# Message Archiving Using the AXS-One System

Messaging Server supports archiving through the AXS-One messaging archive system. Whether you require archiving for regulatory, compliance or legislative purposes, or you wish to manage the growth of your message store or reduce the storage costs, the AXS-One messaging archiving solution Compliance Platform can help them to achieve both, simultaneously. This Technical Note describes how to configure the Messaging Server to run with AXS-One. \

## Archiving Overview

A message archiving system saves all or some specified subset of incoming and outgoing messages on a system separate from Messaging Server. Sent, received, deleted, and moved messages can all be saved in an archive system. Archived messages cannot be modified or removed by email users, and a user's complete email record can be quickly retrieved.

In addition, messages in an archive system can be used in messaging server. For example, you can set up a system to archive messages over 2 years old. Instead of those messages residing in the message store, they would instead reside in a separate archive system. When the user views an archived message, he will see the same header and subject information, but instead of seeing the message he will see a URL called a *stub*. [Q: Can I get a screen shot of a message with a stub?] Clicking on the stub will display the message in a separate browser view window.

## Message Archiving Systems: Compliance and Operational

There are two types of archiving, compliance and operational. Compliance archiving is used when you have a legal obligation to maintain strict retrievable email record keeping. Selected email (selected by user(s), domain, channel, incoming, outgoing and so on) that comes into the MTA (after spam and virus filtering) and is copied to the archive system before being sent to the message store or the internet.

Operational archiving is used for performance or cost purposes. For example:

- To create more space on a message store by moving messages to a more efficient and scalable archiving system
- To recording and track email services even though it is not a legal requirement
- Use of single attachment storage. If an attachment is sent to multiple recipients, only a single copy is kept in the archiving system with pointers to that single copy.
- As another alternative for data back up.

## Accessing Archived Messages

The AXS-One system provides a GUI interface for retrieving messages in a variety of ways. Refer to the AXS-One literature for details. As mentioned earlier, the AXS-One archiving system can also be connected to Messaging Server such that it can provide URL stubs in place of the message content itself.

## Administrator's Perspective: Deploying, Maintaining, and Using the AXS-One Archiving System

This section describes the tasks and responsibilities for administrators deploying and maintaining the AXS-One system with Messaging Server.

[The idea here is to list all the tasks and responsibilities for an archiving system administrator.]

### Deploying AXS-One Archiving System

This section describes the tasks and responsibilities for deploying AXS-One.

- Installing and configuring the AXS-One system
- Configuring Messaging Server to work with AXS-One.
- [Q: Anything else?]

## Maintaining and Monitoring the AXS-One Archiving System

This section describes the tasks and responsibilities for deploying AXS-One. Maintaining the AXS-One system with Messaging Server consists of the following tasks and responsibilities:

- [Q: What are the tasks and responsibilities required for maintaining the AXS-One system with Messaging Server?]

## Using the AXS-One Archiving System

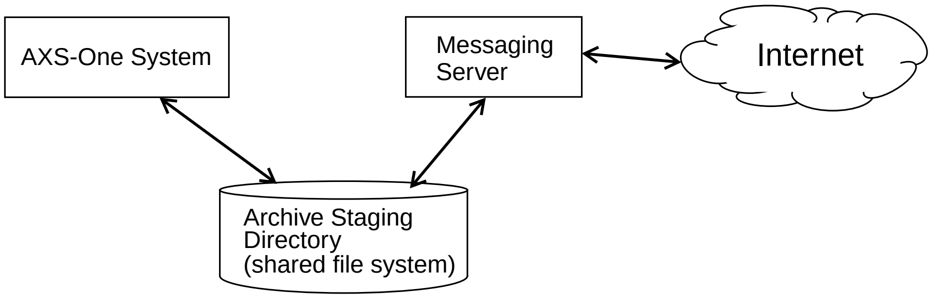
This section describes the tasks and responsibilities for using AXS-One archiving system. Using the AXS-One system with Messaging Server consists of the following tasks and responsibilities:

- Retrieving email records as required.
- [Q: Anything else?]
- [Q: Can we get a screen shot of the AXS-One interface? Or at the least link? The idea is to show what it looks like to use this system.]

# AXS-One Messaging Archiving Theory of Operations

The interface between the AXS-One archiving system and Messaging Server is a shared file system called the archive staging directory. This is shown in Figure 0-1.

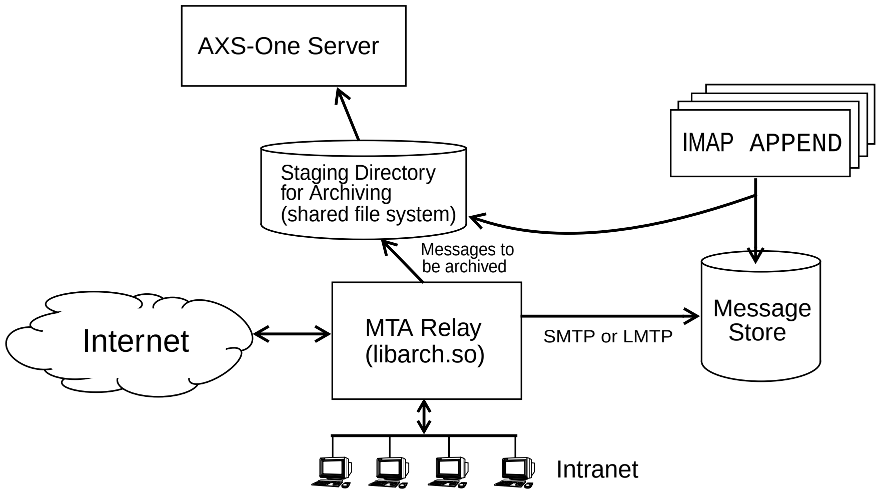
Figure 0-1 High-level View of AXS-One



All Messaging Server incoming, outgoing and moved (folder to folder) messages, are sent to a staging directory after virus and spam filtering. These messages are then moved into the AXS-One archiving system. Where they can be retrieved via an AXS-One client.

Depending on whether a compliance or operational archiving system is used, the architecture will be slightly different. Figure 0-2 shows a lower level view of a compliance architecture.

Figure 0-2 AXS-One Compliance Archiving, Low-level View



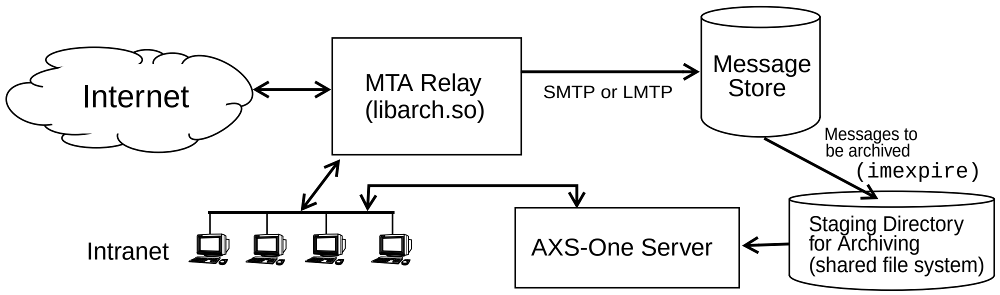
As shown in the previous figure, messages to be archived are copied to a staging directory where messages are moved into the AXS-One system at regular intervals. Because all internet and intranet messages pass through the MTA relay, this is where messages to be archived will be copied. This will happen after all spam and virus filtering occurs.

An AXS-One library file called `libarch.so` is used to implement the archiving functionality on the messaging server side. The archive stream, that is, the messages to be archived, is controlled by the Messaging Server spam filter interface (see [XREF to Spam filter chapter]). This means messages can be archived on a per user, per domain, channel, or per system basis. After messages are copied to the staging directory, they are then send to their destination in the message store or to the internet.

The arrow pointing from the IMAP Append function to the staging directory indicates messages that are moved from one folder to another. For example, let's say the spam filter moves messages it thinks are spam into a trash folder. This allows the user to review his trash folder and move any legitimate messages to the INBOX. These messages also need to get archived. [HAN: is this a correct example?]

Figure 0-2 shows a lower level view of an operational architecture.

**Figure 0-3** AXS-One Operational Archiving, Low-level View



In an operational archiving deployment, messages are archived from the message store instead of the MTA. In this case, the message stream is set by the `imexpire` command (see [ADD: xref to imexpire section]. This means messages can be archived by age, size, message count, message, and so on. Copies of archived messages are sent to the staging directory where they are archived by the AXS-One system. These archived messages can then appear in the user's mailbox as a URL stub. This is done using the `imarchive` command. When the user clicks on the stub, the message will come up in a separate browser viewing window.

## Performance Issues

Each message to be archived is split up and written as a small set of files in the staging directory where it is eventually picked up by the archive system. Once it has been placed in the archive, the files are deleted and the AXS-One server sends a message indicating the message has been archived and that Messaging Server can replace the message with a stub. The archive transfer directory should be placed in an appropriately sized file system for the expected archive message traffic.

[Q: What is the performance hit involved with adding archiving? Are there some workarounds or optimizations that would minimize the performance hit?

# Pre-Deployment Preparation

This section will help you determine your archiving needs. Once you have filled out the form defining these needs, an architecture and deployment implementation can be suggested.

[Q: How do customers determine their archiving needs inter terms of hardware, space, and so on? What questions must they answer? I'm suggesting that we provide an "Archiving Preparation Form" which will help them scope there needs. Is this idea a good approach? If so, what would be on this form?]

## Defining Your Archiving Needs

This section will guide you through filling out a form defining your archiving needs.

[Q: What are the questions that a customer needs to answer in order for us to recommend a particular architecture or deployment set up?]

## Choosing an Archiving Architecture and Deployment Implementation

[Q: Is there really such a choice? hardware? Architecture?]

# Setting Up A Compliance Archiving Deployment

The AXS-One Server must be installed and configured as per the AXS-One Server documentation. It should be installed on a separate server for best performance. Messaging server should be in full operation. Note that archiving can be done with an SMTP MMP proxy in front of the MTA host.

Compliance Archiving uses the MTA's spam filter interface to specify the message stream to be archived. Be sure to understand this interface before setting up a compliance archiving system. Complete documentation is at [ADD NEW X-REF. Temp: http://docs.sun.com/app/docs/doc/819-2650/6n4u4dtr8?a=view

[Q: Should we simply refer customers AXS-One docs for AXS-One configuration info? If so, where? What else do we need to tell customers to do?]

## Hardware and Software Requirements

[What hardware or software requirements are there? Any other preparatory tasks?]

## ▼ Configuring Messaging Server for Compliance Archiving

1. **Make sure the AXS-One Server has been installed and configured as per the AXS-One documentation.**

   [Q: Where is this? Do we need more info on installing AXS-One?]

2. **Set up a shared directory for messaging archival.**

   The staging directory should be a local NFS mounted drive.

3. **Ensure that the** `configutil` **parameters** `store.archive.compliance` **and** `store.archive.path` **are correctly set.**

   `store.archive.compliance` - Set to `ON` (default).

   `store.archive.operational` - Set to OFF

   `store.archive.path` - Set to directory shared by the AXS-One server and Messaging Server for messaging archiving. This should be the same as the `DIRECTORY` variable in the AXS-One configuration file.

4. **Edit `option.dat` file.**

   Include following parameters:

   ```
   SPAMFILTERx_LIBRARY=/opt/SUNWmsgsr/lib/libarch.so
   ```

5. **Where `libarch.so` is the AXS-One API for Messaging.**

   SPAMFILTER*x*_CONFIG_FILE=/opt/SUNWmsgsr/*filename*

   Where *filename* is an arbitrarily named text file detailing the AXS-One archiving information.

   In the above parameters, *x* is a number from 0 to 3 specifying the filtering software, in this case AXS-One.

6. **Create the AXS-One configuration file. (Called *filename* in the previous step).**

   This file must be readable by the `mailsrv` user. It must include the following two entries:

   ```
   STYLE=1
   DIRECTORY=<Message_Archival_Staging_Area>
   ```

   `STYLE` specifies the AXS-One configuration. At this time the only value is 1.

   `DIRECTORY` specifies the directory shared by the AXS-One server and Messaging Server for messaging archiving.

7. **Specify the messages to be archived.**

   Messages can be archived by user(s), domain, channel, incoming or outgoing mail. Use the spam filter interface to specify precisely the message stream you wish to archive. This is described in detail at [ADD NEW X-REF. Temp: http://docs.sun.com/app/docs/doc/819-2650/6n4u4dtr8?a=view

   Below are some examples

   a. **To archive all incoming and outgoing messages for users, specify an user opt-in attribute using `LDAP_OPTIN`*x*. For example:**

      ```
      LDAP_OPTIN1=AXS-One
      ```

      Now add the opt-in attribute-value pair to the LDAP user entry of any user whose mail you wish to archive. For example:

      ```
      AXS-One: archive
      ```

   b. **To archive all incoming and outgoing messages for a particular domain, specify a domain opt-in attribute using `LDAP_DOMAIN_ATTR_OPTIN`*x*. For example:**

      ```
      LDAP_OPTIN1=AXS-One
      ```

      Now add the opt-in attribute-value pair to the LDAP domain entry of the domains whose mail you wish to archive. For example:

      ```
      AXS-One: archive
      ```

c. **To archive all incoming messages to Messaging Server passing through MTAs, add the** `sourcespamfiler`*X*`opt` **value pair to the tcp_local channel block. For example:**

```
tcp_local smtp mx single_sys remotehost inner switchchannel \
identnonelimited subdirs 20 maxjobs 7 pool SMTP_POOL \
maytlsserver maysaslserver saslswitchchannel tcp_auth \
sourcespamfilter1optin archive
tcp-daemon
```

d. **To archive all incoming and outgoing messages to Messaging Server, add the** `sourcespamfilter`*X*`opt` **and** `destinationspamfilter`*X*`optin` **value pairs to the tcp_local channel block. For example:**

```
tcp_local smtp mx single_sys remotehost inner switchchannel \
identnonelimited subdirs 20 maxjobs 7 pool SMTP_POOL \
maytlsserver maysaslserver saslswitchchannel tcp_auth \
sourcespamfilter1optin archive destinationspamfilter1optin
archive
tcp-daemon
```

8. **Compile the MTA configuration with** `imsimta cnbuild` **followed by** `imsimta restart`

## ▼ Testing Your Compliance Archiving Deployment

[How do you do this?]

## Administration and Maintenance of Your Compliance Archiving Deployment

[How do you do this?]

## Troubleshooting Your Compliance Archiving Deployment

[How do you do this?]

# Setting Up An Operational Deployment

Setting up operational archiving consists of two primary steps. In the first step you define what messages are to be archived using the `imexpire` command. The second step involves specifying how to do deal with the messages in the message store after archiving has occurred. That is, should the messages be replaced with a stub or not. This is specified using the `imarchive` command.

Besides the two steps described above, operational archiving is set up in much the same as compliance archiving. The primary difference is that instead of the message archive stream being defined with the MTA spam filter interface, it is defined using the `imexpire` command. Be sure to understand this interface before setting up a compliance archiving system. Complete documentation is at [ADD NEW X-REF. Temp: http://docs.sun.com/app/docs/doc/819-2650/6n4u4dttr?a=view

## Hardware & Software requirements

[What hardware or software requirements are there? Any other preparatory tasks?]

## ▼ Configuring the Messaging Server

1. **Make sure the AXS-One Server has been installed and configured as per the AXS-One Server Manual.**

   [Q: Where is this? Do we need more info on installing AXS-One?]

2. **Set up a shared directory for messaging archival.**

   The staging directory should be a local NFS mounted drive.

3. **Ensure that the message store `configutil` parameters are correctly set.**

   `store.archive.compliance` - Enables compliance archiving. Set to `OFF`.

   `store.archive.operational` - Enables operational archiving. Set to `ON`.

   `store.msghash.enable` - Enables message hash indexing. Must be set to `ON` for operational archiving. [Han: Is this needed or must it be turned off for compliance?]

   `store.archive.path` - This is the directory shared by the AXS-One server and Messaging Server to pass messaging files for archiving. This should be the directory defined in the earlier step.

   `store.archive.reportdir` - This is the directory used by the AXS-One server to pass archiving reports back to Messaging Server. These reports are used by the imarchive command to determine when it can replace the message on an email with a archive stub. This directory is specified on the AXS-One side and this parameter must match that value.

   `store.archive.retrieveserver` - This is the AXS-One server that passes message archive stubs for Messaging Server to put in messages. This is specified on the AXS-One side. [Han: Is this a server name or directory or what?]

4. **Specify the message stream to be archived.**

   Operational archiving can archive messages by folder, user, domain, number of messages in mailbox, age of messages and so on. Use the `imexpire` interface (see http://docs.sun.com/app/docs/doc/819-2650/6n4u4dttr?a=view) to define the messages to be archived. The last rule of a rule set in `store.expirerule` should be the following:

   ```
   action: archive
   ```

5. **Use `imarchive` to specify what to do with archived messages in the message store.**

   After the AXS-One server archives an email message, it sends a log message to the archive report directory. At this point, the email message is safety achieved and the email message on the message store can be replaced with a URL stub. This is done by running `imarchive` at regularly scheduled intervals. Automatic task scheduling is achieved using the `local.schedule.`*taskname* parameter (see http://docs.sun.com/app/docs/doc/819-2650/6n4u4dtnj?a=view).

   In this example, `imarchive` is run at 12:30am, 8:30am, and 4:30pm:

   ```
   configutil -o local.schedule.rm_archived_msgs -v "30 0,8,16 * *
   *" /opt/SUNWmsgsr/sbin/imarchive -s -d -v
   ```

   `-s` specifies that a URL stub is placed in the email message in the message store.

   `-d` specifies that the email message in the message store should be deleted.

   `-v` specifies the verbose mode for returning log messages.

   [Han: How often should imarchive be run? Any guidelines? Also, are there any more commands or tasks that need to be run or is this it?]

## Testing your deployment

[How do you do this?]

## Administration and Maintenance

[How do you do this?]

## Troubleshooting

[How do you do this?]