

**SWADM-19.1**

**M-Switch Administration Guide**

**Isode**

# Table of Contents

<b>Chapter 1</b>	<b>Overview.....</b>	<b>1</b>
	This chapter introduces the Isode M-Switch, giving an overview of its main features and components. It also explains this document and the complementary volume, the <a href="#">M-Switch Operator's Guide</a> and the <a href="#">M-Switch Advanced Administration Guide</a> .	
<b>Chapter 2</b>	<b>Product Activation.....</b>	<b>5</b>
	This chapter provides a brief overview of the Isode Product Activation system. This supersedes the legacy licensing system used prior to R19.0.	
<b>Chapter 3</b>	<b>Installing M-Switch.....</b>	<b>7</b>
	This chapter provides a brief overview of the installation process for M-Switch and its components. Step-by-step instructions for the wizards can be found in the later chapters of this Admin Guide, and relevant evaluation guides.	
	These should be read in conjunction with the Release Notes which include other important information.	
<b>Chapter 4</b>	<b>Configuring Your Messaging System.....</b>	<b>10</b>
	Most messaging systems hold their configuration and information about users and routing in a Directory. This chapter describes how to manipulate this configuration information with the messaging configuration tool MConsole.	
<b>Chapter 5</b>	<b>Configuring an Internet Messaging System.....</b>	<b>28</b>
	This chapter gives instructions on how to set up an Internet messaging configuration using MConsole.	
<b>Chapter 6</b>	<b>Managing Internet Users Using LASER Routing.....</b>	<b>38</b>
	This chapter describes how to setup LASER routing to correctly route email for M-Switch users.	
<b>Chapter 7</b>	<b>Managing Internet Messaging Users.....</b>	<b>52</b>
	This chapter contains references to other manuals which describe how to configure M-Switch so that internet email addresses are routed and delivered by M-Switch into M-Box using MConsole.	
<b>Chapter 8</b>	<b>Connecting to SMTP MTAs.....</b>	<b>54</b>
	This chapter guides you through the way in which SMTP connections to other Internet MTAs are configured.	
<b>Chapter 9</b>	<b>Configuring an X.400 Messaging System.....</b>	<b>71</b>
	This chapter gives instructions on how to set up an X.400 messaging configuration using MConsole.	
<b>Chapter 10</b>	<b>X.400 Message Store.....</b>	<b>79</b>
	The configuration for the P7 Message Store (M-Store X.400) can be stored in the Directory, and is set up using MConsole when an X.400 or MIXER messaging configuration is created. This chapter describes how to create and edit the properties of an existing X.400 Message Store.	
<b>Chapter 11</b>	<b>Managing X.400 Messaging Users.....</b>	<b>86</b>
	The creation and management of X.400 user accounts is described in this chapter.	
<b>Chapter 12</b>	<b>Configuring a MIXER Messaging System.....</b>	<b>105</b>
	This chapter gives instructions on how to set up an Internet/X.400 MIXER messaging configuration using MConsole.	

<b>Chapter 13</b>	<b>DSA Authentication and Authorization.....</b>	<b>121</b>
	This chapter gives an overview of the way in which M-Switch can be configured to access the DSA to obtain its Configuration Information, as well as User Configuration. This includes the used of both SASL IDs and Directory Names.	
<b>Chapter 14</b>	<b>Configuring MTAs.....</b>	<b>131</b>
	This chapter describes the configuration of an MTA using MConsole.	
<b>Chapter 15</b>	<b>Routing.....</b>	<b>164</b>
	This chapter describes how Routing works in M-Switch, listing the different Lookup Policies which can be configured and how they work.	
	ACP127 Routing is not covered in this section. See <a href="#">Chapter 19, ACP127</a> for a description of ACP127 Routing.	
<b>Chapter 16</b>	<b>Connecting to other X.400 MTAs.....</b>	<b>181</b>
	This chapter shows you how to configure connections to other X.400/MIXER MTAs.	
<b>Chapter 17</b>	<b>Connecting to other Military MTAs.....</b>	<b>202</b>
	This chapter contains information about how to configure an ACP 142 (P_Mul) channel. You can send and receive either of Internet or X.400 messages using the ACP 142 channel.	
<b>Chapter 18</b>	<b>Security Labels and Access Control.....</b>	<b>215</b>
	This chapter describes how to configure M-Switch to manage Security Labels, including how it can map between a wide range of Security Label formats and message transport mechanisms when crossing a MIXER gateway or ACP127 gateway. It also discusses how to make use of Security Labels to perform Access Control.	
<b>Chapter 19</b>	<b>ACP127.....</b>	<b>221</b>
	This chapter describes how ACP127 interworking can be set up and configured.	
<b>Chapter 20</b>	<b>Summary View.....</b>	<b>273</b>
	The Summary View provides an easy way for an operator to check the status of an MTA or group of MTAs.	
<b>Chapter 21</b>	<b>Channel Monitor View.....</b>	<b>274</b>
	The Channel Monitor View provides a simple way for operators to monitor specific MTA channels.	
<b>Chapter 22</b>	<b>Corrector Channel.....</b>	<b>276</b>
	This chapter describes how the Corrector channel can be set up and configured.	
<b>Chapter 23</b>	<b>CFTP.....</b>	<b>279</b>
	This chapter describes how CFTP can be set up and configured.	
<b>Chapter 24</b>	<b>FTBE.....</b>	<b>287</b>
	This chapter describes how to configure and operate the File Transfer By Email (FTBE) channel.	
<b>Chapter 25</b>	<b>SLEP.....</b>	<b>296</b>
	This chapter describes how SLEP can be set up and configured.	
<b>Chapter 26</b>	<b>Lists and Recipient Expansion Overview .....</b>	<b>301</b>
	M-Switch has features which enable the recipients of messages to be changed. One such set of features, known as List Expansion, is summarised in this chapter.	

## **Chapter 27 SMTP Directory Based Distribution Lists..... 302**

This chapter describes the creation and maintenance of SMTP (Internet) distribution lists that are stored in the directory.

## **Chapter 28 Directory Based X.400 Conformant Distribution Lists..... 305**

Distribution lists provide a facility for expanding a single list address into multiple recipient addresses. This chapter describes the creation and maintenance of Directory-based X.400 conformant distribution lists.

## **Chapter 29 Military Distribution Lists..... 314**

Distribution lists provide a facility for expanding a single list address into multiple recipient addresses. This chapter describes the creation and maintenance of Military specific address lists.

## **Chapter 30 Profiler Channel..... 317**

A Profiler is a Messaging component which takes an input message and distributes the message to new recipients based on the information in the message.

## **Chapter 31 Edge Channel..... 334**

This chapter describes how the M-Switch Edge channels can be set up and configured to send MIME messages through an M-Guard. There are two distinct channels:

- The Edge Client: this is a dedicated output channel for converting MIME messages to an M-Switch Edge defined XML format, wrapping it in a GCXP wrapper and sending it to an M-Guard (or compatible service).
- The Edge Listener: this is a dedicated input channel for receiving GCXP wrapped messages from an M-Guard (or compatible client), unwrapping the contained XML message and converting it to MIME format for submission to a local Isode M-Switch message queue.

This is expanded on in the section [Section 31.1, “The M-Switch Edge channel”](#).

The aim of this feature is to enable users to send MIME messages through an M-Guard, subject to strict conditions. These include:

- Passing through of only specifically supported headers and message elements by the client sending side and ignoring all others. The supported elements and XML format are defined by an M-Switch Edge XSD.
- Further stripping out of the supported headers depending on configurable rules and content validation according to configurable rules by the client.
- Sending of the resultant message XML (in a GCXP wrapper) to an M-Guard using TLS. This ensures that only permitted clients can attempt to deliver messages via the M-Guard. Furthermore, the M-Guard acts as a one way system (a 'diode'). Only transmission errors are relayed back to the client. No subsequent processing or delivery information is returned to the client side. This ensures that there is no leaking of information (e.g. valid or invalid addresses) from the receiving side back to the sender. Transmission errors can be reported by the client, but all other delivery errors must be handled on the receiving side. This means that there will be manual intervention required on the receiving side in order to handle those errors.
- The M-Guard can be set up to apply strict rules as to what messages, parts of messages and specific content may be transmitted.
- The receiving side can receive and unwrap the GCXP messages to an XML encoded message that is validated against the M-Switch Edge XSD. This message XML is then converted to a MIME format and submitted to the local Isode M-Switch queue. The listener reports all errors to the local operator. No status is reported back to the sender. All error handling in this stage must be handled by the local operator, who will decide what action (including feedback to the sender, if any) is required.



<b>Chapter 32</b>	<b>Message Switch Console.....</b>	<b>350</b>
	Message Switch Console (MConsole) is a graphical management tool that is used to manage multiple aspects of M-Switch.	
<b>Chapter 33</b>	<b>Starting and Stopping Your Messaging System.....</b>	<b>362</b>
	Stopping and starting the MTA on both UNIX and Windows systems are described in this chapter, together with any suggested pre-start checks.	
<b>Chapter 34</b>	<b>Managing Your Messaging System.....</b>	<b>377</b>
	This chapter describes the various ways you can monitor your messaging system, and gives some guidance on troubleshooting and improving performance. The first section describes how to create a backup of a messaging configuration.	
<b>Chapter 35</b>	<b>SNMP.....</b>	<b>409</b>
	This chapter guides you through the way in which you can configure M-Switch to act as an SNMP Agent.	
<b>Chapter 36</b>	<b>Message Audit Database.....</b>	<b>413</b>
	This chapter describes the audit database, which provides a central repository for information about messages passing through your system. It enables you to monitor and manage areas of congestion and where errors have occurred, from a remote location.	
<b>Chapter 37</b>	<b>Clustering.....</b>	<b>434</b>
	If you have suitable clustered hardware, the Message Switch can be configured to run in a hot-standby configuration.	
<b>Chapter 38</b>	<b>Securing Your Messaging System.....</b>	<b>445</b>
	Authentication is proving the identity of someone or of a process; authorization is what that person or process is allowed to do. This chapter describes the configuration of both of these elements in M-Switch.	
	This chapter also describes how the secure storage of passwords can be achieved.	
<b>Chapter 39</b>	<b>Content Conversion and Scanning on Submission.....</b>	<b>451</b>
	This chapter provides a short overview of the way in which the contents of messages can be converted or scanned on submission, and describes the Shaper Configuration File Editor component of MConsole which allows the conversions or scanning to be configured.	
<b>Chapter 40</b>	<b>Content Checking.....</b>	<b>456</b>
	This chapter describes content checking, which allows an administrator to block, filter and alter messages based on their content.	
<b>Chapter 41</b>	<b>SPIF Editor.....</b>	<b>482</b>
	This chapter describes the SPIF Editor application and explains how to use it to create, edit and view a SPIF (Security Policy Information File) and various utility functions.	
<b>Chapter 42</b>	<b>Alert Daemon.....</b>	<b>502</b>
	This chapter describes the Alert Daemon, which can be used to monitor multiple M-Switch servers and generate Alerts in response to the occurrence of specific conditions.	
<b>Appendix A</b>	<b>Messaging APIs.....</b>	<b>513</b>
	The Isode M-Switch product includes a number of programmatic APIs which are formally supported as part of the product set. These are for the most part documented using automatically generated documentation (e.g. using javadoc and doxygen).	

**Appendix B Presentation Addresses..... 515**

These are the fundamental mechanism used by applications to address other application entities. After reading this chapter you should know how to represent any OSI PA in a string format.

**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 2002-2025, 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 2025.

---

## 1 Software version

This guide is published in support of Isode M-Switch R19.1. It may also be pertinent to later releases. Please consult the release notes for further details.

---

## 2 Readership

This guide is intended for administrators who plan to configure and manage the Isode M-Switch message switch. For detailed information on Operating M-Switch, use the complementary volume the [M-Switch Operator's Guide](#). For advanced features of M-Switch, see the [M-Switch Advanced Administration Guide](#).

---

## 3 How to use this guide

You are advised to read through [Chapter 1, Overview](#), before you start to set up your messaging system.

---

## 4 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	Example
File and directory names	<i>isoentities</i>
Program and macro names	mkpasswd
Input to the system	cd newdir
Cross references	see <a href="#">Section 5, “File system place holders”</a>
Additional information to note, or a warning that the system could be damaged by certain actions.	Notes are additional information; cautions are warnings.

Arrows are used to indicate options from the menu system that should be selected in sequence.

For example, **File** → **New** means to select the **File** menu and then select the **New** option from it.

## 5 File system place holders

Where directory names are given in the text, they are often 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.

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

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

Name	Place holder for the directory used to store...	Windows (default)	UNIX
( <i>ETCDIR</i> )	System-specific configuration files.	<i>C:\Isode\etc</i>	<i>/etc/isode</i>
( <i>SHAREDIR</i> )	Configuration files that may be shared between systems.	<i>C:\Program Files\Isode\share</i>	<i>/opt/isode/share</i>
( <i>BINDIR</i> )	Programs run by users.	<i>C:\Program Files\Isode\bin</i>	<i>/opt/isode/bin</i>
( <i>SBINDIR</i> )	Programs run by the system administrators.	<i>C:\Program Files\Isode\bin</i>	<i>/opt/isode/sbin</i>
( <i>EXECDIR</i> )	Programs run by other programs; for example, M-Switch channel programs.	<i>C:\Program Files\Isode\bin</i>	<i>/opt/isode/libexec</i>
( <i>LIBDIR</i> )	Libraries.	<i>C:\Program Files\Isode\bin</i>	<i>/opt/isode/lib</i>
( <i>DATADIR</i> )	Storing local data.	<i>C:\Isode</i>	<i>/var/isode</i>
( <i>LOGDIR</i> )	Log files.	<i>C:\Isode\log</i>	<i>/var/isode/log</i>
( <i>CONFPDUSPOOLDIR</i> )	Large PDUs on disk.	<i>C:\Isode\tmp</i>	<i>/var/isode/tmp</i>
( <i>QUEDIR</i> )	The M-Switch queue.	<i>C:\Isode\switch</i>	<i>/var/isode/switch</i>
( <i>DSADIR</i> )	The Directory Server's configuration.	<i>C:\Isode\d3-db</i>	<i>/var/isode/d3-db</i>

## 6 Support queries and bug reporting

A number of email addresses are available for contacting Isode. Please use the address relevant to the content of your message.

- For all account-related inquiries and issues: [customer-service@isode.com](mailto:customer-service@isode.com). 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.
- For all licensing related issues: [license@isode.com](mailto:license@isode.com).
- For all technical inquiries and problem reports, including documentation issues from customers with support contracts: [support@isode.com](mailto:support@isode.com). 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.

- For all sales inquiries and similar communication: [sales@isode.com](mailto:sales@isode.com).

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/subscribe.html>

---

## 7 Export controls

Many Isode products use TLS (Transport Layer Security) to encrypt data in transit. This means that these products are subject to UK Export Controls.

For some countries (at the time of shipping this release, these comprise all EU countries, United States of America, Canada, Australia, New Zealand, Switzerland, Norway, Japan), these Export Controls can be handled by administrative process as part of evaluation or purchase. For other countries, a special Export License is required. This can be applied for only in context of a purchase order for those Isode products.

You must ensure that you comply with these Export Controls where applicable, i.e. if you are licensing or re-selling Isode products.

The TLS feature of Isode products is enabled by a TLS Product Activation feature. This feature may be turned off, and Isode products without this TLS feature are not export controlled. This can be helpful to support evaluation of Isode products in countries that need a special export license.

Isode products are used to administer sensitive data and so Isode strongly recommends that all operational deployments of Isode products use the export-controlled TLS feature.

All Isode Software is subject to a license agreement and your attention is also called to the export terms of your Isode license.

# Chapter 1 Overview

This chapter introduces the Isode M-Switch, giving an overview of its main features and components. It also explains this document and the complementary volume, the [M-Switch Operator's Guide](#) and the [M-Switch Advanced Administration Guide](#).

---

## 1.1 What is the Isode M-Switch?

M-Switch is a high-performance, versatile Message Transfer Agent (MTA), which can be installed on either Windows or UNIX platforms. It is the main component in a messaging system and supports:

- Internet messaging
- X.400 messaging
- ACP127 messaging
- A mixture of the three variants, converting messages from one form to the other.

The MTA consists of:

- The Queue Manager (`qmgr`)
- Channel processes
- Protocol listeners for messages entering the MTA (`iaed`, `smtpsrvr`, `acp127` or `isode.pp.p3`)

Other components include:

- M-Vault, used to hold configuration
- Management tools (GUIs and command line)

The `qmgr` carries out three functions:

- Manages the message queue, scheduling channels to process messages

A sequence of MTA channels is scheduled on submission of a message to process the message. These channels accept inbound messages as well as relaying or delivering messages as outbound channels. Channels are also used to convert messages from one protocol to another, to redirect them and to perform various management operations, e.g. checking the content.

- Acts as the Switch Operations Management (SOM) protocol server for the Management Tools and other clients
- Downloads the configuration into files such as *mtataylor* file, which contains the MTA's configuration, from the LDAP/X.500 Directory

M-Switch is the main part of an overall messaging system, which consists of M-Switch and a number of tools (applications) that help you to configure and manage it. These applications operate over protocol and are independent of the platform the messaging system is running on. This means that you can set up and monitor an enterprise-wide messaging configuration from a system, such as an administration workstation, which is remote from the system running the messaging software.

- The majority of the MTA's configuration is usually stored in an LDAP/X.500 Directory. This includes general configuration of the MTA, as well as routing and addressing

information. This information is managed using the Message Switch Console (MConsole), which include a graphical Directory User Agent. An overview of MConsole is given in [Chapter 4, \*Configuring Your Messaging System\*](#).

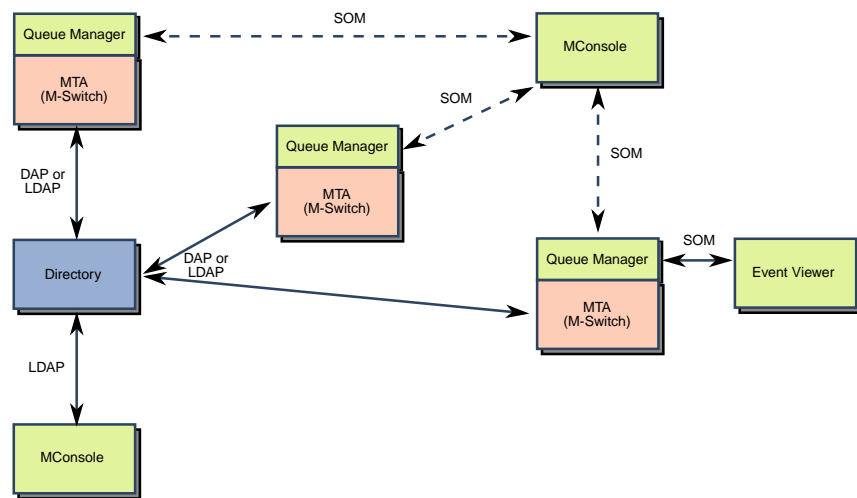
- Although Isode recommends directory-based configuration, for some configurations you may wish to use the legacy table-based configuration option (configuration is held in files), which does not use the Directory. This may be an appropriate approach if, for example, you are configuring an MTA to act as a border or gateway MTA, with minimal routing information and no local users. This topic is covered in the *M-Switch Advanced Administration Guide*.
- M-Switch is monitored and controlled using MConsole, which has a graphical interface. Amongst other Operator and Monitoring features, this application enables you to monitor MTAs as well as fine-tune the MTA and deal with problem messages. MConsole can also perform message tracking functions if the Message Audit Database is configured. An overview of MConsole including how to configure it, is provided in [Section 32.1, “Overview”](#).

For a full description of how to use MConsole see [Chapter 32, \*Message Switch Console\*](#)

- Internet user-related information is accessed and updated via the **Internet Mailbox Management** view. This is part of MConsole see [Chapter 7, \*Managing Internet Messaging Users\*](#). This tool enables users of the system to manage their own message handling and administrators to record routing information for Internet messaging users.
- Information about all of the messages in your system can be stored in the Message Audit Database, enabling you to track messages and audit their progress without interrupting day-to-day operations. Further information can be found in [Chapter 36, \*Message Audit Database\*](#).
- On Windows systems, the MTA's Services are managed using the Isode Service Configuration.

[Figure 1.1, “Overview of Messaging Management System”](#) illustrates where these management applications fit in a typical enterprise messaging network.



**Figure 1.1. Overview of Messaging Management System**

MConsole Switch Operations and MConsole Switch Configuration Management are distanced from each other for clarity in the diagram, because they communicate with different objects in the messaging network and operate over different protocols. In reality, however, both management applications would probably run on the same system.

Each MTA is using a shared configuration held within a single Directory. This simplifies changes as an update to any part of the configuration need only be made once before it is picked up by all the MTAs.

Alternatively, each MTA may have its own Directory holding its own configuration. In such cases only the local MTA has a full entry in the Directory and the other MTAs are represented as 'External MTAs'. An External MTA always appears without tailoring information, and with one or more protocol channels. Non-Isode MTAs are always External MTAs.

---

## 1.2 Using the Administrator Guide

This Administrator Guide contains the information you need to install, configure and maintain your message handling system using the provided tools.

---

**Note:** The M-Switch MTA can be used in Internet Messaging, X.400 and MIXER environments. X.400 information has been excluded from the Internet Messaging version of the document as much as possible, but in some areas the product itself makes reference to both variants.

---

The companion document, the *M-Switch Advanced Administration Guide*, contains lower-level information, which may help you to determine the information that you need to record within the management tools. You will also have to use the information in the *M-Switch Advanced Administration Guide* if you wish to use table-based configuration.

---

**Note:** The document assumes a basic familiarity with the terminology and concepts of messaging systems. The *M-Switch Advanced Administration Guide* contains a comprehensive glossary of the terms you may encounter.

---

# Chapter 2 Product Activation

This chapter provides a brief overview of the Isode Product Activation system. This supersedes the legacy licensing system used prior to R19.0.

---

## 2.1 Overview

Each product (known as a feature to Product Activation), and each product has a number of possible sub-features which can be activated. For example the **M-Switch Gateway** Product can have the **smtp** subfeature activated.

M-Switch must be activated as a feature in order to start. Similarly, some M-Switch programs may need to be activated in order to start.

M-Switch server programs may need to be activated in order to run, e.g. many protocol channels such as **smtp** and **x400p1**.

Activation of product and subfeatures works by requesting a Product Activation Key (PAK) from Isode. Such a request must be accompanied by a Product Activation Request (PAR). Isode messaging products can all be activated using the Messaging Activation Server which is available separately from M-Switch at <http://www.isode.com/>

The Messaging Activation Server manages PAKs, and will put the PAK into *(ETCDIR)/activate.dat*.

Messaging UIs, such as MConsole, although not activated in itself, will check activated features and sub features and adjust behaviour in order to present configurable features in a helpful fashion in which subfeatures which are not activated are not configurable. So, for example, X.400 features or Views will not be presented if the **x400** sub-feature is not activated. This can be disabled in the Options View so that Views which would normally be suppressed, are presented.

---

**Note:** MConsole will not configure or use TLS unless TLS itself is activated in the installed PAK.

---

---

## 2.2 Effects of Product Activation

### 2.2.1 MConsole Use Of Product Activation

MConsole will be aware of Product Activation in one of two ways:

- Reading the installed PAK in *(ETCDIR)/activate.dat* . This is used to present information in the MConsole Help, as well as to control many decisions regarding whether functionality is presented.

The wizards that create Messaging Configurations or MTAs will use the activated sub features to streamline the process of creation.

- Connecting to M-Switch using SOM to the qmgr. This is used to present information regarding Product Activation in M-Switch being managed in the Switch Operations View and Summary View.

## **2.2.2 Server Use Of Product Activation**

When M-Switch starts up the qmgr will check that the M-Switch product (aka feature) is activated. Other checks will take place in some channels to ensure that sub features are activated.

Any use of TLS will always be preceded by a check on TLS activation.

# Chapter 3 Installing M-Switch

This chapter provides a brief overview of the installation process for M-Switch and its components. Step-by-step instructions for the wizards can be found in the later chapters of this Admin Guide, and relevant evaluation guides.

These should be read in conjunction with the Release Notes which include other important information.

---

## 3.1 Prerequisites

Release Notes for current releases are available on the Isode website. These contain detailed information about the operating systems that are currently supported and any specific prerequisites.

You need a valid **Product Activation Key** for each of the applications that you are going to install.

---

## 3.2 Before installation

### 3.2.1 Before Installing on Windows

You need to ensure that you have identified suitable directories in which to install and run M-Switch.

On Windows the default install directory is: *C:\Program Files\Isode* (or equivalent for your local language). You can choose to change this to a different drive or directory. The leaf directory must not already exist unless you are updating/upgrading from a previous install.

M-Switch requires a suitable version of the Java Runtime Environment (JRE), which must be installed before installing M-Switch – see the Release Notes for further details of supported versions and how to install.

In order to run the GUIs such as MConsole, you will need to set suitable environment variables using the Windows system tools. This should look something like this:

```
JAVA_HOME="C:\Program Files\AdoptOpenJDK\jdk-11.0.11.9-hotspot\"  
ISODE_JAVA_HOME="C:\Program Files\AdoptOpenJDK\jdk-11.0.11.9-hotspot\"
```

### 3.2.2 Before Installing on Linux

You need to ensure that you have identified suitable directories in which to install and run M-Switch.

The install directories on Linux are */opt/isode*; */var/isode*; */etc/isode*. The leaf directory must not already exist.

When creating a system on which to run Isode messaging services including M-Switch you should check the available disk space. For example installing on Ubuntu under Hyper-V by default may provide insufficient disk space. You should allocate at least 32Gb.

You should also select a user, creating if necessary, as which unprivileged M-Switch programs and daemons run. By default this is the 'pp' user. You should do this before installing the packages.

On Ubuntu this user needs to be able to login.

On Ubuntu you must disable Wayland in favour of X.

M-Switch requires a suitable version of the Java Runtime Environment (JRE), which must be installed before installing M-Switch – see the Release Notes for further details of supported versions and how to install.

In order to run the GUIs such as MConsole, you will need to set suitable environment variables. One way to do this is to edit */etc/environment* to include something like

```
JAVA_HOME="/opt/AdoptJDK11/jdk-11.0.14+9"
ISODE_JAVA_HOME="/opt/AdoptJDK11/jdk-11.0.14+9"
```

### 3.2.2.1 Installing M-Box on Linux

If running M-Box, you need to do the following :

- Create */var/isode/ms* and */var/isode/ms/user* with suitable ownership and permissions.
- You should also select a user, creating if necessary, as which unprivileged M-Box programs and daemons run. By default this is the 'mbox' user. You should do this before installing the packages.

---

## 3.3 Installation

---

**Caution:** If you are upgrading from an earlier version of M-Switch, you must read the Release Notes in case there are changes to the product between the versions. Failure to take appropriate action may result in loss of data.

---

See the Release Notes for detailed installation instructions for the Isode packages and any dependencies such as Java, PostgreSQL etc.

---

## 3.4 Getting started

You need to use MConsole to create a messaging configuration. See [Chapter 4, Configuring Your Messaging System](#).

---

**Note:** You need to create a DSA to hold the messaging configuration. You should use MConsole to do this.

---

---

## 3.5 After Creating A Messaging Configuration

### 3.5.1 Linux

Once you have created a configuration using MConsole, you can start and stop the Isode services. See [Section 33.1.2, “Isode Messaging Services”](#) for a further explanation.

In addition you are likely to need to tailor which services start when the M-Switch Services are started/stopped by editing `(ECTCDIR)pp.rc` as described in [Section 33.1.3, “Configuring the Messaging startup script”](#). See [Section 33.1.1.2.2, “Startup Steps”](#) for a further explanation.

Java services require `JAVA_HOME` or `ISODE_JAVA_HOME` being set to a suitable value. This can be achieved by altering the `(ECTCDIR)isode.rc` file.

### 3.5.2 Windows

The Isode services are managed using the Isode Service Configuration tool which is started using the **Start** → **Programs**. See [Section 33.2, “Starting an MTA on Windows”](#) for a further explanation.

---

## 3.6 Incorporating existing configurations

Most upgrades from previous releases are straightforward, but you must check the release notes before upgrading. For Major and Minor upgrade releases the Release Notes may contain critical information on upgrading your system.

For update releases, such as when updating from R14.6v6 to R14.6v7 there should be no issues when installing your update. However, it is always wise to read the Release Notes and ensure that your system has been backed up before starting the update.

# Chapter 4 Configuring Your Messaging System

Most messaging systems hold their configuration and information about users and routing in a Directory. This chapter describes how to manipulate this configuration information with the messaging configuration tool MConsole.

---

## 4.1 Introducing MConsole

The Message Switch Console (MConsole) is a GUI which enables client/server management of M-Switch. MConsole comprises a number of views which enable it to act in various different ways to manage, configure, operate M-Switch.

It acts as

- a SOM client, connecting to the `qmgr` to monitor M-Switch
- an authenticated SOM client, connecting to the `qmgr` to monitor and manage M-Switch
- a simple X.400 user agent, enabling you to resend or forward messages from the live queue, the archive or the quarantine
- a Directory User Agent (DUA) which can view and edit the M-Switch configuration and the M-Switch User Mailbox configuration, which are held in the Directory.
- a manager for Authenticated Entities which allows the configuration of the Messaging Administrators such as operators which use the MConsole Switch Operations view and Isode Application Servers which perform authentication by proxy.
- X.400 Mailbox Management which allows the Administrator to configure X.400 Users and Mailboxes
- an Audit Database Client, which performs:
  - message delivery tracking
  - message acknowledgement
  - message quarantine management
  - message transfer history
  - message resubmission (together with a SOM client)
- a Windows Service Manager which performs:
  - Start/Stop of Windows Services
  - Addition/Deletion of Windows Services
  - Configuration of Windows Services

The client/server nature of MConsole means you can install MConsole on any supported platform to connect to and manage Isode M-Switch on any different system to which there is connectivity. A **Product Activation Key** will be required for the client machine, in order to allow TLS to function correctly.

MConsole shares a Directory Bind Profile with other Isode management GUIs such as Sodium.

To start Message Switch Console, ensure (`BINDIR`) is included in your path and type:  
`mconsole`

On Windows, the configuration for Message Switch Console is stored in the Windows Registry.



To start Message Switch Console, right click on the shortcut in **Start → Program Files → Isode** and select **Run As Administrator**.

---

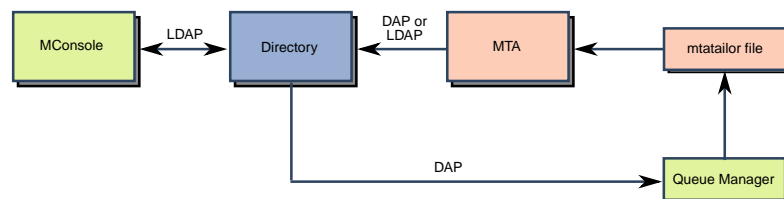
**Note:** If you do not run MConsole as Administrator, some actions cannot be carried out and will fail in unexpected ways.

---

When MConsole first starts, the Welcome View should be shown. If it is not shown, you can display it by using the menu option **View → Welcome View**. This provides a convenient set of starting points for all the views available in MConsole.

Figure 4.1, “Overview of MTA tailoring” adds more detail to the broad outline given in Figure 1.1, “Overview of Messaging Management System”, and illustrates how configuration details and MTA tailoring is accessed by an MTA and MConsole. The shaded part of the diagram would be repeated for each MTA in a messaging network.

**Figure 4.1. Overview of MTA tailoring**



The MTA is configured to use LDAP to access the Directory for configuration and routing details.

From R16.0 X.500 Directory Access Protocol (DAP) is no longer supported for access to the Directory for configuration and routing details.

The `qmgr` in each MTA periodically downloads the MTA's tailoring information from the Directory into an *mtatailor* file, which other processes in the MTA can then access. The MTA processes need to have shared access to the filestore where the *mtatailor* file is located. The *mtatailor* file is by default read from *(ETCDIR)/switch/mtatailor.tai*.

When using Directory-based configuration you still need to configure some tables, which can be held one of four formats:

- linear (held in text files and read as text files)
- dbm (held in text files, converted into a database for efficient reading)
- directory (downloaded from the Directory into files and converted into database for efficient reading)
- empty (do not exist and are not read)

Table overrides can be used to have entries in tables which are otherwise 'empty'. However, note that table overrides cannot be set from MConsole.

---

## 4.2 MConsole profiles

MConsole provides a flexible and powerful interface with many views supporting a wide range of configuration and operational management functions.

In some environments, M-Switch will be managed by Operators who will not perform general System Administrator functions. Some MConsole views (e.g., ACP 127 View, ACP 142 View, Alerts View, Message History, Event Viewer) are specifically targeted at Operators. However, some views are not appropriate for all operators, and the flexibility and large number of views can be confusing.

The views that will be available depend on the Product Activation. If a feature is not activated, the corresponding view will not be available in MConsole.

A new Profile mechanism has been introduced in order to allow M-Switch Administrators and Operators to see Views which are appropriate. E.g. to make sure Administrator Views not needed by an Operator are hidden.

In addition, other restrictions to Views such ACP127 Views and STANAG 5066 may be hidden unless a Military configuration is being managed.

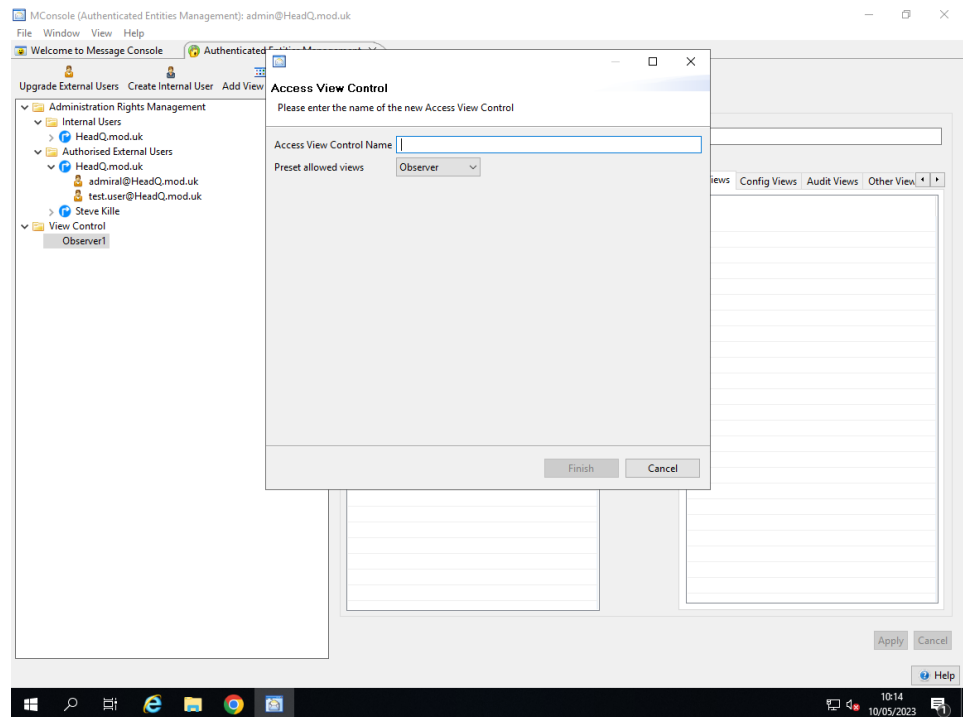
---

**Note:** An administrator must first create a configuration, before any views are restricted.

---

### 4.2.1 Authenticated entities

After a configuration has been created, open the **View** → **Configuration** → **Authenticated Entities Management** view.

**Figure 4.2. View Control Wizard**

Set the name of the view control, and select a default profile

- **Observer** This set of views allows someone to monitor a messaging system. They'll have access to the **Switch Operations View**, but not the **Switch Config View** for example.
- **Operator** This set of views allows someone to monitor and manipulate a messaging system. Other actions such as creating a new messaging system, or DSA are restricted.
- **Administrator** This is a complete set of views.

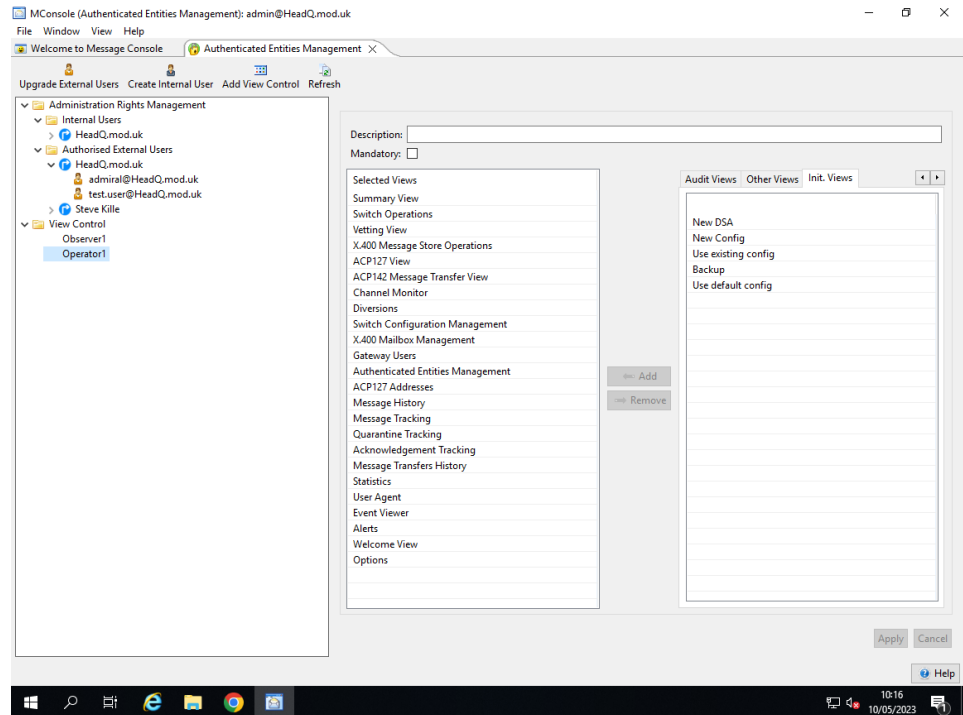
Once a view control is created, it's possible to tweak the control by altering which views are allowed.

---

**Note:** The view control only controls the views a user is allowed. SOM access controls are independent of the view list

---

Create a new user by right-clicking on **Administration Rights Management** and selecting the menu option **Add Qmgr Auth User**. You will need to associate the new user with the view control group. The **user type** will set SOM management attributes. While the **View control** sets the view control group.

**Figure 4.3. Authenticated Entities Management**

Once created you will need to bind to the DSA as the new user.

Select **File** → **DSA Bind Profile editor**

Select **New** and go through the new bind profile creation wizard, creating a bind profile for the user created in the **Authenticated Entities Management** view.

Once created select that user, and click on "bind".

---

**Note:** You may wish to remove the administrator bind profile using the **bind profile gui**.

---

## 4.3 Creating a messaging configuration

To configure an initial messaging system using MConsole you need to carry out the following steps:

1. Start MConsole, as described in [Section 4.4, “Starting MConsole”](#).
2. Create a bind profile for connection to the Directory, as described in [Section 4.4.2, “Creating a new Directory”](#).
3. Create the initial MTA configuration. The procedures to be followed depend upon the type of MTA you select to create (see [Section 4.4.3.3, “Next Steps”](#)).

For each type of MTA, you need to configure:

- a Routing Tree
- the MTA

You may also need to configure an X.400 Message Store for X.400 and MIXER MTAs and an Address Conversion Tree for a MIXER MTA. You may also need to configure a POP/IMAP message store if you wish to configure local Internet mailboxes.

The creation wizard takes you through the steps you need.

4. Set up some mail users.

For details on how to do this using MConsole see [Chapter 7, \*Managing Internet Messaging Users\*](#).

If you have an X.400 system, see [Chapter 11, \*Managing X.400 Messaging Users\*](#).

5. Start the initial messaging system, as described in [Chapter 33, \*Starting and Stopping Your Messaging System\*](#).

---

## 4.4 Starting MConsole

### 4.4.1 Introduction

As MConsole uses a client/server architecture you can install MConsole on any supported platform to connect to and manage the Isode M-Switches, either locally or remotely.

#### 4.4.1.1 Requirements Before Starting MConsole

Before starting MConsole to create a **Switch Configuration Management** view, you need to

- ensure that you have access to an LDAP/X.500 Directory, and are authorized to modify the part of the Directory Information Tree (DIT) that is to hold the messaging configuration. Log on as a user authorized to do this

OR

- create a Directory using MConsole as described below

OR

- create a Directory using M-Vault Console (mvc) as described in the *M-Vault Administration Guide*.

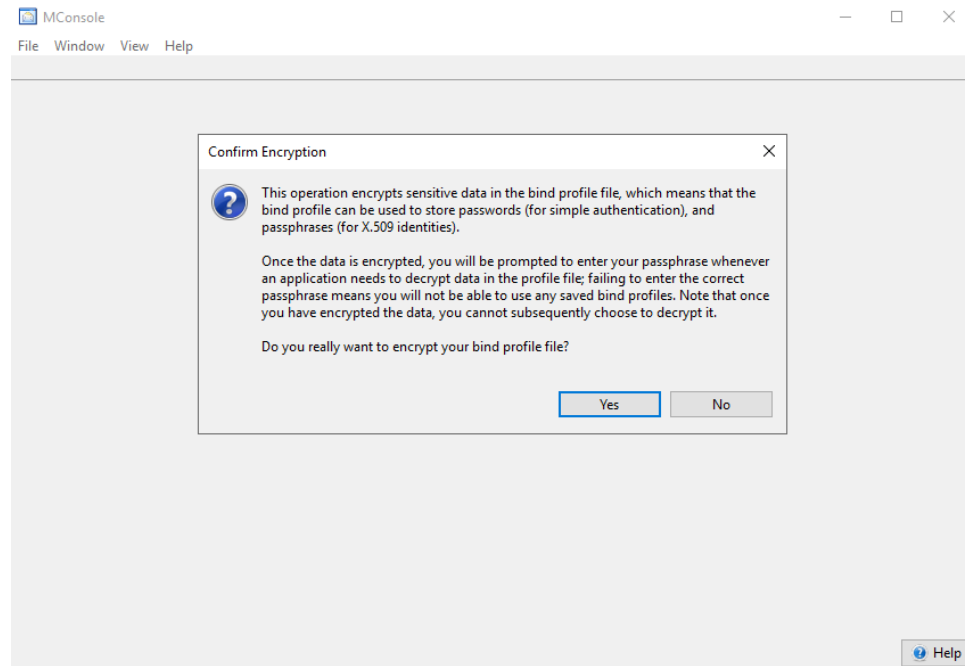
On Unix systems to start MConsole, ensure ( *BINDIR* ) is included in your path, and type:  
mconsole

On Windows right click on **Start** → **Isode** → **MConsole** and choose **Run As Administrator**.

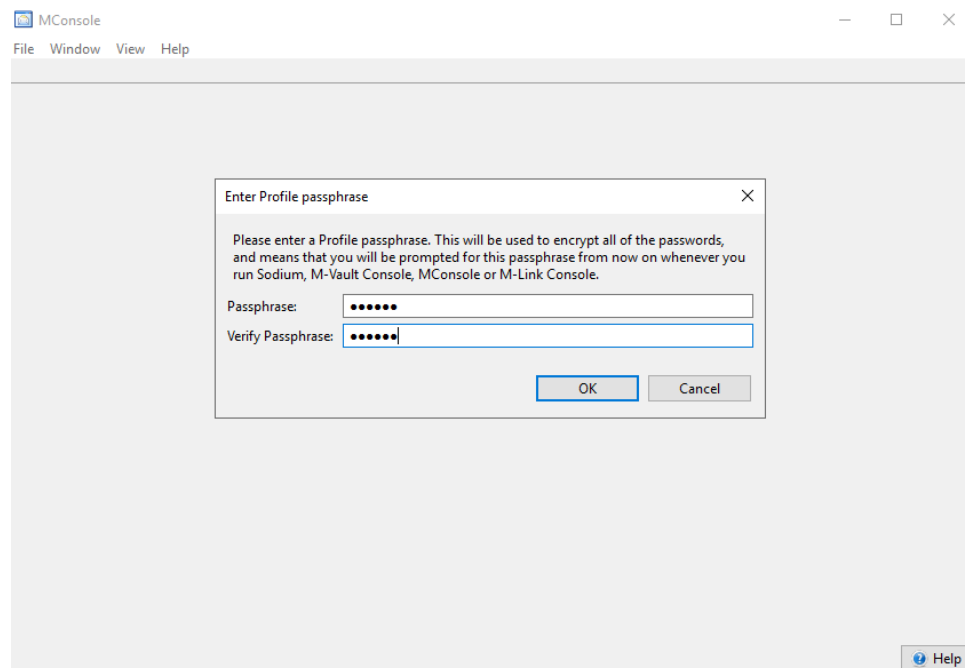
#### 4.4.1.2 Starting MConsole First Time

When MConsole first starts you will need to create a Bind Profile (unless one has been set up for you). You will be prompted to enter a passphrase to be used to encrypt your bind profile. This is required as the Bind Profile contains passwords used to authenticate the Operator to various Isode services.

The initial screen appears as follows:

**Figure 4.4. MConsole Initial Screen**

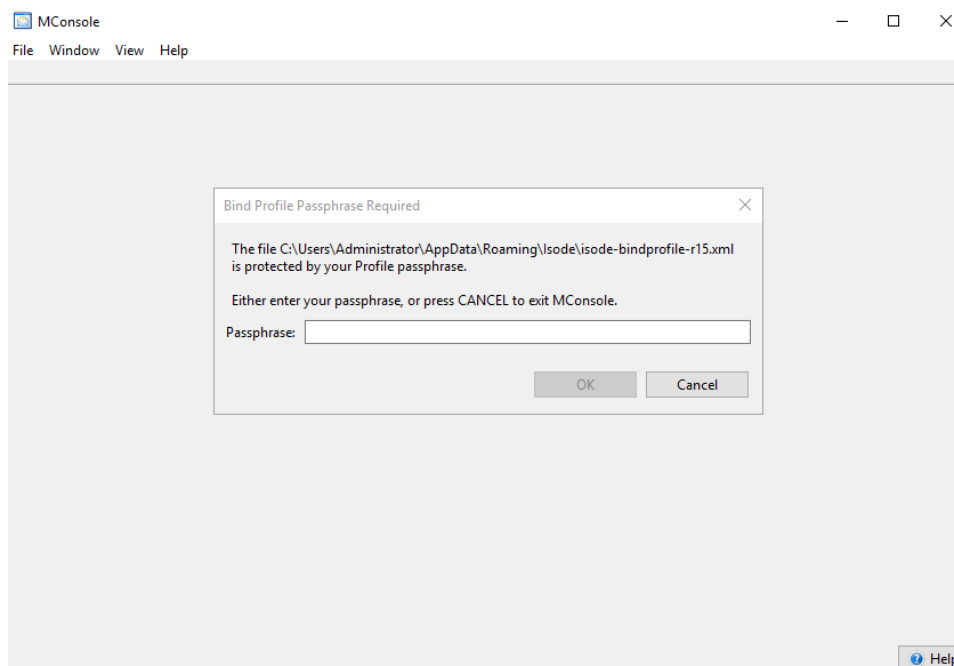
You are recommended to encrypt your Bind Profile, so after clicking yes in answer to the question, you should then set the passphrase, entering it a second time to confirm the correct value.

**Figure 4.5. MConsole Setting Bind Profile Passphrase**

When MConsole starts up subsequently you are prompted for your password. Once completed you now can now login to the M-Switch server with suitable privileges.

### 4.4.1.3 Bind Profiles

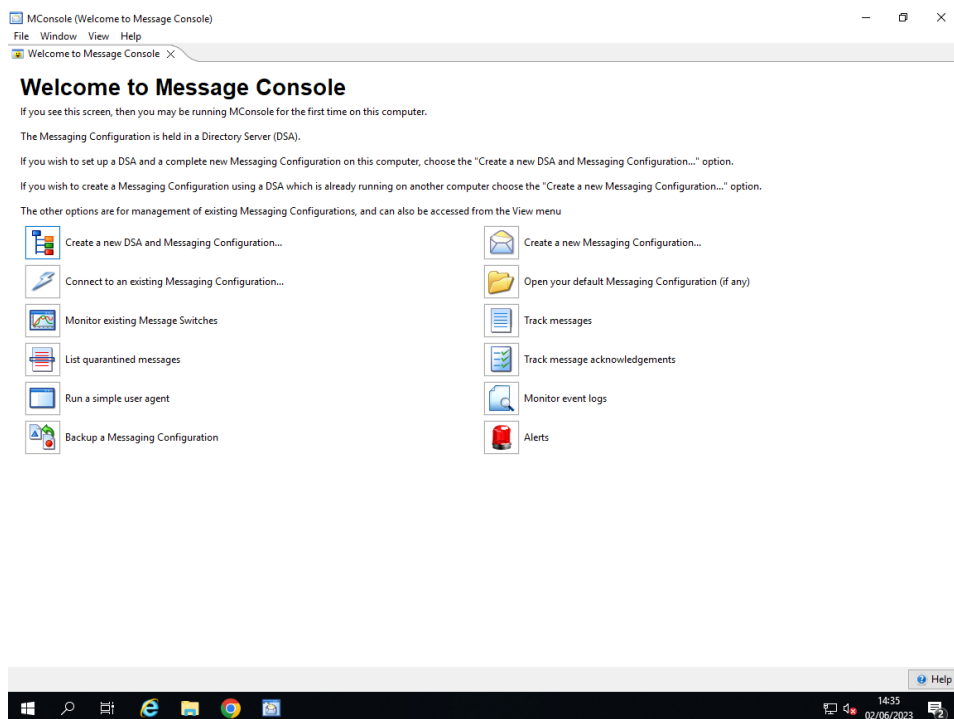
**Figure 4.6. MConsole Entering Bind Profile Passphrase**



### 4.4.1.4 Welcome Screen

The initial Welcome screen now opens. This lists a number of Views which you can now open, see [M-Switch Operator's Guide](#) for more details.

**Figure 4.7. MConsole Welcome**



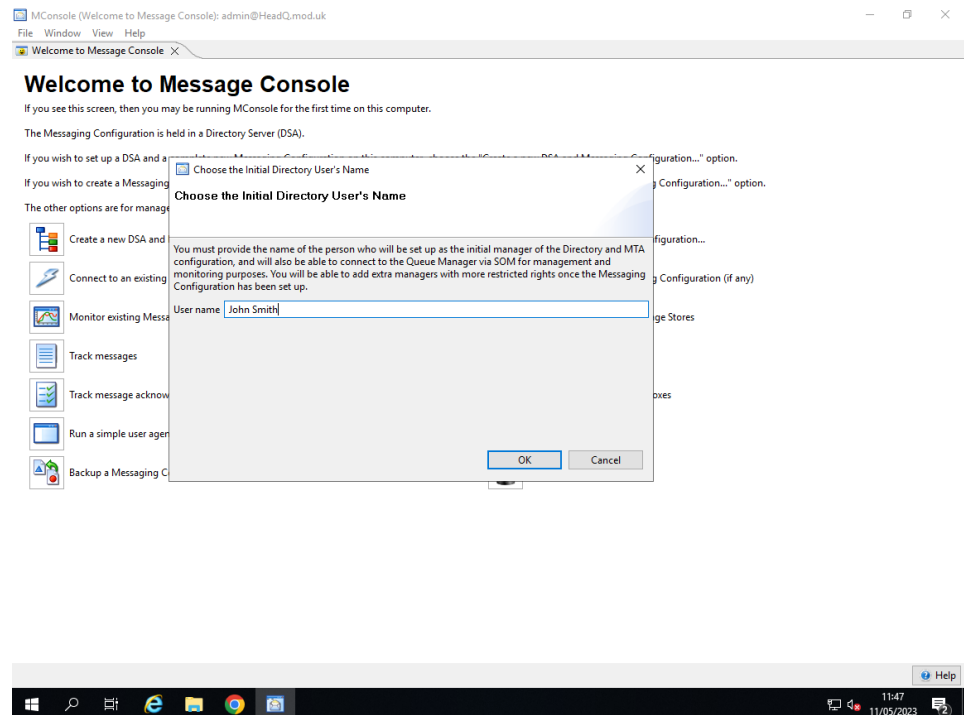
## 4.4.2 Creating a new Directory

### 4.4.2.1 Templates

You now have the choice to amend the details of the DIT structure and roles configured in the template used by MConsole to create a DSA to hold messaging configuration.

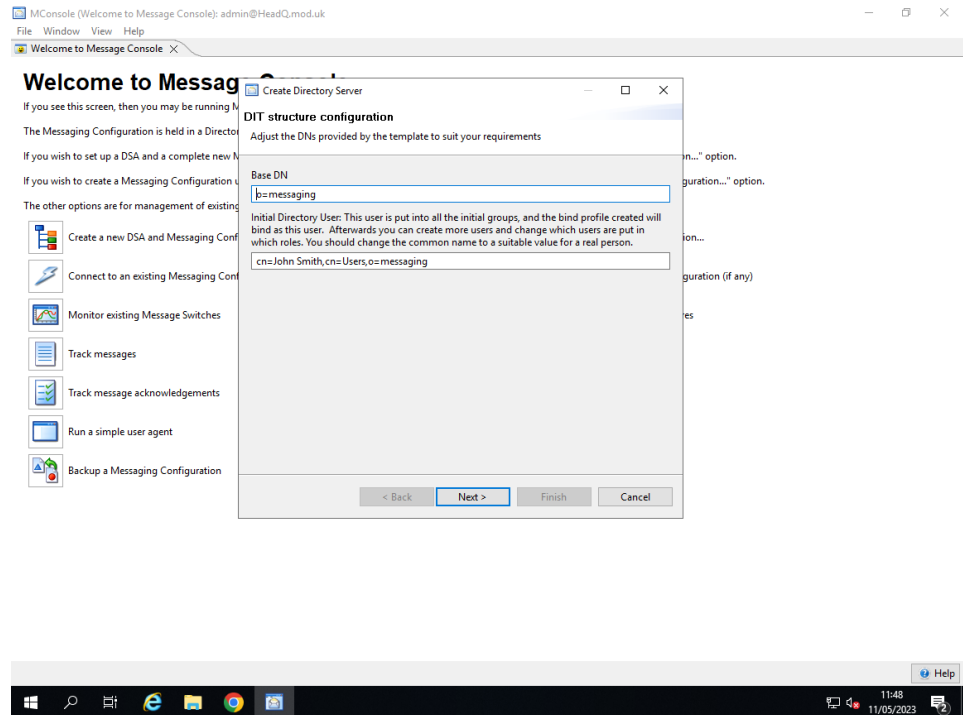
Firstly you need to specify a user who will create and manage the Messaging Configuration in the DSA. See [Section 13.4, “Creating Accounts”](#) for further details. This also describes how you can add further users later which have different privileges.

**Figure 4.8. DSA Creation: Messaging Manager**



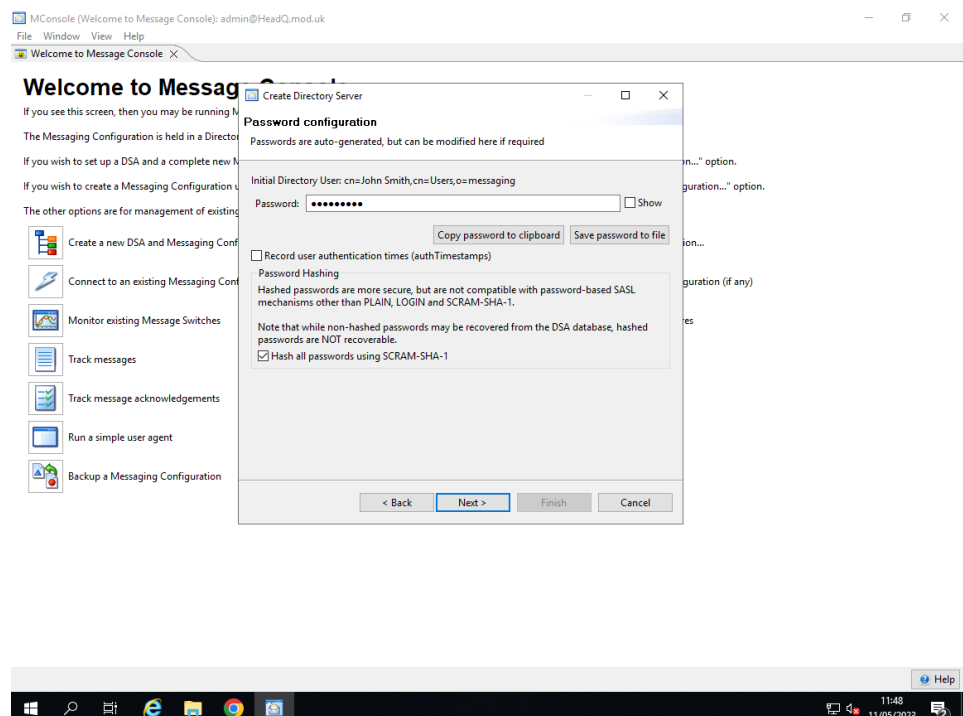
Click on OK when you have entered the User's name



**Figure 4.9. DSA Creation: DIT structure configuration**

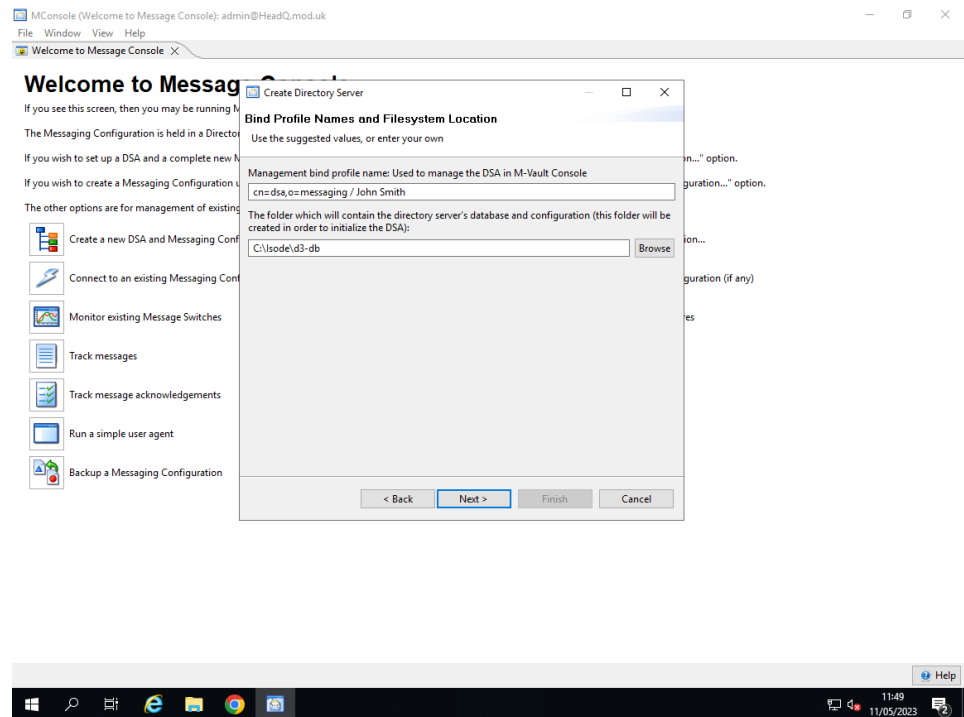
In the next screen you have the chance to change the randomly generated passwords for each of the roles from the template.

You need to consider carefully the choice of SCRAM for password hashing. Use of SCRAM improves the security for your deployment by avoiding holding passwords in the clear. However you need to ensure that in a messaging configuration in which there are multiple DSAs (for example when shadowing is being used), all DSAs must be configured to have this hashing enabled. There are also some specific SASL mechanisms (CRAM-MD5, DIGEST-MD5 and GSSAPI) which will not work if hashing is enabled.

**Figure 4.10. DSA Creation: Password configuration**

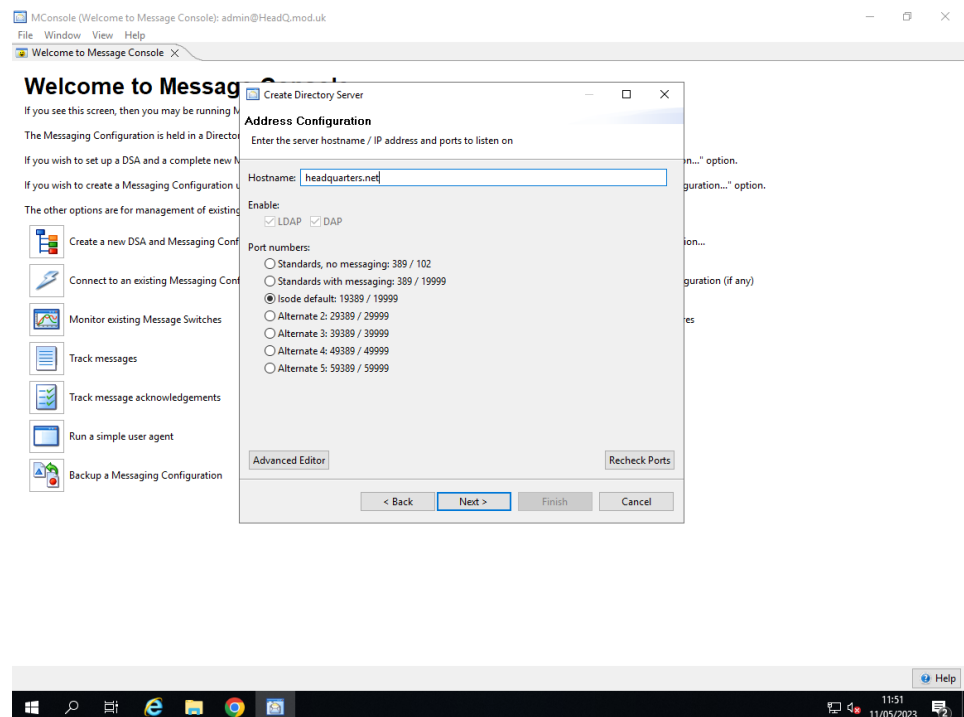
The next screen allows you to select the Bind Profile Name to use to manage the configuration in the Directory. You can also choose the directory and filename under which the Directory keeps its filestore.

**Figure 4.11. DSA Creation: Bind Profile Names and Filesystem Location**



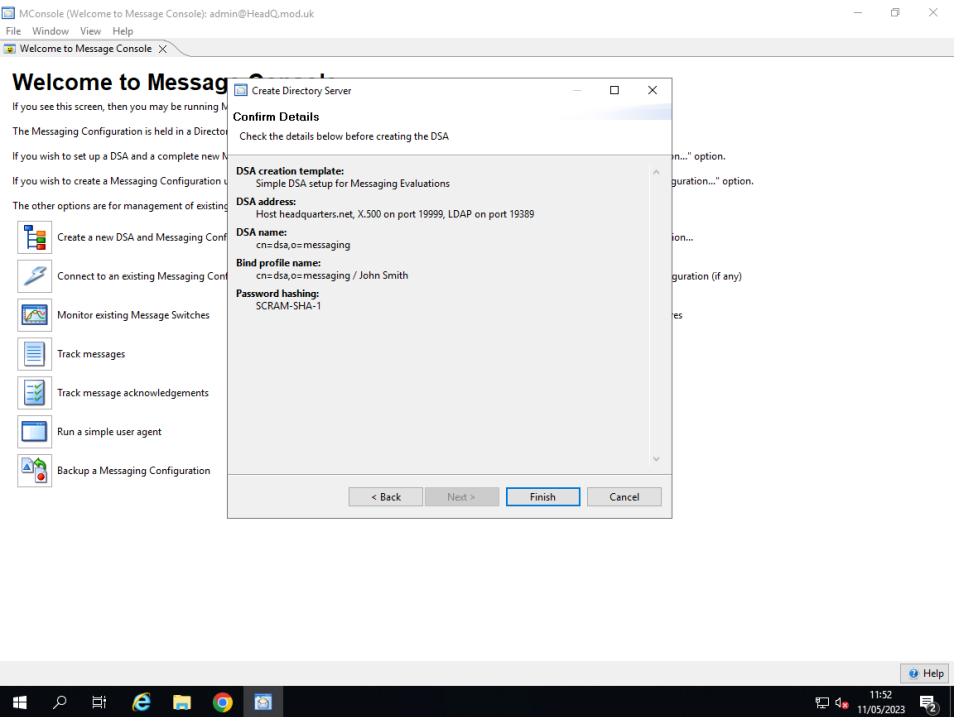
The next screen allows you to configure the addresses on which the DSA will listen.

**Figure 4.12. DSA Creation: Addresses Configuration**



The last screen confirms the details you have entered and prompts you to select **Finish**.

Figure 4.13. DSA Creation: Confirm Details



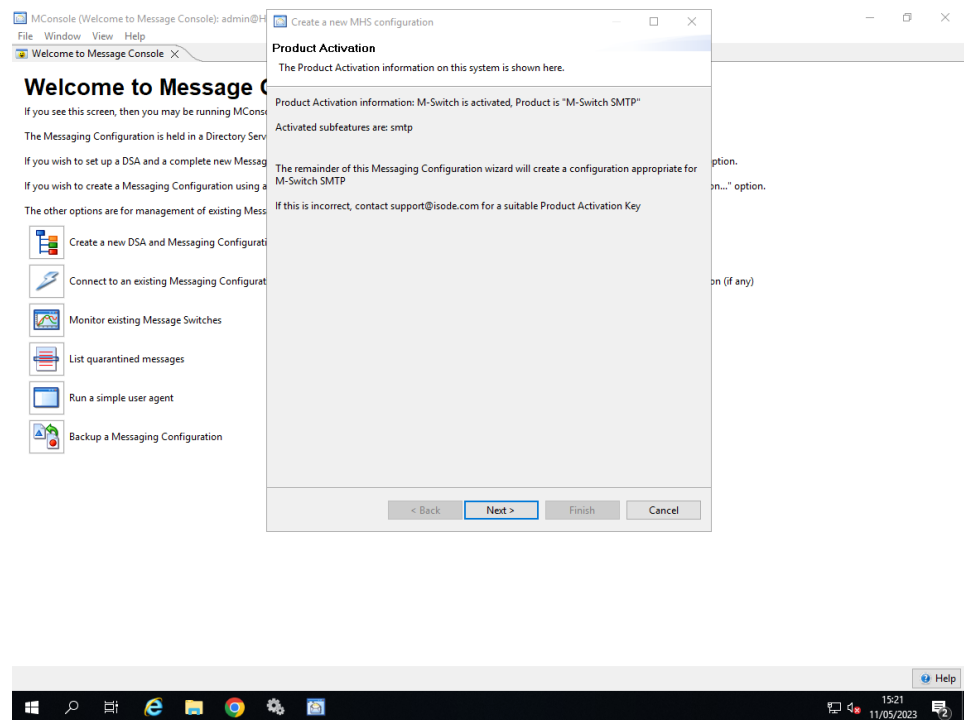
You have now completed the configuration of the DSA to be used to hold your messaging configuration. The DSA created is now running.

4.4.3      **Setting up a new messaging configuration**

4.4.3.1    **Product Activation**

The Isode Product Activation system is described in [Section 2.1, “Overview”](#). MConsole will read the local Product Activation Key (PAK) as installed in (*ETCDIR*)/*activate.dat*.

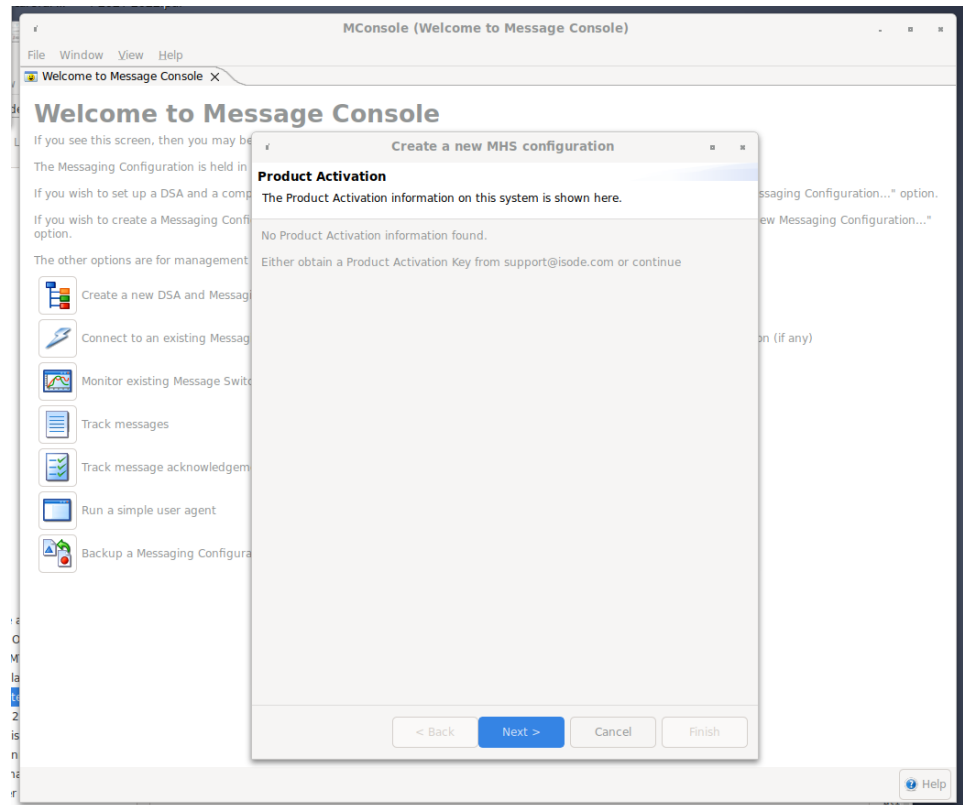
The following wizard page will then be presented.

**Figure 4.14. Messaging Configuration Creation – Product Activation**

The above screenshot shows the details of the Product Activation Key and those features and sub-features which are activated. The wizard will tailor the configuration you are creating based on these details.

Where a PAK has been found, the next pane to appear will be the pane which selects the DIT location in which the configuration is to be created: [Section 4.4.3.2, “Directory location for messaging configuration”](#).

If no `(ETCDIR)/activate.dat` is found, the Product Activation wizard page will appear as follows:

**Figure 4.15. Messaging Configuration Creation – Product Activation (No PAK)**

In this case the two additional screens below will be presented asking you to supply the Type of configuration required, and the Market Segment.

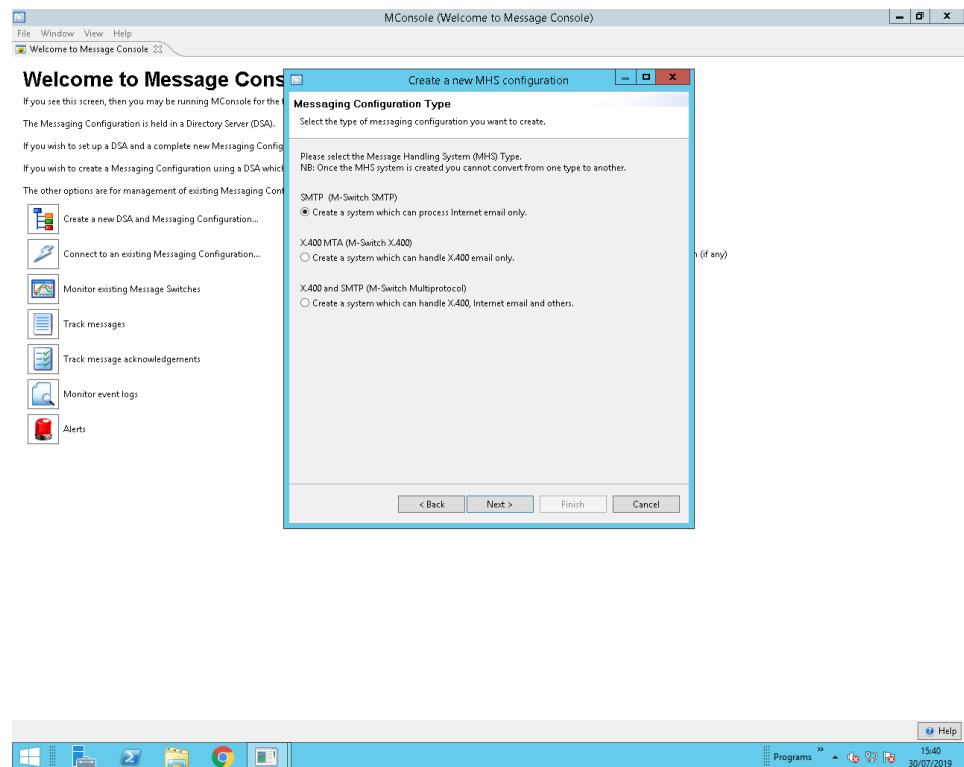
Alternatively, you may choose to ensure a suitable Product Activation Key (PAK) as installed in *(ETCDIR)/activate.dat* is provided and restart configuration creation.

#### 4.4.3.1.1 Selecting the type of messaging configuration

Where no Product Activation Key (PAK) as installed in *(ETCDIR)/activate.dat* is available, you will be prompted in the next two wizard pages for the Type of configuration (protocols), and Market Segment.

In the next wizard page, you are asked what type of messaging configuration you wish to create.

- SMTP (M-Switch SMTP)
- X.400 MTA (M-Switch X.400)
- X.400 and SMTP (M-Switch Multiprotocol)

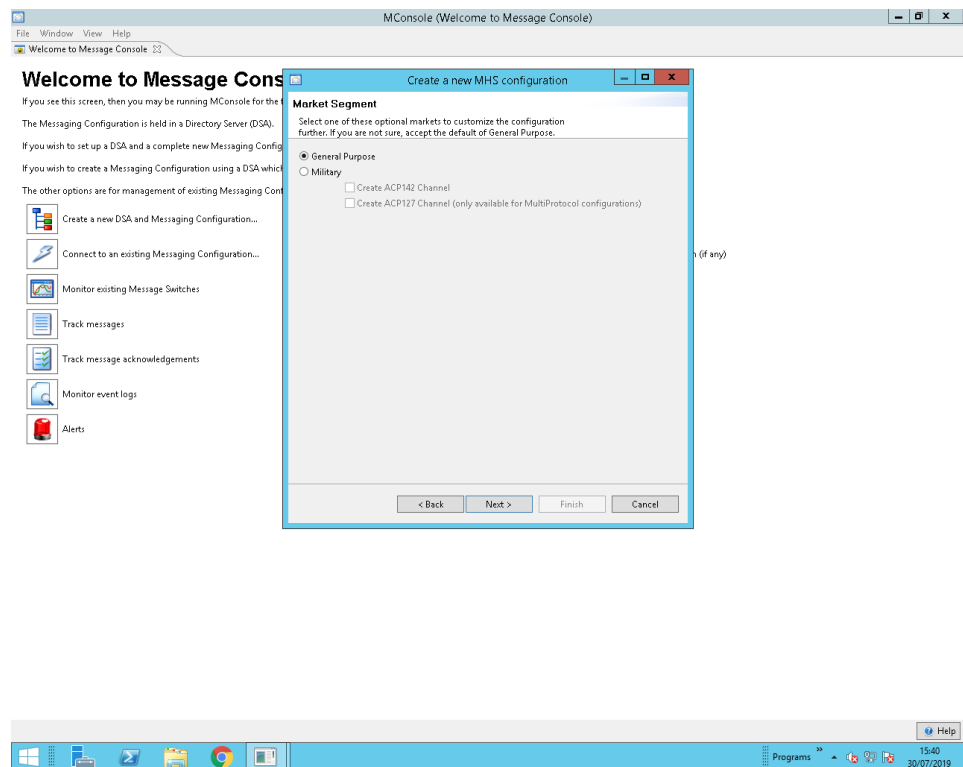
**Figure 4.16. Messaging configuration type selection screen**

#### 4.4.3.1.2 Selecting the Market Segment

In the next wizard page, you are asked what type of messaging Market Segment for the MTA you wish to create.

- **General Purpose**
- **Military**
- **Aviation**

If you are unsure, use **General Purpose**.

**Figure 4.17. Messaging Market Segment selection screen**

#### 4.4.3.2 Directory location for messaging configuration

The location in the DIT to be used for storing messaging configuration is specified in two stages.

In the wizard page shown in [Figure 4.18, “Messaging Configuration Creation – location of configuration”](#), select the **organization** entry below which the configuration is to be placed.

The default starting point is the root of the DIT. MConsole asks you to confirm the name of the organizational entry (**o=**), which is optional, and the **Messaging Configuration** entry (**cn=**) both of which it will create as required.

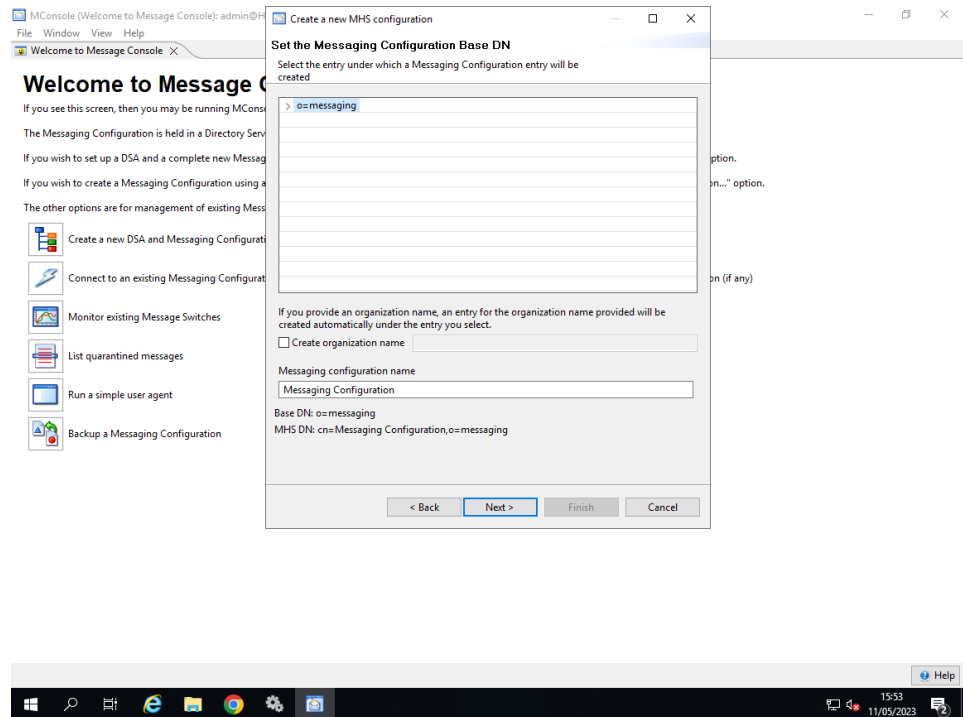
**Figure 4.18. Messaging Configuration Creation – location of configuration**

Figure 4.18, “Messaging Configuration Creation – location of configuration” illustrates an example location for a company's messaging configuration in the DIT.

#### 4.4.3.3 Next Steps

To continue creating your system, go to:

- [Section 5.1, “Creating an Internet profile”](#) to continue creating your Internet/SMTP configuration
- [Section 9.1, “Creating a configuration”](#) to continue creating your X.400 configuration
- [Section 12.1, “Creating a MIXER Configuration”](#) to continue creating your MIXER configuration.

## 4.5 Using an existing configuration

There are two places where you are given the option of accessing an existing Messaging Configuration:

- In the initial screen when you run MConsole for the first time or when all MHS configuration was previously deleted. See [Section 4.4, “Starting MConsole”](#).
- In **Switch Configuration Management** view, you select **Messaging** → **MHS Management** → **MHS Configuration Management**

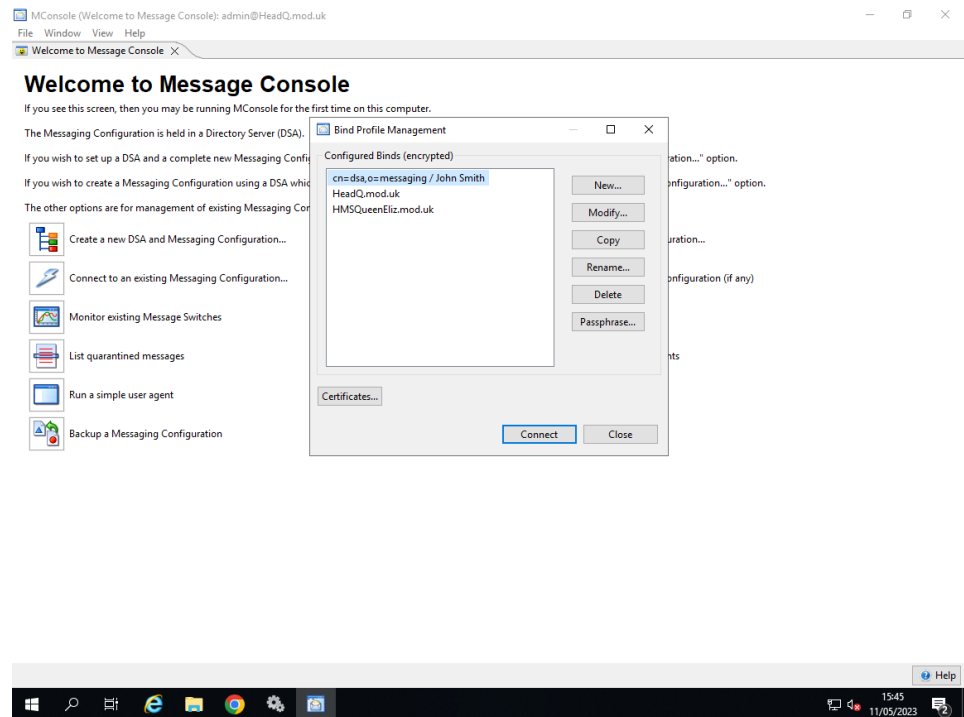
---

**Note:** The existing configuration must be one created previously using MConsole.

---

In the above two cases, you are presented with the following screen. Select the configuration you wish to access using the pull down menu.



**Figure 4.19. MHS configuration selection screen**

You have now created a profile and can now create a new messaging configuration in this profile.

[Chapter 5, \*Configuring an Internet Messaging System\*](#) describes setting up an initial Internet system.

[Chapter 9, \*Configuring an X.400 Messaging System\*](#) describes setting up an initial X.400 system.

[Chapter 12, \*Configuring a MIXER Messaging System\*](#) describes setting up an initial MIXER system.

## 4.6 Managing users

Internet users can be configured using Cobalt

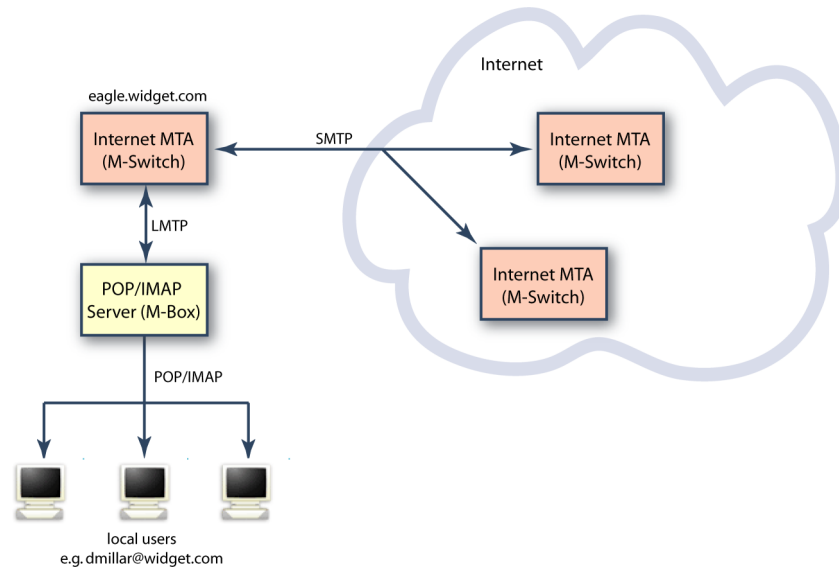
X.400 users are managed using the **X.400 Mailbox Management** view - see [Chapter 11, \*Managing X.400 Messaging Users\*](#).

# Chapter 5 Configuring an Internet Messaging System

This chapter gives instructions on how to set up an Internet messaging configuration using MConsole.

The initial set up instructions are for a basic Internet Mail configuration like the one illustrated in [Figure 5.1, “Example Internet configuration”](#). For this configuration, we will use most of the default values set by MConsole. The latter part of the chapter describes how you can extend and tailor the configuration.

**Figure 5.1. Example Internet configuration**



## 5.1 Creating an Internet profile

[Section 4.3, “Creating a messaging configuration”](#) described how to start the creation of a bind profile and how to specify where the new profile should be held in the Directory. You were then presented with a window, shown in [Figure 4.16, “Messaging configuration type selection screen”](#) inviting you to choose the type of configuration you wish to create.

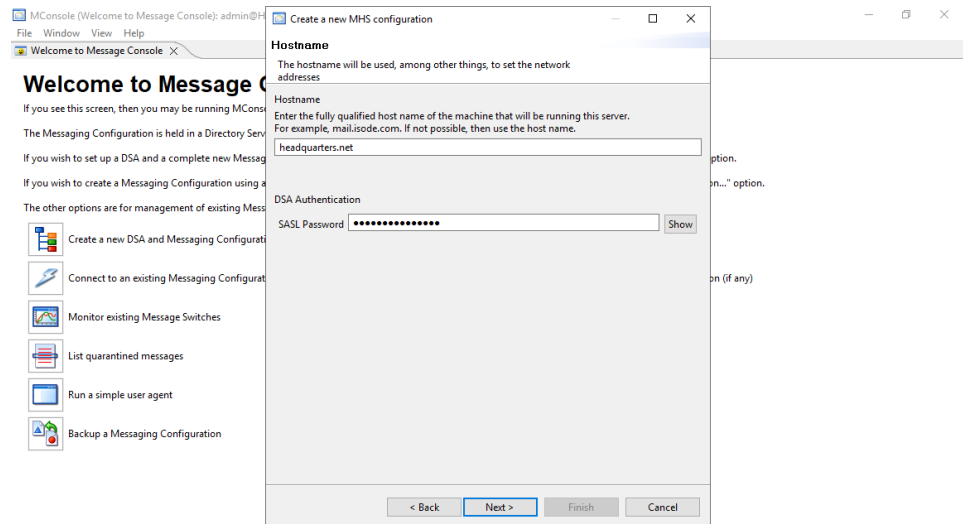
As your Product Activation Key determined that an **SMTP** should be configured, you are now ready to configure the remainder of your Internet/SMTP configuration.

### 5.1.1 Configuring the Hostname

In the next wizard page, you are asked to enter the fully qualified hostname for the MTA you wish to create. The hostname would be in the value of MX records for the Email domain. See [Section 5.1.2, “Configuring the SMTP channel specific entry”](#) to configure this.

You are also asked to provide the SASL Password in order to authenticate to the DSA. See [Section 13.1, “Overview”](#) for a description of SASL authentication.

Figure 5.2. Fully qualified hostname screen



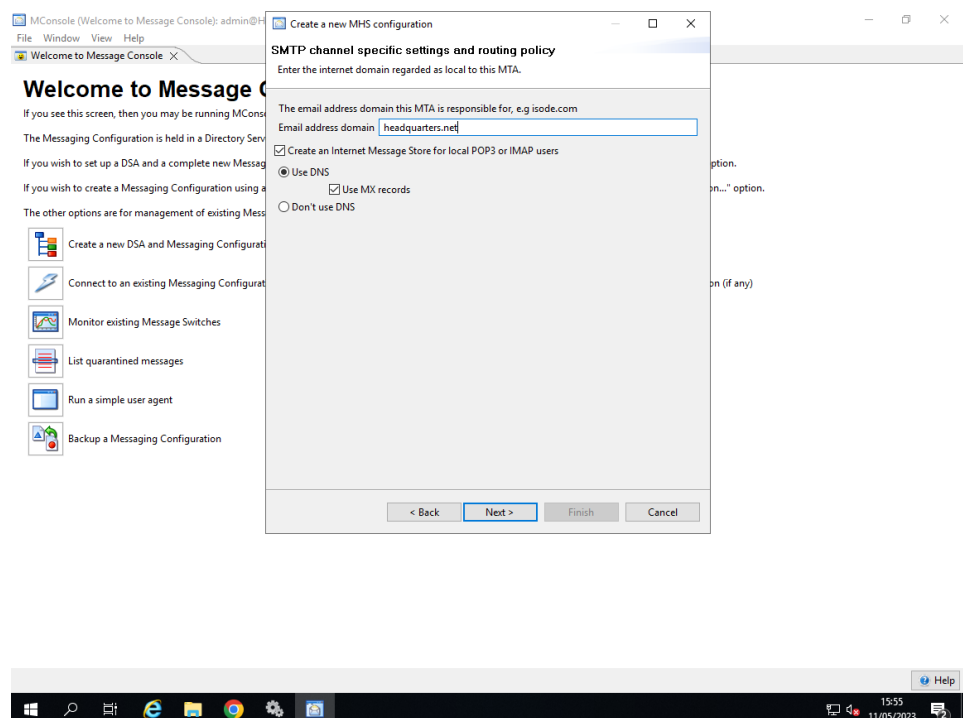
## 5.1.2

### Configuring the SMTP channel specific entry

In the next wizard page, you are asked to enter SMTP channel specific information. You need to enter the internet domain - i.e. the domain local to this MTA, e.g. **mydomain.example.com**. You also need to configure lookup information, i.e. whether DNS is in use. You would normally select the **Use DNS**. If not you will need to read the *M-Switch Advanced Administration Guide* and perform some reconfiguration.

Lastly you can choose whether or not to configure an Internet Message Store for local POP or IMAP users (eg M-Box).

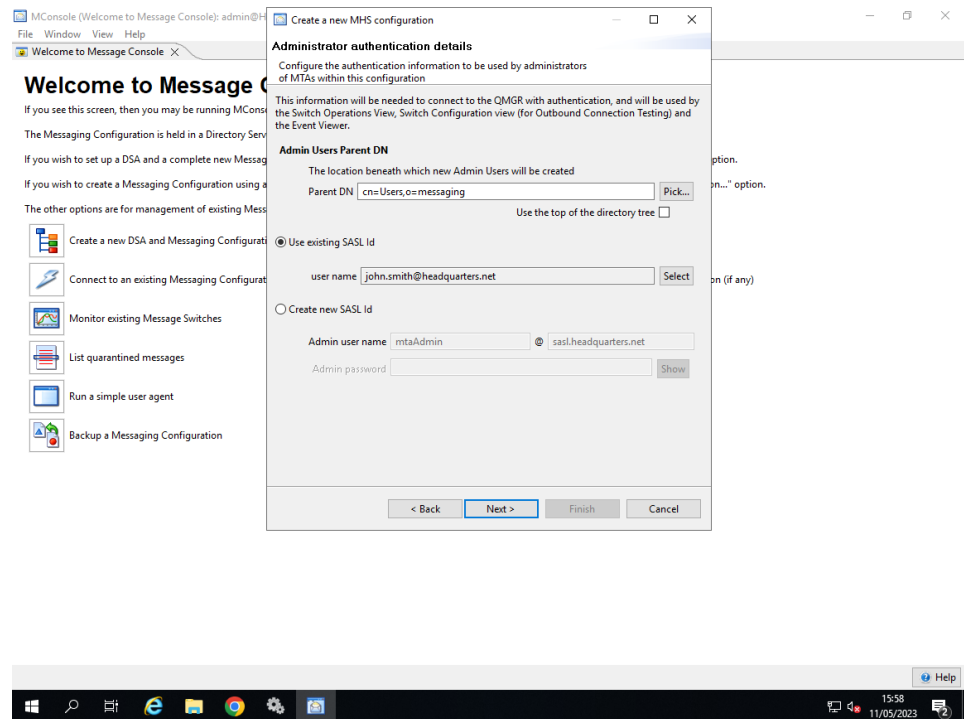
Figure 5.3. Net domain screen



## 5.1.3 Configuring the administrator authentication

In the next wizard page, you are asked to enter Administrator authentication in the form of a SASL User ID and password. These are the credentials used by MConsole when connecting to the Queue Manager using SOM.

**Figure 5.4. Administrator Authentication page**



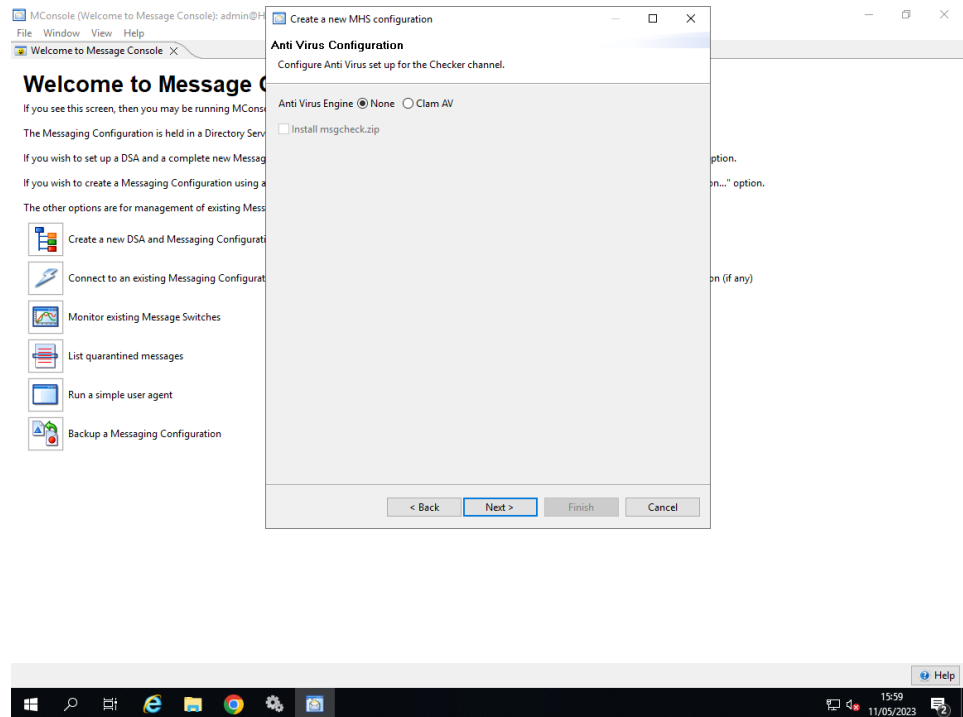
## 5.1.4 Configuring Anti virus and services

M-Switch can be configured to perform content checking in which messages containing Viruses or are considered to be spam are detected and action suitable action taken to counter these abuses.

See [Section 40.1, “Checking message content”](#) for a full description of Content Checking.

This wizard page allows you configure Anti Virus (which you can also configure later, if you wish). You can choose to have no Anti Virus Engine, Clam AV or Sophos. If an engine is selected you will be informed of the platforms on which it is supported and you can choose whether or not to install msgcheck.zip.

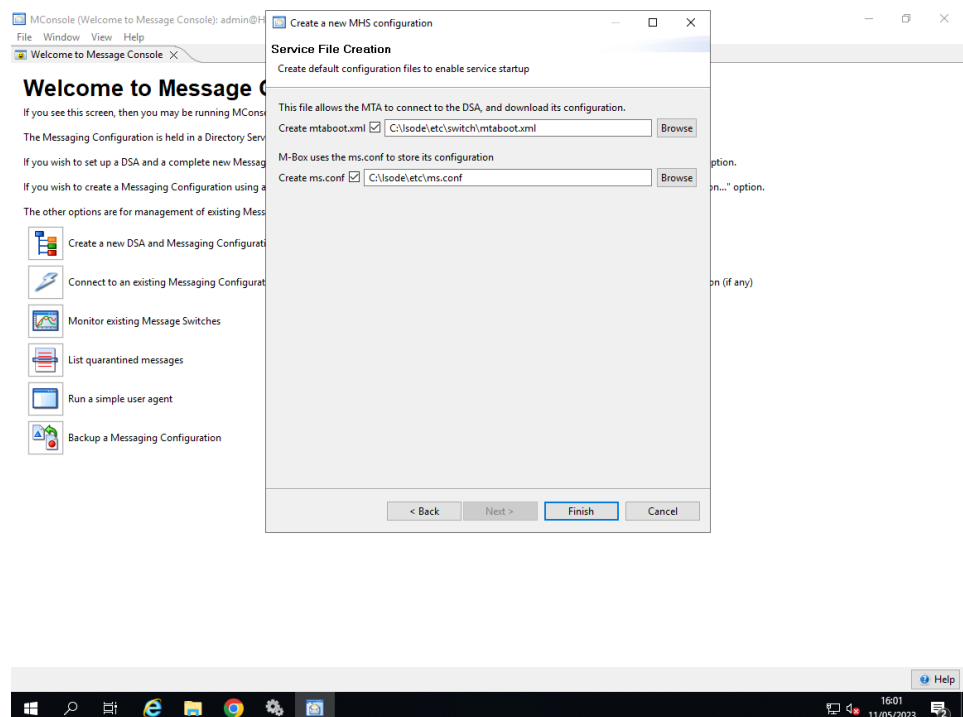
You should only install msgcheck.zip if you have the rights to do so and if MConsole is running on the same machine as the configuration you are creating

**Figure 5.5. Configuring Anti Virus**

## 5.1.5 Service File Creation

You can now save the config files to be used by M-Switch and M-Box onto the filesystem. If the servers are to run on the system on which you are running MConsole, you can save directly into the live config file so the services are ready to be created and run.

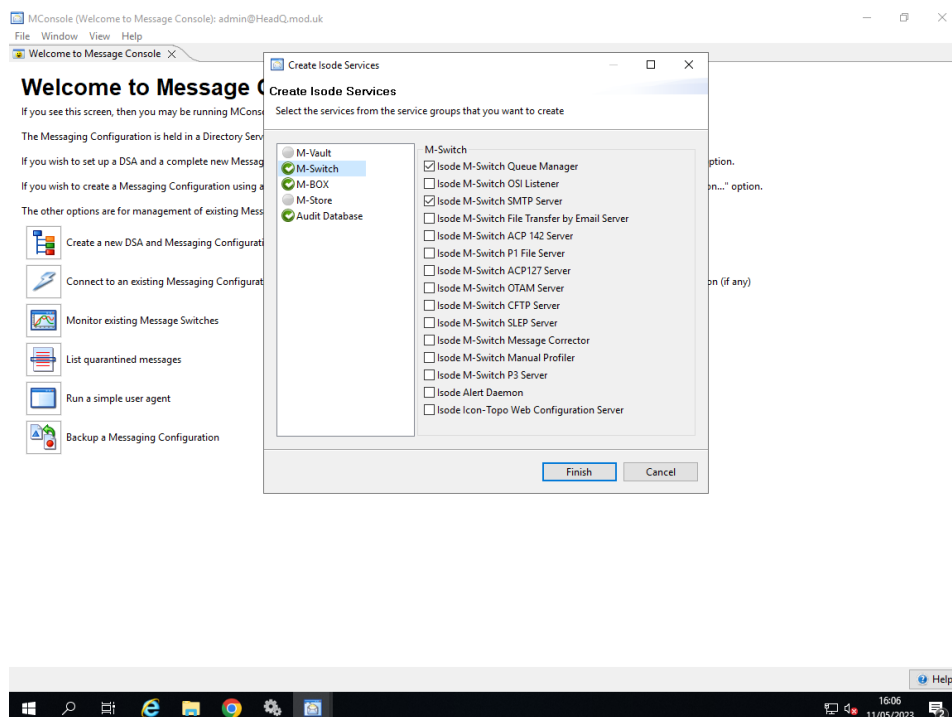
Alternatively, you can save the files elsewhere or not save them at all and do this later from the **Switch Configuration Management** view of MConsole (by right clicking on the MTA).

**Figure 5.6. Service File Creation page**

## 5.1.6 Creating Windows Services

The services are created with default dependencies. You may wish to review the dependencies of the services to make sure they match your environment.

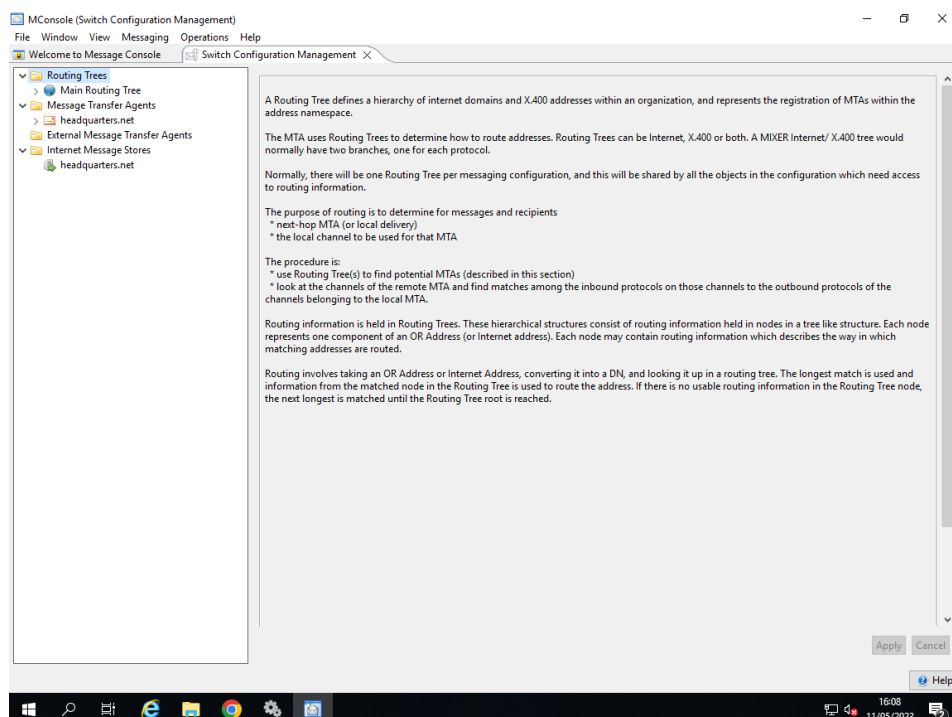
**Figure 5.7. Creating Windows Services**



## 5.1.7 Completed Internet messaging configuration

After clicking on **Finish**, the configuration of the completed SMTP configuration is created and displayed ready to view and edit.

**Figure 5.8. Internet configuration complete screen**



## 5.1.8 Creating and starting services

### 5.1.8.1 Creating and starting services on Unix

You can now start your messaging services.

The services will already have been created and can be started by running the commands

```
service pp start
```

```
service mbox start
```

### 5.1.8.2 Creating and starting services on Windows

On Windows right click on **Start** → **Isode** → **Isode Service Configuration** and choose **Run As Administrator**.

If running for the first time you will only see the DSA service (created by MConsole when the DSA itself was created). In this case select **Operations** → **Create Service**.

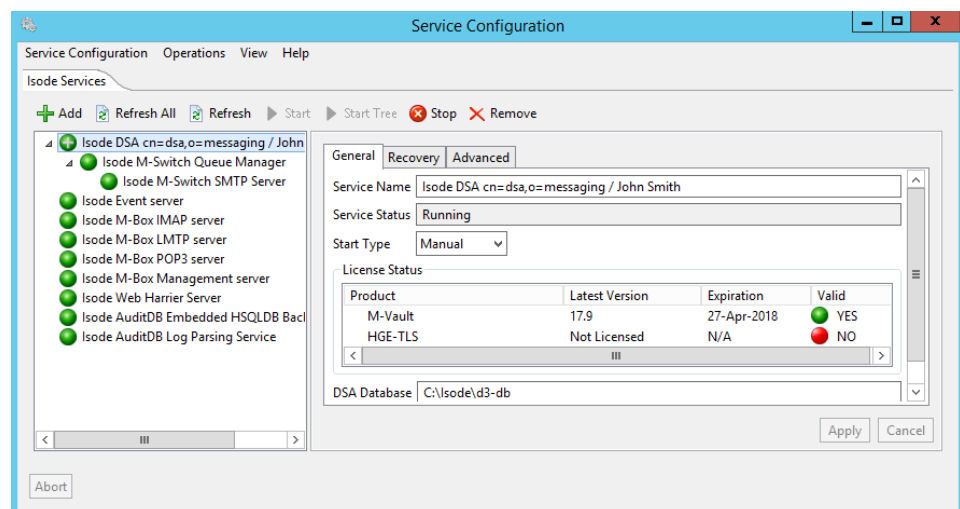
Select the **M-Switch**, **M-Box** services.

You might also want to consider selecting the following services by selecting the **Audit Database** service

You should set the services you wish to start at system boot, or when you click on Start All. Normally this will be the **M-Switch** and **M-Box** services.

You can then start all the services by clicking on **Operations** → **Start All**.

**Figure 5.9. Starting Windows Services**




---

**Note:** This step also configures the embedded SQL database HSQLDB for use by the Message Audit System. See [Chapter 36, Message Audit Database](#) for a description on how to do this. HSQLDB is suitable for evaluations. For deployments you should use PostgreSQL or Microsoft SQL Server.

---

## 5.1.9 Next steps

Your basic configuration is now ready.

Other parts of your system that you may also want to configure are:

- Managing Internet Users, [Chapter 6, Managing Internet Users Using LASER Routing](#).

- Message Audit Database (to configure message tracking). See [Chapter 36, \*Message Audit Database\*](#).
- Content Checking (eg Filtering Spam) [Chapter 40, \*Content Checking\*](#).

---

## 5.2 Editing your configuration

Having created a configuration using the MConsole wizard, you have created a configuration comprising

- one Internet Routing Tree
- one Internet MTA
- (optionally) one POP/IMAP Message Store

The configuration may need amending or enhancing to add additional MTAs. It is likely that for most simple setups you will not need to enhance your configuration in other ways.

---

## 5.3 External MTAs

The MTA configured in [Section 5.1, “Creating an Internet profile”](#) has created a tailoring, shared config MTA. This is an MTA which contains all the tailoring and routing configuration needed by an Isode MTA.

For most purposes you do not need to configure other MTAs as MX records in DNS will enable your MTA to interwork with these MTAs.

To connect to other MTAs you can configure the MTA as

- a full tailoring shared config MTA
- an external MTA

The advantage of a shared config MTA is that a single Directory-based configuration can be shared by multiple Isode MTAs. The means that each MTA is configured in a single place and updates to each MTA take place immediately.

External MTAs are used when the configurations are not shared. When connecting to non-Isode MTAs, you can only configure the remote MTA as an external MTA.

---

## 5.4 What has been created

To check what has been created by the creation wizards, select **Switch Configuration Management** from the **View** menu. Go through the **Switch Configuration Management** view and note the points made in the following subsections.



## 5.4.1 Message Transfer Agents (MTAs)

To see all the MTA tailoring properties, select the MTA from the **Message Transfer Agents** folder in the MConsole **Switch Configuration Management** view.

### 5.4.1.1 MTA properties

Select the MTA in the list on the left and the page on the right shows various attributes for this MTA. The creation wizard has set suitable values for a basic system, so you should not need to alter any of the MTA values at this point. However, should you want to tailor values later on, details of all the fields can be found in the *M-Switch Advanced Administration Guide*.

### 5.4.1.2 Channels

Suitable channels have also been created, which can be seen by expanding the **Channels** node, underneath the **MTA** node. Channel properties are displayed by selecting the channel in the page on the left: the properties are displayed in the page on the right. Again, you may want to tailor channels and their properties later. The various property fields are described in the section on 'Channel Entries' in the *M-Switch Advanced Administration Guide*.

The following gives a summary of the function of the channels created in the basic Internet MTA. Cross references do not need to be followed when creating your initial system, but you may find them useful later when you want to extend the functionality of your system:

#### 822-local

This is the channel used for delivering messages to users configured as local to this MTA. Adding and tailoring local delivery facilities is described in the section on file based distribution lists appropriate for Internet configurations, see the chapter called 'Handling Messages Locally' in the *M-Switch Advanced Administration Guide*.

#### housekeeper

This is a special housekeeping channel used by the MTA to perform tasks such as deleting message when they have been processed, or generating reports. Further information on this channel can be found in the chapter called 'MTA Tailoring' in the *M-Switch Advanced Administration Guide*.

#### ftbe

(optional) This is the channel program which is used to transfer emails to and from the filesystem. Typically it's used in conjunction with Sodium Sync to allow directory replication over email. Configuration information for the fteserver is given within the *M-Switch Advanced Administration Guide*.

#### lmtip

This channel is used for delivery to the POP/IMAP Message Store (usually M-Box).

#### mimecheck

This channel carries out content checking of messages such as virus and spam checking.

#### mimeshaper

This channel carries out content conversion

#### smtp

is the main protocol channel for handling messages coming into the MTA and transferring messages out of the MTA. Three SMTP channels are created: smtp-auth, smtp-dl, smtp-internal and smtp-external.

- smtp-auth is used for authenticated message submission, and usually runs on port 587
- smtp-dl is used for handling the resolution of distribution lists using file-based lists. Configuring distribution lists is described in the chapter called 'Handling Messages Locally' in the *M-Switch Advanced Administration Guide*.

- `smtp-internal` is used for local inbound SMTP transfers, where the source MTA is authorised by matching an **Inchan** rule in the Authorization configuration for the MTA. This is typically a list of local, trusted MTAs
- `smtp-external` is used for all other SMTP transfers.

Typically the authorization system is used to ensure that a message which enters from a trusted MTA (i.e. an MTA which triggers an Inchan Authorization rule) can be treated in a way that is trusted, e.g. by permitting relay or not performing content checking such as checking for spam. See [Section 38.1, “Authorization”](#) for further details.

As we shall see when we come to start the MTA, some protocol channels need to be started explicitly. Simple configuration of SMTP channels is carried out using the various tailoring options available for the channel. These are described in the chapter called ‘MTA Tailoring’ in the *M-Switch Advanced Administration Guide*.

### 5.4.1.3 Tables

A number of tables can be displayed by expanding the **Tables** node in the left hand page, below the **MTA** node.

The MTA creation wizard sets up the domain and channel tables so that simple DNS based routing works. In many cases this will be sufficient for an Internet MTA. For the more advanced features of M-Switch routing the entries in these tables can be modified. See *M-Switch Advanced Administration Guide* for details of how to do this.

Tables may also be required for configuration of facilities such as distribution list handling. Even when using Directory-based routing, some configuration information must be held in tables. The configuration of these tables is mandatory but they can be configured to be empty. If not empty these tables can be held as text files or in the Directory.

Configuration can be held solely in tables instead of in the Directory. Such a configuration is described in the *M-Switch Advanced Administration Guide*.

### 5.4.1.4 Logging

Information about the three log streams created by the MTA creation wizard can also be displayed in this window by expanding the **Logs** node. Their properties can be displayed and edited in the same way as other tailoring entities. The logging tailoring set here is adequate for your basic system. [Section 34.2, “Configuring logging”](#) describes how you can tailor the level of logging required for individual programs, channels and protocol layers. You can also specify that the logging be sent to separate files for each M-Switch program.

---

**Note:** The remaining sections in this chapter describe some of the ways in which you might want to tailor or extend the basic configuration. However, if you are setting up a basic system the next task is to set up some users. We suggest that you turn now to [Chapter 7, Managing Internet Messaging Users](#) for instructions on how to do this. You can return to the following sections later, when you have a basic messaging system working.

---

### 5.4.1.5 Internet Users and Directory Profiles

Directory Profiles configure how access to the Directory (or multiple Directories) take place. This includes hostname/port, search base, authentication, and mapping of attributes by LASER. A "Default" Profile is created when the MTA is created which provides for most straightforward configurations. Most of the values in this "Default" profile are empty, which means that MTA wide values are used to access the Directory.

See [Section 6.2, “Directory Profiles”](#) for a full description of how to use Directory Profiles.

## 5.4.2 Routing Trees

For a full description of the way in which Routing and Lookup Policies work in M-Switch, see [Chapter 15, Routing](#).

Routing information for local Internet users is held in the Directory and looked up using LASER. This involves an LDAP search in a subtree in the Directory. This search is configured using a Directory Profile as described in [Section 6.2, “Directory Profiles”](#). For most configurations, the **Default** Directory Profile is suitable.

This LASER search applies whether domains are Routed using table based Lookup Policies or Directory based Lookup Policies

For a full description of LASER lookup in M-Switch, see [Chapter 6, Managing Internet Users Using LASER Routing](#).

### 5.4.2.1 LASER Routing Using Routing Trees

For Internet Configurations by default a lookup policy of **ds** or **dns-ds** is used which cause routing lookup to use Routing Trees to route domains and sub domains.

In the **Switch Configuration Management** view, click on the **Routing Trees** folder to reveal the **Main Routing Tree**, representing the message routing information which has been set up in the Directory for the domain. Subdomains can be added as nodes or deleted, by right clicking on a particular node, and selecting **Add Node** or **Delete**. Properties of each of the nodes can be displayed by selecting the node. These include:

- Directory subtree information.
- Routing Information which can be held in one of two ways:
  - MTAs which can route to this domain and associated channel information. Configuring the channels to be used to transfer messages is covered in the *M-Switch Advanced Administration Guide*.
  - Routing Nexus which point to a Nexus which provides a logical indirection to MTAs which can route to this domain. See [Section 15.4.2.3, “Nexus Information”](#) for a full description of how Nexus work.

### 5.4.2.2 LASER Routing Using Routing Tables

Internet Configurations can alternatively use a lookup policy of **dns-table-laser** or **table-laser**. This means that routing information for domains is held in the domain and channel tables. In addition DNS lookups are used prior to the table lookups in the case of **dns-table-laser**.

This form of Routing is deprecated in favour of **ds** or **dns-ds**.

# Chapter 6 Managing Internet Users Using LASER Routing

This chapter describes how to setup LASER routing to correctly route email for M-Switch users.

---

## 6.1 Overview

LASER was specified in a draft Internet Standard, and specifies a way of configuring local delivery of internet email, using a standardized (LDAP) Directory schema. The way in which LASER is implemented by M-Switch makes it possible for user-specific information to be held in a separate (possibly non-Isode) Directory from that used for the Messaging Configuration, allowing integration of a messaging system with existing infrastructure. This manual will describe how to configure LASER lookup within the Isode Directory.

The basic idea is very simple: the MTA gets routing information from the LDAP Directory using a search for the email address.

If X.400 addresses are to be routed as well, then two or more lookup policies may need to be configured.

LASER lookups search the Directory, and the way in which this LASER configuration is accessed and used is held in a Directory Profile table. See [Section 6.2, "Directory Profiles"](#).

When the MTA creation wizard creates an MTA this table is by default created to be held in the Directory. Like other tables, this can also be held as a text file.

### 6.1.1 Routing

When routing an Internet email address, the first stage determines

- if the domain is valid
- if the domain is local
- if it is local, if there is a table prefix to be used later.

Processing the address if the domain is not local is done either as for the `dns` policy (i.e. using `smtp` channels), or as for the `table` lookup policy (i.e. using the channel table).

If the domain is local, then the address is looked up using LDAP in a directory server. If no matching entry is found, or if multiple entries are found, then the routing fails. In the latter case a non-delivery is forced with a diagnostic of "ambiguous".

If exactly one matching entry is found, then certain attributes are read from that entry and used for both routing, and for channel-specific purposes. Which attribute type is used for each purpose can be configured, as outlined below.

For a more complete explanation of Routing see [Chapter 15, Routing](#).

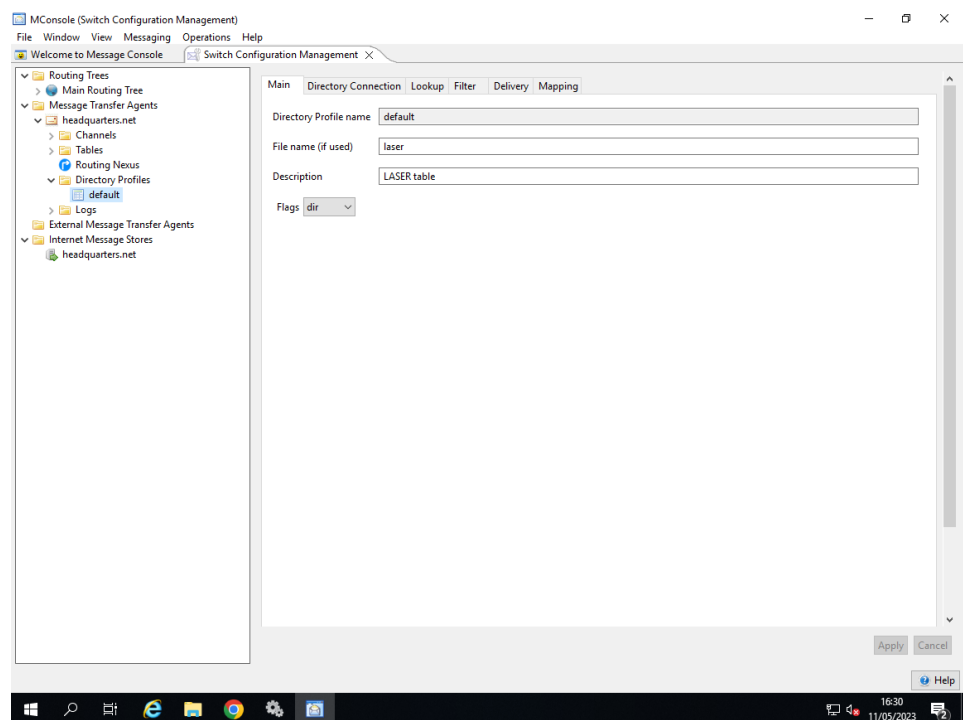
## 6.2 Directory Profiles

Configuration information is held in Directory Profiles as presented by MConsole.

The following figure shows the default Directory Profile as setup when the MTA is created by MConsole.

**Note:** Directory Profile names are stored in the Directory mapped to a different value. Apart from `default` which is stored as `cn=laser`, all other tables are stored as the value `cn=<table-name>-laser`.

**Figure 6.1. Default Directory Profile in MConsole**



When Routing information needs to be looked up using LDAP, this is a result of M-Switch using a Routing Tree to route an address, or an Address Conversion Table to perform a MIXER mapping of an Internet or X.400 address.

Each node of a Routing Tree may have a `laserMapping` attribute configured which is set to the name of a Directory Profile. This causes any LASER lookup resulting from the use of this Routing Tree node to use the Directory Profile values to be used to perform the lookup. This allows, for example, Delivery information to be held in a different Directory or to use different attributes for Routing. The same applies to Address Conversion tables allowing a different Directory to be used to hold MIXER mappings.

A Directory Profile is actually a configuration table. When a configuration is created by MConsole it sets up a table called `default`. Other laser tables have an associated prefix, i.e. `<prefix>-laser`. The prefix can be associated with the lookup policy (e.g. `dns-laser=foobar`) or can be associated with a specific domain in the domain table (value such as `local=foobar`). This enables the MTA to search different parts of the DIT, or even different servers, for different domains in the address.

If values are not found in the appropriate table, then suitable defaults are used, as indicated below. For the actual routing, the defaults use the LASER attributes as defined in the IETF draft.

## 6.3 Directory Profile Content

There are six tabs in a Directory Profile which can be defined. All configuration values can be left blank and sensible defaults are used.

This means that for many straightforward configurations, no values need to be configured. See [Section 6.4, “Simple Configuration”](#) to understand when the simple defaults are likely to be sufficient.

### 6.3.1 Main

The Main tab holds identifying information for the profile.

Directory Profile Name

The name of the profile. Used to reference this profile in other configuration locations. Note that the value of "default" is a special value which will be used unless a different Directory Policy is created and configured to be used.

Filename

The name of the file used to hold the data when Flags values is "linear".

Description

A short text description of the Directory Profile.

Flags

One of the following:

DBM

Stored in binary (database) format. This file is created from the text file when the dbmbuild command is run either from the command line, or the Queue Manager when downloading the configuration.

linear

Stored in a text file.

dir

Stored in the Directory.

empty

Contains no entries. This used in the case where a table is mandatory, or when all the values are in overrides).

### 6.3.2 Directory Connection

The Directory Connection tab controls which LDAP server is used and how to bind (if at all)

If the LDAP bind information is not explicitly set, then the same information is used as is used for the x500\_access=ldap case. To configure them explicitly, use the values below.

LDAP Server

Host (ldap-host)

The name of the ldap host. This can be a space separated list of hosts. The names can be optionally followed by : (a colon) and a port number.

Port (ldap-port)

If ldap-host does not contain a port number, then this is used. The port number defaults to the LDAP standard port: 389.

Server Authentication

Authentication Mechanism (sasl-mechanism)

The SASL mechanism to use. This can be omitted, in which case one of the shared available mechanisms will be used. If set to "simple", it forces a simple bind (DN and password). If set to "none" it forces no binding.

Authentication Data

User (ldap-name)

The DN to use if a simple bind is to be used.

Password (ldap-password)

The password to use in a simple bind.

### 6.3.3 Lookup

The Lookup tab configures how lookups are performed.

When to dereference aliases (ldap-aliases)

Controls the dereferencing of aliases. The default is 3: always. This is set to one of the following numeric values:

- 0 (never)
- 1 (searching)
- 2 (finding)
- 3 (always)

Search Base (search-base)

The base entry for the search (default is the root of the DIT).

Search timeout (search-timeout)

A numeric value for a timeout, in seconds.

Search scope (search-scope)

If set to the string "single-level" then a single-level search is performed. Otherwise a subtree search is performed.

### 6.3.4 Filter

The Filter tab allows the user to configure filters used in LDAP searches.

Routing Search Filter Attributes (filter-atts)

A space separated list of attribute types which are used to construct the search filter. If more than one is specified, then the filter is an 'OR' of the filters for each attribute type. Each attribute filter performs an exact match for the address. Note that if the first search is unsuccessful, then a second search is performed with the local-part of the address removed (but including the '@'). This allows 'wildcard' entries to be configured in the DIT for a whole domain.

Extra Routing Search Filter (extra-filter)

An LDAP search filter, which is combined using 'AND' with the filter constructed using filter-atts. This is used to constrain the entries being matched for routing lookups.

Mixer Internet Attributes (mixer-internet-atts)

A space separated list of attributes to be used to construct the search filter for MIXER address conversion from Internet to X.400. If more than one is specified, then the filter is an 'OR' of the filters for each attribute type. Each attribute filter performs an exact match for the address.

**MIXER X.400 Attributes (mixer-x400-atts)**

A space separated list of attributes to be used to construct the search filter for MIXER address conversion from X.400 to Internet. If more than one is specified, then the filter is an 'OR' of the filters for each attribute type. Each attribute filter performs an exact match for the address.

**Extra Internet Search Filter (mixer-filter-internet)**

An LDAP search filter, which is combined using 'AND' with the filter constructed using mixer-internet-atts. This is used to constrain the entries being matched for MIXER mapping lookups.

**Extra X.400 Search Filter (mixer-filter-x400)**

An LDAP search filter, which is combined using 'AND' with the filter constructed using mixer-internet-atts. This is used to constrain the entries being matched for MIXER mapping lookups.

## 6.3.5 Delivery

The Delivery tab is where the user can configure the default delivery host and default delivery channel.

**Default Delivery Host (default-host)**

Sets the default host to which the mail should be transferred, if non-local.

**Default Delivery Channel (default-channel)**

Sets the default channel to be used for local delivery. Defaults to "lmt".

## 6.3.6 Mapping

The Mapping tab controls the routing or mapping (as appropriate for LASER or MIXER mapping).

**laser-atts**

This is a list of attributes to retrieve from the server for the entry. Each attribute in the list MUST have a corresponding type, so the routing lookup knows how to treat the attribute if found.

Additionally, there should be an entry in the table for each attribute type listing in the laser-atts list. The key is the attribute type name, and the value is one of the keywords which describes how the value is to be interpreted. Note that the concepts are closely related to table-based routing.

Routing values:

**Alias**

The value is an alias address (default mailRoutingAddress)

**Synonym**

The value is a synonym address

**External Synonym**

The value is an external synonym address

**Host**

The host for transfer (default is mailHost)

**Channel**

The channel for local delivery (default is channel)

**Group**

The list of groups to which a user belongs (default is mhsUserGroup)

**Closed User Group**

The list of groups to which a user belongs (default is mhsClosedUserGroup)



**MIXER Internet Mapping**

The attribute used to hold the Internet value of a Per User Mapping. The default value is mail)

**MIXER X.400 Mapping**

The attribute used to hold the X.400 value of a Per User Mapping. The default value is mhsORAddresses)

**Attributes to ignore**

List of any default attributes that aren't to be used if found during a LASER search.

---

**Note:** If the address being used matches the value in an alias or synonym, then the alias or synonym is ignored.

---

If the value for the host matches the local MTA's name, then that value is also ignored.

After removal of attributes which match the address used for lookup, there should be at most one external-synonym value, and at most one alias or synonym. If there is both an external-synonym and an alias or synonym, then the external-synonym is used.

## 6.3.7 Local delivery channel values

These replace the values in the channel configuration.

**numeric-uid**

Numeric UID used for the delivering process (Unix)

**numeric-gid**

Numeric GID used for the delivering process (Unix)

**username**

Username used to get UID, GID, home directory

**mailbox**

Name of mailbox file to use

**directory**

Directory in which mailbox file will be found

**mailformat**

Format of mailbox file

**mailfilter**

Name of mailfilter file

**sysmailfilter**

Name of system mail filter

**path**

Search path

**restricted**

Boolean indicating if user is restricted

**Default values:**

- filter-atts is "mailLocalAddress"
- laser-atts is "mailHost mailRoutingAddress"
- mailHost is "host"
- mailRoutingAddress is "alias".

This matches the LASER routing draft RFC, with local delivery by LMTP.

## 6.4 Simple Configuration

When creating an Internet or MIXER configuration, a default Directory Profile is created. For simple configurations this will be sufficient to allow LASER routing to work if

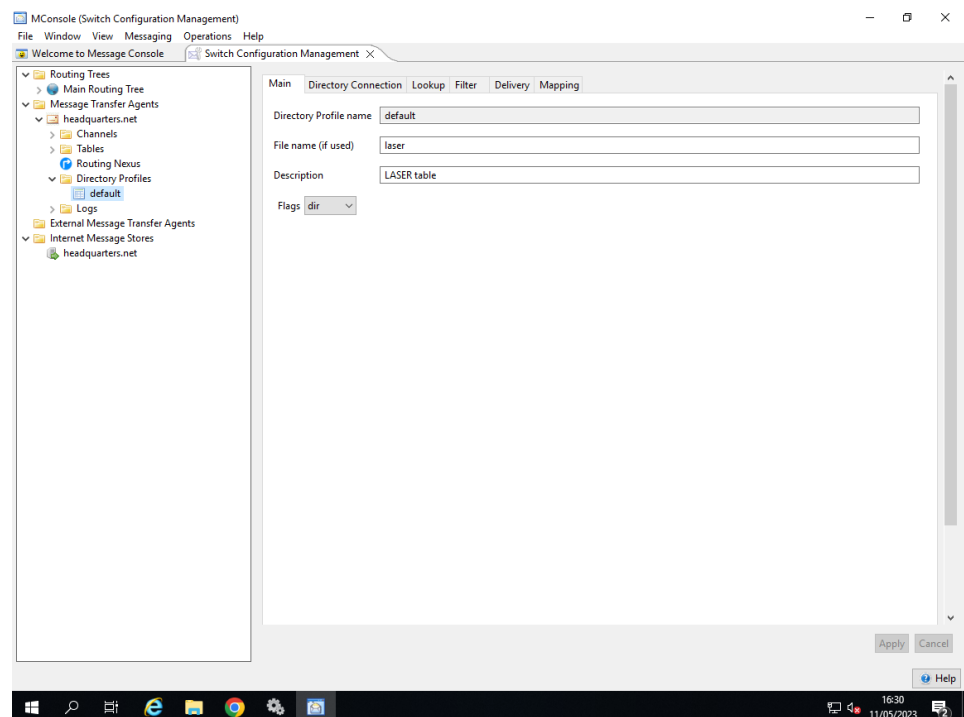
- the users are held in the same Directory as the Messaging Configuration
- the standard attributes are used

### 6.4.1 Lookup Policy

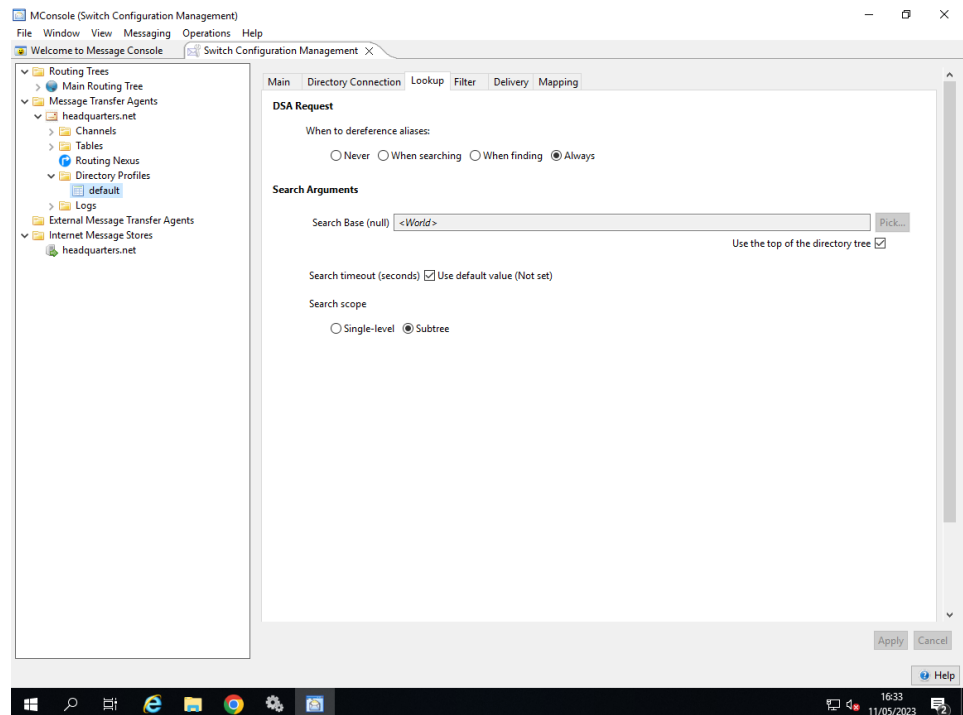
Make sure the lookup policy of the MTA uses a LASER lookup policy, for example "dns-ds". NB: using ds or dns-ds implies the use of LASER after establishing by a DNS and/or Directory lookup that the address is local. You do not need to configured a lookup policy that specifically uses LASER, e.g. dns-table-laser. See [Section 15.3, "Lookup Policies"](#) for a full description of how lookup policies work.

Below is the Directory Profile as created by MConsole with minimal configured values.

**Figure 6.2. Directory Profile in MConsole**



The Figure below shows the Lookup Tab of the default Directory Profile after creation, i.e. using the default values.

**Figure 6.3. Directory Profile in MConsole - Lookup Tab**

You can make the search more efficient by configuring a suitable search base. Note that you may also require a suitable search base to avoid lookup failure due to hitting size limits OR finding multiple entries with the same email address

## 6.4.2 LASER Attributes

When using the default unchanged, you need to ensure that the `mailLocalAddress` attribute contains the email address of the user. For example:

```
objectClass= inetLocalMailRecipient
mailLocalAddress= zoe.brown@headquarters.net
```

You should now find that the address `zoe.brown@headquarters.net` is routed to the local MTA by LMTP (use **ckadr** to check this):

```
/opt/isode/sbin/ckadr zoe.brown@headquarters.net

zoe.brown@headquarters.net -> (rfc822) \
    zoe.brown@headquarters.net

Delivered to headquarters.net by lmtp (weight: 0)
```

## 6.4.3 Using non-default LASER Attributes For Routing

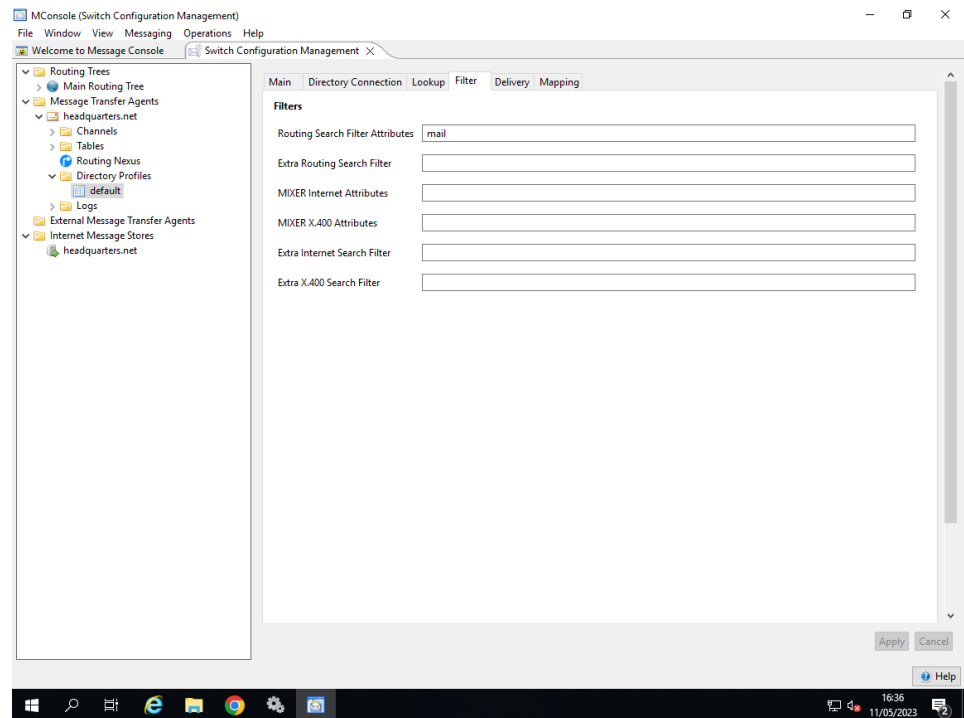
As described in [Section 6.4.2, “LASER Attributes”](#) when using the default Directory Profile unchanged, you need to set the `mailLocalAddress` attribute in the Directory Entry for the user. However you may have your DIT set up so that the email address is held in a different attribute. If this is the case, the following section describes how to configure M-Switch to use this attribute.

This example shows how to configure a Directory Profile to use the `mail` attribute for the email address being looked up.

In this example, the Directory Entry has the email address in the `mail` attribute:

```
objectClass= inetOrgPerson
mail= zoe.brown@headquarters.net
```

**Figure 6.4. Directory Profile in MConsole - Filter Tab**



## 6.4.4 Using non-default LASER Attributes For MIXER mapping

### 6.4.4.1 Internet to X.400

To perform a per user mapping of Internet to X.400 addresses, by default the DIT is searched for a mail attribute, and a corresponding `mhsORAddresses` attribute in the entry which contains the MIXER mapping.

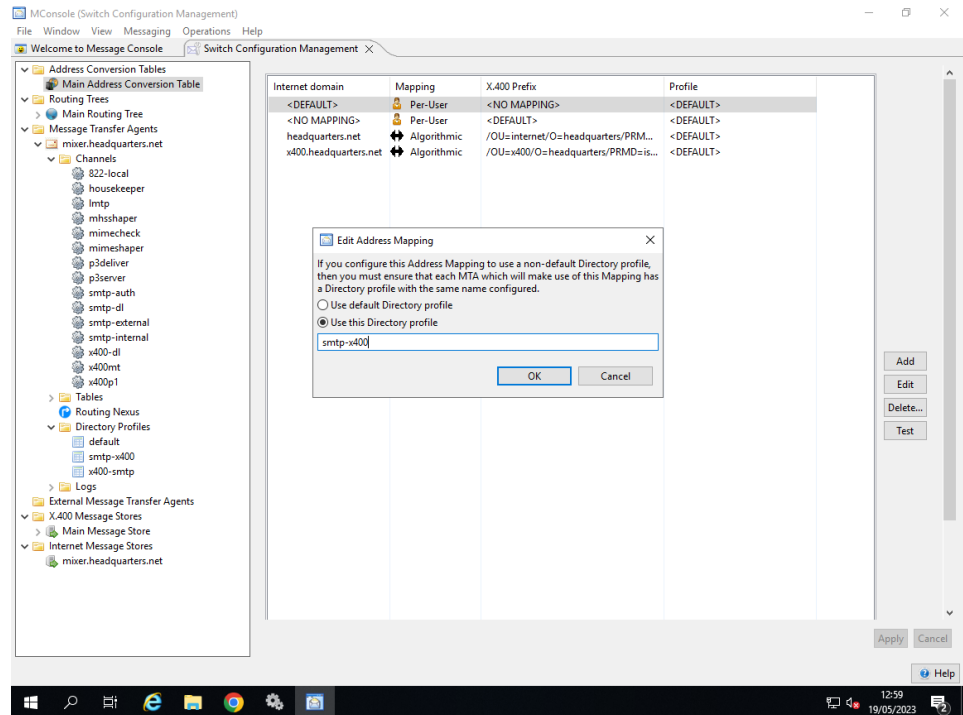
However you may have have your DIT set up so that the X.400 address is held in a different attribute. If this is the case, the following section describes how to configure M-Switch to use this attribute.

This example shows how to configure a Directory Profile to use the `mhsX400Addresses` attribute for the X.400 O/R Address being looked up.

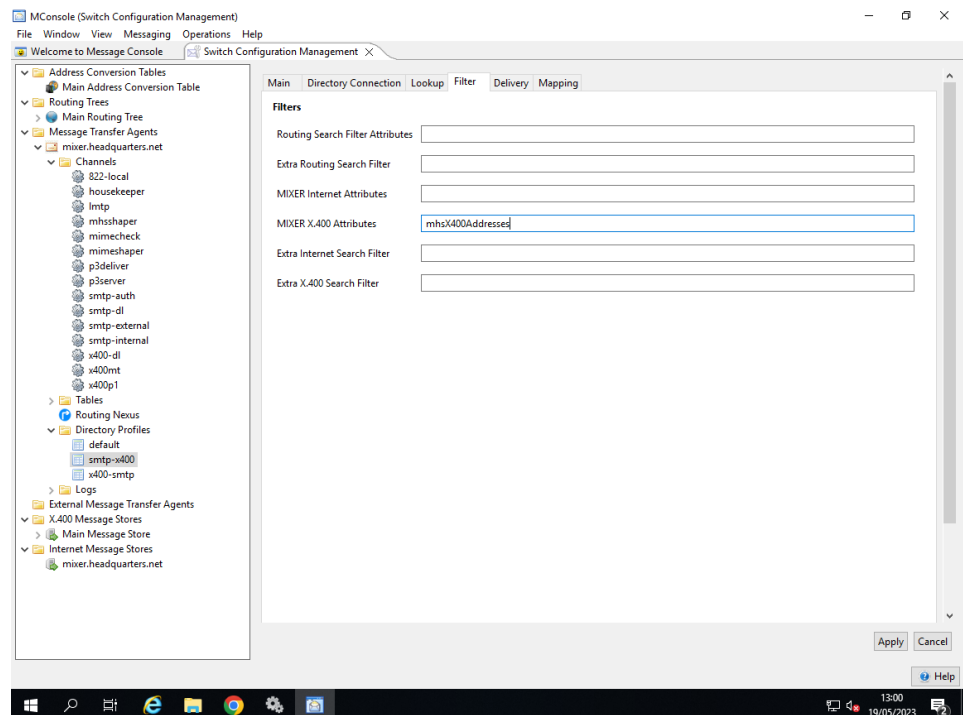
In this example, the Directory Entry has the Internet address in the `mail` attribute and the O/R Address in the `mhsX400Addresses/` attribute.

```
objectClass= inetOrgPerson
objectClass= ftbePeer
mail= zoe.brown@headquarters.net
mhsX400Addresses= /G=p7-x400/S=user3/O=sanders/ADMD= /C=gb/
```

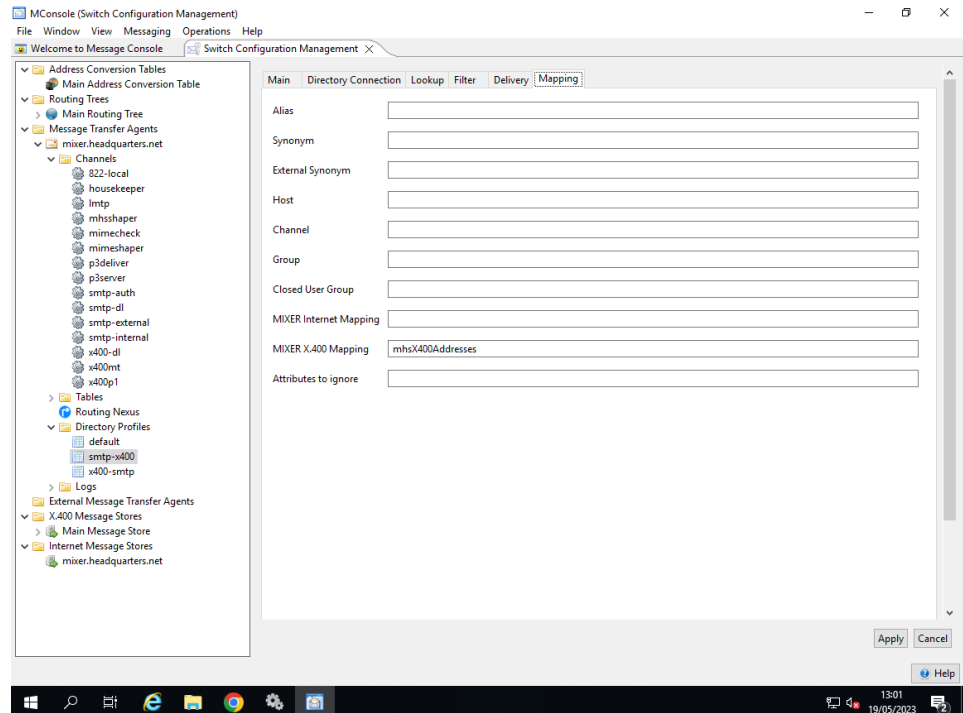
Furthermore in this example, a different DSA is used to hold the MIXER mapping, which requires a different Directory Profile to be used in the Address Conversion Table by setting the `laserMapping` attribute by clicking on the entry in the Address Conversion Table and pressing `Edit` which results in the following dialog:

**Figure 6.5. Directory Profile in MConsole - Address Conversion Table**

Next you need to configure the MIXER X.400 Attributes value in the Directory Profile Filter Tab:

**Figure 6.6. Directory Profile for MIXER Mapping (Filter Tab)**

Finally you need to configure the MIXER X.400 Mapping value in the Directory Profile Mapping Tab:

**Figure 6.7. Directory Profile for MIXER Mapping (Mapping Tab)**

#### 6.4.4.2 X.400 to Internet

To perform a per user mapping of an X.400 O/R Address to Internet address by default the DIT is searched for a `mail` attribute, and a corresponding `mhsORAddresses` attribute in the entry which contains the MIXER mapping.

However you may have your DIT set up so that the Internet address is held in a different attribute. If this is the case, the following section describes how to configure M-Switch to use this attribute.

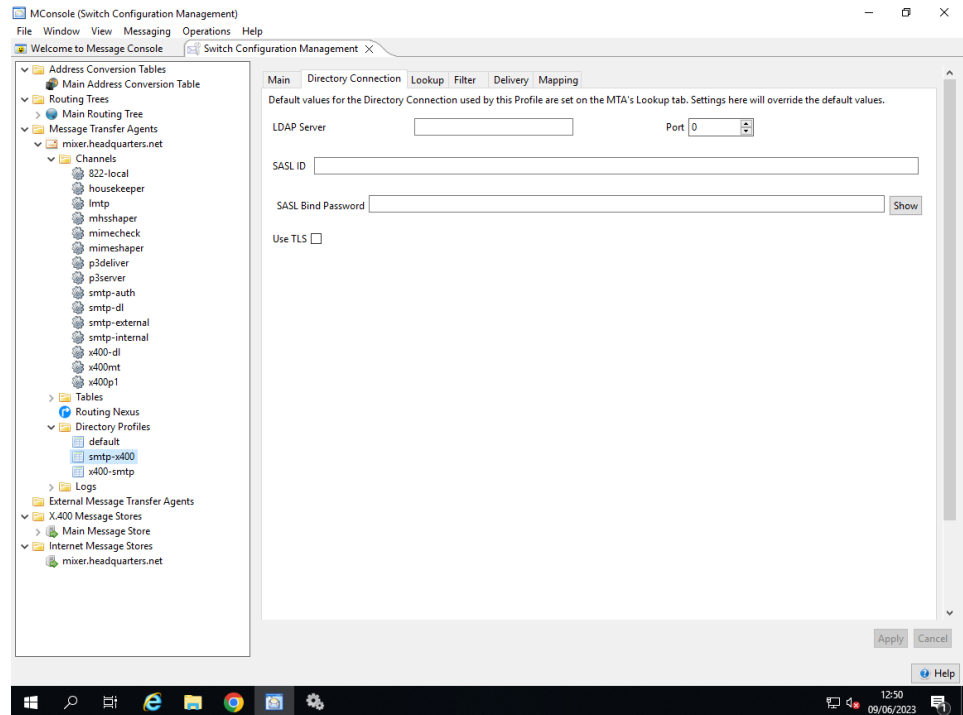
This example shows how to configure a Directory Profile to use the `mailRoutingAddress` attribute for the Internet Address being looked up.

In this example, the Directory Entry has the Internet address in the `mail` attribute and the O/R Address in the `mhsORAddresses` attribute.

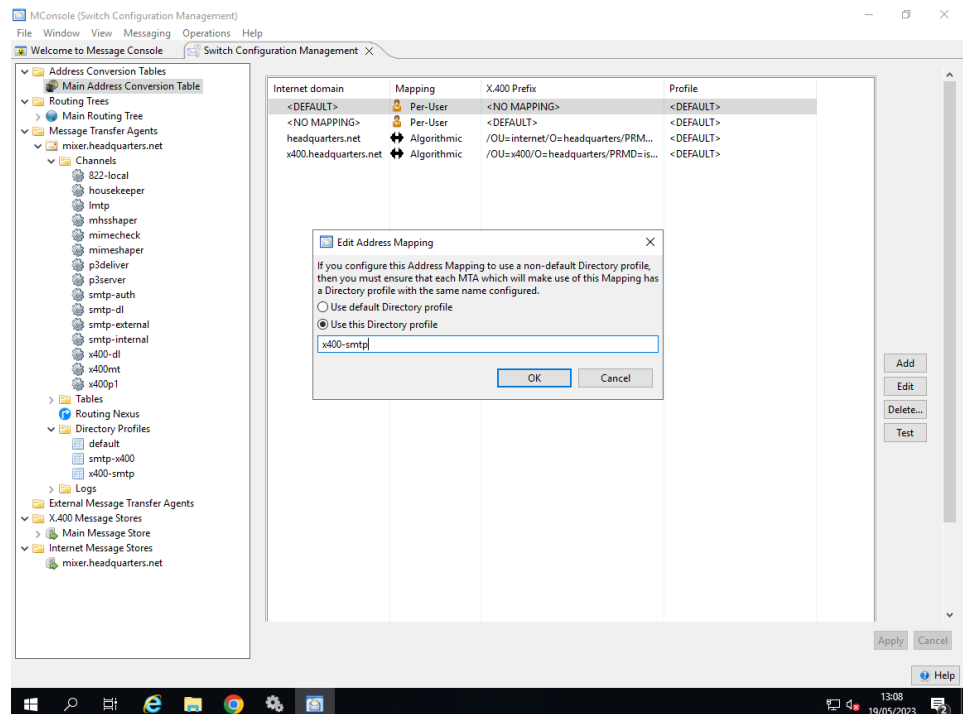
```
objectClass= mhsUser
objectClass= inetLocalMailRecipient
mailRoutingAddress= zoe.brown@headquarters.net
mhsX400Addresses= /G=p7-x400/S=user3/O=sanders/ADMD= /C=gb/
```

Furthermore this example shows how a different DSA is used to hold the MIXER mapping. This requires a new Directory Profile to be setup which contains the hostname, port etc of the DSA. This Directory Profile is used in the Address Conversion Table by setting the `laserMapping` attribute in the Address Conversion Table entry. Clicking on the entry in the Address Conversion Table and pressing `Edit` which results in the following dialog which allows the name of the Directory Profile to be entered.:

This shows the Directory Profile Connection Tab which results in a different Directory being used for the MIXER mapping lookup:

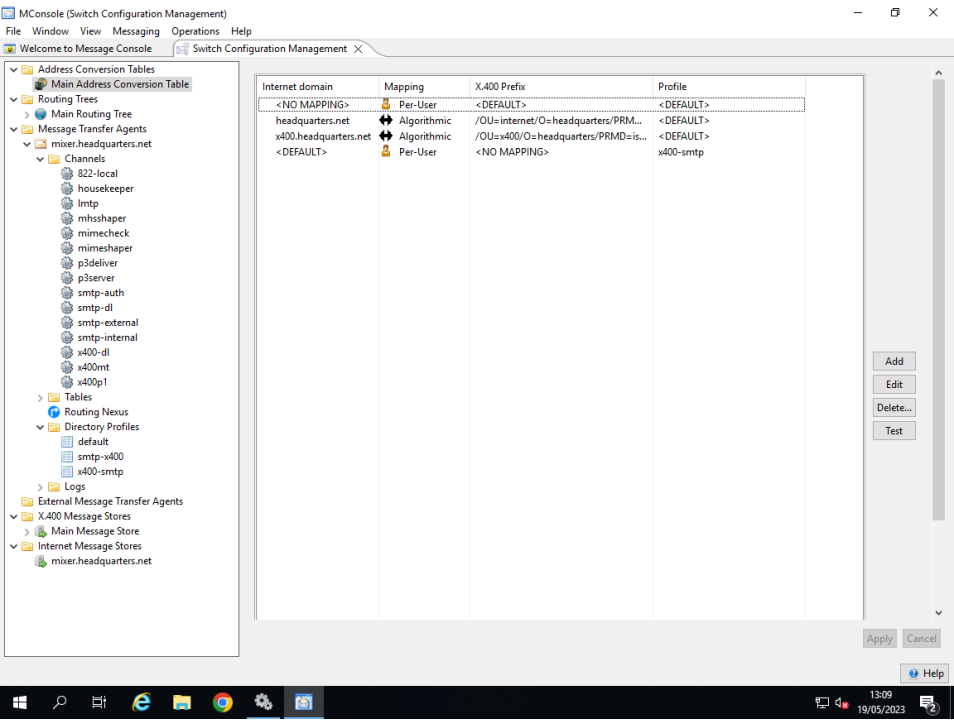
**Figure 6.8. Directory Profile in MConsole - Directory Connection Tab**

To configure a different Directory Profile to be used in the Address Conversion Table set the laserMapping attribute by clicking on the entry in the Address Conversion Table and pressing **Edit** which results in the following dialog:

**Figure 6.9. Directory Profile in MConsole - Address Conversion Table**

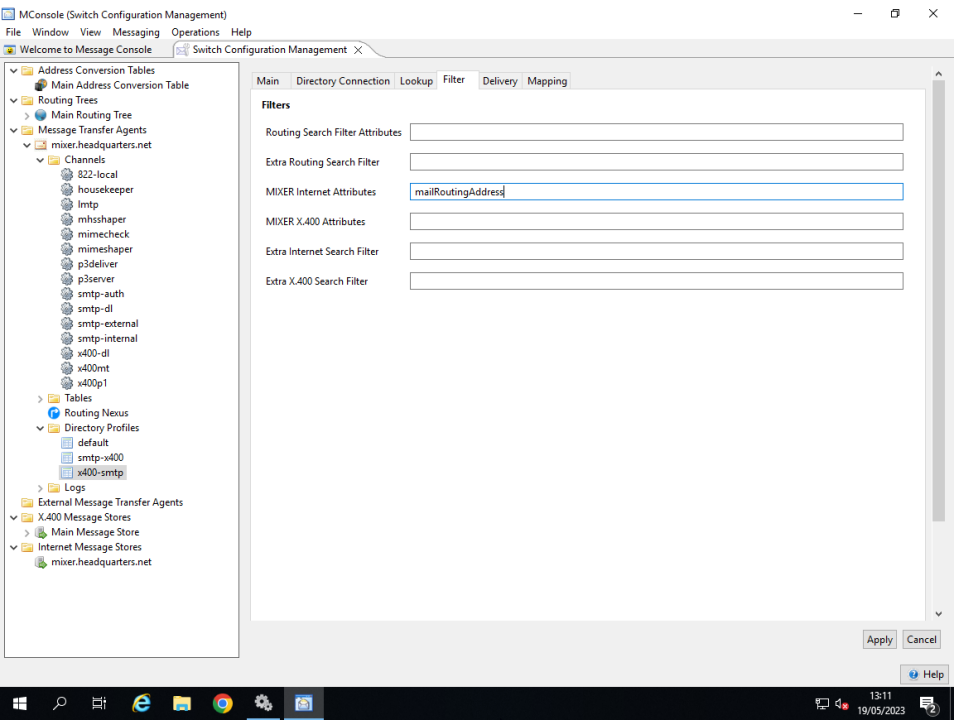
Clicking OK on the Edit results in the Address Conversion Tree looking as follows:

Figure 6.10. Directory Profile in MConsole - Address Conversion Table



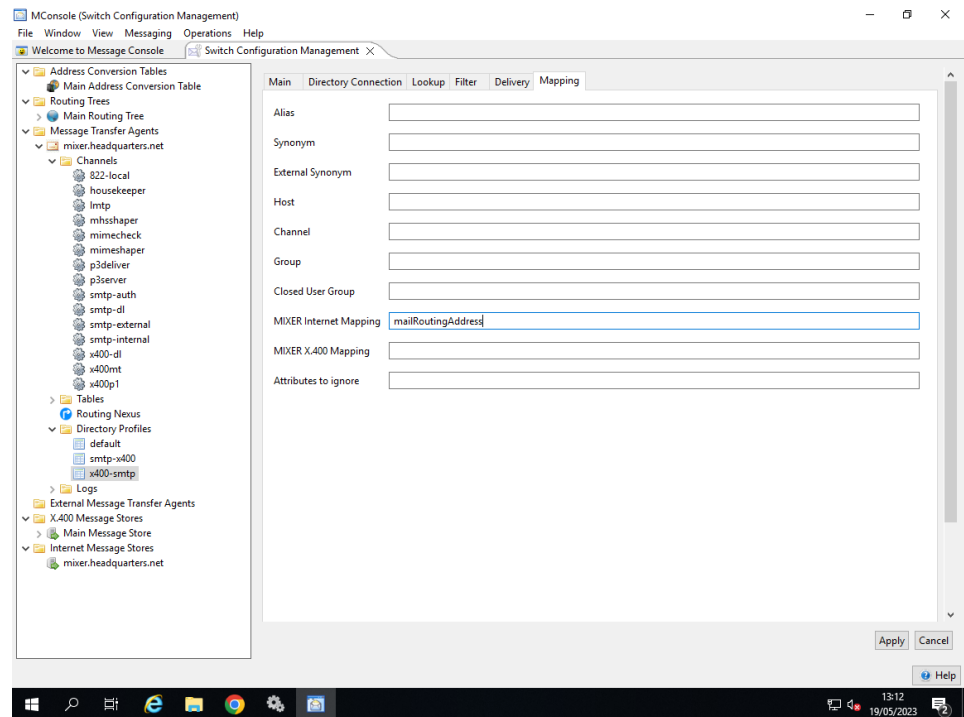
Next you need to configure the MIXER Internet email address Attributes value in the Directory Profile Filter Tab:

Figure 6.11. Directory Profile for MIXER Mapping (Filter Tab)



Finally you need to configure the MIXER Internet email address Attributes value in the Directory Profile Mapping Tab:



**Figure 6.12. Directory Profile for MIXER Mapping (Mapping Tab)**

# Chapter 7 Managing Internet Messaging Users

This chapter contains references to other manuals which describe how to configure M-Switch so that internet email addresses are routed and delivered by M-Switch into M-Box using MConsole.

---

**Note:** In this chapter, all references of an Internet Message Store refer to a POP/IMAP Message Store

---

The configuration of Internet Mailbox Users is held in the Directory, and is managed using Cobalt.

---

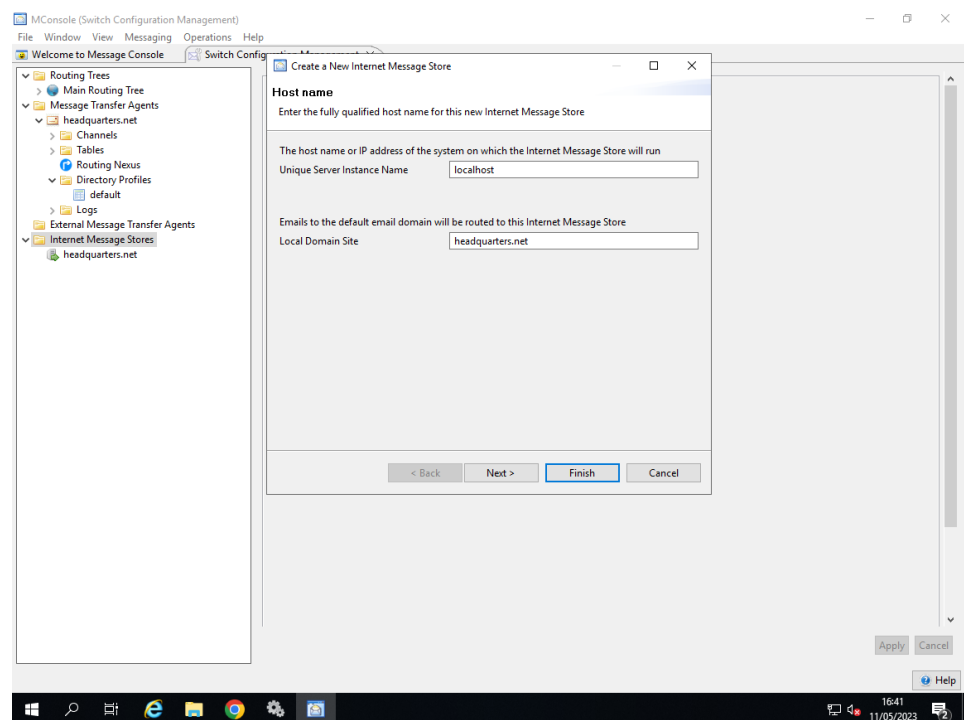
## 7.1 Creating an Internet Message Store

In order to deliver internet messages, you need to have configured an Internet Message Store. You may have created one during your initial configuration (as in [Figure 5.1](#), “[Example Internet configuration](#)”) but, if not, you can follow the steps below to create one from the **Switch Configuration Management** view.

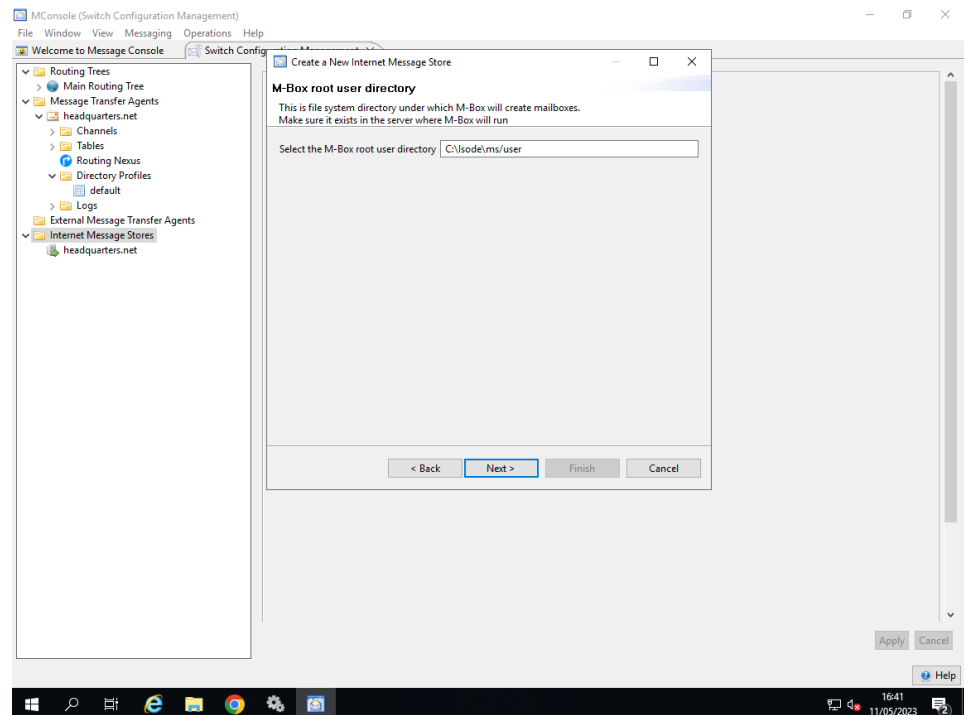
### 7.1.1 Adding an Internet Message Store

Right click on **Internet Message Stores** in MConsole and select **New Internet Message Store**.

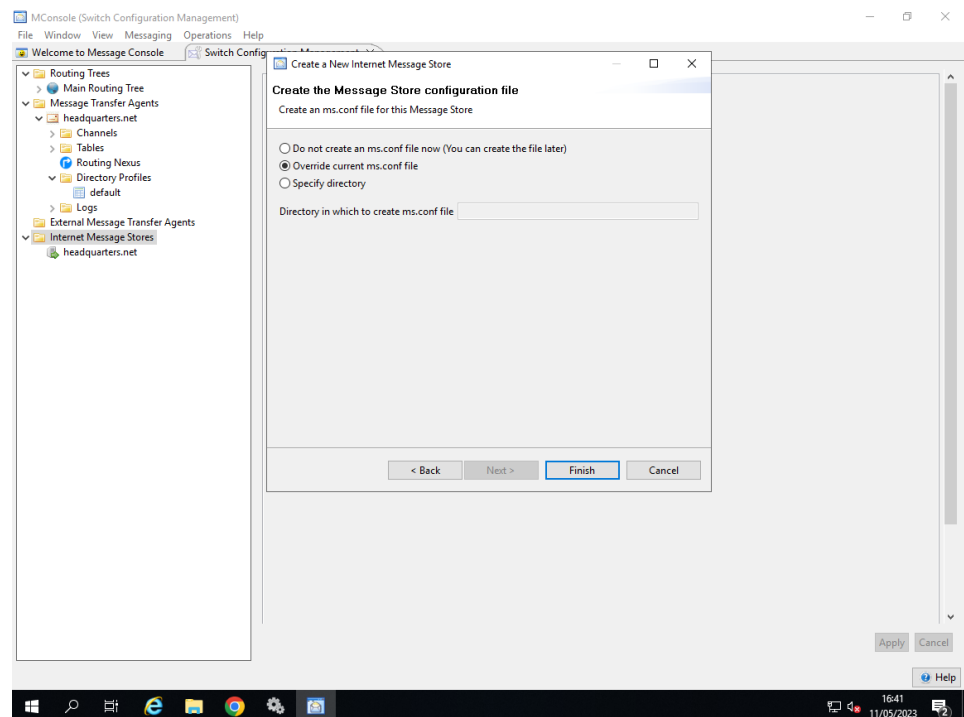
**Figure 7.1. Add new Internet Message Store - Host name**



Enter the host name or IP address of the system on which the Internet Message Store will run, and click on **Next**.

**Figure 7.2. Add new Internet Message Store - Mailbox directory**

Enter the value for the filestore directory to use to store the mailboxes. Click on **Next**.

**Figure 7.3. Add new Internet Message Store - Create ms.conf**

Choose whether you want to create the M-Box configuration file *ms.conf* (normally you would choose **Override current ms.conf file** if M-Box is running on the same system as M-Console), and click on **Finish**.

# Chapter 8 Connecting to SMTP MTAs

This chapter guides you through the way in which SMTP connections to other Internet MTAs are configured.

---

## 8.1 Internet Message Transfer Protocols

Internet MTAs send messages to each other primarily by SMTP. This chapter describes how to configure the many options available in M-Switch for SMTP. Internet MTAs can also use Multicast Email (MULE) over Allied Communications Publication (ACP)142 (RFC8494). See [Chapter 17, \*Connecting to other Military MTAs\*](#) for a description of ACP 142.

SMTP is used server to server to relay messages. In this case one of the servers acts an SMTP client to send messages, and the other acts as a server to receive messages.

In addition SMTP is also used by User Agents (email clients) to submit messages into an MTA.

Both of these two cases are considered in this Chapter.

---

## 8.2 SMTP overview

Isode M-Switch provides full support for Simple Mail Transfer Protocol (SMTP) as specified in RFC 5321, conforming to the Internet host requirements for messaging (RFC 1123), including message submission (RFC 4409). SMTP support includes a number of auxiliary Internet standards related to general capabilities and Lemonade (RFC 4550) support. Supported SMTP extensions are listed in sections describing the SMTP server and the SMTP client respectively.

For connections to Internet MTAs by other protocols, please see [Section 8.4, “Using TLS with SMTP”](#).

Support for the SMTP protocol is provided by two types of processes: one for inbound SMTP (handling messages coming into the MTA) and one for outbound SMTP (transferring messages out of the MTA). Inbound SMTP is handled by a single multi-threaded SMTP server process (channel) named `isode.pp.smtp`, so that new inbound SMTP connections do not require a process to be started.

Message transfer using SMTP protocol out of M-Switch is provided by the `slmtp` process. The `slmtp` process is started by the `Qmgr` process ([Section 34.3, “Queue Manager”](#)), which may run many instances of the `slmtp` program.

Both inbound and outbound SMTP can be represented using one or more channels each sharing the same channel key. For example, a typical Internet configuration (similar to the one created by the `MConsole` utility) will include three SMTP inbound channels: `smtp-external`, `smtp-auth` and `smtp-internal`. In this type of configuration, all three channels will use `key=smtp` but each of them is configured to handle messages differently:

- `smtp-internal` is used for local inbound SMTP transfers, where the source MTA is authorised by matching an **Inchan** rule in the Authorization configuration for the MTA. This is typically a list of local, trusted MTAs See [Section 38.1, “Authorization”](#) for further details.
- The `smtp-auth` channel is used for authenticated message submission, and usually listens on port 587.
- The `smtp-external` channel is used for all other SMTP transfers.

## 8.2.1 Message submission

Message submission uses the SMTP protocol. The Internet Standard for Message Submission (RFC 4409) gives a special port for this (587), which is distinct from the SMTP port used for message transfer (25). Message submission is usually authenticated, whereas this is unusual for message transfer. It is also typical to allow messages arriving on the submission port to be normalized (e.g. for internal addresses to be replaced with externally-visible versions). The SMTP Server can be configured to listen on multiple ports simultaneously and complies with RFC 4409.

## 8.2.2 Binding inbound SMTP channels to specific port numbers

You can configure the port in one of two ways:

- In the Channel entry itself in MConsole. Select the channel and then the Program Tab. The "In" tab now allows you to select the address and port on which this channel is to listen.
- Using the arguments to the invocation of the channel (either the Unix or Windows Service).

The former is relatively simple. The latter is configured as follows.

Each port on which SMTP listens for inbound connections is described by one `listener` parameter to the `isode.pp.smtp` process. Each 'listener' has the form:

```
<key>[ / [ <addr> ] [ / <port> ]
```

where:

`<key>`

is a channel key, matching the channel name or the channel `key=<key>` for a range of channels. The SMTP client name is also used for this, in conjunction with the channels' mtatables.

`<addr>`

is the listen address, in either IPv4 or IPv6 format. If omitted, `smtpsrvr` will listen on 'all interfaces'.

`<port>`

can be a service name or a number. If omitted, the value of the `-p` option is used, or `smtp` if a `-p` option is not specified.

If no listener parameters are specified, the default parameter used is as though the listener was invoked as:

```
isode.pp.smtp smtp//25
```

In the simplest case, the same inbound SMTP channel (or group of channels sharing the same key parameter) can be bound to multiple SMTP ports using multiple listener parameters to the `smtpsrvr` process, each differing only in port number, for example:

```
isode.pp.smtp smtp//25 smtp//587
```

will listen on standard SMTP port 25 and standard submission port 587.

---

## 8.3 Data confidentiality and authentication

Isode M-Switch uses Transport Layer Security (TLS) for data confidentiality and Simple Authentication and Security Layer (SASL, RFC 4422) for authentication. SASL and TLS are particularly important for controlling message submission.

- TLS is supported for both inbound and outbound SMTP, as well as for outbound LMTP. TLS on outbound SMTP can be used to protect server to server communication.

Options controlling TLS are listed in [Section 8.4, “Using TLS with SMTP”](#).

- SASL is only supported for inbound SMTP.

Options controlling SASL are listed in [Section 8.6, “SMTP server”](#).

---

## 8.4 Using TLS with SMTP

TLS can be used with the SMTP server for inbound connections to M-Switch and with the SMTP client for outbound connections from M-Switch, using the **STARTTLS** SMTP extension as defined in RFC 3207. Unlike other protocols, there is no standard (or de facto standard) for use of TLS on the initial TCP connection.

TLS relies on Public Key Infrastructure (PKI) to provide security features of authentication and confidentiality. This in turn relies on Digital Certificates. M-Switch implements TLS using the X.509 standard formats. These Digital Identities and other Security Objects can be held in one two ways in the M-Switch configuration: Security Database or Legacy Variables. The latter is obsolescent and will be withdrawn in a future release.

### 8.4.1 TLS in the SMTP client

Use of TLS by the SMTP client is configured on a per-channel basis, in the **MTA → Channel → Program → TLS** tab (setting the channel variable `tls`) with one of the following values:

`no`

TLS is disabled

`server`

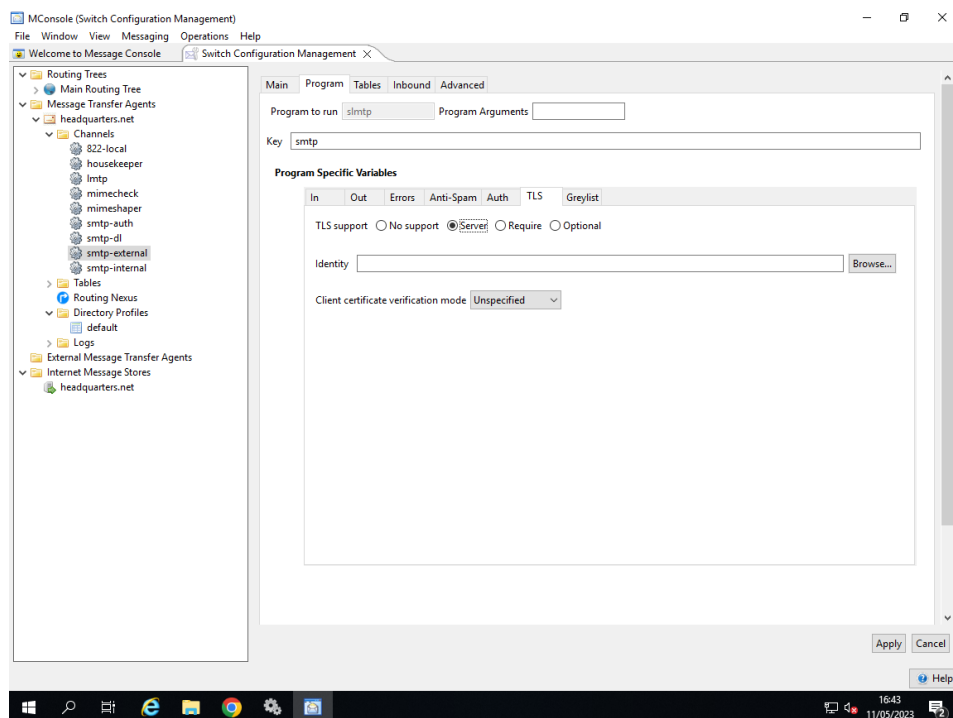
this value specifies that TLS is only enabled on the inbound SMTP channel. This option disables TLS on outbound SMTP channel

`optional`

attempt to establish TLS using **STARTTLS**, but send messages unprotected if this fails

`require`

the **STARTTLS** command must succeed to be able to send messages to a peer MTA over this channel

**Figure 8.1. Configuring TLS for an SMTP channel**


---

**Note:** Inbound SMTP channels with any value other than `no` enables support for **STARTTLS**. The security environment of the client is configured either via the Security Database (see [Section 14.1.6, “Security settings”](#)) or via legacy variables, which are only available if the Security Database is not in use. See below for how to configure this for the server.

---

## 8.4.2 TLS Server

Use of TLS by the SMTP server is configured on a per-channel basis, in the **MTA** → **Program** → **TLS** tab (setting the channel variable `tls`) with any of the TLS values other than `no` as described in [Section 8.4.1, “TLS in the SMTP client”](#).

If the Security Database is in use, legacy configuration of the Digital Identity to be used by the `smtpsrvr` is disabled. If the Security Database is not in use, the directory path containing the Digital Identity can be configured (internally setting the channel variable `tls_path`). See [Section 14.1.7, “Legacy TLS Configuration”](#) for more information.

## 8.4.3 Digital Identities and Trusted Certification Authorities

For information on how to obtain Digital Identities and trusted certification authorities, refer to section [Section 14.1.6.1.5, “Obtaining digital identities and CA certificates”](#)

---

## 8.5 Greylisting

Both the SMTP Client and Server programs contain support for the Greylisting antispam technique. This technique and relevant configuration options are discussed in detail in [Section 40.4.1, “Greylisting”](#).

## 8.6 SMTP server

The SMTP server supports the following SMTP extensions:

Extension	Description
HELP	Help information [RFC 2821]
EXPN	Distribution list expansion (only for X.500 Directory based lists) [RFC 2821]
VERFY	Address verification [RFC 2821]
SIZE	Message size declaration [RFC 1870]
8BITMIME	8bit-MIME transport [RFC 1652]
BINARYMIME	Binary MIME transport [RFC 3030]
PIPELINING	Client side pipelining [RFC 2920]
DSN	Delivery Status Notifications [RFC 3461]
ENHANCEDSTATUSCODES	Enhanced Error Codes [RFC 2034]
AUTH	Authentication [RFC 2554]
BURL	Lemonade extension for message assembly during submission [RFC 4468]
CHUNKING	Efficient sending of large MIME messages
STARTTLS	Secure SMTP over TLS [RFC 3207]
MT-PRIORITY	Priority transfers [RFC 6710]
DELIVERBY	Deliver by a prescribed time [RFC 2852]
FUTURERELEASE	Future message release [RFC 4865]

The server program for inbound messages, `smtpsrvr`, makes use of additional tailoring variables in the channel specification. The per-channel variables (referred to as ‘options’) that are currently recognized are listed below.

For values which configure TLS see [Section 8.4, “Using TLS with SMTP”](#). For values which configure SASL see [Section 8.7, “SMTP authentication”](#).

Most of the following options are configured on the **Program** page of the SMTP channel. The sub-pages on the **Program** page are: **In**, **Out**, **Errors**, **Anti-Spam**, **Auth**, **TLS**, **Greylist**. This list below indicates these using [I], [O], [E], [AS], [A], [T], [G] respectively, (These are held internally (and in the *mtatailor* file) as channel specific or PP variables. Values which have to be configured as channel specific variables in the **Advanced** page are labelled [C]. Values which have to be configured as PP variables (MTA level variables) in the **Advanced** page are labelled [P].)

The following options control which connections/senders are accepted:

`block` [I]

If this option is present, it will cause the channel to block connections by always returning the 521 error code in SMTP greeting.

`noname` [I]

If this key has the value true, any connection is allowed. If it is not set to true, connections are only supported from hosts that can be reverse translated (the IP address, for example, can be converted back to a host name).



If this variable is present but has no value, it defaults to false. Some people consider that setting `noname=false` is a violation of RFC 1123 (Internet host requirements).

#### `rbl` [AS]

If this is set, it enables the Realtime Blackhole List (RBL) feature (see <http://mail-abuse.org/rbl/>). A specific domain can be specified, which is used as a suffix to the calling IP address, for use with local implementation. An alternative target address can also be specified, as some RBL domains use non-standard addresses. Multiple RBL domains can be specified in a list separated using semi-colons: they are used by the SMTP inbound channel in the order in which they are specified. The syntax of the switch is:

```
rbl=<rbl_domain>[ "+"<target_address>][ ";"<rbl_domain>...]
```

Specifying just `rbl` is equivalent to using the normal RBL domain and default target address:

```
rbl=blackholes.mail-abuse.org+127.0.0.2
```

#### `rblheader` [AS]

If this is set, messages from remote MTAs which are found on the configured Realtime Blackhole List are not rejected, but instead have their headers annotated with "X-RBL-FOUND: <name> (<addr>)", where <name> and <addr> are the name and IP address of the sending system.

#### `reject` [AS]

This sets the error code used when connections are rejected (for example, for the RBL or if `noname` is not `true` and there is no domain associated with the calling IP address). It must be a three-digit SMTP status code: for example, 421 for a temporary reject, or 550 for a permanent reject.

#### `spf` [AS]

Presence of this option enables SPF [RFC 4408] lookups on the domain specified by the client in the HELO/EHLO command and on the domain part of the MAIL FROM address. The value of this option specifies a comma separated list of conditions that should cause rejection of the SMTP connection (for HELO/EHLO check failure) or failure of an SMTP transaction (for MAIL FROM check failure). Currently recognized conditions are as follows:

- `fail` – ‘fail’ result from the SPF check
- `soft` – ‘soft Fail’ result from the SPF check, where the domain believes the host is not authorized to send email but is not willing to make that strong a statement. See section 2.5.5 of RFC 4408 for more details.
- `temp` – ‘TempError’ result from the SPF check, where the SPF client encountered a transient error while performing the check. See section 2.5.6 of RFC 4408 for more details.
- `perm` – ‘permanent error’ result from the SPF check, where the domain's published records could not be correctly interpreted. See section 2.5.7 of RFC 4408 for more details.

Unrecognized conditions are ignored. If the result of the SPF check does not match any of the specified conditions, then the X-SPF-HELO-Result (for HELO/EHLO check) and/or X-SPF-Result (for MAIL FROM check) header field is added to the message.

The following options control various aspects of SMTP transactions:

#### `maxrecips` [E]

If present, sets a maximum number of recipient addresses which will be accepted for an individual message. Addresses in excess of this maximum figure will be rejected with a temporary error. The presence of large numbers of recipient addresses in a

message arriving from an external MTA is often an indicator of junk mail. The default value when this option is not present is `no limit`.

`reciplimit [E]`

This option behaves similarly to `maxrecips` except that addresses in excess of this maximum figure will be rejected with a permanent error.

One way in which the originators of SPAM messages obtain email addresses is via Address Harvest attacks, where an SMTP client program is used to test a large range of possible recipient email addresses (for example, `a@headquarters.net`, `b@headquarters.net`, `c@headquarters.net`) in the hope of finding some valid addresses. The SMTP inbound channel supports two configuration switches which can be used to reduce the vulnerability of the system to such attacks:

`maxerr [E]`

If present, sets a maximum number of errors which will be allowed on address-related commands (`MAIL`, `RCPT`, `VRFY`) before the command will no longer be accepted. When this limit has been exceeded, all further commands will be accepted, but both valid and invalid addresses will be faulted. The default value when this option is not present is 'no limit'.

`errdelay [E]`

If present, sets a delay period (in seconds) which will be imposed after each address-related error. This is designed to slow down attempts to harvest addresses (in a similar way to delays on login failure). The default value when this option is not present is 'no delay'.

The following options control various SMTP extensions:

`amms=integer [I]`

This parameter specifies the maximum acceptable message size (in bytes) that the SMTP server is going to allow. It is used in the SMTP dialog with hosts supporting SMTP extensions. The default value is 0, which means 'no limit'.

`verfy[={yes|no|auth}] [AU]`

This option specifies whether the `VRFY` command is enabled on the server. The value of `auth` means that `VRFY` is advertised but is only allowed after successful authentication. If the client is not authenticated, the server responds "530 5.5.1" ("Must authenticate first"). If this option is not present or is present but no value is specified, it defaults to `no` (the `VRFY` command is not supported). See also the `soft_noauth` option.

`expn[={yes|no|auth}] [AU]`

This option specifies whether the `EXPN` command is enabled on the server. The value `auth` means that `EXPN` is advertised but is only allowed after successful authentication. If the client is not authenticated the server will respond "530 5.5.1" ("Must authenticate first"). If this option is not present or is present but no value is specified, it defaults to `no` (the `EXPN` command is not supported). See also the `soft_noauth` option.

`absent_ret_roc [I]`

This option specifies the default value of `ROC` (return of contents), if it is not specified in the `ESMTP RET` parameter to the `MAIL FROM` command. If this option has the value "hdr" then only message headers are returned. If any other value is specified, or if this option is not present, the whole of the message content is returned.

`original text [C]`

If set to `n`, `ROC` (return of contents) is not set when gatewaying the message to an X.400 network when `ESMTP RET` is absent.

`deliveryby=integer [I]`

If set to `n`, then this is the minimum time (in seconds) that a message should request delivery by. If set to 0 or negative, the extension is disabled.

`futurerelease=integer` [I]

If set to *n*, then this is the minimum time (in seconds) that a message should request to be held for delivery. If set to 0 or negative, the extension is disabled.

`priority_profile={profile}` [I]

This indicates a priority profile that should be offered with the MT-PRIORITY extension. Current support values include MIXER, STANAG4406, NSEP and DEFAULT.

The following options control handling of binary messages:

`binary` [I]

Allow receiving of binary messages not labeled with `BODY=BINARYMIME MAIL FROM` parameter.

---

**Note:** The resulting message will probably be changed (converted) when it is relayed to another MTA. Such a message may also cause problems in processing.

---

`line=<integer>` [C]

Set the line length limit used when a `BODY=BINARYMIME` message is converted to 7bit. By default there is no line length limit.

## 8.6.1 8bit and binary data

The MIME specifications [RFC 2045] make a clear distinction between 8bit data and binary data. The former can include data with the 8th bit set (byte values 128 to 255). However, it cannot include the NUL character, nor Carriage Return or Linefeed, except as a CR LF end of line pair. Also, 8bit data is still subject to the SMTP line length restriction of no more than 998 bytes between CR LF pairs. This makes 8bit data unsuitable for the transfer of arbitrary binary data. The SMTP server prevents the transfer in of binary data, unless the 'binary' option is set or `BODY=BINARYMIME MAIL FROM` parameter is specified (in either case the line length restriction is not applied). As a result of the way messages are handled within the Message Switch, data that violates the 8bit constraints and not labeled with `BODY=BINARYMIME` may be changed and also cause problems with operation of the Message Switch.

All bodyparts of messages properly labeled with `BODY=BINARYMIME` are automatically converted to 7bit using base-64 or quoted-printable Content-Transfer-Encoding.

8bit data is only permitted within MIME body parts. It is not permitted to have non-ASCII characters within message or body headers. This is because there is no mechanism for assigning a character set to such characters and the same byte value corresponds to different characters in different character sets. MIME provides a mechanism for encoding non-ASCII characters within heading fields [RFC 2047]. The presence of 8bit characters in message headers can cause problems in the operation of the Message Switch.

---

## 8.7 SMTP authentication

The SMTP inbound channel supports SMTP extension for authentication [RFC 2554]. This allows connections to the SMTP server to be authenticated. RFC 2554 uses the Simple Authentication and Security Layer (SASL) framework. The SASL framework provides a method for adding authentication support with an optional security layer to connection-based protocols. It also describes a structure for authentication mechanisms. The result is an abstraction layer between protocols and authentication mechanisms such that any SASL-compatible authentication mechanism can be used with any SASL-compatible

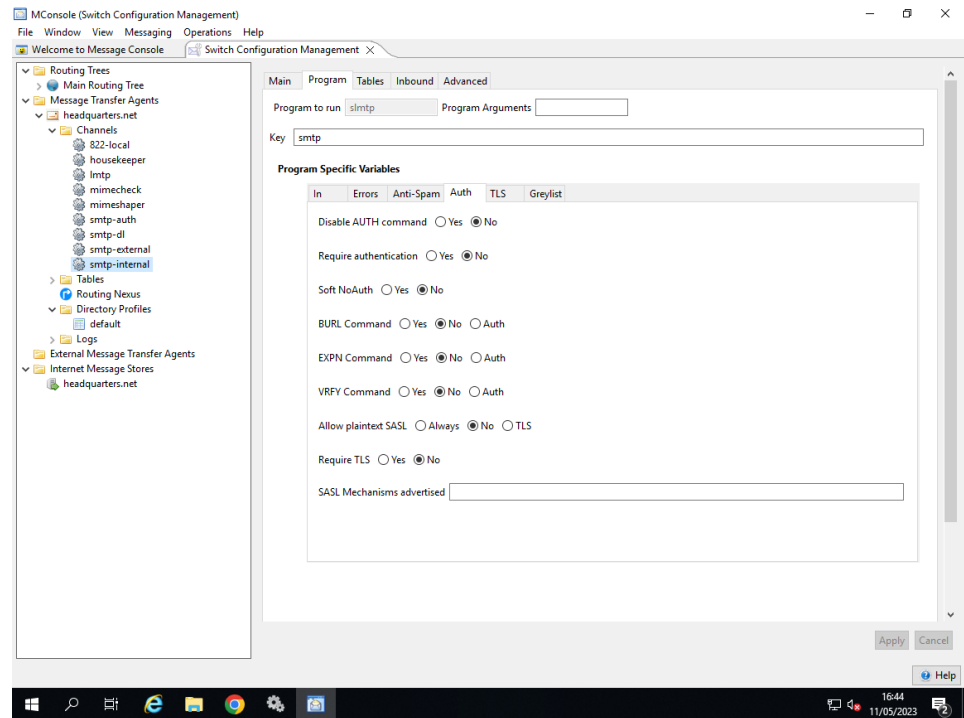
protocol. See *RFC 4422: Simple Authentication and Security Layer (SASL)*, Alexey Melnikov (Editor) and Kurt Zeilenga (Editor), June 2006 for more information.

The SMTP server supports various SASL authentication mechanisms as detailed below.

## 8.7.1 How SMTP authentication works

Authentication is configured on the **Program** page of the SMTP channel under the **Auth** tab.

**Figure 8.2. SMTP Channel Properties (Auth tab)**



## 8.7.2 Options affecting SMTP authentication

The SMTP server program makes use of channel-specific variables to control SMTP authentication. This section lists the per-channel variables (also referred to as ‘options’) that are currently recognized.

```
require_auth[={true|yes|false|no}]
```

This corresponds to the "Require authentication" radio buttons.

This option specifies whether authentication is *required* before starting a mail transaction. If its value is `true`, and the client has not authenticated, the server will reply "530 5.5.1 (Must authenticate first)".

If the option is not specified, it defaults to `false`.

If the option is specified but no value is given, it defaults to `true`. See also `soft_noauth` option.

```
soft_noauth[={true|yes|false|no}]
```

This corresponds to the "Soft NoAuth" radio buttons.

If this option is `true`, all SMTP authentication 5XX error codes (such as “authentication required”) will be reported as 4XX error codes instead.

```
allowplaintext[={always|tls|false|no}]
```

This corresponds to the "Allow plaintext SASL" radio buttons.

This option specifies whether plaintext SASL mechanisms (PLAIN, LOGIN) are allowed.

- If its value is `false` or `no`, those mechanisms will not be advertised in the EHLO response. If the option is not present or is present but no value is specified, it defaults to `false`.
- If its value is `tls`, then these mechanisms are only advertised after a successful TLS negotiation.
- If its value is `always`, then these mechanisms are always advertised.

`sasl_mech_list=<space separated list of SASL mechanisms>`

---

**Note:** This corresponds to the "SASL Mechanisms advertised" field.

---

This option allows you to limit which mechanisms are advertised by the channel in its EHLO response. The intersection of the set of available mechanisms with this list is returned in the EHLO response: for example, if "PLAIN DIGEST-MD5 GSSAPI" are available and the value of this option is "SRP GSSAPI DIGEST-MD5", the EHLO response will list at most DIGEST-MD5 and GSSAPI. Other options, such as `allowplaintext`, may reduce the response still further.

### 8.7.2.1 Options specific to LDAPDB plugin

LDAPDB is an auxiliary property (auxprop) plugin used for password verification by SASL plugins implementing the SASL authentication method. If you want to store usernames and passwords for SMTP AUTH in LDAP, you need to use LDAPDB. This plugin requires some configuration options, which are listed below. The most commonly used of these options can be configured using from the "Lookup" tab for the MTA in MConsole (as noted below): other options must be configured as "PP Internal Variables" using the "Advanced" tab if required.

`sasl_config_entry=<Distinguished Name of DSA's configuration entry>`

This corresponds to the "SASL Configuration Entry" field.

This is the Distinguished Name of the DSA's configuration entry. This normally has the value `<cn=core,cn=config>` .

`ldapdb_uri[=<ldapurl>] [P]`

This option specifies the LDAP server hostname and port number to connect to. If this option is not specified, the default value is going to be constructed from `ldap_host` and `ldap_port` described below.

- If `ldap_host` is not specified, then the value `localhost` is used when constructing `ldapdb_uri`
- If `ldap_port` is not specified, then the default LDAP port 389 is assumed when constructing `ldapdb_uri`

`ldap_host[=<hostname>] [P]`

This option specifies the LDAP server hostname to connect to. LDAPDB ignores this option if `ldapdb_uri` is specified.

`ldap_port[=<portnum>] [P]`

This option specifies the LDAP server port number to connect to. LDAPDB ignores this option if `ldapdb_uri` is specified.

`sasl_ldapdb_dn=<Distinguished Name of target MTA>`

This corresponds to the "DN for Simple Bind" field.

This is the Distinguished Name to be used in the LDAP bind. This DN must have the right to read the **userPassword** attribute for users stored in LDAP. If this option is not specified, the value of `dap_user` option (see below) is used instead. This value is only used if `sasl_ldapdb_mech` has the value `SIMPLE`.

`sasl_ldapdb_id=<SASL username>`

This corresponds to the "ID for SASL Bind" field.

This is the SASL username to be used in the LDAP SASL bind. This user must have the right to read the **userPassword** attribute for users stored in LDAP. If this option is not specified, the value of the `ldap_sasl_user` option is used instead. This value is only used if `sasl_ldapdb_mech` has any value other than `SIMPLE`.

`sasl_ldapdb_pw=<password>`

This corresponds to the "Password" field.

This is the password to be used in the LDAP bind. If this option is not specified, the value of `dap_passwd` option (see below) is used instead.

`sasl_ldapdb_mech={SIMPLE | <SASL mechanism>}`

This corresponds to the "Authentication Mechanism" field.

This option contains the name of the SASL mechanism to use in LDAP SASL bind, or the word `SIMPLE`, if LDAP Simple is to be used. If this option is not specified, the value of the `ldap_sasl_mech` option is used instead. If neither are specified, a default of `SIMPLE` is assumed.

From this list the following options must be specified:

- `sasl_ldapdb_mech`
- either `sasl_ldapdb_dn` or `sasl_ldapdb_id`, depending on the value of the `sasl_ldapdb_mech` option
- `sasl_ldapdb_pw`

The following channel variables affect LDAPDB (note that they need to be specified for each SMTP channel that allows SMTP AUTH):

`sasl_saslMappingRule=<rule number>`

This option specifies how M-Switch users will be located in the corresponding Directory. Valid values are

- 0 ("AD compatible" mapping),
- 1 ("Domain Part search"),
- 2 ("Two searches") or
- 3 ("Single search")

Other options listed below specify which options are used by which mapping rule.

`sasl_saslUsernameAttribute=<attribute name> [P]`

(only used for `sasl_saslMappingRule=0 | 1 | 2`)

This option specifies the attribute that contains the local part of a userid.

`sasl_saslDCMappingSuffix=<base Distinguished Name> [P]`

This option specifies the base DN for all user entries.

`sasl_saslUserParent=<Relative Distinguished Name> [P]`

This option specifies a Relative Distinguished Name (also called a 'container') for all user accounts in the same domain (with the same right hand side). The empty string is used when this option is not specified.

`sasl_saslSearchSuffix=<base Distinguished Name> [P]`

(only used for `sasl_saslMappingRule=1 | 2 | 3`)

This option specifies the base DN for LDAPDB searches for user entries.

`sasl_saslDomainAttribute=<attribute name> [P]`

(only used for `sasl_saslMappingRule=1 | 2`)

This option specifies the attribute that contains the domain (right hand side) part of a userid.

```
sasl_saslFullUsernameAttribute=<attribute name> [P]
(only used for sasl_saslMappingRule=3)
```

This option specifies the attribute that contains a complete userid.

```
sasl_saslDefaultDomain=<domain> [P]
(only used for sasl_saslMappingRule=1 | 2)
```

This option specifies the default SASL domain.

```
sasl_saslUsernamePrefix=<string> [P]
(only used for sasl_saslMappingRule=3)
```

This option specifies an optional prefix to strip from a value of the **sasl\_saslFullUsernameAttribute** attribute.

### 8.7.2.2 How different mapping rules work

Each SASL mapping takes a userid (in `<user>@<domain>` format) and converts it to the corresponding DN in the LDAP Directory. Note that each mapping must be 1-to-1, i.e. if multiple entries end up satisfying an LDAP search used to find the corresponding user, M-Switch assumes that there is no match.

- AD compatible mapping

This mapping rule can be used to construct a DN that is compatible with DNs used in Active Directory. The mapping is a static algorithm.

- The user (left hand side) part of the SASL userid is used to form the least significant RDN, with an attribute type of **sasl\_saslUsernameAttribute** option.
- An additional RDN sequence, specified in the `sasl_saslUserParent` option, is then optionally appended.
- The domain part of the SASL userid is split into multiple subdomains, and each subdomain is used to construct a DC-style RDN sequence which is then appended.
- An additional RDN sequence, specified in `sasl_saslDCMappingSuffix` option, is then optionally appended.

When this configuration is in force, the userid `user@example.com` will have a DN of:

```
<sasl_saslUsernameAttribute>=user, <sasl_saslUserParent>,
dc=example, dc=com, <sasl_saslDCMappingSuffix>
```

- "Domain part search" mapping

This mapping supports holding users with multiple SASL domains in multiple separate subtrees.

- If the SASL userid has a domain that is different from the default domain, then:
  - search from Search base DN for entries matching Domain attribute=domain.

The DN of the matching entry is used for domainsuffix.

Otherwise:

- set the domainsuffix to the Search base DN.
- The user part of the SASL userid is used to form the least significant RDN, with an attribute type of `sasl_saslUsernameAttribute` option.
- The domainsuffix is appended.

For example, with the SASL userid `barabash@example.net`, the default domain is set to `example.net`, the Search base DN set to `o=My Corp, c=US`, and the **Username** attribute set to `cn`, the resulting DN would be `cn=barabash, o=My Corp, c=US`

The Search base DN value is set from the `sasl_saslSearchSuffix` option. The **Domain** attribute is set from the `sasl_saslDomainAttribute` option (the default value is `cn`). The **Username** attribute is set from the `sasl_saslUsernameAttribute` option (the default value is `uid`).

- "Two searches" mapping

This mapping rule is similar to the "Domain Part search" mapping rule, except that instead of forcing all user entries to be directly below the domain suffix, step 2 performs a subtree search under the domain suffix for the user.

- If the SASL userid has a domain that is different from the default domain, then:
  - search from Search base DN for entries matching `Domain attribute=domain`.

The DN of the matching entry is used for `domainsuffix`.

otherwise:

- set the `domainsuffix` to the Search base DN.
- Perform a subtree search from `domainsuffix` for entries matching **Username** `attribute=username`. The resulting match is used as the DN.
- "Single search" mapping

This mapping rule is the most flexible, as it allows users in the same subtree to have different SASL domains. It does this by searching for the complete SASL userid.

- Search from Search base DN for an entry matching `Full username attribute=userid`. The resulting match is used as the DN.

For example with the SASL userid `barabash@example.net`, and **Full username** attribute set to `uid`, the Directory Server would search for a single entry matching `uid=barabash@example.net`

The **Full username** attribute value is set from the `sasl_saslFullUsernameAttribute` option.

---

## 8.8 LEMONADE related extensions

The SMTP server supports several extensions to Message Submission defined by the IETF Lemonade Working Group (<http://www.ietf.org/html.charters/lemonade-charter.html>). The M-Switch SMTP server implements all SMTP extensions listed in the Lemonade Profile document (RFC 4550, section 6). From this list of SMTP extensions, you only need to configure STARTTLS (see [Section 8.4, "Using TLS with SMTP"](#) for more details) and BURL (see below) to enable them, as other extensions are supported automatically and do not need configuring.

---

**Note:** Even though SMTP AUTH is automatically advertised by the SMTP server, you might want to configure it as described in [Section 8.7, "SMTP authentication"](#).

---

To be able to assemble messages from pieces referenced by BURL URLs, the SMTP server needs to know how to authenticate to IMAP servers referenced by such URLs.



Information about allowed IMAP servers is kept in `$(ETCDIR)/switch/burl.xml`. This is an XML file, which has the following format:

- The top level XML element must be `<endpoints>`
- For each trusted IMAP server there is an `<endpoint>` XML element below the top level element, which does not contain any subordinate elements or text.

The `<endpoint>` XML element has one optional and three mandatory attributes:

- `host`, which specifies the hostname or IP address of an IMAP server
- `port`, which specifies the port number of the IMAP server – this is optional and defaults to 143 if not specified
- `username` and `password`, which are the username and password used to authenticate to the server specified by the host and optional port

Here is an example `$(ETCDIR)/switch/burl.xml`:

```
<endpoints>
  <endpoint host="imap.example.com" port="143"
    username="smtpserver" password="some-secret" />
</endpoints>
```

## 8.9 SMTP client

The SMTP client supports the following SMTP extensions:

SIZE	Message size declaration [RFC 1870]
8BITMIME	8bit-MIME transport [RFC 1652]
PIPELINING	Client side pipelining [RFC 2920]
DSN	Delivery Status Notifications [RFC 3461]
ENHANCEDSTATUSCODES	Enhanced Error Codes [RFC 2034]
STARTTLS	Secure SMTP over TLS [RFC 3207]
BY	Deliver by extension [RFC 2852]
MT-PRIORITY	Message priority [RFC 6710]

The SMTP client program for outbound messages makes use of additional tailoring variables in the channel specification. The following per-channel variables (also called ‘options’ below) are currently recognized:

`noesmtplib`

do not attempt to use SMTP extensions

The following options affect how outgoing connections are made:

`nomx`

Do not use DNS MX records

`port=<integer>`

Set the TCP port to be used to integer

The following option affects SMTP client behaviour on unexpected disconnect:

`reconnect_retries=<integer>`

Set the number of reconnect retries to integer

If the SMTP client detects a TCP disconnect while sending a message to another SMTP server and this option is not 0, the SMTP client will immediately try to reconnect to the SMTP server instead of notifying Qmgr about the problem.

If the client receives a 4XX SMTP error code in the SMTP greeting after reconnecting, it will disconnect and reconnect again.

The `reconnect_retries` option contains the number of such reconnects, not counting the first reconnect attempt. The default value is 0, which means that SMTP client will not reconnect following a TCP disconnect.

The following option affects Delivery Status Notification (DSN) extension [RFC 3461] behaviour:

`relaydsn`

If this option is set to `true` and DSN is requested for a message, then the SMTP client will generate 'relayed' DSNs even if the receiving server supports DSN. The default is `false`.

The following options affect how a message content is converted before sending it to another host:

`line=<integer>`

Sets a limit on the length of line that is sent. SMTP and RFC 1652 state that no more than 998 characters should be sent before a line break. By default there is no limit on the line length. This option has no effect if `encode=none` is specified (see below).

`encode=<value>`

Controls how messages are sent. Values of `none` and `default` may result in sending messages which are invalid. The table below describes valid encode values and the resulting action on the SMTP sender.

<code>default</code>	If the body part is mime-unknown and the receiving host indicates support for the 8BITMIME ESMTP extension then the message will be marked as BODY=8BITMIME, otherwise an attempt will be made to downgrade the content to a 7-bit transfer encoding if necessary. Any invalid binary data in headers or content parts will be passed as is. This setting does its best to send only valid messages but will not refuse to transmit a message that it cannot fix.
<code>none</code>	No conversion of content will occur on transmission. If the body part is mime-unknown and the receiving host indicates support for the 8BITMIME ESMTP extension the message will be marked as BODY=8BITMIME. Any invalid binary data in headers or content parts will be passed as is. 8-bit MIME content may be sent to hosts which do not support it. This setting does not alter a message in transit but may send messages which are invalid.
<code>strict</code>	If the body part is mime-unknown and the receiving host indicates support for the 8BITMIME ESMTP extension then the message will be marked as BODY=8BITMIME, otherwise an attempt will be made to downgrade the content to a 7-bit transfer encoding if necessary. Any invalid binary data in headers or content parts will cause non-delivery of the message. This setting will not send invalid messages. It will attempt to downgrade messages with 8-bit content to a 7-bit transfer encoding.Jpara

An 8bit message is one which is flagged as 8bit when transferred into the message switch from a SMTP client, using the 8BITMIME SMTP extension [RFC 1652]. A message is also flagged as 8bit if, when the shaper channel processes a message, MIME body parts that have 8bit or binary content transfer encoding are found.

When an 8bit message is transferred to a host that supports the 8BITMIME extensions, it is flagged as an 8bit message to that host, and transferred unchanged. When the receiving host does not support the 8BITMIME extension, or the message is not flagged as 8bit, then the transfer method depends on the configuration of the `encode` option.

## 8.10 LMTP channel

LMTP (RFC 2033) is a variant of SMTP optimized for delivery to local POP/IMAP Message Stores, also supported by Isode's M-Box server. Message Delivery by LMTP is provided by the `slmtp` process. This process is started by the `Qmgr` process (see [Section 34.3, "Queue Manager"](#)), which may run many instances of the `slmtp` program. (Note that the similarity of the SMTP and LMTP protocols mean that the same program is run when the SMTP and LMTP channels are invoked).

The LMTP channel is configured in the same way as the outbound SMTP channel (see [Section 8.9, "SMTP client"](#)), with the following exceptions:

- The channel name is usually `lmt`
- The channel type is always `out` (`type=out`)
- The destination LMTP server is specified in the `lmt` channel variable. This variable can contain one of the following:
  - a Unix domain socket, for example:

```
"lmt=/var/run/lmt"
```

- TCP `<hostname>:<port>`, for example:

```
"lmt=lmt.example.com:2001"
```

Within MConsole, the `lmt` channel variable can be configured via the "LMTP socket" field on the "Program"/"Out" tab for the LMTP channel.

---

**Note:** The latter form must be used on Windows.

---



---

**Note:** It is also possible to connect to LMTP over TCP by using `lmt=<hostname>` and `port=<port>`

---



---

**Note:** The TCP port number should NOT be set to 25.

---

The following example shows a typical configuration for the LMTP channel:

```
chan lmt prog=slmt show="with LMTP" type=out adr=822
content-out="822,822-8" lmt="lmt.example.com:2001" hdrout=822
```

The following channel variables used by outbound SMTP channel have no effect on the LMTP channel:

- `noesmt`

- nomx
- reconnect\_retries

# Chapter 9 Configuring an X.400 Messaging System

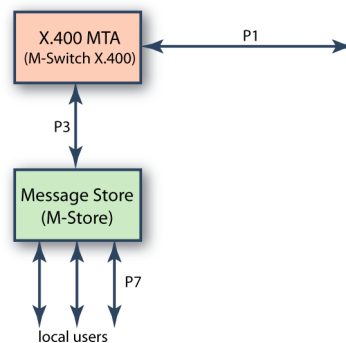
This chapter gives instructions on how to set up an X.400 messaging configuration using MConsole.

The initial set up instructions are for a basic configuration like the one illustrated in [Figure 9.1, “Initial X.400 configuration example”](#). For this configuration, we will use most of the default values set by MConsole. The latter part of the chapter describes how you can extend and tailor the configuration.

The examples given throughout the chapter relate to a simple aviation configuration (AMHS). This setup initially has a single X.400 MTA and an X.400 Message Store. Later, you can extend it by adding another X.400 MTA, or by creating External MTAs.

In our example, MConsole will be used to create the messaging configuration in an LDAP/X.500 Directory, accessed using LDAP. Messages will be routed using information configured in the same LDAP/X.500 Directory.

**Figure 9.1. Initial X.400 configuration example**



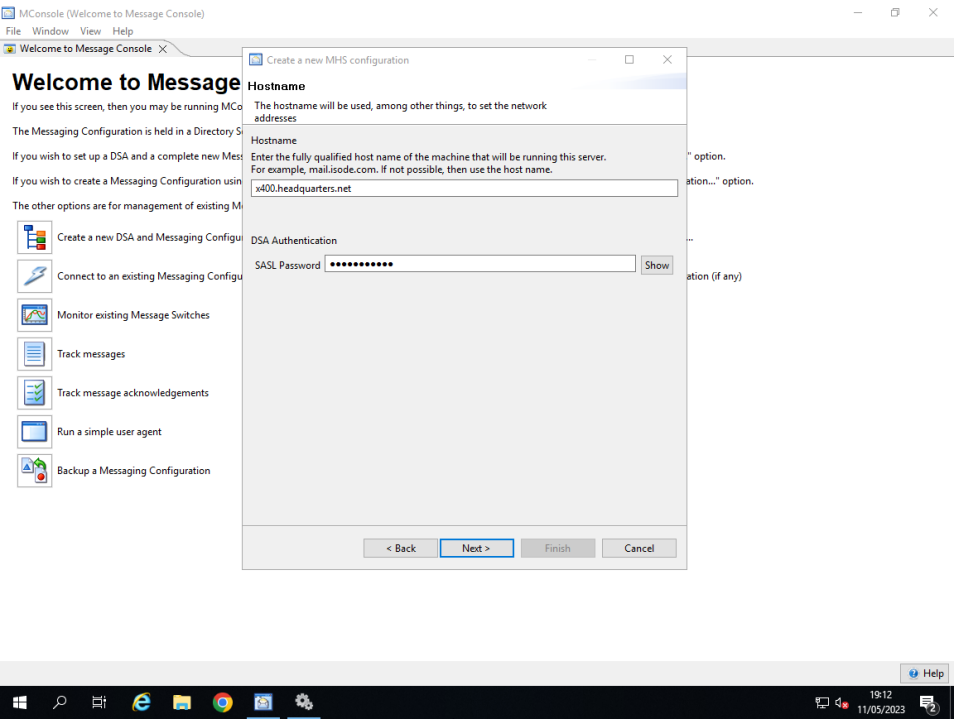
## 9.1 Creating a configuration

After setting up your MConsole Bind Profile and DSA, as described in [Section 4.4.2, “Creating a new Directory”](#) a series of wizard pages enable you to configure a basic X.400 configuration.

### 9.1.1 Configuring the Hostname

In the next wizard page, you are asked to enter the fully qualified hostname for the MTA you wish to create.

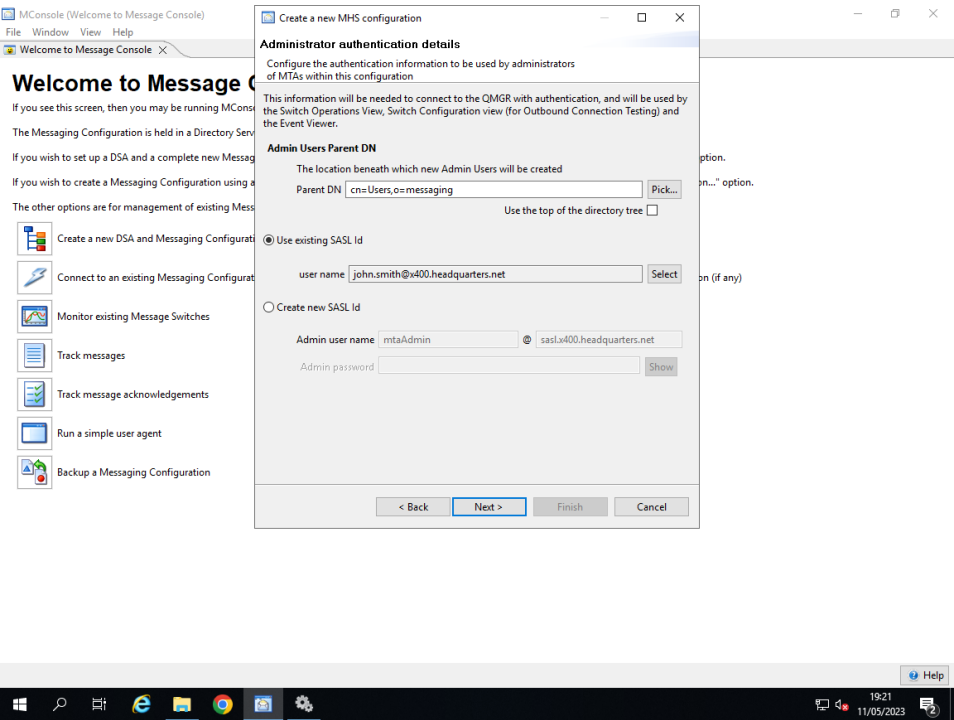
Figure 9.2. Fully Qualified Hostname screen



9.1.2      **Configuring the Administrator Authentication**

In the next wizard page, you are asked to enter Administrator authentication in the form of a SASL ID and password. This information will be used by MConsole to authenticate to the Queue Manager, allowing monitoring and management of the MTA using the SOM protocol. You can either create a new SASL ID or use an existing one. See [Section 13.4, “Creating Accounts”](#) for further details.

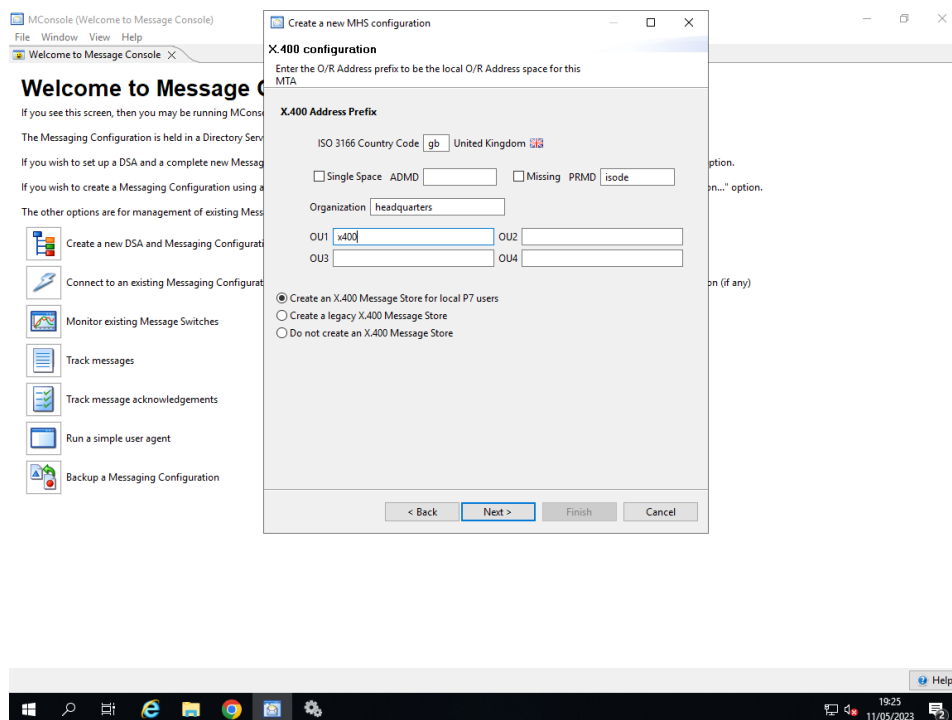
Figure 9.3. Administrator Authentication screen



### 9.1.3 Configuring the X.400 O/R Address (X.400)

In the next wizard page, you are asked to enter the O/R Address regarded as local to the MTA being created. You can also choose whether or not to create an X.400 Message Store (either a new-style Message Store which holds its index entries locally, or a legacy Store which uses the DSA to hold index entries).

**Figure 9.4. X.400 O/R Address screen**



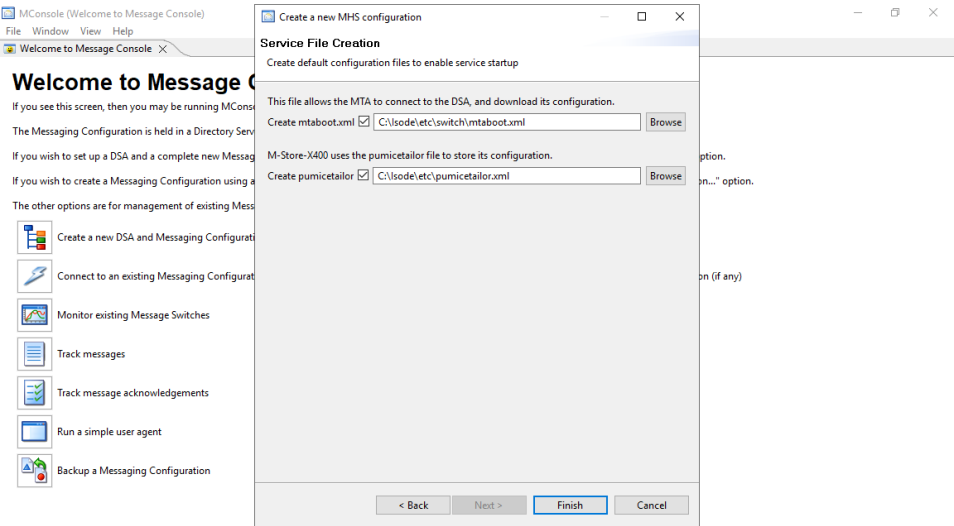
### 9.1.4 Configuring the MTA service file creation

In the next wizard page, you are asked to confirm the creation of the service files. By default the files are copied into the live configuration on the local system. If these already exist, they cannot be overwritten and the **Finish** button does not light and the filename appears in red.

You need to delete the existing file or install the file elsewhere to complete the wizard

- *mtaboot.xml* (for M-Switch)
- *pumicetailor.xml* (for M-Store)

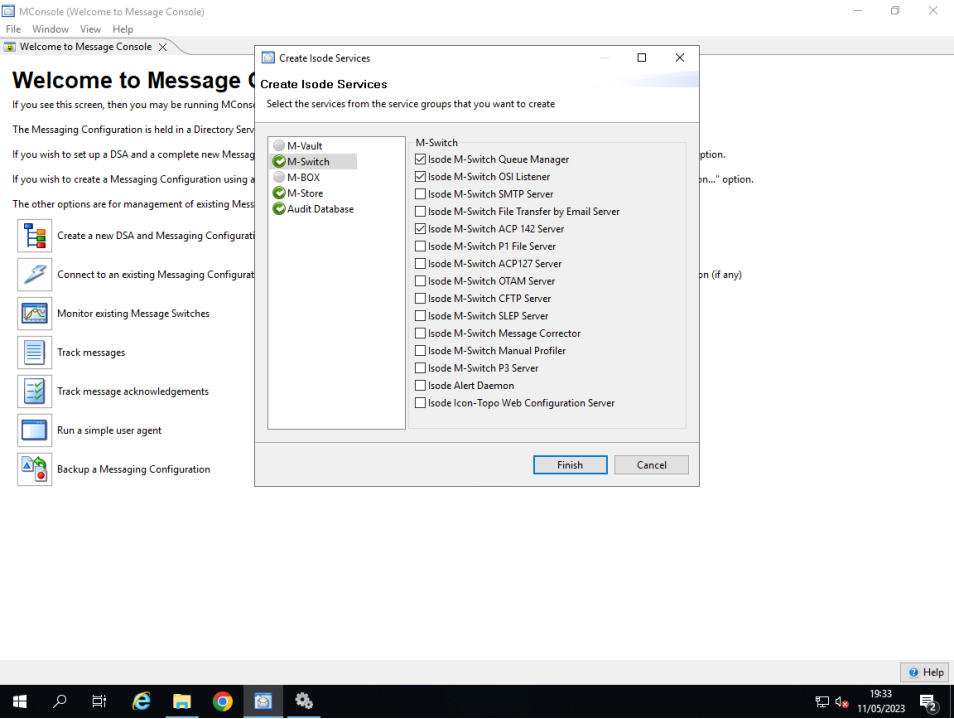
Figure 9.5. MTA Service file creation screen



9.1.5 Services Creation and Completed X.400 MTA

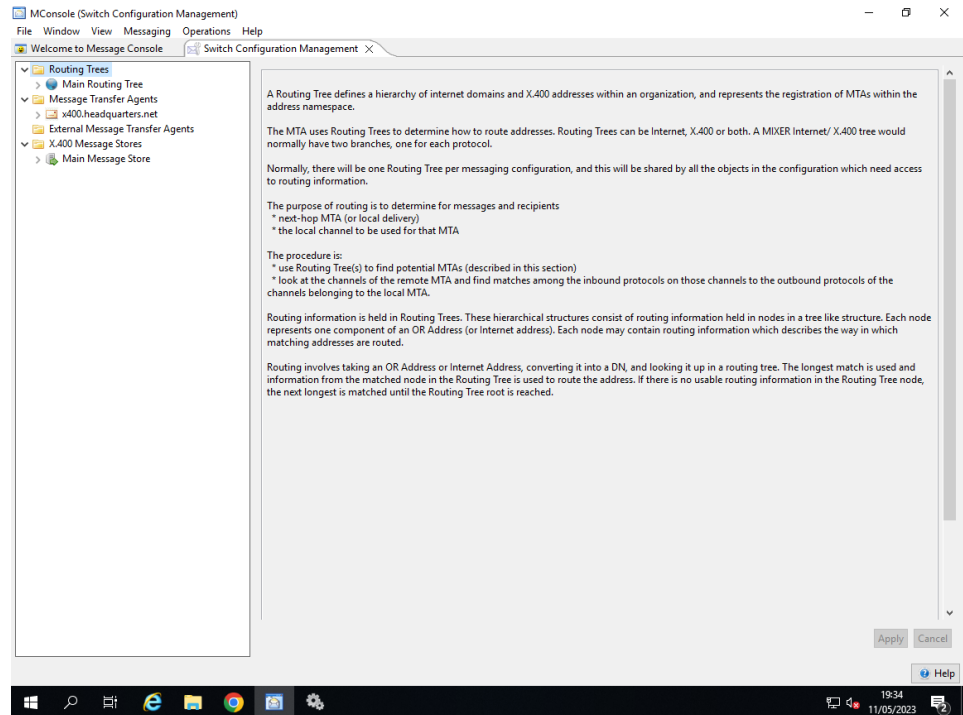
After clicking on **Finish**, you can create the required Windows services.

Figure 9.6. Create X.400 Messaging Services



After clicking on **Finish**, the X.400 configuration is created and displayed ready to view and edit.



**Figure 9.7. X.400 Configuration complete screen**

## 9.1.6 Editing the X.400 Configuration

### 9.1.6.1 Routing trees

Expand the **Routing Trees** folder and you will be able to see that one branch of **Main Routing Tree**, for X.400 messages has been created.

The Routing Tree, **Main Routing Tree**, represents the message routing information which has been set up in the Directory for the namespace

`/OU=x400/O=headquarters/PRMD=Isode/ADMD= /C=gb/.`

From the **Switch Configuration Management** view, more address components can be added as nodes or deleted. Properties of each of the nodes can be displayed. These include:

- Directory subtree information.
- MTAs which can deliver to this address and associated channel information.

Later, when we add users, further routing information will be added to the Routing Tree information in the Directory.

### 9.1.6.2 Message Transfer Agents

The tailoring information for the MTA has been created and placed in the Directory. To see all the MTA tailoring properties, select the MTA within the **Switch Configuration Management** view. This displays the general properties of the MTA within the right hand side window.

You should also expand the folders below the MTA, which will show its Channels, Tables, Directory Profiles and Logs.

#### 9.1.6.2.1 MTA properties

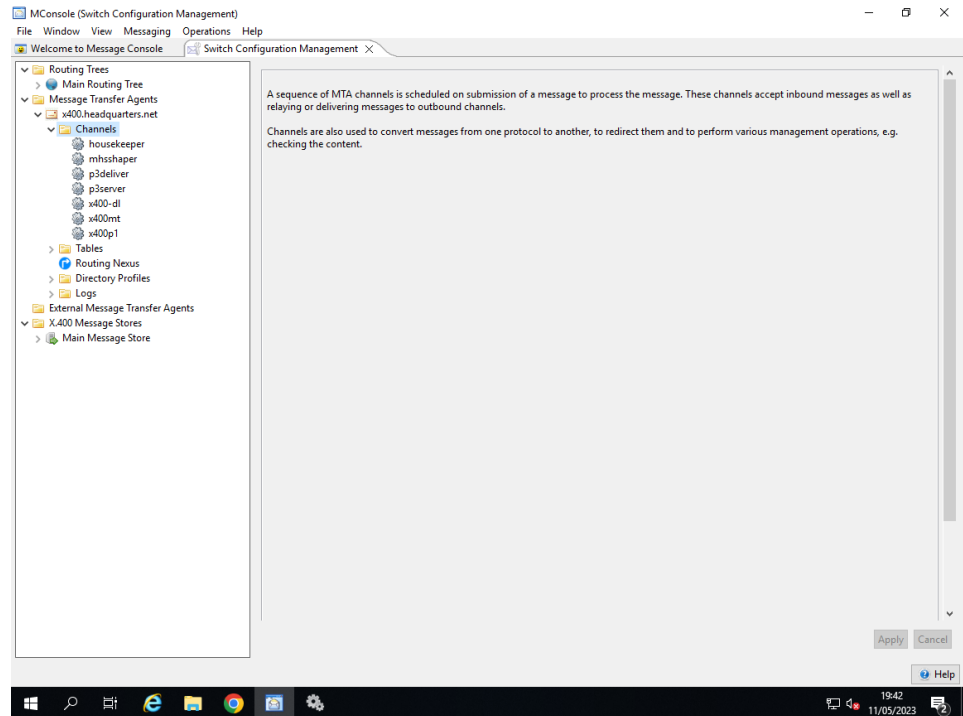
To display the properties set for the MTA, click on the MTA name. The creation wizard page will have set suitable values for a basic system, so you should not need to alter any of the MTA values at this point. However, should you want to tailor values later on, a description of all the fields can be found in the *M-Switch Advanced Administration Guide*.

### 9.1.6.2.2 Channels

Suitable channels have also been created, as shown in [Figure 9.8, “X.400 MTA Channels”](#). The channel properties can be displayed by expanding the **Channels** folder under the MTA window and clicking on the channel in question. Again, you may want to configure the channels and their properties later. The various property fields are described later on in this manual and in the *M-Switch Advanced Administration Guide*.

The following gives a summary of the function of the channels created in the basic X.400 MTA. Further information, including tailoring options, for all of these channels can be found in the *M-Switch Advanced Administration Guide*.

**Figure 9.8. X.400 MTA Channels**



acp142

(optional) This is a channel that implements the military protocol ACP142 (P\_MUL).

ftbe

(optional) This is the channel which is used to transfer emails to and from the filesystem. Typically it's used in conjunction with Sodium Sync to allow directory replication over email. Configuration information for the fteserver is given within the *M-Switch Advanced Administration Guide*.

housekeeper

This is a special housekeeping channel used by the MTA to perform tasks such as deleting messages when they have been processed, or generating reports. Further information on this channel can be found in the chapter called 'MTA Tailoring' in the *M-Switch Advanced Administration Guide*.

hybridList

This channel is for handling hybrid Distribution Lists

mhsshaper

This is used for conversion between MIME and X.400. For simple configurations this will work as created. For details see the *M-Switch Advanced Administration Guide*.

p1file

(optional) This is a channel similar to x400p1, but works by reading/writing P1 APDUs from/to filestore

**p3deliver**

This channel is used to deliver messages into the X.400 Message Store.

**p3server**

This channel is used by a P3 User Agent to submit and accept delivery of messages to/from the MTA, and also to accept submissions from an X.400 Message Store.

**x400-dl**

This channel is for handling the resolution of X.400 distribution lists.

**x400mt**

(optional) If you have selected to create a Gateway channel, this is the channel used for transferring (both in and out) messages from a Gateway application such as the Isode Gateway X.400 API.

**x400p1**

This is the main protocol channel for transferring messages from one X.400 MTA to another. The x400p1 channel can operate in different modes.

### 9.1.6.2.3 Tables

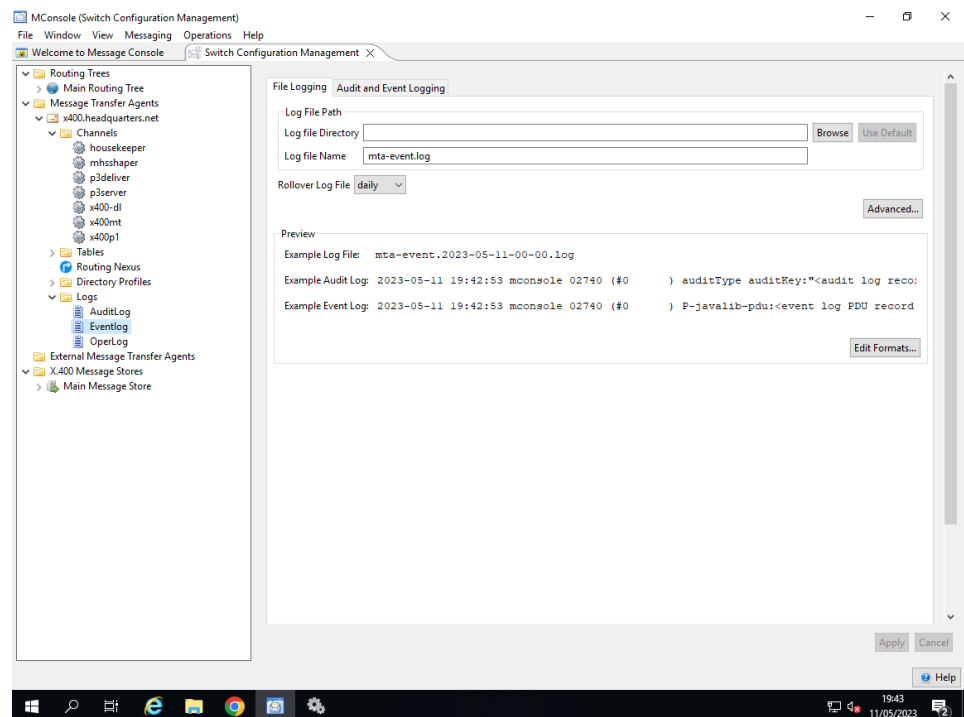
A number of tables can be displayed under the **Tables** folder. Tables can be held in text files, or in the Directory. The only mandatory table is the OR table, which the wizard sets up as empty. The OR table is only used for table based routing. Configuring M-Switch using text tables instead of the Directory is described in the *M-Switch Advanced Administration Guide*.

Tables may be required for setting up certain facilities such as distribution list handling. Even if you are using Directory-based routing, you do need tables for some facilities. However, these tables will generally be configured in the Directory.

### 9.1.6.2.4 Logging

The three logging streams created by the MTA creation wizard are displayed under the **Logs** folder. Their properties can be displayed and edited in the same way as other tailoring entities. The logging tailoring set here is likely to be adequate for your basic system.

**Figure 9.9. Logging Configuration**

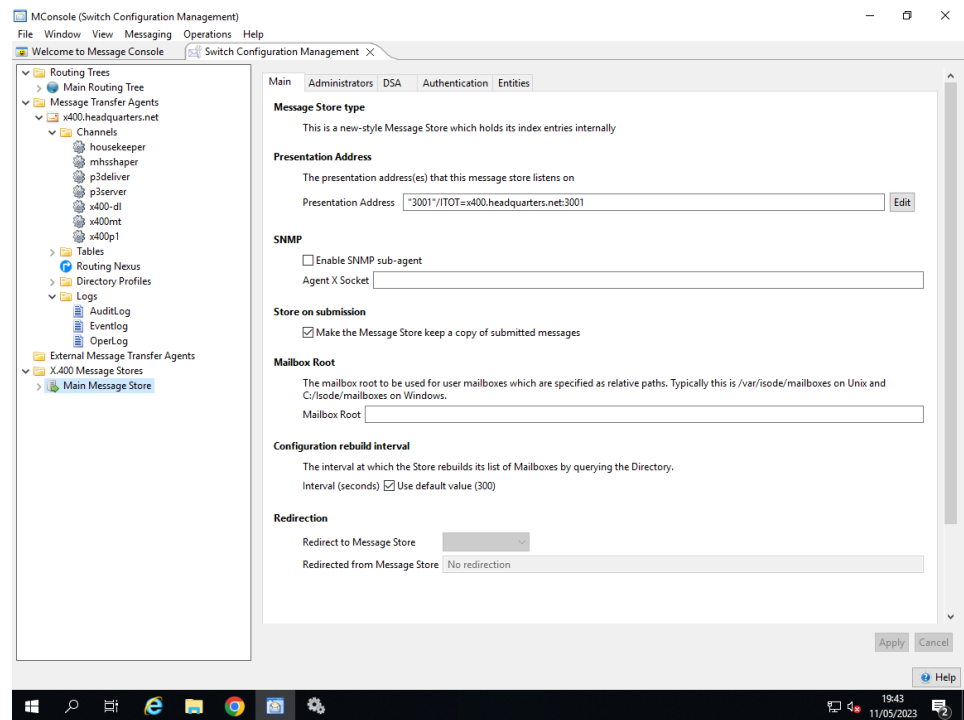


Section 34.2, “Configuring logging” describes how you can tailor the level of logging required for individual programs, channels and protocol layers. You can also specify that the logging for some (or all) programs is sent to separate files.

### 9.1.6.3 X.400 Message Stores

Expand the **X.400 Message Stores** folder and note that the **Main Message Store** has been created. If you have created an X.400 Message Store after going through the wizard to create your messaging configuration then it will have whatever name you gave it when creating.

**Figure 9.10. X.400 Message Store Configuration**



# Chapter 10 X.400 Message Store

The configuration for the P7 Message Store (M-Store X.400) can be stored in the Directory, and is set up using MConsole when an X.400 or MIXER messaging configuration is created. This chapter describes how to create and edit the properties of an existing X.400 Message Store.

## 10.1 Creating a new X.400 Message Store

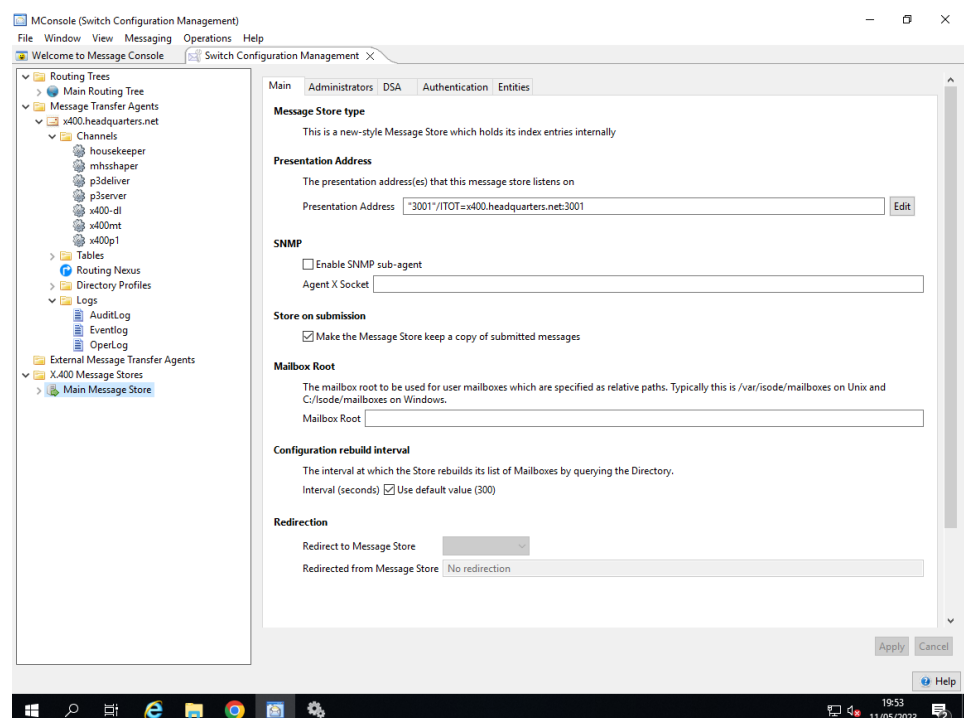
To create a new X.400 Message Store, right click on the **X.400 Message Stores** folder and select the **New X.400 Message Store** option. This starts the **Create a New X.400 Message Store** wizard, which will prompt for some details of the configuration.

## 10.2 Editing an X.400 Message Store

On the **Main** tab below, you can configure the **Presentation Address(es)** on which the Message Store listens for incoming P7 connections.

You can also configure other Message Store wide features such as SNMP, and the directory in which the X.400 mailboxes are held.

**Figure 10.1. Message Store configuration Main tab**

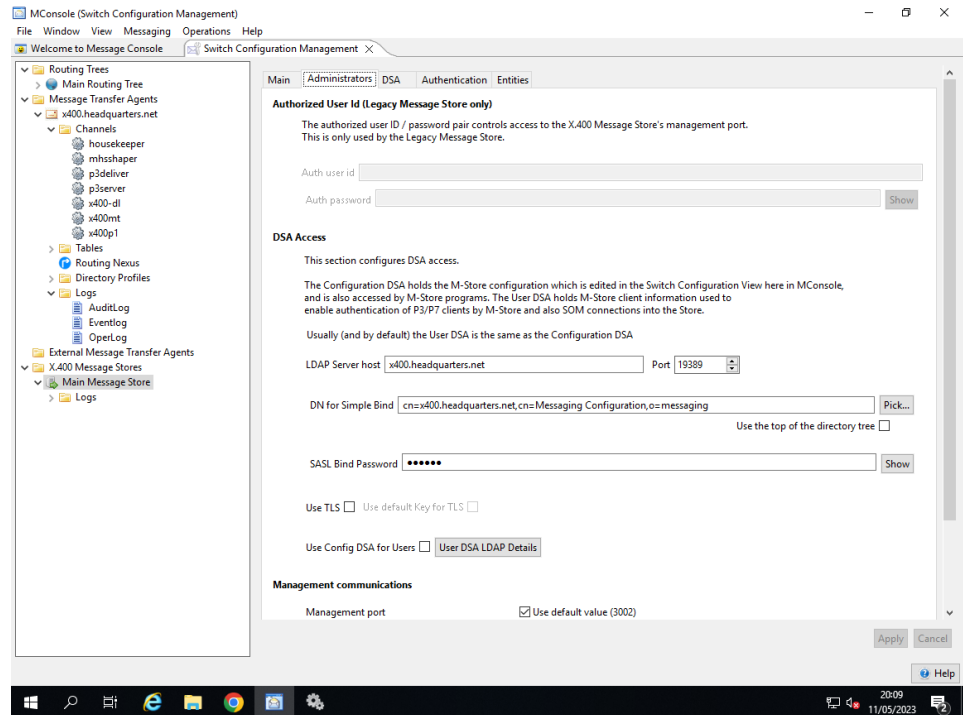


The **Administrators** tab allows configuration of Message Store administration, enabling MConsole's **Store Operations** View and other SOM clients to connect to the Message

Store to manage and monitor its operations. Other SOM clients include the Message Store maintenance applications `mstore_backup.tcl` and `mstore_tidy.tcl`.

Different fields on the **Administrators** tab are enabled or disabled, depending on whether a new-style or Legacy Message Store is being edited.

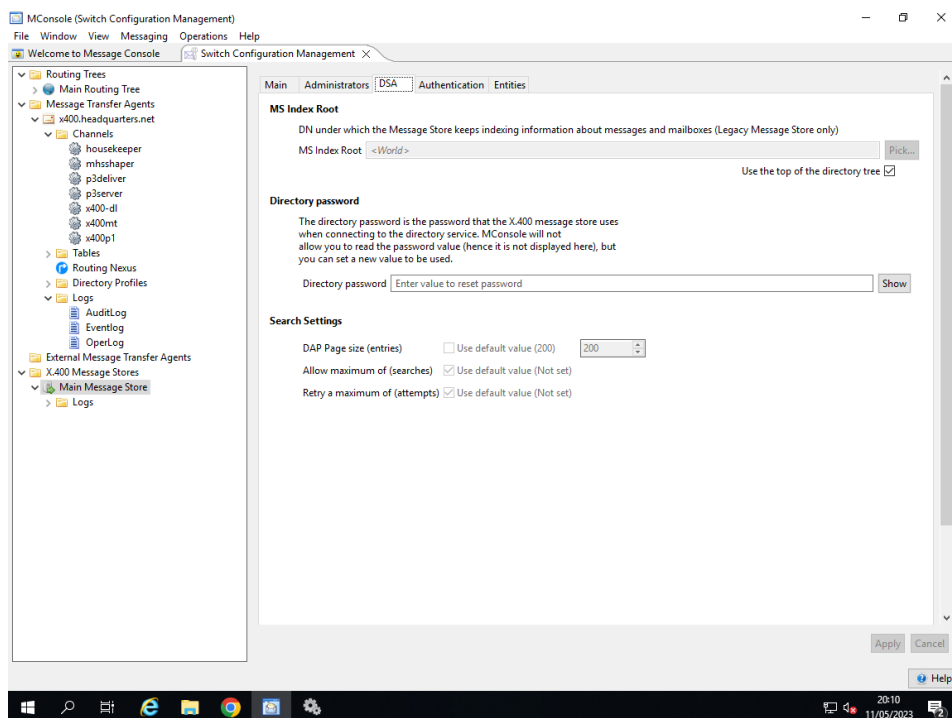
- A new-style Message Store uses SASL for authentication (in the same way that the Queue Manager does). In this case, the "Auth user id" fields are disabled, and the "SASL Server Configuration" fields are enabled.
  - The SASL Configuration Entry is the DN of the DSA's configuration entry. This is usually "cn=core,cn=config".
  - The Authentication Mechanism pulldown allows configuration of the SASL mechanism which the Message Store uses when binding to the DSA. This should normally be left set to "Simple".
  - The Password field allows the password used by the Message Store when binding to the DSA to be specified.
  - The "ID for SASL Bind" field allows a SASL ID to be used by the Message Store when binding to the DSA to be specified. This should only be used in conjunction with an authentication mechanism other than "Simple".
  - The "DN for Simple Bind" field configuration of the Distinguished Name which will be used when binding to the DSA with Simple authentication. The DN chosen must be that of an entry which has the necessary permissions to allow it to read passwords from SASL user entries. When a DSA is created for Messaging use, a suitable entry and associated access control will be set up, with an RDN of `cn=Isode Application Server`.
  - The "Host" field allows the hostname of the DSA to which the Message Store will connect to be specified. If unspecified, this will default to "localhost".
  - The "Port" field specifies the LDAP port to which the Message Store will connect. This defaults to 19389.
- A Legacy Message Store has a single "Authenticated User ID" and associated password. In this case, the "Auth user id" fields are enabled, and the "SASL Server Configuration" fields are disabled.
- The "Management port" field allows configuration of the port on which the Message Store will listen for management connections.
- The maximum number of simultaneous management connections can also be configured.
- Use of TLS encryption for SOM connections can also be enabled. If this is enabled, the path under which TLS identity files are located must also be configured.

**Figure 10.2. Message Store Administrators tab**

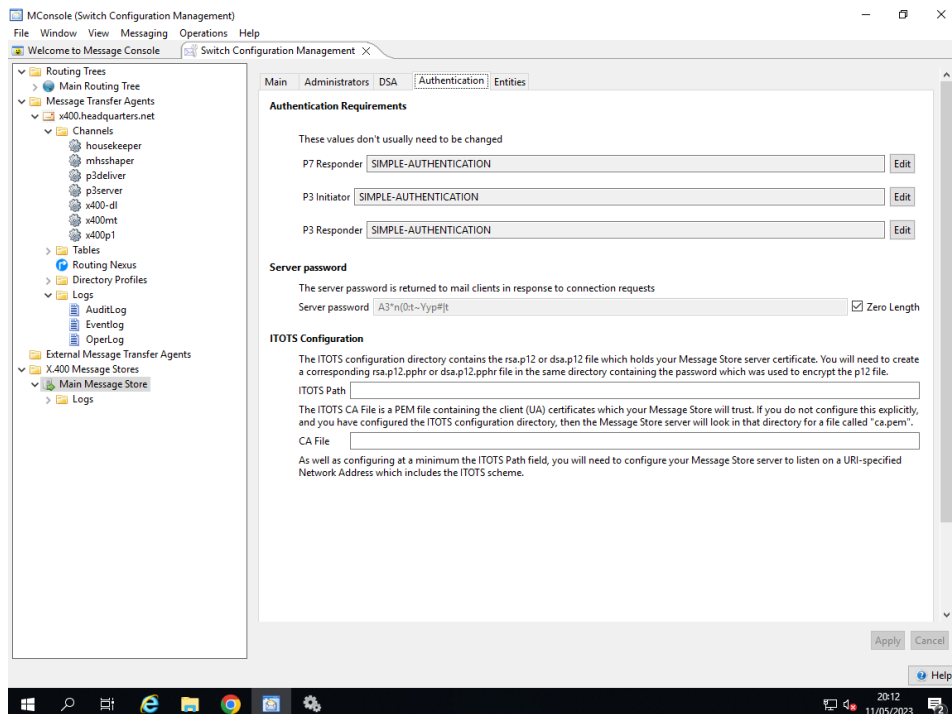
The Legacy Message Store maintains an index of messages in the mailboxes in the Directory. This is configured in the **DSA** tab. A new-style Message Store which does not hold its index in the Directory only makes use of the Directory Password field.

The Message Store's **Directory Password** (i.e. the password which the Message Store uses when binding to the Directory), is also configured, here together with other settings which control access to the DSA.

The **Search Settings** allows you to fine tune the way that the Message Store uses the DSA to store information about messages. For example, under a heavy load, the DSA may return "BUSY" to some operations.

**Figure 10.3. Message Store DSA configuration**

On the **Authentication** tab, the Message Store's **Server Password** (the password which the Message Store returns in a Bind Response, when a User Agent or MTA binds to it), and the Message Store's **Authentication Requirements** can be configured.

**Figure 10.4. Message Store Authentication tab**

Three sets of Authentication Requirements can be configured: those used when acting as a P7 responder (i.e. handling a bind from a User Agent), those used when acting as a P3 Initiator (in other words, the checks which the Message Store makes on the bind response which it receives from the MTA when the Store binds to the MTA for submission), and those used when acting as a P3 Responder - when the MTA binds to the Store for delivery.

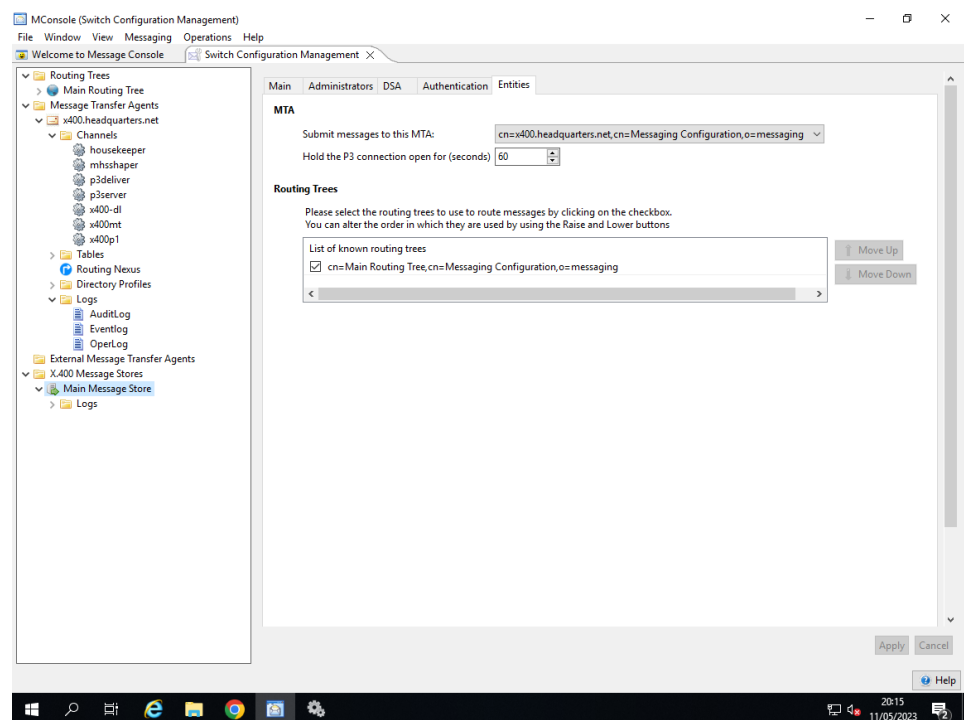


Each set of Authentication Requirements contains the following switches: **MTA Name Present**; **AET Present**; **AET valid**; **Network Address**; **Simple Authentication**; **Strong Authentication** and **Peer Connection**. One or more of these requirements can be set, although some are not appropriate to all situations (for example, **MTA Name Present** is not relevant to P7 binds), and some are mutually exclusive (**Simple Authentication** and **Strong Authentication**, for example).

The ITOTS section allows the Message Store's security environment to be configured so that P7 clients can use TLS to connect to the Message Store in a secure way.

The **Entities** tab specifies the MTA that the Message Store uses for message submission and the Routing Trees which it uses when looking up the O/R Addresses of users who are binding to it.

**Figure 10.5. Message Store entities configuration**



## 10.3 Menu operations on X.400 Message Stores

When a new Message Store has been created a number of new administrator options will be available by selecting the Message Store icon and clicking on it with the right mouse button.

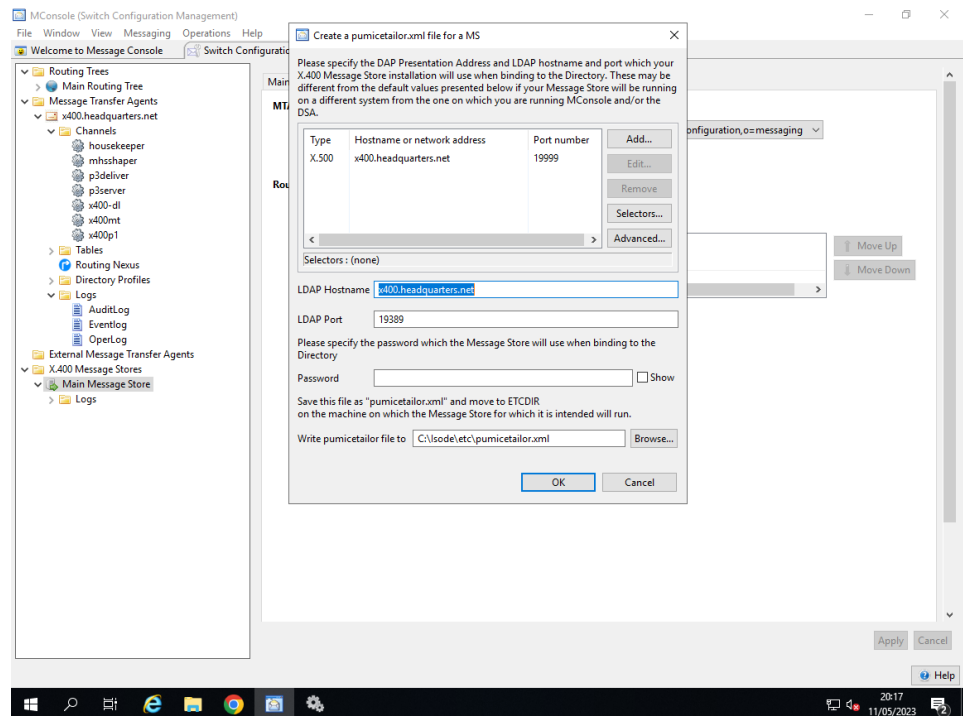
### 10.3.1 The pumicetailor.xml file

When a new Message Store has been created, or the password for an existing Message Store has been changed, a new/updated *pumicetailor.xml* file will be required. This file is read by the Message Store on startup, and contains the information which the Message Store needs to be able to bind to the Directory and read the configured information described above.

You can create a *pumicetailor.xml* file which contains the appropriate information for a Message Store at any time by selecting the Message Store icon, clicking on it with the right

mouse button and then selecting **Create pumicetailor.xml**. This will display the following window:

**Figure 10.6. Creation of new pumicetailor.xml file**



The new tailoring file will be named *pumicetailor.xml*.

### 10.3.2 Service Keys

The *pumicetailor.xml* can contain passwords. These can be obfuscated using the Service Key facility.

To do this, use the **Create Service Key** menu option. On Unix platforms, you will need to be running MConsole as root. In all cases, you need to be running MConsole on the system on which your Message Store will actually run.

You will need to enter a passphrase, which will be used to encrypt the Service Key, and (on Unix only) specify the userid under which the Message Store will run.

Then create a new *pumicetailor.xml* file via the same right-mouse button as used to create the Service Key.

If you have created a Service Key for the Message Store, you will be prompted to enter the passphrase which you entered in the previous step, and passwords will be encrypted using the Service Key before they are written into the *pumicetailor.xml* file.

A password which has been encrypted in this way is prefixed with "{spcrypt3}" when stored.

On Unix, the Service Key will be stored in a file in *(ETCDIR)/servpass*. Standard Unix file protection is used to limit read access to the userid under which the Message Store runs.

On Windows, OS-specific mechanisms are used to protect the Service Key, and the restrictions on the userid do not apply.

### 10.3.3 Configure DSA for Message Store use

This menu option can be used to make sure that the DSA is correctly configured to be used by a Legacy X.400 Message Store which uses the DSA to hold message index entries. It creates the right entries and indexes so that the performance is optimized.

See the *M-Store Administration Guide* for further details.

# Chapter 11 Managing X.400 Messaging Users

The creation and management of X.400 user accounts is described in this chapter.

Before adding any users you need to have created the required Routing Tree(s), MTA(s) and X.400 Message Store(s). If you created an initial messaging configuration using the creation wizard then these should already be in place.

Once you have created a few users, as described in the first part of this chapter, you will have completed all the necessary configuration tasks required for an initial system. We suggest that you configure at least one regular mail user before starting your mail system. You can return to this section of the manual later to tailor this part of your configuration further.

X.400 conformant distribution lists are created and managed as described in [Chapter 28, Directory Based X.400 Conformant Distribution Lists](#).

---

## 11.1 X.400 messaging users and White Pages Entries

Many organizations structure the Directory Service so that Entries which relate to individual users are held in what is termed White Pages. Such Entries can hold a range of information such as Names, Addresses telephone numbers, email addresses etc.

The Isode Messaging Configuration can be set up to either create and manage such a setup, or to integrate with existing data.

In this section each of the X.400 users that are created also allow a White Pages Entry to be used by M-Switch to configure certain features of the user.

### 11.1.1 Aviation (AFTN) Attributes

The Aviation Market has a particular need to specify the ability of users to, for example, receive messages no greater than a certain size, or of a restricted set of Content Types. The AFTN tab in the White Pages Entry can configure a number of these. These can also be used in non Aviation configurations.

### 11.1.2 Groups

The White Pages Tab also allows two sorts of Groups to be specified

- General Groups

This allows White Pages Entries to be grouped together into an arbitrary Group. This group can be referenced by Authorization Rules in the same way as an explicitly configured Authorization Group. See [Section 38.1, "Authorization"](#) for a description of how Authorization and Groups work.

- Closed User Groups

This allows White Pages Entries to be grouped together into a Group who are only allowed to submit messages to each other.

---

## 11.2 Creating X.400 messaging users

X.400 messaging users are created, modified and deleted from the X.400 Mailbox Management view within MConsole. The left-hand window of this display shows a tree-structured representation of the X.400 address space which your Routing Trees define. When an X.400 messaging user has been selected, the right-hand window allows the user's configuration to be edited. Note that new address nodes (e.g. ADMDs, PRMDs etc) need to be added via the **Switch Configuration Management** view.

To create a new mail user, select the point in the address space at which you want the user to be located. A set of buttons along the top of the **X.400 Mailbox Management** window allows the creation of seven basic types of messaging users (you can also right click on the Routing Tree node to retrieve these options):

### **P7 Mailbox**

This enables you to configure access to a X.400 Message Store for users with X.400 User Agents.

### **P3 User Agent**

This allows you to create a user who will access the MTA directly using the P3 protocol for message submission and delivery.

### **X.400 Distribution List**

This allows you to create an X.400 Distribution List. The addresses to which the Distribution List expands are added once the List itself has been created.

### **X.400 Redirect**

This allows you to create an address node which redirects messages to another (possibly external) X.400 address.

### **Other User**

This allows you to create a generic address entry which could be used for message delivery via the shell or FAPI channels.

### **Add File Transfer By Email User**

This allows you to create a generic address entry which could be used for message delivery via the FTBE channel.

### **Add Using Template**

This allows you to create a generic address entry using a template defined in this view under the **User Templates** folder.

The following sections describe how to create users of each type

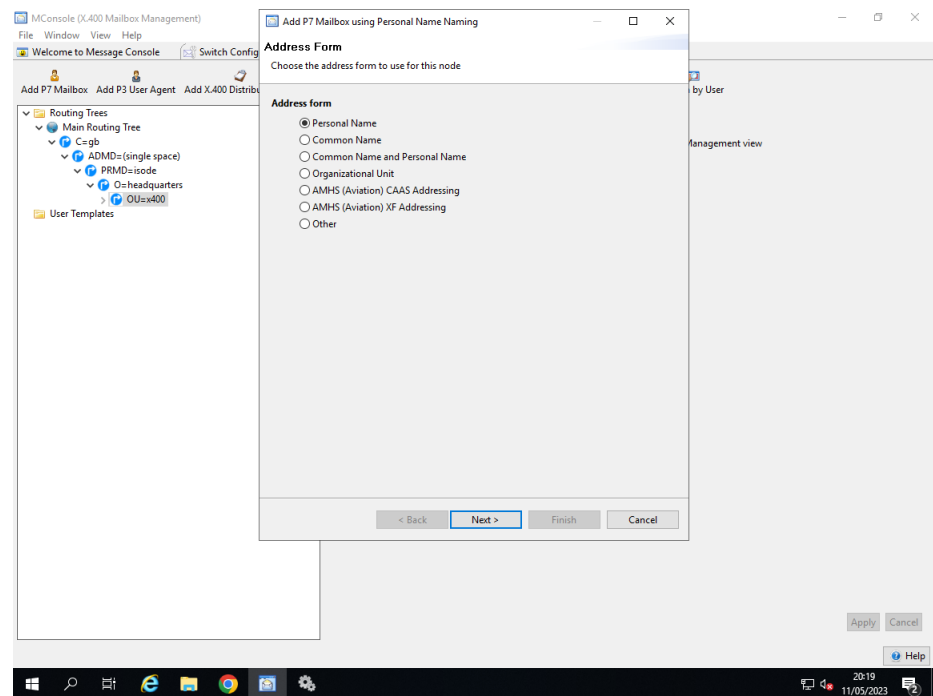
### 11.2.1 P7 Mailbox

1. Choose the address form of the mail user to be created. The Personal Name form allows the user to be named with Surname, Initials, Given Name and Generation Qualifier. The Common Name (also known as CAAS Addressing in the AMHS world) or Organizational Unit (XF Addressing) forms may be chosen instead.

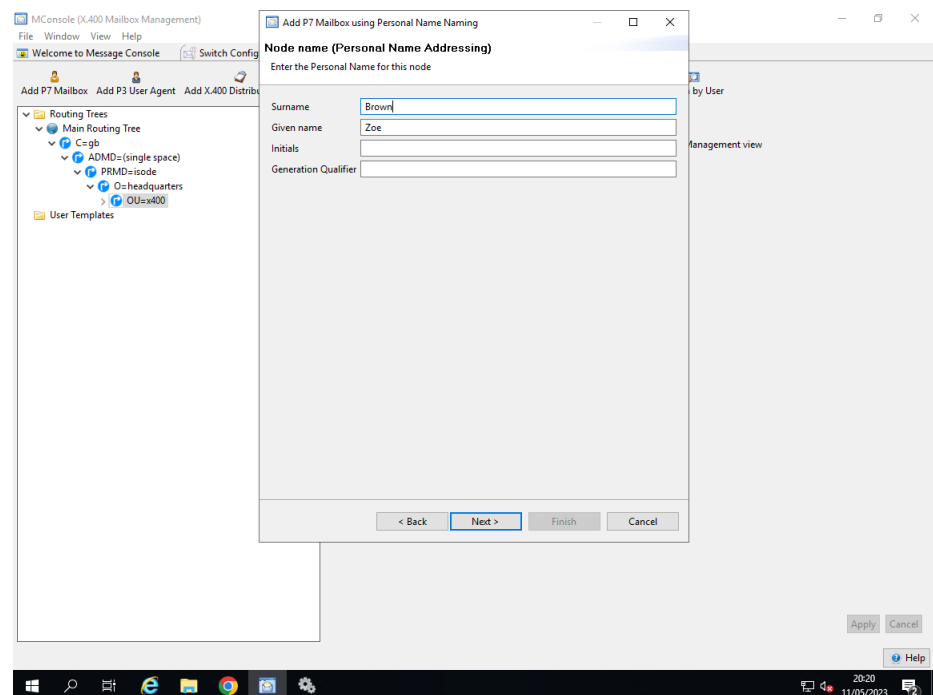
---

**Note:** If you choose to configure a CAAS address or XF address, selecting that choice invokes an intelligent editor which is aware of the address forms and supports intelligent editing of the Common Name and Organizational Unit respectively.

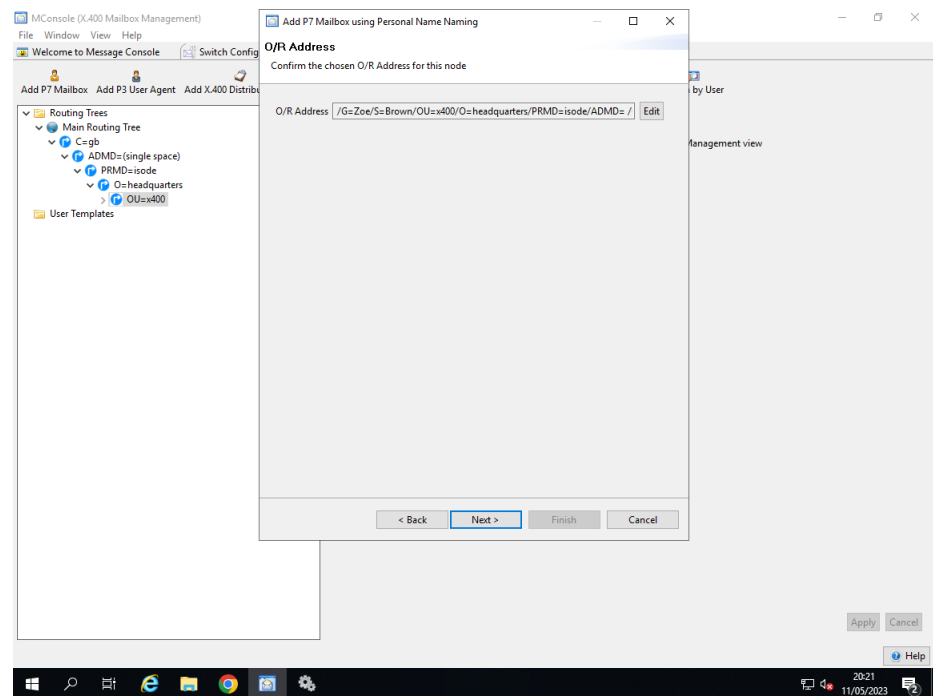
---

**Figure 11.1. Address Form**

2. In the next screen you will need to enter the naming components appropriate for the naming form you have chosen. The screen below shows the values you need to enter for a mailbox named by Personal Name.

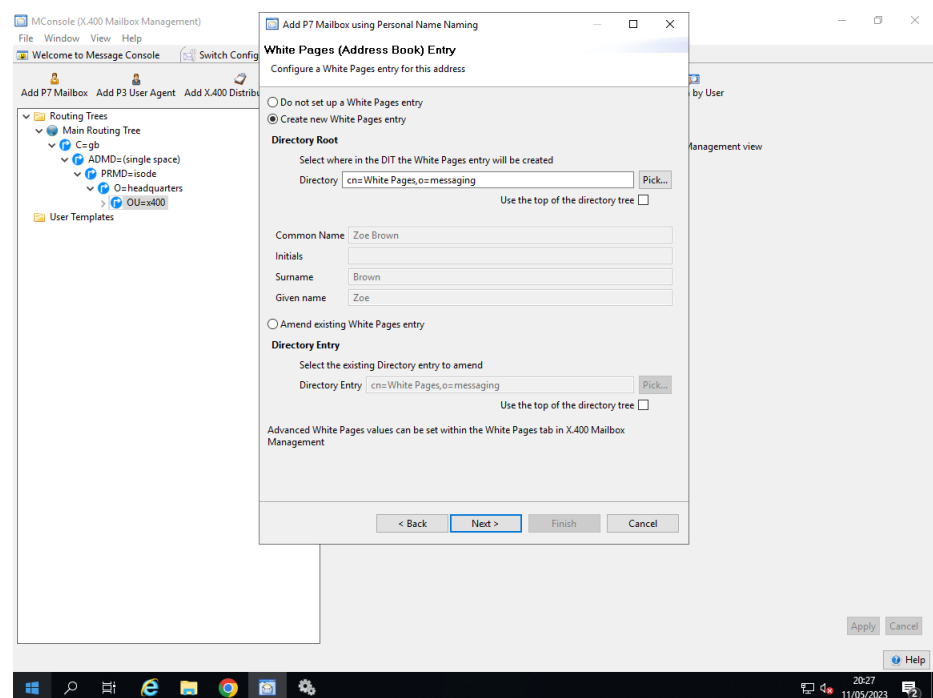
**Figure 11.2. Naming Component**

The point in the OR hierarchy from which the user creation wizard was launched sets the base part of the mail address to be created, and the full resulting O/R Address is displayed on the next wizard page:

**Figure 11.3. O/R Address**

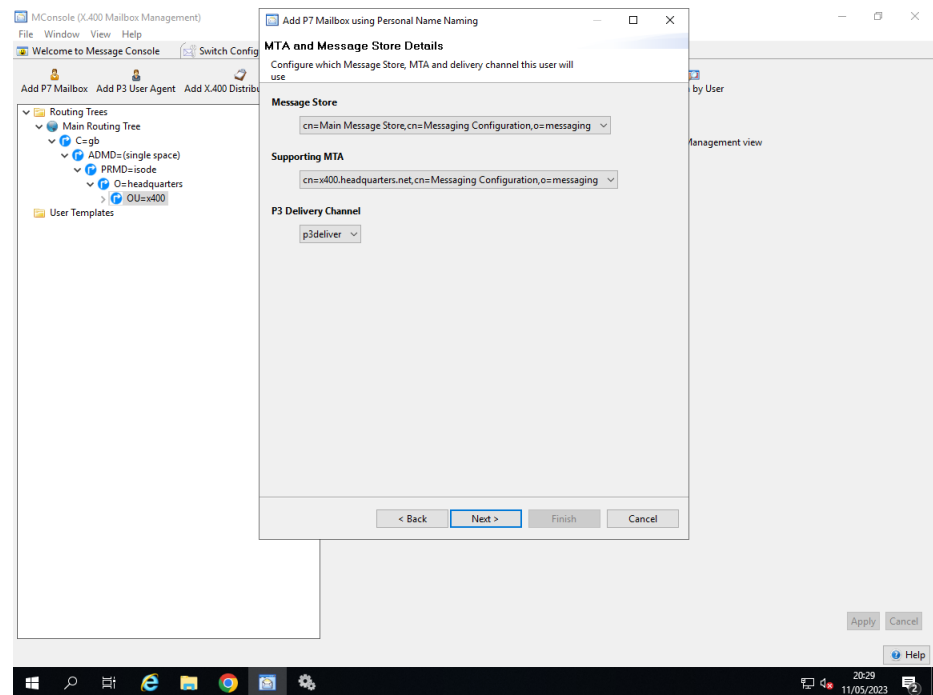
3. Configure White Pages information for the new user. There are three options here:

- Do not set up any White Pages information.
- Create a new White Pages entry (default). The user creation wizard will suggest a location in the DIT under which the entry will be created and naming attributes for the entry based on what has already been entered for the user's O/R Address, but these can be overridden if desired.
- Amend an existing White Pages entry. A Directory Browser is provided to allow selection of the entry to be amended.

**Figure 11.4. White Pages Entry**

4. Select the X.400 Message Store, MTA, and delivery channel which this mail account will use:

**Figure 11.5. Configuring the X.400 Message Store, MTA, and Delivery Channel**

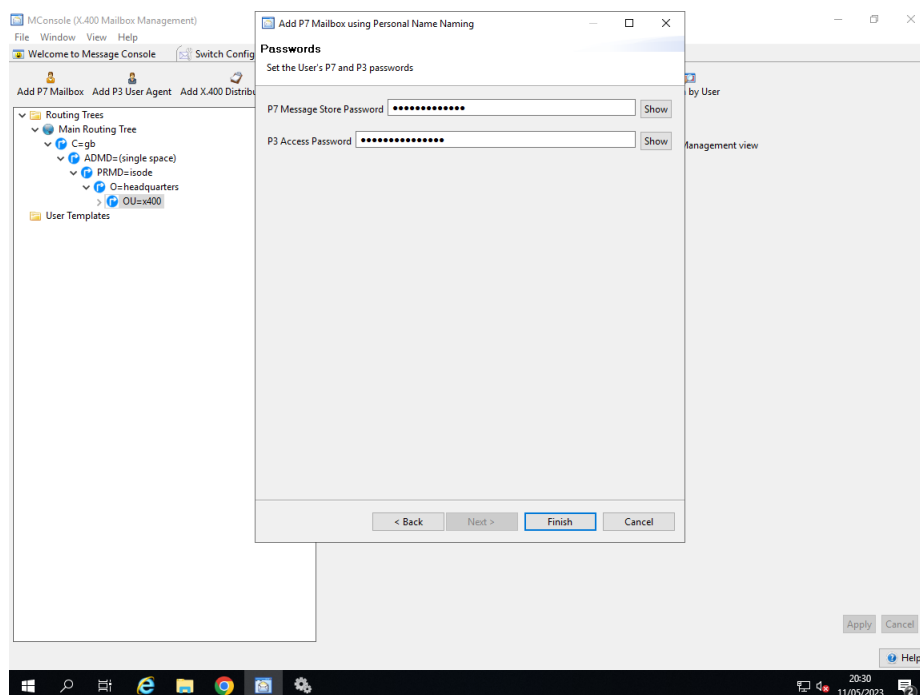


5. P7 and P3 Passwords

This screen is used to configure two passwords: the P7 Message Store Password and the P3 Access Password. The P7 Message Store password authenticates a user to the X.400 Message Store. The P3 Access password is used to bind to the MTA for message submission.

Initial values for these passwords will be randomly generated. These values can be displayed and edited by pressing **Show** on the wizard page. You will need the P7 Message Store Password to be shown so that the owner of the mailbox can log in. You will need to show the P3 Access Password if, for example, P3 client User Agents are to access the MTA. If just the X.400 Message Store is going to use the P3 logon, the random password can be retained.



**Figure 11.6. Configuring the P7 Message Store and P3 Access Passwords**

Once passwords have been set, the **Finish** button will become active; pressing this will create the new X.400 Message Store user.

## 11.2.2 P3 User Agent

Creation of a P3 User Agent follows the same series of steps as the creation of a X.400 Message Store User, except that a Mailbox Path and X.400 Message Store Password are not required.

## 11.2.3 X.400 Distribution List

Creation of an X.400 Distribution List follows the same steps to establish the O/R Address of the List as for a X.400 Message Store User.

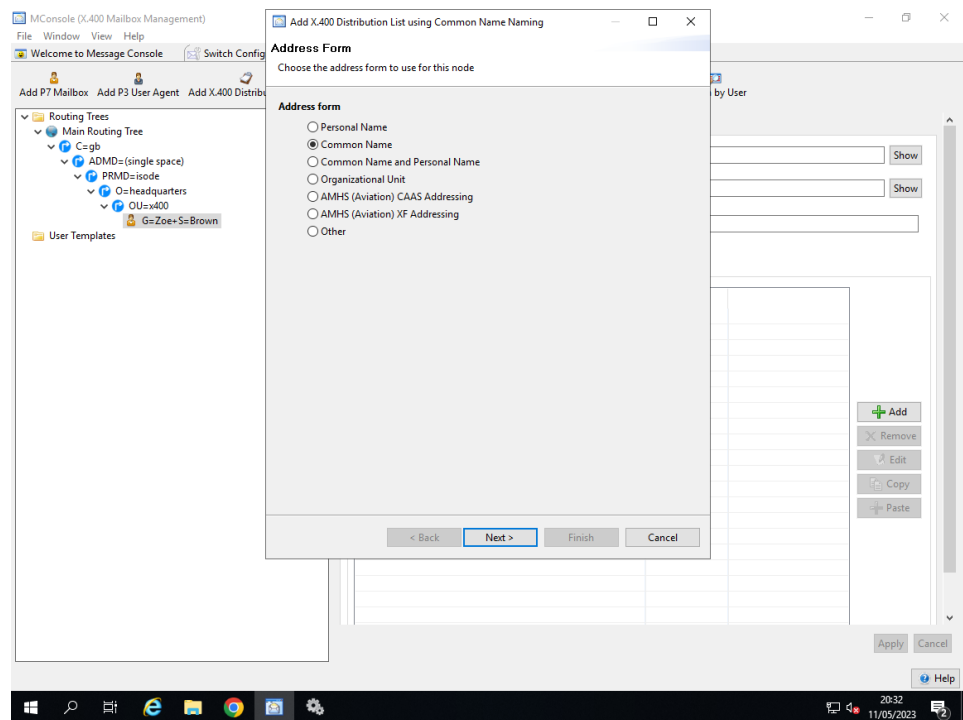
Firstly specify the Name Form of the List (in the same way as the X.400 P7 Mailbox) :

- **Personal Name:** Entry is named by Surname and optional Given Name, Initials and Generation Qualifier.
- **Common Name:** Entry is named by Common Name.
- **Organizational Unit:** Entry is named by Organizational Unit.
- **AMHS (Aviation) CAAS:** Entry is named by Common Name.
- **AMHS (Aviation) XF:** Entry is named by Common Name.

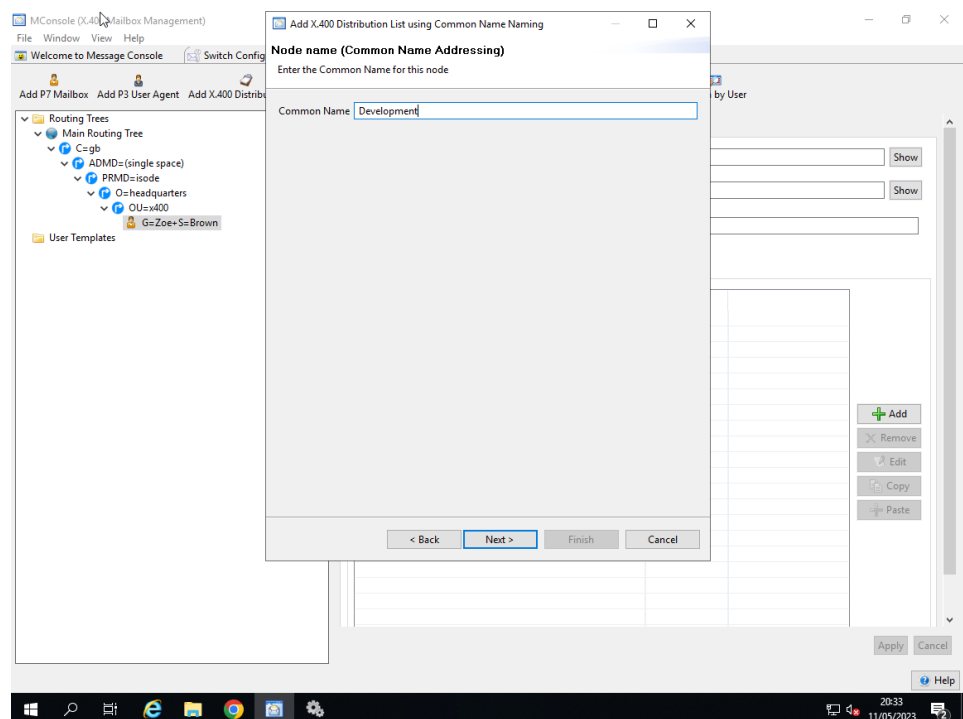
---

**Note:** If you choose to configure a CAAS address or XF address, selecting that choice invokes an intelligent editor which is aware of the address forms and supports intelligent editing of the Common Name and Organizational Unit respectively.

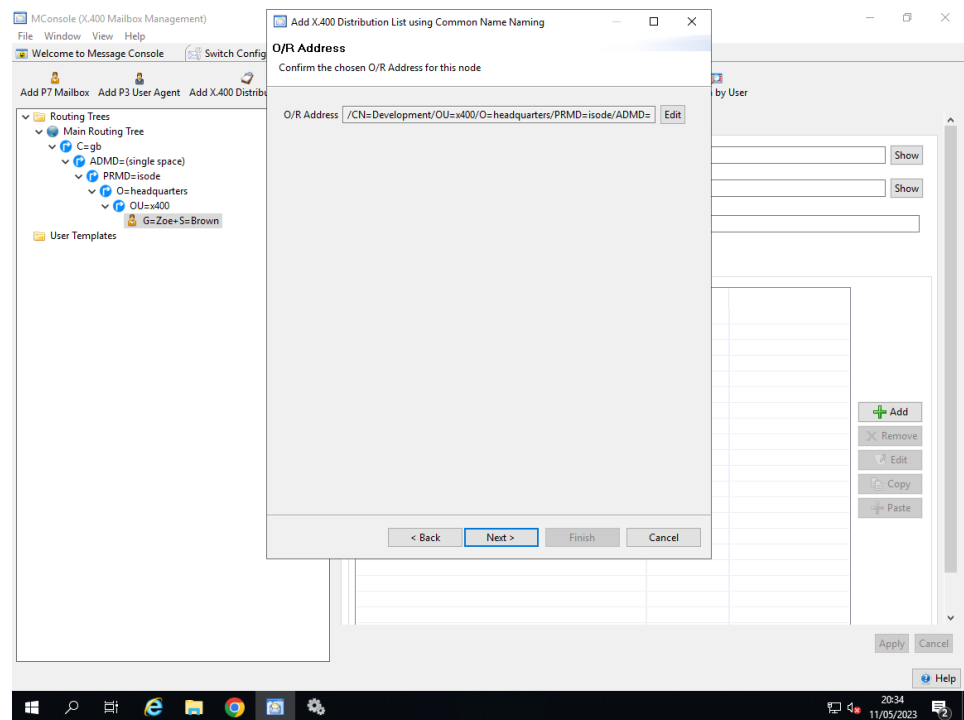
---

**Figure 11.7. Configuring X.400 List Name Form**

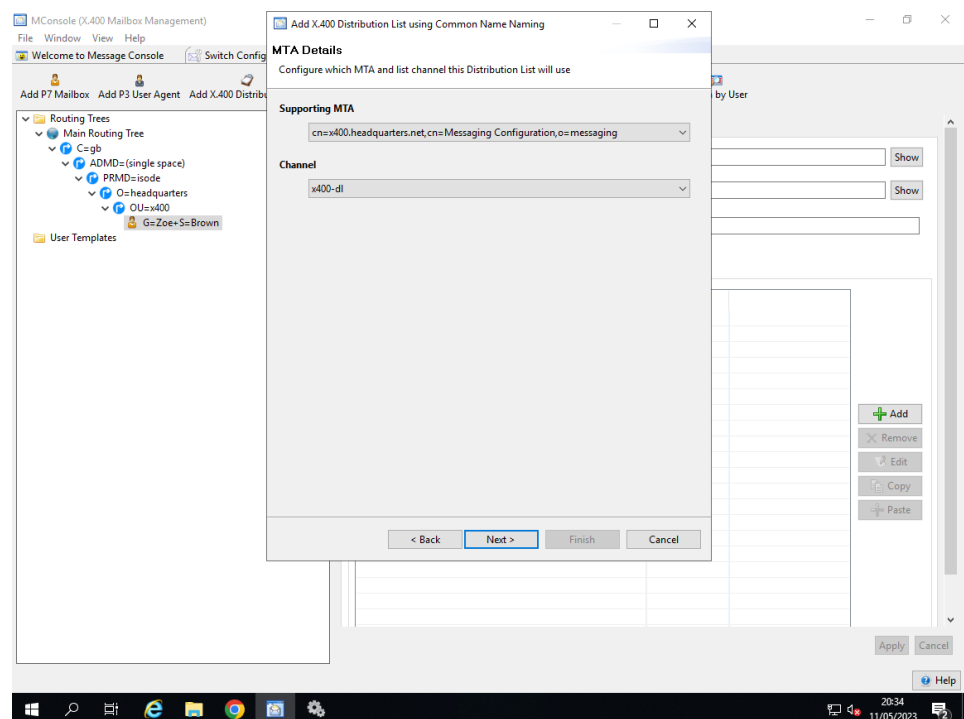
Then enter the value of the Personal Name, Common Name or Organizational Unit name (depending on the Name Form chosen):

**Figure 11.8. Configuring X.400 List Common Name**

Then confirm the Full O/R Address:

**Figure 11.9. Configuring X.400 List Address**

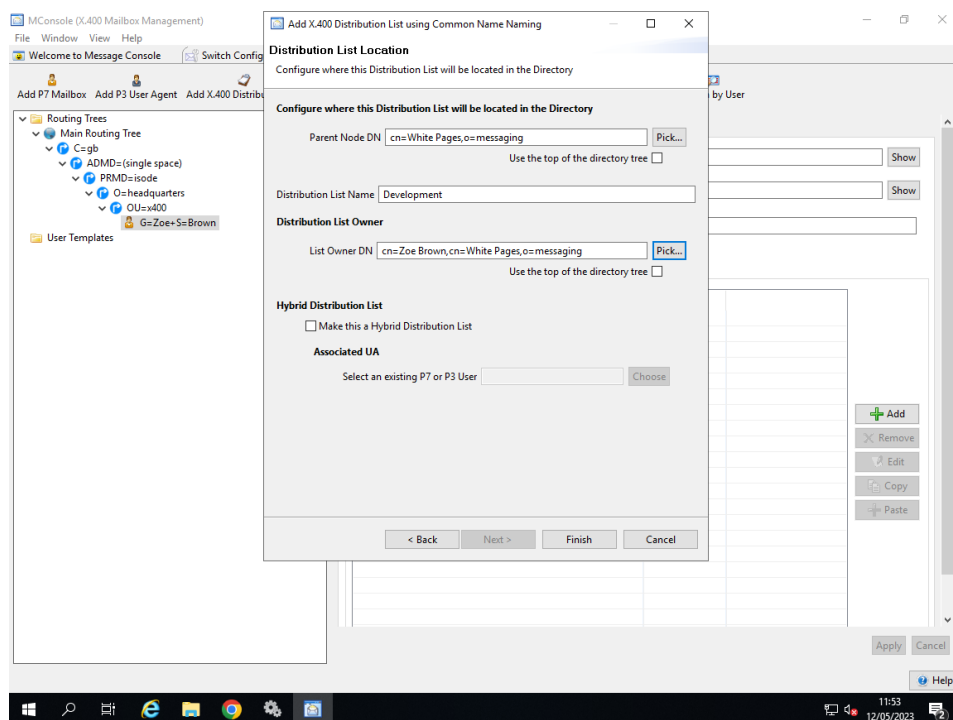
You then need to select the MTA and List channel which will process this Distribution List.

**Figure 11.10. Configuring X.400 List Supporting MTA and channel**

You can now configure the Distribution List Location

The location of the White Pages Entry in the Directory can be selected, as can the list name and Owner.

If you have chosen to configure an Aviation System, you can also configure whether a list uses a standard list channel or the hybridList channel.

**Figure 11.11. Configuring X.400 Distribution List Location**

The primary driver for this capability is Aviation (AMHS) to support configurations that emulate older AFTN configurations that have addresses which behave in this way.

When receiving messages, this address will act like a DL, with messages being delivered to multiple users. However, unlike a standard DL there is an associated “normal” UA with the same address that can send messages.

The core setup of this entity is as a DL, with members and management as a normal DL. There is also a special UA (address) associated with the DL that has a “private address”. This private address is a member of the DL and is used to deliver messages to the associated UA. Having this separate address ensures that there are no loops.

The M-Switch configuration associates BOTH the list address and the private address with the UA. The UA (which can be a standard P3 or P7 UA) will bind using the same address as the DL, and so will be able to send messages from the DL address. M-Switch (for P3) and M-Store X.400 (for P7) will make available messages sent to the private address (so essentially both list and private address are associated with the UA). This means that the UA can be configured as if it has the same address as the DL and will be able to send and receive messages.

For someone sending to the UA/DL address, it will just be “an address”, noting that positive and negative delivery reports will make it look like a DL (positive delivery reports will indicate “expanded”).

The UA associated with the DL will see all messages received as coming via the DL (there will always be DL expansion history). The “this-recipient-name” parameter will be set to the private address, so the UA will be able to detect the “special” behaviour. This is unlikely to have any effect on the UA. If it is desired to have the UA send IPNs (Inter Personal Notifications) to record message receipt, the UA will need to be adapted for this special UA/DL configuration. A standard UA will not send IPNs where a message has been delivered via a DL. The UA will need to recognize this case as special if it is desired to use IPNs.

## 11.2.4 X.400 Redirect

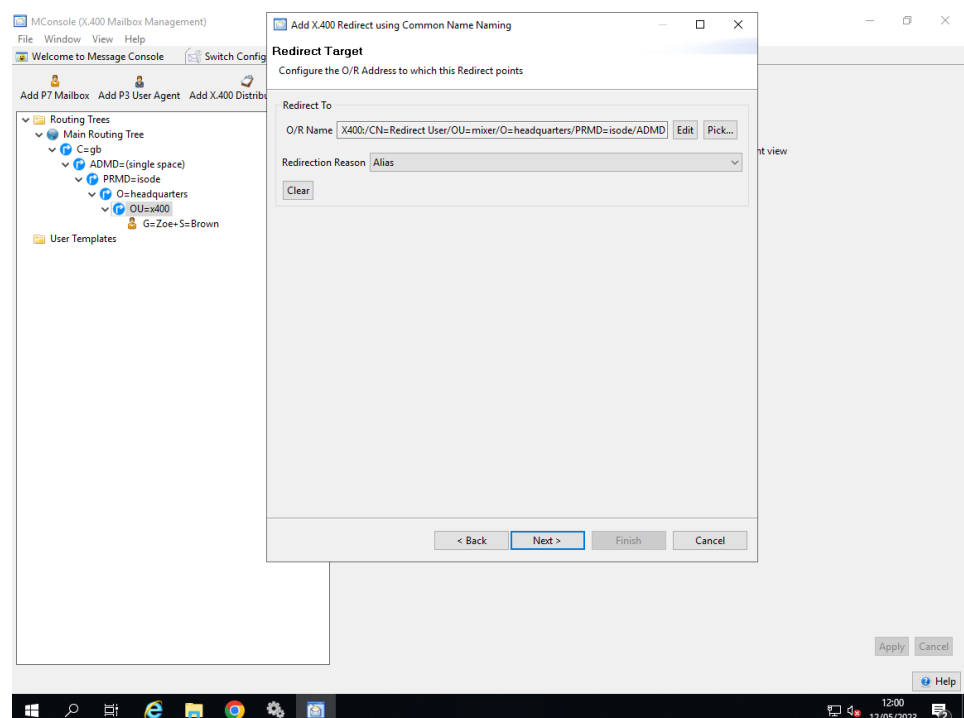
A redirect may be an X.400 address representing, for example, a particular job within an organization such as the postmaster for a particular domain. The postmaster address is a redirected address with a real user configured as the target. You can set up the redirect as either:

- a synonym or alias of an already configured user (for a definition see [Section 11.3.2.1.2, “Synonyms, aliases and redirects”](#)), or
- a redirect to a user which need not be configured in MConsole.

Creation of a Redirect entry follows the same steps for determining the O/R Address as for a X.400 Message Store User.

The only page of the Wizard allows the configuration of the Redirect target address and Redirection Reason, as shown below:

**Figure 11.12. Configuring X.400 Redirect**



## 11.2.5 Other User

This allows you to create a generic address entry which could for example be used for message delivery via the shell or FAPI channels.

## 11.2.6 Add File Transfer By Email User

This allows you to create a generic address entry which could be used for message delivery via the FTBE channel.

## 11.2.7 Add Using Template

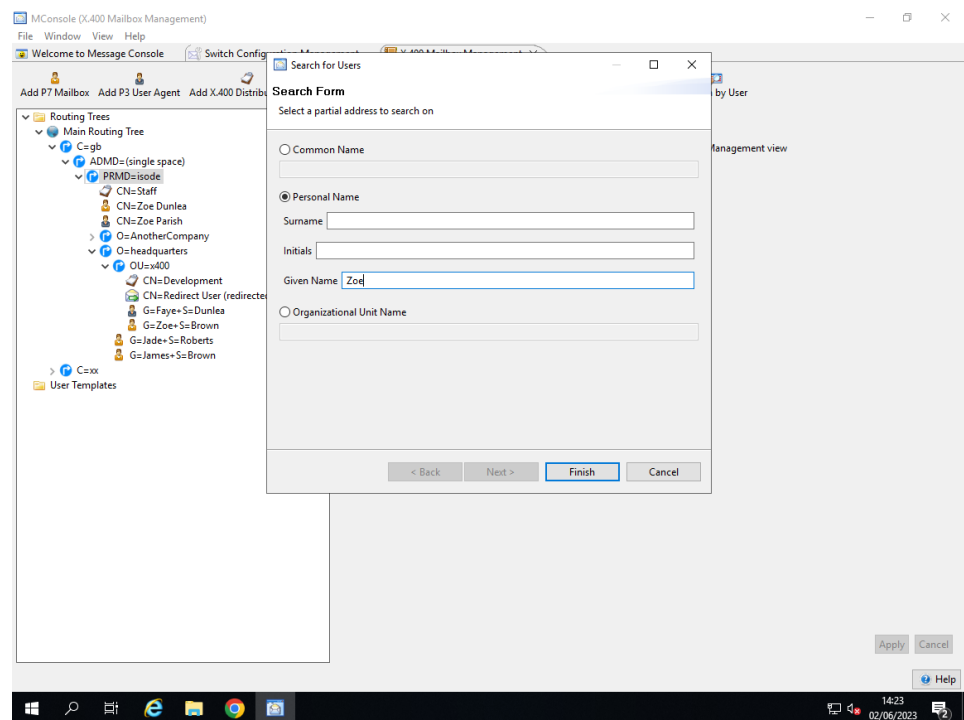
This allows you to create a generic address entry using a template defined in this view under the User Templates folder.

## 11.3 Updating X.400 messaging users

### 11.3.1 Searching for a messaging user

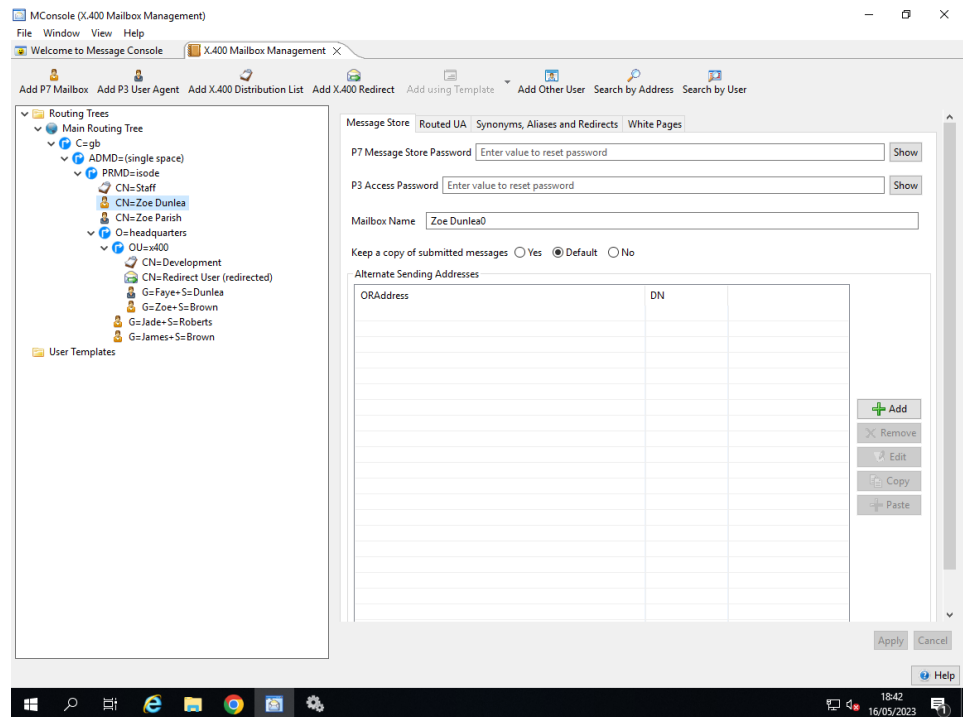
MConsole provides the ability to browse through the configured users. You can also search on full or partial name components from any point in the O/R Address Hierarchy, by selecting a node in the hierarchy and pressing the **Search** button. A search form will be displayed, as shown in [Figure 11.13, “The User Search form”](#). It is also possible to list all the X.400 users and perform partial searches, by using the **Search By Address** button.

**Figure 11.13. The User Search form**



Once you’ve entered one or more values into the search form you can press **Enter** or click on the **Finish** button to start the search. Wildcard values can be used if necessary.

Once the search has completed the set of matching mail users will be displayed on the left hand side of the main window. An example of a set of results matching the given name of Zoe are shown in [Figure 11.14, “Results for a Search on Given Name Zoe”](#).

**Figure 11.14. Results for a Search on Given Name Zoe**

Selecting a **user** node on the left-hand side of MConsole's main window sets up an editor appropriate to the node type on the right-hand side of the display, allowing you to edit the user's configuration.

The editor has a number of tabs displaying different sets of information depending on the node type; the **Routed UA** tab is common to all node types. To make changes simply edit any of the properties and click the **Apply** button; the **Cancel** button will discard any changes you may have made.

The tabs provided for specific user types are described below.

### 11.3.1.1 Listing and Searching for X.400 Users

An alternative way to search for X.400 users is to select the top-level **Routing Trees** folder on the left-hand side of the view, to display the specialized X.400 user browser on the right-hand side of the view. This will show an editor with three tabs: X.400 User List, X.400 User Search and X.400 Integrity Check.

#### 11.3.1.1.1 X.400 User List tab

The X.400 User List tab can display all the X.400 users in the selected messaging configuration. As the operation can be slow, the table is only loaded when the **Refresh Table** button is clicked.

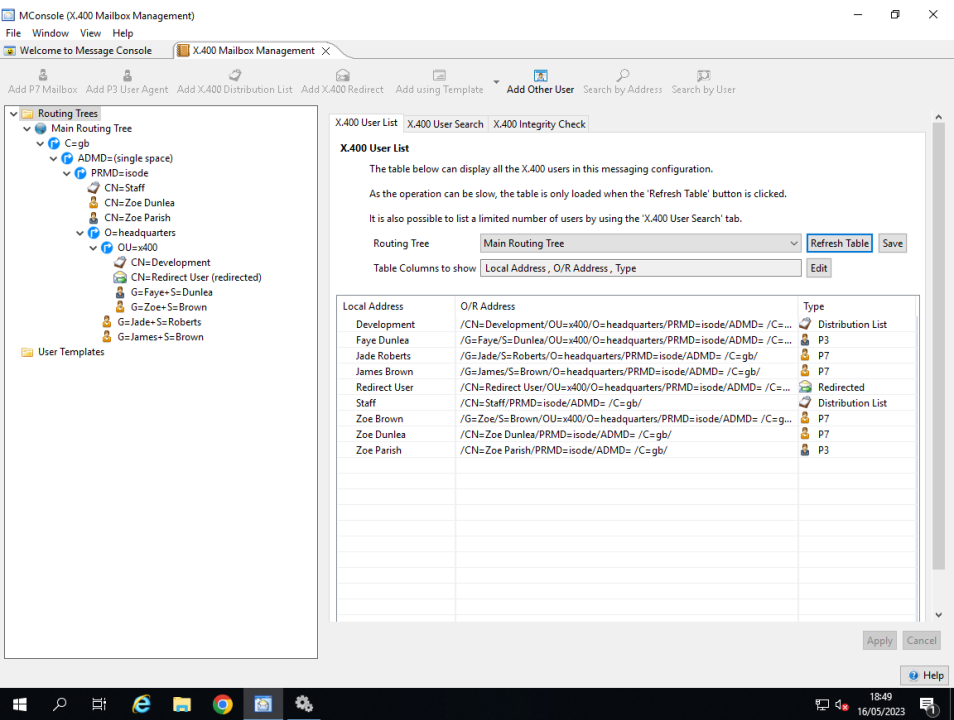
The attributes shown in the X.400 User List tab can be configured by clicking on the **Edit** button and selecting the attributes that you want to appear on the search. The current list of attributes include: Local Address, O/R Address, Type, Redirection Summary, Redirect Reason, Redirect Address Supporting MTA and Max Message Size. In order for the new attributes to be used, you need to click on the **Refresh Table** button.

To sort the result in the table, click on the column names. To edit one of the users, you can right-click on the user in the table, and select the **Edit User** menu option.

It is possible to save the result of the search by clicking on the **Save** button and then selecting a file. There are two formats supported: HTML and CSV (Comma-separated-values).

Bear in mind that this table shows all the X.400 users of the MHS configuration. To list a limited number of users, use the **X.400 User Search** tab.

Figure 11.15. List X.400 Mailboxes



11.3.1.1.2 X.400 User Search tab

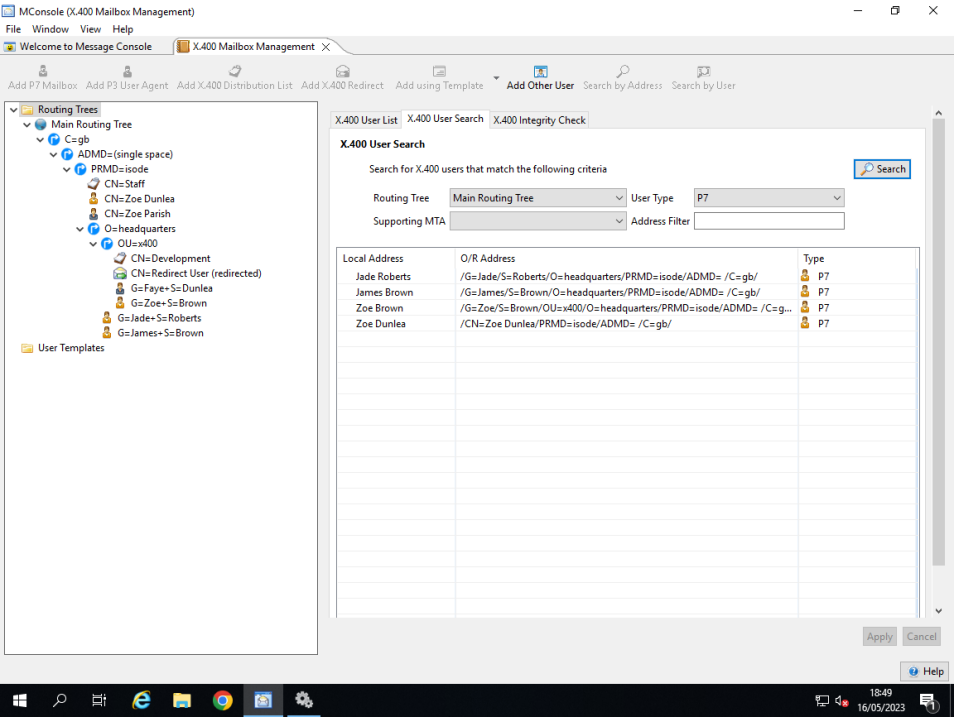
The X.400 User Search tab can narrow down the X.400 users displayed, by selecting some common filters. In case that there is more than one Routing Tree, the **Routing Tree** can be selected, otherwise all Routing Trees are searched. You can also filter by **User Type**: P3, P7, Distribution List, Hybrid List, Routed UA, Redirected FTBE User or FTBE Peer. In case that there is more than one MTA in the MHS configuration, it can be selected in the **Supporting MTAs** combo box. Finally, the search can also be requested to match a part of the O/R address by setting the required value in the **Address Filter**.

The search is performed when the **Search** button is clicked.

To sort the result in the table, click on the column names. To edit one of the users, you can right-click on the user in the table, and select the **Edit User** menu option.



Figure 11.16. Search X.400 Mailboxes



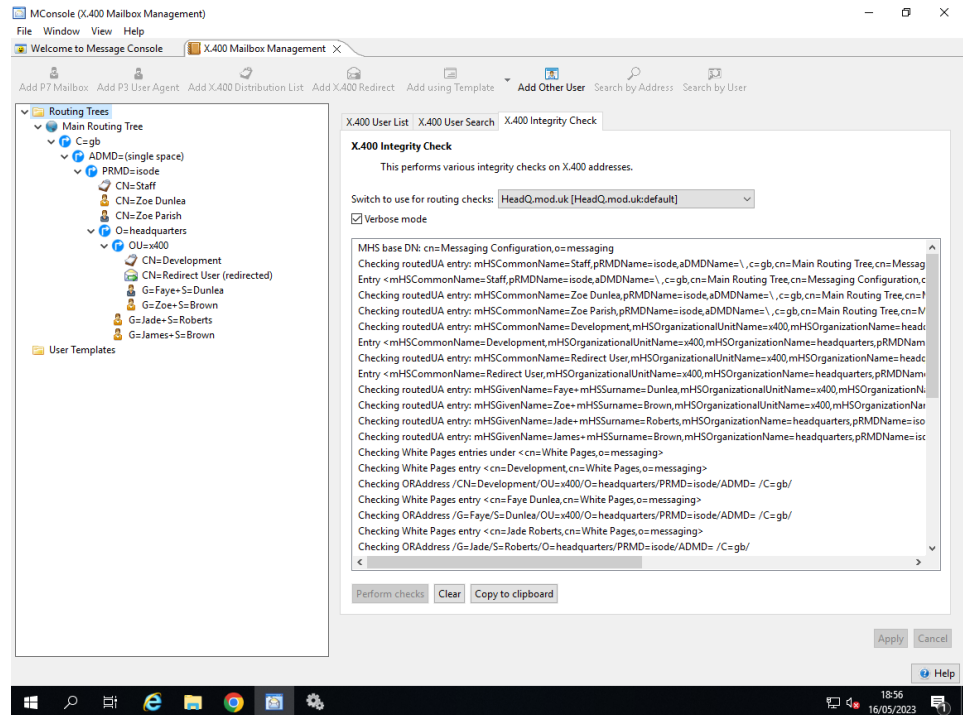
11.3.1.1.3 X.400 Integrity Check

This tab contains a GUI which allows the IntegrityCheck script to be easily called, and displays the output of the script using different colours depending on the level of importance of the log.

Please refer to the *M-Switch Advanced Administration Guide* for more information about the X.400 Integrity Check script and what options there are.

To perform the Integrity Check, click on the **Perform checks** button at the bottom of the window. If the **Verbose** check-button is selected, the output will include more verbose logging.

**Figure 11.17. X.400 Mailboxes Integrity Check**



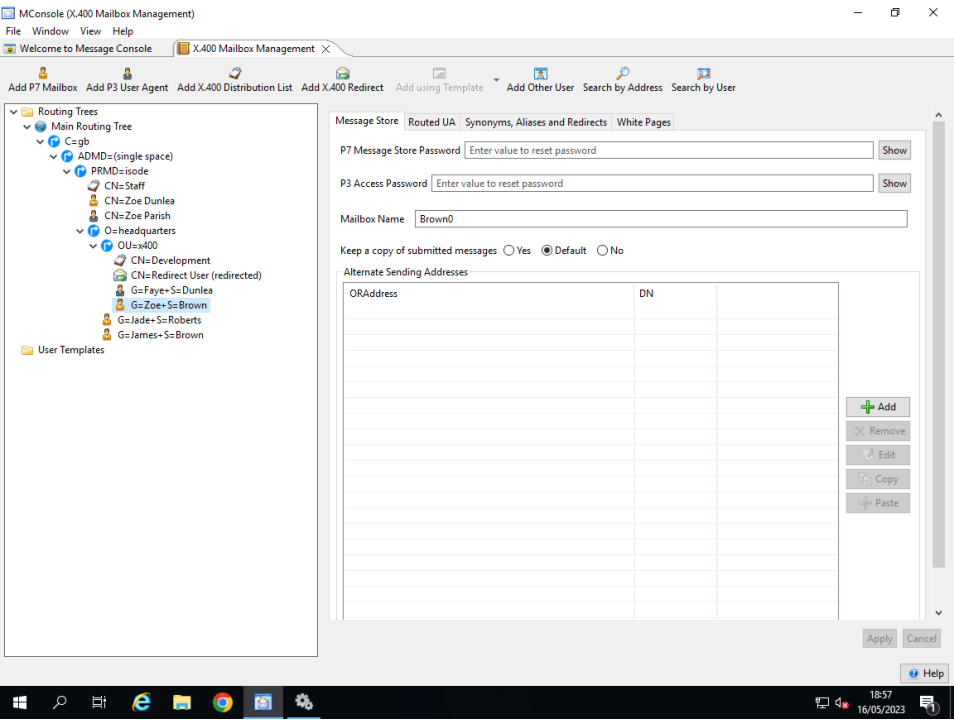
### 11.3.2 Editing Users

### 11.3.2.1 Editing P7 Message Store and P3 User Agent users

The first tab displayed for a X.400 Message Store or P3 User Agent user shows the P7 and MTA passwords, the user's mailbox name and a list of valid alternate sending addresses. For P3 User Agent users the P7 password and mailbox name are ignored.

**Note:** For security reasons, the Access Control for the Entry in the Directory prevents the Messaging Administrator from seeing User Passwords. The Passwords cannot be read, but can be reset.

Figure 11.18. X.400 Message Store User tab



The **Alternate Sending Addresses** field allows the configuration of additional permitted values of the Originator field in messages sent by this user. Normally the O/R Address with which a User Agent has bound to the X.400 Message Store must also be specified as the Originator address in any message sent by that user. By configuring additional addresses in a user’s **Alternate Sending Addresses** field, you permit that user to use that address as the Originator field of messages.

11.3.2.1.1 Routed UA

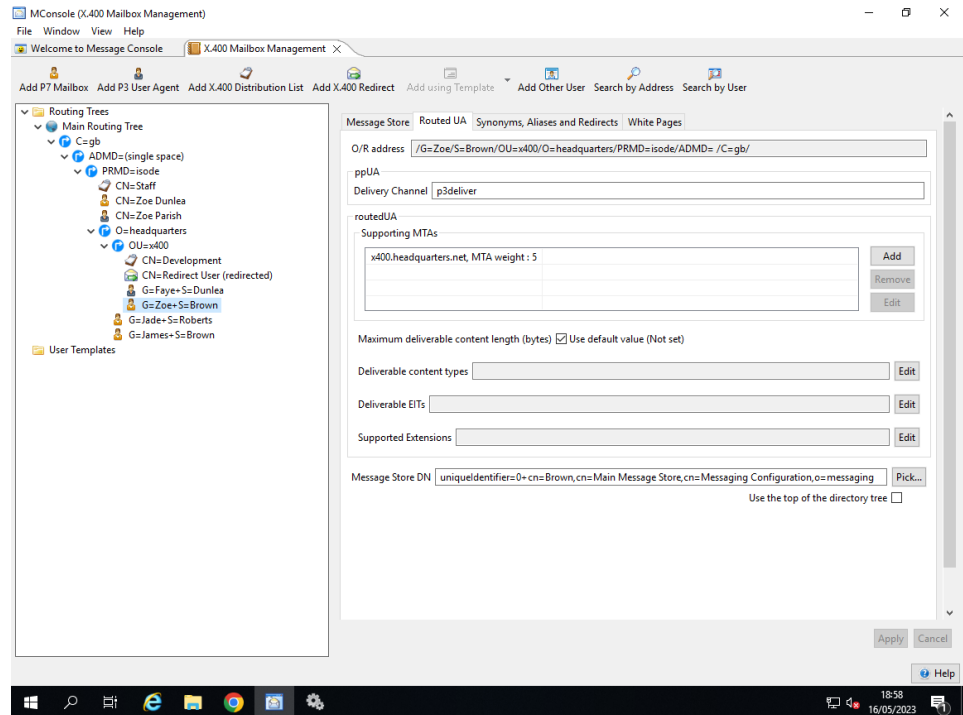
The **Routed UA** tab configures information related to the way in which messages are delivered to the user. The channel which performs delivery, the MTA which is responsible for the delivery and various restrictions on the size and content of the messages which the user will accept, can all be set. For P7 users, the Distinguished Name of the related X.400 Message Store is also displayed: this should not normally be modified.

---

**Note:** The O/R Address of the user is displayed here, but cannot be changed. To do this you need to delete the P7 User and add a new P7 User.

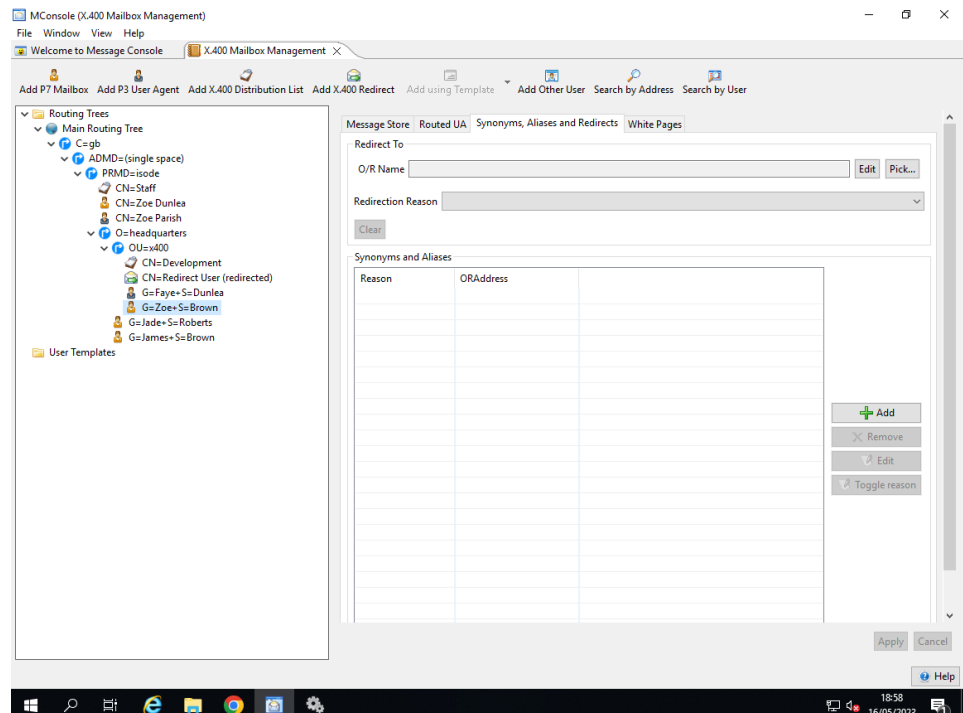
---

A typical view is shown below.

**Figure 11.19. Routed UA Tab for X.400 Message Store User**

### 11.3.2.1.2 Synonyms, aliases and redirects

Synonym configuration is accessed via the **Synonyms, Aliases and Redirects** tab.

**Figure 11.20. Synonyms, Aliases and Redirects Tab**

This tab allows you to configure three different types of synonym/alias. These are:

- **Redirect:** all messages for this user are redirected to another address.
- **Alias,** which means that only the message envelope information is changed and not the header.
- **Synonym,** which means that the message envelope and header are changed.

To add a new alias or synonym, click **Add**. This opens a wizard in which you can define the new alias or synonym's O/R Address. You can then use the **Toggle reason** button to change the alias into a synonym, or vice versa.

To set up a redirection, click **Edit** to define the target O/R Address. **Clear** removes the redirection again.

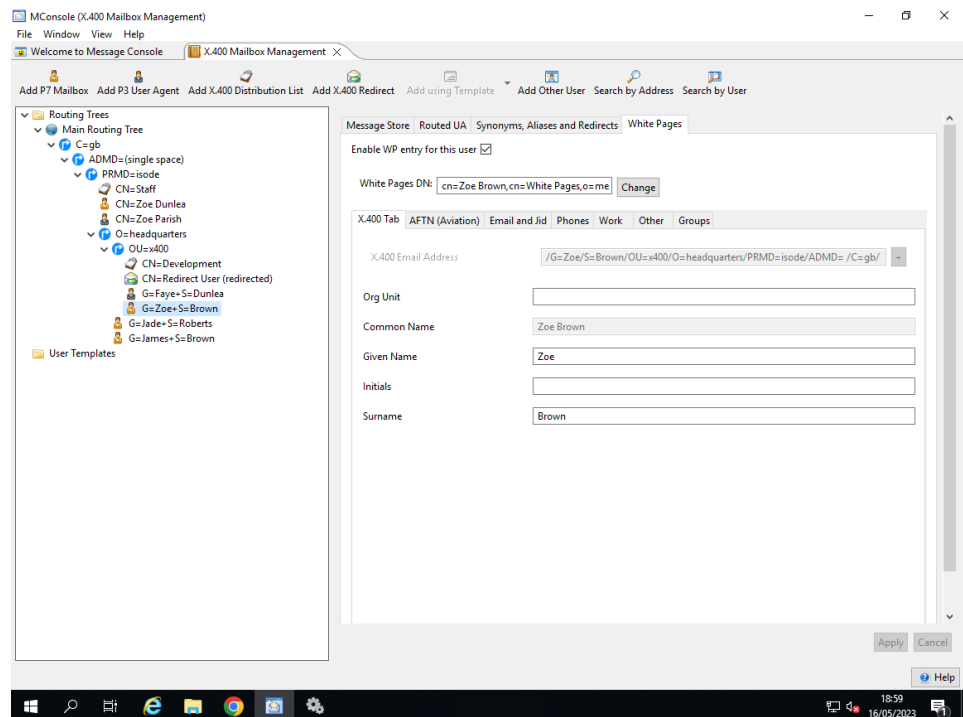
### 11.3.2.1 White Pages

The **White Pages** tab configures information about the White Pages Entry held for the P7 User. (This entry is unrelated to the other Directory Entries that are used to hold information used by M-Switch and M-store for this user).

The White Pages entry can be disabled. If enabled, the Name of the Entry and the values of the attributes to be held in the Entry can be managed from this tab. These are not used by M-Switch and M-Store.

A typical view is shown below.

**Figure 11.21. White Pages Tab for X.400 Message Store User**



### 11.3.2.2 Editing Distribution Lists

This is described in [Section 28.4, “Editing distribution list subscribers”](#).

### 11.3.2.3 Editing Redirections

A Redirection node has a **Redirection** tab which specifies the target O/R Address and reason, and allows them to be edited. The **Routed UA** tab for the Redirection does not contain any editable information.

## 11.3.3 Renaming users

MConsole does not support renaming of mail users (i.e. changing the mail address of a user) as a single function. To do this you must delete the existing mail user from the configuration and then create a new one.

## 11.3.4 Deleting

### 11.3.4.1 Deleting users

To delete a user, highlight the target user in the left hand window and select **Remove** from the right-mouse menu. This will completely remove the mail user from the messaging configuration in the Directory.

### 11.3.4.2 Deleting synonyms and aliases

To delete a synonym or alias highlight the corresponding address in the left hand window and select **Remove** from the right-mouse menu. This will remove the selected entry and also the pointer to it from the original mail user.

Aliases and synonyms can also be removed by modifying the synonyms value of the original mail user.

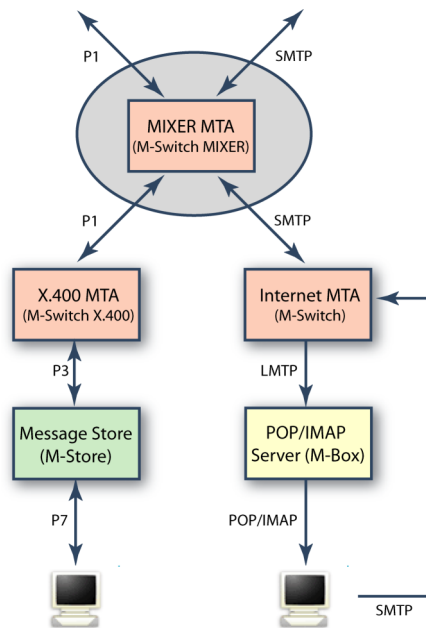
# Chapter 12 Configuring a MIXER Messaging System

This chapter gives instructions on how to set up an Internet/X.400 MIXER messaging configuration using MConsole.

The initial set up instructions are for a basic MIXER gateway configuration like the one illustrated in [Figure 12.1, “Example MIXER configuration”](#). For this configuration, we will use most of the default values set by MConsole. The latter part of the chapter describes how you can extend the configuration.

This chapter concentrates on setting up the MIXER MTA, indicated by the shaded area in the diagram.

**Figure 12.1. Example MIXER configuration**

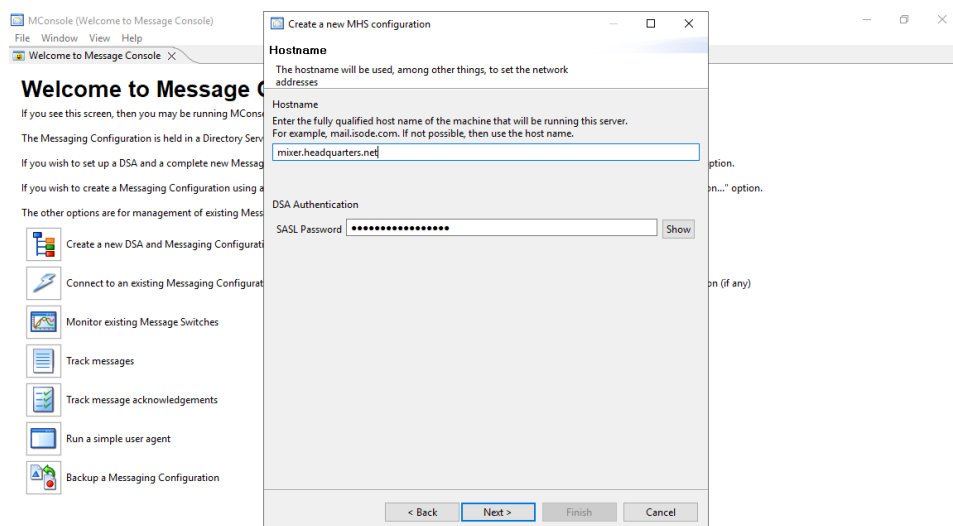


## 12.1 Creating a MIXER Configuration

[Section 4.4.2, “Creating a new Directory”](#) describes how to create a bind profile and DSA and how to specify where the new profile should be held in the Directory. You will now be presented with a series of wizard pages which help you to configure your MIXER system.

### 12.1.1 Configuring the Hostname

In the next wizard page, you are asked to enter the fully qualified hostname for the MTA you wish to create. The hostname will normally be in the values of MX records for the Email domain configured in the next screen.

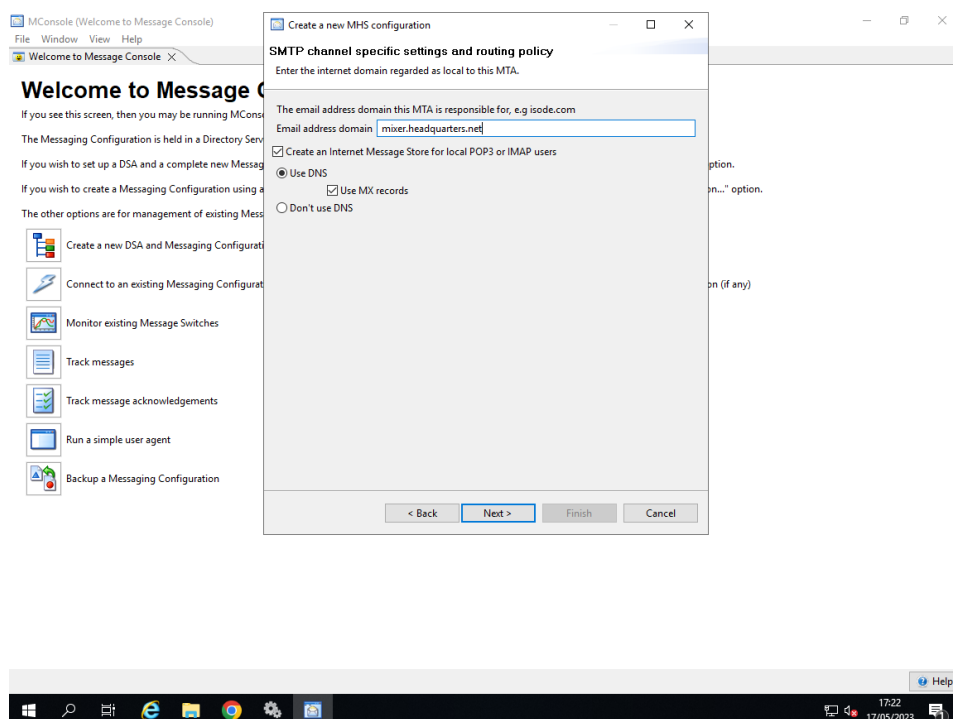
**Figure 12.2. Fully qualified hostname screen**

## 12.1.2

### Configuring the Internet Mail Domain and use of DNS

In the next wizard page, you are asked to enter the Internet Mail domain name to be regarded as local to this MTA.

You can also choose whether or not an Internet POP/IMAP Message Store is to be created in this configuration. This will allow you to configure local POP/IMAP Mailboxes whose users can have messages delivered by the LMTP channel and also submit messages using SMTP.

**Figure 12.3. Domain Name and DNS configuration**

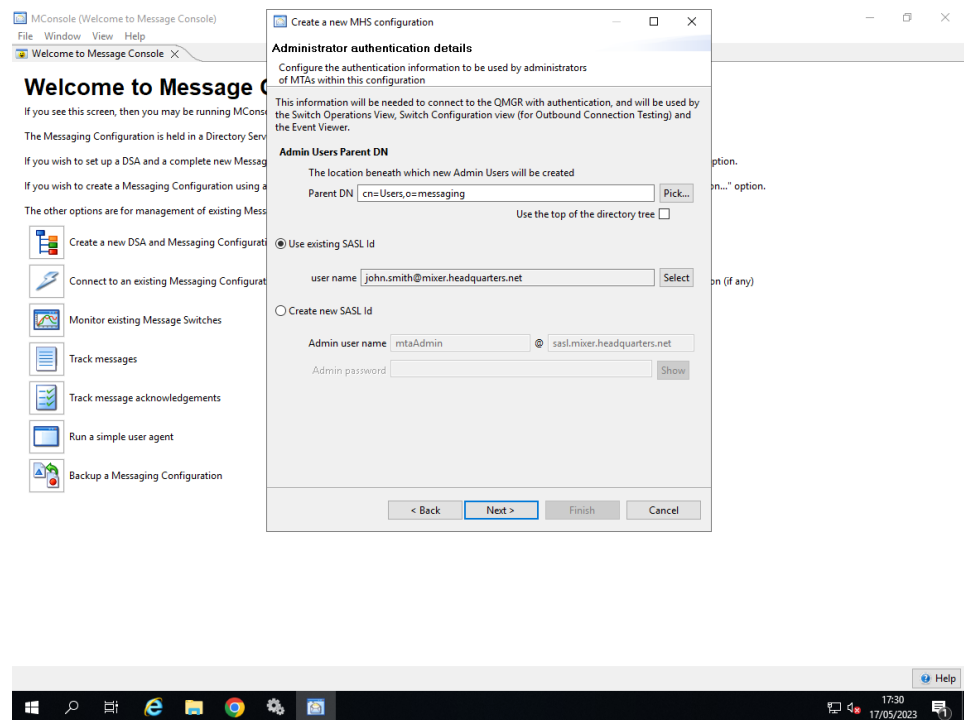


## 12.1.3 Configuring the Administrator Authentication

In the next wizard page, you are asked to enter the SASL ID of the User which will be used by MConsole to connect to the qmgr. This will be used by the **Switch Operations** view, the **Switch Configuration Management** view and the **Event Viewer** view.

The value used is based on the name supplied at the start of the creation of this messaging configuration. You will be able to edit this user in MConsole in the **Authenticated Entities Management** view.

**Figure 12.4. Administrator Authentication screen**

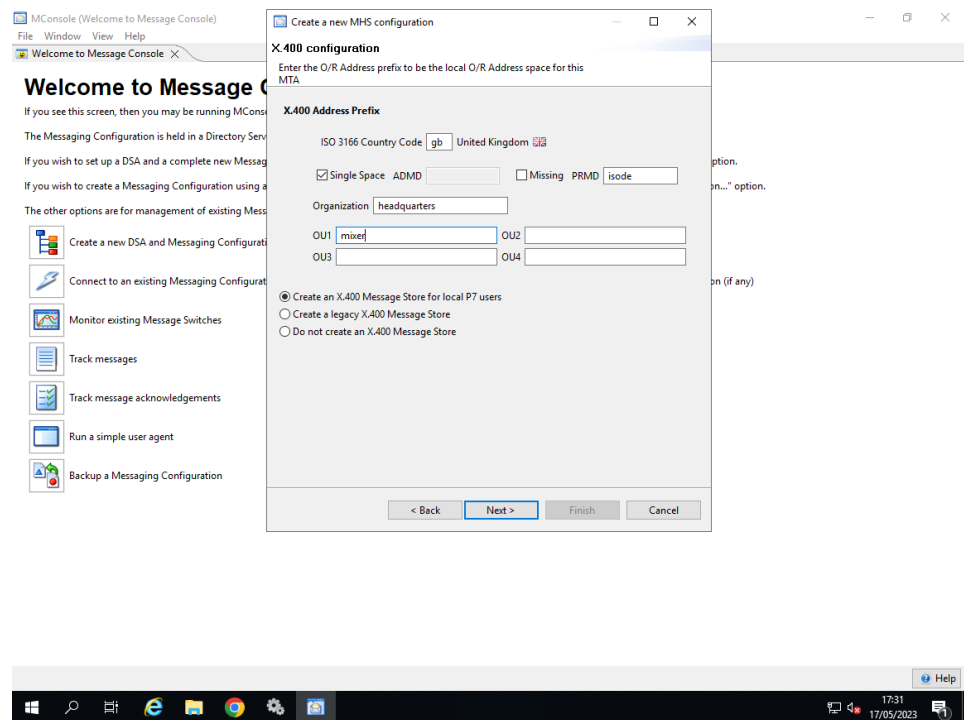


## 12.1.4 Configuring the X.400 O/R Address

In the next wizard page, you are asked to enter the O/R Address regarded as local to the MTA being created.

You can also choose whether or not an X.400 P7 Message Store is to be created in this configuration. This will allow you to configure local P7 Mailboxes which can connect to the Message Store using P7 to submit and retrieve X.400 messages.

If you choose not to create a Message Store, you will still be able to submit and retrieve X.400 messages using P3. Or alternatively, you may wish to use the MTA so X.400 messages can only be handled by either relaying using P1 or converting to SMTP as a MIXER gateway.

**Figure 12.5. X.400 O/R Address screen**

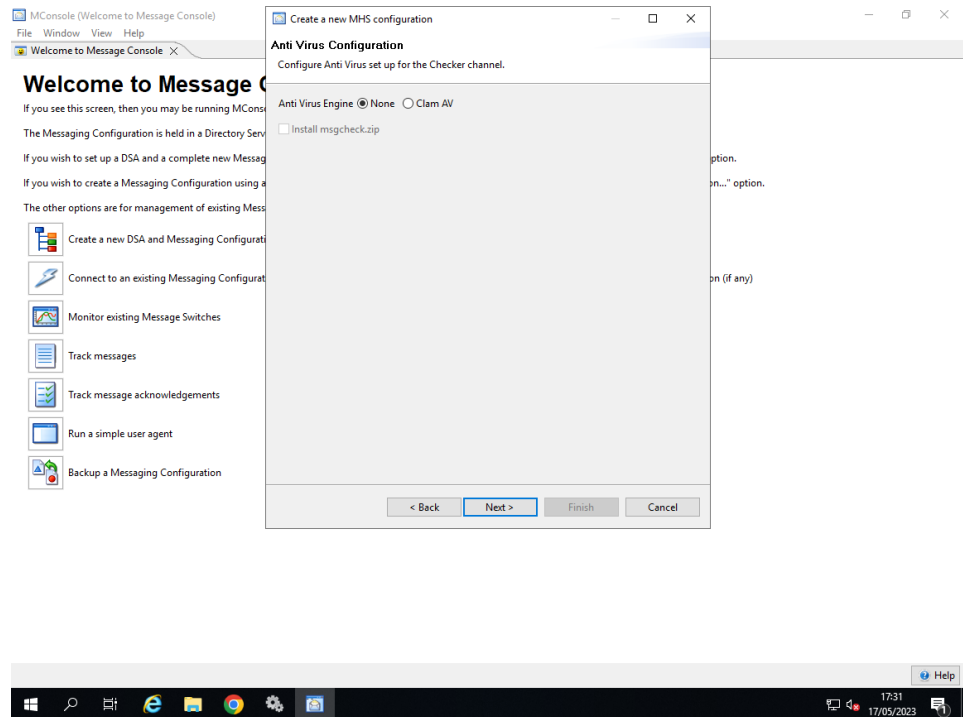
## 12.1.5 Anti Virus Configuration

M-Switch can be configured to perform content checking in which messages containing Viruses or are considered to be spam are detected and action suitable action taken to counter these abuses.

See [Section 40.1, “Checking message content”](#) for a full description of Content Checking.

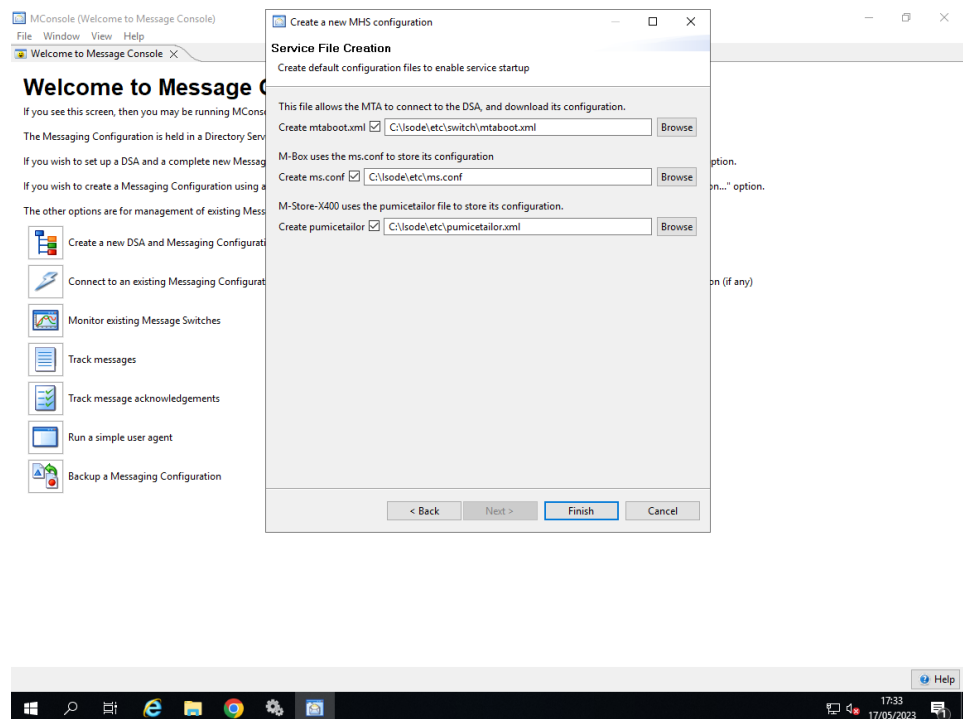
This wizard page allows you configure Anti Virus (which you can also configure later, if you wish). You can choose to have no Anti Virus Engine, Clam AV, or Sophos. If an engine is selected you will be informed of the platforms on which it is supported and you can choose whether or not to install msgcheck.zip.

You should only install msgcheck.zip if you have the rights to do so and if MConsole is running on the same machine as the configuration you are creating

**Figure 12.6. Anti Virus Configuration**

## 12.1.6 Install Service Files

In the next wizard page, you are asked to specify where the configuration file for each of the MTA, X.400 Message Store and POP/IMAP Message Store are to be copied. The defaults assume you are going to run these services on the local system. If not, you will need to either unset the creation checkbox, or specify different directories and copy the files manually to the system which is going to run these services.

**Figure 12.7. MTA Service Files**

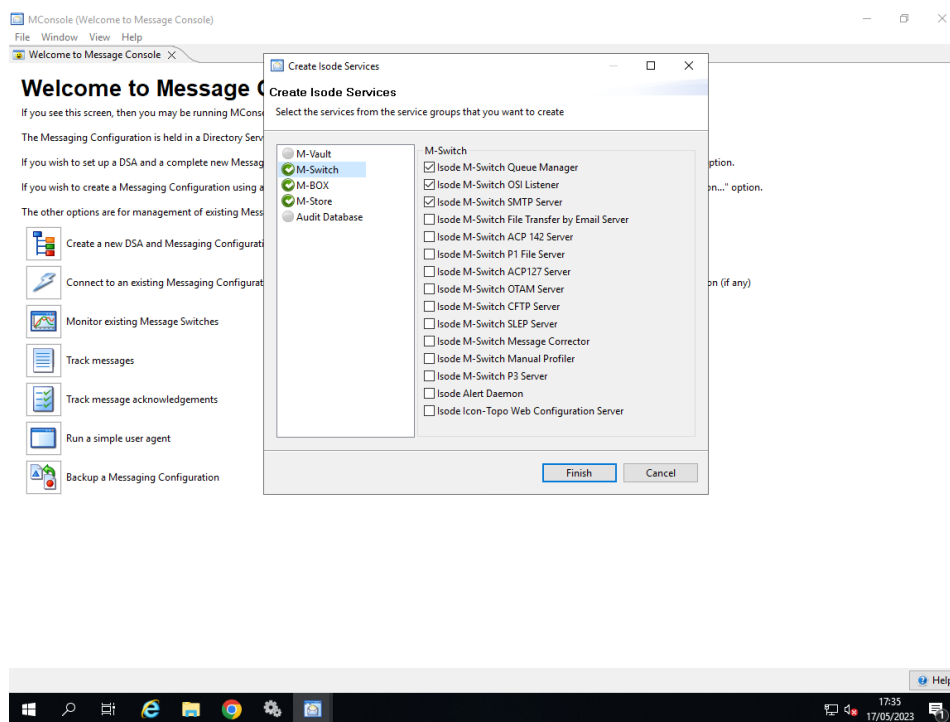
## 12.1.7 Create Windows Services

You can now choose the services you wish to have created. As this is a MIXER MTA you will normally need all the M-Switch Services. If you have created an X.400 P7 Message Store you will need to create the M-Store X.400 Services.

Similarly, if you have created a POP/IMAP Message Store you will need the following Services as a minimum: mseventd, imapd, lmtpd, pop3d, sieve, msadmind.

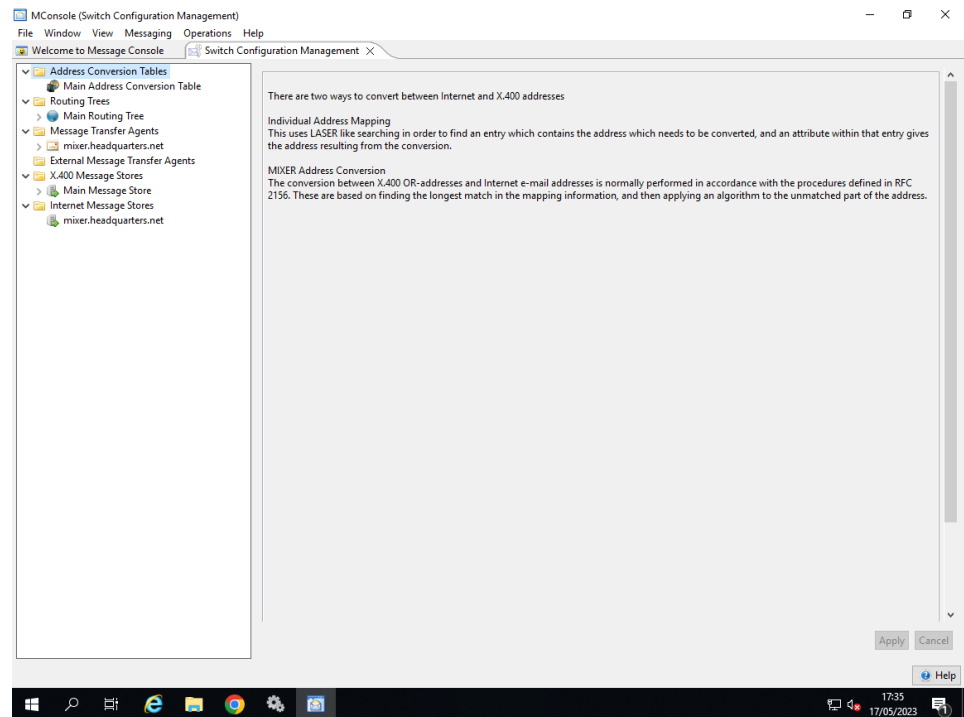
See *M-Box Administration Guide* for a full description of M-Box and its services.

**Figure 12.8. MTA Services Creation**



## 12.1.8 Completed MIXER MTA

After clicking on **Finish**, the configuration of the completed MIXER configuration is created and displayed ready to view and edit.

**Figure 12.9. X.400 Configuration complete screen**

What has been configured is

- an MTA which has X.400 and Internet messaging support
- an X.400 Message Store
- an POP/IMAP Message Store

No support for MIXER address mapping has been added yet. In the next section we'll see how this is added.

---

## 12.2 Editing the MIXER Configuration

The tailoring information for the MTA has been created and placed in the Directory. To see all the MTA tailoring properties, select the MTA within the **Switch Configuration Management** view. This displays the general properties about the MTA in the right hand side window.

You should also expand the folders below the MTA, which will show its channels, tables, directory profiles and logging.

### 12.2.1 MTA properties

To display the properties set for the MTA, click on the MTA's node. The creation wizard will have set suitable values for a basic system, so you should not need to alter any of the MTA values at this point. However, should you want to tailor values later on, a description of the fields can be found in the *M-Switch Advanced Administration Guide*.

## 12.2.2 Address Routing

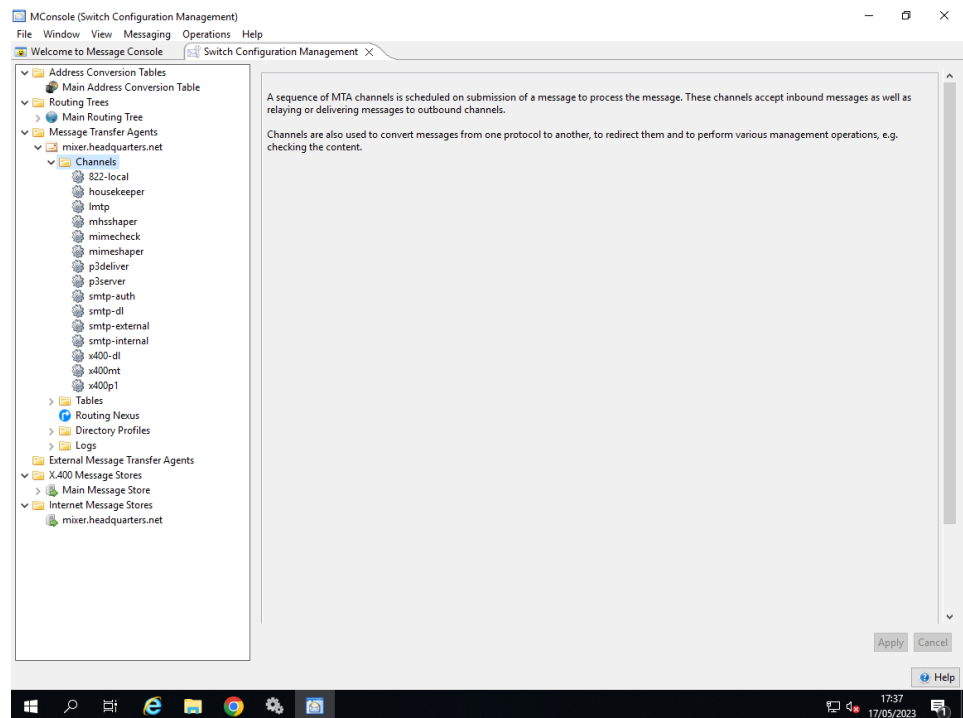
A MIXER MTA needs to be able to route Internet and X.400 addresses. Routing is discussed in [Chapter 15, Routing](#). Address conversion is discussed later in this chapter, see [Section 12.3.1, “MIXER address conversion”](#).

## 12.2.3 Channels

Suitable channels have also been created, as shown in [Figure 12.10, “MIXER MTA Channels”](#). The channel properties can be displayed by clicking on the channel under the MTA. Again, you may want to tailor channels and their properties later. The various property fields are described later on in this chapter and in the *M-Switch Advanced Administration Guide*.

The following gives a summary of the function of the channels created in the basic MIXER MTA. Cross references do not need to be followed when creating your initial system, but you may find them useful later when you want to extend the functionality of your configuration:

**Figure 12.10. MIXER MTA Channels**



### 822-local

Normally messages delivered to local users are delivered using the LMTP channel into M-Box (or another POP/IMAP Message Store). However the 822-local channel can be used to deliver internet messages to users configured as local to this MTA. This uses the well know mbox format. Adding and tailoring local delivery facilities is described in the *M-Switch Advanced Administration Guide*.

### ftbe

(optional) This is the channel program which is used to transfer emails to and from the filesystem. Typically it's used in conjunction with Sodium Sync to allow directory replication over email. Configuration information for the fteserver is given within the *M-Switch Advanced Administration Guide*.

### housekeeper

This is a special housekeeping channel used by the MTA to perform tasks such as deleting message when they have been processed, or generating reports. Further

information on this channel can be found in the chapter called ‘MTA Tailoring’ in the *M-Switch Advanced Administration Guide*.

### **lmtp**

This channel enables the integration of the M-Switch with the POP/IMAP Message Store.

### **mhsshaper**

This is used for conversion between MIME and X.400. For simple configurations this will work as created. For details see the *M-Switch Advanced Administration Guide*.

### **mimecheck**

This channel is a virus and spam checker channel.

### **mimeshaper**

This is used for conversion between X.400 and MIME. For simple configurations this will work as created. For details see the *M-Switch Advanced Administration Guide*.

### **p1file**

(optional) This is a channel similar to x400p1, but works by reading/writing P1 APDUs from/to filestore

### **p3deliver**

This channel is used to deliver messages into the X.400 Message Store.

### **p3server**

This channel is used by a P3 User Agent to submit and accept delivery of messages to/from the MTA, and also to accept submissions from an X.400 Message Store.

### **smtp**

This is the main protocol channel for handling messages coming into the MTA and transferring messages out of the MTA. Four SMTP channels are created: smtp-auth, smtp-dl, smtp-internal, and smtp-external.

- **smtp-auth** is used for SMTP transfers, where the connection is received on port 587 (the SMTP submission port). The channel is configured to mandate Authentication. The Authrozation rules do not block this channel.
- **smtp-dl** is used for handling the resolution of distribution lists using file-based lists. Configuring distribution lists is described in the chapter called ‘Handling Messages Locally’ in the *M-Switch Advanced Administration Guide*.
- **smtp-external** is used for all other SMTP transfers.

Inbound SMTP protocol channels need to be started explicitly. The various tailoring options available for the channel are described in the *M-Switch Advanced Administration Guide*.

- **smtp-internal** is used for local SMTP transfers on port 25, where the source/destination is listed in the localhosts group in the MTA's authorization. This is used where the connecting MTA is in some sense trusted. So (for example) checks for spam can be disabled.

### **x400-dl**

This channel is for handling the resolution of distribution lists. Configuring for distribution lists is described in [Chapter 28, Directory Based X.400 Conformant Distribution Lists](#).

### **x400mt**

if you have selected to create a Gateway channel, this is the channel used for transferring (both in and out) messages from a Gateway application built using the Isode X400 Gateway APIs.

### **x400p1**

is the main protocol channel for transferring messages from one X.400 MTA to another. The x400p1 channel can operate in different modes. The various tailoring options available for the channel are described in the *M-Switch Advanced Administration Guide*.

## 12.2.4 Tables

A number of tables can be displayed under the **Tables** folder. Tables are required for setting up certain facilities such as distribution list handling. If you are using Directory-based routing, you do not need tables for most facilities. Configuring tables, instead of the Directory, to hold message routing information is described in the *M-Switch Advanced Administration Guide*.

## 12.2.5 Logging

The three logging variables created by the MTA creation wizard are displayed under the **Logs** folder. Their properties can be displayed and edited in the same way as other tailoring entities. The logging tailoring set here is adequate for your basic system. [Section 34.2, “Configuring logging”](#) describes how you can tailor the level of logging required for individual programs, channels and protocol layers. You can also specify that the logging be sent to separate files.

## 12.2.6 Message Stores

Expand the **X.400 Message Stores** folder and note that the **Main Message Store** has been created.

Expand the **Internet Message Stores** folder and note that an Internet Message Store has been created.

If you have created a Message Store after going through the wizard to create your messaging configuration then it will have whatever name you gave it when creating.

## 12.2.7 User Configuration

How to create and manage user accounts is described in [Chapter 7, \*Managing Internet Messaging Users\*](#) and [Chapter 11, \*Managing X.400 Messaging Users\*](#).

---

# 12.3 Tailoring the basic MIXER configuration

Many options for extending the basic MIXER configuration have already been described. See [Section 5.2, “Editing your configuration”](#) and [Section 16.1, “X.400 routing”](#).

In addition, there is a lot of background information on MTA tailoring and channel configuration in the *M-Switch Advanced Administration Guide*. However, when you use MConsole to set up your MIXER configuration, the MTA is given default properties suitable in an environment where conversion between different protocols is likely to be necessary and the channels required for conversion are also set up with reasonable defaults.

The MConsole wizard has created a basic MIXER configuration, but there are a number of actions needed for the MTA to be able to act as a MIXER Gateway.

One task which you do need to carry out is to add address mappings for address conversion. How to do this is described in [Section 12.3.1, “MIXER address conversion”](#). You will also need to enable X.400 users to send messages to Internet users via the MIXER gateway already created. An example of how to do this is given in [Section 12.3.4, “MIXER routing”](#).



### 12.3.1 MIXER address conversion

Conversion of Internet address to X.400 and vice versa is described in RFC2164. M-Switch supports MIXER mapping in three ways:

- algorithmic mappings
- per-user mappings
- encapsulation

You can choose to configure algorithmic or per-user mappings in order to map cleanly between the two address forms. If conversion using algorithmic or per-user mappings fails, fallback to encapsulation always succeeds, but is ugly.

Encapsulation of an Internet address in an O/R Address looks like this (lines wrapped for ease of reading):

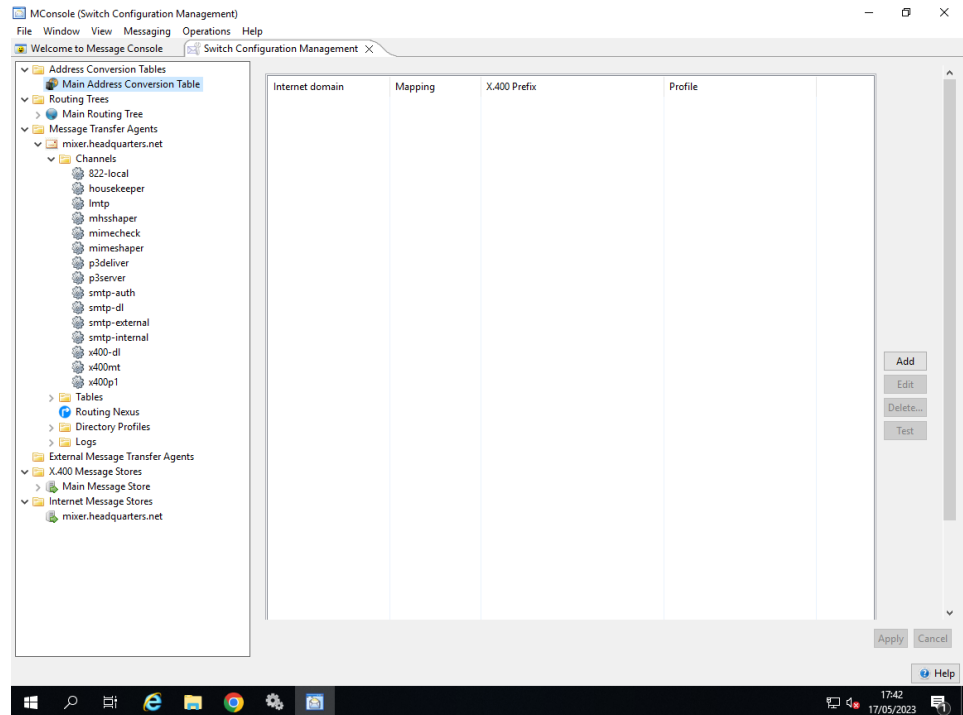
```
C:\Program Files\Isode\bin>ckadr postmaster@headquarters.net
postmaster@headquarters.net -> (rfc822)
postmaster@headquarters.net
postmaster@headquarters.net -> (x400)
/RFC-822=postmaster(a)headquarters.net/OU=internet
/O=headquarters/PRMD=Isode/ADMD= /C=gb/
```

Encapsulation of an O/R Address in an Internet address looks like this (lines wrapped for ease of reading):

```
C:\Program Files\Isode\bin>ckadr -x
"/G=P7/S=User1/OU=x400/O=headquarters/PRMD=Isode/ADMD= /C=gb/"
/G=P7/S=User1/OU=x400/O=headquarters/PRMD=Isode/ADMD= /C=gb/ ->
(x400) /G=P7/S=User1/OU=x400/O=headquarters/PRMD=Isode
/ADMD= /C=gb/
/G=P7/S=User1/OU=x400/O=headquarters/PRMD=Isode
/ADMD= /C=gb/ ->
(rfc822) /G=P7/S=User1/OU=x400/O=headquarters/PRMD=Isode
/ADMD= /C=gb/
@x400.headquarters.net
```

### 12.3.2 Configuring address conversion in the Directory

An Address Conversion Tree has been set up by the wizard. This is presented as a table containing a set of algorithmic and per user mappings. However, the MIXER MTA setup wizard does not configure any conversions between X.400 and Internet addresses. You need to add these after the wizard completes.

**Figure 12.11. Address Conversion Table with No Mappings Configured**

Clicking on **Add** allows an address mapping to be added. There are two sorts of mappings that can be configured:

- algorithmic mappings
- per-user mappings

Algorithmic mappings allow you to specify an internet domain and the O/R Address prefix to which it maps. It also allows a mapping the other way from an O/R Address Prefix to a domain to be specified. This is described in RFC2164. Unmapped portions of the domain and O/R Address to each other are carried out algorithmically.

Similarly the Personal Name or Common Name O/R Address attributes are mapped the local part of the Internet address and vice versa algorithmically. RFC 2156 describes the process which converts the local part to or from the X.400 Personal Name. An Isode extension enables the conversion to or from the X.400 Common Name. This choice is configured by a check box for the algorithmic mapping.

Per-user mappings use a search of the Directory to find an entry with the source address. An attribute in the entry found is used as the converted address. It is possible to use a partial address to control where the search is performed.

### 12.3.2.1 Configuring Algorithmic Mappings

In the diagram a mapping from `/OU=internet/O=headquarters/PRMD=Isode/ADMD=/C=GB/` to `headquarters.net` is configured, as well as a mapping from `x400.headquarters.net` to `/OU=x400/O=headquarters/PRMD=Isode/ADMD=/C=GB/`.

To configure address conversion select the **Main Address Conversion Table** and click on **Add**. Specify the X.400 O/R Address and Internet address which you wish to configure to be mapped. Note that you can configure part of a domain or O/R Address. See [Section 12.3.3, “Address conversion when attributes are missing”](#). If you wish to map the Internet address local part with the X.400 Common Name, check the box.

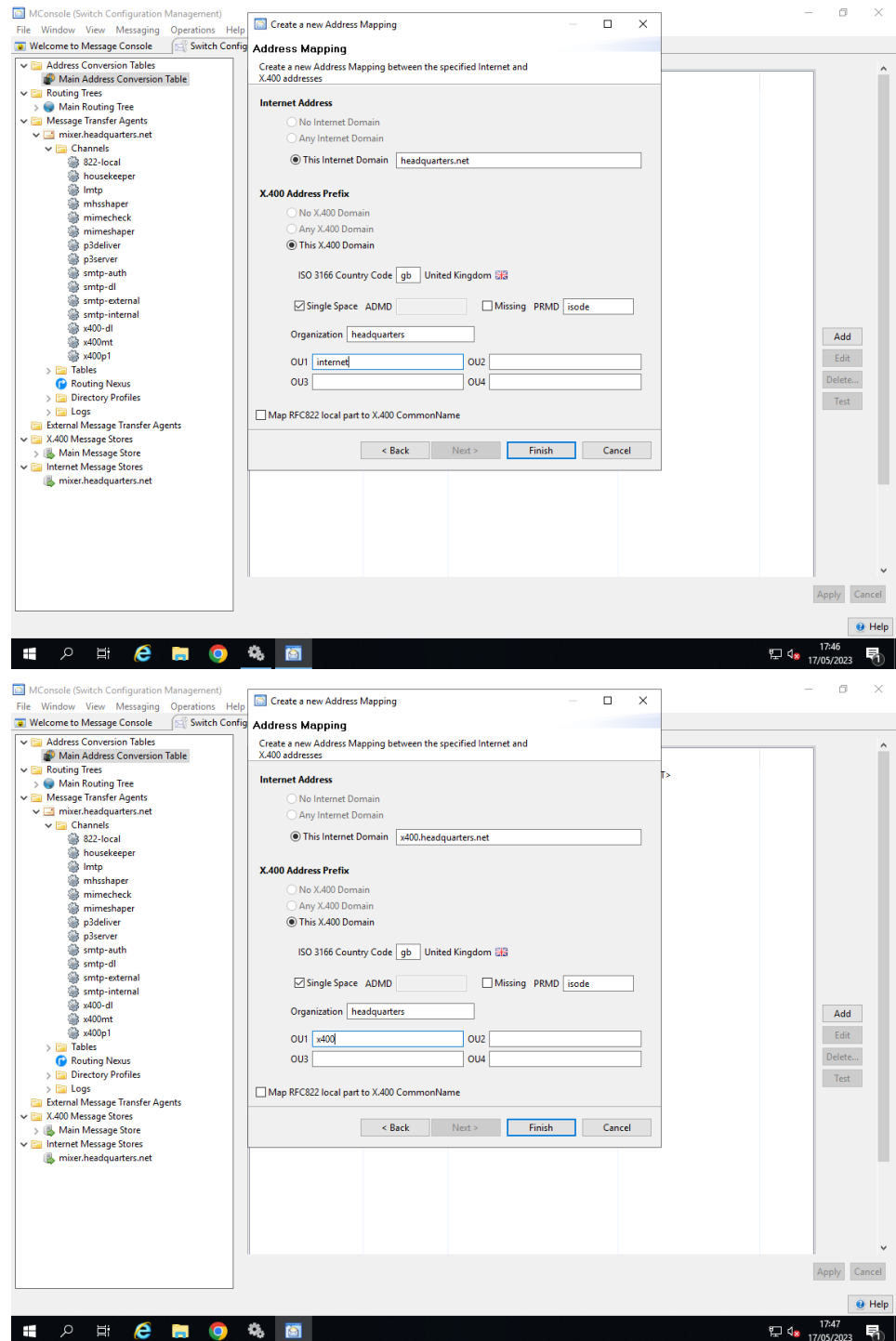
Output of `ckadr` utility before algorithmic mapping looks like this:

```

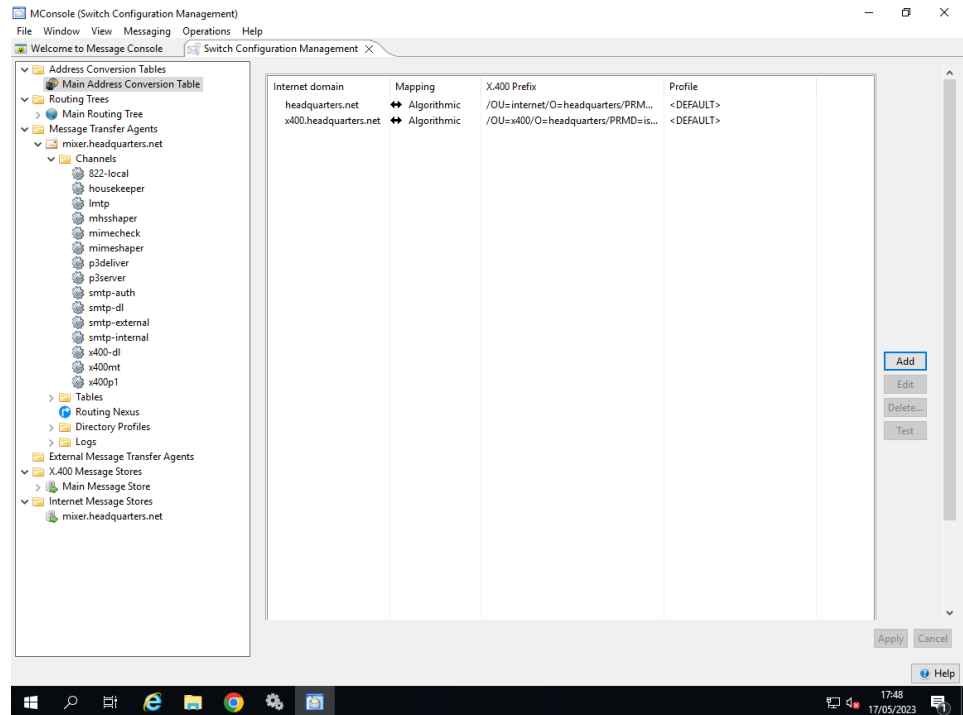
C:\Program Files\Isode\bin>ckadr -p R
test.user1@headquarters.net
test.user1@headquarters.net -> (rfc822)
test.user1@headquarters.net
test.user1@headquarters.net -> (x400)
/RFC-822=test.user1(a)headquarters.net/OU=internet/O=headquarters/
/PRMD=Isode/ADMD= /C=gb/

```

**Figure 12.12. Address Conversion Tree - adding a new algorithmic mapping**



Click on **Finish**.

**Figure 12.13. Address Conversion Tree - New algorithmic mapping added**

Output of ckadr utility after algorithmic mapping looks like this:

```
C:\Program Files\Isode\bin>ckadr test.user1@headquarters.net
test.user1@headquarters.net -> (rfc822)
test.user1@headquarters.net
test.user1@headquarters.net -> (x400)
/G=test/S=user1/OU=internet/O=headquarters/PRMD=Isode/ADMD= /C=gb/
```

Note how the least significant subdomain component, which is not configured as part of the mapping, is mapped algorithmically to an Organizational Unit. The Personal Name is also mapped algorithmically.

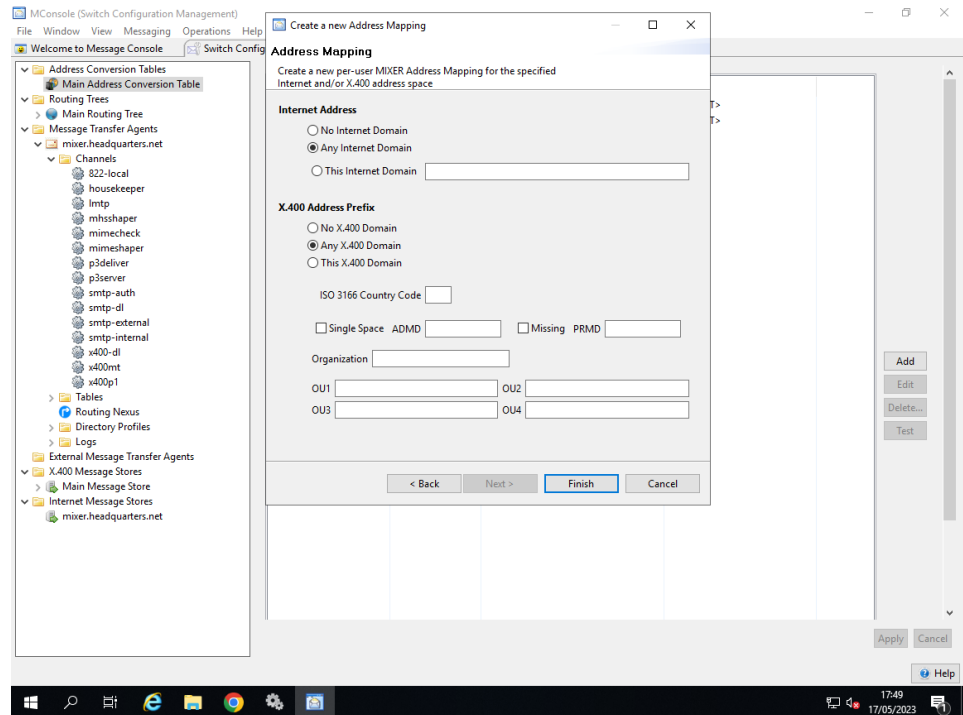
### 12.3.2.2 Configuring per User Mappings

It is also possible to configure per user address mappings. You can configure M-Switch to perform a search in the Directory for entries which configure an Internet to X.400 mapping (or vice versa). By default this requires a search for a Directory Entry which contains a Mail attribute and an mhsOrAddresses attribute.

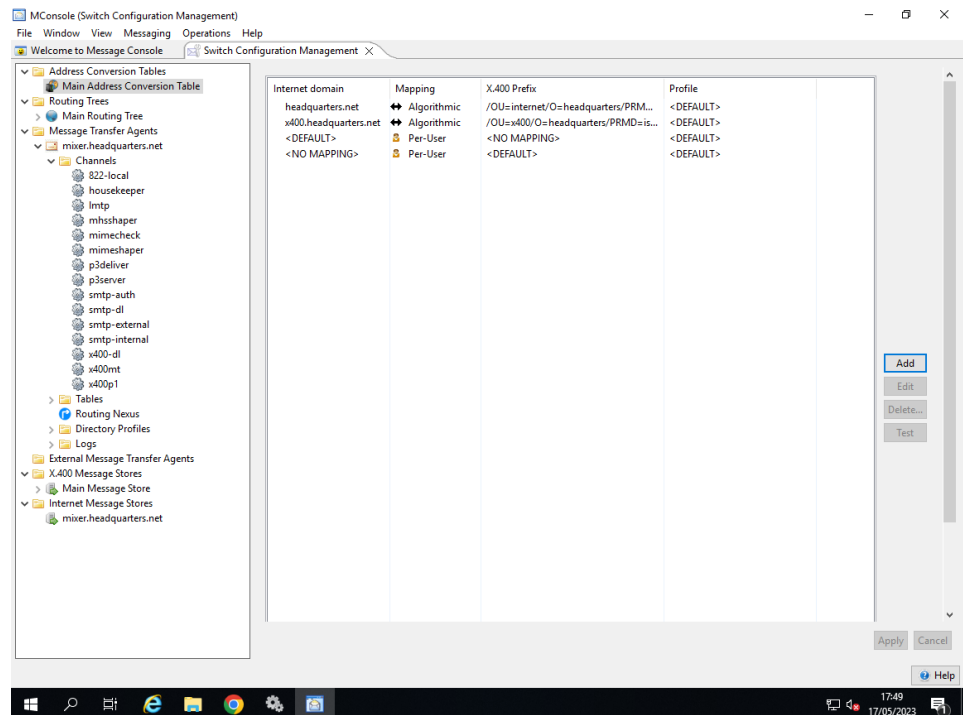
The search needs to result in exactly one result being returned for the source of the mapping, whose entry also contains a single value for the target of the mapping.

By leaving the Internet domain and X.400 O/R Address blank, all Internet domains and X.400 O/R Addresses which are considered to be local to the MTA will result in an attempt to perform address mapping using per-user mappings.

The DSA is searched as configured by the Directory Profile being used. The wizard configures the per-user mapping to use the Directory Profile named "Default". By editing the profile (or by creating another Directory Profile) the way in which the lookup for the mapping is carried out can be changed.

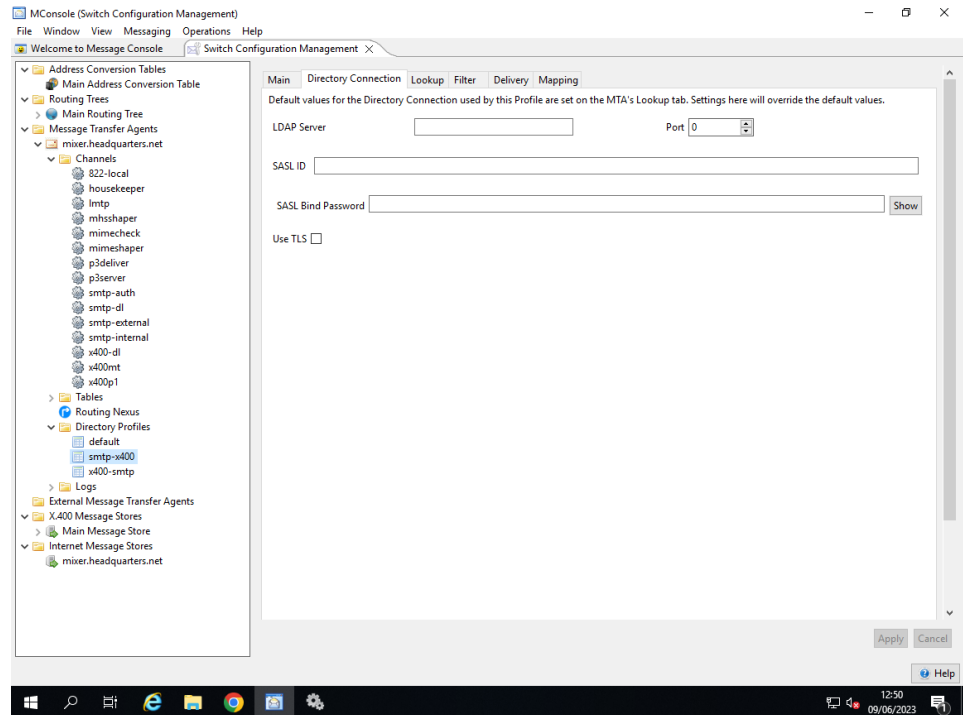
**Figure 12.14. Address Conversion Tree - New per user mapping addition**

Click on **Finish**, and the per user mapping is now displayed.

**Figure 12.15. Address Conversion Tree - New per user mapping added**

To make the per user mapping active (i.e. to trigger a lookup) you must add the Internet domain into the domain table as local

To configure the way the per user mappings are looked up in directory, configure the Directory Profile entry used by the Per User Mappings. You can use the default Directory Profile (called "default"), or create one of your own.

**Figure 12.16. Directory Profile as used by Per User Mappings**

### 12.3.3 Address conversion when attributes are missing

In the above example the O/R Address contains the optional attribute `PRMD`. If this is not used, difficulties arise in applying the mappings symmetrically. The following example shows this.

If the mapping required is `/ADMD=TestADMD/C=GB/` to `test.mixer.com`, i.e. the company does not use an Organization name, the mapping of `sales.test.mixer.widget.com` is not performed correctly, as it will be mapped to `/PRMD=sales/ADMD=TestADMD/C=GB/`, instead of to `/O=sales/ADMD=TestADMD/C=GB/`.

To overcome this you should configure the `PRMD` as a missing component by checking the **Missing** button by the `PRMD` in the **Create a New Address Mapping** window. This causes the conversion process to skip the `PRMD` when constructing the O/R Address.

This procedure conforms to RFC2164.

### 12.3.4 MIXER routing

This section explains how to set up a MIXER network of MTAs consisting of one or more X.400 MTAs, one or more Internet MTAs and one or more MIXER MTAs.

Previous sections in this Chapter have described how to set up the MTAs and the Address Conversion Trees. You must also ensure that the routing is configured to enable X.400 users to send messages to Internet users via your MIXER gateway. For X.400 MTAs you need to ensure that the Routing Tree entries are set up to route messages for Internet side of the MIXER gateway to the MIXER MTA. See [Section 16.1, “X.400 routing”](#) for details of this.

Similarly for Internet MTAs, whether you are using DNS or Routing Trees, you need to configure Internet MTAs to route messages for the X.400 side of the MIXER gateway to the MIXER MTA. See [Chapter 8, \*Connecting to SMTP MTAs\*](#) for details of this.

# Chapter 13 DSA Authentication and Authorization

This chapter gives an overview of the way in which M-Switch can be configured to access the DSA to obtain its Configuration Information, as well as User Configuration. This includes the used of both SASL IDs and Directory Names.

---

## 13.1 Overview

M-Switch uses the DSA in order to perform Authentication and Authorisation for components of M-Switch and other entities.

Some specialised messaging setups (usually when the DSA is not being used, such as when configuration is table based) can use an XML table to configure SASL IDs passwords and associated access rights. This is described in the *M-Switch Advanced Administration Guide*

The rest of this chapter explains the way in which Authentication and Authorisation is performed by M-Switch using the DSA.

There are two main areas in which DSA is used to support Authentication and Authorisation in M-Switch. These two areas can be configured in the same DSA, and by default this is the case.

- Configuration Information which is used to hold tailoring information about M-Switch as managed by MConsole. The qmgr uses this information in order to connect to the Configuration DSA and download into various configuration files. E.g. *mtataylor.tai*.
- User Information, which is used to Authenticate clients connecting into M-Switch server components. This includes:
  - SOM client authentication by qmgr
  - SMTP client authentication by smtp channels
  - LASER lookups of Internet users

See [Section 14.1.3, “Lookup information”](#) for a description of how to configure different DSAs to hold Configuration information and User information.

From R19.0 onwards, M-Switch configurations default to using SASL IDs rather than anonymous access or Simple Binds. This chapter describes how SASL IDs are created, edited and used to make M-Switch features available to authenticated clients.

---

## 13.2 SASL Authentication - Introduction

### 13.2.1 SASL Identifiers

A SASL identifier is a string which resembles an Internet email address, e.g. *user@domain*. The *@domain* may be absent, in which case a default domain which depends on the server's configuration will be assumed. Whether a SASL identifier is actually usable as an email address depends on configuration which is outside the scope of this section.

See [Section 6.1, “Overview”](#) for a description of how to configure Internet Users.

## 13.2.2 Use of Directory

When using the Directory to authenticate SASL IDs, the DSA needs to be able to map the SASL ID to a Directory Entry. A number of attributes located in the DSA's configuration entry (`cn=core,cn=config`) control the algorithms used when locating the authentication and authorization information for a given SASL id. The Queue Manager reads these attributes (or uses defined default values when they are not present). The most important attributes used are:

- The `isodeSaslGenericRule` attribute controls the search strategy used to locate a user's entry, given their SASL id. The default value of 3 indicates that a single search operation should be performed.
- The `isodeSaslGenericFullMatchAttr` attribute specifies the attribute type which should be used when searching on a SASL id. The default value is `mail`.
- The `isodeSaslGenericBase` attribute specifies the DN of the search base which should be used. The default is to search from the base of the DIT.

Once the search strategy and attributes used have been determined, the Queue Manager searches the Directory for information which can be used to authenticate the specified SASL id. Access control considerations mean that both the Queue Manager's initial access to the DSA's configuration entry and the subsequent searching/reading of entries containing authentication information will probably require the Queue Manager to bind to the Directory as a privileged entity, particularly as read access to `userPassword` attributes is required.

If you are using a DSA which has been created using one of the standard Isode templates, then a `userid` (e.g. `"cn=Isode Application Server,cn=Users,o=Messaging"`) with the necessary rights will already exist, and the Queue Manager will already be configured to use this in order to authenticate SOM clients.

## 13.2.3 Passwords in the DSA

When M-Vault is initialised, the DSA being created can be configured to store passwords in hashed form. This is a security measure which avoids the need to store passwords in the DSA in the clear. When MConsole creates a DSA to support M-Switch in R19.0, this option defaults to true.

This means that the available SASL mechanisms as described in [Section 13.2.4, “SASL Mechanisms”](#) do not all work. You are restricted to SCRAM-SHA-1, PLAIN and LOGIN. By default, R19.0 configurations are created to use SCRAM-SHA-1. See [Section 14.1.3.1, “Directory Access”](#) for a description of how you configure PLAIN and LOGIN mechanisms. By default, use of PLAIN or LOGIN only works if TLS is being used.

## 13.2.4 SASL Mechanisms

A number of different SASL mechanisms are available. Standard mechanisms are CRAM-MD5, DIGEST-MD5, SCRAM-SHA-1, PLAIN and LOGIN: these all work without the need for additional (external) resources. Support for GSSAPI (i.e. Kerberos-based authentication) is provided, but this relies on external configuration and will be described later.

SCRAM-SHA-1, PLAIN and LOGIN are the only mechanisms which will work when password hashing is being used (i.e. the Directory in which passwords are being held was created with SCRAM-SHA-1 hashing enabled).



---

## 13.3 Configuration Using MConsole

### 13.3.1 Authenticated Entities

Configuration of SASL IDs is now carried out using the Isode User Provisioning Application Cobalt, and is documented in the Cobalt Administrator's Guide.

Configurations which do not have configured Internet Users (such as pure X.400 configurations, or a MIXER Gateway with no local users) continue to use MConsole to managed the Authenticated Entities as SASL IDs. The rest of this section describes how such SASL IDs in these configurations are configured in MConsole.

### 13.3.2 Use of SASL IDs

The SASL IDs configured here are used to provide authentication in two ways.

- As SOM clients using the SASL ID itself which connects to M-Switch server components in order to provide operational access to M-Switch. For example using the Switch Operations View to connect to the Queue Manager.
- As DSA clients which bind using the DN equivalent of the SASL ID in order to manage configuration data held in the DSA.

The SASL ID and DN used in the above two cases are mapped to each other using the algorithms and DSA configuration described in [Section 13.2.2, “Use of Directory”](#).

When configuring access as a SOM client, the SASL ID itself is used. When DSA access is being configured, the Directory Name of the Entry is used. It is important to understand this distinction when configuring different tabs in the Views described later in this chapter.

### 13.3.3 Configuration of SASL IDs

The configuration of SASL IDs as SOM clients uses the following different Views in MConsole.

- The **Lookup** tab for an MTA in the **Switch Config View** holds the details of the way in which the SASL server (e.g. Queue Manager, ACP127 channel) uses the DSA to authenticate SOM clients.
- The SASL IDs are configured in the **Authenticated Entities View**.

The five tabs available are **Queue and Store Auth**, **ACP127 Monitor Auth**, **Access Control Groups View**, and **Control View**.

### 13.3.4 Configuration of DSA Access

Fields on the **Lookup** tab for an MTA in MConsole allow configuration of the Configuration DSA and User DSA used by M-Switch. This is described in [Section 14.1.3, “Lookup information”](#).

### 13.3.5 SASL User Management

SASL Identifiers and the associated credentials and authorization data can be managed using the **Authenticated Entities Management** view.

The **Authenticated Entities Management** view has the following tabs:

- Authentication
- Queue and Store Auth
- ACP127 Monitor Auth
- Access Control Groups
- View Control

### 13.3.5.1 Authenticated Entities Management View

The **Authenticated Entities Management** view allows SASL identities which are only to be used for SOM authentication to be created, modified and deleted.

It also allows the creation of view controls, allowing an administrator to limit the views exposed to certain users.

Entries are split between:

#### Internal Users

These are users created by MConsole to allow connections via SOM, for example and MTA Administrator.

#### Authorised External Users

These are users that have been created via a separate program, for example **Cobalt**.

---

**Note:** The view is populated via an **Auth Entities Search Parent DN**. New **Internal User** entries are created under the **Queue Manager Authentication Users Paren DN** location. These values are set under **View->Options->Authenticated Entities Management**.

---

#### Upgrade External Users

All users require certain Object Classes and attributes, to be used for SOM authentication. This option allows an administrator to select which user entries need to be modified to have those Object classes and attributes set. Once upgraded users will be displayed within the **Authorised External Users** section of the view.

#### Create Internal User

Creates a basic user entry with enough schema to be able to perform SOM authentication and access control.

#### Add view control

Creates a new view control.

#### Refresh

Reloads the **Administration Rights Management**.

#### 13.3.5.1.1 Authentication

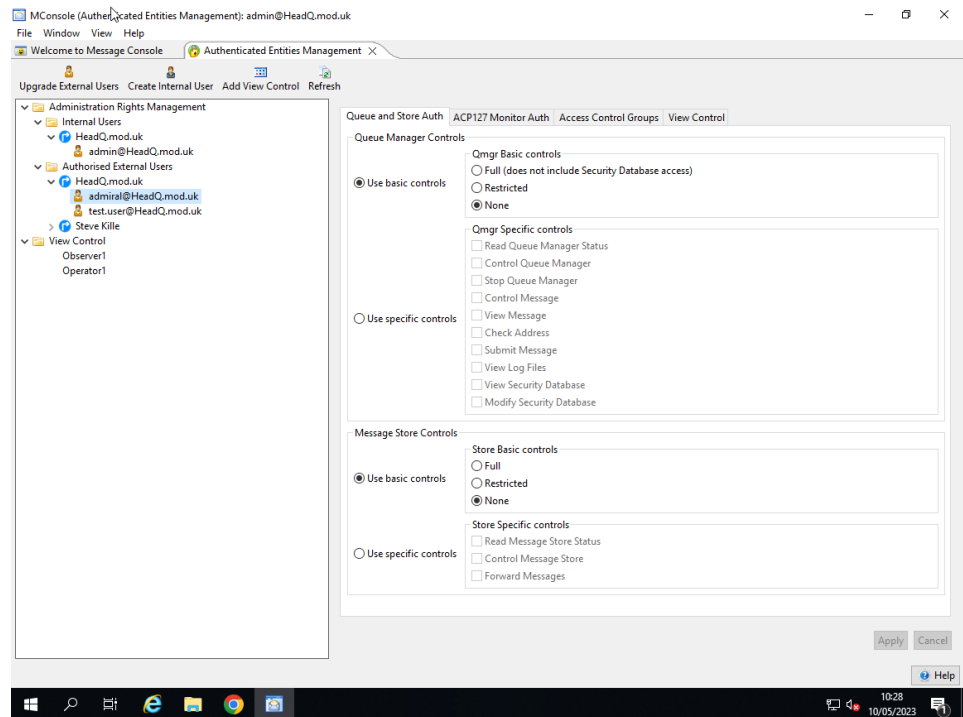
The only attribute that requires configuration is the password that the authenticated entity will use.

#### 13.3.5.1.2 Queue Manager Authentication

The Queue and Store Auth tab allows configuration of the set of SOM operations which this identity is authorized to perform. Either a coarse choice (Full, Restricted or None) can be made, or a fine-grained selection of the specific set of operations available to the identity can be made via the "Specific controls" selector.

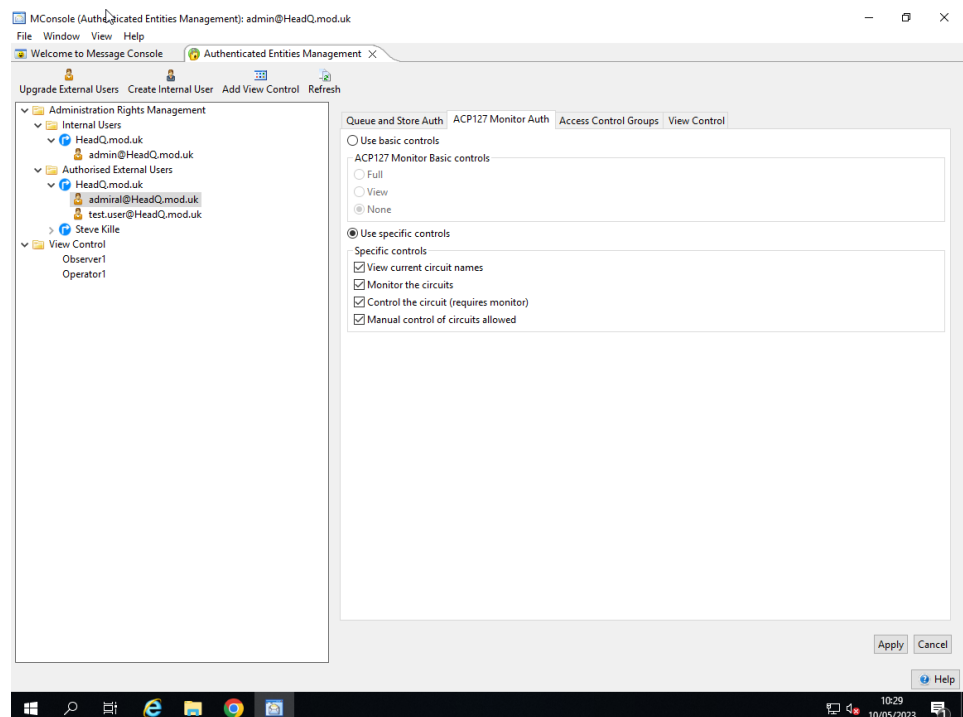
Similarly the controls for this user when connected to the Message Store are configured in the Message Store Controls section.

If the MTA has a Corrector channel, the checkbox can be configured which allows an Operator to authenticate using the Web Interface provided by the Corrector channel.

**Figure 13.1. Queue and Store Auth tab**

### 13.3.5.1.3 ACP127 Monitor Authentication

The ACP127 Monitor Auth tab allows configuration of the set of SOM operations which this identity is authorized to perform in the ACP127 View. Either a coarse choice (Full, View or None) can be made, or a fine-grained selection of the specific set of operations available to the identity can be made via the "Specific controls" selector.

**Figure 13.2. Authenticated Entities Management View - ACP127 Monitor tab**

### 13.3.5.1.4 Access Control Groups

The Access Control Groups tab allows the SASL ID to be associated with differing levels of privilege to access the DSA. An Administrator who is to be assigned full privileges

would be a member of each group, apart from Application Server. This last group is used to enable a Server which requires the DSA to validate SOM client credentials. In practice the privilege granted to such group members is to read passwords from the DSA.

When a messaging configuration is created, the SASL ID `Iside.Application.Server@<local domain>` is made a member of this group.

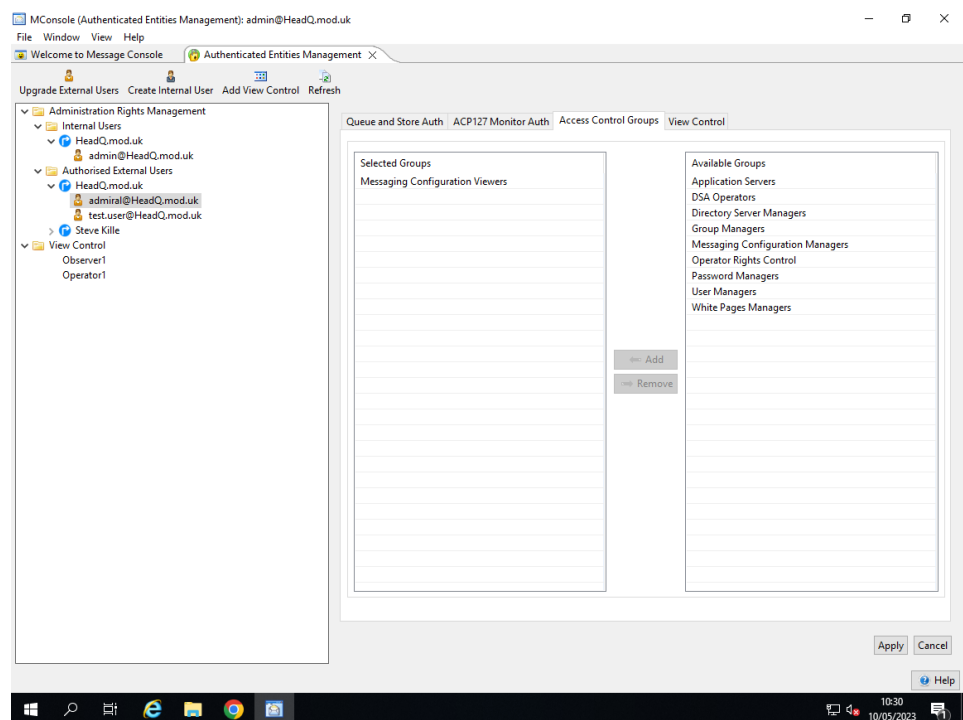
This works as each SASL identity has a DistinguishedName (DN), so it is possible to use the DN and password for any identity in a Directory Bind operation to access the Messaging Configuration from MConsole. These DN and passwords are stored in the User's Bind Profile.

This DN can be copied to the clipboard by selecting the entry on the left hand side and using the right click button.

The **Access Control Groups** tab for each user on the **Authenticated Entities Management** view allows an individual SASL identity to be assigned as a member of various pre-defined groups, granting different levels of access to different areas of the Directory Information Tree.

The following figure shows how a User can be assigned the privileges to view M-Switch configuration.

**Figure 13.3. Authenticated Entities Management View - Access Control Groups tab**



### 13.3.5.1.5 View Control

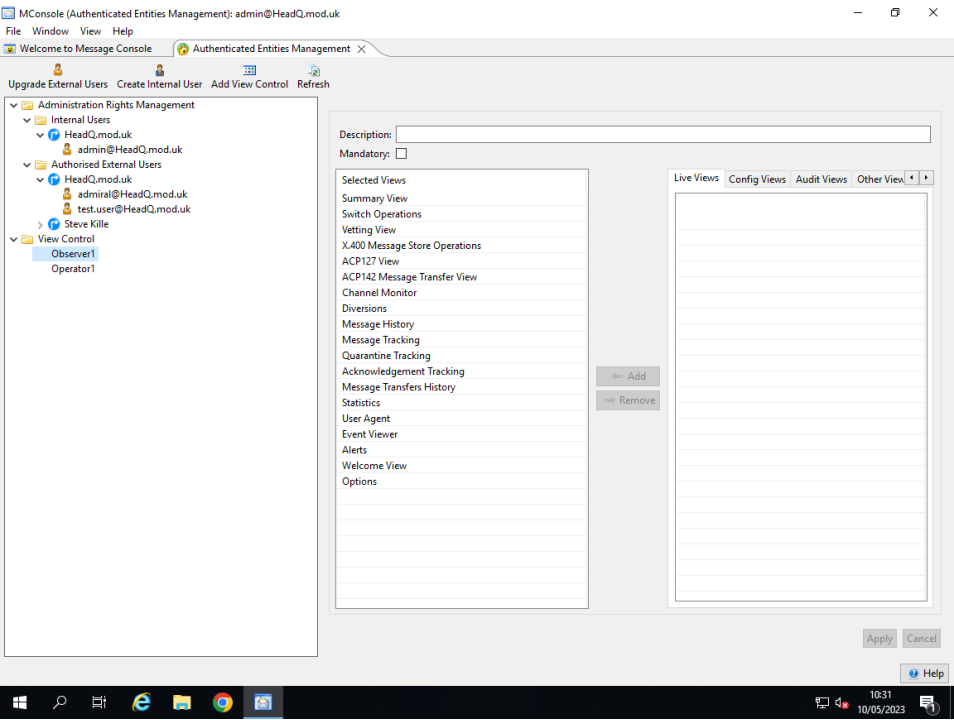
The View Control tab allows the Administrator to control which MConsole Views are presented to the User when logged in with this SASL ID. The MConsole Views are grouped into the following categories:.

- Live
  - Switch Operations
  - Summary
  - Vetting
  - X.400 Message Store Operations

- ACP127
- ACP142 Message Transfer
- Channel Monitor
- Diversions
- Config
  - Switch Configuration Management
  - X.400 Mailbox Management
  - Gateway Users
  - Authenticated Entities Management
  - ACP127 Addresses
- Audit
  - Message History
  - Message Tracking
  - Quarantine Tracking
  - Acknowledgement Tracking
  - Message Transfers History
  - Statistics
- Init
  - New DSA
  - New Config
  - Use Existing Config
  - Backup
  - Use default config
- Other
  - User Agent
  - Event Viewer
  - Alerts
  - Welcome
  - Shaper Config File Editor
  - Options

The View Controls are created, edited and deleted using the View Control editor at the bottom of the left hand side of the View as shown below.

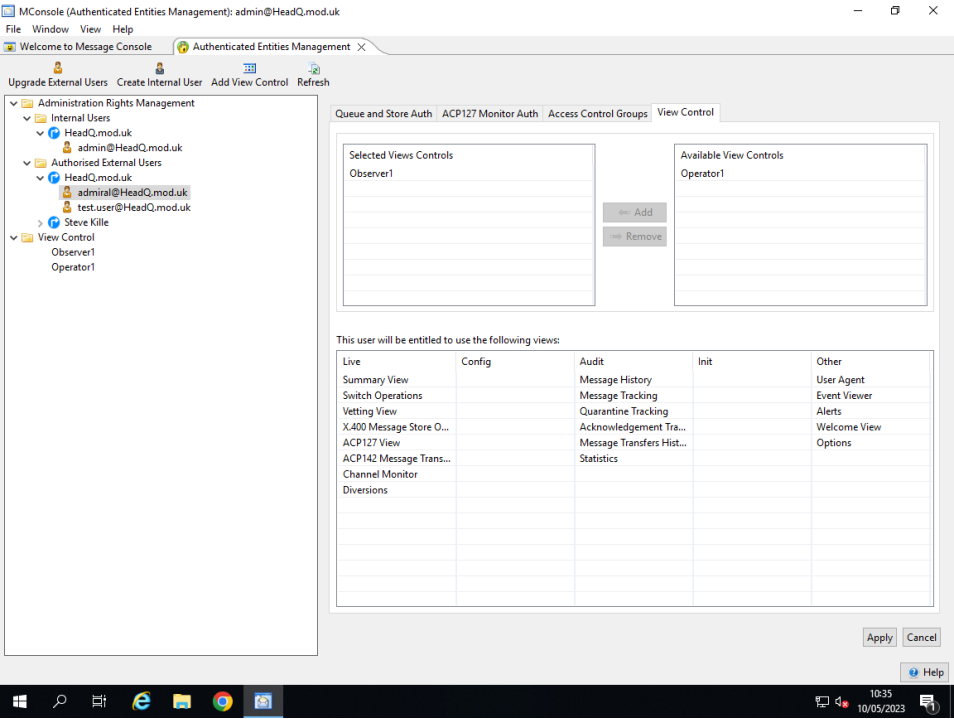
Figure 13.4. Authenticated Entities Management View - View Control Editor



This shows an Observer view control which includes all those views which a User with Observer privileges requires, e.g Switch Operations.

The next figure shows a User being configured with the View Control created in [Figure 13.5](#), “Authenticated Entities Management View - View Controls tab”. Note the lists of Views available.

Figure 13.5. Authenticated Entities Management View - View Controls tab



---

## 13.4 Creating Accounts

The M-Switch Configuration is created originally with an MTA Administrator SASL ID. Once the configuration is complete it is likely you will wish to create other accounts with different levels of privilege. This section describes how to create the two additional accounts you are most likely to need. These are :

- **Operator Account.** This is a user who can view configurations and has some rights to manage the MTA itself.
- **Observer Account.** This is a user who can view some of the MTA's operational state.

### 13.4.1 Creating An Operator Account

The standard MTA Administrator SASL ID which is set up as part of creation of a Messaging Configuration has SOM access rights which allow full control of the running MTA, plus read/write access to the Messaging Configuration area of the DIT.

MTA Operators require full control of the running MTA but should not need to modify the configuration of the MTA stored in the DSA. To create an Operator account:

- Open the **Authenticated Entities Management View** and add an appropriate User.

Add a Queue Manager Authenticated User. You are offered the choice of Administrator, Observer and Operator. Select one of these to provide sensible defaults for the values described below.

Configure the SASL ID with appropriate Queue Manager controls, as described in [Figure 13.1, "Queue and Store Auth tab"](#). If your Messaging Configuration includes ACP127 channels, you can configure appropriate ACP127 Monitor controls as well as described in [Figure 13.2, "Authenticated Entities Management View - ACP127 Monitor tab"](#).

- After creating the SASL ID, select it and open the **Access Control Groups** tab. Add the **Messaging Configuration Viewers** group and **Operator Rights Control** group to the User's **Selected Groups** list as described in [Figure 13.3, "Authenticated Entities Management View - Access Control Groups tab"](#).
- After editing the **Access Control Groups** tab, you may wish to manage the Views available to this Operator. You can restrict the Views a User is permitted to open. You may also wish to make it impossible for the Operator to close certain Views. To do this you need edit the View Controls as described in [Figure 13.4, "Authenticated Entities Management View - View Control Editor"](#). You can then add the View control you have created to the Operator as described in [Figure 13.5, "Authenticated Entities Management View - View Controls tab"](#).
- The **Distinguished Name** which corresponds to the newly-created SASL ID can be obtained from the right-mouse-button menu for the SASL ID: the **Copy DN** option will copy the DN into your paste buffer.

Three pieces of information will be needed when setting up a Bind Profile so that an Operator can use MConsole: the DN referred to above, which is used in conjunction with the password configured for the ID in the DSA Bind Profile, and the SASL ID and the same password, which are needed when setting up SOM access to the Queue Manager.

## 13.4.2 Access Control Groups example: adding an Observer

This example, shows what is needed in order to create a user allowed to run MConsole in a mode where he could inspect the configuration and running MTA but not perform any modifications, you would need to:

Follow the instructions in [Section 13.4.1, “Creating An Operator Account”](#) to create the observer, apart from the **Access Control Groups** tab in which you add the **Operator Rights Control** group only to the ID's **Selected Groups**.

### 13.4.2.1 Logging in as the New User

Once the new User (e.g. Operator) has been added, you need to logout of the OS User acting as the Administrator and log into the OS as the Operator User and configure the User's Bind Profile.

- To create the new Bind Profile, go to **Switch Configuration Management** view and select **Messaging** then **DSA Profile Editor**. From here you can select **New** to add a new bind profile, enter all relevant values and paste your copied DN into the **Bind DN** value box. Once finished you can select **Connect** to connect to the directory using your new bind profile.
- Using the new Bind Profile, connecting to the Directory as "Joe Bloggs", will now give read-only access to the Messaging Configuration.

---

## 13.5 Advanced configuration

Some advanced aspects of Queue Manager authentication can only be configured via PP Internal Variables. These can be set using the **Advanced** tab for the MTA in the **Switch Configuration Management** view.

### 13.5.1 Allowing Anonymous Access

The PP internal variable `qmgr_anon_rights` allows the rights for anonymous SOM connections to be assigned. The value of the variable should be a space or comma separated list of the rights assigned to anonymous connections. The available values are:

- `full` Full read and control access
- `restricted` Read-only access
- `none` No access - the default: anonymous connection attempts will be rejected.
- `qread` Read Queue Manager status
- `qctl` Control Queue Manager
- `qstop` Stop Queue Manager
- `qmsgctl` Control messages
- `msgview` View messages
- `ckadr` Check addresses
- `submit` Submit messages
- `logview` View log files



# Chapter 14 Configuring MTAs

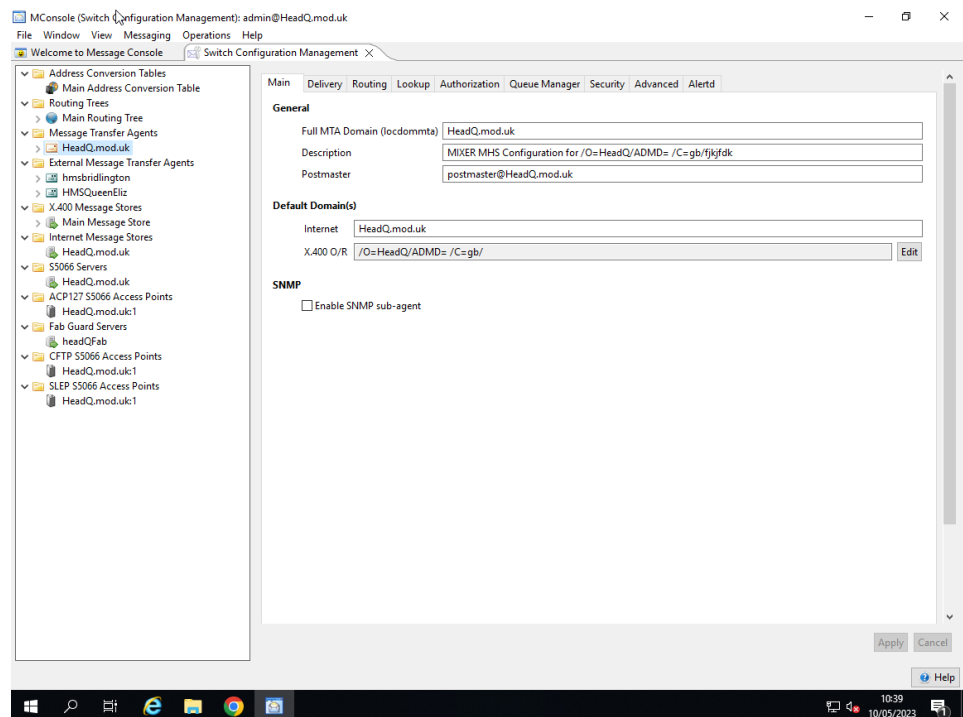
This chapter describes the configuration of an MTA using MConsole.

## 14.1 Editing an MTA's properties

To edit the properties of an MTA that has already been created, expand the **Message Transfer Agents** folder and select the MTA you want to edit. Nine tabs appear on the right hand side window:

- **Main:** The MTA's most important attributes
- **Delivery:** Configuration of the intervals after which to non-deliver messages or send warning messages and options for loop detection
- **Routing:** The routing trees and Address Conversion Tables used by this MTA
- **Lookup:** The lookup policies used by the MTA to resolve Internet addresses and the directory access method
- **Authorization:** The rule-based mechanisms used by the MTA to permit or block messages, control archiving or message checking and select inbound channels
- **Queue Manager:** Some of qmgr settings, used to fine-tune the performance of a system
- **Security:** TLS and SASL configuration, if used
- **Advanced:** Other values which do not usually need to be changed in most configurations.
- **Alertd:** Allows configuration of the Isode Alert Daemon, which can be used to monitor this MTA.

**Figure 14.1. Properties of an MTA (Main page).**



The fields on this screen are:

- **Full MTA Domain:** This is the full domain name of the local MTA. It is used for trapping routing loops and so should be globally unique. For example:  
myhost.mysubdomain.mydomain.com.
- **Description:** A text field which describes the MTA. This is optional.
- **Postmaster address:** The RFC822 address of the postmaster for this MTA. This is not displayed for X.400-only MTAs.
- **Default Domain(s):** These are the default Internet and X.400 domains to which this MTA belongs. If your MTA is X.400-only or Internet-only, you will only see one relevant domain.
- **Enable SNMP subagent:** selecting this option makes the qmgr connect to the master SNMP agent enabling monitoring of M-Switch using SNMP.

See [Section 35.1, “SNMP overview”](#) for a description of SNMP.

## 14.1.1 Delivery details

Figure 14.2. Properties of an MTA (Delivery page).

MConsole (Switch Configuration Management): admin@HeadQ.mod.uk

File Window View Messaging Operations Help

Welcome to Message Console Switch Configuration Management

Address Conversion Tables

- Main Address Conversion Table

Routing Trees

- Main Routing Tree

Message Transfer Agents

- HeadQ.mod.uk

External Message Transfer Agents

- hmsbridlington
- HMSQueenElizabeth

X.400 Message Stores

- Main Message Store

Internet Message Stores

- HeadQ.mod.uk

S5066 Servers

- HeadQ.mod.uk

S5066 Access Points

- ACP127 S5066 Access Points
- HeadQ.mod.uk1
- Fab Guard Servers
- headQFab
- CFTP S5066 Access Points
- HeadQ.mod.uk1
- SLEP S5066 Access Points
- HeadQ.mod.uk1

Main Delivery Routing Lookup Authorization Queue Manager Security Advanced Alerts

Timeouts

Priority mode ☐ Standard priorities ☒ Military priorities

Timeout from Submission	Timeout from Arrival	Target processing time
Override 2 hours		Routine 12 hours
Flash 4 hours		Deferred 18 hours
Immediate 6 hours		Probes 8 hours
Priority 8 hours		Reports 8 hours

Detect loops after

Maximum of (traces) ☒ Use default value (25)

Loops up to (times) ☒ Use default value (5)

Warning messages

Warning Interval Initial Warning Interval

Warning Interval	Initial Warning Interval
Override 1 days	
Flash 1 days	
Immediate 1 days	
Priority 1 days	
	Routine 1 days
	Deferred 1 days
	Probes 1 days
	Reports 1 days

Number of warnings

Send a maximum of (messages) ☒ Use default value (2)

Apply Cancel

Help

10:40 10/05/2023

The fields on this page are:

- **Priority mode:** The timeouts that can be configured depend on the message priority (see below). Standard X.400 priorities are different from Military priorities, so select which kind of priority to display and configure.
- **Timeout from submission:** This is the time after which to expire an undelivered message. The time is calculated from the time a message is submitted to the MTS, or from the deferred time. By default, this time specified for **Normal** is doubled for low priority messages and halved for high priority messages. However, these other values can be overridden by specifying the other times as well as the **Normal** return time. **Reports** (delivery reports) timeouts can also be configured.
- **Timeout from Arrival :** This is similar to **Timeout from submission**, except that the time is calculated from the time a message arrives at the MTA, as opposed to the time a message is submitted to the MTS. The purpose of this variable is to permit the local delivery or transfer of messages which have exceeded the time given in **Timeout from Submission**. This could occur if the message has taken a long time to reach the MTA or if an old RFC 822 message has been resent. Setting values greater than **Timeout from**

**submission** values results in only the time from arrival at the MTA being used. Delivery **Reports** timeouts can also be configured.

- **Detect loops after**

- **Maximum of 'N' traces** is the maximum number of trace fields a message may contain i.e. the maximum number of MTAs through which a message can traverse before being regarded as looping. If there are more than this number, the message will be rejected as looping.

- **Loops up to 'N' times** is the number of times a message may pass through this MTA before being rejected.

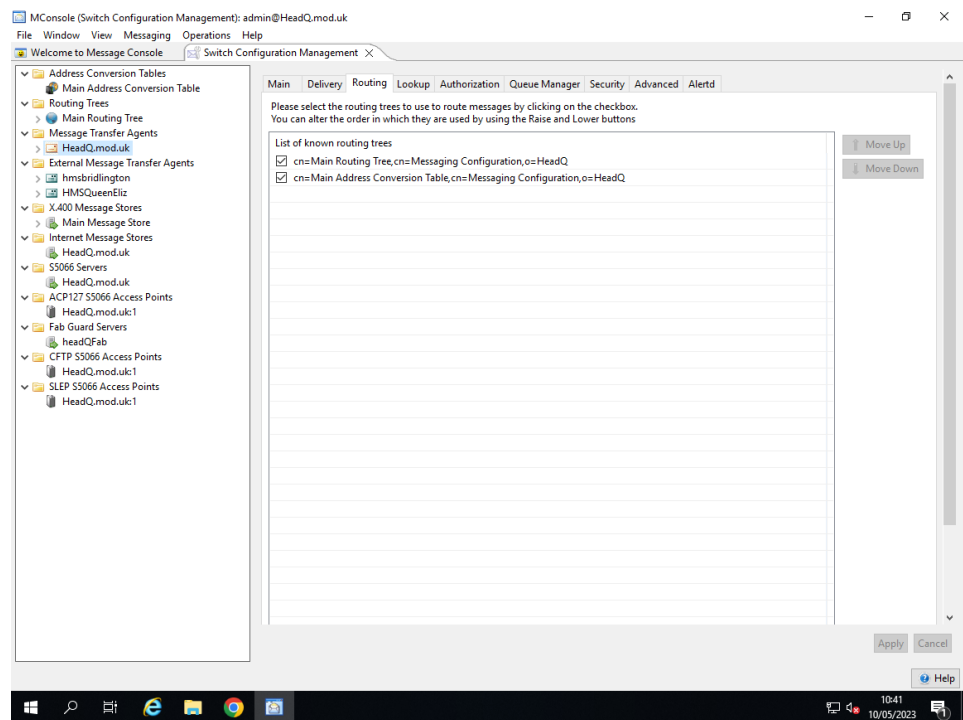
- **Warning Messages**

- **Send a maximum of 'N' messages** is the maximum number of warnings to send for a particular message.

- **Interval 'N' hours** is the time in hours the channel will wait before sending out a warning message.

## 14.1.2 Routing information

**Figure 14.3. Properties of an MTA (Routing page).**



This page allows the set of routing trees and Conversion Tables that this MTA uses to be set to be active or not. A straightforward MIXER system will usually have one Routing Tree and one Conversion Table.

---

**Note:** A Conversion is actually a routing tree but is presented in MConsole as a table to aid clarity.

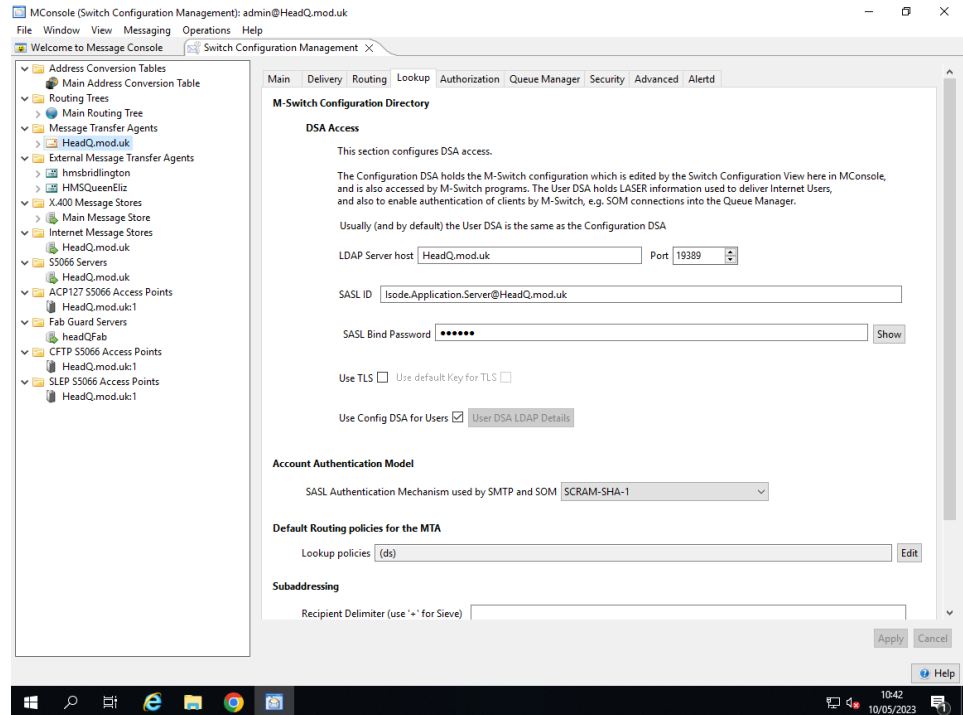
---

The routing trees that are used by the MTA have a checkbox next to them. To make the MTA use one of the routing trees, simply click on the checkbox, and to make it stop using the routing tree, just clear the checkbox.

Once you have selected the routing trees that you want to use, you can edit the order in which they are used by selecting a routing tree on the table and then clicking on the **Raise** and **Lower** buttons.

## 14.1.3 Lookup information

**Figure 14.4. Properties of an MTA (Lookup page).**



This page allows you to edit the DSA connections used by mconsole and the MTA, Lookup policies and Subaddressing.

### 14.1.3.1 Directory Access

M-Switch and MConsole use the X.500/LDAP Directory (DSA) to store both Configuration Information and User Information.

Configuration Information is configured in Mconsole. The qmgr connects to the DSA and downloads the Configuration into the `mtatailor.tai` file for channels and other M-Switch programs to read.

The qmgr also downloads the Configuration into the `mtalogging.xml` file; the `profiler.json` file for the Profiler channel; and the Authorisation Groups and Rules files.

User Information comprises

- Routing Information for addresses and mailboxes (used for LASER lookup)
- authentication and authorisation information for SOM clients which connect to the qmgr and Routing Information for addresses and mailboxes.
- SMTP Authentication information used to provide authentication on submission

Access uses LDAP (DAP access is no longer supported).

Most setups will hold Configuration Information and User Information in the same DSA, and so the **Use Config DSA for Users** will be checked.

Prior to R19.0 anonymous access for Configuration Information was supported and configured by default, and SASL configuration was used for User Information. This has been changed so that by default SASL and SCRAM-SHA-1 is used for both Configuration Information and User Information. As noted, by default the same DSA configuration is used for both as configured in the **Use Config DSA for Users**.

For more details on how SASL IDs are configured and used by M-Switch together with a description of Authenticated Entities, see [Section 13.1, “Overview”](#).

The SASL mechanism PLAIN can still be used by SOM clients in the Switch Operations View, but by default this is only allowed when TLS is being used. You can override this in the Security tab **Allow SASL PLAIN and LOGIN** see [Section 14.1.6, “Security settings”](#).

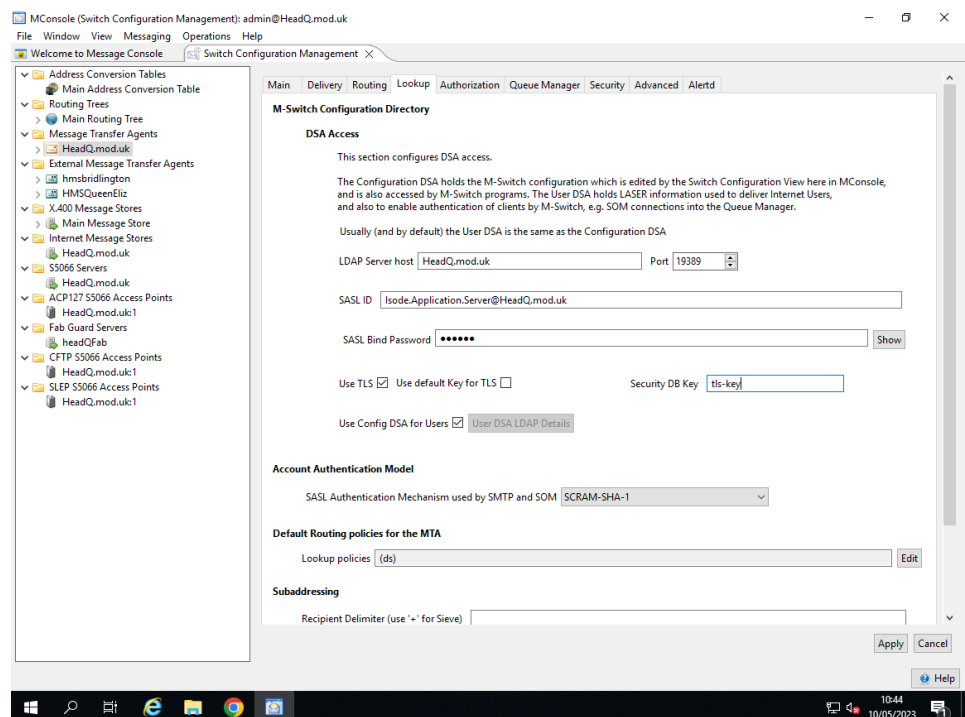
The Authentication model to be used is configured in the Account Authentication Model option. There are three options:

- SCRAM-SHA-1: (default) SOM will use SCRAM-SHA-1; SMTP will default to offering SCRAM-SHA-1 and PLAIN
- LDAP Bind (SASL PLAIN): SOM and SMTP will offer PLAIN only. Passwords are verified using LDAP BIND, which enables use of passthrough authentication to Active Directory.
- Any: SOM Access will use SCRAM-SHA-1. SMTP can be configured to offer any supported SASL mechanisms. This mode requires that passwords are stored in the clear in M-Vault and that M-Switch can read these passwords."

Connecting to the DSA using TLS can be configured by setting the **Use TLS** checkbox. For this to work you must configure the Security Database in the MTA Security Tab. When this is initialised a Certificate and Trust anchor are set up using an ad hoc Certificate Authority. This is suitable only for Testing and Demonstrations. This Certificate has the channel name `default`. Checking the **Use TLS** checkbox will result in the **Use default key for TLS** to be checked.

For deployments you are strongly recommended to use a Certificate issued by a trusted CA and importing it into the Security Database. The Key by which the Certificate is identified in the Security Database needs to be configured in the **Security DB key** box. See [Section 14.1.6.1.4, “Trusted Certification Authorities”](#) for a more complete description.

**Figure 14.5. Using Non Default TLS Key**



The above diagram shows how to configure a non default key for TLS.

In R19.0 a DSA created by MConsole for Messaging Configurations by default holds passwords in hashed form using SCRAM-SHA-1. So passwords can no longer be read

In some setups you may not wish to use default mode (SCRAM-SHA-1). The **Account Authentication Model** allows the mode to be changed to LDAP Bind mode. LDAP Passthrough is supported when in LDAP Bind Mode. This option is needed when using Passthrough to Active Directory (or other DSA's which do not have SASL available).

Also available is a legacy mode (any) which requires the DSA passwords to be held in the plain.

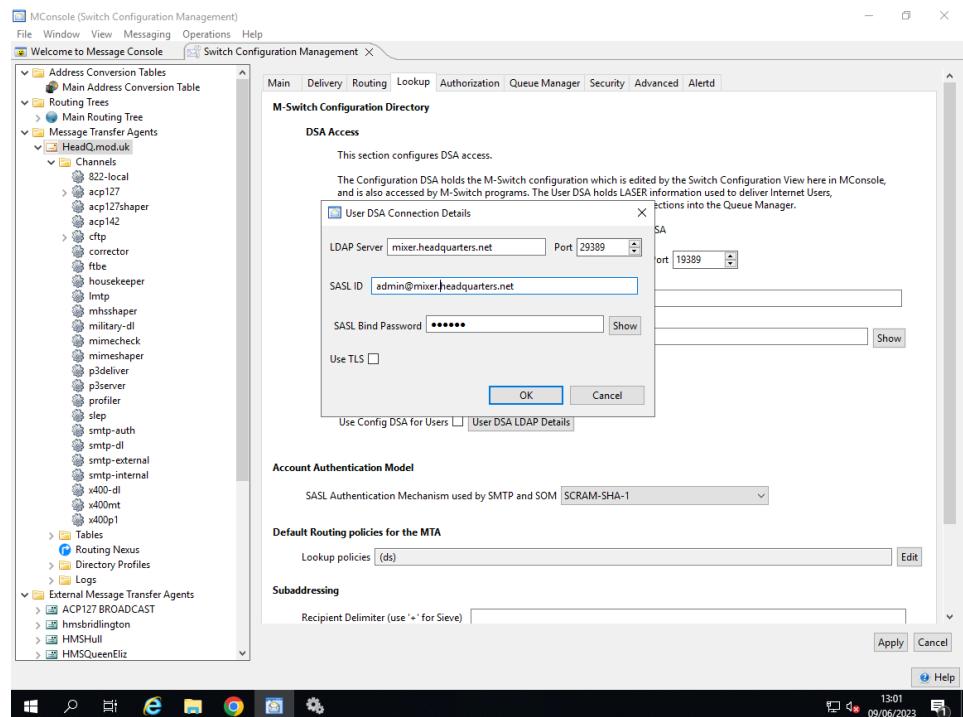
#### 14.1.3.1.1 User DSA Configuration Information

When a different DSA is needed to access User Information, you can uncheck the **Use Config DSA for Users**, which causes the **User LDAP Details** button to be enabled.

Clicking on the **User LDAP Details** button allows the User DSA configuration details to be edited. These have the same set of configuration options as for the Configuration DSA.

The figure below shows the popup which appears allowing the User Configuration to be in a different DSA.

**Figure 14.6. LDAP Connection Properties.**



- **LDAP Server host:** This is the hostname of the system on which the DSA is running.
- **Port:** The IP port on which to connect to the DSA, usually 19389.
- **SASL ID:** The SASL ID to be used in the SASL Bind to the DSA
- **SASL Password:** The SASL Password to be used in the SASL Bind to the DSA. Note that this can be held using servpass
- **Use TLS:** Use TLS to connect to the DSA. If set, a TLS Key must be specified. This defaults to the default key in the Security Database. This is configured in the Security Tab which must be configured to use the Security Database, NOT the legacy database.

You can (and should) import a trusted TLS Key into the Security Database, and name it with a channel Key which should match the value in the Security DB Key entry.

### 14.1.3.2 Subaddressing

**Recipient Delimiter (use '+' for Sieve).** This variable enables a subaddressing separator to be specified so that routing of the RFC822 address can be performed using only the section of the local-part of the address preceding the delimiter.

Commonly this delimiter is “+” and the technique is known as “plus-addressing”. For example:

```
set recipient_delimiter=+
```

Note that this control is not shown for X400-only MTAs.

### 14.1.3.3 Default Routing policies for the MTA

Most configurations use `ds` lookup policies in which only the DSA used to route addresses using Routing Trees and (for `smtp` addresses) `LASER`. `dns-ds` can be used where DNS is used to lookup and normalised domains.

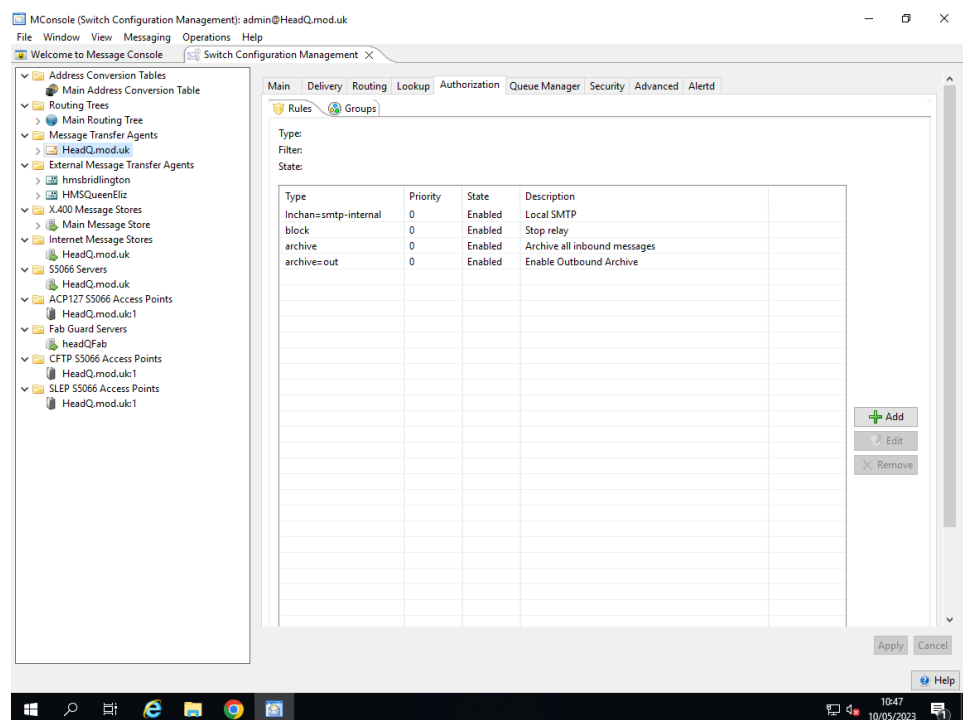
See [Chapter 15, Routing](#) for a full description of Routing and Lookup policies.

To edit the Default Routing policies for the MTA click on the **Edit** button to display the Default Lookup Policy editor.

This editor shows a table with the lookup policies in use. Use the **Add** button to add a new policy to the list. Use the **Remove** button to remove the selected policy from the list. Use the **Raise** button to move the selected policy up new in the list. Use the **Lower** button to move the selected policy down in the list.

## 14.1.4 Authorization

**Figure 14.7. Properties of an MTA (Authorization page)**



These entries configure the MTA's Authorization Rules. The MTA creation wizard sets up three rules for SMTP and MIXER MTAs:

- Inchan rule to determine channel to be used on inbound SMTP connections.

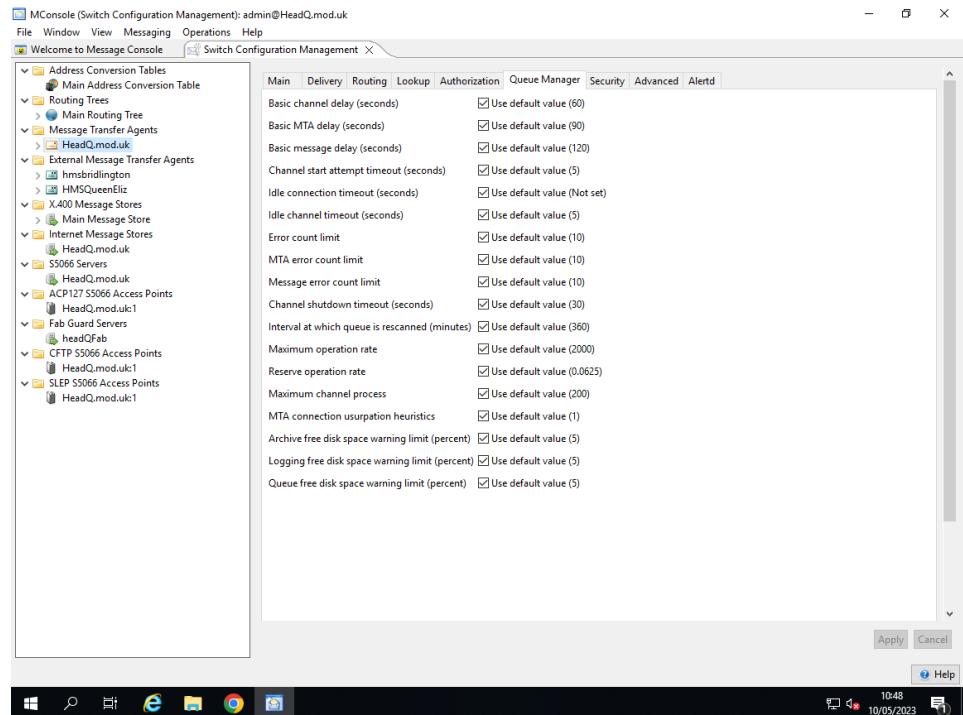
- Block rule to prevent M-Switch acting as an open relay host
- Archive rule which causes all message to be archived as they enter the MTA

For pure X.400 MTAs, only the last of these rules is set up.

See [Section 38.1, “Authorization”](#) for a full description of this feature.

## 14.1.5 Information for Queue Manager

**Figure 14.8. Properties of an MTA (Queue Manager page).**



Items configured in the Queue Manager tab of MConsole configure the way the Queue Manager controls many of the activities of the MTA for which it is responsible.

Values are in seconds unless otherwise specified.

Starting from the top, they are:

- **Basic channel delay:** The base for delays to channels, peer MTAs and messages. The default is 60 seconds.
- **Basic MTA delay:** The base for delays to peer MTAs. The default is 1.5 times the channel delay, e.g. 90 seconds if the channel delay is the default of 60 seconds. Setting this to a non default values overrides the algorithmically derived value
- **Basic message delay:** The base for delays resulting from message errors. The default is 2 times the channel delay, e.g. 120 seconds if the channel delay is the default of 60 seconds. Setting this to a non default values overrides the algorithmically derived value
- **Channel start attempt timeout:** Time to wait before assuming that a process start attempt has failed. The default is 5 seconds.
- **Idle connection timeout:** Time to hold a connection open to a peer when there are no messages. The default is 0. This is the overall default. Channels and MTAs can have their own values.
- **Idle channel timeout:** Time to keep a channel processing running when there is no work for it to perform. The default is 5 seconds.



- **Error count limit:** The basic delay intervals applied to channels, MTAs and messages when errors occur are multiplied by the smaller of the error count and this value. The default is 10.
- **MTA error count limit:** After each MTA error, the delay is calculated by multiplying the basic MTA delay by the error count, until the Error count limit is reached. After the error count reaches the limit, the limit value is used in further delay calculations. The default is 10.
- **Message error count limit:** After each Message error, the delay is calculated by multiplying the basic Message delay by the error count, until the Error count limit is reached. The default is 10.
- **Channel shutdown timeout:** The time the qmgr waits for channels to respond to the instruction to close down. The default is 30 seconds. The qmgr will abort channel processes which take longer than this to respond.
- **Interval at which queue is rescanned:** Time interval between scans of the queue to see messages which may have been missed, in **minutes**. The default is 6 hours (360 minutes).
- **Maximum operation rate:** The maximum operation rate in ops/sec (Default 2000.0). This is the basic replacement for maxchans and is a floating-point value.
- **Reserve operation rate:** The actual operation rate is compared with a value scaled from `qmgr_oprate_max` based on the message priority. It is a percentage value, and floating-point value.

The default is calculated by this function:

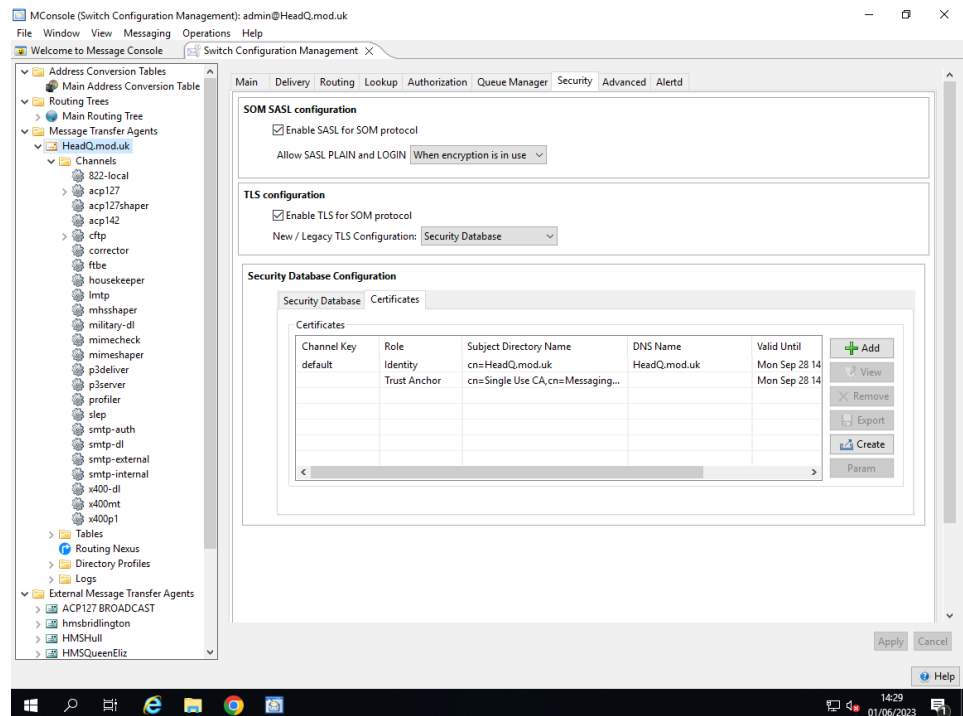
```
scaled = max*(1.0 -0.01*reserve*(priority-lowestprio)/
(highestprio-lowestprio))
```

In effect the processing of lower priority messages is throttled at a lower operation rate than for higher priority messages. `qmgr_oprate_reserve` is the proportion of the overall limit which depends upon message priority.

- **Maximum channel process:** sets the default maximum number of channels that can be run simultaneously. This parameter needs to be set carefully, as a larger number may not increase but decrease performance. The default value is 200.
- **MTA connection usurpation heuristics:** Parameter which controls if the qmgr will use a connection to an MTA for which there are messages with a message for another MTA. It is compared to the difference in priority of the messages for the two MTAs, which includes fraction factors depending on arrival time and message size. A floating-point value, default 1 (which implies that messages of higher message priority will usurp). Increasing this prevents messages of a small priority difference from usurping. Decreasing this value means that smaller, or longer queued messages of the same priority can usurp.
- **Archive free disk space warning limit:** When the available space on the disk to which messages are archived falls below this value (as a percentage), the Queue Manager will generate an Error-level event.
- **Logging free disk space warning limit:** When the available space on the disk to which logs are written falls below this value (as a percentage), the Queue Manager will generate an Error-level event.
- **Queue free disk space warning limit:** When the available space on the disk on which the message queue is held falls below this value (as a percentage), the Queue Manager will generate an Error-level event.

## 14.1.6 Security settings

**Figure 14.9. Properties of an MTA (Security page).**



This page configures the way in which M-Switch uses TLS as part of authentication. This includes:

- SOM client connections to the Queue Manager
- SMTP client connections to the SMTP Server

The Isode Security Database acts as a permanent repository in which Certificates, Public and Private Keys and TLS configuration options for use by the Queue Manager and SMTP server can be stored. The database is held on the system on which M-Switch is installed. It is accessed by SOM clients which make requests over a SOM connection to the M-Switch Queue Manager allowing them to access the Security Database remotely. These requests allow certificates, keys to be loaded, viewed and deleted and for TLS parameters to be edited.

A legacy option in which a specific set of security variables can be configured is available, but this feature is obsolescent.

The security features on this Tab are as follows:

- **SOM SASL Configuration:** The **Enable SASL for SOM protocol** checkbox can be used to enable the support of SASL in the SOM protocol, which is used, among other programs, by MConsole.

**Allow SASL PLAIN and LOGIN:** This allows the administrator to select when to allow SASL PLAIN and LOGIN in the SOM protocol. The options are: 'Never', 'Always' and 'When encryption is in use'.

- **TLS Configuration:** If the **Enable TLS for SOM protocol** checkbox is selected it configures M-Switch to support use of TLS for SOM connections, via the STARTTLS command.

**New / Legacy TLS Configuration:** This widget allows the administrator to select the type of TLS configuration: If the value is 'Legacy TLS Configuration' the corresponding

values will be available to be edited below. If it is 'Security Database' then the Security Database parameters will be displayed instead.

- **Security Database:** when this option is chosen, MConsole will connect to the Queue Manager using the SOM protocol to access a Security Database. The Security Database is a permanent repository for Certificates, Public and Private Keys and TLS configuration options. This is described in [Section 14.1.6.1, “Setting Up The Security Database”](#).
- **Legacy TLS Configuration:** enables configuration of the TLS via the legacy variables, as described below [Section 14.1.7, “Legacy TLS Configuration”](#). The Legacy Variables are obsolescent and will be removed in a future release.

### 14.1.6.1 Setting Up The Security Database

The high level steps needed to setup the Security Database are as follows

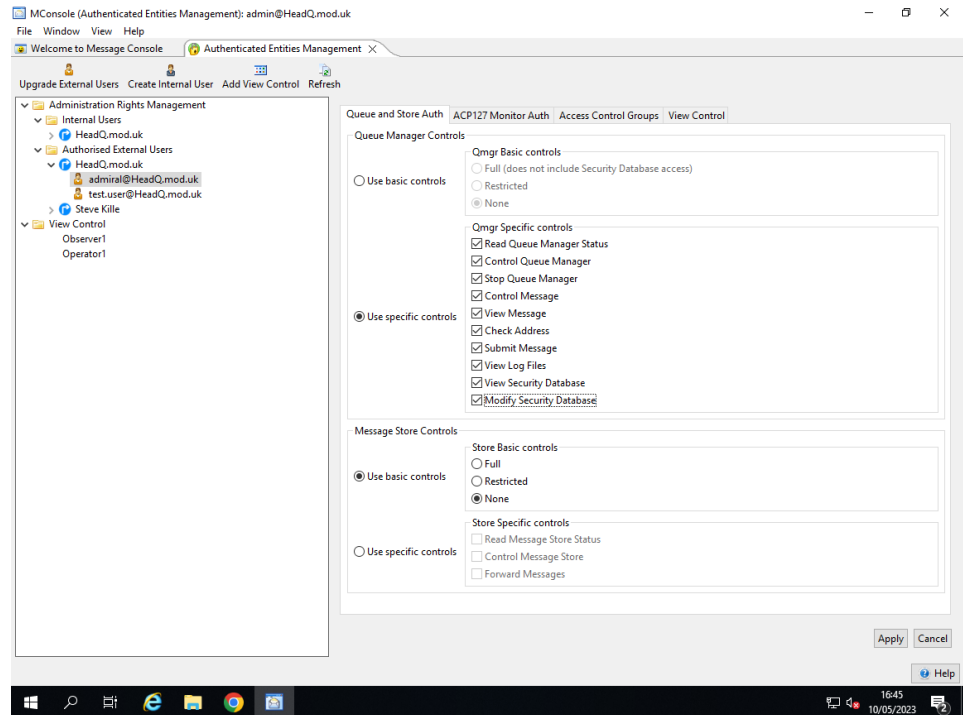
- **Configure a SOM client with privileges to create/modify the Security Database:** see [Section 14.1.6.1.1, “Create SOM Client”](#) for a detailed description.
- **Create the Security Database:** see [Section 14.1.6.1.2, “Create Security Database”](#) for a detailed description.
- **Digital Identities and CA certificates :** see [Section 14.1.6.1.3, “Digital Identities and Trust Anchors”](#) for a detailed description.
- **Testing TLS Access :** see [Section 14.1.6.2, “Testing and evaluation”](#) for a detailed description.

#### 14.1.6.1.1 Create SOM Client

The Security Database is held on the system on which the **qmgr** runs. This means that other M-Switch programs such as channels, can access the contents.

To access the Security Database, you will need to have a SOM connection to your Queue Manager configured (via the Switch Operations View), connecting with a SASL ID which has the **View Security Database** and **Modify Security Database** Queue Authorization rights.

By default the SASL ID created when the Messaging Configuration is created does not have the **View Security Database** and **Modify Security Database** Queue Authorization rights enabled. If that's the case, you will need to use the **Authenticated Entities Management View** to edit the User's **Queue and Store Auth** Tab. You will need to change the **Use Basic Controls** to **Use specific controls** and include the **View Security Database** and **Modify Security Database** Options. See the figure below.

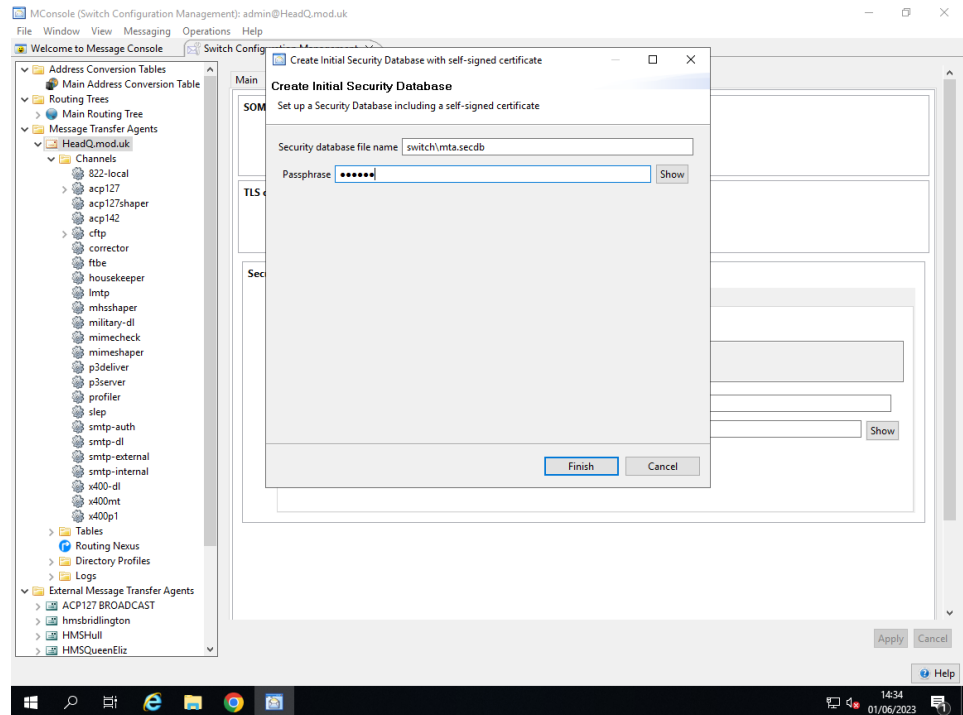
**Figure 14.10. Editing a SASL ID to provide Security DB privileges**

#### 14.1.6.1.2 Create Security Database

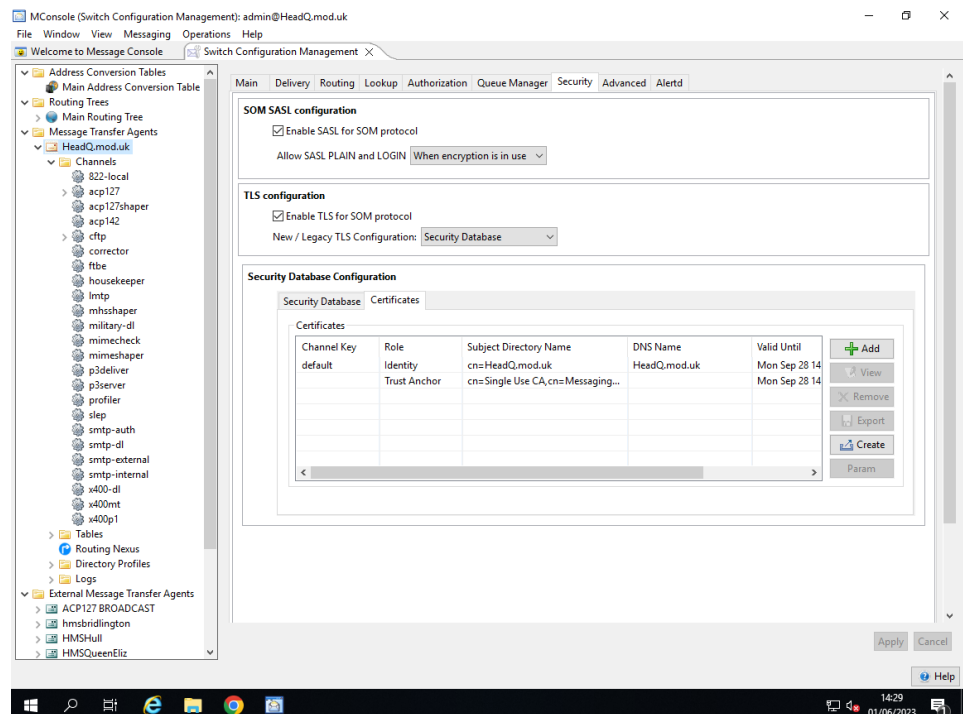
Once you have configured a User in the **Authenticated Entities Management View** with privileges which allow Security Database modifications, and ensured that a SOM connection exists using this User in the **Switch Operations View** it should be possible to create the initial security database as described below.

If you have edited the User for an existing SOM connection in **Switch Operations View**, you will need to Disconnect and Connect in the **Switch Operations View** in order to acquire the additional privileges.

If a Security Database does not currently exist, the Operations menu for the Switch Configuration Management view will include a "Create Initial Security Database" option. This launches a wizard which prompts for the Security Database file path and passphrase, and then causes a new Security Database to be created.

**Figure 14.11. Creating an Initial Security Database.**

The wizard also creates a Certification Authority (with its own self-signed certificate) and uses this to generate a TLS identity for M-Switch. Both the TLS identity and CA certificate are then loaded into the new Security Database, enabling the use of TLS-encrypted connections to the Queue Manager and SMTP server.

**Figure 14.12. Populated Security Database**

### 14.1.6.1.3 Digital Identities and Trust Anchors

TLS servers (i.e. SMTP or SOM) require an 'identity' – a private key and a certificate signed by a certification authority containing the corresponding public key. For SMTP,

the identity is shared among the inbound channels (the identity is a property of the SMTP server process).

TLS clients, such as outbound SMTP channels, do not require a TLS identity to be able to use a TLS-encrypted connection, although one may be supplied if required by the TLS server to which the connection is being made.

---

**Note:** Key usage requirements for a TLS client are different from those of a TLS server. Using default types of certificate, you cannot use the same identity for the SMTP client as the SMTP server.

---

#### 14.1.6.1.4 Trusted Certification Authorities

When a TLS end point receives a certificate, it needs to check that the issuer of that certificate is trusted. Certificates for the trusted certification authorities (CA) are held in the Security Database or in a file (for Legacy configuration): there is no built-in list of these trusted certificates. If legacy configuration is being used, the file contains the sequence of certificates in PEM format.

A TLS client will need to have the CA certificate for the servers to which it is connecting, as servers always send a certificate. A TLS server will need to have the CA certificates for any TLS clients that connect and send their certificate in the TLS handshake.

#### 14.1.6.1.5 Obtaining digital identities and CA certificates

Live deployments need to use a Digital Identity and CA certificate from a proper Certification Authority. The procedure for this is outside the scope of this document. Once obtained these should be installed as described in [Section 14.1.6.3, “Security Database Configuration”](#).

Prior to obtaining Digital Identity and CA certificate from a proper Certification Authority, you may wish to test the use of the Security Database. See [Section 14.1.6.1.2, “Create Security Database”](#) for a description of how to set up the Security Database to set the internally-generated TLS identity and CA certificate for test purposes.

### 14.1.6.2 Testing and evaluation

For testing purposes, the internally-generated TLS identity and CA certificate as described in [Section 14.1.6.1, “Setting Up The Security Database”](#) may be used, or a new TLS Identity can be generated by creating a Certificate Signing Request as described above, importing this CSR into SodiumCA and using this to generate a Digital Identity. This can then be imported into the Security Database.

---

**Caution:** Do not use the initial setup of the Security Database with a self signed certificate for a live deployment. It is for testing and evaluation only.

---

The use of Sodium CA to generate a Digital Identity is discussed in the *M-Switch Advanced Administration Guide*, the *M-Vault Administration Guide* and the *Strong Authentication Evaluation Guide*.

After restarting M-Switch, the smtpserver should now report successful initialisation, as shown in [Example 14.1, “Fragment of log file showing successful SMTP TLS initialisation”](#).

#### Example 14.1. Fragment of log file showing successful SMTP TLS initialisation

```
12/ 7 09:58:25 smtpsrvr 21078 (root      ) N-MTA_SMTP-TLS TLS: \
      initialized
```

### 14.1.6.2.1 Testing with Openssl Test Utility

You can now test the SMTP server's use of TLS using the openssl (installed as isode\_openssl) as a command line client:

```
/opt/isode/bin/isode_openssl s_client -starttls smtp \  
-crlf -showcerts -connect <hostname>:587
```

This will connect to the M-Switch SMTP server running on the host, perform a starttls and the necessary handshaking required up to the point at which the SMTP command EHLO is required.

You can expect that the following will be reported (NB output truncated for clarity):

```
CONNECTED(00000003)  
Can't use SSL_get_servername  
depth=0 CN = hardie.isode.net  
verify error:num=20:unable to get local issuer certificate  
verify return:1  
depth=0 CN = hardie.isode.net  
verify error:num=21:unable to verify the first certificate  
verify return:1  
depth=0 CN = hardie.isode.net  
verify return:1  
---
```

The above output shows that the M-Switch SMTP server Certificate is not trusted by the openssl client. In order to fix this you need to extract the M-Switch CA certificate held in the Security Database, convert it into a suitable form, and then pass the file into the openssl command.

To overcome this you need to extract the self signed CA certificate generated by MConsole from the Security Database by selecting the Trust Anchor of the CA in the Security Tab as shown in [Figure 14.12, "Populated Security Database"](#), and clicking on **Export** to create a Trust Anchor certificate in DER-encoded X.509 form (.crt) in a file with a name such as *ca-certificate.crt*.

Next you need to convert the DER certificate into the PEM form required by openssl:

```
/opt/isode/bin/openssl x509 -in ca-certificate.crt \  
-out ca-certificate.pem -outform PEM -inform DER
```

Note that with multiple trusted certificates, entries in the PEM file can be concatenated.

You can now retest the SMTP server's use of TLS passing the CA certificate as a Trust Anchor to the client openssl (installed as isode\_openssl) as a command line client:

```
/opt/isode/bin/isode_openssl s_client -starttls smtp \  
-crlf -showcerts -CAfile ca-certificate.pem \  
-connect <hostname>:587
```

This will again connect to the M-Switch SMTP server running on the host, perform a starttls and the necessary handshaking required up to the point at which the SMTP command EHLO is required. However this time the M-Switch SMTP server Certificate will be reported as verified by the openssl client but warning that the CA certificate is self signed (NB output truncated for clarity)

```
CONNECTED(00000003)  
depth=1 /C=gb/O=isode/CN=IR Test CA on headquarters.net
```

```
verify error:num=19:self signed certificate in certificate chain
verify return:0
```

The warning indicates that the Certificate was self signed and is therefore not properly trusted. To configure openssl to trust the certificate you need to obtain a Digital Identity and CA certificate from a proper CA as described in [Section 14.1.6.1.5, “Obtaining digital identities and CA certificates”](#). Ensure the CA certificate is in PEM form and pass into the openssl client, the PEM format server certificate reported by openssl into the *ca.pem* file. (With multiple trusted certificates these entries in the PEM file can be concatenated).

Once you have obtained a Digital Identity and a CA Certificate from a proper Certification Authority, run the openssl client again:

```
/opt/isode/bin/isode_openssl s_client -starttls smtp \
-crlf -showcerts -CAfile thawte-ca.pem \
-connect <hostname>:587
```

This time the M-Switch SMTP server Certificate will be reported as trusted as a proper CA by the openssl client: (NB output truncated for clarity)

```
Server certificate
subject=/O=Isode/OU=System/CN=MTA
issuer=/O=Isode/OU=System/CN=CA
---
Acceptable client certificate CA names
/O=Isode/OU=System/CN=CA
/C=ZA/ST=Western Cape/L=Cape Town/O=Thawte
Consulting/OU=Certification Services Division/CN=Thawte Personal
Freemail CA/emailAddress=personal-freemail@thawte.com
```

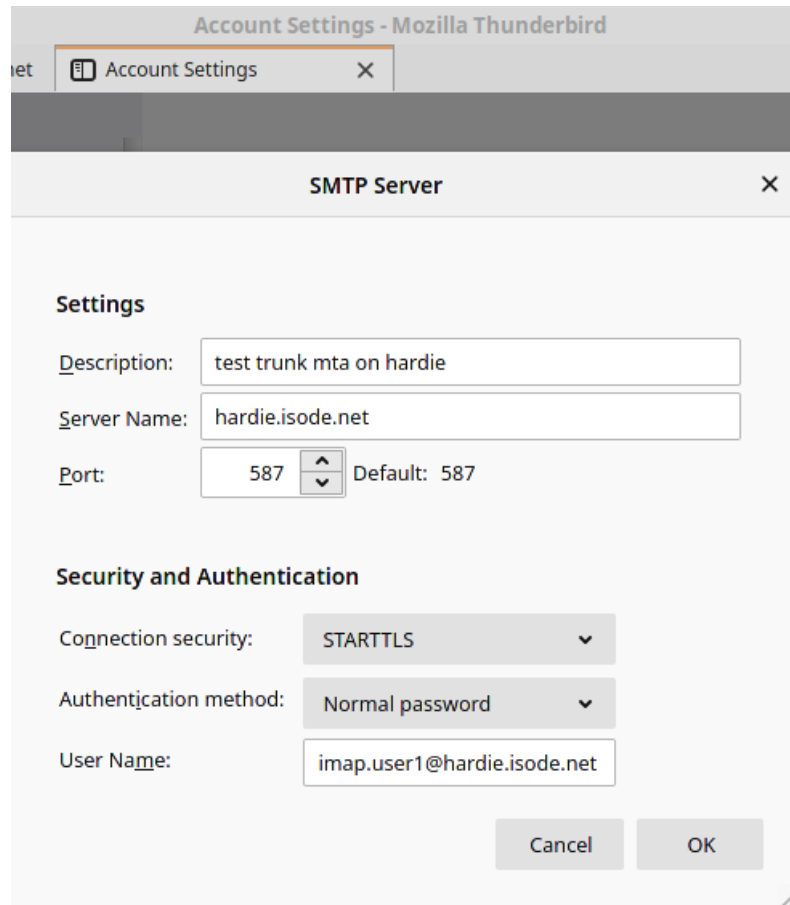
### Example 14.2. Example PEM file

```
-----BEGIN CERTIFICATE-----
MIIByzCCATQCAQAwDQYJKoZIhvcNAQEEBQAuLjEOMAwGA1UEChMFsXNvZGUxDzAN
BgNVBAsTB1N5c3RlbTElMAkGA1UEAxMCQ0EwHhcNMjYxMDA1MTAxMDIxWhcNMjYx
MDAyMTAxMDIxWjAuMQ4wDAYDVQQKEwVJc29kZTEPMA0GA1UECzMGU3lzdGVtMQsw
CQYDVQQDEwJDQTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwGyKCGYEAsoQCD0Gy jVIC
ONFud88rhBWNclXshWwbfhUsMqj/214VuXa2BDROcaAQAmoT0us/T468HZ37xTVJ
xaZDce36AW9zY6j141GVvEKIAoGrSDUOWLY3jL92MA8tDPsel96KVwWloe+0yoZj
Vyva+I05cvzxe+uoltycg+aMOehlW9kCAwEAATANBgkqhkiG9w0BAQQFAAOBgQBr
77U4o0yVBBueq+CmRWmphqQOp5DtWmWXbd3olCKPD9TymqLKObIyOlZpOIFfRzgK
ZoNDi+hC2oQY+kuOSCRbgmL18uivxU2OsPUMfTUMZmA1g/SZBlfxYU0jN/srLMOe
XK8QPruH2unAcwylryKbcdJgvywjygdVZKnU1Z1Q==
-----END CERTIFICATE-----
```

#### 14.1.6.2.2 Testing with an Email Client

Once the tests with the openssl client have been successful, you can use a proper email client to send emails. This example shows how to configure Thunderbird, but other email clients will need very similar configuration.



**Figure 14.13. Configuring Thunderbird**

### 14.1.6.3 Security Database Configuration

The Security Database editor provides the following facilities:

- **Create** launches a wizard which allows a Certificate Signing Request for a new TLS identity to be generated. If the CSR is actioned immediately, the resulting certificate can then be imported using the same wizard. If the CSR is actioned at a later date (e.g. the CSR is sent off via email with the resulting certificate arriving in the same way), the certificate can be imported using the **Import** button.
- **Import** launches a wizard which allows a single TLS identity or one or more Trust Anchors to be imported from local files. TLS identities must be provided as PKCS12 files. Trust Anchors should be contained in .crt or .cer files. When loading a TLS identity you may need to provide a passphrase for decryption.

The selected certificate will not be added to the Security Database until the "Apply" button is pressed.

- **Export** launches a wizard which allows the selected certificate to be exported in encoded X.509 format.
- **Remove** marks the selected certificate for deletion. Deletion from the Security Database will not take place until the "Apply" button is pressed.
- **View** allows the contents of the certificate to be examined, but not modified.
- **Param** allows the contents of the certificate parameters to be examined and edited. It is not available for some certificate types.

Multiple TLS identities may be configured within the Security Database, although in practice only two are used: the "default" identity, which is always used when establishing a TLS-encrypted SOM connection to the Queue Manager, and optionally a second identity with the key "smtp", which may be used by all of the configured SMTP channels.

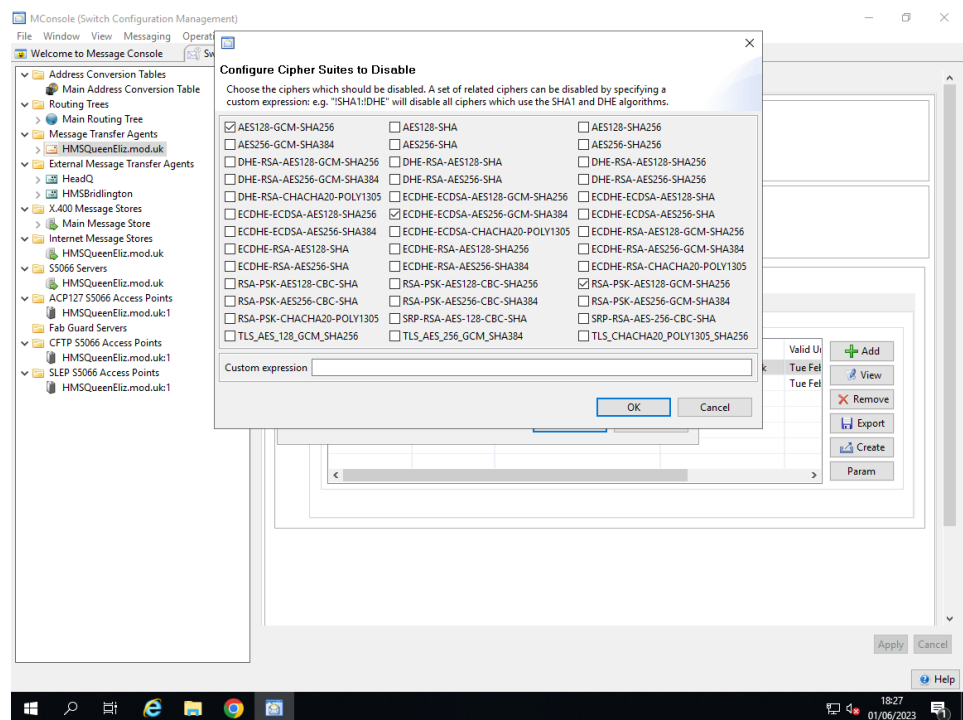
Various parameters can be configured for each of the TLS identities:

- **Client certificate verification mode:** This controls how the TLS server handles certificates which may be supplied by clients. The options are:
  - **Unspecified:** perform default action (verify if present).
  - **Do not verify**
  - **Verify if present:** TLS negotiation will fail if the client certificate cannot be verified.
  - **Required:** a client certificate must be present and verifiable.
- **Suppress TLS v1:** Prevent the use of TLS v1 by the server.
- **Disable Specific Cipher Suites:** This allows a one or more of Cipher Suites to be disabled. Pressing the "Edit" button will produce a list of all of the available Cipher Suites, from which those to disable can be selected. It is also possible to disable sets of Cipher Suites by specifying a custom expression.

There are two actions on certificates that can be accessed by selecting a certificate and then using the right-click context menu. If available, this will display two options: **Copy URI** will copy the selected certificate URI to the clipboard. **Assign TLS Identity** will open an editor that allows the modification of the selected certificate's TLS Identity.

### 14.1.6.4 Cipher Suite List Editor

Figure 14.14. Cipher Suite Editor



The Cipher Suite Editor allows specific ciphers to be selected for deactivation. This means that they will not be used by TLS servers and will not be reported to TLS clients. Any attempt by a TLS client to use the ciphers will be rejected.

A custom expression can also be used to disable a set of related ciphers. For example, the expression "!SHA1:!DHE" will disable all ciphers which use the SHA1 and DHE algorithms.

## 14.1.7 Legacy TLS Configuration

Legacy TLS configuration allows the setup of basic TLS information, using a set of PP internal variables. Note that both the TLS identity path and CA certificate PEM file must be specified in order for TLS encryption to work. The available fields are:

- **Path for TLS identity file:** Path for TLS identity files containing Digital Identities. This is used by both the Queue Manager (for SOM protocol connections) and SMTP server processes. This sets the PP variable `tls_path`. If `tls_path` is not set, it is in the `tls` subdirectory of the configured table directory.

TLS Identity Files can be:

- *rsa.p12* (with passphrase file *rsa.p12.pphr*)
- *dsa.p12* (with passphrase file *dsa.p12.pphr*)
- *ecdsa.p12* (with passphrase file *ecdsa.p12.pphr*)
- *id.p12* (with passphrase file *key.pphr*)

---

**Note:** Passphrase files are only needed if the private key file is passphrase-protected

---



---

**Caution:** Do not put a Carriage Return/Line Feed at the end of a passphrase file.

---



---

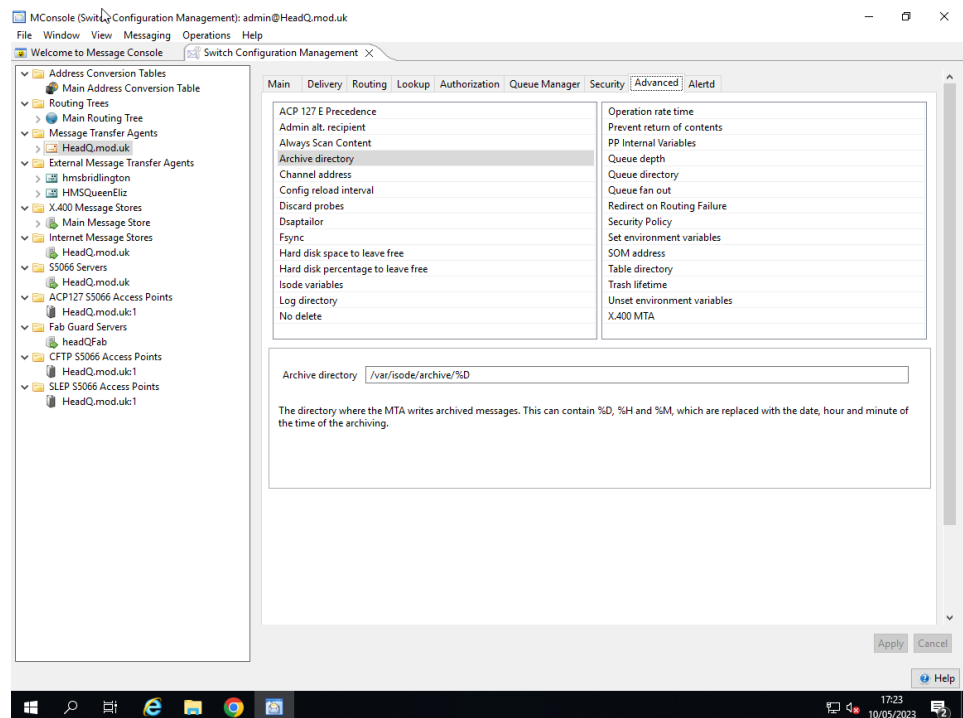
**Caution:** You should ensure that the file permissions, or access control, are such that only the user which is used for the MTA can read these files.

---

- **Name of PEM file containing CA certificates :** This sets the name of the PEM file used to hold the list of trusted Certificate Authorities. This is used by both the Queue Manager and SMTP server processes. This sets the PP variable `tls_cafile`.
- **Client certificate verification mode:** This controls how the Queue Manager and SMTP server handle certificates which may be supplied by SOM or SMTP client applications when they open TLS-encrypted connections. This sets the PP variable `x509_verifymode`. The options are:
  - **Unspecified:** perform default action (verify if present).
  - **Do not verify**
  - **Verify if present:** TLS negotiation will fail if the client certificate cannot be verified.
  - **Required:** a client certificate must be present and verifiable.
- **Available cipher suite:** There is no GUI interface for this, but PP variable `tls_cipher_suites` can be set to a colon-separated list of cipher suites to disable, e.g. `DHE-DSS-AES128-SHA:ECDSA-AES256-SHA`.
- **Disable TLS version 1:** There is no GUI interface for this, but PP variable `suppress_tls_v1` can be set to any value to disable use of TLS version 1.

## 14.1.8 Advanced configuration options

**Figure 14.15. Properties of an MTA (Advanced page).**



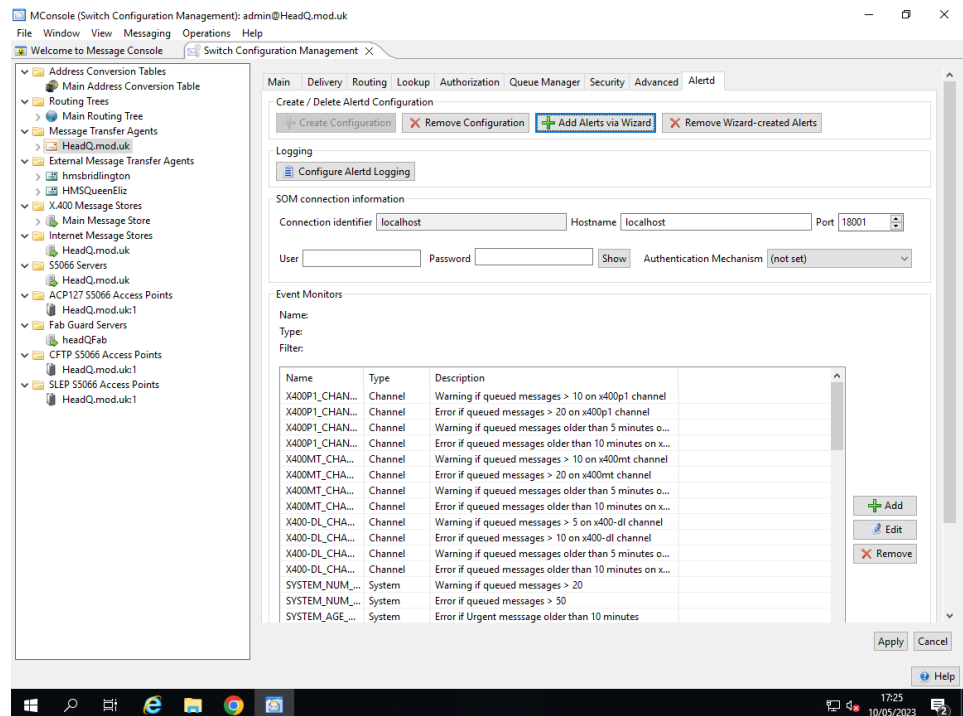
The **Advanced** page allows you to configure various rarely-changed aspects of the MTA.

- **ACP127 E(mergency) Precedence:** Standard ACP127 uses four letters to represent the precedence levels Flash (Z), Priority (P), Immediate (O) and Routine (R). Some environments also use 'Y' for a precedence 'Emergency' which is higher than Flash. When a message is received with 'Y' precedence, this is converted into the precedence 'override'. Normally, precedence 'override' is represented by the same letter as 'Flash'. If 'ACP127 E Precedence' is set, then 'Y' is used for messages of the precedence 'override'.
- **Admin alt recipient(X.400 only).** An alternative recipient address to which wrongly-addressed messages should be redirected.
- **Always scan content** Cause SMTP message entering M-Switch to have their content scanned. This is necessary so that Security Labels other than SIO can be extracted. It is also necessary when verifying S/MIME messages. By default this is disabled as it is a performance overhead.
- **Archive directory:** The directory where the MTA writes archived messages. This can contain %D, %H and %M which are replaced with the date, hour and minute of the time of the archiving.
- **Channel address:** Address on which the qmgr listens for connections from channels. Formatted as `<hostname>:<port>`
- **Config reload interval:** Interval between reloading the qmgr's configuration. This value is in seconds.
- **Discard probes (X.400 only).** Selecting this option will cause incoming Probes to be discarded without the generation of a delivery report. The event will be logged in the stat log.
- **Dsaptailor:** This specifies a Directory Services tailoring file that this MTA uses to connect to the directory. This is not normally required, but may be useful if you wish to override the default logging levels for Directory operations.
- **Fsync:** Force writing data to disk using fsync prior to closing files

- **Hard disk space to leave free:** This field shows how much disk space (in MBytes) should be kept free on the partition holding the MTA's message queue. If both **Hard disk space to leave free** and **Hard disk percentage to leave free** are configured the less permissive is used.
- **Hard disk percentage to leave free:** This field shows what percentage of disk space should be left on the partition holding the MTA's message queue. If both **Hard disk space to leave free** and **Hard disk percentage to leave free** are configured the less permissive is used.
- **Isode Variables:** Isode tailoring variables (overriding those found in *isotailor*).
- **Log directory:** The directory where all the log files go.
- **No delete:** Keeps messages in the queue rather than deleting them when they have been completed.
- **Operation rate time:** The smoothing time for calculation of the operation rate.
- **Prevent return of contents (X.400 only):** Selecting this option for X.400 systems will cause the x400p1 and p3 channel programs not to include any returned content in non-delivery reports. It also cancels the request to return content in messages being transferred to other MTAs.
- **PP Internal Variables:** various parameters used to control the behaviour of the MTA. See *M-Switch Advanced Administration Guide* for a list of those available.
- **Queue depth:** This specifies the level of sub-directories to create. This should not be changed unless extremely large queues are expected (above 50,000).
- **Queue directory:** The directory where messages waiting to be processed by the channels are saved.
- **Queue fan out:** The queue fan out option specifies the number of subdirectories of the main queue. For example, a value of 100 means there are 100 sub-directories of the main queue file. This cuts down the searching of the main queue file when there are very large queues.
- **Redirect on Routing Failure:** Controls whether the administrator-assigned-alternate-recipient address (if set) should be used in the case of routing failure.
- **Set environment variables:** Here you can add environment variables. These can be used to control the behaviour of certain sub-programs. This is described in more detail in the *M-Switch Advanced Administration Guide*.
- **Security Policy:** Filename in ETCDIR or SHAREDIR
- **SOM address:** Address on which the qmgr listens for SOM protocol connections. Formatted as `<hostname>:<port>`
- **Table directory:** This is the directory where all the table files are located
- **Trash lifetime:** Non-message files (e.g. recovery/resumption information) older than this lifetime are deleted from the qmgr's queue. The value is expressed as `<val><units>` where units are one of s (seconds), m (minutes), h (hours) or d (days).
- **Unset environment variables:** Here you can unset environment variables.
- **X.400 MTA:** For X.400 systems, this is the name used to represent this MTA in tracing fields.

## 14.1.9 Alert Daemon

Figure 14.16. Properties of an MTA (Alertd page).



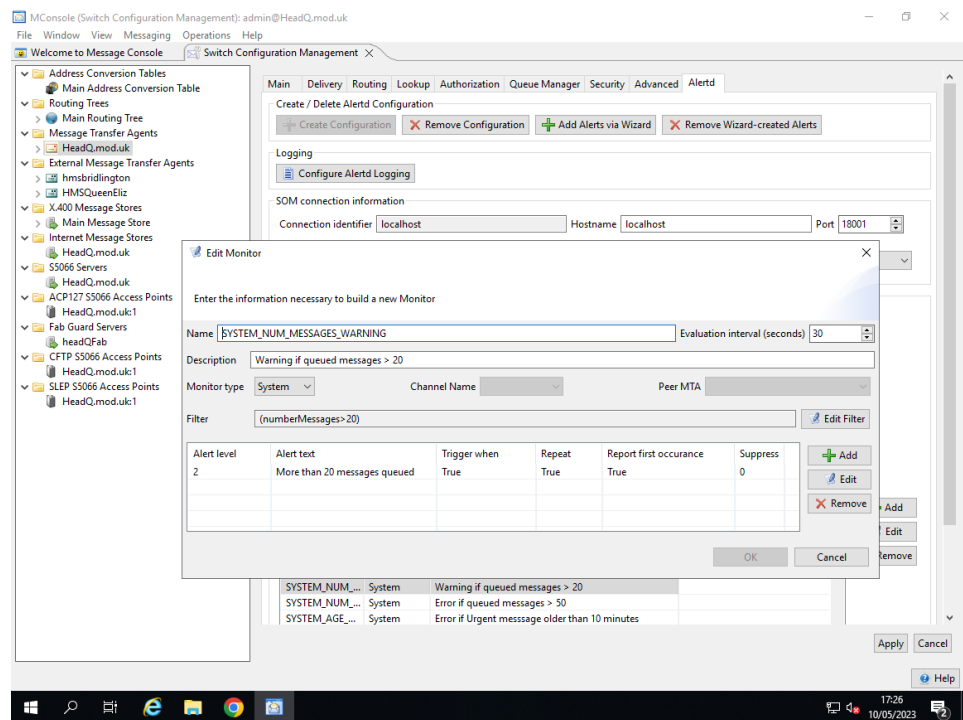
The **Alertd** page allows you to configure the Isode Alert Daemon. The Alert Daemon is used to monitor the MTA, using the SOM protocol. Multiple "Event Monitors" can be set up: each Monitor runs at a pre-set interval and evaluates a filter which tests an aspect of the MTA's performance (for example, the total number of messages queued). Configurable Events can be generated in when the filter evaluates as true (or false). These Events are written to one or more log streams, and can be monitored by MConsole's Alert View.

The pages provides the following facilities:

- **Create Configuration:** if an Alertd Daemon configuration does not currently exist, this creates a minimal configuration framework ready for Event Monitors to be added, plus a default logging configuration which writes events of logging level NOTICE and higher to a log file named "alertd-event.log" which rolls over daily.
- **Remove Configuration:** completely removes the existing Alert Daemon configuration.
- **Add Alerts via Wizard:** this initiates a wizard which will allow you to quickly create a set of Event Monitors which are tailored to your MTA. Choices are:
  - **General Purpose, Aviation or Military:** this selects message age limits which are appropriate to the MTA use.
  - **Channel Alert Thresholds:** this groups channels into three different types: constrained bandwidth channels which can only transfer messages slowly, fast transfer and delivery channels, and internal processing channels. For each group, warning and error message age (i.e. age of oldest message queued on the channel) and volume (number of messages queued on the channel) limits can be set.
  - **System-wide Alert Thresholds:** this allows the configuration of warning and error message age and volume limits which apply across the whole MTA. Separate age limits can be configured for each available priority level.
- **Remove Wizard-created Alerts:** this will remove any Event Monitors which have been created via the Wizard described above, while preserving any other Event Monitors which have been created manually.

- **Configure Alertd Logging:** this launches a pop-up version of the standard Log Configuration Editor which allows the logging configuration for the Alert Daemon to be modified if necessary.
- **SOM connection information:** this area of the page allows the configuration of the information which the Alert Dameon will use when connecting to the MTA (using SOM). Whether an anonymous connection (i.e without **User** and **Password** values) can be used will depend on your MTA's configuration.
- **Event Monitors:** The configured Event Monitors are displayed in a table. Individual Event Monitors can be Added, Edited or Removed.

**Figure 14.17. Edit Event Monitor.**

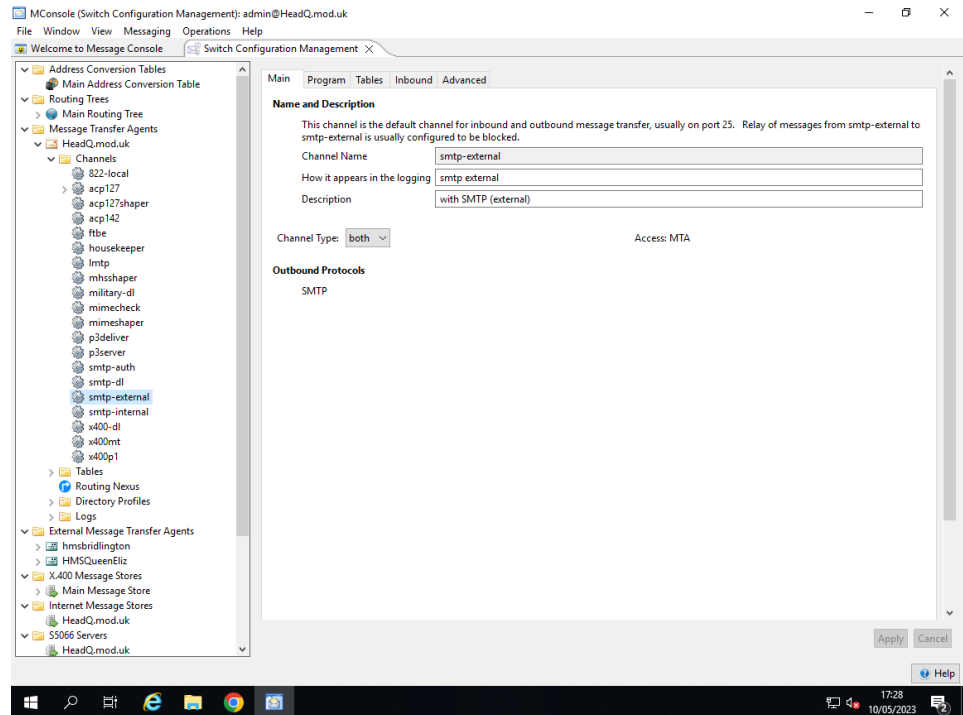


## 14.2 Editing an MTA protocol channel

### 14.2.1 Editing channel properties

Channels are the most complex objects of an MTA configuration. Some channels have configuration options that are not used by others and, in those cases, MConsole will display the values only if it makes sense.

Most channels will have the following pages: **Main**, **Program**, **Tables** and **Advanced**. Protocol channels have, in addition an **Inbound** page, and X.400 channels will have also **Auth** and **RTSE** pages.

**Figure 14.18. Properties of a channel (Main page).**

### 14.2.1.1 Main tab

- **Channel Name:** This is name of the channel currently being edited. This is for display only.
- **How it appears in the logging:** a short phrase describing the channel. If this string starts with the keywords 'with' or 'via' then this value is included in Received lines generated for RFC 822 messages..
- **Description:** You may add descriptive text about this channel here.
- **Channel Type:** this is a choice of the following values:
  - **in:** This channel accepts incoming connections, and messages
  - **out:** This channel creates outgoing connections and sends messages
  - **both:** Allows both incoming and outgoing connections and messages.
  - **shaper:** Typically shaper channels reformat messages, for instance between internet and X.400 messages.
  - **check** Checking channels are used to check messages, for example virus checking.
  - **housekeeper:** The new housekeeper channel amongst other functions, returns messages that have timed out, converts DSNs into DRs and sends a warning back to the originator when a messages is delayed.
- **Outbound Protocols:** This specifies the outbound protocols which this channel provides.

### 14.2.1.2 Program tab

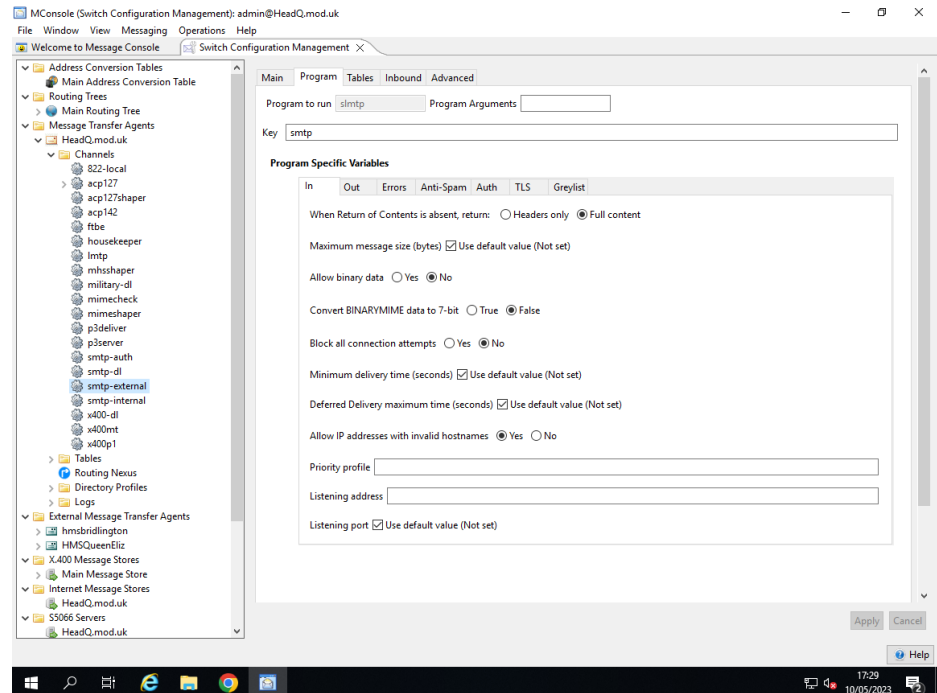
- **Program to run:** The program associated with the channel (the binary that is run).

---

**Note:** All programs need the Program value to be filled apart from Gateway channels which are used by the Isode Gateway API clients to transfer messages in and out of the MTA.

---

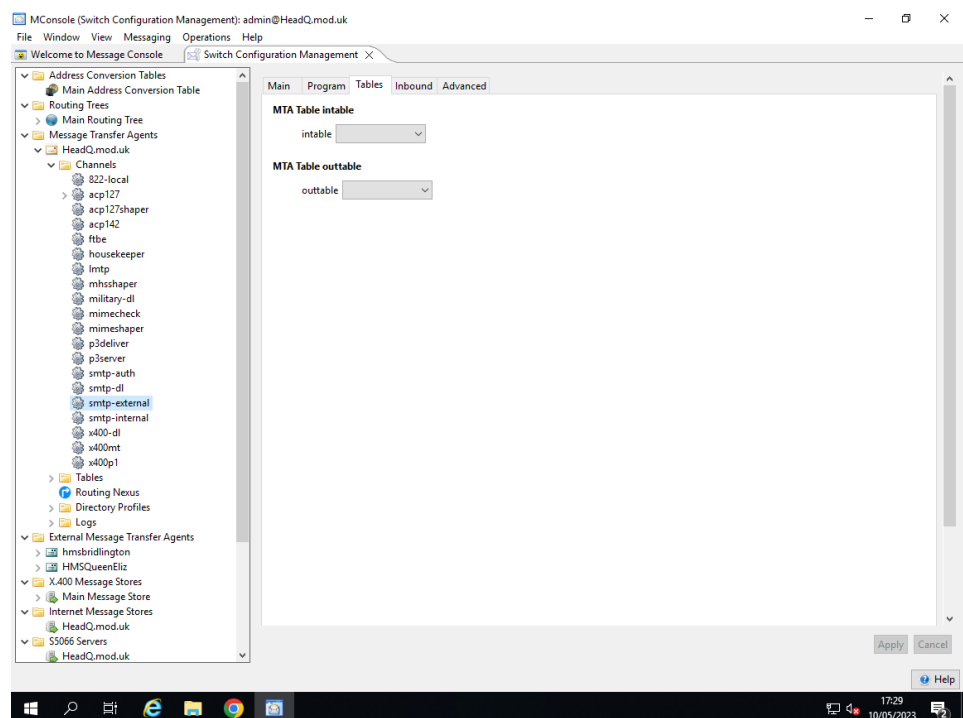


**Figure 14.19. Properties of a channel (Program tab).**

- **Key:** A list of keys (comma separated) by which this channel is known. This can be used to map several logical channels onto one.

Below the **Key** field are some attributes that vary between channels called **Channel Specific Variables**. You will find more information about these variables under the documentation of the specific channel.

### 14.2.1.3 Tables tab

**Figure 14.20. Properties of a channel (Tables tab).**

- **intable:** A table associated with the inbound part of this channel

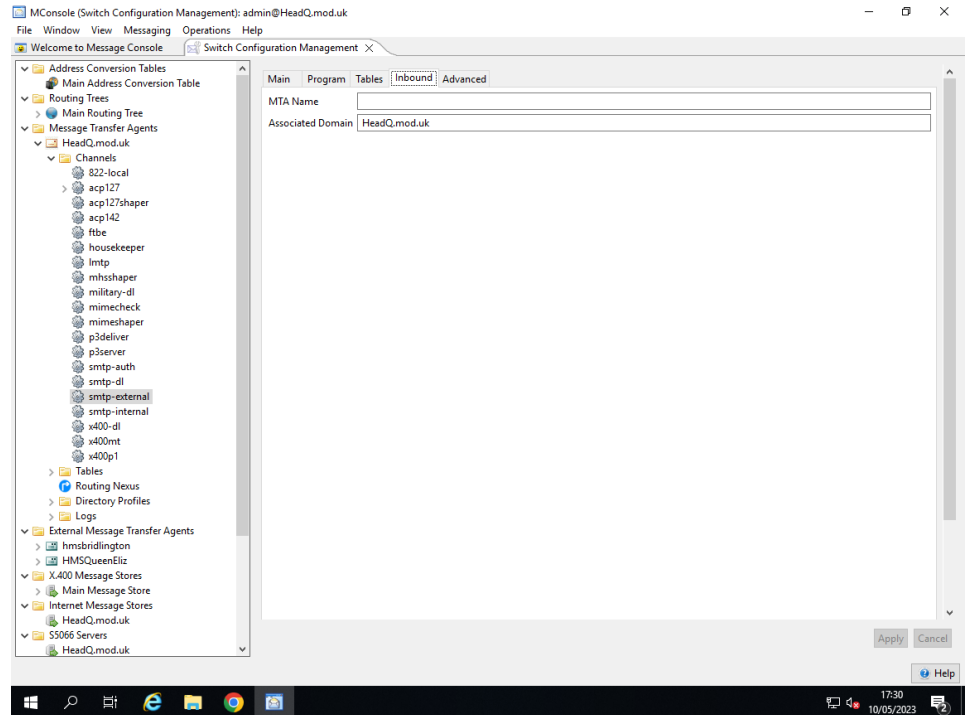
- **outtable:** A table associated with the outbound part of this channel

#### 14.2.1.4 Inbound tab

The contents of the Inbound tab will vary depending on the type of channel being configured.

Because of this, some of the fields listed below may not be present for a specific channel instance. The whole tab may be absent if there are no fields which are relevant to the channel being edited.

**Figure 14.21. Properties of a channel (Inbound tab).**



##### 14.2.1.4.1 Inbound tab: SMTP attributes

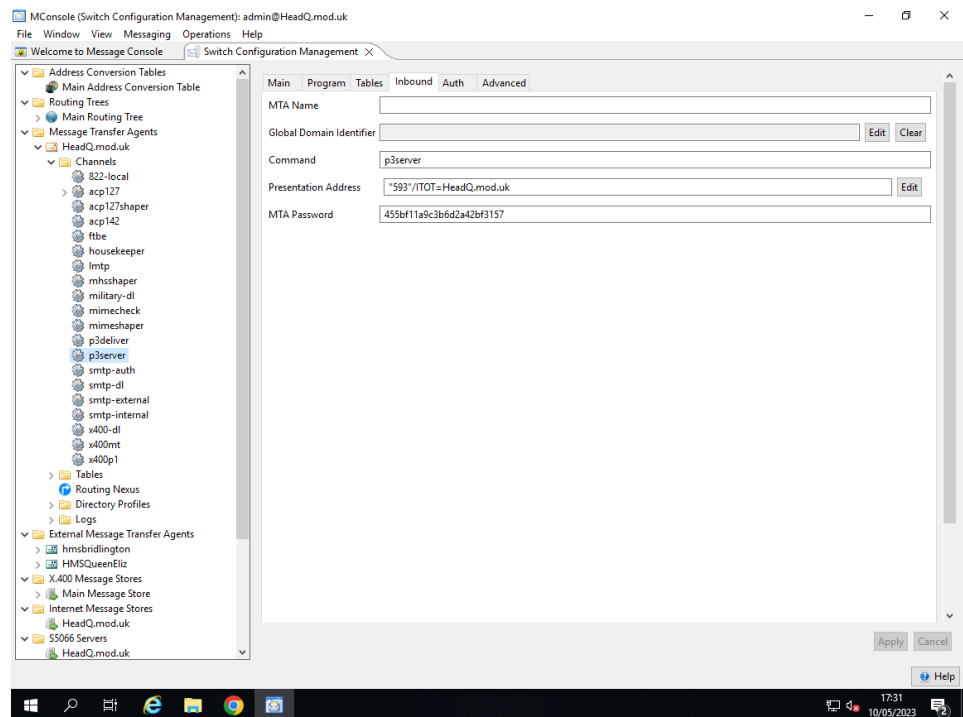
- **Associated Domain:** For Internet Routing, gives the hostname to connect to when attempting to transfer messages to this MTA.

##### 14.2.1.4.2 Inbound tab: common X.400 attributes

- **MTA Name.** A destination MTA for this channel. If this is set, all messages will be delivered to this MTA regardless of the destination MTA given in the message. This is useful for relaying all messages for a given channel via another MTA.
- **Global Domain Identifier:** This is used when verifying a Strong Authentication token, if configured.
- **Command:** The executable program to be run to handle the inbound connection on the configured Presentation Address.
- **Presentation Address:** The address on which the channel will listen for incoming connections.

### 14.2.1.4.3 Inbound tab: X.400 P3 attributes

**Figure 14.22. Properties of a channel (Inbound tab for the p3server channel).**

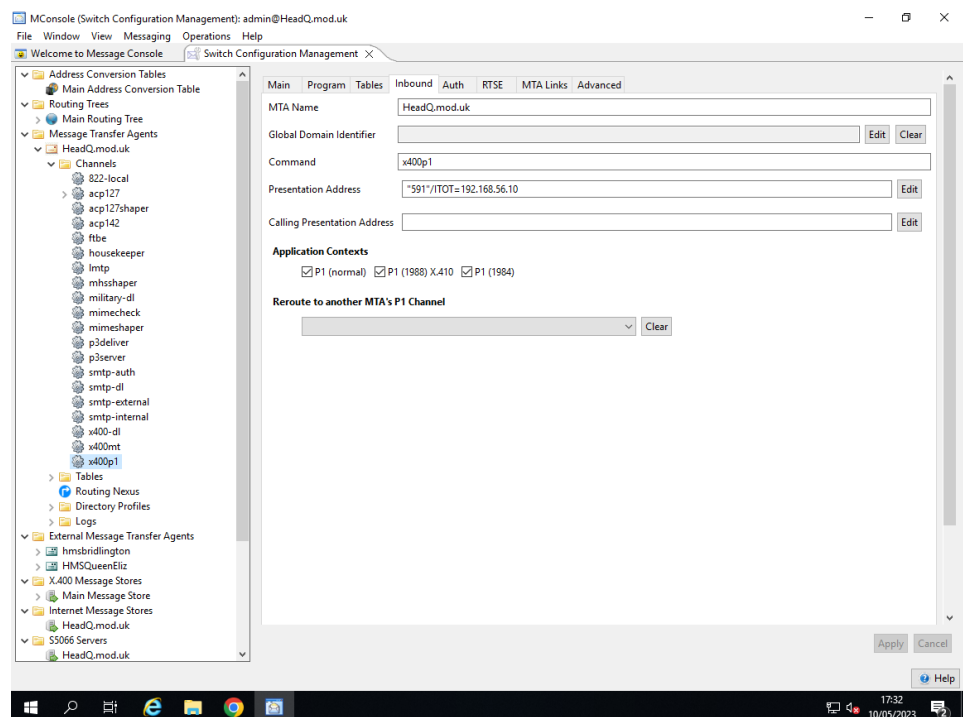


- **MTA Password:** The password this channel will return in a Bind Response when an incoming association is accepted.

### 14.2.1.4.4 Inbound tab: X.400 P1 attributes

This page allows you to configure miscellaneous properties of the x400p1 channel

**Figure 14.23. Properties of a channel (Inbound tab for the x400p1 channel).**



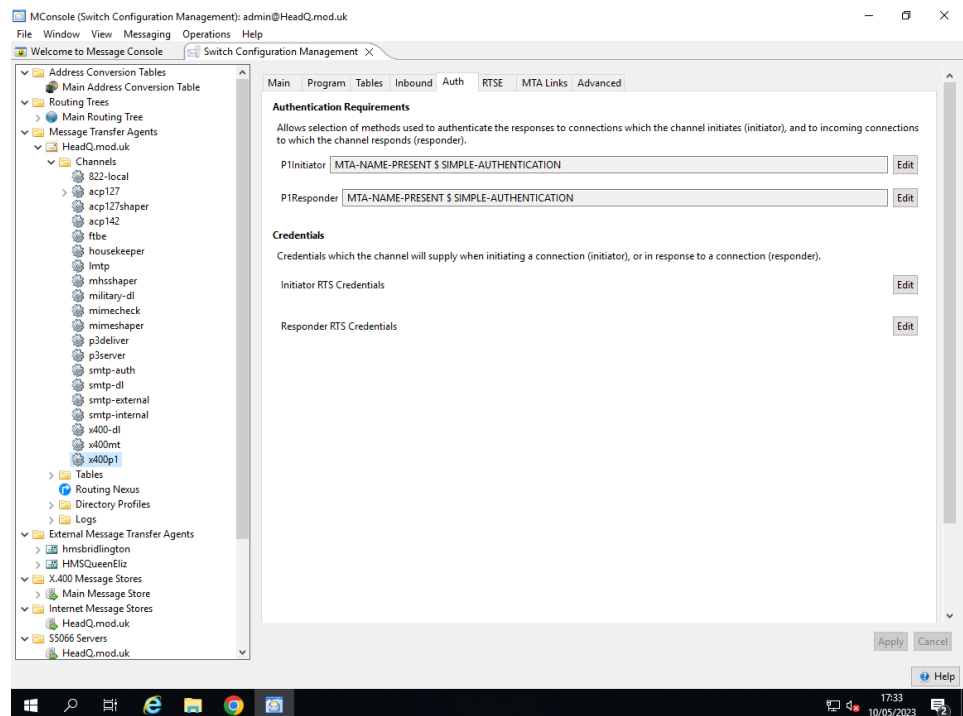
- **Calling Presentation Address:** The Presentation Address which the channel should use in a response to an incoming connection, and which the channel will use in outgoing connection attempts.
- **Application Contexts:** The Application Contexts which the channel supports. The available contexts are: **MTS Transfer**, **MTS Transfer Protocol**, and **MTS Transfer Protocol 1984**. It configures the ways in which the X.400 P1 channel can connect out to an external MTA
- **Reroute to another MTA's P1 Channel.** This sends all messages that would be routed to this MTA to another MTA's P1 channel. You cannot reroute to an MTA/channel which is itself rerouted.

### 14.2.1.5 Auth tab

This tab only appears on P1 and P3 channels.

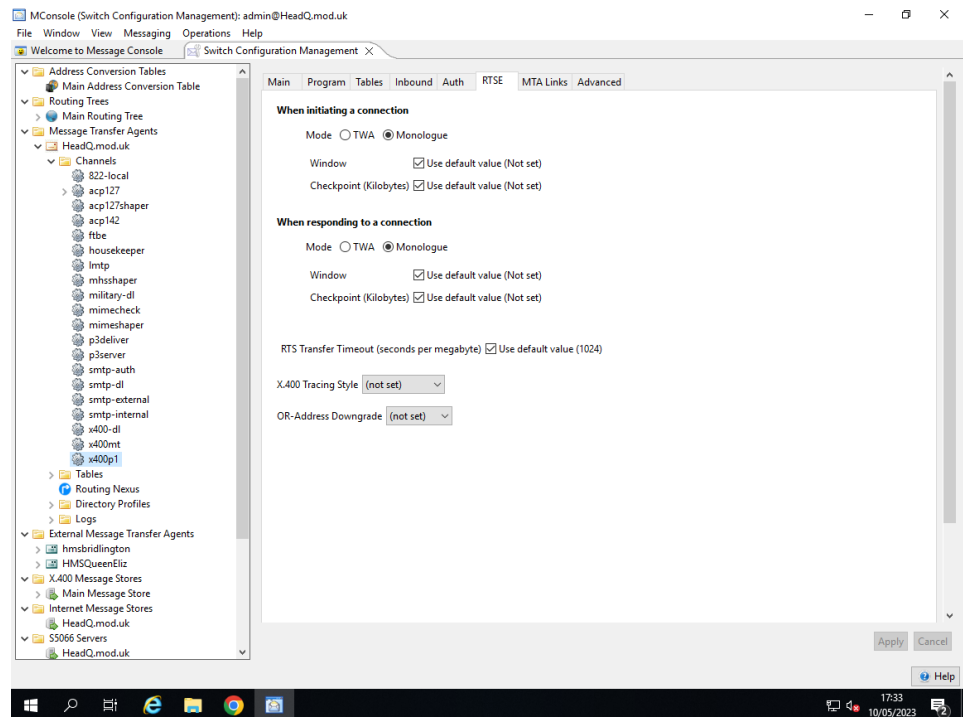
- **Initiator Authentication Requirements:** Allows selection of methods used to authenticate the responses to connections which this channel initiates.
- **Responder Authentication Requirements:** Allows selection of methods used to authenticate incoming connections to which this channel responds.
- **Initiator RTS Credentials:** These are the credentials which the channel will supply when initiating a connection.
- **Responder RTS Credentials:** These are the credentials which the channel will supply when responding to an incoming connection.

**Figure 14.24. Properties of a channel (Auth tab for the x400p1 channel).**



### 14.2.1.6 RTSE tab

Figure 14.25. Properties of a channel (RTSE tab for the x400p1 channel).



The following settings are used by the channel when initiating a connection to, or responding to a connection from, another MTA. The first 3 options have separate initiator and responder equivalents. The last 3 options apply to both initiator and responder.

- **Mode:**
  - **Two-way alternate** means that messages are sent and received on the same connection
  - **Monologue** means that once connected, messages are sent only one way: from the Initiator to the Responder.
- **Window:** This is the number of checkpoints to go through before acknowledgement is sent. If there is a failure in sending information across a network, the connection will be able to resend from the last checkpoint.
- **Checkpoint:** This defines the number of kilobytes in a 'checkpoint'.
- **RTS Transfer Timeout:** The value is an integer, whose units are seconds per kilobyte, i.e. the actual timeout used is based on the size of the message. If message transfer fails because of the timeout, then the message will be tried again.
- **X.400 Tracing Style:** the style of tracing information to include in the envelope of messages transferred out by the channel. This is described in more detail in the *M-Switch Advanced Administration Guide*.
- **O/R Address Downgrade:** The content is downgraded according to *ISO/IEC DISP 12072-1 Annex C*.

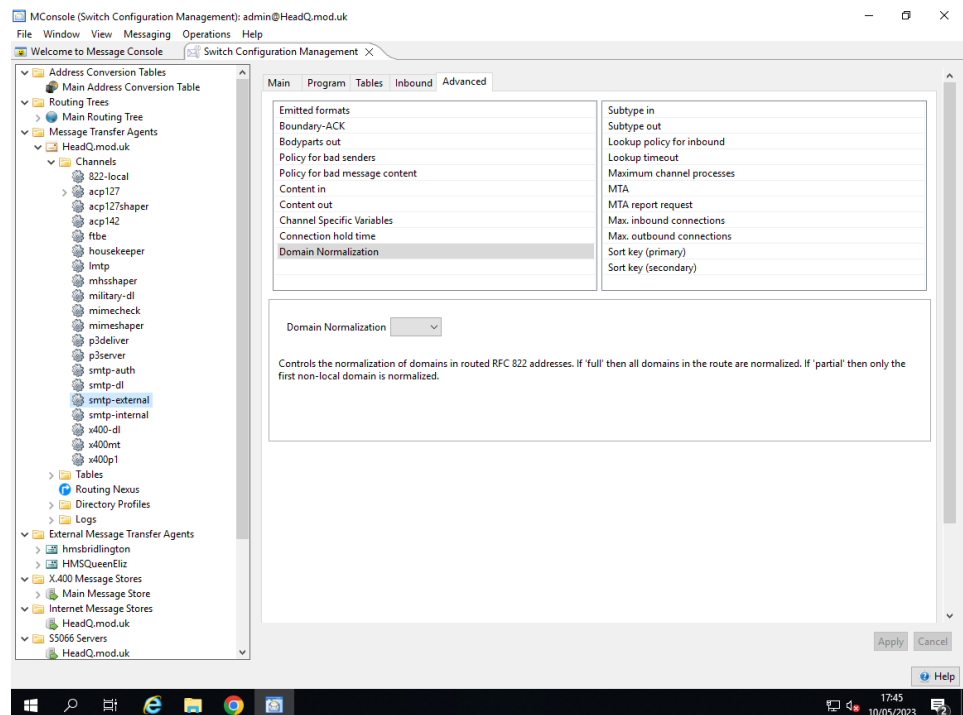
### 14.2.1.7 MTA Links tab

The **MTA name Links** provide the MTA with a way of mapping between the mtaName supplied in the Bind Parameters when an external MTA attempts to connect and the entry in the Messaging Configuration which contains the authentication information for that external MTA. The set of links can be (re)generated by clicking **Generate**: entries can also be added and deleted manually.

### 14.2.1.8 Advanced tab

This tab allows you to configure miscellaneous properties of the channel.

**Figure 14.26. Properties of a channel (Advanced page).**



## 14.3 Creating a new channel

To create a new channel, you have to run the **New Channel Wizard**

The instructions below are to create a new MTA channel called `smtp-internal2`.

If you want to create a Gateway channel, you can follow the instructions in [Section 16.3.2, “Adding a new gateway channel to an MTA”](#).

If you are creating a different kind of channel, you can easily adapt these instructions, as most of the wizard pages will look very similar. You can also clone an existing channel that is similar to the one you want to create. For that, see [Section 14.4, “Cloning an MTA channel”](#).

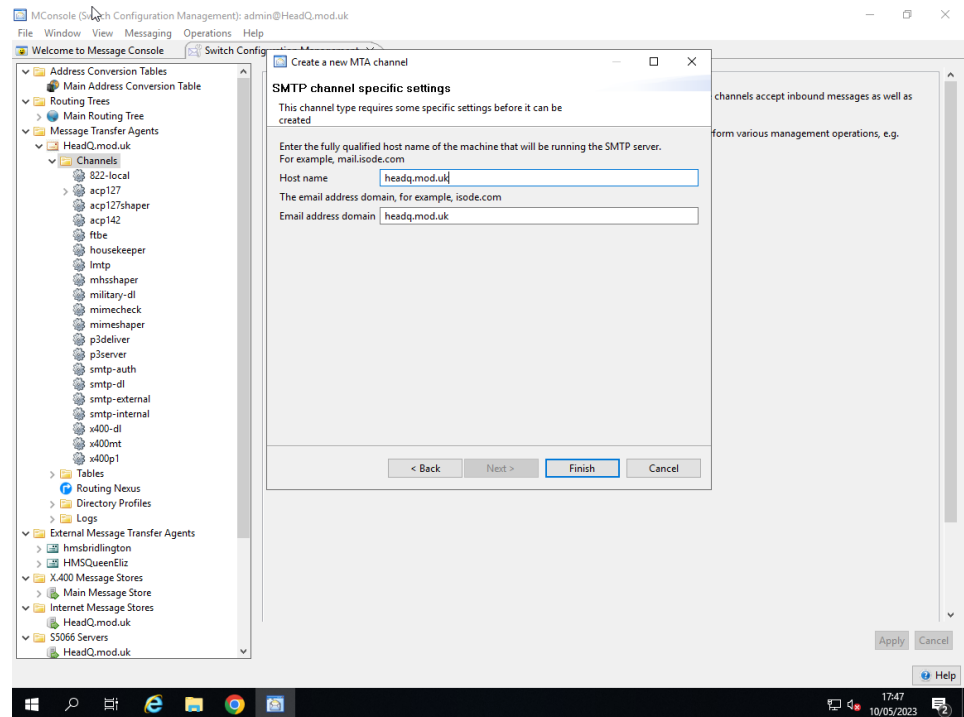
### 14.3.1 Creating an SMTP channel

To create a new channel, start the **New Channel Wizard** by right clicking on the **Channels** folder and selecting **New channel**.

This wizard page allows you to configure the properties of the SMTP channel being created.

The first page of the New Channel Wizard offers a choice of channel types to choose from. For this case, select the **SMTP** radiobutton, enter `smtp-internal2` in the **Channel name** field, and click on the **Next>** button.

In the following page, you will be prompted to enter the Host name and the Email address domain. The values you will be prompted for will depend on the channel type.

**Figure 14.27. Creating an SMTP channel.**

## 14.4 Cloning an MTA channel

Another way to create a channel is to select a similar channel and ‘clone it’, that is, you create a copy of the original channel with another name. You can then edit the new channel’s attributes.

To clone a channel, select it, right-click on it, and choose **Clone**. In the new window, type the new channel’s name. You can then edit the channel configuration after it has been created.

## 14.5 Configuring logging

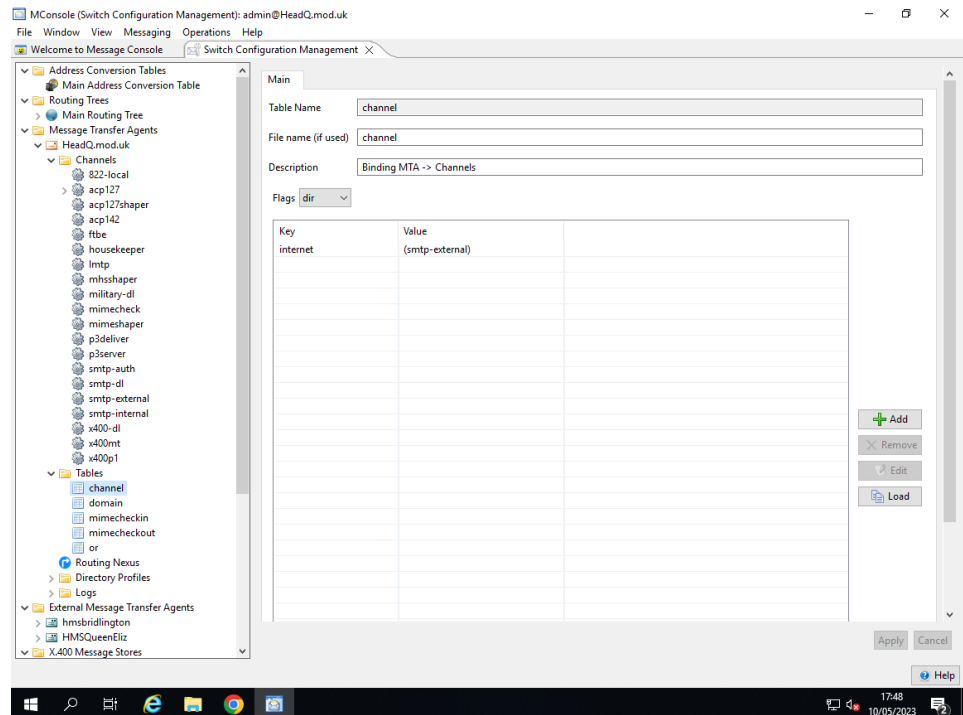
By default logging is configured to generate an audit log and an event log. These logging streams are written to the mta-audit.log and mta-event.log respectively. Also configured is an oper log which you can configure to write to the system log.

You can configure the logging in many different ways to help you to operate M-Switch. This is described in detail in [Section 34.2, “Configuring logging”](#).

## 14.6 Editing an MTA table

To edit an MTA table, expand the **Tables** folder and select the table you want to edit. The table's properties are displayed on the right. The field **Table Name** is read-only in the displayed properties – to change its name, right-click on the table entry on in the left page and select **Rename** from the menu displayed.

**Figure 14.28. Configuring a table (Table ID page).**



- **File name (if used):** Indicates the name of the file that contains this table. It can be blank, as not all tables have files associated with them. The filename can be a relative or absolute pathname and does not have to be the same as the table name. If a relative pathname is given, it will be taken to be relative to the value of **Table directory**, which can be found in the properties window for the MTA, on the **Advanced** page.
- **Description:** Allows you to specify a descriptive string to be used when printing messages about this table (mainly for logging purposes).
- **Flags:** Defines how this table operates:
  - **dbm** (held in text files, converted into a database for efficient reading)
  - **linear** (held in text files and read as text files)
  - **dir** (downloaded from the Directory into files and converted into database for efficient reading)
  - **empty** (does not exist and are not read)

Table overrides can be used to have entries in tables which are otherwise 'empty'. However, note that table overrides cannot be set from MConsole.

The **Table Entry** page allows you to create table entries. It is only used if **Flags** are set to **dir**.



The names of the fields are self-explanatory: **Key** is the key of the entry, **Value** is its value, and the buttons **Add** and **Delete** are used to add and delete entries from the table. What is actually created is an entry on the table with `<key> : <value>`

---

**Note:** You must click **Add** after entering information on the **Table Entry** or **Table Override** pages to add the information to the table. After the changes have been added to the table, click **Apply** to apply them.

---

## 14.7 Storing Credentials in Config Files

M-Switch has a number of configuration files which can include passwords. In order to avoid these being held in the clear, the servpass mechanism allows them to be encrypted.

### 14.7.1 Service Key facility

The following passwords are able to be stored using the Service Key facility to provide password encryption:

- Directory passwords in `mtaboot.xml`
- Directory passwords in `mtatailor.tai`
- Other passwords in `mtatailor.tai`
- X.509 passphrases used to protect private keys in Digital Identities
- Alertrd configuration. See [Section 42.3, “File-based Configuration”](#).
- Message Store configuration held in `pumicetailor.xml`. See [Section 10.3.2, “Service Keys”](#).

The Service Key is used to encrypt and decrypt the passwords. It is stored in `ETCDIR/servpass`. It is important to keep this directory properly protected.

### 14.7.2 Creating a Service Key

Use MConsole to create a Service Key for your MTA.

You need to run MConsole on the system on which your Message server (M-Switch or M-Store) will actually run.

Open the Switch Configuration View and select the MTA for which you wish to create the Service Key. The right-mouse menu for this object includes options for creating and deleting Service Keys.

Enter a passphrase, which will be used to encrypt the Service Key.

### 14.7.3 Using the Service Key facility

Use MConsole to create a new `mtaboot.xml` file for your MTA enter the passphrase which you entered in the previous step, and passwords will be encrypted using the Service Key before they are written into the `mtaboot.xml` file.

A password which has been encrypted in this way is prefixed with “`{spcrypt3}`” when stored. OS-specific mechanisms are used to protect the Service Key i.e. file/directory user permissions.

# Chapter 15 Routing

This chapter describes how Routing works in M-Switch, listing the different Lookup Policies which can be configured and how they work.

ACP127 Routing is not covered in this section. See [Chapter 19, ACP127](#) for a description of ACP127 Routing.

---

## 15.1 Routing Overview

The purpose of routing is to determine for each recipient of a message

- if the MTA performing the routing can deliver to the user, or the user's message store.
- otherwise find the peer MTA to which it should transfer the message. This MTA is the next-hop in the route of the message.

Each recipient of a message undergoes routing, and results in a sequence of MTA/channel pairs being identified. This sequence is ordered by Routing Weight in which the lowest weight is first in the list. The sequence of MTA/channel pairs are tried in order. This provides Routing with the ability to attempt a fallback if a connection cannot be made. If two weights are equal, the order of these MTAs is randomised. This provides load balancing.

The sequence of MTA/channel pairs can also be reduced by using Authorization to remove certain routes which are prohibited if certain conditions are met, e.g. size, content type, Security Label/Clearance. See [Section 38.1, "Authorization"](#) for a description of the M-Switch Authorization features.

---

## 15.2 Routing Information Repositories

The M-Switch Routing procedures use repositories of information to determine how to relay or deliver a message. The way in which these repositories are accessed is configured by the Lookup Policies configured in M-Switch.

The repositories of Routing Information can be part of the M-Switch Configuration, or may be managed externally. The following lists these repositories.

### Routing Trees

Routing information can be held in Routing Trees. Routing Trees are hierarchical structures held in the Directory, and consist of routing information held in nodes in a tree like structure. Each node represents one component of an X.400 O/R Address or Internet address. Each node may contain routing information which describes the way in which matching addresses are routed. Routing involves taking an O/R Address or Internet Address, converting into a DN, and looking up in a routing tree. The *longest match* is used and information from the matched node in the Routing Tree is used to route the address. If there is no usable routing information in the Routing Tree node, the next longest is matched until the Routing Tree root is reached. Routing Trees are used by ds policies.

Note that the root of a Routing Tree can also hold Routing Information allowing a fallback (default) route to be configured.

#### Routing Tables

Routing Information can be held in tables. The Internet MTA creation wizard sets up the `domain` and `channel` tables. DNS lookups are used prior to the table lookups so that simple DNS based routing works. In many cases this will be sufficient for an Internet MTA.

For X.400 or other more advanced features of M-Switch routing, the entries in these tables can be set up or modified. See M-Switch Advanced Administration Guide for details of how to do this. Tables are used by `table` policies.

#### White Page Directories

An LDAP/X.500 Directory can be used to hold routing information, address mapping information or delivery information. This is used by `LASER` policies.

#### DNS (Domain Name Service)

The Domain Name System (DNS) is a distributed directory that resolves human-readable hostnames, such as `www.dyn.com`, into machine-readable IP addresses such as `50.16.85.103`. DNS is also a directory of information about domain names, such as email servers (MX records) and sending verification (DKIM, SPF, DMARC), TXT record verification of domain ownership.

The (official) root DNS servers are maintained by ICANNs, but are generally delegated to subsidiaries in a heirarchical manner. The DNS is used by `dns` policies.

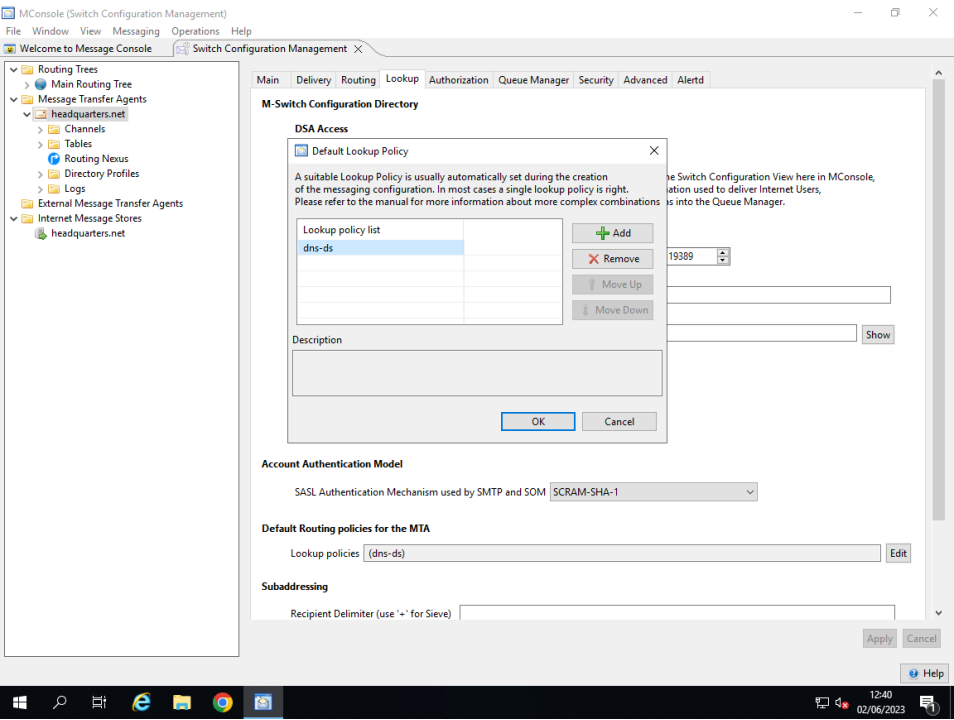
---

## 15.3 Lookup Policies

The lookup policy ([Section 5.3, “External MTAs”](#)) determines where channels and other M-Switch programs are to look for information on how to route messages. A default policy for all channels is set in the configuration for the MTA but you can override this for individual channels.

[Figure 15.1, “Default lookup policy editor”](#) shows how to edit and select the MTA Lookup Policy.

Figure 15.1. Default lookup policy editor



Channel properties can be displayed and modified from the **Switch Configuration Management** view in the same way as MTA properties and the lookup policy fields for both work in the same way.

The effect of the policies is described below in [Table 15.1, “LASER lookup policies for message routing”](#). The prefix window is for use with tables, see the *M-Switch Advanced Administration Guide* for details.

When routing a message, the policies are tried in the order given until routing information is found. If none of the policies can provide the necessary routing information, the message will not be delivered to this recipient and a non-delivery notification is generated, specifying that the recipient cannot be routed.

Note that the use of a policy can result in an 'authoritative failure', which terminates the process, without considering policies which follow in the sequence.

---

**Note:** If `queue` is included anywhere in the list, the message will be queued and processed later if temporary name-server errors occur. If `queue` is the first lookup policy, all messages are queued on the housekeeper channel to be processed subsequently.

---

Table 15.1. LASER lookup policies for message routing

Value	Effect
<code>dns-laser</code>	Verifies the domain using the DNS and if the delivery would be to the local system, the address is deemed to be local.
<code>table-laser</code>	Uses the domain table as in standard table-based routing
<code>dns-table-laser</code>	Uses the DNS to verify the domain, but then proceeds to use the domain table, followed by LASER Lookup

Value	Effect
<code>nssoft</code>	If the previous Domain Name Service or Directory lookup failed because the information could not be read (for example, the DSA or DNS server was down), continue with the lookup policies as if the attempt to read the information was successful but no routing information was found. This policy must be preceded by a value which includes <code>dns</code> or <code>ds</code> .
<code>queue</code>	Queue the message. This policy can be used to speed up message submission, as described in <a href="#">Section 15.3.1, “Queue Lookup Policy”</a> .

The `nssoft` policy enables processing to continue in the event of a DNS or DS read failure. Normally processing for the current message would stop in such circumstances.

**Table 15.2. Other lookup policies**

Value	Effect
<code>dns-ds</code>	The Domain Name Service ( <code>dns</code> ) is looked up initially to attempt to normalize an RFC 822 domain address. Then the Directory ( <code>ds</code> ) is accessed for information on how to route the address. If the Domain Name Service lookup was successful but the Directory lookup failed to produce the necessary routing information, and the address is not local, the Domain Name Service is checked again to find out where to relay this message. If the address is an Internet address, and deemed to be local, a LASER lookup is performed in order to deliver the message.
<code>dns</code>	Uses the domain table as in standard table-based routing
<code>ds</code>	Look up routing information in the Directory. If the address is an Internet address, and deemed to be local, a LASER lookup is performed in order to deliver the message.
<code>dns-tbl</code>	Normalize the address using the Domain Name Service, then look up routing information in tables.
<code>table</code>	Look up routing information in tables.

LASER routing is recommended for Internet messaging, but the above values can be used. See the *M-Switch Advanced Administration Guide* for a description of how to use these values.

## 15.3.1 Queue Lookup Policy

The `queue` lookup policy works differently in that it defers routing, submitting the message instead onto the `housekeeper` channel, which performs routing after the Switch has accepted the message. This channel is set up automatically and does not need to be configured.

There are two reasons `queue` may be chosen:

- the first reason is to prevent the rejection in protocol (e.g. for SMTP), so that unrouteable recipients result in a non-delivery.
- the second reason is if lookup of routing information is slowing down submission (this applies to all protocols)

To configure this action, make `queue` the first default lookup policy.

---

**Note:** Speeding up submission in this way may increase the load on the system.

---

### 15.3.2 Default Lookup policies

When MConsole creates the initial configuration for an MTA, it will set up the Lookup Policies as follows:

**ds**

X.400 MTA

**ds**

Internet MTA not using DNS

MIXER MTA not using DNS

**dns-ds**

Internet MTA using DNS

MIXER MTA using DNS

The following lookup policies are not configured by default and are deprecated. However they can be used for the following ways:

**dns-table-laser**

Internet MTA using DNS

**table-laser**

Internet MTA not using DNS

The Directory lookups which take place when **ds** is part of the Lookup Policy includes both Routing Tree lookups as well as LASER lookups for Internet users.

---

## 15.4 Routing Trees

This section describes how Routing Tree are used to perform routing. This occurs when Lookup Policies including **ds** are configured.

### 15.4.1 Routing Tree Overview

This section provides an overview of routing using Routing Trees in a Directory-based X.400 or Internet configuration.

The procedure is:

- use Routing Tree(s) to find potential MTAs (as described in this section)

- look at the channels of the remote MTA and find matches among the inbound protocols on those channels to the outbound protocols of the channels belonging to the local MTA. This is described in [Section 15.6, “Connectivity”](#)

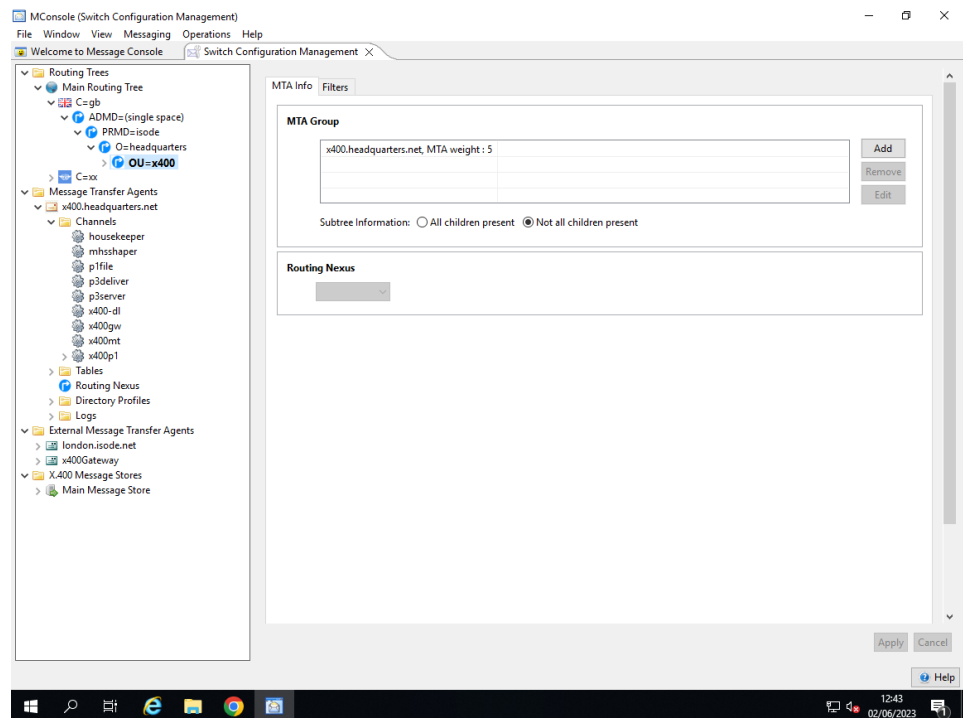
---

**Note:** The remote MTA can be a full tailoring MTA, or an External MTA.

---

The Routing Tree below shows a Routing Tree node selected which contains no routing information, i.e. the MTA Information box is empty. The next section shows the information which may be configured in a Routing Tree node.

**Figure 15.2. A Routing Tree with Routing Tree node selected**



## 15.4.2 Routing Information

A Routing Tree Node can contain:

- **Delivery Information:** which provides information about delivery into a P7 Message Store, or a P3 client.
- **MTA Information:** which provides information about possible "next hop" MTAs to which the recipient could be routed.
- **Routing Nexus:** which provides an indirect link to MTA information in a similar way as direct MTA Information described above.
- **Routing Filter:** allows a set of addresses to be relayed to another MTA using the information in an arbitrary routing node. See [M-Switch Advanced Administration Guide](#)
- **Redirect Filter:** These are very similar to Routing Filters, except that instead of using the Routing Information in an arbitrary Routing Tree node, the address is redirected to a new ORName. In all other respects Redirect Filters and Routing Filters work in the same way. See [M-Switch Advanced Administration Guide](#)
- **No Information:** the routing lookup will backtrack up to the next Routing Tree Node in the Tree.

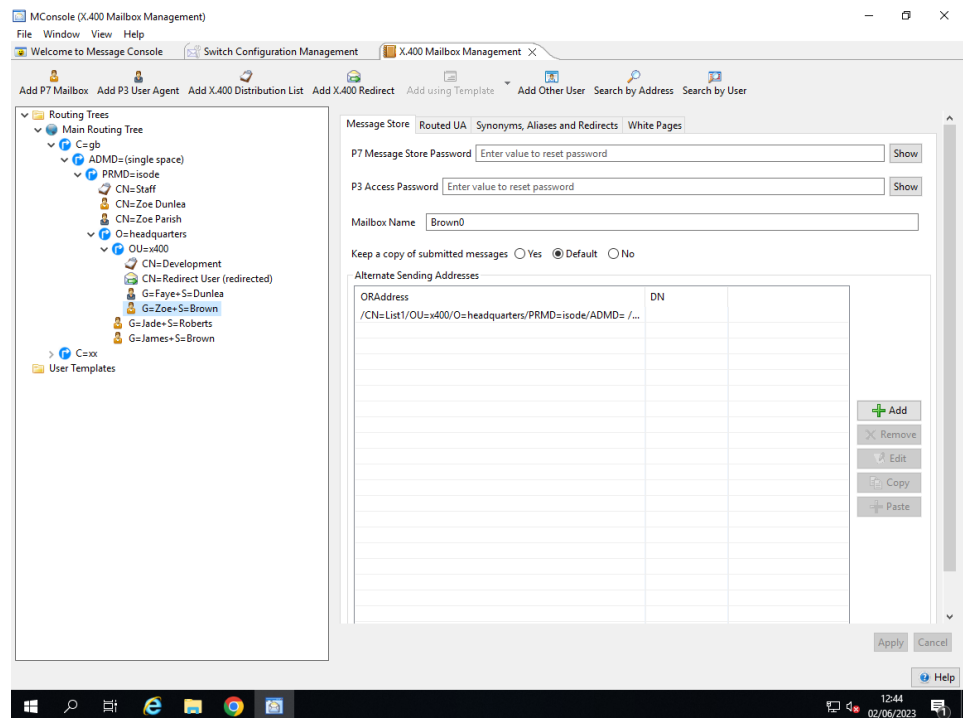
### 15.4.2.1 Delivery Information

Routing Trees when displayed as part of the **Switch Configuration Management** view do not show mailboxes. To see the X.400 you need to open the **X.400 Mailbox Management** view.

Internet mailboxes are managed by **Cobalt**, consult the Cobalt Administration guide for more information.

The figure below shows an X.400 User in the Routing Tree under the node selected in the previous figure.

**Figure 15.3. An X.400 User in the X.400 Mailbox Management view**



For example, the X.400 address of the local X.400 User

```
/G=Zoe/S=Brown/OU=x400/O=headquarters/PRMD=isode/ADMD= /C=gb/
```

is delivered to the local Message Store by the p3delivery channel using the X.400 Mailbox Entry in the Routing Tree.

```
C:\Program Files\Isode\bin>ckadr -v -x
"/G=Zoe/S=Brown/OU=x400/O=headquarters/PRMD=isode/ADMD= /C=gb/"
/G=Zoe/S=Brown/OU=x400/O=headquarters/PRMD=isode/ADMD= /C=gb/ ->
(x400) /G=Zoe/S=Brown/OU=x400/O=headquarters/PRMD=isode
/ADMD= /C=gb/
/G=Zoe/S=Brown/OU=x400/O=headquarters/PRMD=isode/ADMD= /C=gb/ ->
(rfc822)
"/G=Zoe/S=Brown/OU=x400/O=headquarters/PRMD=isode/ADMD= /C=gb/
"@x400.headquarters.net

Delivered to x400.headquarters.net by p3deliver (weight: 0)
```



### 15.4.2.2 MTA Information

When the Routing Tree node holds MTA Information, a recipient is routed to the (Internal or External) MTA in the MTA Information box.

In this example, the X.400 address

```
/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/
```

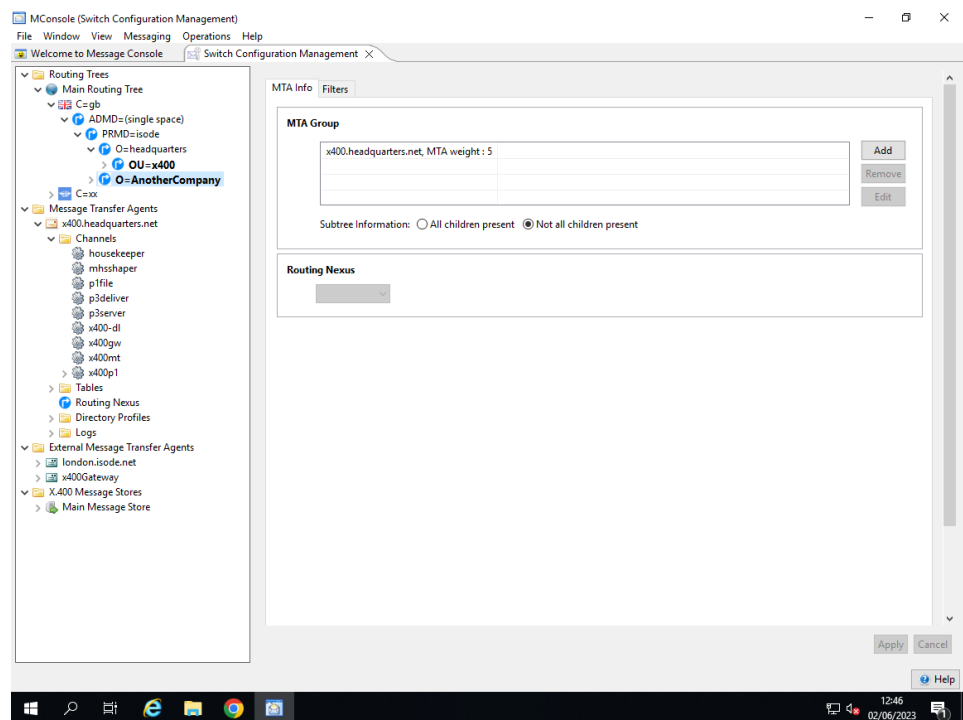
would be routed using the **Routing Tree** node selected in Figure 15.4, “Selecting a routing tree node”, that is, the entry for the O/R Address prefix:

```
/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/
```

is used (because there is no entry for /CN=John Smith/) beneath  
/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/.

The **Routing Tree** node contains the MTA x400.headquarters.net, so the address would be routed to this MTA (subject to other checks such as network connectivity).

**Figure 15.4. Selecting a routing tree node**



```
C:\Program Files\Isode\bin>ckadr -x
"/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/ "
/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/ -> (x400)
/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/

Delivered to cn=x400p1,cn=x400.headquarters.net,
cn=Messaging Configuration,
o=Isode,o=messaging by x400p1 (weight: 5)
```

### 15.4.2.3 Nexus Information

As described in [Section 15.4.2, “Routing Information”](#), a Routing Tree Node can contain different types of Routing Information (or none). MTA Information (which refers to an Internal or External MTA) can be held in the Routing Tree node itself. This is described in [Section 15.4.2.2, “MTA Information”](#).

Instead of MTA Information, a Routing Tree Node may contain a Routing Nexus. This allows an extra level of indirection to be inserted between the Routing Tree Node and MTA Information.

A Routing Nexus specifies one or more MTAs which are to be the target for Routing. Each MTA can be labelled **Enabled** or **Disabled**. It is expected that all but one of the MTAs will be labelled as **Disabled**.

A nexus comprises:

- The Nexus itself
- A description for the Nexus
- Zero or more Nexus Data each of which is a child of the Nexus, and contains:
  - MTA Information (one or more) OR
  - Indirection OR
  - ACP 127 channel
  - Enabled/Disabled indication. If disabled, the MTA Information is ignored for Routing

The Nexus Data items hold different possibilities for routing. These possibilities can be active or inactive. This enables these to be set up once, and alternate routes chosen easily as circumstances necessitates.

Typically, the routing data is just an MTA Information as for a normal routing tree node. By changing which Nexus Data item is enabled, the whole Nexus can change the route from one MTA to another.

For more complex setups involving the use of Icon-Topo, an Indirection can be used instead. An Indirection is a way to point to another Nexus, that is the 'default' Nexus. For example, if ten Nexus are set up to have an Indirection to Nexus "ABC", by changing the routing in the "ABC" Nexus we can indirectly change the routing of those ten Nexus.

A Nexus Data item can contain an ACP 127 channel. In this case, it works like an MTA, but limited to the ACP 127 protocol.

Which ACP127 circuit is used for the message will be determined using the ACP127 routing indicators for the message.

In this example, the X.400 address

```
/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/
```

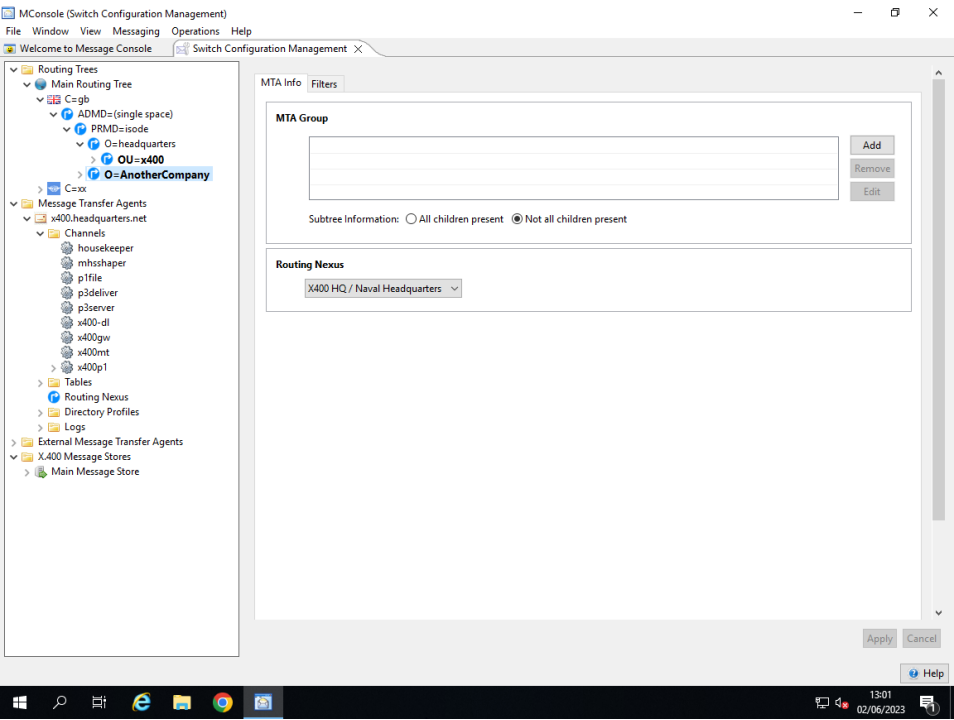
would be routed using the **Routing Tree** node selected in [Figure 15.5, “Selecting a Routing Tree node with Nexus”](#), that is, the entry for the O/R Address prefix:

```
/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/
```

is used (because there is no entry for /CN=John Smith/) beneath  
/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/.

The **Routing Tree** node contains the Headquarters Nexus.

Figure 15.5. Selecting a Routing Tree node with Nexus

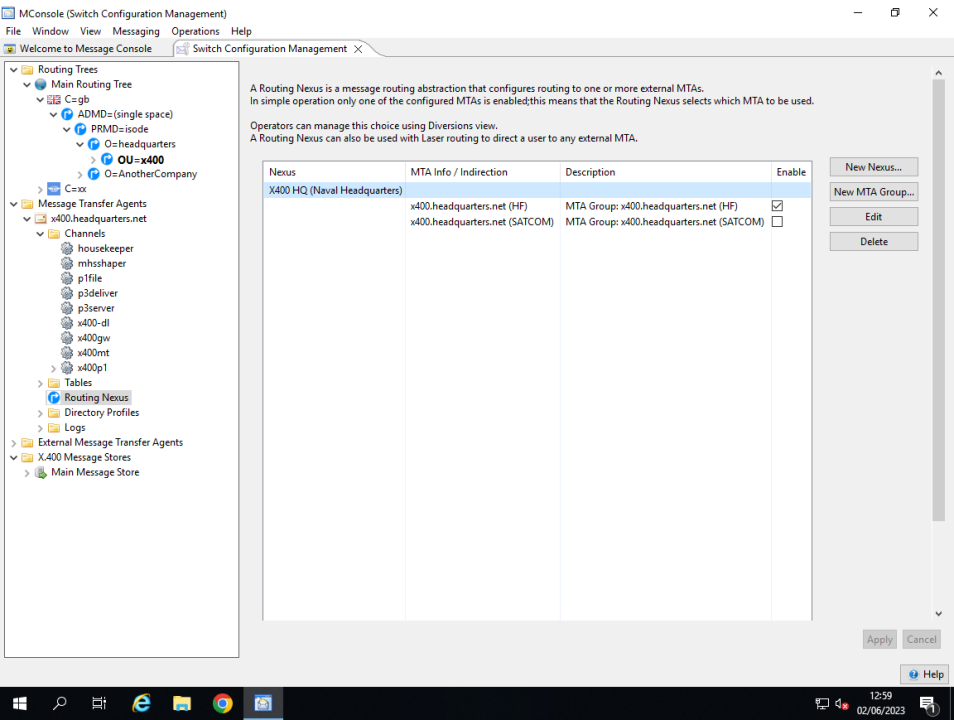


This Nexus contains two Nexus Data (MTA Group) Entries, referring to two external MTAs. One is reached via HF, and one via SATCOM.

The two Nexus Data Entries each contain MTA Information for `x400.headquarters.net`, so the address would be routed to this MTA (subject to other checks such as Authorisation).

When the Nexus Headquarters has HF enabled and SATCOM disabled (as in [Figure 15.6, “Configuring a Routing Nexus \(HF\)”](#) below, the routing command returns information as follows:

Figure 15.6. Configuring a Routing Nexus (HF)

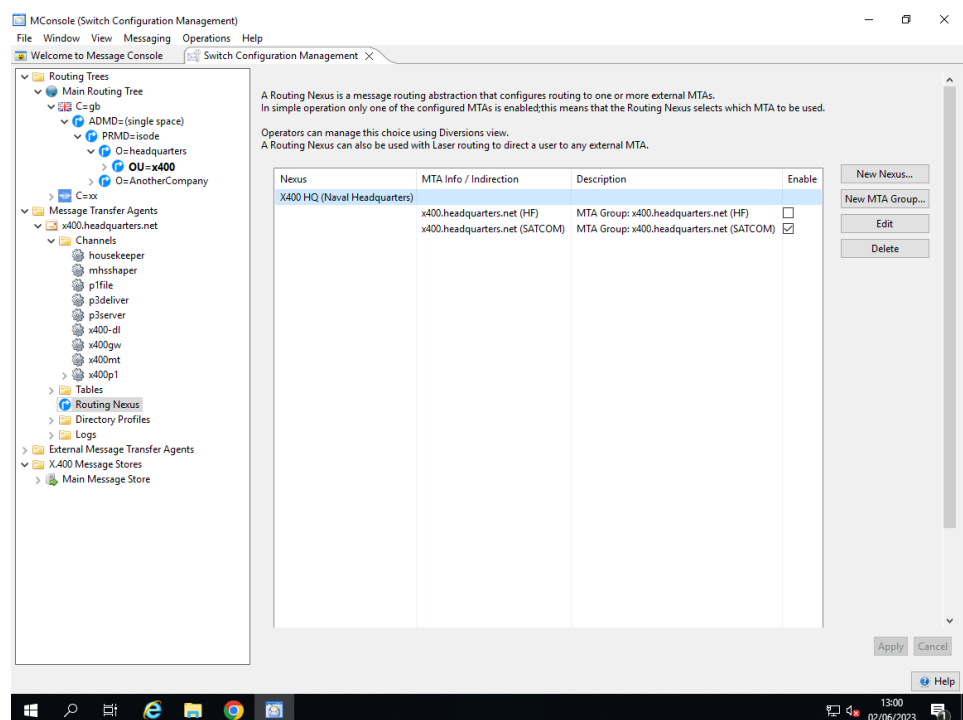


```
C:\Program Files\Isode\bin>ckadr -x
    "/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/ "
/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/ -> (x400)
    /G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/

Delivered to cn=x400p1,cn=X400 Headquarters (HF),
cn=Messaging Configuration,
o=Isode,o=messaging by x400p1 (weight: 5)
```

When the Nexus Headquarters has HF disabled and SATCOM enabled (as in [Figure 15.7](#), “Configuring a Routing Nexus (SATCOM)” below, the routing command returns information as follows:

**Figure 15.7. Configuring a Routing Nexus (SATCOM)**



```
C:\Program Files\Isode\bin>ckadr -x
    "/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/ "
/G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/ -> (x400)
    /G=John/S=Smith/O=AnotherCompany/PRMD=Isode/ADMD= /C=GB/

Delivered to cn=x400p1,cn=X400 Headquarters (SATCOM),
cn=Messaging Configuration,
o=Isode,o=messaging by x400p1 (weight: 5)
```

---

## 15.5 Fallback Routes

A Routing Tree node can hold MTA Information for multiple MTAs. There are two reasons for using this:

- Load Balancing: using weights that are equal
- Fallback routes: so a secondary route (with lower weight) can be used if the preferred route fails

Like MX records, the smaller number is the more preferred.

When routing, the different routes are assessed. If a route is not valid, e.g. because of authorisation, it is removed from the list. Then the lowest weight route is selected.

If there are other routes using the same outbound channel, then these are included with this other route. The message is queued to the group of MTAs on the single channel. Then the system can set up a connection to an alternate MTA if the preferred MTA is not available through not responding. (Or you can have load balancing by use of equal weights).

### 15.5.1 ACP142 and SATCOM/HF Considerations

In this scenario, the SATCOM and HF routes are via different local channels. Thus, nothing automatic would happen. What would be needed is for the routing to be changed to remove the SATCOM route. Then messages on the SATCOM channel need to be aborted - if being transferred, and reprocessed to update the route. (You need to do this in the other order).

In addition, ACP142 is a connectionless protocol which sends out PDUs. It waits for NACKs and ACKs. However, the message would continue to be transmitted until its expiry time is reached, even if no response is received. After all, it works when the receiver is not transmitting at all. So, manual action is required if SATCOM is not working.

---

## 15.6 Connectivity

Once routing information has been obtained by looking up an O/R Address or Internet Address in a **Routing Tree**, the MTA performing the routing needs to determine whether the potential MTAs returned from the **Routing Tree** lookups can be connected to by the local MTA.

This is calculated by considering all the protocol channels of the potential remote MTA and all the protocol channels of the local MTA. Those remote MTAs which share one or more Application Contexts with the local MTA are added to the sequence of MTA/Channel pairs being generated by the routing procedure.

### 15.6.1 Peer Connections

The routing procedure looks up information in the local and remote MTA entries. The local MTA's values may be overridden by a Peer Connection in the local MTA. Peer Connections are configured beneath the protocol channels, e.g. x400p1, SMTP or ACP 142 of non-External MTAs.

See [Section 15.9, “Peer Connections”](#) for a description of Peer Connections.

---

## 15.7 Adding an external MTA

An external MTA is one which is not managed within the current MConsole configuration (or MHS Messaging Configuration). The details for a non Isode MTA will always be an External MTA. Connecting to an Isode MTA may use an External MTA or a full Internal (tailoring) MTA.

There are several different types of External MTA you can create reflecting the different transfer protocols supported by M-Switch. You can also create Gateways which provide ways to transfer messages into different forms of Message Transfer Systems using the Isode Gateways APIs, i.e. The Open Group Gateway API, or the Isode Gateway API. In these cases the Gateway is an Application built using the Isode APIs acting as a co-located MTA.

The following options appear when creating an External MTA:

### **SMTP**

A simple Internet MTA with a single SMTP channel

### **X.400 MTA**

A simple X.400 MTA with a single x400p1 channel

### **SMTP and X.400**

An MTA with both Internet and X.400 channels

### **ACP 127 Station**

An MTA with an ACP 127 channel

### **ACP 142 (STANAG 4406 Annex E or MULE)**

An MTA with an ACP 142 channel

### **CFTP**

An MTA with a CFTP (Compressed File Transfer Protocol) channel

### **SLEP**

An MTA with a SLEP (SIS Layer Extension Protocol) channel

### **X.400 Gateway (using the Isode or Open Group Gateway API)**

This allows a client written using the Isode Gateway APIs to transfer messages in and out of the MTA.

### **X.400 P1 File Gateway**

This allows a client to transfer messages in and out of the MTA to an application which reads/writes P1 file APDUs.

### **X.400 P1 Over HTTP Gateway**

This allows a client to transfer messages in and out of the MTA to an application using P1 APDUs over HTTP.

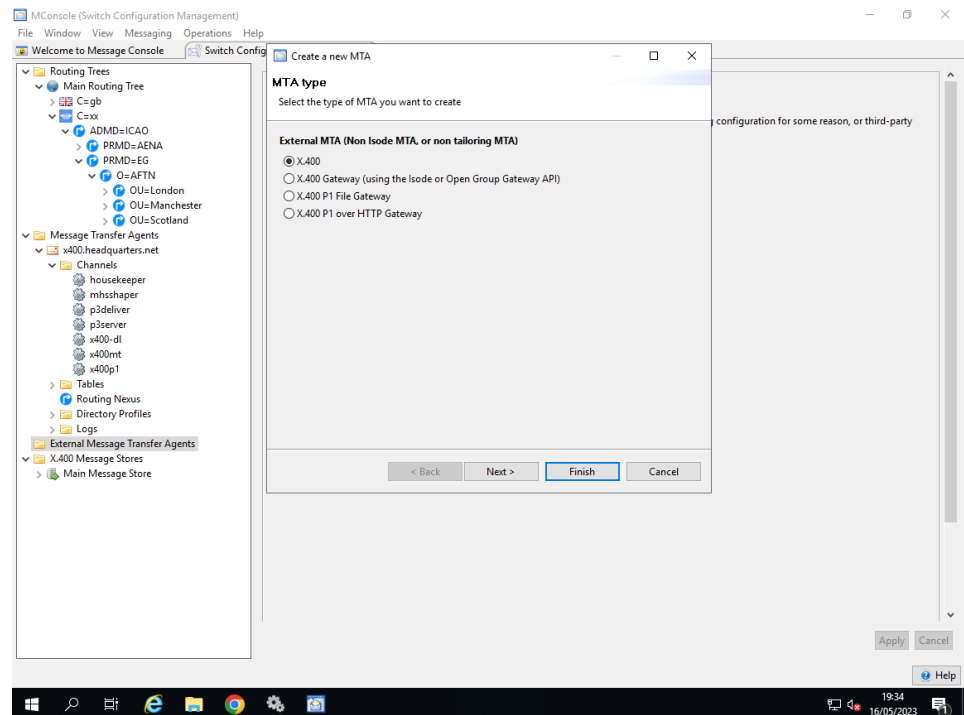
### 15.7.1 Creating an external MTA

Before starting to create an external MTA, you should create a routing tree node for the area of O/R Address or Internet domain space (or both if MIXER) which the external MTA will manage. Refer to [Section 15.8, “Adding a shared configuration MTA”](#) for details of how to do this.

Next, right click on the **External Message Transfer Agents** folder and select the **New External MTA** option. You will be presented with a wizard which, in its first page, lets you choose which type of external MTA should be created. The choices which are available

depend on the type of messaging configuration you are working in. A pure X.400 or pure Internet configuration will not show the Internet and X.400 options (respectively). This dialogue is shown below.

**Figure 15.8. New External MTA Type**



The next screen displayed allows you to specify the MTA name. The contents of this screen will vary, depending on the type of external MTA you are creating.

---

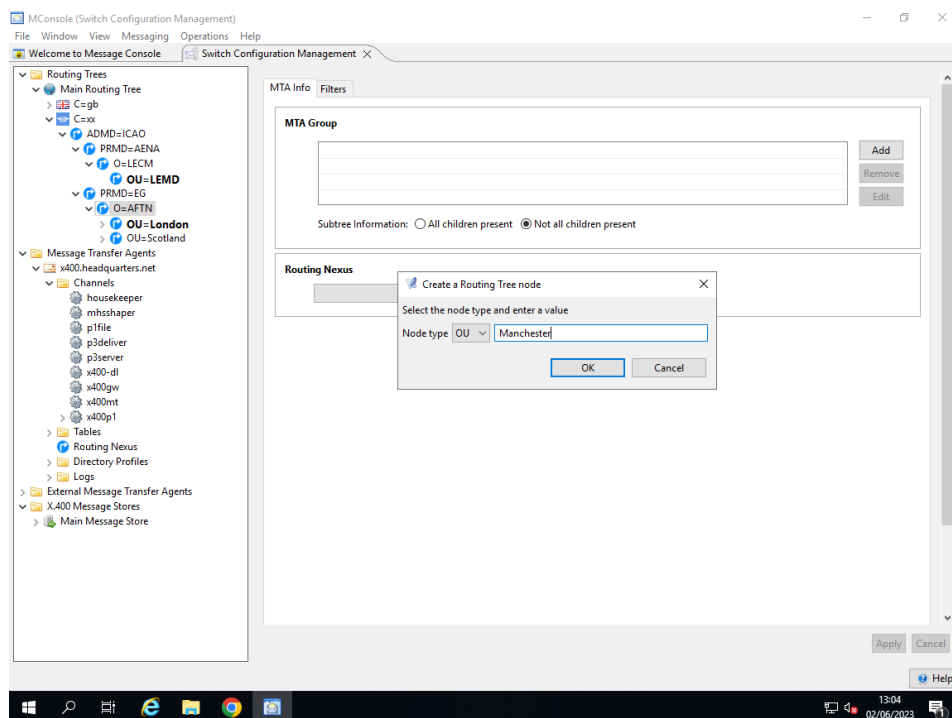
**Note:** For X.400 External MTAs, the name which you specify for the new external MTA must match the `MTAName` which the external MTA sends when binding to your internal MTA. This is necessary so that the internal MTA can locate the correct external MTA when it receives a Bind Request.

---

## 15.8 Adding a shared configuration MTA

This section of the manual describes how to extend an existing messaging configuration to allow a second Isode MTA to use it.

You should start by creating a routing tree node for the area of O/R Address or Internet domain space which the new MTA will manage. This will enable you to select the correct node to associate with the new MTA when you create the new MTA configuration.

**Figure 15.9. Adding a routing tree node**

Right click on an existing node in the routing tree and select **Add Node**. You will be presented with a window asking for a name, and a node type if you are adding a node to an O/R Address. In an Internet domain hierarchy, all nodes are of type Domain Component.

Select a node type and enter the node value. Repeat this until you have created an O/R Address hierarchy appropriate to your new MTA. You are now ready to add the new MTA.

---

**Note:** Internet Domains can be routed in Routing Trees, but are normally routed using DNS.

---



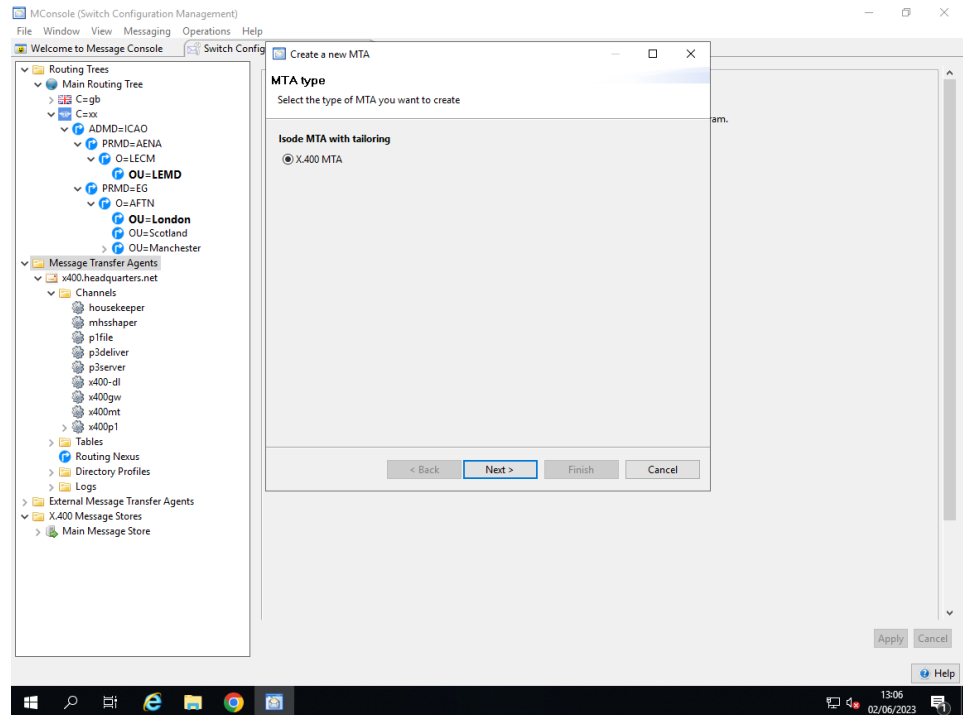
---

**Note:** MConsole can be run on any machine, as long as it connects to the Directory Server which holds the existing configuration.

---

In MConsole, right click the **Message Transfer Agents** folder, and press the **New Tailoring MTA** button. If you are working in a MIXER configuration, you will also be able to choose the type of MTA to be created.



**Figure 15.10. Specifying the type of the new MTA.**

If you choose to create an X.400 MTA, you will go through the same wizard as in [Chapter 14, Configuring MTAs](#).

## 15.9 Peer Connections

**Note:** Peer Connections were formerly known as Bilateral Agreements. This change in terminology is for consistency with other channels such as ACP127 channels which also have Peer Connections configured.

Peer Connections can be applied to pairs of MTAs which have x400p1, SMTP, ACP 127, ACP 142 or CFTP channels configured.

### 15.9.1 When to use peer connections

Peer Connections describe a set of overrides for a specific pair of MTAs which override the default values held in each of the MTA's individual entries.

Additionally, there are some features which are only configurable using a peer connection. Permanent MTAs are the set of MTAs of which the qmgr is always aware. An MTA which has a peer connection configured with another MTA regards the other as a Permanent MTA. Permanent here means that it is always visible in the **Switch Operations** view of MConsole.

**Note:** Permanent MTAs as described above are different from Permanent Associations. The latter are MTAs to which M-Switch creates and maintains an association (connection) on M-Switch startup. You must use a Peer Connection to configure Permanent Associations.

## 15.9.2 What is a peer connection?

By default, a pair of MTAs wishing to interwork must be able to read each other's entry in the Directory either as an internal or an external MTA. In the latter case the lookup is carried out by using the `MTAName` supplied in the bind. They need to do this to obtain information, such as the Presentation Address, which is required for interworking. These lookups are of the x400p1 channel of the MTA. As this entry is used by this MTA for connections to all other MTAs, it follows that the values used for each remote MTA must be the same.

This is obviously undesirable when considering such values as credentials as you may well want to be able to configure a different local MTA password for each remote MTA.

To allow a pair of MTAs to interwork using different information, for example credentials, you need to set up a Peer Connection between them. The information in the peer connection overrides that in each of the MTA's entries in the DIT.

Peer Connections of an MTA are associated with the MTAs outbound channel to be used

Each MTA can have several peer connections configured, each of which contains details of the MTA's own half of an agreement. The other MTA's half of the agreement may be similarly configured in its own peer connections, or may be configured by a different mechanism entirely, such as in tables or in the local configuration of the remote MTA (which may not be an Isode MTA).

For an MTA to determine the configuration for interworking with another MTA, three lookups are involved:

- it reads its own entry.
- it reads the remote MTA's entry (as a shared configuration MTA or external MTA).
- it reads the remote MTA's entry in its own set of peer connections. .

---

**Note:** When using a peer connection, access to the other MTA's actual entry is still required, because only some of the attributes may be overridden.

---

Peer Connections may contain paired attributes, one of which overrides the local MTA's attribute, and the other overrides the attribute read from the remote MTA's entry. Where such attributes have Initiator and Responder equivalents, two pairs are required.

An example of such an attribute is `AuthenticationRequirements` (AR). These have both Initiator and Responder equivalents, as well as Our (local) and Their (remote) equivalents. Thus a peer connection can hold:

- `responderOurAuthenticationRequirements`: overrides the local MTA's Responder authentication requirements.
- `initiatorOurAuthenticationRequirements`: overrides the local MTA's Initiator authentication requirements.
- `responderAuthenticationRequirements`: overrides the remote MTA's Responder authentication requirements.
- `initiatorAuthenticationRequirements`: overrides the remote MTA's Initiator authentication requirements.

# Chapter 16 Connecting to other X.400 MTAs

This chapter shows you how to configure connections to other X.400/MIXER MTAs.

Two types of configuration are possible:

- Shared-configuration MTAs. These are additional Isode MTAs which share this messaging configuration. Creating a shared MTA takes you through the same wizard as when you created your original MTA in [Chapter 9, Configuring an X.400 Messaging System](#).
- External MTAs. These may be either Isode MTAs which you have chosen to configure in a different messaging configuration for some reason, or third-party MTAs which are configured completely separately.

The main difference in configuration terms is that shared-configuration MTAs contain tailoring information, and External MTAs do not. An external MTA is the only way that connections to non-Isode MTAs can be configured.

If you are going to connect to a Military X.400 MTA, using the ACP 142 protocol (P\_MUL), then also read [Chapter 17, Connecting to other Military MTAs](#).

---

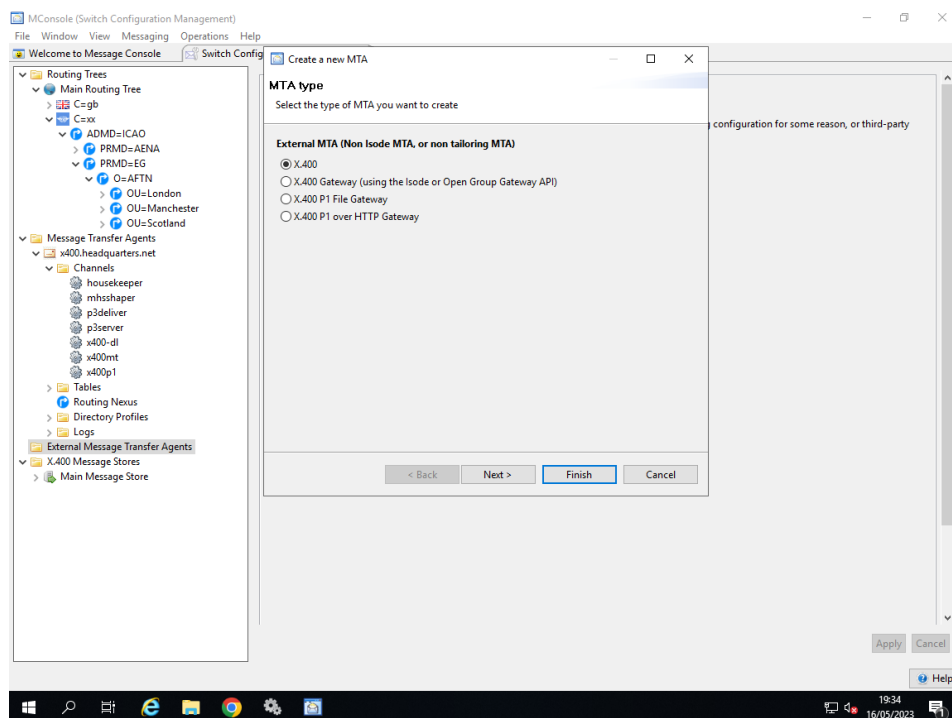
## 16.1 X.400 routing

### 16.1.1 Creating an external X.400 MTA

Before starting to create an external MTA, you should create a routing tree node for the area of O/R Address which the external MTA will manage. Right click on the Routing Tree Node under which you wish to add an entry. Select Add Node. You will be prompted for the Value of the Name of the Routing Tree node.

Next, right click on the **External Message Transfer Agents** folder and select the **New External MTA** option. You will be presented with a wizard which, in its first page, lets you choose which type of external MTA should be created. The choices which are available depend on the type of messaging configuration you are working in. A pure X.400 configuration will only show the X.400 options and a pure Internet configuration only the Internet options. This dialogue is shown below.

Figure 16.1. New External MTA Type



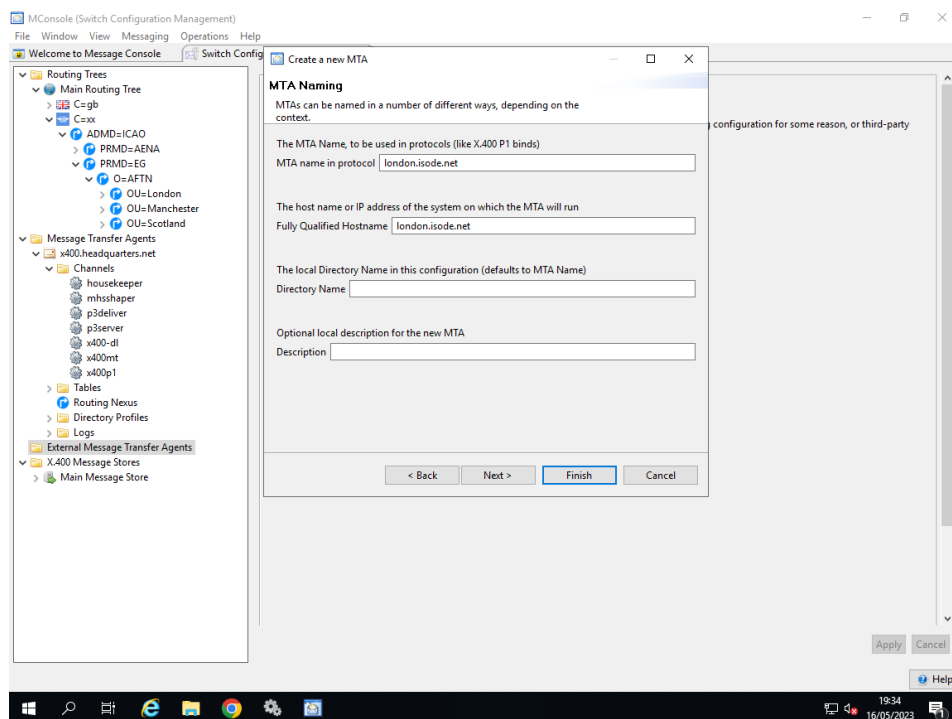
The next screen displayed allows you to specify the MTA name. The contents of this screen will vary, depending on the type of external MTA you are creating.

---

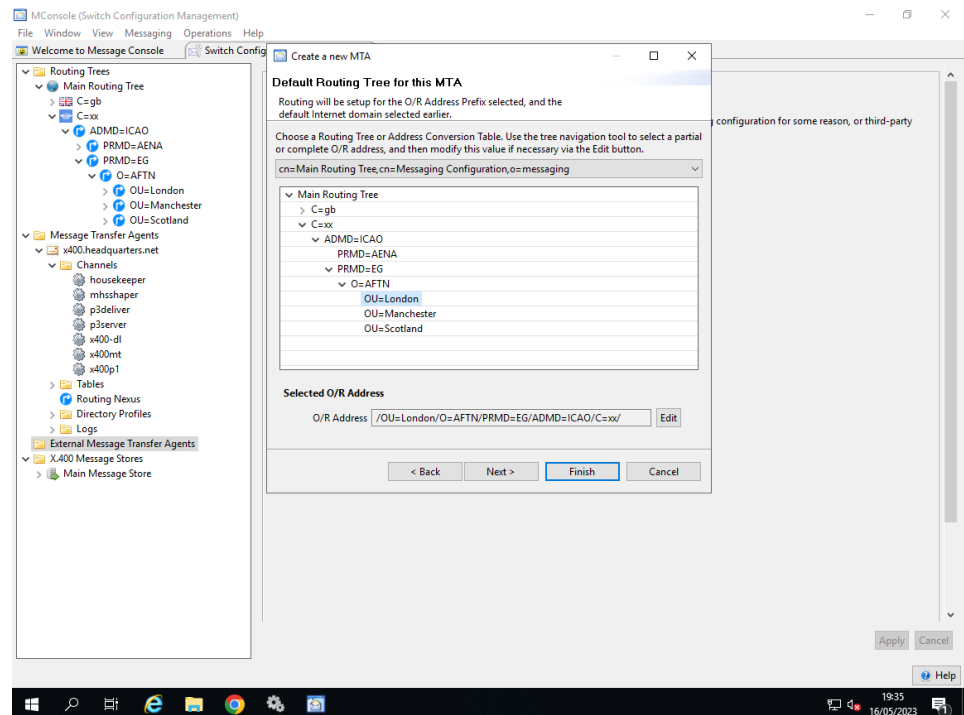
**Note:** The name which you specify for the new external MTA must match the MTAName which the external MTA sends when binding to your internal MTA. This is necessary so that the internal MTA can locate the correct external MTA when it receives a Bind Request.

---

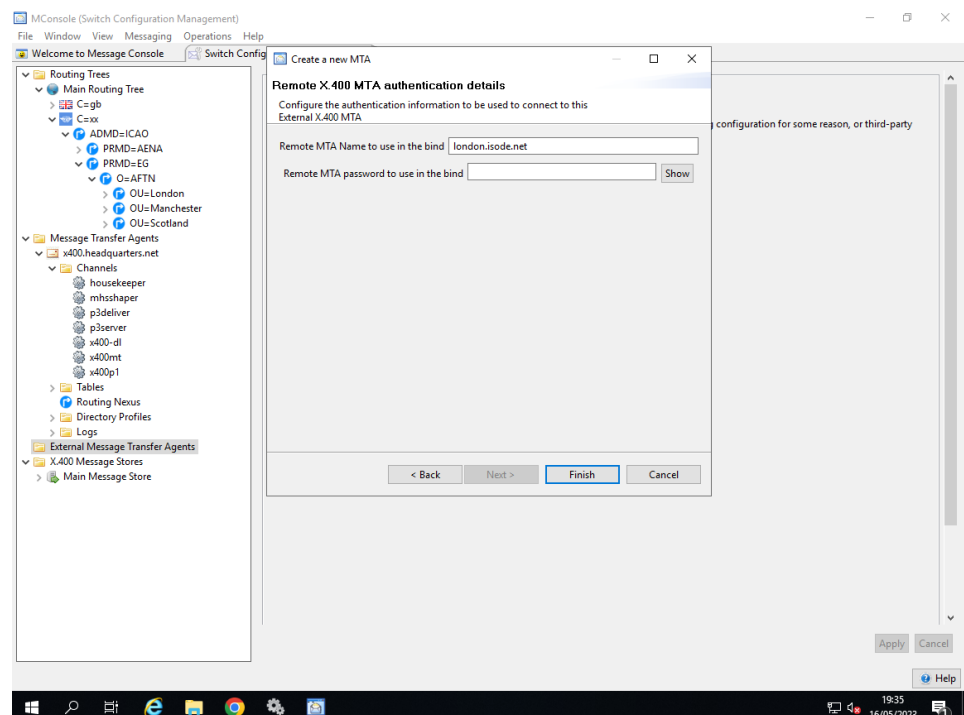
Figure 16.2. Entry of the Host Name for the External MTA (X.400)



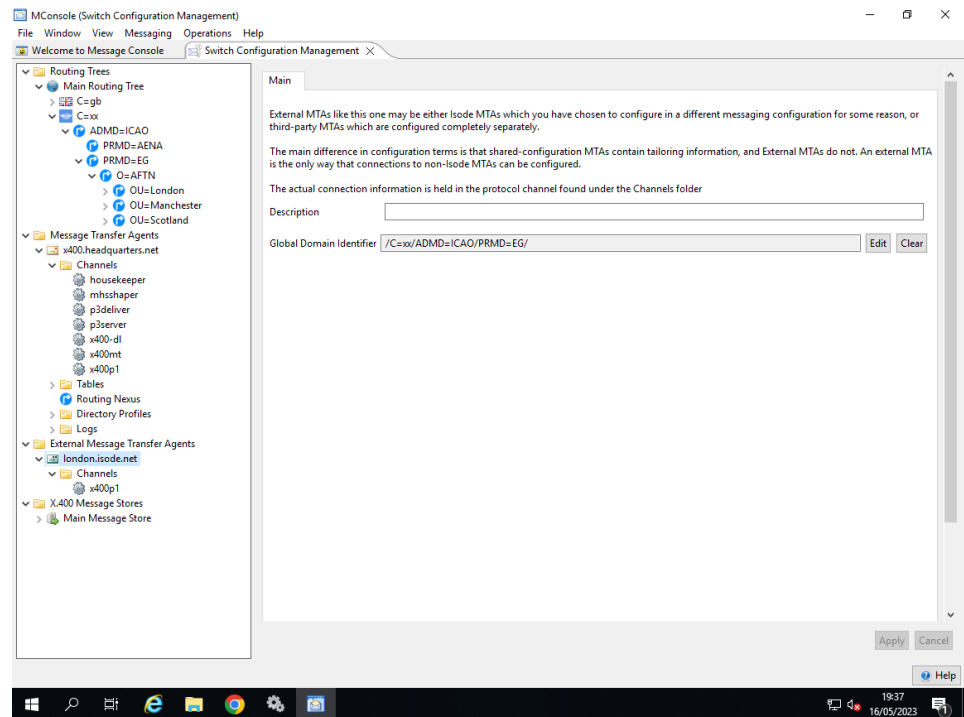
The next screen allows you to select the O/R Address(es) which are routed to this MTA.

**Figure 16.3. Default O/R Address prefix for the MTA**

If you are creating an X.400 or MIXER external MTA, the next page allows you to choose the MTA NAME and Password used in P1 Binds.

**Figure 16.4. Configuring P1 Binds.**

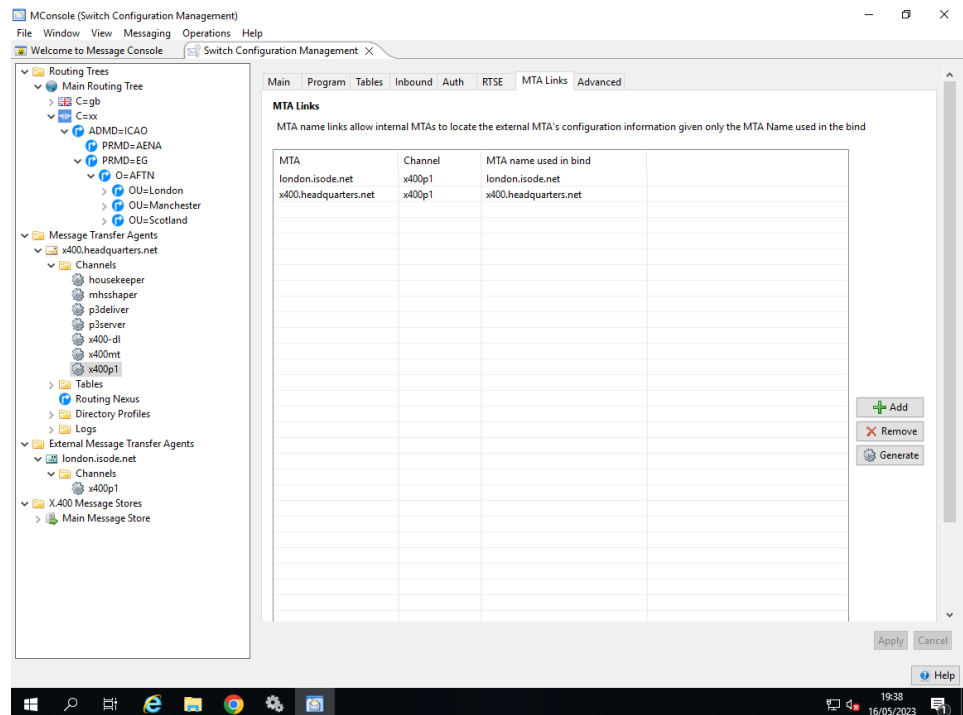
The new external MTA will now appear on MConsole's list of MTAs. It will have a small set of properties and either an X400p1 channel, an SMTP channel or both, depending on whether it is an X.400, MIXER or Internet MTA.

**Figure 16.5. Overview of New External MTA.**

The final task is to set up the information which allows internal MTAs to locate the external MTA's configuration information given only its MTA Name. To do this, either right click on **Messaging Transfer Agents** and select **Create MTA Links**, or select the menu option **Messaging → Create MTA Links**.

This configures all of the internal MTAs so that they are aware of the MTAName of all external MTAs and can therefore identify incoming connections based on the MTAName used in the incoming P1 Association request.

(This can also be done by selecting each internal MTA's X400p1 channel and clicking **Generate** on its **MTA Links** page).

**Figure 16.6. External MTA links.**

## 16.2 Peer Connections

**Note:** Peer Connections were formerly known as Bilateral Agreements. This change in terminology is for consistency with other channels such as ACP127 channels which also have Peer Connections configured.

Peer Connections can be applied to pairs of MTAs which have x400p1 channels configured.

### 16.2.1 When to use peer connections

Peer Connections describe a set of overrides for a specific pair of X.400 MTAs which override the default values held in each of their individual entries.

Additionally, there are some features which are only configurable using a peer connection. Permanent MTAs are the set of MTAs of which the qmgr is always aware. An MTA which has a peer connection configured with another MTA regards the other as a Permanent MTA. Permanent here means that it is always visible in the **Switch Operations** view of MConsole. See [Section 16.2.5, “Peer Connections and Permanent MTAs”](#).

### 16.2.2 What is a peer connection?

By default, a pair of MTAs wishing to interwork must be able to read each other's entry in the Directory either as an internal or an external MTA. In the latter case the lookup is carried out by using the MTAName supplied in the bind. They need to do this to obtain information, such as the Presentation Address, which is required for interworking. These lookups are of the x400p1 channel of the MTA. As this entry is used by this MTA for connections to all other MTAs, it follows that the values used for each remote MTA must be the same.

This is obviously undesirable when considering such values as credentials as you may well want to be able to configure a different local MTA password for each remote MTA.

To allow a pair of MTAs to interwork using different information, for example credentials, you need to set up a peer connection between them. The information in the peer connection overrides that in each of the MTA's entries in the DIT.

Each MTA can have several peer connections configured, each of which contains details of the MTA's own half of an agreement. The other MTA's half of the agreement may be similarly configured in its own peer connections, or may be configured by a different mechanism entirely, such as in tables or in the local configuration of the remote MTA (which may not be an Isode MTA).

For an MTA to determine the configuration for interworking with another MTA, three lookups are involved:

- it reads its own entry.
- it reads the remote MTA's entry (as a shared configuration MTA or external MTA).
- it reads the remote MTA's entry in its own set of peer connections. .

---

**Note:** When using a peer connection, access to the other MTA's actual entry is still required, because only some of the attributes may be overridden.

---

Peer Connections contain paired attributes, one of which overrides the local MTA's attribute, and the other overrides the attribute read from the remote MTA's entry. Where such attributes have Initiator and Responder equivalents, two pairs are required.

An example of such an attribute is `AuthenticationRequirements` (AR). These have both Initiator and Responder equivalents, as well as `Our` (local) and `Their` (remote) equivalents. Thus a peer connection can hold:

- `responderOurAuthenticationRequirements`: overrides the local MTA's Responder authentication requirements.
- `initiatorOurAuthenticationRequirements`: overrides the local MTA's Initiator authentication requirements.
- `responderAuthenticationRequirements`: overrides the remote MTA's Responder authentication requirements.
- `initiatorAuthenticationRequirements`: overrides the remote MTA's Initiator authentication requirements.

---

**Note:** You must use a peer connection to configure Permanent Associations. See [Section 16.2.5.1, "Scheduled associations"](#).

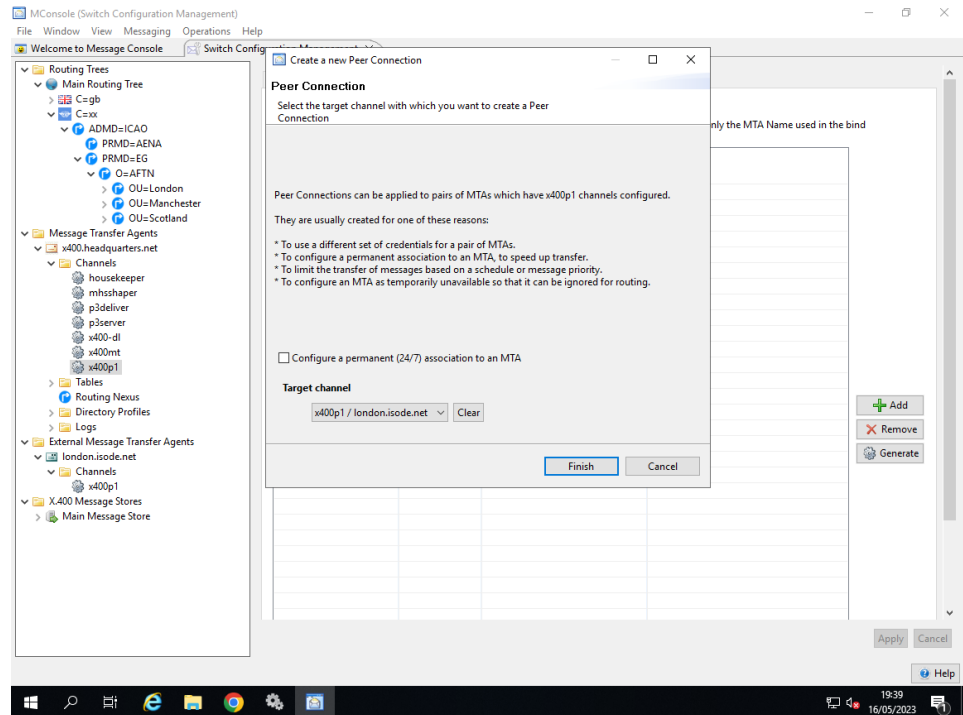
---

### 16.2.3 Creating a new peer connection

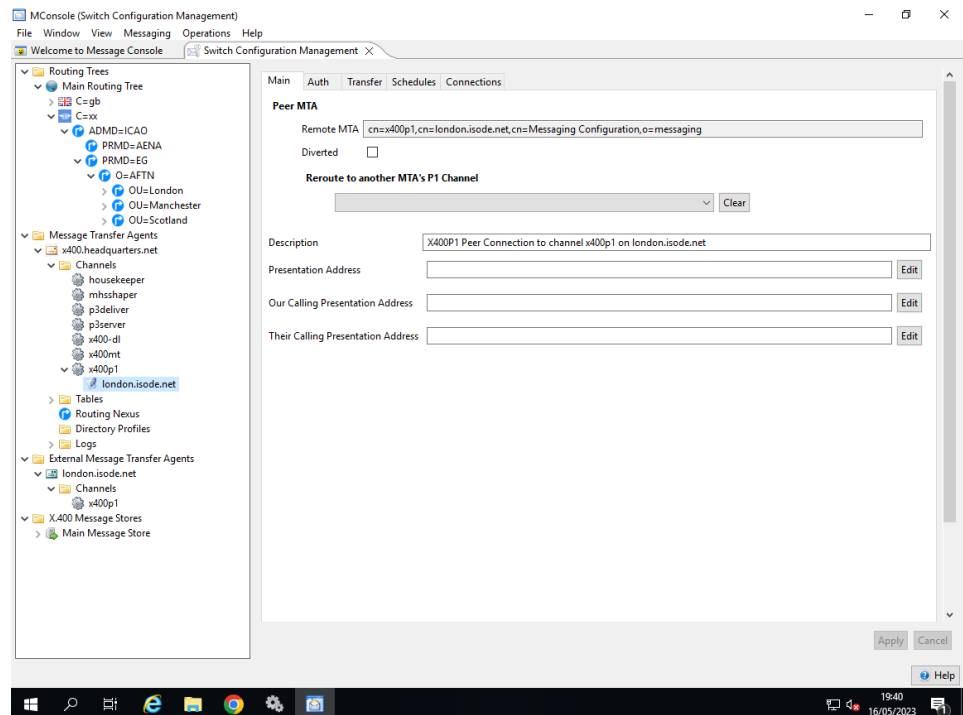
In this example there will be two MTAs: MTA A, a local MTA, and MTA B, an external MTA.

Right-click on MTA A's `x400p1channel` and select **Add Peer Connection** from the menu. From the pull down menu select MTA B and its corresponding X.400 channel. Click **Finish**.



**Figure 16.7. Creation of a new peer connection.**

The Peer Connection is now created and looks as in [Figure 16.8, “Peer Connection.”](#)

**Figure 16.8. Peer Connection.**

You may now edit the Peer Connection to alter the behaviour of this MTA when initiating connections, or responding to connection requests.

The **Main** tab shown in [Figure 16.8, “Peer Connection.”](#) allows the following to be configured:

#### Peer MTA

The remote MTA for which this is a Peer Connection

### Diversion

Set this option to Diverted if this MTA is unavailable and should be ignored when performing routing calculations

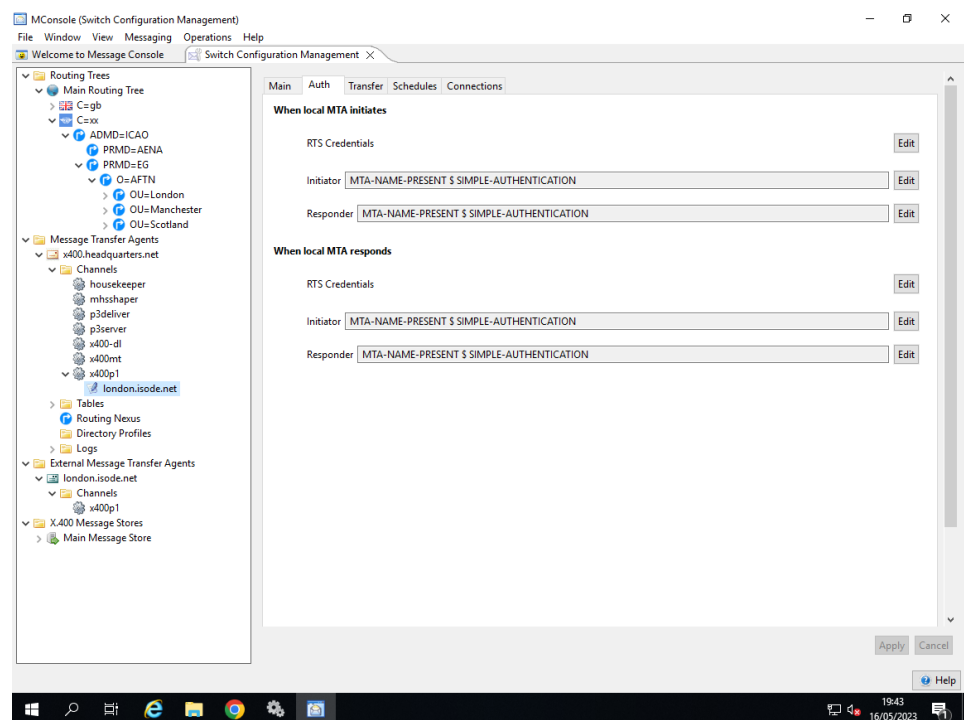
### Presentation Address

Address of the remote MTA

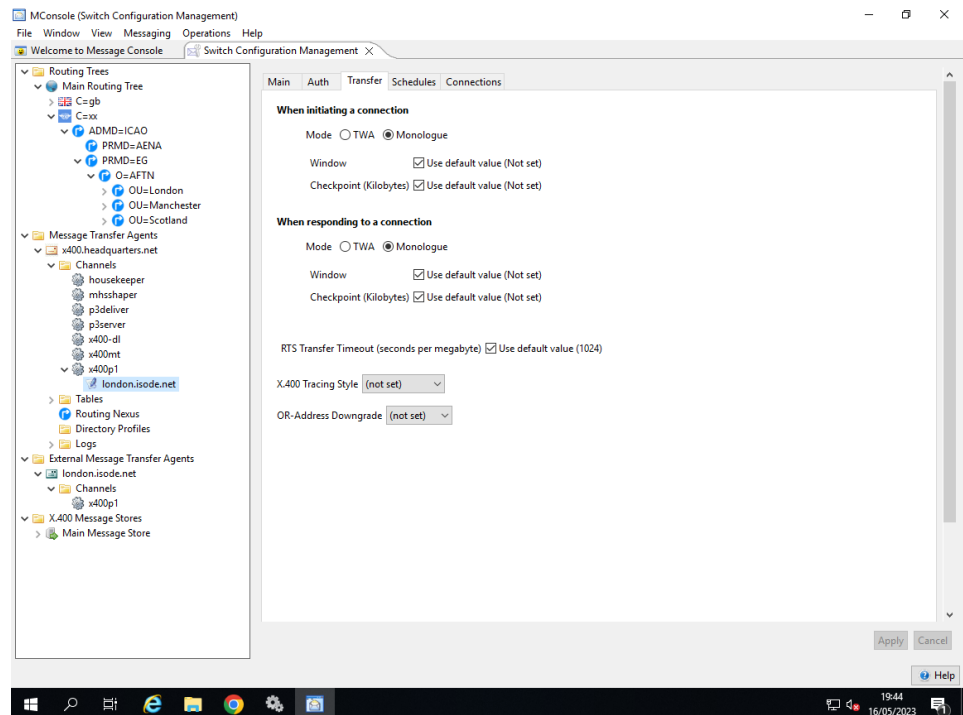
### Calling Presentation Address

Value to use as our calling address when connecting to the remote MTA. In a situation where MTA A is using the peer connection table to contact MTA B, MTA B can use the Our Calling Address value to authenticate the connection (i.e. to check that the entity calling it is calling from the expected address). This is optional and you may leave this blank. Similarly, if MTA B is contacting MTA A, MTA A can use the Their Calling Address value to perform authentication.

**Figure 16.9. Peer Connection Auth tab.**



The various possible credentials used by the P1 Initiator and Responder are configured as in [Figure 16.9, “Peer Connection Auth tab.”](#)

**Figure 16.10. Peer Connection Transfer tab.**

For both the Initiator and Responder modes, the following settings can be configured:

### Mode

this specifies the way in which the connection handles traffic; the choice is between **Monologue** mode, where traffic will only go one way through the connection (this is the default), and **TWA** (Two Way Alternate), where traffic can go in both directions through the connection.

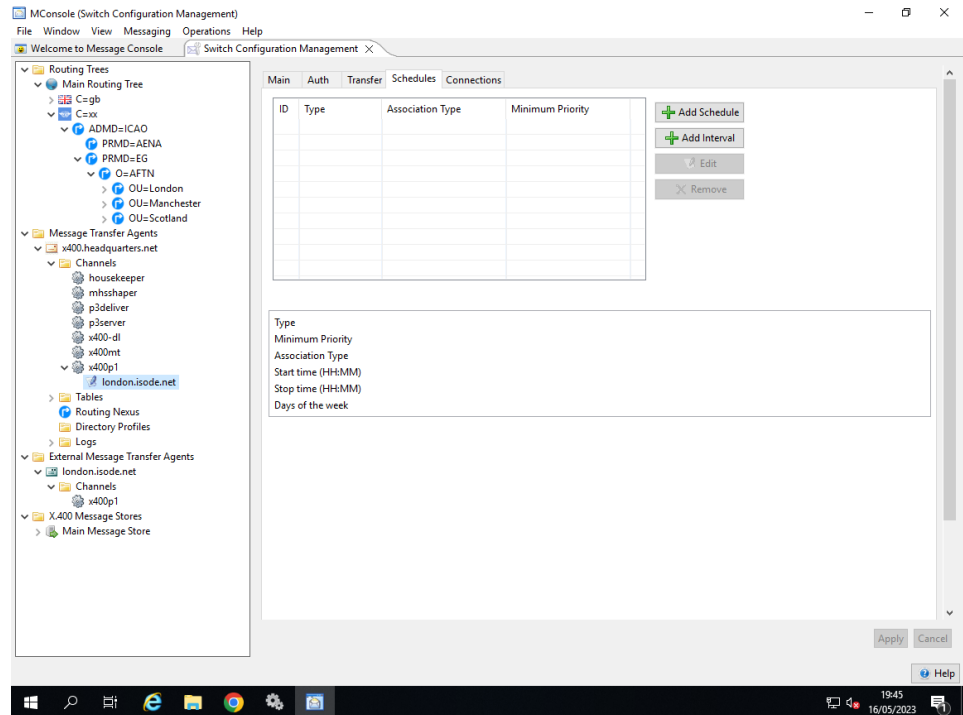
### Window

The window value is the number of minor synchronization points that may occur during a connection. The initiator value specifies the *maximum* number of synchronization points, whilst the responder proposes the *minimum* number of synchronization points.

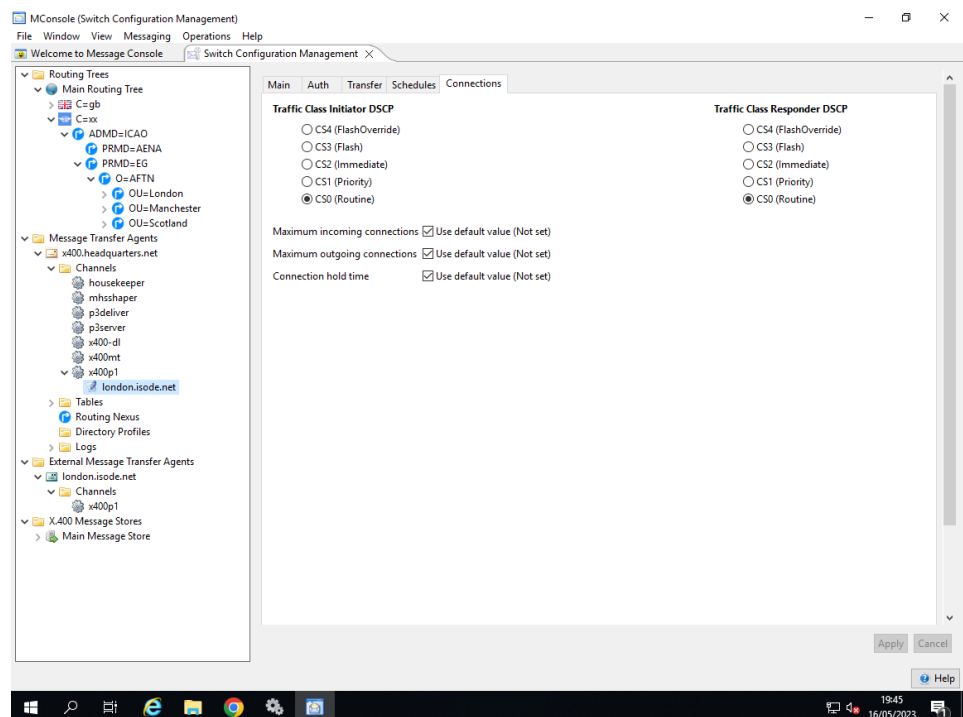
### Checkpoint

The checkpoint value is the size of the RTS data unit (measured in units of 1024 bytes). If the initiator proposes 0, then the responder is free to choose any value.

The **Schedules** tab allows the P1 associations to be configured as Permanent Associations for all or part of each 24 hour period.

**Figure 16.11. Peer Connection Schedules tab.**

The **Connections** tab allows the P1 associations to be configured as Permanent Associations for all or part of each 24 hour period.

**Figure 16.12. Peer Connection Connections tab.**

The sequence of events when MTA A tries to communicate with MTA B will now be:

1. MTA A performs normal routing calculations, and determines that it needs to connect to MTA B's x400p1 channel, using its own x400p1 channel.
2. MTA A then reads entity information from MTA B's x400p1 channel entry.

3. MTA A reads its own x400p1 channel entry and finds that it has a peer connection with the Distinguished Name of MTA B's x400p1 channel. It uses the information found there to override the information read directly from MTA B's x400p1 channel entry.
4. MTA A uses this combined information to make a connection attempt.

## 16.2.4 X.400 Connectivity Example

Some aspects of MTA connectivity configuration may be held in either the MTA channel configuration or a Peer Connection. An example of this is the X.400 MTA bind credentials. The simple credentials (consisting of a simple password) are configured in the Authentication tab of the X.400 P1 channel editor. In this case, the MTA will always use the same credentials on all P1 associations.

If a pair of MTAs wish to use credentials which are different from credentials used for P1 associations to other MTAs, this must be configured in a Peer Connection.

Other features of MTA channels can only be configured in a Peer Connection. An example of this is configuring that an MTA is temporarily unavailable and should be ignored for Routing, as in [Figure 16.8, "Peer Connection."](#)

When a Peer Connection with another MTA is configured, the qmgr treats this as a Permanent MTA and is represented in MConsole **Switch Configuration Management** view as a child entry of the **x400p1** channel. In the **Switch Configuration Management** view, it appears as an entry in the table in the **MTA Links** tab of the X.400 P1 channel editor.

---

**Note:** A Permanent MTA is different from the configuration of a Permanent Association. The latter is the creation of P1 Association by the qmgr when it starts. (This is configured within a peer connection.)

---

For more details on peer connections, see [Section 16.2, "Peer Connections"](#).

## 16.2.5 Peer Connections and Permanent MTAs

Any MTA which has a peer connection configured with another MTA regards the other MTA as a permanent MTA. There are a number of features which may be configured in a permanent MTA which cannot be configured for non-permanent MTAs.

When MConsole has an MTA in its **Switch Operations** view, any permanent MTAs configured for this switch will always be represented (hence the name permanent). You can use MConsole to open an association to such a permanent MTA.

Permanent MTAs are configured by editing the peer connection for a pair of MTAs using MConsole.

### 16.2.5.1 Scheduled associations

You can edit the peer connection of a permanent MTA by using MConsole to edit the values on the **Schedules** page to configure time periods when the association is held open. See [Figure 16.11, "Peer Connection Schedules tab."](#)

If you configure the time period to be permanent, a permanent association is created (NB this is different from a permanent MTA). This means that whenever the MTA is started, the qmgr will automatically open an association to this permanent MTA. This association will stay open until the qmgr is closed down. This is in contrast to MTAs without a permanent association in which associations are opened 'on demand', i.e. when there is a message to transfer.

The advantage of this is that when messages arrive to be relayed to an MTA with a permanent association, there is no cost incurred in opening the association. This will therefore reduce the latency of messages improving performance.

### 16.2.5.2 Limiting by priority

A scheduled or permanent association may have the priority of messages restricted, e.g. to urgent messages only. (Military priorities are also supported.) Again this is configured on the **Schedules** page of the peer connection. Such an arrangement is used when you wish to reserve a permanent association for certain categories of message in order to ensure that throughput of messages above a certain priority is optimised.

See [Figure 16.11, “Peer Connection Schedules tab.”](#)

### 16.2.5.3 Differentiated Service Code Point

Differentiated Service Code Point (DSCP) is defined in RFC 2474 and RFC 2475. It specifies how a TCP connection can be used in which different service characteristics are configured.

DSCP specifies how the top six bits of the Traffic Class for IPv6 and the top six bits of the Type Of Service (TOS) byte in IPv4 can be set.

Values for DSCP of particular relevance are:

- Routine Precedence
- Priority Precedence
- Immediate Precedence
- Flash Precedence
- Override Precedence

You can use this setting to ensure that network traffic is sent at the appropriate priority. This can help to ensure that high priority messages get through even when the network is saturated.

See [Figure 16.12, “Peer Connection Connections tab.”](#)

---

## 16.3 Adding gateways

A gateway MTA is special external MTA which provides a Gateway Application Context. A common reason for using a gateway MTA is as an interface to an alternative communications protocol of some type. In addition to setting up the gateway MTA, any internal MTA which wishes to communicate with the gateway MTA will need to have a channel which can provide the necessary protocol added.

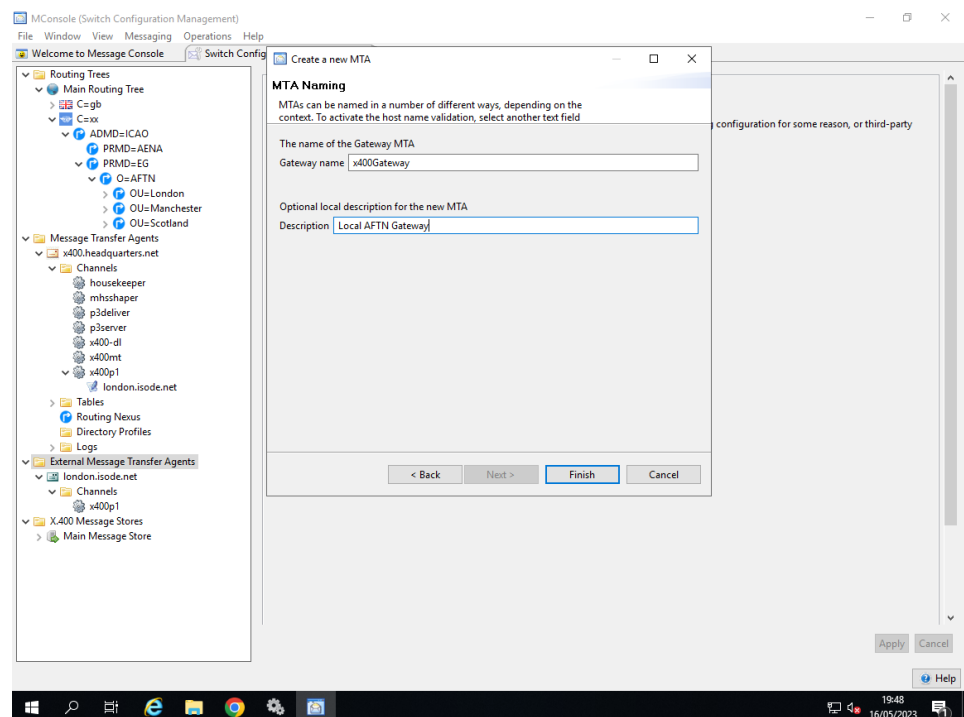
The distinguishing feature of a gateway channel is that there is no program configured in the Program Tab. When a message is queued on a gateway channel, the qmgr cannot schedule a channel to process the message and it stays there until a gateway program such as an XAPI application or an Isode MT Gateway application binds to the qmgr to transfer the message out of the MTA and into the gateway. Similarly for messages to be transferred out of the gateway into the MTA, an XAPI application or an Isode MT Gateway application can connect to the qmgr to transfer the message into the MTA. See [Appendix A, \*Messaging APIs\*](#) and the *M-Switch Advanced Administration Guide* for more details on these APIs.

## 16.3.1 Adding a gateway MTA

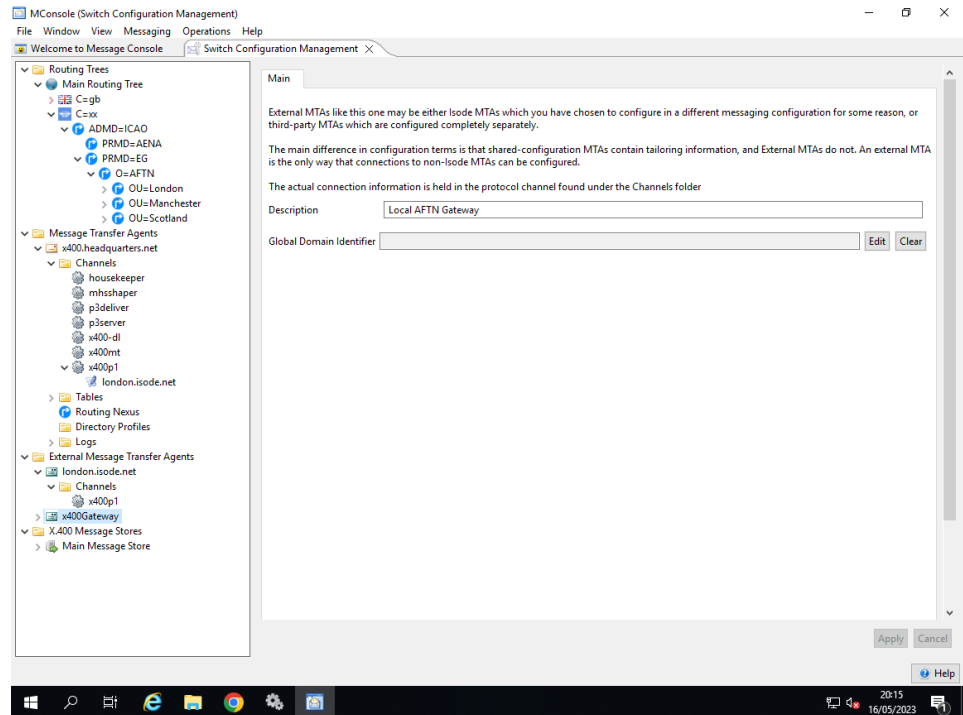
To be able to transfer messages into or from a message transfer gateway, you must create a gateway MTA. A gateway MTA is not a real MTA but represents the X.400 half of the gateway between the X.400 messaging network and the non X.400 messaging network.

Follow the procedure for adding an external MTA which is described in [Section 16.1.1, “Creating an external X.400 MTA”](#), but choose a protocol type of **Gateway**. You will generally not need to provide much information for the gateway MTA – it is really just a placeholder which allows message routing to work correctly. The MTA’s application context will be set to MTS Gateway automatically. Before you create the gateway MTA, set up an appropriate **Routing Tree** branch to represent the X.400 O/R Address space which the gateway MTA will serve.

**Figure 16.13. Creating a gateway MTA.**



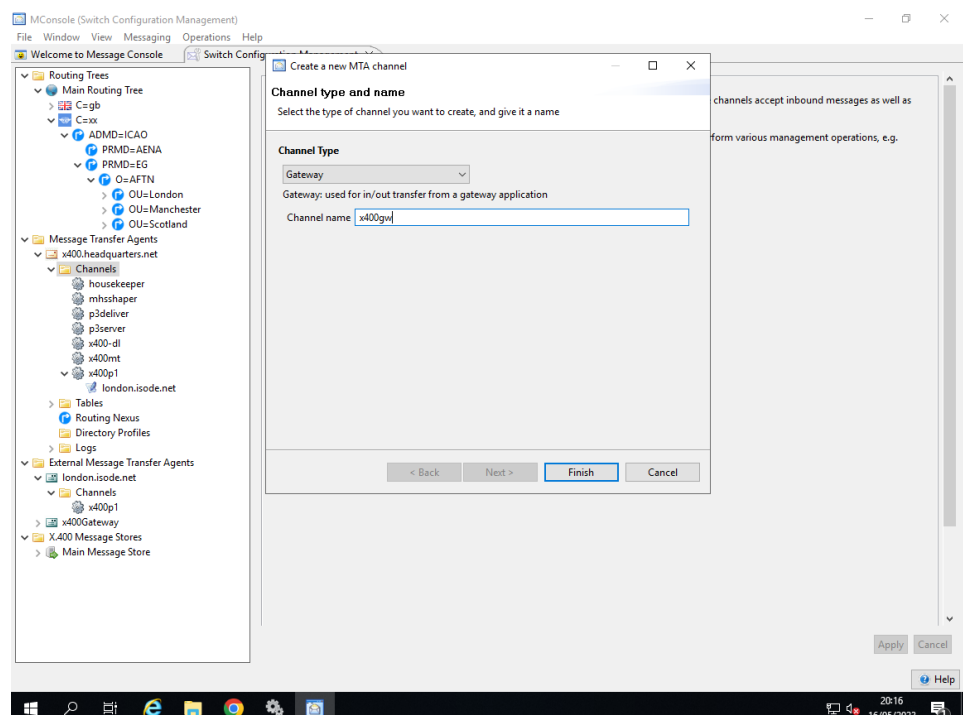
After creation the Gateway MTA appear as in [Figure 16.14, “Gateway MTA after creation.”](#)

**Figure 16.14. Gateway MTA after creation.**

## 16.3.2 Adding a new gateway channel to an MTA

Unlike most M-Switch channels which are started by the qmgr, gateway channels are started as programs, e.g. from the command line, or as Windows Services.

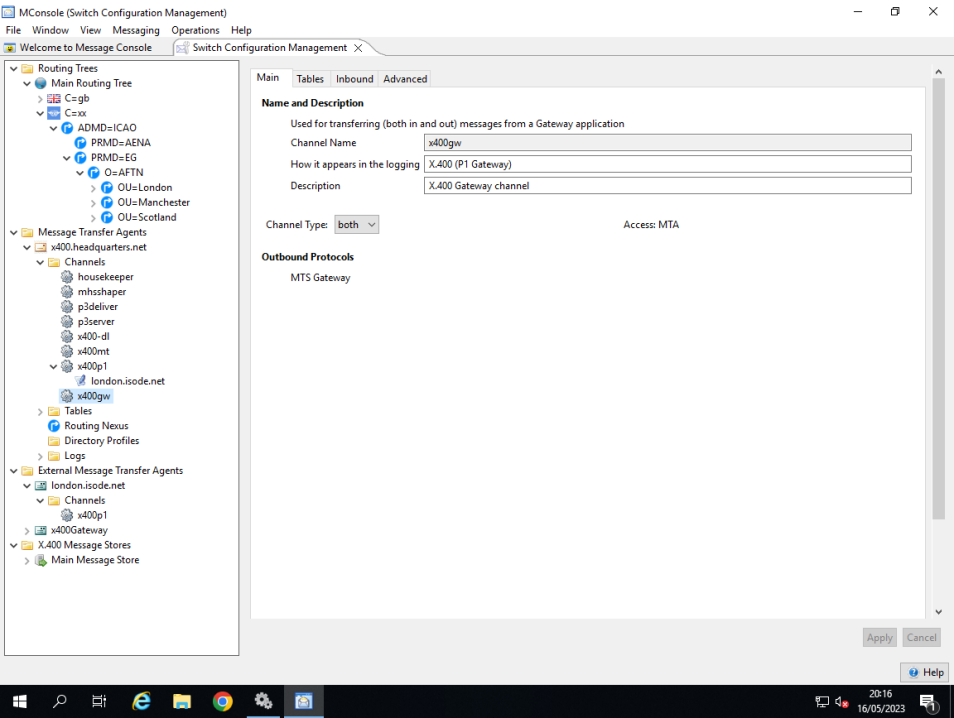
You now need to create a gateway channel on the internal MTA from which you want to transfer messages to the gateway MTA. Right click on the **Channels** folder under the appropriate MTA folder, and select the **New channel** button. A gateway channel is configured as a protocol channel. You only need to specify the name of the channel.

**Figure 16.15. Gateway channel protocol and mode selection.**



The created Gateway channels appear as in [Figure 16.16, “Gateway channel after creation.”](#)

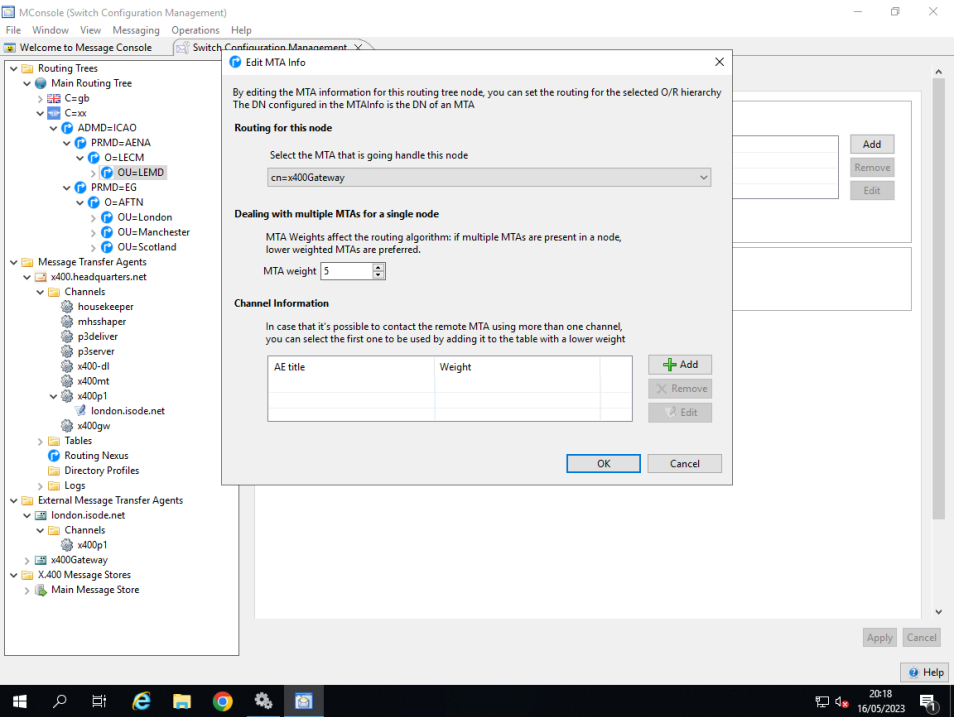
**Figure 16.16. Gateway channel after creation.**



16.3.3 Adding a gateway MTA to the routing tree

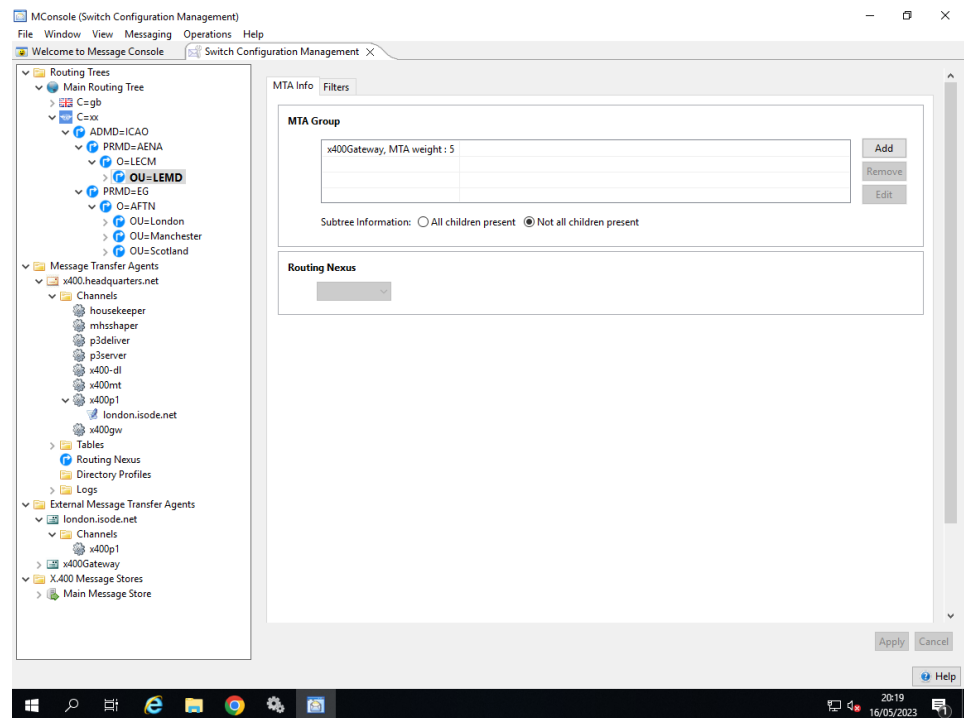
In order to route messages to your gateway MTA, the MTA must be associated with a node on the **Routing Tree**. You may need to add a new node, or series of nodes, to the **Routing Tree** to represent the O/R Address space which is managed by the gateway MTA. Once you have done this, you can set the gateway MTA to be the MTA to which messages for this address space should be routed.

**Figure 16.17. Adding a new gateway MTA to a routing tree.**



To add an MTA to a **Routing Tree** node, select the node and click **Add** on the node's **Main** tab. The pane shown in Figure 16.17, “Adding a new gateway MTA to a routing tree.” will be displayed. Select the gateway MTA from the pulldown list and click **OK**. The MTA Information list for the Routing Tree node will be updated as illustrated below.

**Figure 16.18. The routing tree after adding the gateway MTA.**



## 16.4 Configuring P1 authentication

This section of the manual describes how to configure peer MTA connections. There are three levels of authentication:

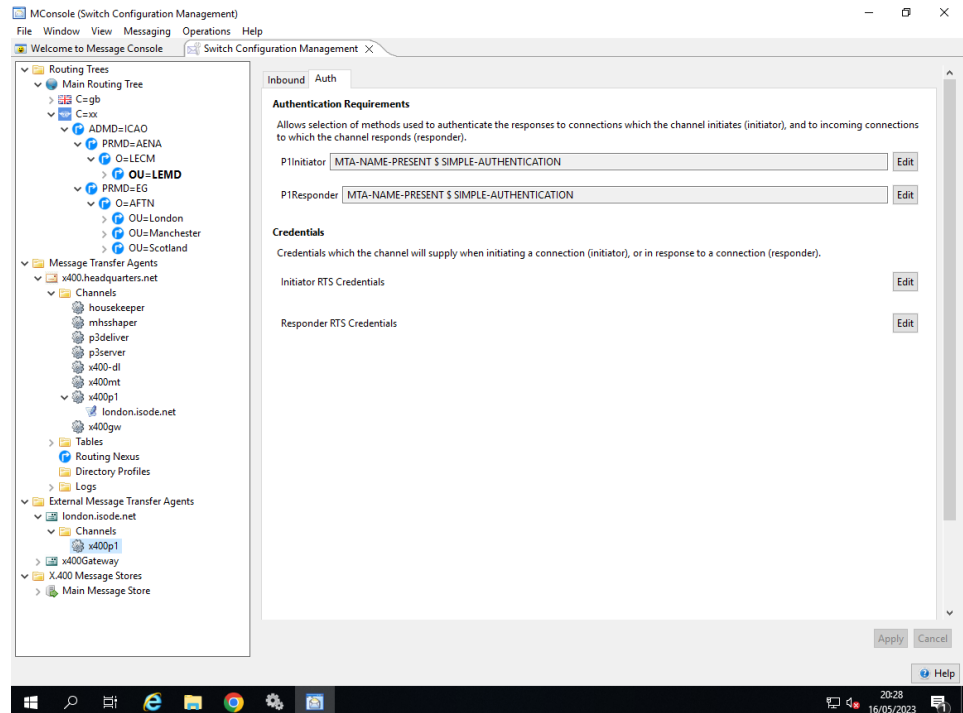
- Anonymous (this is deprecated)
- Simple: using MTA Name and password
- Strong: using MTA Name and a cryptographically signed token

The way in which MTAs authenticate each other is controlled by the **Authentication Requirements** held in the **Auth** tab of the X.400 P1 channel editor and/or Peer Connections editor associated with the MTA pair.

There are a number of different requirements which can be mandated including the use of Peer Connection.

### 16.4.1 Simple authentication

To configure Simple Authentication you should ensure the **Authentication Requirements** for the Remote MTA P1 channel are configured to include **Simple Authentication** and **MTA Name Present** present. You should also select the **AET Valid** if the MTA shares this configuration – do not select **AET Valid** for external MTAs.

**Figure 16.19. Configuring Simple Authentication**

You can configure different credentials to be used when Initiating and Responding, however more often they will be the same. If you leave the credentials empty a default value of a single space is used. The MTA is (by default) configured to treat a single space and a zero length P1 password as matching.

---

**Note:** These default **Authentication Values** and **Authentication Requirements** are used when connecting to all other MTAs. To override the values for a particular MTA pair you need to configure a Peer Connection. See [Section 16.2, “Peer Connections”](#) for details on how this is done.

---

After making any changes, click **OK** and then click on **Apply** to change the channel configuration.

## 16.4.2 Strong authentication

For details on how to set up strong authentication, please refer to the *M-Switch Advanced Administration Guide*.

---

## 16.5 P1 file channel

The p1file channel is another gateway channel similar in operation to XAPI and Isode MT Gateway clients, except that the interface is purely file based. The channel comes in two parts: an inbound (P1 File Server) which reads P1 files from filestore, and an outbound process (P1 File Client) which writes files into a directory. The processes read from and write to binary files containing the BER encoding of P1 messages.

You can configure the P1 File Channel using table-based or Directory based configuration.

## 16.5.1 Directory based configuration

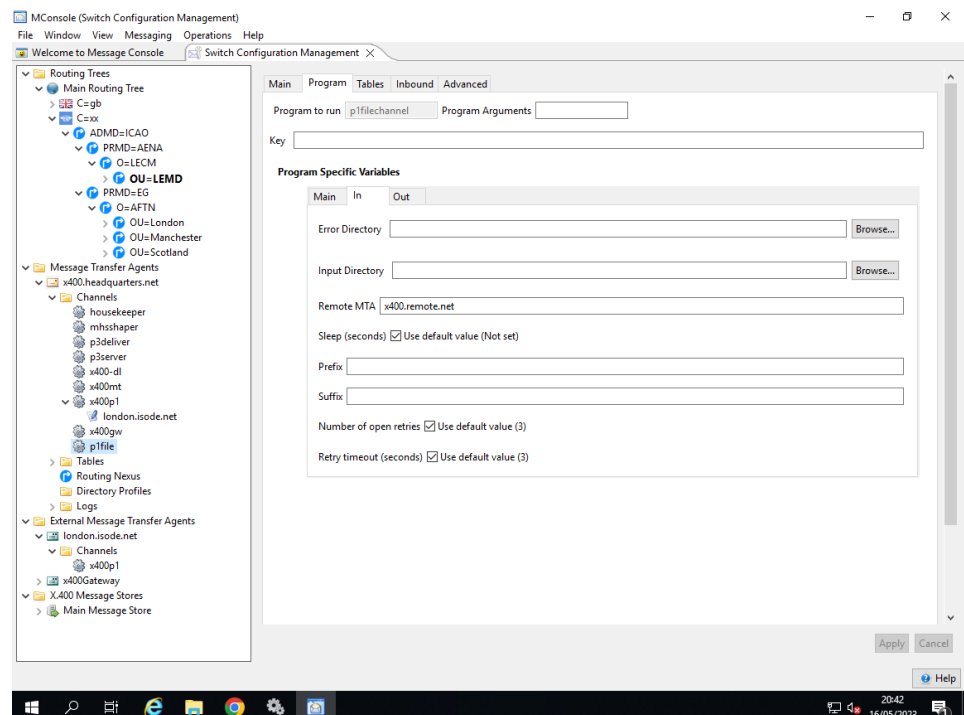
The following discussion assumes that you have already set up a standard X.400 messaging configuration. This is described in [Chapter 9, Configuring an X.400 Messaging System](#).

Next, use MConsole to create P1 File Channel. Expand the MTA where you want to create a P1 File Channel, and right click the **Channels** folder. Select the **New Channel** option. In the first wizard page, select the **P1 File** option. The wizard gives the channel a default name of **p1file**. Either change the name or click on **Next**. Enter the MTA Name in the second wizard page, and then click on **Finish**.

The channel is now created. You should then configure the values on the **In** and **Out** sub-pages on the **Program** tab of the channel editor. Mandatory values are:

- **In: Remote MTA**

**Figure 16.20. P1 file channel: editing the In page.**



Use MConsole to create a new routing tree entry to represent the O/R Address space that will be served by the P1 file channel.

Use MConsole to create a new MTA:

- set **Configuration Type** to **P1 File Gateway**
- set **P1 File gateway name** to the value you want.

After the wizard is complete you will need to fill in a value for the **MTA Name** which should be `p1filemta`.

You will then need to configure one or more Routing Tree nodes to route the O/R Address space for which to be used on the other side of the P1 File gateway.

## 16.5.2 Table-based configuration

### 16.5.2.1 P1 file channel

A typical *mtataylor* entry for the p1file channel would look like:

```
chan plfile type=both name=plfile prog=plfilechannel
key=plfile
show="P1 file transfer channel" content-out="p2,p22"
outinfo="outdir=/var/isode/plfile/outbound/%m"
ininfo="remote_mta=plfile"
bptout="ia5, g3fax, ttx, videotex, national,
encrypted, undefined, voice, tif0, bilateral, odif,
iso6937, external, tif1" hdrout="p2, p22"
inadr="X.400" outadr="X.400"
contentin="p2, p22"
contentout="p2, p22"
```

---

**Note:** Neither inbound nor outbound tables are required.

---

### 16.5.2.2 P1 file client

The outbound side of the plfile channel acts as a standard transfer-out channel. Messages and reports are written to output directories. The output directory paths may include a component which identifies the destination MTA, if required. The `outinfo` settings supported by the plfile outbound channel are:

`outdir=<directory path>`  
 optional; configures where the outbound channel writes outbound messages to. The default value is `/var/isode/plfile/outbound` on Unix and `C:/Isode/plfile/outbound` on Windows.

---

**Note:** On unix, this directory must be readable/writeable by the PP user. If "`%m`" is specified in the path, it will be replaced by the name of the outbound MTA.

---

`outfile=<xxx>`  
 optional; configures a template for outbound file names. If used this must include "`%q`" somewhere - this will be replaced with the `QueueIdentifier` of each message written. If not specified, the default name of output files is just the `QueueIdentifier`.

### 16.5.2.3 P1 file server

The plfile server process `plfileserver` monitors a specific input directory for new files. When a file arrives in the input directory, the process reads the contents of the file and attempts to decode it as a P1 MTS APDU (i.e. an ASN.1 Sequence of envelope and content). If this is successful the resultant message is submitted. Messages which cannot be decoded, or for which submission fails for some other reason, are moved to an 'error' directory. The default behavior of the process is to ignore files in its input directory whose names start with `.` or `~`. Additional configuration allows only files with a particular suffix (e.g. `.msg`) to be recognized, and can also enable high/medium/low priority sorting. On Unix platforms the server process simply scans its input directory regularly, looking for new files (the default is to check once per second). On Windows, use is made of filesystem notification functions, so a regular scan is not required.

Messages which cannot be submitted for some reason (e.g. failure to decode the contents of the file) are moved into an 'error' directory for manual analysis.

On Unix, `plfileserver` should be run as a daemon, in the same way as the SMTP inbound channel. The following command line switches are supported:

`-c <channelname>`

The channel as which the server process should run (this parameter is mandatory).

`-m: <mtaname>`

(optional) The inbound MTA as which the server process should run.

On Windows the p1fileserver program should be installed as a Windows service. This can be done by running a Tcl script through a custom Tcl interpreter:

```
>C:
>cd \Program Files\Isode\bin
>ismsh plfile_install
```

This will install the p1fileserver program with the default service name of isode.pp.plfile, running as channel plfile. Command line options to the Tcl script allow you to override various settings when performing the installation, or even to install multiple versions of the program. For a complete list, type:

```
>ismsh plfile_install -help
```

Once the service is installed, you can start it using the **Isode Service Manager**. If you wish to run the service as a standard program (i.e. on the command line), you will need to specify the -d (debug) command line flag, in addition to the mandatory channel name (-c) command line flag and argument; i.e:

```
>C:
>cd \Program Files\Isode\bin
>p1fileserver -d -c plfile
```

Once the p1file service is running, all of the rest of its configuration information is read from the relevant channel entry in *mtatailor*. As most of the configuration information is shared between the submission and delivery processes, it is described separately, in [Section 16.5.2, “Table-based configuration”](#). The *ininfo* settings supported by the p1file inbound channel are:

*indir*=<directory path>

optional; configures where the inbound channel expects messages for submission to be placed. The default value is */var/isode/p1file/inbound* on Unix and *C:/Isode/p1file/inbound* on Windows. Note that on unix, this directory must be readable/writable by the PP user.

*remote\_mta*=<mta name>

mandatory unless the p1file program has been run with the -m <mtaname> switch. Configures the MTA name which the channel uses when submitting messages.

*sleep*=<nnn>

optional (only used on Unix); configures the length of time for which the channel will sleep between processing one set of inbound messages and checking for a new set.

*acceptall*

```
C:\Program Files\Isode\bin>ckadr -x
"/C=XX/A=ICAO/P=AENA/O=LECM/OU=LEMD/CN=LEMDYMYA/"
/C=XX/A=ICAO/P=AENA/O=LECM/OU=LEMD/CN=LEMDYMYA/ -> (x400)
/CN=John Smith/OU=LEMD/O=LECM/PRMD=AENA/ADMD=ICAO/C=XX/

/C=XX/A=ICAO/P=AENA/O=LECM/OU=LEMD/CN=LEMDYMYA/ -> (rfc822)
"/CN=John Smith/OU=LEMD/O=LECM/PRMD=AENA/ADMD=ICAO/C=XX/"
@x400.headquarters.net

Delivered to x400.headquarters.net by p3deliver (weight: 0)
```

optional. If this is set, any recipient addresses in the inbound messages will be accepted immediately and non-delivery reports generated for any invalid addresses, rather than causing a submission failure.

`prefix=<xxx>`

optional; defines a file prefix which is required for input files (i.e. files which do not have this prefix will be ignored). Can be either a fixed character string (e.g. "msg. ") or "%p". In the latter case, the p1file process will expect all message files to start with a priority indicator character of "h", "m" or "l", indicating high, medium or low priority respectively. Messages will then be processed in priority order.

`suffix=<xxx>`

optional; defines a file suffix which is required for input files (i.e. files which do not have this suffix will be ignored). Must be a fixed character string (e.g. ".msg").

`errdir=<path>`

optional; configures where the inbound channel writes messages which cannot be submitted for some reason. The default value is */var/isode/p1file/errors* on Unix and *C:/Isode/p1file/errors* on Windows.

# Chapter 17 Connecting to other Military MTAs

This chapter contains information about how to configure an ACP 142 (P\_Mul) channel. You can send and receive either of Internet or X.400 messages using the ACP 142 channel.

Please refer to [Chapter 16, \*Connecting to other X.400 MTAs\*](#) for general information about creating MTAs and X.400 specific configuration.

Please refer to [Chapter 15, \*Routing\*](#) for general information about Routing..

---

## 17.1 ACP142 Channel Configuration

ACP 142 P\_Mul 'A Protocol for Reliable Multicast Messaging in Constrained Bandwidth and Delayed Acknowledgement (EMCON) Environments' is a CCEB standard for multicast and EMCON support

ACP 142 is specifically designed to support NATO's *STANAG 4406 Annex E*.

It is a replacement for the full P1 stack and is used in circumstances where the network is characterised by high latency, low bandwidth and high error rates. It includes support for EMCON (Emission Control) which enables the receiver of the message to avoid or delay sending an acknowledgement in protocol. It also is aware of the military priorities and ensures that these are observed so the higher priority messages are optimised.

ACP 142 can be used to carry Internet messages using MULE as specified in *RFC 8494 Multicast Email (MULE) over Allied Communications Publication (ACP) 142..*

ACP 142 is an end to end protocol, providing reliable multicast data transfer, that operates over a datagram service which can either be:

- STANAG 5066 using UDOP (Unreliable Datagram Oriented Protocol)
- IP using UDP (Unreliable Datagram Protocol) or;

ACP142 can be configured to work in three different ways:

- Single recipient (Unicast). ACP 142 is fundamentally a multicast protocol, and unicast is a special case. For Unicast, the standard IP address of the recipient is used.
- Static multicast. Here a multicast IP address is assigned to a fixed set of recipients. This is useful for very small networks and for frequently used combinations of recipients in large networks. A static multicast address can be used without any special negotiation. Most simple deployments will use a single static multicast address that reaches all recipients and will be used for all multicast messages. It will be useful to setup static multicast addresses in complex IP networks in order to optimize network traffic and to send network traffic only to those systems necessary.
- Dynamic multicast. Each sender has a set of IP multicast addresses reserved for dynamic multicast. ACP 142 allows the sender to negotiate a specific set of recipients to be associated with one of these addresses. This allows dynamic multicast to be used for an arbitrary set of recipients. This is useful for nets with a large number of endpoints, as it optimizes performance for systems not receiving a specific message.

Unlike most channels, the acp142 channel is not started by the qmgr. The acp142 channel is a static process which needs to be started when the MTA starts up. The acp142 operates bidirectionally so only one instance is ever needed. Although it is configured as a protocol channel, it needs to be started as a daemon or service.



You can create an ACP142 configuration in one of two ways:

- by creating an MTA with the appropriate channel, which can be a normal shared-configuration MTA with tailoring information or an external MTA with a single ACP142 channel instance
- by adding an ACP142 channel to an existing MTA.

Typically if you wish to configure your MTA to use ACP142 you would need to

- configure the **Routing Tree** with the OR Address space for which the remote MTAs are responsible.
- add the ACP142 channel to your local MTA
- add an external MTA for each of the MTAs with which you are interworking (at least 2 in practice).

---

**Note:** You can interwork with a single peer MTA, but should note that ACP142 is a multicast protocol.

---

## 17.1.1 Examples of ACP configurations

### 17.1.1.1 Unicast example

By default the above example will allow a message to be submitted from the local MTA to a single recipient on MTA B and be relayed by the ACP142 channel. Similarly, a message can be submitted from the local MTA to a single recipient on MTA C and be relayed by the ACP142 channel.

However, a message to two recipients – one on MTA B and one on MTA C – will fail. To make this work, you need to configure either static or dynamic multicast.

### 17.1.1.2 Static multicast example

Static multicast is characterised as follows:

- Fixed group of recipients for each multicast address
- Useful in small configurations
- Specified for each receiver
- Configured on **ACP 142 Std** page
- Lists IP address of each multicast group to which receiver belongs

### 17.1.1.3 Dynamic multicast example

Dynamic multicast is characterised as follows:

- Group of recipients created on demand and assigned to one of a set of multicast addresses
- Dynamic multicast addresses belong to sender
- Configured on **ACP 142 Adv** page
- Lists IP addresses that sender may use for dynamic multicast
- Most efficient if each sender has its own group of addresses – but ACP 142 can resolve conflicts and so addresses may be shared
- Multiple IP addresses needed to support multiple simultaneous transmissions

## 17.1.2 Adding an ACP142 channel to an MTA

When creating a new MTA you are presented with an option to include an ACP142 channel in the new MTA. If you do not choose this option, you can add the channel afterwards.

When creating a new ACP142 channel ensure that you select the correct channel type. Typically S'5066 for STANAG 5066 networks.

Right click on **Channels** and select **New**.

Set the values for which you are prompted as follows (values not described should be left as default or blank):

- Set **ACP142 IP** or **ACP142 S5066** as the channel type
- Set the channel name.
- Click on **Finish**.

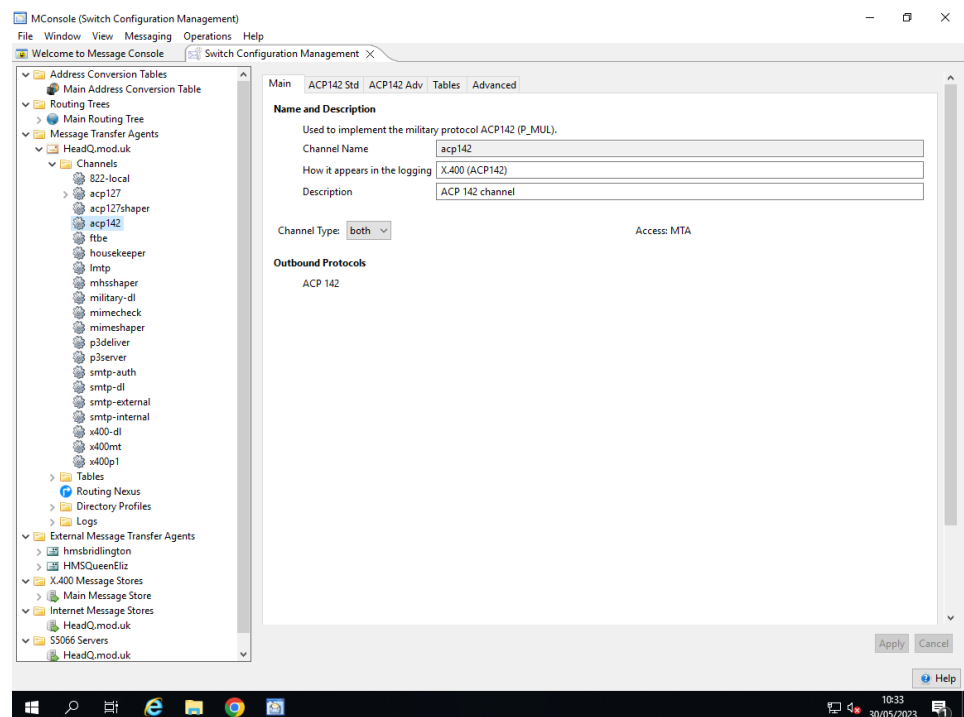
The channel will now be added to the configuration, and can be altered as needed.

## 17.1.3 Configuring the ACP142 Channel

The acp142 channel has tabs which are configured in the channel configuration. The Main, Program and Tables Tab are the same as for most other channels.

### 17.1.3.1 The ACP142 Main tab

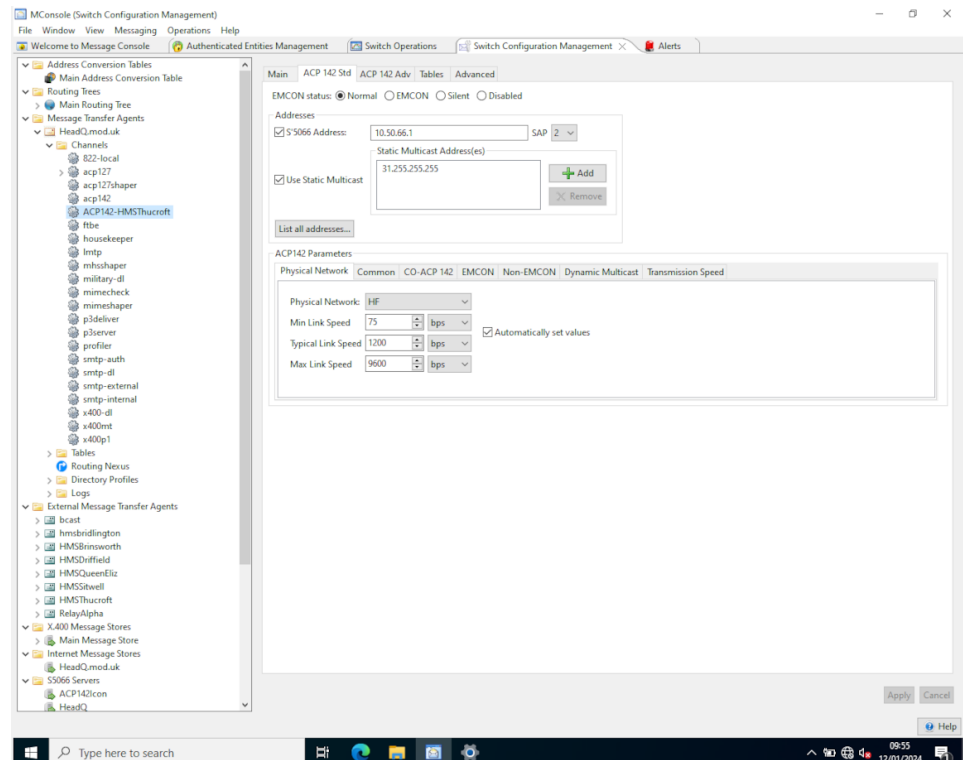
**Figure 17.1. The ACP142 Main tab**



### 17.1.3.2 ACP142 Std Tab

The ACP142 Std Tab configures the more commonly changed values for the ACP142 channel.

Figure 17.2. The ACP142 Std tab



EMCON is an abbreviation of EMmission CONTROL. This is a special mode, which alters the behaviour of the messaging system. Changing the expectation of receiving and transmitting ACKs / NACKs

Possible values are:

- **Normal** Expect ACKS / NACKs to be sent / received normally.
- **EMCON** Expect or transmit ACKS / NACK only when coming out of EMCON.
- **Silent** Never expect / transmit ACKS / NACK
- **Disable** Bounce messages, rather than use this channel.
- Set the unicast address. This will either be the IP address of the interface the ACP142 server should listen on. Or it will be the S5066 node address. The destination ID will be set automatically to this address.
- **SAP (S'5066 networks only)** Default 2. This is the STANAG 5066 SAP ID that defines the SAP ID being used by this application. It has range 0-15.
- Set the static multicast address. The static multicast address is either an IPv4 multicast address (E.G 224.0.0.1) or a S5066 broadcast address (E.G 16.0.0.1). This is optional, and is only of use if there are multiple nodes in your network.

---

**Note:** The **List all addresses...** button shows all the configured unicast and static multicast addresses. This can be useful when picking new addresses for new channels.

---

Some of the settings on this tab are difficult to set, to help administrators, the page will act as an ACP142 "calculator" calculating recommended settings as an administrator works their way down the page.

### 17.1.3.21 ACP142 Parameters

- **Physical Network.**

These settings are used to help calculate other ACP142 Parameters

### **Physical Network**

The type of network being used.

### **Min Link Speed**

The minimum speed of the underlying physical network, measured in bits per second by default. You can also configure this to be measured in units of megabits per second or kilobits per second.

### **Typical Link Speed**

An estimate of how quickly data can be transferred, measured in bits per second by default. You can also configure this to be measured in units of megabits per second or kilobits per second.

### **Max Link Speed**

The maximum speed of the underlying physical network, measured in bits per second by default. You can also configure this to be measured in units of megabits per second or kilobits per second.

- **Common**

### **Max Recv PDU Size (Bytes)**

This is the largest PDU that may be received. This should be set to be larger than the **Max Tran PDU** on all systems on the network.

### **Config Load (Secs)**

This is the interval at which the channel checks and, if necessary, reloads its configuration.

### **Last PDU Timer (Secs)**

This controls how often after the last data PDU is received that the server should check for missing PDUs (ACP 142(a) 332-334).

### **Undef Data Valid (Secs)**

This defines the time for which a data PDU not associated with a message should be discarded (ACP 142(A) 339-40)

### **Max Tran PDU Size (Bytes)**

This is the maximum size of MTU. For IP networks, this will typically be set to MTU size of the network (e.g., 500 or 1500) and for STANAG 5066 it will typically be set to 2048

### **Multicast TTL**

This sets the IP TTL (Time To Live) parameter, reflecting the maximum number of hops a packet can traverse. It should be set according to the IP network configuration and the maximum number of hops expected.

### **Cache Timeout (Secs)**

This is the time the channel caches information on peers, before checking the directory.

### **MaxMissing (PDUs)**

The maximum number of missing PDUs that can be reported in an ack PDU (ACP 142 MM).

### **ACP142 Hub**

If selected this MTA will act as a hub. So if a message is sent from MTA A to two recipients on MTA B and MTA C. Then MTA B will route the message, just for its own recipient. If the hub option is selected, then MTA B will route the message for the recipients using MTA B and MTA C.

- **CO-ACP 142 (IP networks only)**

### **Conn. Timer (Secs)**

This is a timer for Connection Oriented ACP 142, which is how long a connection is held open when idle. There is no protocol disconnect.

### **Conn. Max PDU Size (Bytes)**

This is the maximum RCOP size used in Connection Oriented ACP 142.

- **EMCON**

- **EMCON Time (Secs)**

- When a receiving system is in EMCON the sender transmits the message, waits for this period and then retransmits. (ACP 142 EMCON RTI)

- **EMCON Count**

- The number of times that a message will be transmitted to a destination in EMCON (ACP 142 EMCON\_RTC)

- **Non-EMCON**

- **Retransmit (Secs)**

- When a receiving system is not in EMCON, the sender will retransmit an Address PDU that indicates the message is finished in the event that it has not received an ack from the receiver. This parameter controls the delay before this retransmission. (ACP 142 RETRANSMISSION TIME)

- **Backoff Factor**

- ACP142 transmissions are given an exponential increase. This factor controls the rate of increase of Retransmit (RE-TRANSMISSION\_TIME) parameter (ACP142 BACK-OFF\_FACTOR).

- **Retransmit Delay (Secs)**

- This delay is inserted between the Address PDU and data when a message is retransmitted. This delay was proposed as a change to the original ACP 142, but is not defined in ACP 142(A).

- **Ack Respond (Secs)**

- The time a receiver waits before retransmitting acks (ACP 142 ACK\_PDU\_TIME)

- **Dynamic Multicast**

Dynamic multicast will typically be used for IP systems with a large number of end points. It will not be used for STANAG 5066.

The default broadcast address must be set if dynamic multicast is used in order to negotiate dynamic multicast setup.

- **End Session (Secs)**

- This timer controls how long after the message transmission that a dynamic multicast group is released (para 319 of ACP 142(A)).

- **Announce Count**

- The number of times an announce PDU used for set up of dynamic multicast groups is sent (ACP 142 ANNOUNCE\_CT).

- **Wait for Reject (Secs)**

- The time waited after sending an announce to establish dynamic multicast to check for rejects (ACP 142 WAIT\_FOR\_REJECT\_TIME).

- **Announce Interval (Secs)**

- The delay between transmitting Announce PDUs (To configure ACP 142 (A) para 415).

- **Wait before Send (Secs)**

- The delay transmitting data (Address PDU) after the final Announce PDU (ACP 142 ANNOUNCE\_DELAY).

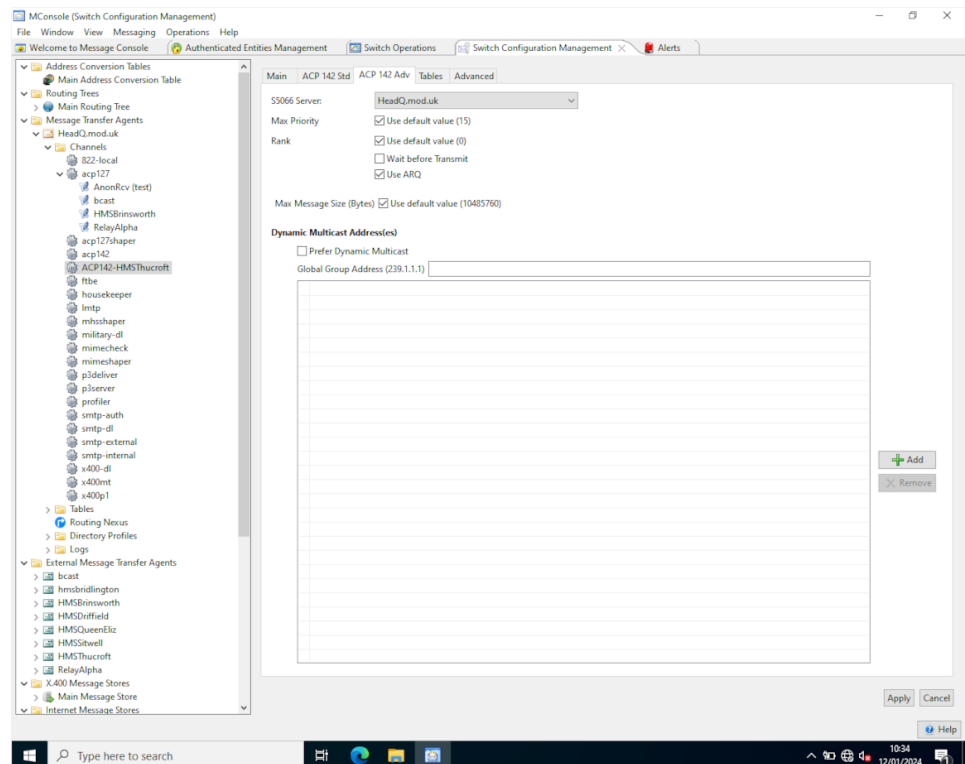
- **Transmission Speed**

The **Transmission Speed** parameter allows you to configure the number of bits per second that the channel will use. You can also configure this to be measured in megabits per second or kilobits per second. For operation over IP, this should be set to a value reflecting the speed at which the network being used operates. For STANAG 5066 transmission, the STANAG 5066 server will control transmission speed and provide flow control back to the application. For STANAG 5066, the rate should be set to a value that is substantially faster than the network. A value of 0 means unlimited.

### 17.1.3.3 ACP142 Adv

The **ACP142 Adv** tab contains information for the sender.

**Figure 17.3. The ACP142 Advanced view for local MTAs**



#### S'5066 Network only parameters

##### S'5066 server:

Select the **S5066** server which this channel will use. If you do not yet have an S5066 Server entity defined in your Messaging Configuration, you should create one before creating the ACP142 channel. This can be done by right-clicking on the **S5066 Servers** folder and selecting **New S5066 Server**. S5066 Servers are described in [Section 19.5.4, "Setting up STANAG 5066 Configuration"](#).

##### Max Priority

Default 15. STANAG 5066 defines priority of data in the range 0-15 with 15 as the highest. STANAG 4406 defines 6 priorities (from DEFERRED to OVERRIDE). ACP 142 defines priority values from 0-256, with 0 as the highest. There is no standard definition of mapping these priorities. Isode uses ACP 142 priorities 0-5 to represent the STANAG 4406 priorities in order. STANAG 5066 priority is defined using this parameter as "Max Priority - ACP 142 Priority". The default setting leads to use of STANAG 5066 priorities 10-15 for the STANAG 4406 priorities.

##### Rank

Default 0. This is the STANAG 5066 Rank that defines the priority of this application relative to other STANAG 5066 clients. It has range 0-15, with 15 as the highest priority rank.

##### Wait before transmit

When using NON-ARQ and S5066 This option will ask the S5066 server to notify M-Switch when a PDU is written out. This improves flow control, and transmission reporting.

##### Use ARQ

When using S5066, this option forces M-Switch to send the S5066 PDUs using ARQ.

#### IP Network only parameters:

**CO-ACP142 Address**

Set the CO-ACP142 address. Set this option if you wish to use CO-ACP142. The value will typically be the same as the unicast address.

**IP flow control**

When performing ACP142 / UDP / IP / S5066 extra flow control operation can be performed by the IP client. The `<hostname>:<port>` for the IP flow control server can be specified here.

**Ports**

The **Ports** section allows the four ports used by ACP142 running over IP to be set. For most configurations, the default values will be used.

**Common parameters:****Max Message Size**

When transferring messages the ACP142 server will automatically compress the message PDUs using the DEFLATE algorithm. After a message is transferred, the receiving ACP142 server will expand the message.

It's possible for an attacker to create a false message with a carefully constructed payload, designed to perform a DOS attack when the receiving server expands the message.

These attacks are called "ZIP Bombs".

By setting the **Max message size**, the ACP142 server will refuse to expand a message beyond this limit.

This stops the "ZIP Bomb", and will cause the receiving ACP142 server to reject the message.

Dynamic multicast will typically be used for IP systems with a large number of end points. It will not be used for STANAG 5066.

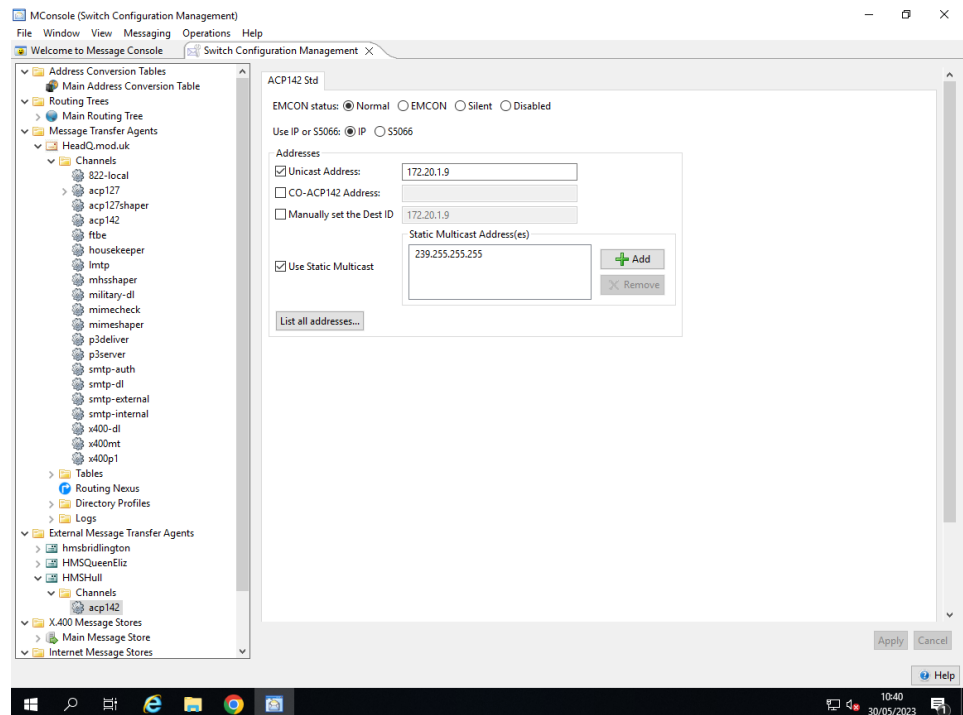
The **Dynamic Multicast Address(es)** section also includes specification of the **Global Group Address**. This is required when dynamic multicast is being used: it is an address used by all servers on the network to negotiate dynamic multicast addresses. This must be set as an IP Multicast address.

If using dynamic multicast, then at least one **Dynamic Multicast Address** must be set up. This should be an IP address. If each MTA has its own set of dynamic multicast addresses, then performance will be optimized.

You can also set a flag to prefer dynamic multicast to static multicast. By default, if a suitable static multicast address can be used, that is preferred as the setup time for the transfer is less. This default is recommended.

### 17.1.3.4 ACP142 In (External MTAs)

When connecting out to other ACP142 MTAs, the remote MTA needs to be configured as an External MTA. The **In** tab represents what other MTAs need to know to connect to it, so external ACP142 MTAs have just this **In** tab.

**Figure 17.4. The ACP142 In view for external MTAs**

The EMCON (EMmission CONTROL or Radio Silence) flag, when set, prevents the owner from sending information. It also alters the way other hosts send to that host, as ACP 142 can optimize transmission for peer MTAs in EMCON. This option has four values.

- Normal.
- EMCON. The remote system will not send any ACKs until it has come out of EMCON.
- Silent. The remote system will never send any ACKs back.
- Disabled. It is impossible to contact the remote system, so fail any messages being transferred to this MTA, and send a Non Delivery Report back to any message originator.

The **Use IP or S'5066** radio button, is used to help configure Static Multicast Addresses. Please see below for more details.

The unicast address is the address to which packets should be sent for this system when multicast is not being used. If ACP 142 is running over IP, this can be an IP address or a domain name that will be resolved to an IP address at run time.

If ACP 142 is being run over STANAG 5066, this is the STANAG 5066 address expressed as a dotted quad. STANAG 5066 addresses are 3.5 bytes long. This is the address of the local system. Which should match the address of the STANAG 5066 server which is being connected to.

The **Connection Address** parameter is used when you are using connection-oriented ACP142. See [Section 17.1.4, "Connection-oriented ACP 142"](#). Set this option when the external MTA is an Isode MTA. CO-ACP142 has performance benefits over regular ACP142.

---

**Note:** CO-ACP142 can only be used in IP based environments

---

The **Dest. ID** is the ACP 142 node identifier. This is a 32 bit number, but is represented as a "dotted quad" like an IPv4 address. It is very commonly set to the IP address of the system. This is mandatory. By default it will be set to the same value as the unicast address.



Static multicast is configured on this tab. For STANAG 5066 and for simple IP configurations a single multicast address should be used and this address should be configured for all nodes.

For complex IP networks, multiple static IP addresses can be used to optimize network traffic. It is recommended to define the set of static multicast addresses in use, and which MTAs belong to which multicast addresses.

The static multicast addresses are IPv4 addresses or STANAG 5066 addresses. Multicast addresses are specified as dotted quads. For IPv4, multicast addresses are in the range 224.\*.\*.\* to 239.\*.\*.\*, but you cannot use 224.0.\*.\* STANAG 5066 defines its own 3.5 byte broadcast address space. Setting the **Use IP or S'5066** radio button will automatically convert the static multicast addresses between an IPv4 and S'5066 addresses.

## 17.1.4 Connection-oriented ACP 142

Connection-oriented ACP 142 is an Isode protocol based on ACP 142 designed to be used over a reliable transport. It provides optimized performance for point to point links. In particular, it is targeted at RCOP/STANAG 5506, and also supports TCP/IP. As it assumes that error correction is performed by the network transport, it is only used when a message is for one destination (i.e. no multicast) and that destination is not in EMCON (i.e. it can transmit).

The configuration of this is performed by configuring the **CO-ACP142 Address** in the **ACP142 Std** tab of the channel properties. This applies to both the local MTA and external MTAs.

The value depends upon the underlying network:

- For S5066, the address is the S5066 address of the system, e.g.

```
1.5.0.2
```

- For TCP/IP the address is the IP address or hostname of the system, followed optionally by : and a port number. If the port number is not specified, then the value of the D-port for the normal UDP/IP network is used (default 2753). E.g.

```
acp142.example.com
acp142.example.com:3753
192.168.0.1:4753
```

## 17.1.5 Testing ACP 142

Checking that messages are routed via the channel by using **ckadr** with the recipient addresses, messages should go through OK. If you turn down the transmission speed, and send a large message, you should be able to follow its progress in MConsole.

If a receiver is in EMCON, then the sender will continue to try sending the message until the receiver sends the ack, or the message times out. You should see the result of the ack by altering the EMCON state in the receiver. The channel should ack messages as soon as it sees the configuration change.

## 17.2 ACP 142 Message Transfer View

The ACP 142 Message Transfer View has been designed for use in Low bandwidth / High latency environments.

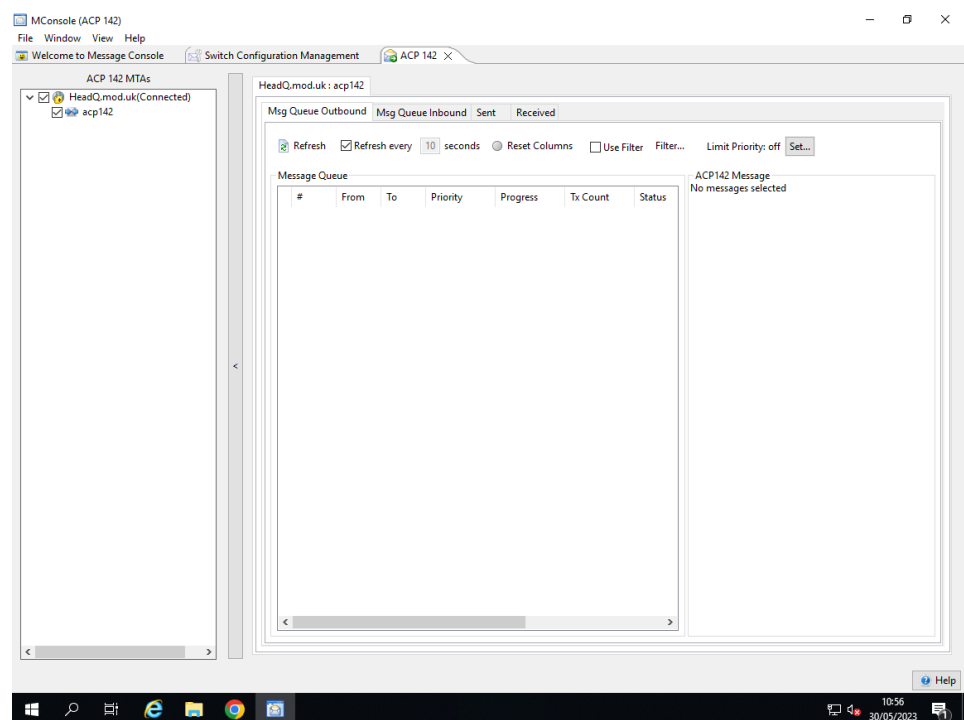
It is recommended that you configure the audit database, before starting the ACP 142 Message Transfer View. The view uses the audit database to show messages that have already been transferred.

It is also advisable that MConsole is connected to your local M-Switch via SOM. If you do not have a switch defined, right click on the left hand side panel, select **Add connection information**.

### 17.2.1 Starting the ACP 142 Message Transfer View

To start the view select **View** → **Live Operations** → **ACP 142 Message Transfer View**

**Figure 17.5. ACP 142 Message Transfer view**



### 17.2.2 SubTabs

#### Msg Queue Outbound

The Msg Queue Outbound tab shows messages being transferred out of the current MTA, on a specific channel.

The top row of buttons and options are described below:

#### Refresh

The Message Transfer View uses SOM to communicate with the M-Switch. In order to gain information from M-Switch the view must poll every few seconds. The **Refresh** option controls that polling.

**Reset Columns**

This option resets the size of the various table columns.

**Use filter**

This option applies a preconfigured filter, so that only a filtered subset of messages are shown.

**Filter...**

This option allows an operator to construct a filter. Messages must meet the criteria set in order to be shown.

The table contains the following columns:

**#**

The message number in the queue.

**From**

The address of the sender or originator.

**To**

The recipient of the message. Can be either an SMTP or X.400 address.

**Priority**

The priority of the message.

**Progress**

A bar graph showing the progress made in transferring a message. Yellow shows the % of the message that has been transmitted. Green shows the % of the message that has been acknowledged by the remote side.

---

**Note:** Under ideal circumstances the graph will turn yellow, with no green being shown. The message will then be ACK'd by the remote side, and removed from the message queue. Only when 1 or more ACP142 data PDUs are missing will an operator see

---

**TX Count**

The number of bytes that has been transmitted

**Status**

The status of the message. This can either be:

**Paused**

The transmission is paused.

**Active**

The message is currently being processed.

**Queued**

The message is queued, waiting to be transmitted.

**Size (bytes)**

Initially the size of the message in bytes. ACP142 messages are compressed however. This means that when a message begins being processed by the channel, the size may shrink.

**Age**

Message Age, based on how long this message has been in the message queue.

**Remaining**

The estimated amount of time until the message finished being transmitted.

**Msg Queue Inbound**

The inbound tab shows messages being transferred in to the current MTA. The various columns that can be selected are:

**From**

Because the message is wrapped up in the ACP142 data PDUs, it's impossible to display the originator / sender information. Instead this is the MTA which is transmitting the message.

**Progress**

A bar graph showing how much of the message has been received.

**Age**

The amount of time the ACP142 server has been receiving this message.

**Remaining**

The amount of time left until the message finished being transferred.

**Messages Sent**

The completed table shows messages that have been successfully transferred in or out of the MTA. The view is based on the Message Tracking View, and is dependent on the Audit Database being set up correctly.

**Messages Received**

The completed table shows messages that have been successfully transferred in or out of the MTA. The view is based on the Message Tracking View, and is dependent on the Audit Database being set up correctly.

### 17.2.2.1 ACP 142 Message controls

When an ACP 142 message is selected, it's possible to issue commands related to that message.

**Hold...**

Holds a non-active message for a set amount of time.

**Let Pass**

Clears any delay Hold... has put on a message.

**Pause**

Pauses the transmission of a message.

---

**Note:** Messages can only be paused, when there are message PDUs to be written out from the ACP 142 layer to the S5066 layer, and the message has started to be transmitted. It's normal for transmission to continue while the S5066 layer continues to write out remaining PDUs

---

**Resume**

Resumes transmission of a paused message.

**Delete...**

Deletes a message from the queue.

**Abort**

Aborts a message being transferred.

# Chapter 18 Security Labels and Access Control

This chapter describes how to configure M-Switch to manage Security Labels, including how it can map between a wide range of Security Label formats and message transport mechanisms when crossing a MIXER gateway or ACP127 gateway. It also discusses how to make use of Security Labels to perform Access Control.

---

## 18.1 ACP127 Security Labels

This chapter covers how M-Switch carries and transfers Security Labels. M-Switch contains powerful and flexible features to carry and manage Security Labels by extracting the Security Label from an incoming message, and inserting a Security Label into an outgoing message.

M-Switch can also use Security Labels in conjunction with Clearances to perform Access Control Decision Functions.

For a full description of these features, see the *M-Switch Advanced Administration Guide*.

### 18.1.1 Security Policy Information Files (SPIF)

A Security Policy is represented as an SDN.801c SPIF in the Open XML SPIF format. A SPIF is structured data which defines for a given policy ID the valid classifications and security categories. It also can define strings to be associated with labels, which are used for mark-up of data for human reading. It can define equivalent policies, which enables labels defined by a different authority to be associated with labels defined in this SPIF. It also defines how the 'Access Control Decision Function' (ACDF) is to be applied.

It is possible to configure M-Switch as a Gateway without a SPIF which will result in Security Labels traversing the Gateway in a very simple manner. Generally it is highly likely that a SPIF will need to be used.

A default SPIF is installed in *(SHAREDIR)/policy.xml*. M-Switch is installed with other configuration files (described later in this section) which refer to this SPIF.

While this SPIF is suitable for demonstration, testing and evaluation it is not suitable for NATO or any other deployment. It uses some values which are aligned to NATO values, but in real world deployments a SPIF tailored to the security environment of the deployment is required.

#### 18.1.1.1 M-Switch Conversion Configuration Files

Configuration of the Shaper channels is held in configuration files installed into *(SHAREDIR)/switch*:

*mimemixer-shaper.xml*

This file configures how the mimeshaper channel converts Internet (SMTP) messages.

*mhsmixer-shaper.xml*

This file configures how the mhsshaper channel converts X.400 messages.

*acp127-shaper.xml*

This file configures how the acp127shaper channel converts ACP127 messages.

Lastly, all channels which accept message submissions use *submitscanconfig.xml* to configure behaviour on submission, e.g. Security Label extraction.

When editing these configuration files, a copy should be placed in *(ETCDIR)/switch* and edited. M-Switch will use this version (if present) to override the value of the installed copy in *(SHAREDIR)/switch*. This avoids having configuration files overwritten on updates or upgrades.

## 18.1.1.2 Configuring M-Switch Conversion

### 18.1.1.2.1 Configuring a SPIF as a security Policy

Each of the configuration files in [Section 18.1.1.1, “M-Switch Conversion Configuration Files”](#) the default SPIF is configured as follows security-policy element at the start

#### Example 18.1. Configuring a Security Policy (submitscanconfig.xml)

```
<?xml version="1.0" standalone="yes"?>
<submitscan>
  <security-policy name="acpl27-scan"
    default-policy-id="1.2.6.1.4.1.453.28.1687253052.1.1">
    <spif file="policy.xml"/>
  </security-policy>
```

#### Example 18.2. Configuring a Security Policy (mhsmixer-shaper.xml)

```
<shaper template="shaper.template.xml" exploder="ppmhs"
  description="Conversion of P2 and P22 content">
  <security-policy name="acpl27-scan"
    default-policy-id="1.2.6.1.4.1.453.28.1687253052.1.1">
    <spif file="policy.xml"/>
  </security-policy>
```

#### Example 18.3. Configuring a Security Policy (mimemixer-shaper.xml)

```
<shaper template="shaper.template.xml" exploder="ppmime"
  description="Conversion of MIME content">
  <security-policy name="acpl27-scan"
    default-policy-id="1.2.6.1.4.1.453.28.1687253052.1.1">
    <spif file="policy.xml"/>
  </security-policy>
```

---

**Note:** In each of the above examples you will need to ensure you have the correct default-policy-id and file values. The file value is the path relative to *(ETCDIR)*.

---

### 18.1.1.2.2 Extract a Security Label from an X.400 Message

The security-policy name in the param element of the scanner element for the ppmhs exploder as in the below example.

#### Example 18.4. Extracting a Security Label from an X.400 Message (submitscanconfig.xml)

```
<scancontent type="p2" exploder="ppmhs">
  <param type="oid.1.2.840.113549.1.9.16.1.6">explode</param>
  <param type="oid.1.3.26.0.4406.0.4.1">explode</param>
  <scanner type="header" cost="1">
    <filter command="ppmhs:heading-extract">
      <param name="security-policy">acpl27-scan</param>
    </filter>
```

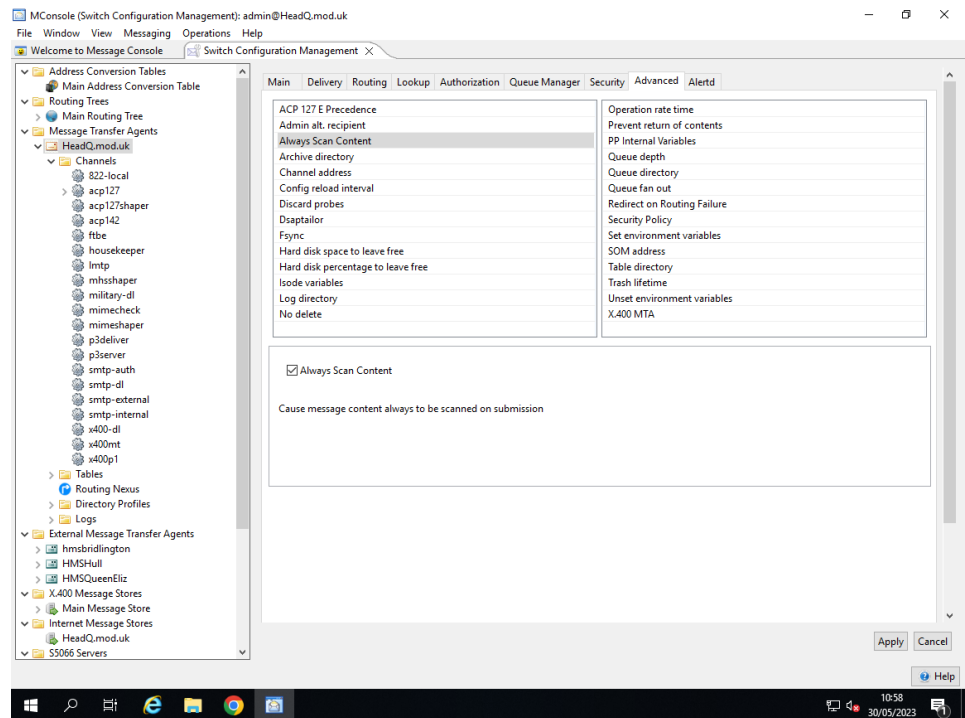
```
</scanner>
</scancontent>
```

### 18.1.1.2.3 Extract a Security Label from an Internet Message

Security Labels in SIO Header of an Internet messages are always extracted automatically when the message is scanned on submission.

However, Internet messages are not always scanned, so to ensure extraction takes place, the PP Internal Variable `always-scan-content = true` must be set as in the following screenshot.

**Figure 18.1. Configuring always-scan-content in the MTA entry**



### 18.1.1.2.4 Extract a FLOT from an Internet Message

Security Labels in the First Line Of Text (FLOT) in an Internet message are configured as in the figure below.

**Example 18.5. Extracting a FLOT (submitscanconfig.xml)**

```
<scancontent type="822" exploder="ppmime">
  <param type="smime-sig">pkcs7-signature x-pkcs7-signature</param>
  <param type="smime-body">pkcs7-mime x-pkcs7-mime</param>
  <!-- other scanner elements omitted for clarity -->
  <scanner type="body" cost="1">
    <filter command="pplablib:flotextract">
      <param name="regexp">
        ( R E S T R I C T E D | C O N F I D E N T I A L
          | T O P S E C R E T | S E C R E T )
      </param>
      <param name="security-policy">acpl27-scan</param>
      <param name="replace">$1</param>
    </filter>
  </scanner>
</scancontent>
```

However, Internet messages are not always scanned, so to ensure extraction takes place see [Figure 18.1, “Configuring always-scan-content in the MTA entry”](#) to configure `always-scan-content`.

### 18.1.1.25 Inserting a Security Label extracted from ACP127 into an Internet Message

There are multiple ways in which a Security Label can be inserted into an Internet message. The example below adds a Security Label extracted from an ACP127 message into SIO, X.411, X-Isode-Label headers and also adds a markup to the subject.

#### Example 18.6. Inserting Security Labels into an Internet message gatewayed from ACP127 (`acp127-shaper.xml`)

```
<!--
    Convert to 822 with only 7 bit encodings
-->
<output type="822" flattener="ppmime">
  <convert type="header" action="convert" cost="1">
    <match name="Depth">1</match>
    <filter command="acp127:header">
      <param name="fold">76</param>
    </filter>

    <!-- various label insertions including subject -->

    <filter command="pplablib:labelinsert">
      <param name="xheader-format">SIO-Label: %e</param>
      <param name="security-policy">acp127-scan</param>
      <param name="convert-label"/>
      <param name="verify-label"/>
      <param name="fixup-label"/>
    </filter>

    <filter command="pplablib:labelinsert">
      <param name="xheader-format">X-Isode-Label: %l</param>
      <param name="security-policy">acp127-scan</param>
    </filter>

    <filter command="pplablib:labelinsert">
      <param name="xheader-format">X-X411: %b</param>
      <param name="subject-format">Subject:
%d (inserted on centos64-2)[SEC=%c]</param>
      <param name="need-policy-id"/>
      <param name="security-policy">acp127-scan</param>
    </filter>

    <filter command="pplablib:labelinsert">
      <param name="xheader-format">SIO-Label: %e</param>
      <param name="security-policy">acp127-scan</param>
    </filter>

  </convert>
```

### 18.1.1.26 Inserting a Security Label extracted from an X.400 message into an Internet Message

There are multiple ways in which a Security Label can be inserted into an Internet message. The example below adds a Security Label extracted from an X.400 message into an SIO header.



### Example 18.7. Inserting Security Labels into an Internet message gatewayed from X.400 (mhsmixer-shaper.xml)

```
<convert type="header" action="convert" cost="1">
  <match name="Depth">1</match>
  <filter command="ppmixer:ipms2rfc">
    <param name="convert-notif-req"/>
  </filter>
  <filter command="pplablib:labelextract">
    <param name="base64">x-classification</param>
  </filter>
  <filter command="pplablib:labelinsert">
    <param name="xheader-format">SIO-Label: %e</param>
  </filter>
</convert>
```

Add the above into the relevant `<output>` section(s). This converts the P22 heading, but only the outer one (matching Depth 1). The `labelextract` filter gets the output fields before the `labelinsert` filter, so it will have found the label and set this as a label for the message prior to the `labelinsert` filter adding the SIO-Label at the end of the message heading fields.

The `%e` means create an ESS type IO-Label value. The `label=` output should have exactly the same as the value in the `x-classification`.

#### 18.1.1.27 Inserting a Security Label extracted from an ACP127 message into an X.400 Message

To insert a Security Label extracted from an ACP127 message into an X.400 message as an X.411 Security Label add the `filter` element as in the example below:

### Example 18.8. Inserting an X.411 Security Label into the Envelope of an X.400 message gatewayed from ACP127 (acp127-shaper.xml)

```
<!-- ===== MIXER =====
      Convert to P22/P2/P772
-->
<output type="p2orp22" flattener="ppmhs" name="To-P22"
      description="Convert to P22">

  <convert type="envelope" action="convert" cost="1">
    <match name="Content-type">acp127</match>
    <!-- insert envelope label -->
    <filter command="pplablib:eninsert">
      <param name="security-policy">acp127-scan</param>
      <param name="verify-label"/>
    </filter>
  </convert>
```

#### 18.1.1.3 Security Labels and Clearances

M-Switch provides Access Control based on a Security Label with each message. M-Switch can require messages to have Security Labels (and reject messages that do not have one) or can add a default Security Label.

The Access Control decision as to whether or not to let the message through is based on comparing the Security Label against a Security Clearance in the context of the Governing Security Policy (which will yield a yes/no decision).

The Security Clearance can be associated with a number of different entities, which enables a number of different sorts of Access Control:

- **Message Recipient.** Here the Security Clearance will typically be obtained from the recipient's entry in the directory. The Security Clearance may be stored directly (e.g., as an X.501/X.509 Security Clearance) or within an X.509 Certificate. This provides control based on recipient.
- **Next MTA.** Here the Security Clearance is associated (in the M-Switch configuration) with a peer MTA, enabling control of next hop MTA. This is currently only available for X.400.
- **Channel.** Enables control by group of MTAs and can also be used to control the route taken to a peer MTA.

M-Switch Access Control prevents messages from going to systems or users who do not have appropriate Security Clearance. This Access Control mechanism can also be used to control message routing based on Security Label.

For a full description of these features, see the *M-Switch Advanced Administration Guide*.

# Chapter 19 ACP127

This chapter describes how ACP127 interworking can be set up and configured.

---

**Note:** The term ACP127 is used here as a general term to refer to ACP127 and similar protocols such as DOI-103s. Distinctions between the forms of protocol are noted where relevant.

---

---

## 19.1 The ACP127 channel

The ACP127 module of M-Switch normally consists of one primary channel that handles all the ACP127 and associated protocols. Messages arrive into M-Switch from the ACP127 world through the channel, where they are either gatewayed to Internet(SMTP) or X.400 networks; or on some occasions relayed on to other ACP127 destinations. In all but very special circumstances, there will only be one ACP127 channel, but there will often be several peer connections. These peer connections are also referred to as circuits.

An ACP127 channel consists of the main channel definition, and a number of peer connections. The peer connections represent ACP127 circuits and are set up on a bilateral basis with remote end stations. Traffic is then routed down the appropriate peer connection as described in [Section 19.3, “ACP127 Routing”](#), utilising whatever ACP127 variant and parameters are appropriate.

Each peer connection points to an External MTA configured as an ACP127 station.

---

**Note:** In ACP127 the connection is also called a 'channel'. This term is not used here to avoid confusion with the use of the term for M-Switch channels.

---

---

## 19.2 ACP127 MConsole Live Operations Views

ACP127 Operations can be monitored in the ACP127 View.

The ACP127 View allows an operator to monitor the state of various ACP127 circuits. The left hand side of the view shows a list of MTAs, selecting an MTA will make the view query the ACP127 Server for a list of available circuits.

---

**Note:** Only viable circuits will be returned by the ACP127 Server.

---

Selecting a circuit will cause a new tab to be opened up for that circuit. This tab contains various subtabs. These subtabs will vary depending on the circuit type.

### 19.2.1 Monitoring

The monitoring tab connects to the ACP127 server and monitors both the outbound and inbound streams if applicable.

It also connects to the Queue Manager to get information on queued messages.

The view combines information from both of these sources to allow an operator control over messages queued on the ACP127 channel.

### Operator controls:

---

**Note:** Not all sending controls are applicable to all circuits. An anonymous receive circuit will not be able to send peer controls (QRT/QRV) for example.

---

### Sending Controls

A circuit can be configured to be two-way, send-only or receive-only. In addition, a two-way circuit can be controlled so that no messages are sent. This control can be done by the operator from the ACP127 View using the 'Change' button associated with the State. The sending state of the circuit can also be changed by the peer station sending a service message. The receipt of QRT from the peer will stop sending, and QRV will resume sending. Whether the receive-only state was set by the operator or by the peer is indicated in the display.

### Peer Control

The peer control is a convenience function allowing an operator to send a service message which, either asks the remote side to stop transmitting (QRT) or asks the remote side to resume transmission of messages (QRV)

### Limit priority

Operators can also set a **MINIMIZE** value. This tells the ACP127 server to restrict messages to at least a given priority. Selecting **Set..** lets an operator do this. It also lets an operator send a service message (ZAN) to the remote peer, requesting the remote peer set its own **MINIMIZE** value.

### Circuit type

If the circuit is configured to allow an operator to switch between automatic and manual control, then this option is present.

The monitoring tab contains the following subsections:

### Outbound traffic

The outbound area shows a stream of outbound bytes, from the ACP127 server. Right clicking and selecting **pop-out**, allows the streaming window to pop out into a separate window. This section is only available when the circuit is a broadcast or two way circuit.

### Inbound traffic

The inbound area shows a stream of inbound bytes, to the ACP127 server. Right clicking and selecting **pop-out**, allows the streaming windows to pop out into a separate window. This section is only available when the circuit is a receiving circuit.

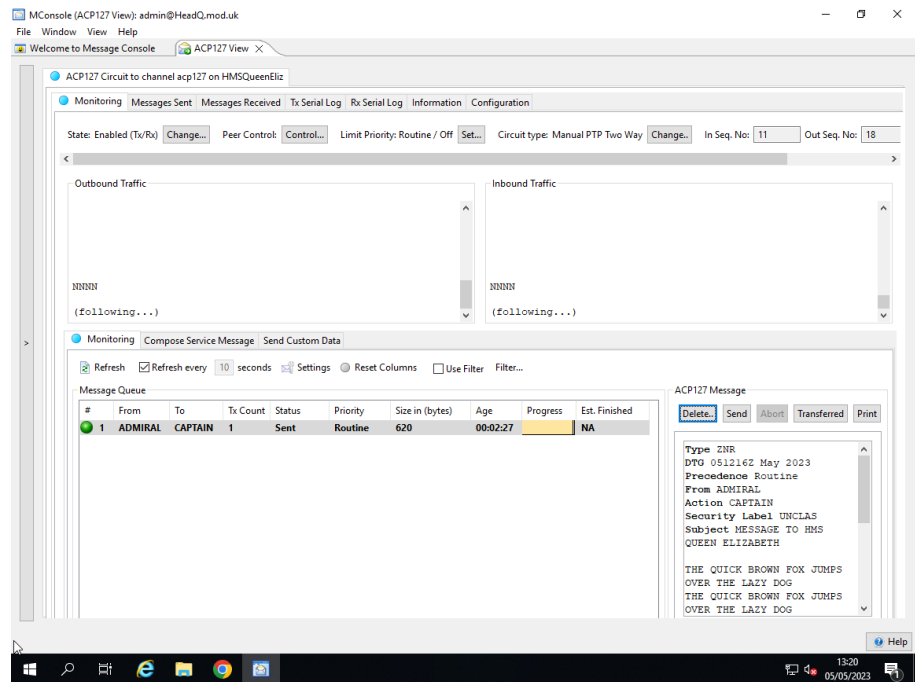
### Inbound FAB monitoring

If the circuit is associated with a FAB Monitoring circuit, then the Inbound FAB broadcast is shown here. This allows an operator to respond to the FAB broadcast appropriately.

### Inbound traffic [Anonymous manual receive]

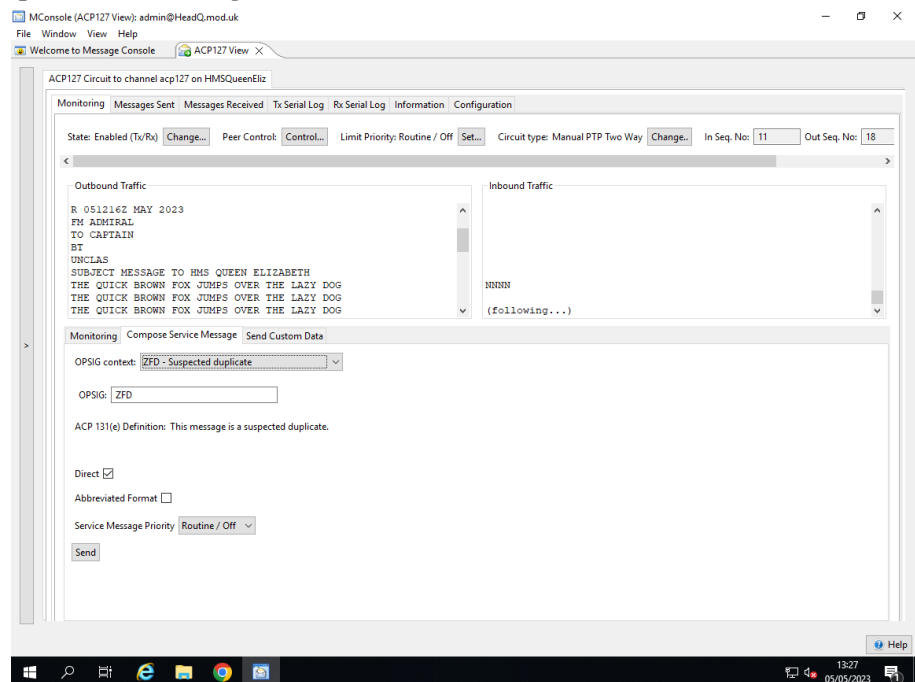
If the circuit is an anonymous manual receive circuit, then the **Inbound Traffic** display will show inbound bytes. Messages will be highlighted in green. The operator can then choose to accept the message via the **Accept** button. Or rejected via the **Reject** button. If the messages are accepted, then they will be processed in the normal way.

## Message Queue



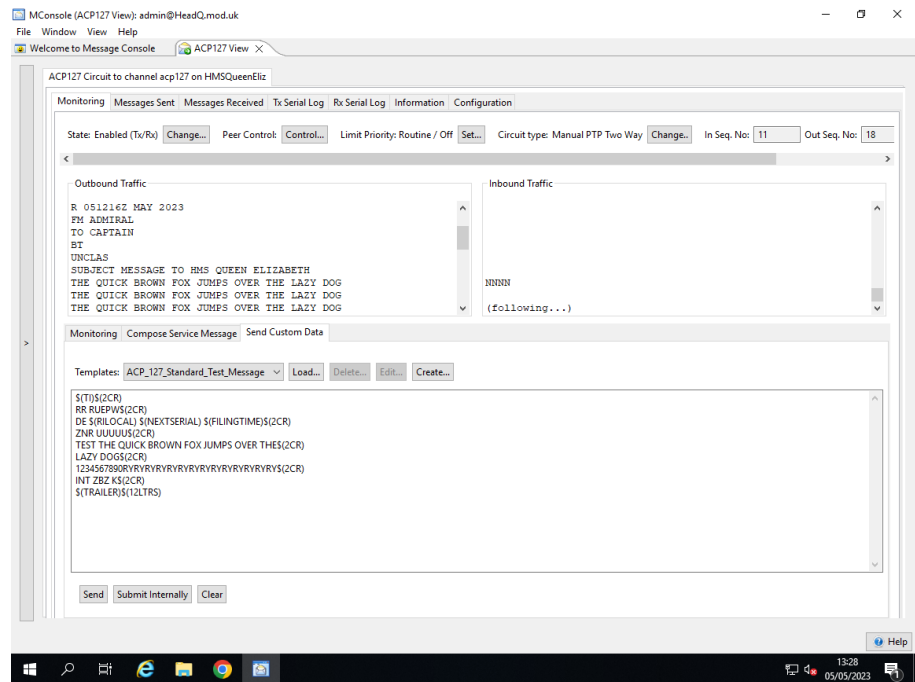
The message queue is used to list outbound messages for this circuit. It also allows the operator control over those messages.

## Compose service message



This subtab allows an operator to create a service message. It's optional, and is only present on some circuits.

## Send Custom Data



This sub tab allows an operator to send arbitrary data over the circuit.

A list of premade templates is shown. This is derived from templates that are shipped with M-Switch but also from custom templates.

---

**Note:** Custom templates are stored in a location defined within the switch configuration. To set this right click on the ACP127 server in the LHS. Then select **Modify Connection Information**. Select the **ACP127 Monitoring** tab. Update the **Template Dir.** with the new location

---

### Load...

Load a template not listed within the drop down menu.

### Delete...

Delete the selected custom template.

### Edit...

Edit the selected custom template using the template editor.

### Create...

Opens a template editor. The editor itself allows the use of macros. These macros allow an operator to insert dynamic values such as the next sequence number into the template.

## 19.2.2 Dynamic Queues

Certain messages require operator intervention. For example a broken message might be placed on a repair queue to be fixed by hand. Should a message be placed on one of these queues a tab will be created allowing the operator to intervene. The circuit needs to be configured so that messages that fall within these categories are processed via the following queues.

The dynamic queues are:

### Repair

If a message is received and it cannot be parsed correctly, and the **Garble action** is set to **Place on repair queue** it is placed onto a **repair queue**. Where an operator can fix the messages before submitting it into the messaging system.

Duplicate

If the **Inbound duplicate message action** is set to **Place on duplicate queue**, and a transmission contains a message that's been received already, then the message is placed on the **duplicate** queue.

Intercept

If the **Intercept Action** is set to **Place on intercept queue**, and a transmission contains a message that has been intercepted by this station, it's placed on the **Intercept** queue. This allows an operator to discard or forward the intercepted message.

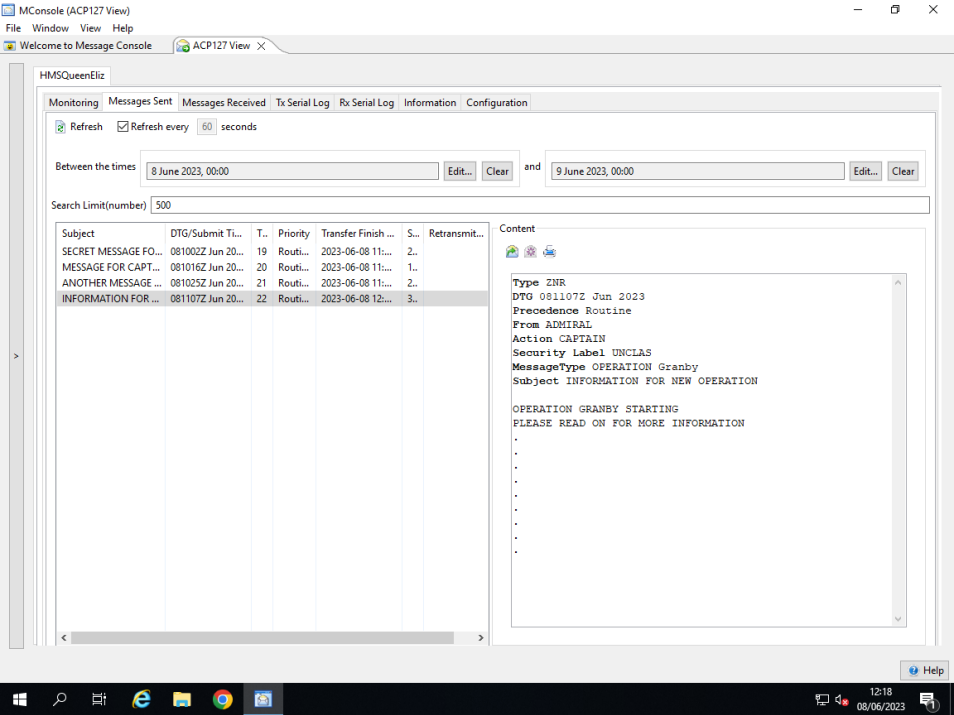
Codress

If the **Encrypted message action** is set to **Place on encrypted queue**, messages that have been encrypted will be placed on the **codress** queue. This allows the operator to export, process the message, and import it back into the message system.

19.2.3 Messages Sent and Received Tab

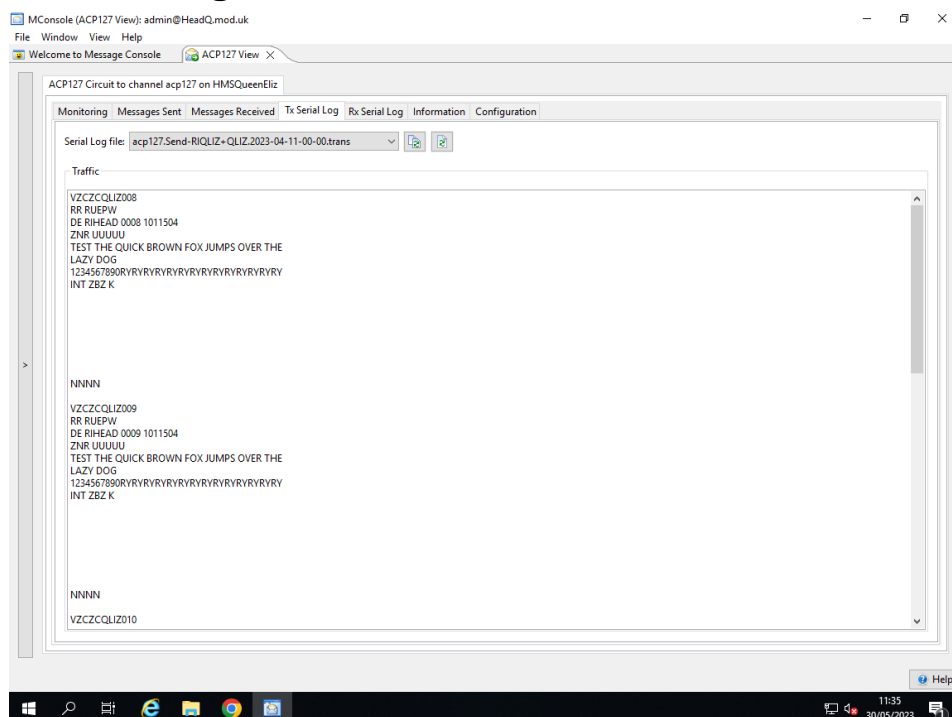
Broadcast and Point to Point sender circuits will have a **Messages Sent** tab. Similarly Broadcast and Point to Point receiver circuits will have a **Messages Received** tab. These views connect to the Audit Database. The Audit Database is configured in the **Options** view. These subtabs provide a way to display a summary of messages that are sent and received on the ACP127 circuit for the day. There is an option to modify the time limits to widen or narrow down the search. On clicking the **Refresh** button or in the next **Auto Refresh** interval, messages in the tab will be displayed as per the search parameters on the tab.

Figure 19.1. ACP127 Messages Sent Tab



## 19.2.4

### TX serial log



The transmission serial log tab is only shown for **Broadcast** and **Two-way** circuits.

The tab will use a connection to the qmgr to request a list of serial logs. The logs are filtered to only show the logs relevant to this circuit and direction.

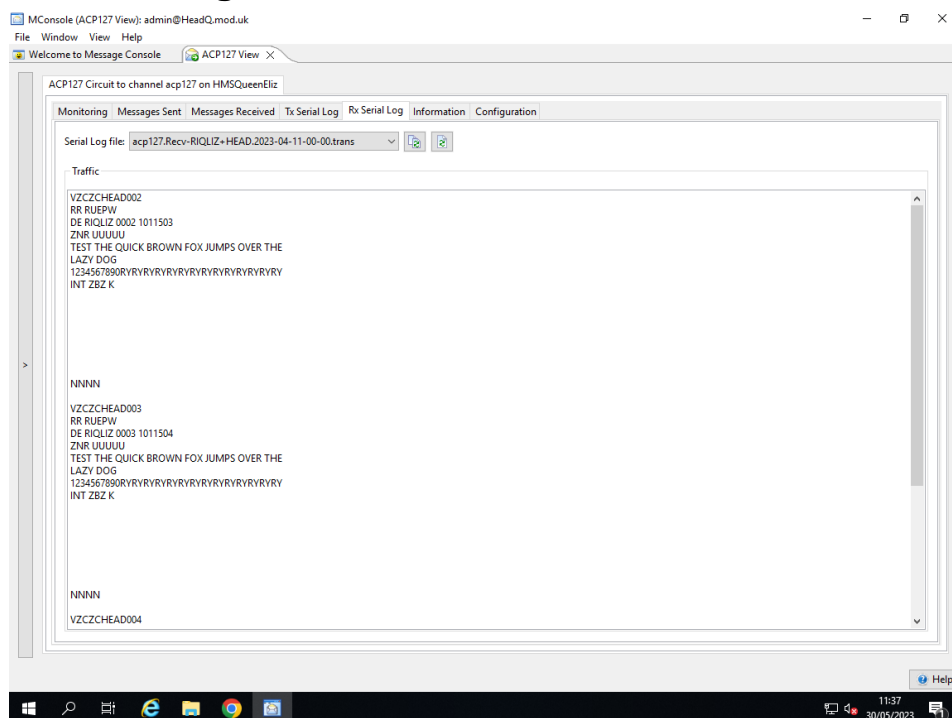
---

**Note:** Files that are loaded are static. An operator has to manually request the file be reloaded in order for it to update.

---

## 19.2.5

### RX serial log





Much like the **TX serial log tab** this tab requests a list of serial logs from the qmgr . The logs are filtered to only show incoming bytes.

---

**Note:** Files that are loaded are static. An operator has to manually request the file be reloaded in order for it to update.

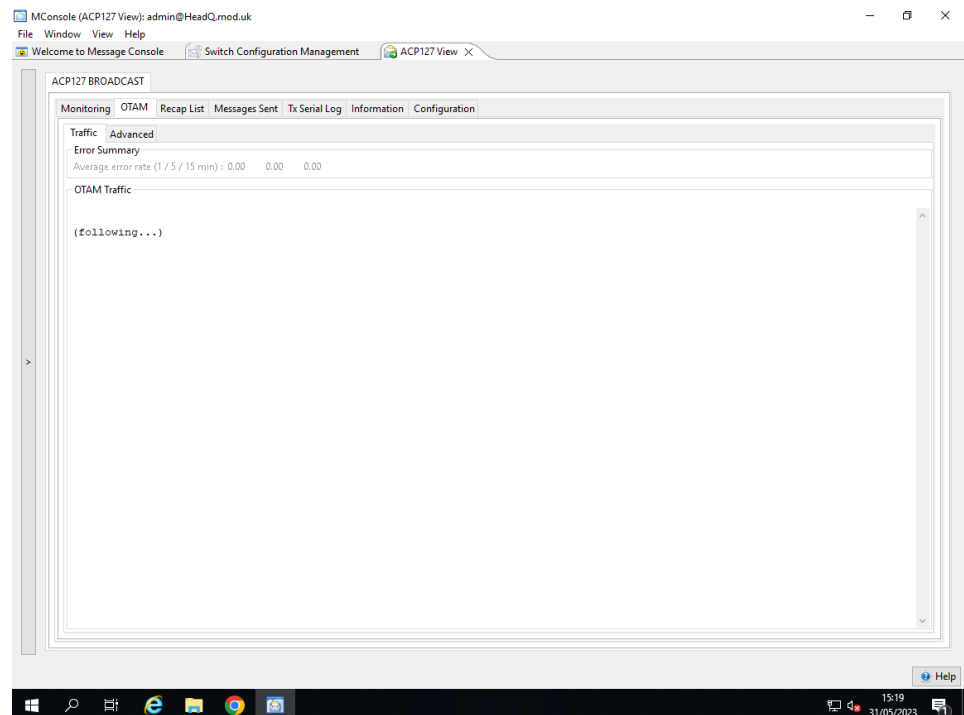
---

## 19.2.6 OTAM

The OTAM tab allows an operator to monitor a broadcast circuit. It works by analysing two connections. One is a monitoring connection to the ACP 127 Server, and the other, a direct serial connection to a modem.

The OTAM server then compares the write output from the ACP 127 Server with the read input from the modems to check for any errors.

### 19.2.6.1 Traffic



The traffic tab gives an average rolling error reading for 1/5/15 minute intervals, as well as the broadcast bytes.

The broadcast bytes are recorded as follows:

#### Green

The read and write streams match.

#### Blue

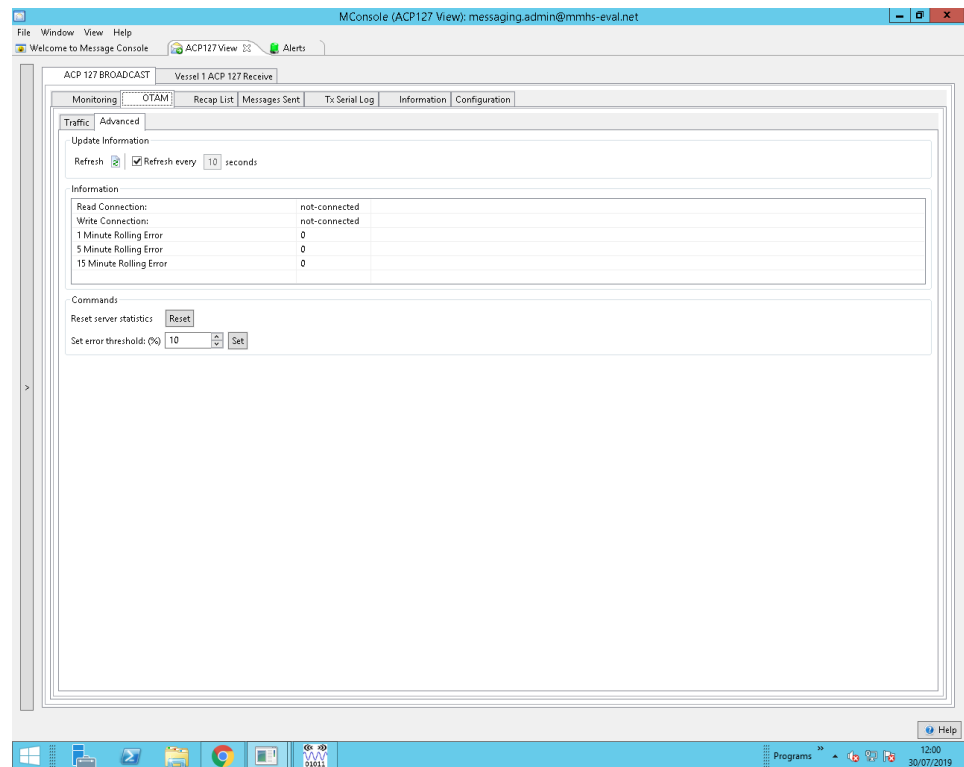
The read stream contains bytes that are not present in the write stream.

#### Red

The write stream contains bytes that are not present in the read stream.

## 19.2.6.2

## Advanced



The advanced tab contains extra information used to gather and report error statistics.

**Update Information**

The refresh button and time interval is used to automatically request new error stats from the server.

**Information**

The information tab shows if the OTAM server has a connection to the ACP127 server, and a direct connection to the modems. It also shows the 1 min/ 5 min/ 15 minute error summary.

**Commands**

The commands section allows the server stats to be reset, and to set the error threshold.

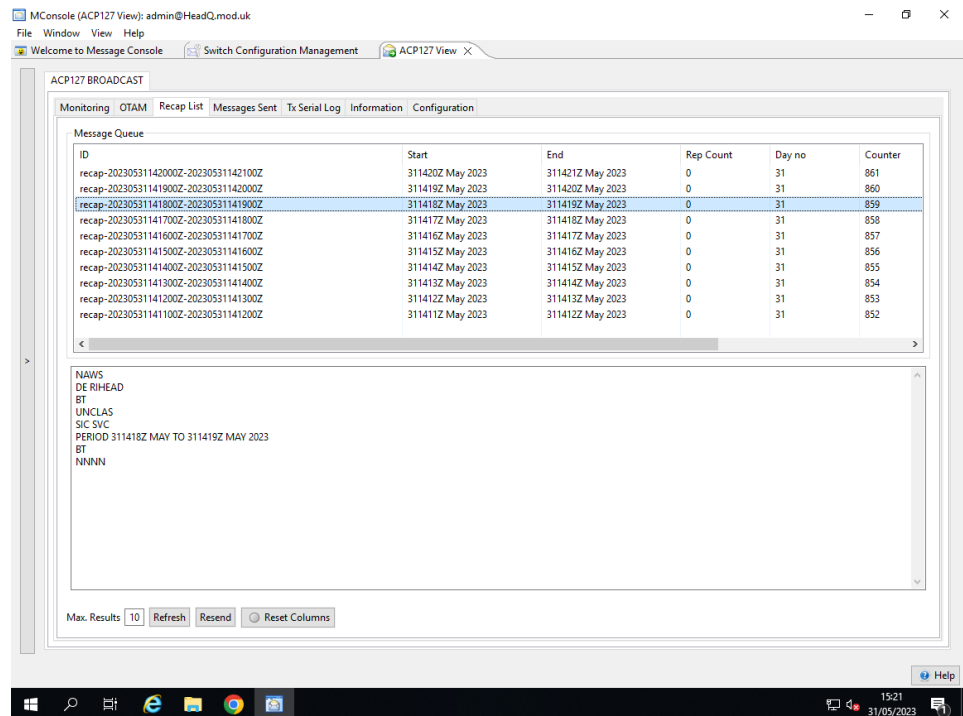
---

**Note:** If the server detects that the OTAM error threshold has been reached, an alert is issued from the server to MConsole.

---

## 19.2.7

### Recap Lists



Broadcast circuits may send out recap messages. A recap message identifies the messages which have been sent over a broadcast circuit within a defined time period, and allow recipients to detect when failure to receive broadcast messages has occurred.

The Recap List tab lists the last few recap messages from the server, and allow an operator to resend one or multiple messages.

If multiple recap messages are being retransmitted, then these will be retransmitted oldest first.

## 19.3

### ACP127 Routing

This is concerned with routing messages to ACP127 destinations, when the addresses used within the M-Switch system are Internet (SMTP) addresses. There are two distinct mechanisms which are used. One is used when all email addresses for a given email domain are to be routed to a single peer ACP127 server. Then the email domain can be used for the routing. However, if different email addresses with the same domain are to be sent to different peers, then a different mechanism is used. Finally, routing needs to be established for ACP127 users who have no address within the SMTP or X.400 networks.

#### 19.3.1

#### Routing using the domain

This is achieved using the standard internet routing using the Routing Tree. The Routing Tree entry for the domain is configured to send to the peer MTA (i.e. External MTA ACP127 Workstation).

#### 19.3.2

#### Routing using the whole address

In this model different addresses with the same domain are routed to different places i.e. different ACP127 stations. The domain concerned needs to be configured as local in the

routing tree node, i.e. that the MTA used is the local MTA. Then a search will be made for an entry which has an email address matching the whole address. This entry should be the same entry which contains the PLA and RI (Routing Indicator) for the user represented by the address. The entry should be configured with the channel set to the ACP127 channel which is to handle the message.

Routing will send messages for the recipient address to the named ACP127 channel on the local MTA. For such a message, the channel has access to the address to RI mapping, and then uses this RI to determine the peer system to which the message should be sent. This procedure is described in the next section [Section 19.3.3, "Routing using the Routing Indicator"](#).

If there are addresses in this domain which are delivered locally into M-Box, these user addresses are configured with the channel set to the LMTP channel. If an address is to be sent to a peer SMTP system, the 'mail host' is set to the hostname of that system.

### 19.3.3 Routing using the Routing Indicator

Each circuit to a peer ACP127 system has a set of routing indicators. There is the RI of the peer itself, and an optional set of RIs in the "Routed RI List". The latter can be a prefix to stand for all RIs which match that prefix. When the channel looks for the peer to which a given RI is to be routed, it searches for the longest match for the RI in these data. The RI itself is sought. Then, if not found, the rightmost character is removed, and another search made. The procedure is followed until a match is found. If no match is found, the message will be non-delivered. If two peers have the same matching RI or prefix, either may be used.

When the peer has been determined, the message for this RI is returned to the queue manager, queued for the appropriate peer (i.e. remote ACP127 station and circuit).

### 19.3.4 Routing for Unmapped ACP127 Addresses

If an ACP127 address has no mapping to an SMTP address, M-Switch will put the RI and/or PLA encoded within the local-part of an SMTP address. The domain used is configured using the channel specific variable 'gateway-domain'. If this domain is not configured, address mapping fails.

This is configured in the ACP127 Tab of the ACP127 channel.

This 'fallback' mapping has two uses:

- for ACP127 relay, when a message arrives by ACP127 and is sent out by ACP127.
- when messages arrive from an ACP127 user which does not have a mapping; the fallback address can be used for replies to that user.

For the latter, the domain chosen should be a domain which is routed in the wider SMTP network to the ACP127 gateway system.

The domain should be local in the Routing Tree. A 'wildcard' local user should be created with just '@' and the domain. This should have the channel set to the ACP127 channel to be used. Note that if the domain value used is one also used for non-ACP127 users, then each of these non-ACP127 users will need to be individually configured.

When an SMTP address in this fallback form is processed by the channel, the PLA and/or RI are found within the local-part. If there is no RI present, then the PLA is mapped to an

RI. Then this RI is used to determine the correct peer to use as described in the previous section.

---

**Note:** The attribute gateway-or-address has a similar function for generating fallback O/R Addresses. This is configured in the ACP127 Tab of the ACP127 channel: Gateway O/R Address.

---

---

## 19.4 ACP127 Circuits

ACP127 circuits can run in five distinct modes, depending on the requirements. These are:

### Normal (PtP)

In the normal mode, the link runs in a point-to-point mode. In this case the link is a straightforward connection between two stations, and messages usually flow bidirectionally between the two. However a PtP circuit can be unidirectional. Service messages may also be issued as required, in both directions. The most common mode for a PtP circuit is bidirectional.

### Broadcast Sender

In the broadcast sender mode, the circuit is designed to be a one to many type broadcast network. This might be shore to ship, or some similar scenario where the messages are broadcast to a collection of end points. No traffic is received on such a circuit, and normally such a circuit uses either modems driven via serial lines, or STANAG 5066 technology to achieve the broadcast. However any supported protocol can be used.

A broadcast sender may generate recap messages. This is a usually hourly message which is broadcast detailing all the messages sent during that period. Broadcast receivers can use this recap message to check that they have received all traffic that they should have.

In some cases, broadcast sender circuits will also send fill messages during periods without message traffic as a keep alive like mechanism to show the circuit is still working. Broadcast receivers will audit and log silence periods of more than the configured time highlighting that there may be a problem with the circuit.

### Broadcast Receiver

In the broadcast receiver mode, the circuit is designed to be the reception end of a link where traffic has been broadcast. Traffic received by this link may not be for this station, and so may be discarded. A broadcast receiver handles recap and fill messages sent from a broadcast sender. It will use the recap and fill messages to keep track of messages and connection state.

### Manual Anonymous Receive

A circuit which is receive only and which is not tied to one sending system. The receipt of messages is controlled by the operator monitoring the received data and changing the status of the circuits to acknowledge receipt, or request resending. This status is sent out over the Frequency Availability Broadcast (FAB) which is monitored by sending systems.

### Auto Anonymous Receive

A circuit which is receive only and which is not tied to one sending system. It uses STANAG 5066 COSS which uses ARQ to obtain reliable data transfer. Manual operation is not used. However, there is a FAB status indicating sent out for monitoring by sending systems.

For Normal (Ptp) and Broadcast sender circuits there are four options which further define the mode:

**Manual Mode (Fixed)**

Messages can only normally be sent with Operator intervention (exceptions are fill and recap messages if so configured). The circuit cannot be changed to Automatic.

For a manual sender (at least for ship-to-shore circuit) there is a need to indicate that a message has been received OK or not.

**Automatic Mode (Fixed)**

Messages are sent without requiring Operator intervention. Circuit cannot be changed to Manual.

**Manual Mode (Changeable)**

As for **Manual Mode (Fixed)**, messages can only be sent with Operator intervention. The circuit can be changed to Automatic in the ACP127 View.

**Automatic Mode (Changeable)**

Messages are sent without requiring Operator intervention. The circuit can be changed to Manual in the ACP127 View.

---

## 19.5 Using MConsole to configure ACP127

When creating a configuration to use ACP127 features of M-Switch, it is important that the MIXER option is selected. This is necessary even if only Internet (SMTP) is being used, or just X.400 is being used.

It is also sensible to configure the configuration as Military and tick the option which causes an ACP127 channel to be created. Doing this means the creation of ACP127 channel and ACP127 Shaper channels as described below can be omitted.

To set up an ACP127 gateway and allow routing to and from it, the following steps are required.

- Create and configure an ACP127 channel.
- Create and configure an ACP127 shaper channel.
- Create one or more external ACP127 MTA(s).
- Create peer connections (circuits) for each external MTA.
- Add a Routing Tree entry for each MTA.
- Configure address mapping to routing indicators. This can be done in two different ways using
  - Gateway Users
  - Mailbox Users

If you are using serial connections and STANAG 5066, you will also need to configure

- S5066 Servers
- ACP127 S5066 Access Points

### 19.5.1 Creating an ACP127 channel

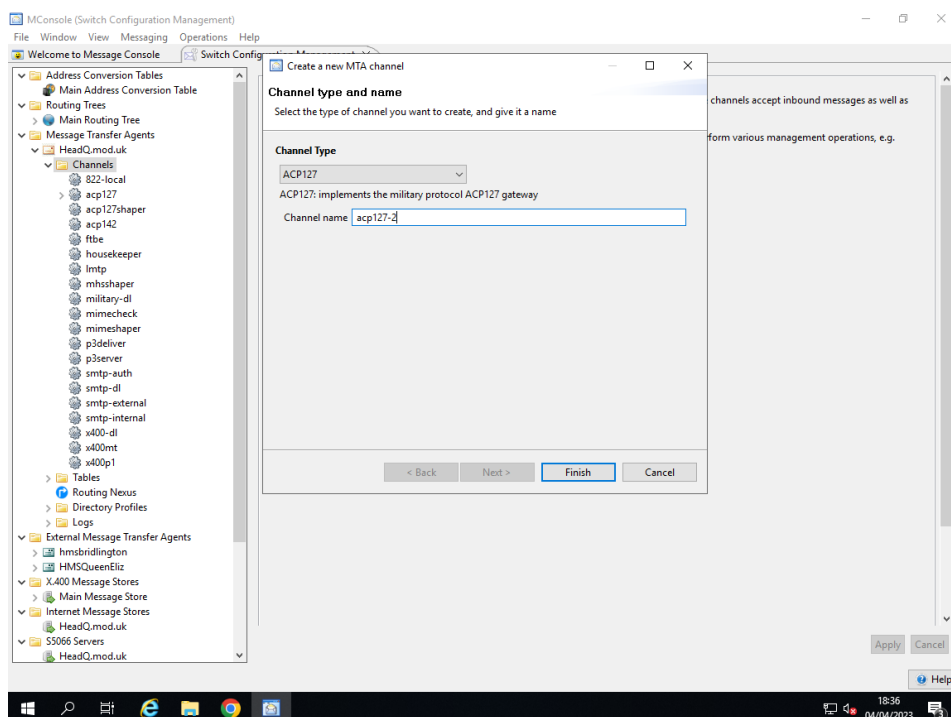
If you do not have an ACP127 channel, or wish to create a second channel to gateway messages, on the **Switch Configuration Management** tab, expand the **Message Transfer Agents** node, and the MTA below that. Then with the **Channels** node selected, either right click and choose **New Channel** or choose **Operations** → **New Channel**.

Then provide the following answers when prompted by the wizard:

- **Channel Type** select **ACP127**
- **Channel Name** name it `acp127` or similar

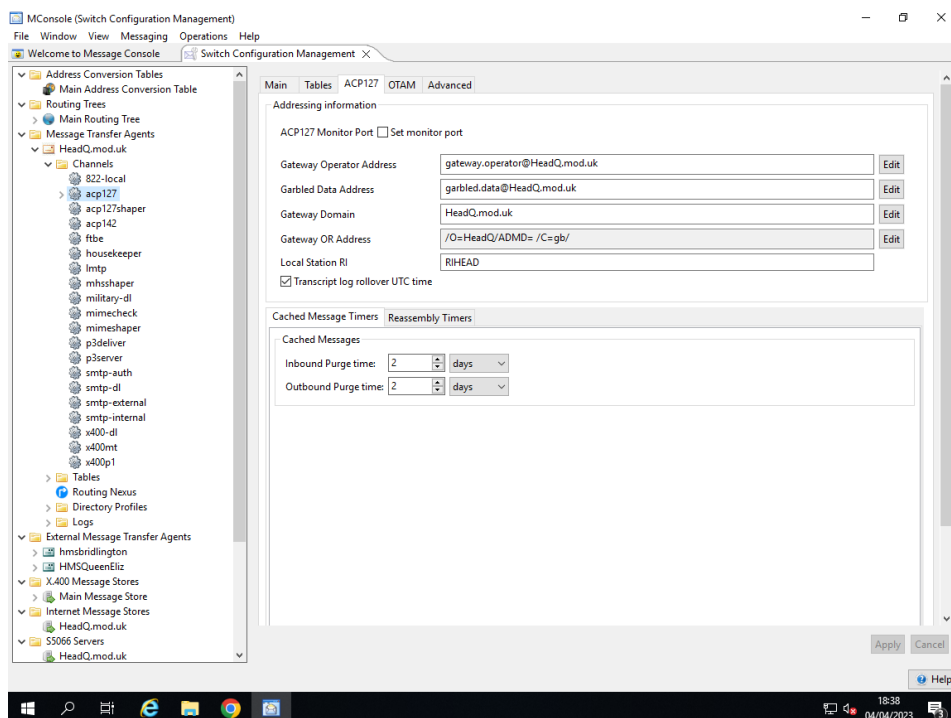
as shown in [Figure 19.2, “ACP127 channel creation”](#).

**Figure 19.2. ACP127 channel creation**



After creating the ACP127 channel, you should edit the channel to set the ACP127 specific parameters on the **ACP127** page [Figure 19.3, “ACP127 channel \(ACP127 page\)”](#) as required.

**Figure 19.3. ACP127 channel (ACP127 page)**



The following features are specific to the ACP127 channels

**ACP127 Monitor port**

This is the port on which the channel listens for incoming monitor connections such as from the MConsole ACP127 View. If this is not set then the default of 18099 is used.

**Gateway operator address**

This is the email address of an operator, which is used for cases where an operator needs to be informed, or used as an originator (such as the enveloper originator of messages being transferred through the gateway from ACP127).

**Garbled Data address**

This is the SMTP or X.400 email address of an operator, which is used for cases where an operator needs to be informed of garbled transmission.

**Gateway Domain**

This is the domain to be used for ACP127 addresses for which there is no mapping to SMTP as described in [Section 19.3.4, “Routing for Unmapped ACP127 Addresses”](#).

**Gateway O/R Address**

This is the O/R Address prefix to be used for ACP127 addresses for which there is no mapping to X.400 as described in [Section 19.3.4, “Routing for Unmapped ACP127 Addresses”](#).

**Local Station RI**

This is the Routing Indicator of this station.

**Transcript log rollover UTC time**

If this is selected the transcript logs of the ACP127 circuits will rollover each day at midnight UTC time. Otherwise it will be midnight local time.

**Inbound/Outbound purge**

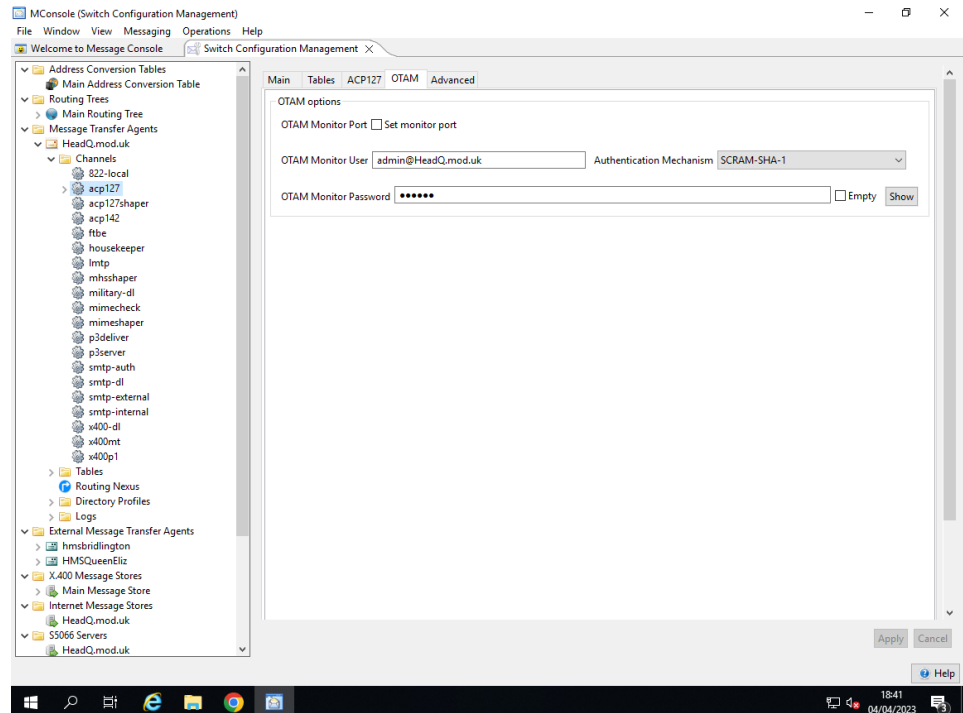
This is the time to keep messages in the channel database to allow for retransmissions on outbound, or duplicate detection on inbound transfers. After this time, the messages will be removed from the database.

**Section message reassembly timers**

These timers control how long the channel will wait for missing fragments of a message that has been split apart. Once the timer expires, the message will be sent with whatever has been received so far with missing parts indicated.

The OTAM process will connect to the ACP127 process to perform monitoring during broadcasts. To do this the OTAM process will connect to the ACP127 process as a monitor. [Figure 19.4, “ACP127 channel \(OTAM page\)”](#)



**Figure 19.4. ACP127 channel (OTAM page)**

The **OTAM** tab allows an Administrator to set:

#### **OTAM Monitor Port**

The port on which the ACP127 process is listening for new monitors.

#### **OTAM Monitor User**

The account used to connect to the ACP127 process.

#### **Authentication Mechanism**

The SASL authentication mechanism that is used when the OTAM server connects to the ACP127 process.

#### **OTAM Monitor Password**

The password for the account used to connect to the ACP127 process.

## **19.5.2 Creating an ACP127 shaper channel**

After creating the main channel, an ACP127 shaper channel is required to allow conversion of ACP127 content to other formats. This is created in a similar manner to the ACP127 main channel. The only important parameter here is the **XML Configuration file**, which contains the rules for conversion. This is normally **acp127-shaper.xml** unless there is a requirement for non-standard conversion.

**acp127-shaper.xml** is a file which is installed into (*SHAREDIR*). It is read and used to configure the acp127 shaper channel. However, if you wish to edit this file, you should copy it into (*ETCDIR*). The acp127 shaper channel will use (*ETCDIR*)/*acp127-shaper.xml* in preference to the original in (*SHAREDIR*).

## **19.5.3 Creating an external ACP127 MTA Connection**

There are five steps to this procedure, or fewer if re-using an existing connection.

- Create an external MTA associated with the remote connection.
- Create or configure a peer connection for the ACP127 Channel with the remote connection. This represents an ACP27 Circuit to the External MTA (i.e. ACP127 Station).

- Create a domain in the Routing Tree entry to route addresses into the ACP127 gateway.

---

**Note:** All addresses to this domain will be routed into ACP127 gateway. It is not currently possible to route different addresses from the same domain differently (e.g. some to the ACP127 Gateway, and some to be routed as Internet addresses, e.g. delivered using LMTP).

---

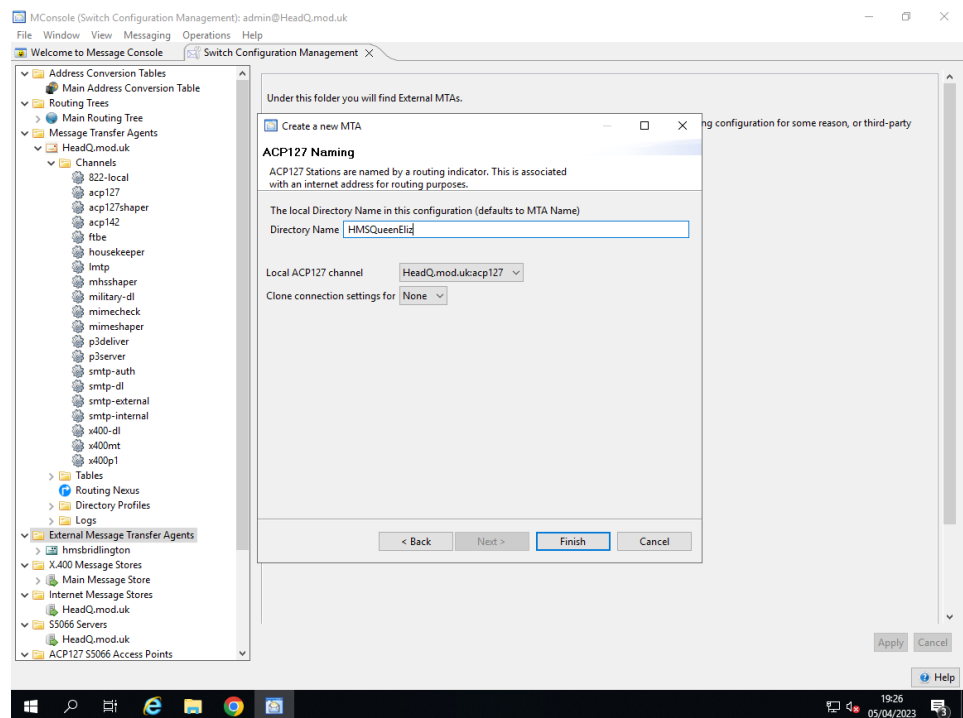
- Assign users in the domain with routing indicators and plain language addresses.
- Assign gateway users in the ACP127 domain with routing indicators and plain language addresses.

### 19.5.3.1 Create a remote external ACP127 MTA

To create a remote MTA that represents the ACP127 network, first create an external MTA. This is done with MConsole on the **Switch Configuration Management** tab. Expand the **Message Transfer Agents** node, and then right click on **External Message Transfer Agents** and choose **New External MTA** from the menu. Alternatively use **Operations** → **New External MTA**.

When the wizard popup dialog appears, select **External ACP127 Station** and click **Next**. In the subsequent dialog, name the external connection with a unique directory name, usually a name relevant to the remote connection - e.g. *Vessel 1 ACP 127 Receive*. Click **Finish** to complete the creation as shown in [Figure 19.5, “ACP127 create external MTA”](#).

**Figure 19.5. ACP127 create external MTA**



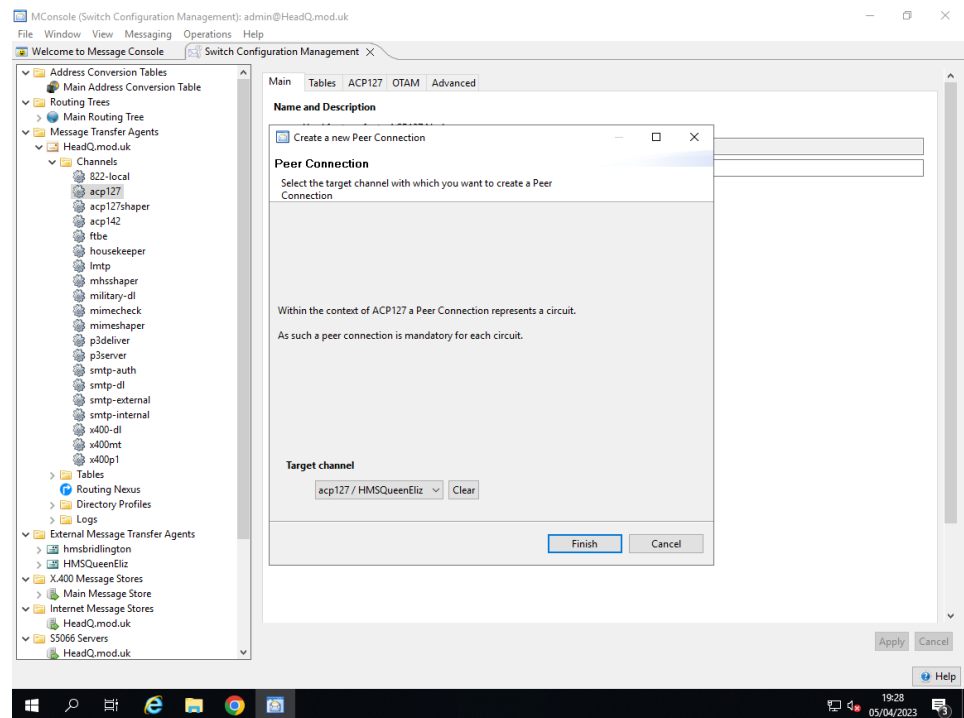
The new external MTA should now be present in the list of MTAs below the **External Message Transfer Agents** node. If the acp127 channel is not named **acp127** then you will need to edit the transfer weightings by right clicking the new node and select **Modify Transfer Channel Weight** to update the channel.

### 19.5.3.2 Set up a peer connection

A peer connection encompasses the details needed to transfer messages between the two entities.

If you do need to create a new peer connection, navigate to the ACP127 channel, which appears below **Message Transfer Agents** and then **Channels**. Right click on the acp127 channel, and select **Add Peer Connection** from the menu as shown in [Figure 19.6, “ACP127 creation of a Peer Connection to the external MTA”](#).

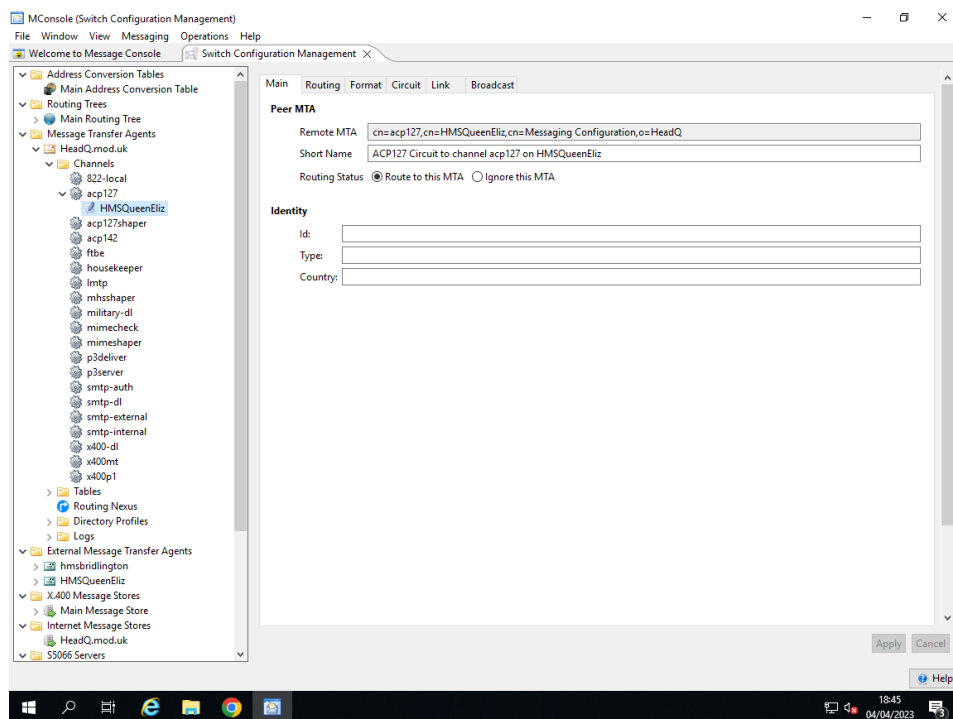
**Figure 19.6. ACP127 creation of a Peer Connection to the external MTA**



After the Peer Connection is created, the entry should be edited to set all the parameters required for the connection to succeed. In particular the variant of the protocol to use (currently acp126, acp127, acp128, doi-103, doi-103s, janap128, bsg, Italian-mrl, Italian-s2s), the TCP/IP details, the local and remote routing indicators, and the station identifier. An example is shown in [Figure 19.7, “ACP127 editing the Main Tab of a peer connection”](#) for the common parameters, and [Figure 19.20, “ACP127 editing the Link Tab of a peer connection \(TCP/IP\)”](#) for TCP/IP case; [Figure 19.21, “ACP127 editing the Link Tab of a peer connection \(5066\)”](#) for the STANAG 5066 case, and [Figure 19.22, “ACP127 editing the Link Tab of a peer connection”](#) for the serial line case.

### 19.5.3.2.1 ACP127 Peer Connection: Main Tab

Figure 19.7. ACP127 editing the Main Tab of a peer connection



The parameters are as follows:

#### Remote MTA

DN of channel of External MTA.

#### Short Name

Short name of External MTA.

#### Routing Status

If **Routing Status** is set to **Route to this MTA** then this peer MTA will be taken into consideration for routing.

If **Routing Status** is set to **Ignore this MTA** then this peer MTA will be ignored for routing.

#### ID

A short, unique identifier for this circuit.

#### Type

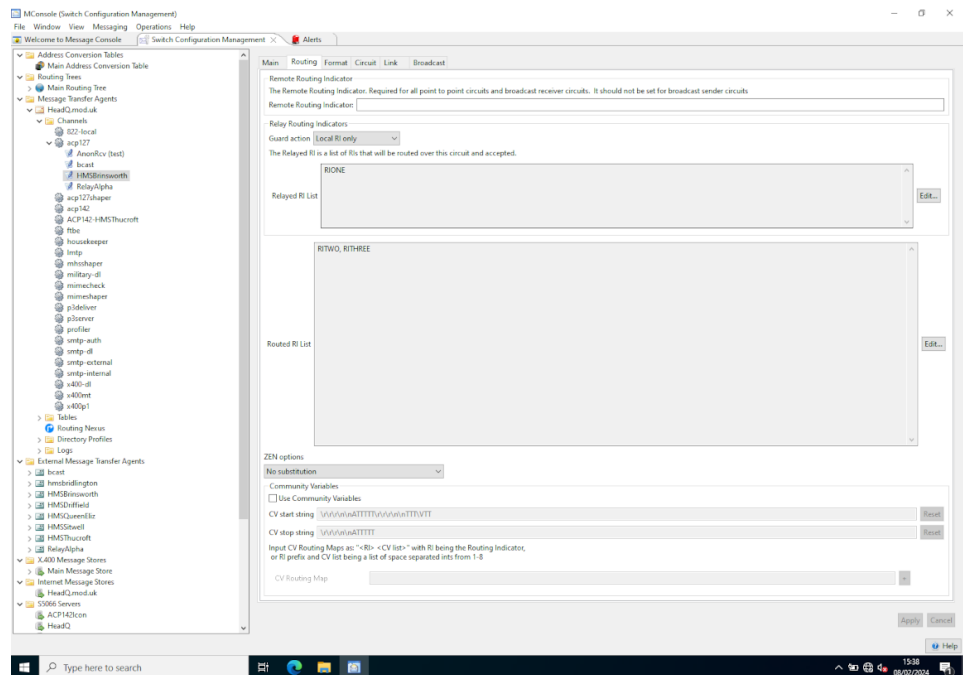
Type of circuit, e.g. MRL, RADIO etc.

#### Country

Country associated with this circuit, NATO, ITALY etc.

### 19.5.3.2.2 ACP127 Peer Connection: Routing Tab

Figure 19.8. ACP127 editing the Routing Tab of a peer connection



The parameters are as follows:

#### Remote Routing Indicator [Point to Point and Broadcast Receive Only]

RI to which Service Messages are to be sent

#### Guard action

This item controls how messages whose destination is for another station are handled by this circuit. It can have one of the following values:

##### Accept all

In this case all messages are accepted and either gatewayed or relayed onwards to other stations. This negates any intercept actions described below.

##### Local RI Only

In this case only if the message is sent to the local RI is the message accepted.

##### Local and Relayed list

The circuit is said to act as a 'guard' for the stations on the Relayed list. In this case only messages marked with the local RI, or those in the relayed list are accepted.

#### Relayed RI List

RIs which will be accepted and routed onwards

#### Routed RI List

RIs which will be routed on this circuit. A partial match is sufficient. If more than one circuit matches, the longest match will be chosen.

#### ZEN options

On outbound messages, recipients can be marked with ZEN indicating that the message has reached them by another route. This option controls which recipients are so marked. This specifies how to handle those recipients that are not directly for us. It can be one of the following:

##### No substitution

No ZEN substitution is done.

##### Use ZEN for recipients not in envelope

Add a ZEN component to all recipients that are not in the envelope component of the message.

**Use ZEN for recipients not routed by acp127**

Add a ZEN component to all recipients that would be routed by protocols other than ACP127 such as SMTP or X.400.

**Use ZEN for recipients not routed down this circuit**

Add a ZEN component for any recipient that is not specifically routed down this circuit.

**Community Variables****Use Community Variables**

If selected this feature is enabled

**CV start string**

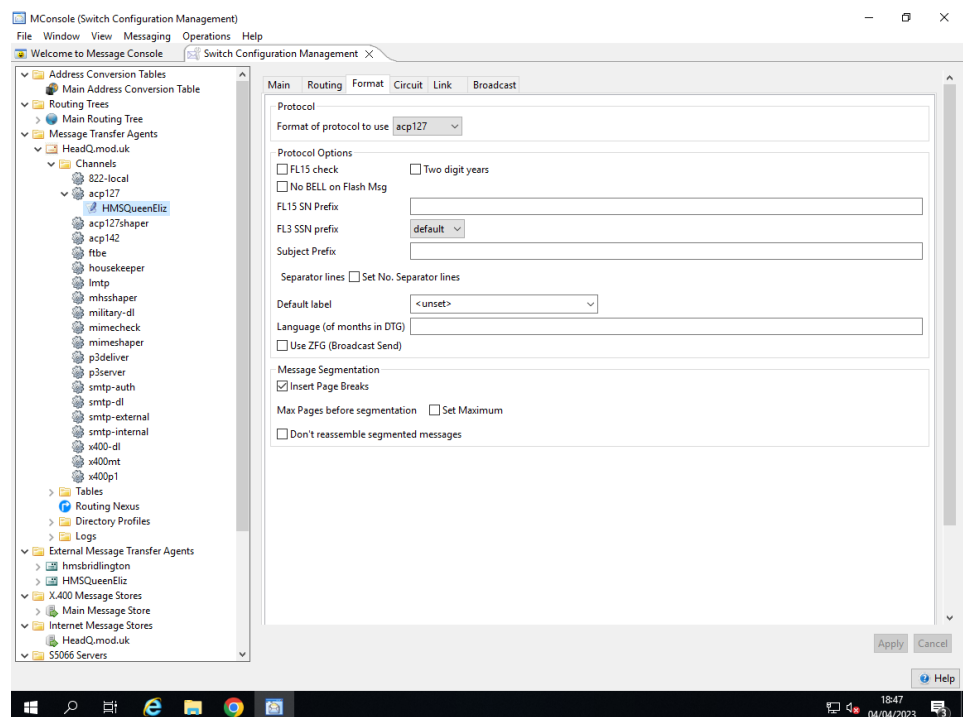
The string used to inform the cryptographic hardware which key to use. The characters '\V' are used to indicate where in the string the digit referring to the key is to be placed.

**CV stop string**

The string used to inform the cryptographic hardware that transmission is ceasing.

**CV Routing Map**

This gives the key numbers which a station, or group of stations can use. Each value starts with a Routing Indicator, or the prefix of a set of Routing Indicators. This is followed by a space-separated list of the numbers of the keys supported. Key 1 is assumed, so need not be included, although it can. An entry is found for a given Routing Indicator by finding the longest match in the set of values.

**19.5.3.23 ACP127 Peer Connection: Format Tab****Figure 19.9. ACP127 editing the Format Tab of a peer connection**

The parameters are as follows:

**Protocol**

The format of the protocol to use. This is currently one of:

- acp126
- acp127
- acp128

- doi-103
- doi-103s
- janap128
- bsg
- italian-mrl
- italian-s2s

This governs the format that is sent out and the parser used for incoming messages.

---

**Note:** Some formats such as the italian-mrl / italian-s2s require callsigns to be set and potentially other options.

---

### **FL15 check**

If selected this inserts a sequence number check into the protocol at format line 15 (FL15). This is of the form #NNNN

### **Two digit years**

A Date Time Group (DTG) can have two digit years or the full four digits. Checking the two digit box causes two digit years to be sent in DTGs

### **No BELL on Flash Msg**

If selected BELL characters are not generated in Flash messages.

### **FL15 SN Prefix**

This is the character inserted in front of the FL15 serial number check. It defaults to #.

### **FL3 SN Prefix**

This controls what character if any is inserted in front of the FL3 serial number. If set to **default** then it does whatever the protocol normally does. For **omit** the character is never inserted, if **insert** then the **FL 15 SN Prefix** value is inserted.

### **Subject Prefix**

When generating body content of a message, the subject part of the FL12 has this prefix. It default to SUBJECT but may be set to other values such as MSGID/.

### **Separator Lines**

The number of blanks lines to insert following the end of message sequence. This is used to separate two messages more clearly.

### **Default label**

This is the default security label to be use don this circuit.

### **Language (of months in DTG)**

What language to specify month names in the DTG component. The default is English, but Italian is also an option currently.

### **Default label**

This is the default security label to be used on this circuit.

### **Use ZFG**

If this option is set, then ZFG OPSIG is used in FL1 to signal a rerun.

### **Page Breaks**

If selected this causes page break to be inserted in outgoing messages. It also affects the segmentation option (Maximum lines) below.

### **Maximum lines**

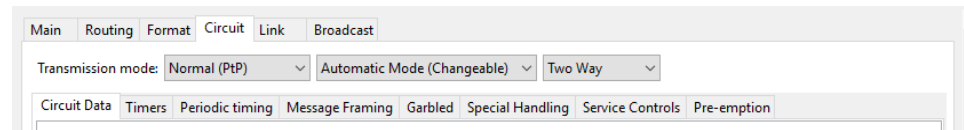
This is the number of lines a message may contain before being subject to segmentation into separate smaller messages. The gateway will normally try and reassemble such messages it receives. A value of 0 stops the message being segmented. If Page Breaks are enabled this instead reflects the number of pages before segmentation and the label changes.

### **Don't reassemble segmented messages**

If selected this causes segmented messages to not be reassembled.

### 19.5.3.2.4 ACP127 Peer Connection: Circuit Tab

**Figure 19.10. ACP127 editing the Circuit Tab of a peer connection**



The parameters are as follows:

#### Transmission Mode

Which style of circuit this represents. One of

- **Normal (PtP)** represents a standard point to point circuit.
- **Broadcast Sender** represents a circuit that broadcasts out to possibly several stations.
- **Broadcast Receiver** represents a circuits that receives incoming broadcast messages but does not transmit.

#### Send Mode

One of the following settings:

##### Manual (Fixed)

The circuit runs with manual intervention. Each message must be explicitly sent and marks as received.

##### Automatic (Fixed)

The circuit runs automatically sending messages as they arrive in precedence order.

##### Manual (Changeable)

The circuit runs as a manual circuit, but the operator may toggle it into automatic mode and back again as required.

##### Automatic (Changeable)

The circuit runs automatically sending messages as they arrive in precedence order by default. However it may be toggled into and out of manual mode as an operator requires.

#### Direction

One of the following settings:

##### Two way

The circuit accepts and receives messages (the default).

##### Transmit Only

The circuit will only send messages. Any incoming traffic will be treated as garbled.

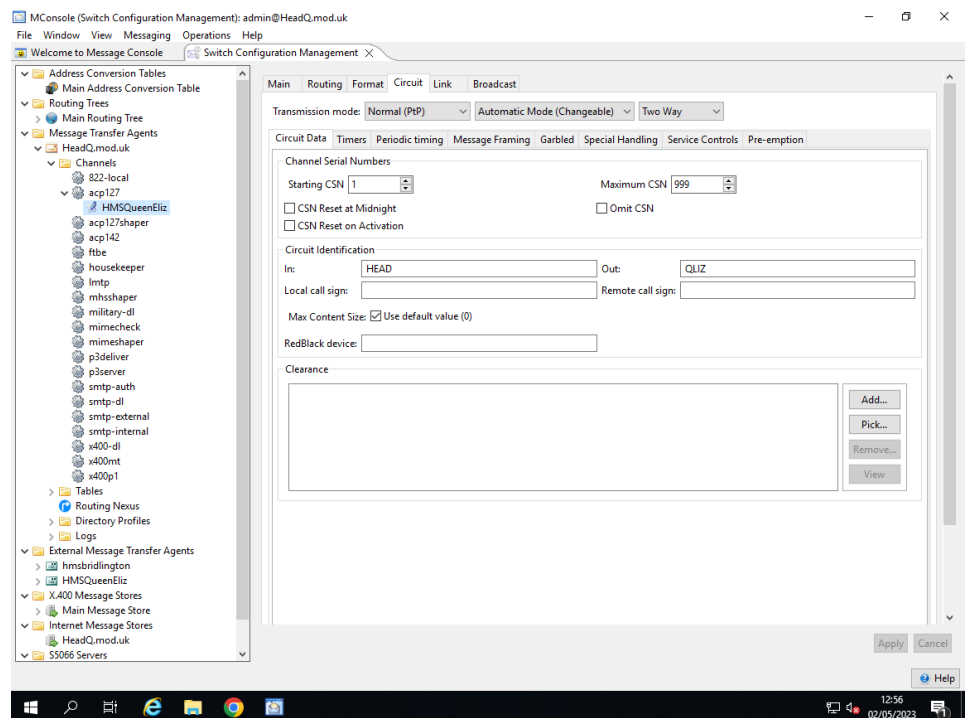
##### Receive Only

The circuit only accepts messages - no messages will be initiated.



## 195.32.4.1 ACP127 Peer Connection: Circuit Tab, Circuit Data SubTab

**Figure 19.11. ACP127 editing the Circuit Tab, Circuit Data**



The Circuit data subtab contains the following settings

### Starting CSN

This is the first number assigned to a circuit serial number, and what this wraps around to after the maximum circuit serial number.

### Maximum CSN

This is the highest number assigned to a circuit serial number, when reaching this value this wraps around to the starting CSN.

### CSN Reset at Midnight

The CSN will be reset to their minimum each midnight period.

### Omit CSN

Do not include the CSN in the message. This is normally done for ACP 126 based formats.

### CSN Reset on Activation

Reset the CSN to the minimum on circuit activation.

### ACP127 Circuit Identification

This is the 3 or 4 letter code of the station that is placed in the first line of the message. The circuit identifiers may be different for incoming and outgoing messages.

---

**Note:** The first and last characters must not be a digit. When broadcasting four characters must be used.

---

### Call Signs

This allows the setting of the call sign for the local and remote site - used mainly in ACP 126 based messaging.

### Max Content size

This sets the maximum allowable content size allowed to be sent over this circuit.

### RedBlack device

This allows linkage to the Red/Black server.

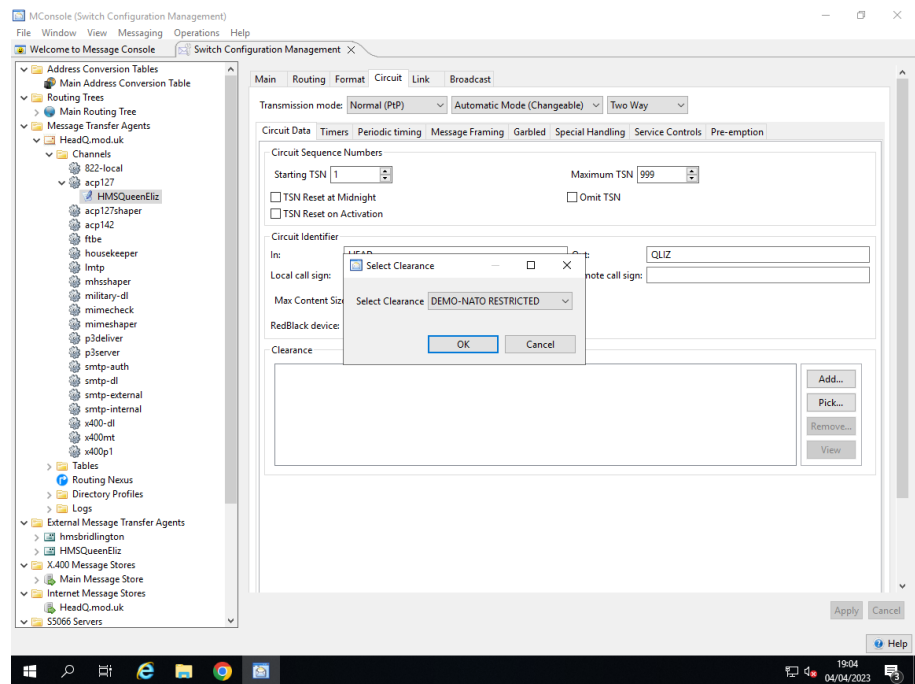
## Clearance

This allows a security clearance to be added associated with the circuit.

This is a multi-valued attribute to configure clearances for the circuit. Clearances can be applied to a circuit for rule based access control using security policy. A circuit can have one clearance per security policy configured for the switch.

The UI for configuring clearances is present on the **Circuit data** tab under the **Circuit** tab. The clearance catalog can be picked from a file named `clearance_catalog.xml` in the *(ETCDIR)*. M-Switch will use this version (if present) to override the value of the installed copy in *(SHAREDIR)/switch*. This avoids overwriting of configuration files on updates or upgrades. The UI also allows you to add clearance from an XML file by browsing the file system.

**Figure 19.12. ACP127 Circuit Clearance Configuration**



## Monitor Fab [Transmit only]

This option allows the operator to associate this circuit with a FAB monitoring circuit. If set, the ACP127 view will show content from the given FAB monitoring circuit.

## FAB Circuit [Receive only]

If selected this circuit will be able to issue FAB status updates.

## Allow free text SVCLINE [Receive only]

If selected the operator will be able to enter their own text as a SVCLINE FAB status update.

## FAB Guard [Receive only]

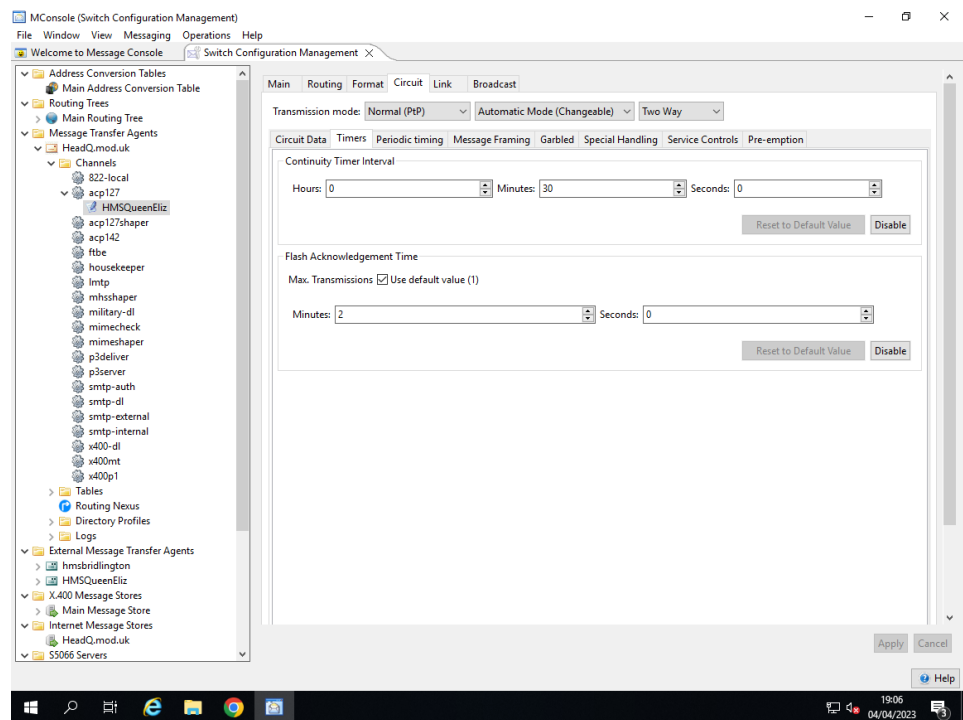
The instance of the guarded fab circuit to use. This has to be configured under [Section 19.5.5, “Setting up a FAB Server Configuration”](#) of the UI.

## Fab Frequency String

The prefix string to send for FAB communications.

## 195.32.42 ACP127 Peer Connection: Circuit Tab, Timers SubTab

**Figure 19.13. ACP127 editing the Circuit Tab, Timers**

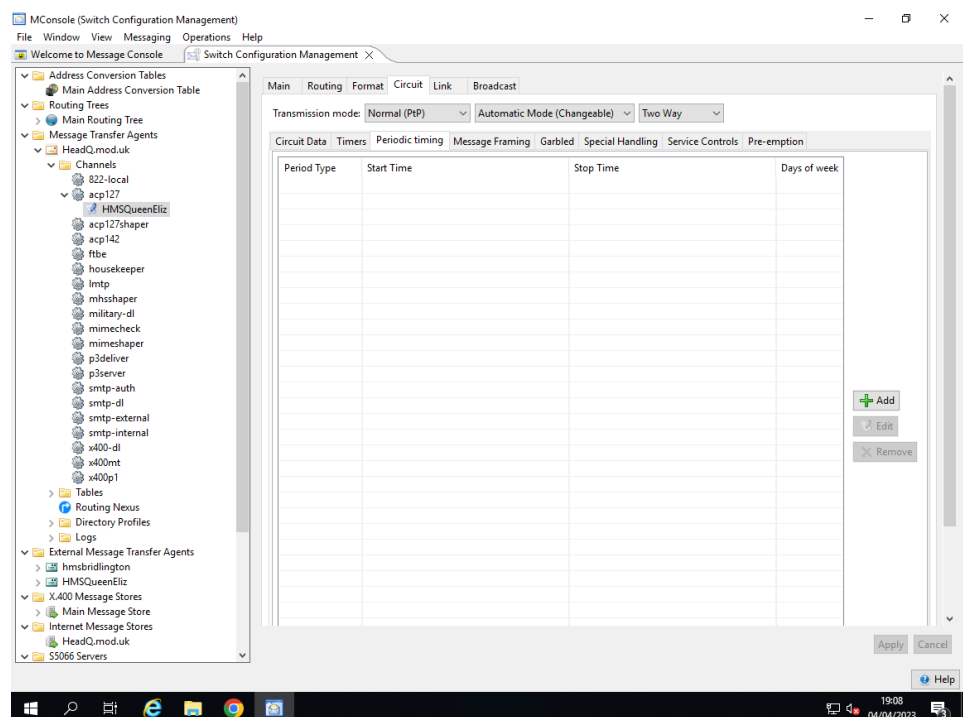


The timers section contains the following:

- **Continuity Timer Interval** Interval between continuity messages (which are sent when there is no other traffic).
- **Flash Ack Time** Time in which an acknowledgement to Flash message is expected.

## 195.32.43 ACP127 Peer Connection: Circuit Tab, Periodic timing subtab

**Figure 19.14. ACP127 editing the Circuit Tab, Periodic timing**

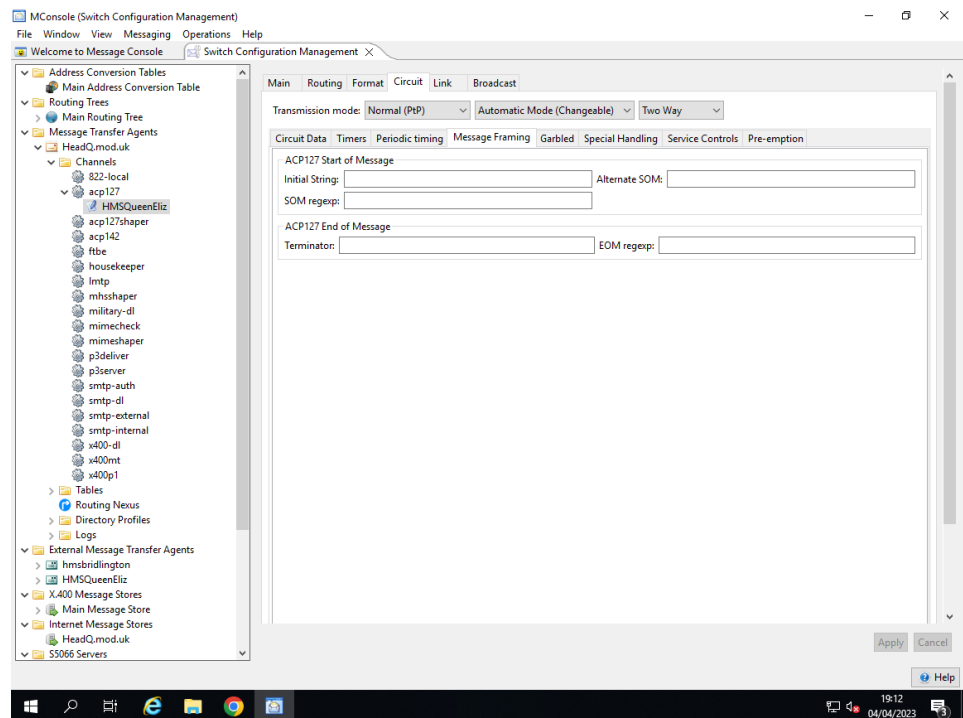


A channel can operate in a mode when it sends message only during predefined periods of time. It is allowed to have overlapping periods. There are two different kinds of period which can be configured:

- **Intervals** An interval has a specific start date and time, and can be configured with a specific end date and time or a duration in hours and minutes.
- **Periodic** This type has a start time and either an end time or a duration. It also has the days of the week to which it applies, which can be all seven days. To configure a period which extends over midnight, the duration option should be used.

#### 1953244 ACP127 Peer Connection: Circuit Tab, Message Framing SubTab

**Figure 19.15. ACP127 editing the Circuit Tab, Message Framing**



This allows configuration of message framing.

- **Initial String** Start of Message value, defaults to ZCZC (or NAWS for broadcast).
- **Alternate SOM** A value to be accepted as an alternative Start of Message sequence (SOM).
- **SOM regexp** A regular expression to match complex or non-standard start of message sequences such as used in Acp126.
- **Terminator** This is the sequence of characters used to terminate a message on the wire. It defaults to NNNN where the characters will be replaced by the end of line marker.
- **EOM regexp** This is a regular expression that will match the terminating string for non-standard cases.

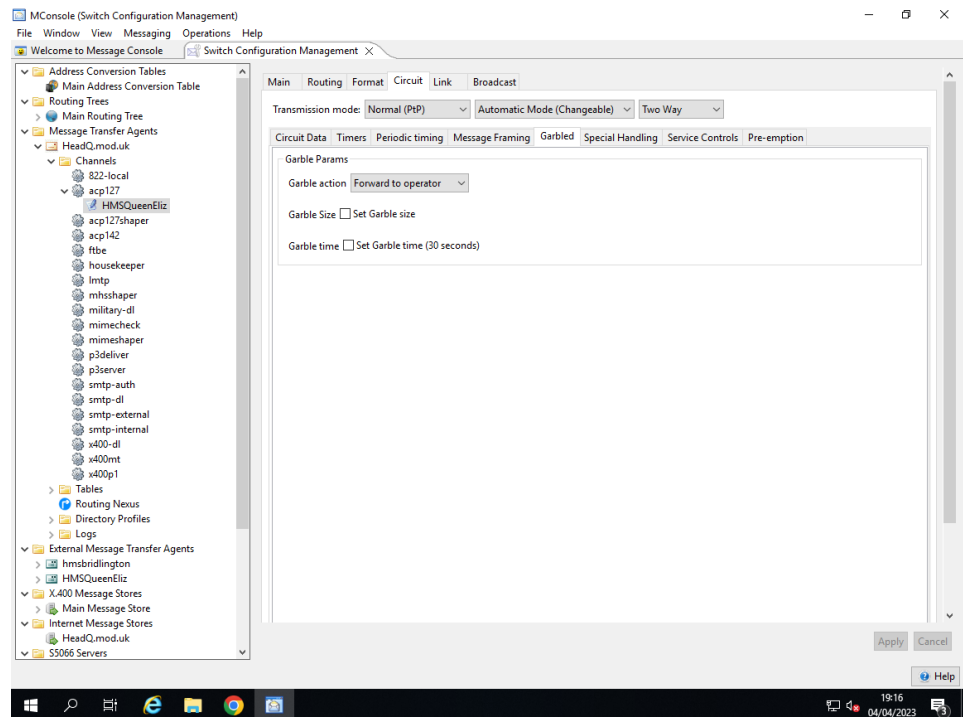
Special characters for the Initial String/Alternative SOM/Terminator are:

Table 19.1. ACP127 special characters in SOM/EOM

Character	Replacement
<	2CR LF (or \r*\n in a regex)
^	LF (or \r*\n in a regex)
_	Space
?	removed unless in a regex

## 195.3245 ACP127 Peer Connection: Circuit Tab, Garbled SubTab

Figure 19.16. ACP127 editing the Circuit Tab, Garbled



This allows configuration of data that is detected to be garbled transmission.

**Garble action**

This controls what happens when a garbled sequence is detected. The options are:

- **unset-** the default action of forward is taken.
- **Forward to Operator-** The garbled transmission is written to a dead letter file, and also forwarded to the configured garble address.
- **Discard** The garbled content is discarded.
- **Place on Repair Queue-** The garbled content is placed on a queue to be further processed and possibly repaired by the appropriate mconsole view.

See [M-Switch Operator's Guide](#).

**Garble size**

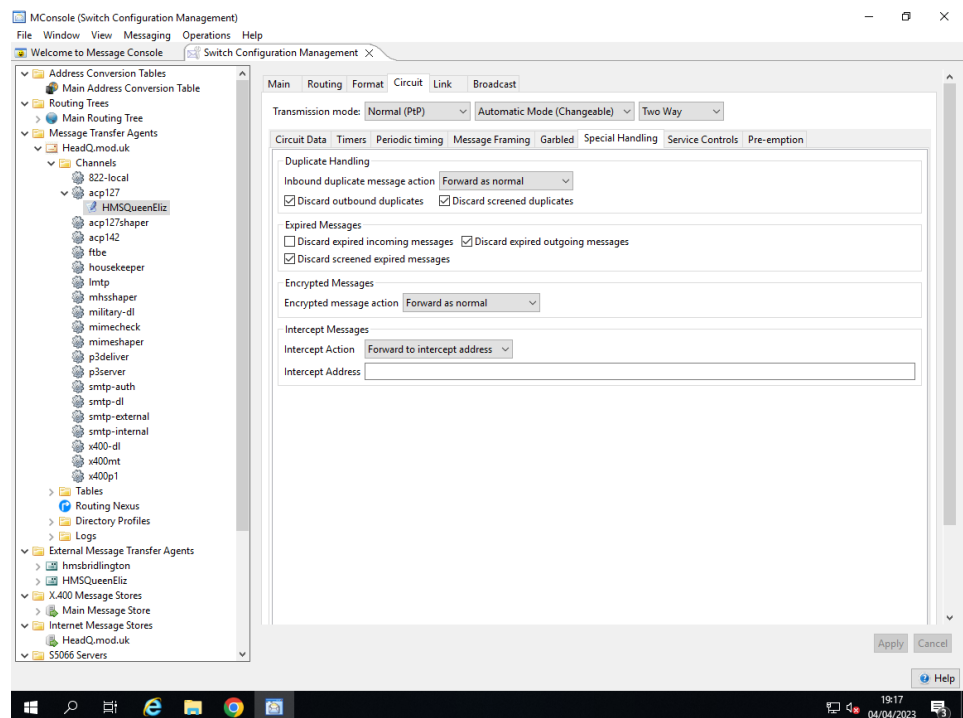
This is a minimum size to treat as garble. It defaults to 10 bytes, so if a garbeled message less than this size will be ignored.

**Garble time**

This is the timeout to wait for a complete message. If this timeout is exceeded then if there is no complete message it will be treated as garbled text.

## 195.32.46 ACP127 Peer Connection: Circuit Tab, Special Handling SubTab

Figure 19.17. ACP127 editing the Circuit Tab, Special Handling



This allows configuration of various special options for message handling.

**Inbound duplicate message action**

This controls what happens when a duplicate message is found. The options are:

- **unset**- the default action (Forward) is taken.
- **Forward as Normal**- treat the message as nothing special and submit as normal.
- **Discard**- The message is deleted and not passed on.
- **Place on Duplicate Queue**- The message is placed in a database queue for further processing by the mconsole interface.

See [M-Switch Operator's Guide](#).

**Discard outbound duplicates**

This is a switch to have outbound messages that are duplicates of previously sent messages discarded.

**Discard screened duplicate**

This is a switch to discard duplicate messages at the point of the screened routing decision.

**Discard expired incoming messages**

This is a switch to discard messages where the expiry time (ZPW) has exceeded.

**Discard expired outbound messages**

This is a switch to discard expired (ZPW) on outgoing connections.

**Discard expired screened messages**

This is a switch to discard expired (ZPW) messages at the point of the screened routing decision.

**Encrypted Message Action**

What action to take on receipt of a grouper (has the GR field present) message. One of

- **Forward as Normal**- treat the message as nothing special and submit as normal.
- **Discard**- The message is deleted and not passed on.

- **Place on Encrypted Queue**- The message is placed in a database queue for further processing by the mconsole interface.

See [M-Switch Operator's Guide](#).

### Intercept Action

If a message is received and it is not for local handling nor to be relayed (see [Guard action](#)), possible options are:

- **unset** the default action of forward is taken.
- **Forward to intercept address** the intercepted message is forwarded to the configured intercept address.
- **Discard** the intercepted messages are discarded.
- **Place on Intercept Queue** the intercepted messages are placed on a queue to be further processed by the appropriate mconsole view.

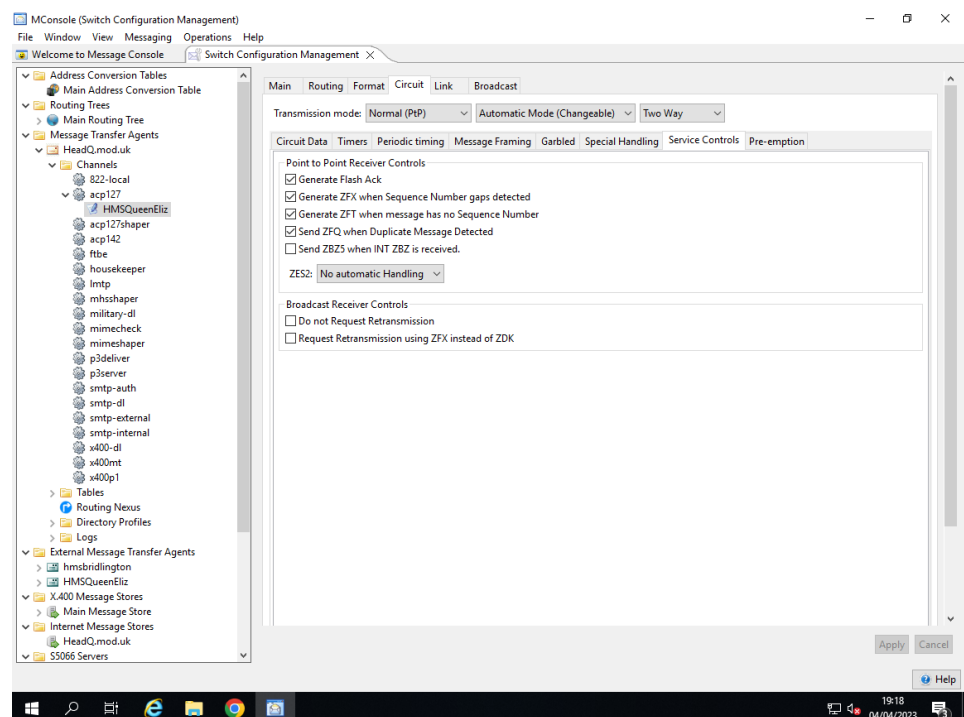
See the [M-Switch Operator's Guide](#).

### Intercept Address

Address to which to send messages received not explicitly for this station.

## 195.324.7 Service Controls

**Figure 19.18. ACP127 editing the Circuit Tab, Service Controls**



These parameters apply to the generation of service messages

### Generate Flash Ack

If set a Flash Ack will be generated for Flash and higher priority messages.

### Generate ZFX when Sequence Number Gaps detected

Generate a ZFX service message when message received and there is a missing sequence number.

### Generate ZFT when message has no sequence Number

Generate a ZFT service message if a message is received which has no sequence number.

### Send ZFQ when Duplicate Message detected

Generate a ZFQ service message when message is received which is the same as a message already received.

### Send ZBZ5 when INT ZBZ is received

A service message with the OPSIG 'INT ZBZ' is a request for information on the quality of the signal being received. If the option "Send ZBZ5 when INT ZBZ is received" is set, when such a service message is received, an automatic response of a service message with ZBZ5 will be returned. This indicates that the signal is "Acceptable - no corruption".

### ZES2 Handling

A service message with the OPSIG ZES2 is reporting the receipt of a garbled message. It is possible for the message to be resent on receipt of this message. The service message must have identified the message using the channel number of the received message. There are three modes of operation:

- **No automatic Handling** The ZES2 does not cause resending (the default)
- **Handles ZES2 local orig.** The message is resent only if it originated at the station receiving the ZES2
- **Handle ZES2 for all** The message is resent

If the message is not resent, then the service message is sent to the operator.

### Do not Request Retransmission

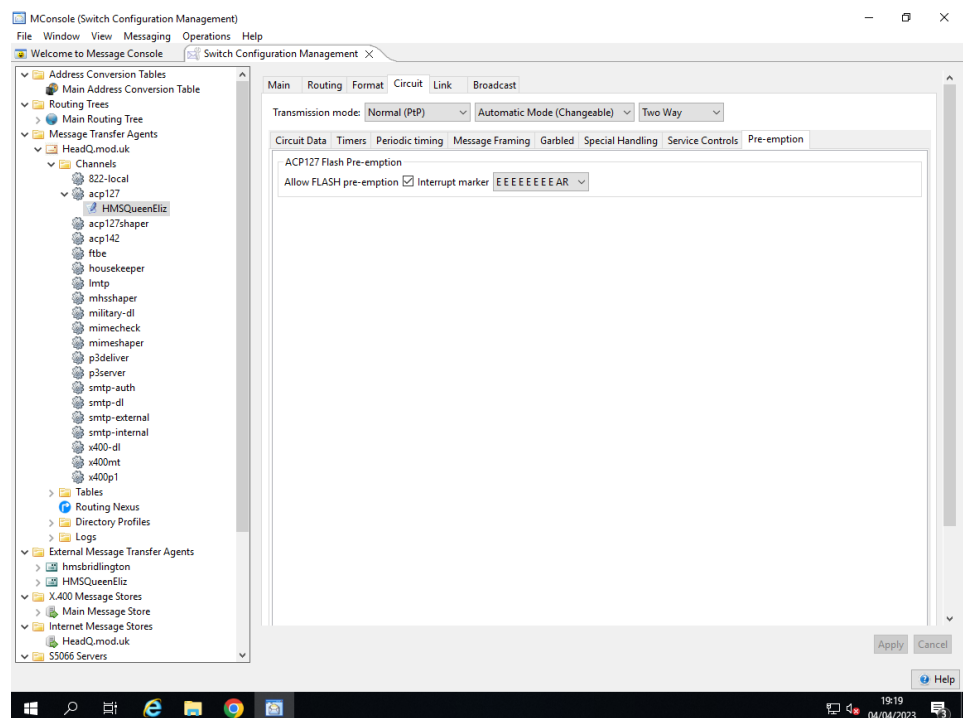
By default, retransmission of missing messages is requested using INT ZDK service messages. Selecting this option means that no retransmission is requested.

### Request Retransmission using ZFX instead of ZDK

Selecting this option causes retransmission of missing messages to be requested using ZFX service messages.

## 1953248 Pre-emption

**Figure 19.19. ACP127 editing the Circuit Tab, Pre-emption**



This section governs the control of interrupted traffic.



### Allow FLASH preemption

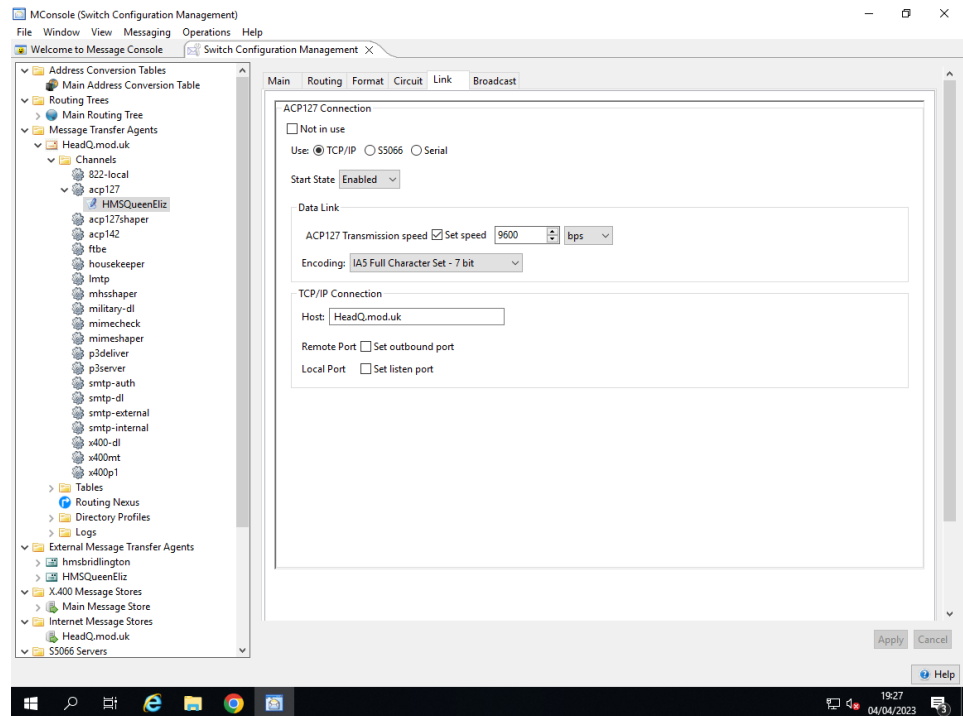
When enabled this allows FLASH traffic and higher to preempt messages in transit by interrupting them and so the FLASH message may overtake it.

### Interrupt Marker

This is the style of interrupted marker to include.

## 19.5.3.25 ACP127 Peer Connection: Link Tab (TCP/IP)

**Figure 19.20. ACP127 editing the Link Tab of a peer connection (TCP/IP)**



The parameters are as follows:

### Not in use

This specifies the circuit is not in use. It is disabled and cannot be enabled in the ACP127 View.

### ACP127 Connection

This specifies the underlying transport for this Link.

- **TCP/IP**
- **S5066**
- **Serial**

### Start State

One of:

- **Enabled**

Attempts to connect when the channel starts.

- **Disabled**

Connects only when specifically enabled.

- **Previous**

Restore previous state, or disabled if no information available.

### Data Link

#### ACP127 Transmission Speed

If set, the transmission speed of the circuit is limited to this value.

### Encoding

One of:

- **IA5 Upper Case Letter Only - 7Bit**
- **ITA2 - 5 bit**
- **IA5 Full Character Set - 7 bit**
- **8 - bit**

### Host

This is the remote hostname to connect to, or accept connections from.

### Remote Port

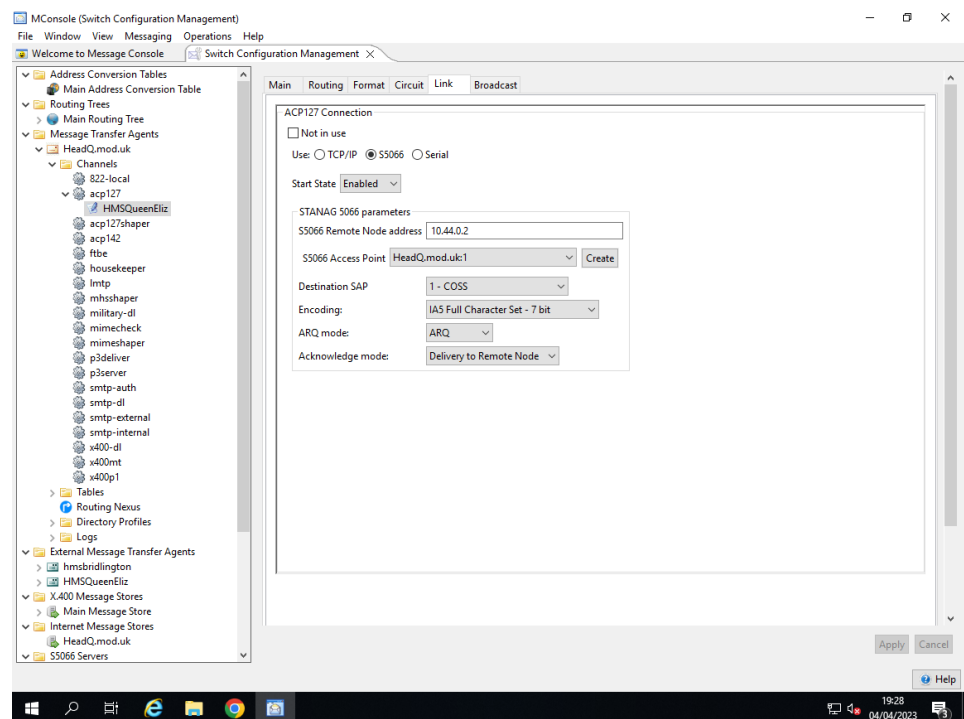
If the circuit is an initiator and this is not zero, it implies a connection will be made to the remote host to deliver messages.

### Local Port

If this is not zero, it is a specific listen port for TCP connections to arrive on associated with this station. The acp127 server will therefore include it in its list of ports on which to listen.

## 19.5.3.26 ACP127 Peer Connection: Link Tab (5066)

**Figure 19.21. ACP127 editing the Link Tab of a peer connection (5066)**



The parameters are as follows:

### Not in use

This specifies the circuit is not in use. It is disabled and cannot be enabled in the ACP127 View.

### ACP127 Connection

This specifies the underlying transport for this Link.

- **TCP/IP**
- **S5066**
- **Serial**

**Start State**

One of:

- **Enabled** Attempts to connect when the channel starts.
- **Disabled** Connects only when specifically enabled.
- **Previous** Restore previous state, or disabled if no information available.

**S5066 Remote Node address**

This is the 5066 address of the remote S5066 entity.

**S5066 Access Point**

This is a Stanag S5066 Access Point which specifies access to a S5066 protocol server that can be selected from the list in the dropdown menu. These servers can be created, managed and remove through the S5066 Access Points as described in [Figure 19.30, “Editing a STANAG 5066 Access Point”](#).

**Destination SAP**

The destination SAP used with the COSS protocol to connect to remotely. It should normally be 1 (COSS) unless special circumstances require it otherwise.

**Encoding**

The type of encoding used on the link. This has to be bilaterally agreed between the sending and receiving stations. The options are:

- **Full 8 bit**- the data is passed as is.
- **IA5**- the data is passed as ascii.
- **ITA2 - Loose Pack**- the data is passed as 5 bit ITA2 characters loosely packed into 8 bit bytes.
- **ITA2 - Dense Pack**- the data is passed as 5 bit data but packed 3 ITA2 characters to two 8 bit bytes.

**ARQ Mode**

The style of ARQ mode in use. It can be one of:

- **ARQ**- Request reliable delivery .
- **Non-ARQ**- Request unreliable delivery (suitable for broadcast circuits).

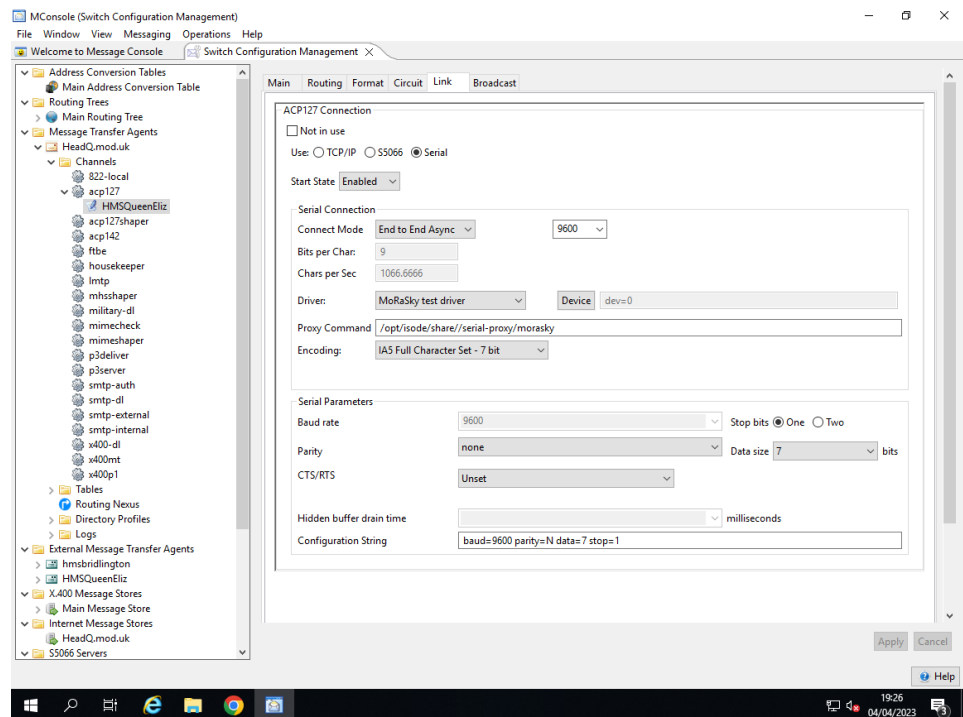
**Acknowledge mode**

The style of acknowledgement to wait for, from one of:

- **No confirmation**- No acknowledgement is expected.
- **Delivery to Remote Node**- Request acknowledgement from the local server when the data unit has been received.
- **Delivery to remote client**- Request confirmation from the remote server when the data has been received.

### 19.5.3.27 ACP127 Peer Connection: Link Tab (Serial)

Figure 19.22. ACP127 editing the Link Tab of a peer connection



The parameters are as follows:

#### Not in use

This specifies the circuit is not in use. It is disabled and cannot be enabled in the ACP127 View.

#### ACP127 Connection

This specifies the underlying transport for this Link.

- **TCP/IP**
- **S5066**
- **Serial**

#### Start State

One of:

- **Enabled** Attempts to connect when the channel starts.
- **Disabled** Connects only when specifically enabled.
- **Previous** Restore previous state, or disabled if no information available.

#### Serial Connection

##### Connect Mode

- **Connect to sync** Used to specify a synchronous connection to the serial device.
- **End to End Async** Used to specify an asynchronous connection to the serial device.

##### Bits per Char

This field shows how many bits will be required to represent a single character, taking into account error correction and encoding etc.

##### Chars per sec

This field shows how many characters will be transferred per second, taking into account error correction and encoding etc.

**Driver**

- **Serial Proxy**
- **Digiport TS Server (RFC2217)**
- **MoRaSky test driver** The Isode Radio simulator.

**Device**

The device parameters. This is a device parameter of the form dev=N, where N is a number from 0 for serial line and MoRaSky drivers. If the Digiport server is selected, then this is a TCP/IP host:port specification of where the TELNET protocol will connect to drive the modem.

**Proxy Command**

For direct serial lines, and the MoRaSky simulator, a proxy command is started that is used to drive the serial line directly. This is specified here.

**Encoding**

What encoding is to be used on the line. This can be full 8 bit data, 7 bit ascii, or 5 bit ITA2. The underlying data size must support the option chosen here.

**Serial Parameters****Baud rate**

The speed of the line - one of the configured values may be chosen, or a supported speed typed directly.

**Stop bits**

Whether to use one or two stop bits on the serial line.

**Parity**

The type of parity to use. There are five options: none, even, odd, space and mark with their conventional meanings.

**Data size**

What size the byte size is on the wire. It can be any from 5-8. Note that sizes smaller than 7 will not support ascii data.

**CTS/RTS**

This option controls how the CTS/RTS flow control options work.

**Unset**

There is no attempt to change the CTS/RTS flow control.

**RTS always off**

The RTS is forced to be off and not used.

**RTS always on**

The RTS signal is always high.

**One way**

Use RTS/CTS sending flow control.

**Two way**

The RTR and CTS are used for flow control.

**Transmit**

The RTS signal is held high during transmission and cleared after.

**Hidden Buffer Drain Time**

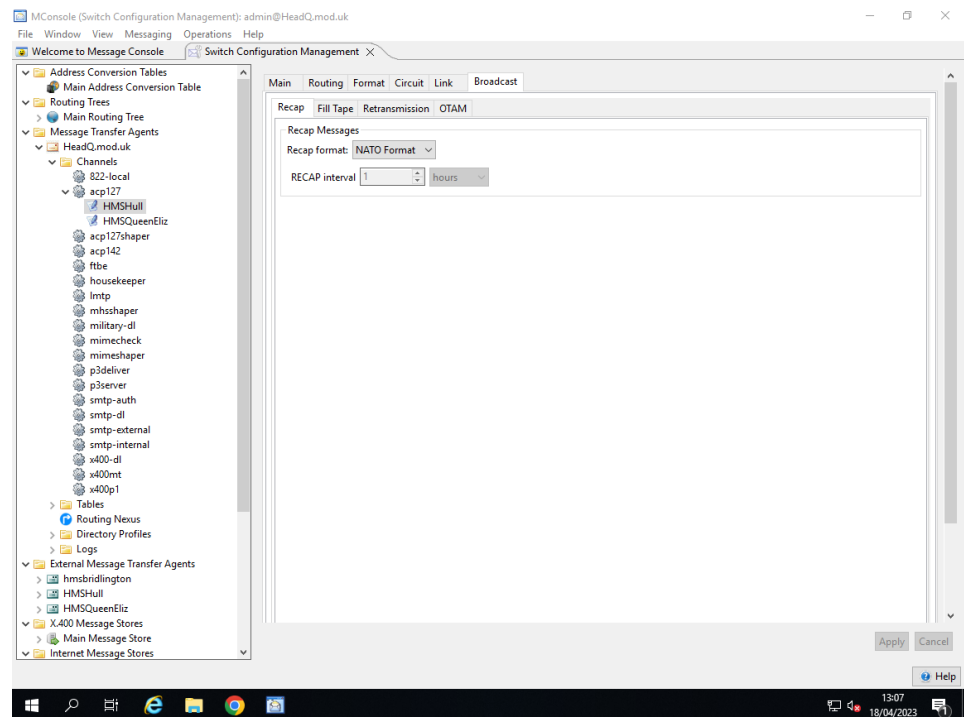
Occasionally there are extra buffers in the modems that take time to drain. This parameter lets a small amount of extra time to be allowed after a message is sent to allow the buffers to drain.

**Configuration**

The last test box shows the configuration summary for the line. Although this is editable to set additional values, it should not normally be changed.

### 19.5.3.28 ACP127 Peer Connection: Broadcast Tab

**Figure 19.23. ACP127 editing the Broadcast Tab of a peer connection (Recap)**



The following values can be configured when the Recap tab is selected.

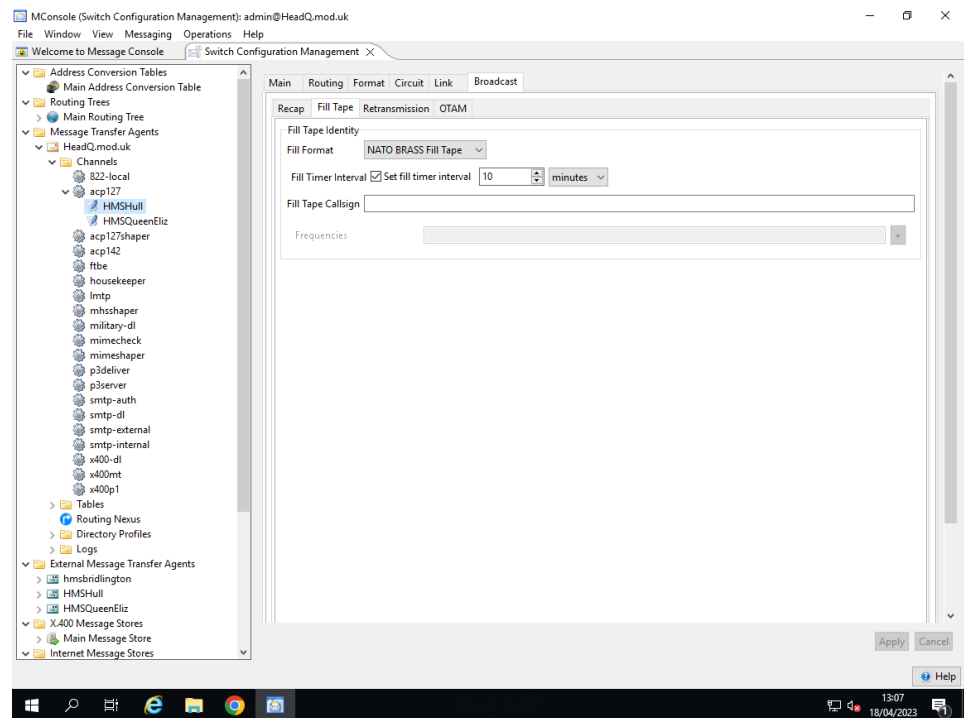
#### **Recap format**

The style of recap message format to be used. This applies to both sender and receiver circuits. For sending circuits it is the format to construct, and for receiving circuits it is the format expected and to parse.

#### **Recap interval**

The time interval for the broadcast sender to send recap messages. The time is rounded up to an appropriate boundary, so if for example they are sent hourly, they will be sent on the hour.

**Figure 19.24. ACP127 editing the Broadcast Tab of a peer connection (Fill Tape)**



The following values can be configured when the Fill Tab is selected.

#### **Fill format**

The style of fill message format to be used. This applies to both sender and receiver circuits. For sending circuits it is the format to construct, and for receiving circuits it is the format expected and to parse.

Allowed values are NATO BRASS format, or Italian format.

#### **Fill timer interval**

The time for the broadcast sender to send a fill tape message if not other traffic has been sent for this period. For a broadcast receiver, this is the time to issue an audit log if not traffic has been received in this time.

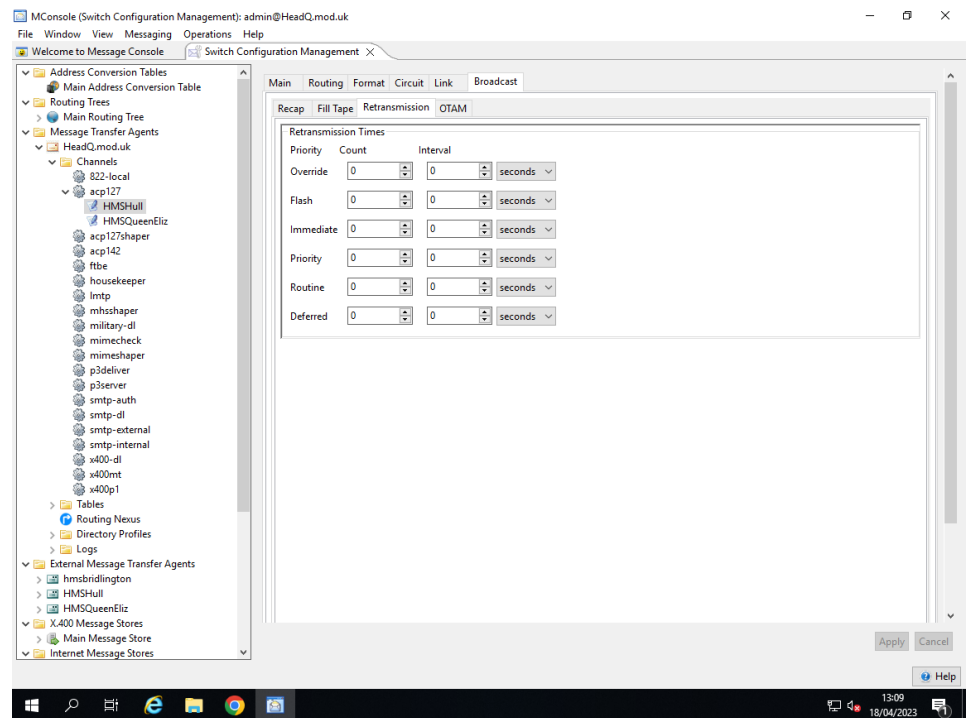
#### **Fill tape call sign**

This is the value for the call sign that is a component of some call tape messages.

#### **Frequencies**

This is a list of frequencies to be specified in the call tape message.

**Figure 19.25. ACP127 editing the Broadcast Tab of a peer connection (retransmission)**

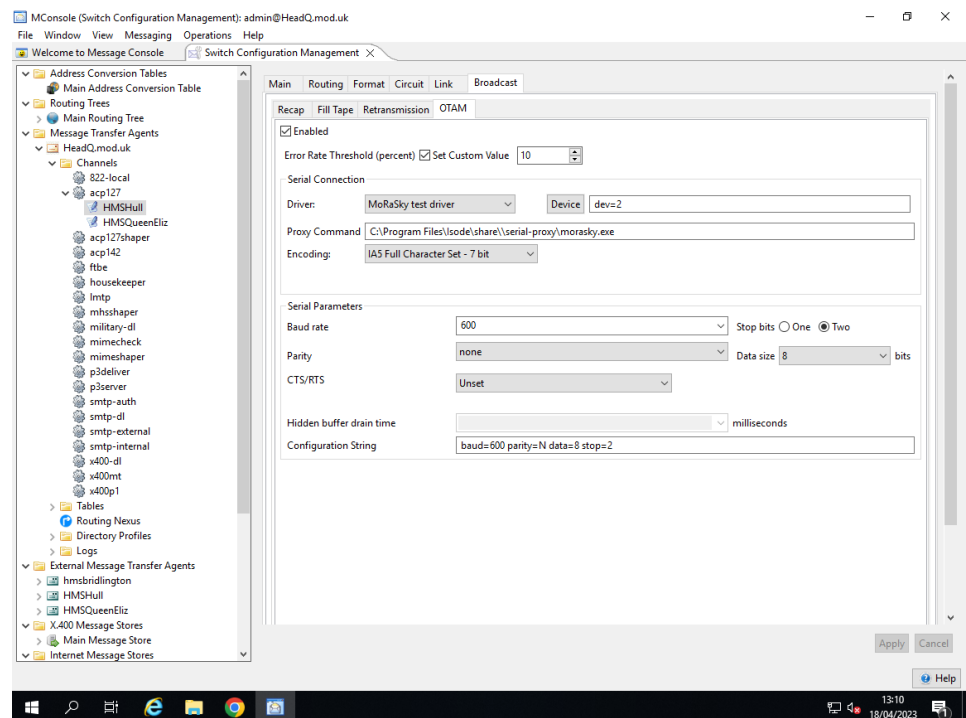


The following values can be configured when the Retransmission Tab is selected.

### Retransmission times

This panel allows the configuration of optional retransmission intervals, selectable by priority. Thus, for example, a FLASH message may be configured to be sent twice, with 20 seconds between each retransmission.

**Figure 19.26. ACP127 editing the Broadcast Tab of a peer connection (OTAM)**



The following values can be configured when the OTAM Tab is selected.



### Enabled

This is available only for broadcast sender circuits. It allows monitoring of the transmission by comparing what is sent with what is received. When the enabled box is ticked, then a serial line monitoring circuit can be configured to listen in to a received transmission and compare.

### Error Rate Threshold

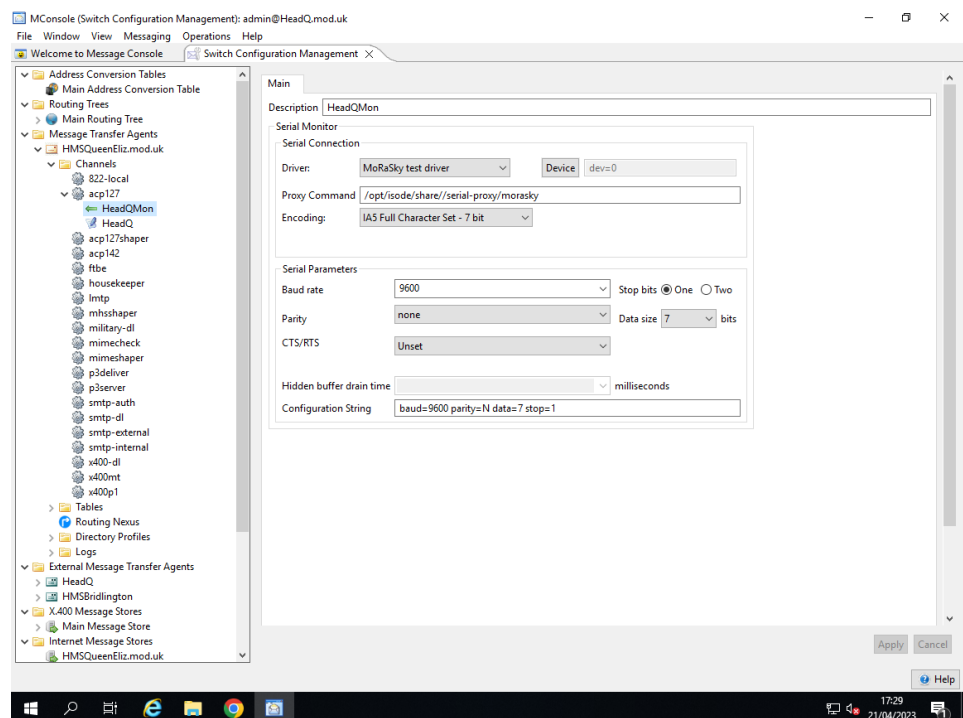
Option to set an Error Rate Threshold percentage. If the error rate exceeds the percentage set, an error will be created.

The other options (for Serial Connection and Serial Parameters) are the same as [Figure 19.22](#), “ACP127 editing the Link Tab of a peer connection”.

## 19.5.3.3 Monitored circuits

Monitored circuits are special circuits used to monitor FAB transmissions. These are serial only links, and are combined with an ACP127 transmission circuit. This allows an operator to see inbound FAB messages, while transmitting.

**Figure 19.27. Editing a monitored link**



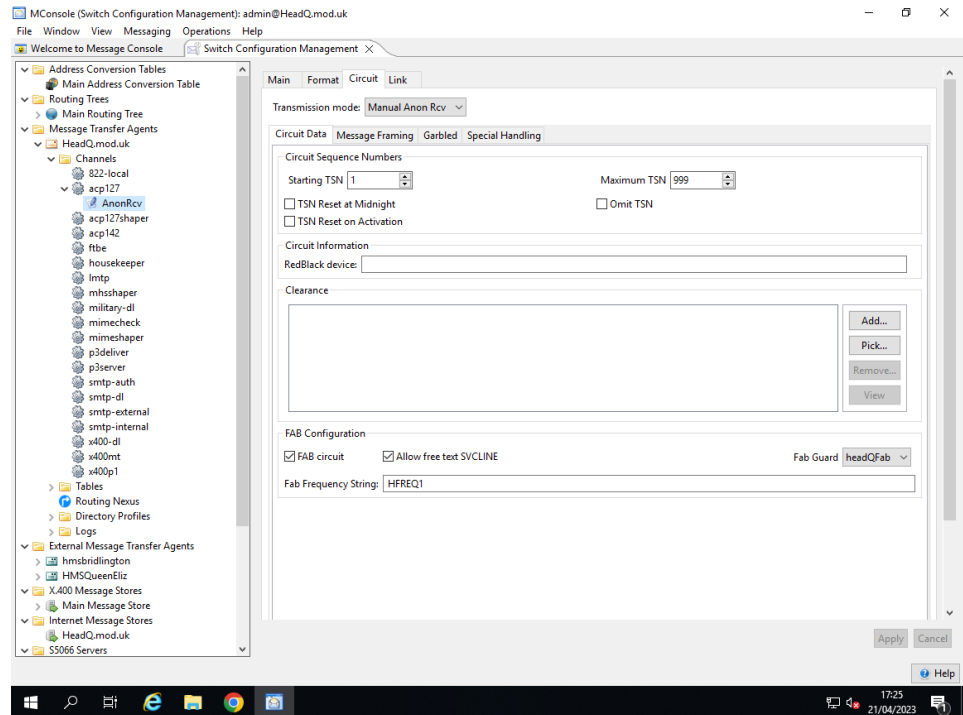
## 19.5.3.4 Receive Only Connection

Anonymous receive only circuits allow a station to receive messages without having setup a remote ACP127 configuration.

---

**Note:** The circuit configuration options are a subset of a normal ACP127 peer connection. With the exception of the **Circuit->Circuit Data->Fab configuration** options.

---

**Figure 19.28. Editing Anonymous Receive FAB options****FAB Circuit**

If selected then this circuit will use FAB to broadcast it's status.

**Allow free text SVCLINE**

If selected then the operator will be able to enter their own text for the SVCLINE FAB status, alternatively the operator will have to pick SVCLINE messages from a predefined catalog. See [Section 19.5.3.5, “FAB SvcLine configuration”](#) for details of that.

**Fab Guard**

This selects the FAB Guard that will be used when transmitting the FAB status. FAB statuses are transferred through to this FAB guard, which will then pass them through to a FAB server. The FAB server will then broadcast those messages.

**Fab Frequency String**

This is a string identifier used to allow the remote side to identify which FAB station is broadcasting the FAB status

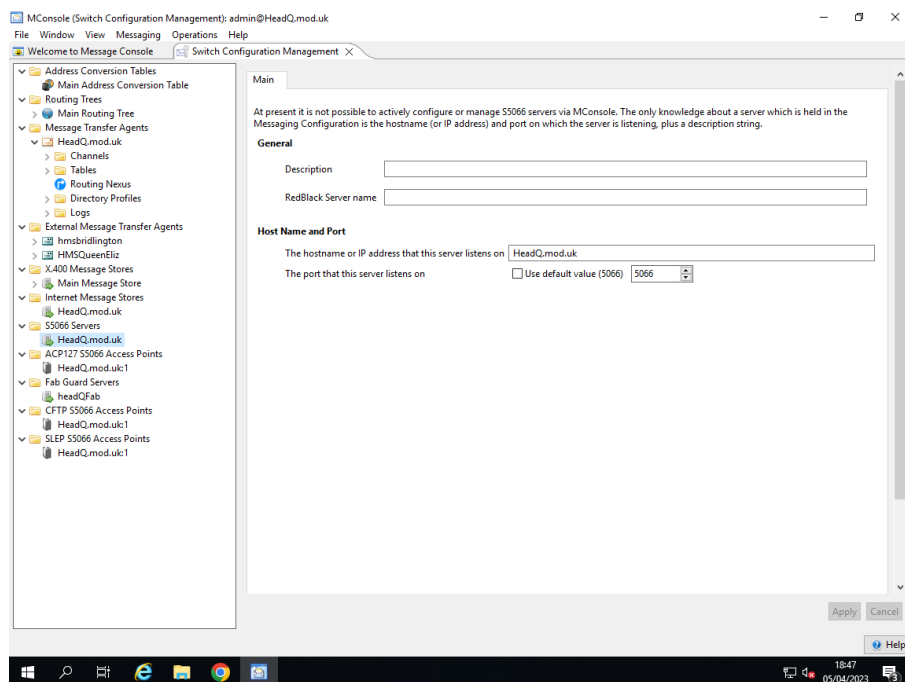
**19.5.3.5 FAB SvcLine configuration**

The FAB protocol allows both freetext (limited to 64 chars) and also items from a catalog of preprepared strings. This catalog is managed by an administrator and must be also installed on the machine running the fabserver.

When using a catalog just the line number is passed across so leading to more secure operation. The format of the file is just single lines of text with the exception of lines beginning with a # or empty lines are ignored. There is a sample file installed which will not be used but shows the format. The file is in *(ETCDIR)/FABserver/svcline.txt.sample* and should be copied to *(ETCDIR)/FABserver/svcline.txt* and edited to suit the local requirements.

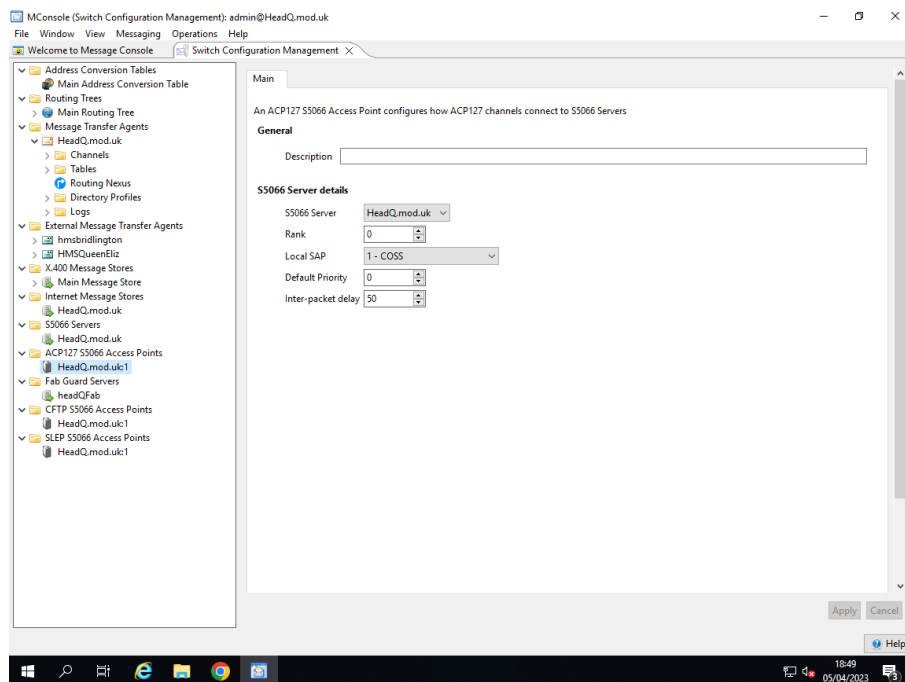
**19.5.4 Setting up STANAG 5066 Configuration****S5066 Servers**

S5066 Servers are represented in MConsole as top level objects, with their own editor. These can be shared by circuits and channels (such ACP127 and ACP142).

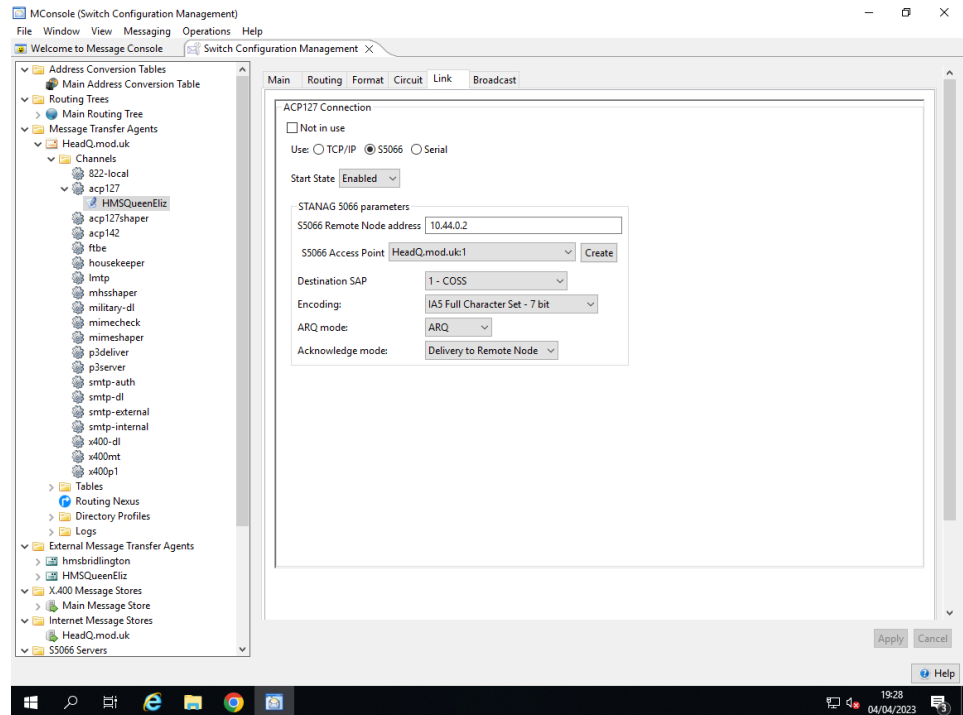
**Figure 19.29. Editing a STANAG 5066 Server**

### S5066 Access Points

S5066 Access Points are represented in MConsole as top level objects, with their own editor. S5066 Access Points are only used by ACP127 channels (ACP142 channels access 5066 Servers directly.)

**Figure 19.30. Editing a STANAG 5066 Access Point**

The following figure shows where the S5066 Access Points can be selected in the ACP127 circuit link Tab.

**Figure 19.31. Selecting a STANAG 5066 Access Point from an ACP127 Circuit**

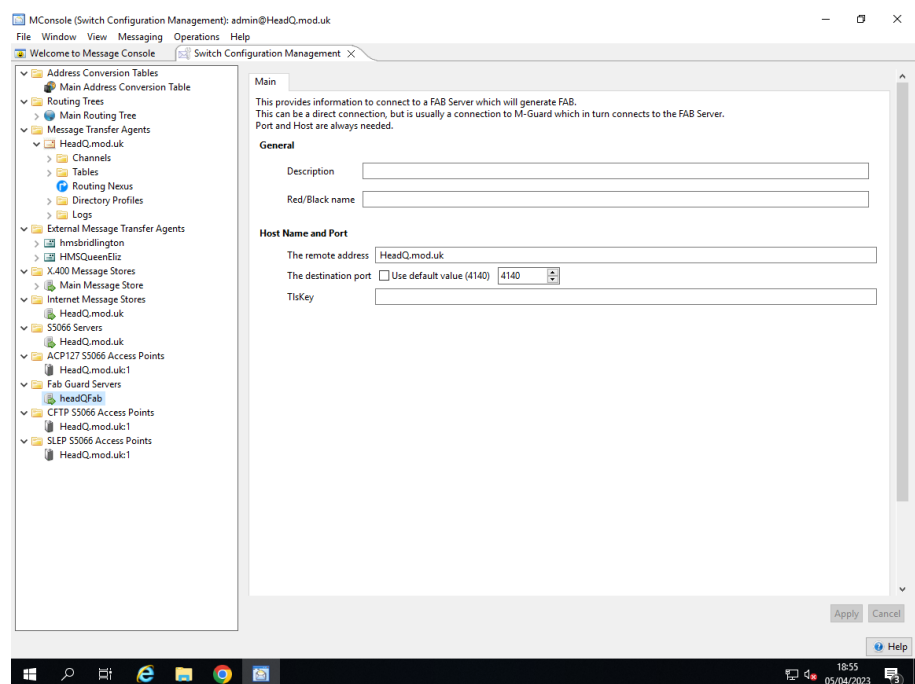
## 19.5.5 Setting up a FAB Server Configuration

### FAB Guard Servers

FAB Guard connections are configured here and used as part of the circuit configuration when appropriate.

### Editing a FAB Guard Server

The connection to each fab server which are normally accessed through an XML guard are configured here,

**Figure 19.32. Editing a FAB Guard Server**

This contains the following settings:

**Description**

A free text description that isn't used for any internal use.

**Red/Black name**

This associates a red/black label with this fab server circuit.

**Remote address**

The hostname/IP address of the guard or remote fab server that is connected to over GCXP.

**Destination port**

The port to connect to with the host for GCXP.

**TlsKey**

This is an optional key into the certificate database to pick a specialised key for use with TLS based on [Section 14.1.6, "Security settings"](#). If not selected the default TLS information will be used.

---

## 19.6 Setting up Routing

The final step of configuring an ACP127 Gateway is to set up the user mappings and routing.

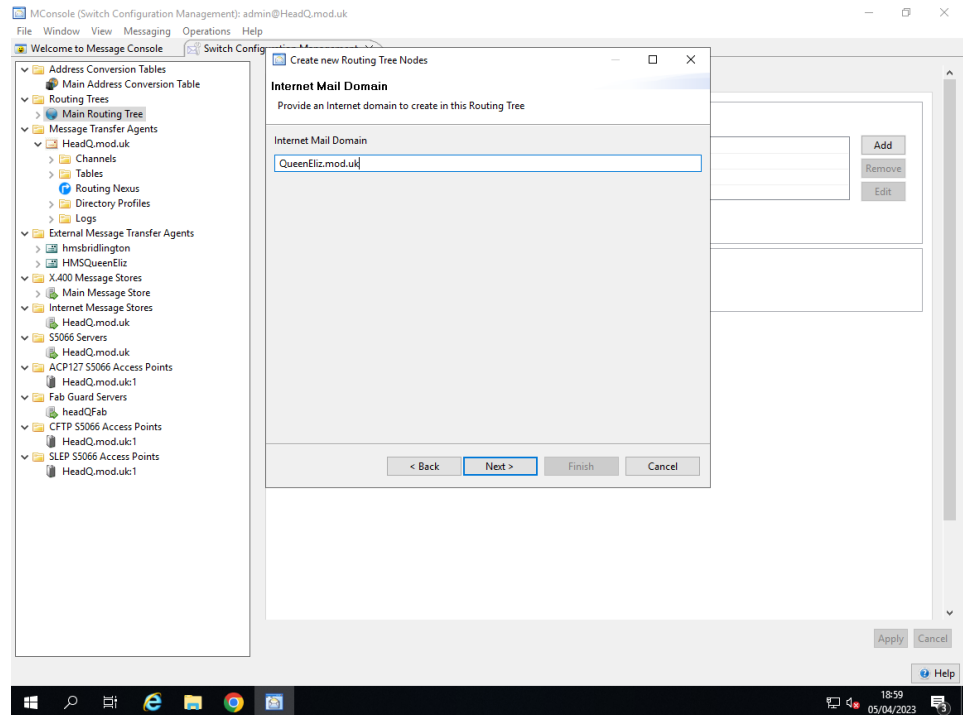
Firstly, the Gateway needs to be configured to map ACP127 RIs to Internet or X.400 addresses (for example RABCA may map to `a@example.net`). Secondly, the Gateway needs to be configured to select the appropriate peer-connection for the external ACP127 Station to which to route.

There are two ways to set this up. The decision depends on whether the domains that the RIs map to are all routed to the same ACP127 Station. If they are then they can be configured using Gateway Users. If they are not (for example you want RABCA mapped to `a@example.net`, and RABCB to mapped to `b@example.net`) but want them routed to different external ACP127 Stations (via different peer connections) then they need to be configured via Mailbox Users. These are described in detail below.

### 19.6.1 Setting up Routing with Gateway Users

#### 19.6.1.1 Add a node to the Routing Tree

To add routing to the new external MTA, create a node representing the Internet address by navigating to the **Main Routing Tree** node under the **Routing Trees** entry in the **Switch Configuration Management** view. Right click and choose **Add nodes** and select the **Create Routing Tree entries representing an Internet domain**. Choose **Next** and enter the Internet domain of the entry, e.g. `vessel1.net` such as shown in [Figure 19.33, "ACP127 creation of Routing Tree node."](#) Select **Finish**.

**Figure 19.33. ACP127 creation of Routing Tree node.**

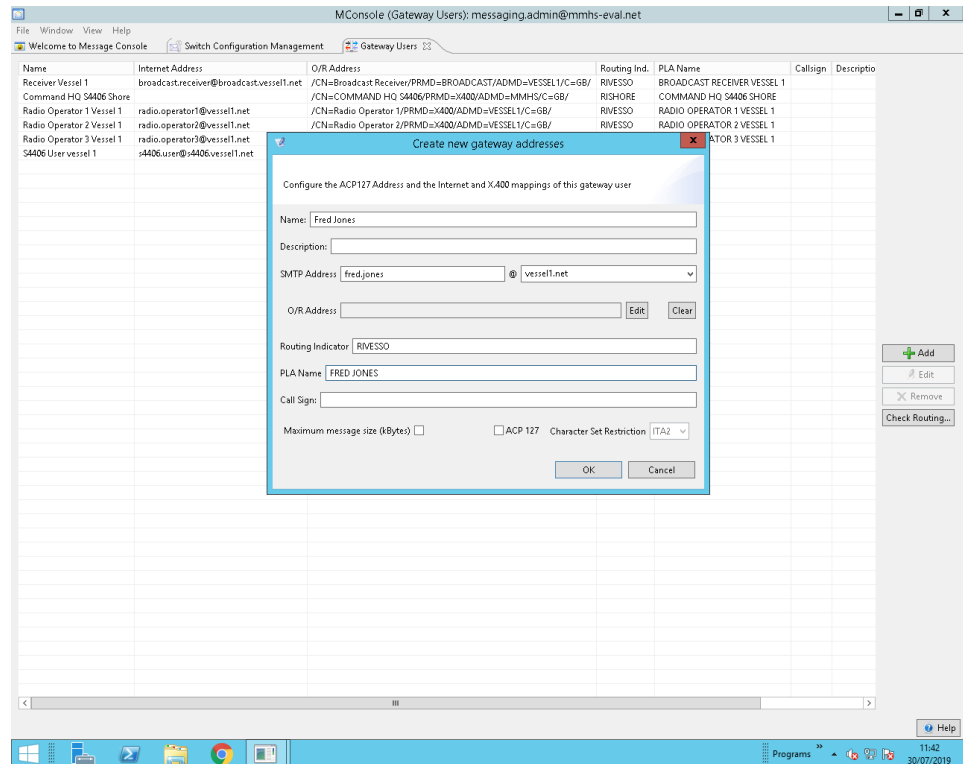
Once this is created, select the entry, and click on the **Add** to connect to the external MTA. Make sure the correct MTA is selected in the drop down menu. The default weight is normally acceptable unless complex setup is required.

## 19.6.2 Adding Gateway Users

### 19.6.2.1 Add remote ACP127 user mappings

To add a remote user mapping which configures the routing indicator and PLAD of an ACP127 mailbox together with the Internet or X.400 address which is to be gatewayed to this address, switch to the **Gateway Users** tab. This can be created by choosing **View** → **Configuration** → **Gateway Users**.

To create the remote ACP127 user click on the **Add** button to bring up a new dialog. The entry currently requires a name, which can describe the mapping. Fill in the email address of the remote user, and also the associated routing indicator, and plain language address as shown in [Figure 19.34, “Gateway user creation for ACP127”](#).

**Figure 19.34. Gateway user creation for ACP127**

### 19.6.2.1.1 Testing remote user mappings

You need to ensure that the user configured in [Figure 19.34, “Gateway user creation for ACP127”](#), has both Internet routing information, and ACP127 routing information, using the command line tools available as described below.

Firstly check that the Internet address is routed to the ACP127 gateway by the acp127 channel. Do this as follows:

```
ckadr fred.jones@vessell.net
fred.jones@vessell.net - (rfc822)
fred.jones@vessell.net
fred.jones@vessell.net - (x400)
/CN=FRED JONES/PRMD=X400/ADMD=MMHS/C=gb/ Delivered
to cn=acp127,cn=Vessel 1 ACP 127 Receive,cn=Messaging
Configuration,o=MMHS Evaluation by acp127 (weight:5)
```

Secondly, check the ACP127 setting of the user are configured correctly. Do this as follows:

```
ckacp127adr -i fred.jones@vessell.net
PLAD: FRED JONES RI: RIVESSO
```

## 19.6.3 Setting up Routing with Mailbox Users

**Note:** This is not available for X.400 mailboxes.

### 19.6.3.1 Add ACP127 Internet User Mailbox mappings

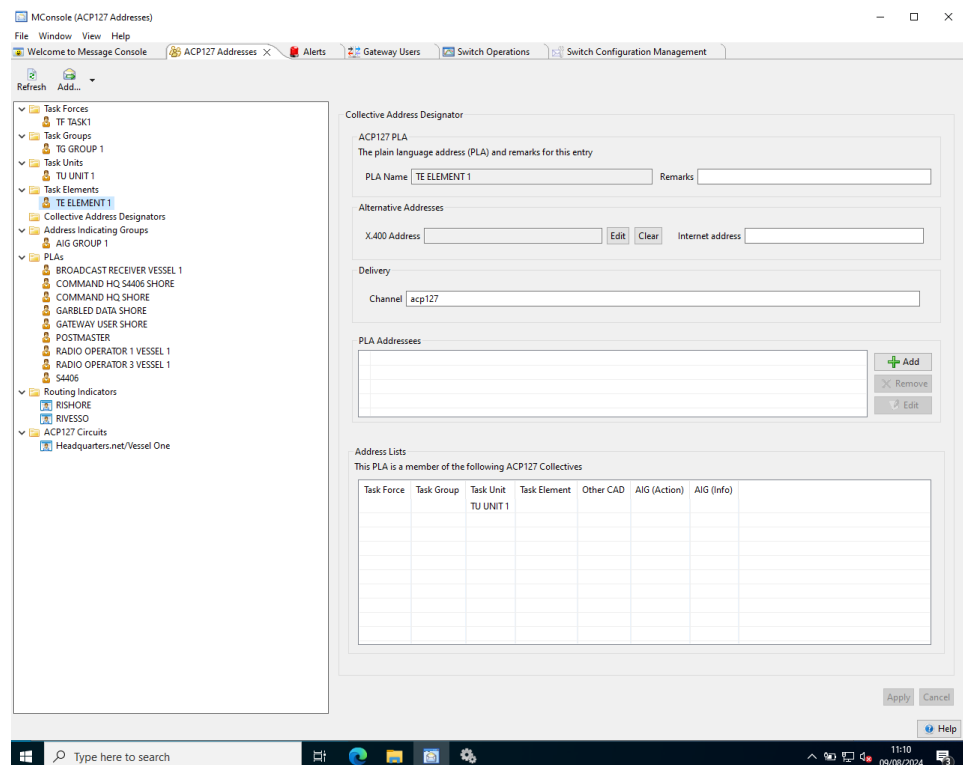
If addresses of the same domain need to be routed to different external ACP127 Stations then they first need to be routed to the local MTA. This is done by creating the domain in the Routing Tree as described in [Section 19.6.1.1, “Add a node to the Routing Tree”](#) but instead of selecting an External MTA (i.e. ACP127 Station), selecting the local MTA.

Internet mailboxes then need to be created for each user that needs to be routed. Consult the **Cobalt Administration Manual** to do that.

## 19.6.4 ACP127 Addresses View

The ACP127 Addresses View allows the creation and configuration of Task Forces, Collective Address Designators and Address Indicating Groups. It also allows the inspection of Plain Language Addresses, Routing Indicators and ACP127 Circuits, including the performance of routing checks.

**Figure 19.35. ACP127 Addresses View**



### 19.6.4.1 Task Forces and CADs

A Collective Address Designator (CAD) is a form of Distribution List - it allows PLAs to be grouped together for easier addressing. A CAD is address using its own PLA, and can have associated X.400 and/or Internet addresses. CAD expansion (i.e. replacing the CAD address with the member PLAs) is performed within the acp127 channel.

Task Forces, and their component Task Groups, Task Units and Task Elements are CADs with fixed PLA naming prefix scheme:

- A Task Force is named using a PLA which starts with the characters "TF". It may have subordinate Task Groups, Task Units, Task Elements or arbitrary PLAs.
- A Task Group is named using a PLA which starts with the characters "TG". It may have subordinate Task Units, Task Elements or arbitrary PLAs.
- A Task Unit is named using a PLA which starts with the characters "TU". It may have subordinate Task Elements or arbitrary PLAs.
- A Task Element is named using a PLA which starts with the characters "TE". It may contain arbitrary PLAs.

New **CADs** can be created by either right clicking on the **Collective Address Designators**, and following the wizard or by selecting **Add..** from the toolbar.



This also applies to other forms of CADs, such as **Task Forces**, **Task Groups**, **Task Units** and **Task Elements**.

Once created the **PLA** of one **CAD** or individual maybe associated with other **CADs**. For instance after creating a **Task Force**, and a **Task Group**, select the Task Force, then add a new **PLA Address**, add the Task Groups PLA, and the Task Group is now part of the Task Force.

#### 19.6.4.2 Address Indicating Groups

An Address Indicating Group is very similar in function to a CAD, except that it has can contain both Action and Information addressees. Address Indicating Groups are created in the same manner as CADs and Task components.

#### 19.6.4.3 Plain Language Addresses

The ACP127 Addresses View allows Gateway users which have PLAs to be viewed but not modified, added or deleted. The "Check Routing" function allows the routing of the address to be tested on available MTAs.

#### 19.6.4.4 Routing Indicators

This folder provides a list of the configured Routing Indicators. For each Routing Indicator which is found, the set of PLAs which have that Routing Indicator is displayed.

#### 19.6.4.5 ACP127 Circuits

This folder displays all of the configured ACP127 circuits in the Messaging Configuration. If a circuit is selected, a list of all of the PLAs whose Routing Indicators route to that circuit is generated and displayed.

---

## 19.7 ACP127 Security Labels

This section covers how Security Labels are gatewayed to ACP127 from either X.400 or Internet networks and vice versa. M-Switch contains powerful and flexible features to carry and manage Security Labels by extracting the Security Label from an incoming message, and inserting a Security Label into an outgoing message. For a full description of these features, see the *M-Switch Advanced Administration Guide*.

### 19.7.1 Security Policy Information Files (SPIF)

See [Chapter 18, Security Labels and Access Control](#) for an overview of Security Labels in M-Switch and a description of SPIFs and how to configure them.

It is possible to configure M-Switch as an ACP127 Gateway without a SPIF which will result in Security Labels traversing the Gateway in a very simple manner. Generally it is likely that a SPIF will be used.

A default SPIF is installed in *(SHAREDIR)/policy.xml*. M-Switch is installed with other configuration files (described later in this section and in [Section 18.1.1.2, "Configuring M-Switch Conversion"](#)) which refer to this SPIF.

While this SPIF is suitable for testing and evaluation, deployments will need to use a SPIF tailored to their own security environment.

## 19.7.1.1 Configuring an ACP127 Gateway to use a SPIF

### 19.7.1.1.1 Configuring a Security Policy

See [Section 18.1.1.1, “M-Switch Conversion Configuration Files”](#) for a description of configuring *mhsmixer-shaper.xml*, *mimemixer-shaper.xml*, *submitscanconfig.xml*.

#### Example 19.1. Configuring a Security Policy (*acp127-shaper.xml*)

```
<?xml version="1.0" standalone="yes"?>
  <shaper exploder="acp127">
    <security-policy name="acp127-map"
      default-policy-id="1.2.6.1.4.1.453.28.1687253052.1.1">
      <spif file="switch/acp127-policy.xml" />
    </security-policy>
  </shaper>
```

---

**Note:** The above example is as the file installed. The `file` value is the path relative to (*ETCDIR*). If you change the name of your SPIF you will need to ensure you have the correct `default-policy-id` and `file` values.

---

### 19.7.1.1.2 Extract a Security Label from an ACP127 Message

Add the `security-policy` name to the `param` element of the `scanner` element as in the below example.

#### Example 19.2. Extracting a Security Label from an ACP127 Message (*submitscanconfig.xml*)

```
<scancontent type="acp127" exploder="acp127">
  <scanner type="content" cost="1">
    <filter command="acp127:scan">
      <param name="security-policy">acp127-map</param>
    </filter>
  </scanner>
</scancontent>
```

### 19.7.1.1.3 Extract a Security Label from an X.400 Message

See [Section 18.1.1.2.2, “Extract a Security Label from an X.400 Message”](#)

### 19.7.1.1.4 Extract a Security Label from an Internet Message

See [Section 18.1.1.2.3, “Extract a Security Label from an Internet Message”](#)

### 19.7.1.1.5 Extract a FLOT from an Internet Message

See [Example 18.5, “Extracting a FLOT \(submitscanconfig.xml\)”](#)

### 19.7.1.1.6 Inserting a Security Label from an Internet Message into an ACP127 Message

ACP127 messages cannot carry a full Security Label, but can only contain a Security Label Marking (i.e. a string representing the Classification). An additional `param` is added to the `filter` element to enable the convertor to use the Security Policy and insert the Marking.

#### Example 19.3. Inserting a Security Label extracted from an Internet Message into an ACP127 messages (*mimemixer-shaper.xml*)

```
<!-- ACP127 CONVERSIONS -->
<!--
  Convert to ACP127
```

```

-->
<output type="acp127" flattener="acp127">
  <convert type="header" action="discard" cost="2"/>
  <convert type="header" action="convert" cost="1">
    <match name="Depth">1</match>
    <filter command="acp127:mime2acphdr">
      <param name="convcharset">transtoascii</param>
      <!-- Add reference to Security Policy -->
      <!-- so that Security Label Marking -->
      <!-- is inserted into the ACP127 message -->
      <param name="security-policy">acp127-map</param>
    </filter>
  </convert>

```

#### 19.7.1.1.7 Inserting a Security Label into an X.400 Message from ACP127

**Example 19.4. Inserting an Security Label extracted from an Internet Message into an ACP127 messages (acp127-shaper.xml).**

```

<!-- ===== MIXER =====>
  Convert to P22/P2/P772
-->
<output type="p2orp22" flattener="ppmhs" name="To-P22"
  description="Convert to P22">

  <convert type="envelope" action="convert" cost="1">
    <match name="Content-type">acp127</match>
    <!-- insert envelope label -->
    <filter command="pplablib:eninsert">
      <param name="security-policy">acp127-map</param>
      <param name="verify-label"/>
    </filter>
  </convert>

```

#### 19.7.1.1.8 Inserting a Security Label into an Internet Message

There are many ways in which a Security Label can be inserted into an Internet message. The example below adds SIO, X.411, X-Isode-Label headers and also adds a markup to the subject.

**Example 19.5. Inserting Security Labels into an Internet message gatewayed from ACP127 (acp127-shaper.xml)**

```

<!--Convert to 822 with only 7 bit encodings-->
<output type="822" flattener="ppmime">
  <convert type="header" action="convert" cost="1">
    <match name="Depth">1</match>
    <filter command="acp127:header">
      <param name="fold">76</param>
    </filter>

    <!-- various label insertions including subject -->
    <filter command="pplablib:labelinsert">
      <param name="xheader-format">SIO-Label: %e</param>
      <param name="security-policy">acp127-map</param>
      <param name="convert-label"/>
      <param name="verify-label"/>
      <param name="fixup-label"/>
    </filter>

```

```

<filter command="pplablib:labelinsert">
  <param name="xheader-format">X-Isode-Label: %l</param>
  <param name="security-policy">acp127-map</param>
</filter>

<filter command="pplablib:labelinsert">
  <param name="xheader-format">X-X411: %b</param>
  <param name="subject-format">Subject:
    %d (inserted on centos64-2)[SEC=%c]</param>
  <param name="need-policy-id"/>
  <param name="security-policy">acp127-map</param>
</filter>

<filter command="pplablib:labelinsert">
  <param name="xheader-format">SIO-Label: %e</param>
  <param name="security-policy">acp127-map</param>
</filter>

</convert>

```

#### 19.7.1.1.9 Inserting a Security Label into an X.400 Message

To insert a Security Label extracted from an ACP127 message as an X.411 Security Label into an X.400 message an additional `filter` element is added as in the example below:

#### Example 19.6. Inserting an X.411 Security Label into the Envelope of an X.400 message gatewayed from ACP127 (acp127-shaper.xml)

```

<!-- ===== MIXER =====
  Convert to P22/P2/P772
-->
<output type="p2orcp22" flattener="ppmhs" name="To-P22"
  description="Convert to P22">

  <convert type="envelope" action="convert" cost="1">
    <match name="Content-type">acp127</match>
    <!-- insert envelope label -->
    <filter command="pplablib:envinsert">
      <param name="security-policy">acp127-map</param>
      <param name="verify-label"/>
    </filter>
  </convert>
</output>

```

## 19.8 ACP127 Services

There are three primary services which are used to provide ACP127 features:

- ACP127 Service (mandatory). This is the main ACP127 channel which sends and receives ACP127 messages over the ACP127 circuits to other ACP127 stations.
- Corrector Channel (Optional). When an incoming SMTP or X.400 message is received, and it cannot be delivered or relayed due to an Authorization Rule (e.g. maximum size, or mandatory Security Label), some or all errors can be configured to be scheduled onto a corrector channel. A web GUI onto this Corrector channel allows the message to be corrected and resubmitted.

On Windows, use Isode Service Configuration to add the Service.

On Unix, see [Section 33.1, “Starting an MTA on UNIX”](#) for details on adding this service.

- OTAM Server (Optional). The OTAM process connects to the ACP127 process, and monitors outbound traffic. It also connects to a receiver, allowing traffic that is being transmitted to be received. By comparing the two streams of read and write data, it is possible to spot errors when transmitting.

On Windows, use Isode Service Configuration to add the Service.

On Unix, see [Section 33.1, “Starting an MTA on UNIX”](#) for details on adding this service.

---

## 19.9 Windows COM device support

M-Switch provides support to connect to Windows COM devices. This allows data to be sent directly over serial based modems, or via 3rd party systems.

### 19.9.1 Preparation to use a COM device

It is advisable to have configured a working ACP127 based M-Switch configuration before going any further, with the circuit connection using TCP. This allows verification of routing, and authentication before configuring the COM device.

The ACP127 channel process will relay information through a **serial proxy**. It may be useful to enable logging for the serial proxy during configuration. To do this, set the environment variable SERIALPROXY\_DEBUG to a debug file of your choice.

### 19.9.2 Configuring a COM device

Once the COM device drivers have been installed, the device settings should be set. For example

- Set **baud rate** to **300bps**
- Set **data bits** to **7**
- Set **Parity** to **Odd**
- Set **Stop bits** to **1**
- Set **Flow Control** to **Hardware**

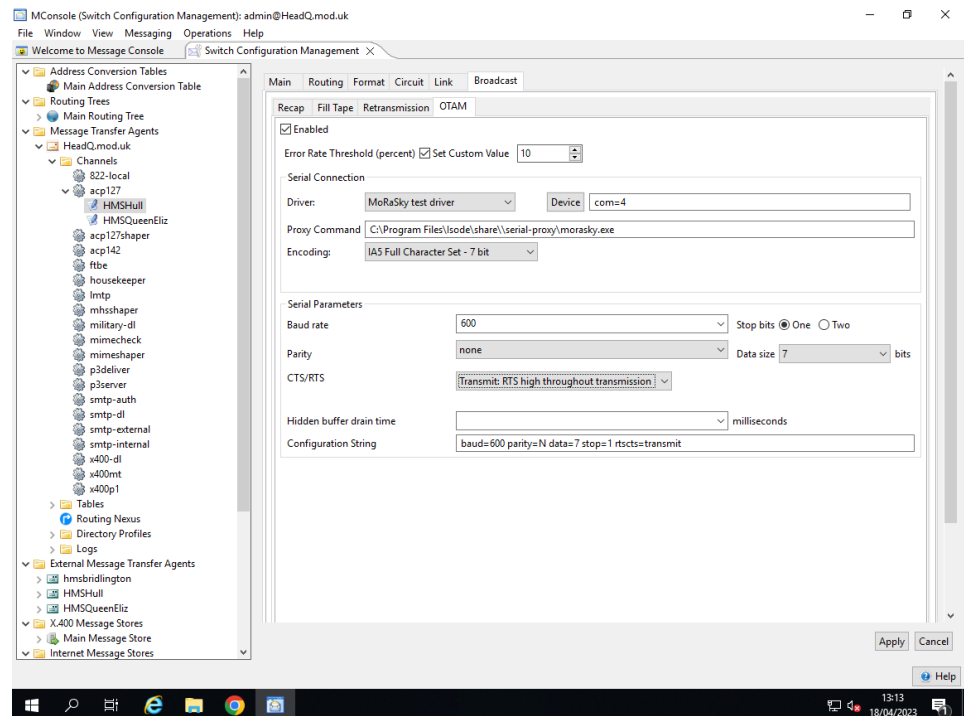
---

**Note:** In this example a data rate of 300bps is being set for the COM device. M-Switch will be told to transfer data at a higher rate to prevent buffer underflows.

---

## 19.9.3 Configuring the ACP127 circuit to use the COM device

Figure 19.36. ACP127 OTAM View Using the COM device



Select the circuit you wish to configure within the **Switch Configuration Management** view of MConsole.

Select the **Circuit** tab.

- Select **Serial** as the connection type.
- Select **Serial proxy** as the driver.
- Set **device** to the COM device. E.G **dev=com4**.
- Set the **baud rate** to be higher than the configured rate of the device. So in this example set it to **600**
- Set **Stop bits** to **1**.
- Set **Parity** to **Odd**.
- Set **Data Size** to **7 bits**.
- Set **CTS/RTS** to **transmit**.

Once configured you should restart the ACP127 process.

# Chapter 20 Summary View

The Summary View provides an easy way for an operator to check the status of an MTA or group of MTAs.

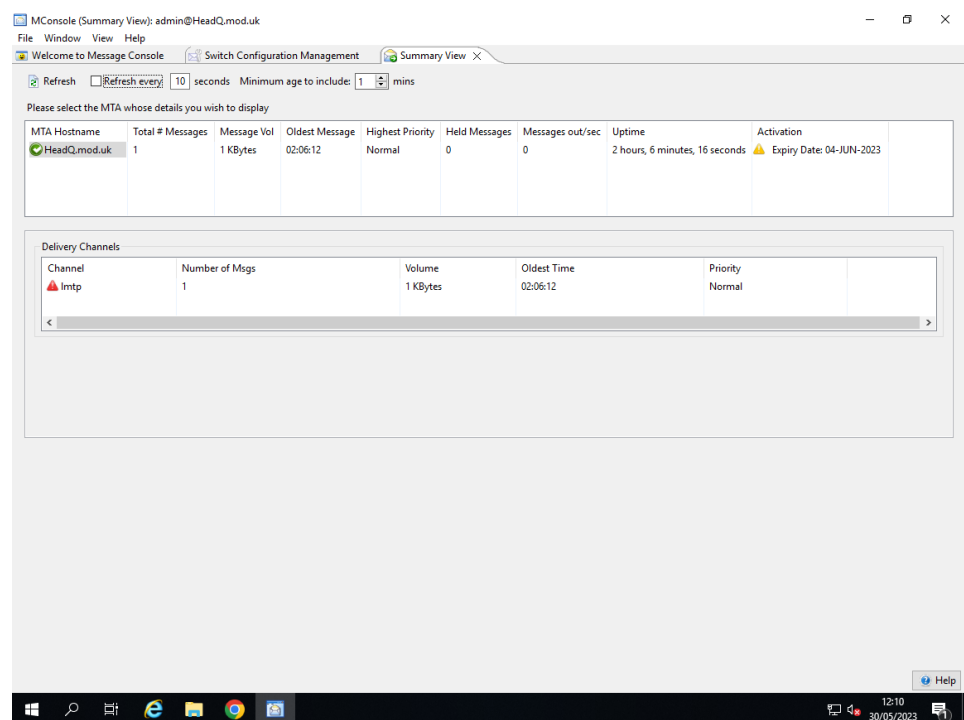
## 20.1 The Summary View

The Summary View is used to check the health of an M-Switch. It does this by querying configured M-Switches, collecting a list of messages, and displaying channels which have messages over a certain time limit.

This allows an operator to quickly see if a channel can't process a message, or a backlog is being created.

Refer to the [M-Switch Operator's Guide](#) for details of the MConsole Summary View.

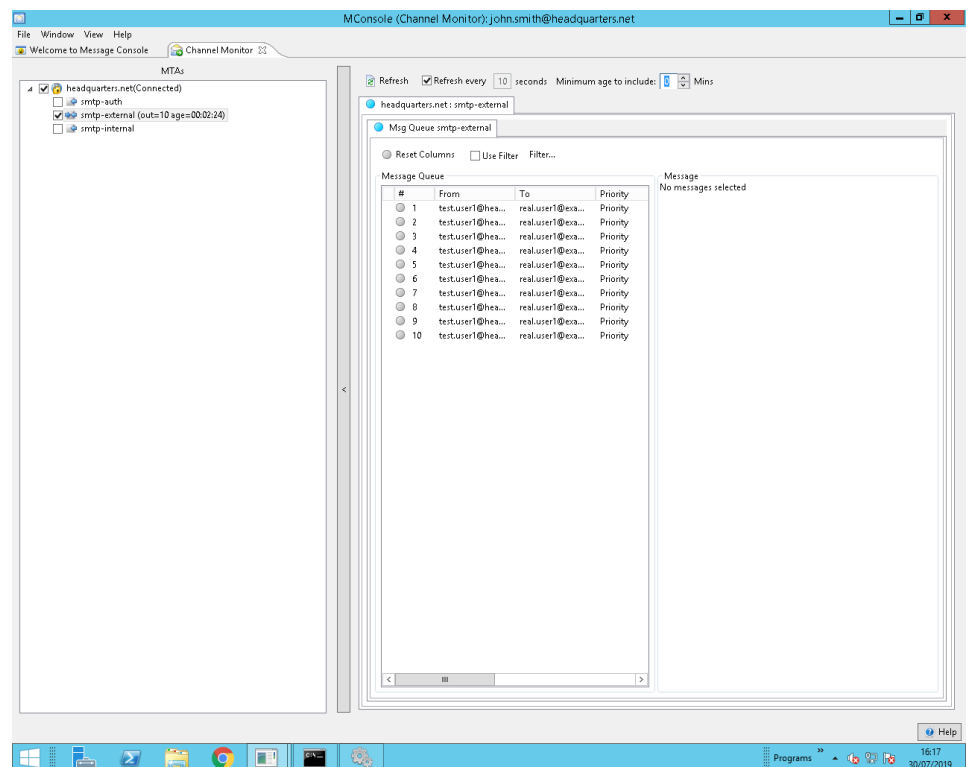
**Figure 20.1. Summary View**



# Chapter 21 Channel Monitor View

The Channel Monitor View provides a simple way for operators to monitor specific MTA channels.

## 21.1 The Channel Monitor View



When opened the left hand side will create a tree structure representing different MTAs.

Channels are shown below those MTAs, and so are peer connections.

---

**Note:** Some channels such as ACP 142 and ACP 127 have their own dedicated views, and so won't be shown as selectable channels here.

---

Next to the channel name is the number of messages currently being sent out of the channel, and the age of the oldest message that is still being processed.

Selecting a channel will make the **Channel Monitoring View** start to monitor that channel.

While monitoring, a tab on the right hand side is shown, which shows all messages currently being processed by that channel.

### 21.1 Toolbar options

The toolbar provides an operator with the ability to

#### Refresh

Update the **Channel Monitoring View** list of messages.



**Refresh every..**

Update the **Channel Monitoring View** list of messages automatically after the given time.

**Minimum age to include**

Only messages that have been in the message queue longer than this value, will be shown.

**Include peers**

Some channels have special peer connections to other MTAs. Selecting this option, will include messages being transferred on specific peers.

## 21.2

### Channel Monitoring View Tab

The **Channel Monitoring View** tab consists of a table showing the status of messages being processed by the selected channel or peer. The messages shown can be filtered to reduce the number of messages shown.

The tab also allows an operator to view messages that are displayed, and perform certain actions on those messages.

These options are:

**Hold..**

Stop the message, by adding a delay to processing it.

**Let Pass**

Remove existing delays, and let a delayed message pass through.

**Delete...**

Delete the message. Can only be performed if the message is no longer being transferred.

**Abort**

Abort transferring the message. This as an option makes sense for channels / protocols with long transfer times.

# Chapter 22 Corrector Channel

This chapter describes how the Corrector channel can be set up and configured.

---

## 22.1 Corrector Channel

When an incoming SMTP or X.400 message is received, and it cannot be delivered or relayed due to an Authorization Rule (e.g. maximum size exceeded, mandatory Security Label missing), the message can be scheduled onto a Corrector channel. A web GUI which communicates with this Corrector channel then allows the message to be corrected and resubmitted (or non-delivered).

The Corrector channel can be used in any environment where manual message inspection and correction/modification is required.

Refer to the [M-Switch Operator's Guide](#) for details of the Corrector Web GUI.

### 22.1.1 Setting up the Corrector Channel

The steps needed to set up the Corrector channel are:

- Add the channel
  - On the **Switch Configuration Management** view, right click on the **Channels** folder under the **Message Transfer Agents** folder and select **New channel**.
  - Select "Corrector" as the channel type
  - Set the channel name: the default of "corrector" will normally not need to be changed.
  - Click on **Finish** to create the channel.
- Add an Authorization Rule to cause messages to be non-delivered.

The M-Switch rule-based mechanisms used by the MTA to permit or block messages need to be configured to cause messages with certain characteristics to be blocked.

Examples would be messages which are too large or do not include a Security Information Code (SIC). These errors, together with unroutable or invalid recipient addresses would normally result in non-delivery.

If a Corrector channel is present, these messages will be scheduled onto the Corrector channel instead of being non-delivered.

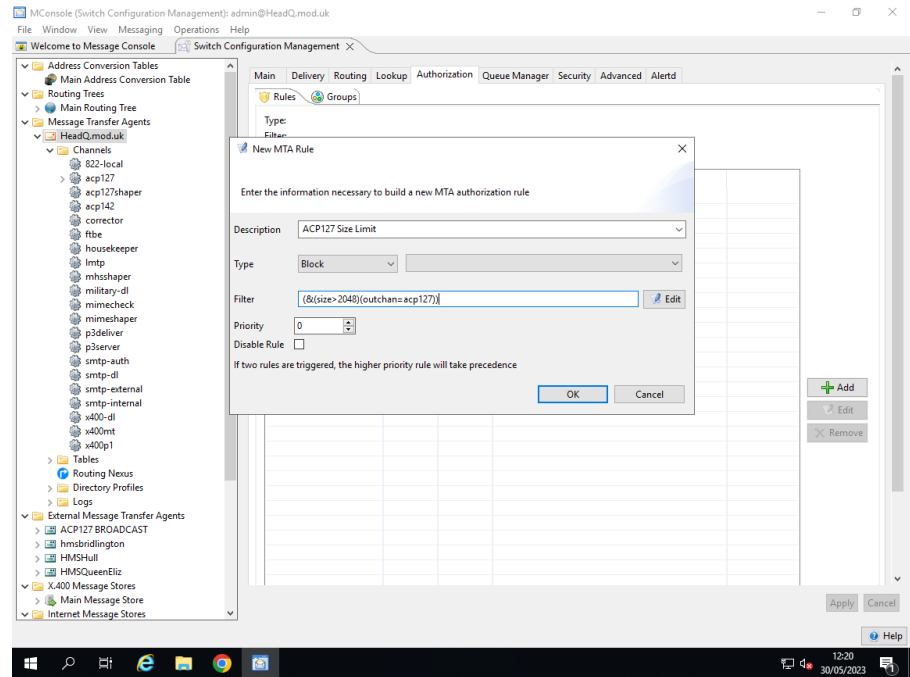
Configuration of the Corrector channel's interpretation of authorization rules and their presentation in the Web Browser are configured in *(SHAREDIR)/switch/corrector-data.txt*.

This file contains details of the way the rule and corrector channel work together.

Some examples of Authorization Rules are:

- To configure a size limit on the ACP127 channel, set up an authorization rule looking like:

Figure 22.1. Authorization Rule



the string "size limit" *must* appear in the rule description as it matches the value in the *(SHAREDIR)/switch/corrector-data.txt*.

The text used for the filter in the example above is: *(&(size>2048)(outchan=acp127))*

```
# Matching authorization error where the rule name contains \
'size limit'
status=5.401.17 regex="size limit" type=attach-error \
text="X.400 Message too large: policy violation"
```

- To configure a mandatory SIC on the ACP127 channel, set up an authorization rule like this:

```
rule "Block msgs with NO SICs"
block
" (&(numsics=0)(outchan=acp127)) "
```

the string "NO SICs" *must* appear in the rule description as it matches the value in the *(SHAREDIR)/switch/corrector-data.txt*.

```
# Matching authorization error where the rule name contains \
'SMTP Message No SICs'
status=5.7.1 regex="no SICs" type=sic-error \
text="SMTP Message has no SICs"
```

- To configure a mandatory Security Label on the ACP127 channel, set up an authorization rule like:

```
rule "Block msgs with Missing Security Label"
block
"(&(classification=0)(outchan=acp127))"
```

the string "Missing Security Label" *must* appear in the rule description as it matches the value in the (*SHAREDIR*)/switch/corrector-data.txt.

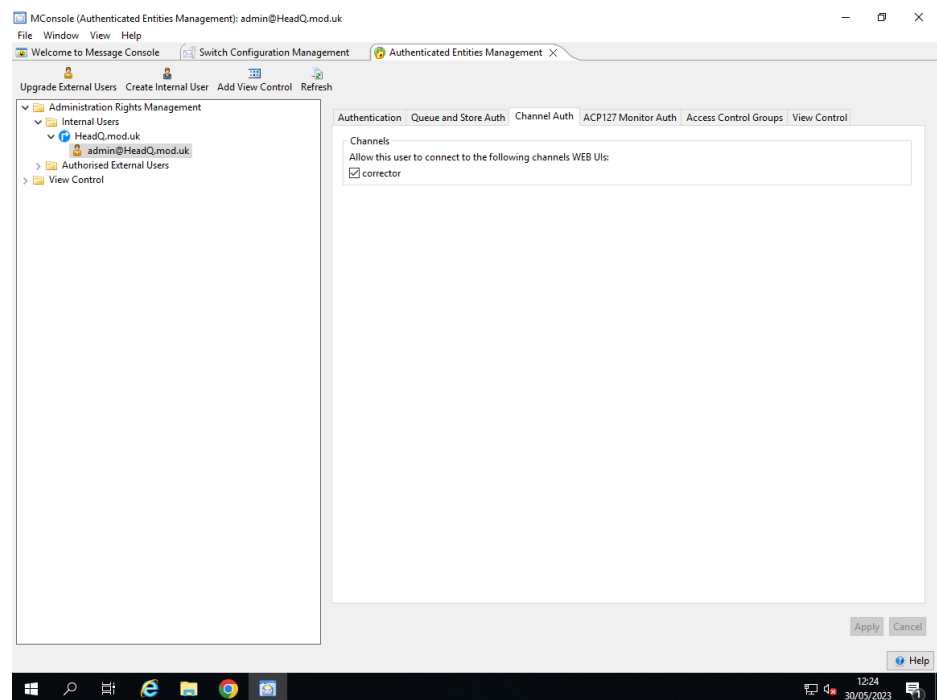
```
# Matching authorization error where the rule name contains \
'Missing Security Label'
status=5.7.1 regex="Missing Security Label" type=label-error \
text="SMTP Message has no Security Label"
```

See [Section 38.1, “Authorization”](#) for a full description of this feature.

- The Corrector client web GUI uses SASL to authenticate to the Corrector channel. To enable this, you need to include the Corrector channel in the "Local channels with Access Rights" list held in the Authenticated Entity entry whose SASL ID will be used. This is shown in [Figure 22.2, “Add Corrector Channel to Authenticated Entity”](#).

You can view the authenticated entities by selecting **View** → **Configuration** → **Authenticated Entities Management** and selecting the user you wish to allow to log into the Corrector channel in your web browser.

**Figure 22.2. Add Corrector Channel to Authenticated Entity**



- To connect to the Corrector channel, point your browser at <http://localhost:18200>.

Messages which result in an Error should now be queued on the Corrector channel instead of being non delivered, and will show up in your browser.

# Chapter 23 CFTP

This chapter describes how CFTP can be set up and configured.

---

## 23.1 The CFTP channel

The CFTP module of M-Switch normally consists of one primary channel that handles all the CFTP traffic over STANAG 5066 based networks. Messages arrive into M-Switch over S5066 through the channel, where they are either gatewayed to Internet(SMTP); or on some occasions relayed on through other protocols. In all but very special circumstances, there will only be one CFTP channel, but there will often be several peer connections.

An CFTP channel consists of the main channel definition, and a number of peer connections. Each peer connection points to an External MTA configured as an CFTP remote node.

---

## 23.2 Routing using the domain

This is achieved using the standard internet routing using the Routing Tree. The Routing Tree entry for the domain is configured to send to the peer MTA (i.e. External MTA CFTP Workstation).

---

## 23.3 Using MConsole to configure CFTP

When creating a configuration to use the CFTP gateway of M-Switch, it is important to note that CFTP only works with the internet protocol. Therefore if other protocols are wanted, then the MIXER option needs to be selected.

To set up a CFTP gateway and allow routing to and from it, the following steps are required.

- Create and configure a CFTP channel.
- Create one or more external CFTP MTA(s).
- Create peer connections for each external MTA.
- Add a Routing Tree entry for each MTA.

You will also need to configure:

- S5066 Servers
- ACP127 S5066 Access Points

### 23.3.1 Creating a CFTP channel

If you do not have a CFTP channel, or wish to create a second channel to gateway messages, on the **Switch Configuration Management** tab, expand the **Message Transfer Agents**

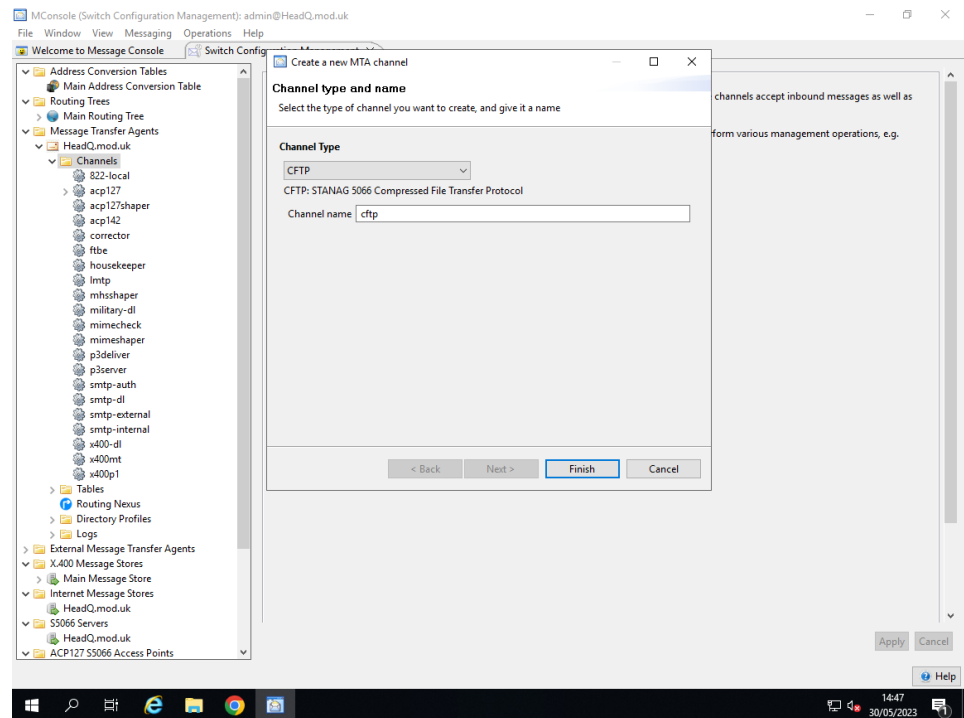
node, and the MTA below that. Then with the **Channels** node selected, either right click and choose **New Channel** or choose **Operations → New Channel**.

Then provide the following answers when prompted by the wizard:

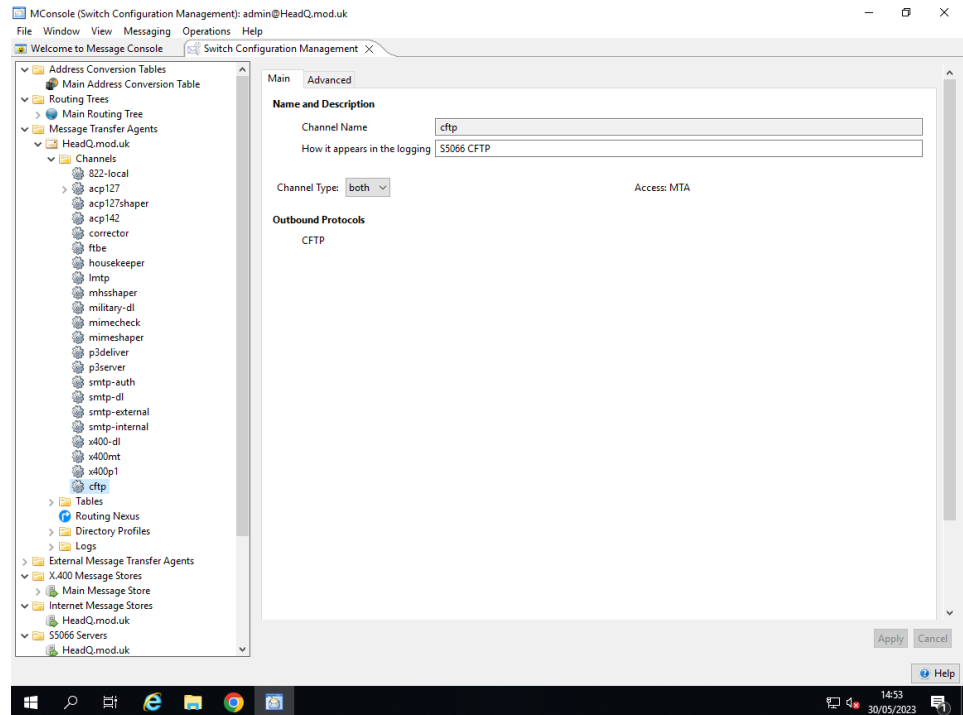
- **Channel Type** select **CFTP**
- **Channel Name** name it `cftp` or similar

as shown in [Figure 23.1, “CFTP channel creation”](#).

**Figure 23.1. CFTP channel creation**



After creating the `cftp` channel, you can edit the channel to change some of the text if required. **CFTP** page [Figure 23.2, “CFTP channel”](#) as required.

**Figure 23.2. CFTP channel**

There are no CFTP specific components of the channel entry currently.

## 23.3.2 Creating an external CFTP MTA Connection

There are five steps to this procedure, or fewer if re-using an existing connection.

- Create an external MTA associated with the remote connection.
- Create or configure a peer connection for the CFTP Channel with the remote connection.
- Create a domain in the Routing Tree entry to route addresses into the CFTP gateway.

---

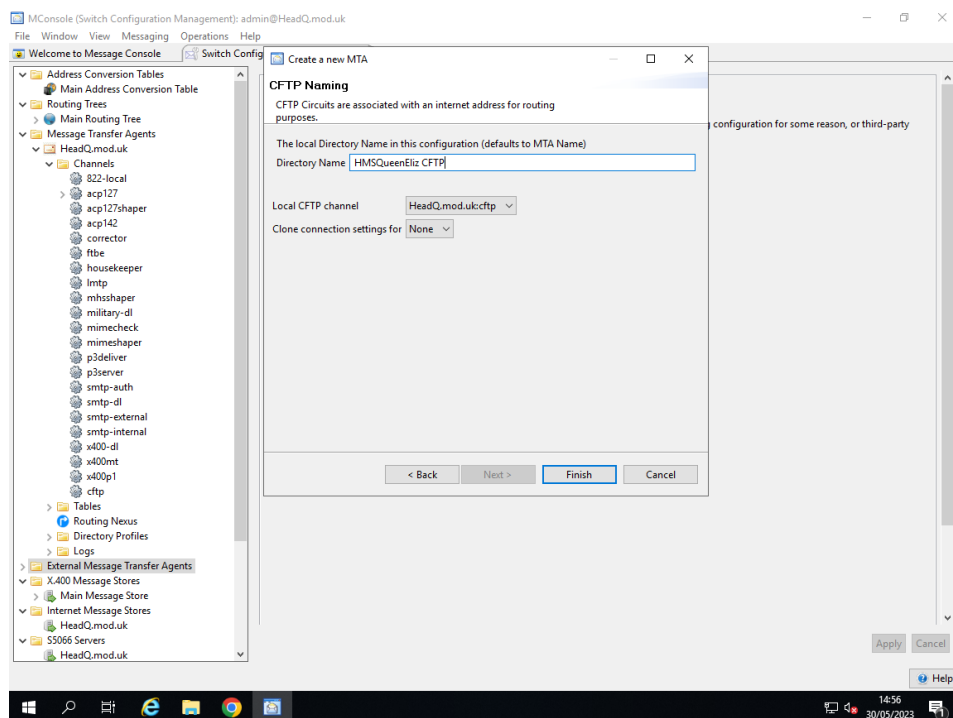
**Note:** All addresses to this domain will be routed into CFTP gateway. It is not currently possible to route different addresses from the same domain differently.

---

### 23.3.2.1 Create a remote external CFTP MTA

To create a remote MTA that represents the CFTP endpoint, first create an external MTA. This is done with MConsole on the **Switch Configuration Management** tab. Expand the **Message Transfer Agents** node, and then right click on **External Message Transfer Agents** and choose **New External MTA** from the menu. Alternatively use **Operations** → **New External MTA**.

When the wizard popup dialog appears, select **External CFTP Connection** and click **Next**. In the subsequent dialog, name the external connection with a unique directory name, usually a name relevant to the remote connection - e.g. *Ship1 Receive*. Click **Finish** to complete the creation as shown in [Figure 23.3, “CFTP create external MTA”](#).

**Figure 23.3. CFTP create external MTA**

The new external MTA should now be present in the list of MTAs below the **External Message Transfer Agents** node. If the cftp channel is not named **cftp** then you will need to edit the transfer weightings by right clicking the new node and select **Modify Transfer Channel Weight** to update the channel.

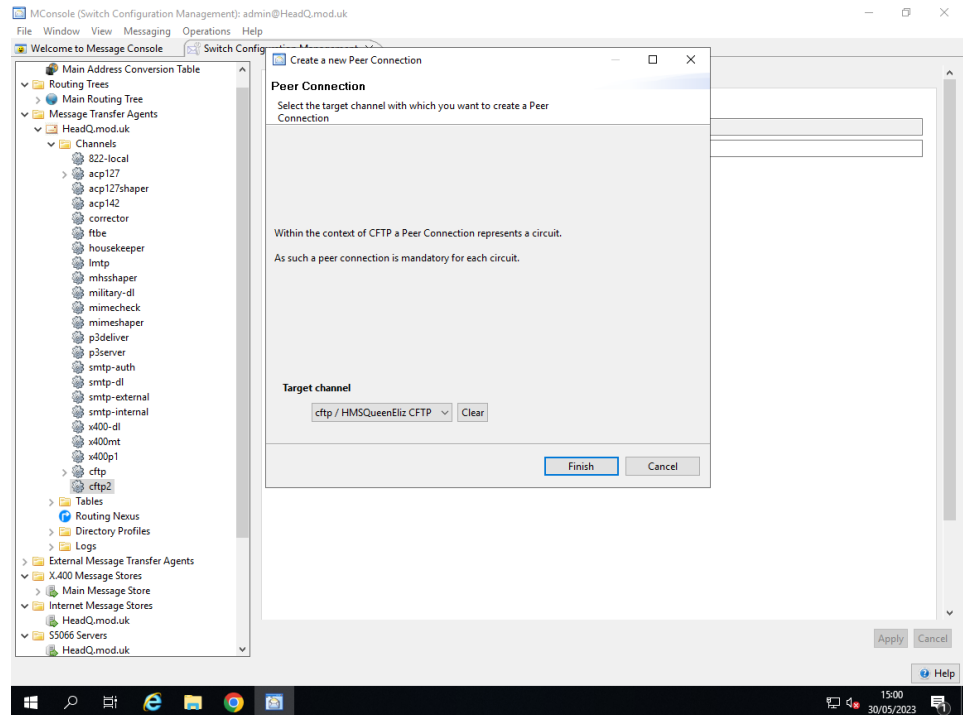
### 23.3.2.2 Set up a peer connection

A peer connection to your external CFTP node will be created for you. A peer connection encompasses the details needed to transfer messages between the two entities.

Creation of a new external CFTP MTA will automatically cause a peer connection to be created for the chosen Local CFTP channel. This peer connection will then need to be suitably edited as described below in [Section 23.3.2.2.1, “CFTP Peer Connection: Main Tab”](#).

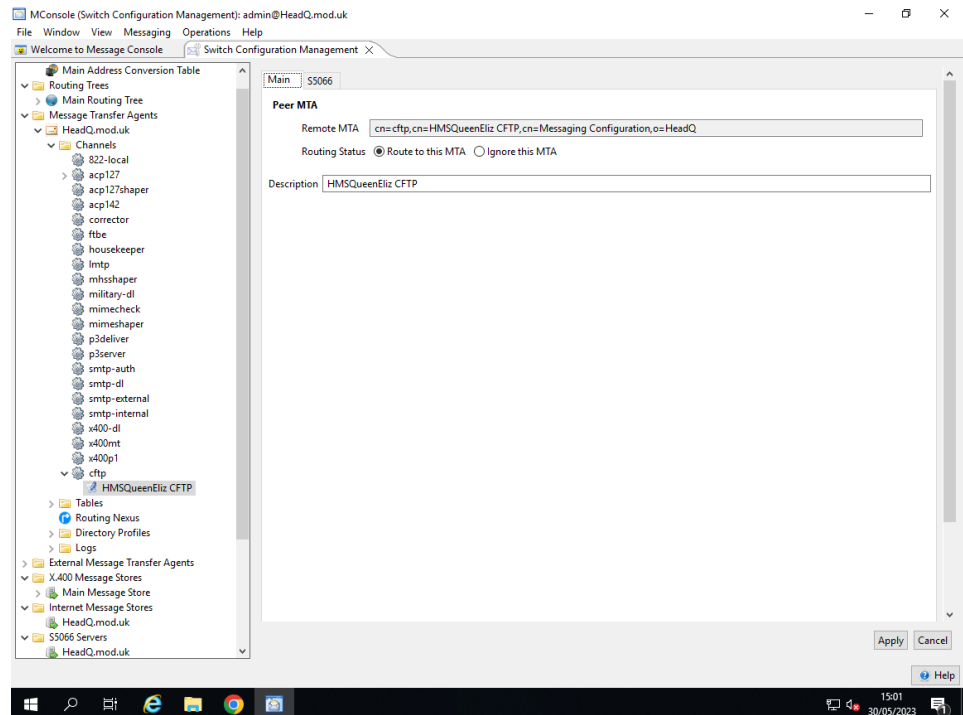
If you do need to create a new peer connection, navigate to the CFTP channel, which appears below **Message Transfer Agents** and then **Channels**. Right click on the cftp channel, and select **Add Peer Connection** from the menu as shown in [Figure 23.4, “CFTP creation of a Peer Connection to the external MTA”](#).



**Figure 23.4. CFTP creation of a Peer Connection to the external MTA**

After the Peer Connection is created, the entry should be edited to set all the parameters required for the connection to succeed. Specifically the S5066 parameters. An example is shown in [Figure 23.5, “CFTP editing the Main Tab of a peer connection”](#) for the common parameters, and [Figure 23.6, “CFTP editing the S5066 Tab of a peer connection”](#) for the STANAG 5066.

### 23.3.2.1 CFTP Peer Connection: Main Tab

**Figure 23.5. CFTP editing the Main Tab of a peer connection**

The parameters are as follows:

## Remote MTA

DN of channel of External MTA.

## Routing Status

If **Routing Status** is set to **Route to this MTA** then this peer MTA will be taken into consideration for routing.

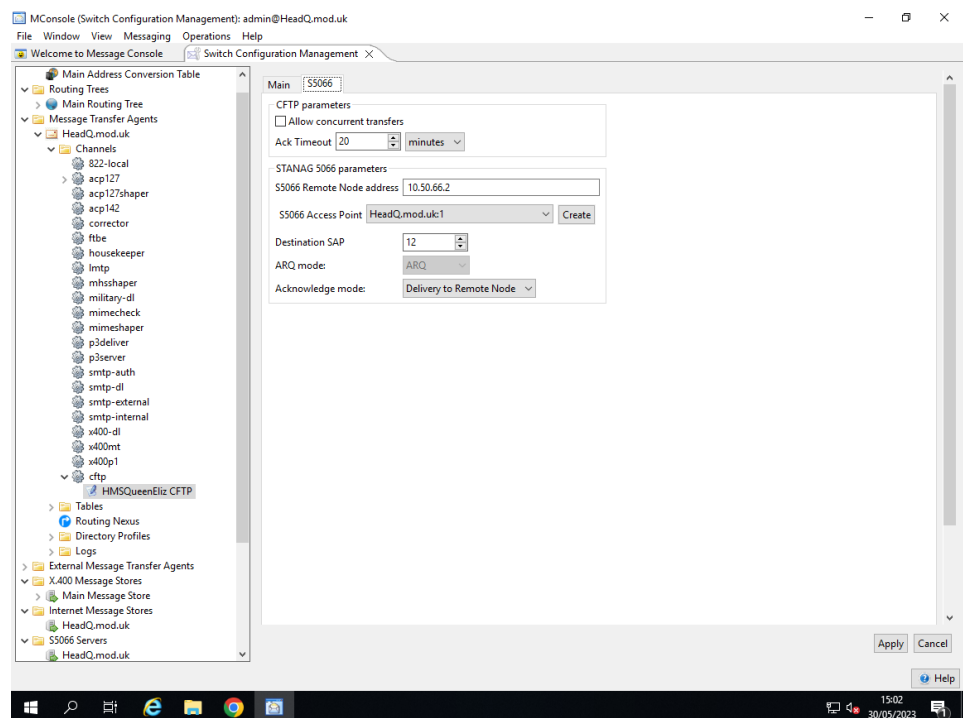
If **Routing Status** is set to **Ignore this MTA** then this peer MTA will be ignored for routing.

## Description

A text description of this channel. This is only used for information to Administrators.

### 23.3.2.2 CFTP Peer Connection: S5066 Tab

**Figure 23.6. CFTP editing the S5066 Tab of a peer connection**



The parameters are as follows:

### Allow concurrent transfers

If this box is ticked, then multiple transfers will be sent one after another with unique IDs. If unticked a message will be sent and a ACK waited for before a subsequent message is sent.

### S5066 Remote Node address

This is the 5066 address of the remote S5066 entity.

### S5066 Access Point

This is a Stanag S5066 Access Point which specifies access to a S5066 protocol server that can be selected from the list in the dropdown menu. These servers can be created, managed and remove through the S5066 Access Points as described in [Figure 19.30, “Editing a STANAG 5066 Access Point”](#).

### Destination SAP

The destination SAP used with the S5066 protocol to connect to remotely. It should normally be 12 (CFTP) unless special circumstances require it otherwise.

### ARQ Mode

The style of ARQ mode in use. It can be one of:

- **None**- No confirmation is requested .

- **ARQ**- Request reliable delivery .
- **Non-ARQ**- Request unreliable delivery (suitable for broadcast circuits).

#### Acknowledge mode

The style of acknowledgement to wait for, from one of:

- **None**- No acknowledgement is expected.
- **Local received**- Request acknowledgement from the local server when the data unit has been received.
- **Remote received**- Request confirmation from the remote server when the data has been received.

## 23.4 Setting up Routing

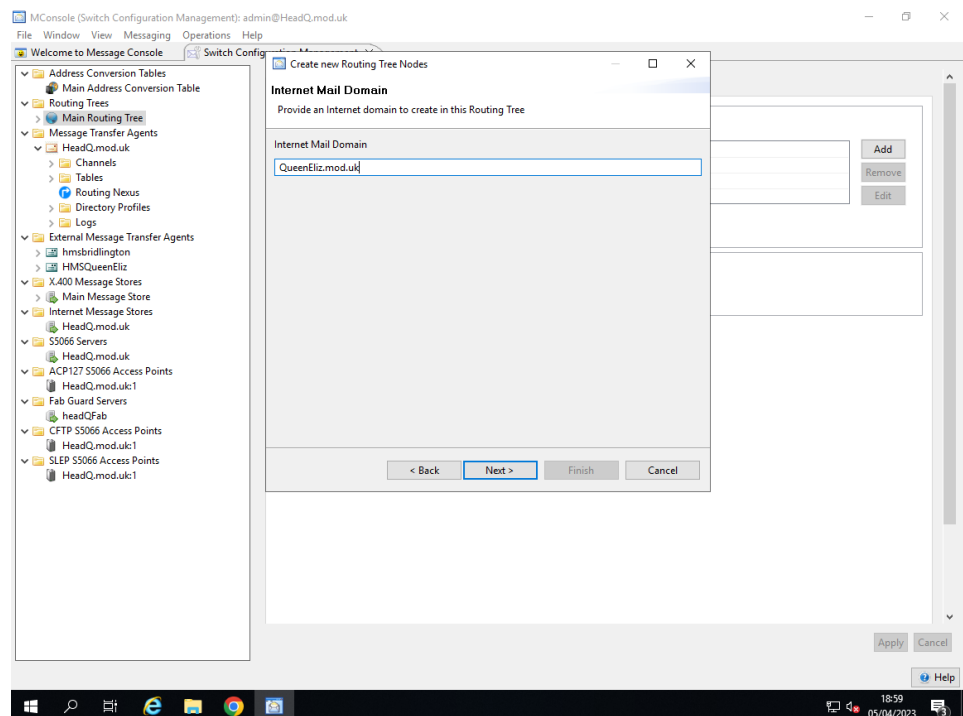
The final step of configuring an CFTP Gateway is to set up the routing.

### 23.4.1 Setting up Routing with Gateway Users

#### 23.4.1.1 Add a node to the Routing Tree

To add routing to the new external MTA, create a node representing the Internet address by navigating to the **Main Routing Tree** node under the **Routing Trees** entry in the **Switch Configuration Management** view. Right click and choose **Add nodes** and select the **Create Routing Tree entries representing an Internet domain**. Choose **Next** and enter the Internet domain of the entry, e.g. `queeneliz.mod.uk` such as shown in [Figure 23.7](#), “CFTP creation of Routing Tree node.”. Select **Finish**.

**Figure 23.7. CFTP creation of Routing Tree node.**



Once this is created, select the entry, and click on the **Add** button to connect to the external MTA. Make sure the correct MTA is selected in the drop down menu. The default weight is normally acceptable unless complex setup is required.

---

## 23.5 CFTP Services

There is one service which is used to provide CFTP features:

- CFTP Service (mandatory). This is the main CFTP channel which sends and receives CFTP messages over the CFTP circuits to other CFTP stations.

# Chapter 24 FTBE

The configuration of FTBE Channel, Server and Users is held in the Directory, and is managed using the MConsole **X.400 Mailbox Management** view for X.400 users, and **Switch Configuration** view for the Channel.

Cobalt is used to manage the Internet FTBE Users.

---

## 24.1 Managing FTBE

The File Transfer by Email Channel allows arbitrary files to be transferred between two systems which are running M-Switch.

Messages can be sent using Internet or X.400 addresses and content. These are both described in this chapter.

### 24.1.1 Configuration Overview

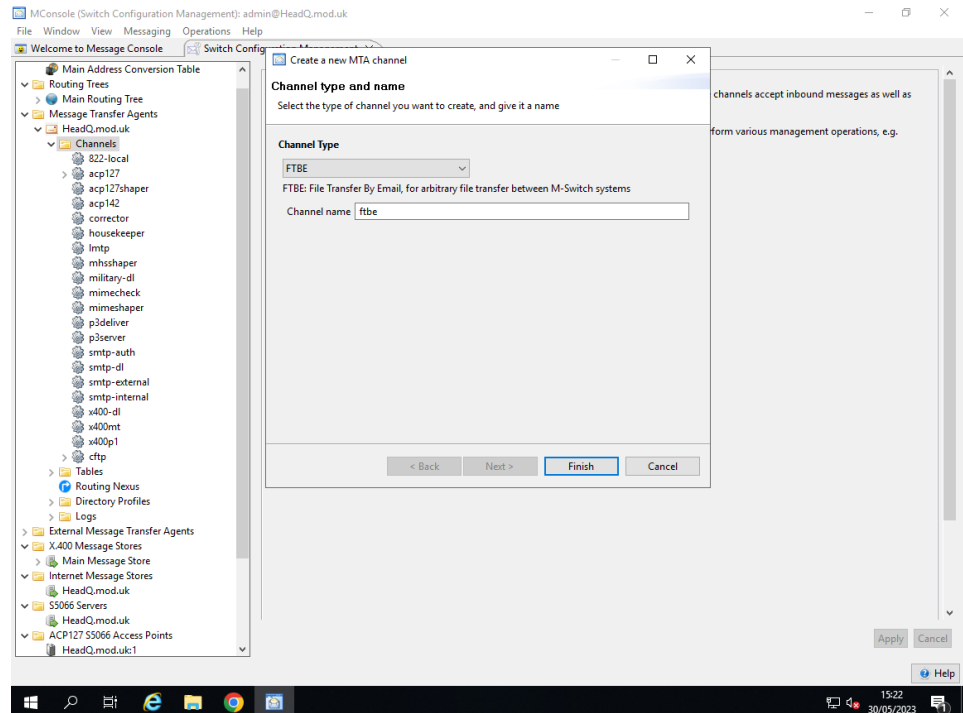
In order to use this feature of M-Switch, you need to do the following:

- Create an FTBE channel, which comes in two parts
  - FTBE Client, which allows messages to be delivered into the filestore by M-Switch.
  - FTBE Server, which allows messages to be submitted into M-Switch from the FTBE managed filestore.
- Configure the FTBE server
- Create an FTBE User which is the mail address
  - of the FTBE mailbox into which messages are to be delivered into the filestore by the FTBE client
  - used as the originator for messages which are to be submitted from the filestore into M-Switch by the FTBE Server
- Create an FTBE Peer which is the mail address
  - of a sender of messages to be delivered into to the FTBE User by the FTBE client
  - of a recipient of messages which are to be submitted from the filestore into M-Switch by the FTBE Server

#### 24.1.1.1 FTBE Channel Creation

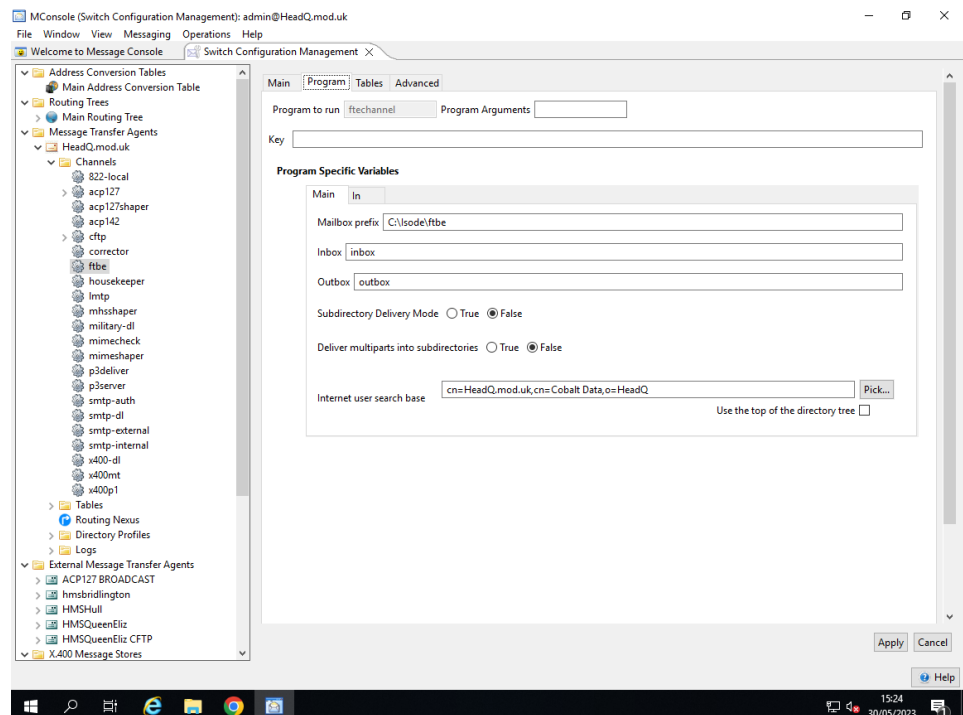
To create a new FTBE Channel using MConsole, start the Channel creation wizard by selecting **New Channel** in the **Switch Configuration Management** View.

Select **FTBE** as the channel type. Select a suitable FTBE channel name, e.g. **ftbe**.

**Figure 24.1. Creating an FTBE Channel**

After creating the FTBE channel, the newly created channel appears as in the following figure. Note that in order to work, the channel needs to have the Internet User Search Base configured correctly. See [Internet User Search Base](#)

The following figure shows an FTBE channel after creation. The **Program** tab is selected, and **Main** tab of the Program Specific Variables.

**Figure 24.2. FTBE Channel Program Tab**

Other values on the FTBE channel **Main** tab are configured as follows:

## Mailbox Prefix

### Inbox

Name of the FTBE User's inbox

### Outbox

Name of the FTBE User's Outbox

### Subdirectory Delivery Mode

If true each message is delivered into a separate folder, with the headers and each Body Part in a separate file.

### Deliver multipart into subdirectories

If true, each MIME/Multipart will be delivered into a separate subdirectory. If false they will be delivered at the same level.

This only affects delivery of Internet messages.

### Internet User Search Base

Configures where FTBE will look for FTBE users.

If using just Internet FTBE Users, the value should be set to wherever Cobalt has been configured to create the users i.e. *cn=FTBE,cn=<domain>,cn=Cobalt Data,cn=Cobalt Data* .

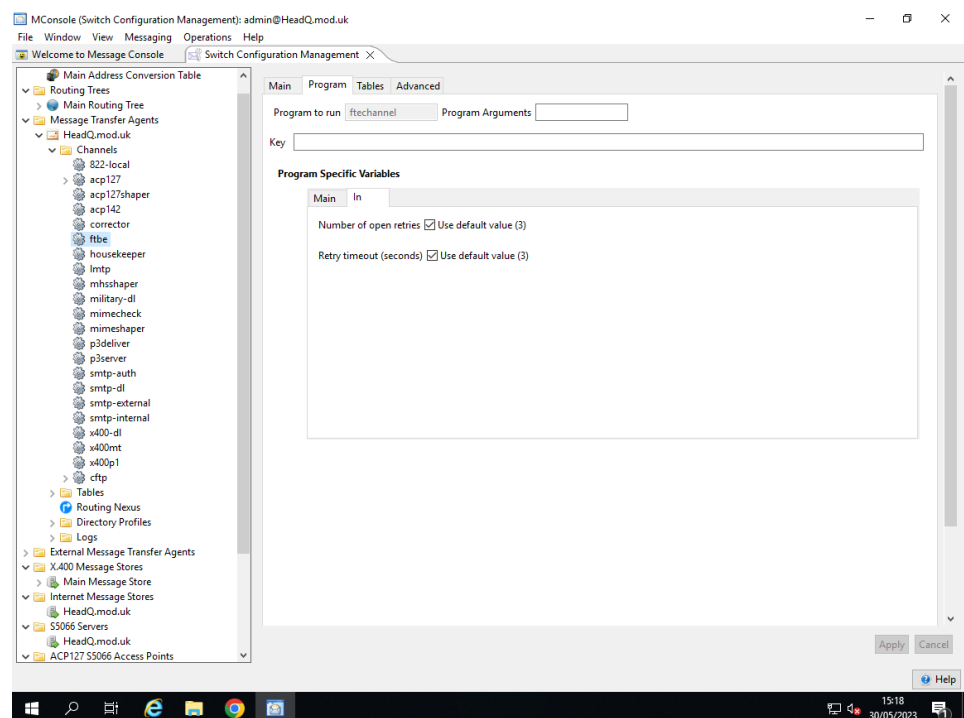
If using just X.400 FTBE Users, the value should be set to *cn=White Pages,o=messaging*.

If using Internet and X.400 FTBE Users, the value should be set to *o=messaging*.

The following figure shows an FTBE channel after creation. The **Program** tab is selected, and **In** tab of the Program Specific Variables.

Values on the FTBE channel **In** tab are configured as follows:

**Figure 24.3. FTBE Channel In Tab**



### Number of open retries

On Windows, the ChangeNotification which alerts the *ftbeserver* process to new files will fire when a new file is created in the input directory, but before the file has

been closed by the process which is creating or copying it. This causes the `ftbserver` to attempt to open the file and discover that it is locked.

When `num_open_retries` is set to non-zero, the `ftbserver` will wait for a short interval and then attempt to open the file again, up to the configured limit on the number of attempts. Defaults to 3 ( 0 implies no retries). Sets the value of the `num_open_retries` channel-specific variable.

#### **Retry timeout**

The time (in seconds) to wait before retrying after a failed file open. Defaults to 3 seconds. Sets the value of the `retry_timeout` channel-specific variable.

You should also check the **Content Out** value set in the **Advanced** Tab of the channel, and make sure that suitable content type are configured, such as **822**, **822-8**, **822-b**. By default only **P2** and **P22** are set.

## **24.1.2 Configuring FTBE Users**

### **24.1.2.1 Configuring FTBE Internet Users**

This is carried out using Cobalt. Consult the **Cobalt Administration Manual** to do that.

### **24.1.2.2 Configuring FTBE X.400 Users**

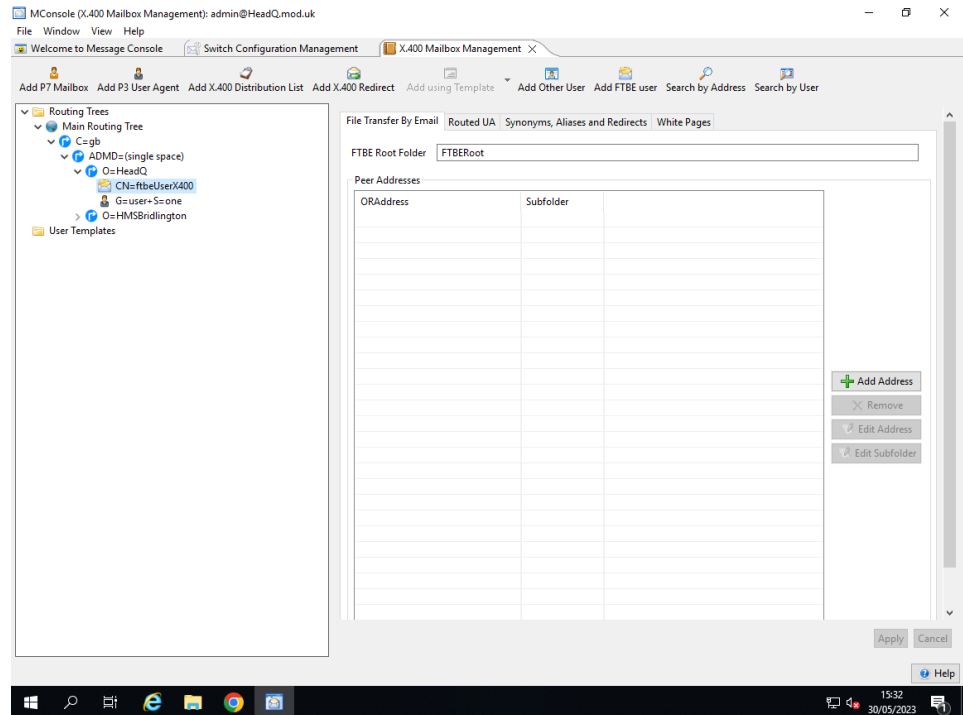
#### **24.1.2.2.1 FTBE X.400 User Creation**

To create a new X.400 FTBE User using MConsole start the FTBE User creation wizard by selecting **Add FTBE User** in the **X.400 Mailbox Management** View.

Step through the creation wizard as follows:

1. Select a suitable address form, such as Personal Name
2. Enter the O/R address attributes for the FTBE User
3. Add a White Pages Entry
4. Select the **Supporting MTA** and **FTBE channel** to be used
5. Set the value of the **FTBE Root Folder** - any appropriately meaningful value is suitable. This will appear as the directory folder in which messages are sent/received.
6. Click on **Finish**



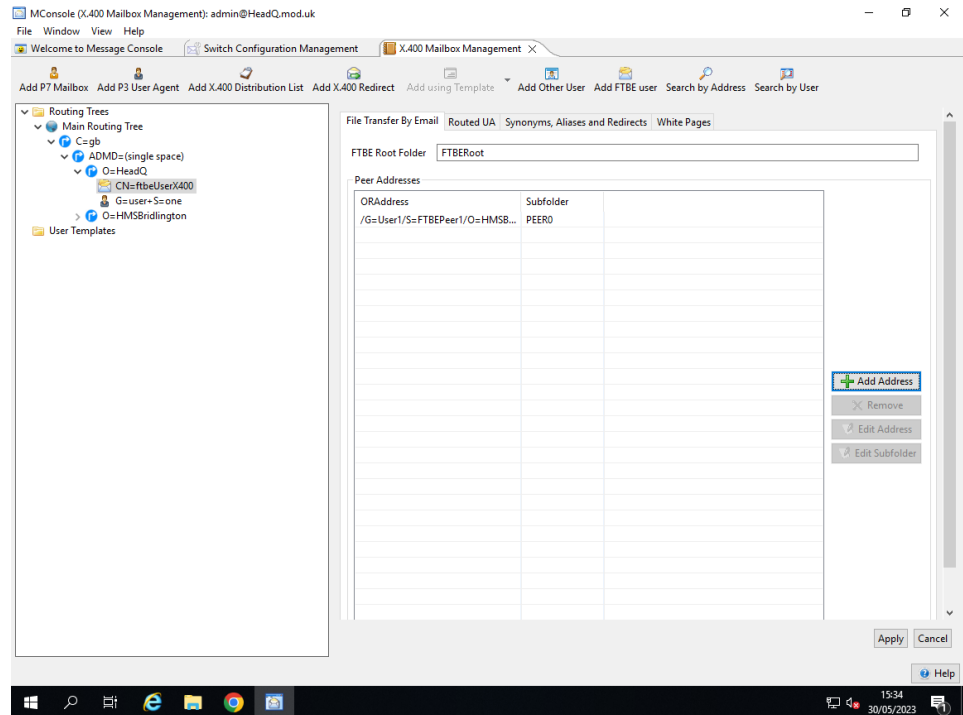
**Figure 24.4. FTBE X.400 User After Creation**

#### 24.1.2.2.2 FTBE X.400 Peer Creation

To create a new FTBE X.400 Peer using MConsole start the FTBE Peer creation wizard by selecting the FTBE User under which you wish to create a Peer and click on **Add Address** in the **X.400 Mailbox Management** View.

This is shown in [Figure 24.5, “FTBE X.400 Peer After Creation”](#).

Enter a suitable **O/R Address** as the O/R address of the FTBE Peer. Click on OK to add the Peer and then click on **Apply**. Note that the Subfolder has had a suitable value of **PEER0** added (although any value will do). You can change this by selecting the Peer and clicking on **Edit Subfolder**.

**Figure 24.5. FTBE X.400 Peer After Creation**

### 24.1.3 FTBE Server Setup

To configure M-Switch so that the FTBE server is started when M-Switch is started, see [Section 33.1.3.2, “Starting FTBE”](#)

### 24.1.4 Initialising FTBE

After all the configuration steps described above have been carried, the FTBE Server will initialise the directory structure when first started. NB if already running, M-Switch will need to be restarted.

#### 24.1.4.1 Internet FTBE Filestore

The configuration examples above will result in the following directory structure being created for internet FTBE users when M-Switch is started. It also shows a message which has been delivered by the FTBE channel into the filestore, and also message which has been submitted into the FTBE server.

The folders are created when the FTBE Server first starts up.

In this example **Subdirectory Delivery Mode** was set to the default value of false.

```
/var/isode/ftbe:
/var/isode/FTBE:
ftbe.peer1

/var/isode/FTBE/ftbe.peer1:
inbox outbox

/var/isode/FTBE/ftbe.peer1/inbox:
unknown

/var/isode/FTBE/ftbe.peer1/outbox:
testmessage.ENVID testmessage.SUBMITTED
```

Note that a message delivered into the inbox can have the filename *unknown*. This is because the mimetype did not include a name (see below for a more complete description).

#### 24.1.4.2 X.400 FTBE Filestore

The configuration examples above will result in the following directory structure being created for X.400 FTBE users when M-Switch is started. It also shows a message which has been delivered by the FTBE channel into the filestore, and also message which has been submitted into the FTBE server.

Like Internet Users the folders are created when the FTBE Server first starts up.

```
/var/isode/switch/FTBEX400:
  /var/isode/switch/FTBEX400:
  PEER0

/var/isode/switch/FTBEX400/PEER0:
inbox  outbox

/var/isode/switch/FTBEX400/PEER0/inbox:
  body-part-name.txt  'body-part-name.txt;1'

/var/isode/switch/FTBEX400/PEER0/outbox:
testx400.FAILED testx400.MTSID testx400.RECIP.1.SUCCESS
testx400.SUBMITTED
```

Note also that a message placed in the outbox with the name *testx400.SUBMIT* has been processed and the file *testx400.ENVID* has been added, and to aid correlation, *testx400.MTSID* has been created.

the original *testx400.SUBMIT* file has been renamed after processing to *testx400.SUBMITTED*.

Note also that because the **Subdirectory Delivery Mode** is set to false, the flatter directory structure has been used.

### 24.1.5 FTBE Detail

#### 24.1.5.1 FTBE Server

The FTBE Server monitors a set of input directory trees. The root of each tree corresponds to an originator address i.e. an FTBE User. Under this directory are a set of subordinate directories which map to a collection of recipient addresses (usually just a single recipient). See [Section 24.1.4, “Initialising FTBE”](#) for examples of the way FTBE sets up and uses its filestore.

Any file which is placed in *outbox* of an FTBE User with the file extension *.SUBMIT* will be used as the sole bodypart of a message with the originator/recipient pairs implied by the directory.

Choice of Internet or X.400 message is made via the address format of the message recipient. For Internet messages, the file will be sent as a MIME application/octet-string bodypart, using base-64 encoding. For X.400 messages a File Transfer bodypart will be used. After successful submission, the file is renamed to indicate whether it has been successfully submitted or not. When successful submission has taken place, a file with the extension “*.MTSID*” is also created - this contains the MTSid or Internet messageId for the message in question. For X.400 messages, positive and negative delivery reports are requested, and an empty ‘recipient file’ is created for each recipient of the message at the point of submission, with the file extension (“*.PENDING*”). Delivery of positive or negative delivery reports corresponding to these recipients will cause the files to be renamed accordingly. So the complete sequence of events when using the channel in X.400 mode might be:

- Write file “update.ldif.SUBMIT” into `/var/isode/fte/sodium/remote-sync-1`
- Message is constructed containing content of this file, and filename “*update.ldif*” in the bodypart parameters, with originator “/cn=sodium/o=isode/p=isode/a= /c=gb/” and recipient “/cn=sodium/o=BoldonJames/a=gold400/c=gb/”.
- The message is submitted, and is allocated an MTS-id of “<infiniteca.0093801-081208.123218>/ADMD= /C=GB/”
- The original file renamed to “*update.ldif.SUBMITTED*”, and a recipient file (in the same directory) is created with name *update.ldif.RECIP.1.PENDING*.
- When a positive delivery report for recipient 1 is delivered, the recipient file is renamed to *update.ldif.RECIP.1.SUCCESS*
- Receipt of a negative DR would cause the recipient file to be renamed to *update.ldif.RECIP.1.FAILURE*

### 24.1.5.2 FTBE Client

The second FTBE component is the delivery channel. This has two modes of operation:

- The default mode follows the same general rules as the submission server: the recipient address for the message is used to locate a top-level directory, and then the message is delivered into the subdirectory which corresponds to the originator address for the message. The channel can deliver either X.400 or Internet content. When delivering X.400 content, the channel delivers any File Transfer Bodyparts in the content as individual files. When delivering Internet content, only *application/octet-string* MIME bodyparts will be processed. Any message which does not contain any deliverable bodyparts will be non-delivered.
- An alternative delivery mode, which is only supported for Internet messages, can be enabled by setting the **Deliver Multiparts into Subdirectories** to **true** as described in [Deliver multiparts into subdirectories](#).

In this mode, each message is delivered into a new subdirectory of the FTBE user's Inbox: these are simply named 1, 2, 3 etc. When operating in this mode, the channel will:

1. Create a “*headers.txt*” file containing the Internet message headers
2. Create an “*envelope.txt*” file containing selected SMTP envelope fields, including the sender address
3. Deliver all MIME bodyparts found in the message. Bodyparts whose MIME parameters include a filename will be delivered as that filename, otherwise a numeric name (e.g. “*0001.txt*”) will be generated. If a bodypart does not have a filename, but is of a known subtype, then an appropriate file extension will be used. For example, a *text/html* bodypart would be delivered as “*0002.html*”. A *message/rfc822* bodypart will always result in a subdirectory being created, within which the bodypart (including headers) will be delivered.

Two alternative methods of handling multipart bodyparts are supported: the default mode is to deliver all components of multiparts (including nested multiparts) at the same level, with the components of a multipart/alternative being assigned the same numeric name (i.e. a multipart/alternative which contained plain-text and HTML versions of the same text would be delivered as “*0001.txt*” and “*0001.html*”. If the **Deliver Multiparts into Subdirectories** is set to **true**, then the channel will create a new subdirectory for every multipart, and deliver the bodyparts making up the multipart within this. See [Deliver multiparts into subdirectories](#) how to configure this. This results in the channel-specific variable “*subdir\_on\_multipart*” being set.

When the FTBE channel is being used for delivery only, and the **Deliver Multiparts into Subdirectories** is set to **true**, then the channel will create the channel-specific variable “*allow\_any\_sender*” to “*true*” will cause the channel to deliver messages from any

sender, without any need for them to be configured via a Recipient entry as described in the preceding paragraph.

# Chapter 25 SLEP

This chapter describes how SLEP can be set up and configured.

---

## 25.1 The SLEP channel

The SLEP module of M-Switch normally consists of at least one channel that handles all the SLEP traffic over a particular STANAG 5066 based network. Messages arrive into M-Switch over S5066 through the channel, where they are either gatewayed to Internet (SMTP); or on some occasions relayed on through other protocols.

Allongside the channel definition are one or more external SLEP channels. One for each remote station.

---

## 25.2 Routing using the domain

This is achieved using the standard internet routing using the Routing Tree. The Routing Tree entry for the domain is configured to send to the external MTA.

---

## 25.3 Using MConsole to configure SLEP

When creating a configuration to use the SLEP gateway of M-Switch, it is important to note that SLEP currently only works with the internet protocol. Therefore if other protocols are wanted, then the MIXER option needs to be selected.

To set up a SLEP gateway and allow routing to and from it, the following steps are required.

- Create and configure a SLEP channel.
- Create one or more external SLEP MTA(s).
- Add a Routing Tree entry for each MTA.

You will also need to configure:

- S5066 Servers
- SLEP S5066 Access Points

### 25.3.1 Creating a SLEP channel

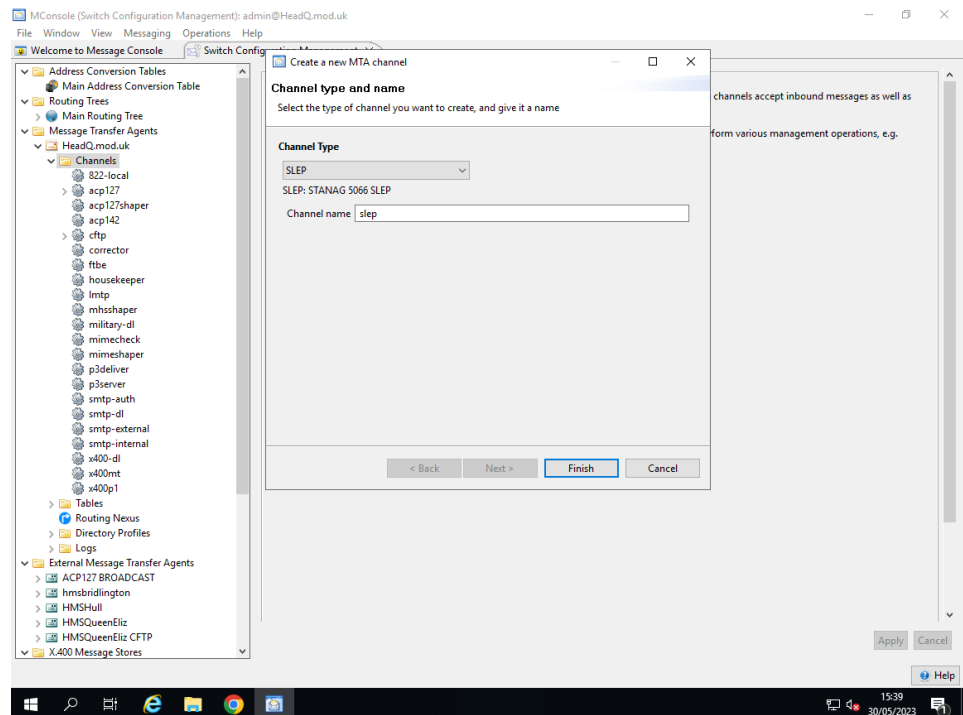
If you do not have a SLEP channel, or wish to create a second channel to gateway messages, on the **Switch Configuration Management** tab, expand the **Message Transfer Agents** node, and the MTA below that. Then with the **Channels** node selected, either right click and choose **New Channel** or choose **Operations** → **New Channel**.

Then provide the following answers when prompted by the wizard:

- **Channel Type** select **SLEP**
- **Channel Name** name it `slep` or similar

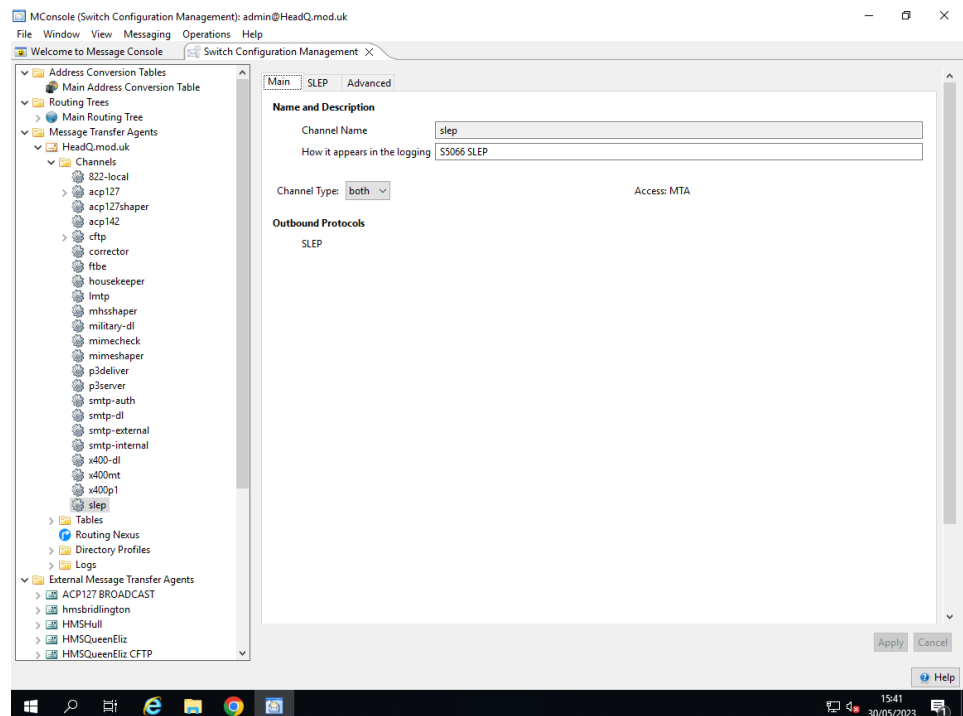
as shown in [Figure 25.1, “SLEP channel creation”](#).

**Figure 25.1. SLEP channel creation**



After creating the `slep` channel, you can edit the channel to select the S'5066 access point by selecting the **SLEP** Notebook Tab [Figure 25.2, “SLEP channel”](#).

**Figure 25.2. SLEP channel**



There are no SLEP specific components of the channel entry currently.

## 25.3.2 Creating an external SLEP MTA Connection

There are 2 steps to this procedure.

- Create an external MTA associated with the remote connection.
- Create a domain in the Routing Tree entry to route addresses into the SLEP gateway.

---

**Note:** All addresses to this domain will be routed into SLEP gateway. It is not currently possible to route different addresses from the same domain differently.

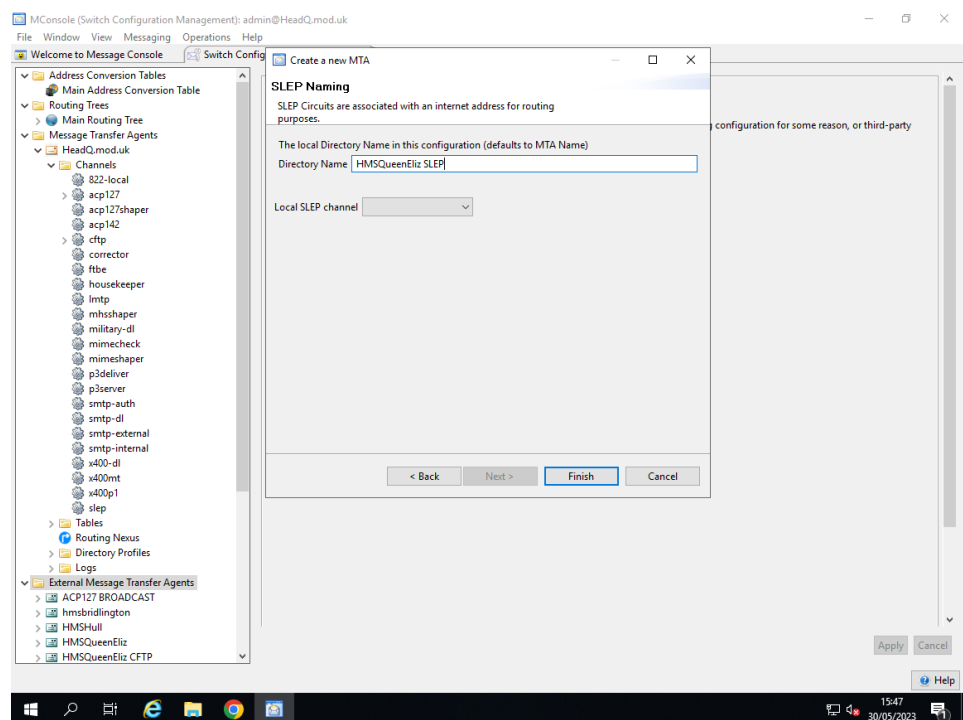
---

### 25.3.2.1 Create a remote external SLEP MTA

To create a remote MTA that represents the SLEP endpoint, first create an external MTA. This is done with MConsole on the **Switch Configuration Management** tab. Expand the **Message Transfer Agents** node, and then right click on **External Message Transfer Agents** and choose **New External MTA** from the menu. Alternatively use **Operations** → **New External MTA**.

When the wizard popup dialog appears, select **External SLEP Connection** and click **Finish** to complete the creation as shown in [Figure 25.3, “SLEP create external MTA”](#).

**Figure 25.3. SLEP create external MTA**



The new external MTA should now be present in the list of MTAs below the **External Message Transfer Agents** node. If the slep channel is not named **slep** then you will need to edit the transfer weightings by right clicking the new node and select **Modify Transfer Channel Weight** to update the channel.

### 25.3.2.2 Configure the external MTAs S'5066 address

After selecting the external MTAs channel, its possible to set the external S'5066 address.

#### S5066 Address

This is a STANAG S5066 address of the remote MTA.



### Destination SAP

The destination SAP used with the S5066 protocol to connect to remotely. It should normally be 10 (SLEP) unless special circumstances require it otherwise.

## 25.4 Setting up Routing

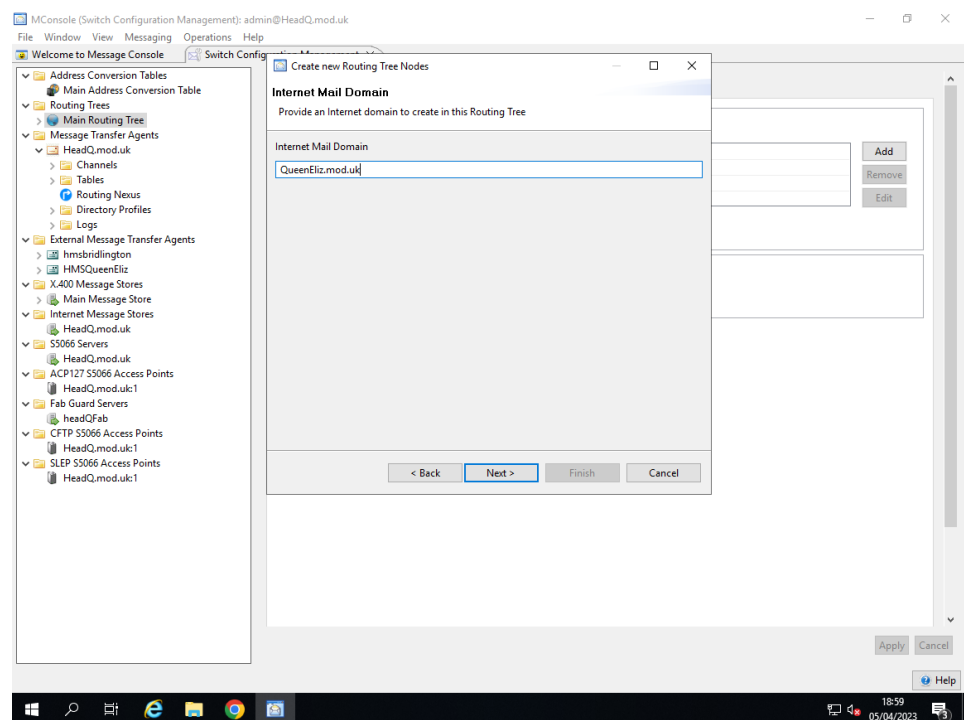
The final step of configuring an SLEP Gateway is to set up the routing.

### 25.4.1 Setting up Routing with Gateway Users

#### 25.4.1.1 Add a node to the Routing Tree

To add routing to the new external MTA, create a node representing the Internet address by navigating to the **Main Routing Tree** node under the **Routing Trees** entry in the **Switch Configuration Management** view. Right click and choose **Add nodes** and select the **Create Routing Tree entries representing an Internet domain**. Choose **Next** and enter the Internet domain of the entry, e.g. `queeneliz.mod.uk` such as shown in [Figure 25.4](#), “SLEP creation of Routing Tree node.”. Select **Finish**.

**Figure 25.4. SLEP creation of Routing Tree node.**



Once this is created, select the entry, and click on the **Add** to connect to the external MTA. Make sure the correct MTA is selected in the drop down menu. The default weight is normally acceptable unless complex setup is required.

---

## 25.5 SLEP Services

There is one service which is used to provide SLEP features:

- SLEP Service (mandatory). This is the main SLEP channel which sends and receives SLEP messages over the SLEP circuits to other SLEP stations.

# Chapter 26 Lists and Recipient Expansion Overview

M-Switch has features which enable the recipients of messages to be changed. One such set of features, known as List Expansion, is summarised in this chapter.

---

## 26.1 Introduction

Distribution lists provide a facility for expanding a single list address into multiple recipient addresses. This chapter summarises the expansion features available in M-Switch and provides links describing how to use each expansion feature.

For information on configuring distribution lists, see the following chapters:

- The creation and maintenance of SMTP (Internet) distribution lists that are stored in the directory is described here in [Chapter 27, \*SMTP Directory Based Distribution Lists\*](#).

Note: This feature can only be used to expand Internet addresses.

Note: A channel of this type named **smtp-dl** is created by the MTA configuration wizard by default for SMTP or MIXER MTAs.

- The contents of Directory-based, X.400 conformant distribution lists can be configured directly using MConsole, or using another DUA, such as Sodium. This is described in [Chapter 28, \*Directory Based X.400 Conformant Distribution Lists\*](#).
- Military Distribution Lists are described in [Chapter 29, \*Military Distribution Lists\*](#). This feature is used to support the ACP 133 schema for lists.

This feature supports the expansion of SMTP (Internet) messages, but can expand to both SMTP and X.400 addresses

- The **Profiler** is often used in Military environments and is a specialised form of recipient expansion. The profiler enables messages to be delivered to recipients based on message content.

The **Profiler** channel can expand and have members which are X.400, ACP127 or SMTP (Internet) addresses.

The **Profiler** is described in [Section 30.1, “Profiler Channel Introduction”](#).

---

**Note:** The file-based distribution lists feature is obsolete. It has been superseded by [Chapter 27, \*SMTP Directory Based Distribution Lists\*](#).

---

# Chapter 27 SMTP Directory Based Distribution Lists

This chapter describes the creation and maintenance of SMTP (Internet) distribution lists that are stored in the directory.

---

**Note:** This feature can only be used to expand to a list of Internet addresses.

---

For information on configuring other types of distribution lists see [Chapter 26, Lists and Recipient Expansion Overview](#).

By default the MTA creation wizard will create an **smtp-dl** channel. If you wish to setup a second Internet directory based distribution list you need to do the following:

- create and configure an Internet Directory **list** channel which will expand distribution lists found in the directory.
- create the list using Cobalt. See the Cobalt Administration Manual for details.
- assign members and other information to the list.

---

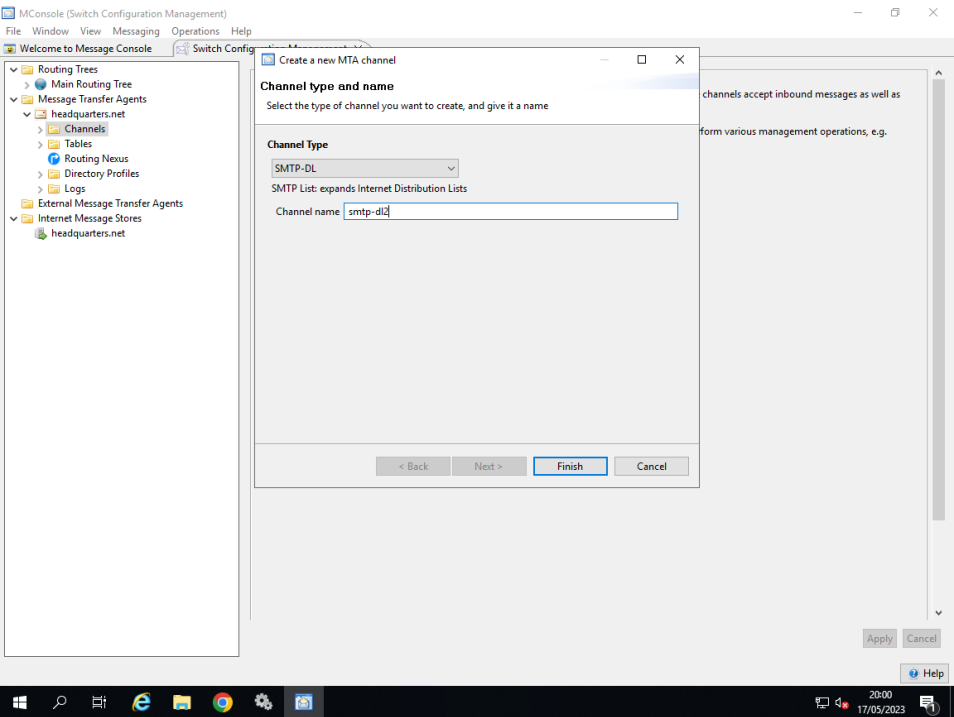
## 27.1 Creating an Internet distribution list channel

If you do not have a Internet distribution list channel, (for example if you have deleted the channel created by the Config/MTA creation wizard, or wish to create a second channel to expand lists, on the **Switch Configuration Management** tab, expand the **Message Transfer Agents** node, and the mta below that. Then with the **Channels** node selected, either right click and choose **New Channel** or choose **Operations** → **New Channel**.

Then provide the following answers when prompted by the wizard:

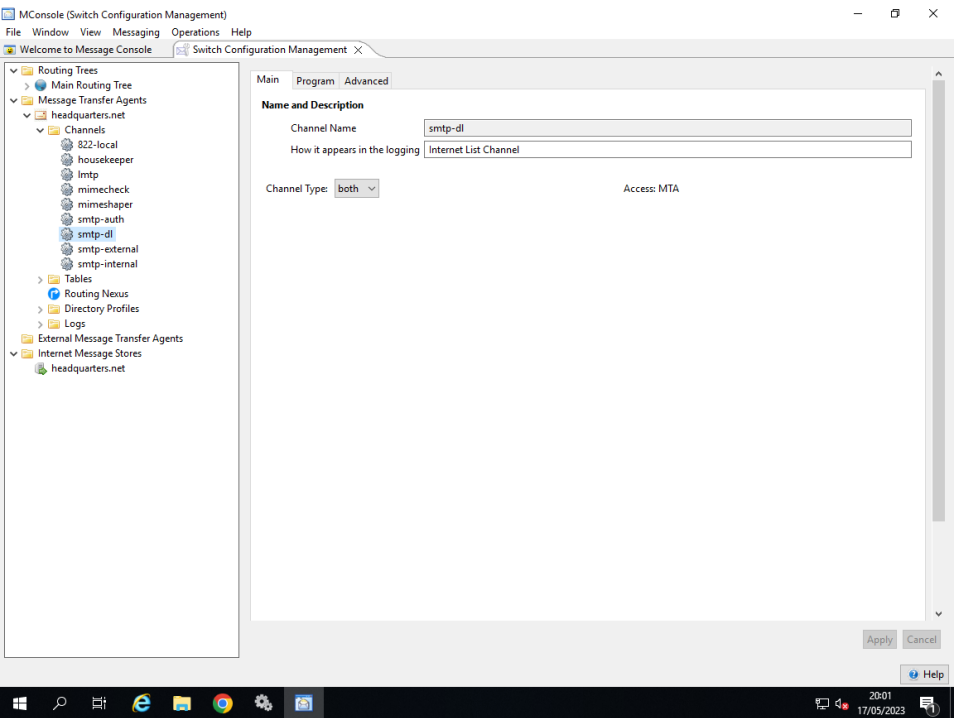
- **Channel Type** select **SMTP-DL**, which will present this description **SMTP List: expands Internet Distribution Lists**
- **Channel Name** you can name it as you wish. If you already have a channel from the MTA creation wizard, you will need to configure a different value from **smtp-dl** for example **smtp-dl2as** shown in [Figure 27.1, “Internet List channel creation”](#).

Figure 27.1. Internet List channel creation



After creating the smtp-dl channel, you should edit it so that the **Main** page looks like [Figure 27.2, “Internet List channel \(Main page\)”](#).

Figure 27.2. Internet List channel (Main page)



---

## 27.2 Creating a list

You can now create a new distribution list. To do this consult the Cobalt Administration Manual, then check the lists routing using ckadr:

### Example 27.1. Output from ckadr

```
C:\Program Files\Isode\bin>ckadr testlist@headquarters.net
testlist@headquarters.net -> (rfc822) testlist@headquarters.net
testlist@headquarters.net -> (x400)
/RFC-822=testlist(a)headquarters.net/O=headquarters/PRMD=Isode/
ADMD= /C=GB/
Delivered to headquarters.net by smtp-dl (weight: 0)
```

# Chapter 28 Directory Based X.400 Conformant Distribution Lists

Distribution lists provide a facility for expanding a single list address into multiple recipient addresses. This chapter describes the creation and maintenance of Directory-based X.400 conformant distribution lists.

For information on configuring other types of distribution lists see [Chapter 26, Lists and Recipient Expansion Overview](#).

The contents of the Directory-based, X.400 conformant distribution lists described in this chapter can be configured directly using MConsole, or using another DUA, such as Sodium.

---

## 28.1 X.400 conformant distribution lists

When running in X.400 conformant mode, the List channel provides distribution list expansion which is aligned with X.411. Each distribution list is held in an entry in the Directory, and includes attributes which store

- the O/R Address of the list itself,
- the OR-names to which the list expands
- an attribute (**DLSubmitPermissions**) which allows control over which originators are allowed to submit messages to the distribution list.

The distribution list entries are located below a single root entry, called the List Owner DN.

Tools to manage the lists and its subscribers are included in MConsole.

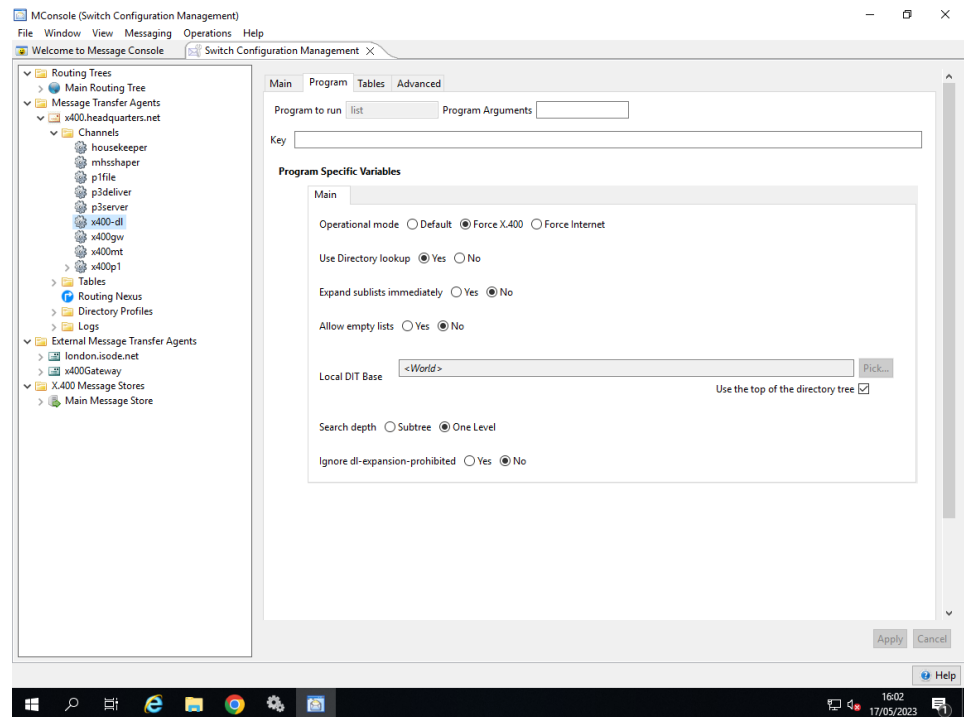
---

## 28.2 X.400 List Channel Creation

A standard X.400 Messaging Configuration already creates an X.400 List channel and a Hybrid List channel, so you will not need to create them. If they are missing, follow the instructions below to create them.

To create a suitable channel to expand X.400 lists, right click on the **Channels** folder, select **New Channel** and select the **X.400-DL** option.

After creating the channel you should also check that the **Program** page looks like [Figure 28.1, “List channel \(Program page\)”](#).

**Figure 28.1. List channel (Program page)**

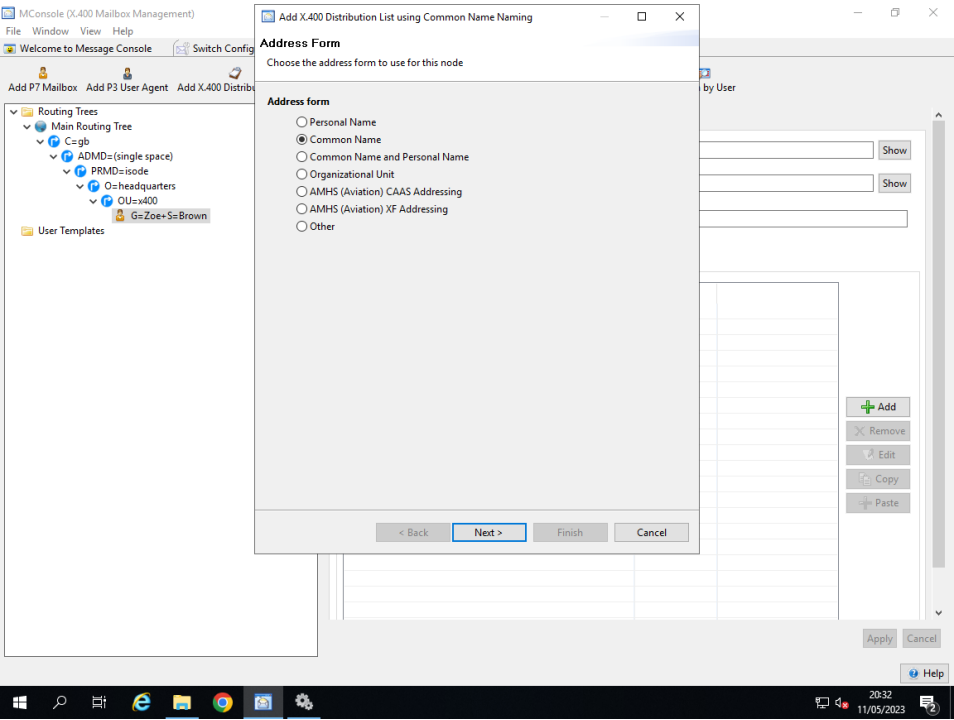
## 28.3 Creating the distribution list

Using MConsole **X.400 Mailbox Management**, right click on the entry under which you wish to create the distribution list, and then click on **Add X.400 Distribution List**

Firstly enter the way in which the list is to be named.

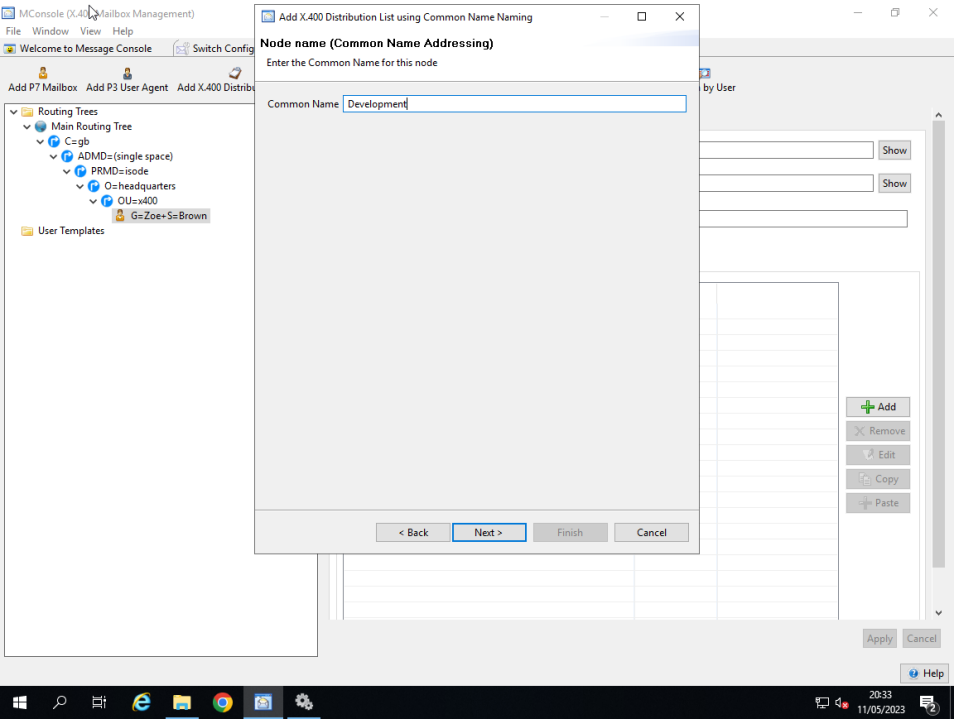


Figure 28.2. The X.400 DL user creation

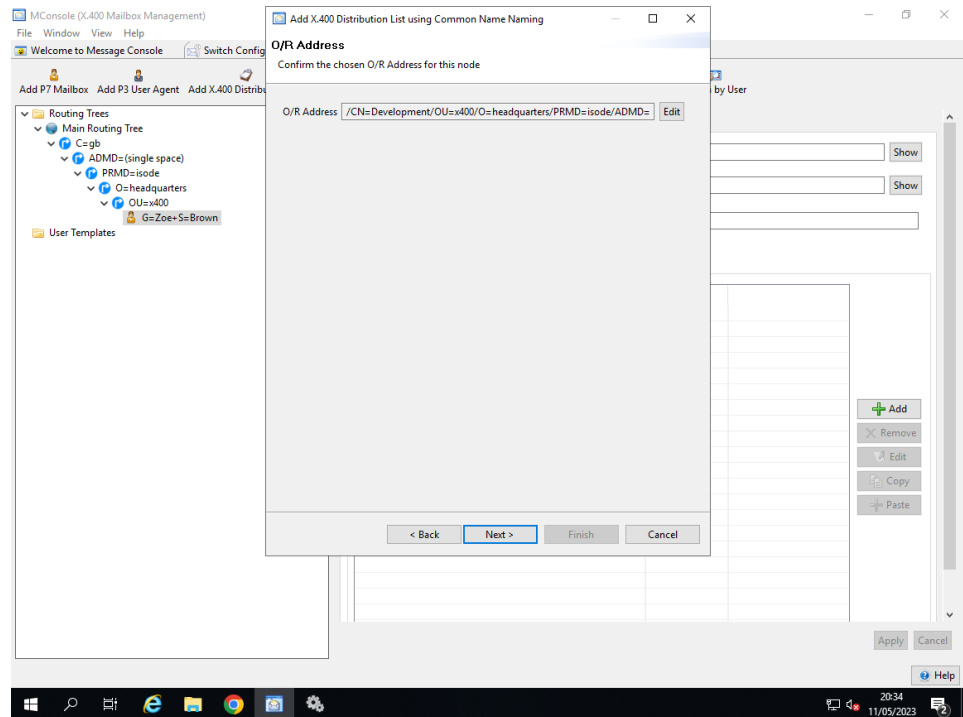


Then enter the actual name of the list.

Figure 28.3. The X.400 DL user creation - name

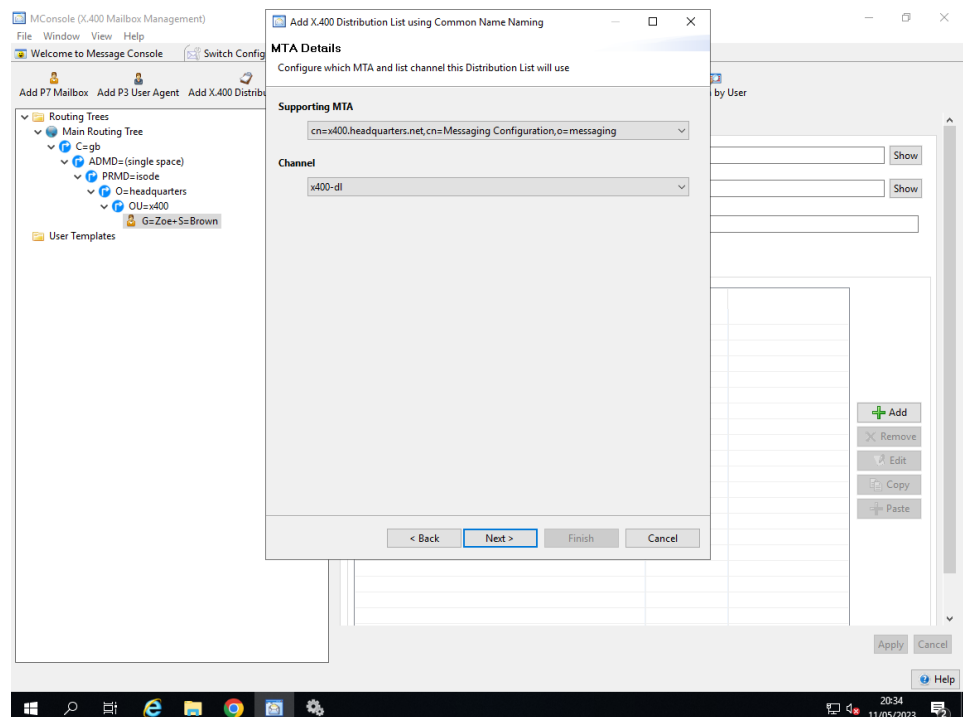


Then confirm the O/R Address of the list.

**Figure 28.4. The X.400 DL user creation - confirm O/R Address**

Then set the supporting MTA and list channel. The channel should be either the standard X.400 List channel **x400-dl** or the **hybridList** channel.

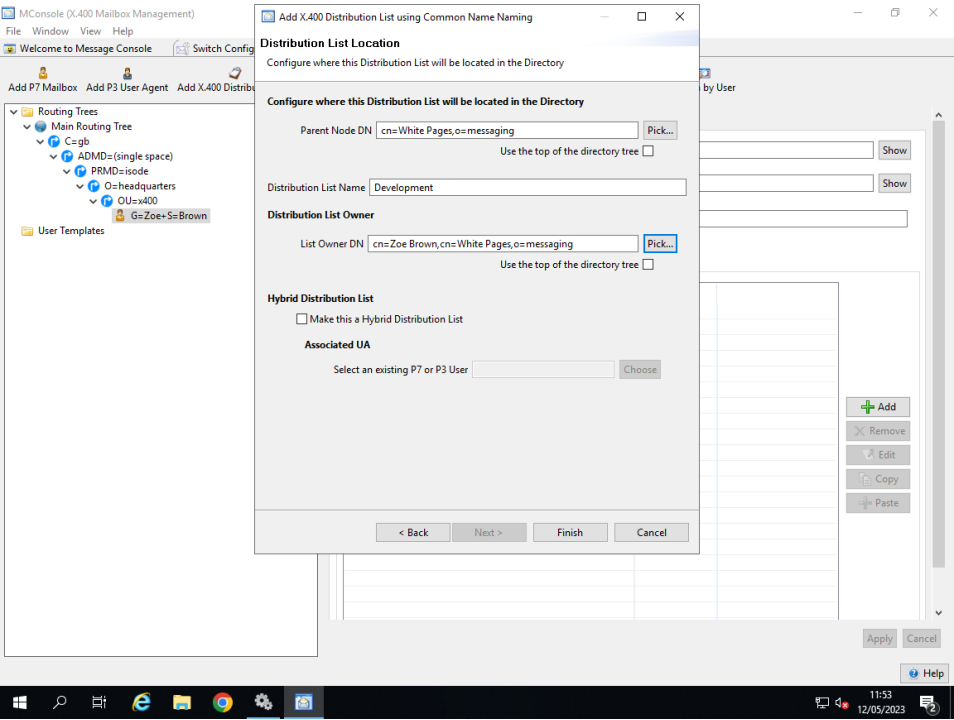
Hybrid Distribution Lists are X.400 Distribution Lists, but have two special features that makes them useful in Aviation configurations. For more information about Hybrid Lists, refer to the section below.

**Figure 28.5. The X.400 DL user creation**

Then confirm the entry under which the list itself is configured.

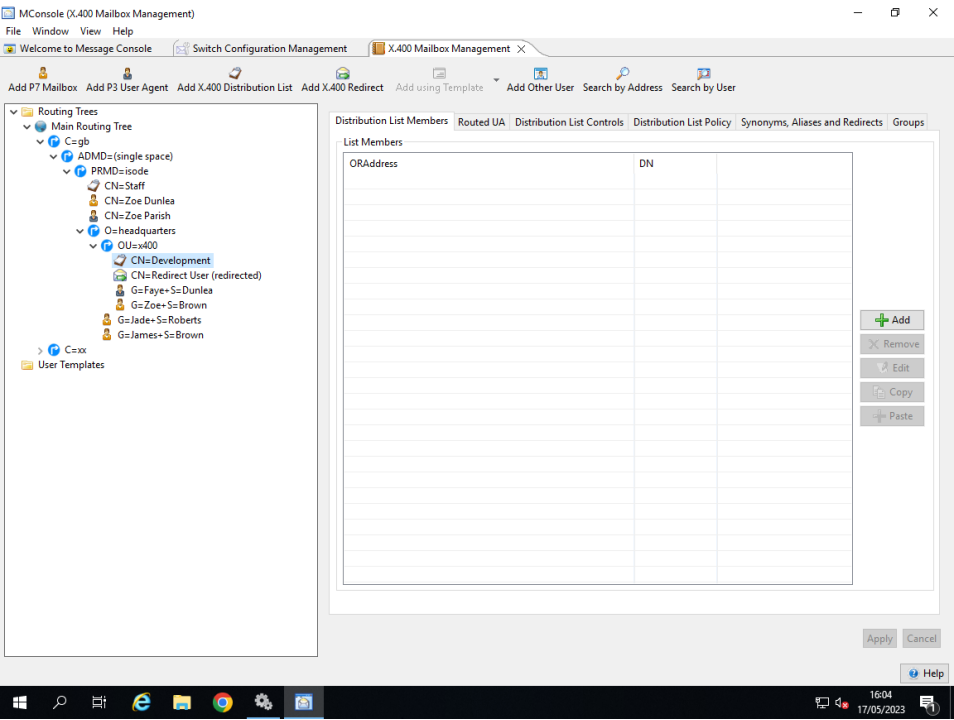
If you want to create a Hybrid List, set the **Make this a Hybrid Distribution List** and select an associated user by clicking on the **Choose** button.

Figure 28.6. The X.400 DL user creation



Then click on **Finish** and the list is displayed.

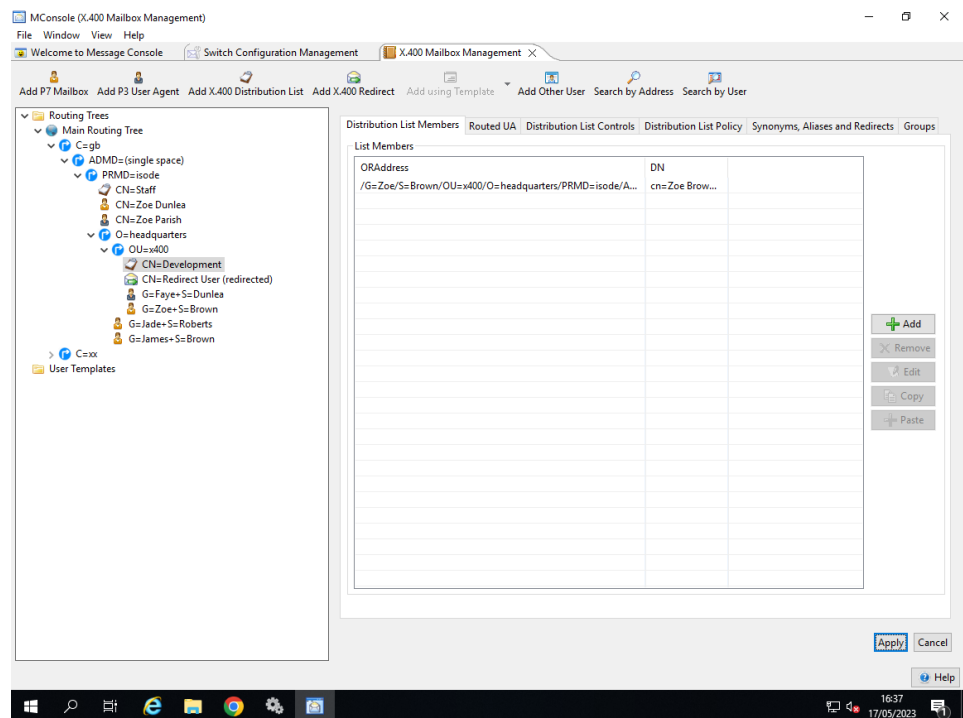
Figure 28.7. The X.400 DL user creation



## 28.4 Editing distribution list subscribers

You can now add the members of the list by clicking on **Add** in [Figure 28.7, “The X.400 DL user creation”](#).

**Figure 28.8. The X.400 DL subscribers editor**



This editor allows you to see and modify the subscribers of the selected X.400 Distribution List. The list is composed of OR-names. OR-Names are made of a Directory Name (like `cn=manager`, `o=isode`, `c=gb`) and an O/R Address (like `/cn=manager/o=isode/admd=/c=gb/`).

You can also add subscribers to a Distribution List by selecting one or more entries on the left-hand display, dragging them on to the target Distribution List entry (also on the left-hand-side) and dropping them. A confirmation box will display all of the O/R Addresses which are about to be added.

## 28.5 Editing distribution list configuration

This is configured by selecting the various tabs in the MConsole **X.400 Mailbox Management** view in [Figure 28.8, “The X.400 DL subscribers editor”](#).

### 28.5.1 Permissions page

This feature is not currently available.

DL Permissions values are a list: you can select one or more different kinds of permissions and if a user matches one of the permissions it is allowed to submit to the list.

By default, anyone can send to an X.400 Distribution List, and that is reflected by the value **everyone** in the **Permissions** field.

If you want to change that, **Delete** the **everyone** entry and click **Add**.

The new window called **Add List Submit Permission** is displayed. Change the **Permission type** option to the value you want.

Some options need you to specify an OR-Name in the lower fields. OR-Names are made of a Directory Name (like `cn=manager, o=isode, c=gb`) and an O/R Address (like `/cn=manager/o=isode/admd= /c=gb/`).

Permission type options:

#### Everyone

Everyone is allowed to submit to this DL

#### OR-name Pattern

Allows you to specify a OR-name pattern, and those OR-names that match the pattern will be able to submit to this DL.

#### OR-name of individual

The OR-name of a specific individual who is allowed to submit to this list.

#### Member of DL with OR-Name

If a user is a member of the Distribution List specified by the OR-name below, then it can submit to this list.

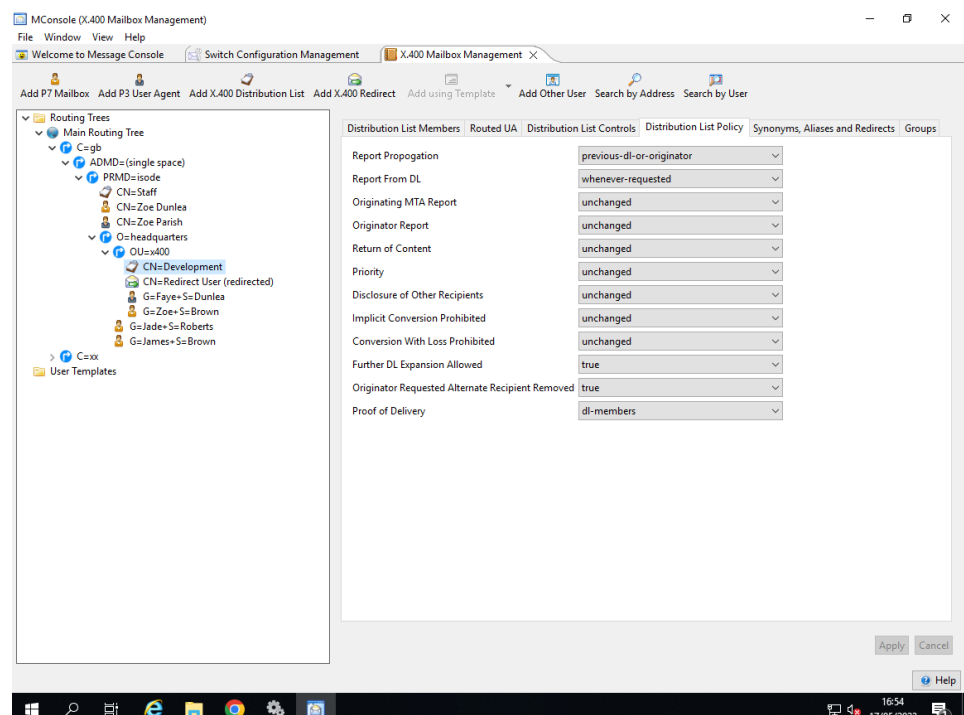
#### Member of Directory Group

If a user is a member of the Distribution Group specified by the Directory Name below, then it can submit to this list.

## 28.5.2 Policy page

A DL policy may specify values for a number of options.

**Figure 28.9. The X.400 DL policy editor**



**Report Propagation**

Whether reports received at the DL expansion point are to be sent to the preceding DL (or the originator if no preceding DL), or to the DL owner, or to both of these.

**Report from DL**

Whether the DL expansion point sends a confirmatory delivery report whenever it expands a message which requests one, or whether such reports are sent only either when report propagation is dl-owner or when originator-report is no-report or non-delivery-report.

**Originating MTA Report**

Whether the MTA report request is unchanged, or set to request both delivery and non-delivery reports, or set to request only non-delivery reports, or set to request audited delivery reports.

**Originator Report**

Whether the originator's report request is unchanged, or set to request no reports, or set to request both delivery and non-delivery reports, or set to request only non-delivery reports.

**Return of Content**

Whether the originator's request for return of content is unchanged, or set to request no return, or set to request return with non-delivery reports.

**Priority**

Whether the originator's setting for priority is unchanged, or set to normal, or set to non-urgent, or set to urgent.

**Disclosure of Other Recipients**

Whether the originator's setting is unchanged, or set to prohibit disclosure, or set to allow disclosure.

**Implicit Conversion Prohibited**

Whether the originator's setting is unchanged, or set to allow implicit conversion, or set to prohibit implicit conversion.

**Conversion With Loss Prohibited**

Whether the originator's setting is unchanged, or set to allow conversion with loss, or set to prohibit conversion with loss.

**Further DL Expansion Allowed**

Whether expansion by any nested DLs is allowed or prohibited.

**Originator Requested Alternate Recipient Removed**

Whether the originator's requested alternate recipient setting is removed (true) or not (false).

**Proof of Delivery**

Can be set to DL members, DL expansion point, neither or both.

Further details of these policy options are in *X.402/ISO/IEC 10021-2*.

---

## 28.6

## Hybrid Distribution Lists

When running in X.400 conformant mode, the List channel provides distribution list expansion which is aligned with X.411. Each distribution list is held in an entry in the Directory, and includes attributes which store

Hybrid Distribution Lists are X.400 Distribution Lists, but have two special features.

The channel that they use, **hybridList**, is set to "Ignore dl-expansion-prohibited", which means that if an X.400 IPM is sent to a hybrid list, and the X.400 flag

"DL-Expansion-Prohibited" is set to **True**, it will be ignored and the message will be expanded anyway.

This is useful in Aviation configurations, as messages with SS priority have to have the value "DL-Expansion-Prohibited" set to **True**, but in certain environments the administrator may want to expand them anyway.

You can associate an existing X.400 P3 or P7 user with the Hybrid Distribution List. This user will then be able to bind to the MTA or Message Store and submit messages with the O/R address of the Hybrid Distribution List as the destination.

These two features put together can be used to implement something similar to a "Copy Traffic" feature, but without the restrictions of using a standard X.400 Distribution List, which will comply with the X.400 flag "DL-Expansion-Prohibited" and won't permit messages being submitted using its O/R address.

# Chapter 29 Military Distribution Lists

Distribution lists provide a facility for expanding a single list address into multiple recipient addresses. This chapter describes the creation and maintenance of Military specific address lists.

For information on configuring other types of distribution lists see [Chapter 26, Lists and Recipient Expansion Overview](#).

---

## 29.1 Military Distribution Lists

The military distribution lists are similar in their process to other distribution lists, but with a few key changes.

- It follows the ACP 133 schema for lists.
- It deals with SMTP format emails, but can be either SMTP (Internet) or X.400 addresses.
- Exempted addresses can be specified, and duplicate addresses are removed.
- Recipients can be marked as either Action (i.e., To:) or Info (i.e., Cc:) recipients. When the list is expanded it may result in two messages being sent, one for the action and one for the info recipients.

Based on these properties it is possible for the recipient to tell if a message arrives as an Action or Info message

---

## 29.2 Channel configuration

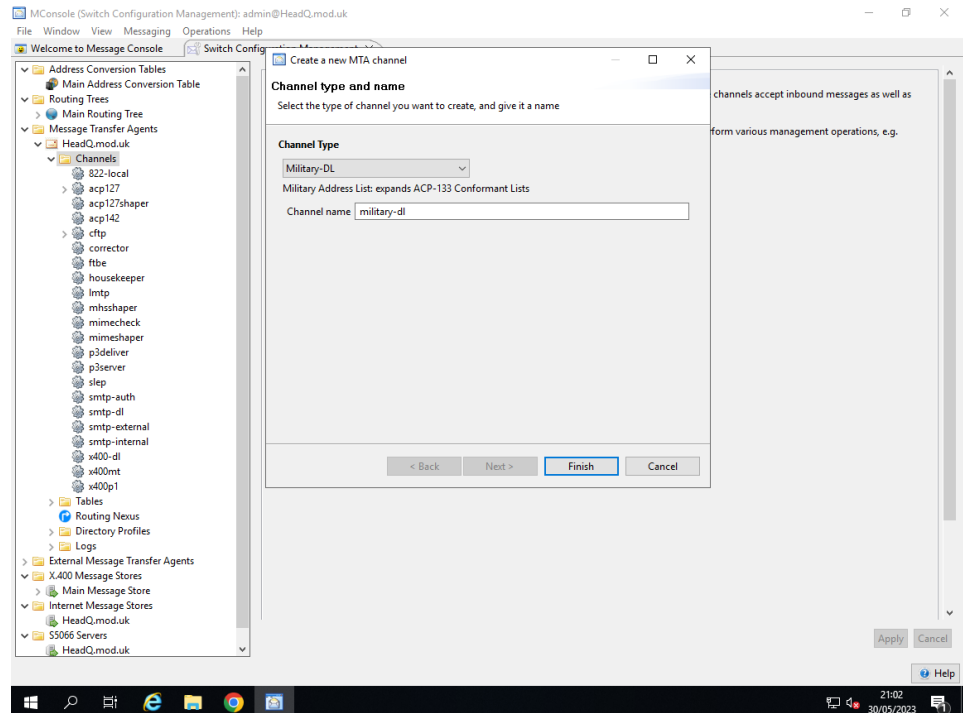
To create a Military List expander, from the **Switch Configuration Management** view, expand the **Message Transfer Agents** node, and the MTA below that. Then with the **Channels** node selected, either right click and choose **New Channel** or choose **Operations** → **New Channel**.

Then provide the following answers when prompted by the wizard:

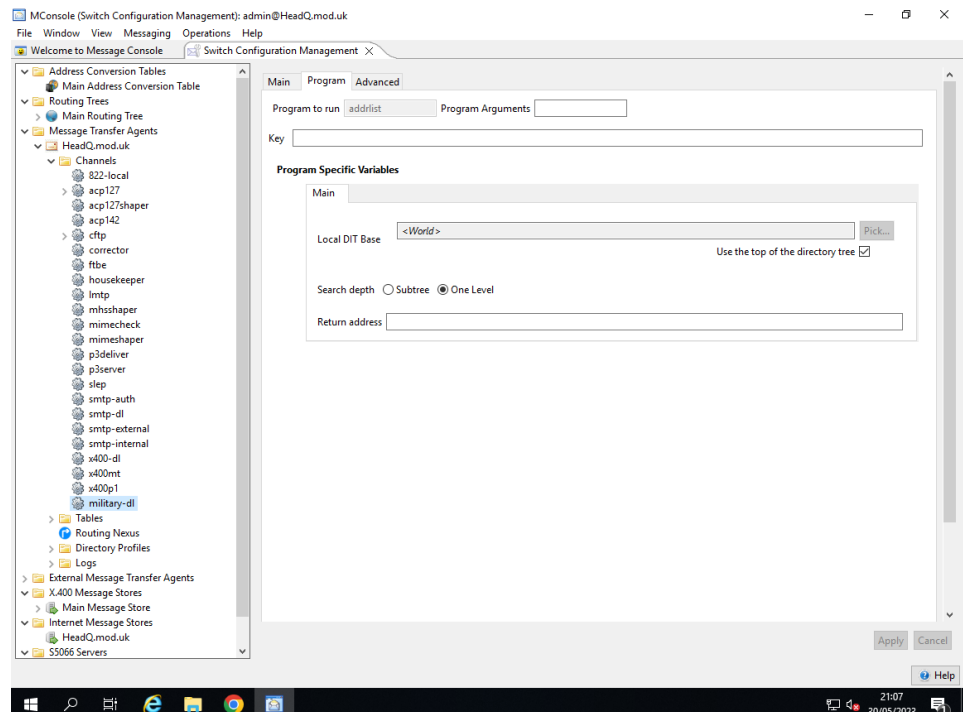
- **Channel Type** select **Military-DL** which will be labelled **AddrList: expands ACP-133 Address Lists**
- **Channel Name** name it `military-dl`

as shown in [Figure 29.1, “Military list channel”](#).



**Figure 29.1. Military list channel**

Once the entry is created, it may need some customisation. Usually the values on the **Main** are suitable. The values on the **Program** tab in [Figure 29.2, “Military list channel editor”](#) show the defaults, and may need some changes.

**Figure 29.2. Military list channel editor**

- The local directory information tree base can be selected by picking an entry in the **Local DIT Base** option, which allows the searching of list entries to be restricted to part of the directory tree.
- The search depth can be configured to either search one level below the DIT point selected above, or can search the whole subtree for entries.

- The return address can be configured to specify an address to originate the messages expanded by the list from. If this is left empty, an empty SMTP email will be used to indicate errors should not be sent.

# Chapter 30 Profiler Channel

A Profiler is a Messaging component which takes an input message and distributes the message to new recipients based on the information in the message.

For information on configuring distribution lists, see the following chapters:

- [Chapter 27, SMTP Directory Based Distribution Lists.](#)
- [Chapter 28, Directory Based X.400 Conformant Distribution Lists.](#)
- [Chapter 29, Military Distribution Lists.](#)

---

## 30.1 Profiler Channel Introduction

The M-Switch Profiler channel allows a message to be redistributed to a set of recipients based on the message itself:

- A message (the distributed message) to a profiled address is configured to be delivered to the **Profiler** channel. This means that any positive delivery reports are generated.
- The profiler determines new recipients for the message. This is done using information in the message, such as the Subject or SICs.
- The new (or distribution) recipients can be Action or Information recipients. The **Profiler** may determine that the original recipient being distributed was an information recipient of the original message. In this case, all recipients are information recipients. If a message has more than one local recipient which is to be distributed, all recipients are processed together. This enables the procedure to ensure that a given distribution recipient only receives one copy of the message.
- A new message (the distribution message) is generated and submitted to the MTS, with the new recipients as the envelope recipients.

There are these inputs to the channel:

- The attributes of the message which are used in conjunction with the configuration.
- Profiler Configuration.
  - The configured **rules**, which are an aggregate of sets of **rules**, called **plans**, which can be enabled or disabled.
  - Information about **Exercises**, **Operations**, **Projects** and **Drills** (generically called **Exercises** here).

---

## 30.2 Configuration of the Profiler Channel.

### 30.2.1 Profiler Channel Config Files

The following files are needed and are described in this section.

*switch/profiler-scan.xml*

See [Section 30.2.3, “Editing The Profiler Channel”](#). A useable version is installed in  $\$(SHAREDIR)$  and will be used by the channel, but if  $\$(ETCDIR)switch/profiler-scan.xml$  is created, this can be edited and will be used in preference.

*switch/profiler.json*

See [Section 30.4.1, “Main Profiler Channel Configuration File”](#).

Examples are installed in  $\$(SHAREDIR)/switch/profiler/examples$ , which can be copied into  $\$(ETCDIR)switch/profiler.json$ .

*switch/profiler-manual-catalog.json*

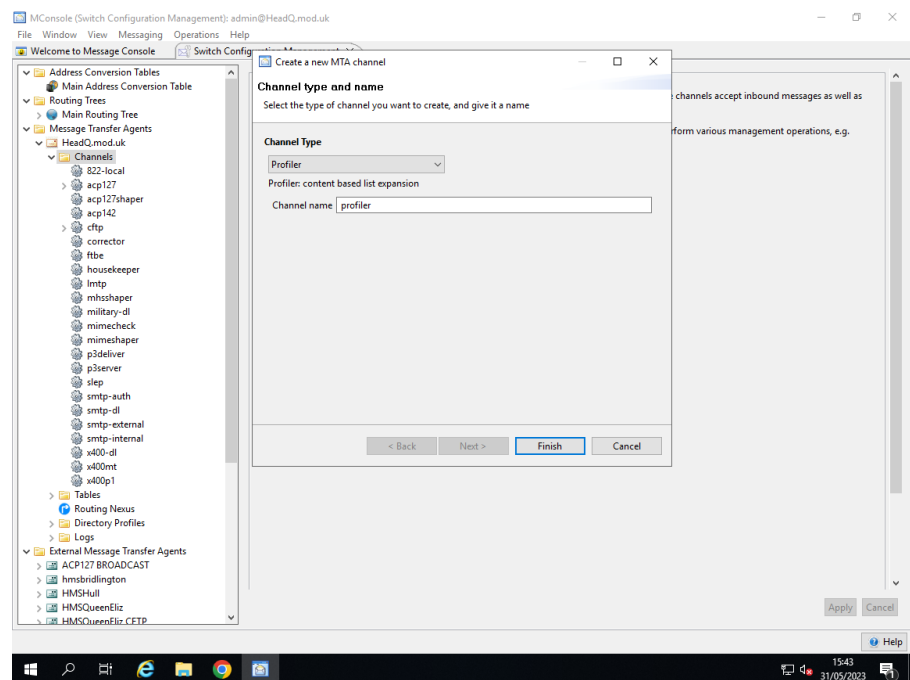
See [Section 30.2.5.4, “Manual Profiler After Logging In”](#).) An example is installed in  $\$(SHAREDIR)/switch/profiler/examples/profiler-manual-catalog.json$ , which can be copied into  $\$(ETCDIR)switch/profiler-manual-catalog.json$ .

## 30.2.2 Setting up the Profiler Channel

The steps needed to set up the Profiler channel are:

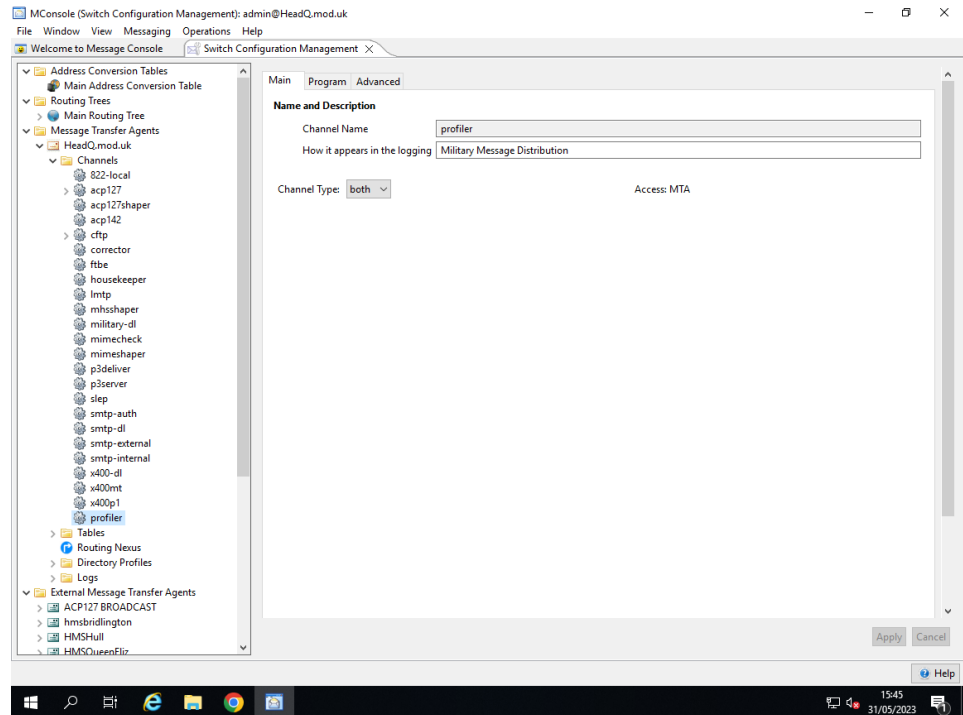
- Add the channel
  - On the **Switch Configuration Management** view, right click on the **Channels** folder under the **Message Transfer Agents** folder and select **New channel**.
  - Select "Profiler" as the channel type
  - Set the channel name: the default of "profiler" will normally not need to be changed.
  - Click on **Finish** to create the channel.

**Figure 30.1. Create Profiler Channel**



## 30.2.3 Editing The Profiler Channel

After creating the Profiler channel, Switch Configuration View appears as follows.

**Figure 30.2. Profiler Channel (Main Tab)**

Configurable items on this screen are:

#### Channel Name

This value is display only and cannot be edited. The default during creation is `profiler`

#### How it appears in the logging

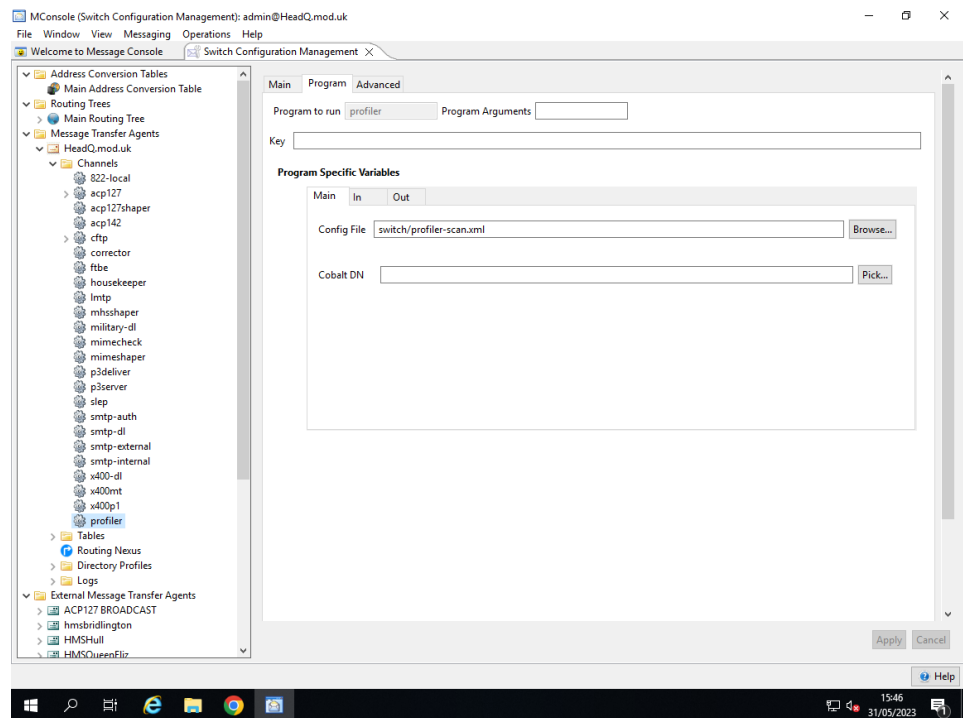
The default is `Military Message Distribution` You can change this to any value, which will result in the logging output changing.

#### Channel Type

This can be **in**, **out**, or **both**. This defaults to **both**.

Selecting the Program Tab of the Profiler channel, causes the Switch Configuration View to appear as follows.

**Figure 30.3. Profiler Channel (Program Tab, and Main Program Specific Variables)**



Configurable items on this screen are:

#### Program Arguments

No values are needed here

#### Key

No values are needed here

#### Config File

The name of the file which controls the extraction of data from the message content to be passed into the Profiler Channel. (This is similar to the way scanning on message submission is configured using `$(OPTDIR)/submitscanconfig.xml`. See [Section 39.2, “Scanning on Submission”](#).)

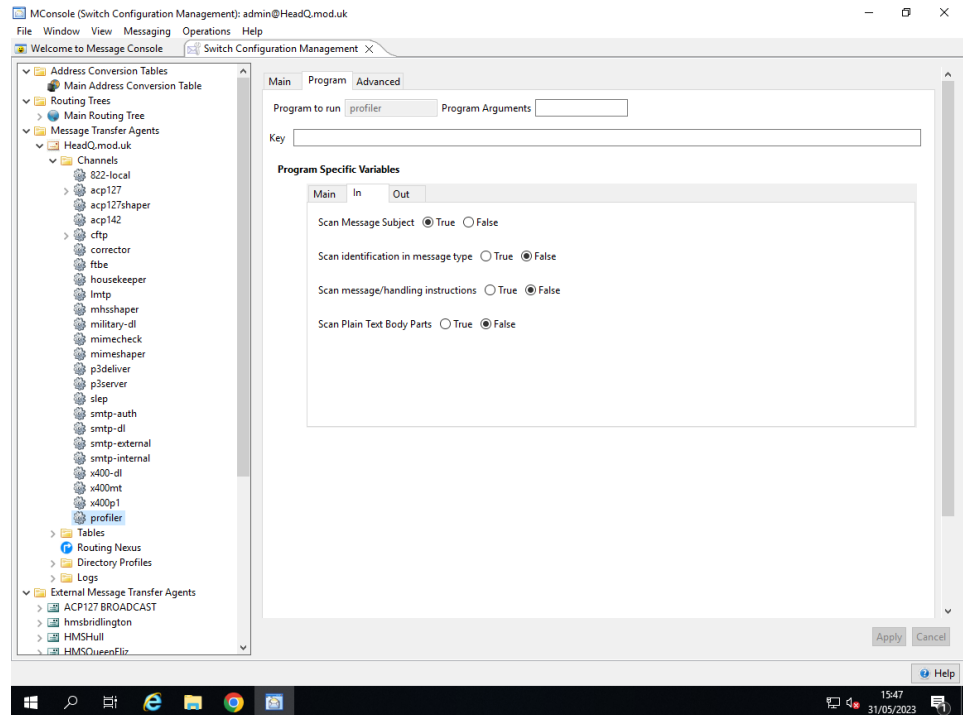
Defaults to `$(SHAREDIR)/switch/profiler-scan.xml`. This is installed as part of M-Switch. In the unlikely event that you need to make changes to this file, make a copy in `$(ETCDIR)/switch/profiler-scan.xml` and edit that. This will prevent your changes being overwritten when upgrading.

#### Cobalt DN

If the Profiler channel configuration is stored in the DSA (and configured via Cobalt), then the DN of this location should be given here.

The file `$(ETCDIR)/switch/profiler.json` is then created (if not already present) by Cobalt from the Profiler configuration in the DSA. This file will be used if the DSA cannot be contacted. If the file is newer than the DSA configuration (local file edits) then it is not overwritten. File edits are preserved until Cobalt updates the DSA configuration or the file is deleted.

Selecting the Program Tab of the Profiler channel and the In tab, causes the Switch Configuration View to appear as follows.

**Figure 30.4. Profiler Channel (Program Tab, and In Program Specific Variables)**

This controls which parts of the message are scanned for keywords. By default 'subject' only is scanned. Configurable items on this screen are:

#### **Scan Message Subject**

If true, the subject is scanned for keywords. Default true.

#### **Scan identification in message type**

If true, the message identification is scanned for keywords. Default false.

#### **Scan message/handling instructions**

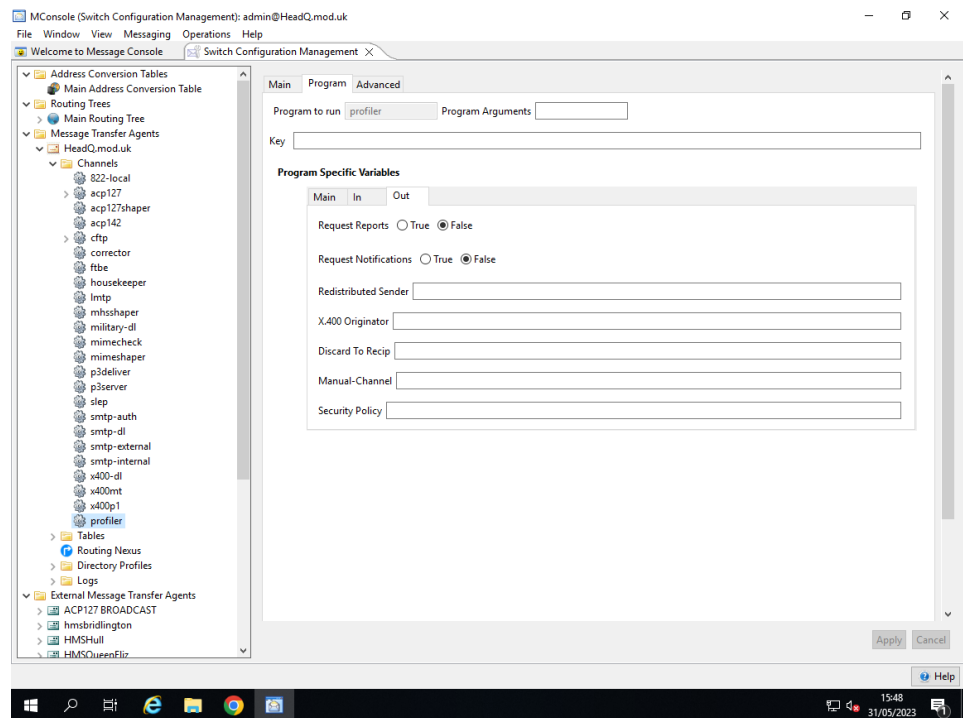
If true, the message Handling Instructions are scanned for keywords. Default false.

#### **Scan Plain Text Body Parts**

If true, plain text body parts are scanned for keywords. Default false.

Selecting the Program Tab of the Profiler channel and the Out Tab, causes the Switch Configuration View to appear as follows.

**Figure 30.5. Profiler Channel (Program Tab, and Out Program Specific Variables)**



Configurable items on this screen are:

#### **Request Reports**

If true, positive delivery reports are requested. Default false

#### **Request Notifications**

If true, read notifications are requested. Default false

#### **Redistributed Sender**

The envelope address for 822 messages from which the message is distributed. If not set, defaults to the postmaster address.

#### **X.400 Originator**

Originator for the message if the original message is an X.400 message.

#### **Discard To Recip**

If manual distribution is not configured, and a recipient of a message is not distributed, then discard the message to this recipient rather than non-delivering.

#### **Manual-Channel**

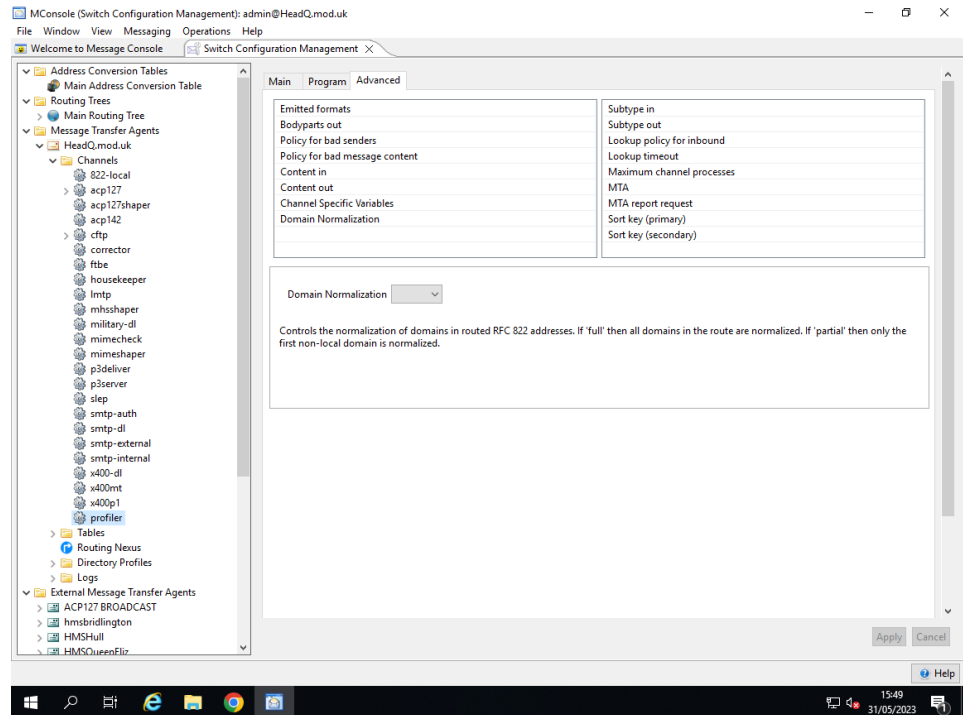
Name for the channel to use for manual distribution. This channel does not need to be explicitly configured. If set, messages being delivered by the Profile Channel will be put onto this dynamically created channel if no Rules are triggered. If this is not set, then manual distribution is not performed and the message is non-delivered.

#### **Security Policy**

The name of the security policy to use for the display of security labels for the manual distribution. The name should be for a security-policy defined in the channel's XML configuration file.

Selecting the Advanced Tab of the Profiler channel, causes the Switch Configuration View to appear as follows.



**Figure 30.6. Profiler Channel (Advanced Tab)**

These values are unlikely to require setting unless instructed by Isode Support.

## 30.2.4 Configure a User

In order to configure a user to be delivered into the Profiler channel for redistribution, the delivery channel of the user needs to be set.

Edit the user in **Cobalt**.

---

**Note:** You must also configure a postmaster address which is routeable in the form `postmaster@<local-domain>` e.g. `postmaster@example.com`.

---

## 30.2.5 Manual Profile Channel

### 30.2.5.1 Channel Configuration

If the Profiler channel attempts to redistribute a message for which none of the configured rules applies, by default the message is non-delivered.

Alternatively the message can be delivered into the **Manual Profile Channel**. This is configured by setting the **Manual Channel** name in the **Profiler** channel itself. See [Figure 30.5, “Profiler Channel \(Program Tab, and Out Program Specific Variables\)”](#).

---

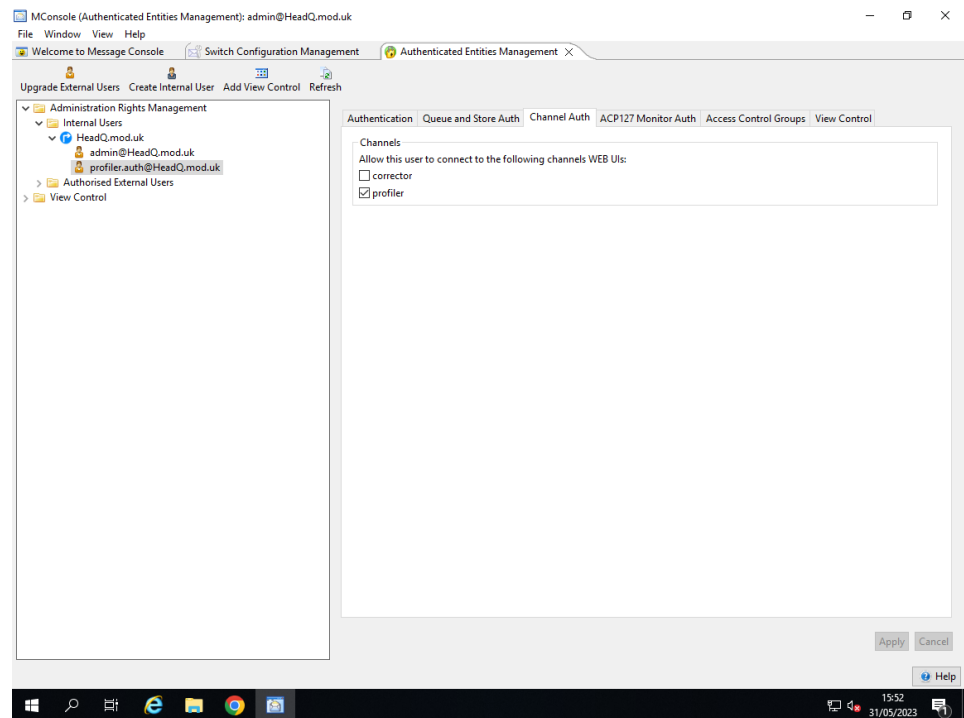
**Note:** This channel *must not* be configured in the usual way. By setting the manual-channel value in the Profiler channel, the Manual Channel is automatically created on demand by the Queue Manager.

---

### 30.2.5.2 User Configuration

To authenticate to the Manual Profiler Channel, you must configure a User with a **ChannelAuth** attribute with the value `profiler`. The following shows how this is done using the **Authenticated Entities View**. Set the checkbox in the **ChannelAuth** tab for the profiler channel.

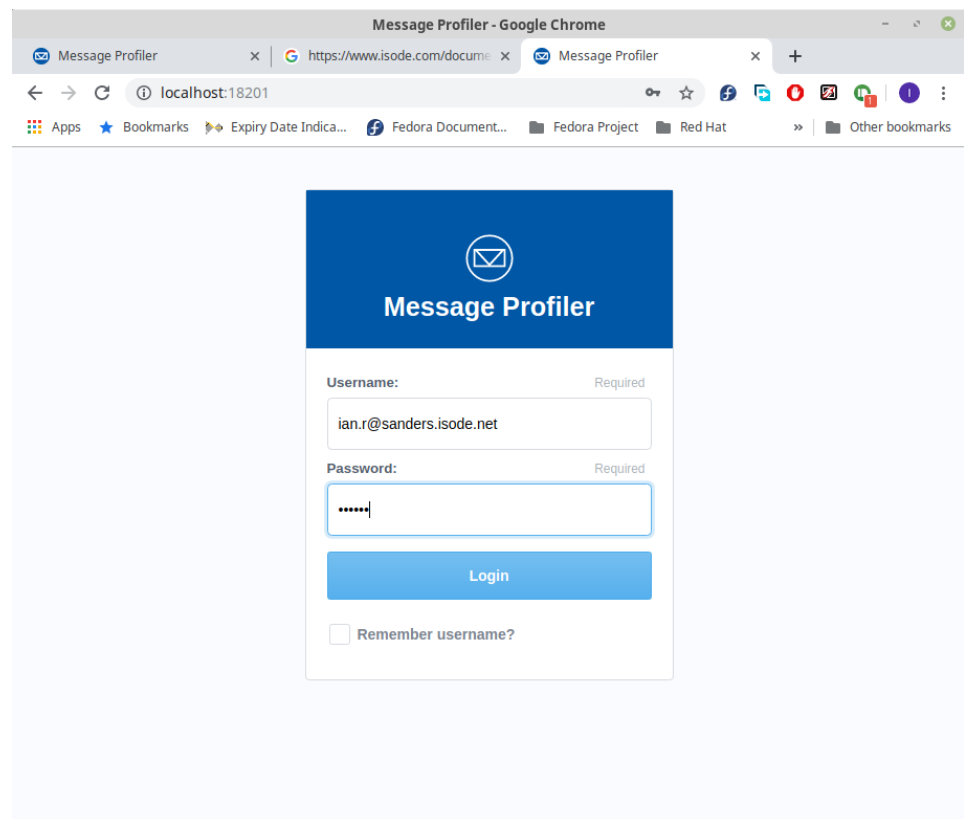
**Figure 30.7. Configure a User To Be Able To Manage Redistribution Of Messages Using The Manual Profiler Channel**



### 30.2.5.3 Connecting To The Manual Profiler

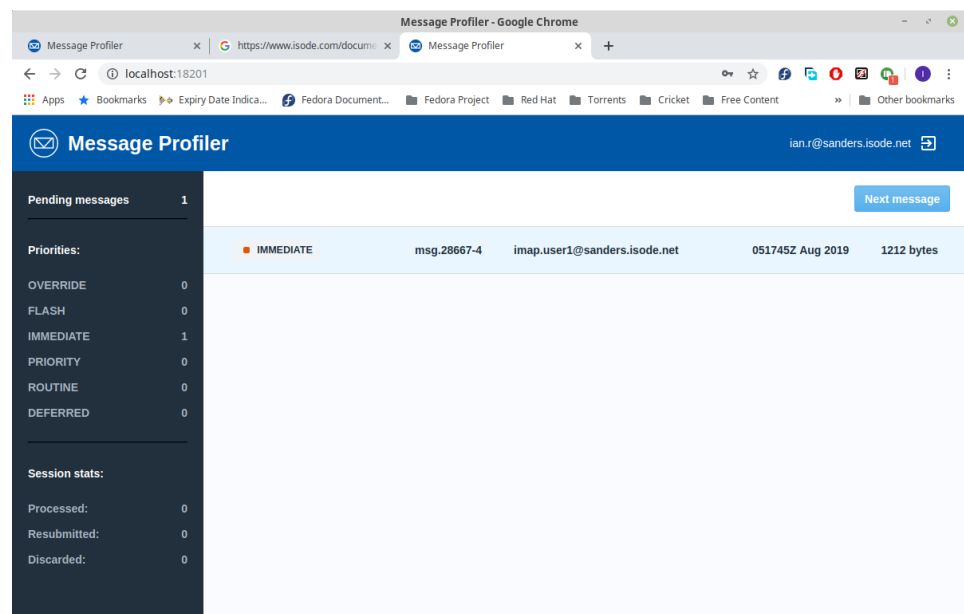
To connect to the Manual Profiler to manage redistribution, you need to use a browser such as Chrome, Firefox etc.

Point your browser at <https://localhost:18201/> and Login using the username and password set up in [Section 30.2.5.2, “User Configuration”](#).

**Figure 30.8. Logging into The Manual Profiler Channel**

### 30.2.5.4 Manual Profiler After Logging In

Once logged in the Browser shows any messages queued on the Manual Profiler Channel.

**Figure 30.9. Logged into The Manual Profiler Channel**

Selecting a queued message allows the User to redistribute the message. The Manual Profiler Channel will load up the distribution lists which are configured in *\$(ETCDIR)/switch/profiler-manual-catalog.json*. The following figure shows a Distribution List in which *\$(ETCDIR)/switch/profiler-manual-catalog.json* has the the following content:

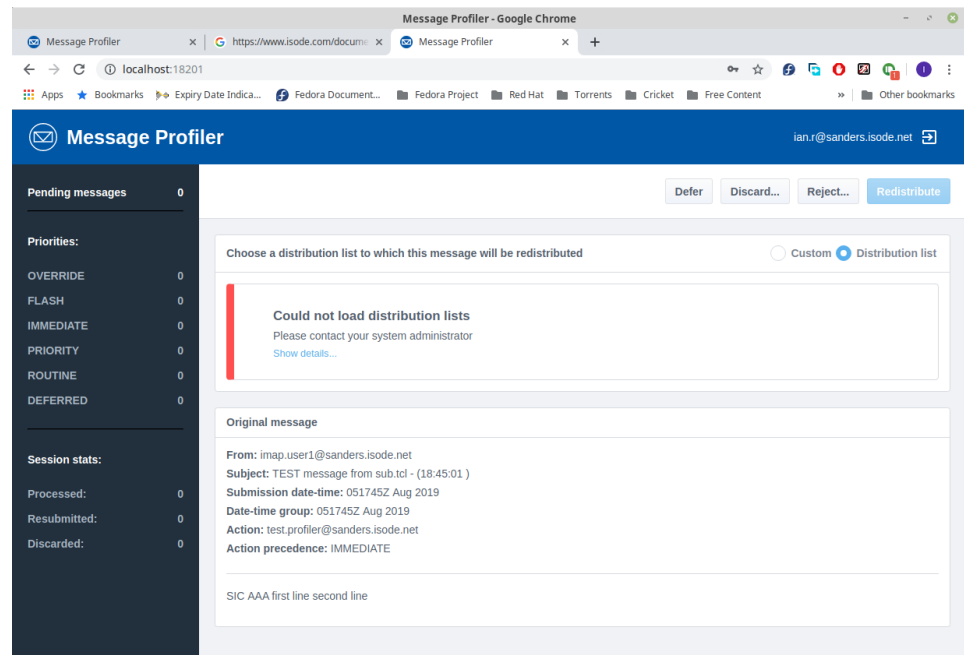
```
[
  {
    "name": "Designated operators",
    "description": "A group of operators designated to receive distributed messages",
    "action": [ "user2@example.com" ],
    "info": [ "user3@example.com" ]
  },
  {
    "name": "Ships crew",
    "description": "Crew members designated to receive distributed messages",
    "action": [ "user4@example.com", "user5@example.com" ],
    "info": [ "user6@example.com" ]
  }
]
```

**Figure 30.10. Manual Profiler Channel Showing Message Detail**

The screenshot shows the Message Profiler web interface in a Google Chrome browser. The address bar shows 'localhost:18201'. The interface has a dark blue header with the 'Message Profiler' logo and the user 'ian.r@sanders.isode.net'. On the left is a sidebar with 'Pending messages' (0) and 'Priorities' (OVERRIDE, FLASH, IMMEDIATE, PRIORITY, ROUTINE, DEFERRED, all 0). Below that is 'Session stats' (Processed, Resubmitted, Discarded, all 0). The main area shows options to 'Defer', 'Discard...', 'Reject...', or 'Redistribute'. A section titled 'Choose a distribution list to which this message will be redistributed' has radio buttons for 'Custom' and 'Distribution list' (selected). Below this, 'Distribution list' is set to 'Designated operators', 'Action recipients' is 'user2@example.com', and 'Information recipients' is 'user3@example.com'. The 'Original message' section shows: 'From: imap.user1@sanders.isode.net', 'Subject: TEST message from sub.tcl - (14:40:41)', 'Submission date-time: 061340Z Aug 2019', 'Date-time group: 061340Z Aug 2019', 'Action: test.profiler@sanders.isode.net', 'Action precedence: IMMEDIATE', and the message body 'SIC AAA first line second line'.

If no distribution lists have been configured in `$(ETCDIR)/switch/profiler-manual-catalog.json`, the following is presented:

**Figure 30.11. Manual Profiler Channel Showing Message Detail But No Distribution Lists**



## 30.3 Profiler Channel Use of Attributes From Message

### 30.3.1 Message Attributes

The message attributes to be passed into the Profiler are:

- the local recipients to be distributed
- the heading originator
- keywords from subject, message type and (optionally) first text body
- addresses from the To: and Cc: fields (primary and copy recipients)
- primary precedence
- message instructions and handling instructions
- SIC list (note: the order is significant here)

## 30.4 Profiler Channel Configuration Files

### 30.4.1 Main Profiler Channel Configuration File

The Profiler Channel Rules are configured in `$(ETCDIR)/switch/profiler.json`. Examples of files which can be used are provided in:

`$(SHAREDIR)/switch/profiler/examples/profiler.json`

A simple Profiler Plan

`$(SHAREDIR)/switch/profiler/examples/profiler-complex1.json`

An example of a complex set of Profiler Plans

`$(SHAREDIR)/switch/profiler/examples/profiler-complex2.json`

An second example of a complex set of Profiler Plans

### 30.4.1.1 Plans

A **Plan** comprises:

- A **Name** which identifies it
- A **Description** for further information
- A set of **Rules**
- A boolean to indicate whether the plan is active or not

For file-based plans, the boolean could be implemented by the location in the filestore, or the name of the file rather than an explicit data item.

### 30.4.1.2 Structure of Rules

The order of the rules in the set is not of significance.

The data items for a rule are as below. Values are always strings. For precedence, either the numeric value or the name can be used. X.400 or SMTP addresses can be used, as the strong forms are distinguishable.

Simple strings are entered surrounded by double quotes, e.g. "value":"immediate"

Array strings are entered surrounded by double quotes, with an outer square bracket e.g. "days":["mon", "tue", "wed", "thu", "fri"]

Each **Rule** comprises (as a string unless otherwise stated):

#### Ident

A string identifying this rule

#### Description

A string describing this rule

#### Type

One of:

Local, Originator, Keyword, Address List, Out-of-Hours, Instructions, SIC

#### Exercise

Needed for SIC Type of **Rule** only.

#### Recipient Address

Optional for some **Types**.

#### Value

Not used for Local **Type**. Optional for Out-of-Hours **Type**

#### Info Addressees (Array of strings)

Target of redistributed message.

#### Action Addressees (Array of strings)

Target of redistributed message.

#### Stop/Not Stop

Ignored for Out-of-Hours **Type** (boolean).

**Schedule**

Only for Out-of-Hours **Type** (Array of schedules).

The schedule is a complex set of items, each of which is:

- A set of days of the week as array of strings for the days rule applies: 'sun', 'mon', 'tue', 'wed', 'thu', 'fri', 'sat', or 'all' If omitted, then applies to all days
- A start time of the form "hh:mm"
- A stop time of the form "hh:mm"

**30.4.1.3 Rule Type**

The list of possible **rule Types** is below.

---

**Note:** All but the SIC rules are called direct distribution rules.

---

Local

Applies to the recipient address always

Originator

Matches the *heading* originator

Keyword

Matches keywords from the message, found in the subject, message type and first text body. It may be necessary to distinguish these sources.

Address List

matching an address in the heading of the message which is some kind of AL.

Out-of-hours

Matches if the message has that precedence or higher. It has a schedule which specifies the day-of-week/time-ranges when it applies (could be always or never). Could have a precedence (defaulting to flash).

Instructions

these match handling and message instructions.

SIC

Multiple SIC values may be used separated by a semi-colon. The match can be a prefix match. SIC rules can be specific to an exercise.

**30.4.1.4 Exercises**

An **Exercise** comprises the following:

**Description**

A string identifying this rule

**Type**

The type: 'exercise', 'operation', 'project', 'drill' or integer value

**ident**

a string used in a message type for the exercise

**esic**

A string giving the SIC for the exercise etc

**info**

a array of strings giving informational DISTAFF addresses

**current**

a boolean indicating if the exercise is current

**plans**

array of plan objects giving rules associated with the exercise when it is current

### 30.4.2 Example Plan (simple)

The figure below shows a simple **Plan** comprising a set of 3 **rules** which form a **Plan**.

```
{
  "plans": [
    {
      "ident": "Plan0",
      "description": "This is Plan number 1",
      "active": true,
      "rules": [
        {
          "ident": "rule1",
          "type": "keyword",
          "value": "foo",
          "recip": "profiler1@example.com",
          "action": [
            "user1@example.com"
          ]
        },
        {
          "ident": "rule2",
          "type": "keyword",
          "value": "bar",
          "recip": "profiler2@example.com",
          "action": [
            "user1@example.com"
          ]
        },
        {
          "ident": "rule3",
          "type": "keyword",
          "value": "foobar",
          "recip": "profiler3@example.com",
          "action": [
            "user1@example.com"
          ]
        }
      ]
    }
  ]
}
```

### 30.4.3 Example Plan (advanced)

The figure below shows a simple **Plan** comprising an inactive Plan, and an Active Plan with multiple **rules**.

```
{
  "plans": [
    {
      "ident": "Plan0",
      "description": "Check that inactive plan not used",
      "active": false,
      "rules": [
        {
          "ident": "rule1",
          "type": "local",
          "recip": "",
          "action": [
            "wrong@tardis.isode.net"
          ]
        }
      ]
    }
  ]
}
```



```

    ]
  },
  {
    "ident": "Plan1",
    "description": "This is Plan number 1",
    "active": true,
    "rules": [
      {
        "ident": "rule1",
        "type": "local",
        "recip": "test1@tardis.isode.net",
        "action": [
          "r1a1@test.isode.net",
          "r1a2@test.isode.net"
        ],
        "info": [
          "r1i1@test.isode.net",
          "r1i2@test.isode.net"
        ]
      },
      {
        "ident": "rule2",
        "type": "originator",
        "recip": "test2@tardis.isode.net",
        "value": "dbw@tardis.isode.net",
        "action": [
          "r2a1@test.isode.net"
        ],
        "info": [
          "r2i1@test.isode.net"
        ]
      },
      {
        "ident": "rule3",
        "type": "keyword",
        "recip": "test3@tardis.isode.net",
        "value": "foo",
        "action": [
          "r3a1@test.isode.net"
        ],
        "info": [
          "R1i1@test.isode.net"
        ]
      },
      {
        "ident": "rule4",
        "type": "instructions",
        "recip": "test4@tardis.isode.net",
        "value": "leave in safe place",
        "action": [
          "r4a1@test.isode.net"
        ]
      },
      {
        "ident": "rule5",
        "type": "address-list",
        "recip": "test5@tardis.isode.net",
        "value": "aig1@tardis.isode.net",
        "action": [
          "mem1-1@test.isode.net",
          "mem1-2@test.isode.net"
        ],
        "info": [
          "mem1-3@test.isode.net",
          "mem1-4@test.isode.net"
        ]
      }
    ]
  }
]

```

```

    },
    {
      "ident": "rule6",
      "type": "out-of-hours",
      "recip": "test6@tardis.isode.net",
      "value": "flash",
      "schedule": [
        {
          "finish": "01:00"
        },
        {
          "days": ["all"],
          "start": "02:00",
          "finish": "03:00"
        },
        {
          "days": ["sat", "sun"],
          "start": "00:00",
          "finish": "24:00"
        }
      ],
      "action": [
        "dutyl@test.isode.net"
      ]
    },
    {
      "ident": "rule6a",
      "type": "out-of-hours",
      "recip": "test6@tardis.isode.net",
      "value": "immediate",
      "schedule": [
        {
          "days": ["mon", "tue", "wed", "thu", "fri"],
          "start": "03:00",
          "finish": "04:00"
        }
      ],
      "action": [
        "dutyl@test.isode.net"
      ]
    },
    {
      "ident": "rule6b",
      "type": "out-of-hours",
      "recip": "test6@tardis.isode.net",
      "value": "priority",
      "schedule": [
        {
          "days": ["mon", "tue", "wed", "thu", "fri"],
          "start": "04:00",
          "finish": "05:00"
        }
      ],
      "action": [
        "dutyl@test.isode.net"
      ]
    },
    {
      "ident": "rule6c",
      "type": "out-of-hours",
      "recip": "test6@tardis.isode.net",
      "value": "routine",
      "schedule": [
        {
          "days": ["mon", "tue", "wed", "thu", "fri"],
          "start": "05:00",

```

```

        "finish": "06:00"
    },
    ],
    "action": [
        "duty1@test.isode.net"
    ]
},
{
    "ident": "rule6d",
    "type": "out-of-hours",
    "recip": "test6@tardis.isode.net",
    "value": "override",
    "schedule": [
        {
            "days": ["mon", "tue", "wed", "thu", "fri"],
            "start": "06:00",
            "finish": "07:00"
        }
    ],
    "action": [
        "duty1@test.isode.net"
    ]
},
{
    "ident": "rule7",
    "type": "sic",
    "recip": "test1@tardis.isode.net",
    "value": "BBC",
    "action": [
        "r7a1@test.isode.net"
    ]
},
{
    "ident": "rule8",
    "type": "sic",
    "recip": "test1@tardis.isode.net",
    "value": "C*",
    "action": [
        "r8a1@test.isode.net"
    ]
}
]
}
]
}
]

```

# Chapter 31 Edge Channel

This chapter describes how the M-Switch Edge channels can be set up and configured to send MIME messages through an M-Guard. There are two distinct channels:

- The Edge Client: this is a dedicated output channel for converting MIME messages to an M-Switch Edge defined XML format, wrapping it in a GCXP wrapper and sending it to an M-Guard (or compatible service).
- The Edge Listener: this is a dedicated input channel for receiving GCXP wrapped messages from an M-Guard (or compatible client), unwrapping the contained XML message and converting it to MIME format for submission to a local Isode M-Switch message queue.

This is expanded on in the section [Section 31.1, “The M-Switch Edge channel”](#).

The aim of this feature is to enable users to send MIME messages through an M-Guard, subject to strict conditions. These include:

- Passing through of only specifically supported headers and message elements by the client sending side and ignoring all others. The supported elements and XML format are defined by an M-Switch Edge XSD.
- Further stripping out of the supported headers depending on configurable rules and content validation according to configurable rules by the client.
- Sending of the resultant message XML (in a GCXP wrapper) to an M-Guard using TLS. This ensures that only permitted clients can attempt to deliver messages via the M-Guard. Furthermore, the M-Guard acts as a one way system (a 'diode'). Only transmission errors are relayed back to the client. No subsequent processing or delivery information is returned to the client side. This ensures that there is no leaking of information (e.g. valid or invalid addresses) from the receiving side back to the sender. Transmission errors can be reported by the client, but all other delivery errors must be handled on the receiving side. This means that there will be manual intervention required on the receiving side in order to handle those errors.
- The M-Guard can be set up to apply strict rules as to what messages, parts of messages and specific content may be transmitted.
- The receiving side can receive and unwrap the GCXP messages to an XML encoded message that is validated against the M-Switch Edge XSD. This message XML is then converted to a MIME format and submitted to the local Isode M-Switch queue. The listener reports all errors to the local operator. No status is reported back to the sender. All error handling in this stage must be handled by the local operator, who will decide what action (including feedback to the sender, if any) is required.

---

## 31.1

## The M-Switch Edge channel

The M-Switch Edge channel is a separately activated feature.

The Edge channel is designed to interact with the Isode M-Guard. The M-Guard is a one way system designed to securely receive, check and deliver or reject messages according to pre-specified rules. The M-Switch Edge feature provides the ability to deliver MIME (email) messages via the M-Guard.

The M-Guard, its operation and the GCXP (Guard Content eXchange Protocol) communications protocol are detailed in the whitepaper [M-Guard: Isode's XML Guard](https://www.isode.com/whitepaper/m-guard-isodes-xml-guard/) [https://www.isode.com/whitepaper/m-guard-isodes-xml-guard/] (external link).

The Edge feature consists of two components:

- A client - the Edge Client that sends messages to the M-Guard.
- A service - the Edge Listener that receives messages from the M-Guard.

The Edge Client (the `guardout` process) acts as a delivery mechanism to the M-Guard. The Edge Channel is used to transfer a message, via a M-Guard, to a recipient which has been routed by a Routing Tree Entry to an MTA by the Edge channel.

The client reads the MIME message and builds a custom XML file representation (defined by a specific Isode M-Switch Edge XSD format). This is then (optionally) processed, wrapped in a GCXP wrapper and delivered to the M-Guard. No response is received from the M-Guard, except a simple successfully or unsuccessfully received status. All processing errors on the receiving side are handled on the receiving side and not reported back to the client.

The receiving side consists of the Edge Listener (the `isode.pp.guard` service). This listener receives the GCXP message from the M-Guard, extracts the XML and then regenerates the MIME message for delivery to the local Isode M-Switch queue manager.

Any errors on the receiving (Edge Listener) side of the M-Guard are not transmitted back to the sending (Edge Client) side. They are reported locally to the receiving (listener) side and will be managed by an operator on that side. This ensures that there is no leakage of information back to the sending side.

---

## 31.2 The M-Switch Edge Client

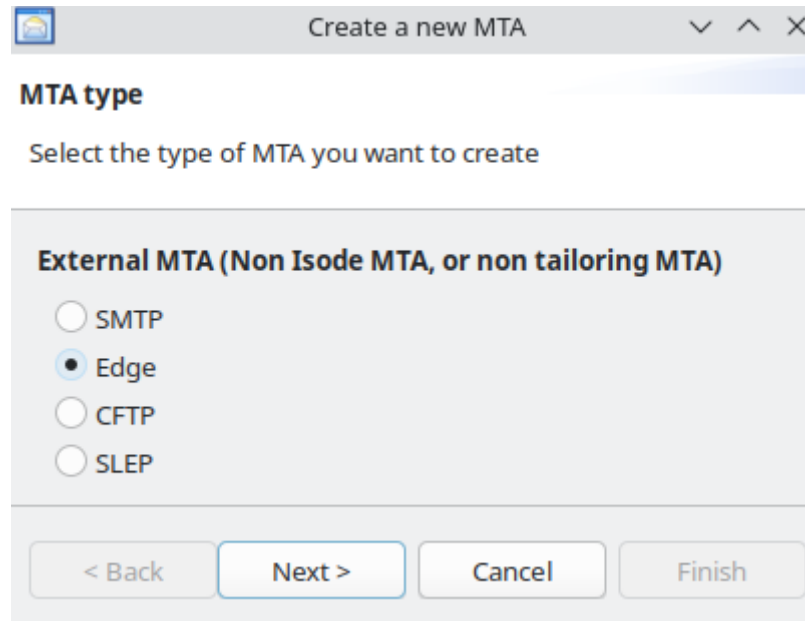
The M-Switch Edge Client receives MIME messages from the Isode queue server, parses them into an M-Switch Edge XML format, optionally parses and filters the message XML (to remove headers or check the existence or non-existence of certain headers, for example) and sends the result - wrapped in a GCXP wrapper - to an M-Guard (or compatible server).

### 31.2.1 M-Switch Edge Client setup

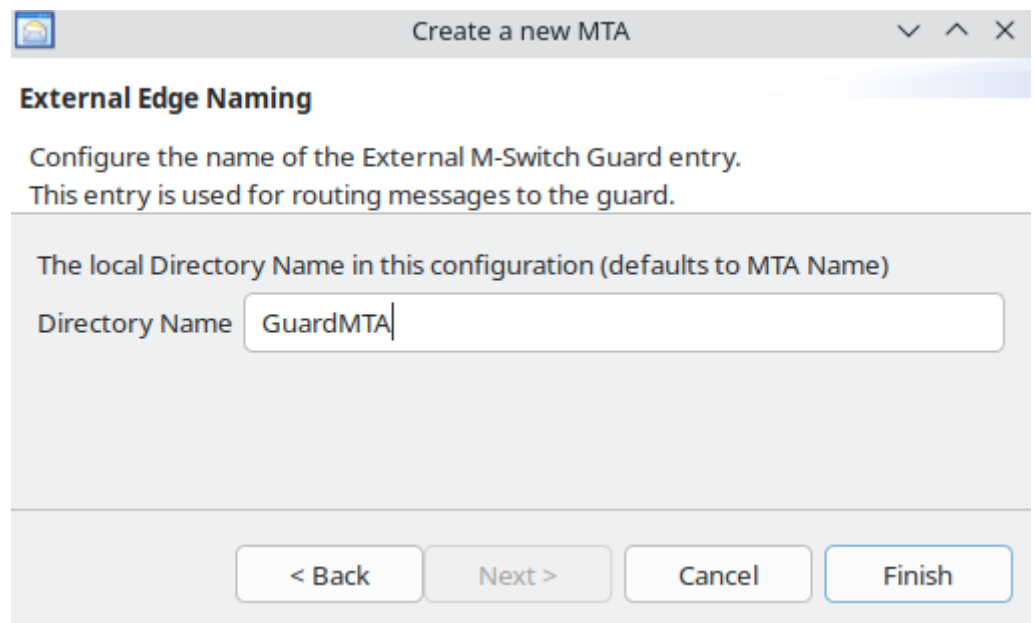
#### 31.2.1.1 Set up an Edge External MTA

To create a new Edge Client channel, an External Message Transfer Agent (MTA) needs to be set up.

To do this, start M-Console, and connect to your **Messaging Configuration**. Open a **Switch Configuration Management** view, and right click on **External Message Transfer Agents**. Finally, select **New External MTA....** This pops up the **Create a new MTA** dialog.

**Figure 31.1. 'Create a new MTA...' dialog**

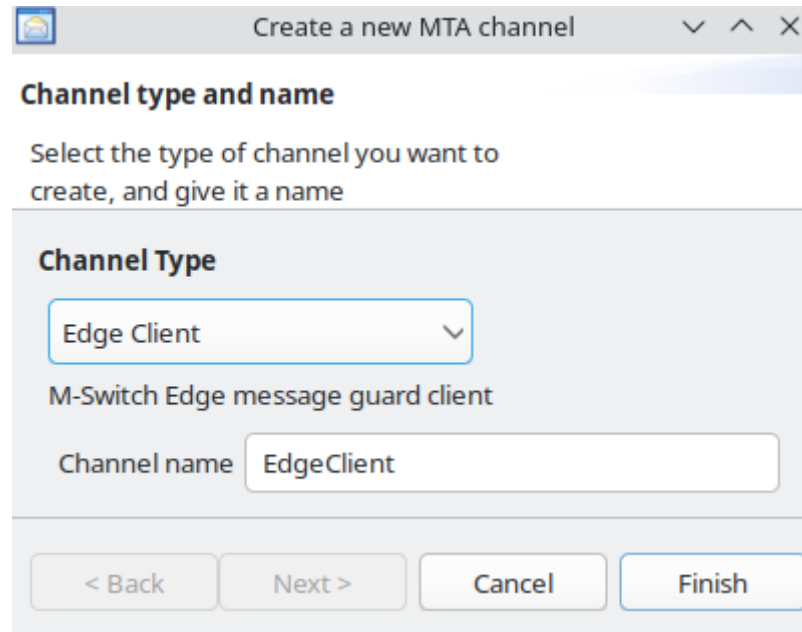
Select the **Edge** option and click on the **Next** button. Enter a name for the MTA and click on the **Finish** button.

**Figure 31.2. Naming the External Edge MTA**

### 31.2.1.2 Add an Edge Client

Now we have our Edge External MTA to which we can now add a client or clients.

In the **Switch Configuration Management** view, expand the tailoring (configurable) MTA you wish to associate the Edge Client with - in this example **acheron.isode.net** - then right click on the **Channels** item, and finally select **New Channel....** This pops up the **Create a new MTA Channel** dialog.

**Figure 31.3. 'Create a new MTA Channel' dialog**

**Channel type and name**

Select the type of channel you want to create, and give it a name

**Channel Type**

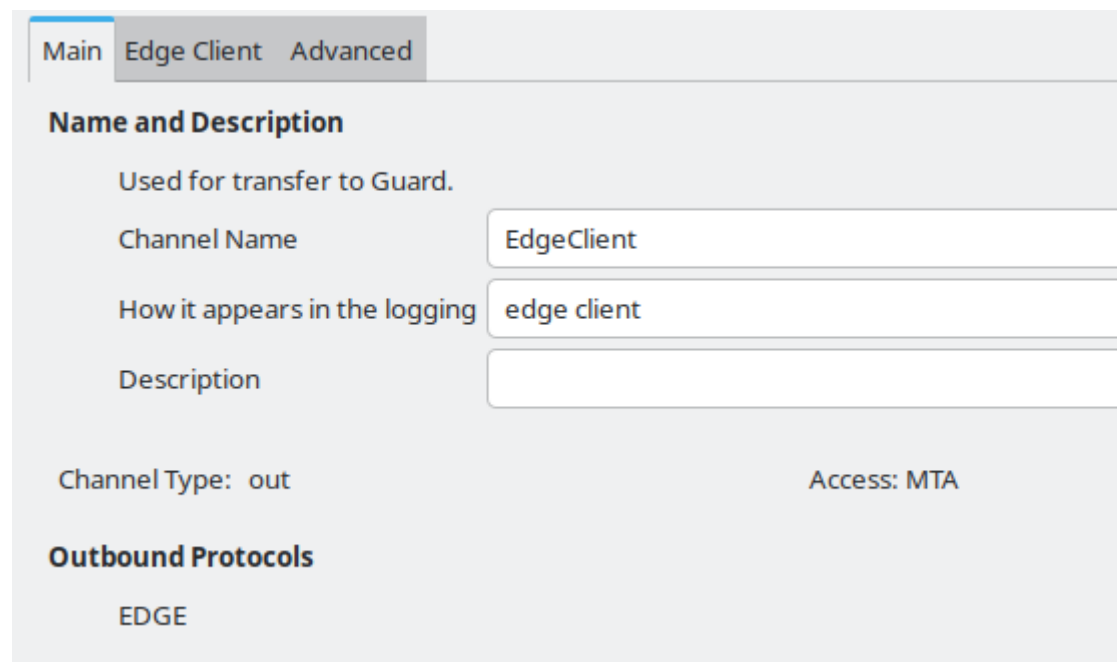
Edge Client

M-Switch Edge message guard client

Channel name EdgeClient

< Back Next > Cancel Finish

In this dialog, select **Edge Client** from the **Channel Type** drop down, and enter a name (alphanumeric only) in the **Channel name** text box. Press the **Finish** button and the channel is created. The Edge Client configuration screen is displayed.

**Figure 31.4. Edge Client Main configuration panel**

Main Edge Client Advanced

**Name and Description**

Used for transfer to Guard.

Channel Name EdgeClient

How it appears in the logging edge client

Description

Channel Type: out Access: MTA

**Outbound Protocols**

EDGE

In the **Main** panel, you can set the general options:

- The **Channel Name**. This is a unique alphanumeric string that identifies the channel.
- A logging string, which can be set to identify log lines generated by the Edge Client.
- A **Description** of the channel.

The next tab is the **Edge Client** panel.

**Figure 31.5. Edge Client configuration panel**

The screenshot shows a web interface with three tabs: 'Main', 'Edge Client', and 'Advanced'. The 'Edge Client' tab is selected. Below the tabs, there are four configuration sections:

- Guard Connection Information:** Contains a 'Guard host' text input field with the value 'localhost' and a 'Guard host port' text input field with the value '182'.
- TLS Config key:** An empty text input field.
- Process file:** An empty text input field.

The **Edge Client** configuration panel allows the user to set Edge Client specific parameters:

- The **Guard host** is either the DNS name or IP address of the M-Guard host.
- The **Guard host port** is the listening port for this Edge Client on the M-Guard.
- The **TLS Config key** is the name of the TLS configuration in the **Security Database** for the Edge Client connection to the M-Guard. The certificate (and chain) for this TLS configuration must be signed by the M-Guard, and the M-Guard signing certificate must be imported in the **Security Database** as a Trust Anchor (TA). This is detailed in [Section 31.5, “Edge TLS configuration”](#).
- The **Process file** option sets the process file to use. This file is described in [Section 31.2.5.1, “The Edge Client process file”](#). If no filename is set or no default file is found, no processing or validation will take place - the message XML will be sent as-is.

The final Edge Client configuration page is the **Advanced** panel. This allows the user to control general parameters that are discussed elsewhere.

## 31.2.2 M-Switch Edge Client parameters

The configuration options above result in the following parameters for the named channel in the *ETCDIR/switch/mtataylor.tai* file.

- *guard\_host* (required): the DNS hostname (resolvable) or dotted IPV4 address for the M-Guard.
- *guard\_port* (required): the incoming port number the M-Guard is listening on. The default is 18201.
- *process\_file* (optional): the process file for the outgoing XML (see [Section 31.2.5.1, “The Edge Client process file”](#)). The default is none, which means that if no default file is found, the XML is transferred as-is.
- *tls\_channel\_cfg* (optional): the TLS channel configuration. The default is none, so the default TLS configuration is used. This should be set up using a TLS configuration signed by the M-Guard (see [Section 31.5, “Edge TLS configuration”](#)).
- *verify\_tls* (optional/hidden): this parameter can only be set by hand-editing the *mtataylor.tai* file. If set to *true*, TLS verification will be bypassed. This is useful for testing, but should never be used in a production environment. This parameter will be removed every time the *mtataylor.tai* is regenerated.

## 31.2.3 M-Switch Edge Client configuration file

There is an additional configuration file that controls the processing of the Edge Client. This should not need to be configured by the user, but is listed here for completeness.



The file is installed in the *SHAREDIR/switch/* directory. If a customisation of the file is required, it should be copied to the *ETCDIR/switch/* directory and edited there. When the Edge Client starts up, it will first look for the customised file, then fallback to the installed one.

- *xmlguard-toxml.xml*: this is the shaper file that defines how the MIME to XML conversion is managed.

## 31.2.4 M-Switch Edge Client command line parameters

The Edge Client has two optional command line parameters. These are *debug* and *<channel name>*. The order is important.

- *debug* when the Edge Client executable is run in this mode, it will run in the foreground and present a command prompt for the user to manually interact with the application.
- *<channel name>* this is the name of the client channel set in the [Figure 31.4, “Edge Client Main configuration panel”](#) configuration panel. This allows the client to find its configuration in the *mtataylor.tai* file. The default is *EdgeClient*.

## 31.2.5 M-Switch Edge Client processing

Once the MIME message has been converted to XML, this message XML can be processed by the Edge Client before being sent to the M-Guard. This processing can be useful for checking the message XML against an XSD or stripping elements from the message XML, for example.

There are four types of operations that may be performed on the message XML.

### 31.2.5.1 The Edge Client process file

The process file is an XML file and may contain none or more of each of the four actions. Each listed action is performed sequentially in the order listed on the message XML. The final resulting message XML is then sent to the M-Guard.

The process file is defined by the *process\_file* option (see the section [Section 31.2.1, “M-Switch Edge Client setup”](#)). If this option is set, that file is looked for, and if not found the process exits. If the option is not set, the file *m-switch\_edge\_process.xml* is looked for in the *ETCDIR/switch*, and (if not found) in the *SHAREDIR/switch*. A default (no operations) file is installed with the distribution in *SHAREDIR/switch*. The first file found is then used. If neither is found, the program continues without any processing.

#### 31.2.5.1.1 Filenames in the processing file

File names referenced in the process file may either contain a path element (full or relative) or not. For the former, the exact location will be used. For the latter, without any path element, *ETCDIR/switch* then *SHAREDIR/switch* will be searched for the file. The first matching file will be used.

So, for example, the *./test.xslt* file will only be looked for in *./.*. The *test.xslt* file will be looked for in *ETCDIR/switch*, then *SHAREDIR/switch*. The first file found will be used.

#### 31.2.5.1.2 Process file XML validation

The last step in any processing should be validation of the final, processed XML. This needs to draw in two schemas: the Isode M-Switch Edge XSD and the NATO STANAG 4774 XSD. These are both installed in *SHAREDIR/switch* as *m-switch\_edge.xsd* and *stanag-4774.xsd*. A wrapper file for both these schemas is also supplied in *SHAREDIR/switch* as *-m-switch\_edge\_wrapper.xsd*.

Sample Edge Client processing XML file

```
<?xml version="1.0" encoding="UTF-8"?>
<actions>
  <verify mode="xslt" file="/opt/mysetup/etc/switch/step1.xslt"/>
  <verify mode="xslt" file="m-switch_edge_filter_headers.xslt"/>
  <verify mode="xpath" path="/descendant::processing-instruction()"/>
  <verify mode="xpath" path="/descendant::comment()"/>
  <verify mode="schematron" file="otherheaders.sch"/>
  <verify mode="schema" file="m-switch_edge_wrapper.xsd"/>
</actions>
```

### 31.2.5.2 The Edge Client processing actions

The actions that may be performed are:

- *xslt* the name of an XSLT file to run against the message XML.

Sample Edge Client processing XSLT file. This copies all elements, except for processing instructions, comments and Date and From elements (these are ignored and not passed through).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:mse="http://isode.com/m-switch/edge/0">
  <!-- Keep indentation -->
  <xsl:output method="xml" encoding="utf-8" indent="yes"/>
  <!-- Identity template : copy all text nodes, elements and attributes -->
  <xsl:template match="@*|node()">
    <xsl:copy>
      <xsl:apply-templates select="@*|node()" />
    </xsl:copy>
  </xsl:template>
  <!-- Remove empty lines -->
  <xsl:strip-space elements="*" />
  <!-- Remove empty text elements -->
  <xsl:template match="*[not(@*|*) and normalize-space()=' ']" />
  <!-- Remove comment and processing instruction elements -->
  <xsl:template match="*(comment()|processing-instruction())" />
  <!-- When matching Date: do nothing -->
  <xsl:template match="mse:Date" />
  <!-- When matching From: do nothing -->
  <xsl:template match="mse:From" />
</xsl:stylesheet>
```

- *xsd* the name of an XSD file to validate the message XML against. There should normally only be one of these entries. If the XSD validation fails, the sending of the message fails. This should be the last action performed, so that the actual resultant message XML that is to be transmitted is validated.
- *schematron* the name of a schematron file to perform customised validation.

Sample Edge Client processing Schematron file. If an OtherHeader element is found, an error is generated, processing fails and the message is not sent.

```
<schema xmlns="http://purl.oclc.org/dsdl/schematron">
  <pattern name="Check no OtherHeader present">
    <rule context="TransferredMessage/MessageAndEnvelope/Message/Heading"
```

```

        name="OtherHeader-Message">
        <assert test="count(OtherHeader) = 0">contains an OtherHeader</assert>
    </rule>
</pattern>
</schema>

```

- *xpath* an XPath statement to validate elements in the message XML. This entry may have *min* and *max* attributes (the default in both cases is 0). If the number of elements found by the *xpath* statement falls outside this range, then processing fails and the message is not sent.

Sample Edge Client processing XPath instructions. These look for processing instructions and comments, and if any are found (*min* and *max* in both cases are unset, so default to zero), an error is generated, processing fails and the message is not sent.

```

<verify mode="xpath" path="/descendant::processing-instruction()" />
<verify mode="xpath" path="/descendant::comment()" />

```

## 31.2.6 Edge Client Error handling

The Edge Client will halt processing on any error. The message will not be sent and an error message will be logged to the MTA log file.

A `Message Error` will be set, and the message retried after a period of backup allowing the admin to reconfigure to prevent the error. This will show up in the M-Switch **Operations View** in `mconsole`.

---

## 31.3 The M-Switch Edge Listener

The M-Switch Edge Listener receives GCXP wrapped XML messages from the Isode M-Guard (or a compatible client), validates the XML against the M-Switch Edge XSD, parses them into a MIME format and submits them to the local Isode M-Switch message queue.

### 31.3.1 M-Switch Edge Listener setup

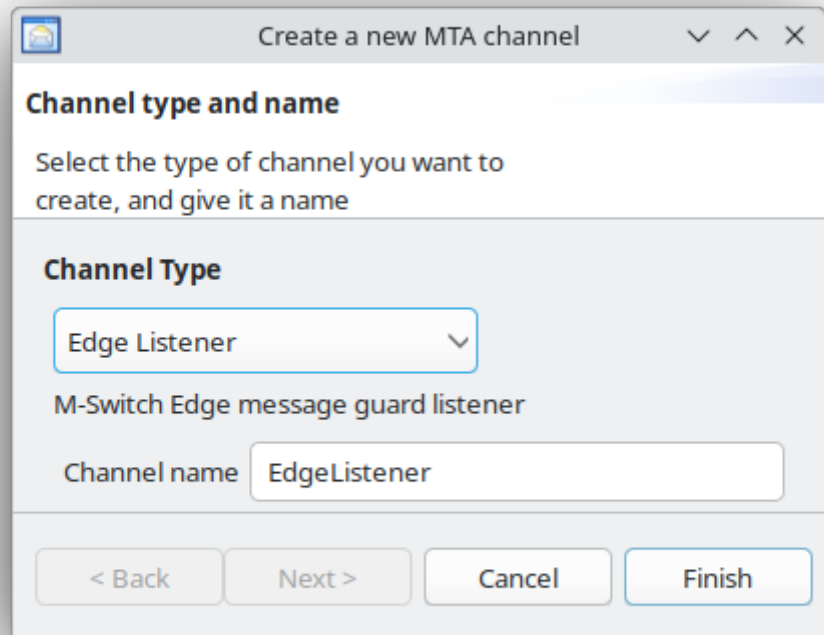
#### 31.3.1.1 Set up an Edge External MTA

This needs to be set up as per the section [Section 31.2.1.1, "Set up an Edge External MTA"](#).

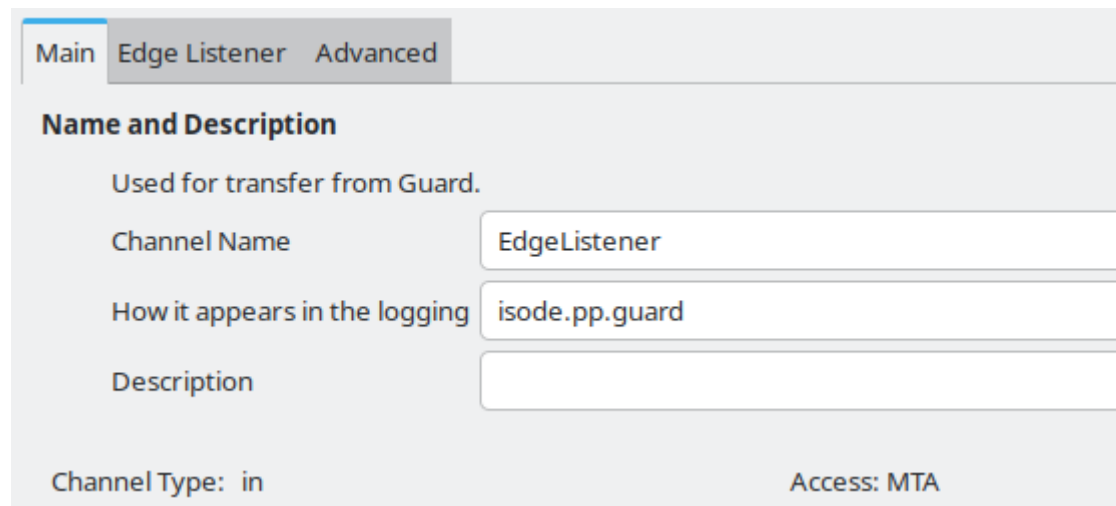
#### 31.3.1.2 Add an Edge Listener

Now we have our Edge External MTA to which we can now add a listener or listeners.

In the **Switch Configuration Management** view, expand the tailoring (configurable) MTA you wish to associate the Edge Listener with - in this example **acheron.isode.net** - then right click on the **Channels** item, and finally select **New Channel....** This pops up the **Create a new MTA Channel** dialog.

**Figure 31.6. 'Create a new MTA Channel' dialog**

In this dialog, select **Edge Listener** from **Channel Type** drop down, and enter a name (alphanumeric only) in the **Channel name** text box. Press the **Finish** button, and the channel is created. The Edge Listener configuration screen is displayed.

**Figure 31.7. Edge Listener Main configuration panel**

In the **Main** panel, you can set the general options:

- The **Channel Name**. This is a unique alphanumeric string that identifies the channel.
- A logging string, which can be set to identify log lines generated by the Edge Listener.
- A **Description** of the channel.

The next tab is the **Edge Listener** panel.

**Figure 31.8. Edge Listener configuration panel**

The screenshot shows a web-based configuration interface for the Edge Listener. At the top, there are three tabs: 'Main', 'Edge Listener' (which is selected and highlighted with a blue border), and 'Advanced'. Below the tabs, the section is titled 'Server Connection Information'. It contains three input fields: 'Server address' with the value '0.0.0.0', 'Server port' with the value '182', and 'TLS Config key' which is currently empty.

The **Edge Listener** configuration panel allows the user to set Edge Listener specific parameters:

- The **Server Address** is the IP address to listen on. The default is 0 . 0 . 0 . 0 (all IP devices).
- The **Server Port** is the listening port for this Edge Listener for the M-Guard or client to connect to.
- The **TLS Config key** is the name of the TLS configuration in the **Security Database** for the M-Guard connection to the Edge Listener. The certificate (and chain) for this TLS configuration must be signed by the M-Guard, and the M-Guard signing certificate must be imported in the **Security Database** as a Trust Anchor (TA). This is detailed in [Section 31.5, “Edge TLS configuration”](#).

The final Edge Listener configuration page is the **Advanced** panel. This allows the user to control generalised parameters that are discussed elsewhere.

### 31.3.2

## M-Switch Edge Listener parameters

The configuration options above result in the following parameters for the named channel in the *ETCDIR/switch/mtataylor.tai* file.

- *port* (optional): the port number the Edge Listener will listen on for the M-Guard to connect to. The default is 18201. Each Edge Listener must have its own dedicated port.
- *address* (optional): the address the Edge Listener will listen on. The default is 0 . 0 . 0 . 0 (all local IP devices).
- *tls\_channel\_cfg* (optional): the TLS channel configuration. The default is none, so the default TLS configuration is used. This should be set up using a TLS configuration signed by the M-Guard (see [Section 31.5, “Edge TLS configuration”](#)).
- *verify\_tls* (optional/hidden): this parameter can only be set by hand-editing the *mtataylor.tai* file. If set to *true*, TLS verification will be bypassed. This is useful for testing, but should never be used in a production environment. This parameter will be removed every time the *mtataylor.tai* file is regenerated.

### 31.3.3

## M-Switch Edge Listener configuration files

There are two additional configuration files that control the processing of the Edge Listener. These should not need to be configured by the user, but are listed here for completeness.

Each file is installed in the *SHAREDIR/switch* directory. If a customisation of the file is required, it should be copied to the *ETCDIR/switch/* directory and edited there. When the Edge Listener starts up, it will first look for the customised file, then fallback to the installed one.

- *xmlguard-fromxml.xml*: this is the shaper file that defines how the XML to MIME conversion is managed.

- *m-switch\_edge.xsd* this is the M-Switch Edge XSD file to validate incoming message XML against.

### 31.3.4 M-Switch Edge Listener command line parameters

The Edge Listener has three optional command line parameters. These are *debug*, *-c <channel name>* and *-p <port>*. The *debug* parameter must be first, if set.

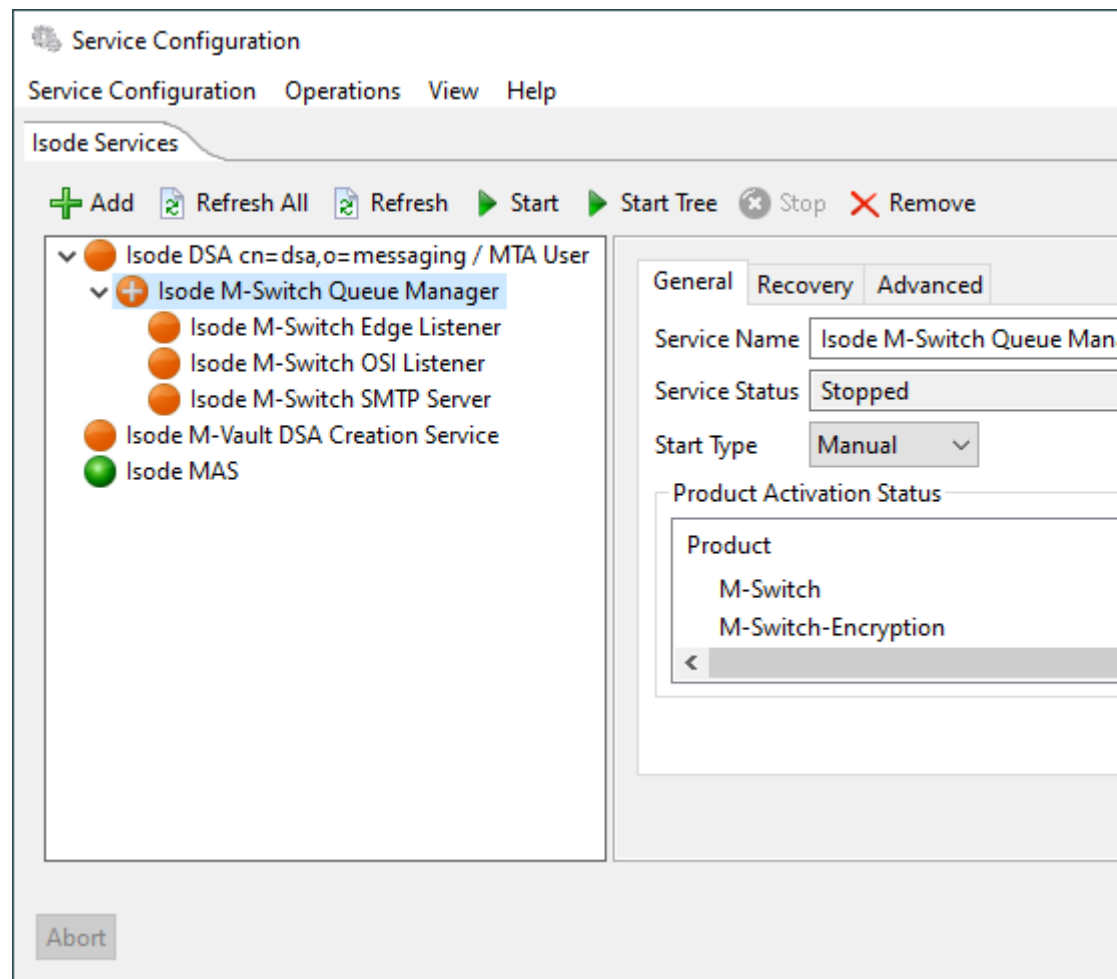
- *debug* when the Edge Listener executable is run in this mode, the listener will run in the foreground.
- *-c <channel name>* this is the name of the listener channel set in the [Figure 31.7, “Edge Listener Main configuration panel”](#) configuration panel. This allows the listener to find its configuration in the *mtataylor.tai* file.
- *-p <port>* this is port for the listener to listen for client connections on. This value overrides the value in the *mtataylor.tai* file. This port must be dedicated to the Edge Listener instance.

### 31.3.5 Edge Listener services

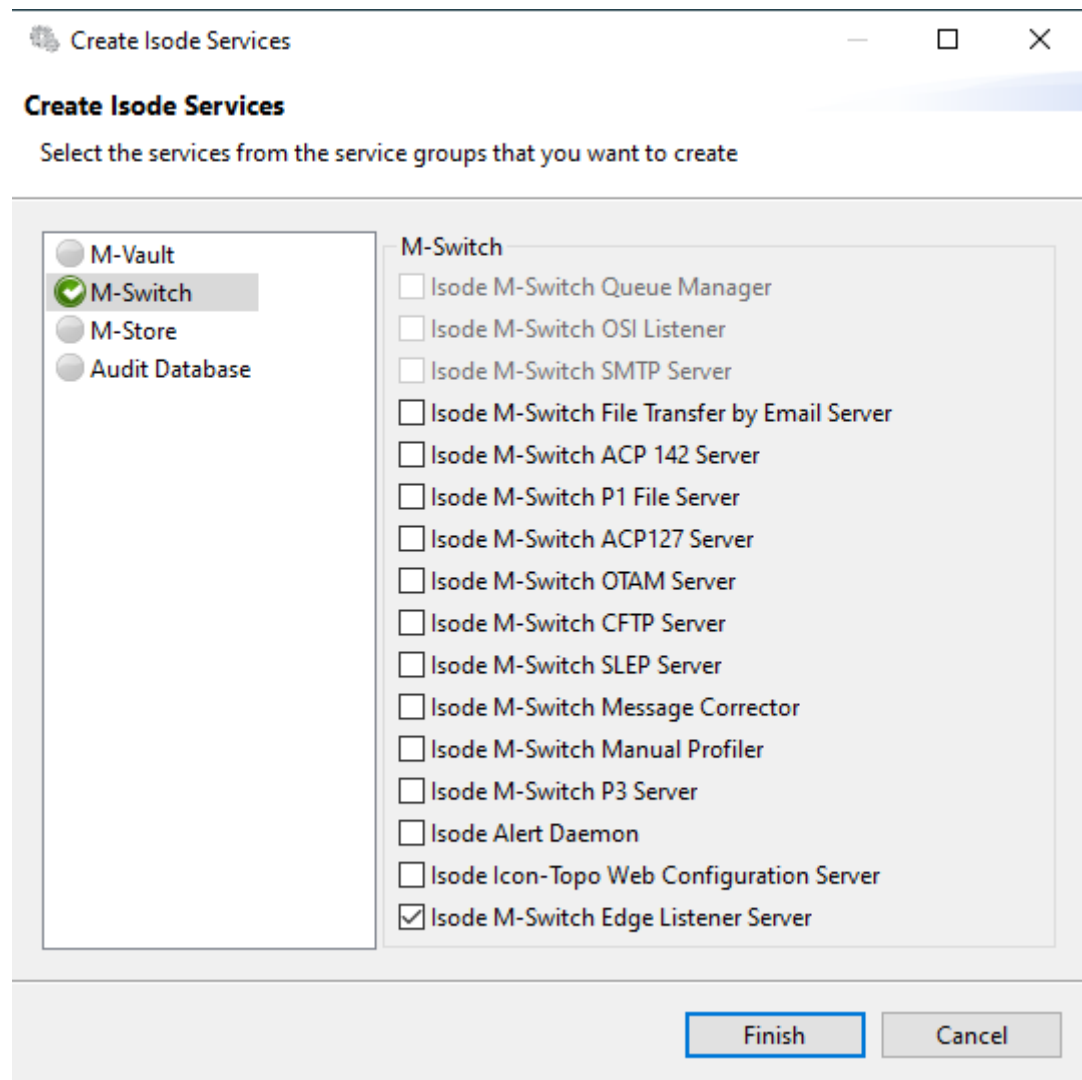
Since the Edge Listener runs as a service, it needs to be started up in order to listen on its port. For Linux and Windows, each listener needs to be configured using the relevant platform method.

#### 31.3.5.1 Edge Listener Service Windows configuration

For MS-Windows, the Isode Service Configuration tool should be used. Start it and highlight the **Isode M-Switch Queue Manager** service. The Edge Listener should depend on this service.

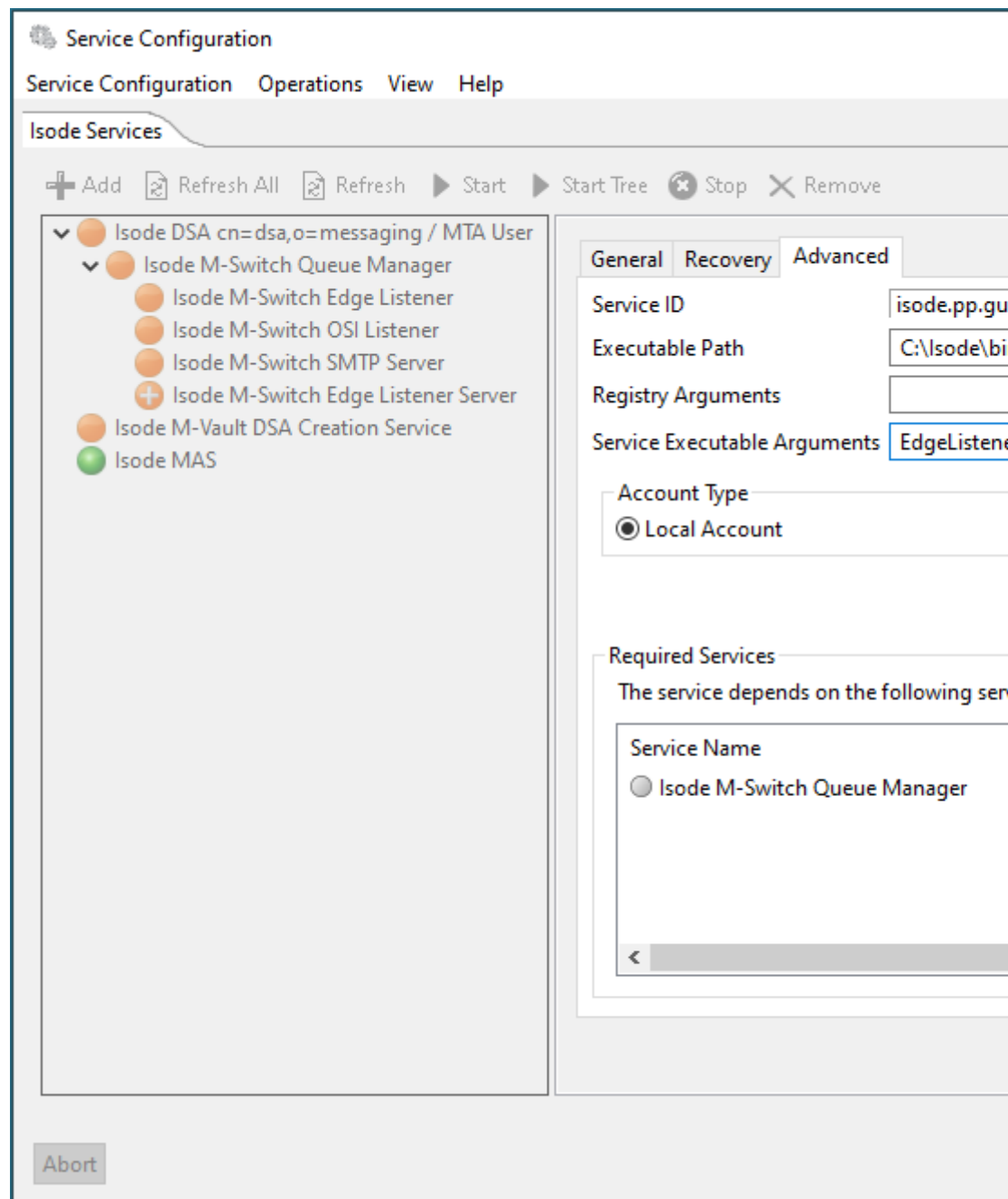
**Figure 31.9. Isode Service Configuration dialog**

Once the queue manager service is highlighted, press the **Add** button to pop up the **Create Isode Services** dialog. Select the **M-Switch** family of services in the lefthand pane, then tick the **Isode M-Switch Edge Listener Service** checkbox that appears in the righthand pane. Finally press the **Finish** button to create the service.

**Figure 31.10. Isode Service Configuration Create Isode Services dialog**

Once the Edge Listener service is created, it needs to be associated with the channel. Click on the newly created service in the main dialog window to select it, and navigate to the **Advanced** tab. Enter the channel name that this service instance will be handling, and then press the **Apply** button.



**Figure 31.11. Isode Service Configuration Advanced configuration tab**

The newly created service may now be managed alongside the other services.

### 31.3.5.2 Edge Listener Service Linux configuration

For Linux, edit the *ETCDIR/pp.rc* file (copying from *ETCDIR/pp.rc.sample* if necessary) and uncomment the `EDGE_LISTENERS` line. Enter the channel name(s) for the service(s) you want started, in double quotes, with each channel name separated by a space.

Editing the *ETCDIR/pp.rc* file.

```
EDGE_LISTENERS="EdgeListener1 EdgeListener2"
```

The Edge Listeners may now be controlled by the `pp` script.

### 31.3.6 Edge Listener Error Handling

The Edge Listener has two levels of error.

- Invalid XML received: if the XML received cannot be parsed, the received message is emailed to the local postmaster, as nothing can be done with it.
- All other errors: these are logged to the MTA log files.

---

## 31.4 Setup for testing

The Edge client and listener can be set up back-to-back for easier testing without an M-Guard. This can be useful for rapid or offline configuration and testing of processing rules, for example.

To do this, set up the external Edge MTA as detailed in [Section 31.2.1.1, “Set up an Edge External MTA”](#). Then set up the Edge listener and client, setting the client **Guard host** and **Guard host port** to be the host and port the listener is running and listening on.

The client and server may be run on the the same machine and the client run in interactive debug mode (using the *debug* command line option) for a simple setup.

---

## 31.5 Edge TLS configuration

Setting up a TLS configuration with a Trust Anchor (TA) is described in [Section 14.1.6, “Security settings”](#).

For the Edge Client and Edge Listener, the certificates must be signed by the M-Guard, as the M-Guard will refuse TLS connections not signed by itself.

There are two main ways for the user to create a TLS configuration:

- The user can generate Private Key (PK) and Certificate Signing Request (CSR) and sign the latter with the M-Guard signing certificate. The resulting certificate chain and PK can then be imported by going to the **Security Database Certificates** panel in M-Console and using the **Import** button to pop up the import dialog, selecting the **TLS Configuration** option.
- The user can use the **Create** button in the above panel to create the PK and CSR. The CSR must be sent to the M-Guard for signing and then imported.

In both cases, it is important that the M-Guard signing certificate is imported as a Trust Anchor (TA) and it (and the signed certificates) are wrapped up in a named TLS configuration. This name is the value used for the TLS configuration parameters for the Edge Client and Edge Listener. Note that the client and listener may have different PKs (they will most likely be in different domains), but will both have the same TA, as they both need to be signed by the intermediate M-Guard.

Figure 31.12. Example Security Database with an M-Guard TLS certificate and TA

Security Database Configuration			
Security Database		Certificates	
TLS Configuration	Role	Subject Directory Name	DNS Name
default	PKCS12 Identity	cn=acheron.isode.net	acheron.isode.net
	Trust Anchor	cn=Single Use CA,cn=Messaging	
red_tls_cfg	PKCS12 Identity	o=M-Guard,cn=red.red.net+cn= red.net red	
	Trust Anchor	o=M-Guard	

# Chapter 32 Message Switch Console

Message Switch Console (MConsole) is a graphical management tool that is used to manage multiple aspects of M-Switch.

---

## 32.1 Overview

As MConsole uses a client/server architecture you can install MConsole on any supported platform to connect to and manage the Isode M-Switches, either locally or remotely.

On Linux, to start MConsole, ensure (*BINDIR*) is included in your path and type `mconsole`

On Windows, to start MConsole, use the shortcut in **Start** → **Program Files** → **Isode**.

On Windows, the configuration for MConsole is stored in the Windows Registry.

When MConsole first starts, the window displayed is empty. Select a **View** to get started.

---

## 32.2 MConsole multi windows – multi view user interface

The MConsole framework for displaying information is to provide views in a tabbed series of windows. Different views are available. There can be broken down into the following groups .

- Live Operations

These are Views used by operators in order to manage Isode messaging systems in real time. Unless otherwise noted, these are documented in the .

- **Summary View.** This provides an overview of the configure M-Switch MTAs.

See [Section 32.6, “MConsole Switch Operations Configuration”](#) for a description of how to configure use the Summary View.

See the [M-Switch Operators Guide](#) for a description of how to use the Summary View.

- **Switch Operations.** This provides realtime management and monitoring of one or more M-Switch MTAs.

See [Section 32.6, “MConsole Switch Operations Configuration”](#) for a description of how to configure the Switch Operations View.

See the [M-Switch Operators Guide](#) for a description of how to use the Switch Operations View.

- **Vetting View.** This enables messages which have been placed in a "held" state (as a result of Authorization Rules) to be released or non-delivered.

See the [M-Switch Operators Guide](#) for a description of how to use the Vetting View.

- **X.400 Message Store Operations.** Provides realtime management and monitoring of one or more M-Store X.400 Message Stores.

See the description of how Operators can use this View.

- **ACP127 View.** This provides detailed information on ACP127 circuits and transfers.

See the [M-Switch Operators Guide](#) for a description of how to use the ACP127 View.

- **ACP142 Message Transfer View.** This provides information on ACP142 message transfers.

See the [M-Switch Operators Guide](#) for a description of how to use the ACP142 View.

- **Channel Monitor View.** This provides information on the various channels, and message progression through the MTA.

See the [M-Switch Operators Guide](#) for a description of how to use the Channel Monitoring View.

- **Diversions.** Configure preconfigured alternative Routing Configurations offered by the Nexus facility generated by different M-Switch components.

See the [M-Switch Operators Guide](#) for a description of how to use the Diversions View.

- Configuration

These are Views used by Administrators in order to configure and manage Isode messaging systems. These are documented here.

- **Switch Configuration Management.** [Section 32.5, “Switch Configuration Management view”](#).
- **X.400 Mailbox Management.** [Section 11.1, “X.400 messaging users and White Pages Entries”](#).
- **Gateway Users.** [Section 19.6.1, “Setting up Routing with Gateway Users”](#).
- **Authenticated Entities Management.** [Section 13.3.5, “SASL User Management”](#).
- **ACP127 Addresses..** [Section 19.6.4, “ACP127 Addresses View”](#).

- Audit Information

These are Views used by operators in order to manage Isode messaging systems in real time by displaying information about messaging events in one or more Isode MTAs. These are documented in

- **Message History.**

See the [M-Switch Operators Guide](#) for a description of how to use the Message History View.

- **Message Tracking.**

See the [M-Switch Operators Guide](#) for a description of how to use the Message Tracking View.

- **Quarantine Tracking.**

See the [M-Switch Advanced Guide](#) for a description of how to use the Quarantine Tracking View.

- **Acknowledgement Tracking.**

See the [M-Switch Advanced Guide](#) for a description of Acknowledgement Tracking and how to use the View.

- **Message Transfers History.**

See the [M-Switch Operators Guide](#) for a description of how to use the Message Transfers History View.

- **Statistics.**

See the [M-Switch Operators Guide](#) for a description of how to use the Message Transfers History View.

- **Miscellaneous**

These are Views which provide ways to configure and navigate in MConsole itself.

- **User Agent.** See the [M-Switch Administration Guide](#) for a description of using the User Agent to forward messages. See [Section 32.8, “User Agent Configuration”](#) for a description of configuring the User Agent

- **Event Viewer.** Review or Monitor Events generated by different M-Switch components.

See the [M-Switch Operators Guide](#) for a description of how to use the Event Viewer.

- **Alerts.** Monitor Alerts generated by different M-Switch components.

See the [M-Switch Operators Guide](#) for a description of how to use the Alerts View.

- **Welcome View**

This is documented in [Section 32.3, “Welcome View”](#).

- **Options**

This is documented in [Section 32.4, “Options View”](#).

- **Shaper Configuration File Editor**

Allows an operator to configure the XML based shapper configuration files.

To display a view, click **View** and select the view you wish to display. Some views are greyed out until you have configured them. You do this by bringing up the **Options** which allows views to be configured.

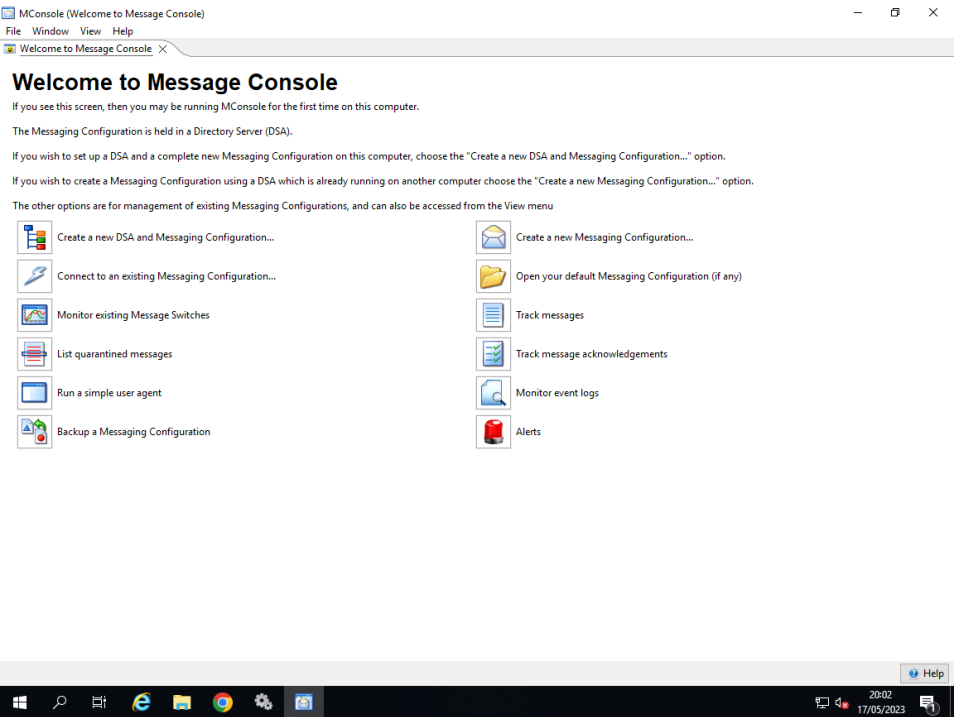
A view can be detached into a new window which opens up a new invocation of MConsole by right clicking the **View's** tab or by dragging the view outside the MConsole window.

---

## 32.3 Welcome View

The **Welcome View** provides the initial view when MConsole is first started. It provides jumping off points for other views.

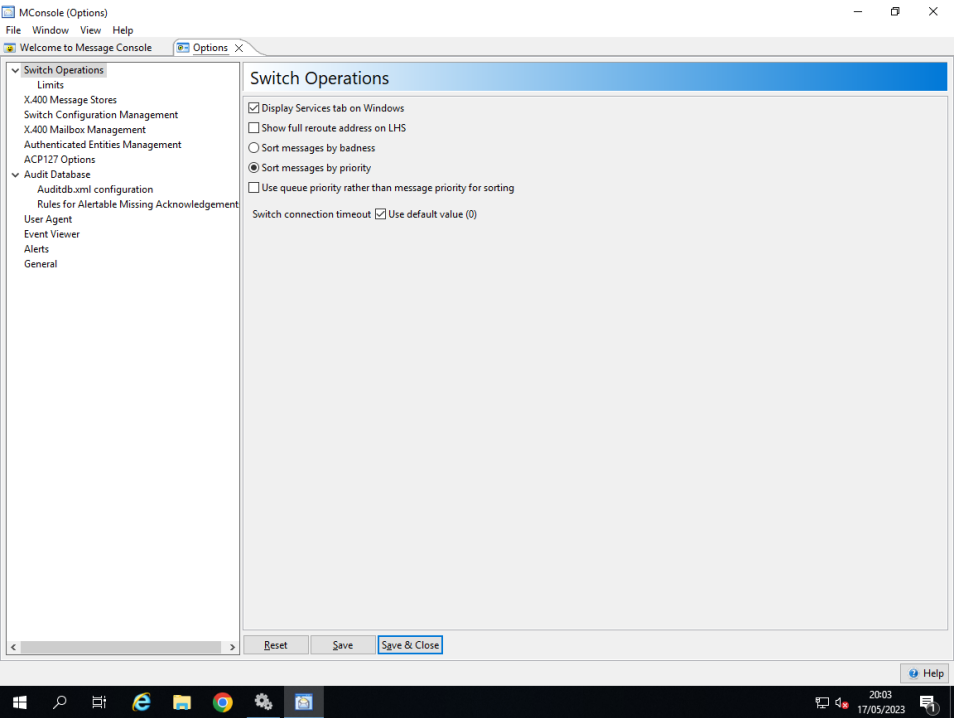
Figure 32.1. Welcome view



# 32.4 Options View

The **Options View** provides a way to configure options which control how others views are presented and work within MConsole.

Figure 32.2. Options View



## 32.5 Switch Configuration Management view

This is described in detail in

- Internet: See the [Chapter 5, Configuring an Internet Messaging System](#) for the chapter on Internet configuration.
- X.400: See the [Chapter 9, Configuring an X.400 Messaging System](#) for the chapter on X.400 configuration.
- MIXER: See the [Chapter 12, Configuring a MIXER Messaging System](#) for the chapter on MIXER configuration.

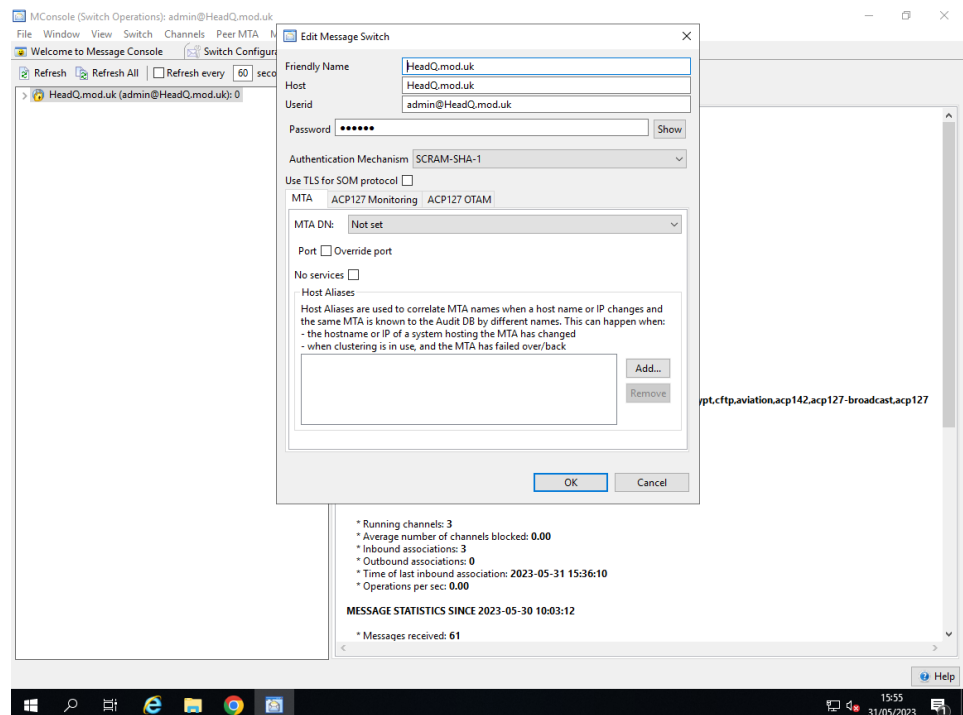
## 32.6 MConsole Switch Operations Configuration

This section describes how to configure the Switch Operations View. For a description of how to operate and use this view, see the .

Each **Switch** view can display information about one or more switches (MTAs).

You can monitor a **Switch**, display a **Switch Operations** view and add a new **Switch** either by right clicking on the empty window, or clicking on the **Switch** menu. This brings up the menu in [Figure 32.3, “Configuring the Switch Operations view”](#).

**Figure 32.3. Configuring the Switch Operations view**



You need to specify the:

- **Host** name (or IP address) of the host



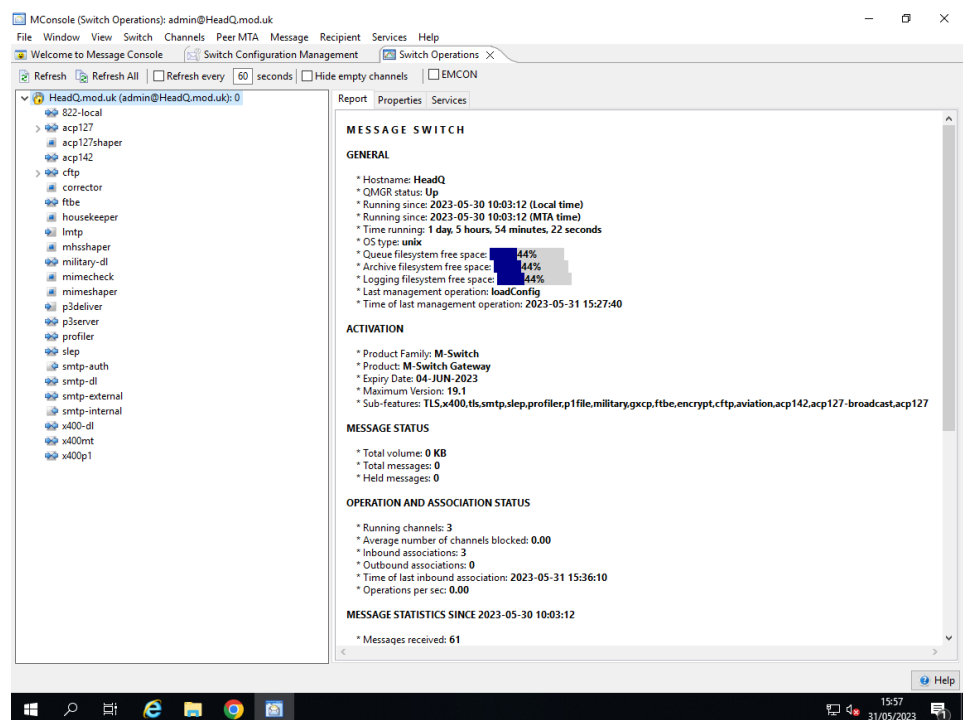
- **Userid**
- **Password.**
- **Authentication Mechanism.**

The Userid is a SASL Identity, as configured using the **Authenticated Entities Management** view (see [Section 13.3.5, “SASL User Management”](#)), and the password is that configured for the Identity. You will need to use a SASL Identity if you wish to perform administration functions (for example, deleting messages, disabling channels) as opposed to simply monitoring the Switch.

Aliases are required when a host is known by alternative names. This can occur, for example, if using clustering (see [Chapter 37, Clustering](#) for a description of this). In such a case you must enter all aliases so that MConsole features work correctly.

The **No services** button can be used to prevent an MConsole instance which is running on a Windows system from attempting to connect to the Service Control Manager on the Windows system on which the M-Switch instance is running. When a connection cannot be established (e.g. because of lack of a trust relationship between the two systems or because MConsole is not being run with Administrator rights when it is colocated with M-Switch), connection timeouts and error dialogs may occur. Checking the "No services" button prevents MConsole from making any connection attempt.

**Figure 32.4. Simple Switch Operations view**



## 32.7 Alerts Configuration

Since the Alerts View is highly configurable, it has introduced two new concepts: the Alert Action and the Alert Group.

## 32.7.1 Alert Actions

An alert action is something that MConsole is able to do to alert the user. There are currently 4 pre-configured alert actions:

- Display in Alerts View
- Play a sound
- Display a temporary notification
- Raise the MConsole window

### Display in Alerts View

This action just displays the given alert event in the Alerts View. Clearly, it is the least important action one can configure.

### Play a Sound

There are currently two sounds that can be played, the “*Error*” and the “*Alarm*” sounds. It is possible to configure how many times a sound is played.

### Display a temporary notification

This action shows the alert on a temporary window that appears on the bottom right hand side of the screen similar, for example, to the ones used by programs to notify about the reception of a new email.

### Raise the MConsole window

In case the MConsole window is minimized or obscured by another window, this alert action raises the MConsole window.

## 32.7.2 Alert Groups

The Alert Group is a new concept, in which a number of Alert Actions are grouped, so they can easily be assigned to an Isode Event.

The Alert Groups can be defined and modified by the administrator, but a number of Alert Groups are created by default.

**Table 32.1. Alert Groups**

Ignore Alert	None, the alert is completely ignored
No Alert	Display in Alerts View
Alert 1	Display in Alerts View. Play Error sound once
Alert 2	Display in Alerts View. Play Error sound once
Alert 3	Display in Alerts View. Play Error sound once. Raise the window
Alert 4	Display in Alerts View. Play Alarm sound 5 times Raise the window Display a temporary notification for 4 seconds
Alert 5	Display in Alerts View. Play Alarm sound until acknowledged Raise the window Display a temporary notification for 4 seconds

## 32.7.3 Editor for Alert Groups

Although this is unlikely to be required by most customers, it is possible to change the configured actions for the Alert Groups. If you want to change the behaviour of MConsole alerts for a given event, you should change the alert configuration for that event, and not the configuration of the Alert Group.

You can open the Alert Groups Editor by using the toolbar button. Alert Groups can be added, modified or removed.

The modified Alert Groups are stored along with the locally modified Alerts in the *alarm-config.ser* file, not editable except by MConsole.

## 32.7.4 Alert Colours

The alerts that are shown in the Alerts Table have a background colour and a foreground colour. The foreground colour is different for every Alert Group, the background colour is yellow only for alerts of levels 4 and 5, in order to highlight the more important alerts.

## 32.7.5 The Alert Configuration Editor

In order to configure Alert Actions, click on the **Alert Configuration Editor** button in the toolbar.

The default alert action for an event is taken from the Isode Event Catalogue. It reads the attribute "*alertgroup*". If it is missing, the event will not be shown in the Alerts View. If it is present, the value will be used as a default alert action.

Isode Events with default values for "*alertgroup*" cannot be removed from the configuration, but they can be set to use the default value.

The administrator is able to select events from the catalogue and choose non-default alert actions.

### 32.7.5.1 The Search Catalogue widget

Since there are many events in the Isode Event Catalogue, it is possible to search the events that match the description by using the filter that is in the box called **Search Catalogue**, as this makes it easier to find the event in question.

### 32.7.5.2 The Commands

On the right-hand side of the Alert Configuration Editor there is a panel with four commands: Add Alert, Edit, Remove and Set Default.

To add an alert, click on the **Add Alert** button. This will open the **Add Alert** window with all the known error messages in the Isode Message Catalogue, sorted by Facility.

It is possible to sort the Alert Wizard table by Alert Code, Facility, Description and Groups. It is also possible to search the known events by typing in the Search box.

Once an event is selected, click on the **OK** button to add it to the Alert Configuration Table. You will notice that in the Alert Configuration Table there is a column called "Actions". A default action has been added to the event that you selected, but this can be edited. It is possible to select and add several events at the same time.

In order to edit an alert, either double-click on it or select the alert from the Alert Configuration Table and click on the **Edit** button to open the Alert Editor. The Alert Editor allows you to map what to do for a given Isode Event.

If you have changed the alert group associated with an event and want to reset it to the default value, click on the **Set Default** button.

If you have manually added an event to the Alert Configuration and want to remove it, click on the **Remove** button. Note that it is not possible to remove the events that have the default value, as it makes no difference.

If you want to prevent an event from being shown in the Alerts View, the way to do this is to edit the event and set the Alert Group to either **Ignore Alert**. If you want it to still appear in the Alerts View but want it to cause no action, set the Alert Group to **No Alert**.

## 32.7.6 Alert Page in the Options View

The Alerts page of the Options View has saved the Protocol value (SOM of Files) to the preference file.

### Alert Source

You can select the sources of the alerts. If SOM server(s) have been configured, they will appear as check boxes. It is also possible to select the **Local files** option.

Bear in mind that if you select both a SOM server and also Local files and if they both correspond to the same system, you will get duplicated alert entries.

### Importing and Exporting an Alerts Configuration

It is possible to import and export the Alerts View configuration. The Alerts View stores its configuration in a file called *alarm-config.ser* in a directory that is operating system and user dependent.

On Linux it is `~/.isode/alarm-config.ser`

On Windows, it depends on the user and the version of Windows, for example, it could be under: `C:\Users\Isode\AppData\Roaming\Isode\alarm-config.ser`

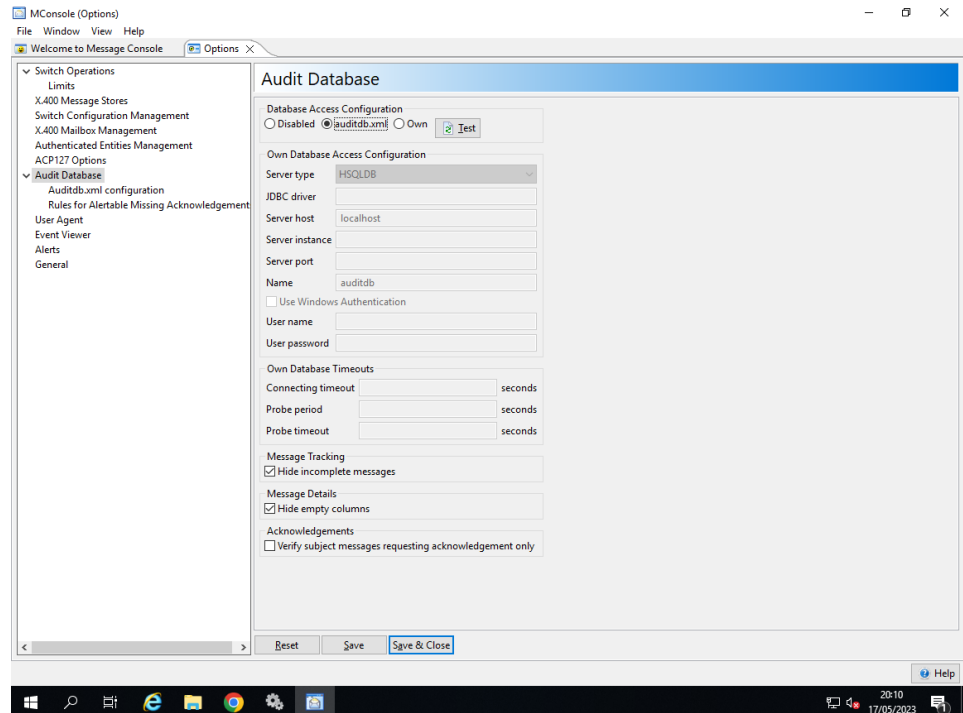
To import or export the configuration you can manually copy the file. Otherwise, you can click on the **Export** button and select a file where you want to save it, or click on the **Import** button and select a file where to import it from.

If you click on the **Edit** button, the Alert Configuration Editor is opened.

## 32.7.7 Configuration of Message Tracking

Before you can display a Message Tracking View you need to configure access to the Message Audit Database. See [Chapter 36, Message Audit Database](#) for details on configuring and running this.

Once you have a Message Audit Database running and populated with data, you can configure MConsole to connect to the database and provide information on message flows across your messaging network.

**Figure 32.5. Messaging tracking view configuration**

Ensure that you have the connection parameters correct by using the **Test** button. You can then apply the changes using the **Save Close** button and use the **View** → **Message Tracking** menu option to display Message Tracking information.

## 32.8 User Agent Configuration

Built into MConsole is the ability to submit messages by copying the content of messages which were delivered, non-delivered or are stuck in M-Switch queues.

---

**Note:** To enable messages which are no longer in the queue to be forwarded, you must have enabled message archiving and message tracking.

---

The message is resubmitted using the Isode X.400 API which will connect to an X.400 Message Store (such as M-Store) using P7 or an X.400 MTA using P3 directly.

To submit a message you need to:

- configure the built-in User Agent
- find the original message in **Message Tracking View** or the related report in **Delivery Tracking View**
- select the **forward** option and send the message.

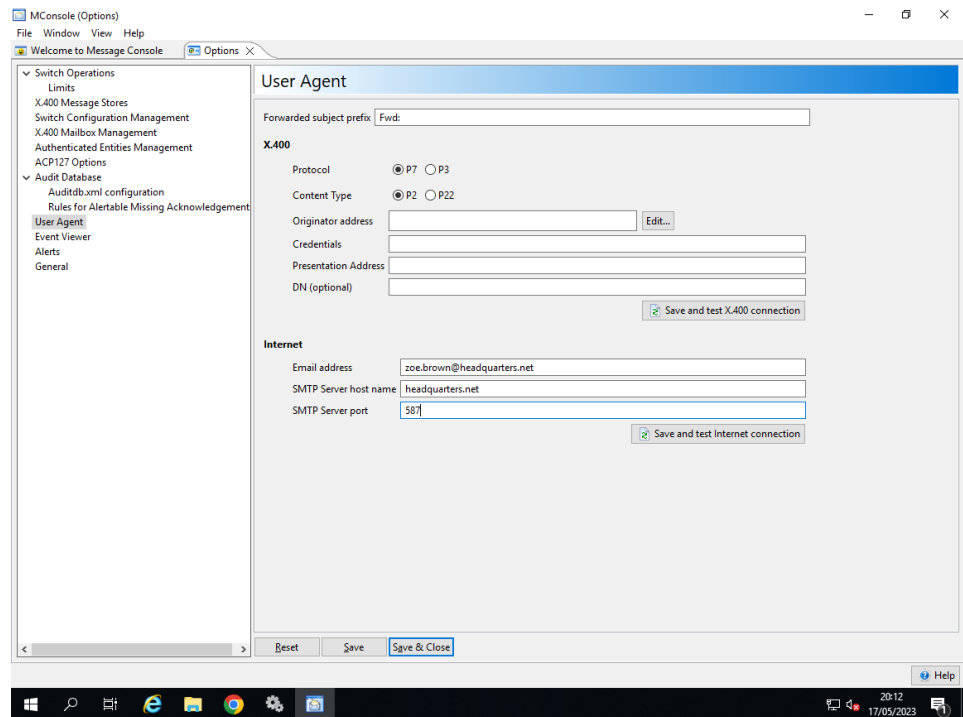
To configure the User Agent you need to specify:

- The protocol (P3, or P7).
- The Sender (an O/R Address used to bind to the X.400 Message Store, from whom the message will appear to originate). This must be an address that is routable on the MTA.

- The password to be used on the P7 bind to the X.400 Message Store or P3 password to the MTA.
- The presentation address of the X.400 Message Store or MTA P3 channel.

Figure 32.6, “MConsole User Agent configuration” shows an example of how the User Agent could be configured.

**Figure 32.6. MConsole User Agent configuration**



## 32.9 Configuring preferences

The Event Viewer is configured in the same way as the other MConsole views: with the **View → Options** dialog. There are currently five options that can be configured:

### Startup mode

What to do when a new **Event Viewer** view is opened. It can do nothing, connect to default server and do nothing, or connect and start monitoring.

### Default SOM server

If more than one SOM server is available, this is the one that will be used by default.

### Maximum lines per log file

The maximum number of lines that the program will request from the SOM server per log file.

### Default file regex

The regular expression used to request files from the SOM server.

### Display the full event text in a separate frame

For some events, the full text be too long to fit in the 'Log Message' column. If this option is selected a new frame will appear at the bottom of the Event Viewer view, containing the full text of the selected event.

## 32.10 ACP127

Monitoring and operating ACP127 components is carried out using the ACP127 View in MConsole. All MTAs known to MConsole can be monitored and managed using this view over the SOM protocol.

The ACP127 View left hand side allows MTAs and circuits to be selected and displayed. Each circuit is presented as a selectable Tab on the right hand side.

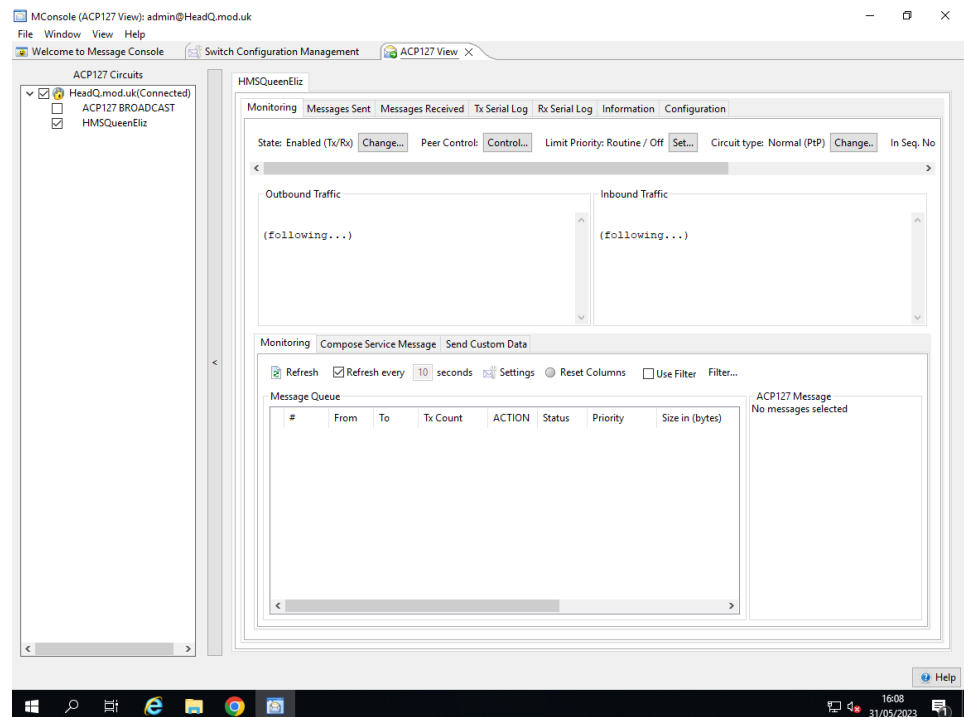
The MTAs available to the ACP127 View are as configured in the Switch Operations View. See [Figure 32.3, “Configuring the Switch Operations view”](#).

It is important to connect using the credentials of a user with monitoring access rights. These are set in the **Authenticated Entities Management** view. For more details, see [Section 13.3.5, “SASL User Management”](#).

Some parts of the ACP127 View also use the Audit DB features to return information about messages, such as displaying content. You will need to have configured your Audit Database in order for this functionality to work. For more details, see [Chapter 36, Message Audit Database](#).

The selection of connected MTAs and Circuits appears as follows:

**Figure 32.7. ACP127 Circuit MTA and Circuit Selection**



Once you have selected these, they will reconnect automatically and you can hide the left hand side of this view using the arrow bar to the right.

# Chapter 33 Starting and Stopping Your Messaging System

Stopping and starting the MTA on both UNIX and Windows systems are described in this chapter, together with any suggested pre-start checks.

A number of checking tools are provided, some of which you should consider running before attempting to start your MTA. In particular, you may wish to:

- Check the validity of addresses, that is, how each MTA will attempt to route given addresses. See [M-Switch Advanced Administration Guide](#) describes how to use the two tools provided for checking addresses, **ckadr** and **probe**.

Other useful checking tools, used once the MTA is running, are described in [M-Switch Advanced Administration Guide](#).

---

**Note:** M-Switch is usually run with M-Vault to hold the configuration. This chapter describes how to start and stop such setups. Starting and Stopping M-Switch when using a Table Based configuration is described in [Section 33.2.2, “Other startup actions”](#).

---

To start the messaging system you need to start the MTAs within your messaging configuration. Starting an MTA involves starting several processes. The way this is done is rather different on UNIX from under Windows, so the starting process is described in separate sections.

- [Section 33.1, “Starting an MTA on UNIX”](#) describes how to invoke and tailor the various processes an MTA may require on UNIX.
- [Section 33.2, “Starting an MTA on Windows”](#) describes how to start the various processes on Windows.

Similarly, stopping the messaging system involves stopping various processes and, again, the way this is done is somewhat different on the two platforms. What you need to do to stop an MTA is described in [Section 33.3, “Stopping an MTA”](#).

---

## 33.1 Starting an MTA on UNIX

### 33.1.1 Simple overview

#### 33.1.1.1 Use of `systemctl`

Unix-like systems nowadays use `systemctl` to start and stop long lived services. This replaces SVR4 startup and shutdown procedures in older Unix-like systems, such as Red Hat 6/Centos 6. This section describes starting and stopping the services using `systemctl` commands.

An MTA comprises a number of processes. Starting an MTA therefore involves starting the various processes associated with that MTA.

M-Switch uses the SVR4 legacy features of `systemctl` in which most of the complexity is hidden behind standard SVR4 startup scripts, which for M-Switch is in files such as `/etc/init.d/pp`. On RedHat Linux, this is installed as a link to the actual script in `(SBINDIR)/pp`. On other unix platforms this may vary slightly.



### 33.1.1.2 Configuration Prior To Starting Services

---

**Note:** It is important that M-Switch services run as an unprivileged user, i.e. not root. The M-Switch services run under the MTA userID (normally pp). The userID used is the owner of the file (*EXECDIR*)/*sendmail*.

---

#### 33.1.1.2.1 Run Time User

The user should be created before the packages are installed and must be a member of the `bin` group. The userID under which the `qmgr` and other M-Switch services run is the owner of the file (*EXECDIR*)/*sendmail*.

On some platforms such as Debian, it must be possible to login as the MTA userID. E.g. a login shell must be configured for the user.

NB the services must all be started as root. The services will switch user ID after starting to the unprivileged MTA userID.

#### 33.1.1.2.2 Startup Steps

The following steps are needed before actually starting the various messaging services described below:

- You must run `mconsole` and create a messaging configuration as described earlier.
- You must run `MConsole` and right click on the MTA entry to create the *mtaboot.xml* file in (*ECTCDIR*). This file is read by the `qmgr` which downloads the messaging configuration for this MTA and installs it for other M-Switch programs to read.
- You are also likely to need to configure messaging sub services by editing the (*ECTCDIR*)/*pp.rc* as described in [Section 33.1.3, “Configuring the Messaging startup script”](#).
- If audit services are going to be used, then **ISODE\_JAVA\_HOME** or **JAVA\_HOME** should be configured in (*ECTCDIR*)/*isode.rc*.

This section summarizes what you must start, and what you may need to start depending on your configuration. Once you have a general understanding of this section, turn to [Section 33.1.2, “Isode Messaging Services”](#), and follow the startup steps for your particular configuration:

1. If the Directory is to be used for the messaging configuration and/or routing, ensure the Directory Server is running before starting your messaging system.
2. Start the MTA processes. These can be started interactively from the command line, and left to run in the background, but normally they are started automatically as part of the system startup. Suitable scripts for doing this are copied into the appropriate places as part of the Isode MTA installation (e.g. the script */etc/init.d/pp* on RedHat Linux).
3. Only the `qmgr` process is essential and should be started before other processes. These run under the MTA userID (normally pp). The userID used is the owner of the file (*EXECDIR*)/*sendmail*.

### 33.1.1.3 Starting Services

In most simple cases, services are started/stopped/restarted by using an invocation such as the following in order to, for example, start an M-Switch service:

```
systemctl command service
```

If you are using Directory-based configuration you first start M-Vault using:

```
systemctl start dsa
```

The main M-Switch services include a small number of mandatory subservices, and a larger number of optional services depending on the type of M-Switch Configuration. See [Section 33.1.2, “Isode Messaging Services”](#) for a list of these sub services. See [Section 33.1.3, “Configuring the Messaging startup script”](#) for a description of how to configure each of them.

You can then start M-Switch using:

```
systemctl start pp
```

If you are using M-Store, you can start it using:

```
systemctl start pumice
```

If you are using the Audit DB services, you may need the following services:

If you are using Postgres, start this as follows:

```
systemctl start postgresql
```

Alternatively, if you are using the Isode supplied test/demo DBMS, start this as follows:

```
systemctl start isode-hsqldb
```

Other Audit DB processes are started as follows:

```
systemctl start adb-lp  
systemctl start adb-hk
```

These Audit DB services are less likely to be in use, but are included to complete the list of available services.

```
systemctl start adb-qosn  
systemctl start adb-qr
```

#### 33.1.1.4 Stopping Services

Stopping services is carried out by stopping services in the reverse order to starting them, for example, if you are using Directory-based configuration you first start M-Vault using:

```
systemctl stop dsa
```

### 33.1.2 Isode Messaging Services

The following lists the Services available as part of the Isode messaging suite. You will need some or all of these depending on the configuration of your system.

1. isode.pp.qmgr (mandatory)

This is the Queue Manager which:

- Manages the queue of messages and schedules the starting of processes, outbound channels and housekeeping channels (as described in [Section 14.2.1, “Editing channel](#)

properties”). Configuring and starting qmgr is described in [Section 33.1.4.1, “Queue Manager \(qmgr\)”](#).

- Updates the (*ETCDIR*)/*switch/mtatailor* file from the configuration held in the Directory.
- Provides management and monitoring services for the Isode MTA.

## 2. isode.pp.smtp (applicable for Internet and MIXER systems)

This process enables the inbound SMTP channel to run as a Service.

You can start this process either from a command line or under the control of inetd(8). Details of both methods are given in [Section 33.1.4.2, “Starting the SMTP server”](#).

## 3. Alert Service (X.400, Internet and MIXER)

The Alert Daemon is a SOM client application which retrieves information about M-Switch entities such as the Queue Manager, channels and Peer MTAs and applies filters to the values obtained. Events can be generated as a result. See [Section 42.1, “Overview”](#).

## 4. OTAM Server (MIXER)

This is for use in ACP127 systems. See [Section 19.2.6, “OTAM”](#).

## 5. SLEP Server (X.400, Internet and MIXER)

This provides SLEP connectivity. See [Section 25.1, “The SLEP channel”](#).

## 6. CFTP Server (X.400, Internet and MIXER)

This provides CFTP connectivity. See [Section 23.1, “The CFTP channel”](#).

## 7. Corrector (X.400, Internet and MIXER)

This provides the features in which Operators can correct messages interactively from a Web Browser. See [Section 22.1, “Corrector Channel”](#).

## 8. fteserver (X.400, Internet and MIXER)

This is a channel program which is used to transfer emails to and from the filesystem. Typically it’s used in conjunction with Sodium Sync to allow directory replication over email. Configuration information for the fteserver is given within the *M-Switch Advanced Administration Guide*.

## 9. ACP142 (X.400, Internet and MIXER)

You need to start this channel if you wish to use the STANAG 4406 Annexe E channel to send and receive X.400 messages using P1 instead of, or in addition to, using the x400p1 channel. You can start it explicitly by running it from the command line (see [Section 33.1.4.3, “Starting the ACP142 channel”](#)) or you can start it as part of the standard startup script (see [Section 33.1.3.4, “Starting ACP142”](#)).

## 10. ACP127 (MIXER only)

You need to start this channel if you wish to use the ACP127 channel to send and receive ACP127 (or variant) messages. See [Section 33.1.4.4, “Starting the ACP127 channel”](#)) or you can start it as part of the standard startup script. See [Section 33.1.3.5, “Starting ACP127”](#)).

## 11. P1File (X.400 and MIXER only)

This is a protocol channel which delivers and accepts P1 APDUs in a file. There are some gateway products which use this format to gateway into legacy networks such as ACP 127. See [Section 16.5, “P1 file channel”](#) for a description of how to configure and run the P1File channel.

## 12. isode.pp.p3 (X.400 and MIXER only)

This is a multithreaded P3 protocol server. It can be used as a replacement for (or in addition to) the `p3server` channel which is started on demand by the `isode.iaed` process. See [Section 33.1.3.6, “Starting the P3 Server”](#)) for details of how to start it as part of the standard startup script.

13. Isode Audit DB Services. Starting and stopping these are described in [Section 36.5, “Starting and Stopping the Audit Database Services”](#).

14. `isode.iaed` (X.400 and MIXER only)

This is needed if the MTA is to receive incoming X.400 P1 or P3 messages.

In X.400 and MIXER configurations the X.400 Message Store process `isode.pumice` will also need to be started if you need to run the X.400 Message Store. Command details for X.400 Message Store processes are given in the *M-Store Administration Guide*.

---

**Note:** Reference is made in the following sections to the files `(SHAREDIR)isoentities` and `(SHAREDIR)isoservices`. These are part of the Isode transport configuration, they are documented in *M-Vault Administration Guide*.

---

### 33.1.3 Configuring the Messaging startup script

This section describes the usual way M-Switch services are started.

The startup script in `/etc/init.d/pp` will by default start these MTA processes :

**isode.pp.qmgr**  
Queue Manager

**isode.pp.smtpsrvr**  
SMTP Server

**isode.iaed**  
X.400 Protocol Listener

However you may need to tailor this startup script to start up various messaging subservices as described in [Section 33.1.2, “Isode Messaging Services”](#) for details of clustering configuration.

To do this create the `/etc/isode/pp.rc` file. A sample file is provided with the possible useful lines commented out. To enable the features, copy `/etc/isode/pp.rc.sample` to `/etc/isode/pp.rc`.

#### 33.1.3.1 Starting iaed

If you are running an X.400 MTA and do not wish to listen for incoming P1 and/or P3 associations, add the following lines to `pp.rc`

```
# If USE_X400 is set to "no", then the Service is not used,
# even if installed
USE_X400=no
```

If you are running an X.400 MTA and wish to use a second iaed listener (if for example you are running the Airtel stack, add the following lines to `pp.rc`:

```
# Run a second IAED process which runs as the MTA login
# This is required for some transport service providers
# Set the value to "yes" to invoke this. The normal Service
# can be prevented by defining USE_IAED1 to "no"
USE_IAED2=yes
USE_IAED1=no

# Options for 2nd IAED
```

```
# When using the Airtel stack, for instance, this should be
# -s xtiairtel
# IAED2_OPTIONS=
```

### 33.1.3.2 Starting FTBE

If you wish to use the FTBE service, uncomment the following lines in *pp.rc*:

```
# # Use this to start the FTE daemon when the MTA is started
USE_FTE=yes
FTE_OPTIONS=" -c ftbe"
```

### 33.1.3.3 Starting Profiler

If you wish to use the Profiler service, uncomment the following lines in *pp.rc*:

```
# Use this to start the profiler daemon when the MTA is started
USE_PROFILER=yes
PROFILER_OPTIONS=" -c profiler"
```

### 33.1.3.4 Starting ACP142

If you are running an X.400 MTA and wish to use the ACP142 channel, uncomment the following lines in *pp.rc*:

```
# Use this to start the acp142 Service when the MTA is started
USE_ACP142=yes
ACP142_OPTIONS=" -c acp142"
```

### 33.1.3.5 Starting ACP127

If you wish to use the ACP127 channel, uncomment the following lines in *pp.rc*:

```
# Use this to start the acp127 Service when the MTA is started
USE_ACP127=yes
ACP127_OPTIONS=" -c acp127"
```

### 33.1.3.6 Starting the P3 Server

If you are running an X.400 or MIXER MTA and wish to use the isode.pp.p3 static P3 listener, uncomment the following lines in *pp.rc*:

```
# Use this to start the P3 server daemon when the MTA is started
USE_P3=yes
P3_OPTIONS=" -c p3server"
```

### 33.1.3.7 Executing arbitrary commands on start/stop

Insert any commands you wish to be run on startup/shutdown in the appropriate section below.

#### Example 33.1. Executing commands on startup/shutdown in *pp.rc*

```
case $1 in
'start')
    # Put commands to be executed when starting up here
```

```
;;

'stop')
    # Put commands to be executed when stopping here
;;

esac
```

### 33.1.3.8 Killing programs when M-Switch Service stops

The variable `SWITCH_PROGS_TO_KILL` can be set as a space-separated list of programs to be killed when the MTA stops. When running a clustered system it is important that all M-Switch programs are killed when M-Switch is closed down. See [Chapter 37, Clustering](#) for details of clustering configuration.

Uncomment the following lines to do this:

```
# SWITCH_PROGS_TO_KILL="\
# ${execpath}example_gw_prog\
# ${execpath}x400p1 \
# "
# # Use this value to kill all programs exec'ed in background
# SWITCH_PROGS_TO_KILL="${execpath}*"
```

## 33.1.4 Starting/Stopping Services Individually

This section will not normally be relevant as the messaging sub services are started/stopped by the `/etc/isode/pp.rc` script. If you do need to manage services from the command line without using the startup script, this sections describes how to do this.

### 33.1.4.1 Queue Manager (qmgr)

The process `isode.pp.qmgr` is the primary MTA process, and resides in (`SBINDIR`). It is managed mostly through the MConsole **Switch Operations** View, which is documented in [Section 32.6, “MConsole Switch Operations Configuration”](#).

However, there are some command line options you can use when starting `qmgr`. These are given below.

- s  
singleshot, which allows you to start the `qmgr` just to update the `mtatailor.tai` file from the Directory. When this is complete, `qmgr` exits.
- d  
debug If you specify `-d`, the `qmgr` runs in debug mode in foreground, otherwise the `qmgr` runs in the background and updates the `mtatailor` file from the Directory at the configured intervals. By default this is 5 minutes. (See [Section 14.1.8, “Advanced configuration options”](#) (**Advanced** tab) to see how to change this using the **config reload** setting).
- S  
Disable inbound channels.
- D  
Disable outbound channels.

The `qmgr` starts by reading the `mtaboot.xml` file in order to determine how to connect to the Directory. The `mtaboot.xml` file is created in MConsole by right clicking on the MTA entry, and creating the `mtaboot.xml` file in (`ECTCDIR`).

Valid values for the `mtaboot.xml` file are:

`loc_mtadnname` *Distinguished Name of target MTA*

This is the Distinguished Name of the MTA whose values are read in order to create the *mtataylor* file.

`dap_user` *Distinguished Name of target MTA*

This is the Distinguished Name to be used on the bind

`dap_passwd` *Distinguished Name of target MTA*

This is the password to be used in the bind

`set ldap_host=hostname`

This is the hostname to connect to

`set ldap_port=portnum`

This is the port to connect to

### Example 33.2. Example *mtaboot.xml* file

```
mtataylor
mtadnnamecn=headquarters.net, cn=Messaging Configuration,
ou=MHS, c=GB/mtadnname
set name="dsa_address"ITOT=localhost:19999/set
set name="ldap_sasl_mech"simple/set
set name="ldap_host"localhost/set
set name="ldap_port"19389/set
/mtataylor
```

The *qmgr* listens on TCP ports in order to receive SOM requests from MConsole. By default this is port 18002. The *qmgr* listens on port 18001 to communicate with channels.

## 33.1.4.2 Starting the SMTP server

The SMTP channel comes in two parts, an inbound and an outbound process. Outbound SMTP connections are started by the Queue Manager on demand. The inbound process is run as a stand-alone server.

The stand-alone SMTP process is called *isode.pp.smtp*, and resides in (*EXECDIR*). A description of the SMTP channel configuration is in the *M-Switch Advanced Administration Guide*.

Start *isode.pp.smtp* either interactively or from the start up script with a command line of the form (lines have been wrapped for readability):

```
(EXECDIR)/smtpsrvr options
channelkey/ipaddress/port
```

The options are listed below. At least one channel key must be provided.

The *ipaddress* is optional.

`-p port`

Alternative way to specify the port if only one to be used.

`-i maxcom`

Set the maximum number of simultaneous SMTP connections that are supported.

`-t timeout`

Nameserver timeout support.

`-d`

Run in debug mode.

`-n`

Disable reverse DNS lookups on connecting clients.

For example:

**Example 33.3. Example isode.pp.smtp startup to listen on ports 25 and 587 on headquarters.net**

```
/opt/isode/libexec/isode.pp.smtp smtp/headquarters.net/25  
smtp/headquarters.net/587
```

The above line is split for convenience.

**Example 33.4. Example isode.pp.smtp startup to listen on ports 25 and 587 on all devices**

```
/opt/isode/libexec/isode.pp.smtp smtp//25 smtp//587
```

### 33.1.4.3 Starting the ACP142 channel

The ACP142 channel comes in two parts, an inbound and an outbound process. Both directions are handled by a single process x400pmul.

Start x400pmul either interactively or from the start up script with a command line of the form:

```
(EXECDIR)/x400pmul options
```

The options are listed below.

`-c channel`

The name the channel runs as.

**Example 33.5. Example ACP142 invocation running as channel acp142**

```
/opt/isode/sbin/x400pmul -c acp142
```

### 33.1.4.4 Starting the ACP127 channel

The ACP127 channel comes in two parts, an inbound and an outbound process. Both directions are handled by a single process acp127.

Start acp127 either interactively or from the start up script with a command line of the form:

```
(EXECDIR)/x400pmul options
```

The options are listed below.

`-c channel`

The name the channel runs as.

**Example 33.6. Example ACP127 invocation running as channel acp127**

```
/opt/isode/sbin/acp127 -c acp127
```



---

## 33.2 Starting an MTA on Windows

On Windows-based systems, the Isode Service Configuration Tool can be used to start and stop the processes which make up the MTA. The publication *M-Vault Administration Guide* describes how to use this tool.

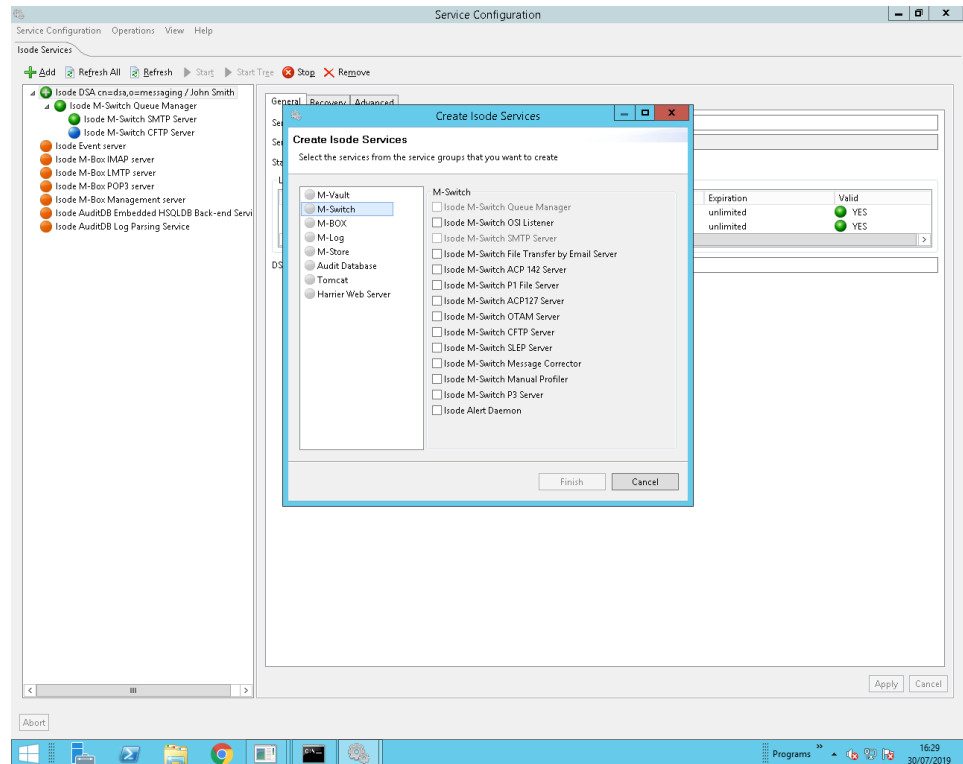
An MTA comprises a number of services. Starting an MTA therefore involves starting the various processes associated with that MTA. This section summarizes what you must start, and what you may need to start depending on your configuration. Subsequent subsections give details of command lines and tailoring options. Each process is introduced using its full name, but subsequent references will use a shortened form. For example, qmgr will refer to the process isode.pp.qmgr.

### 33.2.1 Overview

The Isode services are managed using the Isode Service Configuration tool which is started using the **Start** → **Programs** menu. Make sure you run as Administrator. The Isode services are not created when the Isode packages are installed. You need to determine which of the services you require which depends on the configuration you plan.

The MTA processes are installed as Windows services. These processes run under the Local System account. They can be started manually using the Isode Service Configuration applications, or automatically at system startup by configuring the Isode Services as automatically started services.

Once you have configured your system, you need to create the necessary Isode services. Select **Operations** → **Create Service** in the Isode Service Configuration tool to create specific services. The **Add** button on the toolbar can also be used to create multiple services in one step by selecting the services for each group. Once the services have been created, they can be modified to configure them with requisite parameters and dependencies.

**Figure 33.1. MTA Service Installation**

Select the products you have installed and configure services you wish to add by selecting them and click on **Finish**.

Some of these services are essential: **isode.dsa**, isode.pp.qmgr. The others are optional.

1. isode.pp.qmgr (mandatory)

This is the Queue Manager which:

- Manages the queue of messages and schedules the starting of processes, outbound channels and housekeeping channels (as described in [Section 14.2.1, “Editing channel properties”](#)). Configuring and starting qmgr is described in [Section 33.1.4.1, “Queue Manager \(qmgr\)”](#).
- Updates the (*ETCDIR*)/switch/mtataylor file from the configuration held in the Directory.
- Provides management and monitoring services for the Isode MTA.

2. isode.pp.smtp (applicable for Internet and MIXER systems)

This process enables the inbound SMTP channel to run as a Service.

You can start this process either from a command line or under the control of inetd(8). Details of both methods are given in [Section 33.1.4.2, “Starting the SMTP server”](#).

3. Alert Service (X.400, Internet and MIXER)

The Alert Daemon is a SOM client application which retrieves information about M-Switch entities such as the Queue Manager, channels and Peer MTAs and applies filters to the values obtained. Events can be generated as a result. See [Section 42.1, “Overview”](#).

4. OTAM Server (MIXER)

This is for use in ACP127 systems. See [Section 19.2.6, “OTAM”](#).

5. SLEP Server (X.400, Internet and MIXER)

This provides SLEP connectivity. See [Section 25.1, “The SLEP channel”](#).

6. CFTP Server (X.400, Internet and MIXER)

This provides CFTP connectivity. See [Section 23.1, “The CFTP channel”](#).

7. Corrector (X.400, Internet and MIXER)

This provides the features in which Operators can correct messages interactively from a Web Browser. See [Section 22.1, “Corrector Channel”](#).

8. fteserver (X.400, Internet and MIXER)

This is a channel program which is used to transfer emails to and from the filesystem. Typically it's used in conjunction with Sodium Sync to allow directory replication over email. Configuration information for the fteserver is given within the *M-Switch Advanced Administration Guide*.

9. ACP142 (X.400, Internet and MIXER)

You need to start this channel if you wish to use the STANAG 4406 Annexe E channel to send and receive X.400 messages using P1 instead of, or in addition to, using the x400p1 channel. You can start it explicitly by running it from the command line (see [Section 33.1.4.3, “Starting the ACP142 channel”](#)) or you can start it as part of the standard startup script (see [Section 33.1.3.4, “Starting ACP142”](#)).

10. ACP127 (MIXER only)

You need to start this channel if you wish to use the ACP127 channel to send and receive ACP127 (or variant) messages. See [Section 33.1.4.4, “Starting the ACP127 channel”](#)) or you can start it as part of the standard startup script. See [Section 33.1.3.5, “Starting ACP127”](#)).

11. P1File (X.400 and MIXER only)

This is a protocol channel which delivers and accepts P1 APDUs in a file. There are some gateway products which use this format to gateway into legacy networks such as ACP 127. See [Section 16.5, “P1 file channel”](#) for a description of how to configure and run the P1File channel.

12. isode.pp.p3 (X.400 and MIXER only)

This is a multithreaded P3 protocol server. It can be used as a replacement for (or in addition to) the p3server channel which is started on demand by the isode.iaed process. See [Section 33.1.3.6, “Starting the P3 Server”](#)) for details of how to start it as part of the standard startup script.

13. Isode Audit DB Services. Starting and stopping these are described in [Section 36.5, “Starting and Stopping the Audit Database Services”](#).

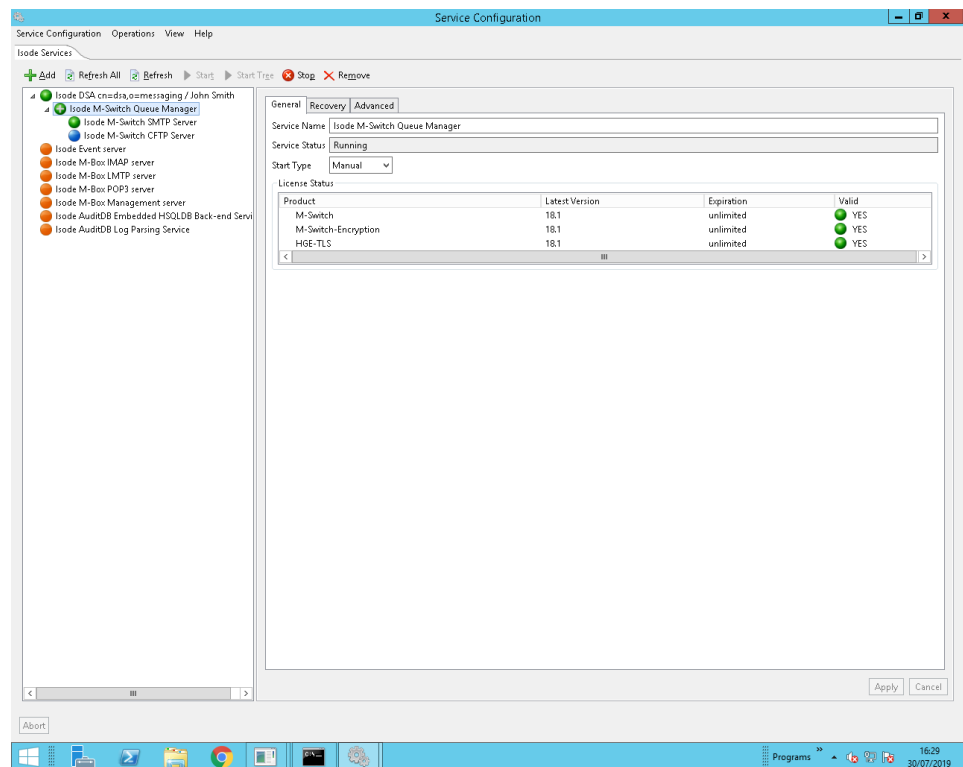
14. isode.iaed (X.400 and MIXER only)

This is needed if the MTA is to receive incoming X.400 P1 or P3 messages.

In X.400 and MIXER configurations, X.400 Message Store processes such as isode.pumice may also need to be started. Command details for X.400 Message Store processes are given in *M-Store Administration Guide*.

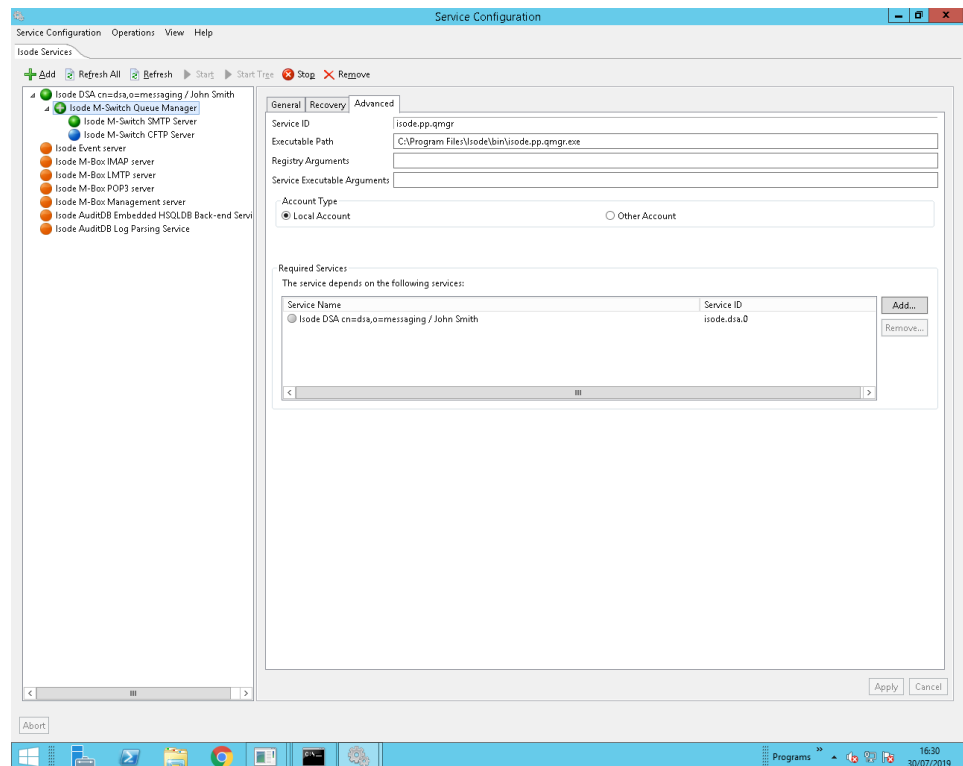
In order to make sure they start and stop in the right order, you need to edit the services in order to set the correct dependencies. Most Isode services are resilient to being started before services on which they are dependent have started, but startup is quicker if the order is correct.

You also need to set the services to automatic/manual/disabled as necessary.

**Figure 33.2. MTA Service Configuration**

You are recommended to set:

- M-Switch services dependent on qmgr
- qmgr dependent on dsa

**Figure 33.3. MTA Service Dependency Configuration**

Reference is made in the following sections to the files (*SHAREDIR*)*isoentities* and (*SHAREDIR*)*isoservices*. As these are part of the Isode transport configuration, they are documented in the *M-Vault Administration Guide*.

### 33.2.2 Other startup actions

If you are using file based tables held in DB format any changes must be consolidated by running **dbmbuild** to ensure that the tables, if required, are up to date – see [Section 34.6, “Dump/Restore and CleanIsode Utilities”](#).

### 33.2.3 Queue Manager (qmgr)

The qmgr can be started with a number of switches which can modify its behaviour. It is unlikely you will need to change these. If you do, the values are documented in [Section 33.1.4.1, “Queue Manager \(qmgr\)”](#).

To change the way the Service is configured, select the Service in the Isode Service Manager and select the **details** button. You can then change the Service arguments.

### 33.2.4 SMTP server

The isode.pp.smtp can be started with a number of switches that can modify its behaviour. If you need to change these, the values are documented in [Section 33.1.4.2, “Starting the SMTP server”](#).

To change the way the Service is configured, select the Service in the Isode Service Manager and select the **details** button. You can then change the Service arguments.

---

## 33.3 Stopping an MTA

### 33.3.1 Stopping an MTA on Unix

Normally, the MTA programs will run without interruption unless something serious happens, such as the machine going down. If you need to stop the MTA, it is advisable to shut down the MTA programs in a controlled manner. In a controlled shut down, all channels should have stopped running before the MTA can be completely stopped.

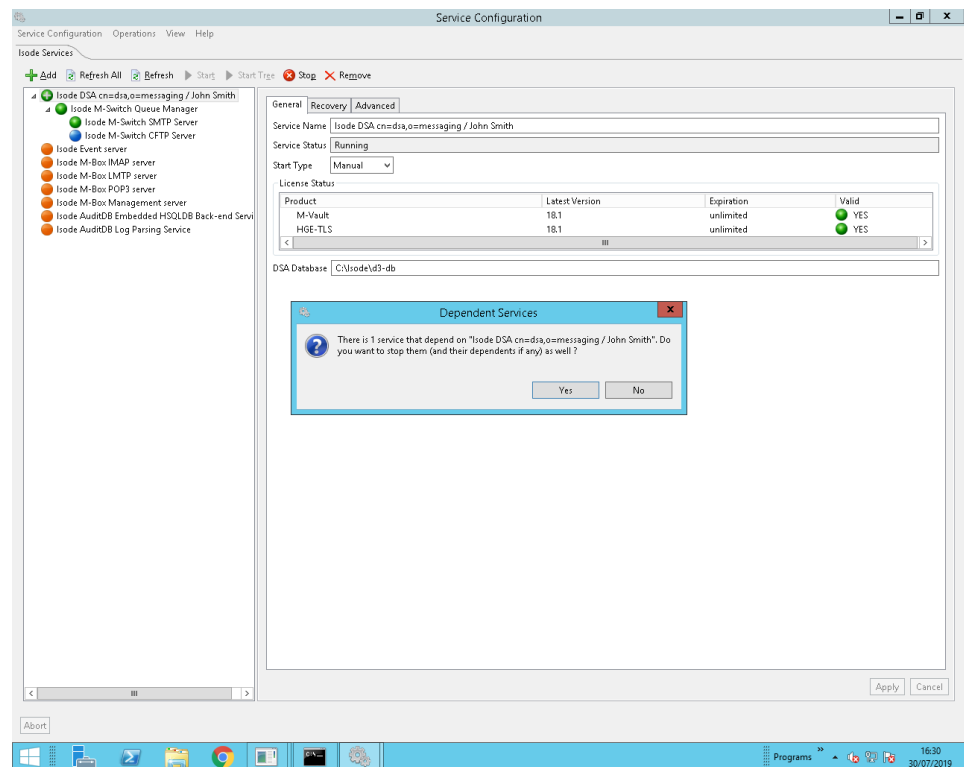
The MTA services should normally be stopped by using the startup / shutdown script which will have been installed as part of the Isode package (e.g. */etc/init.d/pp* on RedHat Linux). If you are not using a script, the following processes should first be killed in the order given:

At this point, you should wait for qmgr to exit. The qmgr sends a terminate message to all the channels it has started, and waits for them to finish. By default this should happen within 30 seconds, but there are configurable timers which can be configured to change this.

1. isode.iaed
2. isode.pp.smtp
3. isode.pp.qmgr

### 33.3.2 Stopping an MTA on Windows

The MTA is stopped using the Isode Service Configuration. You can choose to close individual Services, or a tree of Services which are configured with dependencies.

**Figure 33.4. Stopping an MTA on Windows**

# Chapter 34 Managing Your Messaging System

This chapter describes the various ways you can monitor your messaging system, and gives some guidance on troubleshooting and improving performance. The first section describes how to create a backup of a messaging configuration.

[Section 34.2, “Configuring logging”](#) to [Section 34.4, “Dealing with errors”](#) describes how to monitor the MTA and its logs and deal with errors. [Section 34.5, “What to run each night”](#) describes how you can improve performance by using a spool directory cache. [Section 34.6, “Dump/Restore and CleanIsode Utilities”](#) gives some guidance on daily maintenance of the system.

The MTA can also be monitored using SNMP and an appropriate management agent. This is covered in [Chapter 35, SNMP](#).

---

## 34.1 Backing up a Messaging configuration

Apart from regular directory backups, it is recommended that regular backups are made of the directory-based messaging configuration.

This can be used to identify changes in the configuration or to restore the system to a known state. It is also a very useful source of information for providing support.

There are four ways of creating a backup. Some just create an LDIF file, which can then be either loaded to another directory, or examined with a text processor. Some are more complete and also copy configuration files.

The easiest way is to use MConsole, which basically provides a simple way to run the DumpIsode utility. If you want to create a backup from the command line or from cron, then please refer to [section Section 34.6, “Dump/Restore and CleanIsode Utilities”](#) Dump/Restore and CleanIsode Utilities.

General purpose DUAs such as Sodium and Tcldish can also be used if you want, for example, to include it in a `cron` job. For more information about `tcldish` or `Sodium`, refer to the *M-Vault Administration Guide*.

### 34.1.1 Creating a backup with MConsole

To backup your configuration using the MConsole open the **Welcome View** by selecting **View** → **Welcome View**. Then click on the **Backup a Messaging Configuration**.

If you are using an X.400 P7 Message Store, it is possible to include all the message index entries in the backup, however this is not normally necessary as they can be regenerated by the P7 Message Store.

You can choose the path where the backup will be saved, and if you select the option **Include the date in the subdirectory** the current date will be appended to the directory. This is an easy way to determine when the backup was made.

Once you click on the **Backup** a backup will be made and the result of the command will be shown in a pop-up window.

### 34.1.2 Creating a backup with Sodium

Start (*BINDIR*)/*sodium*, bind to the directory, and then expand the top entry (**The World**). Keep expanding the tree until you find the `cn=Messaging Configuration` entry that corresponds to the messaging configuration you want to save.

Select the `cn=Messaging Configuration` entry, and right click on it, selecting **Bulk Tools** → **LDIF dump**. You can then dump the configuration to LDIF.

### 34.1.3 Creating a backup with tcldish

The first way is using the (*BINDIR*)/*tcldish* program. Invoke it and it will start an interactive Tcl shell. If you are not running it in the same machine as the DSA, refer to the *M-Vault Administration Guide* for more information on how to bind to the right DSA.

After you have bound to the DSA, you can navigate around it using the **dlist** command with the numbers displayed alongside entries on screen.

For example, the first response to **dlist** in [Example 34.1, “Using dlist to navigate a DSA”](#) is a single entry:

```
1 c=XX
```

The command **dlist 1** lists the entries under `c=XX`. The example then drills down through the hierarchy until the entry corresponding to the DSA Manager is displayed. The **dbind** command is used to bind as that user. You could use either the DSA Manager or, as a minimum, a user that has permissions to read passwords under `ou=Messaging Configuration`.

After you have successfully bound, you can issue the **dbulkcommand** to save the contents of the DSA in LDIF format in the file specified after `-o`. The last argument of the **dbulkcommand** is the DN of the entry corresponding to `cn=Messaging Configuration`.

#### Example 34.1. Using dlist to navigate a DSA

```
# tcldish
Welcome to TclDish !

TclDish% dlist
1      c=XX
TclDish% dlist 1
2      cn=dsa
3      ou=MHS
4      cn=dl-1
TclDish% dlist 2
5      cn=gdaml
6      cn=DSA Manager
TclDish% dbind 6
Enter password for "<cn=DSA Manager, cn=dsa, c=XX>":

TclDish% dlist 3
7      cn=Messaging Configuration
8      o=Address Book

TclDish% dbulkdump -dontdereferencealias -file /tmp/myconfig.ldif 7
TclDish% exit
```

You can also do the whole thing directly in one command, if you know the DN of Messaging Configuration (which can be obtained from MConsole → **Messaging** → **MHS Management**).



```
TclDish% dbulkdump -dontdereferencealias -file /tmp/myconfig3.ldif  
"cn=Messaging Configuration, ou=MHS, c=XX"
```

---

## 34.2 Configuring logging

When you create an MTA using MConsole, three logging streams are created by default: `auditlog`, `eventlog` and `operlog`. These are used to control the log output from the MTA's component programs. The `auditlog` log stream provides formatted messages which record the arrival and departure of messages at the MTA, delivery to local recipients and generation of delivery reports. The contents of the stream can be used for statistical analysis. [Section 34.2.7, "Audit record format"](#) describes the format of records on this stream. The `eventlog` is a free-form log stream used for error, warning and tracing messages. The `operlog` is only used for reporting serious problems which the MTA operator should investigate immediately.

Four types of logging stream 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 `xmpp` type, where events are sent to an XMPP server.

The default configuration for the MTA is for the `auditlog` and `eventlog` streams to be file streams (with names *mta-audit.log* and *mta-event.log* respectively), and the `operlog` stream to be a system stream.

Logging can be specific to a program or channel. The properties can be completely separate, or can build on the properties of the default stream of the same name. Accordingly, many items can be configured to be set, to be unset or to use the default.

### 34.2.1 Facilities

Logging is subdivided into Facilities which allow the logging of one or more subsystem to be changed - eg to turn up logging for that Facility to Detail or Debug level.

The complete list of facilities is:

- ACP127: ACP127 protocol operations
- ACP142: P\_Mul
- ACsap: Association level
- Address: Address manipulation routines
- ALETRD: Events generated by the Alertd application
- Application:
- asn1: ASN.1 encoding/decoding
- auditdb: the Message Audit database
- Base: Base library routines
- CFTP: Messages relating to the Compressed File Transfer Protocol
- Compat: Compatibility library routines
- DSA\_DAP: Directory operations using DAP
- DSA\_DISP: Directory operations using DISP

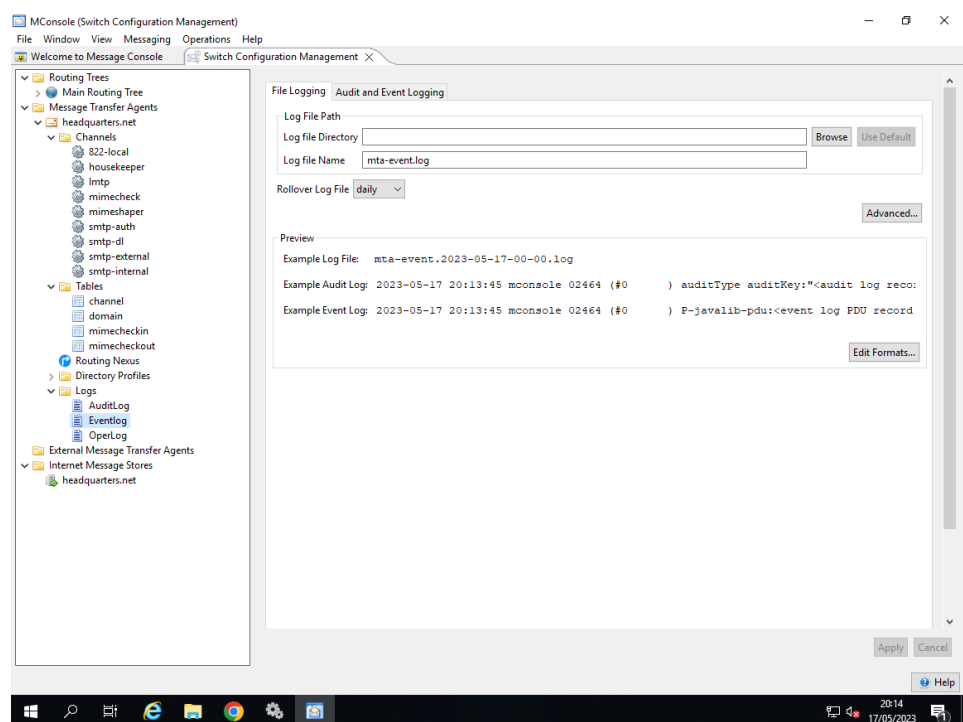
- DSA\_DSP: Directory operations using DSP
- DSA\_GDAM: Directory events from low level GDAM
- DSA\_GEN: General Directory events
- DSA\_INIT: Directory initialisation operations
- DSA\_LDAP: Directory operations using LDAP
- DSA\_SASL: Directory operations using SASL
- DSA\_SEC: Directory security operations
- Dsap: Directory client API operations
- dsapi: Directory client API operations (Java)
- harrier: the Harrier web server
- Icon5066: the ICON5066 server
- ICrypto: the security database
- IOevent: General OSI stack events
- javacryptomsg: java cryptography operations
- javadsapi: Java-DSAPI messages
- javalib: other java operations
- LDAP: Directory operations
- LIST: Messages relating to the various list channel operations
- Logging: the logging subsystem
- MBOX: IMAP operations
- mclient: IMAP client library
- MLink: XMPP Server
- MTA: General MTA operations
- MTA\_Qmgr: MTA Queue Manager
- MTA\_SMTP: MTA SMTP protocol operations
- MTA\_X400: MTA X.400 protocol operations
- MTAnfig: MTA configuration problems
- mvc: M-Vault Console
- nsnmp: SNMP managed object events
- OTAM: the OTAM server
- P3: the P3 server
- Profiler: Messages relating to Profiler
- Psap: Presentation layer
- RFC4158: Certificate check library
- ROsap: ROSE layer
- RTsap: RTS layer
- S5066msg: Stanag 5066 operations
- SASL: Simple Authentication and Security layer
- SecLabel: Security label library
- SLS: Security Label server
- SMIME: SMIME processing
- sodium: The Sodium DUA
- SOM: SOM protocol operations

- `ssap`: Session layer
- `swift`: The Swift XMPP client
- `tsap`: Transport layer
- `x400api`: The X.400 client API
- `x509`: The X.509 cryptographic subsystem
- `xms`: The X.400 Message Store
- `xmsLegacy`: The X.400 Message Store (legacy logging)
- `xmt`: OpenGroup API operations

## 34.2.2 File and TTY streams

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

**Figure 34.1. File page.**



### Log File Path

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

### Log File Directory

If no directory path is specified, then the log file will be written to (*LOGDIR*). If this is a `tty` stream, leaving this field blank will cause output to go to `stderr`, while setting the field to the word `stdout` will cause output to go to `stdout`.

### Log File Name

The file name of the logging file, if this is a file stream. If no directory path is specified, then the log file will be written to (*LOGDIR*). If this is a `tty` stream, leaving this field blank will cause output to go to `stderr`, while setting the field to the word `stdout` will cause output to go to `stdout`. Note that if you have rollover configured (see below) the actual filename will in part be derived from the rollover interval.

### Rollover Log File

Select the period over which roll over takes place.

The following are configurable on the **Advanced** button:

**Set 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. This is ignored for `tty` streams.

**Log to Open File descriptor number**

This enables you to log to an open file descriptor. The integer value of the file descriptor is set in this field. This is ignored for `tty` streams.

**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. This is ignored for `tty` streams.

**Sync log messages to disk**

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. This is ignored for `tty` streams.

**(Windows only) Lock the file prior to writing the message**

(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. This is ignored for `tty` streams.

**Rollover**

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:

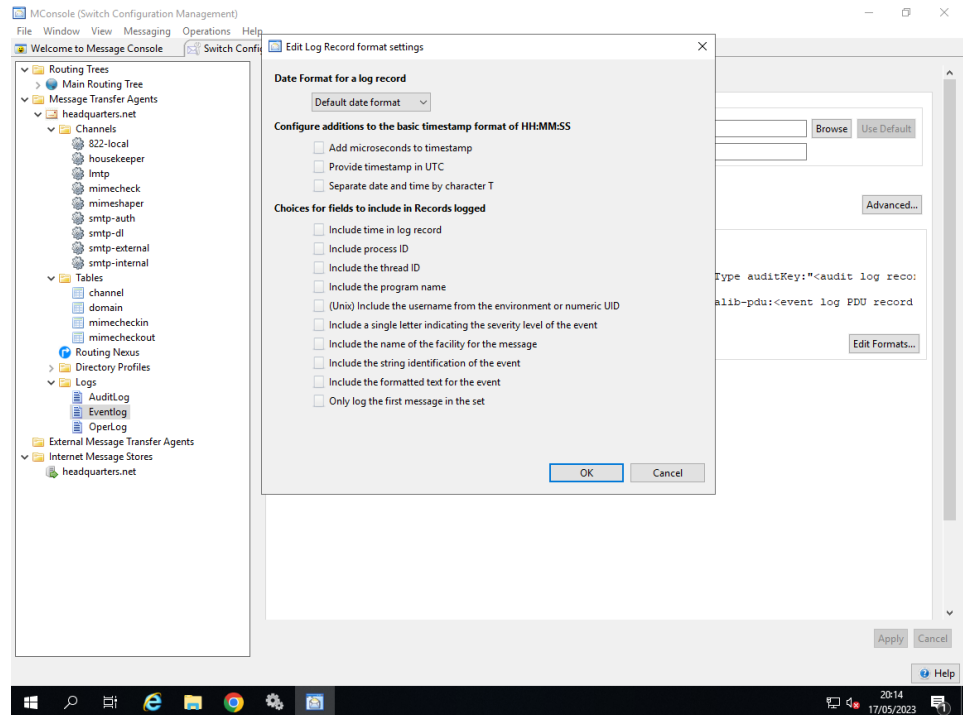
**rollover interval**

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: for example, midnight for intervals of a day or more. The offset enables you to move this start time. This is ignored for `tty` streams.

The **Edit Formats...** page allows control of the format of the common elements of each record logged.

**Figure 34.2. Edit Log Record Format**

### Date Format for a log record

Configure how the date element of the time-and-date field at the start of each log record is formatted. Available choices are:

- **Default date format:** This gives the default date format which is in the form of YYYY-MM-DD.
- **No date field:** Do not include date field
- **MM/DD format:** Month number and day of the month, i.e. "MM/DD"
- **YY-MM-DD format:** two-digit year, month and day in YY-MM-DD format
- **YYYY-MM-DD format:** (default) as above, but four-digit year, i.e. "YYYY-MM-DD"

### Timestamp Format

Configure additions to the basic timestamp format of HH:MM:SS. Choices are:

- **microsec:** 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.
- **separator:** When used in conjunction with the four-digit or two-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, e.g. 2012-06-13T12:34:56.

### Message fields

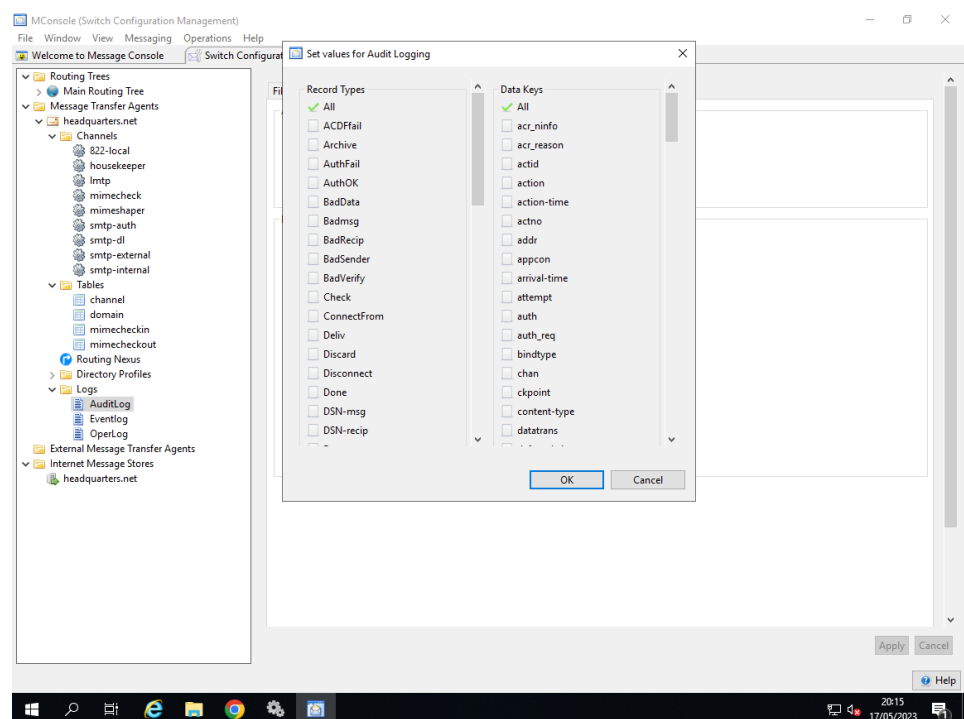
Configure what fields are included in records logged to a file stream. The default setting is for all fields to be included apart from thread id and 'firstonly'.

The available choices are:

- **time:** Include date and time in record.
- **pid:** Include process ID.
- **threadid:** Include the thread ID. This enables you to distinguish events logged by different threads within the process.
- **progrname:** 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

**Figure 34.3. Audit Log Options: audit tab**

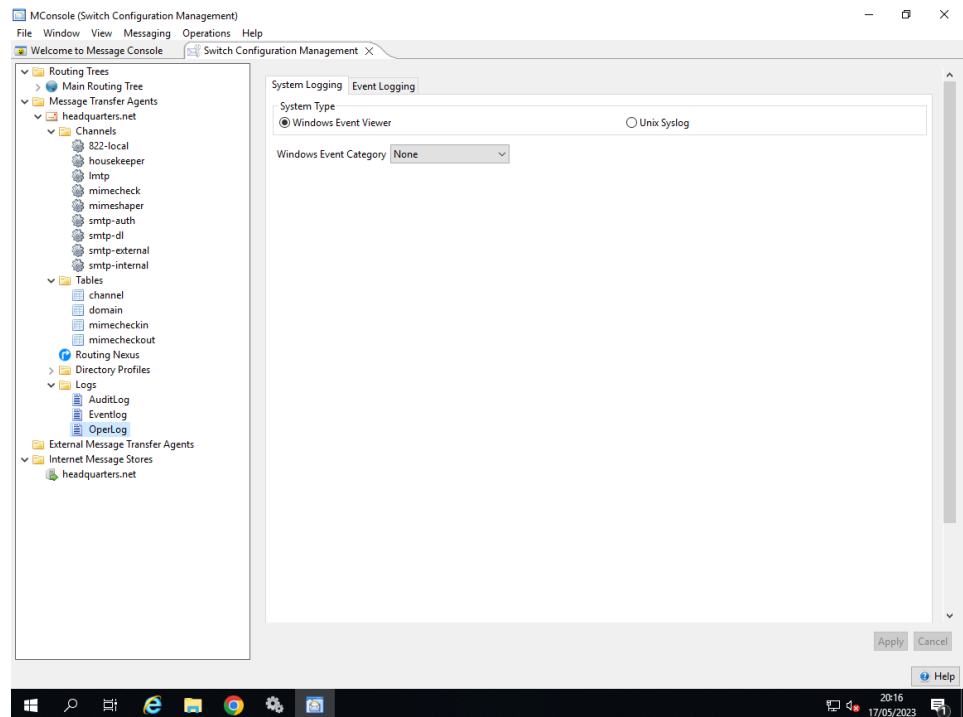


The **Set values for Audit Logging** page configures which of the available audit record types are to be sent to this stream, and which fields in the records are to be included. The default is for all records and all fields to be included. This page is only useful for Audit Log entries and can be found by selecting an audit logging stream, selecting the **Audit Logging** tab (or **Audit and Event Logging** if the stream subtype is AUDITS\_AND\_EVENTS) and then clicking **Edit...**

## 34.2.3 System streams

For system streams (eg OperLog), the following parameters can be configured:

**Figure 34.4. System Stream Properties page.**



- **Windows Event Log Category:** Configures which of the predefined event categories will be used for the event log.
- **Syslog config:** 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.

### **facility**

Include the name of the facility for the message.

### **firstonly**

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

### **ident**

Include the string identification of the event.

### **pid**

Include process ID.

### **severity**

Include a single letter indicating the severity level of the event.

### **stderr**

Print to `stderr` as well.

### **text**

Include the formatted text for the event.

- **Syslog facility:** The facility which should be used to log events.

The following mapping is made between the Message Switch logging levels and `syslog` levels:

Message Switch	Syslog
CRITICAL	CRIT
FATAL, ERROR, AUTHFAIL	ERR
WARNING	WARNING
NOTICE	NOTICE
SUCCESS DETAIL, INFO, AUTHOK	INFO
others	not logged

The **Event Logging** tab is common to both `file/tty` and system streams, and enables you to configure which event records are to be sent to this stream. Configuration is done by facility, where a facility can represent a layer in the ISO stack (e.g. the TSAP facility), a general area of the Isode base functionality (e.g. the `Compat` facility) or a specific area of the MTA (e.g. the `MTA_SMTP` facility). However, the `All` facility can be used to set which severity levels 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 are not available for system streams.

### 34.2.4 XMPP Logging

For XMPP logging, the following parameters can be configured:

#### Login JID

The JID which will be used to log in to the XMPP server

#### Login Password

The password which will be used when logging in to the XMPP server.

#### Login Password

The password which will be used when logging in to the XMPP server.

#### Target Type

The type of the destination JID (CHAT or MUC).

#### Target JID

The JID to which messages will be sent.

#### MUC nickname

If the target JID is a MUC, this allows the nickname to use in the MUC to be requested. Use `@HOST@` to include the local hostname.

### 34.2.5 Logging Streams based on Alert Group

This is a special way to configure a Logging Stream, which is based on the Alert View configuration. The events will be logged in this logging stream if they have at least an Alert Group that is the one configured in the Minimum Alert Level.

In the Alert View it is possible to easily configure alert levels for individual logging events. There are seven Alert Levels, which in order of increasing importance are: **Ignore Alert**, **No Alert**, **Alert 1**, **Alert 2**, **Alert 3**, **Alert 4** and **Alert 5**.

All logging events have a default Alert Level. If a logging event is deemed to be more important for a particular deployment, the alert level can be increased in the Alert View, so that, for example, an alarm is played.

When a Logging Stream is created using an Alert Group the existing Alert View configuration is used as the basis for the customised logging stream configuration, one that matches the choices made in the Alert Group.

If the configuration is later changed in the Alert View, the logging stream would have to be “refreshed” to keep the configuration in sync. This can be done by right-clicking on the



logging stream and selecting the menu option **Refresh**. This operation would use the existing Alert View configuration to update the selected logging stream configuration.

## 34.2.6 Program specific logging

By default all of the individual programs and channels which make up the MTA use the three logging streams described in the preceding sections. In some cases you may wish for the `eventlog` output for all of the programs which make up the MTA to be directed to separate files. Configuration of this is straightforward. The **Split Logging** option can be found by right clicking on the `eventlog` stream and will automatically set up individual `eventlog` child streams for each of the standard executables which make up the Isode MTA.

### 34.2.6.1 The MTA program split logging option

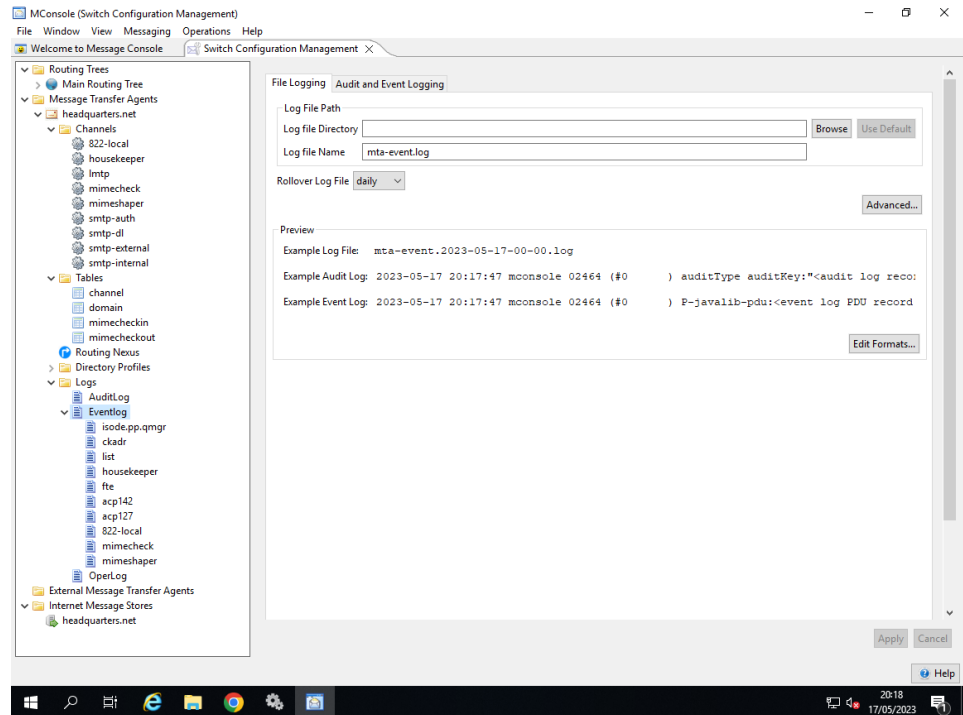
Using the **Split Logging** option causes streams to be set up so that normal logging from related programs will be diverted to the same file. [Table 34.1, “Where logging is redirected with the split logging option”](#) lists where logging is redirected for the various MTA programs. As the pathname given is relative, the files will be created in (`LOGDIR`).

**Table 34.1. Where logging is redirected with the split logging option**

Program	Logfile name
ckadr	<i>ckadr.log</i>
local, mail, sendmail	<i>local.log</i>
smtp	<i>smtp.log</i>
smtpsrvr	<i>smtpsrvr.log</i>
list	<i>list.log</i>
housekeeper	<i>housekeeper.log</i>
isode.iaed	<i>iaed.log</i>
fte	<i>fte.log</i>
p1file	<i>p1file.log</i>
p3deliver	<i>p3deliver.log</i>
p3server	<i>p3server.log</i>
x400mt	<i>x400mt.log</i>
x400p1	<i>x400p1.log</i>
acp127	<i>acp127.log</i>
acp142	<i>acp142.log</i>
isode.pp.qmgr	<i>isode.pp.qmgr.log</i>
mhsshaper	<i>mhsshaper.log</i>

[Figure 34.5, “Separate files for logging”](#) shows what the MTA **Administrator** window (expanded) might look like if you choose the option to split logging. Under **Programs**, a program tailoring object is added for each MTA process. Under each program tailoring object is a **Logging** entry, which contains an override entry for the `normlog` variable. If you display the properties of this entry, you will see that a filename is specified. This means that for this MTA program, `normlog` logging is to be directed to the specified file instead of the MTA default file.

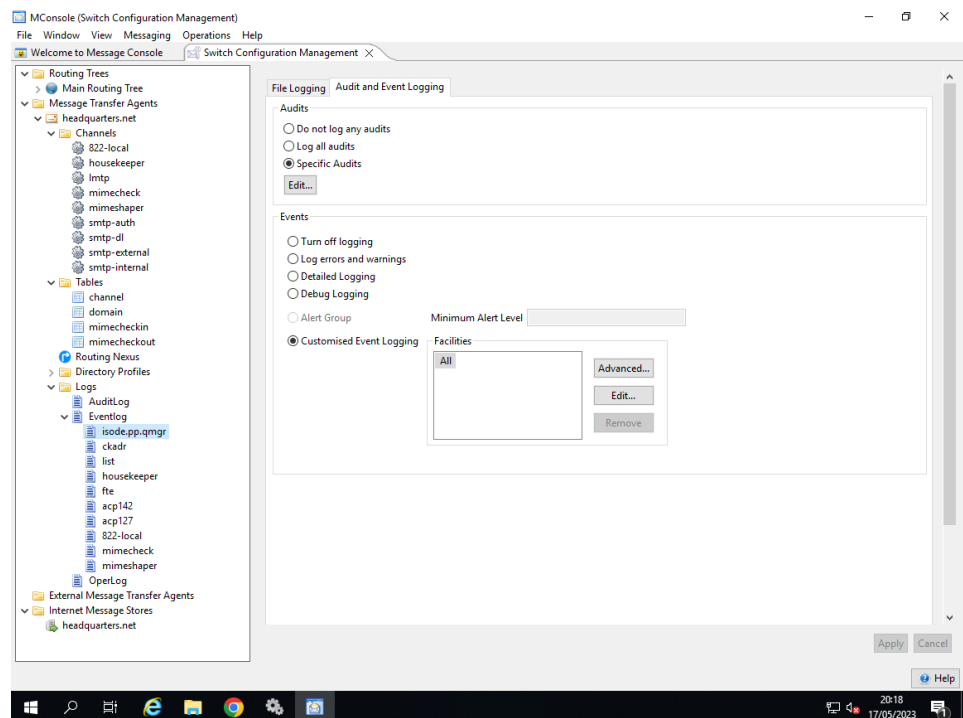
To output the `authlog` stream for this program to the new file, right click the **Logging** node for this program, to initiate the logging variable creation wizard. Select **operlog** on the first screen and enter the log filename on the second screen.

**Figure 34.5. Separate files for logging**

### 34.2.6.2 Tailoring logging for individual programs

This section describes how you can tailor individual MTA programs by creating a new program-specific stream manually.

The steps below illustrate how to tailor logging for the qmgr program.

**Figure 34.6. Tailoring qmgr Logging**

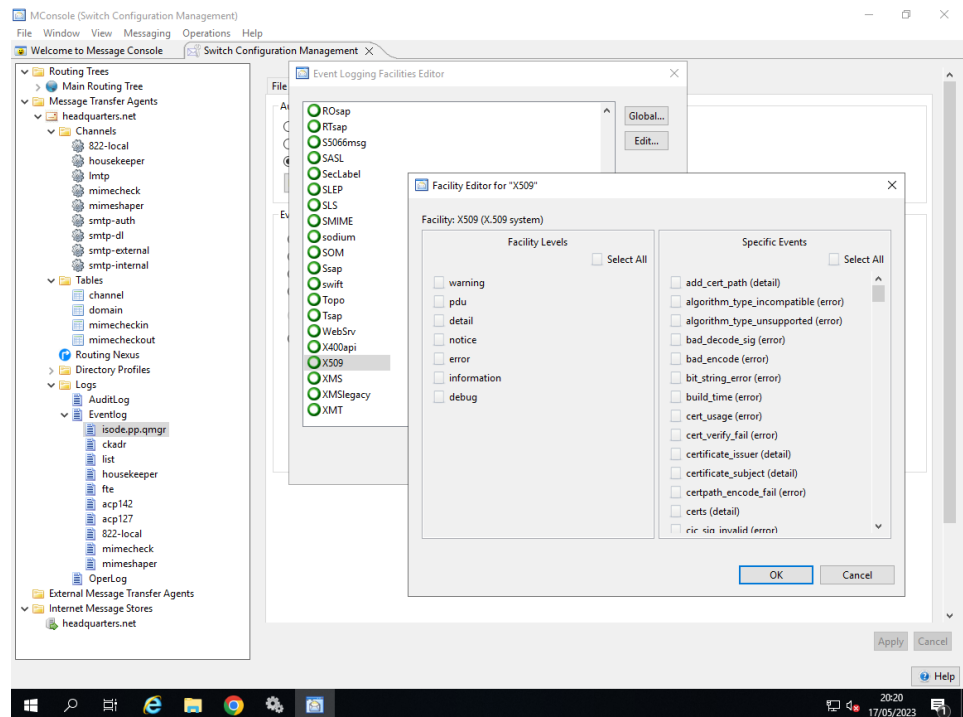
Using the event logging button you can select how the qmgr program logs. Click on **Customised Event Logging** to change the current configuration. Click on the others as a short cut to set to one of four defaults:

- **Turn Off Logging:** stop this program from logging
- **Log Errors and warnings:** do not log Notice or Information
- **Detailed Logging:** log Notice, Information and Detail
- **Debug Logging:** log Notice, Information, Detail and Debug

Using the above will configure all facilities. To set the above differently for different facilities, click on the **Advanced** button to add individual facilities to the list. You can then select the individual **Facility** and configure the required Logging level.

For example you might wish to configure details level X.509 logging. To do this, click on **Advanced**, select **X509** Facility and click on **Edit**. Set the **Facility Levels** and click on **OK**.

**Figure 34.7. Configure Facility Level logging**



## 34.2.7 Audit record format

This section describes the format of audit records written by the MTA. Each record either logs the status of inbound messages or delivery reports that are generated, or notes a message that is delivered or is checked.

Each audit record contains first the date and time at which the log entry was written, then the name of the program (or channel) which created the log entry, the program's process id, the userid under which the process was running, and the record type. Each of these fields is space-delimited. Following the record type is a sequence of space-delimited attribute type/value pairs (the type and value being colon separated), double quoted and escaped where necessary. The attribute types present depend on the record type.

For details of Audit Log key/value formats, see the *M-Switch Advanced Administration Guide*.

## 34.2.8 Setting up logging without MConsole

If you are not using MConsole for configuration of your MTA, then configuration of logging is performed by modifying the file *mtalogging.xml*. A version of this file, containing a

default logging setup, is put into (*SHAREDIR*) as part of M-Switch installation. To change this file:

1. Copy the file from (*SHAREDIR*) to (*ETCDIR*).
2. Either:
  - On Unix, run **logconfig**, from (*SBINDIR*).
  - On Windows, a shortcut will have been set up in the **Isode** folder on the **Start** menu.
3. Open the copy of *mtalogging.xml* in (*ETCDIR*).
4. The operation of the logconfig tool is identical to the log tailoring facilities within MConsole. Refer to the preceding sections of this chapter for more information.
5. When you have finished modifying the logging configuration, save the configuration file.

## 34.2.9 How logging works

### 34.2.9.1 Record types

All Isode applications can generate two types of log records during normal execution: audit records and event records.

Audit records are used to record ‘auditable events’ – Message Switch startup and shutdown, for example. Audit records do not have a severity level associated with them and have a well-defined format, so that they can be easily parsed.

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.

### 34.2.9.2 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, 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 xmpp type, where events are sent to an XMPP server.

### 34.2.9.3 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:

- 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 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.

### 34.2.9.4 Logging configuration

Information about output stream configuration is stored as XML data. All Isode applications will load the XML contained in the file *logtailor.xml*, if it exists, at startup. The search path for *logtailor.xml* is first (*ETCDIR*), and then (*SHAREDIR*). The 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. For Isode Message Switch components, this is contained in the *mtalogging.xml* file, located in either (*ETCDIR*) or (*SHAREDIR*).

When using the Message Switch in a Directory-based configuration, the master version of this configuration is actually held in the Directory, and is editable using MConsole. The *isode.tailord* process then downloads this information and creates an *mtalogging.xml* file for other processes to read.

When using the Message Switch in a table-based configuration, the *mtalogging.xml* file must be edited directly, using the standalone *logconfig* GUI.

---

## 34.3 Queue Manager

As MConsole is used to monitor and control Queue Manager, it may be helpful to consider the role of Queue Manager (*isode.pp.qmgr*) before describing the MConsole application.

The Queue Manager performs three basic functions:

- Manages the message queue by scheduling sequences of programs (channels) to process the recipients of a message
- Connects to the Directory holding the MTA configuration and downloads the information required to generate the *mtataylor* file and *mtalogging.xml*
- Acts as a the server for the SOM protocol which enables clients such as MConsole to send commands and receive responses.

The Queue Manager operates normally without reading the queue on disk. However, it does read the queue on disk on startup and periodically, to check for any messages submitted which it does not know about. At the same time, the Queue Manager will remove files from the queue which are not associated with a message and which are older than a configurable age.

### 34.3.1 How Queue Manager starts a channel

When a channel is invoked by Queue Manager (*qmgr*), the channel connects back to the *qmgr* and is given work to do by it. Although *qmgr* controls the channel, it is the channel which accesses the queue directly.

Some channels are not started by the *qmgr*; these include:

- inbound protocol channels such as the SMTP server (*smtpsrvr*),
- inbound protocol channels such as *x400p1* and *p3server* (these last two being started by the *iaed* listener)
- *fteserver*, *acp142* and *p1file* channel, which run as daemons or services waiting to be given work to do
- API gateway applications, which connect to the *qmgr* to be given work to do.

### 34.3.2 Cleanups, timeouts, and warnings

When all recipients for a message have been processed, the message is deleted from the queue by the last channel processing the message. (However, you can configure the MTA to leave messages on the disk). The MTA calculates a latest delivery time for each message. If that time is reached, the message is non-delivered. The operator can also force non-delivery or deletion of a message using MConsole. If warnings are configured, when

the message has been in the queue for the warning interval, a warning message is generated and sent to the message's originator. Multiple warnings can be generated at the warning interval.

When action on a message is required, such as generating a non-delivery or delivery report, or a warning message, this action is performed by the housekeeper channel.

### 34.3.3 Queue Manager controls and tailoring

The Queue Manager program has some command line flags which can be used to control its behaviour when it starts. On Microsoft Windows, these can be passed as service arguments.

`debug`

'debug' mode. Normally the program will detach, or attempt to run as a service on Windows. This flag prevents this, and enables the queue manager to run in the foreground. This must be the first argument.

`-s`

Single shot mode. The program will see if the tailoring information needs to be updated from the Directory, and create a new *mtataylor* file etc. if required. It will then exit. This is useful for checking the *mtataylor* file without starting the MTA.

`-D`

Disable all outbound channels. This sets all outbound channels into the disabled state when the queue manager starts. They can be enabled using the console, perhaps for just some channels.

`-S`

Disable submission on all inbound channels. Submission can then be enabled using the console for certain channels.

`-b`

Run the queue manager in "benchmark" mode. See the *M-Switch Advanced Administration Guide* for more details.

The queue manager checks the Directory server for configuration changes at the interval set by the **Configuration reload interval** (*qmgr\_config\_time*). The queue manager listens on two different addresses. The Channel address (*qmgr\_chan\_address*) is used by channel programs, and the SOM address (*qmgr\_som\_address*) is that used by MConsole and the EventViewer.

The queue manager will scan the queue on disk at an interval which can be set in the **Queue Manager** tab (*qmgr\_load\_interval*)— it defaults to 6 hours (*qmgr\_load\_interval*). Files which are not part of a valid message will be removed if they are older than the **Trash lifetime** (*trash\_lifetime*).

### 34.3.4 Tuning the Queue Manager

The Queue Manager has to:

- decide which message and recipients to process next
- decide when to process the message
- control starting channel processes
- create connections to peer MTAs
- deal with retrying after temporary errors.

There are a number of controls which can be used to tune the behaviour of the queue manager.

The default configuration is optimised to ensure that the system on which it is running is not overloaded, rather than to maximise performance and throughput. For a very busy

system it is important that you read this section if you need the queue manager to process messages as quickly as possible.

The simplest way to configure the queue manager to optimise for throughput is to change the following values:

- **Maximum operation rate** (`qmgr_oprate_max`): to about 2000
- **Idle connection timeout** (`qmgr_conn_hold_time`): to about 5 (seconds)

### 34.3.4.1 Overall processing

You can limit the number of channel process which run at any time using **Maximum channel processes** (`qmgr_max_chanproc`).

However, the main limit on the overall processing is normally through the **Maximum operation rate** (`qmgr_oprate_max`). The operation rate is the rate at which messages are processed by channels, and will scale roughly with the use of the local CPU and disk resources. Therefore, for a given hardware system, there is a natural limit. If the current operation rate exceeds the operation rate limit, then the queue manager will delay processing the message until the rate has fallen. The operation rate which is used depends upon the message priority.

The maximum operation rate applies to the highest priority messages, and the limit which is applied is reduced in the **Reserve operation rate** (`qmgr_oprate_reserve`) band. This is expressed as a percentage of the maximum operation rate. The 'unreserved' fraction of the operation rate is available for messages of any priority. The object of this is to enable a systems administrator to ensure that there are hardware resources available for messages of higher priority.

### 34.3.4.2 Self tuning

The queue manager keeps data about performance which it uses to do 'self-tuning' to control how many copies of channels to create, or how many connections to open to a given peer MTA. The numbers of these may fluctuate, particularly when the queue manager has recently started.

It is possible to control manually some aspects of this. The channel tailoring allows you to configure a maximum number of channel processes. You can also configure the maximum number of connections to, or from, a peer MTA. The channels have default values, which can be overridden in a per-MTA peer connection.

### 34.3.4.3 Error processing

When temporary errors occur, the queue manager will attempt the operation again after an interval. There are three types of errors: channel, MTA and message. The basic time intervals for these scale as 1:1.5:2. The base time is set using the **Basic channel delay** (`qmgr_delay_time`). If the error recurs, the delay is increased up to a maximum multiple of the basic time interval. That maximum can be controlled by the **Error count limit** (`qmgr_errcount_limit`).

For example, with the defaults of 60s and 10, if repeated failures to connect to a peer MTA would result in delay times of 90s, 180s, 270s, etc. but the interval would not increase beyond 900s.

### 34.3.4.4 Other controls

**Channel start attempt timeout** (`qmgr_proc_start`) is the time limit for a channel process which has been started by the queue manager to make its connection to the queue manager. If this time is exceeded, the queue manager will assume a channel error, and retry later.



**Idle connection timeout** (`qmgr_conn_hold_time`) is the default time a connection is kept open when there are no more messages for the peer MTA. Keeping the connection open avoids the cost of reconnection if another message arrives for the peer after a few seconds. This can be overridden at the channel level and also for individual peer MTAs.

**Idle channel timeout** (`qmgr_chan_hold_time`) is the time a channel process is kept running when there are no messages for that channel. This avoids the process startup cost, if another message for the channel becomes available in a few seconds.

**Channel shutdown timeout** (`qmgr_shutdown_time`) is the time the queue manager will wait when it is shutting down for channel processes to shut down cleanly. If they do not shut down in this time, they may be terminated in an unclean fashion.

**MTA connection usurpation heuristics** (`qmgr_usurp_param`) control how the queue manager decides to use a channel process already connected to one MTA for messages to another MTA when there are messages for the connected MTA. This depends on the priority difference between the messages for the two MTAs. By default, the connection will be taken over for messages of higher message priority.

---

## 34.4 Dealing with errors

When the MTA does not appear to be working correctly, there are two main sources of information:

- MConsole will have information on errors for channels, MTAs and messages.
- The log files will record error conditions

The event viewer can be used to monitor or review log files. Alternatively the `ntail` program can be used to monitor several files. This program is much like running the Unix `tail(1)` command with the `-f` flag, but on several files at once.

The most common reason for problems submitting messages is a problem with routing either the originator or recipient addresses. The `ckadr` program can be used to check the routing of addresses.

If you have updated the configuration, you should check that the `mtataylor` file has been updated. If you have changed table files, `dbmbuild` should be run to update the table database. Note that the queue manager only checks for changes at intervals.

It should not be necessary to restart the queue manager when the configuration changes.

The most common reason for messages not being transferred to a peer MTA is that the local MTA cannot connect. There should be information about the reason for the failure in the log file. Sometimes this is not clear, and you may need to contact the administrator of the other MTA to see why they do not like your connection attempts.

### 34.4.1 Delivery reports

Delivery reports are generated in a two-stage process. First a channel which decides to generate a delivery report for a message (for example, because one of the recipient addresses is invalid) adds report generation information to the subject message. The housekeeping channel then takes the subject message and generates an X.400 report or a NOTARY DSN, and submits this as a new object to the message queue.

An X.400 report generated by the housekeeping channel has a similar envelope structure to a message. The main difference is the report content. If the report is carrying returned content, this is similar to the way in which messages carry their original content.

A NOTARY DSN generated by the housekeeping channel is merely a multipart message, with MIME type as specified by the NOTARY RFCs. If the DSN is carrying returned content, this is similar to the way in which messages carry their original content.

If an X.400 report is transferred or delivered to an Internet MTA or user, the housekeeping channel performs the conversion.

NOTARY DSNs can be converted into X.400 reports.

### 34.4.2 Receipt notifications

These are handled by the UA, and are therefore beyond the scope of the MHS. However IPNs are converted to MDNs by M-Switch as they pass through the MIXER gateway.

---

## 34.5 What to run each night

In addition to running the startup script whenever the machine starts up, there are other things you should run regularly, ideally each night. For example, if you build the MTA tables from other sources, such as the NRS database, local system databases or other external sources, then running **dbmbuild** each night is a useful thing to do. All the nightly routines can be run from a shell script, conventionally named something along the lines of *pp.night*. The sequence of operations should be:

- Backup the day's logs and archive files (see [Section 34.5.1, "Saving logs"](#)).
- Remove outdated logs and archive files (see [Section 34.5.2, "Isode Maintenance"](#)).
- Collect statistics.
- Import new table source data files if necessary.
- Build new tables by running **dbmbuild** (if using tables of type dbm).

### 34.5.1 Saving 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.

The *auditlog* output can be processed to show statistics. There is a Tcl script in the *tools* directory called **statp**. This script processes the *auditlog* output and generates a report providing information on numbers of messages processed, throughput, viruses detected etc. The script can easily be customized to match your particular requirements.

### 34.5.2 Isode Maintenance

Isode Maintenance is a script that can be easily configured to delete, move or compress old log files, archives and any other temporary file generated by Isode software.

The goal is to periodically remove files that are no longer needed, to prevent the disk from filling up with old files and causing operational problems.

Features

- The age of the files can be configurable in an overall way, i.e., files older than 90 days.

- The age of the files can be overwritable on a per-project and module level, i.e., DSA log files older than 30 days.
- The age of the files can be overwritable for trace type, i.e., trace log files older than 10 days.
- Three actions are configurable for file that matches the criteria: delete, move, compress-and-move.
- The script does not delete any file or directory that is not under the configured log or archive directories.
- The script can be configured to compress M-Switch archive directories.

### 34.5.2.1 Running the Isode Maintenance script

On Linux, the script is `(SBINDIR)/isode-maintenance.sh` and on Windows it is `(BINDIR)/isode-maintenance.bat`.

The program can be run from the command line and from Cron on Linux and from Scheduled Tasks on Windows

### 34.5.2.2 Configuration File

The Isode Maintenance script is configured using a properties file called `(ETCDIR)/maintenance.properties`. The Isode Maintenance script requires this file to exist for it to work, but the file is not created by default.

A sample file suitable to be edited is shipped, the name is `(ETCDIR)/maintenance.unix.sample` for Linux systems and `(ETCDIR)/maintenance.windows.sample` for Windows systems.

Here is an example of a Isode Maintenance configuration file (*maintenance.properties*).

```
LogDir=/var/isode/log
OverallMaximumAge=90
OverallAction=Delete
MaximumAge.MTA=60
MaximumAge.DSA=30
MaximumAge.MBOX=120
MaximumAge.MS=60
MaximumAge.Trace=7
Archive.MTA=/var/isode/archive
```

This configuration file would make the Isode Maintenance script examine every log file under `/var/isode/log`, and then delete all log files that are:

- older than 120 days, in the M-Box category
- older than 60 days in the MTA category
- older than 30 days, in the DSA category
- older than 7 days in the Trace category
- older than 90 days for any other type of files. For example, temporary files and directories generated by M-Switch, etc.

This configuration file would also make the Isode Maintenance script delete every file and directory under `/var/isode/archive` that is older than 60 days (since the `MaximumAge.MTA=60`).

If instead of deleting files we want to move them to another location, then we need to add these lines to the *maintenance.properties* file:

```
Archive.MTA.Action=CompressAndMove
Archive.MTA.MoveTarget=/extra/isode/archive
```

The above configuration would also make the Isode Maintenance script compress and move every directory (and therefore every file under those directories) under `/var/isode/archive` that is older than 60 days, (since the `MaximumAge.MTA=60`) and `Archive.MTA.Action=CompressAndMove` to `/extra/isode/archive` (since `Archive.MTA.MoveTarget=/extra/isode/archive`).

---

## 34.6 Dump/Restore and CleanIsode Utilities

A set of command line utilities which simplify the process of dumping (i.e. backing-up) and restoring all of the elements which make up a Messaging Configuration are provided as part of the Isode release. During the restoration process, scripted edits to the Directory-based configuration data can be performed. This allows a "canned" configuration to be adapted to a specific installation, for example by replacing hostname values.

Various configuration and saved option files (and Registry settings, when using Windows) are created during the use of Isode software. When an Isode release is deinstalled from a system, these items are left behind. A **CleanIsode** command line script is provided which will tidy up these extra items prior to release deinstallation, so that the system is left completely clean afterwards.

Note: where user-specific files or settings are involved (e.g. the *isode-bindprofile*, user certificates or Java Preferences), the scripts will only act on those files belonging to the current user.

### 34.6.1 Running the DumpIsode script

The **DumpIsode** script allows a complete Messaging Configuration to be dumped into a single **zip** archive. The script will:

- Perform an LDIF dump of the DSA, starting at the DIT location specified on the script's command line. Typically, this would be the root of your Messaging Configuration, but it could also be the root of the DIT, in which case the DSA's own configuration, changelog and cross/subordinate reference knowledge entries are ignored.

X.400 Message Store index entries are omitted from the dump by default but can be included by specifying an additional command line switch.

- Dump the Isode Service configuration information held in the Registry.
- Dump any other installation-specific information which is held in the Registry.
- Save Java Preferences information for Isode applications.
- Save the system configuration files - i.e. everything under (*ETCDIR*).
- Save any user-specific configuration files - e.g. their *isode-bindprofile* file, plus certificates etc.
- Generate an XML manifest file recording information about the dump, including checksums for the various dump files.
- Create a **zip** archive containing all of the files generated above.

The command line for the **DumpIsode** script takes the following switches:

- b or --bindProfile *<profile name>*  
Specifies the name of the Bind Profile to use. Either a Bind Profile or LDAP credentials (see below) must be specified.
- w or --password *<password>*  
Specifies the password used to bind to the Directory or to decrypt the contents of the user's Bind Profile file.
- p or --port *<port>*  
The port on which the DSA is listening, defaulting to the standard value of 19389. This only needs to be specified if a Bind Profile is not being used.
- h or --hostname *<hostname>*  
The hostname on which the DSA is listening, defaulting to localhost. This only needs to be specified if a Bind Profile is not being used.
- u or --userdn *<userdn>*  
The DN with which to bind to the DSA. This only needs to be specified if a Bind Profile is not being used.
- ldap  
Use LDAP to access the DSA (default). This only needs to be specified if a Bind Profile is not being used.
- dap  
Use DAP to access the DSA. This only needs to be specified if a Bind Profile is not being used.
- c *<configdn>*  
Specify the DN of the Messaging Configuration to dump. If not specified, the root of the DSA's DIT will be used.
- z *<zipfile>*  
The path of the **zip** file to be created.
- f *<dump location>*  
The path of a location in which the dump will be created. This is an alternative to creation of a **zip** file: either a -z or -f switch must be specified when running the script.
- v or --verbose  
Enable verbose logging.
- i or --indexentries  
Include Message Store index entries in the dump.
- l or --list  
Instead of performing the dump operation, list the Bind Profiles and any knowledge about Messaging Configurations which is stored in the user's Bind Profile file.

### 34.6.1.1 Sample DumpIsode Command Lines and Outputs

- List the contents of the BindProfile.

```
DumpIsode -w secret -l
Available Directory Bind Profiles are:
Display name = ldap.isode.com,
    address = ldap://ldap.isode.com:389,
    binding as cn=DSA Manager,cn=DSA,ou=System,o=Isode

Display name = localhost,
    address = URI+0000+URL+itot://localhost:19999,
    binding as cn=DSA Manager,o=users,o=messaging
Known Messaging Configurations for this Profile are:
    cn=Messaging Configuration,o=messaging
    cn=Mixer Messaging Configuration,o=messaging
    cn=AMHS Messaging Configuration,o=messaging
```

- Dump the whole configuration, use the DSA identified as "localhost" in the BindProfile. (See the above output for the available Display Name values).

```
DumpIsode -b localhost -w secret -v -z /tmp/dump.zip
Not dumping message index entries
Dump location is /tmp/dump24
Running LDIF dump from root .....done, dumped 50151
                                         records, errors = 0
Wrote Java preferences to /tmp/dump24/prefs.xml
Wrote system configuration files from /etc/isode to
                                         /tmp/dump24/systemConfig.zip
Wrote user configuration files from /root/.isode to
                                         /tmp/dump24/userConfig.zip
Output is /tmp/dump.zip
```

## 34.6.2 Restoring from Backup using RestoreIsode script

### 34.6.2.1 Using RestoreIsode to create a DSA

RestoreIsode requires a DSA into which to restore the configuration dump using DumpIsode. This can be a DSA which already exists or you have created manually. See [Section 34.6.2.2, “Using RestoreIsode with an existing DSA”](#) if this is what you wish to do.

A simpler alternative is to allow RestoreIsode to create a DSA and set it up suitably for use as the repository for the Configuration.

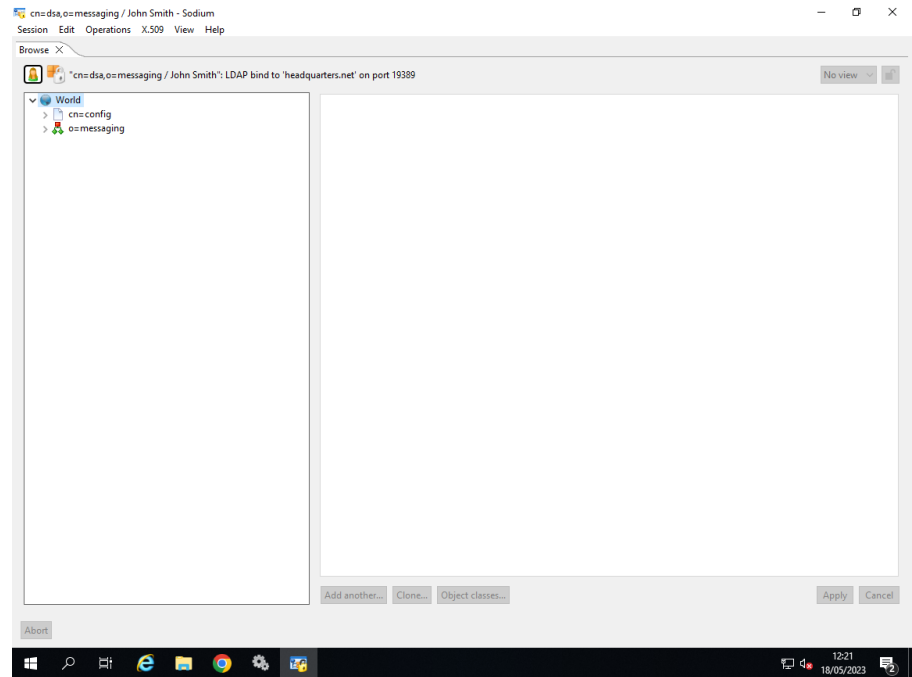
```
# RestoreIsode -u "cn=dsa manager,cn=users,o=messaging"
-w secret -z /tmp/zip -o o=messaging -r "o=new messaging"
-v -C -D 19999 -L 19389 -P /var/isode/dsa-db-restored
-N "o=new messaging" -s /home/tc/SubstitutionSpec.xml
Restore zip file is /tmp/zip
Running LDIF load..done, loaded 94, errors = 1
Loaded service information from Manifest
Existing license file /etc/isode/license.dat
renamed to /etc/isode/lic9169188758892653181.bak
Unzipped system configuration files to /etc/isode/ successfully
Unzipped user configuration files to /root successfully
```

### 34.6.2.2 Using RestoreIsode with an existing DSA

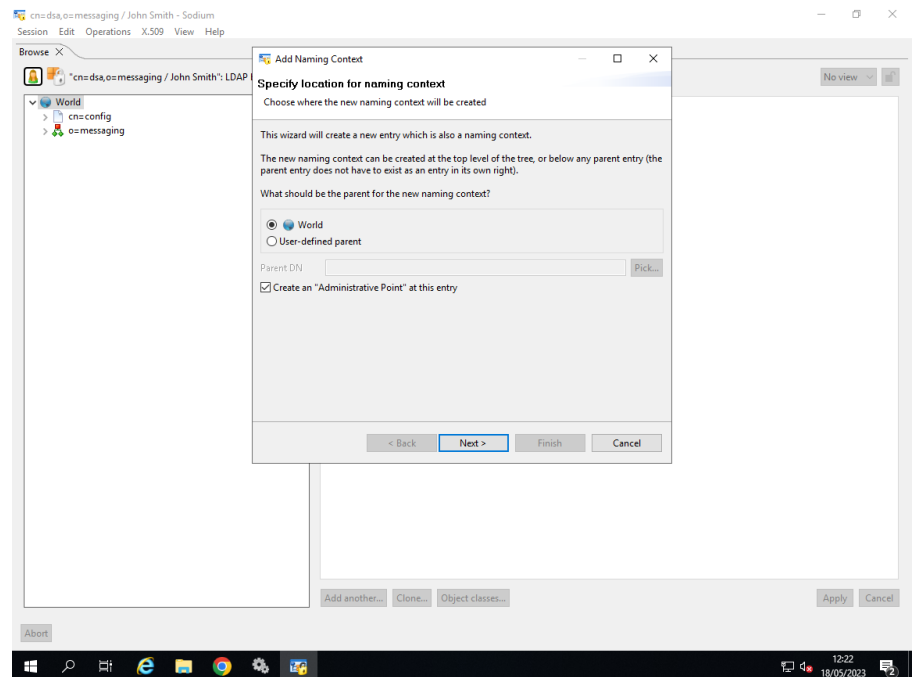
The **RestoreIsode** script performs the reverse of the **DumpIsode** script.

The following steps are required:

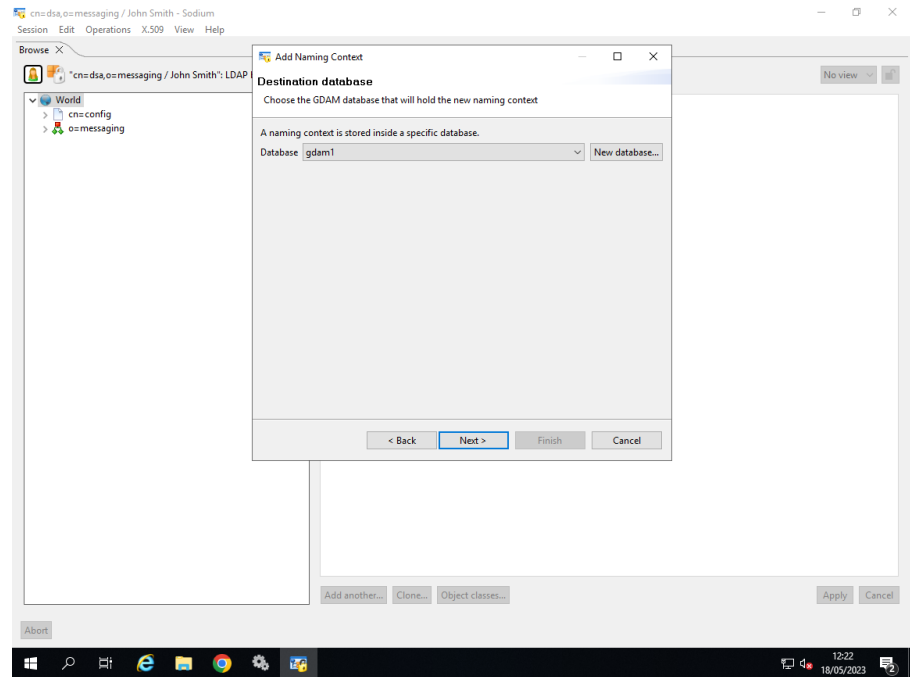
- You must create a DSA. There are several ways of doing this. This is covered in [Section 4.4, “Starting MConsole”](#). The simplest way is to start MConsole and use the "Create DSA and Messaging Configuration" option, but only complete the first of these two steps.
- Exit from MConsole.
- Start Sodium and create a new Naming Context and Admin Point as follows:
  - Right Click on **World** and select "Add Naming Context and Entry".

**Figure 34.8. Add Naming Context**

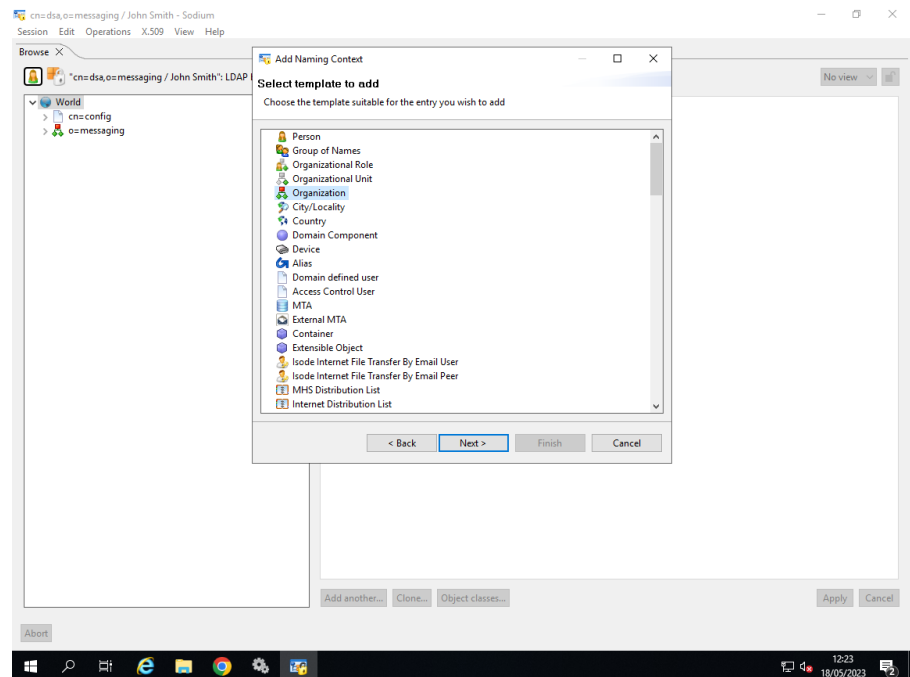
- Check the option to "Create an Administrative Point at this entry" and click on **Next**.

**Figure 34.9. Add Naming Context and Administrative Point**

- Use the existing database (gdam1), which is the default, and and click on **Next**.

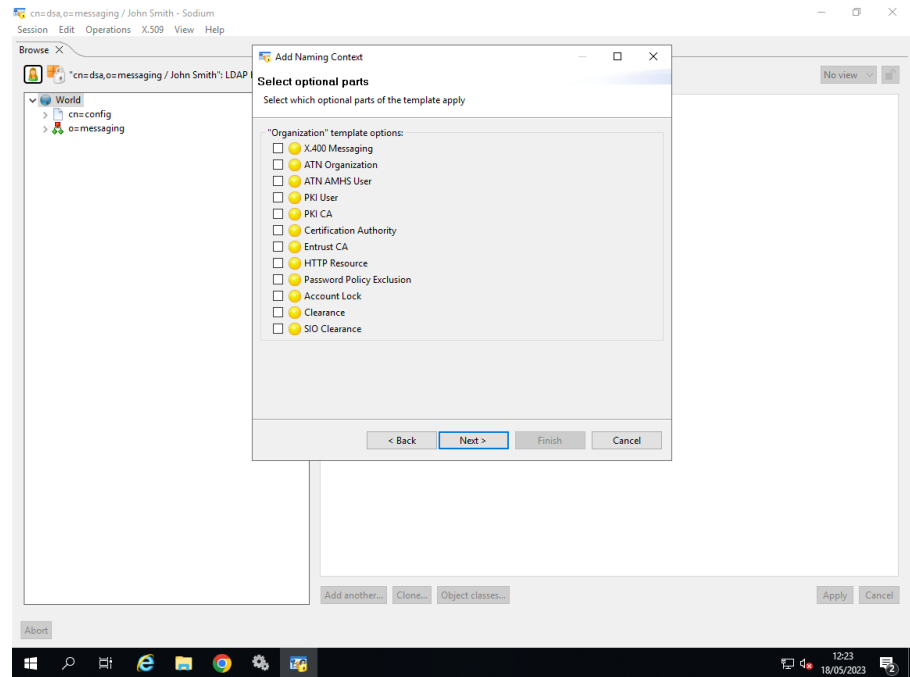
**Figure 34.10. Select Database**

- Select a template then click **Next**.

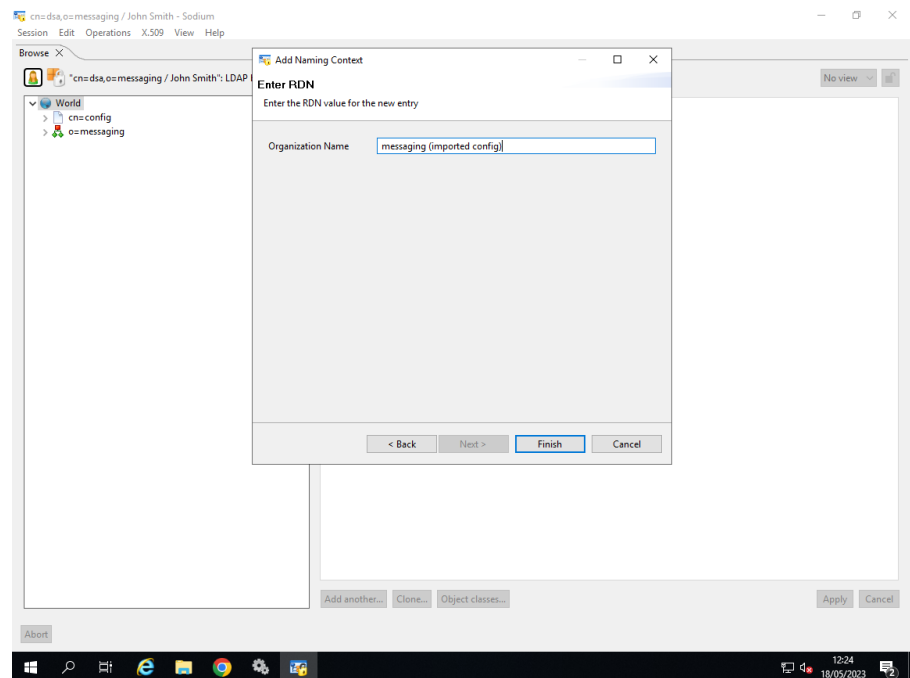
**Figure 34.11. Optional Parts**

- Optional Parts are not needed, so just click on **Next**.



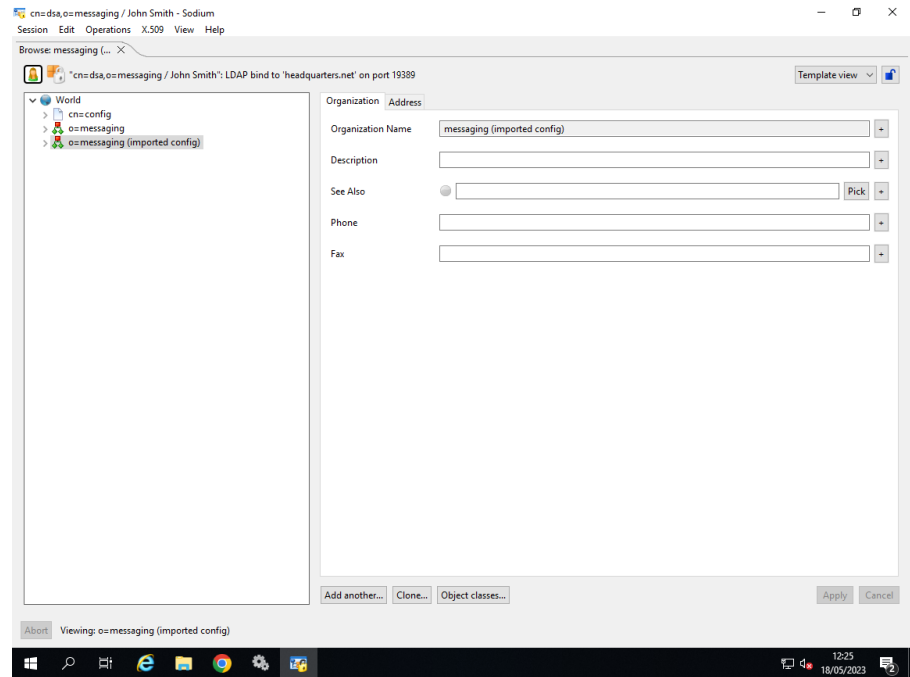
**Figure 34.12. Optional Parts**

- Name the new Naming Context Entry (any meaningful value is OK) which sets the RDN, and click on **Next**.

**Figure 34.13. Name the new Naming Context Entry**

- Finally and click on **Add** to actually create the new Naming Context.

The new Naming Context is now in the DIT as shown below.

**Figure 34.14. Completed Naming Context**

- Run restore, using a command line such as the following:

```
[root@linos restore]# DumpIsode -l -w secret
Available Directory Bind Profiles are:

Display name = cn=dsa,o=messaging / Ian R,
address = ldap://linos.isode.net:19389,
binding as cn=Ian R,cn=Users,o=messaging

[root@linos restore]# RestoreIsode -b
"cn=dsa,o=messaging / Ian R"
-w secret -z `pwd`/centos64-2.zip -s `pwd`/SubstitutionSpec.xml
-v -u secret -o "o=messaging (R16.7)" -r
"o=messaging (imported config)"
Restore zip file is /opt/isode/restore/centos64-2.zip
Running LDIF load..done, loaded 94, errors = 1
Problem removing or rewriting entry for DN:
  o=messaging (imported config) Delete(o=messaging (imported config))
  failed : Update (not on non-leaf)
Loaded service information from Manifest
Existing license file /etc/isode/license.dat
  renamed to /etc/isode/lic9169188758892653181.bak
Unzipped system configuration files to /etc/isode/ successfully
Unzipped user configuration files to /root successfully
```

- Start mconsole.
- Recreate your bind profile (which is overwritten during the RestoreIsode) by selecting **File** → **DSA Bind Profile Editor** and recreate your bind profile.
- Start the Welcome View and click on "Connect to an existing Messaging Configuration...".
- Select **View** → **Configuration** → **Authenticated Entities** and edit the imported DSA Manager User (i.e. not the one as which you are currently bound) adding all the Access Control groups, except Application Server.

- Select **View** → **Configuration** → **Authenticated Entities** and edit the Isode Application Server adding the Access Control group Application Server.
- Edit your bind profile by selecting **File** → **DSA Bind Profile Editor** and change the user in the credentials to be the imported DSA Manager. Click on **Connect**.
- Select **View** → **Configuration** → **Switch Configuration Management**
- In the Switch Configuration Management view, select the **MTA** → **Lookup** → **Connection Details** → **Host** and change to be the host on which the MTA holding the switch configuration is running.
- In the Switch Configuration Management view, select the **MTA** → **Lookup** → **SASL Server Configuration** → **Password** and enter the value used in your substitution spec file. This value is the password to be used when binding to the DSA.
- In the Switch Configuration Management view, select the **MTA** → **Lookup** → **SASL Server Configuration** → **DN** for Simple Bind and click on **Pick** to select the Isode Application Server User in your imported configuration.
- In the Switch Configuration Management view, select the **MTA** → **Security** and select the **Legacy TLS configuration**.
- Right click on the MTA and select **Create a New mtaboot.xml file**.
- You should now be able to start your MTA.

### 34.6.2.3 Running the RestoreIsode script

- Perform DN relocation and attribute value substitution on the contents of the LDIF dump. DN relocation is performed by specifying a portion of the original DN namespace which is to be replaced and the value with which to replace it. This relocation will then be applied to all DN-syntax attributes within the LDIF dump. The usual reason for doing this is to relocate the whole Messaging Configuration. Value substitution for other syntaxes is performed by specifying a *substitution specification file* as a parameter to the **RestoreIsode**. The format of this file is described in detail below.
- If an existing license file (*license.dat*) or activation file (*activate.dat*) is already present, rename it (to a temporary filename) so that it is not overwritten by the version being restored.

The command line for the **RestoreIsode** script takes the following switches:

- n or --dontOverwriteBindProfile  
Don't overwrite bind profile
- d or --dryRun  
dry run which just prints out actions
- b <profile name>  
Specifies the name of the Bind Profile to use. Either a Bind Profile or LDAP credentials (see below) must be specified.
- w <password>  
Specifies the password used to bind to the Directory or to decrypt the contents of the user's Bind Profile file.
- p <port>  
The port on which the DSA is listening, defaulting to the standard value of 19389. This only needs to be specified if a Bind Profile is not being used.
- h <hostname>  
The hostname on which the DSA is listening, defaulting to localhost. This only needs to be specified if a Bind Profile is not being used.
- u <userdn>  
The DN with which to bind to the DSA. This only needs to be specified if a Bind Profile is not being used.

- ldap  
Use LDAP to access the DSA (default). This only needs to be specified if a Bind Profile is not being used.
- dap  
Use DAP to access the DSA. This only needs to be specified if a Bind Profile is not being used.
- z or --zipfile *<zipfile>*  
The path to the **zip** file containing the dump which is to be restored.
- f or --dumplocation *<dump location>*  
The path to a location containing a dump which will be restored. This is an alternative to specification of a **zip** file: either a -z or -f switch must be specified when running the script.
- v  
Enable verbose logging.
- l or --list  
Instead of performing the restore operation, list the Bind Profiles and any knowledge about Messaging Configurations which is stored in the user's Bind Profile file.
- o or --original *<dn>*  
The component of the original base DN to replace.
- r or --replacement *<dn>*  
The value with which to replace the component specified with the -o switch.
- s or --substitutionFile *<filename>*  
The full path to a substitution specification file.
- x or --prefix *<restore prefix>*  
Specifies a prefix to be prepended to the file paths used when restoring the contents of the dump. This allows the dump to be restored into a set of temporary directories so that the contents can be inspected.

These flags are associated with RestoreIsode when being used to create a new DSA.

- C or --CreatedDSA  
Use this flag to create a new DSA rather than restore to an existing DSA.
- D *<DAP port for the new DSA>*  
Use this flag to set the DAP port for the new DSA .
- L *<LDAP port for the new DSA>*  
Use this flag to set the LDAP port for the new DSA .
- P or --dsaDbPath *<path for the new DSA database>*  
Use this flag to set the directory path of the new DSA's database.
- N or --namingContext *<Naming Context to create in new DSA>*  
Use this flag to set the path of the new Naming Context under which the Messaging Configuration is to be restored.

#### 34.6.2.4 Format of Substitution Specification

A Substitution Specification file (i.e. the argument to the -s command line argument) defines a set of attribute value substitutions which will be performed by the **RestoreIsode** script.

The file is structured as XML, and contains one or more substitution elements. Each element is tagged as *subs*. The set of attributes within each substitution element is:

- `syntax="syntaxName"`

Apply this substitution to any attribute of the specified syntax.

- `type="attributeType"`

Apply this substitution to any attribute of the specified type.

- `original="attributeValue"`

Apply this substitution to an attribute of which has the specified value. Wildcard matching may supported, depending on the syntax concerned. If no "original" tag is supplied, all attributes of this type or syntax will be selected.

- `replacement="attributeValue"`

Gives the replacement attribute value to be used.

- `relocate="yes"`

Apply the same DN relocation which is being performed on entry names to this attribute type or syntax. This only has any effect for DN or routingTreeList syntax attributes.

There is a limited set of attribute syntaxes for which substitutions are available. These are:

- presentationAddress (only the HostName within the NetworkAddress component)
- routingTreeList
- CaseIgnoreIA5String
- CaseIgnoreString
- Password
- DN
- MTAInfo
- RTSCredentials

### 34.6.2.5 Sample RestoreIsode Command Lines and Outputs

- Restore into the DSA identified as "localhost" in the BindProfile, without performing any substitutions. Restore configuration files into a temporary location.

```
RestoreIsode -b localhost -w secret -v -z /tmp/dump.zip -x /tmp/r1
Restore zip file is /tmp/dump.zip
Running LDIF load.....done, loaded 48575, errors = 0
Loaded service information from Manifest
Unzipped system configuration files to /tmp/r1/etc/isode/
                                                    successfully
Unzipped user configuration files to /tmp/r1/root successfully
```

- Restore into the DSA identified as "localhost" in the BindProfile, performing any substitutions based on specified substitution specification file.

```
RestoreIsode -b localhost -w secret -v -z /tmp/dump.zip \
                                                    -s /home/root/sub.xml
Restore zip file is /tmp/dump.zip
Running LDIF load.....done, loaded 18775, errors = 0
Loaded service information from Manifest
Unzipped system configuration files to /etc/isode/ successfully
Unzipped user configuration files to /root successfully
```

## 34.6.3 Running the CleanIsode script

The **CleanIsode** script can be used to tidy up those data items and settings associated with an Isode release which are not removed when packages are deinstalled. The net result is that after package deinstallation, the system will be returned to the state that it was originally in.

The script will:

- Save a copy of a license file to somewhere user-specific, if one is present.
- Remove any Isode services. These will need to have been stopped before the script is run.
- Remove the contents of (*ETCDIR*).
- Remove the contents of (*DATADIR*).
- Remove the any user-specific Isode data files (e.g. the user's *isode-bindprofile* file).
- Remove any other Isode-specific Registry entries.
- Remove any temporary or configuration files associated with **XUXA**.
- Remove any Java preferences for Isode products.

The **CleanIsode** accepts a small number of command line options:

-f

Do not ask the user whether they are sure that they want to run the script.

-t

Run in "test" mode - show the actions which would be taken.

# Chapter 35 SNMP

This chapter guides you through the way in which you can configure M-Switch to act as an SNMP Agent.

---

## 35.1 SNMP overview

SNMP can be used in M-Switch to monitor SNMP managed objects.

M-Switch does not speak SNMP directly, instead it communicates with an SNMP Master Agent locally on the host on which it is running. It does this using a protocol derived from SNMP called AgentX, which allows the processes to handle those specific parts of the MIB which they understand and support.

Typically, AgentX is performed over a UNIX socket, present on the filesystem at */var/agentx/master*. M-Switch requires this to be available, with permission to connect. This is normally accomplished by changing the permissions on the directory containing the socket.

On Windows, M-Switch uses a TCP connection to the loopback address on port 705. The Microsoft-shipped master agent does not support AgentX, instead using a proprietary DLL based interface for subagents. However, it is possible to obtain the NetSNMP master agent for Windows which supports both AgentX and Microsoft subagents.

---

## 35.2 Master agent configuration

### 35.2.1 Configuring the SNMP master agent

Most UNIX systems are shipped with Net-SNMP agents, which need configuring as described below to communicate with the Isode sub-agents. If Net-SNMP is not installed, you will need to obtain and install the relevant packages.

Edit the */etc/snmp/snmpd.conf* file. If necessary, add the line:

```
master agentx
```

A section configuring an “isode” community should then be added. Note that the network specification (1.2.3.0/24 in the example) should be set appropriately for your system.

```
## sec.name source community
com2sec localsystem localhost isode
com2sec mynetwork 1.2.3.0/24 isode

## group.name sec.model sec.name
group IsodeRWGroup v1 localsystem
group IsodeROGroup v1 mynetwork
#
group IsodeRWGroup v1 otherv3user
```

```
# ...

## incl/excl subtree mask
view all included .1 80

# Finally, grant the IsodeRWGroup group read-only access to the view.

## group.name context sec.model sec.level prefix read write notif
access IsodeRWGroup "" any noauth exact all none none
```

The master agent should now be restarted using the normal OS tools. This should cause the creation of a UNIX-domain socket with appropriate permissions:

```
# ls -l /var/agentx/master
srwxr-xr-x 1 root root 0 Nov 29 10:56 /var/agentx/master
```

To verify the operation of the master agent, the **snmpwalk** command can be used to request the public managed objects from the master agent:

```
% snmpwalk -v1 -c public localhost 1.3.6.1.2.1.1
SNMPv2-MIB::sysDescr.0 = STRING: SunOS host 5.10 Generic_118833-
17 sun4v
SNMPv2-MIB::sysObjectID.0 = OID: NET-SNMP-
MIB::netSnmpAgentOIDs.10
[...]
```

Microsoft supply an SNMP agent on Windows which is incompatible with AgentX, so Isode recommends that Microsoft's agent be disabled and a Net-SNMP agent be installed instead. The instructions below will result in a configuration supporting (insecure) SNMPv1; in order to get secure SNMPv3 support the Net-SNMP documentation should be consulted.

After stopping and disabling Microsoft's SNMP agent using the Windows Services control panel, the Net-SNMP installer should be downloaded from:

<http://www.net-snmp.org/download.html>

If the version including encryption support is downloaded, an additional OpenSSL installer is also needed, which is available from:

<http://www.slproweb.com/products/Win32OpenSSL.html>

After installing the package (or packages) the agent needs to be registered using the shortcut provided in the Windows **Start** menu.

A new text file should be created at *etc\snmp\agentx.conf*, relative to the directory that Net-SNMP was installed in. The contents should be:

```
agentXSocket tcp:localhost:705
```

Another new text file should be created at *etc\snmp\snmpd.conf*, relative to the directory that Net-SNMP was installed in. The contents should be:

```
rocommunity public master agentx
```

The Net-SNMP agent can now be started using the Windows Services control panel.



## 35.3 M-Switch

### 35.3.1 M-Switch MIBs

M-Switch supports the NETWORK-SERVICES-MIB (RFC2248) and MTA-MIB (RFC2789) MIBs. Copies of these MIBs are included in the Isode release packages, but these are functionally identical to the versions of the same MIBs which are included as part of the Net-SNMP package, so they do not need to be installed for the `snmpwalk` to produce useful results.

### 35.3.2 Enabling SNMP monitoring

To make M-Switch attempt to connect to a master agent, the main **MTA** page in MConsole contains a radio button: **Enable SNMP sub-agent**.

M-Switch will not list all associations within the association table of NETWORKSERVICES-MIB, since some associations are not relevant to mail operations. In particular, connections to a DSA are not listed although if the DSA is M-Vault (and is also configured as an SNMP agent), they will be listed there. M-Switch also has full support for the MTAMIB (RFC2789), and lists all internal channels as MTA groups, as described in RFC2789, pp10-11. In particular, this means that message processing statistics are therefore gathered per-channel, and can be monitored per-channel.

### 35.3.3 Checking SNMP monitoring

Once M-Switch has been configured as a sub-agent, its operation can be verified using the `snmpwalk` command to examine the relevant areas of the MIB space:

```
% snmpwalk -v1 -c isode localhost application
NETWORK-SERVICES-MIB::applIndex.1 = INTEGER: 1
NETWORK-SERVICES-MIB::applName.1 = STRING: statler.isode.com
NETWORK-SERVICES-MIB::applDirectoryName.1 = STRING:
    cn=smtp.isode.com,cn=Messaging Configuration,ou=MHS,o=messaging
NETWORK-SERVICES-MIB::applVersion.1 = STRING: pp 17.0v0-1
    [f5bb5e7b04] (centos7-gcc63) of Wed  2 May 01:45:29 BST 2018
NETWORK-SERVICES-MIB::applUptime.1 = Timeticks: (1461) 0:00:14.61
NETWORK-SERVICES-MIB::applOperStatus.1 = INTEGER: up(1)
NETWORK-SERVICES-MIB::applLastChange.1 = Timeticks: (0) 0:00:00.00
NETWORK-SERVICES-MIB::applInboundAssociations.1 = Gauge32: 0
NETWORK-SERVICES-MIB::applOutboundAssociations.1 = Gauge32: 0
NETWORK-SERVICES-MIB::applAccumulatedInboundAssociations.1 =
    Counter32: 112001
NETWORK-SERVICES-MIB::applAccumulatedOutboundAssociations.1 =
    Counter32: 1163
NETWORK-SERVICES-MIB::applLastInboundActivity.1 = Timeticks:
    (26843461) 3 days, 2:33:54.61
NETWORK-SERVICES-MIB::applLastOutboundActivity.1 = Timeticks:
    (26842661) 3 days, 2:33:46.61
NETWORK-SERVICES-MIB::applRejectedInboundAssociations.1 =
    Counter32: 0
NETWORK-SERVICES-MIB::applFailedOutboundAssociations.1 =
    Counter32: 162
[...]
```

```
% snmpwalk -v1 -c isode localhost mta
MTA-MIB::mtaReceivedMessages.1 = Counter32: 2960
MTA-MIB::mtaStoredMessages.1 = Gauge32: 2
MTA-MIB::mtaTransmittedMessages.1 = Counter32: 1365
MTA-MIB::mtaReceivedVolume.1 = Counter32: 83340 K-octets
MTA-MIB::mtaStoredVolume.1 = Gauge32: 6 K-octets
MTA-MIB::mtaTransmittedVolume.1 = Counter32: 47618 K-octets
MTA-MIB::mtaReceivedRecipients.1 = Counter32: 4188
[...]
```

### 35.3.4 High-priority event monitoring using SNMP

Production systems may require that certain high-priority events, such as “server has stopped” are not simply logged to a file, but instead cause SNMP traps to be sent to interested SNMP management tools.

To enable SNMP traps to be issued, the *etc/snmp/snmpd.conf* file (on Windows) or */etc/snmp/snmpd.conf* file (on UNIX) needs to be edited to contain the line:

```
trap2sink hostname public
```

# Chapter 36 Message Audit Database

This chapter describes the audit database, which provides a central repository for information about messages passing through your system. It enables you to monitor and manage areas of congestion and where errors have occurred, from a remote location.

---

## 36.1 What does it do?

Information about the messages passing through your system is stored in a Database Management System (DBMS). This is an SQL database. You can use the same database to hold information from multiple Isode MTAs, giving you an overview of all the traffic through your Message Transfer System (MTS).

The information in the database can then be viewed and managed using MConsole.

The applications provide the following features.

- Message Statistics

This application displays statistical information about all of the MTAs providing data to the audit database. You can view summary information or perform more sophisticated searches based on specified recipients or viruses.

- Message Viewing (MConsole only)

MConsole is capable of accessing the MTA message archive, Quarantine or MTA queue, and display the content of the message. Please note that the content of the message is never stored in the Message Audit Database. If the message is not in the Quarantine or the MTA Queue, it will only be available if you have set up message archiving.

- Message Tracking

This application can be used to follow messages through your system. You can search using most message attributes; for example, subject or originator.

- Acknowledgement Tracking (MConsole only)

This application can be used to correlate messages and acknowledgements (reports and read receipts) in your system. You can search using time period; message type; acknowledgement status.

- Message Quarantine

Message Quarantine is used to manage messages that have been quarantined (usually by an MTA used to filter messages that are spam or contain viruses or other malware). Information describing the configuration necessary to identify and quarantine suspected spam messages is in [Chapter 40, Content Checking](#).

With the Message Quarantine application, messages that have been placed in quarantine can be viewed, deleted or released for subsequent delivery to the intended recipient.

## 36.2 How does it work?

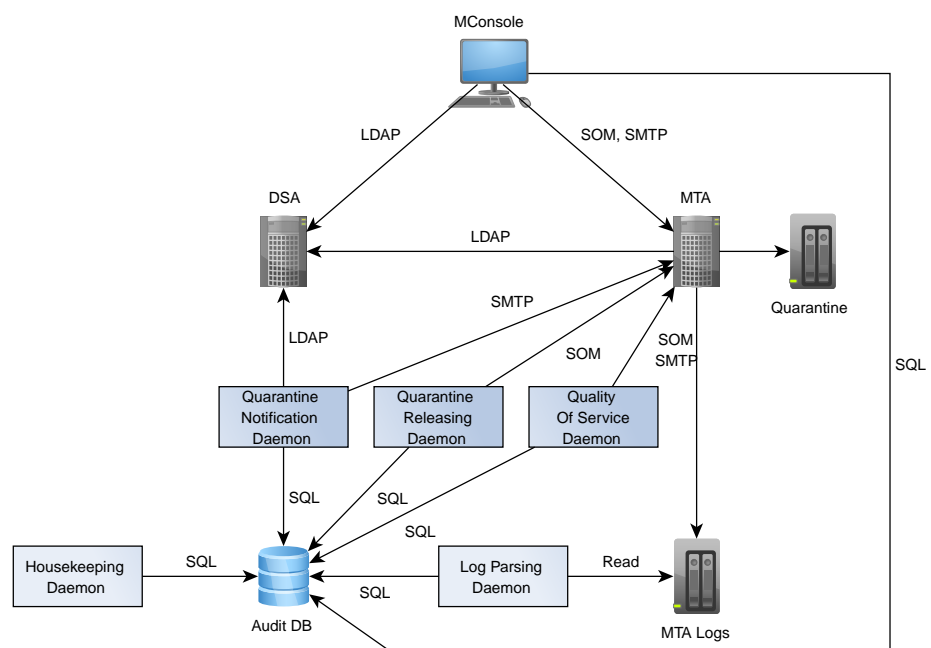
The MTA records audit information in audit log files (sometimes known as the stat log). The contents of these files are read by the Log Parsing daemon, which populates an SQL database with the information contained in the audit logs.

Information from the Message Audit Database can be viewed using the Isode MConsole which has the following views:

- **Message History** to display a quick summary of messages that are sent and received by the the MTA for the day. There is an option to modify the time limits to widen or narrow down the search.
- **Message Tracking View** which allows information about transfer and delivery of messages from the Message Audit Database to be displayed.
- **Quarantine View** which allows information about quarantine from the Message Audit Database to be displayed.
- **Acknowledgement View** which allows correlation information about messages and their acknowledgements held in the Message Audit Database to be displayed.
- **Message Transfers History** displays the transfer history for messages.
- **Statistics** displays statistical information from all the MTAs that feed data to the audit database. This view provides a summary of various statistics like messages, recipients, originators, viruses etc grouped by tabs. It is also possible to narrow down searches to get more specific statistics.

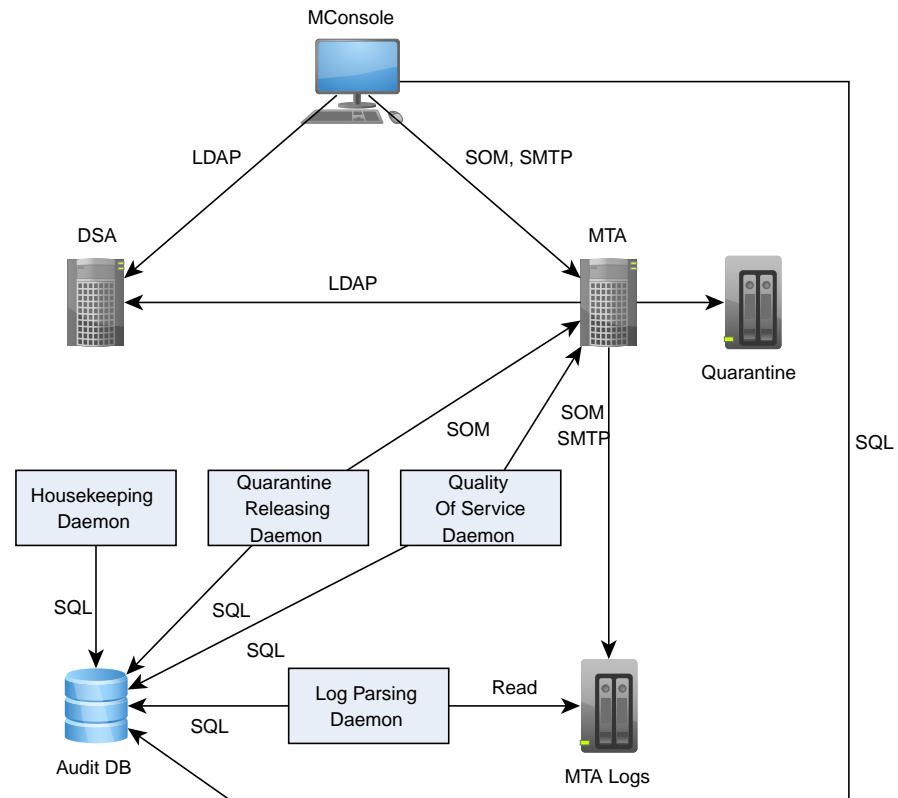
Figure 36.1, “The audit database (overview)” shows the relationships between the various components and the way that information is passed between them. The MTA, MTA log files and the Log File Parser must be physically located on the same machine as the MTA, as information is passed between them using simple file access. The other components (the audit database, other daemons/services) can be located on different systems if required.

**Figure 36.1. The audit database (overview)**



If you intend to run the Quality of Service Notification without the Quarantine Service, the setup is more simple as in [Figure 36.2, “The audit database \(Quality of Service\)”](#):

**Figure 36.2. The audit database (Quality of Service)**



### 36.2.1 Quarantine overview

If content checking results in a message being placed in quarantine, it is given a status of `Quarantined` in the Audit DB (see [Section 40.5.3, “Placing messages in quarantine”](#)). Using the Message Quarantine MConsole **Quarantine View**, this can be changed either to `Freed` – for the message to be resubmitted for delivery to the original recipients – or `Marked for deletion`.

- Messages with a status of `Freed` are resubmitted to the MTA. The daemon changes their status to `Resubmitted` or, if a problem is encountered, `Error on resubmission`.
- Messages with a status of `Marked for deletion` are deleted from quarantine and their status is changed to `Deleted`.

---

**Note:** Only operators, not end-users, can currently delete messages from quarantine.

---

### 36.2.2 Quality of service overview

Messages which are handled by the MTA can request Delivery Reports and/or Read Receipts. These can be positive (the MTA has delivered the message or the Recipient has read the message), or negative (the message failed to be delivered, or the recipient has declined to acknowledge receipt of the message).

Collectively Reports and Read Receipts are termed Acknowledgements.

If you have a requirement to be notified that messages which are expected to have been acknowledged by certain time, this can be provided by the Quality of Service Notification

daemon. Configuration of this service is described in [Section 36.10.1, “Quality of Service Daemon configuration”](#).

---

## 36.3 What is needed to run the Audit DB

A Basic system requires:

- An SQL DataBase Management System. Supported products are:
  - PostgreSQL
  - Microsoft SQL Server
  - HSQLDB is provided in embedded form as part of the Isode product set for evaluations or demos. It is not supported as part of a production system.
- Java.

PostgreSQL and Microsoft SQL Server are not provided by Isode and must be obtained separately.

The release notes for each package give full installation instructions for these third-party packages. You can accept all default values unless specifically instructed otherwise.

---

**Note:** When installing PostgreSQL on Windows, you will be asked if you want to connect from hosts other than local hosts. Answer **Yes**.

---

---

## 36.4 Configuring the software

The core component of the Message Audit Database system is an SQL DBMS (Database Management System) where the Audit Database is stored and processed.

The DBMS has to be configured first, as all other Audit Database components are dependent on it.

### 36.4.1 HSQLDB

Setting up the embedded SQL database for evaluations takes place automatically when starting up isode-hsqldb and parsing daemon/service.

### 36.4.2 PostgreSQL

#### 36.4.2.1 Initialisation and configuration

After successfully installing PostgreSQL, you now need to create an appropriate configuration to make it suitable for Audit DB use. Instructions are provided separately for UNIX and Windows systems.

##### 36.4.2.1.1 Configuring PostgreSQL on UNIX

1. Initialise PostgreSQL. This is carried out automatically by the installer. The database is started.

2. Edit *pg\_hba.conf* to grant access for specific hosts to audit database. The location of the file varies with PostgreSQL version and operating systems. The default location of this file for version 12 of PostgreSQL is */etc/postgresql/12/main/pg\_hba.conf*. Adding:

```
host auditdb postgres 127.0.0.1/32 trust
```

enables local applications (and daemons) to access the database as a "postgres" user. To enable other hosts simply add more entries with appropriate IP addresses. For example:

```
host auditdb postgres 127.0.0.1/32 trust
host auditdb postgres 172.16.0.134/32 trust
```

The /32 parameter means network mask - changing this gives access to address group (network) rather than single host. For example:

```
host auditdb postgres 127.0.0.1/32 trust
host auditdb postgres 172.16.0.134/16 trust
```

The above allows any system with the IP address 172.16.\*.\* to connect.

3. Restart PostgreSQL to use the new settings

```
# /etc/init.d/postgresql restart
```

#### 36.4.2.1.1 Creating an Audit Database

1. To create a database you must switch to user, to the PostgreSQL user (usually created when the PostgreSQL package was installed).

```
# su postgres
```

(You may need to set the postgres user password when you are root, as you will be asked for it at this point)

2. Run the following command:

```
# createdb -E utf8 auditdb
```

#### 36.4.2.1.2 Deleting the database

1. To delete the audit database you have to stop all applications and daemons accessing it.
2. Next as the postgres user run the following command:

```
dropdb auditdb
```

#### 36.4.2.1.3 Resetting the database

If you need to clear the Audit Database the most effective way is just to drop and recreate the database as described above.

#### 36.4.2.2 Configuring PostgreSQL on Windows

1. Install PostgreSQL

During installation you will be asked for the database superuser password, which will be required later to access the database.

2. Edit the *pg\_hba.conf* to grant access for specific hosts to audit database. Adding:

```
host auditdb postgres 127.0.0.1/32 trust
```

enables local applications (and daemons) to access the database as a "postgres" user. To enable other hosts simply add more entries with appropriate IP addresses. For example:

```
host auditdb postgres 127.0.0.1/32 trust
host auditdb postgres 172.16.0.134/32 trust
```

/32 parameter means network mask - changing this gives access to address group (network) rather than single host.

3. Reload the PostgreSQL configuration to use the new settings. This can be accessed by selecting **Programs** and then **PostgreSQL** from the **Start** menu.

#### 36.4.2.1.1 Creating the Audit Database

1. Open the **pgAdmin4** tool from the Windows Program Menu
2. Connect to the database (You will be asked for the password that you provided during installation)
3. Go down the tree and click on the **Databases** item with your right mouse button and select **New Database....**
4. Enter **auditdb** as a database name.
5. On the **Definition** tab, ensure that the selected encoding is UTF8 and click the **OK** button.

#### 36.4.2.1.2 Deleting the database

1. To delete an audit database you have to stop all applications and daemons accessing it first.
2. Open the **pgAdmin4** tool from the Windows Program Menu
3. Connect to the database (You will be asked for the password provided during installation)
4. Go down the tree and below **Databases** select **auditdb** item and with your right mouse button select the **Delete/ Drop** option.

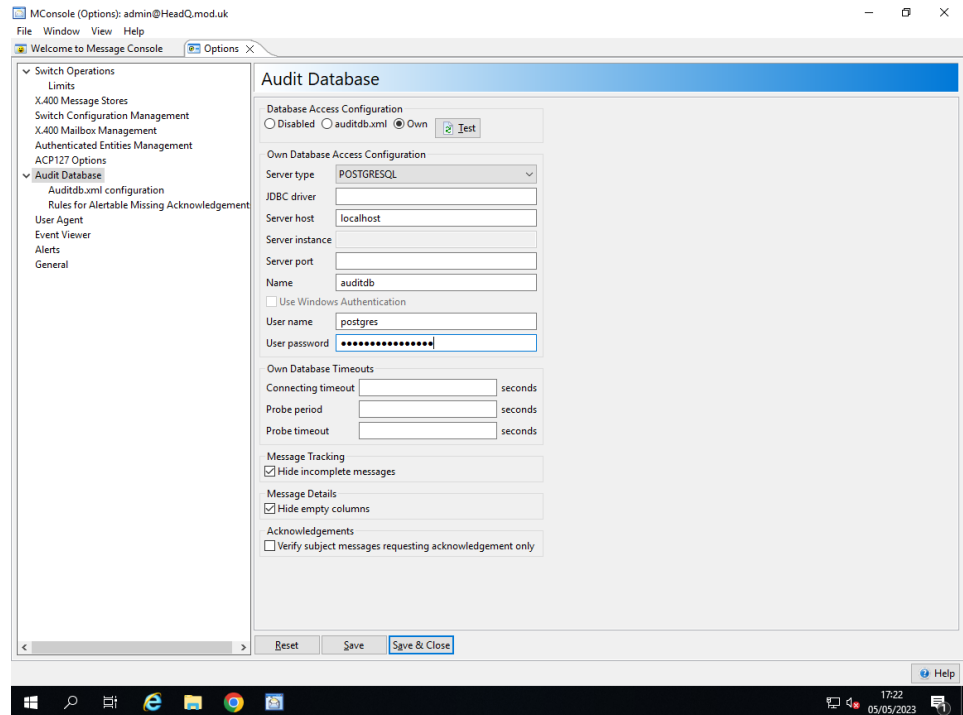
#### 36.4.2.1.3 Resetting the database

If you need to clear the Audit Database the most effective way is just to delete and recreate the database as described above.

### 36.4.2.2 Testing your newly installed PostgreSQL configuration

Start MConsole and select: **View** → **Options** → **Audit Database**. The screen looks as follows:

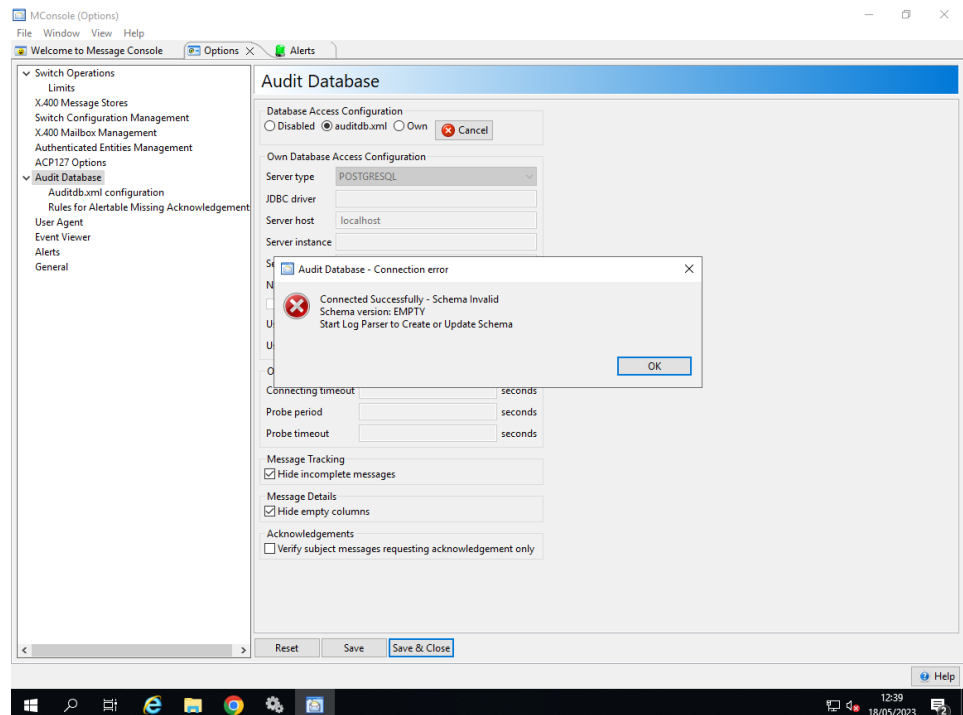


**Figure 36.3. The audit database Options View**

Set the options as in the above diagram and click on **Test**.

You should connect successfully, if not check PostgreSQL is started and try again.

If you do connect successfully you will get a further error reporting that the Audit Database is Empty:

**Figure 36.4. The audit database Options View - Connection Error**

This is because the Audit DB has not been initialised. To do this you need to carry out the instructions in [Section 36.4.4, “Configuring Audit DB”](#).

## 36.4.3 Microsoft SQL Server

### 36.4.3.1 Installing MS SQL Server

Install MS SQL Server.

During installation at "Feature Selection" wizard page select at least "Database Engine Services" and "Management Tools - Basic".

At the "Server Configuration" wizard page you must select Case-sensitive collation (use "Customize" button at "Collation" tab) to make this the default for all new databases. If you do not do this, then you must explicitly configure case-sensitive collation when creating the database.

---

**Caution:** Isode services (such as the log parser) will generate an error and fail to start if they connect to a database and detect that the database has not been configured in this way.

---

At "Database Engine Configuration" wizard page select "Mixed Mode" to allow non windows integrated authentication access and provide SA (System Administrator's) password).

### 36.4.3.2 Creating the Audit Database

1. Open the **Microsoft SQL Server Management Studio** from **Start → Microsoft SQL Server**
2. Connect to the server (You can use "Windows Authentication" or use "SA" login in "SQL Server Authentication" with password that you provided during installation).
3. Go down the tree and click on the **Databases** item with your right mouse button and select **New Database....**
4. Enter `auditdb` as a database name.
5. On the **Options** page, ensure that selected Collation is case sensitive and click the **OK** button.

### 36.4.3.3 Deleting the database

1. To delete an audit database you have to stop all applications and daemons accessing it first.
2. Open the **Microsoft SQL Server Management Studio** from **Start → Microsoft SQL Server**
3. Connect to the server (You can use "Windows Authentication" or use "SA" login in "SQL Server Authentication" with password that you provided during installation).
4. Go down the tree and below **Databases** select **auditdb** item and with your right mouse button select the **Delete** option.

### 36.4.3.4 Resetting the database

If you need to clear the Audit Database the most effective way is just to delete and recreate the database as described above.

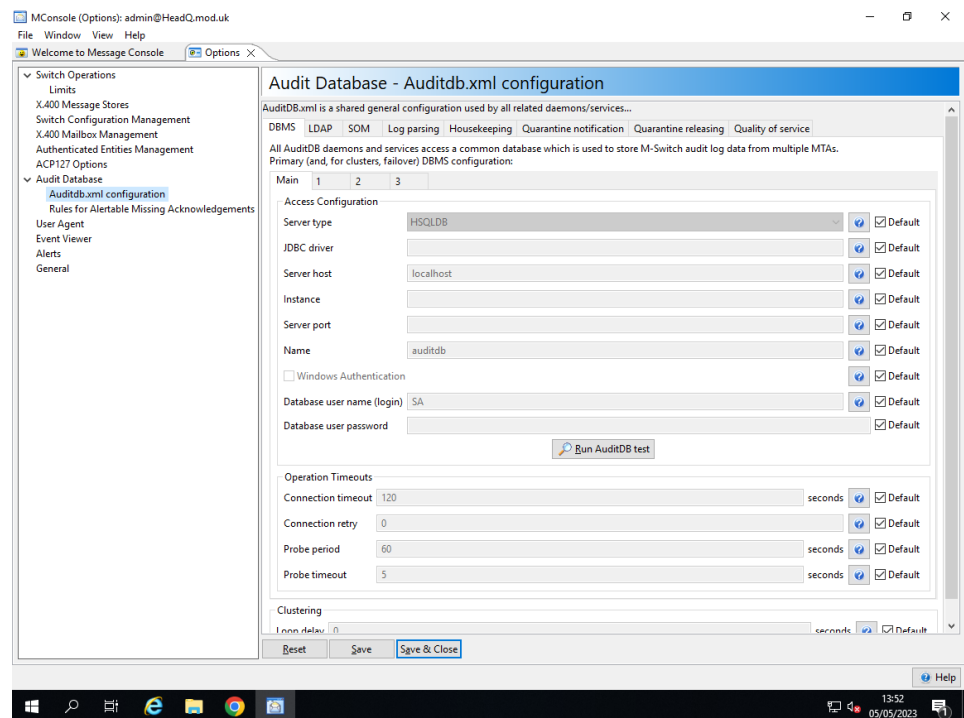
## 36.4.4 Configuring Audit DB

The Audit DB is configured using the configuration file `auditdb.xml`. This is edited using the **Options** view in MConsole. NB you must be running MConsole on the system on which you are running any Audit DB daemons you wish to configure,

### 36.4.4.1 Audit DB Configuration in MConsole

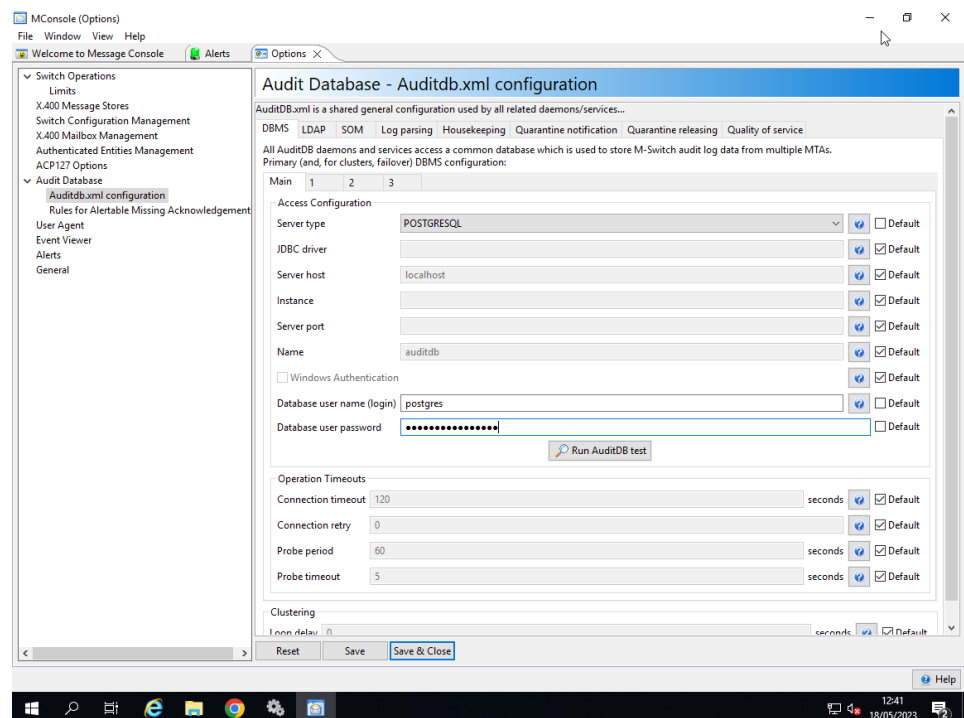
Selecting **View** → **Options** → **Audit Database** → **Auditdb.xml** results in the following configuration screen in which you can edit the values to configure Audit DB.

**Figure 36.5. The Audit Database Options View (HSQLDB)**



If you are using PostgreSQL, the **auditdb.xml** page looks like the page in [Figure 36.6](#), “The Audit Database Options View (PostgreSQL)”:

**Figure 36.6. The Audit Database Options View (PostgreSQL)**



You are recommended to use the GUI but Audit DB can also be configured directly as described in [Section 36.4.4.2](#), “Audit DB Configuration using auditdb.xml directly”.

### 36.4.4.2 Audit DB Configuration using auditdb.xml directly

All Audit Database daemons and applications share a common configuration file. An example file is provided in *ETCDIR/auditdb.xml.sample*. This should be copied to *ETCDIR/auditdb.xml*.

There are various sections in the sample file which are commented out and it may be appropriate to edit these and un-comment them. The sample file isn't meant to be deployed as-is.

You use the `<database>` option to specify which type of SQL database to use. There can be only one `<database>` option in the file, so any other sample configurations should be commented out or removed.

Simple active clustering support is available which means that an application can automatically switch between alternative database configuration(s) in the event of connection failures or operation timeouts.

Alternative configurations can be specified as subentities of "database" and may redefine any parameters from main database configuration (all other parameters are inherited) Up to 16 alternatives databases can be defined as entities named `<alternative_N />` where *N* is the number 1-16 (gaps in numbering are allowed. The number indicates precedence with 1 representing the highest precedence).

If all connection to all alternative databases fail, the switching mechanism works in a loop retrying all of them after a configurable delay. To support this, the main configuration may specify the `loopDelay` parameter. The `loopDelay` parameter is ignored in alternative configurations.

#### 36.4.4.2.1 Common options

The following attributes can be configured for all DBMSes:

`type`

Specifies the database type. Value must be one of: `hsqldb/postgresql/mssql`

`name`

The name of the database, which must be the same as is used during the database creation. `auditdb` is the expected value.

`host`

The hostname or IP address of the DBMS host. `localhost` is the expected value

`port`

The port on which DBMS is listening (Default value depends on selected DBMS type)

`user`

The user name with which to authenticate to the database

`password`

The credentials (password) associated with the user

`driver`

JDBC driver path can be specified here if none is provided within the Icode software or there are multiple versions. (empty - means using default)

`connectingTimeout`

(optional) How many seconds to wait for a connection to succeed before timing it out and giving up on the attempt. (The default value is 120 seconds)

`connectingRetry`

(optional) How many times to retry connecting to a server before assuming that it's not there. (default 0)

**probePeriod**

(optional) This setting determines how frequently to initiate a probe to check that the DBMS is still responsive (if SQL queries are taking a long time to complete). (The default is 60 seconds)

**probeTimeout**

(optional) How many seconds to wait for an SQL probe operation to complete before assuming that it has failed. Not specified or empty means don't use timeout (it may still be timed out if JDBC driver has own builtin timeouts). (The default is 5 seconds)

**loopDelay**

(optional) The number of seconds to wait, after having tried and failed to connect to all server instances, before retrying them all. Not specified or negative value means that if all server instances are non-reachable, then no further attempts will be made to reconnect.

The Audit Database daemons and applications do not require restarting after the *auditdb.xml* configuration is changed.

**36.4.4.2 PostgreSQL**

This is an example PostgreSQL Server config

```
<!-- database configuration -->
<database type="postgresql" name="auditdb" host="localhost"
port="5432" user="postgres" password="postgres" driver="" />
```

PostgreSQL specific attributes:

**type**

Value must be: postgresql

**port**

The port on which PostgreSQL is listening, which is configured in the *postgresql.conf* file (usually 5432)

**36.4.4.3 MS SQL Server**

This is an example MS SQL Server configuration.

```
<!-- database configuration -->
<database type="mssql" name="auditdb"
host="server_main.company.com" port="1433"
windows_auth="on" connectingTimeout="120" connectingRetry="2"
queryTimeout="180" updateTimeout="60" loopDelay="300">
<alternative_1 host="server_alternative_1.company.com" />
<alternative_2 host="server_alternative_2.company.com" />
</database> -->
```

MS Server SQL specific attributes:

**type**

Value must be: mssql

**port**

The port on which MS SQL Server is listening (usually 1433)

**windows\_auth**

Using Windows Integrated Authentication (if set then user and password are not used)

**36.4.4.4 HSQLDB**

This is an example HSQLDB configuration.

```
<!-- database configuration -->
<database type="hsqldb" name="auditdb" host="localhost"
port="9001" driver=""> <user>SA</user>
<password servpass:encrypt="true"></password> </database>
```

The HSQLDB specific attributes:

type

Value must be: hsqldb

port

The port on which HSQLDB is listening (9001 for embedded HSQLDB)

user

The user name with which to authenticate to the database ('SA' for embedded HSQLDB)

password

The credentials (password) associated with the user (empty for embedded HSQLDB)

---

## 36.5 Starting and Stopping the Audit Database Services

If you are using the Audit DB services, you may need the following services:

If you are using Postgres, start this as follows:

```
systemctl start postgresql
```

If you are using Microsoft SQL server, you need to consult the documentation for this in order to start it as the DBMS.

Alternatively, if you are using the Isode supplied test/demo DBMS, start this as follows:

```
systemctl start isode-hsqldb
```

Other Audit DB processes are started as follows:

```
systemctl start adb-lp
systemctl start adb-hk
```

These Audit DB services are less likely to be in use, but are included to complete the list of available services.

```
systemctl start adb-qosn
systemctl start adb-qr
```

---

## 36.6 Populating the database - Log parsing daemon

When an empty database is available (eg on initial installation), the Log parsing daemon is responsible for creating the database schema. It can then populate it with data extracted from M-Switch audit log files.

### 36.6.1 Configuration

All Audit Database daemons and applications share configuration information kept within the *auditdb.xml* file. Several configuration options within the *auditdb.xml* file affect multiple daemons including the log parsing daemon.

#### 36.6.1.1 General daemon parameters

`hostid`

`hostid` used to distinguish log entries from multiple servers (used as a component of message id in the database). Default is current host name (resolved automatically).

`runonce`

(yes/no) - The daemon will not wait for new data after processing the last file and will stop. (default is no).

#### 36.6.1.2 Log parsing daemon specific parameters

`fileinterval`

The interval which the log parsing daemon checks for new data within a log file. (default is 1 second)

`directoryinterval`

The interval which the log parsing daemon checks for new log files within the log file directory. This is used when there is nothing new in the last log file, or when there are no log files at all

`parseWords`

An MTA with archiving and indexing enabled generates an index file per message archive that contains the list of words found in the message. If this option is set, log parsing daemon will parse the index files and populate a table to store these words. This option implies a processing and storage space overhead on the database and should be enabled cautiously.

`transactionmaxlines`

Max number of lines processed after which the database transaction is committed (minimum 1 line - default 1000 lines)

`connectionrecords`

Whether to parse connection records (yes) or ignore (no). Default no.

`reportunknownattributes`

Report all unused logging fields within the daemons own log file. Default is no.

`malformedsubject`

If the daemon comes across a message whose subject field is malformed, then the subject field will be replaced by the value specified in this parameter.

---

## 36.7 Housekeeping

When the Audit Database is used in a busy production environment, it may use very large amounts of disk space.

Generally, it is not useful to retain information in the Audit Database indefinitely, and so the Audit Database Housekeeping Daemon can be configured to perform periodic purging of outdated information from the database.

### 36.7.1 Configuration

All Audit Database daemons and applications share configuration information kept within the *auditdb.xml* file. Several configuration options within the *auditdb.xml* file affect multiple daemons including the housekeeping daemon.

#### 36.7.1.1 General daemon parameters

`hostid`

`hostid` used to filter records for specific MTA. If no `hostid` value is specified (which is the default), then all MTAs are included.

`runonce`

(yes/no) - If set the daemon will not run continuously, but will make a single pass to clear old data (based on the current time), and then stop (default is no).

#### 36.7.1.2 Log parsing daemon specific parameters

`schedule`

processing schedule (cron-tab style) - default="@daily"

`age`

how many days to keep old messages before deleting them (default="7")

`batch`

When removing old messages, a series of separate delete operations is performed, with each operation deleting no more than 'batch' records. This avoids the risk of locking the database for long periods of time when using a single delete operation to delete a large number of records. (default=1000)

---

## 36.8 Tracking messages

### 36.8.1 Message Tracking Views

See for a detailed description of how Operators can use the Message Tracking Views.

The Message Tracking Views enable operators to search the Audit Database and to display the status of messages and other information.

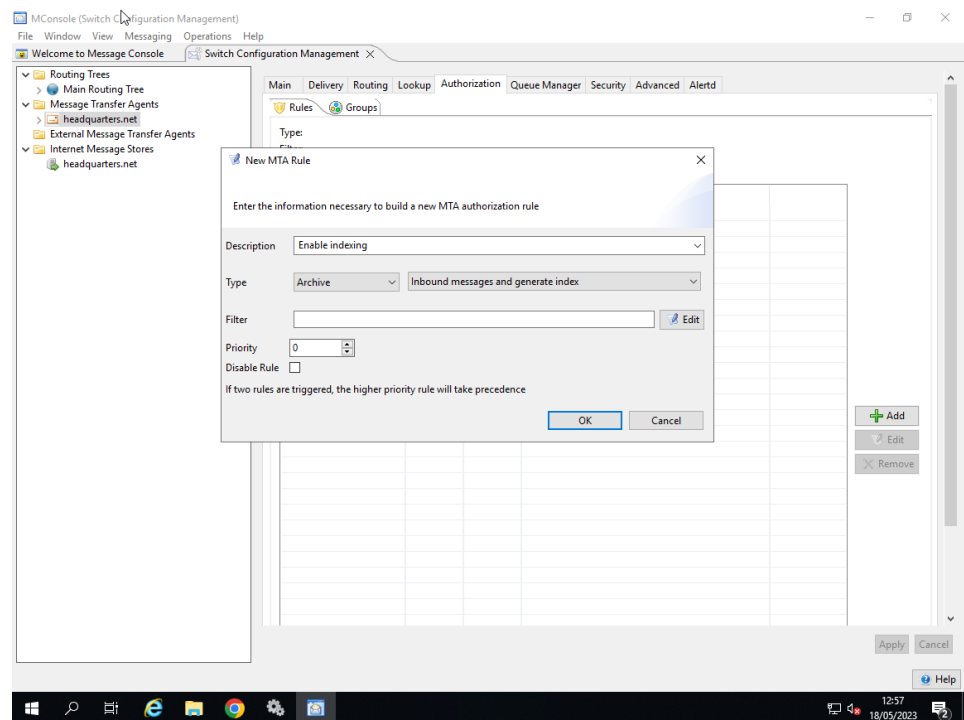
These applications allow the operator to search for specific messages using filters against most message attributes such as originator, subject etc.



## 36.8.2 Message Content

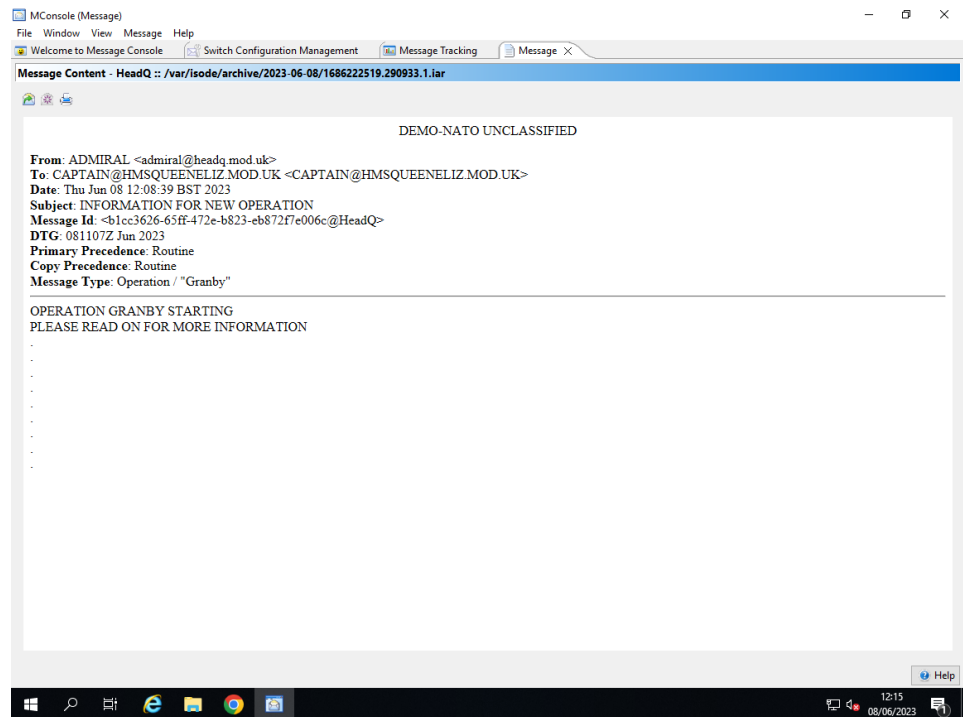
If you want to be able to display message content, you must add an archive rule and add a suitable directory (must exist and be writeable) to the archive directory option. To add the archive rule, open the **Switch Configuration Management** view, select your MTA and, from the **Authorization** tab, select **Add** in the **Rules** tab. Enter a description for the rule and select **archive** for the **Type**. Select **Inbound messages** to enable archiving for inbound messages, **Outbound messages** to enable archiving of outbound messages and **Inbound messages and generate index** to enable archiving of an index file that contains all words found in the inbound message. The index file can be parsed by the parsing daemon to populate a table and this will facilitate searching messages by message content (see [Section 36.6.1, “Configuration”](#)).

**Figure 36.7. Adding an archive rule**



To add your directory, select the **Advanced** tab, select **Archive directory** and add the path for your directory.

This is normally added for you, by default, when creating your messaging configuration. If you find that is not working or fails to create the rule, check that the archive directory is present and correct, and also has the right permissions.

**Figure 36.8. Message content view**

Right clicking each individual recipient in the right hand box provides a similar menu.

## 36.9 Quarantine Management

When a message is quarantined it is removed from the live queue and is placed in a quarantine directory. The Quarantine event is recorded in M-Switch's Audit Log. The Log Parsing Daemon will read this event, and generate a suitable entry within the Audit Database.

The message stays in quarantine until released or deleted. A messaging operator can release or delete messages from quarantine using the Quarantine Manager (MConsole **Quarantine Manager** view).

Releasing a message updates its status to be released. The Quarantine Releasing Daemon subsequently detects this status and resubmits the message into the switch using the SOM API.

### 36.9.1 Quarantine Releasing Daemon configuration

All Audit Database daemons and applications share *auditdb.xml* configuration. The Quarantine Management daemon configuration is kept within the management element.

#### 36.9.1.1 General daemon parameters

**hostid**

hostid used to distinguish log entries from multiple servers (used as a component of message id in the database). Default is current host name (resolved automatically).

**runonce**

(yes/no) - If set the daemon will not work continuously, but will process data once and then stop. (default is no).

### 36.9.1.2 Quarantine Management Daemon specific parameters

somd

SOM connection info: host, port, username and password

interval

After this interval the Quarantine Management Daemon will check the Quarantine Management Database, to see if any actions need to be performed (releasing / deleting messages, etc). This period is measured in seconds.

deletemarked

Delete messages marked for deletion. The default is no.

## 36.10 Quality of Service

The Quality of Service Daemon can be configured to generate alerts for messages which are regarded as unacknowledged - ie for which an acknowledgement is expected but has not been seen within the configured time period.

The alert takes the form of an email and an event which is logged to the Isode logging system. The email summarising the unacknowledged message.

### 36.10.1 Quality of Service Daemon configuration

All Audit Database daemons and applications share the *auditdb.xml* configuration file.

The Quality of Service Daemon configuration is kept within the *qos-notification* element.

#### 36.10.1.1 General daemon parameters

hostid

hostid used to distinguish log entries from multiple servers (used as a component of message id in the database). Default is current host name (resolved automatically).

runonce

(yes/no) - If set the daemon will not work continuously, but will process data once and then stop. (default is no).

#### 36.10.1.2 Quality of Service Daemon parameters

smtp

SMTP server into which notification messages will be submitted. If your SMTP server requires authentication (RFC 2554 (4)) before accepting mail, then you will have to setup a username and password here. (attributes to set: host, port, username, password)

template

template file name (path)

subject

test to be used as the subject of the notification

processingInterval

Time interval between querying the AuditDB to check for unacknowledged messages

processingDelay

Time interval to allow for delay

**catchupProcessing**

Time interval to allow for delay in parsers updating the DBMS

**deltaDelivery**

Time period in which a report is expected, after which it is to be regarded as unacknowledged (can be multivalued - see below)

**deltaRead**

Time period in which a read receipt is expected, after which it is to be regarded as unacknowledged (can be multivalued - see below)

**notifyMissingDelivery**

Yes/No. Send a notification when a Delivery Report is missing (i.e. late).

**notifyMissingRead**

Yes/No. Send a notification when a read receipt is missing (i.e. late).

**notifyNegativeDelivery**

Yes/No. Send a notification when a negative Delivery Report is detected

**notifyNegativeRead**

Yes/No. Send a notification when a negative read receipt is detected

The `deltaDelivery` and `deltaRead` values specify the time in which a report or read receipt (respectively) are expected, after which the message is regarded as unacknowledged. These can vary depending on message priority.

They are specified as a single value or comma separated list of numbers which are the values in seconds for the respective priorities. You can configure

- a single value, in which case this value is used for all priorities, eg : 30
- three values, in which case these apply to the non-military priorities Urgent, Normal and Non-Urgent: sample: 15,30,6
- six values, in which case these apply to the military priorities: Override(0), Flash(1), Immediate(2), Priority(3), Routine(4), Deferred(5), eg : 7,10,15,30,60,120

## 36.10.2 Configuring QoS Rules

The Quality of Service Daemon can be configured to generate notifications for messages which are deemed to be unacknowledged - ie when a Delivery Report or Read Receipt is expected, but has not been seen in the Audit DB.

Not all domains can be relied on to generate Acknowledgements (ie Reports or Read Receipts) reliably. So a facility exists to enable interworking with such domains to take place without flooding the recipient of such notification with high numbers of false alarms.

The Rules for Alertable Missing Acknowledgements enable two sets of rules to be configured which filter notifications of unacknowledged messages.

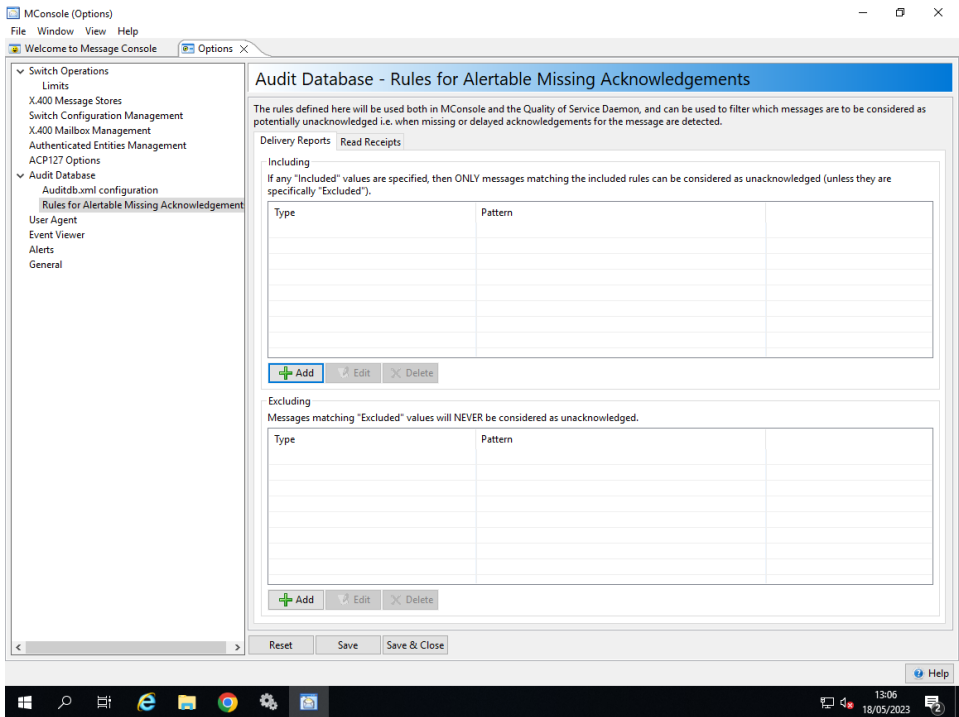
Filters can be set for

- recipient
- sending MTA
- receiving MTA
- originator

One set of rules configure filters for messages which will never be considered as unacknowledged. The second set of rules configure filters for messages only those of which match will result in notifications.

The filters support simple wildcards of \* or % to indicate zero or more characters.

Figure 36.9. The Audit Database Options View (HSQLDB)



# 36.11 Message Statistics: logstats

You can use the logstats program to parse the M-Switch Audit logs directly to obtain information about the messages processed by M-Switch.

Usage:

```
logstats [options]
```

Example invocation:

```
logstats -c "logtime(dd-MM-yyyy), count_msgout IPM Urgent,
count_msgout IPM Normal, count_msgout IPM Non_Urgent,
count_msgout DR, count_msgout NDR, size_sum IPM"
```

**Note:** The above invocation results in the default values being used.

Switches:

-lf,--logfile	parameters and processing errors and information logging file
-ll,--loglevel	[OFF, SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST, ALL] (default INFO)
-i,--input	path and file(s) (default are all audit logs)
-o,--output	output format [CSV, TSV, TAB] (default CSV)
-r,--resolution	Aggregates resolution [DAY, HOUR, MINUTE] (default DAY)

-m, --messages	max size of the message buffer (default=1000, 0=unlimited)
-c, --content	comma separated list of field definitions (default as in sample) (supports comma escaping if necessary)

The -c switch takes an argument as follows:

```
[Aggregated field [Message type selectors]
[Message priority selectors]] | Other field
```

This is passed in as a string to be passed in of comma separated Field Definition.

Valid values for Aggregated Field are:

COUNT_MSGIN	number of 'msgin' records
COUNT_MSGOUT	number of 'msgout' records
COUNT_DELIV	number of 'deliv' records
COUNT_ACK_DELIVERY	number of 'rrecip-pos' or positive 'dsn-recip' records
COUNT_ACK_NON_DELIVERY	number of 'rrecip-neg' or negative 'dsn-recip' records
COUNT_ACK_DELAYED	number of 'dsn-recip' records indicating delay in delivery
COUNT_ACK_RELAYED	number of 'dsn-recip' records indicating relay
COUNT_ACK_EXPANDED	number of 'dsn-recip' records indicating expansion
COUNT_ACK_READ	number of positive 'ipn' or 'mdn' records
COUNT_ACK_NOT_READ	number of negative 'ipn' or 'mdn' records
SIZE_SUM	sum of message sizes
SIZE_MAX	the biggest message size

Valid values for Message Type Selectors are:

IPM	X.400 message
IPN_POSITIVE	X.400 read notification
IPN_NEGATIVE	X.400 not read notification
DR	X.400 delivery report
NDR	X.400 non delivery report
IM	Regular Internet message (RFC 822)
DSN_POSITIVE	Internet message delivery report
DSN_NEGATIVE	Internet message non delivery report
MDN_POSITIVE	Internet message read notification
MDN_NEGATIVE	Internet message not read notification

Valid values for Message Priority Selectors are:

0	OVERRIDE
1	FLASH, URGENT
2	IMMEDIATE
3	PRIORITY, NORMAL
4	ROUTINE

5	DEFERRED, NON_URGENT
---	----------------------



36.12

User Agent

Built into MConsole is the ability to submit messages by copying the content of messages which were delivered, non-delivered or are stuck in M-Switch queues.

# Chapter 37 Clustering

If you have suitable clustered hardware, the Message Switch can be configured to run in a hot-standby configuration.

In this mode of operation, two or more (usually) identical server systems are linked by a shared multi-ported disk. Identical Message Switch configurations are installed on each machine, with the Message Switch's queue directory and other configuration placed on the shared disk. Clustering software ensures that only one server system has the shared disk mounted and the Message Switch processes running at any instant. Any hardware or software failure on the live system causes it to be deactivated, with a standby system being made live. IP address sharing between the systems making up the cluster means that client applications are not aware of the switchover having taken place, other than a brief interruption in service while the switchover is actually happening.

In general, the management of the cluster, switchover from live to standby system, dependency configuration and failure detection is performed by software provided by the manufacturer of the operating system being used. Isode provides scripts which allow the clustering software to start, stop and monitor the Message Switch processes, and instructions on how to configure specific types of clustering software.

---

## 37.1 Microsoft Windows

This section describes two different methods of setting up clustering using the Windows cluster manager software.

Different versions of the Windows Cluster Manager software use different terms, for example "services" are now known as "roles" in Windows 2012.

This chapter uses the term "services" throughout.

- Have 1 clustered IP address / Shared disk for all Isode services. This is simple to configure.
- Split the services up, and run them on separate IP addresses / disks. This means the services can be run at the same time on separate nodes. This is more difficult to configure.

To enable a High Availability Cluster, you need to have

- Shared disk(s). This can be either 1 shared disk for the Isode services, or multiple disks 1 for each Isode service. In addition to the above you may need extra disks / partitions for the Windows cluster manager itself.
- IP Addresses. Either 1 shared IP address for the Isode services, or multiple IP addresses for each Isode service.

### 37.1.1 Creating a simple failover cluster

These instructions explain how to create a failover cluster with M-Vault / M-Switch and M-Store X.400 all operating using the same IP address and shared disk.

1. Create your cluster environment, with at least 2 nodes.
2. Install M-Vault and M-Switch / M-Store.
3. Run the Service Configuration application on both nodes, select **Actions** → **Install Isode Services**.



4. Create a new **Empty Clustered** service, and rename it `Isode`.
5. Add the Clustered IP address to the service, and bring it online.
6. Add the shared disk.
7. Create a new DSA using M-Vault Console. Choose the shared drive for the d3-db directory. Choose the shared hostname / IPaddress for the hostname.
8. The DSA will start up automatically. Stop it.
9. Add a new generic service to the cluster, call it M-Vault.
10. Add the shared disk and ip address as dependencies for the M-Vault clustered service.
11. Change the startup parameters of the DSA using the cluster manager. The startup parameters should be `J:\d3-db` (assuming J is the shared disk).
12. Bring the DSA online.
13. Create a new M-Switch in a normal way. This includes creating configuration files within `C:\Isode`
14. Create a new switch directory on the shared disk.
15. In MConsole select the newly created MTA, then select **Advanced**. Select the **Queue directory**, and enter in the newly created directory on the shared disk.
16. Create a new *Mailboxes* directory on the shared disk.
17. In MConsole select the **Main Message Store**, set the mailbox root to be the newly created **Mailboxes** directory.

The configuration must now be shared across the nodes.

Either:

Copy `C:\Isode` to every other node.

Or:

Copy `C:\Isode` onto the shared disk.

Then alter `HKEY_LOCAL_MACHINE\SOFTWARE\Isode\Isode\16.0\libisode` so that the **datapath** value is set to `J:\Isode` (Assuming J is the shared disk). You will then need to alter the registry for every node.

18. Add generic services for:

- Isode M-Switch QMGR.
- Isode OSI Listener.
- Isode X.400 Server.

Make sure the startup parameters are blank for all of the services.

Also make sure that every service is dependent on M-Vault the IP address and shared disk.

Additionally the OSI Listener is also dependent on M-Switch Queue Manager.

Other M-Switch services such as the File Transfer By Email server and SMTP server are also dependant on M-Switch Queue Manager.

These may need to be configured depending on your configuration.

19. Bring the different services online, and check they can failover.

If you have difficulty bringing services online, make sure that you check:

- a. The service arguments for a service is correct. Typically the cluster manager will enter "Files\Isode\bin\Name of executable" into the service arguments. This is

incorrect, and in most cases should be empty. (The DSA should have the d3-db directory specified as mentioned previously)

- b. Remove any existing service dependancies. You can do this using the Isode Service configuration tool. Note this is different to resource dependancies setup within the cluster manager software

### 37.1.2 Creating an Independent Failover Cluster

These instructions explain how to create a failover cluster with M-Vault, M-Switch and M-Store X.400 operating independently of each other, each product can have its own IP address and disk.

One product can run on one node, and another product can run on a different node.

1. Create your clustered environment with at least 2 nodes.
2. Install M-Vault / M-Switch and M-Store X.400.

---

**Note:** M-Store X.400 is part of the same package as M-Switch.

---

3. Run the Service Configuration application on both nodes, select **Actions** → **Install Isode Services**.
4. Create a new **Empty Clustered Service**, and rename it **M-Vault**.
5. Add the M-Vault Clustered IP address to the service, and bring it online
6. Add the M-Vault Clustered shared disk, and bring that online too.
7. Create a new DSA using M-Vault Console.
  - a. Choose the shared drive to when creating the *d3-db* directory.
  - b. Choose the shared hostname / IP address for the hostname.
8. The DSA will startup automatically. Stop it.
9. Add a new generic service to the "M-Vault" clustered service, call it M-Vault.
10. Add the shared disk and IP address as dependencies for the M-Vault clustered service.
11. Change the startup parameters of the DSA using the cluster manager. The startup parameters should be "J:\d3-db" (assuming J is the shared disk).
12. Bring the DSA online.
13. Create a new Empty Clustered service, and rename it "M-Switch".
14. Add the M-Switch Clustered IP address to the service, and bring it online.
15. Add the M-Switch shared disk.
16. Create another new Empty Clustered service, and rename it "M-Store".
17. Add the M-Store Clustered IP address to the service, and bring it online.
18. Add the M-Store shared disk.
19. Create a new Messaging configuration using MConsole.
20. Create a new switch directory on the M-Switch shared disk.
21. In MConsole select the newly created MTA, then select **Advanced**. Select the **Queue directory**, and enter in the newly created directory on the shared disk.
22. Create a new **Mailboxes** directory on the M-Store shared disk.
23. In MConsole select the **Main Message Store**, set the mailbox root to be the the newly created *Mailboxes* directory.

Update the presentation address so that the network component reflects the shared IP address for the X.400 Message Store.

24. Copy the C:\Isode directory to all nodes.

This means that all nodes have the same *C:\Isode\switch\mtaboot.xml*, *C:\Isode\pumicetailor.xml* files.

When the MTA starts up it reads from the DSA and (re)generates *C:\Isode\switch\mtatailor.tai*

The MTA's configuration is kept within the DSA, this means that the *mtatailor* file will automatically be updated when the MTA is transferred across nodes.

25. For the "MSwitch" service within the cluster manager add generic services for:

- a. Isode M-Switch QMGR.
- b. Isode OSI Listener.

You may also need to add other services, such as the SMTP server depending on your configuration.

26. For the "MStore" service within the cluster manager, add a generic service "Isode X.400 Server"

27. Make sure the startup parameters are empty for all of the newly added services.

28. Bring the "MSwitch" service online starting with the queue manager.

Make the queue manger dependant on the M-Switch shared disk and shared IP address.

Make the other M-Switch services dependant on the shared IP address, disk and the queue manager.

This will make sure that the queue manager is stopped last, and started first.

29. Bring the "MStore" service online, and make the service dependant on the shared M-Store IP address / disk.

You should now have all of the services online to be able to send and receive messages. M-Vault, M-Switch and M-Store X.400 can run independently on each of the cluster nodes.

---

## 37.2 Centos / RedHat Linux Enterprise Linux 9

### 37.2.1 Notes

Before you start, you will need:

- Shared storage (e.g. a SAN)
- Primary Server IP Address
- Failover Server IP Address
- Shared IP Address

All of the IP Addresses must be fixed.

You must ensure that the "High Availability" option is installed on both the servers

### 37.2.2 Setting up the Cluster

In this example we will use the following Hostnames/IP Addresses:

Server	Hostname	IP Address
Primary	clusterp.isode.net	172.20.3.201

Server	Hostname	IP Address
Failover	clusterf.isode.net	172.20.3.202
Shared	cluster.isode.net	172.20.3.203

### 37.2.2.1 Add the Hosts on Both Nodes

Edit the `/etc/hosts` file on each node and add:

```
172.20.3.201 clusterp.isode.net clusterp
172.20.3.202 clusterf.isode.net clusterf
172.20.3.203 cluster.isode.net cluster
```

### 37.2.2.2 Set up the Shared Storage

Configure the logical volume manager (LVM) on both nodes by editing `/etc/lvm/lvm.conf`. Alter the `system_id_source` so that the following line is included:

```
system_id_source = "uname"
```

Check that lvm is configured correctly by running:

```
lvm systemid
```

The output should match `uname -n`

Configure the volume on the Primary Server. In this example, the shared storage is on `/dev/sdb` with a partition created as `/dev/sdb1`.

Create the logical and physical volumes:

```
# pvcreate /dev/sdb1
```

Create a volume group, the `--setautoactivation n` flag ensures that the volume group will not be automatically started during startup.

```
vgcreate --setautoactivation n isode_VG /dev/sdb1
```

Create a logical volume within the volume group.

```
# lvcreate -n isode_v isode_vg
```

Make the filesystem:

```
# mkfs.xfs /dev/isode_vg/isode_v
```

The failover node needs to be configured to accept /dev/sdb1:

```
lvmdevices --adddev /dev/sdb1
```

Now create a folder on each server called /san . This is where the shared storage will be mounted.

### 37.2.2.3 Configure Firewall

Allow the High Availability service through the firewall on both nodes:

```
# firewall-cmd --permanent --add-service=high-availability
# firewall-cmd --add-service=high-availability
```

### 37.2.2.4 Configure and Start the Cluster Service

Set up the Cluster Administrator User (hacluster). If the userid has already been created, simply set the password for it. The same password should be used on both nodes.

```
# passwd hacluster
```

Start the cluster service and set it to start automatically on system startup

```
# systemctl start pcsd.service
# systemctl enable pcsd.service
```

Add the nodes to the cluster:

```
# pcs cluster auth clusterp.isode.net clusterf.isode.net
Username: hacluster
Password:
clusterp.isode.net: Authorized
clusterf.isode.net: Authorized
```

Start the cluster:

```
# pcs cluster setup --start --name isode_cluster clusterp.isode.net\
clusterf.isode.net
# pcs cluster enable --all
```

---

**Note:** In order to quickly test out clustering it's been suggested to disable fencing and the quorum. This should not be done on a production system.

---

Disable Fencing:

```
# pcs property set stonith-enabled=false
```

Disable Quorum:

```
# pcs property set no-quorum-policy=ignore
```

### 37.2.2.5 Create Shared IP and Storage Resources

Put the failover node into standby:

```
# pcs node standby clusterf.isode.net
```

```
# pcs resource create vip1 IPAddr2 ip=172.20.3.203 \
    cidr_netmask=16 --group isodegroup
# pcs resource create my_lvm ocf:heartbeat:LVM-activate \
    vgname=isode_vg vg_access_mode=system_id --group isodegroup
# pcs resource create my_fs Filesystem \
    device="/dev/isode_vg/isode_v" directory="/san" fstype="xfs" --group
```

These "Resources" will be assigned to the Primary Server while the Failover is down. Test the "Resources" failing over by bringing the failover node out of standby, and putting the primary node into standby.

```
# pcs node unstandby clusterf.isode.net
# pcs node standby clusterp.isode.net
# pcs status
```

The `pcs status` command should show the resources being active on the failover node.

## 37.2.3 Install Isode Server Software

Use the `pcs` command to put the failover node into standby, and make the primary node active.

```
# pcs node unstandby clusterp.isode.net
# pcs node standby clusterf.isode.net
# pcs status
```

Install and configure your Isode packages (e.g M-Switch, M-Vault and optionally M-Store and M-Box) in the normal way, using the "cluster" hostname or IP address when prompted during the configuration process.

Now stop all the Isode services and run the following commands on the Primary Server:

```
# cd /san
# mkdir etc
# mkdir var
# cp -rp /etc/isode /san/etc
# cp -rp /var/isode /san/var
```

Start the Failover Server and install the same Isode packages on it as you installed on the Primary Server. There is no need to perform the configuration steps.

On both the Primary and Failover servers:

- Copy the `/etc/isode/isotailor.sample` file to `/etc/isode/isotailor` and make the following changes:
  - `etcpath: /san/etc/isode/`
  - `logpath: /san/var/isode/log/`
  - `datapath: /san/var/isode/`

Start M-Vault on the Primary Server, and run MConsole to make the following changes to the Messaging Configuration using the Switch Configuration View:

- Set the Archive Folder for your MTA to be `/san/var/isode/archive/%D/`.
- There may be other explicit references to `/etc/isode` within your MTA's configuration - for example, on the Security tab or as an Advanced variable. These should be changed to reference `/san/etc/isode`.
- If you have M-Box installed, find the server configuration with the Internet Message Stores folder, and set the Root Directory value to `/san/var/isode/ms/user`.
- If you have M-Store X.400 installed, find the server configuration with the X.400 Message Stores folder, and set the Mailbox Root value to `/san/var/isode/mailboxes`.

Copy the `/etc/isode/isotailor` across to the failover node, so that they are in sync.

At this point it is worth starting the rest of the Isode Services on the Primary Server and checking that everything works as expected. Then stop the Isode Services and shut down the Failover Server.

```
# pcs node unstandby clusterf.isode.net
# pcs node standby clusterp.isode.net
# pcs status
```

## 37.2.4 Activate isode services using MAS

Start MAS on both nodes in the usual way, once started activate the products being clustered.

An `activate.dat` file will be created in `/etc/isode`. This should be copied to the shared disk.

```
# cp /etc/isode/activate.dat /san/etc/isode/activate.dat
```

Next switch the shared resources over to the failover node.

```
# pcs node unstandby clusterp.isode.net
# pcs node standby clusterf.isode.net
# pcs status
```

Activate isode service using MAS on the failover node. This will create another `/etc/isode/activate.dat`. Concatenate the failover nodes `activate.dat`

```
# cat /etc/isode/activate.dat >> \
/san/etc/isode/activate.dat
```

## 37.2.5 Configure Isode Services as Resources

On both nodes create a symlink pointing from `/var/isode/d3-db` to `/san/var/isode/d3-db/`

```
ln -s /san/var/isode/d3-db/ /var/isode/d3-db
```

On both nodes, prevent the Isode Services from starting on boot (the list of services depends on what you have installed:

```
# systemctl disable isode-dsa@d3-db
# systemctl disable pp
# systemctl disable pumice
```

Set the Isode Services up as Resources (the list of services depends on what you have installed:

```
# pcs resource create M-Vault systemd:isode-dsa@d3-db --group isodegroup
# pcs resource create M-Switch systemd:pp --group isodegroup
# pcs resource create M-Store systemd:pumice --group isodegroup
```

The order of the resources listed above is important as this is the order in which they will be started: for example, M-Vault cannot start until the Shared IP and Filesystem have started, and M-Switch cannot start until M-Vault is running.

Restart the Primary Server and check the status of the cluster:

```
# pcs status
```

You should see a result which indicates that the Primary Server is online and the Failover Server is offline, with the vip1, clusterfs and Isode services running, for example:

```
Cluster name: isode_cluster
Last updated: Tue Aug 22 15:07:28 2017
Last change: Mon Aug 21 11:29:20 2017 via cibadmin on
```



```

clusterp.isode.net
Stack: cocsync
Current DC: clusterp.isode.net (1) - partition WITHOUT quorum
Version: 1.1.10-29.el7-368c726
2 Nodes configured
7 Resources configured

Online: [ clusterp.isode.net ]
OFFLINE: [ clusterf.isode.net ]

Full list of resources:

Resource Group: Isode
vip1          (ocf::heartbeat:IPaddr2):
Started clusterp.isode.net
clusterfs     (ocf::heartbeat:Filesystem):
Started clusterp.isode.net
M-Vault       (lsb:dsa):   Started clusterp.isode.net
M-Switch      (lsb:pp):    Started clusterp.isode.net
M-Store-X00   (lsb:pumice): Started clusterp.isode.net

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: inactive/enabled

```

Now check that the Isode software is running as expected, and then bring the Failover Server online. Running # `pcs status` again should now produce a result like:

```

Cluster name: isode_cluster
Last updated: Tue Aug 22 15:07:28 2017
Last change: Wed Aug 23 10:26:20 2017 via cibadmin on
clusterp.isode.net
Stack: cocsync
Current DC: clusterp.isode.net (1) - partition with quorum
Version: 1.1.10-29.el7-368c726
2 Nodes configured
7 Resources configured

Online: [ clusterp.isode.net clusterf.isode.net ]

Full list of resources:

Resource Group: Isode
vip1          (ocf::heartbeat:IPaddr2):   Started
clusterp.isode.net
clusterfs     (ocf::heartbeat:Filesystem): Started
clusterp.isode.net
M-Vault       (lsb:dsa):   Started clusterp.isode.net
M-Switch      (lsb:pp):    Started clusterp.isode.net
M-Store-X00   (lsb:pumice): Started clusterp.isode.net

PCSD Status:
clusterp.isode.net: Online
clusterf.isode.net: Online

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled

```

Repeat your test of the Isode software, and then shut down the Primary Server. Running `# pcs status` again should now produce a result like:

```
Cluster name: isode_cluster
Last updated: Tue Aug 22 15:07:28 2017
Last change: Wed Aug 23 10:26:20 2017 via cibadmin on
clusterp.isode.net
Stack: cocsync
Current DC: clusterp.isode.net (1) - partition with quorum
Version: 1.1.10-29.el7-368c726
2 Nodes configured
7 Resources configured

Online: [ clusterf.isode.net ]
OFFLINE: [ clusterp.isode.net ]

Full list of resources:

Resource Group: Isode
vip1          (ocf::heartbeat:IPaddr2):    Started
clusterp.isode.net
clusterfs      (ocf::heartbeat:Filesystem): Started
clusterp.isode.net
M-Vault        (lsb:dsa):    Started clusterp.isode.net
M-Switch        (lsb:pp):    Started clusterp.isode.net
M-Store-X00     (lsb:pumice):    Started clusterp.isode.net

PCSD Status:
clusterp.isode.net: Offline
clusterf.isode.net: Online

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

This indicates that the Failover Server has taken over running of the Isode services. Again, you should check that the Isode services are running as expected.

If you want the services to run on the Primary when it becomes available again, then you need to add the following configuration.

```
# pcs constraint location Isode prefers clusterp.isode.net
```

# Chapter 38 Securing Your Messaging System

Authentication is proving the identity of someone or of a process; authorization is what that person or process is allowed to do. This chapter describes the configuration of both of these elements in M-Switch.

This chapter also describes how the secure storage of passwords can be achieved.

## 38.1 Authorization

The authorization mechanisms in M-Switch allow you to permit or block messages in a rule based manner. This takes place in a separate way from routing or other decisions about how messages may be handled.

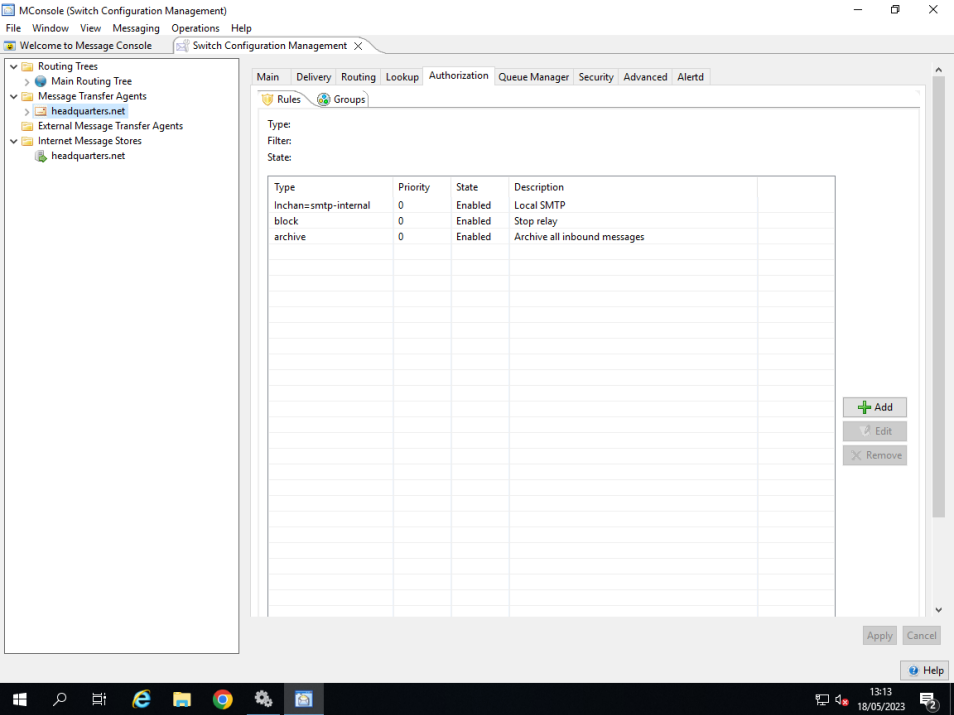
The M-Switch authorization system is a complex and flexible subsystem which is designed to allow administrators to configure many different policies into the MTA. For the most part administrators need only use the authorizations system in a very simple way.

These are

- prevention of SMTP relay
- enable archiving of messages that pass through M-Switch
- use different SMTP internal and SMTP external channels based on table of trusted MTAs

When creating an Internet MTA, each of these three is configured for you in a way that is likely to meet the needs of the majority of Internet configurations. If you need a more complex configuration you should use the *M-Switch Advanced Administration Guide*.

Figure 38.1. Default MTA authorization rules



The three rules configured set up the MTA to carry out the authorization checks generally required so that the MTA operates in an appropriate manner.

### 38.1.1 Message Archiving

There are two ways in which Messages can be configured to be archived as they enter the MTA:

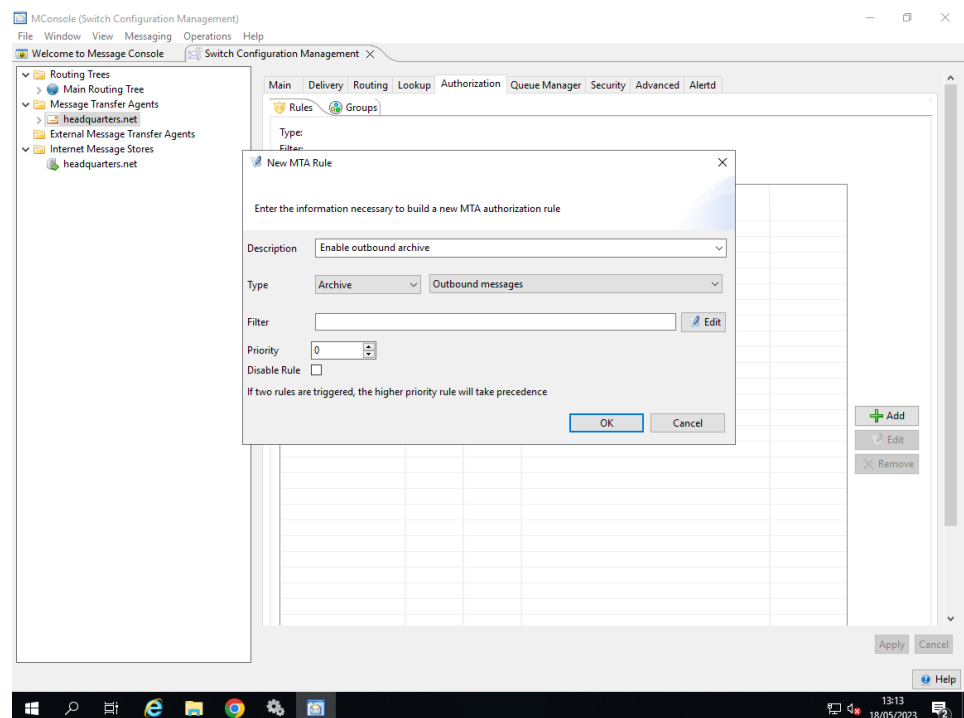
- Archived to disk
- Archived to an address

By default M-Switch is configured to archive inbound messages to disk, as there will be an Archive Rule of type 'Archive' and subtype 'Inbound messages'. However, you can configure M-Switch to archive to an email address by configuring an Internet or O/R Address in the Authorization Rule, using an Archive Rule of Type 'Archive by address'. Inbound Messages will then be archived by sending a copy of the message to that address.

There is a separate rule for enabling outbound archive that can be configured as shown in the figure below. It gets enabled by default when a MIXER or ACP127 messaging system is created. For other system types, you can archive the outbound messages by creating an Archive Rule of Type 'Archive' and subtype 'Outbound messages'

In addition to enabling archives, there is a rule to enable generation of index files for inbound archives. Index files contain list of words found in message content and can be optionally used by the log parser to populate the database with words to enable searching of messages by words. This is an optional feature in the switch as well as auditdb log parser due to the space and processing overhead associated with storing words.

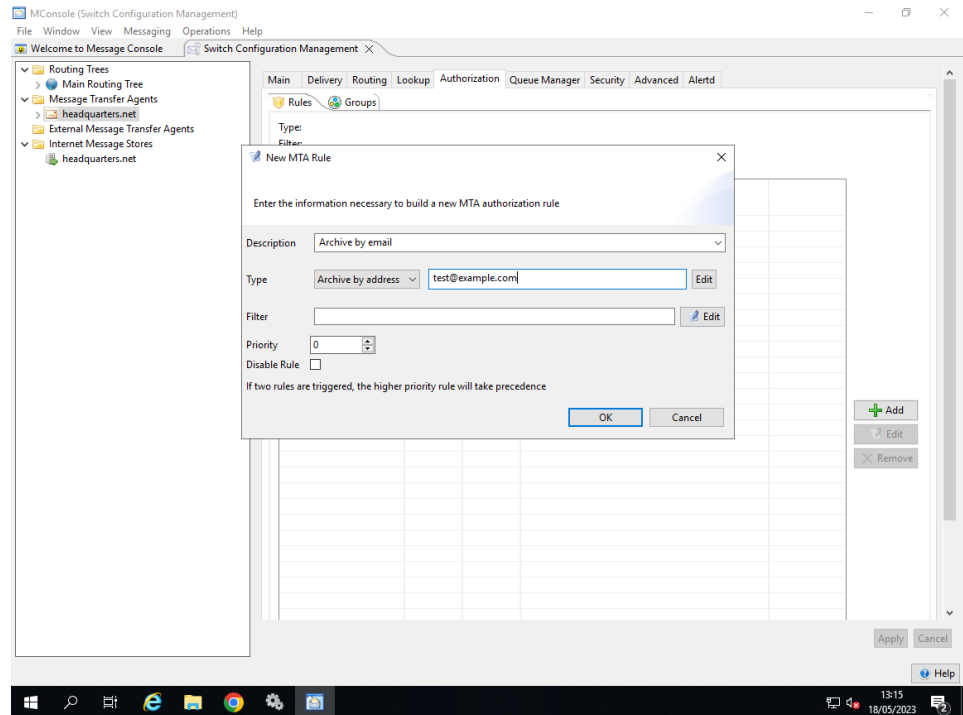
**Figure 38.2. Outbound Archive authorization rule**



---

**Note:** If you wish to configure both Archiving to an address and Archiving to disk, you need to configure two Archive Rules. The archive rule for generating index is separate and requires you to first create rule for archiving inbound messages. Also note that Archive Rules are not affected by priority: i.e. all Archive Rules will be acted on (as long as they match any Filter configured).

---

**Figure 38.3. Configuring Archive by Email**

## 38.2 Secure Storage of Passwords

In a number of different places M-Switch needs to configure credentials for connections to other components of the system. These are stored in some cases as text passwords.

To avoid vulnerabilities inherent in such an approach, it is possible to store these passwords in an encrypted form ensuring protection from a security failure leading to compromise of the filestore and the files containing such passwords. This feature is described in the *M-Switch Advanced Administration Guide*.

The following passwords are able to be stored using this encryption feature:

- Directory passwords in *mtaboot.xml*
- Directory passwords in *mtataylor.tai*
- other passwords in *mtataylor.tai*
- X.509 passphrases used to protect private keys in Digital Identities

If you need to use this feature you should read the *M-Switch Advanced Administration Guide*.

---

## 38.3 Security Issues: Filestore and Users (Windows only)

As the M-Switch Services can be started automatically (i.e. without an operator), there are certain issues regarding security which you need to consider: for example, it is generally not possible to request a passphrase at startup, or to obtain random number data from operator input.

As a result, decisions about the following issues need to be made carefully concerning:

- account selection
- file system access control

The security of M-Switch relies on file system access controls. This means that protection afforded to M-Switch filestore is only as strong as the login security of the system (which is usually password-based), including the ease with which a user can acquire the privileges of another user.

### 38.3.1 Running Services With Least Privilege

The Isode Service Configuration GUI (like the Windows Service Manager) allows you to configure the account under which the service is to run. You can choose to run under the system account or any other specified account.

You must ensure the account has sufficient privileges to be able carry out its functions.

Deploying M-Switch on a default Windows installation will need the user to have the privilege to run as a Service. In typical configurations, no other privileges are required.

If M-Switch is configured to use GSSAPI against Active Directory, you will also need `SE_IMPERSONATE_NAME`.

### 38.3.2 Isode Installation and ACL

Files and directories installed are created with an owner which is either a user or group (i.e. the CREATOR OWNER is: SYSTEM; Administrators; Users)

Files and directories are also installed with ACL (Full Control; Modify; Read & Execute; List Folder Contents; Read; Write; Special Permissions)

They are also set to inherit permissions from the object's parent (in most cases *C:\Isode*)

So by default, mostly objects are created which allow Administrators to do anything, and users to read/execute.

You should consider carefully whether this model is suitable for your deployment.

### 38.3.3 Example Setup

You might wish to create a specific Group (e.g. Isode Sysadmin) and a specific user as a member of that Group to have the necessary privileges to be able to run as the Isode Messaging Services, without having system wide privileges if the Services were to run as the local System user.

This example assumes you wish to create Windows Group (Isode Sysadmins), with a single user (isode), as which Isode Services are to run. The Isode Services must then be set to run as that user. The C:\Isode filestore must then be set with the appropriate ACL. You can configure the new User, or any member of the new Group to have full control of the C:\Isode directory.

---

**Note:** If you change ACLs they are overwritten when the Isode packages are updated.

---

### 38.3.3.1 Creating a Windows Group and User

1. Select **Start** → **Administrative Tools** → **Computer Management**
2. Select **System Tools** → **Local Users and Groups** → **Groups**
3. Right click **Groups** and select **New Group**
4. Fill in the following
  - **Group Name**
  - **Description**
  - **Members**

Click on **Create** and then on **Close**

5. Click **System Tools** → **Local Users and Groups** → **Users**
6. Right click on **Users** and select **New User**
7. Fill in the following
  - **User Name**
  - **Full Name**
  - **Description**
  - **Password**
  - **Confirm Password**

Click on **Create** and then on **Close**

8. Now add the user to the group by right clicking on the newly created user and selecting **Properties** and selecting the **Member Of** tab.
9. Click on **Add** and then click on **Advanced** then click on **Find Now**.
10. Select the **Group** into which the User is to be added.
11. Click on **OK (Select Groups)**, **OK (User Properties)**.

If M-Switch is configured to use GSSAPI against Active Directory, you will also need the user or group to have the privilege SE\_IMPERSONATE\_NAME.

You have now created the User and Group under which the Isode Services can be run, either of which can be used to assign permissions to access the C:\Isode directory and the files and directories therein.

### 38.3.3.2 Configure the Isode Windows Services

The **Isode Windows Services** can be configured using the Isode Service Configuration GUI, or the Windows Service GUI. Using the Windows Service GUI is slightly simpler in this case as this will automatically cause the user to be given the necessary privilege to run as a service.

Select **Start** → **Administrative Tools** → **Services**

For each of the Isode Services you want to run as the newly created user, do the following:

1. Double click the service
2. Select the **Log On** tab
3. Change the user under which the Service is to run from the Local System account to be the newly created user created in [Section 38.3.3.1, “Creating a Windows Group and User”](#).
  - use **Browse** to select the user

---

**Note:** The value will appear as `.\<user>`.

---

- Configure the password to be the user password assigned when creating the user

You will be warned that the user has been given the privilege to run as a service.

### 38.3.3.3 Configure Filestore Access Control

The newly created user also needs to be given privileges to access the filestore used by the Isode Services, ie. `C:\Isode`.

1. Right click on `C:\Isode` and select **Properties**.
2. Select the **Security** tab and then click on **Edit**.
3. Click on **Add**.
4. Click on **Advanced**.
5. Click on **Find Now**.
6. Select the Windows Group or User you have configured the Services to run as, and double click on the entry.
7. Click on **OK (Select User)**.
8. Click on **OK (Service Properties)**.
9. Ensure the User or Group has full control and click on **OK** and click on **OK**.

You should now be able to start the Messaging Services so they run as the new user.



# Chapter 39 Content Conversion and Scanning on Submission

This chapter provides a short overview of the way in which the contents of messages can be converted or scanned on submission, and describes the Shaper Configuration File Editor component of MConsole which allows the conversions or scanning to be configured.

---

**Note:** The default setup will cover most deployment requirements and so this is covered in detail in the *M-Switch Advanced Administration Guide*.

---

---

## 39.1 Overview

M-Switch can perform various kinds of conversion on the content of messages. The main types are:

- Conversion to content type for a different messaging system, normally from Internet to X.400, or X.400 to Internet.
- Modification of Internet messaging headings; for example, changes to addresses from internal to external or the removal of trace fields.
- Downgrading of P22 to P2 content for transfer to X.400 (1984) systems.

A message, when submitted to the M-Switch system will have associated with it:

- a content type
- optionally, a content subtype (this is derived from the "subtype in" setting for the inbound channel)
- a header type (if Internet or X.400 P2 or P22)
- a set of encoded information types (if X.400), which are related to the different kinds of X.400 body parts within the content.

Conversion of the content is required for a recipient of the message if at least one of the following is true:

- the message's content type is not listed among the content types of the outbound channel for the recipient.
- the message has a content subtype which does not match the outbound content subtype of the outbound channel for the recipient.
- the message's header type is not listed in the `hdrout` for the outbound channel. If the list is empty, then all header types are accepted.
- the message has encoded information types which are not listed in the `bptout` list for the outbound channel. If the list is empty, then all encoded information types are accepted.
- There is a Directory entry for the recipient, and that entry has deliverable-content-types or deliverable-eits values which do not match those in the message.

When content conversion is required, this is performed by a Shaper channel. The Shaper channel chosen is identified by matching the content type of the message against one of the content types in the content-in list for the channel. If no matching Shaper channel can be found, the message is non-delivered.

The conversions which apply are determined by the data items above which were used to decide that conversion was required, so that the result will be accepted by the outbound channel and recipient. The possible actions are configured using an XML file associated with the channel. The default name for the file is derived from the channel name, but it can be overridden. The format of the XML file is described in the *M-Switch Advanced Administration Guide*. The default configuration should fulfill the requirements for most systems.

---

## 39.2 Scanning on Submission

By default, the content of a message is scanned at the point of submission. For X.400 messages (i.e. P2, P22 and P772 content types), the header is extracted so that fields from it (e.g. the Subject) can be logged. For Internet messages, DSN and MDN fields are extracted for logging (information such as the Subject line from an Internet message is obtained via other mechanisms). Security label and signature information can also be obtained in the same way, and this can then be used as input to the MTA's Authorization Rules.

Scanning on submission uses much of the same functionality as the content conversion subsystem, and is configured via a single XML file named `submitscanconfig.xml`. This configuration file shares many components with those used by the Shaper channels, and MConsole's Shaper Configuration File Editor allows both types of file to be edited.

---

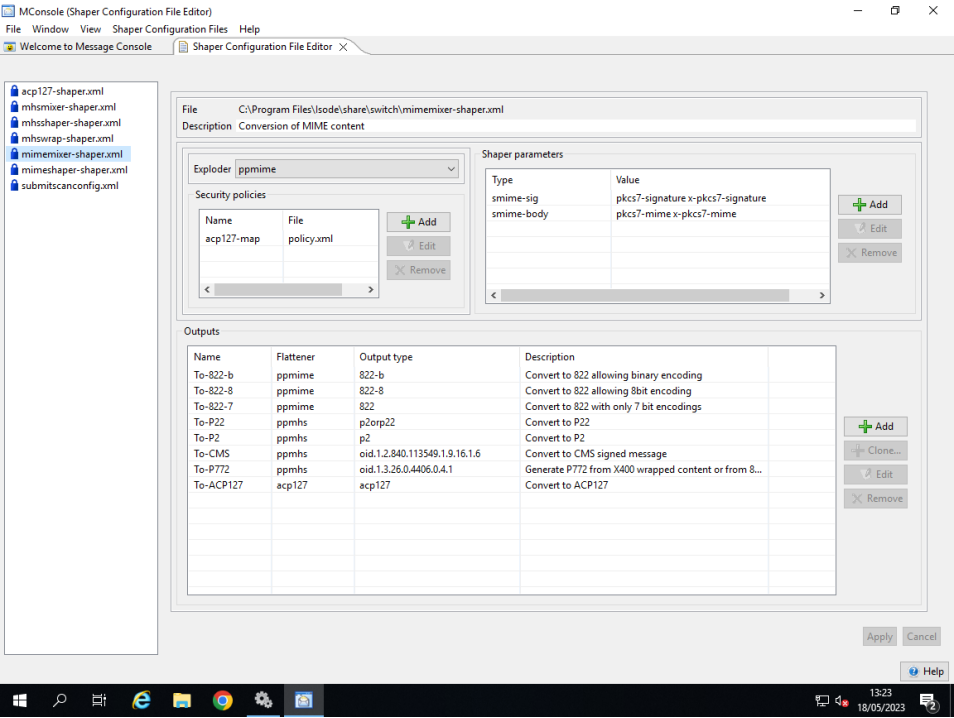
## 39.3 Shaper Configuration File Editor

### 39.3.1 Overview

The Shaper Configuration File Editor facility within MConsole allows structured editing of the XML configuration files which are used by both the content conversion subsystem and the scan-on-submission facility.

A detailed description of the format of Shaper Configuration Files can be found in the *M-Switch Advanced Administration Guide*.

Figure 39.1. Shaper Configuration File Editor



Default versions of these configuration files are installed in the `switch` subdirectory beneath (`SHAREDIR`). When editing one of these files manually, the recommended procedure is to copy the file into the `switch` subdirectory beneath (`ETCDIR`) and edit the copy. The Shaper channel processes first look for their configuration files under (`ETCDIR`) and then under (`SHAREDIR`), and so will find the modified version first.

The Shaper Configuration File Editor follows this recommendation: the default version of each file, located below (`SHAREDIR`) is listed with a closed padlock icon next to it. If this is modified using the Editor, the updated version will be saved back under (`ETCDIR`), and will then be accompanied by an open padlock icon. If you do not have write access to (`ETCDIR`), modified versions of configuration files will be saved to a temporary directory, for subsequent manual copying to the correct location.

**Note:** Because the editor makes use of direct file access, it needs to either be run on the same system as the MTA installation or with access to a copy of the Shaper configuration files, with modified files being copied manually back to the MTA system.

### 39.3.2 Shaper Configuration File Components

A Shaper configuration file consist of a number of components:

- **Exploder.** This selects the way in which messages are disassembled into their components. The chosen exploder must be compatible with the Content Type that the Shaper channel using this configuration file is configured to accept: the `ppmime` exploder handles MIME (i.e. 822 content type), the `ppmhs` exploder handles P2, P22 and P772 content types and the `acp127` exploder handles the ACP127 content type.
- **Security Policies.** This section configures a list of Security Policy Information Files (SPIFs) which will be used by Output components of the Shaper. An (arbitrary) name is associated with each SPIF, and this is then referenced in the Output configuration.
- **Shaper Parameters.** There are some parameters which are applicable to the Shaper as a whole. The available parameters vary depending on the selected Exploder.

- **Outputs.** This section configures the set of Content Types which the Shaper can generate. Each Output has the following components:
  - **Name and Description.** These are arbitrary and do not affect the operation of the Output.
  - **Output Type.** This defines the Content Type which this Output generates, and is used by the Shaper channel to decide which Output to run for a given message recipient.
  - **Flattener Type.** This configures the way in which a message is reassembled from converted components.
  - **Converters.** The `converters` do the work of converting individual message components (headers, bodyparts, complete content etc). Converters have the following components:
    - **Converter Type.** This defines the message component on which this converter acts
    - **Converter Action.** Configures the transformation which the converter performs.
    - **Cost.** Where several converters could be used, the one with the lowest cost will be preferred.
    - **Output type.** An output content subtype may be specified.
    - **EITs.** For X.400 content types, this defines the EITs which will be added to the message when X.400 bodyparts are generated.
    - **Matches.** This configures the matching rules which control whether a specific Converter is selected for use. An individual `match` specifies a message attribute (e.g. Content-type), a value to test the attribute against, and optionally a child bodypart number.
    - **Filters.** A filter may run a set of `filters` to perform transformations on message data. Each `filter` can take a list of key/value parameters.
    - **Parameters.** The `converter` itself may accept parameters which control its behaviour

### 39.3.3 Scan-on-Submission Configuration File Components

A single configuration file, with the name `submitscanconfig.xml` is used to configure message scanning on submission. This has the following component:

- **Security Policies.** This section configures a list of Security Policy Information Files (SPIFs) which will be used by `Scanners`. An (arbitrary) name is associated with each SPIF, and this is then referenced in the `Scanner` configuration.
- **Content Scanners.** This section configures the ways in which different message content types are scanned. Each Content Scanner has the following components:
  - **Content type scanned.** Specifies the content type of messages to which this Content Scanner will be applied.
  - **Description.** This is arbitrary and does not affect the operation of the Content Scanner.
  - **Is an Indexer.** Specifies whether the Content Scanner generates index entries.
  - **Process like.** This allows (for example) P2 messages to be handled by the P22 Content Scanner.
  - **Exploder.** This configures how the message is exploded into its component parts. The chosen exploder must match the content type scanned.
  - **Parameters.** The Content Scanner may accept parameters which control its behaviour.
  - **Component Scanners.** The `Component Scanners` do the work of scanning individual message components (headers, bodyparts, complete content etc). Component Scanners have the following components:
    - **Scanner Type.** This defines the message component on which this scanner acts

- **Cost.** Where several scanners could be used, the one with the lowest cost will be preferred.
- **Filters.** A scanner may run a set of `filters` to extract different information from message data. Each `filter` can take a list of key/value parameters.
- **Matches.** This configures the matching rules which control whether a specific scanner is selected for use. An individual `match` specifies a message attribute (e.g. Content-type), a value to test the attribute against, and optionally a child bodypart number.

# Chapter 40 Content Checking

This chapter describes content checking, which allows an administrator to block, filter and alter messages based on their content.

You need to know what is in a message so you can decide whether you want to block it or not. You may want to block it because it is unwanted (spam), offensive or potentially carries a virus or other malware.

The action taken when a message is set to be blocked can be to annotate the message, quarantine the message, discard the message, non-deliver the message.

---

## 40.1 Checking message content

### 40.1.1 How checking works

When a message is received, the sequence of events is as follows:

1. The Authorization Rules are checked in order to see if they are configured to mandate checking for the channel. See [Section 38.1, “Authorization”](#) for a general description of the M-Switch Authorization system feature, and the *M-Switch Advanced Administration Guide*, Rules Filter section, for a detailed description of Rule Filters.
2. The message is passed to a special checking channel if it has been determined that checking is required.

Different content types (i.e. RFC822 / MIME messages and X.400 messages) require different checking channels. Normally the channel that is used is determined by searching the available channels for a checking channel with a matching content type in its `content-in` list. Alternatively, a list of checking channels can be given in the `auth.channel` table. Only these channels are then searched for one with a matching `content-in` value, and the normal channel search is avoided.

3. There are two different checking channel programs. Described here is the main program which is used. The alternative uses the Content Checking and Conversion Protocol (CCCP) to communicate with a server process which performs the required operation, returning status values back to the channel. The CCCP channel is described in the *M-Switch Advanced Administration Guide*. Only one checking channel will be selected for one message.
4. The checking channel extracts the bodyparts of the message and presents them to a routine which can perform the checking on the *raw* data. Bodyparts within the message content are often held in some encoded form and thus require decoding before they can be checked.
5. The checking routine uses a set of rules and data which you can configure via a graphical interface (described in [Section 40.1.2, “Checking configuration GUI”](#)) to perform both virus and spam checking. When a problem message is discovered, the configurable rules govern the action taken.

For a message containing a virus this may involve:

- Replacing the infected body part with a text body part informing the recipient of the removal.
- Repairing the message. This is supported by some virus checking software for some body part types.

- Adding a text report to the beginning of the message if either replacement or repair has been performed.
- Generating non-delivery reports. Non-delivery reports do not include the returned content.
- Moving the message into an area ('the quarantine'), where it can be held for later examination.

There are a number of different techniques which the Isode Message Switch can use to detect and reject unsolicited junk emails:

- Explicit word/phrase blocking. This relies on detecting specific preconfigured words or phrases in the message being checked. A match will trigger the `BLOCK` action rule.
- Address blacklisting. A blacklisted email address, mail domain or URL will trigger the `BLACKLISTED_ADDRESS`, `BLACKLISTED_HOST` or `BLACKLISTED_URL` action rules, respectively. A corresponding whitelisting facility is also available - matching one of these triggers the `WHITELISTED_ADDRESS`, `WHITELISTED_HOST` or `WHITELISTED_URL` action rules.
- Telephone number blacklisting. As the presence of a telephone number in a message may be a spam indicator, these may be blacklisted (and whitelisted) in the same way as addresses. A match triggers the `BLACKLISTED_TEL` or `WHITELISTED_TEL` action rules.
- Character set blacklisting. Use of unusual character sets may be an indication of spam: this facility enables specific values of the MIME charset attribute to be blacklisted or whitelisted. A match triggers the `BLACKLISTED_CHARSET` or `WHITELISTED_CHARSET` action rules.
- MIME type and file extension checking. This allows specific MIME bodypart types or file extensions to be checked, marking them as unchecked, bad or blocked. In the latter three cases the `NOTCHECKED`, `BADATTACHMENT` or `BLOCK` action rules will be invoked respectively.
- Scoring rules. This is the heart of the anti-spam configuration. A complex set of rules assign positive or negative scores to particular words, phrases, regular expressions and combinations of these. The default score assigned to each rule has been calculated offline by analysing a large number of messages which have been sorted into 'spam' and 'non-spam' categories.

#### 40.1.1.1 What you need to configure

To set up a checking policy for a Message Switch, you need to:

- Configure the checking channel(s), using the graphical interface described in [Section 40.1.2, "Checking configuration GUT"](#). This interface allows configuration of both the virus and anti-spam checking facilities.
- Enable checking by configuring the authorization. See [Section 38.1, "Authorization"](#) and the *M-Switch Advanced Administration Guide*.
- Decide which content types are to be checked. If there are certain content types to be excluded from checking, these need to be set in the internal variable `nocheckcontent` (see [Section 40.1.1.2, "Tailoring channels for checking"](#)).
- Configure checking channels to handle the required content types and the types of body parts they may contain (see [Section 40.1.1.2, "Tailoring channels for checking"](#)).
- You may want to set up message template files to format the warning messages to the sender/recipient of problem messages. Template files and template file macros are described in [Section 40.3, "Preventing address harvest attacks"](#).
- You may need to alter the XML configuration file which controls how the channel checks the different components of the content. This configuration file is described in the *M-Switch Advanced Administration Guide*.

### 40.1.1.2 Tailoring channels for checking

As different checking channels are needed to check different content types, you need to create one or more new checking channels to check a given content type. This section describes how to configure these checking channels.

If you want to check some content types but not others, you set up checking channels for the content types you want checked, and set the internal MTA tailoring variable, `nocheckcontent`, to the content types you do not want checked. In MConsole, you can set this variable in the MTA properties editor, on the **Advanced** tab. For example, to disable checking of P35 and P42 content types, press the **Edit** button to the right of the **PP Internal Variables**. On the new dialog, click on **Add**, and then enter `nocheckcontent` as the **Variable Name** and `p35,p42` as the **Optional Value**. Finally, click on the **OK** button, then on the **OK** button.

If you are not using MConsole to configure the MTA, you would add the following line to the `mtatailor` file:

```
set nocheckcontent="p35,p42"
```

The order of the channels in the MTA tailoring is significant: if you have more than one checking channel for a given content type, the channel you wish to be used by default for a given content type should be positioned above the others by using the **Set Channel Order editor**, which is available when you right click on the MTA entry in MConsole. If you are not using MConsole, then the channel order is the one found in the `mtatailor` file.

If checking is required and no suitable checking channel can be found, the message is non-delivered.

To create a new checking channel using MConsole:

1. Start the channel creation wizard by selecting **New Channel** from the **Channels** node in the **MTA Administrator** window.
2. For the **Channel Type** select either **Checker (Internet)** for an RFC 822/MIME checking channel or **Checker (X.400)** or for an X.400 (P2 and P22) checking channel.
3. The channel **name** should be `mimecheck` for an RFC 822/MIME checking channel and `mhscheck` for an X.400 (P2 and P22) checking channel. If these names are not used, then the channel should have its config file value set (in the **Program** tab for the channel) to `mimecheck-checker.xml` for RFC 822/MIME content, and `mhscheck-checker.xml` for X.400 content. Select **check** for the channel **type**.

---

**Note:** For X.400 checking to be restricted to FTBP body parts only, `mhscheck-checker-ftbp-only.xml` is provided and should be used instead of the default `mhscheck-checker.xml` file.

---

4. The value in the **prog** field should be `checkchan`
5. The content type that a channel is to check is configured in the **content-in** field. An X.400 checking channel would normally have the value `p2,p22`. For an RFC 822/MIME checking channel, it would be `822,822-8,822-b`.

The checker channel uses a library of Tcl functions to perform checking, which reads from the configuration set up by the checking configuration GUI (described in [Section 40.1.2, "Checking configuration GUI"](#)).

If you are not using MConsole for configuration, you will need to manually edit a checking channel entry into your `mtatailor` file, along with the channel's table entries. An example can be found in an appendix of the *M-Switch Advanced Administration Guide*.



6. The channel will require inbound and outbound tables. These should be configured as `flags=dbm`. Corresponding table files should be created in the table directory (usually `(ETCDIR)/switch`). You need to build these table files into the Message Switch's configuration database by running `(SBINDIR)/dbmbuild`.

---

**Note:** Not all X.400 body parts can be checked. Only those whose data can be converted to a single sequence of bytes can be checked. In particular, File Transfer Body Parts (FTBP) need to have a data part of a single octet string (conforming to the MAWG profile).

---

After you have created the channel, you can use the editor to change some of the values that weren't available in the wizard. In particular, you can choose to configure the channel to use per-user configuration read from the directory. This is done by setting the variable **ldap\_userconfig** to **yes** in the channel's **Program** tab. The configuration setup is related to the laser routing configuration, with extra variables.

### 40.1.1.3 Configuring individual checking overrides

The per-channel checking configuration described in [Section 40.1.1.2, "Tailoring channels for checking"](#) allows messages using different inbound/outbound channel combinations to be checked with different checker channels (and thus different checking configurations). There are two further ways in which checking can be varied between individual email addresses: use of multiple checking channels, with different configuration for each channel, and the use of variable overrides to alter the behaviour of a single checking channel for different recipients (or a combination of the two methods).

#### 40.1.1.3.1 Using multiple checking channels

To configure the Message Switch so that addresses which share the same inbound/outbound channel pair are checked differently:

- Add the desired address to the inbound table for one of the checker channels (all that is actually needed is the address as a key; no value is required, but to stop annoying error messages being logged, we suggest you use `"dummy=dummy"` as the value).
- Configure the inbound table as the checker channel's mtatable.

An example with two checker channels is shown below. The first checker channel defined in the `mtatailor` file is used as the default. The configuration causes all email addressed to `John.Smith@headquarters.net` to be checked with the second checker channel (`mimecheck2`). In the example below, the address has been tailored via a table override.

```
tbl mimecheckin show="Mimecheck: inbound table" flags=empty
#
tbl mimecheckout show="Mimecheck: outbound table" flags=empty
#
tbl mimecheck2in show="Mimecheck2: inbound table" flags=empty
override="John.Smith@headquarters.net: dummy=dummy"
#
tbl mimecheck2out show="Mimecheck2: outbound table" flags=empty
#
chan mimecheck type=check name=mimecheck prog=mimecheck
show="Check MIME content" content-in="822"
hdrin="822,822-norm" intable=mimecheckin
outtable=mimecheckout
#
chan mimecheck2 type=check name=mimecheck2 prog=mimecheck
show="Check MIME content (test channel)" content-in="822"
hdrin="822,822-norm" intable=mimecheck2in
outtable=mimecheck2out mtatable=mimecheck2in
```

This default value can be overridden for an individual address by configuring an mtatable entry for the address as shown below.

```
tbl mimecheckin show="Mimecheck: inbound table" flags=empty
override="John.Smith@headquarters.net: variable=score1_level=15.0"
#
tbl mimecheckout show="Mimecheck: outbound table" flags=empty
#
chan mimecheck type=check name=mimecheck prog=mimecheck
show="Check MIME content" content-in="822"
hdrin="822,822-norm" intable=mimecheckin
outtable=mimecheckout mtatable=mimecheckin
```

## 40.1.2 Checking configuration GUI

You need to contact Isode support to obtain the latest standard rules files.

These need to be unzipped into (*ETCDIR*)/switch/msgcheck.

The graphical interface which allows the checking configuration to be created and modified is provided by the TclTk GUI tool scanconfig.

This needs to be run though the wish interpreter which is not installed as part of the Isode release.

If you want to do this, please contact Isode support for instructions.

On Unix (bash shell) you should run it as follows:

```
export WISH=/opt/isode/bin/wish
/opt/isode/sbin/scanconfig
```

On Windows, start the **Message Scanning** program from the **Start** menu.

If you have more than one channel of type=check configured, you will be presented with a list of channels from which to choose:

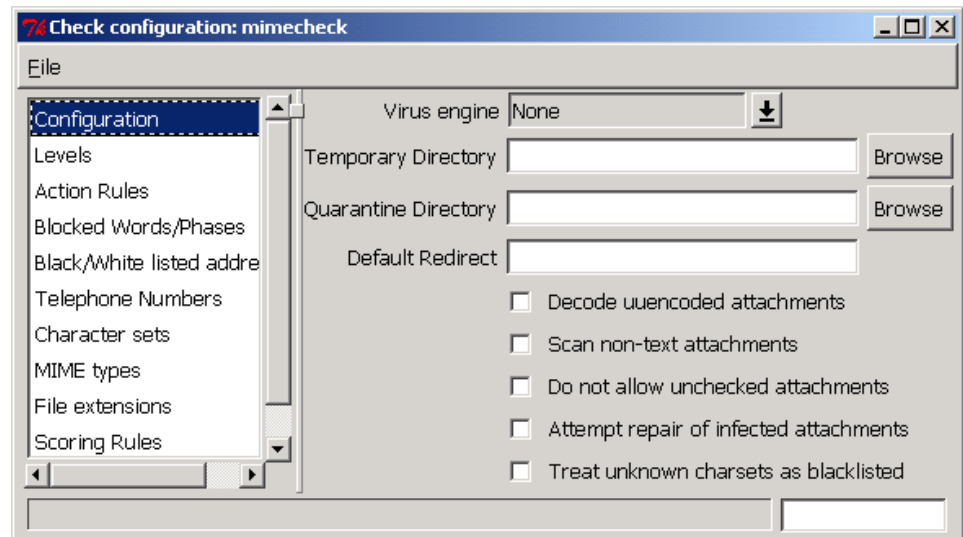
**Figure 40.1. Message Scanning - Open Channel**

If you have only one checker channel configured, you will simply be asked to confirm that you wish to open the checker channel. The main checker configuration window is then displayed:

The **Configuration** page allows the parameters of the virus checking function to be configured. You can choose between the available virus checking implementations or configure no virus checking (this is the default). An interface to the Sophos anti-virus engines is supplied as standard. Interfaces to other anti-virus systems can be scripted if required, as described in [Section 40.1.4.1, “Adding an anti-virus interface”](#).

A temporary directory into which messages will be placed for checking can be specified (if none is specified, a temporary subdirectory of the message in the queue will be created and used). A quarantine directory, into which messages can be moved using the ‘quarantine’ action, can be specified: this defaults to `/var/isode/quarantine` on UNIX and `C:/Isode/quarantine` on Windows.

A default address to which rejected messages will be redirected can be specified if required. Checkboxes allow other features of the checker channel to be enabled:

**Figure 40.2. Message Scanning - Configuration**

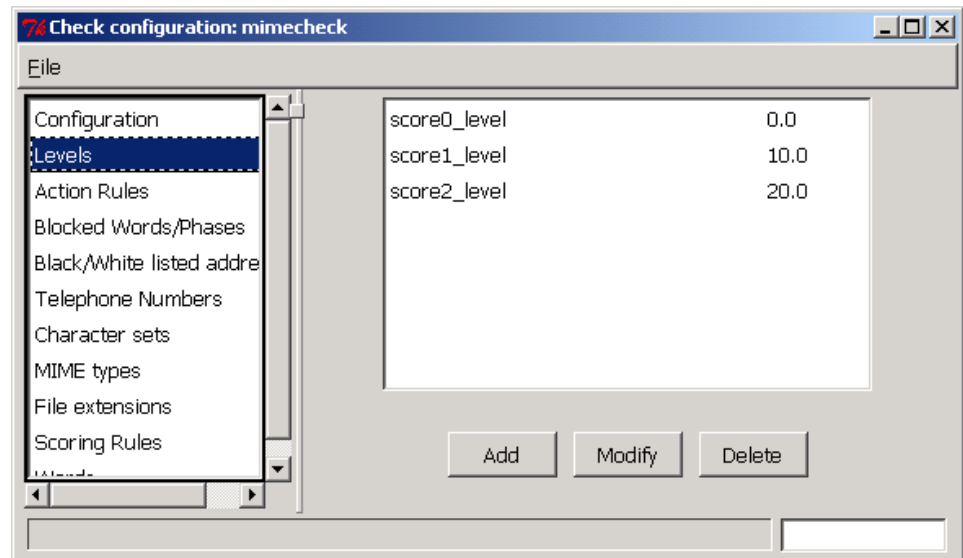
- Whether uuencoded bodyparts should be decoded or not.
- Whether non-text bodyparts should be scanned by the anti-spam checking (all bodyparts are always scanned for viruses). This may be useful when attempting to detect undesirable content in Word documents, for example.
- Whether unchecked bodyparts to a message should trigger the NOTCHECKED rule.
- Whether the virus checking code should attempt to repair any infected bodyparts it finds.
- Whether unknown character sets should trigger the BLACKLISTED\_CHARSET rule.

The **Priorities** page allows you to add per-user configuration. Each action rule has a priority, the rule which is triggered with the highest priority will be applied. However, per-user configuration can give a cut-off in the priority, so that if the priority of a rule is lower than the cut-off, it will not be applied even if triggered. There are four user levels:

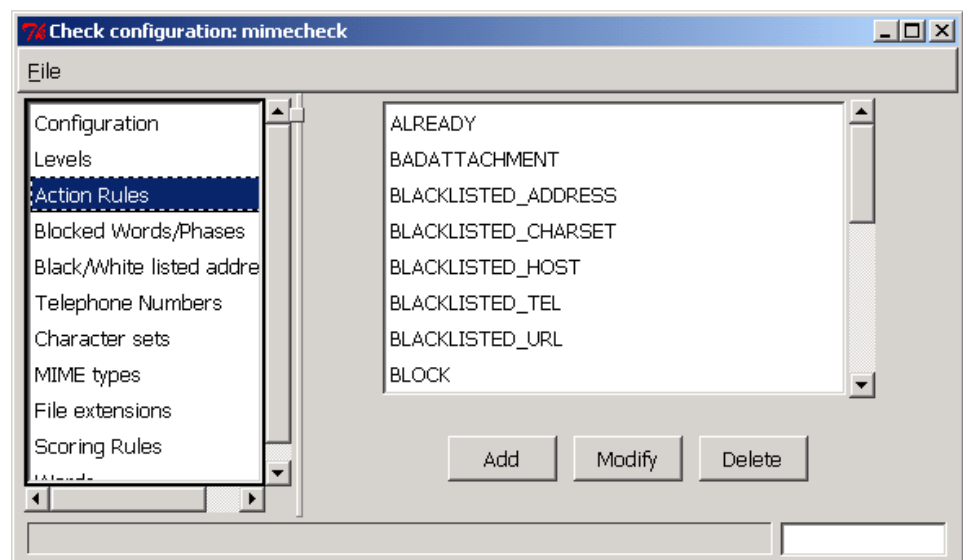
- aggressive
- standard
- mild
- none

The priority cut-off for aggressive is always zero. The values for the other levels are configured in the **Priorities** page. It is possible that the administrator may also need to adjust the priorities of the action rules, so that only rules which should be applied when the user's level is aggressive have a priority less than the standard cut-off, and so on for the other levels. Only anti-virus rules, for instance, would have priorities higher than the 'none' level cut-off.

The **Levels** box allows you to configure the score levels which trigger various action rules. By default there are three scoring levels and associated action rules defined. More levels can be defined if required.

**Figure 40.3. Message Scanning - Levels**

The **Action Rules** box allows the configuration of sets of actions which will be performed on a message when it meets certain criteria.

**Figure 40.4. Message Scanning - Action Rules**

Selecting a rule and pressing the **Modify** button (or attempting to **Add** a new rule) displays the **Modify Action Rule** page:

**Figure 40.5. Message Scanning - Modify Action Rule**

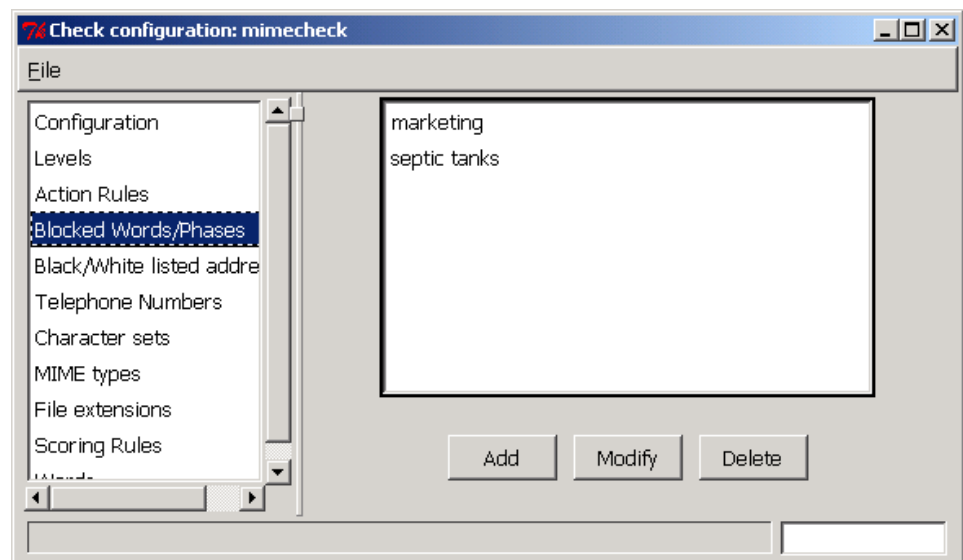
This page allows an individual action rule to be defined. Various parameters can be configured:

- **Action:** The basic function of the rule, which are:
  - **ok** – allow the message to pass (with possible modification)
  - **redirect** – redirect the message
  - **reject**– reject the message (nondelivery)
  - **discard** – silently discard the message
  - **quarantine** – move the message into the quarantine directory.
- **Priority:** this is used to decide which rule to process when multiple rules are activated – the rule with the highest priority is chosen.
- **Do not return content:** whether the original content of the message should be included when a nondelivery report or other warning message is generated.
- **Status message:** a status message which will be logged when the rule is triggered.
- **Warn recipient:** the text of a warning message which should be sent to the original recipient if the message has been rejected. The warning text can either be directly entered, or can be < followed by the name of a file containing the text (assumed to be located in the (*ETCDIR*)/*switch/msgcheck* directory). Message templates are described more fully in [Section 40.1.3, “Template files”](#).

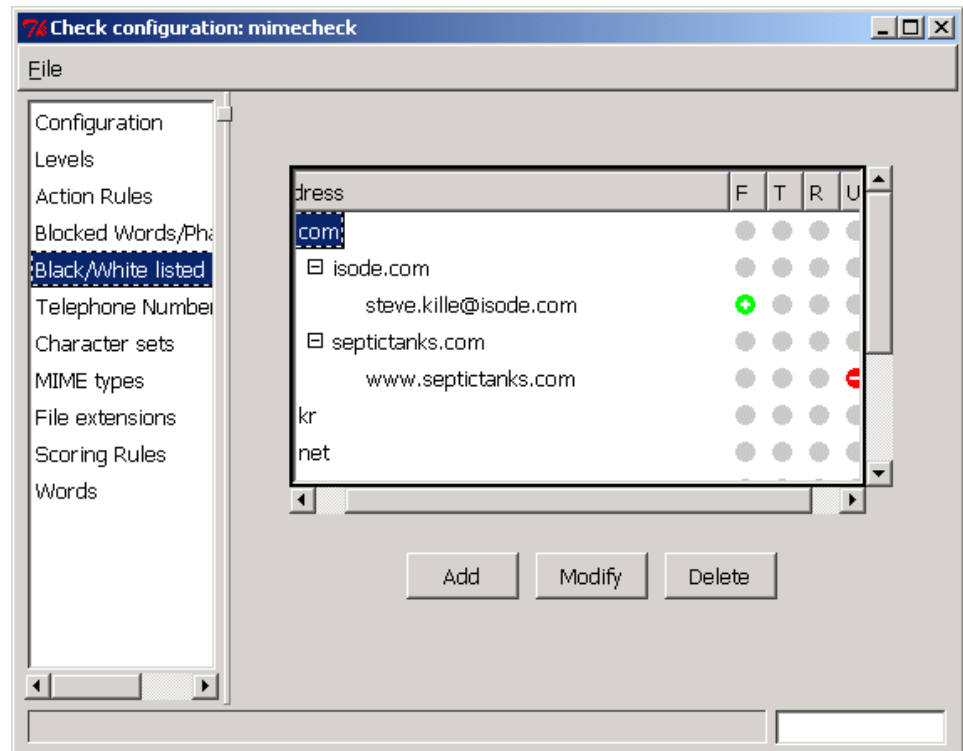
- **Warn sender:** The text of a warning message which should be sent to the originator of the message being processed, if the message has been rejected. Direct text or a file can be specified as described above.
- **Subject:** the subject line to be used for the message.
- **Header:** the header to use for the message.
- **Insert text:** text which should be inserted if the message is being delivered to the intended recipient (i.e. the action is **ok**). The intended recipient of the original message will actually receive a new message with this text as the first bodypart and the original message included as a message bodypart.
- **Replacement:** text which should be used to replace a bodypart which has been found to be infected with a virus. The same ‘blank line’ rule applies as above.
- **Rule code:** the instructions which actually make up the rule. These are expressed in a reverse Polish notation stack-based language. The language is described more fully in [Section 40.5.1, “Action rule programming language”](#).

The **Blocked Words/Phrases** page allows configuration of specific words and phrases which will cause the ‘block’ action rule to be invoked:

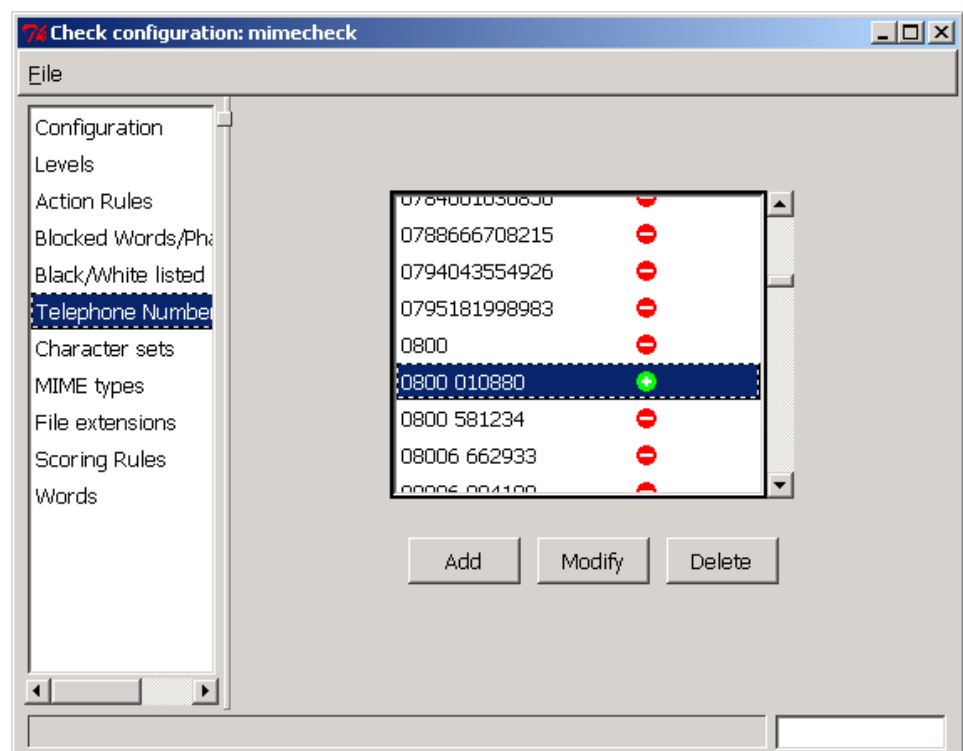
**Figure 40.6. Message Scanning - Blocked Words/Phrases**



The **Black/White listed addresses** form allows mail domains, URLs and individual email addresses to be blocked (i.e. trigger the ‘block’ action rule) or explicitly unblocked. In the example shown below, any message which contains the URL ‘www.septictanks.com’ will be blocked (the ‘Stop’ sign in the **U** column indicates this), as will any message whose From: address contains ‘isode.com’ (indicated by a ‘Stop’ sign in the **F** column), *except* ‘steve.kille@isode.com’ (indicated by a + sign in the **F** column). Blocking on addresses in the To: and Received from: fields (the **T** and **R** columns respectively) can also be performed using this mechanism.

**Figure 40.7. Message Scanning - Black/White listed addresses**

Telephone numbers in messages can also cause messages to be blocked. The black/white listing works in the same manner as for addresses: for example, you could configure all messages which contain telephone numbers including the digits 0800 to trigger the 'block' rule, *except* those which contain the number 0800010880 as illustrated below.

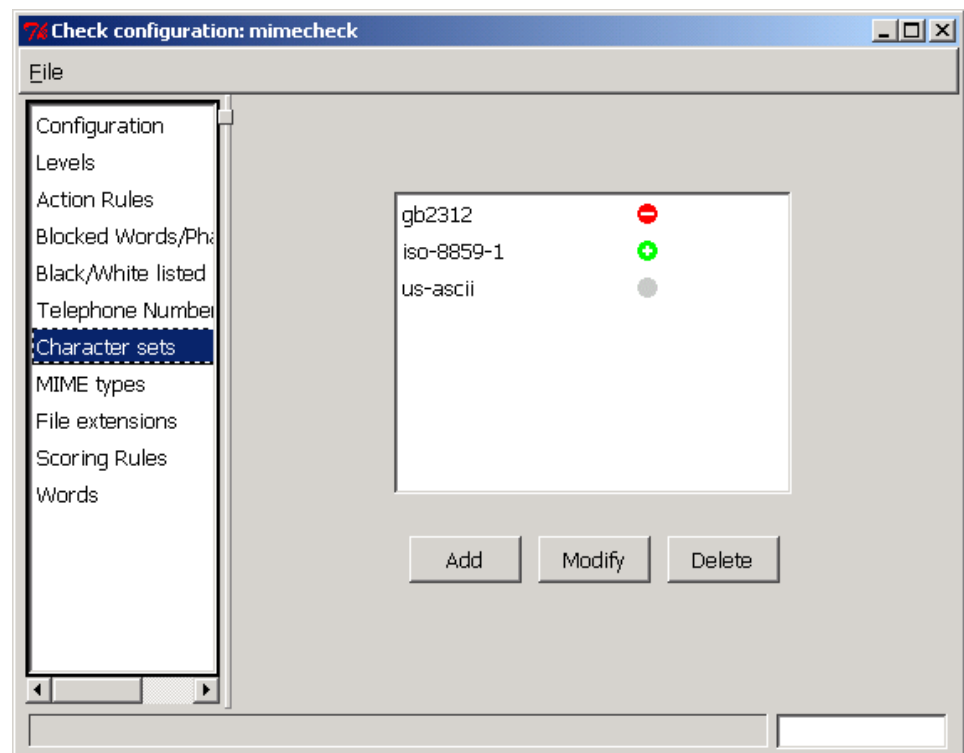
**Figure 40.8. Message Scanning - Telephone Numbers**

Similar configuration is available for character sets : a particular character set (expressed as a MIME charset parameter value – e.g. 'us-ascii' or 'iso-8859-1') can be marked as 'not listed', 'known' or 'blacklisted'. In the example shown below, the 'gb2312' character set



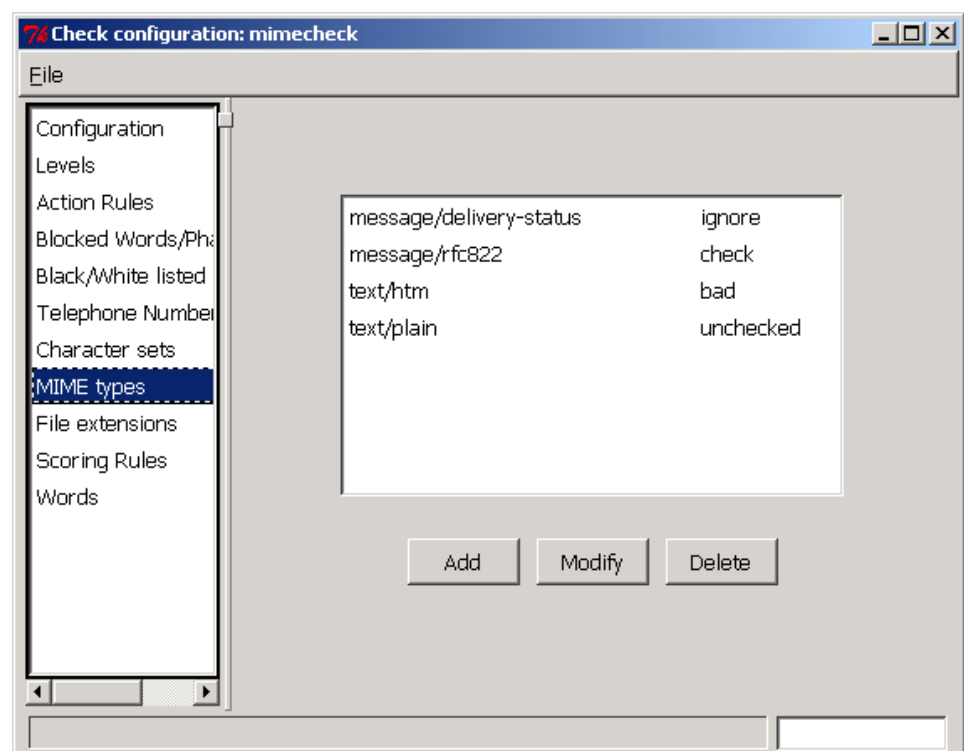
is marked as blacklisted – this is a Chinese language character set which is often seen in unsolicited email originating from the Far East.

**Figure 40.9. Message Scanning - Character sets**



Specific MIME type parameters and file extensions in 'filename' parameters can be marked for normal checking, marked not to be checked, immediately marked as bad or ignored. The configuration screen for MIME types is shown, but the screen for file extensions is identical:

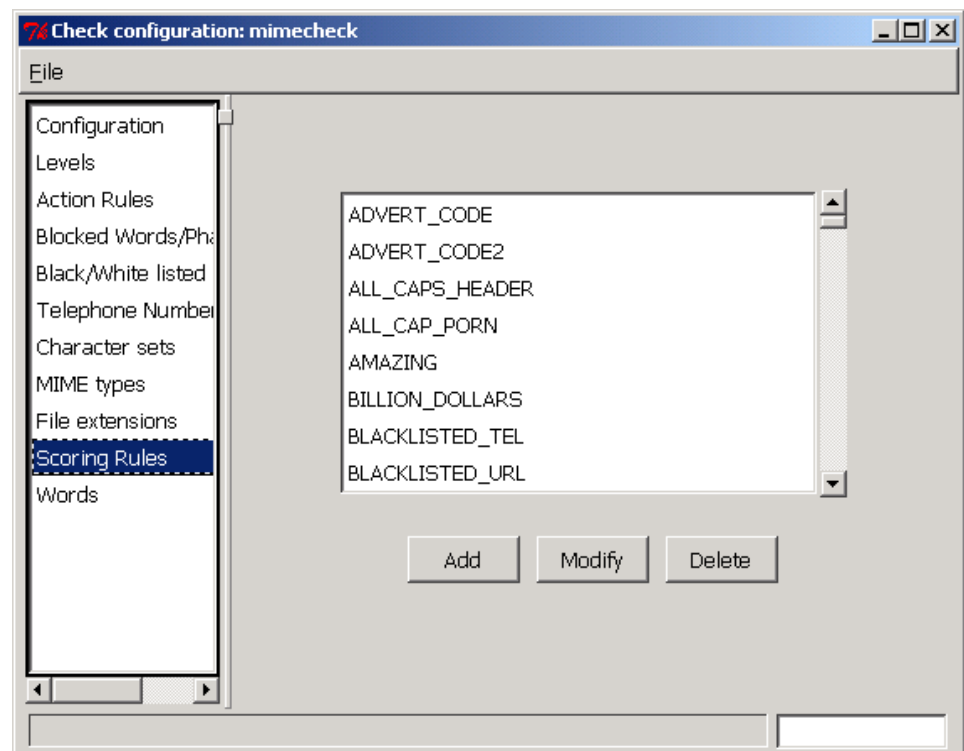
**Figure 40.10. Message Scanning - MIME types**



The ‘unchecked’ case shown above means that the bodypart will not be checked, but if ‘no unchecked’ is set, then that triggers a rule. This is why it is different from ‘ignore’. It might be used for bodypart types which it is known there is no point in looking at, such as encrypted messages. The virus checker can also report that it is unable to check a bodypart, which sets the same status on the bodypart.

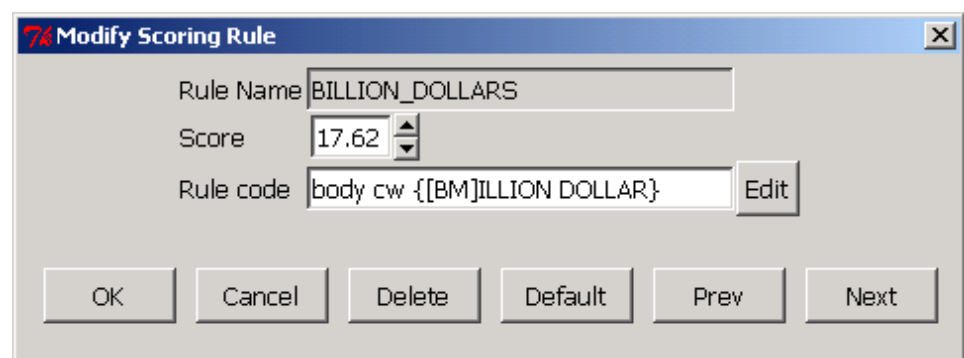
The **Scoring Rules** page allows configuration of one of the most important elements of the Message Switch’s spam-detection capabilities: a large set of rules which match words, phrases or particular words close together (for example, the words ‘adult’ and ‘entertainment’ within 10 words of each other) and many other attributes of a message (for example, the presence of a large number of tags in an HTML bodypart is a reasonable indicator of spam) and assign a score to the rule. Some rules have negative scores – in other words, a match on such a rule indicates that the message is *less* likely to be spam. The default scores are generated by analysis of a large number of messages, which have been pre-sorted into spam and not-spam categories.

**Figure 40.11. Message Scanning - Scoring Rules**



The configuration page for an individual rule allows the score assigned to the rule and the code which makes up the rule itself to be modified. The rule codes are described more fully in [Section 40.5.1, “Action rule programming language”](#). The configuration page for a typical rule is shown below.

**Figure 40.12. Message Scanning - Modify Scoring Rule**

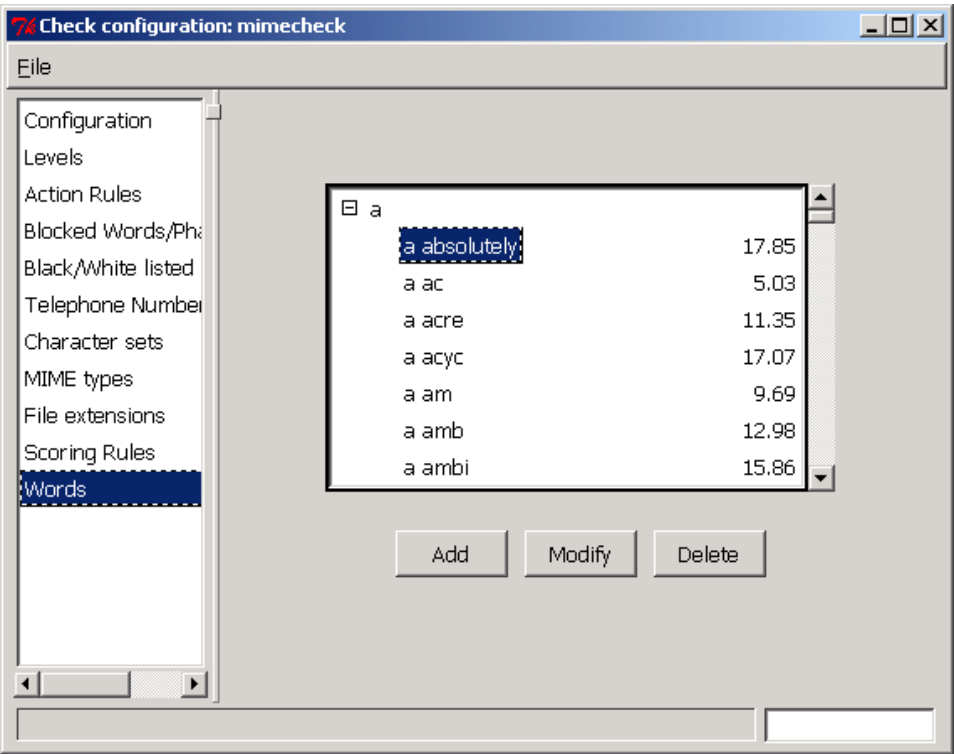


Scoring rules actually make up the bulk of the anti-spam checking data. The rules are normally supplied in a preconfigured form by Isode Limited (there are more than 1000 of them in the standard dataset).

Any changes which you make to the standard scoring rules are stored as overrides/additions to the original scoring data, so that your changes will not be lost if a new basic dataset is downloaded from Isode Limited.

The final configuration screen allows scores to be assigned to the presence of individual words in the message. Words can be added, deleted or have the score (positive or negative) assigned to them modified as required.

Figure 40.13. Message Scanning - Words



When modifications have been made to the data, it is necessary to perform a 'build' before the data can be used by the checker channel. This process loads the data in a pre-compiled form into a cache, from which it is read by the checker channel.

40.1.3 Template files

As described in the previous section, pointers to several warning message template files can be configured for the checker channel. These message templates are normally located in the (*ETCDIR*)/switch/msgcheck directory. Template files can contain the text macros shown in Table 40.1, "Template file macros". In the template file use the macros in the following form, including the brackets:

```
$(macro)
```

The values substituted are derived from the problem message header or the status text for the bodypart itself.

Table 40.1. Template file macros

Macro	Expansion
from	the sender of the message

Macro	Expansion
to	a list of the primary recipients
cc	a list of the copy recipients
bcc	a list of the blind copy recipients
id	the P2 or RFC 822 message ID
date	the message date
subject	the subject
report	for a replacement text, the report for the single bodypart. For cover notes and warning messages, a list of the status text for all faulted bodyparts.

P2 Teletex string values are converted to ASCII, and O/R Addresses are represented in standard O/R Address syntax (as defined in RFC 1327), but using ; as a separator. RFC 822 values are copied verbatim.

To maintain readability, macros which are liable to be replaced by several lines of text should be placed at the start of a line. The following example shows how a simple message template might be constructed.

```
This is to inform you that you were sent a message by:
$(from)
Id:      $(id)
Subject: $(subject)
This message contained a virus, and has been returned to the sender.
$(report)
```

## 40.1.4 Virus checking

With the increasing use of messaging to distribute complex documents and also executable files, there is a need to check the bodyparts in messages for viruses. While this can be done at the User Agent, it is easier for an organization to enforce checking by the use of suitable policies in central Message Switches. This section describes how to configure such policies.

---

**Note:** The Message Switch provides an interface which enables virus checking programs to scan messages. The programs themselves are not included. You will therefore need to acquire a suitable virus checking package for your platform. Many packages are available commercially. The Isode Message Switch is optimised to work with the Sophos anti-virus product, but can be configured to work with almost any virus checker which offers a command line interface.

---



---

**Note:** In addition to the above commercial product M-Switch can now use the free ClamAV product. This is available on all platforms including Windows.

---

The configuration interface described in [Section 40.1.2, “Checking configuration GUI”](#) automatically lists all of the available virus interfaces. Three interfaces are provided as standard: one each for the Sophos and ClamAV products and a dummy 'None' interface, which simply returns 'OK' instead of checking messages. The interfaces are provided by Tcl scripts, enabling the addition of new interfaces as required, as described below.

### 40.1.4.1 Adding an anti-virus interface

For the purpose of this example, suppose you wished to provide an interface to the "FreeAV" virus engine. The following steps should be followed:

1. Copy `(LIBDIR)/msgcheck/virusnone.tcl` to `(LIBDIR)/msgcheck/freeav.tcl`.
2. Edit `freeav.tcl`, changing the line:

```
package provide VirusCheck 1.0
```

to read:

```
package provide VirusFreeAv 1.0
```

---

**Note:** The package name *must* start with `Virus`.

---

3. Fill in the three procedures which the script exports - these are `Init`, `Files` and `Done`.
  - The `Init` procedure should normally contain the line:

```
set ::MsgCheck::config(nodelete) 1
```

to prevent deletion of files from the temporary directory in which the checking is performed. Any other once-off initialization which can be performed here – this function is called when the checker channel starts up. The `nodelete` option only applies if a temporary directory is defined. It can be set if the interface to the actual virus checker can deal with a list of filenames. In some cases, the interface is passed not a list of files but the directory (folder) in which the files are found. In this case `nodelete` should NOT be specified. This is not relevant if the temporary directory is not specified in the main configuration page.

The main checking procedure is `Files`. The first argument passed to it is a Tcl list. Each element of the list is relative pathname (relative to the current working directory) of a file containing bodypart data. Each file should be checked.

If the second argument is a `true` value, then the procedure can attempt to mend any problems with the bodyparts.

The procedure should return a Tcl list. Each element of this list should correspond to a file passed in the input list. Each element of this list is itself a Tcl list. The first element is the file path, as in the input. The second element is the status of the bodypart, and should be one of the strings:

OK

if there is no problem with the bodypart.

NOTOK

if there is a problem.

REPAIRED

if there is a problem which has been fixed.

UNCHECKED

if the procedure is unable to check the bodypart.

The third element of the list is an optional text message. This can be used to report the problem with the bodypart.

- The `done` procedure, which has no parameters, is called when the checker channel process exits. This can be used to clean up any resources allocated in the initialization procedure.
4. Edit the file `pkgIndex.tcl`, located in the same directory as your new anti-virus interface. Add a new line to the end of the file:

```
package ifneeded VirusFreeAv 1.0 [list source [file join $dir
freeav.tcl]]
```

5. Run the configuration interface. You should now find that 'FreeAv' appears on the list of available virus engines.

#### 40.1.4.2 Adding the Clam AV anti-virus interface

1. Download the ClamAV for Windows from:

<http://w32.clamav.net/downloads/clamAV.msi>

to a local folder e.g. *C:\ClamAV*.

2. Run the *clamAV.msi* installer accepting all default settings.
3. Ensure that there is a folder *C:\Windows\Temp*, if not create it as this is what ClamAV uses as the temporary directory
4. In the file *C:\Program Files\clamAV\conf* you may want to change the following line:  
#TCPAddr 127.0.0.1

See comments in the file for more information.

5. Get the latest Virus signatures by running the following from the DOS command line:  
*C:\Program Files\clamAV\freshclam*
6. You can now run the ClamAV daemon from the DOS command line to test as follows:  
*C:\Program Files\clamAV\clamd*
7. You will need to configure your Scanconfig to use the ClamAV Virus scanner.
8. Test using the EICAR Test File sent to one of your email recipients.
9. The recipient should get a message with the subject line "Found Eicar-Test-Signature"
10. You should now install the ClamAV daemon as a service by following the instructions below.
11. Ensure that the Tcl scripts have your ClamAV installation path correct.
12. Using the Iside Service Manager tool, select **Service** → **Add** to create a new service. Give the service a name of *isode.pp.clamd*, a suitable description string, an executable path value of: "*C:/Program Files/Isode/bin/ismsvc.exe*" -service *isode.pp.clamd* and an **Arguments** value of: *C:/Program Files/Isode/bin/clamd\_service.tcl* Remember to set the **Startup Type** value for the service to **Manual**.
13. Repeat step 2 with service name "*isode.pp.freshclam*" and Tcl script name "*freshclam\_service.tcl*".
14. Go to the Configure Startup Order window, and ensure that the two new services are configured to start before any of the MTA services.

By default, the *clamd\_service.tcl* script will run the freshclam program every 60 minutes. You can change this period by editing the script.

---

## 40.2 Rejecting messages from senders

This section describes how you can use various aspects of the Message Switch's configuration to prevent junk email ever reaching its intended recipients.

### 40.2.1 Using authorization

The aim of authorization configuration is to enable the MTA to either reject junk mail at the SMTP level (i.e. the SMTP inbound channel rejects the message), or to generate a

non-delivery report once the message has been accepted. In either case, the message will never be seen by the intended recipient.

The way you configure authorization will depend on the problems being addressed and on the total system configuration. The sections which follow describe how to implement four controls:

1. Dealing with techniques where the identification of the source host is hidden or faked. This is described in [Section 40.2.3, “Hosts not registered in the DNS”](#).
2. Blocking mail from identified hosts. This is described in: [Section 40.2.5, “Known bad sender domains”](#) and [Section 40.2.6, “Specific hosts”](#).
3. Blocking mail from specific message originators or originators matching a specific pattern. This is described in [Section 40.2.7, “Specific users”](#) and [Section 40.2.8, “Bad addresses from good domains”](#).
4. Making use of third-party blacklists to block incoming mail. This is described in [Section 40.2.4, “Using realtime blackhole lists”](#).

Additional controls are available which make it harder for originators of junk email to obtain information about valid recipient addresses within the MTA’s local domain. This is described in [Section 40.3, “Preventing address harvest attacks”](#).

## 40.2.2 Rejection mechanisms

The controls described above involve configuring the MTA to reject Internet Mail messages at one of three levels:

- SMTP connection level.
- SMTP protocol level.
- Application level, by using the authorization system to block the message and generate a non-delivery report. Note that if the non-delivery report cannot be delivered, which is common with junk email, it is likely that the local postmaster will get the resulting report.

The two primary keys required to implement these rejection mechanisms are:

- The true domain name of the calling Internet Mail host.
- The address of the sender as given in the MAIL FROM: line in an RFC 822 message header.

The following table shows the level at which the controls operate and the key information required for the mechanism to work:

	Key Information	
	Domain name of calling Internet host	Sender address
Reject SMTP connection	Use authorization to block relay (see <a href="#">Section 38.1, “Authorization”</a> ).	
SMTP protocol level rejection	Blocking traffic from specific hosts (see <a href="#">Section 40.2.6, “Specific hosts”</a> ).	Known ‘bad sender’ domains (see <a href="#">Section 40.2.5, “Known bad sender domains”</a> ).
		Use of realtime blackhole lists (see <a href="#">Section 40.2.4, “Using realtime blackhole lists”</a> ).

	Key Information	
	Domain name of calling Internet host	Sender address
Generate non-delivery report	Configuring the MTA to prevent relay (see <a href="#">Section 38.1, “Authorization”</a> ).	Blocking traffic from specific users (see <a href="#">Section 40.2.7, “Specific users”</a> ).
	Blocking traffic from specific hosts (see <a href="#">Section 40.2.6, “Specific hosts”</a> ).	Blocking bad addresses from good domains (see <a href="#">Section 40.2.8, “Bad addresses from good domains”</a> ).

Several of the control techniques:

- Configuring the MTA to prevent relay (see [Section 38.1, “Authorization”](#)),
- Use authorization to block relay (see [Section 38.1, “Authorization”](#)),
- Blocking traffic from specific hosts (see [Section 40.2.6, “Specific hosts”](#)),
- Rely on good results from reverse lookup of IP addresses in the DNS. For this the resolver being used must verify that the result of a reverse lookup (IP address → domain name) is actually satisfied by a corresponding forward entry (domain name → IP address list).

Recent releases from most major UNIX manufacturers have resolver libraries which meet this requirement, but it is worth checking.

On Windows 2000, the inbuilt resolver library does this.

If this requirement is not fulfilled, an originator of junk email could create a reverse entry in the DNS for one of his own IP addresses, which resolves to fake-host.naive.example.net and then get mailhost.naive.example.net to relay the message believing it to come from an internal host, even if external relay is blocked (see [Section 38.1, “Authorization”](#)).

### 40.2.3 Hosts not registered in the DNS

A common tactic of people sending junk email is to make the messages come from, or appear to come from, hosts which are not properly registered in the DNS and for which it is not possible to perform an IP address reverse lookup. To reject such calls, ensure the SMTP channel is *not* configured with `ininfo=sloppy`.

---

**Note:** This will reject mail from badly configured systems but not from well configured ones. However, as badly configured systems seem to be commonplace on the Internet today, this mechanism should be used with caution.

---

Strictly speaking, this and some of the other control techniques described here are in contravention of the Internet hosts requirements, RFC 1123. These requirements are probably too weak to help deal with abuses of the Internet such as junk email.

### 40.2.4 Using realtime blackhole lists

The Realtime Blackhole List is maintained by the Mail Abuse Prevention System (MAPS). At its simplest, this is a list of the IP addresses of hosts from which junk email is known to originate. Access to the list is via reverse DNS lookup – i.e. if the Message Switch receives an incoming connection from IP address 192.5.5.1, a reverse DNS lookup of 1.5.5.192.blackholes.mail-abuse.org is performed. If an 'A' record with a specific value is found (the default is "127.0.0.2"), the IP address in question is considered to be blacklisted.



Use of the Realtime Blackhole List is controlled by the presence of the `rb1` switch in the SMTP inbound channel's `ininfo` configuration entry. If this is set, it enables the Realtime Blackhole List (RBL) feature. A specific domain can be specified, which is used as a suffix to the calling IP address, for use with local implementation. An alternative target address can also be specified, as some RBL domains use non-standard addresses. Multiple RBL domains can be specified, in a semicolon-separated list (they are used by the SMTP inbound channel in the order in which they are specified). The syntax of the switch is thus:

```
rb1=<rb1_domain>[ "+"<target_address>][ ";"<rb1_domain>... ]
```

Specifying just `rb1` is equivalent to using the normal RBL domain and default target address, as if:

```
rb1=blackholes.mail-abuse.org+127.0.0.2
```

were used.

## 40.2.5 Known bad sender domains

The following technique can be used to block specific domains if tables are being used for routing (i.e. the `inlookup` for the SMTP channel starts with `table` or `dns-tbl`). The domain is deliberately misconfigured by placing an entry like

```
bad-domain.example.net:mta=nowhereland
```

in the domain table, and no entry for `nowhereland` in the channel table. There will be some logging messages from submit complaining about the misconfiguration, but these are harmless.

## 40.2.6 Specific hosts

To block specific calling hosts, see the *M-Switch Advanced Administration Guide*, 'M-Switch Rules' section.

## 40.2.7 Specific users

To block specific calling users, see the *M-Switch Advanced Administration Guide*, 'M-Switch Rules' section.

## 40.2.8 Bad addresses from good domains

Sometimes a generally good domain is used as the source of messages but the local part of the mailbox address is invalid and follows a particular pattern. For example, it could be all digits.

To configure this, see the *M-Switch Advanced Administration Guide*, 'M-Switch Rules' section.

---

# 40.3 Preventing address harvest attacks

One way in which the originators of SPAM messages obtain email addresses is via Address Harvest attacks, where an SMTP client program is used to test a large range of possible

recipient email addresses (e.g `a@headquarters.net`, `b@headquarters.net`, `c@headquarters.net` etc) in the hope of finding some valid addresses.

The SMTP inbound channel supports two configuration switches which can be used to reduce the vulnerability of the system to such attacks.

- By setting `maxerr=<n>` in the channel's `ininfo` entry, a limit of `<n>` can be placed on the number of address related errors generated via SMTP MAIL, RCPT and VRFY commands. When this limit has been exceeded, all further commands will be accepted, but both valid and invalid addresses will be faulted.
- Setting `errdelay=<n>` in the channel's `ininfo` entry will cause the response to failed MAIL, RCPT and VRFY commands to be delayed by `<n>` seconds. This is designed to slow down attempts to harvest addresses, and is akin to delays on login failure.

---

## 40.4 Reducing SPAM

### 40.4.1 Greylisting

Greylisting is a technique for reducing the amount of junk email accepted by an MTA. The technique relies on the fact that a large proportion of junk email is transferred directly into the target MTA using an SMTP-aware script, rather than a 'real' MTA. Such scripts generally do not have any retry logic – they simply traverse a list of email addresses, making a single attempt to transfer a junk mail to each one in turn. Thus if a temporary error (i.e. "Please try again later") is returned in response to a message transfer attempt, a lot of junk email can be blocked.

Messages which originate from MTAs rather than scripts will be retried after a (usually) short interval, at which point they can be accepted.

Tracking of whether this is the first or a subsequent attempt to transfer a message to a particular recipient is achieved by using a database to store a tuple containing the originator address, recipient address and IP address of the sending system for each recipient of a message. To guard against scripts which do retry immediately, a short (e.g. 10 seconds) timer is associated with each tuple: any retries which occur before the timer expires will be ignored.

The main danger with the use of Greylisting is that there are some real MTA implementations in use which do not retry properly in response to temporary errors. This can mean that non-junk messages may end up being non-delivered. To guard against this it is possible to exclude specific senders, recipients or sending systems from the greylisting process. Tools are provided which allow examination and cleanup of the greylisting database.

#### 40.4.1.1 Support in inbound channel

Greylisting is enabled by adding configuration to the SMTP inbound channel's `ininfo` entry. The keys and their values used are:

`Greylist delay (greylist)`

If this key has a value associated with it, this is taken to be the delay in seconds which the SMTP channel should apply before a message with a new IP address/originator/recipient tuple is accepted. If the key is present without a value, the default delay of 10 minutes is assumed. If the key is not present at all, no use of greylisting will be made by the channel.

**Greylist Database (greylistdb)**

If this key is present, the associated value gives the path for the greylisting database to be used by this channel, overriding the default value. An absolute file path is interpreted as such, while a relative path is interpreted as being relative to the parent directory of (*QUEDIR*).

**Greylist Expiry (greylist\_expiry)**

If this key is present it must have a value associated, which is the interval in seconds after which an entry in the greylisting database is considered to have timed out. If this key is not present, a default value of  $60 * 60 * 24 * 30$  (i.e. 30 days) is used. Once an entry has timed out, a subsequent message with the same IP/originator/recipient tuple will be treated as if the tuple had been seen for the first time. Expired records are also considered suitable for deletion by the "tidygreylist" program.

**Whitelist address (whitelist\_addr)**

If this key is present, the associated value must contain a comma separated list of recipient or originator email addresses for which greylisting should not be applied.

**Whitelist IP (whitelist\_ip)**

If this key is present, the associated value must contain a comma separated list of IP addresses for which greylisting should not be applied. A network can be specified by replacing the host part of the IP address with zero.

### 40.4.1.2 Support in outbound SMTP channel

Support for greylisting can also be enabled in outbound SMTP channels. In this case, successful transfer of a message causes a greylist database record to be added such that if the message recipient replies to the message originator, sending the reply message from an MTA which shares the network part of its IP address with the IP address of the system which received the original message no greylist delay will be imposed. A single configuration key is required in the SMTP channel's *outinfo* entry:

```
greylist
```

In addition, the *greylist\_expiry* and *greylistdb* keys described for the inbound channel above can also be specified.

### 40.4.1.3 Greylist database tools

#### 40.4.1.3.1 dumpgreylist

The *dumpgreylist* utility allows the contents of the greylist database to be examined. The default mode of operation is to print out every record in the database. Switches allow information for specific originators or recipients to be selected. The complete set of command line switches supported is:

- s  
just print a summary instead of all the records in the database.
- d  
print detailed information for each record.
- D  
print very detailed information for each record.
- r *<email address>*  
print information for specified recipient only.
- o *<email address>*  
print information for specified originator only.
- i *<IP address>*  
print information for specified IP address only.

- m *<nn>*  
only print records created within last *nn* minutes.
- h *<nn>*  
only print records created within last *nn* hours.
- t *<nn>*  
only print records created since *nn*, where *nn* is seconds since the start of the Unix epoch.

#### 40.4.1.3.2 tidygreylis

The tidygreylis utility deletes 'expired' records from the database. The following switches are supported:

- s  
print summary of records deleted when complete.
- v  
verbose mode - print information for every record considered for deletion.

---

## 40.5 Checking Channel Configuration

This section describes some advanced features to configure actions on identified messages.

### 40.5.1 Action rule programming language

The program field for an Action Rule contains a small program written in a simple programming language. This language uses reverse Polish notation and operates on a stack of integers. Note that scores are treated as integers, by scaling them by a factor of 100. Boolean values are zero for false and -1 (i.e. all bits set) for true.

There are a set of internal named variables. For each message these are set to zero, except that some can be set to other values from the channel or per-recipient configuration.

There are a number of program instructions, some of which take arguments. The current instructions are:

- exit  
exits the current program
- bool  
takes the top value off the stack and pushes `true` if the value is non-zero, or `false` otherwise.
- not  
takes the top value off the stack, pushes `true` if the value is zero and `false` otherwise.
- neg  
negates the value on the top of the stack
- inv  
performs a bitwise inversion of the value on the top of the stack
- cond  
pops three values off the stack. If the original top value was non-zero, push the second from top, otherwise push the third from top.
- add  
pop two values and push the sum of the two

`sub`  
 pop two values and push the result of subtracting the top value from the next to top.

`mul`  
 pop two values and push the product.

`div`  
 pop two values and push the quotient

`rem`  
 pop two values and push the remainder

`and`  
 pop two values and push the bitwise and

`or`  
 pop two values and push the bitwise or

`eq, ne, ge, le, gt, lt`  
 comparison operators. Pop two values, and push the boolean for the result of the comparison. The ordering is (next to top) op (top).

`drop`  
 discard the top value on the stack

`dup`  
 push a copy of the top value

`swap`  
 swap the top two values

`rot <val>`  
 moves the value at depth <val> to the top of the stack pushing the intervening values down one.

`pushd <val>`  
 push the floating point (score) value, scaled to be an integer

`pushi <val>`  
 push the integer value

`pushs`  
 push the current value of the score

`pushv <name>`  
 push the current value of the named variable

`sets`  
 set the value of the score (scaled to be an integer).

`setv <name>`  
 set the value of the named variable

`body <flags> <RE>, header <flags> <RE>, subject <flags> <RE>, word <flags> <RE>`

These add a regular expression to those associated with the given set; body applies to the contents of message bodies after the transfer encoding has been removed, word applies to the data dividing into words, and for text/html applies to the data stream after removal of HTML tags etc, subject means the message subject, and header means other header fields. <flags> can be an empty string or can contain the flags:

- `c` - match exact case (normally upper and lower case match)
- `w` - match the string within a word (normally markers for the start and end of a word are added).

Regular expressions are described in [Section 40.5.2, “Regular expressions”](#).

`seq <data>...`

This defines a sequence of regular expressions. These need to be found in order. The data consists of: the name of the area in which the search is made and the number of initial REs to be used, followed by that number of initial REs, which have each:

```
<flags> <RE> <offset>
```

and then there is the final

```
<flags> <RE>
```

If the offset is non-zero, then for `body`, `header` and `subject` locations, it gives the maximum offset in bytes between the places the REs are recognised. If it is zero there is no limit to the separation.

Effectively, each RE in sequence is only found if it is present and preceded by any predecessor within the given offset. If the last one is found, then the number of occurrences is pushed onto the stack.

```
tag <RE> <list>
```

The RE should be an expression which matches an HTML tag. The list is itself a program. It is executed at the close of the tag. It will normally contain `attr` opcodes to match attributes and values within the tag. The program can set the score. If on exit the score is non-zero, then this item will cause a non-zero value to be pushed onto the stack.

```
attr <name> <value>
```

Used only in programs attached to tags. The name is a pattern to match an attribute name, and the value is a pattern to match a value. The value can be the empty string, in which case the presence of the attribute will be tested.

For a scoring rule, after evaluating the program, the value on the top of the stack is popped. If it is non-zero, then the score associated with the rule is added to the overall score.

For an action rule, after evaluating the program, the value on the top of the stack is popped. If non-zero, and the priority of the rule is larger than the priority of the current rule to be used, then this action rule replaces the current rule.

It is possible to configure program fragments, for example, to set variable values. However, this cannot be done from the GUI interface. These fragments are executed prior to the code for any rules.

Scoring rules are all evaluated prior to the action rules. The order of evaluation in these sets is undetermined.

The program text is treated as a Tcl list, and so should be escaped suitably.

## 40.5.2 Regular expressions

Regular expressions can have most standard items. Note that special characters need to be escaped using `\`. There are the following special escaped values:

Value	Matches
<code>\a</code>	BEL
<code>\f</code>	FF
<code>\n</code>	LF
<code>\r</code>	CR
<code>\t</code>	TAB
<code>\x&lt;xx&gt;</code>	The character whose code is given by the two hex digits.
<code>\w</code>	Any alphabetic character
<code>\W</code>	Any non-alphabetic character
<code>\s</code>	Any white space character, includes TAB, CR, LF, FF, VT

Value	Matches
\s	Any non-white-space character (the opposite of \s)
\d	A decimal digit
\D	The opposite of \d
.	Any single character
\^	Does not match a character - toggles the case exact/case ignore state.
[<set>]	Matches the characters in the set. If the first character is ^, then this inverts the match. - can be used to form character ranges. Sets are always case exact. So [A] will only match an upper case letter A.

These single character expressions can be combined with:

RE {n}	n occurrences of RE
RE {n, }	n or more occurrences of RE
RE {n, m}	between n and m (inclusive) occurrences of RE
RE*	The same as RE{0, }
RE+	The same as RE{1, }
RE?	The same as RE{0, 1}
( )	can be used to group: RE1   RE2 means RE1 or RE2.

The regular expression is added to those recognised for the given match area. When the program is executed, the number of occurrences of the expression in the message is pushed onto the stack.

REs to be used in the header context must start with a pattern to match the fieldname, and then :. Note that this applies to patterns in sequences as well.

### 40.5.3 Placing messages in quarantine

The action taken when the checker results in the message being rejected is configurable using scanconfig.

For information on configuring the quarantine feature of M-Switch see [Chapter 36, Message Audit Database](#).

# Chapter 41 SPIF Editor

This chapter describes the SPIF Editor application and explains how to use it to create, edit and view a SPIF (Security Policy Information File) and various utility functions.

The term SPIF refers to Security Policy Information File. A Security Policy is represented as an SDN.801c SPIF in the Open XML SPIF format. A SPIF is structured data which defines for a given policy ID the valid classifications and security categories. It also can define strings to be associated with labels, which are used for mark-up of data for human reading. It can define equivalent policies, which enables labels defined by a different authority to be associated with labels defined in this SPIF. It also defines how the 'Access Control Decision Function' (ACDF) is to be applied.

---

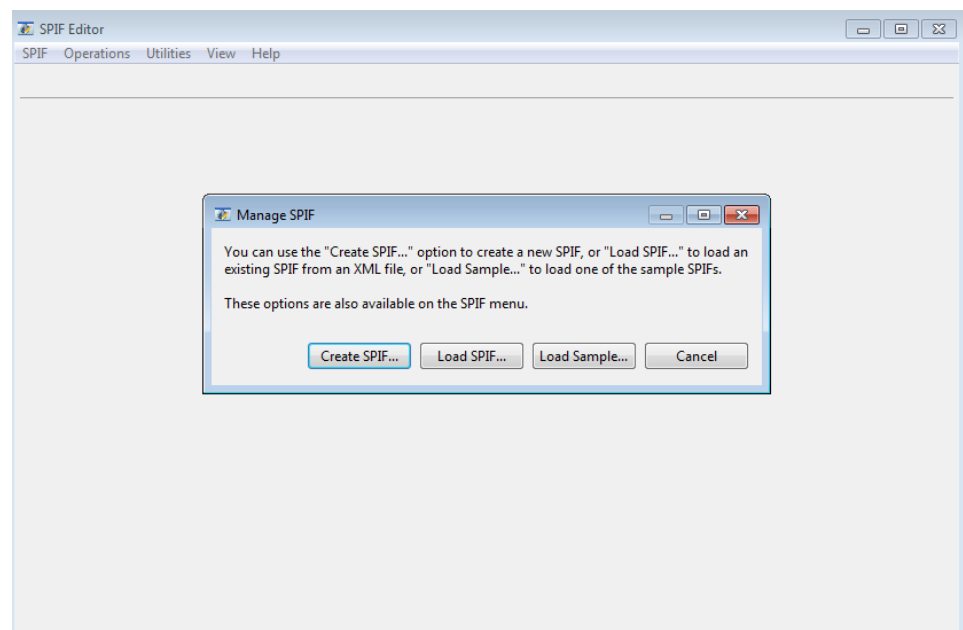
## 41.1 SPIF Editor Overview

SPIF Editor is a GUI that allows you to create, edit and view SPIFs. The primary purpose of the editor is to make it easy to create and manage security labeling for Isode servers and clients. It is not necessary to use the SPIF Editor in order to use security labeling in Isode products, but in many cases it may prove to be the simplest means of doing so.

### 41.1.1 Getting started

On launching the SPIF editor, a dialog will appear that provides options to create a new SPIF, load an existing SPIF XML file or load one of the samples provided as part of its installation.

**Figure 41.1. SPIF Editor Launch Screen**



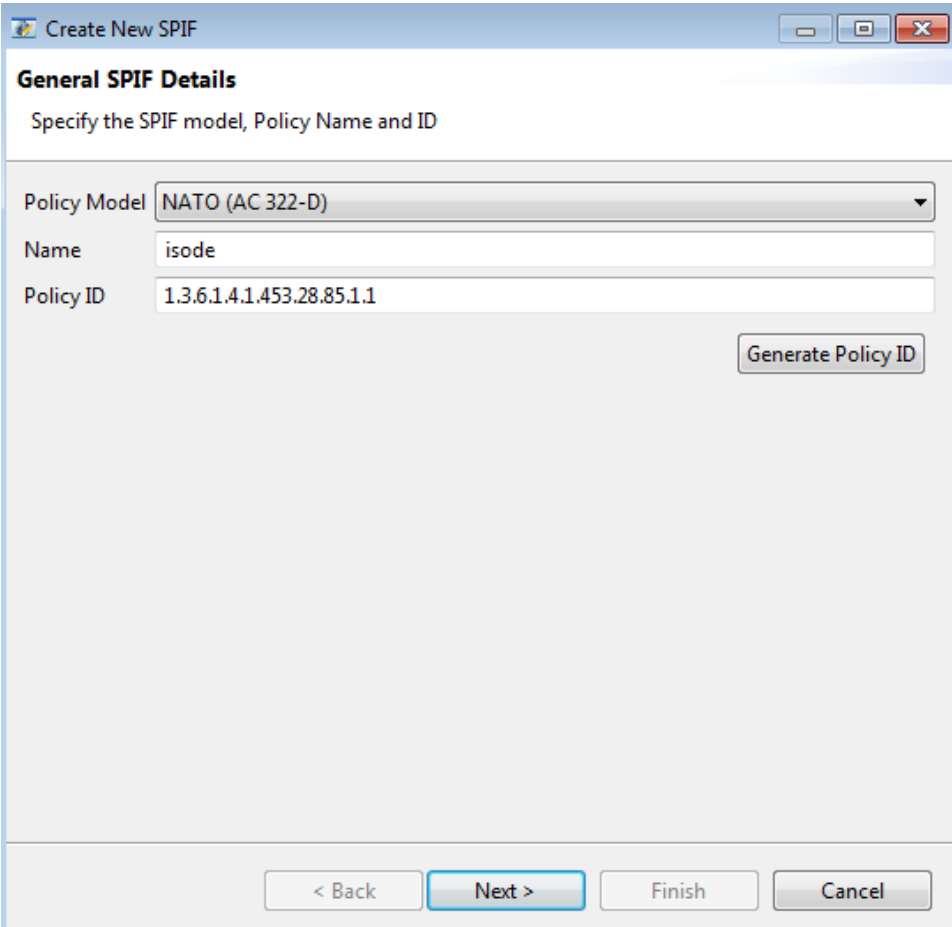


## 41.2 Creating New SPIF

A new SPIF can be created by choosing the "**Create SPIF...**" button on the launch dialog. The option is also available on the **SPIF** → **Create...** menu. The wizard for creating a new SPIF is shown in the figure below.

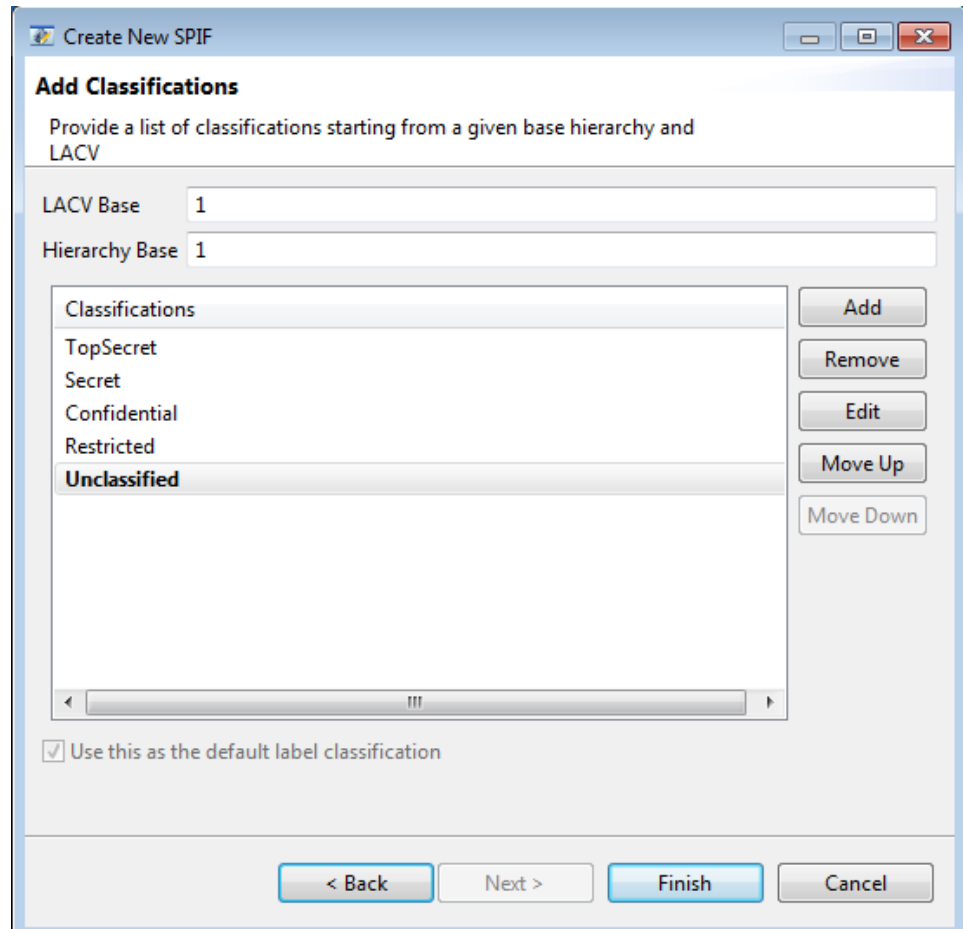
Provide the policy model, name and ID for the SPIF on the first page.

**Figure 41.2. Create SPIF**



The screenshot shows a Windows-style dialog box titled "Create New SPIF". The dialog has a tab labeled "General SPIF Details" with the instruction "Specify the SPIF model, Policy Name and ID". It contains three input fields: "Policy Model" with a dropdown menu showing "NATO (AC 322-D)", "Name" with the text "isode", and "Policy ID" with the text "1.3.6.1.4.1.453.28.85.1.1". A "Generate Policy ID" button is located to the right of the Policy ID field. At the bottom of the dialog are four buttons: "< Back", "Next >" (which is highlighted with a blue border), "Finish", and "Cancel".

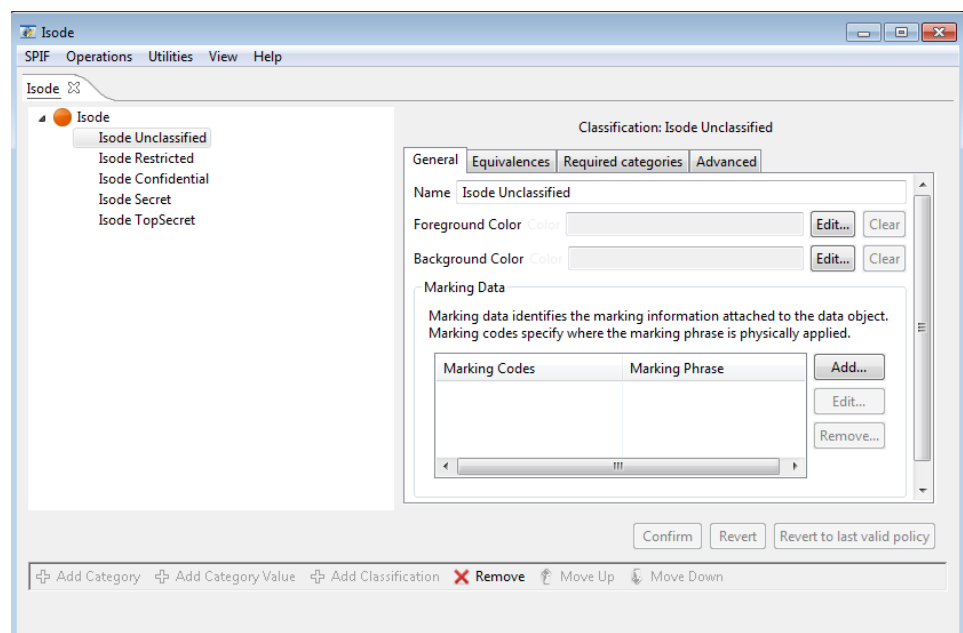
On pressing **Next** button, list of standard classifications will be offered as a default.

**Figure 41.3. Create SPIF Classifications**

The 'Create New SPIF' dialog box is titled 'Create New SPIF'. It has a section 'Add Classifications' with the instruction 'Provide a list of classifications starting from a given base hierarchy and LACV'. Below this, there are two input fields: 'LACV Base' with the value '1' and 'Hierarchy Base' with the value '1'. A list box titled 'Classifications' contains the following items: 'TopSecret', 'Secret', 'Confidential', 'Restricted', and 'Unclassified'. To the right of the list box are five buttons: 'Add', 'Remove', 'Edit', 'Move Up', and 'Move Down'. Below the list box is a checkbox labeled 'Use this as the default label classification' which is checked. At the bottom of the dialog are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

The list can be modified to add or remove classifications. The name of the classifications can be modified using **Edit...** button. The LACV value stands for the classification value whereas the hierarchy governs the ordering of the classifications in the SPIF.

On pressing **Finish**, the SPIF will appear on the SPIF editor as shown below.

**Figure 41.4. Created SPIF**

The 'Isode' SPIF Editor window shows the 'Isode' SPIF in the left pane. The right pane displays the 'Classification: Isode Unclassified' details. The 'General' tab is active, showing the 'Name' as 'Isode Unclassified'. Below this are 'Foreground Color' and 'Background Color' fields, each with an 'Edit...' button and a 'Clear' button. The 'Marking Data' section contains a description: 'Marking data identifies the marking information attached to the data object. Marking codes specify where the marking phrase is physically applied.' Below this is a table with two columns: 'Marking Codes' and 'Marking Phrase'. To the right of the table are three buttons: 'Add...', 'Edit...', and 'Remove...'. At the bottom of the right pane are three buttons: 'Confirm', 'Revert', and 'Revert to last valid policy'. The bottom status bar contains a series of icons and labels: 'Add Category', 'Add Category Value', 'Add Classification', 'Remove', 'Move Up', and 'Move Down'.

## 41.3 Managing Existing SPIF

An existing SPIF XML file can be loaded in a SPIF editor using the **SPIF** → **Load...** menu.

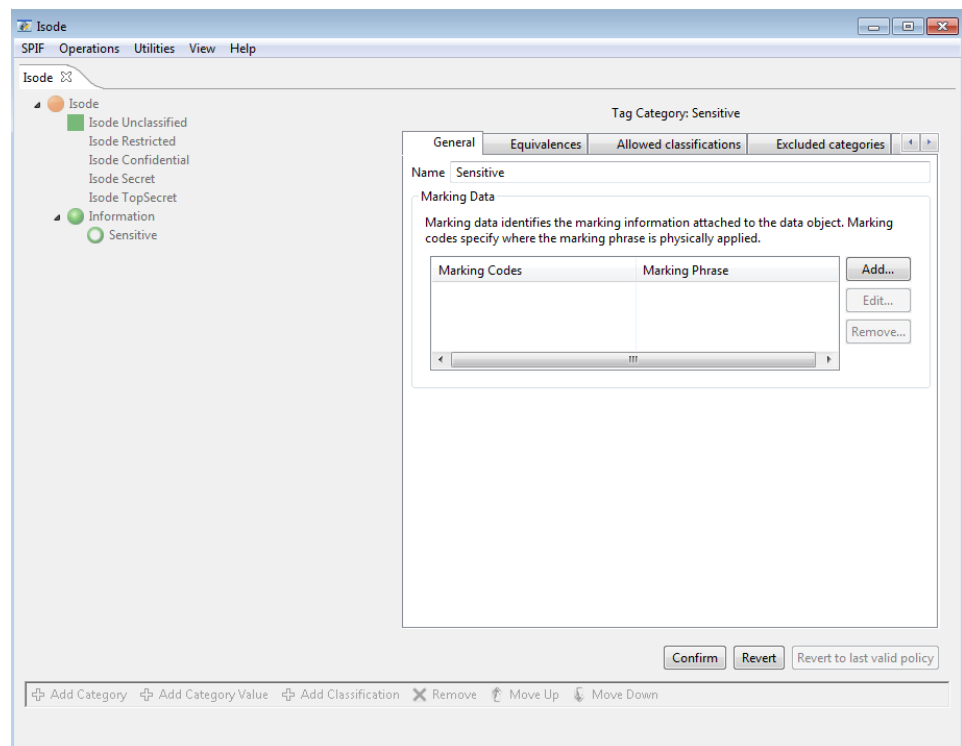
Once a SPIF has been created or loaded from an XML file, it can be viewed or edited using the SPIF editor. The left hand side presents the classifications and categories of the SPIF in a tree format. The classifications are listed on top of the tree followed by categories.

Classifications are displayed using their background color icon and categories are displayed using green circle icons. Note that categories are optional and may not exist in most SPIFs.

On selecting a classification or a category, the right hand side pane will display the details of selected classification or category.

When the selected classification or category is edited, the **Confirm** and **Revert** buttons will get enabled to let you apply the current set of changes or cancel them. Note that the **Confirm** button will not get enabled until the current set of changes made are complete and valid.

**Figure 41.5. Category Edit**



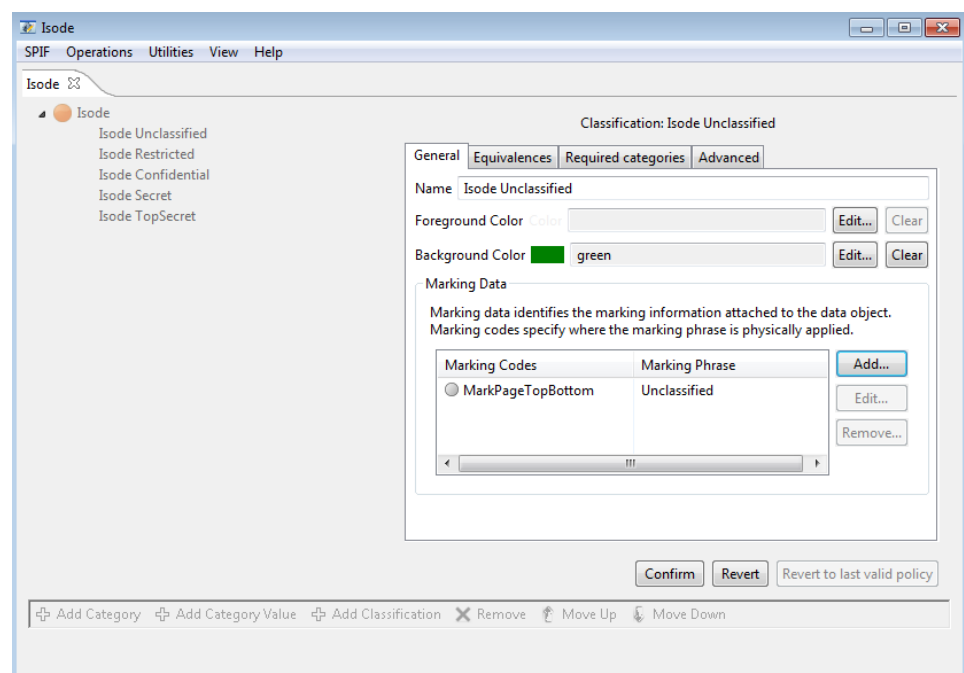
The editor allows you to modify only one classification or category in one operation. For complex policies, a change in more than one classification or category may be required to create a valid policy. If an edit makes the policy invalid **Revert to last valid policy** will get enabled to allow you to revert to last valid state to undo the changes that lead to the invalid policy.

## 41.4 SPIF Classifications

Select a classification on the left hand side in order to edit it. After making the required changes, click the **Confirm** button.

The following figure displays the SPIF after adding a marking code and changing the background color of the classification.

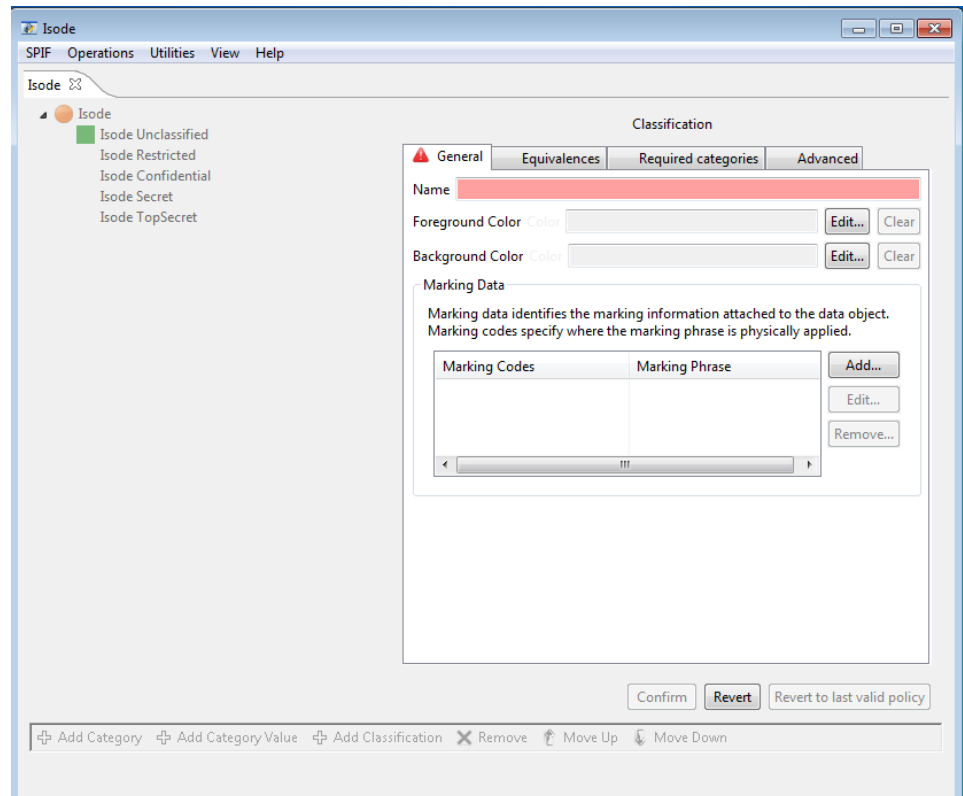
**Figure 41.6. Classification Edit**



The **General** tab lists the most commonly used attributes. Rest of the tabs define advanced parameters that are required for complex SPIFs.

### 41.4.1 Adding Classifications

Select the topmost tree item for the policy and click **Add Classification** button to add a new classification. Provide the details of the new classification to be added on the right hand side pane. The tabs that require mandatory parameters for completing classification creation will display a red icon on the top.

**Figure 41.7. Add Classification**

Once the details of the new classification have been provided, press the **Confirm** button to create the new classification. By default, the editor will fill in default values and in simple cases you will only need to provide the classification name. Colors and markings can be added to suit the requirements.

## 41.4.2 Removing Classifications

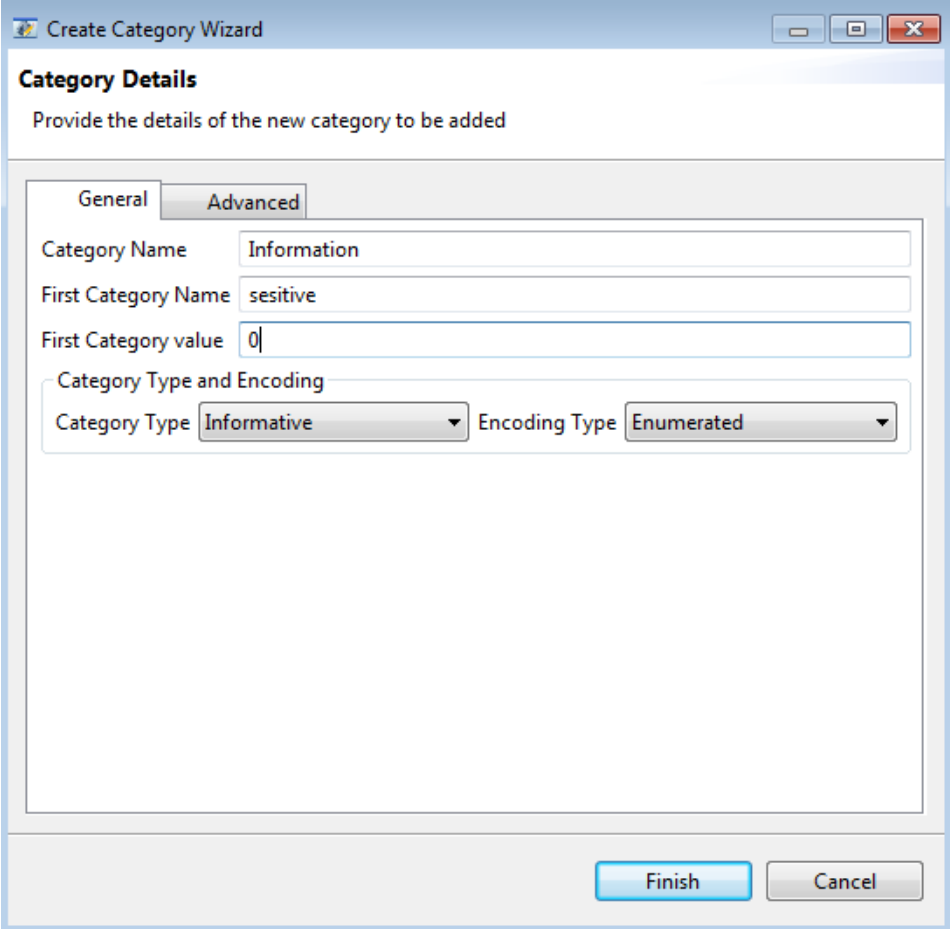
Select a classification and click **Remove** button and confirm to remove the selected classification from the policy. Errors will be reported if removal of a classification invalidates the security policy.

---

## 41.5 SPIF Categories

### 41.5.1 Adding Category

In order to add a new category group, select the topmost tree item for the policy and click **Add Category** button. A wizard to add a new category will be displayed.

**Figure 41.8. Adding Category**

The screenshot shows a window titled "Create Category Wizard" with a standard Windows-style title bar (minimize, maximize, close buttons). The main content area is titled "Category Details" and contains the instruction "Provide the details of the new category to be added". Below this, there are two tabs: "General" (selected) and "Advanced". The "General" tab contains the following fields and controls:

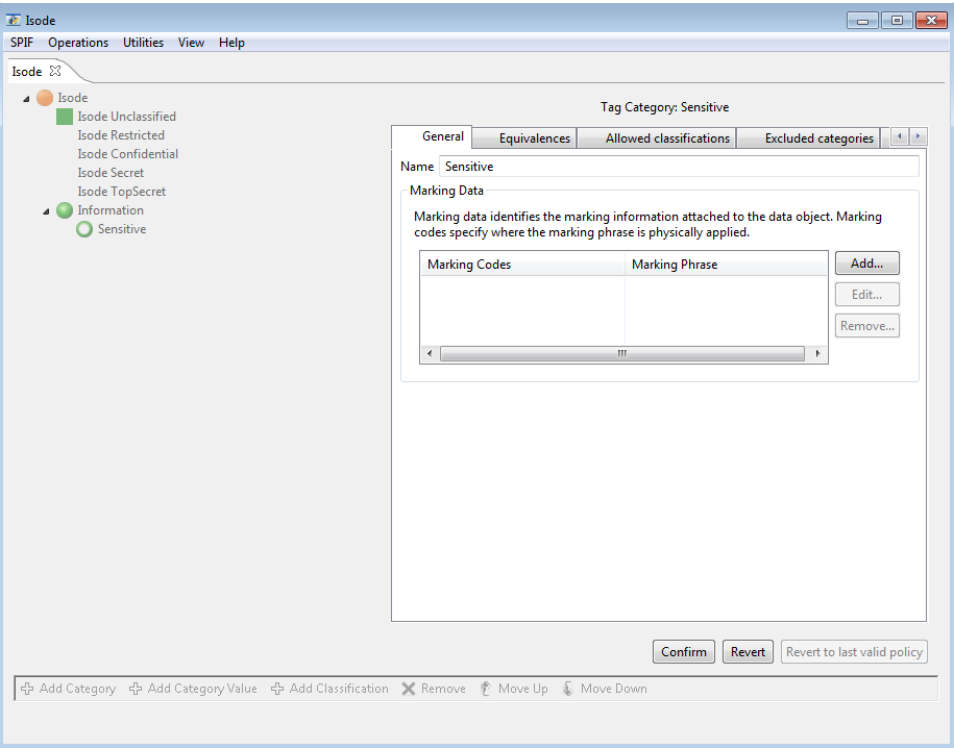
- Category Name:** A text input field containing the text "Information".
- First Category Name:** A text input field containing the text "sesitive".
- First Category value:** A text input field containing the text "0".
- Category Type and Encoding:** A section containing two dropdown menus:
  - Category Type:** A dropdown menu with "Informative" selected.
  - Encoding Type:** A dropdown menu with "Enumerated" selected.

At the bottom right of the dialog, there are two buttons: "Finish" (highlighted in blue) and "Cancel".

The wizard page will ask you to provide the details of the new category group and first value in the group. For simple case, category name and type will have to be provided. See the tooltips on the widgets for more information on the parameters.

On pressing the **Finish** button the editor will display the category to be added on the editor. You can make further changes to the values for this category. Press **Confirm** button to add the category to the SPIF.

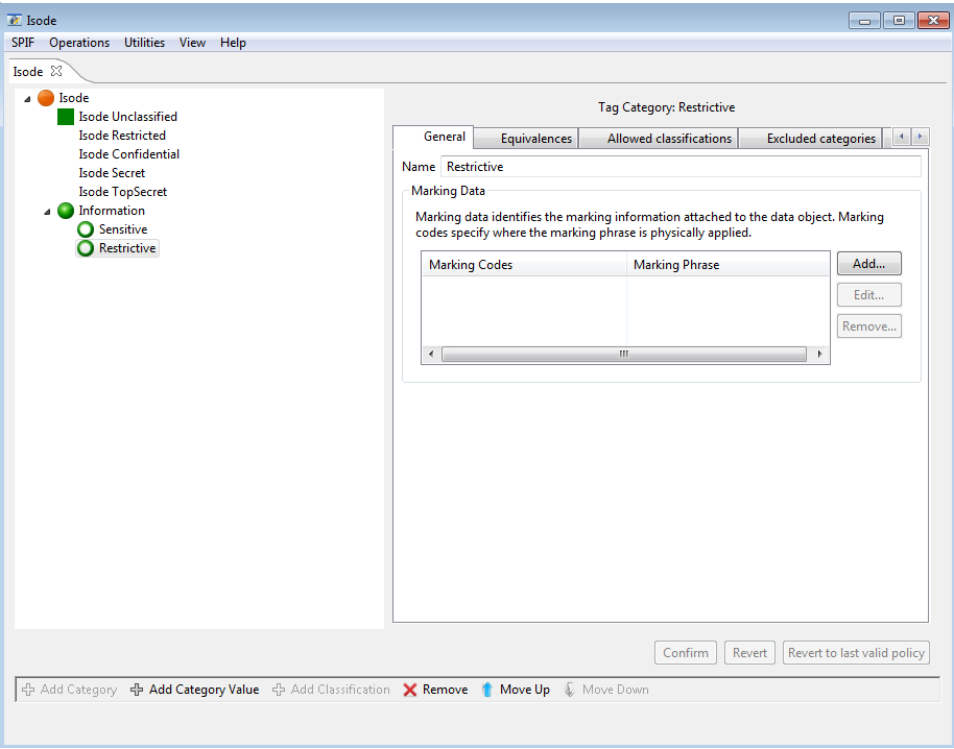
Figure 41.9. New Category



41.5.2 Adding Category Value

To add a new category value to an existing category group, select the category group and click the **Add Category Value** button. The right hand side pane will change to a mode for adding a new category. Provide the name of the category value and other details if required and press the **Confirm** button. The new value will be added to the SPIF as shown below.

Figure 41.10. Adding Category Value



### 41.5.3 Removing Category

Select the category value and click **Remove** button to confirm removal of the category value. Similarly, select a category group and click **Remove** button to remove the selected category group and all its values from the policy. Errors will be reported if removal of a category invalidates the security policy.

### 41.5.4 Moving Categories

Select a category group or value and click on **Move Up** or **Move Down** button to move it up or down the hierarchy. Press **Confirm** to apply the changes to the SPIF.

---

## 41.6 SPIF Utilities

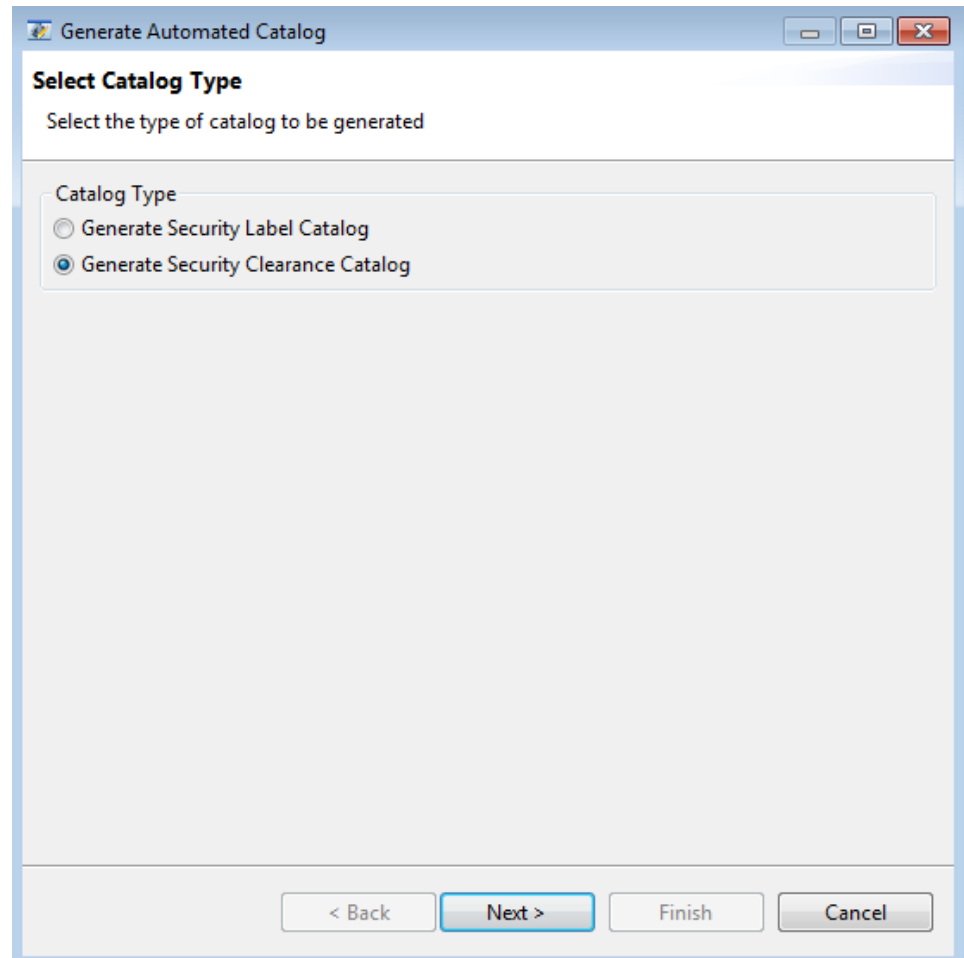
The SPIF editor provides commonly used functions that are available via the **Utilities** menu.

### 41.6.1 Generate Catalog

Security label and clearance catalogs are collections of security labels and clearances. Click the **Utilities** → **Generate Catalog...** menu to launch a wizard to auto generate label and clearance catalogs.

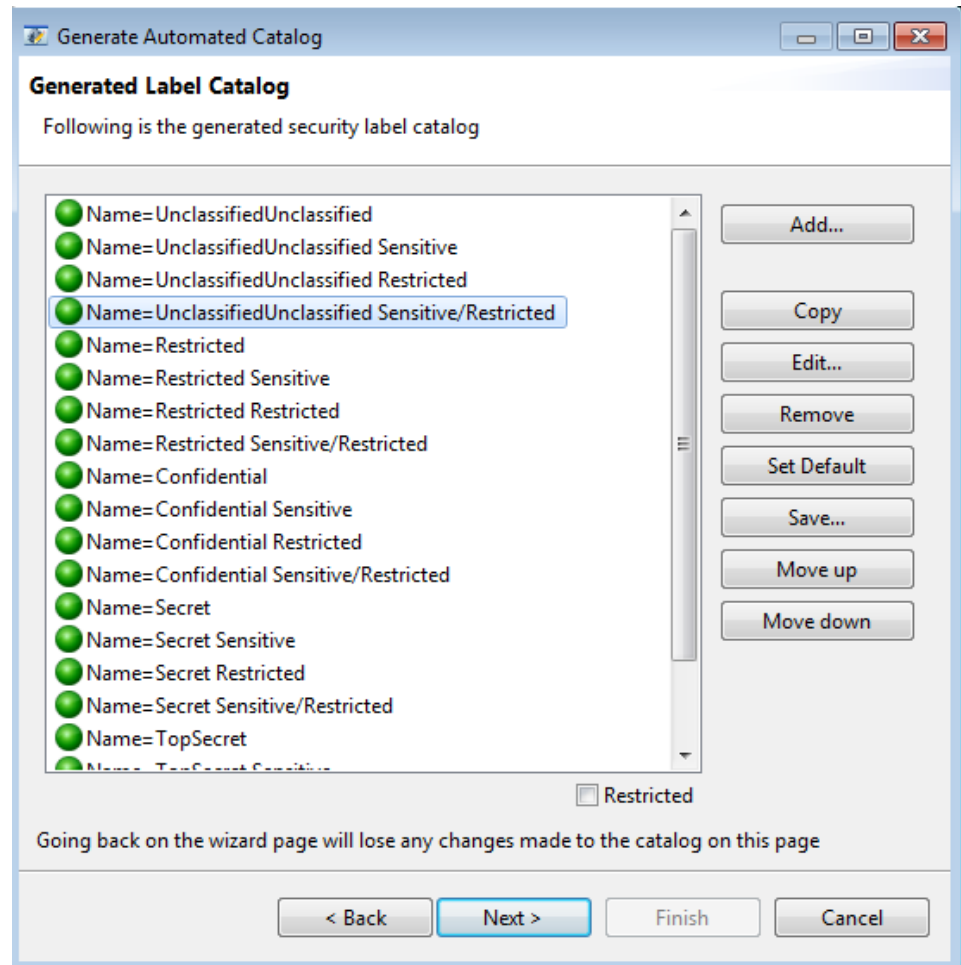
First select the type of catalog to be generated.



**Figure 41.11. Select Catalog Type**

If the policy is simple with few classifications and categories (optional), the wizard will generate a catalog with all possible combinations of classifications and categories. The generated catalog will appear as shown in the figure below that displays a sample auto generated label catalog.

Figure 41.12. Label Catalog



However, for a policy which has large number of classifications and categories, the wizard will present a page to choose a set of classifications and categories to be included in the catalog.

**Figure 41.13. Selected Classifications and Categories**

**Generate Automated Catalog**

**Select Classifications and Categories for Catalog generation**

A catalog will be generated from all possible valid combinations of selected classifications and categories

**Selected Categories**

- UK Restrictive Codewords : BRONCO
- UK Informational Nicknames : CHARGING BULL
- UK Informational Coverwords : RODEO
- UK Enumerated Restrictive Coverwords : ROUNDUP, LONGERCOVERWORDTHANUSUA

**Security Classifications**

☒ Select All

- ☒ UK UNCLASSIFIED
- ☒ UK RESTRICTED
- ☒ UK CONFIDENTIAL
- ☐ UK SECRET

**Security Categories**

- ☒ UK Informational Markings
- ☒ UK Enumerated Permissive Markings
- ☒ UK Informational Codewords
- ☒ UK Restrictive Codewords
- ☒ UK Informational Nicknames
- ☒ UK Informational Coverwords

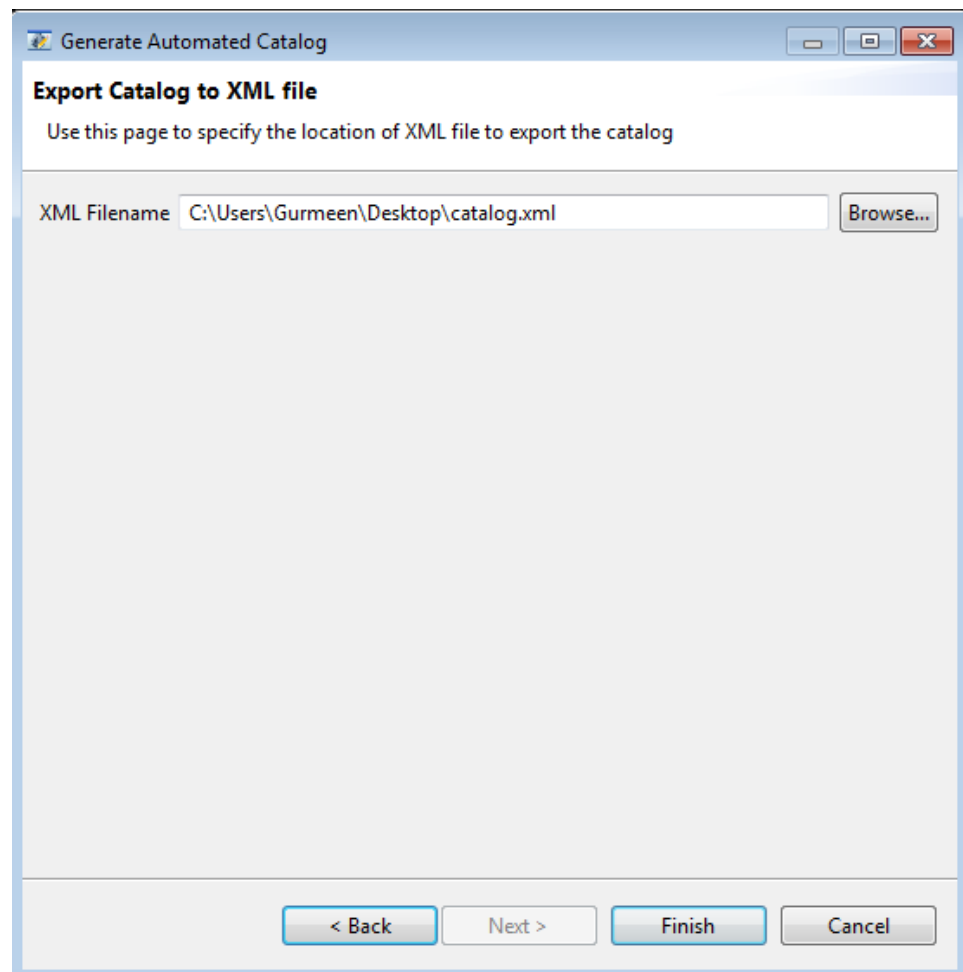
☐ COWBOY

**Warning:** The specified selections would result in 7168 possible labels in the catalog.

< Back   **Next >**   Finish   Cancel

The wizard will attempt to generate a catalog from all possible combinations of selected classifications and categories. A warning will be displayed on the bottom of wizard page if the number of possible combinations is high (in terms of hundreds) and an error will be displayed if the number of combinations is very high (in terms of thousands).

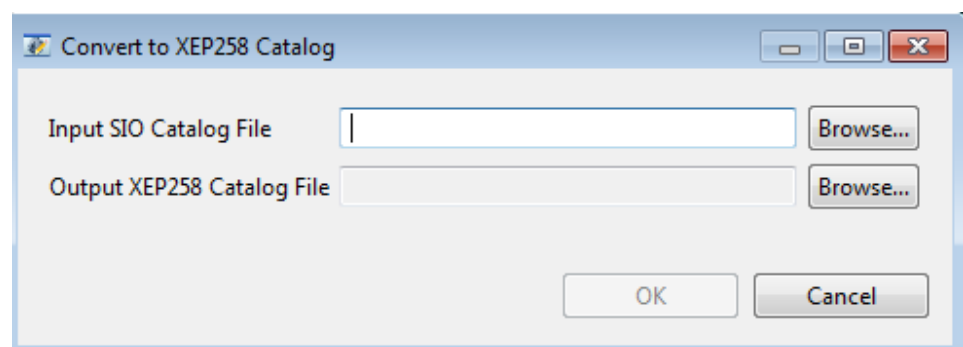
Once a label or clearance catalog has been generated, it can be edited if required on the wizard page that displays the generated list as shown in figure [Figure 41.12, “Label Catalog”](#). On pressing the **Next** button, the page will prompt you to select an XML file location to save the catalog.

**Figure 41.14. Export Catalog**

Press **Finish** to complete the catalog generation.

## 41.6.2 Converting Label Catalog to XEP-258 Format

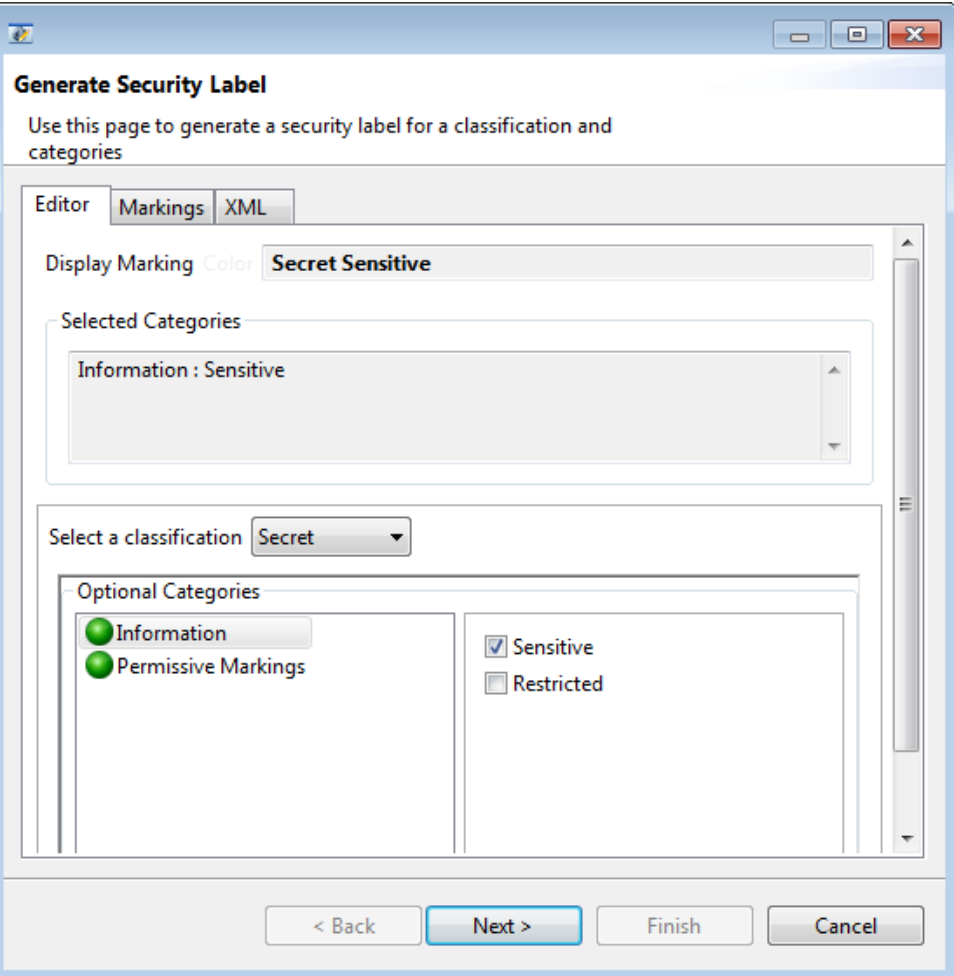
The **Utilities** → **Convert to XEP258 Catalog...** can be used to convert a label catalog to a format that conforms to the XEP 258 format of the XMPP standards.

**Figure 41.15. Convert to XEP258 Catalog**

## 41.6.3 Generate Label

For simple policies, select a classification and one or more categories to generate a label.

Figure 41.16. Generate Label for Simple Policy



Label generation for complex policies is described below.

**Figure 41.17. Generate Label for Complex Policy**

**Generate Security Label**

Use this page to generate a security label for a classification and categories

Editor | Markings | XML

Display Marking **DEMO-UK PROTECT APPOINTMENTS**

Selected Categories

UK Informational Descriptors : APPOINTMENTS

Select a classification **UK PROTECT**

Mandatory Categories

☒ Select One or More

**UK Informational Descriptors**

☒ APPOINTMENTS

☐ BUDGET

☐ COMMERCIAL

☐ CONTRACTS

☐ CONTROL

Optional Categories

☒ UK No Foreign Transmission

☒ UK Enumerated Permissive Nation

☒ UK Informational Descriptors

☒ UK Informational Markings

☒ UK Enumerated Permissive Marki

☒ UK Informational Codewords

< Back | Next > | Finish | Cancel

First select a classification from the drop-down list. Selecting a classification may or may not require inclusion of certain categories. The required categories if any will be displayed as a list in the **Required Categories** pane. The **Optional Categories** pane lists all the categories in the configured policy from which the user can select certain categories to be added to the label. The categories which are disallowed based on the selection of a certain category or the classification will be disabled automatically on the editor. The obsolete categories will be allowed for editing based on whether **Edit obsolete elements** is selected or not.

Selection rules of a category group determine whether it allows selection of single or multiple categories in the group. For single category selection, the categories are displayed using radio buttons and for multiple category selection they are displayed as check-boxes.

The markings get updated on the **Markings** tab when a valid combination of categories has been selected.

---

**Note:** Rules for label editing are based on SDN.801c and are configured in the security policy.

---

## 41.6.4 Generate Clearance

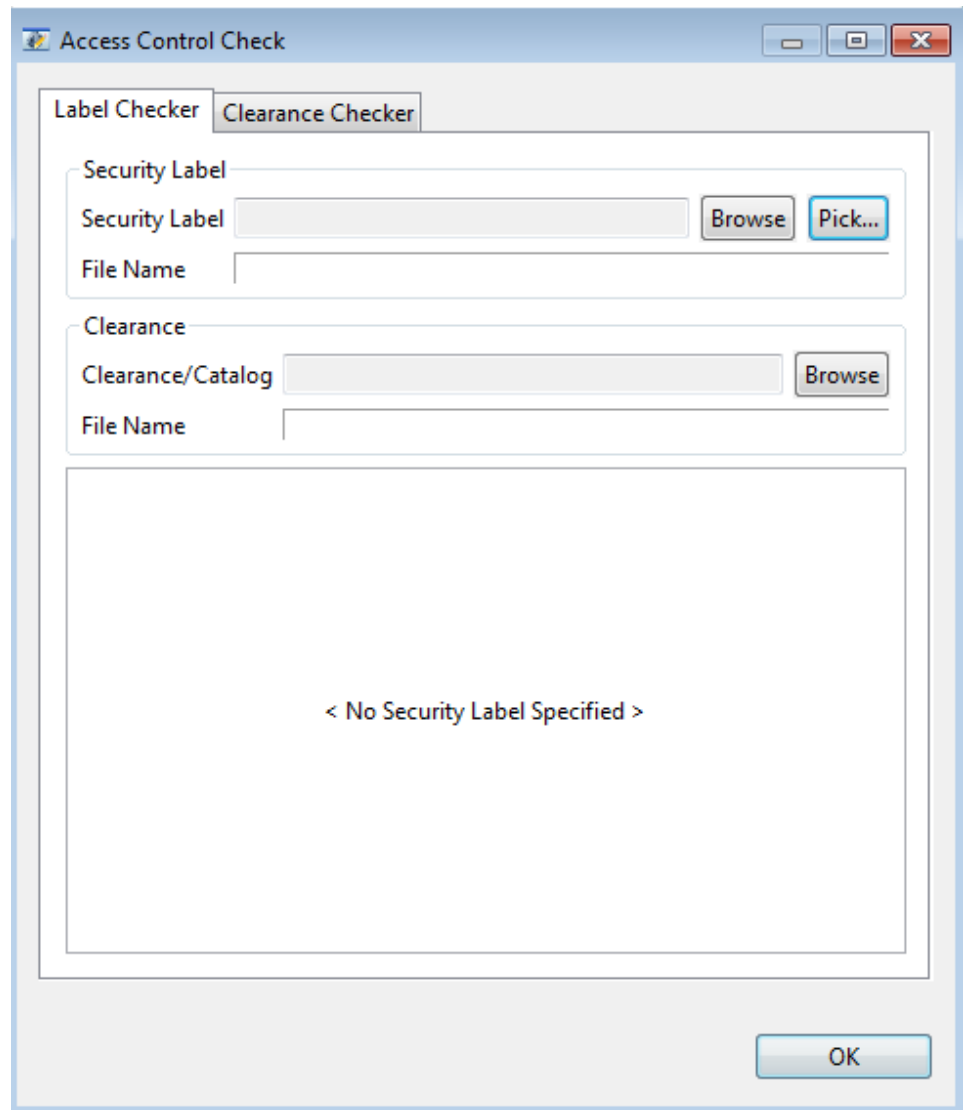
The **Utilities** → **Generate Clearance...** can be used to generate a security clearance using the selected policy. The following wizard will be presented for clearance generation.

**Figure 41.18. Generate Clearance**

One or more classifications can be selected from the **Security Classifications** pane to be added to the clearance. The **Security Categories** pane lists all the categories in the configured policy from which the user can select certain categories to be added to the clearance. The markings get updated on the **Markings** tab as and when the clearance is edited.

## 41.6.5 Access Control Checks

SPIF editor can be used to verify a security label against a security clearance and vice versa. To check access controls, select **Utilities** → **Access control checks....** menu. A dialog for performing these checks will be presented.

**Figure 41.19. Access Control Checks**

Select **Label Checker** to check access control of a label against a clearance or a catalog of clearances. Select **Clearance Checker** to check access control of a clearance against a label or a catalog of labels.

The label can be selected from an XML file by browsing the file system using the **Browse** button or picked up from a label catalog using the **Pick...** button. The **Pick...** button will offer the labels in the catalog in the form of a dropdown list as shown below.

**Figure 41.20. Pick Label from Catalog**

The label can be checked against a clearance or a clearance catalog that can be selected from the file system using the **Browse** button.



Once a label and clearance/catalog has been selected, the result of access control checks will be displayed in the bottom pane.

**Figure 41.21. Access Control Check of Label with Clearance Catalog**

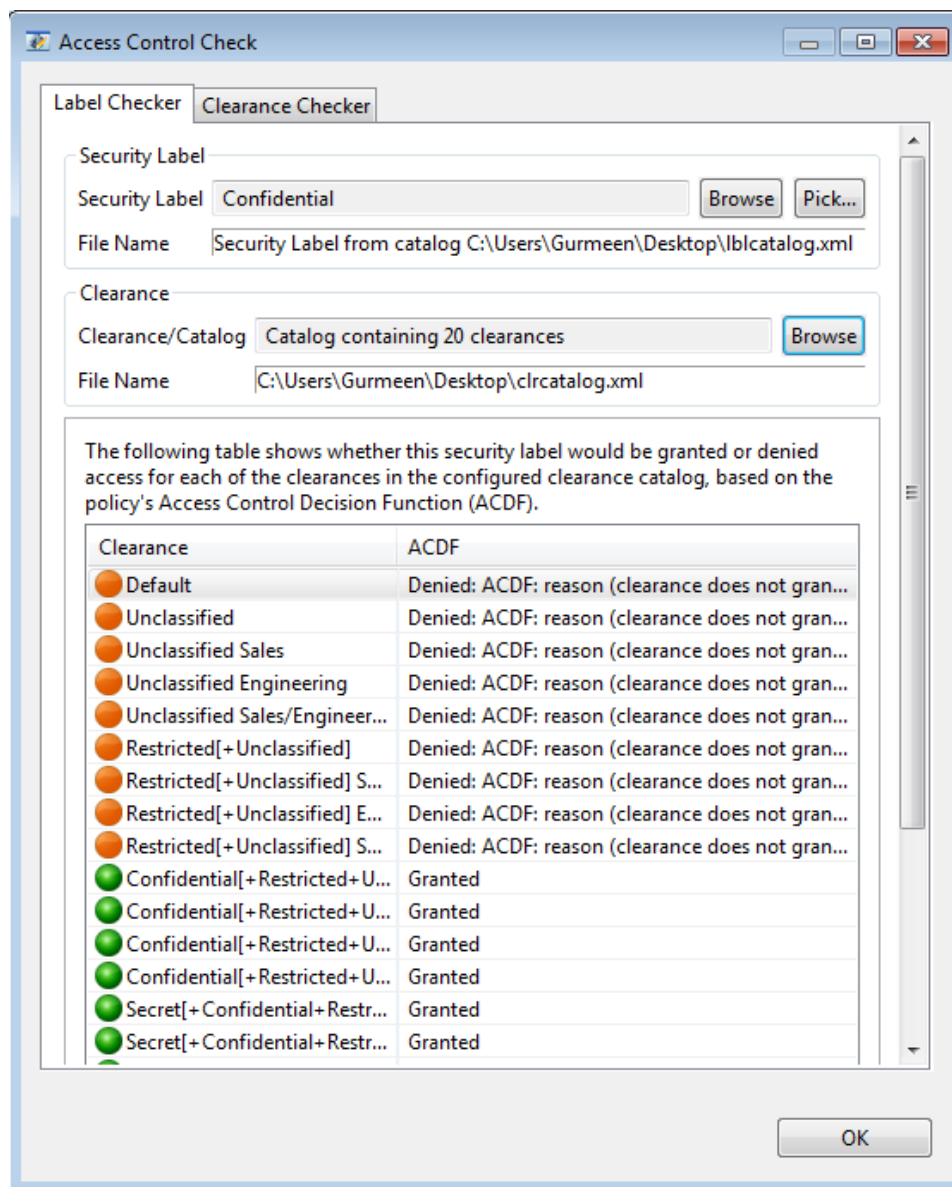
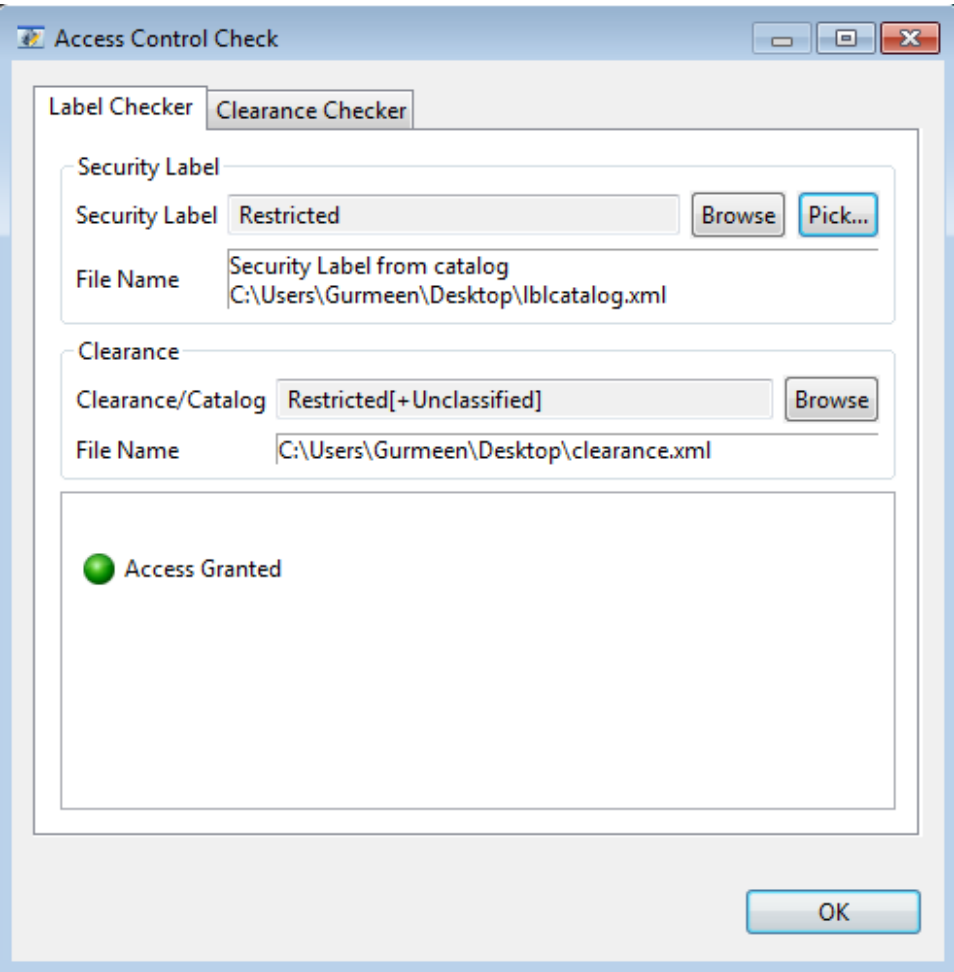
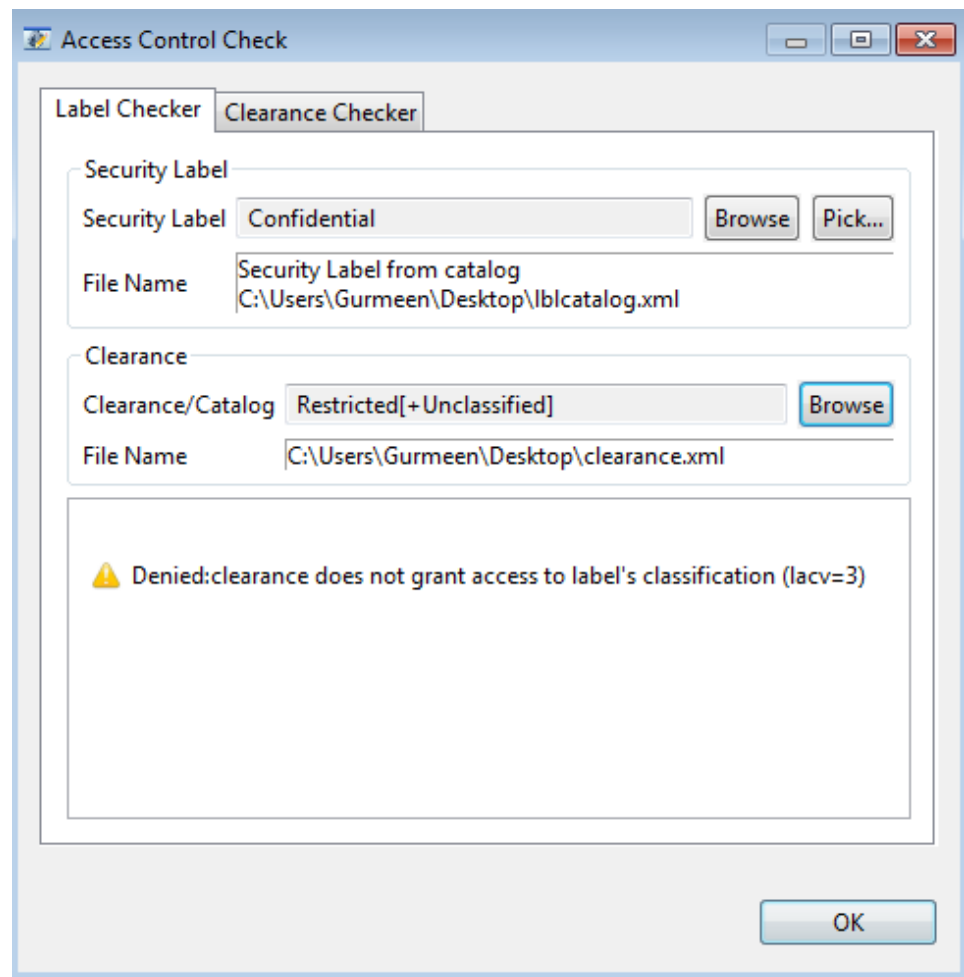


Figure 41.22. Access Control Check of Label - Access Granted



**Figure 41.23. Access Control Check of Label - Access Denied**

Follow the above steps in a similar way to check a clearance against a label or a catalog of labels by selecting the **Clearance Checker** tab.

# Chapter 42 Alert Daemon

This chapter describes the Alert Daemon, which can be used to monitor multiple M-Switch servers and generate Alerts in response to the occurrence of specific conditions.

---

## 42.1 Overview

M-Switch provides data on the state of the Queue Manager as a whole, individual channels, and subordinate Peer MTAs via the SOM protocol. The Alert Daemon is a SOM client application which retrieves information about these objects and applies filters to the values obtained. If a filter matches, an Event can be generated. For example, the number of messages queued on a specific channel can be checked at regular intervals, with an Event being generated if the number queued exceeds a configurable limit.

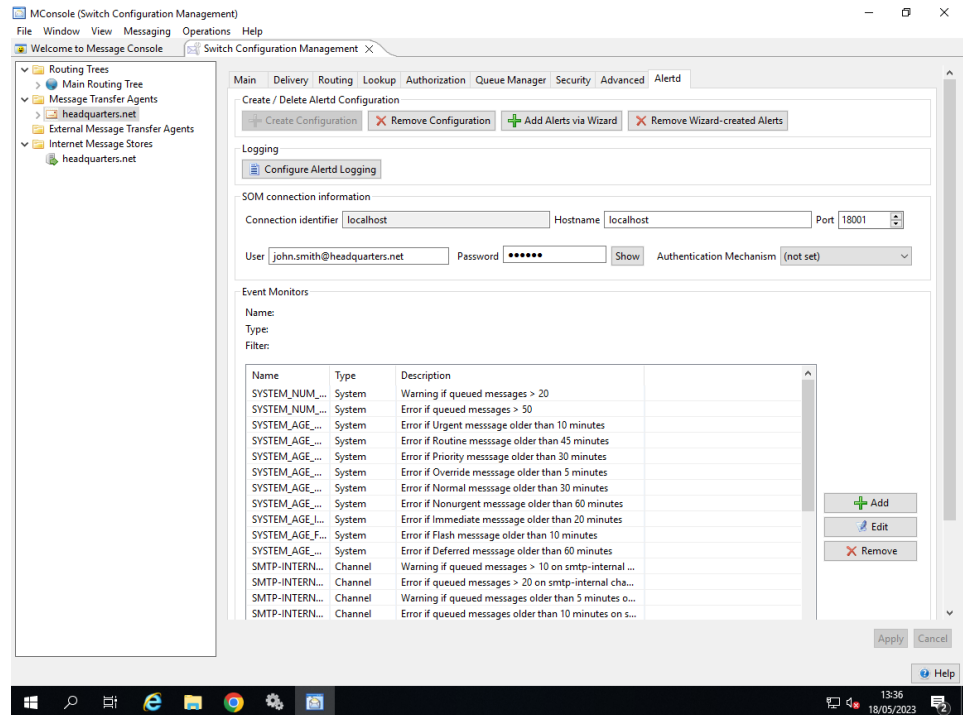
The Alert Daemon is configured using an XML file. This allows the specification of a set of data points (referred to as Monitors) to be monitored at one or more Queue Managers, together with the conditions under which an Event should be generated. Three levels of Event (Error, Warning and Notice level) are supported. Events are reported using the standard Isode event logging mechanisms, configured via their own XML file.

In a Directory-based Messaging Configuration, the XML data used to configure the Alert Daemon, along with the Alert Daemon's logging configuration, can be held as attributes of an MTA's configuration. This allows a built-in editor within MConsole to be used to create and edit the Alert Daemon's configuration, along with an integrated editor for the Alert Daemon's logging configuration. This XML data is then downloaded by the Queue Manager and written into the Alert Daemon's `(ETCDIR)/alertd.xml` and `(ETCDIR)/alertdlogging.xml` files. In this situation, the Alert Daemon only monitors the Queue Manager with which it is associated.

---

## 42.2 Alert Daemon Configuration Editor

This editor provides a wizard which allows a typical configuration to be simply set up for your MTA. This can then be fine-tuned using the editing facilities. It also allows the logging configuration for the Alert Daemon to be created and edited (using the standard Logging Configuration Editor).

**Figure 42.1. Alert Daemon Configuration Editor**

## 42.2.1 Alert Wizard

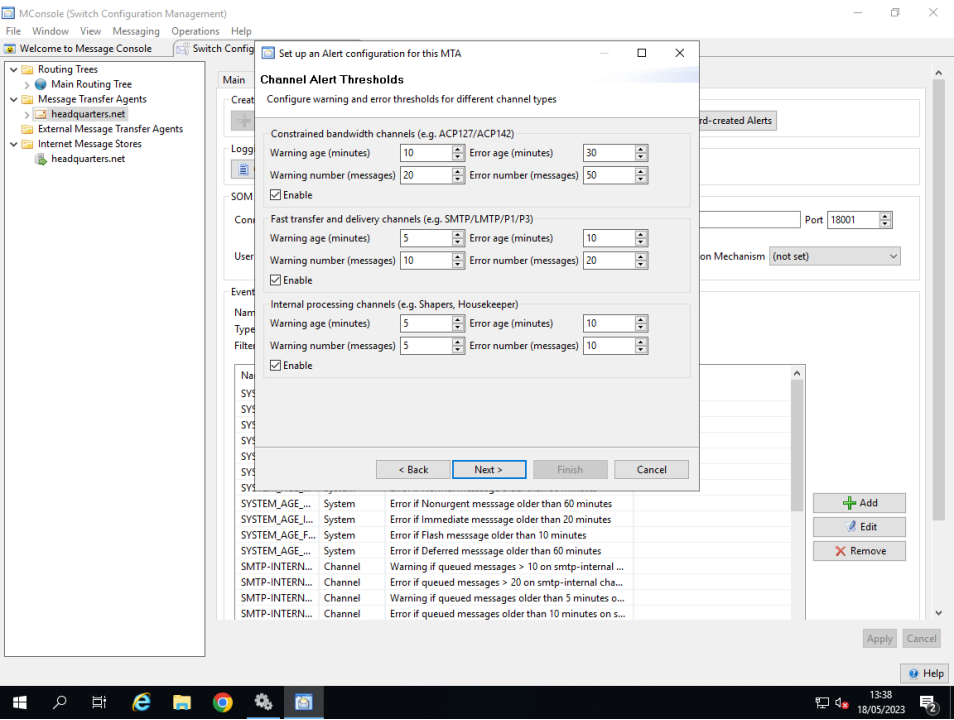
The Alert Wizard allows the quick creation of a set of Monitors with reasonable filters for different types of Messaging Configuration. Each Monitor is configured to create a unique Alert.

After selecting the type of configuration (General Purpose, Aviation or Military), the next page allows you to specify Warning and Error age and volume values for three different categories of channel:

- Constrained bandwidth channels such as ACP127, where reasonably large queues with messages remaining queued for many minutes can be expected under normal operation.
- High bandwidth transfer and delivery channels such as SMTP, LMTP and X.400 P1 and P3, where messages should not remain queued for very long.
- Internal channels such as Shapers or the Housekeeper channel, which should process their messages very quickly.

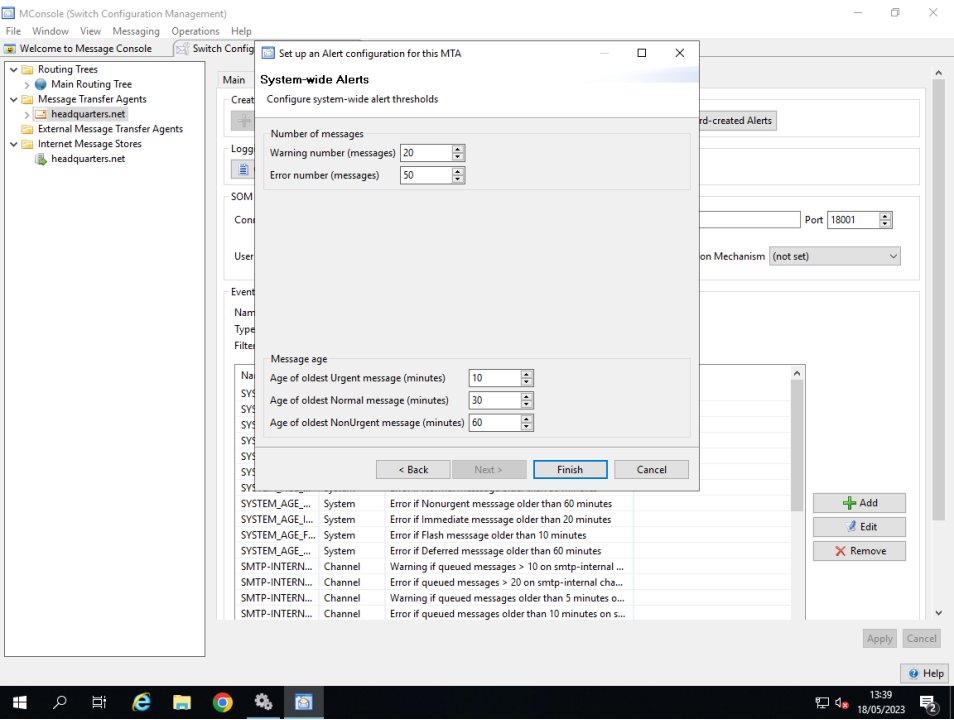
You can modify the default values presented for each category, and disable creation of specific sets of Monitors completely if required.

Figure 42.2. Alert Daemon Wizard: Channel Alert Thresholds



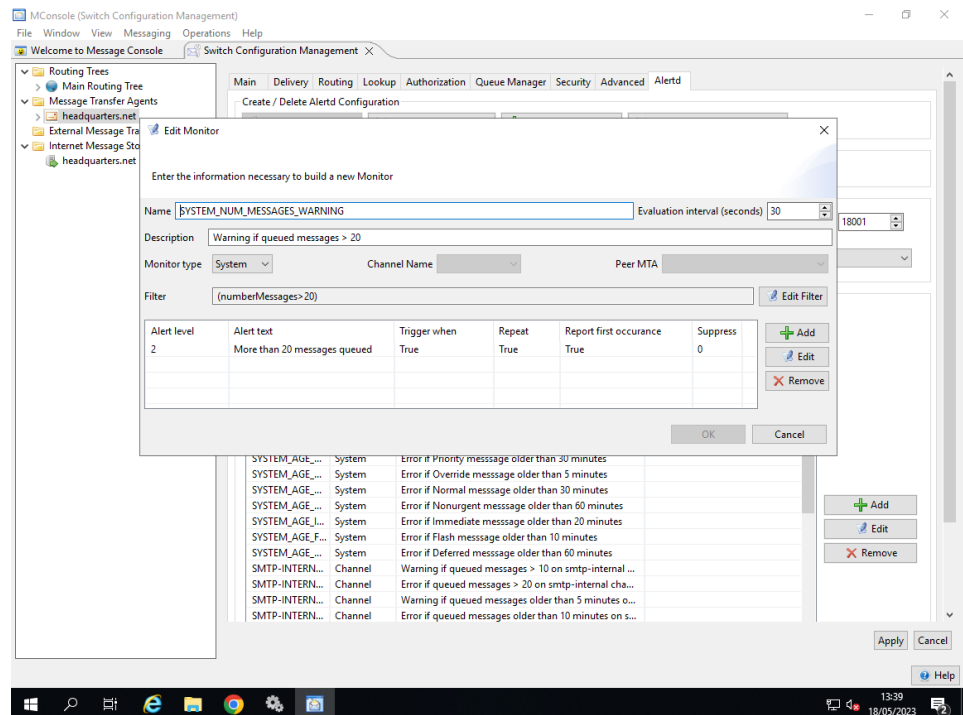
The next wizard page allows the configuration of system-wide Alerts.

Figure 42.3. Alert Daemon Wizard: System-wide Alert Thresholds



### 42.2.2 Monitor Editor

Once a set of Monitors has been created using the Wizard, individual Monitors can be edited to fine-tune the Filter used by the Monitor, to change the description, the interval at which the Monitor is evaluated or the Alert which is generated when the Filter evaluates to "true".

**Figure 42.4. Alert Daemon: Monitor Editor**

You can also create arbitrary Monitors using the same editor. The list of data items which are available for use in Filters is shown in [Section 42.4, “Data Items”](#).

### 42.2.3 Saving Configuration Changes

When new Monitors have been created, or existing Monitors have been edited or deleted, pressing the **Apply** button will cause the changes to the `Alertd` configuration to be saved back to the MTA's entry in the Directory, and will also trigger the Queue Manager (if running) to download this information into the `(ETCDIR)/alertd.xml` file. If the Alert Daemon is running, it will then re-read this configuration file and update its set of Monitors accordingly, without needing to be restarted.

## 42.3 File-based Configuration

A sample Alert Daemon configuration file is included in the the release, in `(SHAREDIR)`.

There are three distinct sections to the Alert Daemon configuration file:

- The standard `servpass` section allows the configuration of the Service Password service and verifier information. This allows SOM passwords elsewhere in the configuration file to be encrypted using the `spasscrypt` tool: these passwords are then decrypted at the point of use. For more information on the Service Password mechanism, see [Section 42.7, “Use of ServPass”](#)
- Multiple `som` sections may be present. Each one defines a SOM connection to a different Queue Manager. Each section requires a unique `key` field which provides the reference point for the connection. The standard SOM connection information of host, port, userid, password and SASL mechanism then need to be specified.
- Multiple `monitor` statements define the data items to be examined and the events to be generated. Each `monitor` statement includes the following information:

- A `name` attribute. This will be included in the text of any event generated by this statement.
- An `evaluate` attribute which defines the interval (in seconds) at which the `monitor` is evaluated.
- A `som` attribute which identifies the SOM connection which is to be used by this `monitor`.
- An optional `channel` attribute which specifies the channel to which the `monitor` applies. If this is not present, the test is assumed to apply to the Queue Manager as a whole.
- An optional `mta` attribute which specifies the Peer MTA to which the `monitor` applies. A `channel` attribute must be present as well in this case. The value of this attribute should be the full Distinguished Name of the Peer MTA within the Messaging Configuration.
- A `description` attribute. This is not used by the Alert Daemon itself.
- A `filter` clause. This defines the set of tests which the `monitor` statement will apply. A `filter` contains:
  - A `type` attribute which specifies the filter type - this can have values `item`, `or`, and `or not`.
  - One or more `filteritem` statements containing:
    - A `key` field which specifies the data item from the MTA, channel or Peer MTA which is to be tested. A list of available keys is given below.
    - An `operator` field which specifies the operator to be applied. This can have values `gt`, `lt` or `eq`, corresponding to "greater than", "less than" and "equals".
    - A `value` field which specifies the value to be used in the test. For "time" syntax data items, the `value` field specifies a number of seconds (e.g. so that you can test `oldestmessage gt 300`).
- One or more `event` clauses. These define the Events which are to generated when the `filter` clause produces a result (either `true` or `false`), and contain:
  - A `level` field, with values `error`, `warning` or `notice`.
  - A `filtervalue` field, which selects whether the Event is triggered when the Filter evaluates to `true` or `false`. `notice`.
  - A `repeat` field, which controls whether the Event is generated every time that the Filter evaluates to the specified value or only on the first occurrence.
  - A `reportfirst` field, which controls whether the Event is generated on the first time that the Filter evaluates to the specified value. This is useful when specifying an Event which reports that an error condition has cleared.
  - The `text` to be included in the Event. The channel and Peer MTA name (if set for this Monitor) will be included in the Event text automatically.

---

## 42.4 Data Items

This table lists the data items which are available for use in filters, together with their syntax and availability.



**Table 42.1. Available Data Items**

Name	Syntax	Availability	Notes
addrIn	Integer	All	Number of inbound addresses processed
addrOut	Integer	All	Number of outbound addresses processed
chanRate	Integer	Qmgr	Average number of blocked channels
connecting	Integer	Mta	True/false
currChans	Integer	Qmgr	Number of current running channels
diverted	Integer	Mta	True/false
enabled	Integer	Channel/Mta	True/false
errorCount	Integer	Channel/Mta	Count of errors
failedAssoc	Integer	Qmgr	Number of failed inbound associations
failedConv	Integer	Qmgr	Number of failed message conversions
failedOutAsoc	Integer	Channel/Mta	Number of failed outbound associations
inAssoc	Integer	All	Current number of inbound associations
inenabled	Integer	Channel/Mta	Enabled for inbound connections
loopsDetected	Integer	All	Number of loops detected
maxChans	Integer	Qmgr	Maximum number of channels which can run
maxinconn	Integer	Mta	Maximum number of inbound connections allowed
maxoutconn	Integer	Mta	Maximum number of outbound connections allowed
maxprocs	Integer	Channel	Number of channel processes allowed
messagesIn	Integer	All	Number of inbound messages processed
messagesOut	Integer	All	Number of outbound messages processed
msgsInPerSec	Integer	Qmgr	Rate of messages inbound per second
msgsOutPerSec	Integer	Qmgr	Rate of messages outbound per second
numberActiveProcesses	Integer	Qmgr	Number of active processes
numberAddresses	Integer	All	Number of recipients queued

Name	Syntax	Availability	Notes
numberReports	Integer	All	Number of reports queued
numberMessages	Integer	All	Number of messages queued
numberMtas	Integer	Channel	Number of Peer MTAs
numberProcesses	Integer	Channel	Number of channel processes running
opsPerSec	Integer	Qmgr	Operations being processed per second
outAssoc	Integer	All	Current number of outbound associations
receivedVol	Integer	All	Volume of received messages in kB
refuseStatus	Integer	Channel/MTA	Status for last refusal
rejectedAssoc	Integer	All	Number of rejected associations
rejectedMsg	Integer	Channel/MTA	Number of rejected messages
rejectstatus	Integer	Channel/MTA	Status for last message rejection
runnableChans	Integer	Qmgr	Number of runnable channels
somversion	Integer	Qmgr	SOM protocol version
status	Integer	Qmgr	1=up, 2=down, 3=halted, 4-congested, 5=restarting, 6=quiescing
successConv	Integer	Channel/Mta	Number of successful conversions
totalInAssoc	Integer	All	Total number of inbound associations
totalOutAssoc	Integer	All	Total number of outbound associations
transmittedVol	Integer	All	Total volume of messages transmitted (KB)
volumeMessages	Integer	All	Volume of messages queued (KB)
archiveFree	Integer	Qmgr	Percentage of free space on archive filesystem
loggingFree	Integer	Qmgr	Percentage of free space on logging filesystem
queueFree	Integer	Qmgr	Percentage of free space on queue filesystem

Name	Syntax	Availability	Notes
oldest-override	Time	Channel/MTA	Age of oldest message with priority "override"
oldest-urgent	Time	Channel/MTA	Age of oldest message with priority "urgent"
oldest-immediate	Time	Channel/MTA	Age of oldest message with priority "immediate"
oldest-normal	Time	Channel/MTA	Age of oldest message with priority "normal"
oldest-routine	Time	Channel/MTA	Age of oldest message with priority "routine"
oldest-nonurgent	Time	Channel/MTA	Age of oldest message with priority "non-urgent"
oldest-bulk	Time	Channel/MTA	Age of oldest message with priority "bulk"
oldest-bulk	Time	Channel/MTA	Age of oldest message with priority "junk"
boottime	Time	Qmgr	Time at which MTA was started
creation	Time	Channel/MTA	Creation time of channel or MTA
delayedUntil	Time	Channel/MTA	The time at which the channel or MTA will next be processed
lastAttempt	Time	Channel/MTA	The time at which the channel or MTA was last attempted
lastSuccess	Time	Channel/MTA	The time at which the channel or MTA last succeeded
lastinbound	Time	All	The time of the last inbound message
lastoutbound	Time	All	The time of the last outbound message

---

## 42.5 Starting and stopping the Alert Daemon

On Windows this is done using a Windows service using M-Switch Service Configuration.

On Unix you need to edit `/etc/isode/pp.rc` to uncomment the ALERTD lines as described in [Section 33.1.3, “Configuring the Messaging startup script”](#)

## 42.6 Example Alert Daemon configuration file

A sample configuration is shown below. It is set up to generate Events when:

- There are more than 100 messages queued overall.
- There is an urgent or immediate priority message which is older than 10 minutes queued on the x400p1 channel.
- There is no connection open from the x400p1 channel to a specific Peer MTA.

```
<?xml version="1.0" standalone="yes"?>
<alertd>

  <!-- Configuration of servpass verifier -->
  <servpass:info service="isode.alertd" verifier="AlsuJCfnoNUo"/>

  <!-- SOM connection to localhost.localdomain -->
  <som key="localhost">
    <host>localhost.localdomain</host>
    <port>18001</port>
    <user>mtaadmin@mydomain.com</user>
    <password servpass:encrypt="true">
      {spcrypt3}p3rQ3KsKt4vLnLSwsWnu5MSpGeQ4/Yud</password>
    <saslmech>CRAM-MD5</saslmech>
  </som>

  <!-- Monitor for urgent messages on x400p1 older than 600s -->
  <monitor name="URGENT1" evaluate="10"
    som="localhost" channel="x400p1">
    <!-- Description string is useful when editing config -->
    <description>Urgent messages on x400p1
    channel older than 600s</description>
    <!-- The filter which defines when the monitor is true -->
    <filter type="or">
      <filteritem key="oldest-override" operator="gt" value="600"/>
      <filteritem key="oldest-urgent" operator="gt" value="600"/>
    </filter>
    <!-- Event to generate when filter evaluates to true -->
    <event level="warning" filtervalue="true" repeat="true">
      <text>Urgent messages older than 600 seconds</text>
    </event>
    <!-- Event to generate when filter evaluates to false -->
    <event level="notice" filtervalue="false" repeat="false"
      reportfirst="false">
      <text>Backlog of urgent messages now cleared</text>
    </event>
  </monitor>

  <!-- Monitor for number of messages queued overall -->
  <monitor name="QUEUE SIZE" evaluate="10" som="localhost">
    <!-- Description string is useful when editing config -->
    <description>Number of messages queued on system</description>
    <!-- The filter which defines when the monitor is true -->
    <filter type="item">
      <filteritem key="numberMessages" operator="gt" value="100"/>
    </filter>
    <!-- Event to generate when filter evaluates to true -->
    <event level="warning" filtervalue="true" repeat="true">
      <text>More than 100 messages queued</text>
```

```

</event>
<!-- Event to generate when filter evaluates to false -->
<event level="notice" filtervalue="false" repeat="false"
reportfirst="false">
  <text>Backlog of messages now cleared</text>
</event>
</monitor>

<!-- This monitor checks for connection loss to a particular MTA
on the local MTA's x400p1 channel -->
<monitor name="CONNECTIONLOSS" evaluate="30" som="localhost"
channel="x400p1"
mta="cn=x400p1,cn=jpamhs,cn=Messaging Configuration,c=gb">
  <!-- Description string is useful when editing config -->
  <description>Lost connection to Japan</description>
  <!-- The filter which defines when the monitor is true -->
  <filter type="item">
    <filteritem key="outassoc" operator="lt" value="1"/>
  </filter>
  <!-- Event to generate when filter evaluates to true -->
  <event level="warning" filtervalue="true" repeat="true">
    <text>No connection open</text>
  </event>
  <!-- Event to generate when filter evaluates to false -->
  <event level="notice" filtervalue="false" repeat="false"
reportfirst="false">
    <text>Connection reopened</text>
  </event>
</monitor>

</alertd>

```

---

## 42.7 Use of ServPass

The ServPass facility allows passwords in alertd.xml to be obfuscated to prevent them being read, and then decrypted at the point of use. GUI support for the use of ServPass by the MTA and Message Store is provided in MConsole (and described elsewhere in this manual), but it is currently necessary to use command line tools when setting up the alertd.xml file. The steps to follow are described below.

- Create the Service Key for the isode.alertd service. You will need to perform this operation as root on Unix platforms.

```

[root@diabolo /]# spassmgt set isode.alertd root
Passphrase: 1234567890abcABC
Re-enter: 1234567890abcABC

```

The final parameter in the command line above specifies the userid under which the isode.alertd daemon will run on Unix. Choose a memorable passphrase, which must include a mixture of upper- and lower-case letters and digits.

- You then need to obtain a verifier for the service:

```
[root@diabolo /]# spassmgt get isode.alertd
AlsuJCfnoNUo
```

- Edit this verifier into your alertd.xml file (the example file shown above already contains this):

```
<servpass:info service="isode.alertd" verifier="AlsuJCfnoNUo"/>
```

- Configure the SOM password(s) in the alertd.xml file in the clear:

```
<password servpass:encrypt="true">secret</password>
```

- Encrypt the password(s) in the alertd.xml file:

```
[root@diabolo /]# spasscrypt -e -s isode.alertd -x /etc/isode/alertd.xml
```

# Appendix A Messaging APIs

The Isode M-Switch product includes a number of programmatic APIs which are formally supported as part of the product set. These are for the most part documented using automatically generated documentation (e.g. using javadoc and doxygen).

The documentation is available on the Isode website <http://www.isode.com/products/x400-gateway-api.html>

More detailed documentation along with a FAQ is available in the *M-Switch Advanced Administration Guide*.

A brief overview is provided here.

The main APIs are:

- Switch Operations and Management (SOM) API. This is for use by management clients. There are a number of features of the switch which can be managed through this API which is primarily aimed at the writer of systems integrator tools who wishes to embed management of the Isode M-Switch into an integrated management system.
- X.400 Messaging APIs. This is for use primarily by those who wish to write client applications which submit/receive messages to/from M-Switch.

---

## A.1 SOM API

SOM servers available are

- M-Switch Queue Manager
- M-Store

The main components of the SOM API for M-Switch Queue Manager are:

- Event Viewing: an API allowing the viewing of events logged by M-Switch of its operations
- Queue Viewing: an API providing access to information about the MTA's queues
- Queue Manager Operations: an API allowing operations such as restarting the qmgr, or clearing delays.
- Message Resubmission: an API allowing messages in quarantine or archive to be resubmitted.
- Message Viewing: an API allowing messages in an M-Switch repository to be viewed. Available repositories are: the live queue; the archive; the quarantine.
- X.400 P1 ping: an API which causes the x400p1 channel on the MTA to check P1 connectivity with another MTA

The main components of the SOM API for M-Store are:

- Messages: an API for examining messages in M-Store
- Connections: an API for examining connections in M-Store
- Mailboxes: an API for examining mailboxes in M-Store

The SOM API is described at <http://www.isode.com/Documentation/messaging-api/som-c-api/index.html>

---

## A.2 X.400 API

The main component of this API is the Isode P3/P7 Client API. This API provides a relatively simple abstraction of X.400 P3 or X.400 P7 to enable Messaging User Agents to be easily written by those without a high level of familiarity with the rich and extensive set of X.400 standards.

An overview of this API is at <http://www.isode.com/products/x400-client-api.html>.

---

## A.3 Isode Gateway APIs

The gateway APIs allow applications to be written which act as a gateway to a different messaging system, particularly legacy messaging systems.

See <http://www.isode.com/products/x400-gateway-api.html> for an overview of the Isode Gateways.

There are two different Isode Gateway APIs:

- Isode Gateway API: this API provides a similar model of interface as the P3/P7 Client API, providing a relatively simple abstraction of the X.400 transfer protocol P1.
- Open Group X.400 API: this API provides a similar set of features to the Isode Gateway API, but implements the API as specified by the Open Group.

---

## A.4 API Model

With the exception of the Open Group APIs, the Isode APIs are provided in a simple style using integers, strings and objects. Strings and integers and sub objects can be extracted from objects or inserted into objects using the API calls.

---

## A.5 Language Bindings

All APIs have C language bindings.

The P3/P7 APIs and the SOM API also have Java and Tcl bindings.



# Appendix B Presentation Addresses

These are the fundamental mechanism used by applications to address other application entities. After reading this chapter you should know how to represent any OSI PA in a string format.

---

## B.1 Introduction

There are several reasons for needing to be familiar with PA string formats. First, this string format is sometimes needed in configuration files. Second, occasionally they are required as run-time arguments to OSI applications. Third, this format is a mechanism that allows people to exchange PAs in a medium such as electronic mail messages. Fourth, the string format of network address (a component of PAs) is used to define network communities; we will see how to do this in [Section B.2.2.3, “Network addresses as macros”](#).

You should be aware of the limited aims of this chapter. The intention is to present enough information on addressing that you can configure your system. It is not intended to be a general tutorial on addressing. While the details of the presentation, session and transport selectors are fairly straightforward, the details of network addressing are less so. For further information on network addressing, refer to *Information processing systems – Data communications – Network service definition Addendum 2: Network layer addressing*, ISO 8348, 1987. Extensions to OSI network addressing, to allow non-OSI addresses to be represented within OSI network addresses, are described in RFC 1277.

The string representation of PAs is fully, if tersely, described in RFC 1278. This document also describes additional string formats for encapsulated network addresses. The following discussion is intended to cover string representations of PAs in a more descriptive manner. Examples are given for all the formats

---

## B.2 Presentation Address format

The string representation of a Presentation Address uses the following syntax:

```
[[[psel / ]ssel / ]tsel / ]na1[|na2 ]...[|na8 ]
```

The brackets ([ ]) indicate the enclosed element is optional. Thus, the only mandatory component of a Presentation Address is a single network address. If it is necessary to specify a presentation address when no network address is appropriate, the empty network address can be specified by using the form:

NS+

### B.2.1 The selectors

The Presentation Selector (PSEL), Session Selector (SSEL) and Transport Selector (TSEL) are all optional components of a PA.

The selectors PSEL, SSEL and TSEL are octet strings of up to 64, 16 and 32 octets respectively.

They may be specified in any of the following ways:

`"IA5-string"`

an IA5 (ASCII) string enclosed in double quotes. The length of the selector is the number of characters in the string. Note that the double quote characters may need to be escaped, by prefixing them with a backslash character (\), in some configuration files. See the descriptions of the *isoservices* and *isoentities* files.

`#digitstring`

a string of decimal digits preceded by #; only numbers 0-65535 are legal for this method and the length of the resulting selector is two octets.

`'hexstring'H`

a string of an even number of hexadecimal digits enclosed in single quotes and followed by H. The length of the selector is half the number of digits in the string. This is the most general form and can be used to represent any selector value.

Examples of these formats are:

```
"ACCNTS1"
#43775
'001aff'H
```

If a PSEL is supplied, then if SSEL or TSEL is null (not used), its '/' must be included. Similarly, if SSEL is supplied (but no PSEL) and TSEL is null, its / must be included. For example, for

```
PSEL = "ACCNTS1"
SSEL null
TSEL = '001aff'H
```

enter:

```
"ACCNTS1"// '001aff'H/network-addresses...
```

and for

```
PSEL null
SSEL = "BUY-TRANS"
TSEL null
```

enter:

```
"BUY-TRANS"//network-addresses...
```

For all selectors, an absent selector has the same meaning as one of length zero. Thus the following would be encoded the same as the previous example:

```
"BUY-TRANS" / " /network-addresses...
```

## B.2.2 Network addresses

In the string representation of PAs, multiple network addresses must be separated by the vertical bar character (|).

A Presentation Address can include from one to eight network addresses.

---

**Note:** A network address is sometimes referred to as a *Network Service Access Point* (NSAP) address. In this manual, only the term *network address* is used.

---

A network address can:

- Be used directly by the OSI network layer, for the Connection Oriented Network Service (CONS) or the Connectionless Network Service (CLNS). These addresses are restricted to 20 octets.
- Encapsulate the connection information for the case where OSI transport is being used, but the network service is not an OSI network service.
- Encapsulate connection information used for protocols other than OSI, for example, LDAP.

### B.2.2.1 Unstructured network addresses

This is the simplest form of representation but the least manageable. This form can be used to represent any network address. In practice it is not often used for writing network addresses by the applications using these services, but it can be used to define macros (see later in this chapter). It is often used in the documentation of OSI transport services and other OSI products. Using this format, a network address is written as follows:

```
NS+network-address
```

where:

NS+

is a literal

network-address

is the entire network address specified as a string of a hexadecimal digits in preferred binary form. Note that significant technical complexity is embodied by this term which is mostly transparent. See *Information processing systems – Data communications – Network service definition Addendum 2: Network layer addressing*, ISO 8348, 1987 for details.

An example of this form is:

```
NS+491234567890
```

### B.2.2.2 Structured network addresses

The first octet of a network address is termed the Authority and Format Identifier (AFI). As it suggests, this octet takes a value that indicates a registration authority and the format of (at least part of) the remainder of the network address. In practical terms, different types of network address (such as TCP Internet and CONS) use different AFIs. This section shows how network addresses can be structured for each of the network types.

#### B.2.2.2.1 TCP Internet

TCP addresses are used to hold TCP connection information for ISO Transport over TCP/IP (ITOT) and for other TCP based protocols when the host is defined by an IPv4 address. This scheme is described in RFC 1277. The structure of TCP Internet addresses is as follows:

```
TELEX+value+RFC-1006+prefix+domainstring[+port]
```

where:

TELEX

is a literal.

value

is a fixed eight digit telex number.

RFC-1006

is a literal

prefix

is a two digit code (value 03 to 99). For ITOT the value is 03. Other values indicate other protocols

domainstring

is a sequence of octets specifying an IP address or a domain name. If a domain name is used, then when the string format is interpreted, the corresponding IPv4 addresses are looked up in the name service. If there are multiple names, then multiple network addresses are generated.

port

is a string of decimal digits specifying the TCP port number. The default for ITOT is 102.

Examples are:

```
TELEX+00728722+RFC-1006+03+192.32.255.01+4321
TELEX+00728722+RFC-1006+03+foo.bar.co.uk
```

#### B.2.2.2.2 X.25 addresses

There are two ways of representing X.25 SNPA addresses. The first method can only be used for X.121 addresses, with no Protocol Identifier (PID) or Call User Data Field (CUDF) fields.

The structure of the address format is:

```
x121+dte
```

where:

x121

is a literal.

dte

is a string of decimal digits specifying the DTE address.

A more general way of specifying X.25 addresses is to use the TELEX AFI. This should be used when the DTE is not an X.121 address connected to the international Packet Switched Data Network (such as a JANET or IXI X.25 address), and/or where a PID or CUDF is required. The formats are as follows:

```
TELEX+value+X.25(80)+prefix+dte[+CUDF+cudf]
TELEX+value+X.25(80)+prefix+dte[+PID+pid]
```

where:

TELEX

is a literal

`value`

is a fixed eight digit telex number. This value is fixed for the pre-defined X.25 networks, and should be set to your organization's international telex number when defining a new community.

`X.25(80)`

is a literal.

`prefix`

is a two digit code (value 01 to 02)

`dte`

is a string of decimal digits specifying the DTE address.

`CUDF`

is a literal.

`cudf`

is a string of hexadecimal digits giving the value of the Call User Data Field

`PID`

is a literal.

`pid`

is a string of hexadecimal digits giving the value of the protocol identifier.

Some examples of the form are as follows:

```
X121+123456654321
TELEX+00728722+X.25(80)+02+123456654321+CUDF+00FF
```

### B.2.2.2.3 Encapsulated URL addresses

This form of network address holds a URL for protocol and connection information. The network address itself can therefore hold hostnames or IPv6 addresses, which cannot be held in other network address formats. Hostnames are resolved to addresses when the address is used by the relevant protocol. The network address is defined by X.519 (2008) 11.4. The string format is defined here. Note that the network addresses defined in this section cannot be used for normal OSI networking.

The format of the address is:

```
URI+idi+URL+url
```

where:

`URI`

is a literal.

`idi`

is the four digits 0000 for OSI transport use (ITOT), and 0001 for non-OSI access points, e.g. LDAP.

`URL`

is a literal.

`url`

is a URL in standard format. The protocol for ITOT is 'itot'. The URL can use either a hostname or IP address (v4 or v6). A TCP port number can be specified. If the port number is not specified, then the default port number for the protocol is used (102 for ITOT).

Examples:

```
URI+0000+URL+itot://x400.example.com
URI+0000+URL+itot://[FEFF:1]:10102
URI+0001+URL+ldap://ldap.example.org
```

---

**Note:** The port number is preceded by a colon, not a plus sign.

---

There are macros to simplify the specification of these network addresses. For example, the first example can be specified by:

```
ITOT=x400.example.com
```

#### B.2.2.2.4 CONS/CLNS addresses

Any format of network addresses can be used for real OSI Connection Oriented Network Service (CONS) and ConnectionLess Network Service (CLNS) addresses but the formats above should be avoided. They are often written in the unstructured format, otherwise the formats below may be used. There are five different AFIs, which we can treat in three groups.

The format of network addresses with DCC and IDC AFIs is as follows:

```
DCC+reg-number+domainspecificpart
ICD+reg-number+domainspecificpart
```

where:

DCC

is a literal

ICD

is a literal.

reg-number

is the number assigned by the registration authority.

domainspecificpart

is the locally defined portion of the network address. It can take one of two formats:

- decimal format: where the string is the letter d followed by a string of decimal digits, e.g. d123456
- binary format: where the string is the letter x followed by a string of an even number of hexadecimal digits, e.g. x0aff.

The format of addresses with PSTN (Public Switched Telephone Network) and ISDN (Integrated Services Digital Network) AFIs is as follows:

```
PSTN+phonenumber+domainspecificpart
ISDN+isdnumber+domainspecificpart
```

where:

PSTN

is a literal

ISDN

is a literal

phonenumber

is an international telephone number

`isdnumber`

is an international ISDN number

`domainspecificpart`

is the locally defined portion of the network address: the format options for localpart are defined above.

The format of addresses with LOCAL AFI is as follows:

```
LOCAL++domainspecificpart
```

where:

`LOCAL`

is a literal. Note the two pluses. (This is required because the LOCAL AFI has a zero length IDI (Initial Domain Identifier)).

`domainspecificpart`

is the remainder of the network address: it can take the decimal and binary formats defined above, but also a string format where the string is the letter I followed by an IA5 string, e.g. ILONDON.

Examples of all these forms are given below, along with the corresponding unstructured format for each address:

DCC+826+dl1000012022	NS+3882611000012022
DCC+826+xl1f0000123a2	NS+39826f1f0000123a2
ICD+0003+xl1f0000123a2	NS+4700031f0000123a2
PSTN+441813329091+x0af2	NS+434418133290910af2
PSTN+11813329091+x0af2	NS+430118133290910af2
ISDN+441813329091+x0af2	NS+454418133290910af2
LOCAL++lLONDON	NS+504c4f4e444f4e

### B.2.2.3 Network addresses as macros

Both the structured and unstructured forms are often unwieldy representations of network addresses. A much simpler representation of network addresses can be achieved by using macros. Macros may be used to replace the leading octets in network addresses. Macros are defined in the *isomacros* file in the (*SHAREDIR*) directory.

The basic form of the *isomacros* file is:

```
macro-name address-prefix
```

where:

`macro-name`

is a string of characters.

`address-prefix`

is the leading portion of a network address. The address prefix may consist of an existing macro plus some additional characters. In other words macros can be used to define macros.

Some examples should explain the use of macros. An IP address of 128.86.8.56 with a port number of 17003 could be represented as below using the structured form of network address:

```
TELEX+00728722+RFC-1006+03+128.86.8.56+17003
```

Since we may wish to specify other network addresses in the same community, we can define a macro for the first part of the address. If the definition in the *isomacros* file is:

```
ipnet TELEX+00728722+RFC-1006+03+
```

then the above address could be expressed as:

```
ipnet=128.86.8.56+17003
```

A further macro could be specified to identify that host only. This could be specified by:

```
iphost ipnet=128.86.8.56+
```

The above address can then be expressed as:

```
iphost=17003
```

Another example, using CONS addresses, is based on entries in the *isomacros* file:

- The UK's CONS network addresses are defined using a DCC AFI with a DCC registration of 826.
- The JANET network's CONS addresses are all prefixed with a decimal value of 11000.
- The University of Leeds uses network addresses beginning with 120 within the JANET address space.
- A host at Leeds has the address 123 within the university's address space.

The host network address can thus be written directly as:

```
DCC+826+d11000120123
```

Alternatively we can define some macros as below:

```
UKNS DCC+826+d  
JanetNS UKNS=11000  
LeedsNS JanetNS=120
```

Then the above address can now be expressed in any of the following ways:

```
UKNS=11000120123  
JanetNS=120123  
LeedsNS=123
```