"
 - year (2):** two-digit year, month and day in YY-MM-DD format
 - year (4):** as above, but four-digit year, i.e. "YYYY-MM-DD"
- **Time options:** Configure additions to the basic timestamp format of HH:MM:SS. Choices are:
 - usec:** Add microsecond field to timestamp, so format is HH:MM:SS.UUUUU
 - utc:** Record all timestamps in UTC rather than local timezone. There is no difference in output format when this option is set.

tsep: When used in conjunction with the four-digit year option, provides an ISO standard timestamp, with the date and time fields separated by a "T" character rather than the default of single space, i.e. "12:34:10T2005-06-13".

- message fields: Configure what fields are included in records logged to a file stream. The default setting is for all fields to be included.

time and date: include time and date in log record.

process ID: Include process ID.

threadid: Include the thread ID. This enables you to distinguish events logged by different threads within the process.

progname: Include the program name. Any "isode" prefix is removed, and the program name is truncated to 8 characters

username: (Unix) include the username from the environment. If not available then the numeric UID is logged.

severity: Include a single letter indicating the severity level of the event. Letters are:

S - Success

X - Debug

P - PDU

D - Detail

I - Info

N - Notice

L - AuthOK

W - Warning

E - Error

F - Fatal

C - Critical

A - Authfail

facility: Include the name of the facility for the message.

ident: Include the string identification of the event.

text: Include the formatted text for the event

firstonly: If a message set is being logged, only log the first message in the set

3.2.3.3 Audit Tab

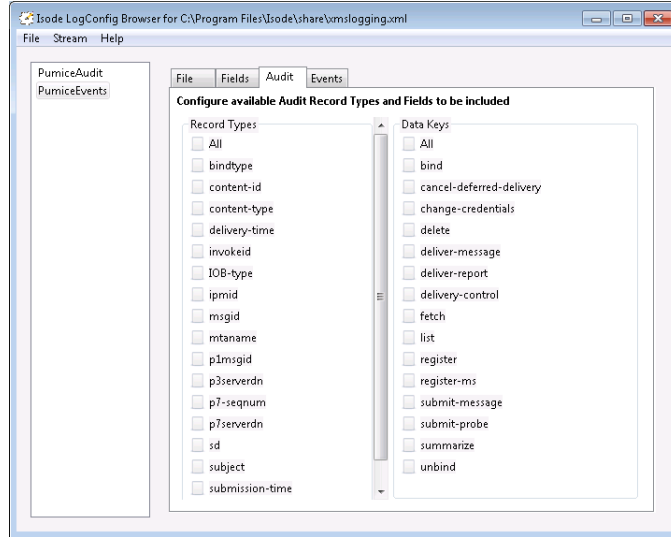


Figure 3.4: Audit Tab

This tab configures which of the available audit record types are to be sent to this stream, and which fields in the records are to be included. The default is for all records and all fields to be included.

3.2.4 System Streams

For *system* streams, the following parameters can be configured:

3.2.4.1 Properties Tab

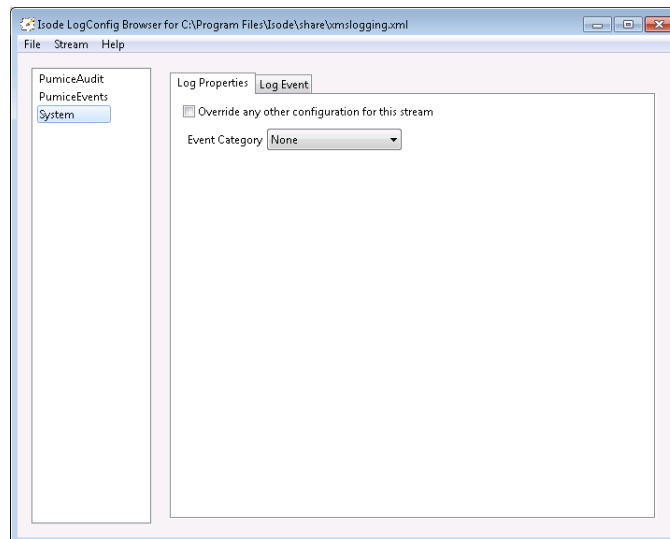


Figure 3.5: Properties Tab

- **override:** If set, any other configuration for this stream, including which events are logged, is overridden. This is particularly useful for program or channel specific logging.

- windows event log category: configures which of the predefined event categories will be used for the event log.
- syslog config: (only displayed on Unix systems) this allows control over various aspects of messages written to syslog. Available options are:
 - console: Write directly to system console if there is an error while sending to system logger.
 - stderr: Print to stderr as well.
 - pid: Include process ID.
 - severity: Include a single letter indicating the severity level of the event.
 - facility: Include the name of the facility for the message.
 - ident: Include the string identification of the event.
 - text: Include the formatted text for the event.
 - firstonly: If a message set is being logged, only log the first message in the set.
- syslog facility: (only displayed on Unix systems) the facility which should be used to log events

3.2.5 MPP Streams

For *mpp* streams, the only configuration items are:

- host: configures the hostname to which events will be sent (on Windows) or the named pipe to which events will be sent (on Unix).
- port: on Windows only, configures the port to which events will be sent.

3.2.6 Events Tab

This tab is common to all types of stream, and enables you to configure which event records are to be sent to this stream. Configuration is done by facility. However, the 'All' facility can be used to set which severity level are logged for all facilities, by default. This setting can then be overridden for each facility. In addition individual messages can be configured to be logged or not logged. Note that *pdu* and *debug* level logging is not available for *system* streams. The example shown below has Critical and Fatal level events configured to be logged for all facilities, and Error, Warning, AuthFail and AuthOK level events to be logged for the Dsap facility only.

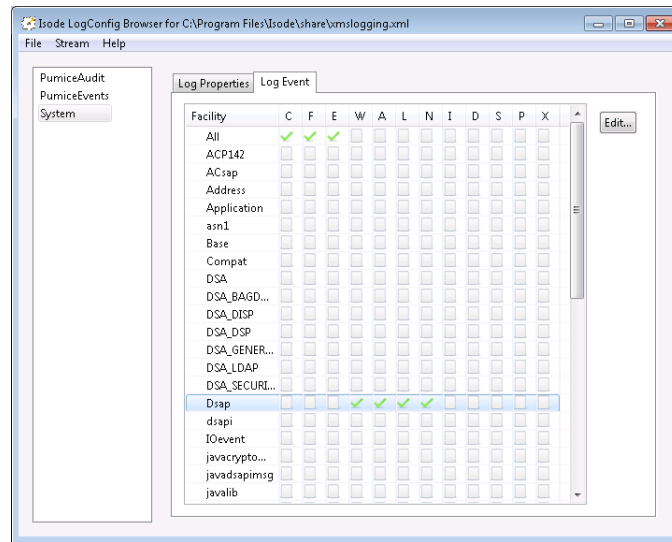


Figure 3.6: Event Configuration Tab

3.3 Message Store Audit Logging

An Audit record is generated for each operation performed by the Message Store. Each record consists of:

- A keyword which identifies the operation being performed. These are: BIND, LIST, SUMMARIZE, FETCH, DELETE, REGISTER-MS, SUBMIT-MESSAGE, SUBMIT-MESSAGE-RESULT, SUBMIT-PROBE, SUBMIT_PROBE_RESULT, CANCEL-DEFERRED-DELIVERY, DELIVER-MESSAGE, DELIVER-REPORT, DELIVERY-CONTROL, CHANGE-CREDENTIALS, REGISTER and UNBIND.
- The identifier of the association performing the operation.
- A sequence of key=value pairs which is specific to the type of operation being recorded. These are described below.

3.3.1 BIND Operation

- bindtype: identifies type of bind operation: value can be “to MTA” “by MTA” and “by UA”.
- p7serverdn: the DistinguishedName of the Message Store.
- p3serverdn: the DistinguishedName of the p3server entity to which the Store is binding, when bindtype = “to MTA”.
- mtaname: the name with which the MTA is binding to the Message Store.
- uaname: the ORName with which the UA is binding to the Message Store.

3.3.2 SUBMIT-MESSAGE Operation

- **invokeid:** an identifier for the operation being invoked on the MTA - this allows correlation with a SUBMIT-MESSAGE-RESULT audit record.
- **content-type:** the content type of the message being submitted.
- **content-id:** the content-id from the message submission envelope, if present.
- **IOB-type:** For P2/P22 content types, indicates whether an IPM or IPN is being submitted.
- **subject:** For IPMs, the subject field from the IPM heading.
- **ipmid:** For IPMs, the this-IPM field from the heading. For IPNs, this will be the subject-ipm field.

3.3.3 SUBMIT-MESSAGE-RESULT operation

- **invokeid:** an identifier for the operation which was invoked on the MTA - this allows correlation with a SUBMIT-MESSAGE audit record.
- **p1msgid:** the MTSIdentifier from the submission result.
- **submission-time:** the time at which the message was submitted.
- **content-id:** the content-id from the message submission envelope, if present.

3.3.4 SUBMIT-PROBE Operation

- **invokeid:** an identifier for the operation being invoked on the MTA - this allows correlation with a SUBMIT-PROBE-RESULT audit record.
- **content-type:** the content type of the probe being submitted.
- **content-id:** the content-id from the probe submission envelope, if present.

3.3.5 SUBMIT-PROBE-RESULT operation

- **invokeid:** an identifier for the operation which was invoked on the MTA - this allows correlation with a SUBMIT-PROBE audit record.
- **p1msgid:** the MTSIdentifier from the submission result.
- **submission-time:** the time at which the probe was submitted.
- **content-id:** the content-id from the probe submission envelope, if present.

3.3.6 DELIVER-MESSAGE operation

- **p1msgid:** the MTSIdentifier of the message being delivered
- **delivery-time:** the time at which the message was delivered.
- **p7-seqnum:** the P7 sequence number assigned to the message after delivery.

3.3.7 DELIVER-REPORT operation

- **subjectmsgid:** the MTSIdentifier of the message to which the report being delivered pertains.
- **content-id:** the content-identifier field from the report.
- **p7-seqnum:** the P7 sequence number assigned to the report after delivery.

3.3.8 CANCEL-DEFERRED-DELIVERY Operation

- msgid: the MTSIdentifier of the message whose delivery is being cancelled.

3.3.9 FETCH Operation

- p7-seqnum: the P7 sequence number of the message being fetched.

3.3.10 DELETE Operation

- p7-seqnum: the P7 sequence number of the message being deleted.

3.3.11 Other Operations

None of the other operations generate operation-specific key=value pairs.

3.4 User Agent Configuration

Your User Agent will need to be configured with the Presentation Address on which the **isode.pumice** process is listening. The default value for this is "3001"/Internet=<IPaddr>+3001 - i.e. the **isode.pumice** process will listen on port 3001 for incoming connections which specify a Transport Selector of "3001" (text). Note that the Message Store will only listen on the specific IP address for which it is configured.

3.5 Use of the X.500 Directory

The Message Store uses the X.500 Directory to authenticate and configure users. This is based on work in *RFC 1836: Representing the O/R Address Hierarchy in the X.500 Directory Information Tree* and *RFC 1801: MHS use of Directory to Support MHS Routing*, which allows configuration of Message Stores, Message Store Users and the protocol links to and from MTAs, using a specially designed Directory User Agent. The Lightweight Directory Access Protocol (LDAP) is used by the Message Store to access the Directory for authentication.

The Message Store also uses the X.500 Directory to provide its message indexing service. For this access it uses Directory Access Protocol (DAP).

3.6 Other Configuration

The directory specified as a containing user's root folder must exist prior to Message Store use and must be writable by the Message Store process. The Message Store will create individual users' root folders and subordinate folders as required.

References

Refer to the documents listed in this appendix for more information on issues raised or referred to in Isode documentation. Where specific documents can be obtained electronically, this is shown in the reference; otherwise refer to *Obtaining Documents*, on page 25.

Cited Documents

Refer to the documents listed below for more information regarding issues raised or referred to in this Guide.

ITU-T Recommendation X.400 | ISO/IEC 10021-1:1990: Information Technology - Text Communication - Message-Oriented Text Interchange System (MOTIS) - System and Service Overview.

ITU-T Recommendation X.402 | ISO/IEC 10021-2:1990: Information Technology - Text Communication - Message-Oriented Text Interchange System (MOTIS) - Overall Architecture.

ITU-T Recommendation X.413 | ISO/IEC 10021-5:1990: Information Technology - Text Communication - Message-Oriented Text Interchange System (MOTIS) - Message Store: Abstract Service Definition.

ITU-T Recommendation X.419 | ISO/IEC 10021-6:1990: Information Technology - Text Communication - Message-Oriented Text Interchange System (MOTIS) - Protocol Specifications.

ITU-T Recommendations Series X.500 | ISO/IEC 9594-1: 1993: Information Technology - Open Systems Interconnection - The Directory: Overview of Concepts, Models, and Services.

RFC 1278: A String Encoding of Presentation Address; S. Kille

RFC 1327: Mapping between X.400(1988)/ISO 10021 and RFC822; S. Kille

RFC 1777: X.500 Lightweight Directory Access Protocol; W. Yeong, T. Howes, & S. Kille

RFC 1779: A String Representation of Distinguished Names. S. Kille

RFC 1801: MHS use of Directory to Support MHS Routing; S.Kille

RFC 1836: Representing the O/R Address Hierarchy in the X.500 Directory Information Tree; S.Kille.

Other Publications

- Chadwick, D. W. *Understanding X.500 (The Directory)*. International Thompson Publishing, July 1996. ISBN 1-85032-281-3.
- Gardner, Ella and Ginsburg, Elliot. *Defense Message System Unclassified Directory Schema*. Mitre Corporation, Washington C3 Center. 5 February 1993.
- National Institute of Standards and Technology. *Announcing the Data Encryption Standard (DES)*. Federal Information Processing Standards Publication 46-2, Dec. 1993.
- National Institute of Standards and Technology. *Digital Signature Standard (DSS)*. Federal Information Processing Standards Publication 186, May 1994.
- National Institute of Standards and Technology. *Secure Hash Standard*. Federal Information Processing Standards Publication 180, May 1993.
- National Security Agency. *SDNS Directory Specifications for Utilization with SDNS Message Security Protocol*. Specification SDN.702, 8 December 1992. Revision 2.0.
- North American Directory Forum. *SD-5: An X.500 Naming Scheme for National DIT Subtrees and its Application for c=CA and c= US*.
- OIW Implementor's Workshop. *Stable Implementation Agreements for Open Systems Interconnection Protocols: Part 12 - OS Security*. September 1994.
ftp://nemo.ncsl.nist.gov/pub/oiw/agreements/12S_9409.ps
- Ousterhout, J. *An X11 Toolkit Based on the Tcl Language*. Winter 1991 USENIX Conference Proceedings.
<ftp://ftp.scriptics.com/pub/tcl/doc/tkUsenix91.ps>
- Ousterhout, J. *Tcl: An Embeddable Command Language*. Winter 1990 USENIX Conference Proceedings.
<ftp://ftp.scriptics.com/pub/tcl/doc/tclUsenix90.ps>
- Ousterhout, J. *Tcl and the Tk Toolkit*. Addison-Wesley, 1994. ISBN 0-201-63337-X.
- Roe, M. *PASSWORD R2.5: Certification Authority Requirements*. Nov. 1992.
<ftp://cs.ucl.ac.uk/password/r25.ps>
- Rose, M. *The Little Black Book: Mail Bonding with OSI Directory Services*. Prentice-Hall, 1991. ISBN 0-13-683219-5.
- RSA Data Security Inc. *PKCS #1: RSA Encryption Standard*. Nov. 1993.
- RSA Data Security Inc. *PKCS #6: Extended-Certificate Syntax Standard*. Nov. 1993.
- RSA Data Security Inc. *PKCS #7: Cryptographic Message Syntax Standard*. Nov. 1993.
- RSA Data Security Inc. *PKCS #8: Private-Key Information Syntax Standard*. Nov. 1993.
- RSA Data Security Inc. *PKCS #9: Selected Attribute Types*. Nov. 1993.
- Stonebraker, M. and Kemnitz, G. *The POSTGRES next-generation database management system*. Communications of the ACM, Oct. 1991, vol. 34,

(no. 10): 78-92.

<http://s2k-ftp.cs.berkeley.edu:8000/postgres/papers/ERL-M91-62.ps.Z>

versit Consortium. *vCard. The Electronic Business Card Version 2.1*. September 18, 1996.

<http://www.imc.org/pdi/vcard-21.ps>

Welch, B. B. *Practical Programming in Tcl and Tk*. Prentice Hall. ISBN 0136168302.

X.400 API Association and X/Open Company Ltd. *API to Electronic Mail (X.400) CAE Specification: Issue 2*. 1994.

X.400 API Association and X/Open Company Ltd. *Guide to Selected X.400 and Directory Service APIs*. 1991.

X.400 API Association and X/Open Company Ltd. *OSI-Abstract-Data Manipulation API (XOM) CAE Specification: Issue 2*. 1994.

X.400 API Association and X/Open Company Ltd. *X/Open API to Directory Services CAE Specification: Issue 2*. 1994.

Obtaining Documents

All Documents

You can obtain ITU-T (CCITT) Recommendations, ISO/IEC Standards and draft standards, and paper copies of RFCs from:

Omnicom PPI Ltd.
 Forum Chambers
 The Forum
 Stevenage
 Herts SG1 1EL
 ENGLAND
 Telephone: +44 438 742424
 Fax: +44 438 740154

Phillips Publishing Inc.
 7811 Montrose Road
 Potomac
 MD 20854
 USA
 +1 301 424 3338
 +1 301 309 3847

ISO/IEC Documents

Contact your national Standards Organization.

ITU-T (CCITT) Documents

CCITT/ITU-T Recommendations and draft documents can be obtained from:

International Telecommunications Union
 General Secretariat - Sales Section
 Place des Nations
 CH-1211 Geneva 20
 Switzerland

RFCs

Electronic copies of RFCs are available from the following FTP servers:

- <ftp://ftp.isi.edu/in-notes/>
- <ftp://nis.nsf.net/internet/documents/rfc/>

X/OPEN Documents

You can obtain X/OPEN Company Limited and X.400 API Association documents by post from:

X/OPEN Company Limited
Apex Plaza
Forbury Road
Reading
Berkshire, RG1 1AX England

or by electronic mail submission to:

XoSpecs@xopen.co.uk

Glossary & Abbreviations

The terms and abbreviations necessary for use of this book are explained below.

Distinguished Name: A set of attributes that uniquely identifies an object or user within a global set of network addresses. It is constructed of concatenated values, in a similar way to the construction of a file name. In the case of a Distinguished Name, the values identify such things as the country, organization, organizational group and user name associated with the user.

Directory Information Tree: A way of defining relationships between, and access routes to, the complete set of information contained in the X.500 Directory.

DIT: See Directory Information Tree.

DN: See Distinguished Name.

LDAP: See Lightweight Directory Access Protocol.

Lightweight Directory Access Protocol: Protocol used to provide simple access to the X.500 Directory.

Message Store: A means of providing a continuously available, secure storage place for messages being transferred either to or from a UA. Each Message Store services only one O/R address. However, a Message Store database consisting of many Message Stores can be configured to service multiple O/R addresses.

MS: See Message Store.

Message Transfer Agent: A component of the MTS, which, in co-operation with other MTAs, transfer and deliver messages to the intended recipients.

Message Transfer System: The MTS delivers, to one or more recipients, messages submitted to it by UAs. An MTS consists of a number of MTAs

MHS-DS: An abbreviation used (here) to refer to a method of configuring MTAs and Message Stores using an X.500 Directory.

MTA: See Message Transfer Agent.

MTS: See Message Transfer System.

O/R: Originator/Recipient - the User Agent sending a message or to whom a message is to be delivered.

O/R Address: A collection of information (an *attribute list*) that uniquely identifies the User Agent to whom a message is to be delivered or notification of delivery returned. The O/R address also identifies the user's point of access to the MHS.

Open Systems Interconnection: Set of design rules and protocol specifications defined to allow systems from any source to talk to each other, thus evading problems of hardware and software incompatibility.

OSI: See Open Systems Interconnection

P3: The X.400 protocol used in communications to and from an MTA.

P7:The X.400 message handling protocol used to access a Message Store.

UA: See User Agent.

User Agent: A User Agent is an application process that interacts with the MTS or a Message Store, to submit messages for a single user. It can also accept delivery of messages, either directly from the MTS or by retrieving them from a Message Store.