In this chapter:

- *History of Cyrus*
- *Cyrus Concepts and Features*
- *Cyrus Server Configuration*
- *The Future of Cyrus*
- *Strengths and Weaknesses of Cyrus*
- *When Is Cyrus the Right Choice?*

6

Introduction to the Cyrus IMAP Server

This chapter provides a technical overview of the features and design concepts that make up the Cyrus IMAP server. The Cyrus server wasn't the first IMAP server, nor was it written by the author of the IMAP standard. True believers in the Cyrus server and its derivatives are likely to tell you, though, that once you've deployed the Cyrus server, you're unlikely to switch to another server. Cyrus boasts a very attractive feature set. It offers robust administration, scalability, and leading-edge IMAP extensions, such as mailbox quotas, plug-in authentication mechanisms, and support for server-side filtering. Detractors are likely to say that UW can be deployed more easily and that Cyrus may not make much sense for a small or medium-size user base.

The Cyrus IMAP server is based on IETF standard protocols, including IMAP4, IMSP, SMTP, RFC 822, MIME, and SASL. IMAP2bis is also supported for backward compatibility with earlier IMAP clients. POP3 is included to support POP users while they're going through the process of selecting an IMAP client.

The Cyrus server is feature-rich, and implements several IMAP protocol extensions, including the IMAP QUOTA and the IMAP ACL extensions. Cyrus is normally run as a "black box" server—users are not meant to access the system by any means other than the IMAP protocol. Mailboxes are stored in a central location and in parts of the filesystem that are private to the Cyrus server. Cyrus was designed with centralized mail storage in mind to make the system easier for administrators to manage. The centralized mailstore also allows such features as mailstore partitioning (discussed later in this chapter). The ability to partition the mailstore makes the server very scalable—as the mailstore grows, the system administrator may add new partitions to the mailstore to accommodate new growth, without affecting the operation of the server. Cyrus shared folders are very

flexible—the server not only allows concurrent read connections to the same mailbox, but also supports concurrent write connections.

The Cyrus server differs from other IMAP server implementations in that it is meant to run on sealed, or black box, servers, where normal users are not permitted to log in. Users access their mail strictly via IMAP. Even if a user were allowed to log on to the server where Cyrus runs, she would not be able to do things like "grep" her mail folders for someone's username or compress her folders. That is because the Cyrus mailstore is private to the Cyrus IMAP system. The private mailbox design can be difficult for users to get used to, especially if in the past they've logged in to a Unix shell account and had direct access to mail. Cyrus's black box design, however, gives the server large advantages in efficiency, scalability, and ease of administration.

Cyrus did not turn out to be fantastically scalable by accident. An original goal of Project Cyrus was to produce a mail system that scaled to tens of thousands of active mail readers.* The number of large† sites using Cyrus (Carnegie Mellon University, the University of North Texas, North Carolina State University, the University of Florida, Ohio University, the University of Otago in New Zealand, for example) are examples of Cyrus's scalability. There are even three IMAP server products released commercially—by Netscape, Mirapoint, and MessagingDirect—which are built on the original Cyrus distribution. Although Cyrus is suitable for large organizations, it can be used equally well in small organizations. Although not as simple to deploy as the UW server, it is capable of being installed, configured, and maintained by small computing departments.

History of Cyrus

The Cyrus IMAP server traces its roots back to CMU's Andrew Messaging System (AMS). AMS, the world's first large-scale multimedia mail and bulletin board system, was a very successful proprietary system developed and used at CMU to support the electronic communication needs of the university community. Project Cyrus began in 1992 in an effort to develop a replacement for AMS, which by then was reaching its end-of-life. The first Cyrus IMAP4 server was released in late 1995, one year after IMAP4 was approved as a Proposed Internet Standard. The Cyrus server retained many of the same features that made AMS a success.

---

* Project Cyrus was named after Cyrus the Great of Persia (599–530 BC), who initiated one of the first known postal systems.

† A large site is a site with more than 20,000 active users.

AMS used a file access protocol, Andrew File System (AFS) to access mail. By the time AMS comprised 9,000 users, accessing mail from a mix of Macintosh, PC, and Unix machines, it was painfully clear that a mail system using a filesystem protocol for mail access does not scale well. AFS was not implemented for Macs and PCs. To make mail stored on the AFS servers available to those machines, it had to be served from a "translator" machine. The "translator" was an AFS client set up as a server that acted as sort of a middleman and made mail available to the Macs and PCs. As it turned out, limitations in the AFS client resulted in more translator machines than there were actual AFS servers serving up mail. If that wasn't bad enough, the translator machines were plagued with performance problems. A goal of Project Cyrus was to serve mail using a network protocol that was explicitly designed for mail—IMAP.

An important feature of AMS was its integration of email and bulletin boards. AMS supported controlled access to bulletin boards by allowing a system administrator to apply an access control list to the bulletin board. AMS also provided an addressing scheme that allowed authorized users to post messages directly to bulletin boards via email. Bulletin boards were an essential feature of AMS and were widely used by the user community at CMU. Project Cyrus continued support for bulletin boards, including all the bulletin board features offered in AMS, in the Cyrus IMAP server. AMS, however, did not allow users to make their personal mail folders available to other users. The lack of support for shared personal folders was problematic. There were individuals and small groups who wished to set up private forums, but could not do so without administrative intervention. Project Cyrus addressed the limitation by extending bulletin board support to allow users to share their own folders with other users on the system.

Kerberos was the authentication method used at CMU at the time that AMS was in production. Even in the early 1990s, there was a movement within the IETF to discourage network protocols that relied on cleartext passwords from making it through the standards process. Kerberos support in the Cyrus server (and in the SASL RFC) is a carryover from AMS. Challenge-Response Authentication Mechanism (CRAM) was added to accommodate sites that need encrypted authentication but do not have a Kerberos framework in place. The latest versions of Cyrus support Simple Authentication and Security Layer (SASL), which is a way of adding authentication support to IMAP and other connection-based protocols.

AMS supported MIME, which was a standard that was quickly becoming the accepted way of interchanging multimedia documents via email. The project Cyrus developers included MIME support in the Cyrus server for its growing popularity, but also because it would make for a smooth transition from AMS to Cyrus.

A Late-Breaking Note

Cyrus IMAP 2.0 was in the pre-release stage when we went to press. Cyrus 2.0 represents a major architectural overhaul over previous versions. One of the largest changes is that it no longer runs out of a separate "superdaemon" like *inetd*. The Cyrus IMAP server now runs as a standalone daemon. Here's an overview of the new features in Cyrus 2.0:

- **Improved mailboxes file structure**. The mailboxes file is now a Berkeley-style transaction-protected database. This greatly streamlines the contention between different *imapd* processes for read/write access to the mailboxes file and, by extension, streamlines overall operation.
- **Support for LMTP**. Delivery to the mailboxes from the MTA is now accomplished with Local Mail Transport Protocol (LMTP). LMTP can be used either from one process to another on the same host with Unix sockets, or via Internet sockets. By using LMTP for delivery, an MTA can receive a reliable confirmation or denial of the success of delivery to each local mailbox. This often wasn't possible with other delivery methods. Because this same feature could be used to confirm or deny the existence of specific mailboxes, some system administrators highly restrict the access to LMTP over Internet sockets, especially from remote hosts. The Cyrus *deliver* program is now simply a wrapper to the LMTP delivery agent.
- **Clustering support**. Cyrus Murder: The IMAP Aggregator, provides support for IMAP server clustering.
- **Sieve support**. Server-side filtering through Sieve is enhanced by the addition of regular expression, notification, and IMAP flag features.
- **Simple Network Management Protocol hooks**. You can now roll your Cyrus server into your SNMP managed network scheme. Cyrus now comes with an SNMP agent daemon called *tug-o-war*.
- **Administration via Perl**. Brandon Allbery's Perl-based version of *cyradm* is now included with Cyrus IMAP. Eventually, it will replace the Tcl-based version.
- **MULTIAPPEND extension**. The MULTIAPPEND extension is implemented. Clients may now append an arbitrary number of messages to a mailbox in a single atomic action.
- **LIBWRAP**. Since it's not possible to launch the server(s) out of *inetd*, *libwrap*, which comes with TCP Wrappers, is presented as an alternative way to do IP-based access control.
- **Hierarchical RENAME**. The IMAP RENAME command now works hierarchically.

Cyrus Concepts and Features

This section explains the basic concepts of the Cyrus server design and discusses Cyrus's feature set.

The Cyrus Mailbox Namespace

The Cyrus IMAP server uses a hierarchical mailbox naming convention similar to the naming scheme used in Usenet for naming newsgroups. The mailbox namespace is described in this section from the server's point-of-view. As we will see later in the section, the mailbox names that the user sees in his IMAP client are not necessarily what the system administrator sees when she lists the mailbox names on the server.

Let's take a look at a mailbox hierarchy. Our mailbox belongs to a user whose username is *johndoe* and contains a mailbox for incoming mail and three folders* named *drafts*, *saved-messages*, and *sent-mail*. On a Cyrus server, the incoming mailbox and the three folders would be named:

user.johndoe
user.johndoe.drafts
user.johndoe.saved-messages
user.johndoe.sent-mail

There are several restrictions on mailbox names. Only ASCII characters are allowed; shell metacharacters, /, a leading or trailing period, and two periods in a row are not allowed. Additionally, mailbox names are case-sensitive.

A Word on a Special IMAP Mailbox: INBOX

Although no two mailbox names are the same from the Cyrus server's point of view, that is not necessarily true from the point of view of the IMAP client. RFC 2060 (IMAP) allows for a special mailbox name, *INBOX*. *INBOX* is case-insensitive and refers to the authenticated user's primary mailbox on the server. When an IMAP client communicates with a Cyrus IMAP server, the user's top-level mailbox may be referred to as either *user.username* or as the name *INBOX*. When configuring an IMAP client, it is more common to define the incoming mailbox or incoming mail folder as *INBOX*, and in fact, many clients come preconfigured that way.

* Mailboxes other than the incoming mailbox are sometimes also referred to as "folders."

The Cyrus Mailstore

On a Cyrus IMAP server, a filesystem is allocated for the sole purpose of mail storage (under ideal circumstances, this filesystem also resides on its own physical media, but we'll discuss that in a later chapter). Each user's mailbox is represented by a directory on the mailstore's filesystem. Cyrus stores mail in a format that is similar to MH format—instead of storing many messages in a single file as in standard Unix mail (sometimes known as *mbox* or *Berkeley format*), each message is stored in an individual file. Each individual file is in RFC 822 format, like Berkeley mail, the only difference being that the Cyrus mail file terminates each line with CRLF.*

Each user has his or her own subdirectory in the Cyrus mailstore directory. The subdirectory and all files within it are owned by the Cyrus user. As mentioned earlier in this chapter, the mailstore is private to the owner of the Cyrus server (usually user *cyrus*), and all other users are restricted from direct access to the mailstore. Incoming mail is written to the user's top-level directory, one message per file. Folders consist of subdirectories below the user's top-level directory. Like the top-level directory, the folder contains individual files, each of which contains a single mail message. Both the top-level directory (incoming mail) and subdirectories (folders) are referred to as "mailboxes." The filename of each mail message within a mailbox is a numeral† appended with a period. The numerals are strictly ascending in the order that the message was added to the mailbox.

Each mailbox contains four additional binary files that the Cyrus server uses to manage the mailbox on a per-session basis:

cyrus.index
: This file contains a header of information that applies to the whole mailbox, and one fixed-length record per message. The information stored in the header includes the number of times the mailbox has been expunged (i.e., messages marked for deletion have been permanently removed); the unique identifier (UID) of the last message that was added to the mailbox; the number of messages in the mailbox; the mailbox's quota usage; and the time a message was last inserted into the mailbox. The per-message records contain the UID of the message, the Date: header, the date of last modification, and the size of the mail message.

cyrus.header
: This file contains a magic number, the name of the mailbox's quota root, a copy of the mailbox's ACL, and the names of user-defined flags.

---

* This is important to keep in mind when converting other mailbox formats to Cyrus mailboxes.

† The numbers are mapped to UIDs in the *cyrus.index* file. Although they are strictly ascending, they are not necessarily contiguous.

cyrus.cache
: This file is a performance enhancement. It contains cached headers of every message that the client requests, so that the server doesn't have to go back and peek in the message for the headers. The file contains a counter that keeps track of the number of times *cyrus.cache* has been rewritten by an expunge operation. Each record in *cyrus.cache* contains an IMAP envelope; an IMAP body structure and body; and the size, offsets, character sets, and encodings of the MIME sections of the message. The IMAP envelope, body, and body structure are in a format that the IMAP FETCH command can use.

cyrus.seen
: This file contains the username, the time the mailbox was last opened, the sequence of numbers of messages that have been seen, and the message number of the last message that is not recent.*

One caveat to keep in mind is that, once mail is stored in Cyrus format, it can be challenging to extract the mail and revert to the standard Unix mail format. Scripts to tackle this task are provided in Chapter 9, *Cyrus System Administration*.

Cyrus Features

Cyrus IMAP server is feature rich and implements several protocol extensions, including Quota (RFC 2087) and ACL (RFC 2086). In this section we'll talk about how Cyrus implements those extensions. We'll also look at some implementation-dependent "bonus" features that are derived from the Cyrus server design, such as mailstore partitioning and shared folders.

Access control

Permission to access a mailbox is defined in an Access Control List (ACL) on the Cyrus server. An ACL specifies the users or groups who have permission to access the mailbox. Each mailbox on a Cyrus server has an associated ACL. The ACL contains one or more entries that consist of a user or group that the ACL applies to and a list of access rights. Two special groups, *anonymous* and *anyone*, are defined on every Cyrus server by default. The *anonymous* group refers to all unauthenticated users and is most commonly used to provide public access to a shared folder. CMU provides access to their Cyrus Mailing List archives in this manner. The *anyone* group is made up of all users, including the *anonymous* users.

---

* A recent message is one that has recently arrived in the mailbox. The current session is the first IMAP session to have been notified about the message.

ACLs are discussed in detail in Chapter 9. ACLs are very flexible and allow a fine granularity of control over access to a mailbox. It is possible using ACLs to allow different levels of access to different users on the same mailbox.

Shared folders and bulletin boards

Cyrus allows users to share private mail folders with other users on the system without requiring the intervention of a system administrator or any person with special privileges. Shared folders are simply mailboxes with ACLs that allow more than one user to access the mailbox via an IMAP client. Cyrus also supports multiple concurrent read/write connections by different users to the same shared folder.

Bulletin boards are shared mail folders that have the specific feature of allowing many users not only to read, but also to post messages directly to the folder. Users post a message to the board by mailing a message to the submission address of the bulletin board. Bulletin boards are reminiscent of Usenet newsgroups. Many sites manage their own Usenet news server specifically so that the site can provide site-specific, private news groups to their users. For those sites, Cyrus bulletin boards could replace the site-specific newsgroups and make it possible to outsource the public newsgroups. The details of setting up shared folders and bulletin boards are covered in Chapter 9.

Mailstore partitioning

The Cyrus server supports distribution of the mailstore across physical disks. Each mailbox hierarchy is known as a partition. Mailstore partitioning allows the mailstore to be expanded as needed. If the mailstore approaches the capacity of the filesystem it resides on, the system administrator has the option of expanding the mailstore onto another filesystem by adding a new Cyrus partition to that filesystem. He would then either move a subset of existing mailboxes to the new partition or perhaps create any new mailboxes on the new partition.

A mailbox is assigned to a partition at the time the mailbox is either created or renamed (moved). Although a user's hierarchy of mailboxes is usually kept together on one partition, a mailbox need not reside in the same partition as its parent mailbox.* As an example, *user.johndoe* and *user.johndoe.sent-mail* may reside on separate partitions. Mailstore partitioning is invisible to the end user. How to partition the mailstore and assign mailboxes to partitions is described in Chapter 9.

---

* The authors recommend keeping an individual user's mailbox hierarchy contained on one partition for ease of management unless there is a compelling reason not to.

Storage quotas

Storage quotas allow the system administrator to set limits on mail storage for both incoming mail and mail stored in folders. Cyrus IMAP storage quotas are not related to Unix quotas and apply to the Cyrus server only. Cyrus quotas do not apply to a specific mailstore partition—a quota can apply to more than one mailbox on more than one mailstore partition. Mailbox index and cache files are not counted against the quota—only mail message files count.

Quotas on the Cyrus server are hierarchical and very flexible. Quotas can be set at any level of a mailbox hierarchy. The level at which the quota is set is known as a *quota root*. A mailbox hierarchy may have more than one quota root at any level of the hierarchy. The quota set on a quota root applies to the mailbox at that level and all mailboxes below it. As an example, recall that *johndoe* has four mailboxes in his hierarchy:

user.johndoe
user.johndoe.drafts
user.johndoe.saved-messages
user.johndoe.sent-mail

The following quota roots have been set on *johndoe*'s mailboxes:

user.johndoe
user.johndoe.saved-messages

The quota root *user.johndoe* applies to the *user.johndoe*, *user.johndoe.drafts*, and *user.johndoe.sent-mail* mailboxes. The quota root *user.johndoe.saved-messages* applies only to the mailbox *user.johndoe.saved-messages*. If the quota root on *user.johndoe.saved-messages* didn't exist, then the quota root on *user.johndoe* would cover *user.johndoe.saved-messages*.

Because quotas on the Cyrus server are hierarchical, it's possible to set quotas on folders without setting a quota on incoming mail. However, it is not possible to do the reverse and set quotas on incoming mail without implicitly setting quotas on folders. This is so because of the hierarchical nature of mailboxes on the Cyrus server. The quota applies to a mailbox and all mailboxes below it in the hierarchy, unless quotas are explicitly applied to mailboxes lower in the hierarchy.

Quota limits and usage are stored in a separate file per user in the Cyrus configuration directory, described later in this chapter.

Quota warnings. Cyrus provides a means of specifying a soft quota limit by setting the `quotawarn` option in */etc/imapd.conf*. The value of `quotawarn` is expressed as a percent usage of the quota. When a user selects a mailbox, the

server will send an ALERT warning the user that she's exceeded the soft limit if two conditions are met:

1. The user's usage in the selected mailbox exceeds the `quotawarn` threshold.
2. The user has delete rights on the selected mailbox.

Although many IMAP clients claim to be fully IMAP4 compliant, the truth of the matter is that many of them do not handle quotas cleanly. For example, in some clients, if mailbox usage exceeds the soft limit while an IMAP session is open, the client will not handle the alert message when the mailbox is selected. In that case the IMAP session must be closed and reopened and the mailbox selected before the user will see the alert.

Quotas and mail delivery. When mail storage in a given mailbox exceeds the quota limit, mail delivery fails. Delivery attempts to the mailbox in question are deferred and handled in whatever way you choose to make the MTA handle it. Usually, such mail is kept in a local queue and delivery is re-attempted with each queue run for three days or so before it's either delivered successfully, or bounced back to the sender with an error.

If the usage is over the soft limit but has not reached the hard quota limit, however, a message of any size will be delivered, regardless of whether or not it would result in usage that exceeds the hard limit. One may argue that this behavior would open the doors to denial of service (DOS) attacks, but that is not a strong argument against the design. DOS attacks can be prevented by setting a limit on the size of messages in your MTA configuration.* Besides, on the traditional Unix mail systems that use */var/spool/mail* or */var/mail* for incoming mail, system administrators rarely, if ever, set UFS quotas on the mail spool partition. It is considered poor practice because when a user's usage exceeds the quota, mail bounces without the user ever knowing that she is unable to receive mail. Cyrus IMAP quotas are a different story. The Cyrus server alerts the user when usage approaches or exceeds the quota limit. By allowing one message to be delivered to the mailbox and bringing the user over quota, the effect is that the next time the user selects the mailbox, she receives the alert immediately and can begin to correct the problem.

* Generally, 10 to 15 MB is considered an adequate maximum message size limit. The greatest benefit of setting such a limit is to disable denial-of-service attacks originating from software that's designed to flood your server with an endless stream of content for a single message until the server fills a filesystem and croaks. Good MTA tuning will prevent such attempts before the message is ever handed off to Cyrus to deliver.

Usenet news integration

The Cyrus server supports exporting Usenet newsgroups as mailboxes. The Cyrus distribution includes programs that interface with an *inn* server to collect news articles and add pointers to them to the appropriate mailboxes on the IMAP server, to remove pointers to expired and canceled articles from the IMAP mailboxes, and to synchronize the IMAP news mailboxes with the news active file. Version 2.0 has much easier news integration than earlier versions of Cyrus. News is discussed in further detail in Chapter 9.

Authentication

With Cyrus, your users need never send their passwords over the network in cleartext. Cyrus natively supports Kerberos 4. The latest version releases of Cyrus (1.6 and higher) include support for the SASL library. SASL makes it possible to authenticate your users with any of the additional encrypted authentication methods supported by SASL, including CRAM-MD5, DIGEST-MD5, and GSSAPI (MIT Kerberos 5). However, not all IMAP clients support Kerberos.

Cyrus IMAP also supports Unix authentication. Unix authentication permits a user to log in to the server using her plaintext Unix password, provided that she appears in */etc/passwd* (or is a member of a netgroup that is included in */etc/passwd*). Unfortunately Cyrus does not provide a way of supporting both Kerberos and plaintext login on the server side—it's either one or the other.

Anonymous login is supported regardless of which authentication method is used, as long as `allowanonymouslogin` is defined in */etc/imapd.conf*.

Many sites that do not implement Kerberos would still prefer not to send plaintext passwords over the network. For these sites, SSL is an option. The Cyrus IMAP server does not support SSL, but SSL support can be added to Cyrus by wrapping *imapd* with the free SSLeay library and *stunnel* and using clients (web browsers, for example) that support SSL. *stunnel* creates an SSL pipe between the web browser and *stunnel* server over the network, then sends the password in plaintext over the loopback interface to *imapd*. The 2.0 release of Cyrus is SSL enabled—it has built-in SSL-enabled IMAP and POP daemons that make an SSL wrapper like *stunnel* unnecessary. For sites using an older version of the Cyrus server, instruction on how to add SSL support to IMAP is provided in Appendix B, *Adding SSL Support to IMAP*.

With the growing popularity of LDAP, many sites that implement both IMAP and LDAP may be interested in using LDAP as an authentication mechanism. Although LDAP authentication is not supported officially in the Cyrus release, there is at least one freely available drop-in LDAP authentication module for the Cyrus server.

The module replaces Cyrus's external authentication daemon (*pwcheck*) with another daemon (*pwcheck_ldap*) that uses an LDAP directory for authentication.*

CMU has also developed an implementation of RFC 2222 (Simple Authentication and Security Layer), the Cyrus (SASL) API, which is integrated into Cyrus releases later than 1.6. The SASL API can be used to provide authentication from either the client or server side. Cyrus SASL includes support for anonymous, DIGEST-MD5, CRAM-MD5, Kerberos Version 4, MIT Kerberos Version 5 or Heimdal Kerberos Version 5, and plaintext.

Selecting an authentication method is beyond the scope of this book, but instructions on how to set up Cyrus to use the both the supported and not-yet-supported authentication methods are outlined in Chapter 8, *Configuring the Cyrus Server*, and Chapter 9.

Cyrus Server Configuration

The parts that make up the server configuration, what each part does, and where to find each part, are described in this chapter. Details on setting up the configuration are given in Chapter 9.

The imapd Server Configuration File

The *imapd* daemon reads its configuration options from the file */etc/imapd.conf*. The location of the configuration file itself is not configurable. Examples of the options specified in this file include:

`umask`
: The default permissions used by the various Cyrus programs

`allowanonymouslogin`
: Whether to allow anonymous login to the server

`configdirectory`
: The location of the Cyrus server configuration directory

`quotawarn`
: The threshold for over quota warning messages

The `plaintextloginpause` option sets the number of seconds to pause after a successful plaintext login. This option allows users to perceive the cost of using plaintext passwords on systems that support strong authentication. The complete list of options is discussed in Chapter 9 and listed in the *imapd*(5) online manual page.

---

* The source and installation instructions for *pwcheck_ldap* are available from *http://www.linc-dev.com/*.

Server Configuration Directory

The server configuration directory contains databases that describe the server's dynamic configuration in terms of content, status, processes, quotas, and mailbox subscriptions. By default, the server configuration is installed in */var/imap*.

Special files

configdirectory/msg/shutdown
: If *configdirectory/msg/shutdown* exists, then *imapd* will send the first line in the file to the client in the form of an IMAP ALERT* message and shut down the connection. New connections will be denied until the file is removed.

configdirectory/msg/motd
: If *configdirectory/msg/motd* exists under the *config* directory, *imapd* sends the first line in the file to the client as an ALERT message when a connection is made. Many clients handle ALERTs poorly (by connecting more frequently than is necessary and thus displaying the alert for each connection and annoying the user) or not at all.

Mailboxes file

configdirectory/mailboxes is a text file that lists every mailbox name, the mailbox's quota root, and the mailbox's ACL. *mailboxes* is critical to the operation of the Cyrus server. In releases 1.6.22 and earlier, the *mailboxes* file is a plaintext file. The 2.0 release replaces the text file with a transactional Berkeley DB database file. The *mailboxes* file is updated frequently. It can also grow very large as the number of users on the system increases. CMU warns that the *mailboxes* file should never reside on an NFS-mounted partition due to locking problems with NFS.

In Cyrus 1.6.22 and earlier, although the Cyrus system is well-engineered and has proven to be robust, the *mailboxes* file is perhaps its weak link. If the file is corrupted, the road to recovery is not a simple matter of restoring the file from tape. A very recent copy of *mailboxes* is required to avoid a long and laborious reconstruction process. Chapter 9 describes that recovery process in detail. In practice, the text format *mailboxes* file is not necessarily a bad thing—our personal experience has shown that even on large systems (30,000+ users), the *mailboxes* file does not create any noticeable degradation in performance. Cyrus Version 2.0 replaces the *mailboxes* file with a Berkeley DB database that has transactional updates.

---

* ALERT is a server status response that IMAP-compliant clients are required to display to the user in a way that calls the user's attention to the response.

Logging and process information

Cyrus IMAP logs messages to the *local6* facility of *syslog* with severity levels of *crit*, *err*, *warning*, *notice* and *info*. Errors requiring the system administrator's attention are logged to *crit*. I/O errors, including the specific file and Unix error, are logged with severity level *err*. Client timeouts and authentication failures are logged with severity level *warning*. Successful and unsuccessful authentications are logged with severity level *notice*. Mailbox openings and closings and suppression of duplicate deliveries are logged with severity level *info*.

The server also supports per-user telemetry logging. Telemetry logging is very useful in troubleshooting problems, especially in determining whether a bug resides in the server or in the client. Telemetry logging becomes active for a user when the system administrator creates a subdirectory with the user's name under the *configdirectory/log*. The server keeps a protocol level log of sessions that are authenticated as the user in a file named after the server process identification number. The Cyrus user must have write access to *configdirectory/log*. Files created in that directory match the PID of the IMAP process.

A file per active server process is stored in the *configdirectory/proc* subdirectory under the configuration directory. The name of the file is the process identification number of the process. Each file contains the client's hostname; the username, if the user is logged in; and, if a mailbox is selected, the mailbox name.

Quota root

Each quota root has a corresponding file, named after the quota root, under *configdirectory/quota*. The quota root file contains the quota limit in kilobytes and the current usage in bytes. For example, user *johndoe* has a quota limit of 20 MB on his top-level mailbox (*user.johndoe*) with a usage of 13,194 bytes. *johndoe* also has a quota limit of 10 MB one of his folders, *user.johndoe.sent-mail*, and has a current usage of 4,336 bytes. To use the Cyrus terminology, *johndoe* has two *quota roots*, and those quota roots are *user.johndoe* and *user.johndoe.sent-mail*. If we list the contents of the directory *configdirectory/quota*, we see two files named after *johndoe*'s quota roots:

```
$ ls /var/imap/quota | grep johndoe
user.johndoe       user.johndoe.sent-mail
```

This is what we see inside each of the files:

```
$ cat user.johndoe
13194
20480
$ cat user.johndoe.sent-mail
4336
10240
```

The delivered database

When the Cyrus *deliver* program is invoked with duplicate delivery suppression turned on, Cyrus keeps a database of delivered messages, used to suppress duplicate deliveries. The database contains the mailbox name, the message-ID, and the delivery time. Versions through 1.5.24 store the information in DBM format, in *configdirectory/delivered.dir* and *configdirectory/delivered.pag*. Versions 1.6 and up store a database file for each letter of the alphabet. File locking is accomplished by the *configdirectory/deliver.lock* file, which the server locks before updating the DBM database.

Mailbox subscriptions

Mailbox subscriptions are kept under *configdirectory/user*. Each user on the system has a file named *username.sub*. The file contains a list of the mailboxes the user is subscribed to. When migrating from another mail system to Cyrus, it may be possible to collect subscription information, parse it, and write it directly to the users' subscription files before bringing Cyrus up.

The Future of Cyrus

Cyrus is a popular IMAP server. The Cyrus server distribution is downloaded over 3,500 times a month, and there are dozens of sites running Cyrus on a large scale. Three commercial IMAP solutions are based on Cyrus. There is a very active Cyrus mailing list that uses Cyrus as a context for discussing implementation of enterprise messaging systems. Cyrus is here to stay.

Version 2.0a is the latest release. Version 1.6.22 is still considered the stable version at the time of this writing, but a large number of sites still run Version 1.5.19 in their production environments.

Versions of Cyrus later than 1.5.19 have begun to move away from cleartext authentication to methods supported by SASL. Project Cyrus needed to offer more authentication methods for people to use. It's possible to configure Cyrus 1.6.22 to work with cleartext authentication, but it requires some extra work (e.g., you must either configure SASL or PAM appropriately or change the ownership on your shadow password file). The introduction of completely new code into the previously comfortingly consistent Cyrus code base has turned a population of easy-going mail system administrators into a curmudgeon collective.

Many sites have held back from upgrading to newer versions of Cyrus because changing a site's authentication mechanism has serious implications to the infrastructure. Sites that are not ready to replace shadow password authentication with another mechanism may not want to open a security hole by changing permissions on the shadow password file. Solaris sites in particular have found it

challenging to make SASL work with PAM. The original 1.6 release excluded support for *pwcheck*,* and some sites complain that 1.6 "force-fed" new features by making that exclusion. However, 1.6.22 does include support for *pwcheck* with the SASL PLAIN mechanism, allowing sites to deal with that as they will. The latest version of the SASL library also includes support for *pwcheck*, so sites can once again pass around cleartext passwords if they so desire. It appears that Cyrus project management is doing its best to accommodate current Cyrus sites, while keeping development right on track with the IETF, which no longer ratifies protocols that rely on cleartext authentication.

Project Cyrus is planning to recommend that sites use CRAM-MD5 and */etc/sasldb* for local authentication instead of PAM. Users in */etc/sasldb* do not have to be listed in */etc/passwd* in order to authenticate. Because they're not listed in */etc/passwd*, nothing special need be done to keep them from logging in to system.

Strengths and Weaknesses of Cyrus

Although you can bring up the UW IMAP server on a multipurpose Unix box with little or no forethought, the same cannot be said of Cyrus. Even under the best of circumstances, putting Cyrus on your machine is a major challenge. The only way to know if you want to "go there" is to weigh the strengths and weaknesses of Cyrus within the context of your own needs and resources.

The strengths of the Cyrus server are that it's popular, scalable, and secure. Here are the strengths in detail:

Popular

Cyrus is very popular. The community of Cyrus administrators communicates and cooperates via the *info-cyrus* mailing list, which has over 400 subscribers. The Cyrus server is downloaded over 3,500 times every month. In other words, if you operate a Cyrus site, you're in good company. The attitude of other Cyrus sites towards questions and problems posted to the list is always helpful and cooperative. The Cyrus developers often contribute helpful advice and tips to the list, as well.

Scalable

Cyrus supports quotas on mailboxes—your */var/mail* partition will never fill up unexpectedly. Growth of your mailstore can be controlled using quotas and IMAP partitions.

Secure

Cyrus natively implements the Kerberos v4 SASL mechanism.

---

* *pwcheck* is a daemon that allows the Cyrus IMAP server to check the shadow password file without requiring the permissions or ownership on the shadow file to be modified.

Supports shared mailboxes
: Cyrus supports shared folders by way of ACLs and allows concurrent read-write connections from more than one user.

Here are some of the weaknesses of the Cyrus server:

Cyrus doesn't use Unix mail format
: Cyrus's private mailstore and mail storage format make it non-trivial to convert to other mail formats, such as standard Unix format.

Users may feel locked out
: What you consider a strength, your users may consider a weakness: users accustomed to directly accessing their mail folders can no longer do so—they must use an IMAP client to get at their mail on a Cyrus server.

Server-side filtering is green
: Server-side filtering is still in the early stages. Cyrus does not easily support popular add-on filtering programs like Procmail. Sieve, built into versions of Cyrus later than 1.6, is in development, but there is no practical mechanism or clients available yet that allow users to edit their filters.

Recent changes in direction
: Development of authentication methods in the very latest releases of Cyrus (1.6 and later) has not stabilized yet.

I/O intensive
: Like other IMAP systems, Cyrus is I/O intensive and requires careful choice of an operating system and hardware configured to handle a heavy I/O load.

Learning curve
: There's more of a learning curve involved in managing a Cyrus system.

When Is Cyrus the Right Choice?

The Cyrus server's design is a quite different approach to IMAP than the UW server. The UW server emphasizes personal mailboxes, while Cyrus supports both personal mailboxes and shared mailboxes equally. The Cyrus server's closed, or black box, design makes access to the Cyrus mailstore faster than access through the UW server. There is, however, a trade-off for that performance improvement, such as the inability of users to log in to the server. If there's a need to move mail from a Cyrus server to a traditional Unix mail system or UW server, then there's more to moving it than a simple file copy. The mail must either be translated back through the IMAP protocol or suitably munged using a translating script that copies the mail back into the appropriate format via a filesystem protocol (such a script is provided in Chapter 9).

Here are some criteria that a site can use to determine whether the Cyrus server is a good match:

- Do users need to share private mail folders with other users?
- Is there a per-user limit on the amount of space the user can occupy in the mailstore?
- Does the site need to provide public forums with controlled access (i.e., bulletin boards)?
- Does the site require encrypted authentication?
- Does the site anticipate fast growth in the number of users on the system?

If a site answers yes to the above criteria, then Cyrus is the best choice.