S U P P L E M E N T A L  M A N U A L

# InterMail®

## Supplement to v4.0 Manual Set

**Version 1.4 of Manual**
*Published August 11, 1999*

Software.com™
THE INTERNET INFRASTRUCTURE COMPANY™

*Document number: SUPP990811*

# *Table of Contents*

# 1

# *Introduction*

This document identifies additions and changes to the existing InterMail 4.0 manual set. It supplements information recorded in the standard manuals and should be used in conjunction with those items identified below:

- The *InterMail Installation Guide*
- The *InterMail Migration Guide*
- The *InterMail Upgrade Manual*
- The *InterMail Operations Guide*
- The *InterMail Reference Guide*
- The *Integrated Services Directory Guide*
- The *WebMail Reference Guide*
- The *InterManager Administration Guide*

For ease of reference, the structure of this document closely follows the structure of the original manuals with separate chapters devoted to individual system components, specific operational tasks, and particular interface tools (configuration keys, administration commands, etc.)

If descriptions or instructions found in the original manuals conflict with information in this supplement, the information in the supplement should be considered authoritative as the supplement is revised more frequently and is therefore more current.

***Note:*** *Each new version of the supplement replaces all prior versions. Check the version number and publication date on the cover of this document to determine relative age.*

## 1.1 Significant Changes in Functionality

The most significant change since publication of the original InterMail manual set is the introduction of an additional server to support the web-based SelfCare and WebMail applications. A detailed description of the Web Server, both architecture and operation can be found in Chapter 3 of this manual.

Additional information on the SelfCare application can be found in Chapter 2. The WebMail application is documented independently in the *WebMail Reference Guide*.

In InterMail 4.1, ISP's have greater control over who can retrieve mail from their POP/IMAP Servers. In addition, a new per-user filtering mechanism is added, that extends InterMail's SIEVE filtering

A multijournaling feature is added to the Message Store Server. This new feature enhances CPU performance and improves MSS capacity.

The WebMail application is enhanced to include an improved user interface, robust paging functionality, and improved logging that is easily customizable to satisfy the ISP's needs. In addition, error messages are restructured to allow more efficient methods of troubleshooting.

***Note:*** *Per-user mail filtering is only available for the SUN operating system.*

# 1.2 Controlling POP/IMAP Access by Location

It is desirable for ISP's to limit user access to their POP/IMAP Servers, based on the location of the user. By programming their routers to block access to all unknown users (for example, anyone connecting from the Internet), ISP's can completely deny all unknown access.

However, there are times when an ISP may want to grant users access to their POP/IMAP Servers, even if retrieval is being attempted from an unknown location. InterMail offers a feature that supports this. By utilizing specific paths through the ISP's routers, traffic coming from an unknown location will arrive at one interface, while traffic coming from a trusted location (internal network) will arrive at different interface. InterMail is able to identify trusted and unknown locations by the interface through which the connection is made.

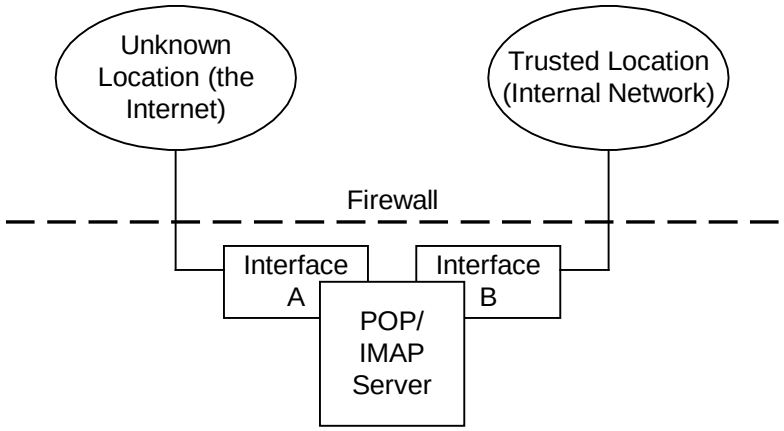


**Figure 1: Trusted Locations vs. Unknown Locations.**

### *Configuring Trusted Interfaces*

In order to limit user connections to the ISP's internal network, each server host needs to know which interfaces are considered trusted. The `/*/common/trustedInterfaces` configuration key allows the ISP to list a set of trusted interface addresses. Values are specified in the form of an IP address, as shown in the example below:

```
/uranus/common/trustedInterfaces:
      [10.2.7.49]
      [10.2.7.50]
      [10.2.5.51]
```

Any IP address that does not appear in the `/*/common/trustedInterfaces` configuration key is considered untrusted.

 

### *Controlling User Access to Designated Interfaces*

Class of service options gives ISPs further flexibility when choosing the locations from which users can access their mailboxes. Users can be granted permission to connect to the ISP's server from any location, from trusted locations only, or not at all.

The following class of service attributes specifies the login permission of an account:

- pref_popaccess – defines access restrictions for standard POP retrieval.
- pref_popsslaccess – defines access restrictions for POP retrieval with SSL.
- pref_imapaccess – defines access restrictions for IMAP.

The previous class of service attributes have three string values:

| Value | Description |
|---|---|
| `all` | Users can log in from any location |
| `trusted` | Users can only log in from interfaces that are specified in the `/*/common/trustedInterfaces` configuration key. |
| `none` | Users cannot log in from any location. |

***Note:*** *For more information on these attributes, see Section 8.1.*

The SSL option for the POP Server is listed independently and this allows specified access for encrypted locations.

For example, John Doe has the following account settings:

pref_popaccess – trusted
pref_popsslaccess – all
pref_imapaccess – none

In the scenario above, John Doe would be allowed standard POP access from trusted locations only, POP with SSL access from any location, and IMAP access would never be allowed.

***Note:*** *These options can be set on either a class of service level or a per-account level, with the per-account setting having precedence.*

When a user, with a "trusted" account setting, tries to connect to a corresponding POP/IMAP service, the server compares the server host interface on which the connection arrives to the list of trusted IP addresses. If the IP address is in the list, and the user's access restrictions allow entry from a trusted interface, then access is allowed.

# 1.3 Per-User Mail Filtering (SUN only)

InterMail allows you to set customizable mail filtering rules on a per-user basis. This functionality extends the system level filtering provided in InterMail 4.0. Since per-user filters are specified on a per-account level, there is greater flexibility when configuring spam prevention methods.

For example, Mario Garcia can have a rule to block all mail from a known spammer; Julie Williams can have all mail from her friend, Brock Lee, sent to her "Friends" folder; and Marge Harding can have no filters at all, allowing mail to be delivered as normal.

Per-user filters are run inside the MTA, just before the message is delivered to the Message Store database. These filters are only run on messages destined for delivery to a local mailbox.

***Note:*** *For a complete description on mail filter functionality and syntax, see Chapter 4 of the InterMail Operations Guide*

## 1.3.1 Extensions to the SIEVE Language

To accommodate per-user filtering, the SIEVE filtering language was extended to include four new actions. The actions are described below:

`Write-header <"quoted-string"> [,] ["quoted string"]`

This action takes the current message being delivered to the user and changes the header portion of the message. The quoted string contains the header attribute value pair (with a ":" as a delimiter). If the attribute exists, the old value is replaced with the new value.

***Note:*** *The command `write-header` does not effect forwarded messages. Only the header changes following the `HEADER` keyword are honored.*

**Example 1:**

If you want to add an X-Header to the message with the value "sieve," do the following:

```
write-header "X-Header: sieve";
```

**Example 2:**

If you want to add two headers to the message, do the following:

```
write-header "X-Header: sieve" , "X-JunkMail: true";
```

```
Sendmessage <"recipient's address"> [,] ["recipient's address"] DATA \
<"quoted string"> [,] ["quoted string"]
```

This action sends a brand new message to the list of recipients specified. The header and body of the message is specified at a list of quoted strings following the DATA keyword:

**Example:**

If you want to send a new message to Brock Lee, do the following:

```
sendmessage "Brock.Lee@software.com" DATA
                                "To: Brock.Lee@software.com",
                                "From: Marge.Harding@accordance.com",
                                "Subject: Hi",
                                "",     # mark end of header
                                "Hello, Marge Harding. How are you?",
                                " –Brock Lee";
```

`Fileinto <"foldername">`

This action files the current message into the user's folder. If the folder does not exist, it is instantly created. This command implies that "keep" was called.

**Example 1:**

If you want to file the current message into the "jokes" folder, do the following:

```
fileinto "jokes";
```

**Example 2:**

If you want to file the current message into the "funny" folder (inside the joke folder), do the following:

```
fileinto "jokes/funny";
```

```
forward <"recipient's address"> [,] ["another recipient's address"] \
[HEADER "quoted string" [,] ["quoted string"]]
```

The `forward` action was available in InterMail 4.0. However, modifications have been made that allow you to specify multiple recipients, as well as change the header on the forwarded message. For more information on the `forward` action, see Chapter 4 of the *InterMail Operations Guide*.

**Example 1:**

If you want to forward the current message to brock.lee@software.com (no header changes), do the following:

```
forward "brock.lee@software.com"
```

**Example 2:**

If you want to forward the current message to Brock Lee and Marge Harding, with one header change, do the following:

```
forward "brock.lee@software.com" , "Marge.Harding@accordance.com" HEADER "subject:
This is my forwarded message";
```

**Example 3:**

If you want to forward the current message to Brock Lee, change the header sent to Marge Harding, and file the original in the default inbox folder, do the following:

```
forward "brock.lee@software.com",;
forward "marge.harding@accordance.com" HEADER "subject: This is a forwarded message";
keep;
```

## 1.3.2 Setting Per-User Mail Filters

Per-user mail filters are configured in three ways:

1. Using the `pref_mtaFilterPerUser` class of service attribute.
2. Using the `imfilterctrl` command line.
3. Using the C-APIs.

### *Enabling Filters via Class of Service Options*

The system administrator can enable/disable class of service attributes by using `pref_mtaFilterPerUser`. This is necessary to on/off SIEVE rules on a per-user basis or on a class of service level.

**Example:**

```
imdbcontrol sac Brock.Lee software.com pref_mtaFilterPerUser 1
```

---

***Note:*** *If pref_`mtaFilterPerUser` is on (set to 1), an extra call is made to the Message Store database to retrieve the user's SIEVE filter text, even if the filter text file is empty. This may cause a small performance issue. It is recommended that pref_`mtaFilterPerUser` is disabled when no filters are set.*

---

### *Setting Filters via the Command Line*

The `imfilterctrl` utility is used to set per-user mail filters in the Message Store database. To execute `imfilterctrl`, you must be logged in as the InterMail user on any InterMail host. The `imfilterctrl` command is used to set, retrieve, or remove filter rules for an individual account.

**Syntax:**

```
imfilterctrl <option> <address> <domain> [- |{<filter-file>|<filter-
file to stdout>}]
```

**Where:**

| | |
|---|---|
| `option` | Specifies the operation to be performed. There are three choices: get, set, and clear. |
| `get` | Retrieves per-user filtering information for a specified user. |
| `set` | Sets a new per-user rule for a specified user. |
| `clear` | Removes all rules for a specified user. |
| `address` | SMTP address of the specified user. |
| `domain` | Domain name of the specified user. |
| `filter-file` | A text file containing filtering rules. |
| `filter-file to stdout` | The file to which filter data will be written. |

**Example:**

```
imfilterctrl set Mario.Garcia software.com sievefilename.txt
```

### *Setting Per-User Filters via the C-API*

Three routines that are added to the C-API that allow system administrators to add and remove user filters from the Message Store database:

## IM_ValidateMtaFilter

```
int  IM_ValidateMtaFilter(IM_Account*, IM_Reply*, IM_Error*);
```

Compiles and validates the filterText. This is useful in verifying that the SIEVE filter will compile (valid syntax).

**Return Values:**

| | |
|---|---|
| `IM_SUCCESS` | Filter is valid. |
| `IM_FAILURE` | Parser error. Possible values of error->number. |
| `IM_ERROR_FILTER_PARSER` | SIEVE compiler error. Error messages contained in error->string (same output as `imfiltercheck –e`) |
| `IM_ERROR_MISSING_ARG` | Missing argument. |

## IM_UpdateMtaFilter

```
int  IM_UpdateMtaFilter(IM_Account*, IM_Reply*, IM_Error*);
```

Stores the filterText for a given user. This function also validates the filterText and returns an error if the test cannot be parsed by SIEVE.

**Return Values:**

| | |
|---|---|
| `IM_SUCCESS` | Filter is valid and stored in the user's mailbox. |
| `IM_FAILURE` | Error. Possible values of error->number. |
| `IM_ERROR_FILTER_PARSER` | SIEVE compiler error. Error messages contained in error->string (same output as `imfiltercheck –e`) |
| `IM_ERROR_MISSING_ARG` | Missing argument. |
| `IM_ERROR_INVALID_ARG` | One of the arguments was invalid (same as `IM_CreateReply`). |

## IM_ReadMtaFilter

```
int  IM_ReadMtaFilter(IM_Account*, IM_Reply*, IM_Error*);
```

Reads the user SIEVE filter text and places it in the filterText structure.

**Return Values:**

| | |
|---|---|
| `IM_SUCCESS` | Filter is valid and stored in the user's mailbox. |
| `IM_FAILURE` | Error. Possible values of error->number. |
| `IM_ERROR_FILTER_PARSER` | SIEVE compiler error. Error messages contained in error->string (same output as `imfiltercheck –e`) |
| `IM_ERROR_INVALID_ARG` | One of the arguments was invalid (same as `IM_CreateReply`). |

## 1.3.3 Sample Filters

The following are examples of per-user SIEVE filtering methods. For a complete description of the SIEVE filtering syntax, see Chapter 4 of the InterMail Operations Guide.

**Example 1:**

Bounce all messages from “Badman@spam.org”:

```
if header "from" contains "Badman@spam.org" {
  bounce "This message was bounced because we think it’s spam";
}
```

**Example 2:**

Silently drop all mail from “junkmail.com”:

```
if header "from" contains "junkmail.com" {
  discard;
}
```

**Example 3:**

Forward a message to a pager and deliver the message to an inbox:

```
if header "subject" contains "urgent" {
  forward "mypager@skytel.com";
}
```

**Example 4:**

Forward a message to a fax server and do not deliver the message to an inbox:

```
if any-of (header "to" contains "fax", header "cc" contains "fax") {
  forward "myfax@server.com";
}
```

**Example 5:**

File all messages that have no “from” or “date” field into a spam folder:

```
if not exists ("From" "Date") {
  fileinto "spam";
}
```

# 1.4 Modifications to WebMail Customization

The performance of the WebMail application has been improved. Changes associated with the WebMail enhancements require modification of the instructions for customizing WebMail, message text, images, and banners. See the sections that follow for detailed information.

***Note:*** *See Chapter 5 of the WebMail Manual for more information on WebMail customization.*

### *Customizing Message Text*

Text customization is performed by editing the `WM-lang_English.pl` file, found in the `$INTERMAIL/httpd/htdocs/webmail` directory. Customizing message text is described in detail in Chapter 5 of the WebMail Manual. The existing instructions are still valid, however it should be noted that all of the modifiable values in the `WM-lang_English.pl` file are not text strings. Some are subroutines that generate string values based on passed arguments. The following is an example of a subroutine:

```
$WebMail::read_msgs =
{
  # …
  FolderPosText => sub { qq(Message $_[0] of $_[1]); },
  # …
};
```

Modifying the subroutine definition is easy if you know Perl, just be careful not to change the way the arguments are used and keep the subroutine definition and invocations consistent.

### *Changing Image Location*

By default, WebMail looks for images in the `$INTERMAIL/httpd/htdocs/webmail/images` directory. There is a new feature that allows you to move the `images` directory to a different location. This option may be desirable for two reasons:

1. A different file system organization is preferred, allowing images to be shared between several applications.
2. Serving images from a different host can reduce the load on the primary Web Server. For example, serving the WebMail images from a Linux box is an inexpensive method in increasing I/O performance.

**Warning!** In order to guarantee that the system finds the image files required for use in help text displays, duplicates images must be kept in the default images directory, (`$INTERMAIL/httpd/htdocs/webmail/images`).

### *Banner Generation*

There is a new method for managing banner generation. Previously, all banner images were referenced by values recorded by the `/*/web/banners` configuration key. Now, the description of the banners is contained in the `banners.pl` file. A new subroutine in the `WM-lang_English.pl` file controls this feature:

```
$WebMail::banners_generator = sub
{
  my $file = 'fancy_banners.pl';
  -f $file or $file = 'banners.pl';
  do $file;
  my @keys = keys %$banners;
  my $i = int(rand @keys);
  my $key = $keys[$i];
  return ($key, $WebMail::banners->{$key});
};
# undef $WebMail::banners_generator;
 1;
```

As shown in the subroutine above, WebMail will search for banner files using the following steps:

1. First, WebMail looks for banners in the `fancy_banners.pl` file. This file is a generic placeholder for custom banner images. By default, `fancy_banners.pl` is not present. If you wish to use custom banners, you will need to create this file.
2. If the fancy_`banners.pl` file does not exist, the `banners.pl` file will be used. A random image will be selected from among the items listed in this file:

   ```
   $WebMail::banners =
   {
     "$image_root/imap1.gif" => 'http://www.software.com',
     "$image_root/ldap-rev.gif" => 'http://www.software.com',
     "$image_root/pop31.gif" => 'http://www.software.com',
     "$image_root/internet1.gif" => 'http://www.software.com',
     "$image_root/mailserver1.gif" => 'http://www.software.com',
     "$image_root/smtp.gif" => 'http://www.software.com',
   };
   ```

   If `banners.pl` is not defined or the entire banner subroutine is undefined, the `/*/web/banners` configuration key will be used.
3. In the `/*/web/banners` configuration key, special formatting indicates that the value represents a file:

   `/*/web/banners: [file=example_file.pl]`

   If the initial value in the `banners` configuration key is a file, the systems looks for banner images within that file.
4. If no file is defined in the `/*/web/banners` configuration key, the key is used in its normal manner:

   ```
   /*/web/banners: [images/imap1.gif,http://www.software.com]
                   [images/ldap-rev.gif,http://www.software.com]
   ```

---

***Note:*** *For a complete description on the /*/web/banners configuration key, see Chapter 4 of the Supplemental Manual.*

---

By default, the new mechanism for banner generation is enabled. In order to disable this feature, do the following:

1. Open the WM-lang_English.pl file.
2. Uncomment the `# undef $WebMail::banners_generator;` line.

# 1.5 MSS Journaling

The InterMail system is enhanced to allow the MSS to run with multiple journal files instead of just one. Previously, when journaling was enabled and the CPU was under heavy load, a system bottleneck was encountered. Because the MSS formerly wrote to a single journal file, more CPU resources were needed to get desired throughput. By adding multiple journal files, the MSS has improved its capacity.

The multi-journal feature of the system will be off by default. Therefore, the default behavior will be exactly the same as the previous versions of the 4.x software.

***Note:*** *Please consider that you may have site specific considerations to resolve before using this new feature. For example, ensure that your backup and recovery scripts take into consideration a multi-journal file system as well as special site-specific scripts that use journal files.*

The main component benefiting from this change is the MSS. Also benefiting are the garbage collector (`immssgc`) and the journal recovery tool (`imjrnrecover`).

### *Multijournaling functionality*

Prior to these changes, the MSS created journal files in the following form:

```
YYMMDD.HHMMSS.mss.0
YYMMDD.HHMMSS.mss.1
YYMMDD.HHMMSS.mss.2
...
```

Journal entries for the `mss.0` process are applied to the `YYMMDD.HHMMSS.mss.0` file. Journal entries for the `mss.1` process are applied to the `YYMMDD.HHMMSS.mss.1` file, and so on.

If the `multiJournaling` configuration key is set to `false` or omitted entirely from the configuration database, nothing changes and the MSS creates journal files and writes journal entries just as it always has.

***Note:*** *For a complete description on the `multiJournaling` configuration key, see Section 4.1.*

If the `multiJournaling` key is set to `true`, the MSS creates journal files in the following form:

```
YYMMDD.HHMMSS.mss.0.0
YYMMDD.HHMMSS.mss.0.1
YYMMDD.HHMMSS.mss.0.2
YYMMDD.HHMMSS.msd.0.0
YYMMDD.HHMMSS.mss.1.0
YYMMDD.HHMMSS.mss.1.1
YYMMDD.HHMMSS.mss.1.2
YYMMDD.HHMMSS.mss.1.3
YYMMDD.HHMMSS.msd.1.0
YYMMDD.HHMMSS.mss.2.0
YYMMDD.HHMMSS.mss.2.1
...
```

In this new scheme, the `msd` files are reserved specifically for journal DELETE entries and the `immssgc` entries. Journal DELETE entries for `mss.0` are applied to the `YYMMDD.HHMMSS.msd.0.0` file, journal delete entries for `mss.1` are applied to the `YYMMDD.HHMMSS.msd.1.0` file, and so on. There is only one `msd` file for each MSS process.

Journal CREATE entries are written to the more conventional `mss` files. In the example above, journal CREATE entries for `mss.0` would appear in `YYMMDD.HHMMSS.mss.0.0`, `YYMMDD.HHMMSS.mss.0.1`, and `YYMMDD.HHMMSS.mss.0.2`.

Notice that in the new scheme, each MSS process has multiple files for its journal CREATE entries, each with an additional "`.<n>`" suffix, where `<n>` is some number.

The `numJournalsPerServer` configuration key sets the maximum value of `<n>` and the maximum number of journal CREATE files for an MSS process. For example, if the `numJournalsPerServer` configuration key is set to `5`, journal CREATE entries for `mss.0` would appear in any of the following 5 files:

```
YYMMDD.HHMMSS.mss.0.0
YYMMDD.HHMMSS.mss.0.1
YYMMDD.HHMMSS.mss.0.2
YYMMDD.HHMMSS.mss.0.3
YYMMDD.HHMMSS.mss.0.4
```

---

***Note:*** *Not all of these files will necessarily be created. The MSS only creates as many as it needs. For a complete description on the `numJournalsPerServer` configuration key, see Section 4.1.*

---

# 1.6 Errors and Omissions

The information presented in this section provides clarification of discussions that were previously incomplete and acknowledges errors in the original documentation.

The chapters that follow provide detailed information on features that were added or functionality that changed after the original documentation was published.

### *Setting up the Welcome Message*

If specified, a welcome message can arrive automatically in every new mailbox. In order to enable this option, before going into production, you must create the welcome message for new

 

users. A well-written welcome message can create a positive first impression and also convey important information, such as contact numbers and e-mail addresses for technical support, information on quota policies, and the URL for your organization's Web site.

To create a new `admin` mailbox with a welcome message, perform the following steps:

1. Create an account for the `admin` mailbox.

   This is required to run `immsinit`. The value of the `internalId` field specified when the account is created will be the `admin` mailbox name. Note that the `internalId` field is numeric only.

2. Write a welcome message in a text file. The message should include the following information:

   ```
   From: <Your ISP>
   To: <Your ISP's new customers>
   Subject: <Welcome to Your ISP>
   Message-ID: <unique id, such as "<Welcome.052699>">
   <The body of the message>
   ```

   Make sure that the `Message-ID` header is unique and is in angle brackets. Adding a date stamp can be useful.

3. Set the configuration key `/*/mss/weclomeMsgID` key to the value of the `Message-ID` header in the message.

   Make sure that you include angle brackets if they are present. For example:

   ```
   /*/mss/welcomeMsgID: [<Welcome.052699>]
   ```

   The `welcomeMsgID` key refers to the `messageid` field in the `im_message` table. If the Message-ID is not unique, this is the value that must be changed.

4. Set the configuration key `/*/mss/adminMessageStoreName` to the value of the `internalId` field specified in Step 1.

5. Run the `immsinit` command:

   ```
   immsinit -host <hostname> -w <welcome message filename> -a <value
   /*/mss/adminMessageStoreName key>
   ```

This creates the `admin` mailbox and inserts the defined welcome message into it. If you do not use the `-w` flag, the default welcome message is inserted, with its `Message-ID` of `<Welcome>`. If you do not use the `-a` flag, `immsinit` creates the `admin` mailbox by default (you need an existing account with an `internalId` of `admin` for this to work).

---

***Note**: You cannot run `immsinit` if the specified `admin` mailbox already exists; if you do, the command will produce a `MsAlreadyExists` error.*

---

### *Changing the Welcome Message*

You can change the welcome message in two ways: by sending a new welcome message to the account that points to the `admin` mailbox, or by deleting the `admin` mailbox and running `immsinit` again with a new welcome message.

To send a new welcome message to the `admin` mailbox:

1. Connect to the MTA using any mail client or using telnet.

2. Send the new welcome message to the account that points to the `admin` mailbox.

   Be sure that the required headers are included in the welcome message and that the `Message-ID` header is unique. If there is already a message in the database with the same message ID, the MSS will change the message ID of the new message.

3. Check the arrival date or review the message file on disk to verify that the message has been delivered properly and that the message ID has not been changed.

4. Change the `/*/mss/welcomeMsgID` key to the value of the `Message-ID` header of the new message.

To delete the `admin` mailbox and run `immsinit` to create a new welcome message:

1. Run the `imboxdelete` command.

   ```
   imboxdelete <hostname> <admin mailbox name>
   ```

2. Run `immsinit` again making sure to use a unique message ID in the new welcome message.

3. Set the welcomeMsgID and adminMessageStoreName keys.

### *MoveMessages*

In the Perl API documentation, it incorrectly states that, “It is possible to pass one or more arrays of msgRef’s instead of individual msgRef’s.” However, it is only possible to pass one reference to an array at a time.

### *Setting Message Aging on a Class of Service Level*

Figure 2 on page 19 of the *Integrated Services Directory Guide* and Figure 3 on page 4 of the *InterManager Manual* incorrectly refer to message aging being set on a class of service level. This is incorrect and the only way message aging can be set is via a configuration key, not a class of service attribute. Therefore, all mailboxes controlled by a particular MSS server must use the same message aging policy.

### *Significance of Account Status*

Although Suspended, Locked, and Deleted are three independent status types for an InterMail account, they are handled the same way by the MTA and the POP Server: delivery attempts result in a bounce notice and retrieval attempts are refused.

If an account is in any one of these three states, the Directory Cache reports the status as “inactive” without saying which specific state the account is in.

See the *Integrated Services Directory Guide* for the original discussion of account status.

### *Limits on Domain Name Length*

InterMail version 4.0 allows a maximum of 64 characters when specifying a domain name.

See the *Integrated Services Directory Guide* for the original discussion of domains.

### *Migration of Post.Office Quotas*

The *InterMail Migration Guide* says that the `-q` (quota) option for `immigacctdump` will migrate all Post.Office quotas to InterMail. However, this is not an automatic process and Post.Office quotas have to be migrated individually, on a per-account basis.

See the *InterMail Migration Guide* for more information.

### *Details on Deleting Domain Information*

There are two methods for deleting domain information from the Integrated Services Directory:

1. Mark the domain as deleted in the Integrated Services Directory, but retain information that points to the deleted domain by typing:

   ```
   imdbcontrol deletedomain <domain>
   ```

   This invocation will disallow use of the specified domain; however, it will leave a referential pointer to the deleted domain (i.e., the domain will be shown as a deleted domain).

2. Erase any mention of the domain from the Integrated Services Directory by typing:

   ```
   imdbcontrol deletedomain <domain> force
   ```

### *Stopping InterMail Servers via the imctrl command (kill vs. exit)*

The `imctrl` command includes both `kill` and `exit` options. The `kill` option is equivalent to the UNIX `kill -9` command and *may* leave the host machine in a corrupted state. If the `exit` option is used `imctrl` employs an RME call which instructs the server to shutdown immediately.

Although the difference between the two options would not be visible to end users (both result in the immediate cessation of service), the `exit` option is recommended as it provides a more graceful means of shutting down the server.

See the *InterMail Reference Guide* for complete discussions of all administration commands.

### *Where to Find Results of the imdbspacequickcheck Command*

Output of the `imdbspacequickcheck` utility is significant as it warns of an impending tablespace crisis. Therefore, it is important to note that the output of this command is *only* recorded in the `imdbspacequickcheck.log`.

InterMail system administrators must monitor the `imdbspacequickcheck.log` for urgent messages like the example that follows:

```
19980806 181659265 mss1 imdbspacequickcheck 20601 1 1
Urgt;DbToolsSpaceEmergencyCrisis(103/14) IMM1 POX02PK_IM_HDRADDRESSINDEX
```

### *Relationships between Database Connection Configuration Keys*

The configuration keys listed below control access to the Directory database. The primary keys control access for general requests, the update keys control access for the update thread, and the expire keys control access for the expire thread.

- `primaryDBuserInfo`
- `primaryDBconnection`
- `expireDBconnection`
- `expireDBuserInfo`
- `updateDBconnection`
- `updateDBuserInfo`

If the either of the expire keys is left unspecified, the expire thread will use the connection specified by the primary keys. Similarly, if the either of the update keys is unspecified, the update thread will use the connection specified by the primary keys. However, it should be emphasized that *if the expire and update keys are specified, they override the values in the primary keys with respect to their particular task.*

See the *InterMail Reference Guide* for a complete discussion of each configuration key.

### *Changes to Header, Body, and Control File Format*

The SMTP Server process generates unique filenames for Control, Body, and Header files. The format for those filenames is as follows:

```
<date.<sequence-string<pid.<mtahost@<helo-hostname-or-ip-Type
```

where `Control`, `Body`, or `Header` is substituted for the variable `Type` as appropriate.

All Control, Header, and Body filenames are created with the HELO/EHLO hostname (if it appears legitimate) or the Peer IP address (if it does not), assuming that one or the other exists.

The HELO hostname is considered legitimate if both the following conditions are met

- the hostname has at least one "." in it
- the first character is not a '`[`' as in `[my.ip.address]`

If the HELO hostname is not considered legitimate, the IP address of the connection is used in the filename.

The following examples illustrate the appearance of this new filename format. Note that the name of the MTA host is included, since multiple MTAs could write to a single Queue Server.

```
19980601171253.AHA6542.mta1@foo-Body
19980601171253.AHA6542.mta1@sbs-devsun1.software.com-Body
19980601171253.AHA6542.mta1@[10.2.6.98]-Body
```

### *Clarification on MTA Acting as Backup for the Queue Server*

Chapter 10 of the *InterMail Operations Guide* states that "if mail needs to be stored temporarily and no Queue Server is available, the MTA assumes this role and stores messages in its local `spool` directory." That statement is true *only* if the `localFallback` configuration key is set to `true`. If the `localFallback` key is set to `false`, the messages will not be accepted.

### *Notes on Message Sidelining*

A new header record, the `Ok-To-Sideline` header, has been created to assist in the processing of sidelined mail.

The `Ok-To-Sideline` header is found in the Control file for each message and can have a value of 0 or 1. A value of 0 indicates that sidelining tests have already been performed and the message either passed the tests or was successfully deposited in the `sideline` directory. A value of 1 indicates that sidelining tests still need to be performed, either because those tests have not yet been executed, or because the message failed the tests but remains in the `deferred` directory because the `sideline` directory was unavailable (for whatever reason).

Use of this header is important for two reasons. First it uses system resources more efficiently by ensuring that messages are tested against sideline criteria only once. Second it facilitates processing of sidelined mail that has been reviewed by the administrator, approved for delivery, and re-submitted for processing via the `immsgprocess` command. (The header value of 1 bypasses subsequent sidelining tests, which would have deposited the message right back in the sidelined directory even after it, had been reviewed and approved!)

### *imoraupgrade4 and imoraupgrade4batch*

Chapter 12 of the *InterMail Reference Guide* describes the `imoraupgrade4` and `imoraupgrade4batch` utilities in great detail. These utilities aid in the InterMail upgrade process from 3.2.x to 4.0.x., but are not used for general administrative functions.

### *Erroneous "@" Symbol*

In the *InterMail Migration Guide*, there is an error in Chapter 2. There is an incorrect reference in the `popProxyHost` configuration key example. Instead of an "`@`" sign, there should be a "`.`" See the following correction:

Set the `popProxyHost` configuration key to the hostname of your source (POP3) system. For example: `/*/popserv/popProxyHost: [jupiter.accordance.com].`

### *Blocking Mail from Non-Existent Local Addresses*

There is an error in Chapter 4 of the *InterMail Operations Guide*. The correct information is provided below:

To prevent the use of non-existent return addresses, make the following changes to the configuration database:

Locate the key `/*/mta/blockAddresses` in the configuration database. If this key is set to `false`, change it to `true`. If the key does not exist in the configuration database, add it as a new configuration entry:

```
/*/mta/blockAddresses: [true]
```

### *Error Handling*

Chapter 6 of the *Integrated Serviced Directory Guide* describes error handling within the InterMail Perl API. However, an inaccurate example of an error handling Perl subroutine is referenced. The correct example is provided below:

```
{
      my ($n, $s);
      if($n = im_errnum()) {
           $s = im_errstr() || im_errormnemonic() ||
                '(no further information)';
           print "** Error $n: $s\n";
           return;
      }
      print "** OK\n";
}
```

### *WebMail Internationalization*

The internationalization steps found in Chapter 5 of the WebMail Manual should be revised as follows:

To change all text in all of the Mail Management Forms to a different language, such as Spanish, do the following:

1. Make a copy of the `WM-lang_English.pl` file in the `httpd/htdocs/webmail` directory:

   `cp WM-lang_English.pl WM-lang_Spanish.pl`

2. Open the `WM-lang_Spanish.pl` file in a plain text editor, such as vi or emacs.

3. Translate the text values for each variable into Spanish. For example, find the line in the file that says "`Name`":

   `Name => 'Name'`

   `Translate the value to "nombre":`

   `Name => 'Nombre'`

4. Save the changes to the `WM-lang_Spanish.pl` file.

5. Next, open the `WM-lang.pl` file in a text editor and insert the appropriate pattern-matching code:

   ```
   @WebMail::languages =
   (
     # { pattern => '/french', file => 'WM-lang_French.pl' },
     # { pattern => '/spanish', file => 'WM-lang_Spanish.pl' },
     # { pattern => '/german', file => 'WM-lang_German.pl' },
     { pattern => '/english', file => 'WM-lang_English.pl' },

     # default language should come last, with a pattern of "."
     { pattern => '.', file => 'WM-lang_English.pl' }
   );
   ```

   The French, Spanish, and German translation pattern-matches are already in the file, but they are commented out with the # sign. Remove the # sign to activate that language.

6. If the language you want to insert is not one of those three, create a new line appropriately:

   ```
   { pattern => '/russian', file => 'WM-lang_Russian.pl' },
   ```

   and insert that line in the file before the "`default`" line.

7. Insert an alias definition in the `httpd` server configuration file. Edit the file, `$INTERMAIL/httpd/conf/httpd_conf` and find the section that starts with the comment:

   ```
   ### Aliases

   ScriptAlias /cgi-bin/ /scratch/imail/httpd/cgi-bin/

   Alias               /icons/         /scratch/imail/httpd/icons/
   AliasMatch          "^/(SelfCare|selfcare|SELFCARE)(.*)"
                       /scratch/imail/httpd/htdocs/InterManager/EU/$2
   AliasMatch          "^/InterManager/(SelfCare|selfcare|SELFCARE)(.*)"
                       /scratch/imail/httpd/htdocs/InterManager/EU/$2
   AliasMatch          "^/(WebMail|webmail|WEBMAIL)(.*)"
                       /scratch/imail/httpd/htdocs/webmail/$2
   AliasMatch   "^/(ConfEdit|confedit|CONFEDIT)(.*)"
                       /scratch/imail/httpd/InterMail/CFG/
   ```

   In that section, there will be multiple lines of the form:

   ```
   AliasMatch  "^/(WebMail|webmail|WEBMAIL)(.*)"
               /imail/imail2/httpd/htdocs/WebMail/$2
   ```

   A new alias should be entered *above* the line shown, since matching is done in order. Insert the following line:

   ```
   AliasMatch  "^/(WebMail|webmail|WEBMAIL)/Russian(.*)"
   /imail/imail2/httpd/htdocs/WebMail/$2
   ```

8. Save that file and exit the editor.

9. Restart the Web Server so the changes take affect:

   ```
   imctrl <hostname> restart httpd"
   ```

   ***Note:*** *You may have as many languages as you choose, simultaneously active for a single WebMail server, and users may select any language they wish by their choice of the URL."*

10. Log on to the WebMail Authentication Form using the appropriate URL:

    `http://<your_domain_name>WebMail/spanish/authenticate.cgi`

11. The URL is case sensitive. In order to change `spanish` to `Spanish`, the `WM-lang.pl` file must be changed from:

    ```
    >    { pattern => '/spanish', file => 'WM-lang_Spanish.pl' },
    ```

    to:

    ```
    >   { pattern => '/Spanish', file => 'WM-lang_Spanish.pl' }
    ```

### WebMail Address Book

Chapter 2 of the *Integrated Services Directory Guide* has a list of class of service attributes. The `pref_webmailaddressbooklistlimit` is incorrectly defined as, "The maximum number of mailing lists allowed in an address book in WebMail." The correct definition is, "The maximum allowed number of addresses in a single address book entry."

### WebMail Signatures

In Chapter 2 of the *Integrated Services Directory Guide*, it says that `pref_webmailusesignature` has a precedence rule of "L." However, the correct rule is "A," as shown in the example below:

| Attribute | Rule | Type | Description |
|---|---|---|---|
| `pref_webmailusesignature` | A | Boolean | Insert Signature in WebMail messages. |

### Correction to imbatchextract Argument

In the *InterManager Guide*, there is an incomplete description for the –o argument for the `imbatchextract` utility. The correct description is as follows:

`-o`

Specifies the object type to be dumped. Valid object types are accounts and classes of service. The parameter used by –o should be a valid account or class of service name.

For example, -o premium would produce records for the premium class of service.

If no object type is specified, then information about all accounts and classes of service is listed in the output file.

### Directory Cache Server Configuration Options

The following configuration key is added to the list of keys in Chapter 4 of the *InterMail Reference Guide*

`cacheAuthoritativeOnDbFail`

Indicates whether or not the information in the directory cache should be considered authoritative if the Integrated Services Directory cannot be reached.

### pref_bypassauthentication

All mentions of the `pref_bypassauthentication` preference should be deleted from Chapter 2 *Integrated Services Directory Guide*.

### sslPop3Port Configuration Key

In the *InterMail Reference Guide*, it erroneously says that if you set the `sslPop3Port` configuration key to `–1`, secure POP Server operations are disabled. However, in order to disable secure POP operations, you must remove the configuration value, as in the following example:

```
/*/popserv/sslPop3Port: []
```

### *logNamedPipeMode Configuration Key*

The *InterMail Reference Guide* provides an incorrect description for the `logNamedPipeMode` configuration key. The correct description is as follows:

**Description:** Determines the behavior of the named pipe that transmits log file information.

If the value is set to zero, a named pipe is not to be created.

If the value is set to 1, a named pipe is created if necessary, and log entries transmitted to the pipe whether or not there is a reader on the pipe.

If the value is set to 2, the named pipe is blocked until there is a reader.

If you plan to use a named pipe, the recommended value is 1.

# 2

# *Understanding Self Care*

SelfCare is a web-based interface for InterMail that allows end users to view and modify selected information related to their e-mail accounts: passwords, forwarding addresses, vacation messages, etc.

Access to SelfCare–and individual options within it–can be controlled through the setting of class of service attributes. The SelfCare interface can also be customized to include site-specific product information, form text, or user interface presentation.

## 2.1 Web Server Configuration

The standard InterMail Web Server supports the SelfCare application. Before enabling SelfCare for end users, certain Web Server configuration options should be noted and modified, if desired.

There are two files that can be directly modified when using SelfCare: the standard InterMail configuration database (config.db) and the Web Server configuration database (httpd.conf).

### 2.1.1 Modifying config.db to Support SelfCare

The `cookieDomain` configuration key should be modified. It should be set to the top level domain of all web server hosts.

For instance, if an ISP had hosts named `www1.isp.com`, `www2.isp.com`, etc. you would set the value to `isp.com` so that the cookie would to be readable by all servers in the network.

### 2.1.2 Modifying httpd.conf to Support SelfCare

The `httpd.conf` file is a text-only configuration file read by the Web Server. You can use any text editor to make changes to this file. Once changes have been made, restart the Web Server (via the `imctrl` command) to reload the configuration values.

The `ServerAdmin` directive should be modified for SelfCare. It should be changed to an actual account for a system administrator or network operator who will receive administrative mail. By default, this directive is set to ‘imail@software.com.”

## 2.2 Enabling SelfCare Access

In order to use the SelfCare interface to manage his or her e-mail account, a user's account must be associated with a class of service that allows SelfCare access. The class of service attribute that allows SelfCare login access is pref_selfcare.

The default class of service does not allow SelfCare login access, so by default, users cannot access SelfCare. To add login access to a class of service, use the `imdbcontrol` utility with the `SetCosAttribute` option to set the required attribute.

For example, to set SelfCare login access for all accounts associated with the default class of service, type the following:

```
imdbcontrol default pref_selfcare 1
```

The value 1 for this boolean attribute indicates 'true' (0 indicates 'false').

Additional class of service attributes determine the options that are available to users within the SelfCare interface. For example, the permission attributes `perm_forwarding` and `perm_localdelivery` control whether users will see the form fields related to mail delivery options. For information on these (and other) class of service attributes, refer to the *Integrated Services Directory Reference Guide*.

## 2.3 SelfCare Files

The following files comprise the SelfCare interface, and are installed in the `httpd/htdocs/EU` directory as part of the InterCore installation:

| File | Description |
|---|---|
| `EU.pm` | Perl module that defines common form code, such as basic layout. |
| `EU_msgs.pl` | File that contains the text displayed on SelfCare web forms. All label text, error messages, and result text in the SelfCare interface is defined in this single file. |
| `account.cgi` | Authentication (login) form. |
| `authenticate.cgi` | the path(s) and file name(s) where documents will be stored |
| `display.cgi` | (Reserved for use with WebMail.) |
| `images/*` | Images used in the SelfCare interface. |
| `password.cgi` | Password options form. |
| pref_saved.cgi | Form that displays confirmation of user changes. |
| rules.cgi | Form used to set auto-replay, mail delivery, and junk mail policies. |
| saveUser.cgi | Script that saves user data to the Integrated Services Directory. |
| signature.cgi | (Reserved for future use with WebMail.) |

The Content of these files is complete as delivered. No customization is required to Use SelfCare in its standard state.

## 2.4 Interface Customization

Customization is the process of modifying the implementation of the SelfCare user interface to provide a desired look and feel. There are many different levels of modification that are possible, depending on the amount of development, maintenance and future integration that you are willing to undertake.

The SelfCare interface is designed to be easily customized. The directions provided in this section provide general guidelines for customizing the interface, but they are not intended to serve as detailed instruction sets. Customizing the interface requires expert knowledge of Perl.

**Warning!** We strongly recommend that you undertake customizing the product only if it is necessary for the success of the product. You must also plan for maintenance and integration work with future versions of InterMail.

The general types of customizations that can be performed are:

- Branding
- User interface presentation
- Form text modifications

Each of these customization techniques are described in the following sections.

### 2.4.1 Branding

Branding of a user interface involves the representation of names and logos of the product and its provider. A common branding operation is replacing the initial product logo graphics with the service provider's name and logo. Branding typically does not require editing of the scripts that define the SelfCare interface forms, and is therefore the easiest type of customization.

Branding is carried out by replacing graphic files in the SelfCare images directory with graphic files created by the service provider.

**Example:**

In this example, we want to replace the default background image used by the SelfCare forms with a site-specific graphic that includes the service provider's logo. To make this change, execute the following steps:

1. Locate the httpd/htdocs/EU/images directory. This is the directory that contains all SelfCare interface images.
2. Move (or rename) the graphics file self-background.gif. This is the background image used for all SelfCare forms. Create your custom graphic, and save it as self-background.gif.
3. Copy your graphic file to the httpd/htdocs/EU/images directory.

After making these changes, all users will see the new background image when viewing SelfCare forms.

***Note**: When replacing graphics, the new file must have the same name (including case) as the replaced file. It is also highly recommended that the new graphic be the identical size (height and width) as the replaced graphic.*

## 2.4.2 User Interface Presentation

User interface presentation customization involves modifications to the way that information is presented in the SelfCare web forms, but not the information itself. This type of customization typically involves changes to the form layout. For example, you may want to change the location of the execution buttons on a form.

Some form components--such as the width of the tab bar--are very easy to change. Others are more complicated, such as replacing the tabs with frames. The code that defines the form presentation is separate from the code used to process input and output, so it is relatively easy to make presentation-related changes.

To make these types of presentation changes, you must edit a Perl module named EU.pm, which contains the presentation-related code. By modifying the contents of this file, you can change the user interface presentation for all SelfCare forms.

**Example:**

In this example, we want to remove the folder tabs at the top of SelfCare forms and replace them with navigation buttons on the left side of the screen.

1. Mock up the new layout using the HTML editor of your choice. You should encapsulate the navigational elements from the rest of the document, which makes the process simpler.
2. Make the appropriate changes to `EU.pm`. These changes require editing the following functions:
   - **print_formatting_top().** Replace the current default background graphics and/or colors with your own. Make any modifications necessary to the HTML table so that your new layout will fit in correctly.
   - **print_navigation().** Replace the contents of this function with your own HTML code that includes the navigation buttons.
   - **print_formatting_bottom().** Close up any tables you opened the first two functions, and generally wrap things up (including the closing </HTML> tag).

If you view the source code to any of the SelfCare web forms (after they have been processed and returned to your browser), you will see a series of comments that indicate where each of these functions has begun and ended its work. They look like:

```
<!-- Begin print_formatting_top() -->
<!-- End print_formatting_top() -->
... etc.
```

From this, you can get a good idea of how the pages are laid out and you will see how the entire look of the web pages can be radically changed by the modification of those routines.

 

## 2.4.3 Form Text Modifications

Another common method of interface customization is form text modifications. This type of customization involves modifying the text that is shown in the SelfCare interface, but not the structure or layout of the forms themselves. Form text modifications are useful for providing additional user help, site-specific information, or translating the SelfCare interface into other languages.

All SelfCare form text--including field labels, error messages, and ALT text shown in place of images--is defined in a single file (EU_msgs.pl). Changing form text is as simple as editing this file, which contains separate sections for each form script and message type. Each line in EU_msgs.pl contains a variable definition like the following:

```
$msg_TitleAccount = "Self Care: Account";
```

The first part of each line (in this case, `$msg_TitleAccount`) specifies the name of a variable contained in a SelfCare form script. When a script is executed, each variable contained in the

script is replaced by the value defined for it in `EU_msgs.pl` (in this case, “Self Care: Account”).

**Example:**

In this example, we want to add site-specific technical information to the message shown to users who enter incorrect login data. To make this change, execute the following steps:

1. Open EU_msgs.pl in a text editor.
2. Locate the section of the file that is preceded by the line

   ```
   # General Messages
   ```
3. Locate the variable named `$msg_AuthFailed` in this section.
4. Modify the value of the variable to include your site-specific information. For example, you might change the default entry

   ```
   $msg_AuthFailed = ’Authentication failed; please try again.’;
   ```

   to

   ```
   $msg_AuthFailed = ’Authentication failed, please try again. If this
   problem continues, contact Mega-ISP Technical Support at 805-555-
   1234.’;
   ```
5. Save and close the file.

After these changes have been made, all subsequent SelfCare users who enter incorrect login data will see the new message.

# 3

# *Understanding the Web Server*

The standard InterMail server set now includes a Web Server. The Web Server operates in support of new web-based applications, SelfCare and WebMail. This chapter introduces the new server and provides detailed information on the following topics:

- the role of the Web Server in mail delivery and end user account management
- Web Server configuration, including available configuration options
- server controls for security and event logging

## 3.1 The Role of the Web Server

The Web Server supports all applications that offer access to InterMail via web browsers. At present, there are two such applications: SelfCare and WebMail.

SelfCare allows end users to modify selected account attributes such as their password and/or vacation message. WebMail provides end users with the ability to send and receive mail via a web browser. WebMail and SelfCare are seamlessly integrated so that WebMail users can access SelfCare options from within the WebMail framework.

The WebMail and SelfCare applications are implemented as Perl scripts that run within the Web Server architecture and interact with other InterMail servers to service requests. Figure 2 shows the relationship between the applications and how they interact with other InterMail components.

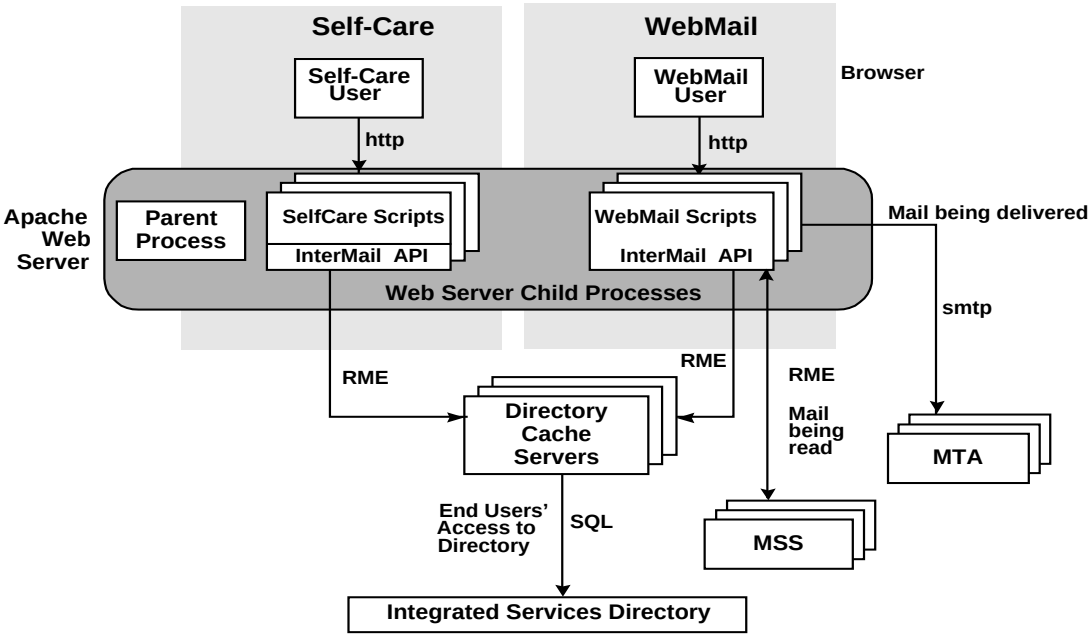


**Figure 2. WebMail and SelfCare.**

The additional components referenced in Figure 2 are described as follows:

- The web browser. which communicates with the InterMail Web Server allowing end user access to InterMail.
- The Web Server which runs the WebMail and SelfCare web applications, communicating with other InterMail servers as necessary and returning relevant results to the web browser.
- The InterMail APIs that access data maintained by the Integrated Services Directory and Message Store databases.
- The WebMail scripts executed by the Web Server to run WebMail. These scripts call the InterMail APIs, which in turn, make calls to the Directory Cache Server, the Message Transport Agent (MTA), and the Message Store Server, to obtain account information, deliver messages, and retrieve mail, respectively.
- The SelfCare scripts executed by the Web Server to run SelfCare. These scripts call the InterMail APIs, which in turn, make calls to the Directory Cache Server to retrieve and write account information.
- The MTA that handles the receipt of all incoming messages and delivers them appropriately.
- The Message Store Server (MSS) responsible for persistent storage of messages.
- The Directory Cache Server that responds to requests for account information.
- The Integrated Services Directory, the centralized database where complete account information is maintained.

---

***Note:*** *Complete descriptions of the operation and architecture of the MTA, the MSS, and the Directory Cache Server can be found in the* InterMail Reference Guide *and the* InterMail Operations Guide. *Details on the Integrated Services Directory and the InterMail APIs are in the* Integrated Services Directory Reference Guide. *For information about WebMail, please see the* WebMail Reference Guide.

---

## 3.2 How the Web Server Works

The InterMail Web Server is an implementation of the popular APACHE web server. Like all APACHE web servers, it uses a parent-child process model of execution. A single, simple parent process handles all network requests. The parent process is responsible for handing requests off to child processes and, when necessary, stopping and restarting child processes.

Each child process receives the request from the designated port (typically port 80) and executes the appropriate SelfCare and/or WebMail scripts. These scripts make calls to the underlying InterMail APIs. The APIs make calls to the Directory Cache Server, the MSS, and the MTA.

The InterMail API maintains a reusable Directory Cache Server connection and a set of reusable MSS connections within each Web Server process to avoid initialization with each request. This speeds up access to the Directory Cache Server and the MSS. Only one Directory Cache Server connection is needed per child process, but since there may be more than one MSS that the Web Server needs to contact, the size of the MSS connection pool is configurable. The configuration key `mssConnPoolSize` specifies the number of connections a single Web Server process should cache between requests.

 

To illustrate the operation of the Web Server in greater detail, the sections that follow outline the steps required for the tasks of mail delivery and message retrieval, two WebMail functions facilitated by the work of the Web Server.

## *Mail Delivery*

This section describes the message and communications flow that occurs while delivering mail via WebMail.

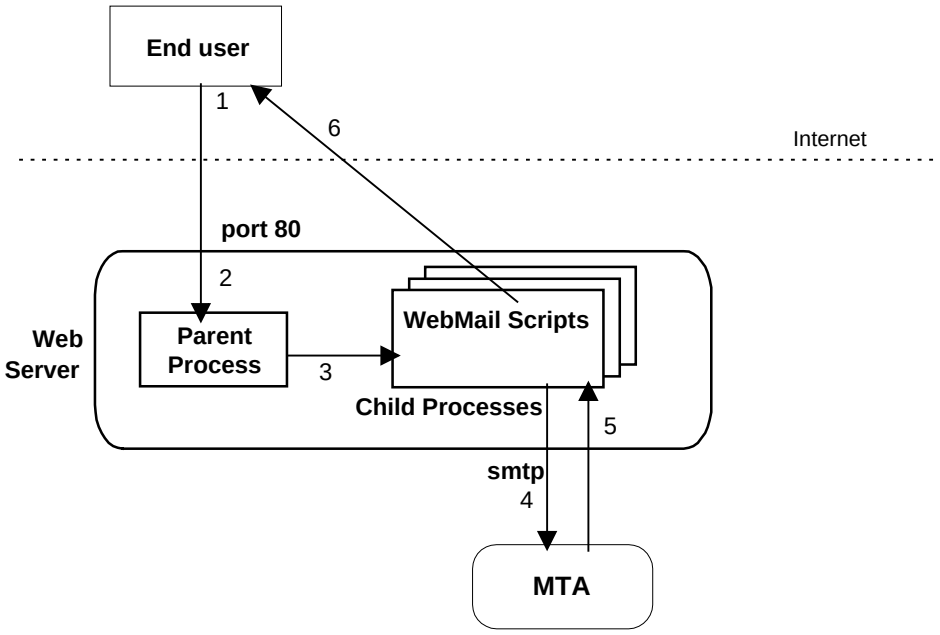


**Figure 3. Mail Delivery.**

When an end user creates and sends mail using the WebMail client, the following steps are executed on the Web Server:

1. The end user passes a `send` request to the Web Server via the WebMail client using the *Compose* HTML form.
2. The `send` request is received by the Web Server, typically via port 80, and handled by a single parent process.
3. The parent process hands off the request to a child process, which executes the WebMail script associated with the `Compose` form identified in the URL.
4. The script associated with the *Compose* form calls an internal routine that sends mail via SMTP to the MTA.

---

***Note:*** *The configuration key `mtaHosts` defines a list of MTA hosts that will deliver mail sent from the Compose form. Each time mail delivery is required, a random entry is picked from the list defined in this key.*

---

5. The MTA signals acceptance of the message.
6. This signal is relayed to the end user.

### *Mail Retrieval*

This section describes the message and communications flow that occurs while retrieving mail via WebMail.

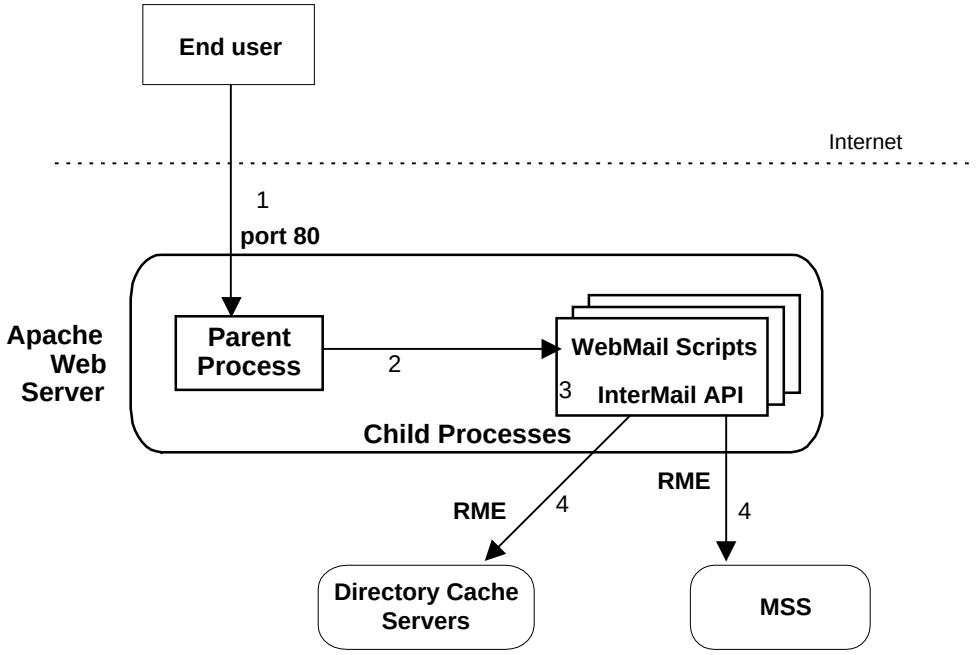


**Figure 4. Mail Retrieval.**

When an end user attempts to retrieve mail using the WebMail client, the following steps are executed on the Web Server:

1. The `Web Server receives the get request`, typically via port 80.
2. The request is handled by a single parent process that hands off the request to a child process.
3. The child process executes the WebMail script, which, in turn, makes calls to the underlying InterMail API.
4. The InterMail API makes calls to the Directory Cache Server to retrieve directory information and to the appropriate MSS to access the user's mailbox.

   The message headers are displayed on the user's screen. To retrieve an entire message, the user clicks on the message header.

## 3.3 Web Server Installation

The Web Server is installed as part of the standard InterMail server set, and all scripts required to run WebMail and SelfCare are installed during the InterCore installation.

As with all the other InterMail servers, the Web Server is installed on each host machine that is part of the InterMail system and can be run on any host or hosts desired.

***Note:*** *Although you can run the Web Server on the same host as any other InterMail server, it is strongly recommended that you* not *do so. Web Servers require more system resources than other InterMail servers, and running them on the same machine as other InterMail servers may negatively impact overall system performance. For information on capacity planning, see the* WebMail Reference Guide.

## 3.3.1 Web Server Directory Structure

After installation, the Web Server's directory structure will appear as illustrated in Figure 5

The root directory is titled `httpd`. The subdirectories in the tree contain most (though not all) of the files related to the Web Server and the web-based applications. Other important files can be found in the `$INTERMAIL/config` directory and the `$INTERMAIL/log` directory.

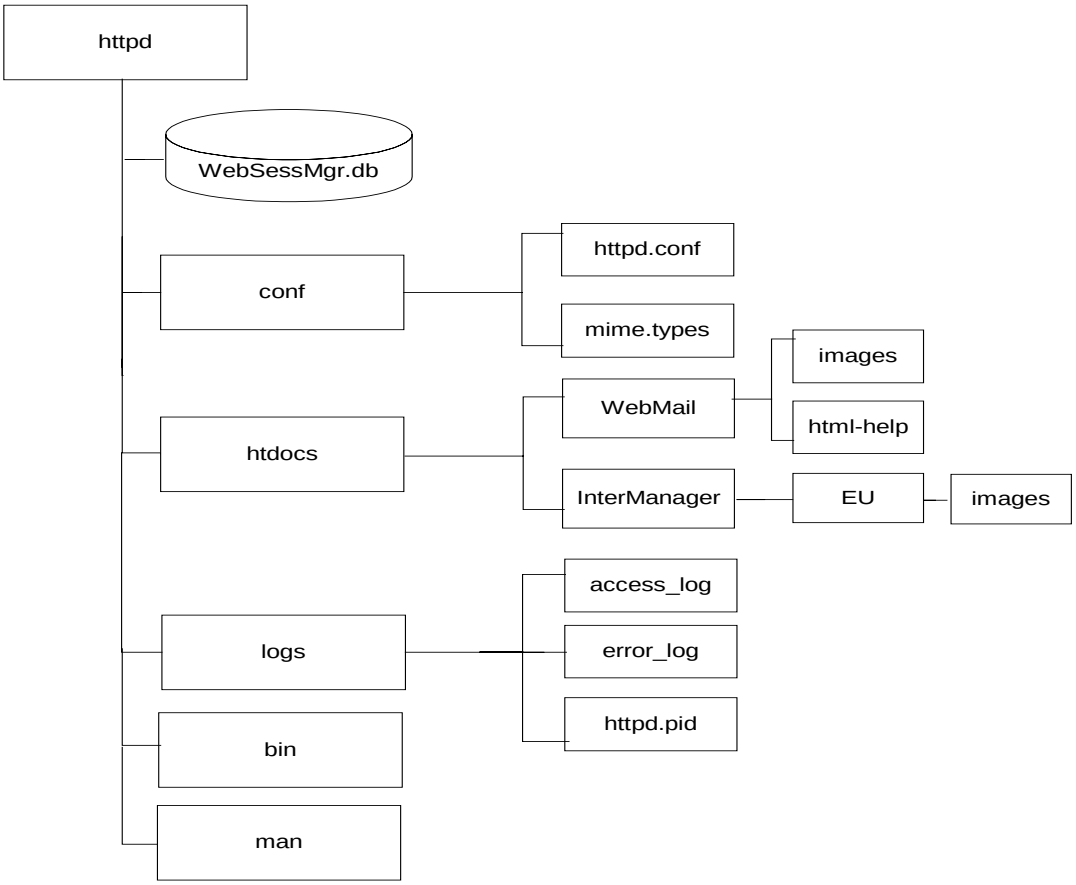


**Figure 5. Web Server Directory Structure.**

The root Web Server directory, `httpd`, contains the following:

- **The Web Session Manager database (`WebSessMgr.db`).** All Web Server transactions are session based. When a user logs in, the Web Session Manager database stores the user's token as well as cached account information from the Integrated Services Directory. When the user ends the session (by logging out), the stored information is deleted.

  ***Note:*** *A session is a series of interactions between a web browser client and the Web Server. It consists of an initial authentication phase followed by a sequence of authenticated interactions between the user and the Web Server.*

- **The `conf` directory** containing the `httpd.conf` and `mime.types` configuration files that control behavior of the Web Server. (See Section 3.4 for instructions on configuring the Web Server.)
- **The `htdocs` directory** with WebMail and InterManager subdirectories containing the Perl scripts and image files for the WebMail and SelfCare applications.
- **The `logs` directory** containing log files generated by the Web Server to record actions, errors, and the Web Server's current process ID (`access_log`, `error_log`, and `httpd.pid`, respectively).
- **The `bin` directory** containing the `httpd` binary and a number of useful utilities. Some of the utilities are part of the standard APACHE distribution, while Software.com developed others specifically for use with WebMail and SelfCare.

  ***Note:*** *The exact content of this directory will change from one InterMail release to the next, as new tools are added or existing tools changed. The function and use of these tools are documented in their respective man pages.*

- **The `man` directory** containing the documentation for `httpd` (the Web Server) and relevant utilities. The man page `httpd(8)` lists the command line arguments for `httpd`.

# 3.4 System Configuration

The Web Server and the applications it supports are independently controlled. The performance and functionality of the web-based applications is controlled via a series of configuration keys located in InterMail's standard Configuration database (the `config.db` file). The behavior of the Web Server, however, is controlled by directives stored in the `httpd.conf` file.

## 3.4.1 Configuring Web-based Applications

Changes to the InterMail configuration keys have no direct bearing on the performance of the Web Server, however they can affect the performance of the SelfCare and WebMail applications.

All configuration keys are contained in the master configuration database file, `config.db`.

The configuration keys that affect the behavior of web-based applications can be identified by the word `web` in the path name, as for example: `/*/web/sessionExpireTimeoutSecs: [1800]`

 

Figure 6 shows the portion of `config.db` that contains the web-related configuration keys.

```
/*/web/banners:
    [images/imap1.gif,http://www.software.com]
    [images/ldap-rev.gif,http://www.software.com]
    [images/pop31.gif,http://www.software.com]
    [images/internet1.gif,http://www.software.com]
    [images/mailserver1.gif,http://www.software.com]
    [images/smtp.gif,http://www.software.com]
/*/web/cookieDomain:     []
/*/web/cookieExpireTimeoutDays: [10]
/*/web/MSSConnPoolSize: [0]
/*/web/mtaHosts: [localhost]
/*/web/sessionDBFilePath: [httpd/WebSessMgr.db]
/*/web/sessionExpireTimeoutSecs: [1800]
/*/web/sessionFlushIntervalSecs: [360]
```

**Figure 6. The Configuration Database.**

These keys, like all other InterMail configuration keys, can be edited via the `imconfedit` utility. Refer to the *InterMail Operations Guide* for a general discussion of system configuration.

## 3.4.2 Configuring the Web Server

The Web Server configuration file (`httpd.conf`) contains a number of *directives* (server controls) whose settings can be adjusted in order to manage the performance of the server. These options allow you to control server-specific tasks, such as setting the maximum number of requests that each `httpd` child process can handle.

Changes to Web Server directives affect the performance of the Web Server itself; however, any change in the performance of the Web Server may also have an indirect effect on the performance of web-based applications running on that Web Server. For example, changes that speed the performance of the Web Server under certain circumstances may, as a *consequence*, enhance the performance of WebMail and/or SelfCare.

### *The httpd.conf File*

The `httpd.conf` file is located in the `httpd/conf` directory, beneath the Web Server's root, and can be edited directly using any text editor. Before attempting to edit `httpd.conf`, it is crucial that you back up the file. This will give you a record of your original settings, and also make it easy to restore them should you be unhappy with the results of your changes.

---

***Note:*** *In most cases, the default settings for these directives are appropriate and do not require alteration. Before attempting to change the values for any of the Web Server's directives, you should be familiar with APACHE configuration procedures in general. If you have not already done so, we suggest you consult the APACHE documentation.*

---

No changes made to `httpd.conf` will take effect until the Web Server is restarted, or until it receives a `SIGHUP` signal.

### *Directive Syntax*

The syntax for most Web Server directives is

```
<directiveName> <directiveValue>
```

as for example:

```
maxRequestsPerChild 30
```

The `<directiveName>` and `<directiveValue>` elements should be separated either by one or more white spaces, or else by a tab.

Directive names are not case sensitive, so any of the following would be acceptable:

- `maxRequestsPerChild`
- `MAXREQUESTSPERCHILD`
- `MaXrEqUeStSpErChIlD`

However, while the *name* of the directive is not case sensitive, its values often are. This is especially true if the value contains a file name or regular expression.

### *Editing Instructions*

The following example illustrates the procedure for changing the value of a Web Server directive. In this case, the `maxRequestsPerChild` value will be raised from `30` to `32`.

1. Create a backup copy of `httpd.conf` by typing the following command:

   ```
   cp httpd.conf httpd.conf.bak
   ```

2. Open `httpd.conf` in any text editor and change the value of the `maxRequestsPerChild` directive to read: `maxRequestsPerChild 32`.
3. Save the edited configuration file.
4. Restart the Web Server by invoking the following command:

   ```
   imctrl <hostname> restart httpd
   ```

---

***Note:*** *You can also issue a* `reset httpd` *command.*

---

## 3.5 Web Server Directives

While values for any of the Web Server directives can be changed, it is recommended that you use only the following directives to enhance Web Server performance.

---

***Note:*** *There are more Web Server directives than those described in this section, however it is not recommended that you attempt to adjust any of the others. For a description of the other Web Server directives, consult the APACHE documentation.*

---

## accessConfig

**Description:** Sets the path for the `accessConfig` file.

This file is read by the server to obtain additional directives after it reads the `resourceConfig` file. The file name is relative to the server root.

This feature can be disabled by setting the value to:

```
accessConfig /dev/null
```

This file may contain any server directive allowed in the server config context.

***Note:*** *The initial value set for this directive is correct. It should* never *be changed.*

**Related Keys:** `resourceConfig`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** a valid path and file name

**Initial Value:** `conf/access.conf`

**Default Value:** none

**Example:**

```
accessConfig conf/access.conf
```

## alias

**Description:** Allows documents to be stored in the local file system rather than under the `documentRoot`. URLs with a (`%-decoded`) path beginning with URL-path will be mapped to local files beginning with `<directory-filename>`.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** the path(s) and file name(s) where documents will be stored

**Initial Value:**

```
alias /icons/    $image_dir/
alias /SelfCare/ $install_root/htdocs/InterManager/EU/
alias /WebMail/
$install_root/htdocs/InterManager/WebMail/
```

**Default Value:** none

**Example:**

```
alias /image /ftp/pub/image
```

## allowOverride

**Description:** When the server finds an `.htaccess` file (as specified by `accessFileName`) it needs to know which directives declared in that file can override earlier access information.

***Note*** *For the most part,* `allowOverride` *should be left at the default setting of* `All`.

**Related Keys:** `accessFilename`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** The name of the directive that will override earlier access information. `All` or `None` can also be used to have all directives or no directives override earlier access information.

Override can be set to `None`, in which case the server will not read the file, `All` in which case the server will allow all the directives, or one or more of the following:

`authConfig:`

Allow the use of the authorization directives: (`authDBMGroupFile`, `authDBMUserFile`, `authGroupFile`, `authAuthName`, `authType`, `authUserFile`, etc.).

`fileInfo:`

Allow the use of the directives controlling document types: (`addEncoding`, `addLanguage`, `addType`, `defaultType`, `errorDocument`, `languagePriority`, etc.).

`indexes:`

Allow the use of the directives controlling directory indexing: (`addDescription`, `addIcon`, `addIconByEncoding`, `addIconByType`, `defaultIcon`, `directoryIndex`, `fancyIndexing`, `headerName`, `indexIgnore`, `indexOptions`, `readmeName`, etc.).

`limit:`

Allow the use of directives controlling host access (`allow`, `deny`, and `order`).

`options:`

Allow the use of directives controlling specific directory features (`options` and `xBitHack`).

***Note:*** *The other directives referenced in this section are described in detail in the Apache documentation*.

**Initial Value:** `All`

**Default Value:** `All`

**Example:** `allowOverride None`

## documentRoot

**Description:** Sets the directory from which `httpd` will serve files. Unless matched by a directive like `alias`, the server appends the path from the requested URL to the document root to make the path to the document.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** should be the directory from which the Web Server will serve files

**Initial Value:** `/vol1/imail/httpd/htdocs`

**Default Value:** none

**Example:** `documentRoot /usr/web`

## <directory> … </directory>

**Description:** `<Directory>` and `</Directory>` are used to enclose a group of directives which will apply only to the named directory and sub-directories of that directory.

Any directive that is allowed in a directory context may be used. `<directory>` is either the full path to a directory, or a wild-card string. In a wild-card string, `'?'` matches any single character, and `'*'` matches any sequences of characters.

As of Apache 1.3, you may also use `'[]'` character ranges like in the shell. Also as of Apache 1.3, none of the wildcards match a `'/'` character, which more closely mimics the behavior of UNIX shells.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** a group of directives which apply only to the named subdirectory

**Initial Value:**

```
<Directory /imail/imail/httpd/htdocs>
Order Deny,Allow
Allow from All
Options +ExecCGI
</Directory>
```

***Note:*** *This is just one of the initial values, see `httpd.config` for others.*

**Default Value:** none

**Example:**

```
<Directory /usr/local/httpd/htdocs>
Options Indexes FollowSymLinks
</Directory>
```

## errorLog

**Description:** Sets the name of the file to which the server will log any errors it encounters. If the filename does not begin with a slash (/) then it is assumed to be relative to the `serverRoot`.

Using `syslog` instead of a filename enables logging via `syslogd(8)` if the system supports it. If the first character of the argument is a pipe (`|`), the rest of the argument indicates the program to which you want the messages piped.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** a valid path name to the `error_log` file.

**Initial Value:** `/vol1/imail/httpd/logs/error_log`

**Default Value:** `logs/error_log`

**Example:** `errorLog /vol1/imail/httpd/logs/error_log`

## <location> … </location>

**Description:** The `<Location>` directive provides for access control by URL. It is similar to the `<Directory>` directive, and starts a subsection which is terminated with a `</Location>` directive. `<Location>` sections are processed in the order they appear in the configuration file, after the `<Directory>` sections and `.htaccess` files are read, and after the `<Files>` sections.

Note that URLs do not have to line up with the file system at all, it should be emphasized that <Location> operates completely outside the file system.

For all origin (non-proxy) requests, the URL to be matched is of the form `/path/`, and you should not include any `http://servername prefix.` For proxy requests, the URL to be matched is of the form `<scheme://servername/path>`, and you must include the prefix.

The URL may use wildcards In a wild-card string, `'?'` matches any single character, and `'*'` matches any sequences of characters.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** the URL for access control

**Initial Value:**

```
<Location /server-status>
SetHandler server-status
Order Deny,Allow
Deny from All
### To enable access to the server-status feature,
uncomment the following
### lines and edit the domain as appropriate:
# HostnameLookups on
# Allow from foo.bar
</Location>
```

***Note:*** *This is just the initial appearance of a location block. You need to edit it as appropriate.*

**Default Value:** none

**Example:** See Initial Value

## maxClients

**Description:** Sets a limit on the number of simultaneous requests that can be supported; not more than this number of child server processes will be created.

The memory consumed by each `httpd` process is proportional to the number of requests it serves. This parameter should be tuned based on the amount of memory available on your system.

A value of 32 is recommended.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any positive integer

**Initial Value:**

**Default Value:**

**Example:** `maxClients 32`

## maxRequestsPerChild

**Description:** Sets the limit on the number of requests that an individual child server process will handle. After the `maxRequestsPerChild` requests limit is reached, the child process will die. If `maxRequestsPerChild` is set to `0`, there is no limit on the number of requests a child process will handle.

Setting a non-zero limit establishes a finite lifetime for each process and thereby reduces the number of processes when the load on the server diminishes.

The recommended setting is `30`, however you may wish to experiment with other settings.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart optional

**Possible Values:** any non-negative integer (including zero)

**Initial Value:** `30`

**Default Value:** `0`

**Example:** `maxRequestsPerChild 30`

## maxSpareServers

**Description:** Establishes an upper limit on the number of idle child server processes. (An idle process is one that is not handling a request.) If there are more than maxSpareServers idle, then the parent process will kill off the excess processes.

A value of 10 is recommended.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any positive integer

**Initial Value:**

**Default Value:**

**Example:** `maxSpareServers 10`

## minSpareServers

**Description:** Establishes a lower limit on the number of idle child server processes. If there are fewer than minSpareServers idle, then the parent process will create new children at a maximum rate of 1 per second.

A value of 5 is recommended.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any positive integer

**Initial Value:**

**Default Value:**

**Example:**
```
minSpareServers 5
```

## passEnv

**Description:** Specifies one or more environment variables to pass to CGI scripts from the server's own environment. You may add entries to this list, but do not remove any existing entries.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** one or more environment variables to pass to CGI scripts

**Initial Value:**
```
INTERMAIL PERLHOME NLSPATH LD_LIBRARY_PATH
```

**Default Value:** none

**Example:**
```
passEnv LD_LIBRARY_PATH
```

## pidFile

**Description:** Sets the file to which the server records the process id of the daemon. If the filename does not begin with a slash (/) then it is assumed to be relative to the `serverRoot`.

It is often useful to be able to send the server a signal, so that it closes and then reopens its `errorLog` and `transferLog`, and re-reads its configuration files. This is done by sending a `SIGHUP (kill -1)` signal to the process id listed in the `pidFile`.

***Note:*** *The initial value set for this directive is correct. It should never be changed.*

**Related Keys:** `serverRoot`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** a valid path to `pidFile`

**Initial Value:** `/vol1/imail/httpd/logs/httpd.pid`

**Default Value:** `logs/httpd.pid`

**Example:**
```
pidFile logs/httpd.pid
```

## resourceConfig

**Description:** After reading the `httpd.conf` file, the server will read the file specified in this directive, and use any additional directives found there. Filename is relative to the `serverRoot`. This feature can be disabled by entering:

```
resourceConfig /dev/null
```

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** the path and name for the file containing additional configuration directives.

**Initial Value:** `conf/srm.conf`

**Default Value:** `conf/srm.conf`

**Example:**
```
resourceConfig conf/srm.conf
```

## scriptAlias

**Description:** The `scriptAlias` directive has the same behavior as the `alias` directive, except that in addition it marks the target directory as containing CGI scripts. URLs with a `%-decoded` path beginning with `<url-path>` will be mapped to scripts beginning with directory-filename.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** the path(s) and file name(s) where documents will be stored

**Initial Value:** `scriptAlias /cgi-bin/ /imail/imail/httpd/cgi-bin/`

**Default Value:** none

**Example:** `scriptAlias /cgi-bin/ /web/cgi-bin/`

## scriptLog

**Description:** Sets a CGI script error `logfile`. If no `scriptLog` is defined, no CGI script error log is created. If defined, any CGI errors are logged into the filename given as the argument. If this is a relative file or path it is taken relative to the server root.

This log will be opened as the user specified in the main `user` directive. This means that either the directory the script log is in needs to be writable by that user, or the file needs to be manually created and set to be writable by that user. If you place the script log in your main `logs` directory, *do not* change the directory permissions to make it writable by the user for the child processes.

***Note:*** *Script logging is meant to be a debugging feature when writing CGI scripts, and is not meant to be activated continuously on running servers. It is not optimized for speed or efficiency, and may have security problems if used in a manner other than that for which it was designed.*

**Related Keys:** `scriptLogLength`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** the name of the script error log file

**Initial Value:** none

**Default Value:** none

**Example:** `ScriptLog /imail/imail/httpd/logs/cgi_log`

## scriptLogLength

**Description:** Limits the size of the CGI script `logfile`. Since the `logfile` may log a great deal of information per CGI error (all request headers and all script output) it can grow quite large. This directive can be used to prevent problems due to unbounded growth of the CGI `logfile`. If the file exceeds this size, no more information is written to it.

Values are in bytes.

**Related Keys:** `scriptLog`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** the maximum number of bytes to which the `logfile` may grow.

**Initial Value:** `1000000`

**Default Value:** `1000000`

**Example:**
```
scriptLogLength 1000000
```

## serverAdmin

**Description:** Sets the e-mail address that the server includes in any error messages it returns to the client.

This directive should be set to an actual account for a system administrator, or for a network operator who will receive administrative mail.

It may be worth setting up a dedicated address just for this purpose.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** a valid e-mail address at this domain

**Initial Value:** none

**Default Value:** none

**Example:**
```
serverAdmin johndoe@software.com
```

## serverName

| | |
|---|---|
| **Description:** | Sets the hostname of the server. This is only used when creating redirection URLs. If it is not specified, then the server attempts to deduce the server name from its own IP address. |
| **Related Keys:** | none |
| **Servers Affected:** | Web Server |
| **Change Impact:** | server restart required |
| **Possible Values:** | the hostname of the Web Server |
| **Initial Value:** | the value written in InterMail's `config/hostname` file |
| **Default Value:** | none |
| **Example:** | `serverName lex-devsun.software.com` |

## setHandler

| | |
|---|---|
| **Description:** | When placed into an `.htaccess` file or a <Directory> or <Location> section, this directive forces all matching files to be parsed through the handler given by <handler-name>. For example, if you had a directory you wanted to be parsed entirely as image map rule files, regardless of extension, you might put the following into an `.htaccess` file in that directory:<br>`setHandler imap-file`,<br>where<br>`imap-file` is the <handler-name> |
| **Related Keys:** | none |
| **Servers Affected:** | Web Server |
| **Change Impact:** | server restart required |
| **Possible Values:** | a handler name |
| **Initial Value:** | `server-status` |
| **Default Value:** | none |
| **Example:** | `setHandler server-status` |

## transferLog

**Description:** Adds a log file in the format defined by the most recent `logFormat` directive, or in `commonLogFormat` (see the Apache documentation) if no other default format has been specified.

This should be a filename relative to the `serverRoot`.

If the first character of the argument is a pipe (`|`), the rest of the argument is then the program to which you want to pipe the messages.

***Note:*** the new program will not be started for a *`virtualHost`* if it inherits the *`transferLog`* from the main server.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** the name of the log file to be added

**Initial Value:** `/vol1/imail/httpd/logs/access_log`

**Default Value:** none

**Example:**
```
transferLog /vol1/imail/httpd/logs/access_log
```

## typesConfig

**Description:** Sets the location of the mime types configuration file. Filename is relative to the `serverRoot`. This file sets the default list of mappings from filename extensions to content types; changing this file is not recommended. Use the `addType` directive instead (see the Apache documentation). The file contains lines in the format of the arguments to an `addType` command:

`mime-type extension` extension

The extensions are lowercased. Blank lines, and lines beginning with a hash character `('#')` are ignored.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** the location of the `mime.types` file.

**Initial Value:** `/vol1/imail/httpd/conf/mime.types`

**Default Value:** `conf/mime.types`

**Example:**
```
typesConfig /vol1/imail/httpd/conf/mime.types
```

# 3.6 Security

The Web Server ensures security for user data by session management. A session is the series of interactions between a client running a Web browser and the Web Server. It consists of an initial authentication phase followed by a sequence of authenticated interactions between the user and the Web Server. Session management is the means of retaining information associated with the session for the duration of the session. This enhances security by ensuring that unattended browsers will not normally grant access to a user's mail account.

## 3.6.1 Managing User Sessions

Session information is stored in the session database file. The path and name of this file should be specified in the `sessionDBFilePath` configuration key. The default value for this key is `$INTERMAIL/httpd/WebSessMgr.db`. Any path that you enter here is interpreted to be relative to `$INTERMAIL`. For example, to set the name and path of the session database file to `httpd/WebSessionMgr.db`, you would enter the following:

```
/*/web/sessionDBFilePath: [httpd/WebSessionMgr.db]
```

If the user does not make any requests to the Web Server for some configurable amount of time, his/her session data will expire and be deleted from the session database.

Two keys control the expiration of session data:

- `sessionExpireTimeoutSecs`
- `sessionFlushIntervalSecs`

To set the length of inactivity after which the session data will be expunged, use the `sessionExpireTimeoutSecs` configuration key. The value for this key is expressed in seconds. For example, to allow 30 minutes (1800 seconds) of inactivity before expiring a user session, you would enter the following:

```
/*/web/sessionExpireTimeoutSecs: [1800]
```

To specify the amount of time between flushes, use the `sessionFlushIntervalSecs` configuration key. A flush occurs when the system goes through the session database and looks for inactive sessions. When such sessions are found, they are expired. For example, to set the flush interval to five minutes, enter the following:

```
/*/web/sessionFlushIntervalSecs: [300]
```

If end users wish to continue use of WebMail or SelfCare after a session has expired, they must repeat the authentication process to establish a new session. The application attempts to resume the new session at the same place the user was when his/her previous session expired.

### 3.6.2 Security for Password Information

After authenticating successfully for the first time, it is possible for the user to bypass the login screen and authenticate automatically for subsequent sessions. This feature can be set using the `perm_bypassauthentication` class of service attribute. If this feature is set, the Web Server can instruct the web browser to store the user's password as a cookie on the user's machine. The next time the user logs in to WebMail or SelfCare, the Web Server fetches the user's username and hashed password from the cookie, without requiring the user to log in again.

---

***Note**: This feature will not work if the end user's browser has disabled the acceptance of cookies.*

---

Security for cookie information can be provided by expiring cookies after a specified amount of time has elapsed. The `cookieExpireTimeoutDays` specifies the number of days before a cookie expires. For example, to set the expiration limit to ten days, enter the following:

```
/*/web/cookieExpireTimeoutDays: [10]
```

## 3.7 Logging

Log files aid in debugging by recording the steps that led to an event or problem. You can also use information in log files to monitor system performance.

Because the Web Server incorporates third-party software, there are two different logging mechanisms supported:

- Standard logging stored in the `$INTERMAIL/log` directory that contains the log files for all InterMail servers and applications.
- Web Server logging stored in the `access_log, error_log`, and `httpd.pid` files in `$INTERMAIL/httpd/logs` directory.

### 3.7.1 Standard InterMail Logging

The web-based applications generate standard InterMail log files that keep a running record of operations and connections to and from InterMail servers. These log files roll over after a configurable period of time and a new log file is started.

The file `web.log` contains a reference to the current log file for the Web Server. Archived logs are similarly referenced in files that are identified by the prefix `web`, a host name, a date/time stamp, and the log type (for example, `web.venus.199807240000-0700.log`).

Standard InterMail logging controls apply to these files. Refer to the *InterMail Operations Guide* for a discussion of available logging options.

## 3.7.2 Web Server Logging

Log files generated by the Web Server itself are written to the following two files in the `httpd/logs` directory:

- `access_log` contains log entries for every Web Server action from the time at which the log file was started. This log file is defined by the `transferLog` directive.
- `error_log` contains a record of all Web Server errors from the time at which the log file was started.

These files record entries pertaining to all InterMail web applications that use the Web Server.

The example that follows suggests a possible entry in the `error_log`.

```
[Thu Jul 23 10:39:06 1998] access to
/htdocs/InterManager/EU/help.html failed for venus.software.com,
reason: File does not exist
```

Entries in the `access_log` file use the common log format that is standard across most web servers. The format is as follows:

```
remotehost rfc931 authuser [date] <request> <status> <bytes>
```

where:

| | |
|---|---|
| `remotehost` | Remote hostname (or IP number if DNS hostname is not available. |
| `rfc931` | Remote logname of the user. |
| `authuser` | Username of the user. |
| `[date]` | Date and time of the request. |
| `<request>` | Request line exactly as issued by the client. |
| `<status>` | HTTP status code returned to the client. |
| `<bytes>` | Content-length of the document transferred. |

Figure 7 shows an example of a message in the `access_log` file.

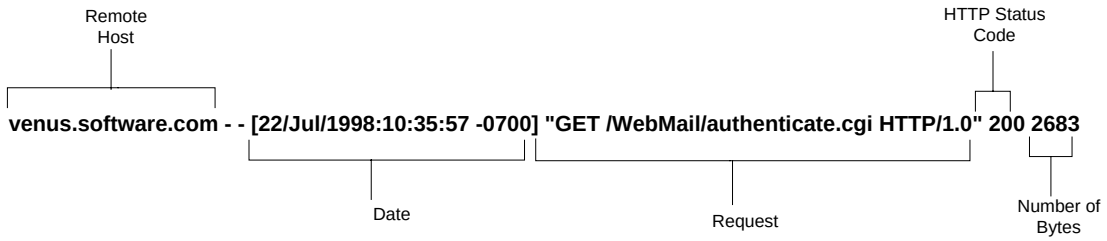


**Figure 7. An entry in the access_log file.**

### Resetting Log Files

Log files can grow very quickly. The `access_log` file typically grows by 1Mb for each 10,000 requests. Therefore, it is necessary to roll over log files periodically and start a new log file.

### The rotatelogs Utility

The `rotatelogs` program in the `ServerRoot/bin` directory is a good filter for piping `httpd` messages if you wish to control the length of the log files by "chopping" and saving them periodically. For details on using `rotatelogs`, see the man pages for that program.

### Customizing Log Files

Logging operations can be customized to provide additional information, such as the user's browser type. Two directives are available for this purpose; `transferLog` creates a log file, and `logFormat` specifies a custom format.

---

***Note:*** *A third directive, `customLog` can be used to define both a log file and format using a single directive (see the Apache documentation for an explanation of the `customLog` directive).*

---

The `transferLog` and `customLog` directives can be used multiple times in each server to cause each request to be logged to multiple files. By default, log files created by `transferLog` are in standard Common Log Format (CLF) unless otherwise specified in the `logFormat` directive.

### CGI Script Error Logs

The `scriptLog` directive offers the option of defining a script log file. This file records errors in the execution of CGI scripts.

While there is no way to pipe these error messages to a filter, you can limit the length of the file by using the `scriptLogLength` directive. No further CGI script execution errors will be written to the script log file after the file length defined in `scriptLogLength` is reached.

# 4

# *Additional Configuration Keys*

This chapter supplements the Configuration Key chapter in the *InterMail Reference Guide*. It includes updated information for previously documented configuration keys and complete descriptions for keys not documented in the original manual. The format for full descriptions is defined in the table that follows.

## sampleKeyName

| | |
|---|---|
| **Description:** | Explains the purpose of the key, describes the format of key entries, and provides suggested settings where appropriate. |
| **Related Keys:** | Lists additional configuration keys (if any) that work together with this key to achieve a specific result. |
| **Servers Affected:** | Indicates the InterMail servers affected by this key. |
| **Change Impact:** | Describes the implications of changing the value of a particular configuration key. |
| **Possible Values:** | Describes allowable values for a key, e.g., `true` or `false`, a text string, an integer, etc. |
| **Initial Value:** | Defines the "starter" entry that is set for this option when the server is installed. |
| **Default Value:** | Specifies the value the system will insert if an explicit value has not been set for this particular option.<br>A "null" entry indicates that no default value exists. |
| **Example:** | Sample syntax for the configuration key, including the complete configuration hierarchy, the key name, a colon, a space, and the value of the key enclosed in square brackets. For example:<br>`/*/web/SampleKeyName: [12]` |

InterMail allows wildcard matching when entering IP address values for a configuration key. `0.0.0.0` will match any IP address since `0` is used as the wildcard value. For example, if `0.0.0.0` is used as the IP address value for the `relaySourceRemoteIPList` configuration key, anyone is allowed to relay.

Another use of wildcard matching is found in the following example:

```
10.2.7.0
```

In this example the trailing 0 is stripped off, thereby making `10.2.7.0` the same as `10.2.7.*`.

# 4.1 List of New/Modified Keys

The configuration keys documented in this section are listed alphabetically by key name. If you do not find a listing for the key you seek, please refer to the *InterMail Reference Guide* for additional information.

## allowedIPs

**Description:** Provides a list of addresses in CIDR notation [x.x.x.x/x] that are allowed to connect to the POP Server. Those that cannot connect are instantly dropped. If the list is empty, or the configuration key is not present, there are no restrictions on connections.

**Related Keys:** `none`

**Servers Affected:** POP Server

**Change Impact:** Trivial, no server restart required

**Possible Values:** list of entries, or null

**Initial Value:** `null`

**Default Value:** `null`

**Example:** `/*/popserv/allowedIPs: []`

## alwaysTryFirst (modified)

**Description:** Indicates whether or not immediate mail delivery should be attempted for domains that are already queuing mail.

In standard InterMail operation, if a queue already exists for a particular domain, subsequent mail that arrives for that domain is automatically added to the queue, with delivery re-attempted at intervals defined by the value of the `outboundDeferProcessInverval` configuration key. This behavior can be altered by using the `alwaysTryFirst` and `alwaysTryFirstList` configuration keys.

There are four possible settings for the `alwaysTryFirst` configuration key. The first two, `false` and `true`, affect *all* domains that are queuing mail. The other two settings, `allowlisted` and `denylisted`, affect only a limited set of domains, those listed in the `alwaysTryFirstList` configuration key. The settings and their effect on system behavior are as follows:

- If the value of `alwaysTryFirst` is set to `false` , InterMail will never attempt immediate delivery of a message destined for any domain that is already queuing mail. Instead, additional messages for such domains will be added to the existing queues and held until the next deferred mail processing interval. Use of this setting conserves system resources by limiting delivery attempts to domains whose mail host may be temporarily out of service.

 

- If the value of `alwaysTryFirst` is set to `true`, InterMail will always attempt immediate message delivery of all messages. A message will be added to a queue only after InterMail learns that the mail host for the desired domain is unavailable.
- If the value of `alwaysTryFirst` is set to `allowlisted`, the system can accommodate immediate delivery to a specified subset of the domains that are queuing mail. If the domain to which the message is addressed is listed among the entries in the `alwaysTryFirstList` configuration key, delivery will be attempted immediately regardless of queuing status. However, if mail is already queuing for a domain and that domain name is *not* listed among the values for the `alwaysTryFirstList` key, InterMail will simply add the message to the existing queue without attempting immediate delivery.
- If the value of `alwaysTryFirst` is set to `denylisted`, the system can accommodate immediate delivery to most domains while denying this option to a specified subset of the domains that are already queuing mail. If mail is already queued for the domain to which the message is addressed *and* the name of that domain appears among the values for the `alwaysTryFirstList` key, InterMail will simply add the message to the existing queue without attempting immediate delivery. For all other messages, delivery will be attempted immediately.

***Note:*** *The alwaysQueue configuration key takes precedence over the alwaysTryFirst key. If the alwaysQueue configuration key is set to true, the value in alwaysTryFirst will be ignored.*

**Related Keys:** `alwaysTryFirstList`

**Servers Affected:** MTA

**Change Impact:** trivial, no server restart required

**Possible Values:** `true, false, allowlisted, denylisted`

**Initial Value:** `false`

**Default Value:** `false`

**Example:**
```
/*/mta/alwaysTryFirst: [allowlisted]
```

## alwaysTryFirstList

**Description:** Specifies a list of domains for which the system will allow or disallow immediate delivery attempts (when mail is already queued for these domains). Multiple entries are allowed, however each listed domain must appear on its own line, between its own set of square brackets.

This key works in conjunction with the `alwaysTryFirst` configuration key. If `alwaysTryFirst` is set to `allowlisted`, then InterMail will always attempt immediate delivery for messages addressed to any domain in this list, even if mail is already queued for that domain. If `alwaysTryFirst` is set to `denylisted`, then InterMail will never attempt immediate delivery for messages addressed to any domain in this

list if that domain is already queuing mail. Instead, new messages will simply be added to the domain's existing queue.

***Note:*** *If `alwaysTryFirst` is set to either `true` or `false`, then the domains listed in `alwaysTryFirstList` have no bearing on the behavior specified in `alwaysTryFirst`.*

**Related Keys:** `alwaysTryFirst`

**Servers Affected:** MTA

**Change Impact:** trivial, no server restart required

**Possible Values:** a list of domains

**Initial Value:** `null`

**Default Value:** `null`

**Example:**
```
/*/mta/alwaysTryFirstList: [software.com]
                            [hardware.com]
```

## badPasswordDelay

A value of 0 (zero) means there will be no delay between re-authentication attempts when the user submits a bad password.

## banners

**Description:** Defines a list of banners (anonymous advertisements in the form of banner graphics). Each banner is identified by a separate entry. Multiple entries are allowed; however, each entry must appear on its own line, between its own set of square brackets.

Each entry must be in the form:

```
<image>,<link-url>
```

where:

`<image>` The path to the file containing the banner image.

`<link-url>` URL indicating the destination to which the user will be taken after clicking on the banner image.

***Note****: Both entries may be set relative to the WebMail directory. If you need a comma in either URL, use %2C (escaped comma) instead.*

**Related Keys:** `none`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** One or more entries, each consisting of a valid file path, a comma, and a valid URL.

**Initial Value:** `null`

**Default Value:** `null`

**Example:**
```
/*/web/banners: [ads/ad1.gif,http://www.software.com]
```

```
[ads/ad2.gif,http://www.hardware.com]
[ads/ad3.gif,http://www.wetware.org]
[ads/ad4.gif,http://www.adbusters.org]
```

## bucketizeMtaDefer

**Description:** Defines whether or not the `deferred/MTA` directory uses buckets. This feature is implemented in two places: the MTA and the Queue Server. For this reason, it is required that you upgrade both the MTA and the Queue Server to Mx 4.1.2 before enabling the key.

***Note:*** *This configuration key is really meant only for transition from off to on. While it is possible to switch back (on to off), it would require additional Software.com assistance.*

**Related Keys:** `None.`

**Servers Affected:** MTA and Queue Server

**Change Impact:** Server restart required.

**Possible Values:** true/false

**Initial Value:** `false`

**Default Value:** `false`

**Example:** `/*/common/bucketizeMtaDefer: [false]`

## cacheAlwaysReadThru

**Description:** This key instructs the Directory Cache Server to always read-through to the Integrated Services Directory when performing lookups, rather than reading first from the cache.

***Note:*** *This key does not supercede `cacheDisableReadThrus`, nor is it superceded by `cacheDisableReadThrus`. If both keys are set to true, then the Directory Cache Server will always respond with `MS_UNKNOWNUSER`.*

**Related Keys:** `CacheDisableReadThrus`

**Servers Affected:** Directory Cache Server

**Change Impact:** When activated, requests to the Directory Cache Server will automatically read-through to the Integrated Services Directory rather than first looking at the local cache.

**Possible Values:** `true`- Only read through to the Integrated Services Directory.

`false` - Look to the local cache first, before reading through to the Integrated Services Directory.

**Initial Value:** `false`

**Default Value:** `false`

**Example:** `/*/imdircacheserv/cacheAlwaysReadThru: [false]`

## cacheAuthoritativeForMTA

**Description:** Determines whether or not information in the directory cache file should be considered authoritative for account queries from the MTA.

If the value of the `cacheAuthoritativeForMTA` configuration key is set to `false` and the Directory Cache Server does not find requested information in its local cache file, the server will attempt to read through to the Integrated Services Directory to obtain the necessary information.

If the value of the `cacheAuthoritativeForMTA` configuration key is set to `true`, cached information will be considered authoritative and no attempt will be made to read through to the Integrated Services Directory.

**Related Keys:** `cacheAuthoritativeOnDbFail`

**Servers Affected:** Directory Cache Server

**Change Impact:** trivial, no server restart required

**Possible Values:** `true` or `false`

**Initial Value:** `false`

**Default Value:** `false`

**Example:** `/*/imdircacheserv/cacheAuthoritativeForMTA: [false]`

## cacheDisableReadThrus

**Description:** Under normal circumstances, the Directory Cache Server reads through to the Integrated Services Directory if it cannot find a particular user or the proper password in its local cache. However, if a large amount of users or passwords cannot be located, CPU resources may be drained, and the option to disable read-thrus should be considered.

When read-thrus are disabled, the Directory Cache Server is considered the authoritative source of information.

If this key is set to true, no read-thrus to the Integrated Services Directory will take place. If set to false, the Directory Cache Server will function as normal.

This key takes precedence over the `cacheAuthoritativeForMTA and cacheAuthoritativeOnDbFail` configuration keys.

**Related Keys:** `cacheAuthoritativeOnDbFail, cacheAuthoritativeForMTA, cacheDisableWriteThrus`

**Servers Affected:** Directory Cache Server

**Change Impact:** no read-thrus to the directory will take place when set to true.

**Possible Values:** true, false

**Initial Value:** `false`

**Default Value:** `false`

**Example:** `/*/imdircacheserv/cacheDisableReadThrus: [false]`

## cacheDisableWriteThrus

**Description:** Under normal circumstances, the Directory Cache Server writes through to the Integrated Services Directory if it cannot find a particular user or the proper password in its local cache. However, if a large amount of users or passwords cannot be located, CPU resources may be drained, and the option to disable write-thrus should be considered.

When write-thrus are disabled, the Directory Cache Server is considered the authoritative source of information.

If this key is set to true, no write-thrus to the Integrated Services Directory will take place. If set to false, the Directory Cache Server will function as normal.

**Related Keys:** `cacheDisableReadThrus`

**Servers Affected:** Directory Cache Server

**Change Impact:** no write-thrus to the directory will take place when set to true.

**Possible Values:** true, false

**Initial Value:** `false`

**Default Value:** `false`

**Example:** `/*/imdircacheserv/cacheDisableWriteThrus: [false]`

## checkAuthentication

In the *InterMail Reference Guide*, it states that if `checkAuthentication` is set to `true`, authenticated SMTP login is required. In addition, bogus addresses will be denied if `checkAuthentication` is set to `true`.

If this is set to `false`, bogus addresses (`MAIL FROM:` and `From:` address lines) are allowed and SMTP authentication is not required for any address.

## configRecorderSize

**Description:** Adjusts (only upward) the size of the recorder buffer used for recording the current configuration values being used by InterMail. Since this buffer is not written to any file, it will grow as needed to accommodate the key/value pairs.

**Related Keys:**

**Servers Affected:** All servers.

**Change Impact:**

**Possible Values:** any non-negative integer (zero disables the recorder buffer).

**Initial Value:** `10240`

**Default Value:** `10240`

**Example:** `/*/common/configRecorderSize: [10240]`

## cookieDomain

**Description:** Specifies the domain suffix to which autologin cookies may be passed.

A cookie is a file on the user's computer that contains authentication information. It allows the user to perform a series of actions without having to manually authenticate each for each action or subsequent log in. (However, the cookie does expire automatically if it is not refreshed within the time period defined by the `cookieExpireTimeOutDays` configuration key.)

The value for this key can be specified as the simple domain to which the Web Server host belongs (e.g., `software.com`), or it may be specified as a particular host within that domain (e.g., `venus.software.com`).

If a host name is included in the value for this key, the cookie will only work for a Web Server referencing that specific host. For example, if the value is specified as `venus.software.com`, the cookie will only work for the host `venus.software.com`. It will not work for `mercury.software.com`.

If no host name is included, the cookie will work for any Web Server host in the domain. For example, if the value is specified as `software.com`, the cookie will work for `mercury.software.com`, `venus.software.com`, `neptune.software.com`, etc.

***Note:*** *There is no default setting for this key. If no domain is specified, then none is sent with the cookie. The recommended setting is the top level domain.*

**Related Keys:** `cookieExpireTimeOutDays`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any valid domain name (with or without a host name pre-pended)

**Initial Value:** `null`

**Default Value:** `null`

**Example:**

```
/*/web/cookieDomain: [software.com]
```

## cookieExpireTimeoutDays

**Description:** Specifies the amount of time before a cookie (used for login bypass) expires. The browser will expire a cookie if it is not refreshed within the interval specified by the value of this key.

The value is specified in days. A value of zero is *not* recommended.

***Note:*** *A cookie is refreshed whenever a subscriber uses it to log in, or whenever the subscriber submits the preferences form with login bypass enabled.*

**Related Keys:** `CookieDomain`

**Servers Affected:** Web Server

**Change Impact:** Server restart required

**Possible Values:** any integer greater than zero

**Initial Value:** `10`

**Default Value:** `10`

**Example:** `/*/web/cookieExpireTimeoutDays: [10]`

## createMboxes (modified)

Mailboxes can now be created automatically in response to delivery and retrieval attempts initiated by the Web Server. To take advantage of this option the `createMboxes` configuration key should be set as follows:

```
/*/web/createMboxes: [true]
```

## dbCacheSizeInKb (modified)

The value of the `dbCacheSizeInKb` configuration key formerly represented the in-memory size of an individual cache file. A change in design now allows the content of a single directory cache to be divided among multiple files. The value of the `dbCacheSizeInKb` configuration key has been modified accordingly. It now represents the total size of all cache files on a given host (as opposed to the size of each individual file). This improvement ensures that the memory footprint stays constant when the number of cache files is changed.

---

***Note:*** *The number of cache files per Directory Cache Server is determined by the value of the `numUserDbFiles` configuration key. See separate entry for details.*

---

## dbReconnectInterval

| | |
|---|---|
| **Description:** | Period in which a server attempts to restore a lost connection to the database. A value of `0` indicates lost connections are not restored until server restart. |
| **Related Keys:** | `none` |
| **Servers Affected:** | Directory Cache Server |
| **Change Impact:** | trivial |
| **Possible Values:** | any integer 0 and greater |
| **Initial Value:** | `0` |
| **Default Value:** | `10 seconds` |
| **Example:** | `/*/imdircacheserv/dbReconnectInterval: [5]` |

## dirCacheUpdateThreshold

| | |
|---|---|
| **Description:** | Maximum amount of time in seconds a cache update can take before a `CacheUpdateTimeTooLong` notification is logged.<br><br>***Note:*** *If this configuration key is `0`, no `CacheUpdateTimeTooLong` message is logged.* |
| **Related Keys:** | `None.` |
| **Servers Affected:** | Directory Cache Server |
| **Change Impact:** | trivial |
| **Possible Values:** | any integer 0 and greater |
| **Initial Value:** | `0` |
| **Default Value:** | `0` |
| **Example:** | `/*/imdircacheserv/dirCacheUpdateThreshold: [5]` |

## Error-Actions
## Error-Text

The `Error-Actions` configuration keys define the system's response to a given event. Specifically, they indicate whether or not the event should be logged, and whether the message in question should returned or held for review.

The `Error-Text` configuration keys specify the text of the bounce message sent when the value of the corresponding `Error-Actions` configuration key indicates that a message should be returned.

---

**Warning!** Default `Error-Text` entries are *only* provided for `Error-Actions` keys whose default value is set to `return`. If you set the value of any other `Error-Actions` keys to `return`, you must also enter the appropriate text value in the corresponding `Error-Text` key.

---

## filterSendMessageCharset

**Description:** Defines the character set used to by the sieve when sending new messages.

**Related Keys:** none

**Servers Affected:** MTA

**Change Impact:** trivial

**Possible Values:** us-ascii, any valid MIME character set

**Initial Value:** us-ascii

**Default Value:** us-ascii

**Example:**
```
/*/mta/filterSendMessageCharset: [us-ascii]
```

## hostnameAliasList

**Description:** Describes other domain names or IP addresses for the MTA that appear in the MX records. Below is an example of what the MX records may look like if the domain name is “foo.com”:

```
foo.com.      MX      5 foo.com
foo.com.      MX      10 <MTA of ISP>
```

If `foo.com` is unavailable, the mail is delivered and stored with the ISP. However, an ISP may have different names for the MTAs (internal and external). For example, the external name is “mta.isp.com,” and the internal name is “logos.isp.com.” When the ISP tries to send mail to foo.com, and foo.com is not available, it will try to send to “mta.isp.com,” the secondary location. Since the MTA does not recognize that name (mta.isp.com) as itself, a mail loop is created.

The `/*/mta/hostnameAliasList` key is used in these situations, defining a name that the MTA should look for in the MX records if the actual domain name or IP does not exist.

All records at this MX preference or higher will be removed when sending mail to the foo.com domain.

**Related Keys:**

**Servers Affected:**

**Change Impact:**

**Possible Values:** Hostnames, IP addresses, or a mixture of the two.

**Initial Value:**

**Default Value:**

**Example:**
```
/*/mta/hostnameAliasList: [mta.isp.com]
                         [10.2.6.98]
```

## inDeliveryDeferKb

In the manual, it incorrectly states that if the value for this key is set to `0`, there are no limits for mail delivery. However, when inDeliveryDeferKb is set to `0`, the key is completely disabled.

## journalRollHours
## journalRollMbytes

The `journalRollHours` configuration key controls the number of hours of journal information that may be recorded in a single journal file. The `journalRollMbytes` configuration key controls the amount of information that may be recorded in a single journal file. If either of these limits is reached, the current journal file is closed and a new journal file opened.

The limits defined by these keys apply to *both* the message journal files created by the Message Store Server *and* the queue journal files created by the Queue Server.

## loginAliases

| | |
|---|---|
| **Description:** | If set to true, this allows the user to login (authenticate) using an SMTP alias as a login name. |
| **Related Keys:** | `none` |
| **Servers Affected:** | Directory Cache Server |
| **Change Impact:** | trivial |
| **Possible Values:** | true, false |
| **Initial Value:** | `false` |
| **Default Value:** | `false` |
| **Example:** | `/*/common/loginAliases: [false]` |

## loginDefaultDomain (modified)

When a login connection is attempted without an explicit domain (e.g., `login user` versus `login user@domain`), *and* the IP address for the connection is not listed among the values of the `loginDefaultDomainTable` configuration key, then the value of the this key is automatically appended to the username as the default domain.

---

***Note:*** *See the entries for the `loginNameConvertFrom`, `loginNameConvertTo`, and `loginDefaultDomainTable` configuration keys for additional information.*

---

## loginDefaultDomainTable

| | |
|---|---|
| **Description:** | Provides a list of default domains based on the IP address to which the POP connection is made.<br>Multiple entries are allowed; however, each entry must appear on its own line, between its own set of square brackets and each entry must be in the form:<br>`<IPaddress>:<domain>` |

where:

| | |
|---|---|
| `<IPaddress>` | the IP address from which the POP connection is made |
| `<domain>` | the default domain associated with users connecting from that IP address |

The `loginDefaultDomainTable` key provides flexibility. If a POP server machine has multiple IP interfaces or if one of the interfaces has multiple IP aliases, then clients can connect to different IP addresses and get different default domains.

For example, if the entries for the key were as follows:

```
[128.123.46.22:software.com]
[128.123.46.23:hardware.com]
```

A user logging in as "joe" to IP address `128.123.46.22` would be recognized as joe@software.com, while a user logging in as "joe" to IP address `128.123.46.23` would be recognized as joe@hardware.com.

***Note:*** *This table is only consulted if the login name contains no @ sign. If a match is found among the entries in the table the corresponding default domain is added to the end of the login name with an @ before it. If none of the IP addresses match, then the value of the* `loginDefaultDomain` *key is used. If that value is not specified, no default domain is added.*

| | |
|---|---|
| **Related Keys:** | `LoginNameConvertFrom, loginNameConvertTo` |
| **Servers Affected:** | POP server, IMAP server |
| **Change Impact:** | trivial, no server restart required |
| **Possible Values:** | one or more entries, each consisting of an IP address, a colon, and a valid domain name |
| **Initial Value:** | null |
| **Default Value:** | null |
| **Example:** | `/*/common/loginDefaultDomainTable: [128.123.46.22]` |

## loginFilter

**Description:** A list of `regex` substitution filters that are applied, in sequence, to every name supplied in a POP authentication query before the name is looked up in the Directory database.

The value of the loginFilter configuration key may include multiple entries, but each entry must appear on a separate line contained within its own set of square brackets.

Entries should be in the form:

```
s/pattern/replace/opts
s/pattern2/replace2/opts2 ...
```

where the "/" can be any single character not used in "pattern" or "replace", and "opts" can be empty or a string containing "g" and/or "i" (similar to Perl).

Values are separated by a white space and/or punctuation.

Examples:

```
s%xxx%yyy%i, s^Apple^Oracle^gi, s/^(.*)$/Z-\1/i
```

The pattern argument is a POSIX regular expression. The replace string can be simple text, or it can contain expressions of the form "\N", where N is a digit. If the pattern contains at least N parenthesized groups, then the text that matched the Nth group in the input will be substituted for the corresponding "\N" in the output.

***Note:*** *If the IMDIAG environment variable contains "popFilter=3" before starting the* `imdircacheserv` *process, trace output will be generated that may be useful in researching issues surrounding use of this configuration key,*

**Related Keys:** none

**Servers Affected:** Directory Cache Server

**Change Impact:** server restart required

**Possible Values:** a string with multiple substitution expressions

**Initial Value:** `null`

**Default Value:** `null`

**Example:** `/*/imdircacheserv/loginFilter: [s%xxx%yyy%i]`

## logRecorderSize

**Description:** Adjusts (only upward) the size of the recorder buffer used for the log file. A value of '0' disables the recorder buffer (but still writes to file).

**Related Keys:** none.

**Servers Affected:** all servers

**Change Impact:** trivial, no server restart required

**Possible Values:** Any non-negative integer

**Initial Value:** `100000`

**Default Value:** `100000`

**Example:** `/*/common/logRecorderSize: [100000]`

## masterAgentHost

When installing InterMail, the option to turn on SNMP is provided. If "`yes`" is the option selected, the `/*/common/masterAgentHost` configuration key is set automatically. However, if SNMP is turned off during installation (selecting "`no`" as the option), and it is later turned on, the `/*/common/masterAgentHost` configuration key will need to be set manually in the configuration database. SNMP functionality will not work if this key is not properly configured.

See the *InterMail Reference Guide* for more information on this configuration key.

## maxAutoReplyMsgLenKb

**Description:** Server-wide limit on the size of any mailbox "property" that can be stored in the `IM_AutoReplyMessage` table. Size is expressed in kilobytes.

The `maxAutoReplyMsgLenKb` configuration key limits the size of the vacation message used for vacation-mode and auto-reply. In addition, this key controls the size of the address book and signature text managed by WebMail for each account.

**Related Keys:** None.

**Servers Affected:** MTA

**Change Impact:** trivial.

**Possible Values:** 0 to 1024

**Initial Value:** `100`

**Default Value:** `100`

**Example:** `/*/mta/maxAutoReplyMsgLenKb: [377]`

## maxBadPassword

Specifies a limit on the number of bad password attempts allowed before a client connection is dropped. Do *not* set the value to 0.

## maxBadPasswordAddrs

A value of 0 (zero) means the system will *not* maintain a list of the IP addresses of clients that issued incorrect login information.

## maxBadPasswordDelay

A value of 0 (zero) means there is no delay; however you should *not* disable this feature by setting its value to 0. Instead, set the values of the `maxBadPasswordAddrs` and `maxBadPasswordUsers` configuration keys to 0.

## maxBadPasswordUsers

A value of 0 (zero) means the system will *not* maintain a list accounts for which an incorrect password was given.

## maxCursorsPerConn

| | |
|---|---|
| **Description:** | Controls maximum size of a cursor pool. |
| **Related Keys:** | `primaryDBnumConnections` |
| **Servers Affected:** | Directory Cache Server |
| **Change Impact:** | trivial |
| **Possible Values:** | integers greater than 0 |
| **Initial Value:** | `None (not in initial config.db)` |
| **Default Value:** | `8` |
| **Example:** | `/*/imdircacheserv/maxCursorsPerConn: [8]` |

## maxDirectDelivery

This key sets the maximum number of recipients for messages that may be delivered directly from memory. The original documentation states that any message with more than the specified number of recipients gets written to the queue directory, however, this only occurs when the `localFallback` configuration key is set to `true`. If the `localFallback` configuration key is set to `false`, messages that exceed the maximum number of recipients are rejected.

## maxMSSRetrieveConnectionList

| | |
|---|---|
| **Description:** | Indicates limits on RME connections used for retrieving mail from the specified MSS host. |
| **Related Keys:** | `maxMSSRetrieveConnectionTimeout` |
| **Servers Affected:** | POP and IMAP Servers |
| **Change Impact:** | server restart required |
| **Possible Values:** | |
| **Initial Value:** | `empty list` |
| **Default Value:** | `empty list` |
| **Example:** | `/*/common/maxMSSRetrieveConnectionList: [mss1:100] [mss2:200]` |

## maxMSSRetrieveConnectionTimeout

| | |
|---|---|
| **Description:** | In order to retrieve mail from the MSS, an RME connection has to be established. This key indicates the amount of time (in seconds) an RME connection waits if the value in the `maxMSSRetrieveConnectionList` configuration key is exceeded. |
| **Related Keys:** | `maxMSSRetrieveConnectionList` |
| **Servers Affected:** | POP and IMAP Servers |
| **Change Impact:** | trivial |
| **Possible Values:** | any integer greater than zero |
| **Initial Value:** | `5 seconds` |
| **Default Value:** | `5 seconds` |
| **Example:** | `/*/common/maxMSSRetrieveConnectionTimeout: [5]` |

## maxQueueTimeInHours

This key replaces the `maxQueueTimeInDays` configuration key. Instead of being configured in days, this key is now set using hours as its possible value. The initial defaults remain the same, but the entry for days has been mapped to hours. The example below shows the conversion:

minimum allowed value = 12 hours

default = 96 hours

default no entry = 96 hours

---

***Note:*** *It is very important that you update your configuration database when upgrading to 4.1. If this is not done, these changes to* `MaxQueueTimeinHours` *will not take effect.*

---

## maxPasswordFailures

Specifies a limit on the number of authentication failures on a given account or from a particular IP Address before connections are dropped. The recommended minimum setting for this key is 1. Do *not* set the value to 0.

## maxSessionsPerUser

**Description:** Limits the number of concurrent sessions for a single user.

**Related Keys:** none

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any integer greater than zero

**Initial Value:** 3

**Default Value:** 3

**Example:**
```
/*/web/maxSessionsPerUser: [3]
```

## maxThreads

**Description:** Limits the number of threads created per process.

Recommended values range from 100 to 10000.

***Note:*** *This key is only significant for installations which run on the IRIX operation system.*

**Related Keys:** none

**Servers Affected:** all servers

**Change Impact:** server restart required

**Possible Values:** all non-negative integers (including zero)

**Initial Value:** 2048

**Default Value:** 2048

**Example:**
```
/*/common/maxThreads: [2048]
```

## messageFilesDir

There is an error in the documentation concerning the messageFilesDir configuration key. The correct information is provided below:

**Servers Affected:** MSS

**Change Impact:** server restart required

## messageJournaling (MSS)

| | |
|---|---|
| **Description:** | Determines whether the MSS and garbage collection utility will generate journal files while processing. MSS journals are fundamental to the recovery of deleted messages, and the garbage collector journals are important when pruning the backup Message File System. |
| **Related Keys:** | `none` |
| **Servers Affected:** | MSS |
| **Change Impact:** | server restart required |
| **Possible Values:** | true/false |
| **Initial Value:** | `true` |
| **Default Value:** | `true` |
| **Example:** | `/*/mss/messageJournaling: [true]` |

## messageJournaling (Queue Server)

**Description:** Determines whether the Queue Server will generate journal files while processing. The Queue Server journals are fundamental to the recovery of deferred and undelivered messages.

**Related Keys:** `none`

**Servers Affected:** Queue Server

**Change Impact:** server restart required

**Possible Values:** true/false

**Initial Value:** `true`

**Default Value:** `true`

**Example:** `/*/imqueueserv/messageJournaling: [true]`

## mssConnPoolSize

**Description:** Specifies the number of connections a single web server process should cache between requests.

A setting of 0 means there is no connection pool. No connections are cached, and any open connections are closed as soon as they are no longer in use.

**Related Keys:** `none`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any non-negative integer (including 0)

**Initial Value:** `10`

**Default Value:** `0`

**Example:** `/*/web/mssConnPoolSize: [0]`

## mtaHosts

**Description:** Defines a list of MTA hosts that will deliver mail sent from the Compose form (the WebMail interface page that allows users to compose and send messages).

Each entry must be in the following form :

```
<hostname>:<port#>
```

where:

| | |
|---|---|
| `<hostname>` | The name of the host on which the MTA runs. The string `localhost` is a special value indicating the current host (that is, the host on which the Web Server generating the request is installed). |
| `<port#>` | The port number on which to attempt connection. The port number entry is optional. If no port number is specified, port 25 is used as the default port. |

Multiple entries are supported, as long as each appears on its own line, between its own set of square brackets. A random entry is picked from the list for each delivery operation.

**Related Keys:** `none`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** one or more entries, each consisting of a valid hostname, a colon, and a port number

**Initial Value:** `localhost:25`

**Default Value:** `localhost:25`

**Example:**

```
/*/web/mtaHosts: [venus:25]
                [neptune:25]
```

## multiJournaling

**Description:** Switches between the old, single-file journaling functionality and the newer, multiple-file journaling functionality.

Because the MSS formerly wrote to a single journal file, more CPU resources were needed to get desired throughput. By adding multiple journal files, the MSS has improved its capacity

Affected servers are fully backward compatible when this key is unset or set to `false`.

**Related Keys:** `numJournalsPerServer`

**Servers Affected:** MSS

**Change Impact:** server restart required

**Possible Values:** `true` or `false`

**Initial Value:** `false`

**Default Value:** `false`

**Example:**

```
/*/mss/multiJournaling: [false]
```

## mutexSerialNumbering

**Description:** Controls the numbering of mutexes created by the InterMail servers.

**Warning!** This key should not be modified except at the express direction of Software.com technical support personnel.

**Related Keys:** none

**Servers Affected:** all servers

**Change Impact:** server restart required

**Possible Values:** `true` or `false`

**Initial Value:** `false`

**Default Value:** `false`

**Example:**
```
/*/common/mutexSerialNumbering: [false]
```

## numDBFiles

**Description:** This key controls the number of `session.db` files that get created. Once this number is changed, none of the data in the existing `session.db` files is valid. It is recommended that the `session.db` files get recreated if this value is modified.

This number should be set high if a large amount of records in the `session.db` files are expected.

It is suggested that a prime number is used: (e.g. 1,2,3,5,7,11,13,19,23,29...).

**Related Keys:** `none`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any integer greater than zero (prime number is recommended)

**Initial Value:** `11`

**Default Value:** `11`

**Example:**
```
/*/web/numDBFiles: [13]
```

## numJournalsPerServer

**Description:** Allots the given number of journal files to the server for its exclusive use. Threads within the server process gain access to the journal files on a first in, first out basis. If this key is omitted, the server uses a single journal file.

If the `multiJournaling` configuration key is set to `false`, this key is ignored.

**Related Keys:** `multiJournaling`

**Servers Affected:** MSS

**Change Impact:** server restart required

**Possible Values:** any integer greater than zero

**Initial Value:** `1`

**Default Value:** `1`

**Example:**
```
/*/mss/numJournalsPerServer: [1]
```

## numUserDbFiles

**Description:** Controls the number of directory cache files is use by a single Directory Cache Server. All required directory data is divided into these files. The more accounts in the Integrated Services Directory, the larger the number of files should be.

***Note:*** *As a rule of thumb, take the total number of accounts you expect in the Directory database and divide by 100,000. Round the result to the nearest even number, and use that number as the value for this key.*

**Related Keys:** `DbCacheSizeInKb`

**Servers Affected:** Directory Cache Server; `imdirsync` and `imcachecontrol` utilities

**Change Impact:** server restart required

**Possible Values:** any integer greater than zero

**Initial Value:** `1`

**Default Value:** `1`

**Example:**
```
/*/imdircacheserv/numUserDbFiles: [2]
```

---

***Note:*** *This key CANNOT be specified for individual hosts. For example, the number of database files is common to all Directory Cache Server hosts in the system. This is documented for completeness (in practice, it is highly unlikely that the number of database files changes from one Directory Cache Server host to another).*

---

## partitionRouterData

**Description:** Controls whether or not router data in the cache database should be partitioned. This is finer control over the cache database, and should not

be used unless instructed by Software.Com.

**Related Keys:** `partitionUserData`
`realms`

**Servers Affected:** Directory Cache Server

**Change Impact:** All router data will be present in each cache if this key is set to `false`. By default, when realms support is enabled (see the realms configuration key), router data is always partitioned.

***Note:*** *The `impatch` utility will not install this key*

**Possible Values:** true, false

**Initial Value:** `true`

**Default Value:** `true`

**Example:** `/*/imdircacheserv/partitionRouterData: [false]`

## partitionUserData

**Description:** Controls whether or not user data in the cache database should be partitioned. This is finer control over the cache database, and should not be used unless instructed by Software.Com.

**Related Keys:** `partitionRouterData`
`imdircacheserv/realms`

**Servers Affected:** Directory Cache Server

**Change Impact:** All user data is present in each cache database if this key is set to `false`. By default, when realms support is enabled (see the realms configuration key), router data is always partitioned.

***Note:*** *The `impatch` utility will not install this key.*

**Possible Values:** true, false

**Initial Value:** `true`

**Default Value:** `true`

**Example:** `/*/imdircacheserv/partitionUserData: [false]`

## queueServerRetrieveHosts

**Description:** These are used in cases where you want to separate out functionality of the various Queue Servers. This is a list of the machines that run Queue Server processes that you want the MTA to pull deferred email from and try and reprocess it.

**Related Keys:** `queueServerStoreHosts`

**Servers Affected:** MTA

**Change Impact:** trivial

**Possible Values:** (empty list) = none

[all] = all Queue Servers

[machineA] [machineB] = just machine A and machine B

**Initial Value:** `all`

**Default Value:** `all`

**Example:** `/*/mta/queueServerRetrieveHosts: [all]`

## queueServerStoreHosts

**Description:** These are used in cases where you want to separate out functionality of the various Queue Servers. This is a list of the machines that run Queue Server processes that you want the MTA to store deferred email on.

**Related Keys:** `queueServerRetrieveHosts`

**Servers Affected:** MTA

**Change Impact:** trivial

**Possible Values:** (empty list) = none

[all] = all Queue Servers

[machineA] [machineB] = just machine A and machine B

**Initial Value:** `all`

**Default Value:** `all`

**Example:** `/*/mta/queueServerStoreHosts: [all]`

## realms

**Description:** Controls the use of partitioning (via realms) of the Directory Cache Server. Partitioning data into realms reduces the size of the cache database that each Directory Cache Server maintains. It uses smaller storage and provides better performance for both lookups and updates.

Realms are configured by supplying a multi-line value. Each line contains a realm number followed by a colon (:) and a comma-separated list of hosts on which Directory Cache Server is running. Realm numbers are 0-based, and there can be as many as 32 realms. A host can appear on more than one list with the understanding that the Directory Cache Server on that host supports more than one realm. (Each realm is associated with a subset of the total data that is present in the Directory).

The first host in the comma-separated host list is known as the “primary host”. This host is responsible for all lookups for that realm. In the event of a failure, the secondary host assumes that responsibility and so on.

Realms can be configured as the following simple example illustrates:

```
/*/imdircacheserv/realms: [0: pluto, venus]
                          [1: mercury, venus]
                          [2: uranus, venus]
```

In the above example, there are 3 realms. “`pluto`” is the primary host for

realm `0`, "`mercury`" for realm `1` and "`uranus`" for realm `2`. "`venus`" is a backup server holding all the realm data (a full replica). The host list behaves much like the `dirCacheHosts` configuration key and is much like the `dirCacheHosts` list in behavior. The host, "`venus`," is not queried unless the primary host in that realm fails.

There can be more than 2 hosts listed for each of the realms, and realm configurations are not required to have one host ("`venus`" in the above example) that acts as a backup with a full replica.

**Related Keys:** `partitionUserData`
`partitionRouterData`

**Servers Affected:** Directory Cache Server

**Change Impact:** If you already have a full copy of the data on each Directory Cache Server host, you should get rid of database files that no longer belong in that realm. Run `imdirsync -reportRealms` to show you what database files belong in what realm.

***Note:*** *The `impatch` utility will not install this key*

**Possible Values:** See the Description for more information.

**Initial Value:** `None.`

**Default Value:** `None. Realms are disabled.`

**Example:**
```
/*/imdircacheserv/realms: [0: pluto, venus]
                           [1: mercury, venus]
```

## rejectInvalidFromAddr

**Description:** Specifies whether or not senders are required to provide a RFC 821 compliant `MAIL FROM:` address.

If the configuration key is set to true, senders must provide a RFC 821 compliant address with the `MAIL FROM:` command in order for it to be accepted. Note that in the case of rejection, the reply code and reply text returned to the sender are defined by the values of `blockReplyCode` and `blockReplyText`, respectively.

If `rejectInvalidFromAddr` is set to false, senders do not have to provide an RFC-821 compliant `MAIL FROM` address.

**Related Keys:** `blockReplyCode, blockReplyText`

**Servers Affected:** MTA

**Change Impact:** trivial, no server restart required

**Possible Values:** `true` or `false`

**Initial Value:** `false`

**Default Value:** `false`

**Example:**
```
/*/mta/rejectInvalidFromAddr: [false]
```

## rejectSenderBadAddress

**Description:** If set to true, the sender address must parse correctly to be accepted (refer to RFC821 for the definition of invalid). If the message is not validated, it is rejected with a `550` response code and text "Sender address is invalid."

**Related Keys:** Below is complete set of anti-spam keys that define rules for rejecting senders:

```
/*/mta/rejectSenderBadAddress: [false]
/*/mta/rejectSenderBadDomain: [false]
/*/mta/rejectSenderIPDomain: [false]
/*/mta/rejectSenderNoDomain: [false]
```

**Servers Affected:** MTA

**Change Impact:** trivial

**Possible Values:** `false/true`

**Initial Value:** `false`

**Default Value:** `false`

**Example:** `/*/mta/rejectSenderBadAddress: [true]`

## relayMaxRCPTs

Chapter 11 of the InterMail Reference Guide, describes the `relayMaxRCPTs` configuration key incorrectly; see the following description for the correct information.

**Description:** In Chapter 11 of the InterMail Reference Guide, the `relayMaxRCPTs` configuration key is described as a key that "defines the minimum number of recipients...before relay restrictions are applied to it." However, the key actually defines the *maximum* number of recipients allowed, not the minimum, before checking the relay restrictions.

For example, if the value of the `relayMaxRCPTS` configuration key were set to 3, then all messages with less than three recipients would be exempt from established relay restrictions, however all messages with three or more recipients would be subject to those restrictions.

**Related Keys:** `relayLocalMustExist, relayDestAllowList, relayDestDenyList, relaySourcePolicy, relaySourceDomainList, relayLocalDomainsOK, relaySourceRemoteIPList, relaySourceLocalIPList, and relayNullRestricted`

**Servers Affected:** MTA

**Change Impact:** trivial, no server restart required

**Possible Values:** any integer greater than zero

**Initial Value:** `0`

**Default Value:** `0`

**Example:** `/*/mta/relayMaxRCPTs: [0]`

Zero is the default value, and is recommended because it causes *all* messages to go through relay checks.

## relayNullRestricted

The original documentation states that the value of the `relayNullRestricted` configuration key applies only if the `relaySourcePolicy` configuration key is set to `denyListed`; this is not correct. The value of the `relayNullRestricted` configuration key applies if the `relaySourcePolicy` configuration key is set to *either* `denyListed` or `allowListed`.

## requireSecureAuth

The description for the `requireSecureAuth` configuration key in Chapter 11 of the *InterMail Reference Guide* is erroneous; see the following correction:

**Description:** If this key is set to true, the MTA does not allow the `AUTH LOGIN` command unless the user is connecting via SSL.

If this key is set to false, the MTA allows `AUTH LOGIN` whether or not SSL is enabled.

This variable does not cause authentication to be required; however, if the

 

user attempts to authenticate, it is required that they have a secure connection.

The setting of this variable only applies if `mta/checkAuthentication` is set to true; if it's set to false, `AUTH LOGIN` is never allowed.

**Related Keys:** `checkAuthentication`

**Servers Affected:** MTA

**Change Impact:** trivial, no server restart required

**Possible Values:** true or false

**Initial Value:** `false`

**Default Value:** `false`

**Example:** `/*/mta/requireSecureAuth: [true]`

## retryOnConnectFail

**Description:** , Sets the number of times InterMail attempts to reconnect to a remote host when a connection returns `ECONNREFUSED`.

This is only for host reconnections within the InterMail system (i.e. MTA talking to an MSS)

**Related Keys:** none

**Servers Affected:** all servers

**Change Impact:** trivial.

**Possible Values:** any non negative integer

**Initial Value:** `0`

**Default Value:** `0`

**Example:** `/*/common/retryOnConnectFail: [2]`

## rmeAccessList

**Description:** Indicates IP addresses or hostnames that are allowed to connect to the RME port of any server. This serves as a protection for the RME port.

**Related Keys:** `none`

**Servers Affected:** all servers

**Change Impact:** trivial

**Possible Values:** valid IP address or hostname

**Initial Value:** `no restrictions, access allowed by all`

**Default Value:** `no restrictions, access allowed by all`

**Example:**
```
/*/common/rmeAccessList: [venus] [10.2.6.82]
```

## scanOldMessagesIntervalInHours

**Description:** Scanning for volumes of old mail can be resource intensive since each piece needs to be inspected. This key allows the user to specify how often to check the mail that has been in the system for more than the interval specified by the `maxQueueTimeInDays` configuration key.

If it were set to 24 hours, then we would check each outbound domain once a day for mail that was older than `maxQueueTimeInDays.`

**Related Keys:** `maxQueueTimeInDays`

**Servers Affected:** MTA Queue Server

**Change Impact:** trivial

**Possible Values:** integer number of hours

**Initial Value:** `24`

**Default Value:** `24`

**Example:**
```
/*/mta/scanOldMessagesIntervalInHours: [24]
```

## sessionDBFilePath

**Description:** Specifies the path to and name of the session database file used by all web-based applications (SelfCare and WebMail).

Paths may be absolute or relative. Relative paths are interpreted in relation to the `$INTERMAIL` directory.

***Note:*** *SelfCare and WebMail transactions are session based. When a user logs in, the user's token and selected account information from the Integrated Services Directory is cached in the session database*

**Related Keys:** `sessionFlushIntervalSecs, sessionExpireTimeoutSecs, maxSessionsPerUser`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** a valid path and file name

**Initial Value:** `httpd/WebSessMgr.db`

**Default Value:** `httpd/WebSessMgr.db`

**Example:** `/*/web/sessionDBFilePath: [httpd/WebSessMgr.db]`

## sessionExpireTimeoutSecs

**Description:** Specifies the amount of inactivity after which a session is marked as eligible for expiration.

Values are indicated in seconds.

For example, if the value of the `sessionExpireTimeoutSecs` configuration key were set to 1800, any session without activity for 1800 seconds (30 minutes) would be marked for expiration and flushed from the database at the next expiration interval.

***Note:*** *Inactive sessions are flushed from the database at intervals defined by the `sessionFlushIntervalSecs` configuration key.*

**Related Keys:** `sessionFlushIntervalSecs`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any non-negative integer (including zero)

**Initial Value:** `1800`

**Default Value:** `1800`

**Example:** `/*/web/sessionExpireTimeoutSecs: [1800]`

## sessionFlushIntervalSecs

**Description:** Specifies the interval at which the Web Server goes through the session database, looks for inactive sessions, and expires them.

Values are measured in seconds.

**Related Keys:** `sessionExpireTimeoutSecs`

**Servers Affected:** Web Server

**Change Impact:** server restart required

**Possible Values:** any integer greater than zero

**Initial Value:** `360`

**Default Value:** `360`

**Example:** `/*/web/sessionFlushIntervalSecs: [360]`

## sslSmtpPort

The following is an addendum to the existing description of `SslSmtpPort` in the *InterMail Reference Guide*.

**Description:** Port number (usually `465`) that the MTA listens on for SSL (Secure Socket Layer) connection requests from other MTAs.

When null, or `-1`, it disables the SSL operation.

If the port specified is `25`, then the secure SMTP operates in the mode defined by the SMTP Service Extension for Secure SMTP over TLS (Transport Layer Security). TLS is essentially SSL, but with a negotiation on the standard SMTP port (25) to indicate at the start of the transaction that you want to talk securely. By setting `sslSmtpPort` to 25, any client that uses the standard SMTP port and is SSL-compliant will negotiate a secure connection.

**Related Keys:** none

**Servers Affected:** MTA

**Change Impact:** server restart required

**Possible Values:** any valid, unused port number

**Initial Value:** `465`

**Default Value:** `-1`

**Example:** `/*/mta/sslSmtpPort: [465]`

## snmpReconnectNapTime

**Description:** Determines the length of time a server will wait after detecting that the SNMP master agent is down, before attempting to reconnect to the master agent.

Values are measured in seconds.

***Note:*** *Each server retries only once per SNMP event; however, further connection attempts will be made when additional SNMP subagent events are generated.*

**Related Keys:** none

**Servers Affected:** all servers

**Change Impact:** trivial

**Possible Values:** any integer from `5` to `300`

**Initial Value:** `15`

**Default Value:** `15`

**Example:** `/*/common/snmpReconnectNapTime: [15]`

## statRecorderSize

**Description:** Adjusts the size of the recorder buffer used for the stat file. A value of `0` disables the recorder buffer (but still writes to file).

**Related Keys:**

**Servers Affected:** Possibly all servers

**Change Impact:** Trivial

**Possible Values:** Any non-negative integer (including `0`).

**Initial Value:** `10240`

**Default Value:** `10240`

**Example:** `/*/common/statRecorderSize: [10240]`

## traceRecorderSize

**Description:** Adjusts the size of the recorder buffer used for the trace file. A value of 0 will disable the recorder buffer (but still writes to file).

**Related Keys:**

**Servers Affected:** All servers.

**Change Impact:** Trivial

**Possible Values:** Any non-negative integer (including `0`)

**Initial Value:** 10240

**Default Value:** 10240

**Example:**
```
/*/common/traceRecorderSize: [10240]
```

## trustedInterfaces

**Description:** List of host interfaces specified by IP address which are considered to be "trusted." Used in conjunction with the `pref_popaccess`, `pref_popsslaccess`, and `pref_imapaccess` class of service attributes to limit locations from which users can access their mailboxes.

**Related Keys:** none

**Servers Affected:** POP Server, IMAP Server

**Change Impact:** Trivial

**Possible Values:** List of IP addresses

**Initial Value:** `empty, no addresses`

**Default Value:** `empty, no addresses`

**Example:**
```
/*/common/trustedInterfaces: [10.2.7.49]
                              [10.2.7.50]
                              [10.2.5.51]
```

## verfiyDeferOK

There is an error in the *InterMail Reference Guide* regarding the description for the `verifyDeferOK` configuration key. The following revised description is correct:

**Description:** This key controls what happens in cases when the Directory Cache Server is down and only has any affect when `relayLocalMustExist`, `blockLocalNoAcct`, `verifyRCPTs`, or `blockPerAccount` are set.

When `relayLocalMustExist` or `blockLocalNoAcct` are set, the MTA will attempt to look up `MAIL FROM` addresses in the Directory Cache Server. When `verifyRCPTs` is true, or when `blockPerAccount` is true and the sender is blocked, the MTA will attempt to look up `RCPT TO` addresses in the Directory Cache Server.

If the Directory Cache Server is down, and `verifyDeferOk` is true, the

 

message is NOT deferred locally. Instead, the MTA responds to the client with a 450 response, rejecting the message. The 450 response means that the client should retry layer. Therefore, the client will defer the message, not the server.

If the `verifyDeferOk` configuration key is false, the message is accepted.. If `relayLocalMustExist` is true, and the Directory Cache Server is down, the MTA stops enforcing `relayLocalMustExist` until the Directory Cache Server comes back up. This same scenario is true for `blockLocalNoAcct`, `verifyRCPTs`, and `blockPerAccount`.

**Related Keys:** `relayLocalMustExist`, `blockLocalNoAcct`, `verifyRCPTs`, and `blockPerAccount`

## vxfsSpaceReservation

**Description:** Enables an improved algorithm for the allocation of disk space for InterMail message files.

If this key is set to `true`, the system makes better use of available disk space and reduces fragmentation.

This option works on Veritas filesystems only; it should only be used if each and every file system mounted beneath the Messages directory (defined by the `messageFilesDir` configuration key) is a Veritas filesystem.

**Warning!** Using this option with non-Veritas filesystems will generate an Urgent `FioVxfsOptFailed` error in the `mss log` and unnecessary impact on performance for each message created by the MSS.

**Related Keys:** none

**Servers Affected:** MSS

**Change Impact:** trivial

**Possible Values:** `true` or `false`

**Initial Value:** `false`

**Default Value:** `false`

**Example:**
```
/*/mss/vxfsSpaceReservation: [true]
```

# 5

# *Additional Administration Commands*

This chapter supplements the Administration Commands chapter in the *InterMail Reference Guide*. It includes updated information for previously documented commands and complete descriptions for commands not documented in the original manual.

Commands are listed in alphabetical order for ease of reference.

## imaccountquery

The documentation for `imaccountquery` shows the parameters for the `-r` option in both lower case and mixed case. However, these parameters should only be used as lower case entries. See the correct usage below:

**Where:**

```
-r addr
-r count
```

## imboxcopy

Although the `imboxmove` command will be used far more frequently when relocating mailboxes, there are several scenarios in which use of the `imboxcopy` command may also be appropriate. Those scenarios are discussed in the sections that follow.

**Scenario 1: Using the `imboxcopy` command to copy an existing mailbox from one account to another on the same MSS host.**

In this scenario, a subscriber requests a new e-mail name. The service provider provisions a new account with the new name, then copies the old mailbox contents into the new account. To use `imboxcopy` for this purpose, follow these steps:

1. Provision the second (new) account.
2. Create an associated mailbox for this account (using `imboxcreate`, `imboxsync`, or some other mechanism).
3. Run `imdirsync` to synchronize the Directory Cache with the updated contents of the Integrated Services Directory (i.e., the newly created account).
4. Find out the Internal ID of the source (original) and target (second) accounts by running `imdbcontrol listaccounts` or `imaccountquery`.
5. Invoke `imboxcopy` as in the following example (where “555”: is the Internal ID of the source mailbox and “7777” is the target mailbox):

   ```
   venus% imboxcopy ms:/DB/555 ms:/DB/7777
   ms:/DB/555 -> ms:/DB/7777: Move succeeded
   ```

***Note:*** *There are other options that *can* be used with `imboxcopy` in this scenario. For example, using the -force option will create a mailbox if there is no existing target mailbox. See the InterMail Reference Guide for more information on possible `imboxcopy` options.*

**Scenario 2: Using the `imboxcopy` command to copy a mailbox from HostA to HostB**

This scenario assumes that `imboxmove` was previously executed to move a mailbox (or batch of mailboxes) associated with an account (or accounts) from HostA to HostB; however, this move was erroneous because it was discovered that some of the original accounts on HostA contained mail. In order to use imboxcopy for this scenario, follow these steps:

1. On HostB, find out the Internal ID of the mailbox that you wish to copy.
2. Invoke imboxcopy as in the following example (where "venus" is the source MSS host, "neptune" is the target MSS host, and "34567" is the Internal ID of the account):

```
venus% imboxcopy ms://venus/DB/34567 ms://neptune/DB/34567
```

Invoking `imboxcopy` in this manner will copy the mailbox on `venus` to the mailbox on `neptune`. Please note, that as in Example A, other `imboxcopy` arguments may be appropriate to the task you wish to accomplish.

## imboxdelete

The `imboxdelete` command deletes an InterMail message store. An inaccurate usage statement was documented in the *InterMail Reference Manual*. The following description is correct and replaces the previously documented `imboxdelete` usage.

***Note:*** *Deleting an InterMail message store is NOT the same as deleting an account. Please refer to the InterMail Mx Operations Guide for more information on how to delete an account.*

**Syntax**

```
imboxdelete [-v] [-a|-m <file>] <host> <InternalID>
```

**Where:**

| | |
|---|---|
| `-v` | Verbose operation. |
| `[-a <file>]` | An input file containing e-mail addresses. |
| `[-m <file>]` | An input file containing Message Store IDs. |
| `<host>` | Host upon which the message store resides. |
| `<InternalID>` | Message Store ID. |

***Note:*** *You can find the Message Store ID and the name of the MSS host for a given e-mail address with the `imboxget` command.*

**Example**

To delete a message store, run `imboxdelete` as in the following example:

 

```
paris% imboxdelete paris 654321

imboxdelete:  Message store 654321 deleted!
```

## imboxmove

In Chapter 12 of the *InterMail Reference Guide*, the usage statement for `imboxmove` is incomplete as the `–d` option has been omitted. The correct usage statement is as follows:

```
imboxmove [-i|-b] [-d <sourceHost>] <dstHost> <file> [<file>...]
imboxmove [-e] <dstHost> <srcHost> [<exclude>]
```

Where:

| | |
|---|---|
| `-i` | Runs in "interactive mode," prompting the user to hit `<Enter>` before `imboxmove` proceeds to the next step. |
| `-b` | Runs in "batch" mode where there is no prompting. |
| `-d <sourceHost>` | Deletes successfully moved mailboxes from the specified source host. |
| `<dstHost>` | Target MSS host to which all accounts will move. |
| `<file>` | File containing the list of user accounts to move to the target MSS host. |
| `-e` | Moves only empty mailboxes from one host to another. |
| `<srcHost>` | Source MSS host from which mailboxes will move. |
| `<exclude>` | List of mailboxes to exclude from the `imboxmove` process. |

***Note:*** *Any MTA servers that are running must stop before running `imboxmove` with the `-e` option; for this reason, you rarely use the `-e` option.*

When using the `imboxmove` command on a particular host, verify that the `dirCacheHosts` configuration key (on that host) contains a list of all servers that run a Directory Cache Server process. If all the servers are not listed in this configuration key, mail will continue to be delivered on the original MSS host.

## imboxstats (modified)

This command can be operated in verbose mode with a `-v` flag. A modification to the usage statement has also been made and the hostname parameter has been removed.

**Syntax:**

```
imboxstats [-v] {storeId | smtpAddress}
```

***Note:*** *If the e-mail address is specified, the hostname is not needed.*

## imboxsync (modified)

The `imboxsync` command's new behavior retains backward compatibility for input and output mechanisms; however, three new options are available as well.

If -c is specified, the user is prompted before `imboxsync` creates mailboxes, but not prompted before `imboxsync` deletes mailboxes. If -d is specified, the user is prompted before `imboxsync` deletes mailboxes, but not prompted before it creates mailboxes. With the -q option (which may be used along with any one of the other options), the `imboxsync` does not report each ID affected, but, instead reports the number of mailboxes to be created.

Mailboxes can be deleted safely under a running MSS, however, when using `imboxsync` with newly created mailboxes, precautions must be taken. The best approach is to use the -y flag. This option allows the `immsscall` input to be optionally saved for later execution. When the prompt appears, asking to add/delete the mailboxes, answer "no." This will cause `imboxsync` to generate lists of the mailboxes which should be created or deleted. When the site is at an off-peak hour, run `immsscall` to create or delete these mailboxes. Note that any mailbox which is currently loaded by the MSS will not get deleted. The `mss.log` will report errors if the deletion of any mailbox has failed.

## imbroadcast

The `imbroadcast` command is used to send email to mailboxes on an MSS host, without using the MTA. Instead of using the SMTP port, `imbroadcast` uses RME and communicates directly with the MSS.

**Syntax:**

```
imbroadcast -f <filename> [-m <mssHost>] [-p <mssPort>] [-a
<adminID>] [-t <seconds>] [-h]
```

**Where:**

| | |
|---|---|
| `-f <filename>` | Specifies the pathname to the message file. If the filename is "-", text is read from `stdin`. This flag is required. |
| `-m <mssHost>` | Specifies the logical hostname of an MSS; defaults to the current host. |
| `-p <mssPort>` | Specifies the MSS port for the broadcast operation. The default is randomly selected from valid MSS ports if there is more than one. |
| `-a <adminID>` | Specifies the `msName` of the administration mailbox. The default is `admin`, which is the default used by `immsinit` to create the administration mailbox. |
| `-t <seconds>` | Specifies the timeout in seconds for the broadcast operation. The default is 8000 seconds (2.2 hours). |
| `-h` | Prints the usage statement and exits. |

It is possible for the broadcast operation to take a very long time, but it is not necessary for the `imbroadcast` command to actually wait for the results. If the timeout is reached, `imbroadcast` will print a warning and exit gracefully, but the broadcast operation will continue on the MSS.

 

## imcachecontrol

The imcachecontrol command allows the user to view account information that is currently stored in the Directory Cache Server, such as aliases and POP and SMTP addresses.

**Syntax:**

```
imcachecontrol -h
```

**Where:**

| | |
|---|---|
| `-list <directory>` | List all information about all user accounts contained in the cache. In general, this will be all of the InterMail accounts. |
| `-lm <directory>` | Report the "serial number" of the latest directory transaction which was updated to the cache on this machine. |
| `-h` | Provides usage statement. |

By default, `imcachecontrol` accesses the InterMail cache file, specified by the `/host/imdircacheserv/dbFilePath` configuration variable, where `host` is the logical hostname of the local host on which you are running.

The Directory Cache Server will periodically compare the latest "serial number" for the directory with the latest "serial number" for the cache. If these numbers are the same, then the cache is up-to-date. If the number in the directory is higher than the number in the cache, then the cache must be updated with the intermediate directory transactions. The `imdirsync` utility does this operation on demand.

## imcachecopy

The `imcachecopy` command is used to copy directory cache files in previous versions of InterMail; however, it is a reserved command and should only be used in conjunction with Software.com technical support.

## imcacheread (modified)

A new argument (`-c`) has been added in order to display class of service information for a given input address. If the `imcacheread -f` command does not work properly; use the `imdbcontrol` utility's `ListAccountForwards` option instead.

## imctrl

The correct usage statement for `imctrl` is as follows:

```
imctrl [-lists] [-verbose] [-dryrun] {<host>[:<host>]|localhost|
allhosts} {start|drain|stop|kill|restart|drainStart|stopStart|
exitStart|killStart} {<server>[:<server>:<server>]|mailservers|
allservers}
```

The –`lists` argument displays a sorted output of servers that are being affected.

See the *InterMail Reference Guide* for a complete explanation of the options.

## imdbcontrol

This warning was listed in the Release Notes, but bears repeating here.

**Warning!** The `DeleteCosAttribute` argument to the `imdbcontrol` utility is *extremely* dangerous. *Do not use this argument!*

**New Additions to the imdbcontrol utility**

Previously, if there was no data in the table when `imdbcontrol gfm` was called, an Oracle error would result. The `imdbcontrol` utility now returns the following:

```
Earliest Log Entry: 0 (Log table empty)
```

**imdbcontrol CreateAccount**

The usage statement for `imdbcontrol CreateAccount` is documented incorrectly in the *Integrated Services Directory Guide.* For the correct usage statement, see the following:

**Usage:**

```
imdbcontrol ca <pSmtpAddress> <delivery-host> <internal-id>
[<PopAddress> [<password> [-convert] (clear | md5-po | unix)
[<Domain> [(A | L | D | M | S | P) [(S | A) [<CosName>]]]]]]
```

**Where:**

| | |
|---|---|
| `pSmtpAddress` | The local part of the SMTP address for the account (the part which precedes the "@" symbol in the e-mail address). |
| `delivery-host` | The name of the MSS host on which the mailbox for this account will be located. |
| `internal-id` | A unique ID shared between the Directory database and the MSS. |
| `PopAddress` | POP/IMAP login name. |
| `password`<br>`-convert`<br>`clear\|md5-po\|unix` | Password for POP/IMAP access. If a password is specified, the hashing scheme (or lack thereof) must be indicated. Note that if the -convert option is omitted, imdbcontrol does not hash passwords. |
| `Domain` | Domain with which the account is associated. If none is specified, the default domain is assumed. |
| `A\|L\|D\|M\|S\|P` | Status of the account. The possible values for this parameter are: A (active), L (locked), D (deleted), M (maintenance), S (suspended), P (proxy). The default is A (active). |
| `S\|A` | Account type of the account. The possible values for this parameter are A (administrative) and S (standard). The default is S (standard). |
| `CosName` | Class of service with which the account is associated. If none is specified, the default class of service is assumed. |

## imdbcopyarchredo (modified)

Two new arguments, `-backupkeepcount` and `-copiedbackupkeepcount` have been added.

| | |
|---|---|
| `-backupkeepcount <n>` | Keeps around enough archived redo logs to roll forward the last `<n>` backups made by `imdbhotbackup` in the database's `archive_dest`. Delete all older archived redo logs. |
| `-copiedbackupkeepcount <n>` | Like `backupkeepcount`, except it determines how much archived redo is kept in the `-copytodisk` directory. |

## imdbhotbackup

When using the `imdbhotbackup` command, you must set the `/*/common/logDir` configuration key. If this is not specified, the command will not function properly.

## imdbindexrecreate

The `imdbindexrecreate` command drops and recreates corrupt indexes. Oracle 7.3.4.1 has an issue where indexes can become partially corrupted. When this occurs, the `imdbindexreorg` utility cannot successfully compress the index, and using the "ALTER INDEX REBUILD" SQL command also fails. The only way to repair a corrupted index is to drop it and recreate it from scratch. The `imdbindexrecreate` utility automates this task.

Only run `imdbindexrecreate` during a maintenance window. The utility breaks all connections to the database and prevents InterMail software from reconnecting to the database as it runs. Stop the MSS before running the utility against its database.

Run the `imdbindexrecreate` utility as the oracle user to ensure that the utility has permission to stop and start up the database:

% su oracle

**Syntax**

```
% imdbindexrecreate -id <user>/<pass> -dbname <ORACLE_SID>
```

Where:

| | |
|---|---|
| `<user>/<pass>` | The username and password for the Oracle InterMail user (`imail/imail` by default). |
| `<ORACLE_SID>` | The name of the Integrated Services Directory database instance (`IMD1` by default). |

## imdbspacequickcheck

Output of the `imdbspacequickcheck` utility is significant as it warns of an impending tablespace crisis. Therefore, it is important to note that the output of this command is *only* recorded in the `imdbspacequickcheck.log`.

InterMail system administrators must monitor the `imdbspacequickcheck.log` for urgent messages like the example that follows:

```
19980806 181659265 mss1 imdbspacequickcheck 20601 1 1
Urgt;DbToolsSpaceEmergencyCrisis(103/14) IMM1 POX02PK_IM_HDRADDRESSINDEX
```

## imdirsync

The `imdirsync` documentation notes that the "`imdirsync` utility will bring the directory cache up to date with the Integrated Services Directory database itself." However, it should be noted that if the Directory Cache Server is running when the `imdirsync` utility is executed, the Directory Cache Server will not use the new cache file until the period defined by the `dirCacheUpdatePeriod` configuration key has expired.

The net effect is a potential delay, as account attributes that are modified in the database are not immediately propagated to the directory cache in use.

## imfilterctrl

The `imfilterctrl` is used to set or retrieve per-user filter rules for an individual account.

**Syntax:**

```
imfilterctrl <option> <address> <domain> {-} {<filter-file>|<filter-
file to stdout>}
```

**Where:**

| | |
|---|---|
| `option` | Specifies the operation to be performed. There are three choices: get, set, and clear. |
| `get` | Retrieves per-user filtering information for a specified user. |
| `set` | Sets a new per-user rule for a specified user. |
| `clear` | Removes all rules for a specified user. |
| `address` | SMTP address of the specified user. |
| `domain` | Domain name of the specified user. |
| `filter-file` | A text file containing filtering rules. |
| `filter-file to stdout` | The file to which filter data will be written. |

## imfolderlist

The `imfolderlist` command retrieves all messages in a message store and prints their headers to standard output. This command lists the header information for all messages in a specified message store's folder.

**Syntax**

```
imfolderlist { storeName | email-address} [-folder <folderName>
[-s]] [-e] [-d] [-all] [From] [Subject] [Date] [Message-ID] [other
headers]
```

Where:

| | |
|---|---|
| `<MessageStoreName>` | Internal id of the message store. |
| `<emailAddress>` | Account's email address, including the domain name. |
| `-folder` | Lists contents of the specified folder (defaults to /INBOX) |
| `<folderName>` | Specific folder to list |
| `-s` | Also list messages in all subfolders of <folderName> |
| `-e` | Also list contents of .ERROR folder |
| `-d` | If command fails, -d will give detailed error messages |
| `-all` | Prints all the header attributes. The default is to print out the `From`, `Subject`, `Message-ID`, and `Date` attributes. |
| `otherHeaders` | Lists each header specified (if none, defaults to -all). |

**Example**

To list the `To:`, `From:`, `Subject:`, `Date:`, and `Message-Id:` lines from the headers of each message in the `/INBOX` of the Message Store identified by the internal ID 999, enter the following.

```
imfolderlist 999
```

To list the `From:` and `Content-Type:` lines from the headers of each message in the `/TRASH` folder of the mailbox which corresponds to johnDoe's account, enter the following:

```
imfolderlist johnDoe@myDomain.com -folder /TRASH From Content-Type
```

---

***Note:*** *In addition to the standard headers shown with the attribute option, you may include additional client- and server-specific headers for a given message. The header output for each header attribute consists of a message index, a colon followed by the header attribute type (for example, From), a colon and the value of the header attribute. A blank line separates the output for each message.*

---

## immovemsgfiles

The `immovemsgfiles` command will move a group of message files from the source paths to the destination paths listed in the buckets file. The `immovemsgfiles` command will also update the MSS database information for each message that was moved.

This command is useful if it is necessary to take a message file system offline and move message files to a new location. It must be executed on the same machine where the MSS is running and where the message files reside.

While running the command, the MSS may be left running. In this case, use the `-link` option to prevent unnecessary errors in mailboxes that are open. These old links should then be removed after a 24 hours.

**Syntax:**

```
immovemsgfiles [-help] [-verbose] [-noop] [source-paths...]
[-src source-path-list] [-dst buckets] [-link] [-remove]
```

where:

| | |
|---|---|
| `-help` | Reports usage statement. |
| `-verbose` | Causes command to report additional informational messages as it runs. |
| `-noop` | In this mode, no operation that would change the system is allowed to take place. Operations that would change the system are reported to `STDOUT` (preceded by `NOOP`). |
| `sourcePaths…` | A list of directories or message files which should be moved. For each directory that is listed, all message files found beneath that directory will be moved. |
| `-src sourcePathList` | Each file specified with the -src option contains a list of source paths. Each source is treated identically to the sourcePaths that were listed on the command line (as explained for the previous parameter). In order for `immovemsgfiles` to run, at least one source message file or directory must be specified, either via the sourcePaths or the sourcePathList parameters. This option may be repeated. |
| `-dst buckets` | Each specified buckets file contains a list of those directories that the command can use as target directories. Each directory listed in the buckets file is assumed to be relative to the `messageFilesDir` configuration key, unless the directory is an absolute path (beginning with slash). If no buckets file is specified, the `messageFilesDir/buckets` file is used. This is the same buckets file which the MSS uses in order to determine where to store message |

| | |
|---|---|
| | files. This option may be repeated. |
| `-link` | Once the messages have been copied to their target locations, the original message source files are replaced by links to the corresponding target message files. |
| `-remove` | Once the messages have been copied to their target locations, the original message source files are removed. |

***Note:*** *The `-link` and `-remove` cannot both be specified. If neither `-link` nor `-remove` options are specified, then the source message files are left in place, although the MSS database will no longer refer to them.*

## immsgfind

The `immsgfind` utility now looks for message-ids in the message header, not the MSS database. In order to disable this new method, use `-u` option, and immsgfind will operate as it previously did. See the *InterMail Reference Guide* for more information on the `immsgfind` utility.

## immsgverify

The `immsgverify` command prints a list of all “widows” and “orphans” to standard output. “Widows” are messages occurring in the database which have no corresponding message files in the Message File System. “Orphans” are files in the Message File System for which no database entries can be found. Since the `immsgverify` command needs to compose complete lists of messages in both the database and the Message File System, it usually takes a long time to run.

The `immsgverify` command has no options or parameters and should be run as the `imail` user on a MSS host.

## immsinit

To create the administrative message store, you may use `immsinit` with the `-w` argument. This argument allows you to specify a message in a text file. In order to perform this procedure, you may need to edit the `/*/welcomeMsgId` configuration key to a unique value (such as the combination of "Welcome" and a date/time stamp), as in the following example:

```
<Welcome.042899>
```

Next, format the welcome message with a `From:`, `To:`, `Message-ID:`, and `Subject:` header, as in the following:

```
From: Admin
To: New Users
Message-ID: <Welcome.042899>
Subject: Welcome to Everyone!
================================================================================
Welcome to InterMail by Software.com. This is your first electronic mail message.
May you have many more!
================================================================================
```

The `–a <adminID>` flag is used with the `immsinit` command to indicate the Message Store ID of the Administrative User. By default, the Administrative User is the postmaster. If you want to designate someone other than the postmaster as the Administrative User, change the default value using the `-a <adminID>` flag. See the *InterMail Reference Guide* for additional information on use of the `immsinit` command.

## impopuserstats

The `impopuserstats` utility parses the popserv.log files in the `$INTERMAIL/$logDir` directory. The output of the log file lists the number of POP user activities within each day and hour. The hour is printed as a single 24-hour number.

Sample output appears below. The first number in each row is the number of transactions by testuser1 within a single hour. The next number indicates the date (in YYMMDD format). The third column indicates GMT time.

```
venus% impopuserstats testuser1

2 970102 13
1 970102 15
1 970102 16
1 970203 14
4 970108 11
```

The output indicates that there were 2 user transactions logged by the pop-server on January 2, between 1:00 pm and 1:59 pm, one transaction that same day between 3:00 pm and 3:59 pm, and one between 4:00 pm and 4:59 pm. There was one transaction on January 3 between 2:00 pm and 2:59 pm, and four transactions on January 8 between 11:00 am and 11:59 am.

## imqueuesplit

The `imqueuesplit` command allows the user to split outgoing e-mail queues. During some service periods, when the system cannot deliver outgoing e-mail because the remote domain is not available, it must defer the mail and outgoing mail queues can get very large. Deferred mail messages reside in outgoing queues: there is one queue for each unreachable remote domain.

Periodically, InterMail will scan all out-going queues. For each queue still containing deferred e-mail messages, InterMail will attempt to connect to the corresponding domain. Once InterMail successfully connects to the remote domain, it will attempt to deliver all messages in the queue, one at a time.

In a heavily loaded InterMail system, thousands of messages could accumulate in the deferred queue while the remote domain is rejecting connections. In this case, once the remote server is again accepting connections, it could take some time to deliver all the messages, since InterMail is only delivering one message at a time for each deferred queue. In order to get InterMail to deliver several messages to the same remote domain simultaneously, it is necessary to divide the mail in the deferred queue between several different queues. `imqueuesplit` will split these outgoing queues for reprocessing messages.

**Syntax**

```
imqueuesplit [-q] <destdomain> <newdest> [<newdest> ....]
```

where:

| | |
|---|---|
| `-q` | Uses the `$queueDir` directory rather than on the `$MTASpool` directory. |
| `<destdomain>` | Domain name for outgoing queued mail. |
| `<newdest>` | Additional domain name(s) for outgoing queued mail. |

## imspoollistoldfiles (modified)

This command has changed with the introduction of the Queue Server in InterMail 4.0.

**Syntax:**

```
imspoollistoldfiles [-help] [-days <number>] [queue|spool]
```

**Where:**

| | |
|---|---|
| `-help` | Print usage information. |
| `-days <number>` | Find spool files older than `<number>` days. (Four days by default.) |
| `queue` | Examine only the queue directory. |
| `spool` | Examine only the spool directory. |

If neither `queue` nor `spool` is specified, both directories will be queried (provided they exist). The age is configurable from the command line using the `-days` option and there are no limitations on this `<number>` with respect to “today's” date.

# 6

# *Additional API Information*

This chapter supplements the API information in the *Integrated Services Directory Guide*. It includes updated information for previously documented APIs and complete descriptions for APIs not documented in the original manual.

The APIs are divided into three sections for ease of reference:

- InterMail C-API
- InterMail Perl API
- InterManager Perl API

## 6.1 InterMail C-API

This chapter supplements the InterMail C-API chapter in the *Integrated Services Directory Guide.*

### 6.1.1 InterMail C-API Function Summary

The following is a summary of API functions found in Chapter 5 of the *Integrated Services Directory Guide*.

#### *Library*

```
int IM_InitLibrary();
int IM_InitApplication(const char* appName, IM_Error*);
int IM_SetMSSConnPoolSize(int poolSize, IM_Error*);
```

#### *IM_Error*

```
int IM_InitError(IM_Error*);
int IM_FreeError(IM_Error*);
int IM_SetError(IM_Error*, int errNum, const char* string);
int IM_GetErrorMnemonic(const IM_Error*, char* buf, int bufsize);
int IM_Errno2Mnemonic(int errNum, char* buf, int bufsize);
```

### *IM_StringArray*

```
int IM_InitStringArray(IM_StringArray*);

int IM_FreeStringArray(IM_StringArray*);

int IM_CopyStringArray(IM_StringArray* dst, const IM_StringArray*
src);

int IM_ZeroStringArray(IM_StringArray*);
```

### *IM_Domain*

```
int IM_InitDomain(IM_Domain*);

int IM_FreeDomain(IM_Domain*);

int IM_CopyDomain(IM_Domain* dst, const IM_Domain* src);

int IM_ReadDomain(IM_Domain*, IM_Error*);

int IM_ReadDomains(char* from, char* till, int max,
        IM_StringArray* domains, IM_Error*);

int IM_ReadSubDomains(char* topDomain, char* from, char* till,
        int max, IM_StringArray* domains, IM_Error*);

int IM_CreateDomain(IM_Domain*, IM_Error*);

int IM_UpdateDomain(IM_Domain*, IM_Error*);

int IM_DeleteDomain(IM_Domain*, IM_Error*);
```

### *IM_Account*

```
int IM_InitAccount(IM_Account*);

int IM_FreeAccount(IM_Account*);

int IM_ReadAccount(IM_Account*, IM_Error*);

int IM_ReadAccounts(char* topDomain, char* from, char* till,
        int max, IM_StringArray*, IM_Error*);

int IM_ReadPOP3(const char* pop3, IM_Account*, IM_Error*);

int IM_CreateAccount(IM_Account*, IM_Error*);

int IM_UpdateAccount(IM_Account*, IM_Error*);

int IM_UpdateAccountAddr(IM_Account*, IM_Account* newAddress,
        IM_Error*);

int IM_DeleteAccount(IM_Account*, IM_Error*);

int IM_ReadAlias(const char* alias, IM_Account*, IM_Error*);

int IM_ReadAccountAliases(IM_Account*, IM_StringArray*, IM_Error*);

int IM_CreateAlias(IM_Account*, const char* alias, IM_Error*);

int IM_DeleteAlias(const char* alias, IM_Error*);

int IM_ReadAccountForwards(IM_Account*, IM_StringArray*, IM_Error*);

int IM_CreateForward(IM_Account*, const char* forwardTo, IM_Error*);

int IM_DeleteForward(IM_Account*, const char* forwardTo, IM_Error*);
```

```
int IM_EnableAccountForwards(IM_Account*, IM_Error*);

int IM_DisableAccountForwards(IM_Account*, IM_Error*);

int IM_ResetAcCos(IM_Account*, IM_Error*);

int IM_UpdateAcCos(IM_Account*, char* name, char* value, IM_Error*);

int IM_DeleteAcCos(IM_Account*, char* name, IM_Error*);

int IM_HashPassword(char* password, char* hashPassword, int len,
        IM_PwHashType);

int IM_CheckPassword(char* password, char* hashPassword,
IM_PwHashType);
```

### *IM_Mbox*

```
int IM_InitMbox(IM_Mbox*);

int IM_FreeMbox(IM_Mbox*);

int IM_ReadMbox(IM_Mbox*, IM_Error*);

int IM_CreateMbox(IM_Account*, const char* welcomeMsgId,
        IM_Mbox*, IM_Error*);

int IM_DeleteMbox(IM_Mbox*, IM_Error*);
```

### *IM_Folder*

```
int IM_InitFolder(IM_Folder*);

int IM_FreeFolder(IM_Folder*);

int IM_ReadFolder(IM_Mbox*, IM_Folder*, IM_Error*);

int IM_CreateFolder(IM_Mbox*, IM_Folder*, IM_Error*);

int IM_RenameFolder(IM_Mbox*, IM_Folder*, const char* dstname,
        IM_Error*);

int IM_DeleteFolder(IM_Mbox*, IM_Folder*, IM_Error*);

int IM_ScanFolderMsgs(IM_Mbox*, IM_Folder*, int numMsgs, IM_Msg**,
        IM_Error*);

int IM_DeleteFolderMsgs(IM_Mbox*, IM_Folder*,
        int numMsgRefs, int* msgRefs, IM_Error*);

int IM_MoveMsgs(IM_Mbox*, IM_Folder* src, IM_Folder* dst,
        int numMsgRefs, int* msgRefs, IM_Error*);

int IM_CopyMsgs(IM_Mbox*, IM_Folder* src, IM_Folder* dst,
        int numMsgRefs, int* msgRefs, IM_Error*);
```

### *IM_Msg*

```
int IM_InitMsg(IM_Msg*);

int IM_FreeMsg(IM_Msg*);

int IM_ReadMsg(IM_Mbox*, IM_Folder*, IM_Msg*, IM_Error*);

int IM_ReadMsgHeader(IM_Mbox*, IM_Folder*, IM_Msg*,
        const char* hdrLabel, char** hdrValue, IM_Error*);

int IM_ReadMsgBody(IM_Mbox*, IM_Folder*, IM_Msg*, int offset,
        int size, char** text, IM_Error*);

int IM_CreateMsg(IM_Mbox*, IM_Folder*, const char* from,
        const char* rfc822message, IM_Msg*, IM_Error*);

int IM_UpdateMsgFlags(IM_Mbox*, IM_Folder*, IM_Msg*,
        const char* flags, IM_Error*);

int IM_DeleteMsg(IM_Mbox*, IM_Folder*, IM_Msg*, IM_Error*);
```

### *IM_MimeInfo*

```
int IM_InitMimeInfo(IM_MimeInfo*);

int IM_FreeMimeInfo(IM_MimeInfo*);

int IM_ReadMsgMimeInfo(IM_Mbox*, IM_Folder*, IM_Msg*,
        IM_MimeInfo* parent, int index, IM_MimeInfo* output,
IM_Error*);
```

### *IM_Reply*

```
int IM_InitReply(IM_Reply*);

int IM_FreeReply(IM_Reply*);

int IM_ReadReply(IM_Account*, IM_Reply*, IM_Error*);

int IM_CreateReply(IM_Account*, IM_Reply*, IM_Error*);

int IM_UpdateReply(IM_Account*, IM_Reply*, IM_Error*);

int IM_DeleteReply(IM_Account*, IM_Error*);
```

### *IM_Cos*

```
int IM_InitCos(IM_Cos*);

int IM_FreeCos(IM_Cos*);

int IM_ReadCos(IM_Cos*, IM_Error*);

int IM_ReadCosNames(IM_StringArray*, IM_Error*);

int IM_CreateCos(IM_Cos*, IM_Error*);

int IM_SetCosAttribute(IM_Cos*, char* name, char* value, IM_Error*);

int IM_UnsetCosAttribute(IM_Cos*, char* name, IM_Error*);

int IM_DeleteCos(IM_Cos*, IM_Error*);
```

### *IM_CosAttrDef*

```
int IM_InitCosAttrDef(IM_CosAttrDef*);
int IM_FreeCosAttrDef(IM_CosAttrDef*);
int IM_CreateCosAttribute(IM_CosAttrDef*, IM_Error*);
int IM_UpdateCosAttribute(IM_CosAttrDef*, IM_Error*);
int IM_DeleteCosAttribute(IM_CosAttrDef*, IM_Error*);
```

### *IM_CosAttrDefArray*

```
int IM_InitCosAttrDefArray(IM_CosAttrDefArray*);
int IM_FreeCosAttrDefArray(IM_CosAttrDefArray*);
int IM_ReadCosAttributes(IM_CosAttrDefArray*, IM_Error*);
```

### *IM_Config*

```
int IM_InitConfig(IM_Config*);
int IM_FreeConfig(IM_Config*);
int IM_ReadConfig(IM_Config*, IM_Error*);
```

### *IM_LogContext*

```
int IM_InitLogContext(IM_LogContext*);
int IM_FreeLogContext(IM_LogContext*);
int IM_ReadLogContext(IM_LogContext*, IM_Error*);
int IM_UpdateLogContext(IM_LogContext*, IM_Error*);
```

### *IM_LogMsg*

```
int IM_InitLogMsg(IM_LogMsg*);
int IM_FreeLogMsg(IM_LogMsg*);
int IM_ReadLogMsgText(IM_LogMsg*, IM_Error*);
int IM_WriteLogMsg(IM_LogMsg*, IM_Error*);
int IM_CreateLogFile(IM_LogMsg*, IM_Error*);
```

### 6.1.2 Additional InterMail C-API Information

The information presented in this section provides clarification of discussions that were previously incomplete and acknowledges errors in the *Integrated Services Directory Guide*.

#### *IM_DeleteAccount vs. IM_DeleteMsg*

IM_DeleteAccount only deletes the account information from the database. This is not the same as removing a mailbox. In order to delete a mailbox, you must use the IM_DeleteMbox API.

#### *IM_CreateAccount*

The `IM_CreateAccount` function returns an error when an account is created with a `PWHASH` value of `IM_PWHASH_NO_CHANGE`. The `IM_PWHASH_NO_CHANGE` constant is only valid when used with `IM_UpdateAccount()`.

#### *Case corrections*

In the *Integrated Services Directory Guide*, there are several errors regarding case-sensitive commands. These commands appear in mixed-case throughout the manual when they are actually all lower-case. The correct case appears below:

```
im_init ()
im_version ()
im_errnum ()
im_erstr ()
im_errmnemonic ()
```

All mentions of `void` in the InterMail C-API documentation should be changed to `int`.

## 6.2 InterMail Perl API

The information presented in this section provides clarification of discussions that were previously incomplete and acknowledges errors in Chapter 6 of the *Integrated Services Directory Guide*.

Refer to the InterMail Perl API chapter for introductory information on InterMail Perl APIs.

### DeleteAlias ( *addr* )

An incorrect description of `DeleteAlias` was documented in the *Integrated Services Directory Guide*. The following description is correct and replaces the old definition.

**Prerequisites**

The `addr` argument must refer to an existing alias for this account.

**Implementation**

Calls `IM_DeleteAlias();`.

**Return Values**

Returns “1” on success, “0” on failure.

---

***Note:*** *The Account `DeleteAlias` method is not a member function. It is a class method. Therefore, it is not called with an Account object, but as in the following example:*
*`Account::DeleteAlias($alias);`*

---

## 6.3 InterManager Perl API

The information presented in this section provides clarification of discussions that were previously incomplete and acknowledges errors in Chapter 7 of the *Integrated Services Directory Guide*.

Refer to the InterMail Perl API chapter for introductory information on InterMail and InterManager Perl APIs.

### *Provider::Create ()*

In the InterManager Perl API, the `SwCom::Imgr::Provider::Create ()` class does not take any arguments.

***MailCos::Add ()*** Class of service attributes cannot be modified using AddAttributes. Use SetAttributes instead of AddAttributes to change a class of service.

# 7

# *Additional Event Logging*

This chapter supplements the event logging information in the *InterMail Reference Guide*. It includes updated information for previously documented events and complete descriptions for events not documented in the original manual.

---

***Note:*** *Occasionally, a message with the type "dead" is appears in the log. This is usually an error that happens when you try to retrieve a message from the MSS that is already deleted. Before the new POP session lock was put in place, this used to be a normal occurrence. Now it can happen if you use your own API's to retrieve messages that don't exists and allow more than one session at a time.*

---

Log entries are listed in alphabetical order for ease of reference.

## AcctForwardInfoInvalid

**Description:** An account has forwarding turned on, but one or forwarding addresses are unparseable.

**Parameters:** `<addr>` The primary SMTP address for the account.
`<fwda>` The returned (invalid) forwarding information.

**Cause:** Misconfigured forwarding information for account.

**Effect:** This error is of major severity (Error), but only affects the account in question. It means no mail will be delivered to that account until this is fixed.

**Action(s):** Analyze and correct the forwarding information for the account in question.

## AcctForwardInfoMissing

**Description:** An account has forwarding turned on but no forwarding info.

**Parameters:** `<addr>` The primary SMTP address for the account.

**Cause:** Misconfigured account information.

**Effect:** This severity of this event is Error. This only affects the account in question. It means no mail will be delivered to that account until this is fixed.

**Action(s):** Analyze and fix the account.

## AcctForwardTo

**Description:** A request for forwarding information failed.

**Parameters:** none.

**Cause:** The Directory Cache Server was not able to obtain information from the authoritative directory service, because of a cache lookup error.

**Effect:** This severity of this event is an error.

**Action(s):** Rebuild the directory cache using `imdirsync`.

## AcctNotTrustedLogin

**Description:** The user is not allowed to connect to the server via the specified interface address.

**Parameters:** *addr*: The address of the interface to which the client is connected.

**Cause:** The user's class of service attributes (`pref_popaccess`, `pref_popsslaccess`, or `pref_imapaccess`) do not allow access the service through the specified interface. If the class of service attributes are set to "trusted" and the interface is not listed in the `/*/common/trustedInterfaces` configuration key, the user will be denied access.

**Effect:** This message is of informational severity.

**Action(s):** Verify that the user is authorized to use the server in question and that the host's `trustedInterfaces` configuration key is correctly setup.

## CacheBadConfig

**Description:** While trying to load the cache files, the Directory Cache Server encountered an error. It found the configuration information in the cache file but the information was not correct. This information tells it how many cache files there are and other details.

**Parameters:** None.

**Cause:** The most likely cause of this error is corruption of the cache file.

**Effect:** This is a major error. The Directory Cache Server will not be able to start up till this error is rectified. No authentication will be provided by this server.

**Action(s):** Run `imdirsync` to synchronize the cache with the Directory database. Or, if another cache is determined to be healthy, run `imcachecopy` on that machine, and copy the cache over to this machine.

## CacheConfigDbFilesMismatch

**Description:** There is a mismatch between the number of files that the cache is configured to use, and the number of files specified by the `NumUserDbFiles` configuration key.

**Parameters:** `<number>` Number of files in Cache Configuration.

**Cause:** Most likely, the number of files is incorrectly specified in the Configuration database (`config.db`).

**Effect:** This is a major error. The Directory Cache Server will not start, and hence will not provide any authentication.

**Action(s):**

1. If you wish to use the cache as it is, set the configuration key `NumUserDbFiles` to the number printed as event parameter.
2. If you wish to change the number of files used by the cache, run `imdirsync` to create a new cache with the same number of files as specified by the configuration key.

## CacheConfigHashAlgUnknown

**Description:** The Cache configuration has a hash algorithm that is not known by the software.

**Parameters:** `<Number>` Hash Algorithm Number from the Cache Configuration.

**Cause:** The most likely reasons for this are either:

1. The cache is corrupted, or
2. This software is running on an installation that it is not a part of.

**Effect:** This is a major error. The software accessing the cache will not run. If it is the Directory Cache Server, it will not be able to start up.

**Action(s):** Run `imdirsync` or copy the cache from a different machine.

## CacheConfigInfoMismatch

**Description:** There is a mismatch between the information configured in the cache and the information specified by `config.db`. The cache configuration is encoded in the `common001.db` cache file.

**Parameters:** Information in the cache configuration.

**Cause:** The number of files is incorrectly specified in `config.db` or the number of files was changed and `imdirsync` needs to be run. Since this information is used in partitioning data, it is a serious error.

**Effect:** This error is of major severity. The Directory Cache Server will not start.

**Action(s):** If you wish to change the number of files used by the cache, run `imdirsync` to create a new cache with the same number of files as specified by `config.db`.

## CacheConfigNotFound

**Description:** The Directory Cache Server in trying to load the cache files encountered an error. It could not find the configuration information in the cache file. This information tells it how many cache files there are and other details.

**Parameters:** None.

**Cause:** The most likely cause of this error is corruption of the cache file.

**Effect:** This is a major error. The Directory Cache server will not be able to start up till this error is rectified. No authentication will be provided by this server.

**Action(s):** Run `imdirsync` to synchronize the cache with the Integrated Services Directory database. Or, if another cache is determined to be healthy, run `imcachecopy` on that machine, and copy the cache over to this machine.

## CacheConfigVersionMismatch

**Description:** There is a mismatch between the cache's version number and the version number that the software expects.

**Parameters:** Version in cache configuration and the version that software expects.

**Cause:** The cache is old and `imdirsync` needs to be run.

**Effect:** This error is of major severity. The Directory Cache Server will not start.

**Action(s):** Run `imdirsync` to solve the problem.

## CacheCopyFailed

**Description:** While attempting to copy the cache files to the specified directory, the Directory Cache Server encountered an error.

**Parameters:** `<sourceFile>` Source file for the copy operation.

`<destFile>` Destination file for the copy operation.

`<errorNumber>` Reason for failure.

**Cause:** The most likely cause for this failure is related to the file-system. Either the copy failed because of permissions or lack of storage resources.

**Effect:** This is a minor error. The Directory Cache Server is unaffected by this failure.

**Action(s):** Remedy the file-system permissions/storage situation, and try to run `imcachecopy` again.

## CacheGenericMsg

**Description:** A generic notification of informational messages that could be used for debugging the server.

**Parameters:** *message*: informational Message about the state of the server.

**Cause:** Multiple causes, all trivial.

 

**Effect:** None. This error is of informational severity.

**Action(s):** None. This is a notification of expected behavior.

## CacheOpenFail

**Description:** Unable to open Directory cache file pathname.

**Parameters:** `<pathname>` The pathname of the Directory cache file.

**Cause:**
1. The Directory cache or its directory does not exist, or the permissions are wrong.
2. The value for the `DBFilePath` configuration key is incorrect.

**Effect:** This error can be either minor or critical. The Directory Cache server will exit immediately, which may cause incoming mail to be deferred and POP access to be suspended. If other `imdircacheserv` processes are still running, and if the configuration on the current host is set up to fall back to these servers, then the effect will be minor. If no other `imdircacheserv` processes are running, or if fallback behavior has not been specified, the effect will be critical, since all Directory access will fail.

**Action(s):** Check the existence and permissions on the Directory cache file. Verify the configuration value for `DBFilePath`.

## CacheRealmConfig

**Description:** There was an error in the realm specification in `config.db`. The Event Parameter indicates the line and the nature of the error.

**Parameters:** Line in `config.db` which has the error.

**Cause:** The specification of the key `/*/imdircacheserv/realms` in `config.db` is incorrect.

**Effect:** This error is of major severity. The Directory Cache Server will not start.

**Action(s):** Correct the realm specification in `config.db`.

## CacheRealmDataDeleteFailed

**Description:** There was an error deleting the data from the master server of the indicated realm.

**Parameters:** The DB name where that data resided.
The DB file number.
The key of the data.

**Cause:** The remote server does not have the data, most likely because its cache is not in sync with the database.

**Effect:** This error is of minor severity. The Directory Cache Server ignores the error and updates on the remote server will catch up.

**Action(s):** Check the cache in the indicated realm to see if it is in sync with the database.

## CacheRealmDataFetchFailed

**Description:** There was an error retrieving the data from the master server of the indicated realm.

**Parameters:** The DB name where that data resided.

The DB file number.

The key of the data.

**Cause:** The remote server does not have the data, most likely because its cache is not in sync with the database.

**Effect:** This error is of minor severity. The Directory Cache Server will read-through to Oracle.

**Action(s):** Check the cache in the indicated realm to see if it is in sync with the database.

## CacheRealmDataStoreFailed

**Description:** There was an error storing the data into the master server of the indicated realm.

**Parameters:** The DB name where that data resided.

The DB file number.

The key of the data.

**Cause:** The remote server already has the data or there was a storage error in the remote server's cache.

**Effect:** This error is of minor severity. The Directory Cache Server ignores the error.

Action(s): Check the cache in the indicated realm to see if it has any storage problems. If there are storage problems, all write operations should fail.

## CacheRealmServerDown

**Description:** The directory cache server is not operational.

**Parameters:** *realm-number*: realm number of the downed server

*hostname*: name of host that is down

**Cause:** The Directory Cache Server on a particular hostname is not responding. The process could have terminated unexpectedly or configured incorrectly.

**Effect:** This error is of critical severity. The directory cache server needs to be running for customers to be authenticated to retrieve their mail.

**Action(s):** Verify the Directory Cache Server is NOT running. Investigate the log file for this process to determine the cause of this failure. If the Directory Cache Server is running, verify that its port configurations are correct.

## CacheRealmServerUp

**Description:** The Directory Cache Server is running on a particular hostname. This is not an error but an informational event.

**Parameters:** *realm-number*: realm number of the downed server

*hostname*: name of host that is down

**Cause:** Not applicable

**Effect:** None. This message is of informational severity.

**Action(s):** None.

## CacheReopenCloseError

**Description:** The Directory Cache Server, detecting that the cache was overwritten, attempted to close the old cache and reopen the new one. It encountered an error while closing the file.

**Parameters:** None.

**Cause:** This error is rare and should not happen. Likely cause is a possible file-system problem.

**Effect:** This is a major error. The Directory Cache Server will not be able to service any requests from its cache. It will have to go to the Directory database to authenticate.

**Action(s):** Check the file-system for problems. Restart the Directory Cache server.

## CacheReopenError

**Description:** The Directory Cache Server, detecting that the cache was overwritten, attempted to close the old cache and reopen the new one. It encountered an error while reopening the new cache.

**Parameters:** None.

**Cause:** This error is rare and should not happen. Likely cause is a possible file-system problem.

**Effect:** This is a major error. The Directory Cache Server will not be able to service any requests from its cache. It will have to go to the Integrated Services Directory database to authenticate.

**Action(s):** Check the file-system for problems. Restart the Directory Cache server.

## CacheUpdateCompleted

**Description:** Cache is updated to the log id (cache update id). It counts update log entries that are processed in time seconds.

**Parameters:** *cache update id* - The id of the most recent update log entry processed by Directory Cache Server.

*count* - The number of update log entries that were processed.

*time* - The time needed to process the entries in seconds.

**Cause:** The Directory Cache Server process has finished processing a set of updates from the master Oracle Directory.

**Effect:** This message is of informational severity.

**Action(s):** None.

## CacheUpdateOutOfRange

**Description:** The log entry ids are out of range, `cache`=cache log id `oldest`=oldest log id `newest`=newest log id.

**Parameters:** The `cache` id is the most recent log entry processed by Directory Cache Server.

The `oldest` log id is the oldest log entry still available from the master directory.

The `newest` log id is the most recent log entry in the master directory.

**Cause:** The Integrated Services Directory cache file is out of sync with the master, most likely because it has fallen too far behind in processing log entries. The Directory Cache Server can no longer be automatically synchronized with the master directory.

**Effect:** Urgent

Directory queries will return some outdated information. For example, accounts and entries that have been updated will continue to use old information and entries that have been deleted or disabled will still be accessible. Newly created accounts and updates to account passwords, will generally be recognized due to read-thru. This problem can become more critical as updates accumulate in the InterMail directory.

**Action(s):** There should be at least one other Directory Cache Server process to act as a backup for this one. Stop the Directory Cache Server on the machine with the error. Replace the current directory cache with an updated cache, either by copying one from another server using `imcachecopy`, or by running `imdirsync`. Restart the Directory Cache Server.

## CacheUpdateStarted

**Description:** Cache update started with the log ids, `cache`, `oldest`, or `newest`.

**Parameters:** *cache log id* - The id of the most recent log entry processed by Directory Cache Server.

*oldest log id* - The oldest log entry still available from the master directory.

*newest log id* - The most recent log entry in the master directory.

**Cause:** The Directory Cache Server is starting to process a set of updates from the master Oracle Directory.

**Effect:** This message is of informational severity.

**Action(s):** None.

## CacheUpdateTimeTooLong

**Description:** The cache update took an unusually long time to complete (update time is in seconds).

**Parameters:** The time it took to update the cache.

**Cause:** Something is making the Directory Cache Server take more than the configured time to process a set of updates from the master, as defined by the `dirCacheUpdatePeriod` configuration key. This could be due to a very large number of changes to the database in a short amount of time.

**Effect:** WARNING

There is no direct effect on service, but there could be secondary effects. If there was a large amount of new updates to the master oracle directory, then the next update may take even longer than the previous update, then the next update will take even longer, ad infinitum. Therefore, it is important to find this log event early, as this problem has the potential to become more critical.

**Action(s):** Verify that the connection between the master Oracle Directory and Directory Cache Server is working properly. Also make sure the `dirCacheUpdatePeriod` configuration key is not set too low.

## ConfNoDomainName (modified)

This event previously had a severity rating of `Urgent`. Now, when this event is reported, it has a `Fatal` severity level. If this event occurs, the MTA will exit and will not run.

## ConfParmError

**Description:** A bad key was found in the configuration file.

**Parameters:** *badKey*: the bad key found in the configuration file.

*error*: the actual problem with the bad key.

**Cause:** This indicates that a bad key was found in the configuration file and it was not specified in the format required. Refer to the manual for the proper specification of the key in question.

**Effect:** The configuration key may not be producing the desired effect. It is possible that InterMail is ignoring the key.

**Action(s):** Fix the configuration key using the `imconfedit` utility.

## DbDatabaseNote

**Description:** The database operation `sqlop` was requested by the InterMail Directory server, the Message Store server, or `imdirsync` performed the operation

**Parameters:** *sqlop* - The database operation that was in progress when the event occurred.

*event* - The event that the database server wanted to report.

**Cause:** The event is a normal event that is being logged for notification purposes only.

**Effect:** None.

**Action(s):** None.

## DbFeatureUnavailWithoutSchemaPatch

**Description:** The InterMail MSS is designed to adapt to several different versions of the MSS database schema. However, if the schema is not the latest version, some features of the MSS might not be available.

***Note:*** *Schema patching instructions can be found in the release notes.*

**Parameters:** *string*: a sentence that briefly describes the unsupported feature.

**Cause:** There was an attempt to use a feature that is not available without a schema patch.

**Effect:** This error is of minor severity. Whatever actions are associated with the feature do not occur, but all other processing takes place. Sometimes the consequence is merely decreased performance.

**Action(s):** Determine what feature is involved, and if it is really needed or not. If the feature is needed, then the schema must be patched.

## DbRemovingConnection

**Description:** The server is unable to restore a connection it previously had to the database. It is therefore removing the connection from its internal pool of database connections and is now running with reduced database access.

**Parameters:** None.

**Cause:** A database connection was lost and the `dbReconnectInterval` key is set to prevent automatic reconnection. See also the `DbTempDisconnect` HTML page.

**Effect:** The server runs with reduced database access.

**Action(s):** Choose one of the following:

- Continue running with reduced database access.
- Restart the server to re-initialize the database connection pool.
- Set the `dbReconnectInterval` key to a non-zero value. This will enable periodic server reconnection attempts in the future.
- Restart the server after setting the key.

## DbTempDisconnect

**Description:** The server temporarily lost a connection with the database, causing a failure of the attempted operation.

**Parameters:** None.

**Cause:** High CPU load.

Interruption of the Oracle listener process.

Interruption of an Oracle client process.

Interruption of an Oracle server process.

**Effect:** A single operation fails.

**Action(s):** Monitor system CPU load. Run `$ORACLE_HOME/bin/lsnrctl status` to verify the integrity of the Oracle listener. Examine the Oracle alert log for recent database failures.

## DbTnsAdminUndefined

**Description:** A definition of `TNS_ADMIN` could not be found in either the configuration database or the user's runtime environment. Without `TNS_ADMIN` defined in the runtime environment, a connection to Oracle cannot be established.

**Parameters:** Description of failed operation.

**Cause:** `tnsAdmin` is not defined in the configuration database and `TNS_ADMIN` is NOT defined in the users environment.

**Effect:** FATAL

**Action(s):** Verify that tnsAdmin is defined in the configuration database or `TNS_ADMIN` is defined in the user environment.

## DbOracleHomeUndefined

**Description:** A definition of `ORACLE_HOME` could not be found in either the configuration database or the user runtime environment.

**Parameters:** `<explanation>` Description of failed operation.

**Cause:** The `oracleHome` key is not defined in the configuration database and `ORACLE_HOME` is NOT defined in the user's environment.

**Effect:** The severity of this event is fatal. Without `ORACLE_HOME` defined in the runtime environment, a connection to Oracle cannot be established.

**Action(s):** Ensure that the `oracleHome` key is defined in the configuration database or `ORACLE_HOME` is defined in the user environment.

## DbTnsAdminUndefined

**Description:** A definition of `TNS_ADMIN` could not be found in either the configuration database or the user's runtime environment.

**Parameters:** `<explanation>` Description of failed operation.

**Cause:** The `tnsAdmin` key is not defined in the configuration database and `TNS_ADMIN` is not defined in the user's environment.

**Effect:** The severity of this event is fatal. Without `TNS_ADMIN` defined in the runtime environment, a connection to Oracle can not be established.

**Action(s):** Ensure that the `tnsAdmin` key is defined in the configuration database or `TNS_ADMIN` is defined in the user environment.

## DbWritesBlocked

**Description:** Writes to the directory cache file have been temporarily blocked, presumably so that the directory cache file(s) can be copied using `imcachecopy` without getting a corrupted result. Corruption could occur if the source was written to while it was being copied, so that the target only contained part of the write. This is a normal event when using `imcachecopy`. When `imcachecopy` finishes, it will automatically re-enable writes and a `DbWritesUnblocked` message will be indicated in the log file.

**Parameters:** This is the name of the directory cache file or directory.

**Cause:** Writes to the directory cache file have been temporarily blocked.

**Effect:** Writes are blocked, but directory cache file access and read-through ability is unaffected. Writes will automatically be re-enabled when `imcachecopy` completes, at which point the Directory Cache Server will catch-up on the missed updates via the periodic update sync mechanism.

 

**Action(s):** Perform a ps(1) and verify that `imcachecopy` is running. If it is not running, something is very wrong, and writes can be re-enabled via `imcachecopy` -on, but this should only be never be required, and only exists for emergencies. If this option is used during normal operation, it will most likely corrupt the `imcachecopy` operation.

## DbWritesUnblocked

**Description:** Writes to the directory cache file have been unblocked, presumably because `imcachecopy` finished copying the directory cache file(s) and automatically re-enabled write activity. Any writes that were missed by the Directory Cache Server will be performed by the periodic update sync mechanism. This is a normal event when using `imcachecopy`.

**Parameters:** This is the name of the directory cache file or directory.

**Cause:** The `imcachecopy` utility finished copying the directory cache file(s).

**Effect:** None.

**Action(s):** None.

## FiltActionRewriteHeader

**Description:** A mail filter has processed a message and caused it to rewrite it's header.

**Parameters:** *filterName*: the name of the filter being executed.

*attributes*: a list of the changed header attributes.

**Cause:** The “write-header” or “forward header” action was used in the filter.

**Effect:** This message is of informational severity.

**Action(s):** none required.

## FiltActionSendMessage

**Description:** A mail filter has processed a message that has triggered a sendmessage action. This action will initiate and deliver a new message specified by the source filter. This action is a side effect and will not prevent the triggering message from being delivered.

**Parameters:** *filterName* – The name of the filter being executed.

**Cause:** The “sendmessage” action was used in the filter.

**Effect:** This message is of informational severity.

**Action(s):** None.

## FiltHeaderRewriteFailed

**Description:** An attempt to change the header of a message failed.

**Parameters:** *filterName*: the name of the filter being executed.

*attributes*: a list of the changed header attributes.

**Cause:** The machine is running out of memory or disk space.

**Effect:** The current message will be placed in the deferred queue until system resources become available

**Action(s):** none

## FioWriteFail

The description for the FioWriteFail log event should read, “The file system is full,” not “The system file is full.” For a complete description of the FioWriteFail log event, see the *InterMail Reference Guide.*

## ImapNoProxyLogin

**Description:** The IMAP server does not support proxy mode. Accounts that are in proxy mode are not accessible via the IMAP server.

**Parameters:** none.

**Cause:** An IMAP client tried to log into an account which is in proxy mode.

**Effect:** The severity of this event is Informational.

**Action(s):** None.

## JrnBadNumJournalsParm

**Description:** The configuration file specifies to run the MSS with <n> journal files, which is an illegal value.

**Parameters:** *numJournalfiles* - Specified number of journal files.

**Cause:** Number of journal files specified is invalid.

**Effect:**

**Action(s):**

## MsDeliverFolderNotFound

**Description:** The attempt to deliver a message to the message store failed because it the destination folder did not exist.

**Parameters:** none

**Cause:** The folder did not exist. The mailbox's quota might be full, or system resources might not be available.

**Effect:** This error is of minor severity. Affects a single message store only: the message will be delivered to the inbox folder.

**Action(s):** none.

## MsFolderNotRemovable

**Description:** A client of the message store server attempted to remove a folder that cannot be removed. This folder is probably the trash folder.

**Parameters:** *folderName*: the name of the folder.

**Cause:** An attempt to remove the trash folder by a Message Store Server client.

**Effect:** This error is of informational severity; it affects a single message store only.

**Action(s):** This is a user error.

## MsInitFailure

**Description:** The MSS constructor failed to initialize successfully.

**Parameters:** None.

**Cause:** See previous Error events in the MSS log file.

**Effect:** The severity of this event is an error. This affects a single mailbox. This mailbox will have no access.

**Action(s):** Check the MSS logs for the immediately preceding Error event, correct the problem indicated, and restart the MSS.

## MsNoSuchObject

**Description:** The specified object no longer exists in this mailbox. This can happen if an MSS client attempts an operation on a folder or message which has been totally expunged from the database and/or the MSS.

**Parameters:** *MsName*: the mailbox in which the object exists.

*objRef*: the reference number of the object.

*realTypeName*: the real type name of the object.

*RmeOp*: the most recent operation performed in this thread.

**Cause:** Caused by a client-server timing issue or a synchronization issue between separate MSS sessions.

**Effect:** This event should have no effect beyond the failure of the operation regarding the specified object.

**Action(s):** Ignore this error unless it occurs in mass quantities. If there are numerous `MsNoSuchObject` errors, verify the integrity of the database, restart the MSS and contact Software.com support.

## MsObjTypeMismatch

**Description:** A handle to a MSS object was resolved but produced an object of the wrong type. The object name consists of `messageStoreName:refNumber:typeName`.

**Parameters:** *msName*: the mailbox in which the object exists.

*objRef*: the reference number of the object.

*realTypeName*: the real type name of the object.

*RmeOp*: the most recent operation performed in this thread.

**Cause:** Memory corruption. This is a programming error.

**Effect:** This error is of critical severity. It affects all message stores served by this MSS.

**Action(s):** The continued viability of the MSS is seriously in question. It is recommended that you restart the MSS's. Check for other InterMail and system errors. If the error continues, contact Software.com customer support.

## MsgTrace (modified)

Some changes have occurred to this message as it is reported in an `mta.log`. Previously, when an undeliverable message was moved to the system's `errors` directory, a log message like the following would appear in the MTA log file:

```
19980406 143557723-0400 venus mta 20333 9 26 Note;MsgTrace(65/26)
moved to error directory:from=<:size=454:msgfile=/nfs/sandbox/ \
granoff/imsrc/main/mercury/testing/spool/mail/errors/ \
19980406182655.AAH20333@venus.software.com-Control:
msgid=19980406182645.AAG20333@venus.software.com:
queuehost=venus
```

However, in 4.0, the MTA typically runs in stateless mode and messages are moved to a different errors directory on a Queue Server host (noted here by the `queuehost=` argument).

Therefore, the "moved to error directory" log entry now has the following form:

```
19980406 171752369-0400 venus mta 18012 9 27 Note;MsgTrace(65/26)
moved to error directory:from=<:size=449:msgfile=/nfs/sandbox/
granoff/imsrc/main/mercury/testing/spool/mail/errors/
19980406211635.AAF18012@venus.software.com-Control:
msgid=19980406211614.AAE18012@venus.software.com
```

This message is logged when the message was *not* moved to a Queue Server `error` directory. Note the absence of the `queuehost=` argument as well as full path to the control file.

In addition, the following, slightly different log entry has been added in the event that the message was moved to a Queue Server's errors directory:

```
19980406 165841713-0400 venus mta 18012 9 21 Note;MsgTrace(65/26)
moved to queue/errors directory:from=<:size=453:
```

 

```
msgfile=19980406205240.AAC18012@venus.software.com-Control:
msgid=19980406205232.AAB18012@venus.software.com:
queuehost=venus
```

Note that `queuehost=` is present and is an important clue to finding the control file. Also, `msgfile=` only indicates the base name of the control file. But with that, and the `queuehost` name as well as the clue that the file was moved to `queue/errors`, the file can be located.

## MsStaleRef

**Description:** The specified object has been deleted in the MSS and is no longer available to MSS clients. This can happen if a MSS client attempts an operation on a folder or message which has recently been deleted in a separate MSS session.

**Parameters:** *msName*: the mailbox in which the object exists.

*objRef*: the reference number of the object.

*realTypeName*: the real type name of the object.

*RmeOp*: the most recent operation performed in this thread.

**Cause:** A client-server timing issue or a synchronization issue between separate MSS sessions.

**Effect:** This event should have no effect beyond the failure of the operation regarding the specified object.

**Action(s):** Ignore this error unless it occurs in mass quantities. If there are numerous `MsStaleRef` errors, verify the integrity of the database, restart the MSS and contact Software.com support.

## MtaHeaderNotTerminated

**Description:** One or more lines of the `-Header` file are not properly terminated with end of line characters.

**Parameters:** None.

**Cause:** The `–Header` file was possibly corrupted while on disk.

**Effect:** The message cannot be delivered and is placed in the `errors` directory.

**Action(s):** Check the disk that the message was stored on (`queueserver` or `mtaSpool` directories) for disk errors or other problems.

## MtaHeaderRewriteFailed

**Description:** An attempt to change the header of a message failed.

**Parameters:** none

**Cause:** The machine is running out of memory or disk space.

**Effect:** The current message will be placed in the deferred queue until system resources become available.

**Action(s):** none

## MtaQueueServerIterationMismatch

**Description:** The MTA maintains consistency with the Queue Server through the use of iteration numbers. Any time the MTA disconnects from the Queue Server and reconnects (for instance, the Queue Server was rebooted), then the iteration number for the Queue Server is incremented. Each message that the MTA is currently manipulating on the Queue Server is also tagged with that same iteration number. It should always be the case that the iteration number of the message matches the MTAs iteration number for the Queue Server. If the iteration number does not match, then the `MtaQueueServerIterationMismatch` event is logged.

***Note:*** *The iteration number is not maintained with the file, so a deferred piece of email does not maintain the iteration number it had while the MTA was processing it. When the MTA reprocesses that deferred e-mail message, it will get assigned a new iteration number that matches the current iteration number of the Queue Server connection. The iteration number is only maintained for messages that are currently being processed by the MTA and are secured on the Queue Server.*

**Parameters:** *filename*: the file that the MTA attempted to use.
*operation*: The operation attempted by the MTA.
*Queue Server iteration number*: the MTA's Queue Server connection number
*MTA iteration number:* the message's Queue Server connection number.

**Cause:** Unclear. This should not happen during normal operation.

**Effect:** The MTA will stop handling that message. It may re-process it later, or it may have already been processed and needs to be removed.

**Action(s):** Inspect the `mta.log` to see if the message was delivered; if so, simply remove it. Otherwise make sure that the message Control file is in `queue/deferred/MTA` and that the corresponding Header and Body parts exist under queue/messages. Report this error to Software.com.

## NioConnNotAllowed

**Description:** An attempted connection was dropped because the IP address was not listed in the access list for that port. The access list is stored in the `accesstype` configuration key.

 

**Parameters:** *ipaddress*: the IP address of the machine that was rejected.

*keyname*: the configuration key that controls this type of access.

**Cause:** The access control list does not allow connections from the IP address given.

**Effect:** None. This might be a warning of an attempted intrusion to the machine. The IP address listed should be investigated to determine why it was attempting to connect. It might also be possible that if the access control lists for this port are incorrect, a host that should be able to connect is being denied service.

**Action(s):** Verify the access control list, and determine the identity of the remote host is that is attempting access if the message is frequent or concerning.

## NioConnectionLimitTableBadEntry

**Description:** An entry for the limit on number of connections for an IP address is invalid. Each entry must be of the form:

```
ip_address /significant bits] : num_connections
```

This event occurs if the entry is not in this form.

**Parameters:** None.

**Cause:** An invalid entry in the `config.db` for the configuration key `clientConnectionLimitTable`.

**Effect:** This affects the MTA. If this event occurs, the table entry is ignored.

**Action(s):** Change the invalid entry of `clientConnectionLimitTable` in the `config.db` using `imconfedit`.

## NioGetHostNameFail

**Description:** An attempt was made to lookup a hostname, typically with the call to `gethostname`, which failed with error reason.

**Parameters:** *hostname*: the hostname we were trying to lookup.

*operation*: the method we were using to lookup the hostname.

*reason*: the reason that it failed.

**Cause:** The machine could be misconfigured in DNS or NIS or the hostname could be invalid.

**Effect:** The effect on service depends on what hostname lookup is actually failing.

**Action(s):** Using the error information reason determine which condition caused this error. Fix DNS, NIS as appropriate, the configuration database or other source of the non-existent hostname. If none of these troubleshooting tactics work, contact Software.com.

## NioGetSockNameFail

**Description:** When trying to get the name (address) of the socket a client connected to, the system call, `getsockname()`, failed with a `systemErrorString` system error.

**Parameters:** *systemErrorString* – The string the system reported as being the cause of the error.

**Cause:**
- Insufficient memory.
- Insufficient streams resources.
- Programming Error.

**Effect:**

**Action(s):** Use the `systemErrorString` system error to determine the cause of the event. If it is determined to be a programming error, contact Software.com.

## NioFcntlFail

**Description:** The call to `fcntl()` failed with error `systemerrstring`..

**Parameters:** *systemerrstring* – The system error string.

**Cause:** It could be one of several causes. The `systemerrsting` should indicate the reason.

**Effect:** The severity of this error is major.

**Action(s):**

## NioRecvFail

**Description:** The system call to recv failed. This is due to either a system error or a network connection problem.

**Parameters:** None.

**Cause:** System error or network connection problem.

**Effect:** The severity of this event is MAJOR in the context of one of the servers.

**Action(s):** Look in the logs for events that would indicate either a system error or network connection problem.

## NioServerGoneDown

In the manual, it incorrectly states that this log event only applies to errors with the Message Store database. However, `NioServerGoneDown` reports errors for *all* servers, not just the Message Store database.

## PopConnectionNotAllowed

**Description:** This warning message is reported to the popserv.log file when a connection is attempted to the POP Server from an IP address that is not allowed, as defined by the `/*/popserv/allowedIPs` configuration key.

**Parameters:** IP Address that was trying to connect.

**Cause:** A connection is attempted to the POP Server from an IP address that is not allowed, as defined by the `/*/popserv/allowedIPs` configuration key.

**Effect:** The connection is dropped.

**Action(s):**

## PopDomainTableEntryError

**Description:** At least one of the values in the `loginDefaultDomainTable` (found in the configuration database) contains an entry which does not follow this format:

`<ipaddr>:<defaultDomain>`

The `ipaddr` should be in the form, xxx.xx.xx.xx, not xxx:xx:xx:xx.

**Parameters:** *popdomaintable* – The table entry which caused the problem.

**Cause:** An entry in the `loginDefaultDomainTable` was incorrectly entered in the configuration database.

**Effect:** The `loginDefaultDomainTable` entry mentioned in the log message, as well as any entries which follow it (within the `loginDefaultDomainTable` table), will be ineffective.

**Action(s):** Make the appropriate corrections to the `loginDefaultDomainTable`, using `imconfedit`. Services will not need to be restarted.

## PopMaxSessions

**Description:** This warning message is reported to popserv.log file when the maximum number of POP sessions is reached, as defined by the `/*/popserv/maxSessions` configuration key.

**Parameters:** None.

**Cause:** The maximum number of POP sessions is reached., as defined by the `/*/popserv/maxSessions` configuration key.

**Effect:** Until a pop session ends, any additional pop sessions will hang, waiting.

**Action(s):** Increase the `popserv/maxSessions` configuration variable value or add more POP Server machines.

## ProcChownFail

**Description:** An attempt to change the ownership of a file with chown(2) failed.

**Parameters:** *filename* - The name of the file.

*userId* - The (numeric) value of the user id.

*groupId* - The (numeric) value of the group id.

*systemErrorSTring* - The system error string

**Cause:** Refer to the system error for the exact cause. Most likely the error will be the result of a permissions problem.

**Effect:** The effect on service currently is minimal. The only files which we `chown()` are our log files.

**Action(s):** Check that user and group ids are correct for the configuration parameter named `commonUser`.

## ProcCloseFail

**Description:** The file `<fileDescriptor>` failed to close and reported a system error `<systemerrstring>`. This happens when trying to close a message file.

**Parameters:** `<fileDescriptor>` The file descriptor that failed to close.

`<systemerrstring>` The system error string.

**Cause:**

**Effect:** Messages from third-party libraries (specifically SNMP) could be sent to standard error.

**Action(s):**

## ProcDupFailed

**Description:** A call to dup(2) return an unexpected fd.

**Parameters:** `<fdExpected>` The fd expected from the call of dup(2).

`<fdReturned>` The fd returned from the call of dup(2).

**Cause:**

**Effect:** Messages from third-party libraries (specifically SNMP) could be sent to standard error.

**Action(s):**

## ProcDupSurprise

This event is the same as the `ProcDupFailed` event previously documented.

## ProcNoRunDir

**Description:** The server has not changed its directory to its `runDir`

**Parameters:** None.

**Cause:** If no `runDir` was specified in the configuration database for this server, then the message will be of severity "`Warning`" to indicate this fact.

**Effect:** Depends on the severity, because if "`Fatal`", the server will not run.

**Action(s):** Check for existence, ownership, and permissions of the directories.

## ProcObjBadRefCnt

**Description:** A code object had a reference count which was invalid for the place in the code where it was detected. This is an internal error.

**Parameters:** Object Name
Reference Count
Source File
Source Line
Additional Data

**Cause:** Invalid reference count in a code object.

**Effect:** Indeterminate.

**Action(s):** Contact Software.com support.

## ProcPipeCreateFailed

**Description:** A call to pipe(2) failed.

**Parameters:** `<systemError>` The system error after the failed call to pipe(2).

**Cause:**

**Effect:** Messages from third-party libraries (specifically SNMP) could be sent to standard error.

**Action(s):** Analyze reasons for failure.

## ProcReExecingProg

**Description:** Records re-execution of a program.

## ProcSetuidSucceed

**Description:** Indicates the successful setting of the unique ID.

## ProcUncaughtCPlusPlusException

**Description:** A C++ exception was raised that was not caught in the code. InterMail does not raise or catch C++ exceptions.

**Parameters:** *exceptionName*: The name of the exception.

**Cause:** Either a system library or third-party library has raised a C++ exception.

**Effect:** This is a major or critical error, and may cause a disruption of service.

**Action(s):** Contact Software.com Customer Support.

## ProcWriteToStderr

**Description:** A non-InterMail library has written a line to standard error. Normally, InterMail servers do not write to standard error unless explicitly instructed to with the `-warnToConsole` option, or while tracing in debuggable code.

**Parameters:** <message> The message that was written to standard error.

**Cause:** Ill-behaved third-party libraries.

**Effect:** Unexpected behavior from third-party libraries.

**Action(s):** Contact Software.com Technical Support

## QsrvNotLocked

**Description:** In response to a client request, the Queue Server attempted to perform an operation on a file that should have been previously locked by the client, but wasn't.

**Parameters:** *filename*: the file that the MTA attempted to use.

*operation*: the operation that the MTA attempted to do with the file.

**Cause:** Unclear. This should not happen in normal operation.

**Effect:** This event should not affect mail delivery, but it should be reported to Software.com.

**Action(s):** Report the error to Software.com.

## RmeConnectionUnavailable

**Description:** The RME connections for retrieving mail from an MSS are limited (see the `maxMSSRetrieveConnectionList` configuration key) and an RME connection could not be obtained within the timeout specified by the `maxMSSRetrieveConnectionTimeout` configuration key.

**Parameters:** *mssHost*: The MSS host we were trying to connect to.

**Cause:** The MSS specified is very busy, running slowly, or the limit for that host in the `maxMSSRetrieveConnectionList` configuration key is too low.

**Effect:** The client trying to retrieve mail thinks that the mail services are currently unavailable. This affects just that single client.

**Action(s):** Look at the performance of the MSS and the number of connections to it. Check the value of the `maxMSSRetrieveConnectionList` configuration key.

## RmeInvalidCOSAttribute

**Description:** The current server requested an undefined or invalid class of service attribute.

**Parameters:** None.

**Cause:** If a server (MTA, POP Server or IMAP Server) requests the value of an undefined or invalid class of service attribute, this error is returned. This can happen if the Directory Cache Server's primary directory database is missing this requested class of service attribute.

**Effect:** The server that received the error will gracefully shutdown and print this error message.

**Action(s):** Upgrade the schema on the primary directory database.

## RmtDupFailed (modified)

**Description:** A call to dup(2) failed.

**Parameters:** `<systemError>` The system error after the failed call to dup(2).

**Cause:**

**Effect:** The actions requested of the Manager Server may not have been carried out.

**Action(s):** Analyze reasons for failure.

## RmtDupSurprise

**Description:** A call to dup(2) return an unexpected fd.

**Parameters:** `<fdExpected>` The fd expected from the call of dup(2).

**Cause:**

**Effect:** The actions requested of the Manager Server may not have been carried out.

**Action(s):** Analyze reasons for failure.

## RunMssQuotaMaxMsgs

**Description:** A message was rejected because the destination mailbox already contained the maximum number of messages, as dictated by the class of service for this mailbox's account.

**Parameters:** None.

**Cause:** Too many messages in the destination mailbox.

**Effect:** Mail for the destination mailbox will continue to be rejected until the class of service is changed or messages are deleted out of the mailbox.

**Action(s):** Change the class of service or delete messages, whichever is desired.

## SidelineFailure (modified)

Now reports that "The message (id=%s) was sidelined in the deferred directory," where %s is a Message ID.

## SmtpBadRelayPolicy

**Description:** The anti-spam relay policy is misconfigured. The `config.db` entry is not recognized as a valid mode and the default policy of "allow all" is used instead.

**Parameters:** An incorrect mode was used in the anti-spam relay policy.

**Cause:** User error. The `config.db` file has an invalid entry for the `relaySourcePolicy` configuration key.

**Effect:** The relaySourcePolicy configuration key will default to "allow all."

**Action(s):** Fix the entry in the `config.db` file.

## SmtpSenderAddrInvalid

**Description:** An incoming message was rejected because the sender's address was invalid (e.g. improperly formatted).

**Parameters:** Invalid address.

**Cause:** An invalid `MAIL FROM:` address was entered.

**Effect:** Mail coming from the `MAIL FROM:` address will be blocked.

**Action(s):** Verify that the `MAIL FROM:` address is correct and resend the mail.

## SnmpCantConnectToMasterAgent

**Description:** The Emanate library code cannot connect to the master SNMP agent.

**Parameters:**

**Cause:**

**Effect:** The server will not be able to report SNMP traps or handle requests.

**Action(s):** If the SNMP server has crashed or not been started, doing that will eliminate the problem. If it is not your intention not to use SNMP at your site, you should have answered the appropriate questions to that effect in the installation process. This would have set the _opt configuration parameters to "-nosnmp," in addition to any other appropriate options. If this has been subsequently undermined, you may need to restore these configuration parameters to their original post-install values, or at least to `-nosnmp`.

## SnmpInitFailed

**Description:** The call to InitSubAgent() in the Emanate library failed.

**Parameters:**

**Cause:**

**Effect:** The particular server will run, but will not be SNMP-enabled.

**Action(s):**

# 8

# *Additional Class of Service Attributes*

This chapter supplements the class of service attribute information in the *Integrated Services Directory Guide*. It includes updated information for previously documented attributes and complete descriptions for attributes not documented in the original manual.

## 8.1 Preferences

The following table provides a list of the available class of service preference attributes. You can also add custom attributes to this list if you need to extend the available class of service options. Note that the Rule column uses the following for precedence rules:

- `C` - Class of service value used
- `A` - Account value used
- `L` - Lesser value used
- `G` - Greater value used

For further information setting class of service attribute permissions, see the *Integrated Services Directory Guide*.

| Attribute | Rule | Type | Description |
|---|---|---|---|
| `pref_webmailusesignature` | A | Boolean | Insert Signature in WebMail messages. |
| `pref_popaccess` | A | String | Defines access restrictions for standard POP retrieval. |
| `pref_popsslaccess` | A | String | Defines access restrictions for POP retrieval with SSL |
| `pref_imapaccess` | A | String | Defines access restrictions for IMAP. |
| `pref_mtaFilterPerUser` | A | Boolean | Turns on/off per-user filtering |

## 8.2 Permissions

The following table defines the account permissions associated with a class of service. In all cases, permissions are numeric data types, with a value of `1` indicating `true` (access allowed), and a value of `0` indicating `false` (access denied).

For further information on setting class of service permissions, see the *Integrated Services Directory Guide*.

| Attribute | Rule | Type | Description |
|---|---|---|---|

| Attribute | Rule | Type | Description |
|---|---|---|---|
| `perm_mtaFilterPerUser` | L | Numeric | Controls end user's ability to set `pref_mtaFilterPerUser` |

# 9

# *Server Statistics*

The list of statistics reported by the Directory Cache Server has changed. The accurate listing appears in the table that follows.

| Event | Description |
|---|---|
| `StatStartUp` | Directory Cache Server startup time |
| `StatQueriesProcessed` | Total number of queries processed |
| `StatQueriesSucceeded` | Total number of queries that succeeded |
| `StatQueriesFailed` | Total number of queries that failed |
| `StatUnknownUserReturns` | Number of user validation requests that returned UNKNOWN USER |
| `StatBadPasswordReturns` | Number of validation requests that returned BAD PASSWORD |
| `StatOKPasswordReturns` | Number of validation requests that returned PASSWORD OK |
| `StatConnections` | Total number of current connections |
| `StatFailedConnections` | Number of failed connections (connections fail when an attempt to create a socket for the client fails, typically because the maximum number of connections has been reached.) |
| `StatAccumulatedReadThrus` | Number of total read-throughs to the directory |
| `StatAccumulatedConnections` | Number of total connections accumulated |
| `StatUpdateTime` | Number of seconds spent in processing (`imdircacheserv`) cache updates |
| `StatRecentUpdates` | Number of updates from the Integrated Services Directory |
| `StatWritesProcessed` | Total number of write-through commands processed |
| `StatWritesSucceeded` | Total number of write-through commands that succeeded |
| `StatWritesFailed` | Total number of write-through commands that failed |
| `StatWritesUnknownUser` | Total number of write-through commands that returned UNKNOWN USER |
| `StatInactiveUserReturns` | Total number of write-through commands that returned INACTIVE USER because the account's status was `deleted`, `suspended`, `locked`, or `maintenance` |

| | |
|---|---|
| `StatReadThrusFailed` | Total number of read-throughs that failed |
| `StatAuthReadThrus` | Number of read-throughs for queries which included a non-NULL password parameter |
| `StatBackupDbReadThrus` | Total number of read-throughs to the backup database |
| `StatAvgCacheHitPercent` | The percentage of requests for directory information that we satisfied by information already in memory. Events are tracked since the last time the server was started. (This statistic is calculated on each update interval, but reported on each “stat” interval.) |
| `StatCurCacheHitPercent` | The percentage of requests for directory information that we satisfied by information already in memory. (This statistic is calculated on each update interval, but reported on each “stat” interval.) |

# *Index*