Internet Exchange 4

Message Store Administrator's Guide

INTERNATIONAL MESSAGING ASSOCIATES

Version 1.0
July 1999

ISBN: 962-8137-04-2

International Messaging Associates Limited
Hong Kong Computer Center, 20/F
54-62 Lockhart Rd.
Wanchai
HONG KONG

Tel:+1 (408) 481-9985
+1 (888) 562-3564
+852 2520-0300
+63 (2) 811-3999
Fax:+1 (888) 562-3561
+852 2648-5913
+63 (2) 811-3939
Email:*info@ima.com*
WWW:*http://www.ima.com*

IMA Philippines, Inc.
The Peak Tower, 15/F
107 Alfaro Street
Salcedo Village, Makati
PHILIPPINES

USA - Sunnyvale, California
USA - Message Center
Hong Kong
Philippines - Makati
USA
Hong Kong
Philippines - Makati

The following are copyrights of their respective companies or organizations:

Apache HTTP Server Copyright © 1995-1999 The Apache Group. All rights reserved.

McAfee VirusScan Copyright © 1998 Network Associates, Inc.

F-PROT Professional Copyright © 1999 Data Fellows Ltd. All rights reserved.

S|O|P|H|O|S Copyright © 1997-1999 Sophos Plc. All rights reserved.

cc:Mail is a trademark of cc:Mail Inc., a wholly owned subsidiary of Lotus Development Corporation, an IBM subsidiary.

Internet Exchange is a trademark of International Messaging Associates, Ltd.

Lotus Notes is a trademark of Lotus Development Corporation, an IBM subsidiary.

MS-DOS and MS-Windows are trademarks of © 1999 Microsoft Corporation. All rights reserved.

Portions of this product are based on software developed by the following universities/organizations:

CGI script Copyright © 1997 by Eugene Kim (eekim@eekim.com).

DiamondBase Copyright © 1993 by Darren Platt, Andrew Davison, Kevin Lentin of the Monash University Melbourne, Australia.

IMAPD Copyright © 1999 by Mark Crispin of the University of Washington (MRC@CAC.Washington.EDU).

LDAP support is based on software developed by the University of Michigan and its contributors.

SSL Copyright © 1995-1998 by Eric Young (eay@cryptsoft.com).

Table of Contents

Overview

THE MESSAGE STORE ADMINISTRATOR'S GUIDE

The Internet Exchange Message Store is a dedicated mail repository for storing, retrieving, and manipulating messages. It enables users to access their mailboxes via POP3 and/or IMAP4 capable clients such as Microsoft Outlook, Eudora, and others.

To provide the system administrator with a well-defined tool for using, configuring, and managing the Internet Exchange Message Store, this manual is organized as follows:

Part 1, *"System Architecture"*, provides an overview of the technologies used in the Internet Exchange Message Store together with a complete diagram showing how the module connects to the Internet Exchange Messaging Server. This section also provides an overview of the key features of the Message Store.

Part 2, *"Installation"*, describes in detail the steps that must be followed by the user in installing and setting up the Internet Exchange Message Store.

Part 3, *"Administration and Operation"*, describes the procedures for configuring the many features of the Internet Exchange Message Store and managing its operations.

Part 4, *"Troubleshooting Tools"*, describes the tools needed by the system administrator for troubleshooting purposes.

Part 5, *"Appendix"*, provides a detailed description of every technology and standard used by the Internet Exchange Message Store.

PART 1

System Architecture

Message Store Architecture

INTRODUCTION

Internet Exchange 4's IMAP4 Optimized Message Store is a dedicated mail repository for storing, retrieving, and manipulating messages. In addition, the Message Store enables users to access their mailboxes via POP3 and/or IMAP4 capable clients such as Microsoft Outlook and Eudora.

SYSTEM ARCHITECTURE

The Internet Exchange Message Store consists of the following structured databases:

- Users Database
- Shared Mailboxes Database
- Mailbox Database
- Message Status Database
- Message Envelope Database
- Message Body Database

The *Users Database* contains the name, password, and home directory of all valid users. It also contains a list of the names of the shared mailboxes available to a particular user. The *Shared Mailboxes Database* stores the names and home directories of all shared mailboxes in the system, while the *Mailbox Database* holds information regarding the status of different mailboxes, including the shared mailboxes. IMAP4-related attributes and RFC822 header information are stored in the *Message Status Database* and *Message Envelope Database*, respectively. The *Message Body Database* stores the body structure of all messages in a given mailbox. Access to the different databases in the Message Store is carried out via the Message Store API.

Aside from containing multiple internal databases, the Message Store also includes both the IMAP4 and POP3 servers. Each of these servers is capable of creating multiple threads to support simultaneous access of the Message Store and the retrieval of multiple messages.

Another Message Store module is MailSort, Internet Exchange's filtering utility. It delivers messages to specific mailboxes/folders and/or forward messages based on the mail filtering rules defined by the user. If no filtering rules are defined for the user, the messages are forwarded directly to his/her INBOX.

The Local Mail Delivery Agent is responsible for the delivery of messages from the Internet to the Message Store. The Local Mail Delivery Agent also communicates with the

MailSort engine to customize the delivery of messages based on user defined rules.

The key features of the **Internet Exchange 4** IMAP4 Optimized Message Store are:

- IMAP4 support
- POP3 support
- MailSort filtering utility
- Support for shared mailboxes

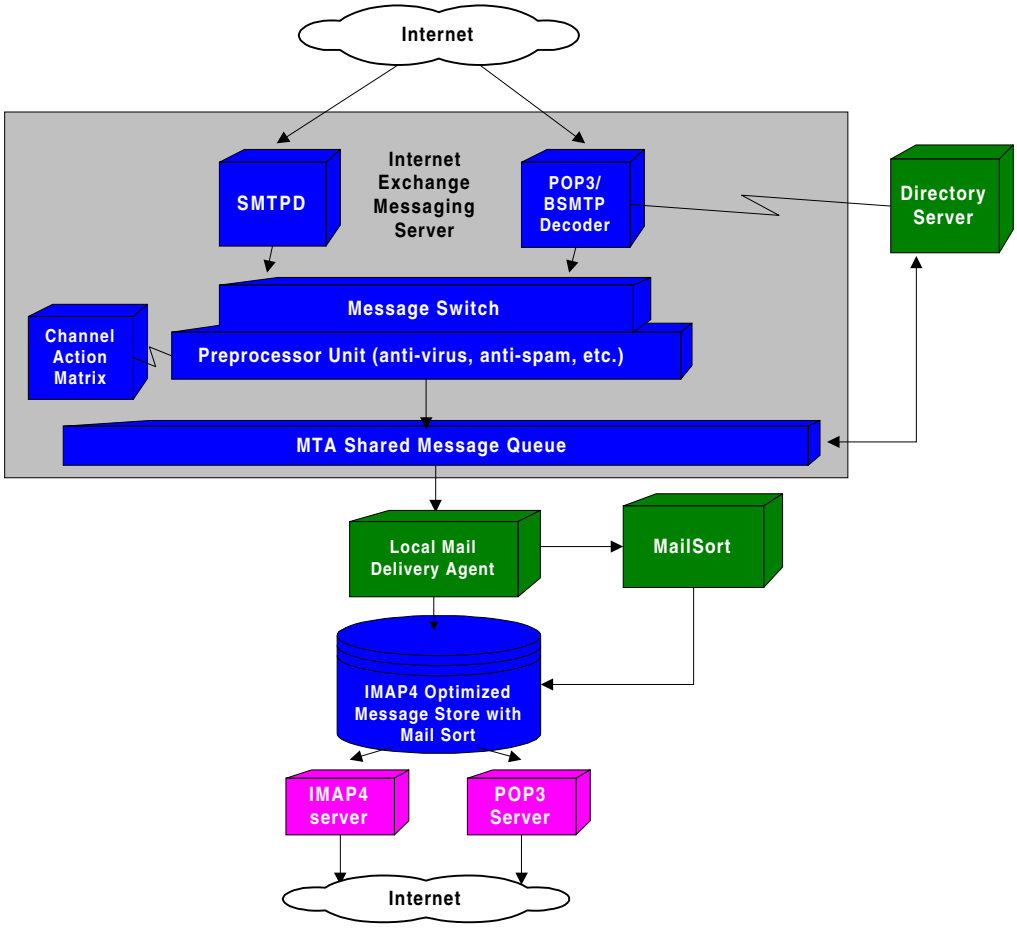


Figure 1. IMAP4 Optimized Message Store

System Components

MESSAGE STORE DATABASES

Internet Exchange 4's IMAP4 Optimized Message Store consists of the following databases. Access to these databases is carried out via the Message Store API.

Users Database
The Message Store's Users Database contains the name, password, and home directory of all valid users. It also contains a list of the names of the shared mailboxes available to a particular user.

Shared Mailboxes Database
The Shared Mailboxes Database stores the names and home directories of all shared mailboxes in the system.

Mailbox Database
The Message Store's Mailbox Database holds information regarding the status of different mailboxes, including the shared mailboxes.

Message Status Database
The Message Store's Message Status Database contains all the IMAP4-related attributes.

Message Envelope Database
The Message Envelope Database contains RFC822 header information.

Message Body Database
The Message Body Database stores the body structure of all messages in a given mailbox.

POP3 SERVER

Although IMAP4 access is more user friendly and extensive than POP3 due to the former's ability to handle remote folders, many email clients still use the latter. **Internet Exchange 4's** POP3 Server provides POP3 capable clients with a means for accessing their mailboxes. With POP3, users retrieve messages from the POP3 server and store them in a local hard disk so they can be read in an off-line or disconnected state.

While the POP3 Server is running, it listens on port 110 for a connection request from a POP3 client. Once such a request is received, the POP3 Server creates a thread that will handle further client transactions. Upon start-up, this server thread sends out an initial greeting to the client, signifying that a connection between the client and server has been established. After a connection has been established, the POP3 Server and its client communicate by a sequence of command and response exchanges (see RFC 1081 for POP3 command and response specifications), the goal of which is to retrieve all messages currently in the user's INBOX. Specifically, the client sends POP3 commands to the server,

the server then executes the appropriate actions in response to these commands. Based on the result of command execution, the server forms an appropriate response, which it sends back to the client. If the POP3 Server encounters an error during execution, it sends back an error message to the user via the client screen.

In order to download messages, the user must first identify itself to the POP3 Server through his POP3 client account. Thus, the initial commands that a POP3 client usually issues are the USER and PASS commands, which send the user name and password, respectively, to the POP3 Server for validation. When the USER and PASS commands have been received, the server verifies the given user account information by checking for its existence in the Users Database. Once the user name and password have been verified, the user's incoming mailbox is opened for the POP3 client's exclusive use. If another user, which may either be a POP3 client or an IMAP4 client, has already opened the incoming mailbox, access to that particular mailbox will not be allowed and the connection will be terminated.

Internet Exchange 4's POP3 Server is a 32-bit application that supports multithreading for simultaneous processing of messages, thereby assuring fast message delivery.

Incoming mailbox as a subdirectory of MsgStore directory

In the Message Store, the mailbox for incoming messages or INBOX is implemented as a subdirectory of the MsgStore directory containing message databases and actual message files. Because mailbox and message status information is stored in relational databases, retrieval of this information is relatively fast. Furthermore, because each message is assigned its own file, as opposed to a scheme wherein all messages are stored in a single file, message text is obtained without having to determine and seek for its position within a file. Deletion of a message is likewise more straightforward as there is no need to move message text in order to overwrite the message that is to be deleted.

Auto-logout timer

The POP3 Server has an inactivity logout timer that causes the severance of a client connection once no command is received from the client within a period of 30 minutes, though this value is fully configurable. The presence of this timer ensures that resources are not wasted on idle clients or on clients that have encountered problems and are not able to communicate with the server anymore.

IMAP4 SERVER

IMAP4 offers users added flexibility in managing their mail over other post office access protocols, such as POP3. **Internet Exchange 4's** IMAP4 Server allows users to access their mailboxes via any IMAP4 capable clients, such as Microsoft Outlook Express, and Netscape Communicator. With IMAP4 support, users can manipulate their mailboxes/ folders on the server without having to download them to a local hard disk. End users can also create multilevel mailboxes on the server that can be easily renamed or deleted by them (with the proper authorization from the system administrator), as well as shared mailboxes which can be viewed concurrently in real time from multiple platforms. Another advantage is that users have the option to search for messages on the server based on various attributes such as message size, headers, and message sender, and to separate

attached files from the text and header portions of a message, with the searches being performed by the back-end message store.

While the IMAP4 Server is running, it listens on port 143 for a connection request from an IMAP4 client, which is usually initiated by the user selecting a mailbox to access from the client screen. Once such a connection request is received, the IMAP4 Server creates a thread that will handle further client transactions. Upon start-up, this server thread sends out an initial greeting to the client, signifying that a connection between the client and server has been established.

After a connection has been established, the IMAP4 Server and its client communicate by a sequence of command and response exchanges. Specifically, the client sends IMAP4 commands to the server upon instigation of the user, that is, each user action on the client screen corresponds to a sequence of one or more IMAP4 commands that are sent to the server. The server then executes the action corresponding to each command it receives and responds accordingly (see RFC 2060 for IMAP4 command and response specifications). Based on the server response to a command, the client is able to deduce the result of a user operation and informs the user via the client screen.

To read his/her messages, the user must first identify itself to the IMAP4 Server through his IMAP4 client account. Thus, one of the first commands an IMAP4 client issues is the LOGIN command, which sends the user name and password to the IMAP4 Server for validation. Once the LOGIN command is received, the server verifies the given user account information by checking with the Users Database. In the process, both the personal and shared mailboxes that a user can rightfully access are also determined. Consequently, when the client requests for a complete listing of the user's mailboxes, the server returns the names of both personal and shared mailboxes to which the user has access.

Because the IMAP4 Server supports multi-accessed mailboxes, any changes made by a user on a particular mailbox or the messages contained therein are seen by other users who simultaneously viewing the mailbox. Thus, an IMAP4 Server thread periodically checks for time stamp changes in the underlying message databases in order to determine whether any message in the currently selected mailbox has been read, deleted, expunged, etc. by another user. It also periodically checks for the existence of new messages in the selected mailbox by comparing the unique identifier value of the last message it has accessed with the unique identifier value of the most recent message in the mailbox. When any of these mailbox or message changes occurs, a server thread notifies its client of the event at the soonest time possible. The client, in turn, notifies the user by displaying appropriate markings on the client screen.

Incoming mailbox as subdirectories of the MsgStore directory

In the message store, mailboxes are implemented as subdirectories of the MsgStore directory containing message databases and actual message files. Because message structure information, as well as mailbox status information, are stored in relational databases, retrieval of this information is relatively fast. Furthermore, because each message is assigned its own file, as opposed to a scheme wherein all messages are stored in a single file, message text is obtained without having to determine and seek for its position within a file. Deletion of a message is likewise more straightforward as there is no need to move message text in order to overwrite the message that is to be deleted.

Support for nested mailboxes

The IMAP4 Server supports nesting of mailboxes without the user having to explicitly specify that he/she intends to create subfolders under a particular mailbox. In fact, the IMAP4 Server even allows a user's primary mailbox, the INBOX, to have subfolders. Because of this, users have greater flexibility in managing their messages. They can easily structure their mailboxes in such a way that groups and subgroups of messages are created. In addition, they do not have to know beforehand whether they will be having subfolders under a mailbox that they are intending to create. Because all mailboxes can be nested, there is no need to specify at creation time whether a mailbox will contain subfolders.

Support for shared mailboxes

The IMAP4 Server allows users to have both personal and shared mailboxes, all of which can be accessed using a single account. This does away with the need to use different account names in order to view either a personal or shared mailbox. Thus, with shared mailboxes, e-mail for a group of people can easily be managed without having to create a group login name and without having to distribute several copies of a single message to different people.

Shared mailboxes are created and assigned to users through an administrative utility. This administrative utility adds a mailbox to the shared mailboxes database and updates the profile of users to which the shared mailbox is assigned in the users database. By accessing the appropriate users and shared mailboxes database entries, the IMAP4 Server is then able to determine and locate the shared mailboxes a particular user has access to. In order for the IMAP4 Server to distinguish between personal and shared mailbox references, shared mailbox names have to be prefixed by "shared\". Consequently, this precludes a personal mailbox from being named "shared."

Unsolicited mailbox updates

Because the IMAP4 Server supports the IDLE command, it is able to give out unsolicited mailbox updates to the client. Thus, the client need not poll the server for changes (i.e. new mail, deletions) to the selected mailbox. Instead, the server transmits updates to the client in real-time while the IDLE command is in effect. This permits the user to have a consistent and current view of his mailbox.

Auto-logout timer

The IMAP4 Server has an inactivity logout timer that causes the severance of a client connection once no command is received from the client within a period of 30 minutes. The presence of this timer ensures that resources are not wasted on idle clients or on clients that have encountered problems and are not able to communicate with the server anymore. It also prevents users from having an incorrect view of their mailbox, the contents of which may have already been changed by another user.

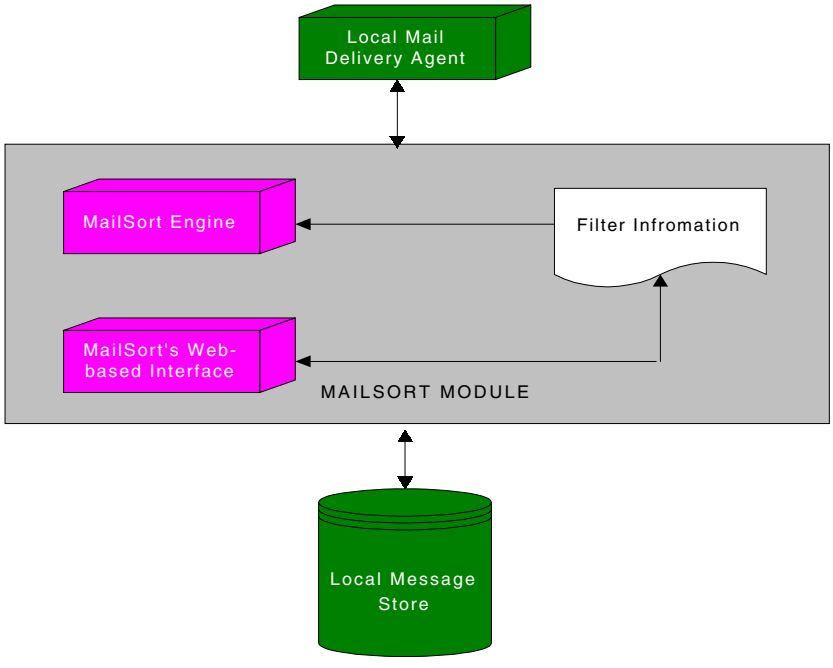
MAILSORT

Internet Exchange 4 features the Mail Sort filtering utility for defining rules so that the local mail delivery agent can direct messages to preselected mailboxes/folders other than the INBOX. It can also selectively forward messages to other addresses. This feature

enables users to sort incoming mail based on attributes such as the message sender and subject without having to go through all the messages. For example, a user may want to store messages in folders according to sender or subject line. This can easily be done in Internet Exchange using the MailSort utility. The sorting actions are done in the background by the server at message delivery time and are transparent to the users.

Mailsort is divided into two modules, the engine and the web interface. The engine is used by the Local Mail Delivery Agent to determine the destination of messages whose recipients maintain filtering information in their respective Message Store directories. The web-based interface allows the users to create and edit filter files used by the engine to inform the Local Mail Delivery Agent of the destination of the messages.

The Internet Exchange Mailsort Module provides users with a utility for preprocessing incoming email on a per user basis. Incoming messages that will be delivered to the Message Store through the Local Mail Delivery Agent will be sorted according to the rules set by the user via the web-based interface. The engine will read the filter file (*filter.txt*), and perform a very simple recursive-decent parsing to speed up the interpretation of the filter file.



The recursive-decent parsing scheme is divided into two steps, scanning or lexical analysis and parsing. The engine uses a lexical analyzer on the filter file to divide the file input into meaningful units. A parser is used to determine the relationships among these units. For this kind of input, the units would probably be lines of text, with a distinction between lines that contain a match of the target strings and the lines that do not. The division into units, which are usually called tokens, is known as *lexical analysis*. The token descriptions are regular expressions that are used by the lexical analyzer to scan the input text. As the input is divided into tokens, the parser establishes the relationships among them. This task is known as parsing and the list of rules that define the relationships is a grammar. The

parser automatically detects whenever a sequence of input tokens matches one of the rules in the grammar and also detects a syntax error whenever its input does not match any of the rules.

Sorting Incoming Messages

MailSort has several options for sorting incoming messages:

Move to a folder

This command tells the local mail delivery agent to deliver the message to the designated mailbox folder of the recipient as configured in the web-based interface.

Copy to a folder

This command tells the local mail delivery agent to deliver the message to the INBOX of the recipient and deliver a copy of the message to a designated mailbox folder as configured in the web-based interface.

Reject Incoming Messages

This command tells the local mail delivery agent to reject the message by not delivering it to its designated mailboxes.

Automatic Responses

Forward to predefined email addresses

This command tells the local mail delivery agent to forward the message to an email address(es) specified by the message's recipient via the web-based interface.

Vacation Utility

This command tells the local mail delivery agent to automatically send a specified reply message to all incoming email messages if configured by the recipient. For example, a user who is on vacation may have no access to his/her email for a week. He/she can configure MailSort to send replies to all incoming messages for that period, informing the message senders that he/she is on vacation.

LOCAL MAIL DELIVERY AGENT

The Local Mail Delivery Agent operates by polling the Shared Message Queue for local bound messages. When a message is available for local mail delivery, it retrieves the recipient name from the message envelope provided by the Shared Message Queue. This recipient name is verified against the user's Message Store database. Every local recipient has to be defined in the user's database before a message can be delivered.

The Local Mail Delivery Agent performs several operations when delivering a message. First, it writes the message file to a temporary buffer and parses the contents. Parsed envelope information and body information are saved in the internal data structure used by message store databases. Second, it reads the mailbox database of the user for the mailbox status. The mailbox status includes the next unique identifier to be assigned to the message when it is saved in the message store databases. Third, it gets the system time and uses it as the local time delivery of the message. This is saved in the message status data-

base. Fourth, the envelope and body data structures are saved in the message envelope and message body databases respectively. Lastly, the actual message file is saved in the user mailbox. The default user mailbox is the INBOX.

The Local Mail Delivery Agent may consult the Mailsort Engine module before delivering the local message. The Mailsort Engine provides a way to customize the delivery of messages instead of using the default mailbox. A user may create several mailboxes in his/her home directory or he/she may be a member of a shared folder. In this case, the Local Mail Delivery Agent will be dependent on the Mailsort Engine module in what to do and where to deliver the local message.

Using the Mailsort Engine module, the Local Mail Delivery Agent can customized the delivery of local messages. The Mailsort engine module supports the following functions:

- *Copy a message to different folder* – copies a message destined for the user's INBOX to the user's other mailbox.
- *Move a message to different folder* – delivers the message to a different user's mailbox without delivering the message to the user's INBOX.
- *Reject a message* – denies unwanted messages from an unknown sender.
- *Forward a message to different recipient* – forwards the message to a specified recipient.
- *Send vacation replies* – sends vacation replies to a message's sender whenever the user is unavailable.

PART 2

Installation

System Requirements

HARDWARE / SOFTWARE BASE CONFIGURATION

For optimum performance, it is recommended that Internet Exchange 4 and its components be run using the following minimum configurations:

Windows 95/98

- Pentium or higher
- Minimum recommended RAM: 64 MB
- Minimum recommended hard disk space for applications: 40 MB
- Minimum recommended hard disk space for message storage: 1GB

Windows NT 4.0 Server

- Pentium or higher
- Minimum recommended RAM: 96 MB
- Minimum recommended hard disk space for applications: 40 MB
- Minimum recommended hard disk space for message storage: 1GB

INTERNET EXCHANGE 4 COMPONENTS

Internet Exchange 4 consists of the following modules, which are in turn divided into several components.

Message Transfer Agent (MTA)

The MTA consists of the following components:

- SMTP Daemon (SMTPD)
- SMTP Client (SMTPC)
- MQ Router
- LDAP Server
- Distribution List Manager
- Preprocessor
- Btrieve Database Engine
- Anti-virus Module
- Anti-spam Module
- Auto-insertion Engine
- Auto-loop Detection DLL
- Administrative Tools
- Responder
- Web Server

IMAP4 Optimized Message Store

The Message Store consists of the following components:

- Message Store Server
- Local Mail Server
- Local Mail Delivery Agent (LMDA)
- IMAP4 Daemon
- POP3 Daemon

cc:Mail Connector

The cc:Mail consists of the following components:

- CCIN
- CCOUT

Notes Connector

The Notes consists of the following components:

- NOTESIN
- NOTESOUT

MEMORY USAGE

The hardware/software requirements mentioned above are only for running the machine's OS and other software needed by the OS to run Internet Exchange 4 properly. To determine the minimum memory requirement needed by your machine to run the OS and the Internet Exchange 4 modules installed on the machine, you must add the memory requirements of those modules to the base hardware configuration. Use Table 3a for reference to compute the minimum memory requirement of your machine.

For example, if you have a machine running Windows 95/98, you need a minimum of 64MB of RAM to run the OS. If you wish to install the Message Store on that machine, then you will have to install additional RAM of 2MB for the Message Store Server, 4MB for the POP3 Server, 6MB for the IMAP4 Server, 2MB for the Local Mail Server, and 4MB for the Local Mail Delivery Agent (as shown in Table 3a). Thus, the machine needs at least 82MB of RAM in order for the Message Store to run smoothly.

Internet Exchange 4 Modules	Memory Usage (MB)
CCIN	8
CCOUT	8
NOTESIN	8
NOTESOUT	8
SMTP Daemon (SMTPD)	6
SMTP Client (SMTPC)	4
MQ Router	2
LDAP Server	4
Local Mail Delivery Agent	4
Local Mail Server	2
Distribution List Manager	4
Message Store Server	2
IMAP4 Daemon	6
POP3 Daemon	4
Preprocessor	8
Btrieve Database Engine	4
Anti-virus Module	4
Responder	2
Web Server	2
Anti-spam Module	2
Auto-loop Detection DLL	2
Administrative Tools	8

Table 3a - Minimum memory requirements of Internet Exchange 4 components

Installation

INSTALLING THE MESSAGE STORE

In order to install the Internet Exchange 4 Message Store, simply run the install program "*Setup.exe*". The following dialog box will appear:

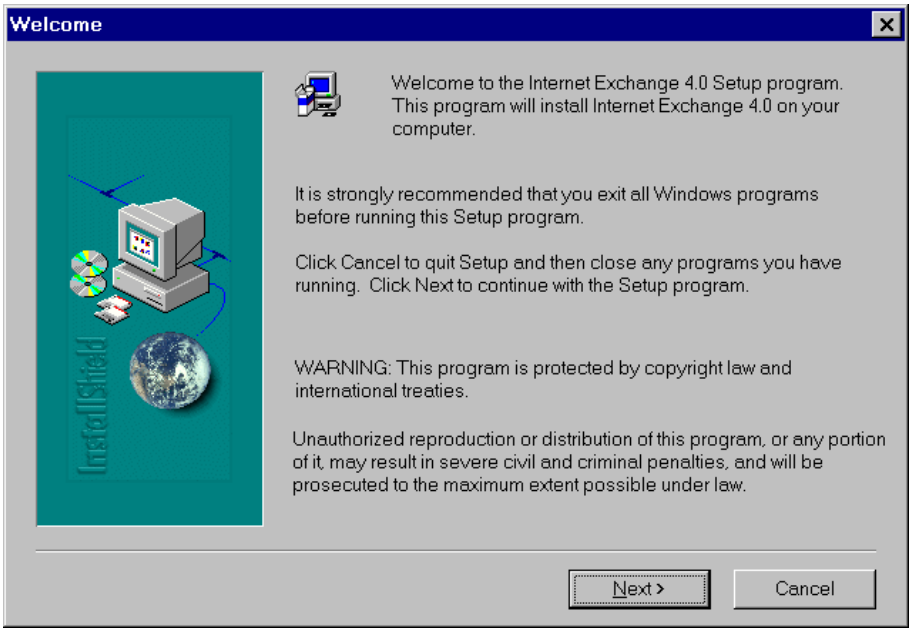


Figure 4a - Welcome Screen

Click on the *Next* button of the initial dialog box. A new dialog box (see Figure 4b) that allows the system administrator to choose the folder or directory where the **Internet Exchange** executable files will be copied will appear. By default, the executable files will be transferred to the folder Program Files\IMA\Internet Exchange 4. Click on the *Next* button to continue.

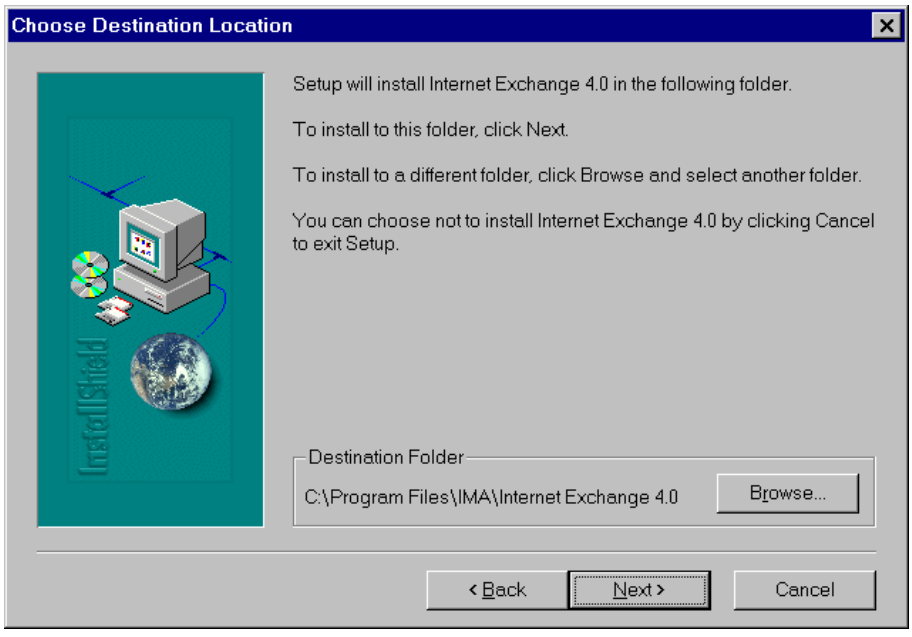


Figure 4b - Choose Destination Location

The next screen (Figure 4c) allows the system administrator to select which **Internet Exchange** components to install on the machine that currently runs the installer. If you have already installed the other Internet Exchange 4 modules on other separate machines, you may select *Message Store*. By selecting this option, the Message Store, together with MailSort and the Local Mail, IMAP4, and POP3 Servers, will be installed on the machine. Click on the *Next* button to proceed.

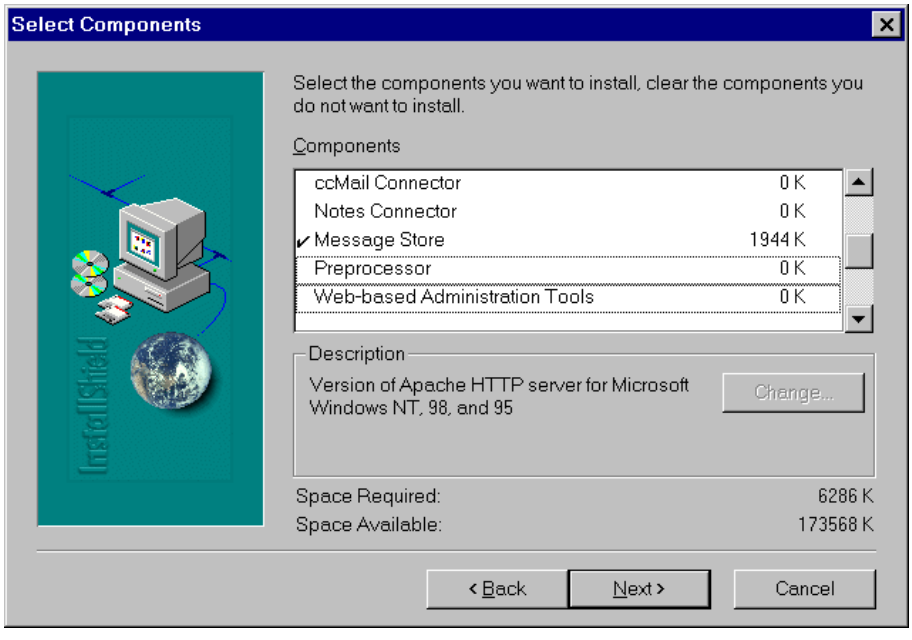


Figure 4c - Select Message Store

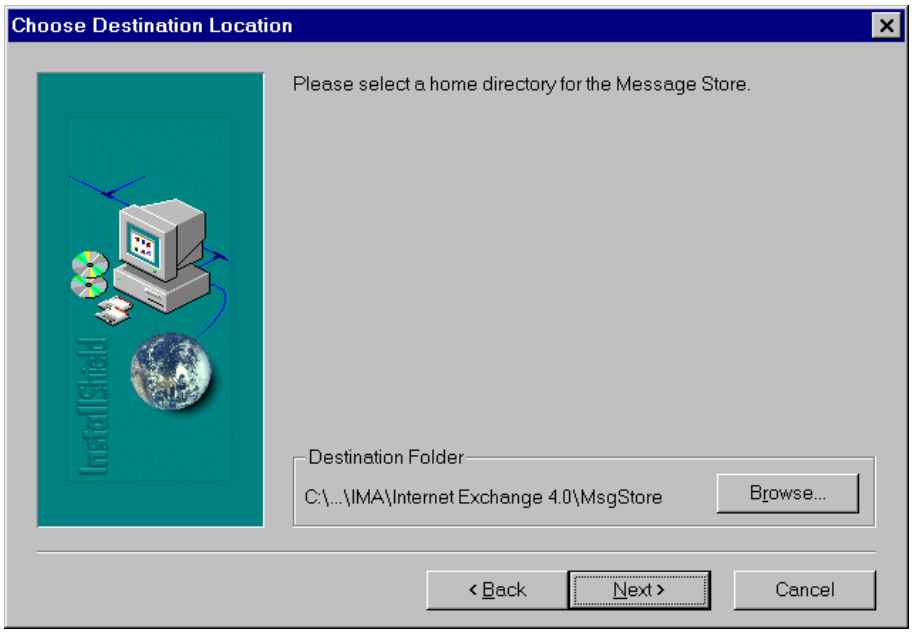


Figure 4d - Message Store Directory

The next dialog box (Figure 4d) enables the system administrator to select/create the working directory for the Message Store. By default, the Message Store will reside in the MsgStore subdirectory under the executable directory. Click on the *Next* button to continue.

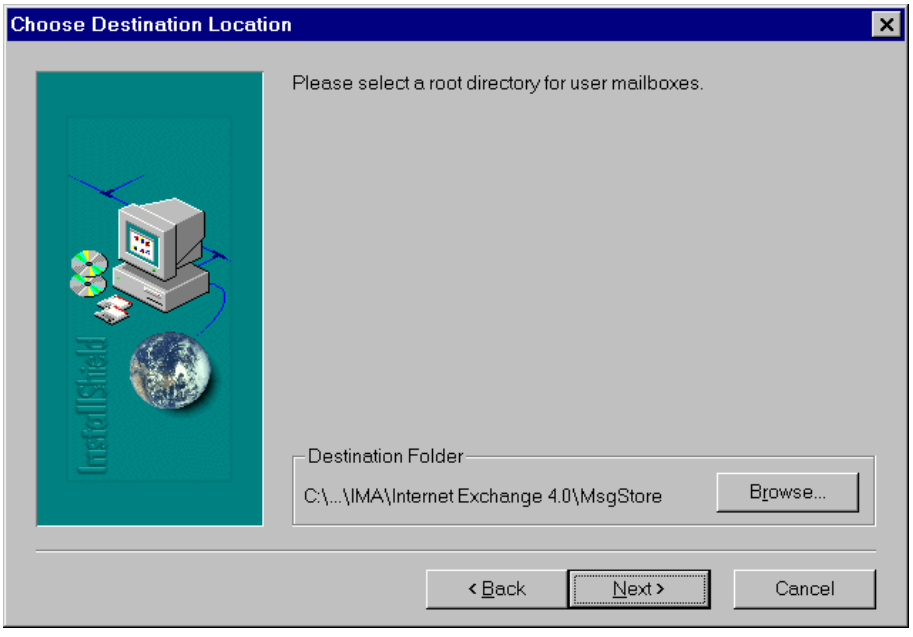


Figure 4e - Screen for creating user mailboxes directory

The next dialog box (Figure 4e) enables the system administrator to select/create the working directory for the users' mailboxes. By default, the users' mailboxes subdirectory

will reside under the executable directory. Click on the *Next* button to continue.

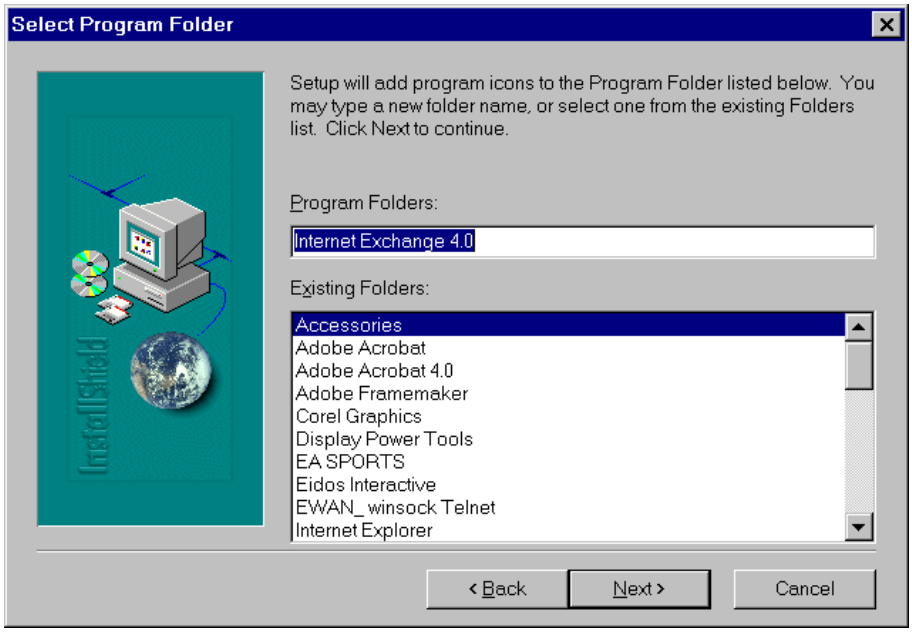


Figure 4f - Screen for creating user mailboxes directory

The next dialog box (Figure 4f) allows the system administrator to create a new program folder in the Start Programs menu or choose an existing folder in which the Message Store program icons will be placed. By default, a new program folder named **Internet Exchange 4** will be created in the Start Programs menu. Click on the *Next* button to start the installation. After the installation process is done, a new screen will appear (Figure 4g). Click on the *Finish* button to complete the installation process.

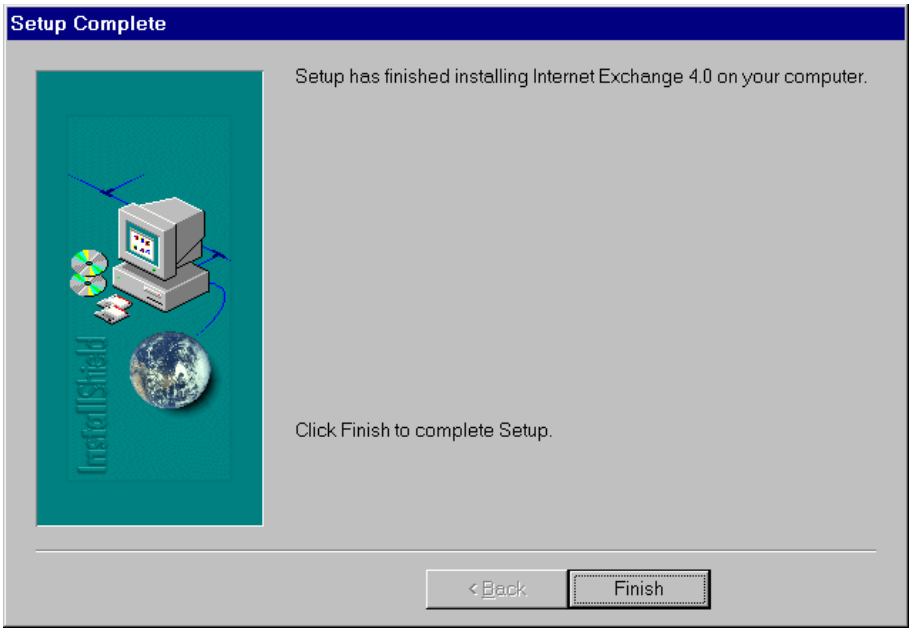


Figure 4g - Finish Installation

INSTALLING THE LICENSES

License certificates are used to enable the software. To install **Internet Exchange 4**, a license certificate containing the license information needed to activate the license keys is required.

Installing the software does not already mean access to the software. License keys need to be requested from an authorized license manager. After registration, a certificate which contains information on the licensed modules will be issued via email. This certificate is needed to identify the user when installing the license key. Validation of the certificates and the license keys are needed in order to be able to use the applications. Both elements are needed to install the license for particular module.

License Types

There are three types of licenses for Internet Exchange 4: *Evaluation*, *Interim,* and *Permanent*.

Evaluation Keys

These keys are time-limited keys (normally 30 days) and are used with the freely available evaluation copies of Internet Exchange 4. Once a registration form is received from the customer, the authorized license manager generates this key and gives it to the client.

Interim Keys

These keys are also time-limited keys, except that an *Interim* license can be updated to a permanent license at a later date. These keys are used for serialized or purchased copies of Internet Exchange 4.

Permanent Keys

These keys are used for the conversion of a given interim key into a permanent license, and are only applied to serialized copies of the software. Unlike *Evaluation* and *Interim* licenses, *Permanent* licenses are based on the Internet Exchange 4 serial number and the Fully Qualified Domain Name (FQDN) of the gateway machine. *Permanent* keys are generated only by an authorized license manager.

Running the License Manager

The Internet Exchange 4 licenses are installed/updated via the *License Update* pages provided by the Web Administration Interface. To install/update licenses, look for the Internet Exchange icon in the *Programs* menu. Click on the *Apache Web Server* (or you may run any Web server on your machine) to start the Web-based administration utilities. Then run your Web browser and type the name of your host in the URL field. If the machine running the Web-based Administration Tools is named *cuenca.ima.com*, type *cuenca.ima.com* in the URL field. The authentication page for the main Web Administration Interface will appear (see Figures 4h). Enter the user name and the corresponding password in the pop-up dialog box then click on the *OK* button.

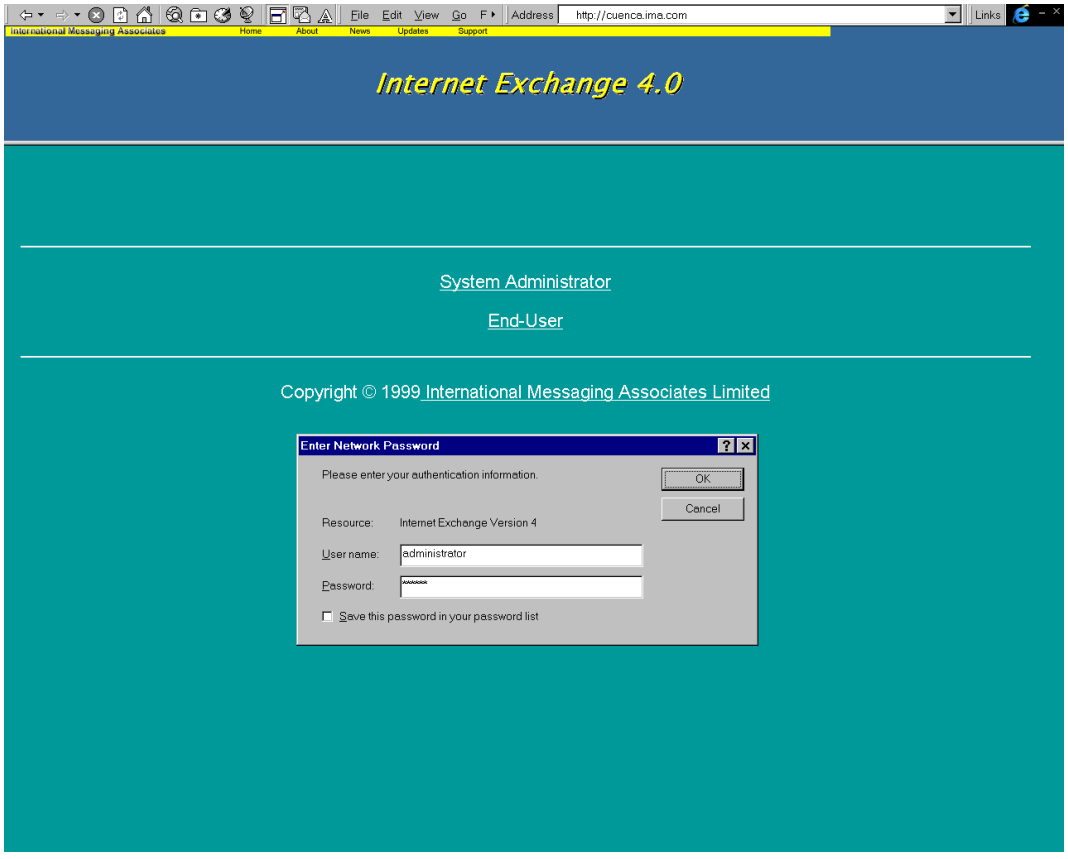


Figure 4h - Web Administration Interface Authentication Page

If the user name and password that you entered have been verified to be correct, the main Web Administration Interface will appear (see Figures 4i.1 and 4i.2)

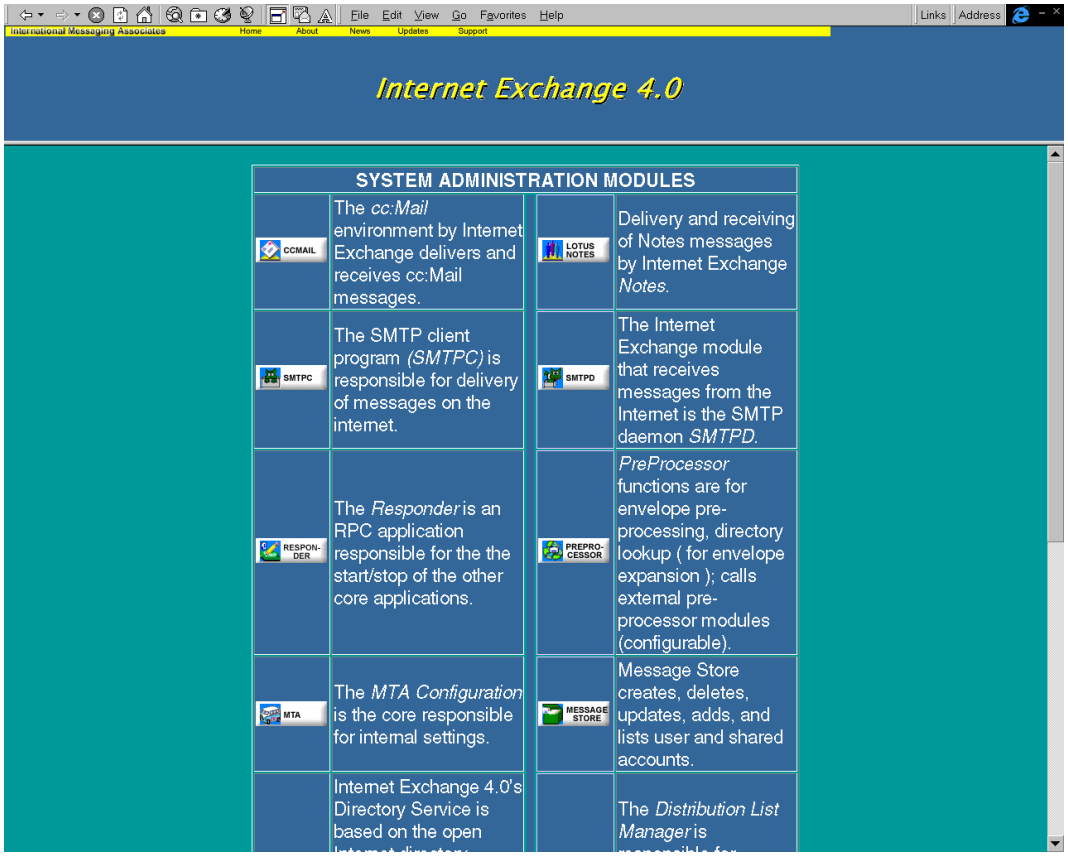


Figure 4i.1 - Main Web Administration Interface

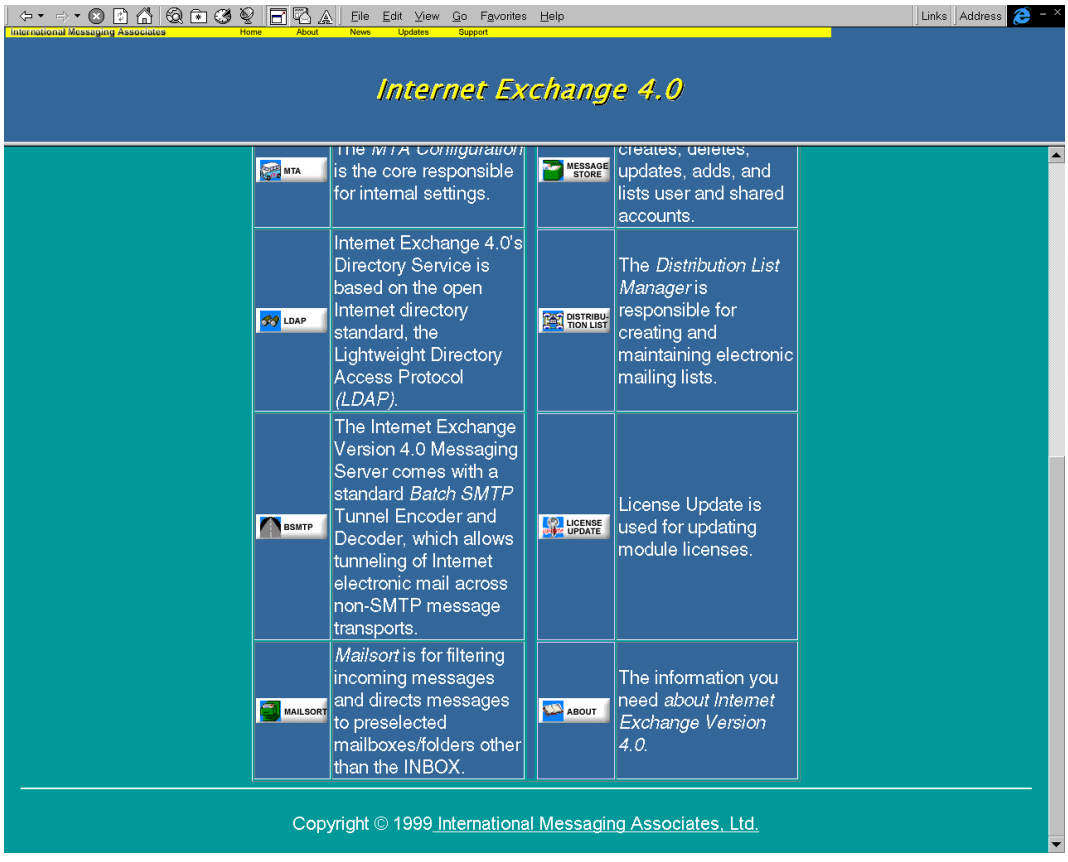


Figure 4i.2 - Main Web Administration Interface

Click on the *License Update* button to go to the main Licensing Tools page (see Figure 4j). In this page, click on the *License Manager* link. The next screen (Figure 4k) will ask for the directory where the certificates are stored and the name of the module to be licensed.

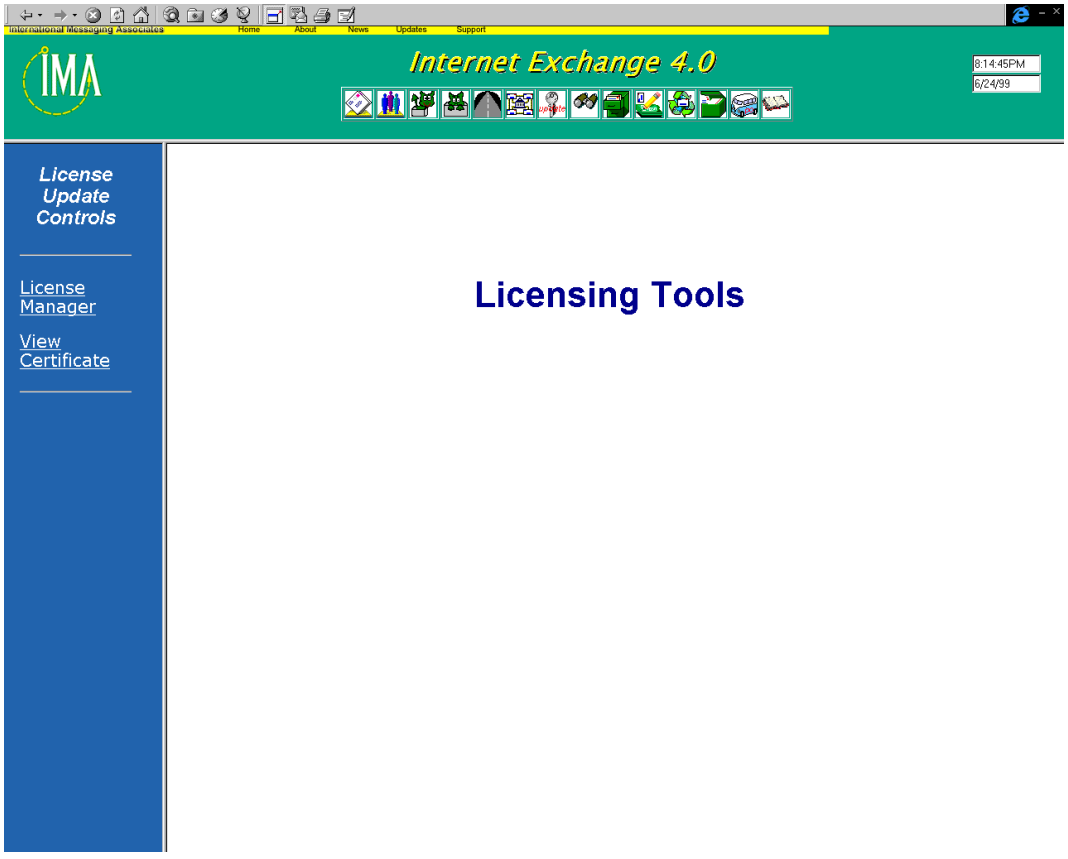


Figure 4j - Main Licensing Page

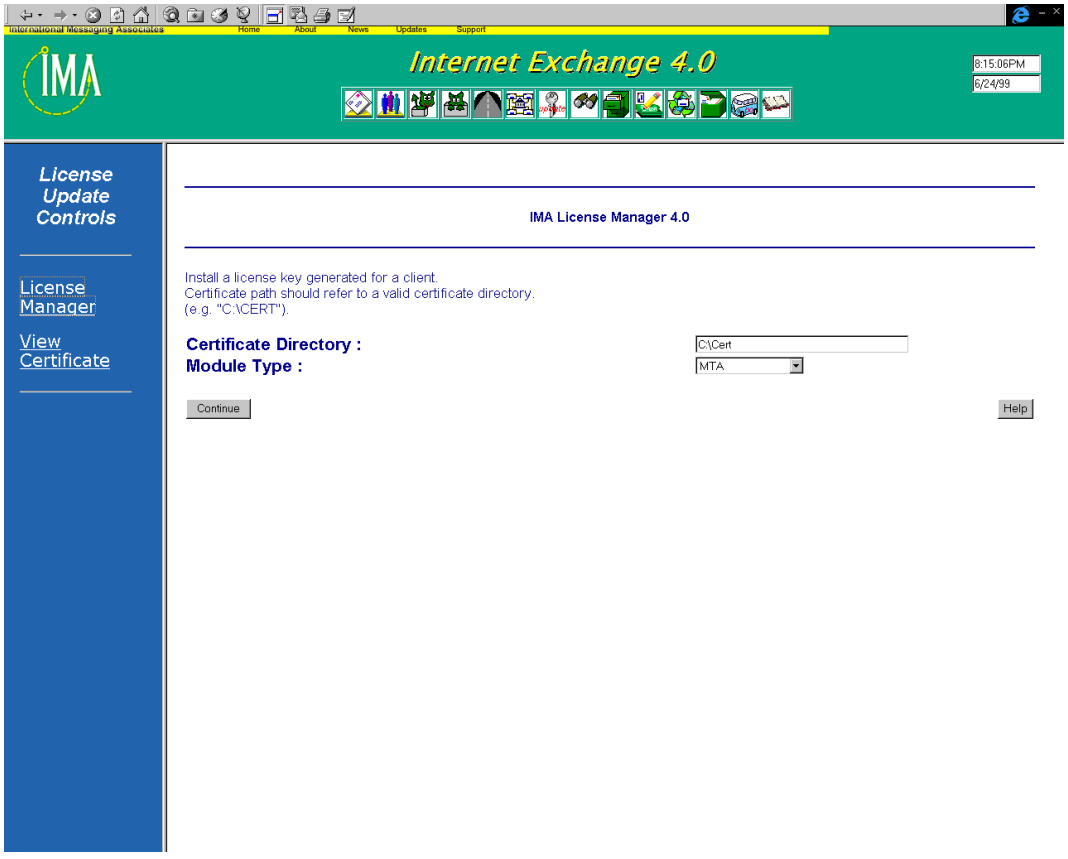


Figure 4k - Internet Exchange License Manager

Certificate Directory

The initial directory entry displayed is based on the IEMTA.INI file entry. This entry should point to the directory containing the certificate files that were sent to you by IMA via email (e.g. c:\CERT).

Module Type

Internet Exchange 4 is made up of the *Internet Exchange MTA*, *Internet Exchange Message Store*, *Internet Exchange cc:Mail Connector Module*, and *Internet Exchange Notes Connector Module.*

After entering the path of the directory that contains the certificate files and selecting the module to be licensed, click on the *Continue* button to start the installation of the license key. The License Manager then verifies if the certificate exists. If this file is missing, the licensing process will terminate. If the certificate is found, its contents are extracted and displayed in a new screen (Figure 4l). This screen displays license information extracted from the certificate, except the *FQDN* value, which is based on the IEMTA.INI file.

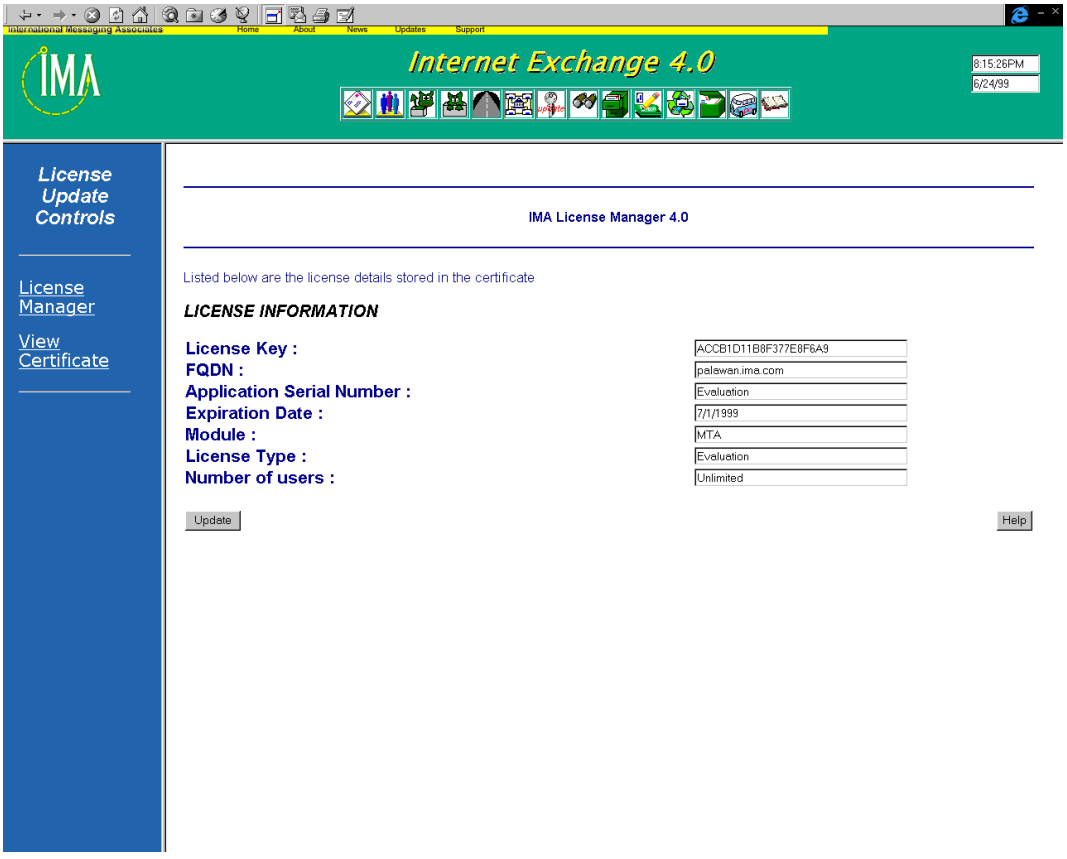


Figure 4l - License Information Page

License Key
This field displays the license key stored in the certificate.

FQDN
This field displays the Fully Qualified Domain Name (FQDN) based on the INI file entries, GatewayHostName, and GatewayDomain entries.

Application Serial Number
This field displays the application serial number for the module being licensed.

Expiration Date
This field displays the date of validity for the module's certificate/license key. The date format should be *mm/dd/yyyy.*

Module
This field displays the name of the module to which this particular license certificate is assigned. The module types are: *Internet Exchange MTA*, *Internet Exchange Message Store*, *Internet Exchange cc:Mail Connector* and *Internet Exchange Notes Connector.*

License Type
This field displays the type of license that will be issued for the client. The different

license types are: *Evaluation, Interim,* and *Permanent*.

Number of Users

This field displays the number of allowed users for the module being licensed.

After the information displayed on the screen has been verified to be correct, click on the *Update* button to continue the licensing process. If the licensing process is successful, the IEMTA.INI file will be updated to reflect the license key used for the module.

An option to view the contents of the certificate is provided. To do this, click on the *View Certificate* link on the License Information Page. The *Certificate Page,* which displays the information for the licensed module(s) will appear (Figure 4m).

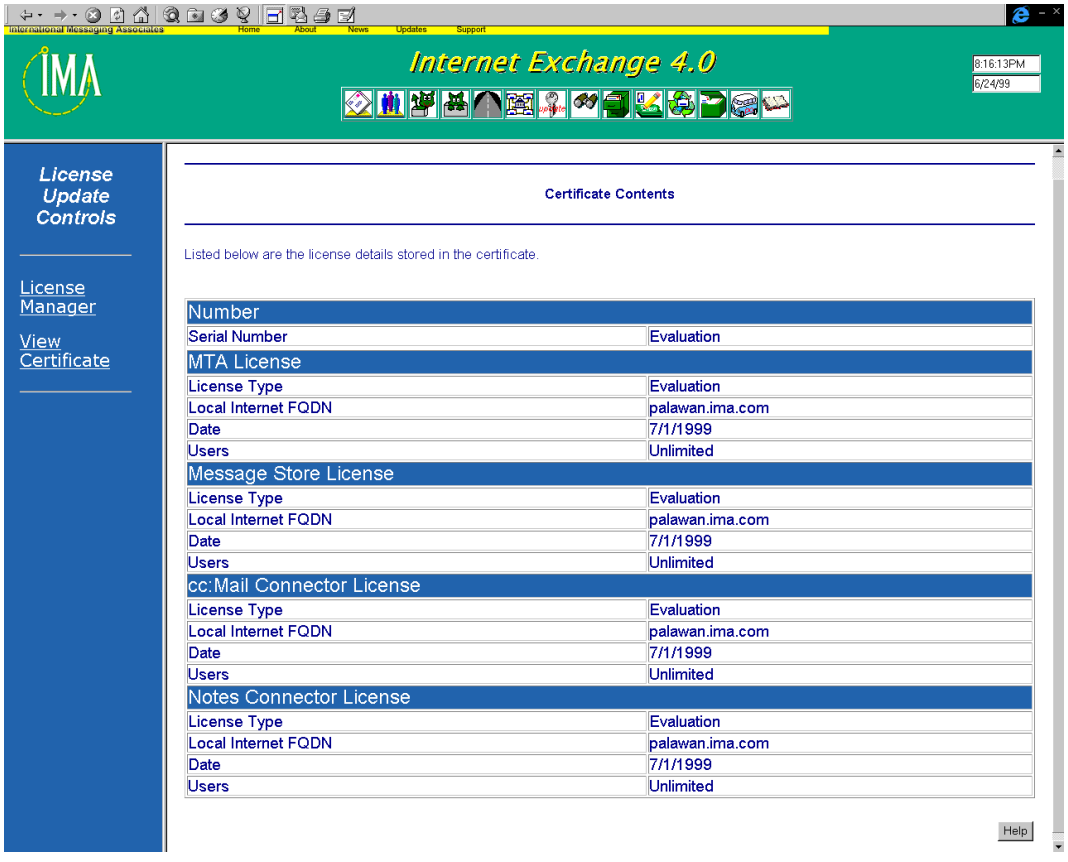


Figure 4m - View Certificate Page

PART 3

Operation and Administration

Chapter 5

Configuring the Message Store

CONFIGURING THE USERS DATABASE

The IMAP4 Optimized Message Store allows the system administrator to effectively manage individual user accounts via a Web-based interface. To configure the Users Database, go the main Web Administration Interface and click on the *Message Store* button. A screen for configuring the Message Store's various options will appear.

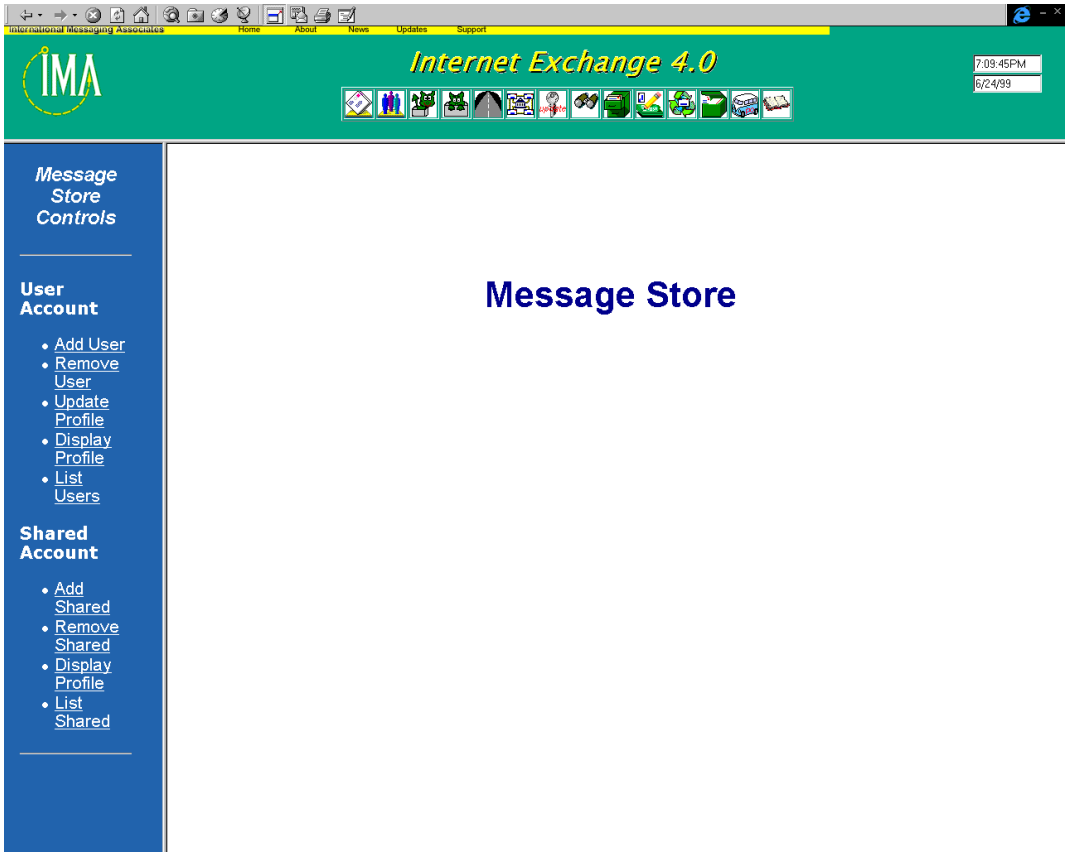


Figure 5a - User Management Interface

Using these options, the system administrator can manage user accounts using:

- *Creation of user accounts* - creates or to adds a new user account.
- *Deletion of user accounts* - deletes an existing user account.
- *Update of user accounts* - changes the password of a user account/adds or removes shared accounts.
- *Information on user accounts* - displays information on a particular user account, such as user account name, home directory, personal mailboxes and shared mailboxes.
- *Listing of user accounts* - lists all the registered user accounts.

All registered user accounts are stored in the Message Store.

Adding Users

To add a new user to the Message Store, click on the *Add User* link on the left frame of the User Management Interface. A new screen will appear.

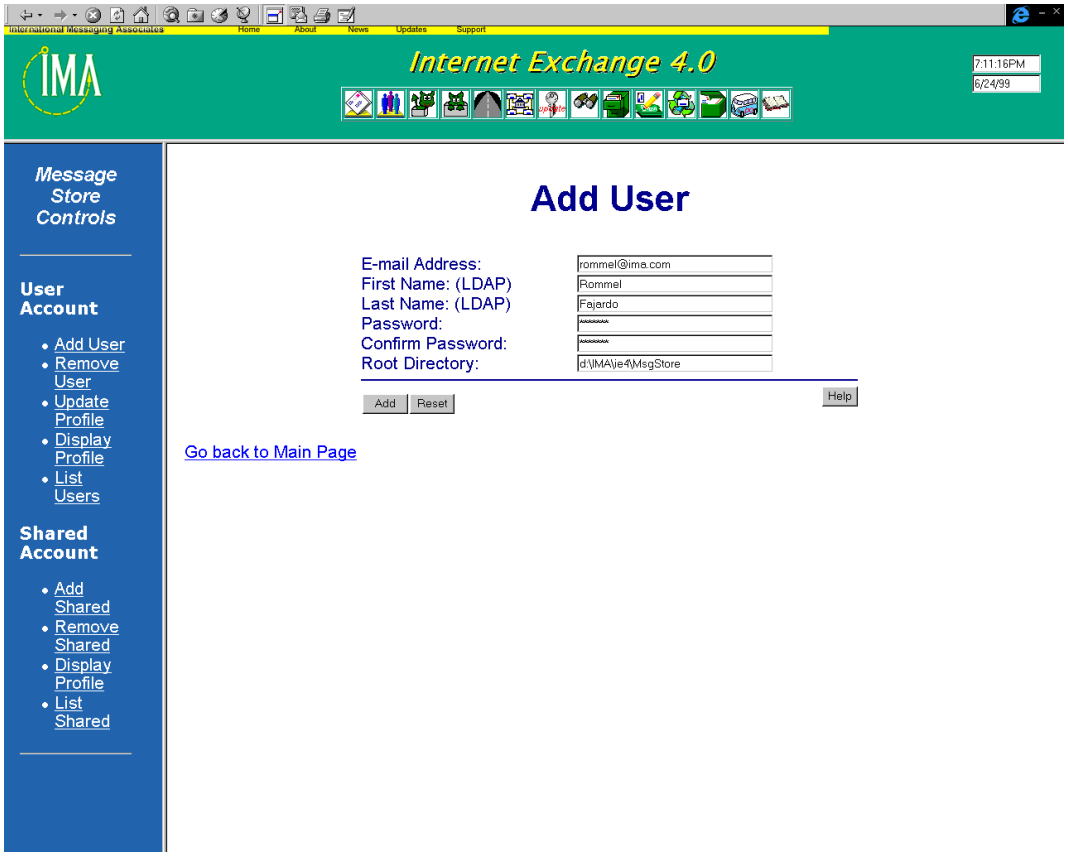


Figure 5b - User Addition To Message Store

Email Address

The email address of the new user to be registered. The email address entered provides users with a reference to their personal mailboxes. Senders use these email addresses to reach the user's (recipient) mailbox.

First Name
The first name of the user to added to the Message Store.

Last Name
The last name of the user to added to the Message Store.

Password/Confirm Password
This specifies the password to be used by the new user. The password should be entered twice to make sure that it is typed correctly.

Root Directory
This specifies the physical location of the user's mailboxes and messages. A default location is already defined in the IEMTA.INI that is displayed in the form. The system administrator can easily change the home directory of the user by altering the default value.

After providing these information, click on the *Add* button and the new user will be added to the Message Store. To clear all the fields on this screen, click on the *Reset* button. This will erase all previous information entered.

Removing Users
To delete a user from the Message Store, click on the *Remove User* link on the left frame of the User Management Interface. A new screen will appear.

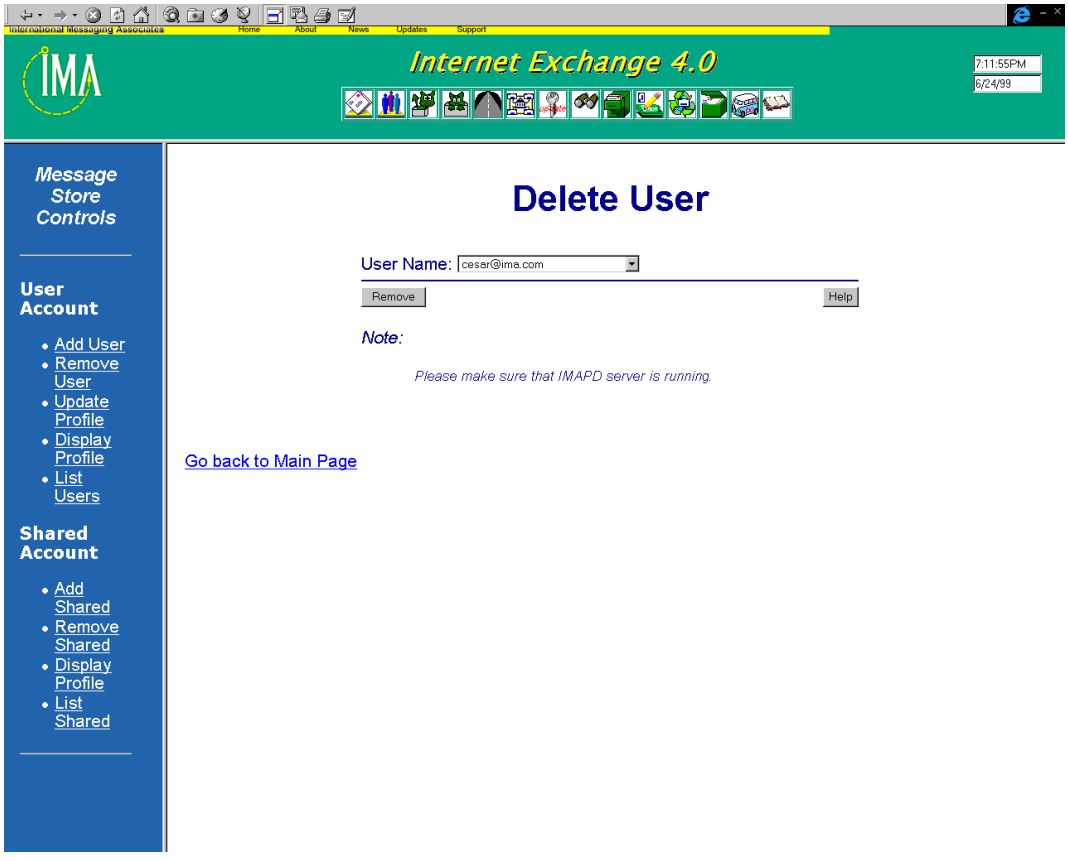


Figure 5c - Message Store User Removal

Update User Profile

To update user information, click on the *Update Profile* link on the left frame of the User Management Interface. A new screen will appear.

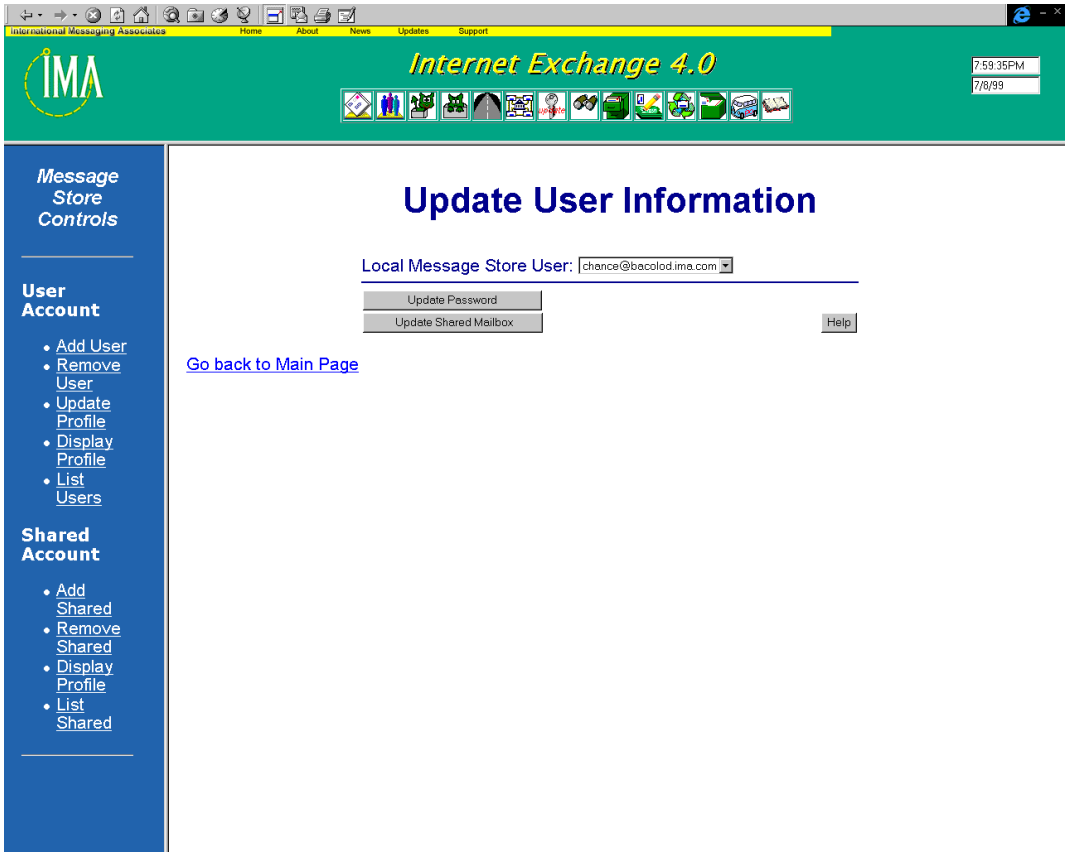


Figure 5d - Updating Users Information

The user password and shared mailboxes can be updated. By clicking on the profile (password or shared mailboxes) to be edited, a new form will appear displaying the current information on the selected profile (see Figures 5e and 5f).

Update Password

To change the password of the selected user, click on the *Update Password* button. A window to modify the current password will be displayed. Enter/confirm new password.

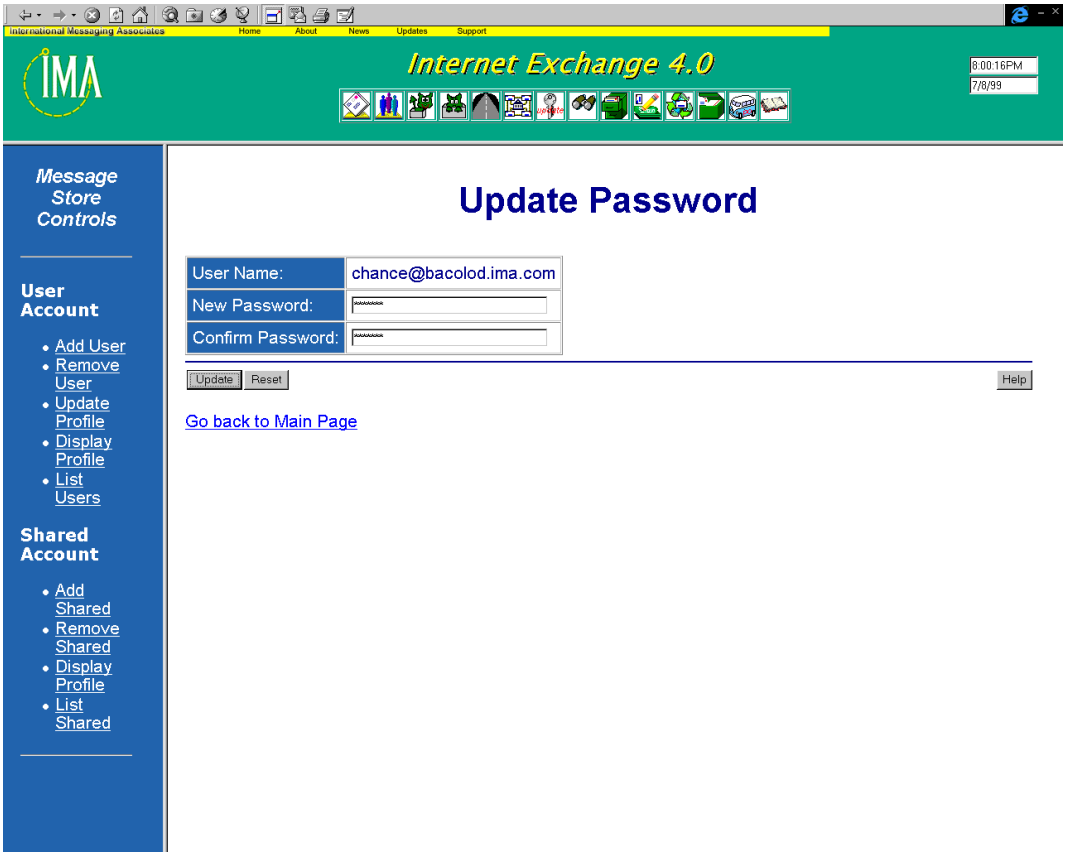


Figure 5e - Update Password Screen

Update Shared Mailbox

Adds/removes a shared mailbox to/from the list of shared mailboxes subscribed to by the user given in the User Name field. To update a shared mailbox, the following fields should be set:

Remove List Box

Selects the mailbox that is to be removed from the list of mailboxes subscribed to by the user given in the User Name field.Click the *Remove* button to delete the selected mailbox from the user's shared mailbox list in the users database of the Message Store.

Add List Box

Selects the mailbox that is to be added to the list of mailboxes subscribed to by the user given in the User Name field. Click the *Add* button to add the selected mailbox to the user's shared mailbox list in the users database of the Message Store.

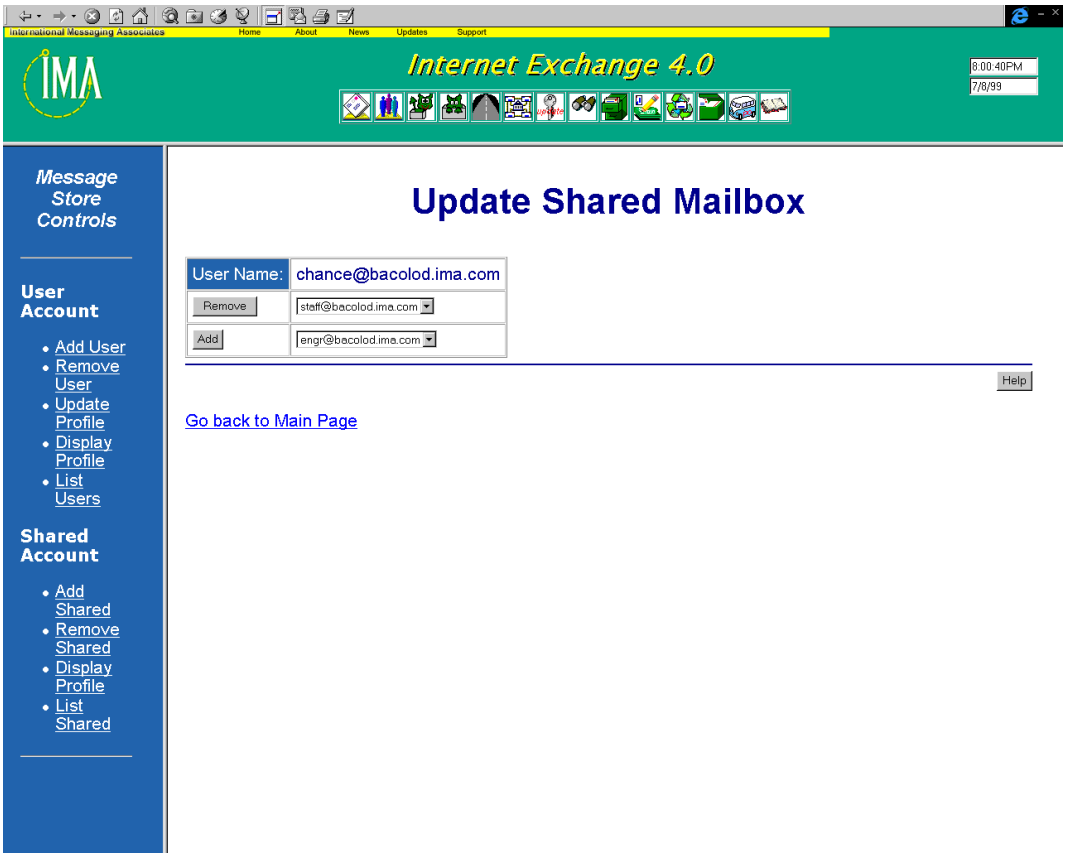


Figure 5f - Update Shared Mailbox

View List of All Registered Users

To list all users registered in the Users Database, click on the *List Users* link on the left frame of the User Management Interface. A new screen will appear which contains a list of all registered users.

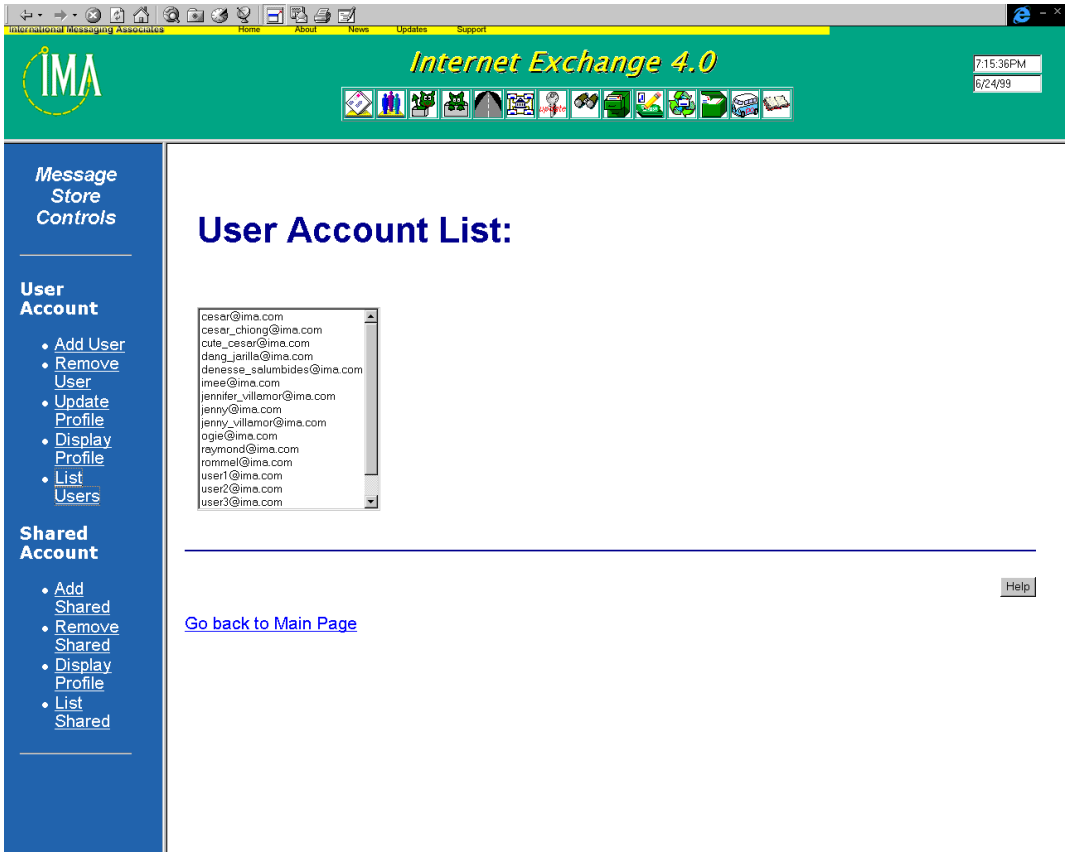


Figure 5g - User Account List

CONFIGURING SHARED MAILBOXES DATABASE

The IMAP4 Optimized Message Store also features a Web-based interface for creating, deleting, finding, and/or listing all shared mailboxes in the Message Store's Shared Mailboxes Database.

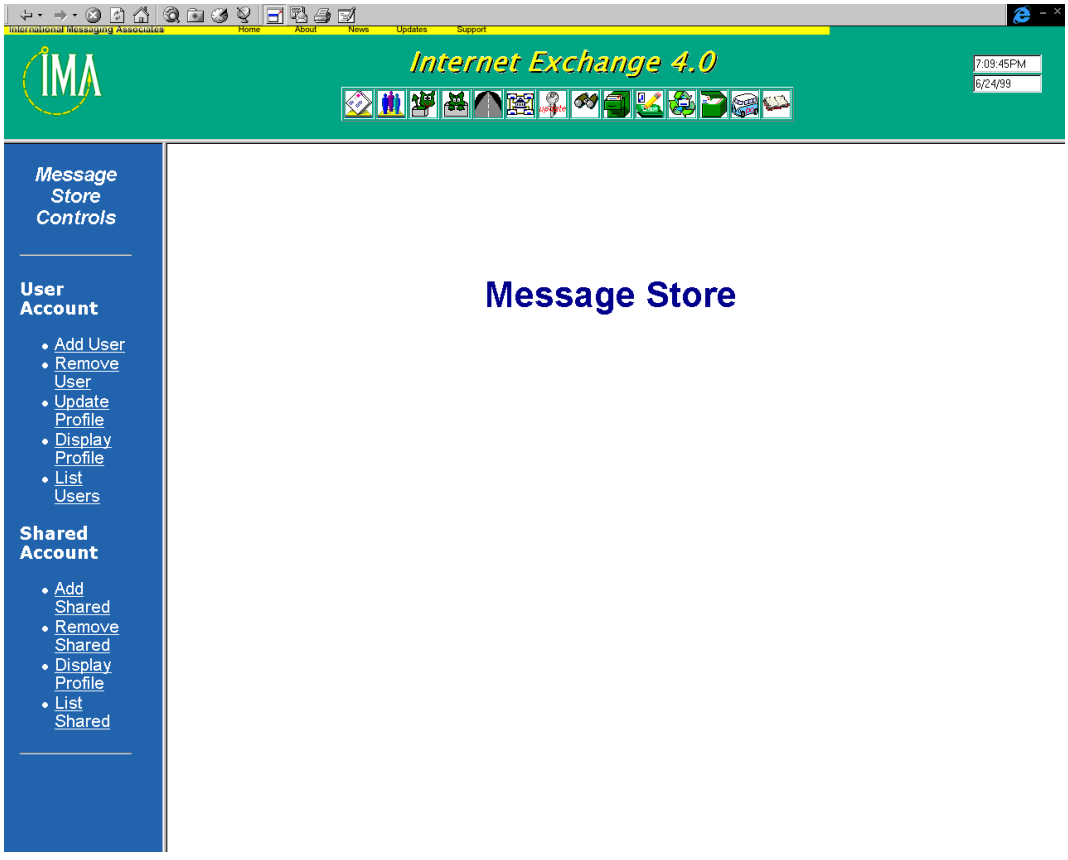


Figure 5h - Configuring Shared Mailboxes Database

Creating Shared Accounts

To add a shared mailbox to the Shared Mailboxes Database, click on the *Add Shared Mailbox* link on the left frame of the User Management Interface. A new screen will appear (see Figure 5i).

The following information must be entered for a shared mailbox to be added to the Shared Mailbox Database:

Email Address

The email address of the shared account to be created in the Message Store.

User name in LDAP

The name of the shared account as it is entered in the Directory Server.

Root Directory

The location of the actual messages for a particular shared account. The members will be

the initial Local Message Store users that will have access to this shared account. The *Home Directory* text box displays an initial path that can be pointed at other locations.

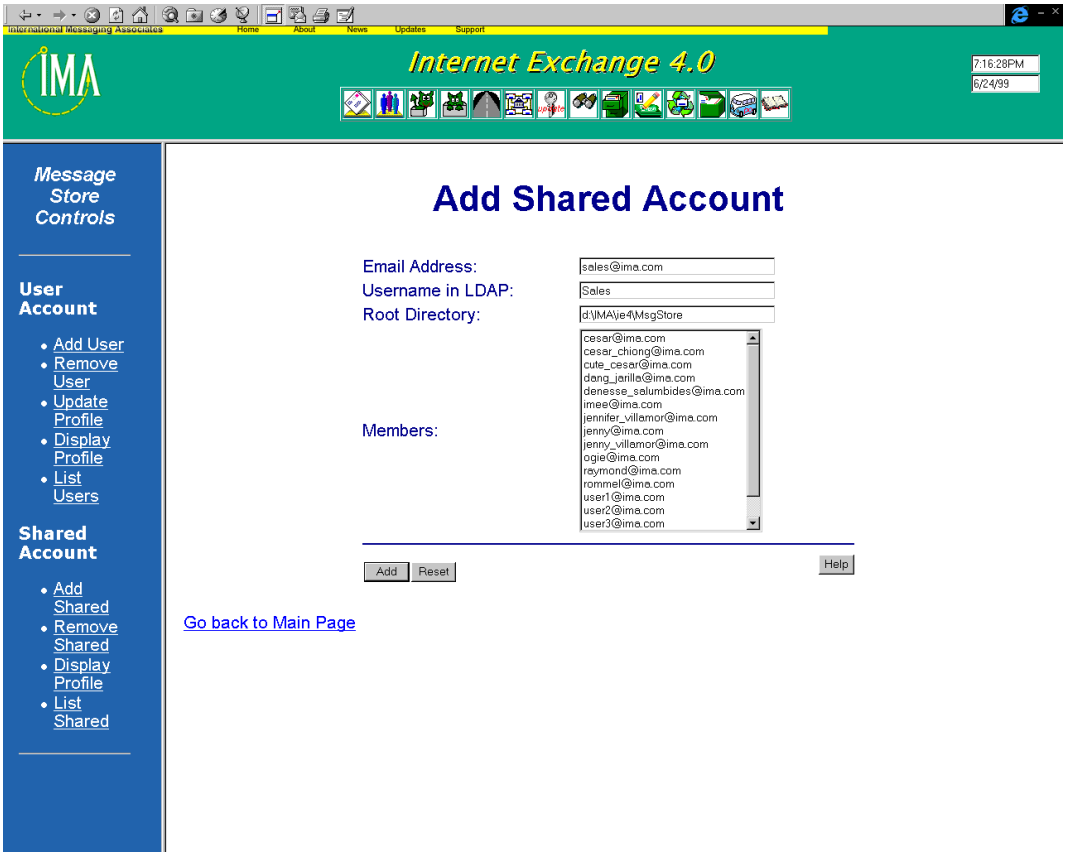


Figure 5i - Web-based Interface for Adding Shared Accounts

Members

The system administrator can choose from a list box that contains all the registered users of the Message Store initial members of the shared account name entered. At least one user must be selected for a shared account. To select the members of a shared account, highlight the names of the users and click on the *Add* button. This activates the CGI program that will add the shared account in the Local Message Store and register it in the LDAP directory service.

Deleting Shared Accounts

To remove a shared mailbox from the Shared Mailboxes Database, click on the *Remove Shared Mailbox* link on the left frame of the User Management Interface. A new screen will appear.

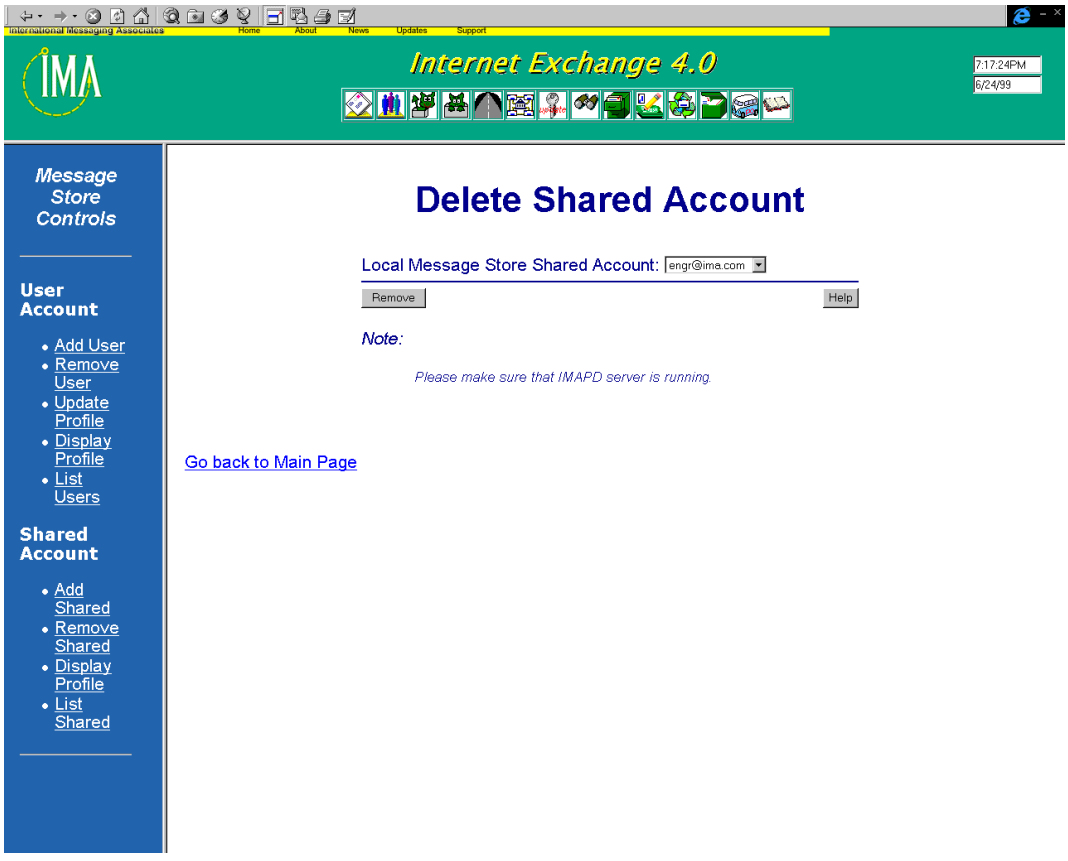


Figure 5j - Delete Shared Account

To remove a shared account from the Shared Mailboxes Database, select the name of the shared account to be deleted and click on the *Remove* button.

Finding Shared Accounts

To search for a shared mailbox from the Shared Mailboxes Database, click on the *Find Shared Mailbox* link on the left frame of the User Management Interface. A new screen will appear (see Figure 5k).

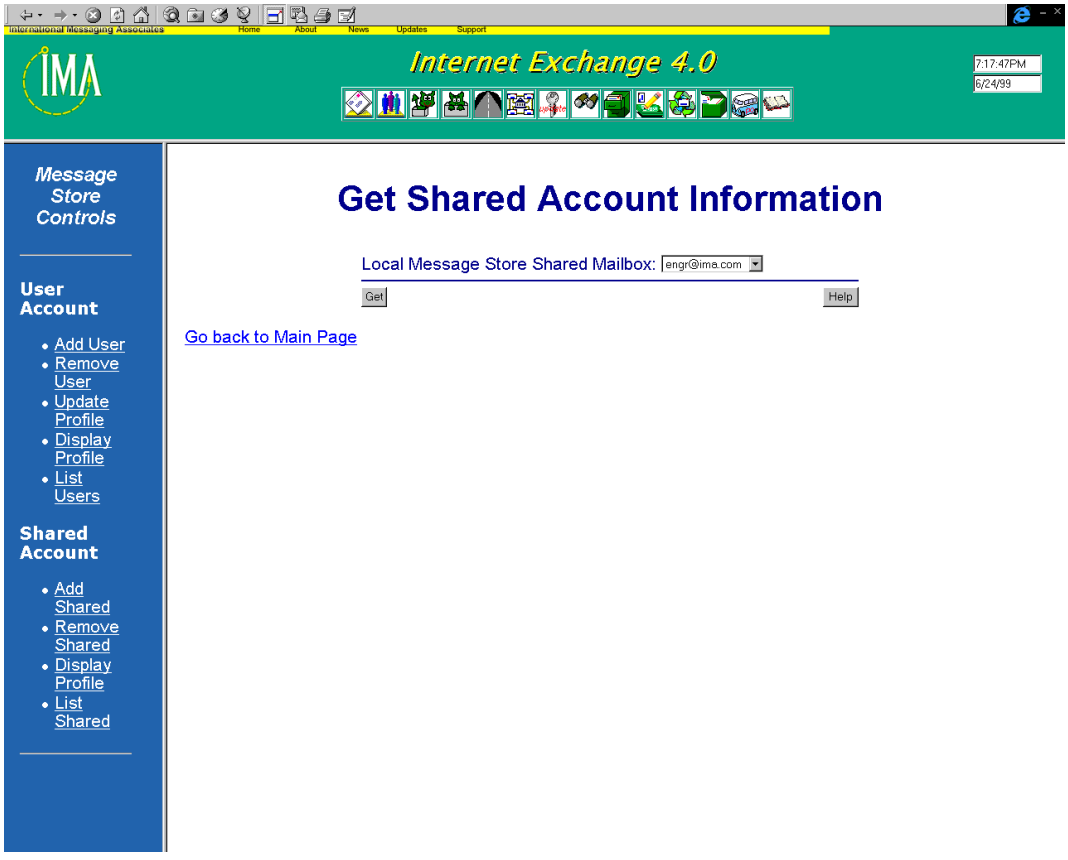


Figure 5k - Get Shared Account Information

To display the properties of shared mailbox, select the name of the mailbox and click on the *Get* button. A screen displaying the home directory of that particular shared mailbox and its registered users will be displayed (see Figure 5l).

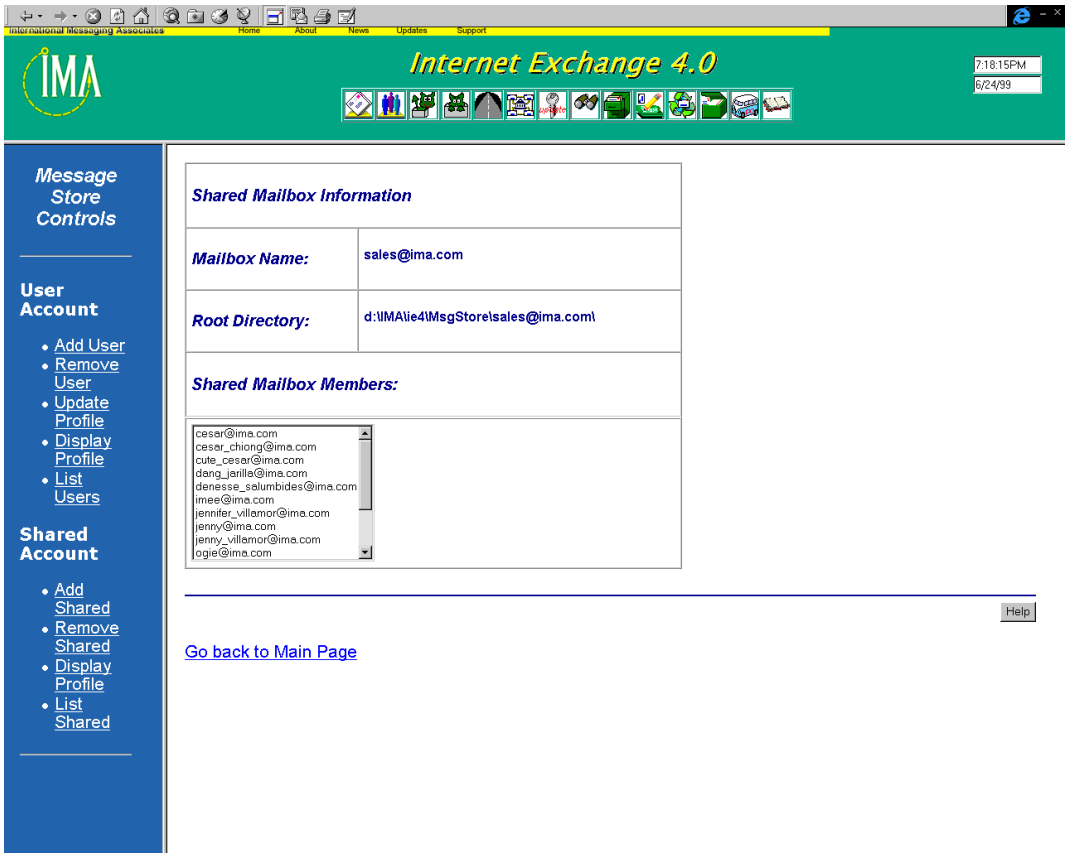


Figure 5l - Shared Account Properties

View List of All Registered Shared Accounts

To see a list of all the shared accounts in the *Shared Mailboxes Database*, click on the *List Shared Mailbox* link on the left frame of the User Management Interface. A new screen will appear (see Figure 5m).

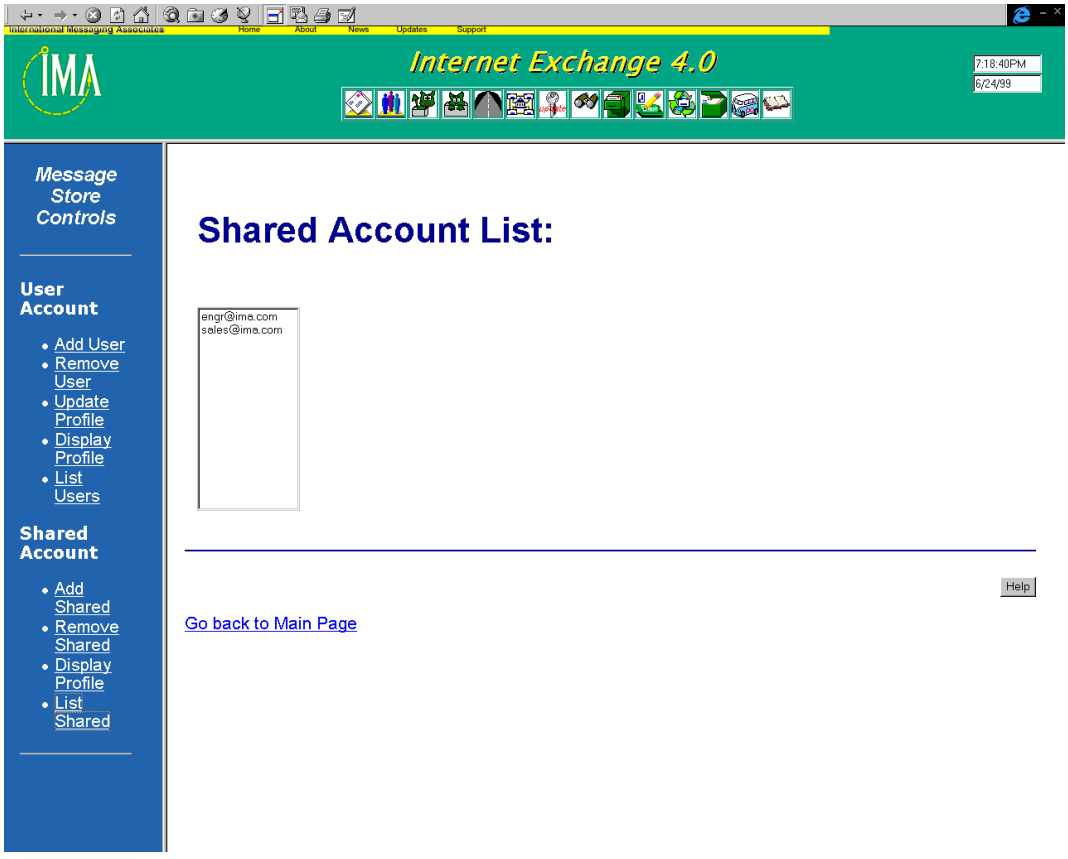


Figure 5m - Shared Account List

CONFIGURING MAILSORT

Individual users of the Local Message Store can configure their own filtering mechanism. To be able to use the web interface for configuring the Mailsort engine, the user will have to be authenticated. To go the authentication page (see Figure 5n), click on the MailSort button on the main Web Administration Interface.

Log on to the MailSort engine

To log on to the MailSort engine, all the system administrator has to do is select the user name from the pull-down menu. By clicking on the *Submit* button, MailSort engine is invoked.

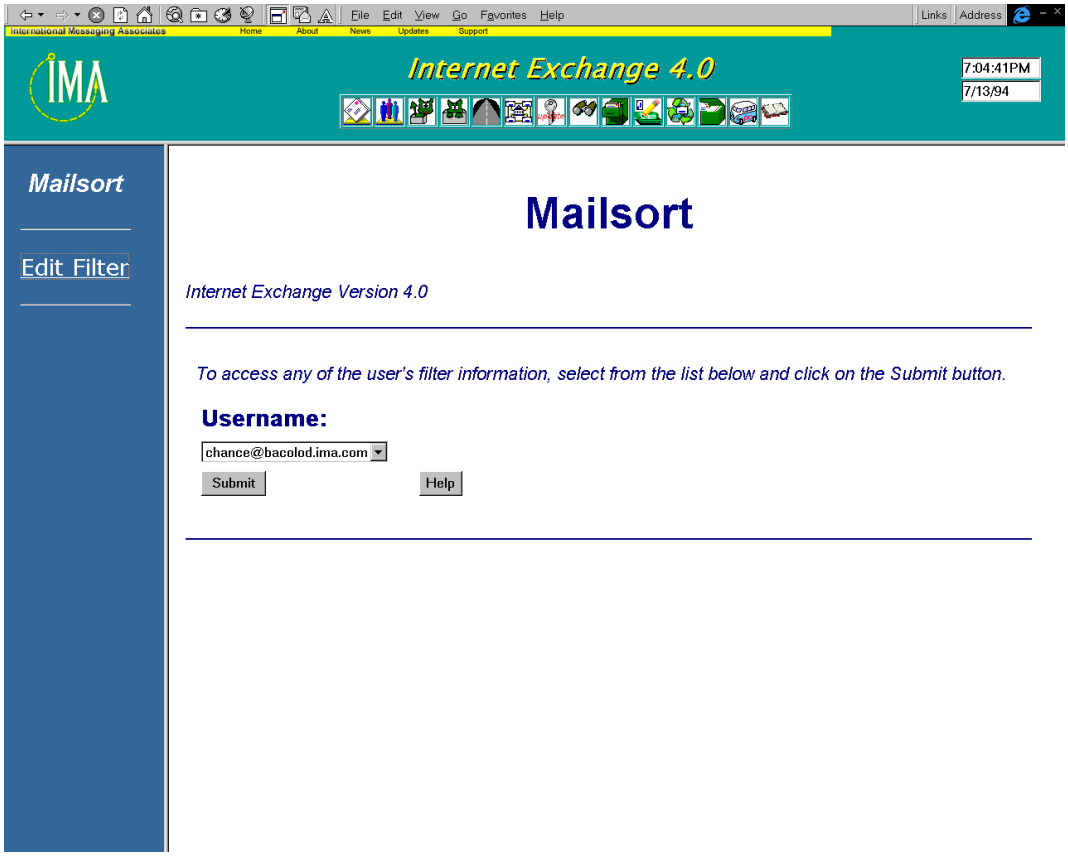


Figure 5n - Mailsort Authentication Page

Creating a filter file

After the user has been authenticated by the Message Store, a web-based interface for creating filter files will appear (see Figure 5o). By clicking on the *New* button, another web-based interface for entering the information needed to create a filter file is invoked (see Figure 5p), provided that there is still no filter file that exists for the user.

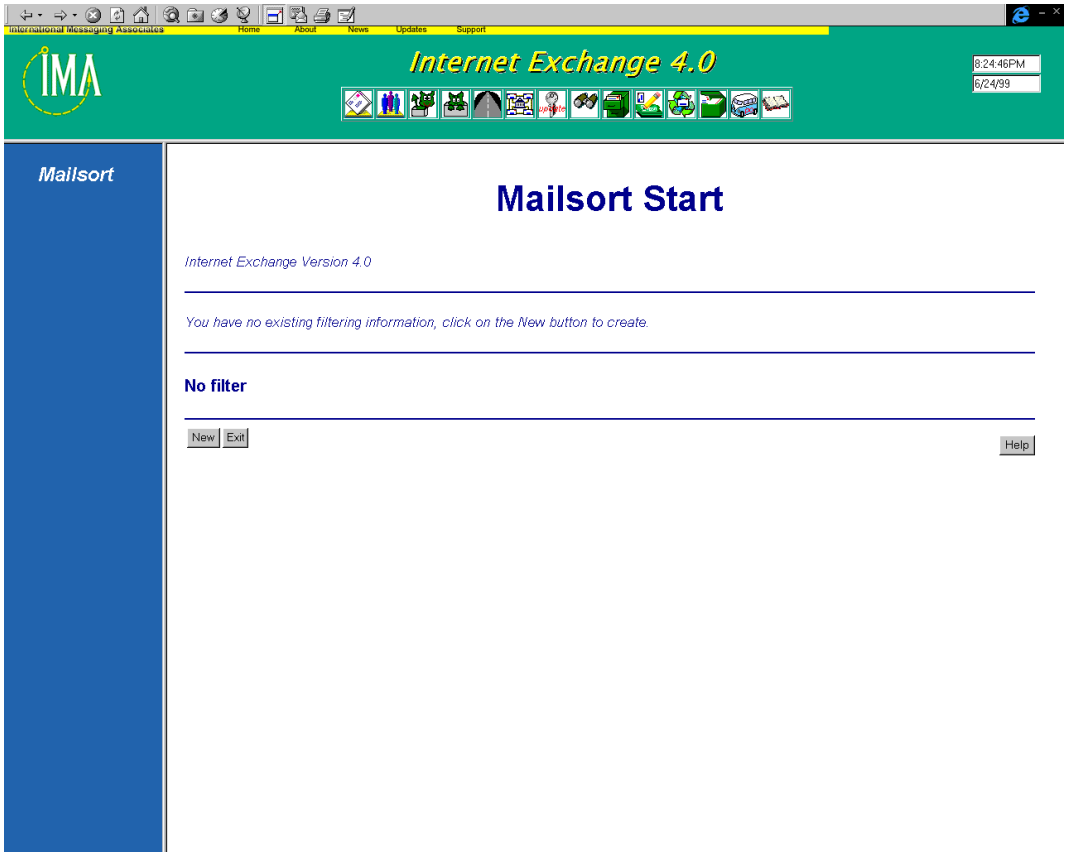


Figure 5o - Create New Filter File

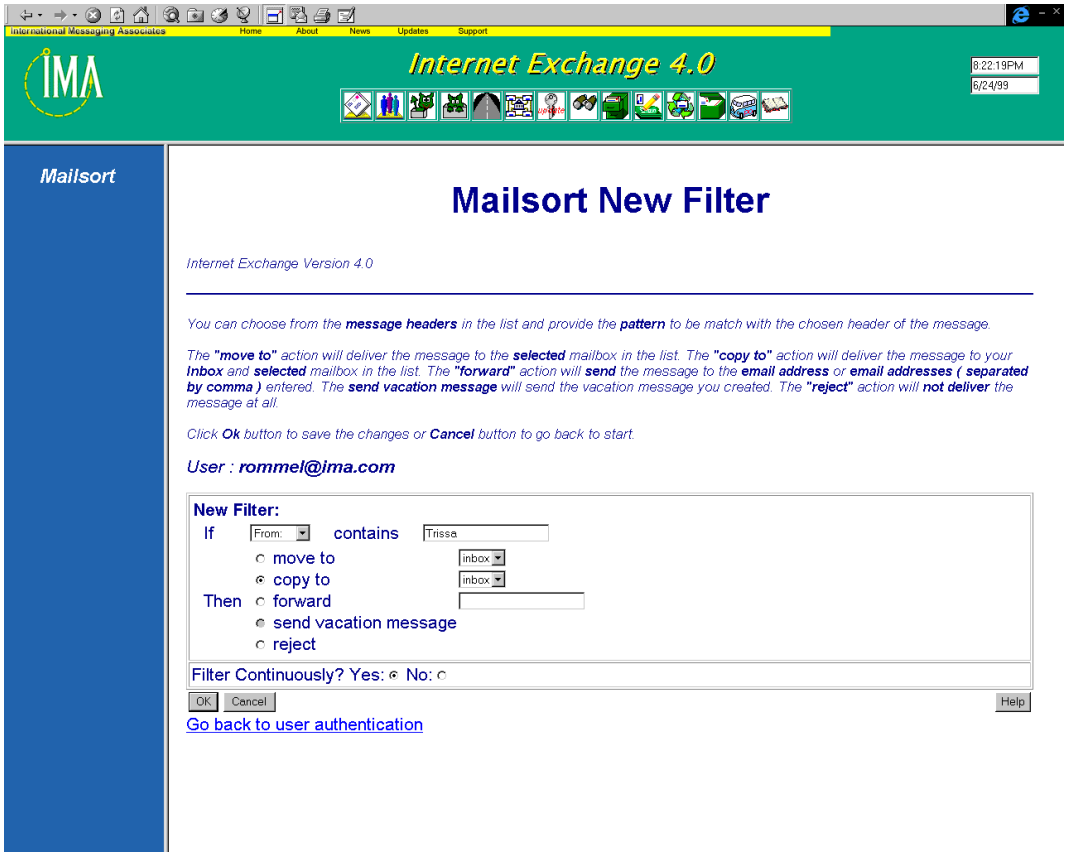


Figure 5p - Create New filter file

To create a new filter block, fill up the text boxes with information that will tell the local mail delivery agent where to send a particular message.

For example, a user may want the local mail delivery agent to deliver all messages with a *From:* field containing *John Doe* to be delivered to the *engr* mailbox (which has been created by the user in the Message Store). To do this*:*

1. Select the *From:* header and enter *John Doe* in the opposite text box.

2. Select the option *move to* and choose the *engr* mailbox from the list provided. The user also has options to copy the message to another mailbox or forward it to another email address. An option to send an automatic reply is available (you must have an existing filter file to activate this option). Activating the *reject* option will tell MailSort to reject the message.

3. Select *Yes/No* to configure filtering action.

4. Click the *OK* button to create a new filter block.

5. To create another filter block, repeat the procedure.

Editing an existing filter file

Information contained in existing filter blocks can be changed or updated using the *MailSort Filter Information* window. Users with existing filter files are automatically brought to this window upon logging on to MailSort. To display and edit a filter block, click on the *Edit* button for that filter block.

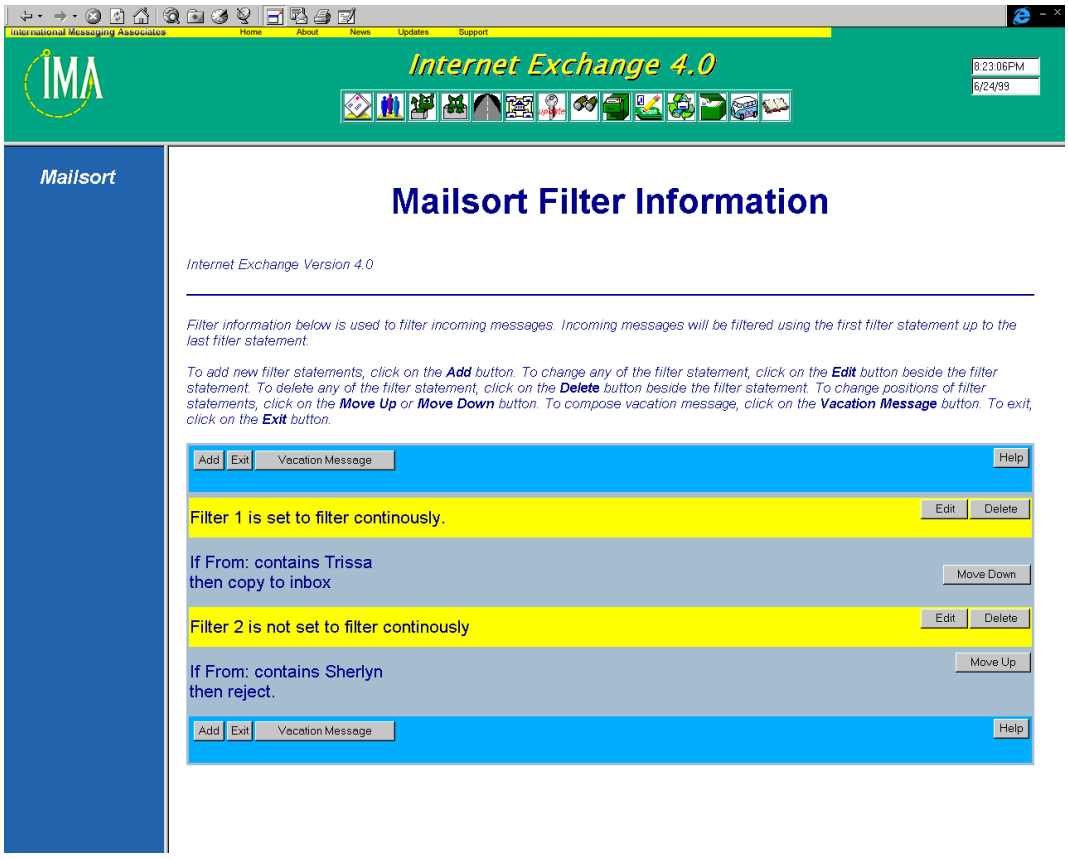


Figure 5q - Mailsort Filter Information.

In the Edit page (see Figure 5r), the user can update the one filter data at a time.

1. Select the header field which the Mailsort engine must scan (i.e. *From:, To:, Cc:, Bcc:, Subject:*) to compare the pattern.

2. In the opposite text box, enter the word or phrase that the Mailsort engine must search for in the selected header.

3. Check the action that you want to be taken by the Mailsort engine for messages that meet the defined criteria (i.e. *move to, copy to, forward to, send vacation message, reject*).

4. Select *Yes/No* to configure filtering action.

5. Click on the *OK* button to save the new filter information for that particular filter block.

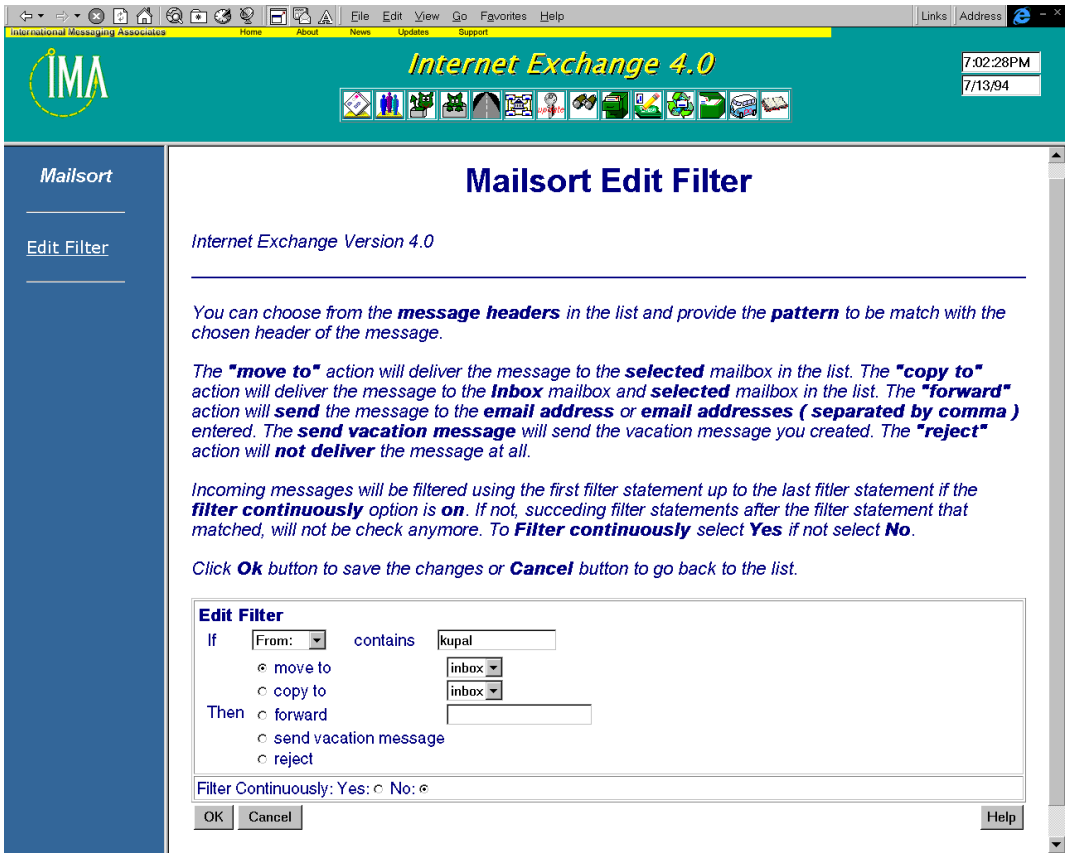


Figure 5r - Web-based interface for editing filter files

Vacation Utility

In the Edit page (Figure 3d), an option to specify a vacation message is available. This feature is useful when an automatic reply needs to be sent to incoming messages when the *Send Vacation Message* is set in the filter block. Click on the *Vacation Message* button on Figure 3d, and the screen shown on (Figure 3f) will be displayed.

The following information needs to be specified for this feature.

Message Subject

Use this field to specify the message subject/header.

Message Body

Use this field to compose the message that needs to be sent out.

NOTE: Vacation messages will not be generated for standard formatted distribution lists. Also, the MailSort Vacation Utility only sends replies to a specific sender every seven days. Thus, if the Vacation Utility has already replied to a sender, it will not send any more messages to that sender until after seven days.

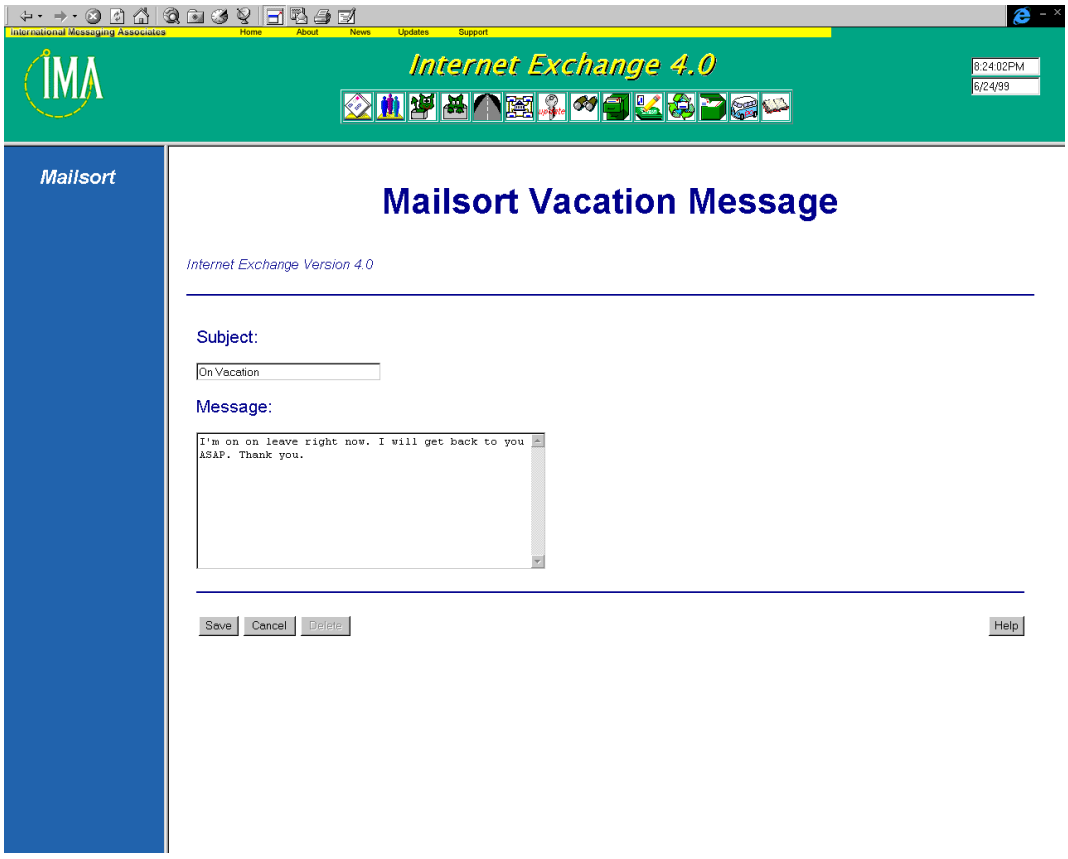


Figure 3f - Mailsort Vacation Message

Click the *Save* button to save the message. This message will be used when replying to incoming messages.

Click the *Cancel* button to cancel the current message being composed.

Click the *Delete* button to delete the saved message.

Error Handling

ERROR HANDLING FOR IMAP4 OPTIMIZED MESSAGE STORE

The following is a list of errors that may be encountered by the Message Store and the possible reasons why they occur:

User database cannot be opened
– User database does not exist in message store home directory.
– User database is corrupted.
– MS_CANNOT_OPEN_USER_DB.

User database is locked
– User database is in use by another process.
– MS_LOCKED_USER_DB.

User name already exists
– A particular user is already defined in user database.
– User name should be unique.
– MS_USER_REC_ALREADY_EXIST.

User name not found
– A particular user is not defined in user database.
– MS_USER_REC_NOT_FOUND.

User name cannot be added
– This error code is the result of MS_CANNOT_OPEN_USER_DB, MS_LOCKED_USER_DB, MS_USER_REC_ALREADY_EXIST, MS_USER_REC_NOT_FOUND.
– MS_CANNOT_ADD_USER_REC.

User name cannot be deleted
– This error code is the result of MS_CANNOT_OPEN_USER_DB, MS_LOCKED_USER_DB, MS_USER_REC_NOT_FOUND.
– MS_CANNOT_DELETE_USER_REC.

User name cannot be updated
– This error code is the result of MS_CANNOT_OPEN_USER_DB, MS_LOCKED_USER_DB, MS_USER_REC_NOT_FOUND.
– MS_CANNOT_UPDATE_USER_REC.

Shared database cannot be opened
– Shared database does not exist in message store home directory.
– Shared database is corrupted.

– MS_CANNOT_OPEN_SHARED_DB.

Shared database is locked
– Shared database is in use by another process.
– MS_LOCKED_SHARED_DB.

Shared name already exists
– A particular shared name is already defined in shared database.
– Shared name should be unique.
– MS_SHARED_REC_ALREADY_EXIST.

Shared name not found
– A particular shared name is not defined in shared database.
– MS_SHARED_REC_NOT_FOUND.

Shared name cannot be added
– This error code is the result of MS_CANNOT_OPEN_SHARED_DB, MS_LOCKED_SHARED_DB, MS_SHARED_REC_ALREADY_EXIST, MS_SHARED_REC_NOT_FOUND.
– MS_CANNOT_ADD_SHARED_REC.

Shared name cannot be deleted
– This error code is the result of MS_CANNOT_OPEN_SHARED_DB, MS_LOCKED_SHARED_DB, MS_SHARED_REC_NOT_FOUND.
– MS_CANNOT_DELETE_SHARED_REC.

Shared name cannot be updated
– This error code is the result of MS_CANNOT_OPEN_SHARED_DB, MS_LOCKED_SHARED_DB, MS_SHARED_REC_NOT_FOUND.
– MS_CANNOT_UPDATE_SHARED_REC.

Mailbox database cannot be opened
– Mailbox database does not exist in message store home directory.
– Mailbox database is corrupted.
– MS_CANNOT_OPEN_MBX_DB.

Mailbox database is locked
– Mailbox database is in use by another process.
– MS_LOCKED_MBX_DB.

Mailbox name already exist
– A particular mailbox name is already defined in mailbox database.
– Mailbox name should be unique.
– MS_MBX_REC_ALREADY_EXIST.

Mailbox name not found
– A particular mailbox name is not defined in mailbox database.
– MS_MBX_REC_NOT_FOUND.

Mailbox name cannot be added
– This error code is the result of MS_CANNOT_OPEN_MBX_DB, MS_LOCKED_MBX_DB, MS_MBX_REC_ALREADY_EXIST, MS_MBX_REC_NOT_FOUND.
– MS_CANNOT_ADD_MBX_REC.

Mailbox name cannot be deleted
– This error code is the result of MS_CANNOT_OPEN_MBX_DB, MS_LOCKED_MBX_DB, MS_MBX_REC_NOT_FOUND.
– MS_CANNOT_DELETE_MBX_REC.

Mailbox name cannot be updated
– This error code is the result of MS_CANNOT_OPEN_MBX_DB, MS_LOCKED_MBX_DB, MS_MBX_REC_NOT_FOUND.
– MS_CANNOT_UPDATE_MBX_REC.

Message status database cannot be opened
– Message status database does not exist in message store home directory.
– Message status database is corrupted.
– MS_CANNOT_OPEN_MSGSTAT_DB.

Message status database is locked
– Message status database is in use by another process.
– MS_LOCKED_MSGSTAT_DB.

Message status ID already exist
– A particular message status ID is already defined in message status database.
– Message status ID should be unique.
– MS_MSGSTAT_REC_ALREADY_EXIST.

Message status ID not found
– A particular message status ID is not defined in message status database.
– MS_MSGSTAT_REC_NOT_FOUND.

Message status ID cannot be added
– This error code is the result of MS_CANNOT_OPEN_MSGSTAT_DB, MS_LOCKED_MSGSTAT_DB, MS_MSGSTAT_REC_ALREADY_EXIST, MS_MSGSTAT_REC_NOT_FOUND.
– MS_CANNOT_ADD_MSGSTAT_REC.

Message status ID cannot be deleted
– This error code is the result of MS_CANNOT_OPEN_MSGSTAT_DB, MS_LOCKED_MSGSTAT_DB, MS_MSGSTAT_REC_NOT_FOUND.
– MS_CANNOT_DELETE_MSGSTAT_REC.

Message status ID cannot be updated
– This error code is the result of MS_CANNOT_OPEN_MSGSTAT_DB, MS_LOCKED_MSGSTAT_DB, MS_MSGSTAT_REC_NOT_FOUND.
– MS_CANNOT_UPDATE_MSGSTAT_REC.

Message envelope database cannot be opened
– Message envelope database does not exist in message store home directory.
– Message envelope database is corrupted.
– MS_CANNOT_OPEN_MSGENV_DB.

Message envelope database is locked
– Message envelope database is in use by another process.
– MS_LOCKED_MSGENV_DB.

Message envelope ID already exist
– A particular message envelope ID is already defined in message envelope database.
– Message envelope ID should be unique.
– MS_MSGENV_REC_ALREADY_EXIST.

Message envelope ID not found
– A particular message envelope ID is not defined in message envelope database.
– MS_MSGENV_REC_NOT_FOUND.

Message envelope ID cannot be added
– This error code is the result of MS_CANNOT_OPEN_MSGENV_DB, MS_LOCKED_MSGENV_DB, MS_MSGENV_REC_ALREADY_EXIST, MS_MSGENV_REC_NOT_FOUND.
– MS_CANNOT_ADD_MSGENV_REC.

Message envelope ID cannot be deleted
– This error code is the result of MS_CANNOT_OPEN_MSGENV_DB, MS_LOCKED_MSGENV_DB, MS_MSGENV_REC_NOT_FOUND.
– MS_CANNOT_DELETE_MSGENV_REC.

Message envelope ID cannot be updated
– This error code is the result of MS_CANNOT_OPEN_MSGENV_DB, MS_LOCKED_MSGENV_DB, MS_MSGENV_REC_NOT_FOUND.
– MS_CANNOT_UPDATE_MSGENV_REC.

Message body database cannot be opened
– Message body database does not exist in message store home directory.
– Message body database is corrupted.
– MS_CANNOT_OPEN_MSGBODY_DB.

Message body database is locked
– Message body database is in use by another process.
– MS_LOCKED_MSGBODY_DB.

Message body ID already exist
– A particular message body ID is already defined in message body database.
– Message body ID should be unique.
– MS_MSGBODY_REC_ALREADY_EXIST.

Message body ID not found
– A particular message body ID is not defined in message body database.
– MS_MSGBODY_REC_NOT_FOUND.

Message body ID cannot be added
– This error code is the result of MS_CANNOT_OPEN_MSGBODY_DB, MS_LOCKED_MSGBODY_DB, MS_MSGBODY_REC_ALREADY_EXIST, MS_MSGBODY_REC_NOT_FOUND.
– MS_CANNOT_ADD_MSGBODY_REC.

Message body ID cannot be deleted
– This error code is the result of MS_CANNOT_OPEN_MSGBODY_DB, MS_LOCKED_MSGBODY_DB, MS_MSGBODY_REC_NOT_FOUND.
– MS_CANNOT_DELETE_MSGBODY_REC.

Message body ID cannot be updated
– This error code is the result of MS_CANNOT_OPEN_MSGBODY_DB, MS_LOCKED_MSGBODY_DB, MS_MSGBODY_REC_NOT_FOUND.
– MS_CANNOT_UPDATE_MSGBODY_REC.

Directory cannot be created
– Directory path does not exist.
– Directory is locked by another process.
– MS_CANNOT_CREATE_DIR.

Directory cannot be deleted
– Directory is locked by another process.
– MS_CANNOT_DELETE_DIR.

Home directory cannot be created
– This error code is the result of MS_CANNOT_CREATE_DIR.
– MS_CANNOT_CREATE_HOMEDIR.

Home directory cannot be deleted
– This error code is the result of MS_CANNOT_DELETE_DIR.
– MS_CANNOT_DELETE_HOMEDIR.

Mailbox directory cannot be created
– This error code is the result of MS_CANNOT_CREATE_DIR.
– MS_CANNOT_CREATE_MBX_DIR.

Message store databases not found
– Message store home directory not specified in IEMTA.INI file.
– Databases not found in home directory.
– MS_SOURCE_DB_NOT_FOUND.

Message store databases cannot be opened
– Message store databases not found in home directory.
– Message store databases are corrupted.

– MS_CANNOT_OPEN_SOURCE_DB.

Message store databases cannot be read
– Message store databases not found in home directory.
– Message store databases are corrupted.
– MS_CANNOT_READ_SOURCE_DB.

Message store databases cannot be copied
– Message store databases not found in home directory.
– Message store databases are corrupted.
– MS_CANNOT_COPY_SOURCE_DB.

Message store databases cannot be created in user/shared home directory
– No disk space.
– MS_CANNOT_CREATE_TARGET_DB.

Message store databases cannot be written to user/shared home directory
– No disk space.
– MS_CANNOT_WRITE_TARGET_DB.

Message file cannot be created in mailbox directory
– No disk space.
– MS_CANNOT_CREATE_MSGFILE.

Message file cannot be deleted in mailbox directory
– Message file is in use by another process.
– MS_CANNOT_DELETE_MSGFILE.

No filter file for user
– User did not define a filter in Mailsort.
– MS_NO_FILTERFILE.

Not enough memory
– Resources are low due to low disk space.
– Several applications are running at the same time.
– MS_NO_MEM.

ERROR HANDLING FOR THE POP3 SERVER

The following is a list of errors that may be encountered by the POP3 Server and the possible reasons why they occur:

Unable to connect to the server
– Maximum number of client connections has been reached.
– Network error.
– Memory allocation failure.

Unable to log in to the server
– Incorrect user name or password.
– Database error.
– Memory allocation failure.

Unable to open incoming mailbox
– Mailbox is in use or cannot be locked.
– Database error.
– Memory allocation failure.

Unable to obtain INBOX status
– Database error.
– Memory allocation failure.

Unable to fetch message information / text
– Database error.
– Memory allocation failure.

Unable to delete / expunge message
– Database error.
– Memory allocation failure.

Lost connection
– Auto-logout timer has expired.
– Server thread crashed.
– Network error.

ERROR HANDLING FOR THE IMAP4 SERVER

The following is a list of errors that may be encountered by the IMAP4 Server and the possible reasons why they occur:

Unable to connect to the server
– Maximum number of client connections has been reached.
– Network error.
– Memory allocation failure.

Unable to log in to the server
– Incorrect user name or password.
– Database error.
– Memory allocation failure.

Unable to open a mailbox
– Mailbox does not exist.
– Database error.
– Memory allocation failure.

Unable to create a mailbox
– Invalid mailbox name.
– Mailbox already exists.
– Database error.
– Memory allocation failure.

Unable to delete a mailbox
– INBOX being deleted.
– Mailbox does not exist.
– Database error.
– Memory allocation failure.

Unable to rename a mailbox
– Mailbox to be renamed does not exist.
– Mailbox to rename to already exists.
– Database error.
– Memory allocation failure.

Unable to obtain a listing of mailboxes
– Database error.
– Memory allocation failure.

Unable to obtain mailbox status
– Mailbox does not exist.
– Database error.
– Memory allocation failure.

Unable to append a message to a mailbox
– Mailbox does not exist.
– Error in flags, date/time or message text.
– Database error.
– Memory allocation failure.

Unable to expunge a message
– Database error.
– Memory allocation failure.

Unable to fetch message information / text
– Database error.
– Memory allocation failure.

Unable to change / update message flags
– Database error.
– Memory allocation failure.

Unable to copy a message from one mailbox to another
– Message has been expunged.
– Mailbox does not exist.
– Database error.

– Memory allocation failure.

Message parsing error
– Unrecognized element in a message's RFC-822 or MIME-IMB header is encountered.

Message search failure
– Database error.
– Memory allocation failure.

Lost connection
– Auto-logout timer has expired.
– Server thread crashed.
– Network error.

ERROR HANDLING FOR THE LOCAL MAIL DELIVERY AGENT

The following is a list of errors that may be encountered by the Message Store's Local Mail Delivery Agent module and the possible reasons why they occur:

Message queue cannot be initialized
– Preprocessor is not running.
– LM_CANNOT_INIT_MQ.

Message queue cannot be opened
– Preprocessor is not running.
– LDAP server is not running.
– LM_CANNOT_OPEN_MQ.

Not enough memory
– Several applications are running.
– LM_NO_MEM.

Message queue envelope is invalid
– Message queue cannot open the message file.
– LM_MQ_MSGFILE_NOT_FOUND.

Message queue message file is empty
– An error was encountered during creation in Message Queue module.
– LM_MQ_MSGFILE_EMPTY.

Message queue message file cannot be opened
– MQ message file is corrupted.
– LM_CANNOT_OPEN_MQ_MSGFILE.

Message queue message file cannot be read
– MQ message file is not open.
– LM_CANNOT_READ_MQ_MSGFILE.

Message header is invalid
– Message header file is empty.
– LM_INVALID_MSGHDR.

Message envelope is invalid
– Message file is not an RFC-822 conformant.
– LM_INVALID_MSGENV.

Message body is invalid
– Message file is not an RFC-822 conformant.
– LM_INVALID_MSGBODY.

Local mail cannot connect to LDAP server
– LDAP server maybe down.
– LDAP section in INI file points to invalid server name.
– LM_CANNOT_CONNECT_TO_LDAP_SERVER.

ERROR HANDLING FOR THE MAILSORT ENGINE

The following is a list of errors that may be encountered by the Message Store's Mailsort utility and the possible reasons why they occur:

Message is corrupted
– Message being delivered by local mail delivery agent contains an invalid header format (RFC822).

Local Message Store user not found
– The recipient of the message does not exist in Local Message Store.

Local Message Store shared mailbox account not found
– The shared mailbox account does not exist in Local Message Store.

Error in opening the filter file (filter.txt)
– An error occurred while opening the filter file.

Error in reading filter file
– An error occurred while reading the filter file.

Syntax error in the filter file
– The information in the filter file is not valid.

Corrupted Filter Information
– The filter file in memory has been corrupted.

System Out of Memory
– The system runs out of memory.

ERROR HANDLING FOR THE MAILSORT WEB-BASED INTERFACE

The following is a list of errors that may be encountered by the MailSort Web-based interface and the possible reasons why they occur:

No user name entered
– The user did not enter any username.

No password entered
– The user did not enter any password.

User not found
– The user is not registered in the Local Message Store.

User not authorized
– The user entered an invalid password.

Filter file (filter.txt) does not exist
– The filter file (filter.txt) that contains the filter information does not exist.

Error in reading filter file
– An error occurred while reading the filter file.

Syntax error in the filter file
– The information in the filter file is not valid.

Corrupted Filter Information
– The filter file in memory has been corrupted.

System Out of Memory
– The system run out of memory.

Out of disk space
– The system runs out disk space to save the filter file.

PART 4

Troubleshooting

Troubleshooting Tools

TROUBLESHOOTING THE POP3 SERVER

Error messages related to a server shutdown or the inability of a client to log in to the POP3 Server are logged on the server screen. On the other hand, once a client has been able to log in to the POP3 Server, connection requests from the client are logged in a file located in the log subdirectory of the message store home directory. If no error is encountered for the entire duration of a particular client connection, the log file for that client is removed from the log subdirectory when the client logs out. Otherwise, if any error is encountered while processing a client request during a connection, the client log file remains in the log subdirectory until manually deleted or reused by succeeding client connections.

Log files for POP3 clients are named PXXX.log, where the X's represent a sequence of one or more digits. In order to determine the cause of errors in a particular client transaction, one would have to locate the file containing the name used to log in to the server in the first line of the file. If there are several such files, the file time-stamp should be used to pick out the correct log file.

The second up to the last line of the POP3 log file contains the sequence of commands issued by the POP3 client intermixed with any error responses generated by the server. Server response lines may be distinguished from client command lines by a leading asterisk ('*'). Based on the logged server responses, which may also be displayed by the POP3 client on its screen, appropriate action may then be taken to remedy an error.

TROUBLESHOOTING THE IMAP4 SERVER

Error messages related to a server shutdown or the inability of a client to log in to the IMAP4 Server are logged on the server screen. On the other hand, once a client has been able to log in to the IMAP4 Server, connection requests from the client are logged in a file located in the log subdirectory of the message store home directory. If no error is encountered for the entire duration of a particular client connection, the log file for that client is removed from the log subdirectory when the client logs out. Otherwise, if any error is encountered while processing a client request during a connection, the client log file remains in the log subdirectory until manually deleted or reused by succeeding client connections.

Log files for IMAP4 clients are named *IXXX.log*, where the X's represent a sequence of one or more digits. In order to determine the cause of errors in a particular client transaction, one would have to locate the file containing the name used to log in to the server in the first line of the file. If there are several such files, the file time-stamp should be used to pick out the correct log file.

The second up to the last line of the IMAP4 log file contains the sequence of commands issued by the IMAP4 client intermixed with any error responses generated by the server. Client command lines start with an alphanumeric string, while server response lines start with an asterisk ('*'). Based on the logged server responses, which may also be displayed by the IMAP4 client on its screen, appropriate action may then be taken to remedy an error.

TROUBLESHOOTING THE MAILSORT ENGINE

The system administrator (sysadm) can check the log files produced when errors occurred. The log files contains the last action performed by the system before a particular error occurred. Through these log files, the sysadm can trace which modules or even function of a module the error showed up.

Log files are in text mode. Ordinary text editors such as "Notepad" or "Wordpad" can be used to open these log files.

TROUBLESHOOTING THE LOCAL MAIL DELIVERY AGENT

Local Mail Delivery Agent module uses log file to record all transactions during delivery of local messages. All Internet Exchange 4 modules use this log file. The administrator can view this file named *IEMTA.LOG* for troubleshooting purposes.

Please refer to Error Handling section for the error messages logged by the Local Mail Delivery Agent.

PART 5

Appendices

Appendix A

Internet Standards

POST OFFICE PROTOCOL VERSION 3 (POP3)

The Post Office Protocol version 3 (POP3), as defined in RFC 1939, is an "off-line" protocol designed to provide the Internet community with a tool for connecting to mail servers and retrieving email messages. Since the Simple Mail Transfer Protocol (SMTP) functions only as a transport protocol between mail servers, it is not capable of delivering mail from the post office to a users' mailbox. It also does not allow remote users to get mail from the server. Thus, a delivery protocol like POP3 is needed so that the mail received by the server from the Internet can be delivered to their intended recipients.

As an off-line protocol, POP3 retrieves messages from the server (either at an ISP or on a LAN) and transfers them to a workstation's hard disk. It deletes the messages from the server if this option is explicitly mentioned in the configuration. In short, POP3's function is to get email from a remote mailbox and store it on a user's local machine so it can be read later in a disconnected or "off-line" state. POP3 is designed as a single user, single mailbox system (one account per user). The POP3 connector is usually a combined POP3 retriever and SMTP sender.

Using POP3, smaller nodes in the Internet that are incapable of maintaining a message transport system (MTS) can retrieve mail. An example of such nodes are workstations that do not have the resources needed to maintain an SMTP server and associated mail delivery system. Another example is a personal computer that connects to the Internet only at certain times due to economic reasons. A node capable of supporting an MTS can offer maildrop service to these smaller nodes via POP3 service. Through this setup, a workstation can dynamically access a maildrop on a host that offers the POP3 service, otherwise known as the server host. The host that makes use of the POP3 service is called the client host.

The server host starts the POP3 service by listening to TCP port 110. Users who want to retrieve their mail must log in to the server host POP3 service at this port with their account names and passwords. The client host wishing to use the POP3 service establishes a TCP connection with the server host. After the connection has been made, the POP3 server sends a greeting. The client and the POP3 server then issue commands and responses until the connection is terminated.

A POP3 session goes through a number of states during its lifetime. After TCP connection has been established and the POP3 server has sent a greeting, the session enters the AUTHORIZATION state. While in this state, the client identifies itself to the POP3 server. After the this stage, the server acquires resources associated with the client's maildrop. The session then enters the TRANSACTION state, wherein the client requests actions on the part of the server. Once the client has issued the QUIT command, the POP3 server releases all the resources acquired during the TRANSACTION state and says good-

bye. The TCP connection is then terminated.

When a client gives an unrecognized, un-implemented or syntactically invalid command, the server automatically responds with a negative status indicator. The same response is given when a command is issued in an incorrect state. There is no general method for a client to distinguish between a server that does not implement an optional command and another server that is unable or unwilling to process the command.

A POP3 server may have an inactivity autologout timer. Such a timer must be set to a duration of at least 10 minutes. If the server receives a command from the client during that interval, the autologout timer is reset. When the timer expires, the session is prevented from entering the UPDATE state, that is the server terminates the TCP connection without removing any messages or sending any response to the client.

INTERNET MAIL ACCESS PROTOCOL VERSION 4 (IMAP4)

RFC 2060, which obsoletes RFC 1730, defines the Internet Message Access Protocol Version 4 (IMAP4) as a protocol that allows manipulation of remote message folders, known as mailboxes, in a manner that is functionally equivalent to local mail folders. Unlike the POP3 protocol, IMAP4 has a number of features which make it more user-friendly. Whereas POP3 is designed primarily to function as an off-line protocol, IMAP4 is used primarily as an on-line protocol, but can also support an off-line mode as well.

In the on-line mode of IMAP4, the user manipulates messages/message folders on the server without having to download them to a local hard disk. It also supports efficient folder management by allowing users to create multi-level folders or mailboxes on the server (with proper authorization from the administrator) that can easily be renamed, moved, or deleted by them. Folder and message updates made through a IMAP4 client will be seen by other clients accessing the same message store.

These features are particularly useful to multiple-computer users. In IMAP4, a user with different computers – say a PC at home, Mac in the office, and a laptop on the road – can move freely among them and access the same message store or post office on the server. Moreover, IMAP4 allows multiple users to access a shared mailbox on the server from multiple platforms concurrently and view the actual status of each message.

Unlike POP3, IMAP4 has the ability to search for messages on the server using various message attributes such as message size, headers, and message sender, among others. It can also separate attached files from the text and header portions of the message. This is particularly useful for handling multipart MIME messages. In POP3, a message must be downloaded in its entirety, including unwanted embedded attachments, before they can be read. IMAP4 gives users the capability to retrieve the header or certain portions of a multipart MIME message first. Using this feature, users can scan the header information on a MIME message and decide which part(s) of the message they need.
Another advantage of IMAP4 over POP3 is that the former has a built-in security feature that prevents a user's password from being transmitted in the clear over the network. It uses an authentication mechanism that is based on a series of server challenges and client answers. POP3 has a similar feature invoked by the APOP command. However, very few email clients support this feature.

Appendix B

Request for Comments (RFC's)

The Internet Exchange Message Store complies with a number of RFC's to ensure optimum compatibility with proven Internet standards that guarantee safe and trouble-free message delivery.

REQUEST FOR COMMENTS: 2342 IMAP4 NAMESPACE

IMAP4 [RFC-2060] does not define a default server namespace. As a result, two common namespace models have evolved:

The Personal Mailbox model
In which the default namespace that is presented consists of only the user's personal mailboxes. To access shared mailboxes, the user must use an escape mechanism to reach another namespace.

The Complete Hierarchy model
In which the default namespace that is presented includes the user's personal mailboxes along with any other mailboxes they have access to.

These two models, create difficulties for certain client operations. This document defines a NAMESPACE command that allows a client to discover the prefixes of namespaces used by a server for personal mailboxes, other users' mailboxes, and shared mailboxes. This allows a client to avoid much of the manual user configuration that is now necessary when mixing and matching IMAP4 clients and servers.

Personal Namespace
A namespace that the server considers within the personal scope of the authenticated user on a particular connection. Typically, only the authenticated user has access to mailboxes in their Personal Namespace. It is the part of the namespace that belongs to the user that is allocated for mailboxes. If an INBOX exists for a user, it MUST appear within the user's personal namespace. In the typical case, there SHOULD be only one Personal Namespace on a server.

Other Users' Namespace
A namespace that consists of mailboxes from the Personal Namespaces of other users. To access mailboxes in the Other Users' Namespace, the currently authenticated user MUST be explicitly granted access rights. For example, it is common for a manager to grant to their secretary access rights to their mailbox.In the typical case, there SHOULD be only one Other Users' Namespace on a server.

Shared Namespace
A namespace that consists of mailboxes that are intended to be shared amongst users and do not exist within a user's Personal Namespace.

Clients often attempt to create mailboxes for such purposes as maintaining a record of sent messages (e.g. "Sent Mail") or temporarily saving messages being composed (e.g. "Drafts"). For these clients to inter-operate correctly with the variety of IMAP4 servers available, the user must enter the prefix of the Personal Namespace used by the server. Using the NAMESPACE command, a client is able to automatically discover this prefix without manual user configuration.

In addition, users are often required to manually enter the prefixes of various namespaces in order to view the mailboxes located there. For example, they might be required to enter the prefix of #shared to view the shared mailboxes namespace. The NAMESPACE command allows a client to automatically discover the namespaces that are available on a server. This allows a client to present the available namespaces to the user in what ever manner it deems appropriate. For example, a client could choose to initially display only personal mailboxes, or it may choose to display the complete list of mailboxes available, and initially position the user at the root of their Personal Namespace.

A server MAY choose to make available to the NAMESPACE command only a subset of the complete set of namespaces the server supports. To provide the ability to access these namespaces, a client SHOULD allow the user the ability to manually enter a namespace prefix.

REQUEST FOR COMMENTS: 2060
INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4REV1

The Internet Message Access Protocol, Version 4rev1 (IMAP4rev1) allows a client to access and manipulate electronic mail messages on a server. IMAP4rev1 permits manipulation of remote message folders, called “mailboxes”, in a way that is functionally equivalent to local mailboxes. IMAP4rev1 also provides the capability for an offline client to resynchronize with the server. IMAP4rev1 supports a single server. A mechanism for accessing configuration information to support multiple IMAP4rev1 servers is discussed in [ACAP].

IMAP4rev1 includes operations for creating, deleting, and renaming mailboxes; checking for new messages; permanently removing messages; setting and clearing flags; parsing; searching; and selective fetching of message attributes, texts, and portions thereof. Messages in IMAP4rev1 are accessed by the use of numbers. These numbers are either message sequence numbers or unique identifiers.

IMAP4rev1 does not specify a means of posting mail; this function is handled by a mail transfer protocol such as SMTP[RFC 821]. IMAP4rev1 is designed to be upwards compatible from the IMAP2[RFC 1176] and unpublished IMAP2bis protocols.

IMAP4rev1 protocol transactions, including electronic mail data, are sent in the clear over

the network unless privacy protection is negotiated in the AUTHENTICATE command. A server error message for an AUTHENTICATE command which fails due to invalid credentials SHOULD NOT detail why the credentials are invalid.

Use of the LOGIN command sends passwords in the clear. This can be avoided by using the AUTHENTICATE command instead. A server error message for a failing LOGIN command SHOULD NOT specify that the user name, as opposed to the password, is invalid. Additional security considerations are discussed in the section discussing the AUTHENTICATE and LOGIN commands.

References

[ACAP] Myers, J. “ACAP -- Application Configuration Access Protocol", Work in Progress.

REQUEST FOR COMMENTS: 2061 IMAP4 COMPATIBILITY WITH IMAP2BIS

The Internet Message Access Protocol (IMAP) has been through several revisions and variants in its 10-year history. Many of these are either extinct or extremely rare; in particular, several undocumented variants and the variants described in RFC 1064, RFC 1176, and RFC 1203 fall into this category.

One variant, IMAP2bis, is at the time of this writing very common and has been widely distributed with the Pine mailer. Unfortunately, there is no definite document describing IMAP2bis. This document is intended to be read along with RFC 1176 and the most recent IMAP4 specification (RFC 2060) to assist implementors in creating an IMAP4 implementation to interoperate with implementations that conform to earlier specifications. Nothing in this document is required by the IMAP4 specification; implementors must decide for themselves whether they want their implementation to fail if it encounters old software. An implementor who wishes to interoperate with both RFC 1730 and RFC 2060 should refer to both documents. For detailed information on interoperating with other old variants, refer to RFC 1732.

IMAP4 client interoperability with IMAP2bis servers

A quick way to check whether a server implementation supports the IMAP4 specification is to try the CAPABILITY command. An OK response will indicate which variant(s) of IMAP4 are supported by the server. If the client does not find any of its known variant in the response, it should treat the server as IMAP2bis. A BAD response indicates an IMAP2bis or older server.

Most IMAP4 facilities are in IMAP2bis. The following exceptions exist:

- CAPABILITY command
- AUTHENTICATE command
- LSUB, SUBSCRIBE, and UNSUBSCRIBE commands
- LIST command
- SEARCH extensions (character set, additional criteria)
- BODYSTRUCTURE fetch data item

- BODY.PEEK[section]
- FLAGS.SILENT, +FLAGS.SILENT, and -FLAGS.SILENT store data items
- UID fetch data item and the UID commands
- CLOSE command

IMAP4 server interoperability with IMAP2bis clients

The only interoperability problem between an IMAP4 server and a well-written IMAP2bis client is an incompatibility with the use of "\" in quoted strings. This is best avoided by using literals instead of quoted strings if "\" or <"> is embedded in the string.

REQUEST FOR COMMENTS: 2062
INTERNET MESSAGE ACCESS PROTOCOL - OBSOLETE SYNTAX

This document describes obsolete syntax which may be encountered by IMAP4 implementations which deal with older versions of the Internet Mail Access Protocol. IMAP4 implementations MAY implement this syntax in order to maximize interoperability with older implementations.

Obsolete Commands and Fetch Data Items

The following commands are OBSOLETE. It is NOT required to support any of these commands or fetch data items in new server implementations. These commands are documented here for the benefit of implementors who may wish to support them for compatibility with old client implementations.

Obsolete Commands

- FIND ALL.MAILBOXES
- FIND MAILBOXES
- SUBSCRIBE MAILBOX
- UNSUBSCRIBE MAILBOX
- PARTIAL

Obsolete FETCH Data Items

- BODY[<...>0]
- RFC822.HEADER.LINES
- RFC822.HEADER.LINES.NOT
- RFC822.PEEK
- RFC822.TEXT.PEEK

Obsolete Responses

- MAILBOX
- COPY
- STORE

REQUEST FOR COMMENTS: 2177
IMAP4 IDLE COMMAND

The Internet Message Access Protocol [IMAP4] requires a client to poll the server for changes to the selected mailbox (new mail, deletions). It's often more desirable to have the server transmit updates to the client in real time. This allows a user to see new mail immediately. It also helps some real-time applications based on IMAP, which might otherwise need to poll extremely often (such as every few seconds).

This document specifies the syntax of an IDLE command, which will allow a client to tell the server that it's ready to accept such real-time updates.

The IDLE command may be used with any IMAP4 server implementation that returns "IDLE" as one of the supported capabilities to the CAPABILITY command. If the server does not advertise the IDLE capability, the client MUST NOT use the IDLE command and must poll for mailbox updates.

In particular, the client MUST continue to be able to accept unsolicited untagged responses to ANY command, as specified in the base IMAP specification. The IDLE command is sent from the client to the server when the client is ready to accept unsolicited mailbox update messages. The server requests a response to the IDLE command using the continuation ("+"), response. The IDLE command remains active until the client responds to the continuation, and as long as an IDLE command is active, the server is now free to send untagged EXISTS, EXPUNGE, and other messages at any time.

The IDLE command is terminated by the receipt of a "DONE" continuation from the client; such response satisfies the server's continuation request. At that point, the server MAY send any remaining queued untagged responses and then MUST immediately send the tagged response to the IDLE command and prepare to process other commands. As in the base specification, the processing of any new command may cause the sending of unsolicited untagged responses, subject to the ambiguity limitations. The client MUST NOT send a command while the server is waiting for the DONE, since the server will not be able to distinguish a command from a continuation.

The server MAY consider a client inactive if it has an IDLE command running, and if such a server has an inactivity time-out it MAY log the client off implicitly at the end of its time-out period. Because of that, clients using IDLE are advised to terminate the IDLE and re-issue it at least every 29 minutes to avoid being logged off. This still allows a client to receive immediate mailbox updates even though it need only "poll" at half hour intervals.

REQUEST FOR COMMENTS: 2180
IMAP4 MULTI-ACCESSED MAILBOX PRACTICE

IMAP4[RFC-2060] is a rich client/server protocol that allows a client to access and manipulate electronic mail messages on a server. Within the protocol framework, it is possible to have differing results for particular client/server interactions. If a protocol does not allow for this, it is often unduly restrictive.

For example, when multiple clients are accessing a mailbox and one attempts to delete the mailbox, an IMAP4 server may choose to implement a solution based upon server architectural constraints or individual preference. With this flexibility comes greater client responsibility. It is not sufficient for a client to be written based upon the behavior of a particular IMAP server. Rather the client must be based upon the behavior allowed by the protocol.

By documenting common IMAP4 server practice for the case of simultaneous client access to a mailbox, we hope to ensure the widest amount of inter-operation between IMAP4 clients and servers. The behavior described in this document reflects the practice of some existing servers or behavior that the consensus of the IMAP mailing list has deemed to be reasonable. The behavior described within this document is believed to be [RFC-2060] compliant. However, this document is not meant to define IMAP4 compliance, nor is it an exhaustive list of valid IMAP4 behavior. [RFC-2060] must always be consulted to determine IMAP4 compliance, especially for server behavior not described within this document.

Deletion/Renaming of a multi-accessed mailbox

If an external agent or multiple clients are accessing a mailbox, care must be taken when handling the deletion or renaming of the mailbox. Following are some strategies an IMAP server may choose to use when dealing with this situation.

The server MAY fail the DELETE/RENAME command of a multi-accessed mailbox. In some cases, this behavior may not be practical. For example, if a large number of clients are accessing a shared mailbox, the window in which no clients have the mailbox accessed may be small or non- existent, effectively rendering the mailbox undeletable or unrenamable.

The server MAY allow the DELETE command of a multi-accessed mailbox, but keep the information in the mailbox available for those clients that currently have access to the mailbox. When all clients have finished accessing the mailbox, it is permanently removed. For clients that do not already have access to the mailbox, the 'ghosted' mailbox would not be available. For example, it would not be returned to these clients in a subsequent LIST or LSUB command and would not be a valid mailbox argument to any other IMAP command until the reference count of clients accessing the mailbox reached 0.

In some cases, this behavior may not be desirable. For example, if someone created a mailbox with offensive or sensitive information, one might prefer to have the mailbox deleted and all access to the information contained within removed immediately, rather than continuing to allow access until the client closes the mailbox. Furthermore, this behavior, may prevent 'recycling' of the same mailbox name until all clients have finished

accessing the original mailbox.

The server MAY allow the DELETE/RENAME of a multi-accessed mailbox, but disconnect all other clients who have the mailbox accessed by sending a untagged BYE response. A server may often choose to disconnect clients in the DELETE case, but may choose to implement a "friendlier" method for the RENAME case.

The server MAY allow the RENAME of a multi-accessed mailbox by simply changing the name attribute on the mailbox. Other clients that have access to the mailbox can continue issuing commands such as FETCH that do not reference the mailbox name. Clients would discover the renaming the next time they referred to the old mailbox name. Some servers MAY choose to include the [NEWNAME] response code in their tagged NO response to a command that contained the old mailbox name, as a hint to the client that the operation can succeed if the command is issued with the new mailbox name.

Expunging of messages on a multi-accessed mailbox

If an external agent or multiple clients are accessing a mailbox, care must be taken when handling the EXPUNGE of messages. Other clients accessing the mailbox may be in the midst of issuing a command that depends upon message sequence numbers. Because an EXPUNGE response can not be sent while responding to a FETCH, STORE or SEARCH command, it is not possible to immediately notify the client of the EXPUNGE. This can result in ambiguity if the client issues a FETCH, STORE or SEARCH operation on a message that has been EXPUNGED.

Fetching of expunged messages

The server MAY allow the EXPUNGE of a multi-accessed mailbox but keep the messages available to satisfy subsequent FETCH commands until it is able to send an EXPUNGE response to each client.

In some cases, the behavior of keeping "ghosted" messages may not be desirable. For example if a message contained offensive or sensitive information, one might prefer to instantaneously remove all access to the information, regardless of whether another client is in the midst of accessing it.

The server MAY allow the EXPUNGE of a multi-accessed mailbox, and on subsequent FETCH commands return FETCH responses only for non-expunged messages and a tagged NO. After receiving a tagged NO FETCH response, the client SHOULD issue a NOOP command so that it will be informed of any pending EXPUNGE responses. The client may then either reissue the failed FETCH command, or by examining the EXPUNGE response from the NOOP and the FETCH response from the FETCH, determine that the FETCH failed because of pending expunges.

The server MAY allow the EXPUNGE of a multi-accessed mailbox, and on subsequent FETCH commands return the usual FETCH responses for non-expunged messages, "NIL FETCH Responses" for expunged messages, and a tagged OK response.

If all of the messages in the subsequent FETCH command have been expunged, the server SHOULD return only a tagged NO. In this case, the client SHOULD issue a NOOP command so that it will be informed of any pending EXPUNGE responses. The client may

then either reissue the failed FETCH command, or by examining the EXPUNGE response from the NOOP, determine that the FETCH failed because of pending expunges.

REQUEST FOR COMMENTS: 2192
IMAP URL SCHEME

IMAP [IMAP4] is a rich protocol for accessing remote message stores. It provides an ideal mechanism for accessing public mailing list archives as well as private and shared message stores. This document defines a URL scheme for referencing objects on an IMAP server.

The IMAP URL scheme is used to designate IMAP servers, mailboxes, messages, MIME bodies [MIME], and search programs on Internet hosts accessible using the IMAP protocol. The IMAP URL follows the common Internet scheme syntax as defined in RFC 1738 [BASIC-URL] except that clear text passwords are not permitted. If :<port> is omitted, the port defaults to 143.

An IMAP URL takes one of the following forms:

imap://<iserver>/
imap://<iserver>/<enc_list_mailbox>;TYPE=<list_type>
imap://<iserver>/<enc_mailbox>[uidvalidity][?<enc_search>]
imap://<iserver>/<enc_mailbox>[uidvalidity]<iuid>[isection]

The first form is used to refer to an IMAP server, the second form refers to a list of mailboxes, the third form refers to the contents of a mailbox or a set of messages resulting from a search, and the final form refers to a specific message or message part.

IMAP User Name and Authentication Mechanism

A user name and/or authentication mechanism may be supplied. They are used in the "LOGIN" or "AUTHENTICATE" commands after making the connection to the IMAP server. If no user name or authentication mechanism is supplied, the user name "anonymous" is used with the "LOGIN" command and the password is supplied as the Internet e-mail address of the end user accessing the resource. If the URL doesn't supply a user name, the program interpreting the IMAP URL SHOULD request one from the user if necessary.

Since URLs can easily come from untrusted sources, care must be taken when resolving a URL which requires or requests any sort of authentication. If authentication credentials are supplied to the wrong server, it may compromise the security of the user's account. The program resolving the URL should make sure it meets at least one of the following criteria in this case:

• The URL comes from a trusted source, such as a referral server which the client has validated and trusts according to site policy. Note that user entry of the URL may or may not count as a trusted source, depending on the experience level of the user and site policy.

• Explicit local site policy permits the client to connect to the server in the URL. For example, if the client knows the site domain name, site policy may dictate that any hostname ending in that domain is trusted.

• The user confirms that connecting to that domain name with the specified credentials and/or mechanism is permitted.
• A mechanism is used which validates the server before passing potentially compromising client credentials.
• An authentication mechanism is used which will not reveal information to the server which could be used to compromise future connections.

IMAP server

An IMAP URL referring to an IMAP server has the following form:

imap://<iserver>/

A program interpreting this URL would issue the standard set of commands it uses to present a view of the contents of an IMAP server. This is likely to be semantically equivalent to one of the following URLs:

imap://<iserver>/;TYPE=LIST
imap://<iserver>/;TYPE=LSUB

The program interpreting this URL SHOULD use the LSUB form if it supports mailbox subscriptions.

Lists of mailboxes

An IMAP URL referring to a list of mailboxes has the following form:

imap://<iserver>/<enc_list_mailbox>;TYPE=<list_type>

Lists of messages

An IMAP URL referring to a list of messages has the following form:

imap://<iserver>/<enc_mailbox>[uidvalidity][?<enc_search>]

A specific message or message part

An IMAP URL referring to a specific message or message part has the following form:

imap://<iserver>/<enc_mailbox>[uidvalidity]<iuid>[isection]

Relative IMAP URLs

Relative IMAP URLs are permitted and are resolved according to the rules defined in RFC 1808 with one exception. In IMAP URLs, parameters are treated as part of the normal path with respect to relative URL resolution. This is believed to be the behavior of the installed base and is likely to be documented in a future revision of the relative URL specification.

The following observations are also important:
The <iauth> grammar element is considered part of the user name for purposes of resolving relative IMAP URLs. This means that unless a new login/server specification is included in the relative URL, the authentication mechanism is inherited from a base IMAP URL. URLs always use "/" as the hierarchy delimiter for the purpose of resolving paths in relative URLs. IMAP4 permits the use of any hierarchy delimiter in mailbox names. For this reason, relative mailbox paths will only work if the mailbox uses “/” as the hierarchy delimiter. Relative URLs may be used on mailboxes which use other delimiters, but in that case, the entire mailbox name MUST be specified in the relative URL or inherited as a whole from the base URL.

The base URL for a list of mailboxes or messages which was referred to by an IMAP URL is always the referring IMAP URL itself. The base URL for a message or message part which was referred to by an IMAP URL may be more complicated to determine. The program interpreting the relative URL will have to check the headers of the MIME entity and any enclosing MIME entities in order to locate the "Content-Base" and "Content-Location" headers.

REQUEST FOR COMMENTS: 1939
POST OFFICE PROTOCOL - VERSION 3

On certain types of smaller nodes in the Internet it is often impractical to maintain a message transport system (MTS). For example, a workstation may not have sufficient resources (cycles, disk space) in order to permit a SMTP server [RFC821] and associated local mail delivery system to be kept resident and continuously running. Similarly, it may be expensive (or impossible) to keep a personal computer interconnected to an IP-style network for long amounts of time (the node is lacking the resource known as "connectivity").

Despite this, it is often very useful to be able to manage mail on these smaller nodes, and they often support a user agent (UA) to aid the tasks of mail handling. To solve this problem, a node which can support an MTS entity offers a maildrop service to these less endowed nodes. The Post Office Protocol - Version 3 (POP3) is intended to permit a workstation to dynamically access a maildrop on a server host in a useful fashion. Usually, this means that the POP3 protocol is used to allow a workstation to retrieve mail that the server is holding for it.

POP3 is not intended to provide extensive manipulation operations of mail on the server; normally, mail is downloaded and then deleted. A more advanced (and complex) protocol, IMAP4, is discussed in [RFC1730].

Initially, the server host starts the POP3 service by listening on TCP port 110. When a client host wishes to make use of the service, it establishes a TCP connection with the server host. When the connection is established, the POP3 server sends a greeting. The client and POP3 server then exchange commands and responses (respectively) until the connection is closed or aborted. Commands in the POP3 consist of a case-insensitive keyword, possibly followed by one or more arguments. All commands are terminated by a CRLF

POP3 Command Summary

Minimal POP3 Commands:

- USER name valid in the AUTHORIZATION state
- PASS string
- QUIT
- STAT valid in the TRANSACTION state
- LIST [msg]
- RETR msg
- DELE msg

- NOOP
- RSET
- QUIT

Optional POP3 Commands:

- APOP name digestvalid in the AUTHORIZATION state
- TOP msg n valid in the TRANSACTION state
- UIDL [msg]

POP3 Replies:

- +OK
- -ERR

Note that with the exception of the STAT, LIST, and UIDL commands, the reply given by the POP3 server to any command is significant only to "+OK" and "-ERR". Any text occurring after this reply may be ignored by the client.

All messages transmitted during a POP3 session are assumed to conform to the standard for the format of Internet text messages [RFC822]. It is important to note that the octet count for a message on the server host may differ from the octet count assigned to that message due to local conventions for designating end-of-line. Usually, during the AUTHORIZATION state of the POP3 session, the POP3 server can calculate the size of each message in octets when it opens the maildrop. For example, if the POP3 server host internally represents end-of-line as a single character, then the POP3 server simply counts each occurrence of this character in a message as two octets. Note that lines in the message which start with the termination octet need not (and must not) be counted twice, since the POP3 client will remove all byte-stuffed termination characters when it receives a multi-line response.

It is conjectured that use of the APOP command provides origin identification and replay protection for a POP3 session. Accordingly, a POP3 server which implements both the PASS and APOP commands should not allow both methods of access for a given user; that is, for a given mailbox name, either the USER/PASS command sequence or the APOP command is allowed, but not both. Further, note that as the length of the shared secret increases, so does the difficulty of deriving it. Servers that answer -ERR to the USER command are giving potential attackers clues about which names are valid.

REQUEST FOR COMMENTS: 1725
POST OFFICE PROTOCOL - VERSION 3

This memo is a revision to RFC 1460, a Draft Standard. It makes the following changes from that document:

- removed text regarding “split-UA model", which didn't add anything to the understanding of POP
- clarified syntax of commands, keywords, and arguments
- clarified behavior on broken connection
- explicitly permitted an inactivity autologout timer

- clarified the requirements of the "exclusive-access lock"
- removed implementation-specific wording regarding the parsing of the maildrop
- allowed servers to close the connection after a failed authentication command
- removed the LAST command
- fixed typo in example of TOP command
- clarified that the second argument to the TOP command is non-negative
- added the optional UIDL command
- added warning regarding length of shared secrets with APOP
- added additional warnings to the security considerations section

On certain types of smaller nodes in the Internet it is often impractical to maintain a message transport system (MTS). For example, a workstation may not have sufficient resources (cycles, disk space) in order to permit a SMTP server [RFC821] and associated local mail delivery system to be kept resident and continuously running. Similarly, it may be expensive (or impossible) to keep a personal computer interconnected to an IP-style network for long amounts of time (the node is lacking the resource known as "connectivity").

Despite this, it is often very useful to be able to manage mail on these smaller nodes, and they often support a user agent (UA) to aid the tasks of mail handling. To solve this problem, a node which can support an MTS entity offers a maildrop service to these less endowed nodes. The Post Office Protocol - Version 3 (POP3) is intended to permit a workstation to dynamically access a maildrop on a server host in a useful fashion. Usually, this means that the POP3 is used to allow a workstation to retrieve mail that the server is holding for it.

Initially, the server host starts the POP3 service by listening on TCP port 110. When a client host wishes to make use of the service, it establishes a TCP connection with the server host. When the connection is established, the POP3 server sends a greeting. The client and POP3 server then exchange commands and responses (respectively) until the connection is closed or aborted. Commands in the POP3 consist of a keyword, possibly followed by one or more arguments. All commands are terminated by a CRLF pair.

A POP3 session progresses through a number of states during its lifetime. Once the TCP connection has been opened and the POP3 server has sent the greeting, the session enters the AUTHORIZATION state. In this state, the client must identify itself to the POP3 server. Once the client has successfully done this, the server acquires resources associated with the client's maildrop, and the session enters the TRANSACTION state. In this state, the client requests actions on the part of the POP3 server. When the client has issued the QUIT command, the session enters the UPDATE state. In this state, the POP3 server releases any resources acquired during the TRANSACTION state and says goodbye. The TCP connection is then closed.

A POP3 server MAY have an inactivity autologout timer. Such a timer MUST be of at least 10 minutes' duration. The receipt of any command from the client during that interval should suffice to reset the autologout timer. When the timer expires, the session does NOT enter the UPDATE state--the server should close the TCP connection without removing any messages or sending any response to the client.

POP3 Command Summary

Minimal POP3 Commands:

- USER name valid in the AUTHORIZATION state
- PASS string
- QUIT
- STAT valid in the TRANSACTION state
- LIST [msg]
- RETR msg
- DELE msg
- NOOP
- RSET
- QUIT valid in the UPDATE state

Optional POP3 Commands:

- APOP name digestvalid in the AUTHORIZATION state
- TOP msg n valid in the TRANSACTION state
- UIDL [msg]

POP3 Replies:

- +OK
- -ERR

All messages transmitted during a POP3 session are assumed to conform to the standard for the format of Internet text messages [RFC822]. It is important to note that the octet count for a message on the server host may differ from the octet count assigned to that message due to local conventions for designating end-of-line. Usually, during the AUTHORIZATION state of the POP3 session, the POP3 server can calculate the size of each message in octets when it opens the maildrop. For example, if the POP3 server host internally represents end-of-line as a single character, then the POP3 server simply counts each occurrence of this character in a message as two octets. Note that lines in the message which start with the termination octet need not be counted twice, since the POP3 client will remove all byte-stuffed termination characters when it receives a multi-line response.

It is conjectured that use of the APOP command provides origin identification and replay protection for a POP3 session. Accordingly, a POP3 server which implements both the PASS and APOP commands must not allow both methods of access for a given user; that is, for a given "USER name" either the PASS or APOP command is allowed, but not both.

REQUEST FOR COMMENTS: 1730
INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4

The Internet Message Access Protocol, Version 4 (IMAP4) allows a client to access and manipulate electronic mail messages on a server. IMAP4 permits manipulation of remote message folders, called "mailboxes", in a way that is functionally equivalent to local mailboxes. IMAP4 also provides the capability for an off-line client to resynchronize with the

server. IMAP4 supports a single server. A mechanism for supporting multiple IMAP4 servers is discussed in [IMSP].

IMAP4 includes operations for creating, deleting, and renaming mailboxes; checking for new messages; permanently removing messages; setting and clearing flags; RFC 822 and MIME parsing; searching; and selective fetching of message attributes, texts, and portions thereof. Messages in IMAP4 are accessed by the use of numbers. These numbers are either message sequence numbers (relative position from 1 to the number of messages in the mailbox) or unique identifiers (immutable, strictly ascending values assigned to each message, but which are not necessarily contiguous).

IMAP4 does not specify a means of posting mail; this function is handled by a mail transfer protocol such as [SMTP]. IMAP4 is designed to be upwards compatible from the [IMAP2] protocol.

The IMAP4 protocol assumes a reliable data stream such as provided by TCP. When TCP is used, an IMAP4 server listens on port 143. An IMAP4 session consists of the establishment of a client/server connection, an initial greeting from the server, and client/server interactions. These client/server interactions consist of a client command, server data, and a server completion result response. All interactions transmitted by client and server are in the form of lines; that is, strings that end with a CRLF. The protocol receiver of an IMAP4 client or server is either reading a line, or is reading a sequence of octets with a known count followed by a line.

IMAP4 protocol transactions, including electronic mail data, are sent in the clear over the network unless the optional privacy protection is negotiated in the AUTHENTICATE command. A server error message for an AUTHENTICATE command which fails due to invalid credentials should not detail why the credentials are invalid. Use of the LOGIN command sends passwords in the clear. This can be avoided by using the AUTHENTICATE command instead. A server error message for a failing LOGIN command should not specify that the user name, as opposed to the password, is invalid.

References

[IMAP2] Crispin, M., "Interactive Mail Access Protocol - Version 2", RFC 1176, University of Washington, August 1990.
[IMSP] Myers, J. "IMSP -- Internet Message Support Protocol", Work in Progress.
[SMTP] Postel, Jonathan B. "Simple Mail Transfer Protocol", STD 10, RFC 821, USC/ Information Sciences Institute, August 1982.

REQUEST FOR COMMENTS: 1732
IMAP4 COMPATIBILITY WITH IMAP2 AND IMAP2BIS

This is a summary of hints and recommendations to enable an IMAP4 implementation to interoperate with implementations that conform to earlier specifications. None of these hints and recommendations are required by the IMAP4 specification; implementors must decide for themselves whether they want their implementation to fail if it encounters old software.

IMAP4 has been designed to be upwards compatible with earlier specifications. For the most part, IMAP4 facilities that were not in earlier specifications should be invisible to clients unless the client asks for the facility. In some cases, older servers may support some of the capabilities listed as being "new in IMAP4" as experimental extensions to the IMAP2 protocol described in RFC 1176.

IMAP4 client interoperability with old servers

In general, a client should be able to discover whether an IMAP2 server supports a facility by trial-and-error; if an attempt to use a facility generates a BAD response, the client can assume that the server does not support the facility. A quick way to check whether a server implementation supports the IMAP4 specification is to try the CAPABILITY command. An OK response that includes the IMAP4 capability value indicates a server that supports IMAP4; a BAD response or one without the IMAP4 capability value indicates an older server.

The following is a list of facilities that are only in IMAP4:

- CAPABILITY command
- AUTHENTICATE command.
- LSUB and LIST commands
- SEARCH extensions (character set, additional criteria)
- BODYSTRUCTURE fetch data item
- RFC822.HEADER.LINES and RFC822.HEADER.LINES.NOT fetch data items
- BODY.PEEK[section], RFC822.PEEK, and RFC822.TEXT.PEEK fetch data
- UID fetch data item and the UID commands
- FLAGS.SILENT, +FLAGS.SILENT, and -FLAGS.SILENT store data items

The following IMAP4 facilities were introduced in the experimental IMAP2bis revisions to RFC-1176, and may be present in a server that does not support the CAPABILITY command:

- CREATE, DELETE, and RENAME commands
- APPEND command
- SUBSCRIBE and UNSUBSCRIBE commands
- EXAMINE command
- BODY, BODY[section], and FULL fetch data items
- PARTIAL command

IMAP4 server interoperability with old clients

In general, there should be no interoperation problem between a server conforming to the IMAP4 specification and a well-written client that conforms to an earlier specification.

REQUEST FOR COMMENTS: 1733
DISTRIBUTED ELECTRONIC MAIL MODELS IN IMAP4

Distributed Electronic Mail Models

There are three fundamental models of client/server email:

- offline,
- online, and

- disconnected use.

IMAP4 can be used in any one of these three models.

The *offline* model is the most familiar form of client/server email today, and is used by protocols such as POP-3 (RFC 1225) and UUCP. In this model, a client application periodically connects to a server. It downloads all the pending messages to the client machine and deletes these from the server. Thereafter, all mail processing is local to the client. This model is store-and-forward; it moves mail on demand from an intermediate server (maildrop) to a single destination machine.

The *online* model is most commonly used with remote filesystem protocols such as NFS. In this model, a client application manipulates mailbox data on a server machine. A connection to the server is maintained throughout the session. No mailbox data are kept on the client; the client retrieves data from the server as is needed. IMAP4 introduces a form of the online model that requires considerably less network bandwidth than a remote filesystem protocol, and provides the opportunity for using the server for CPU or I/O intensive functions such as parsing and searching.

The *disconnected* use model is a hybrid of the offline and online models, and is used by protocols such as PCMAIL (RFC 1056). In this model, a client user downloads some set of messages from the server, manipulates them offline, then at some later time uploads the changes. The server remains the authoritative repository of the messages. The problems of synchronization (particularly when multiple clients are involved) are handled through the means of unique identifiers for each message.

Each of these models have their own strengths and weaknesses:

Feature	Offline	Online	Disc
Can use multiple clients	NO	YES	YES
Minimum use of server connect time	YES	NO	YES
Minimum use of server resources	YES	NO	NO
Minimum use of client disk resources	NO	YES	NO
Multiple remote mailboxes	NO	YES	YES
Fast start-up	NO	YES	NO
Mail processing when not online	YES	NO	YES

Although IMAP4 has its origins as a protocol designed to accommodate the online model, it can support the other two models as well. This makes possible the creation of clients that can be used in any of the three models. For example, a user may wish to switch between the online and disconnected models on a regular basis (e.g. owing to travel).

IMAP4 is designed to transmit message data on demand, and to provide the facilities necessary for a client to decide what data it needs at any particular time. There is generally no need to do a wholesale transfer of an entire mailbox or even of the complete text of a message. This makes a difference in situations where the mailbox is large, or when the link to the server is slow.

More specifically, IMAP4 supports server-based RFC 822 and MIME processing. With

this information, it is possible for a client to determine in advance whether it wishes to retrieve a particular message or part of a message. For example, a user connected to an IMAP4 server via a dialup link can determine that a message has a 2000 byte text segment and a 40 megabyte video segment, and elect to fetch only the text segment.

In IMAP4, the client/server relationship lasts only for the duration of the TCP connection. There is no registration of clients. Except for any unique identifiers used in disconnected use operation, the client initially has no knowledge of mailbox state and learns it from the IMAP4 server when a mailbox is selected. This initial transfer is minimal; the client requests additional state data as it needs.

As noted above, the choice for the location of mailbox data depends upon the model chosen. The location of message state (e.g. whether or not a message has been read or answered) is also determined by the model, and is not necessarily the same as the location of the mailbox data. For example, in the online model message state can be co-located with mailbox data; it can also be located elsewhere (on the client or on a third agent) using unique identifiers to achieve common reference across sessions. The latter is particularly useful with a server that exports public data such as netnews and does not maintain per-user state.

The IMAP4 protocol provides the generality to implement these different models. This is done by means of server and (especially) client configuration, and not by requiring changes to the protocol or the implementation of the protocol.

Index

A

C

E

I

L

M

P

R

S

T

U

W