

Comparing List Servers

By John Buckman

John Buckman is President of Lyris Technologies, Inc. and programming architect behind the Lyris list server.

Copyright 1999 Lyris Technologies, Inc.

Contents

- Introduction 1
- What is a List Server? 1
- List Server Issues 1
- Comparing List Servers 3
 - Subscribing 3
 - Unsubscribing 5
 - Sending Email 8
 - Archives 10
 - Error Mail 11
 - Mail Loops 14
 - Administrative Tools 15
 - Security 16
 - Scalability 17
 - Non-English Languages 18
 - Customizability 18
 - Compatibility and Availability 19
- Conclusion 21

Introduction

There are many list servers on the market with varying sets of features. It can be difficult to decide which one is best for you. This article gives an overview of the issues that list servers attempt to address, and compares features of four major list servers as they relate to these issues.

What is a List Server?

A list server is software that distributes email to members of a mailing list. A minimal list server sends email to members and maintains the member database. People use list servers primarily for two functions: (1) to host email discussions and (2) to send one-way email announcements or publications.

Some mailing lists exist without a list server. In those cases, an administrator manually maintains a list of addresses and personally sends email. However, manual maintenance of mailing lists can be tedious and time consuming. Automated list servers save administrators time, allow members to get faster mail responses, and provide advanced features that benefit administrators and list members.

Although programmers can write very simple list servers in a short period of time, these programs are not effective when mailing lists are fairly large. Simple list servers also require considerable human effort to maintain the list. Companies that wish to maximize email marketing and customer support efforts require highly sophisticated list servers that can automatically send mailings to hundreds of list members. To fulfill this need, a growing market for list server software has emerged and encouraged the development of increasingly advanced features.

List Server Issues

Most list servers attempt to address the following 12 issues:

Subscribing: People need a simple way to subscribe to mailing lists. List servers need user-friendly subscription features that also help prevent list abuse (such as people subscribing other individuals without prior consent).

Unsubscribing: People also need a quick and easy way to unsubscribe from lists. Unsubscribe features need to effectively handle challenges

that arise when people change their email address after subscribing to a mailing list.

Sending email: Sending mail reliably and quickly is especially relevant to companies that send mailings to thousands of list members. Servers need to optimize how they work with external mail servers or come with built-in mail servers to maximize mail delivery methods.

Archives: Discussion or announcement list archives can become an electronic library of valuable information. Advanced list servers not only provide archives of posted messages but also facilitate reading messages with web interfaces.

Error mail handling: Each month, approximately 5% of all email addresses on mailing lists change or become invalid. When list servers try to deliver mail to these “bad” addresses, mail servers send error reports back to list servers. Unless list servers can analyze error reports with proficiency, list administrators with large mailing lists may spend a great deal of time analyzing these reports and unsubscribing problematic addresses from the list. List servers that handle error mail efficiently, reduce administration time.

Mail loops: A mail loop occurs when one automated program sends a message to another automated program, which replies with a message that starts the cycle over again. Since there are many people on the Internet who use automated programs to send email, the probability for mail loops is very high. It is important that list servers have the capability to detect loops and end them, or mailing list members may receive hundreds of email messages.

Administrator tools: List administrators often need to perform varied tasks. The ease in which list servers help administrators accomplish these tasks is very important. List servers may offer an email-only interface, a native operating system tool, or a web interface that allows administrators to manage list tasks.

Security: List servers need to offer robust security features that allow list administrators to protect their lists from non-legitimate users. Providing effective security features is especially important in light of the growing number of spammers (people who send unsolicited junk e-mail to others). Security features on list servers need to be simple for end-users, but flexible and robust for administrators.

Scalability: High usage sites may require list servers to process and send a high volume of messages to thousands of members. Well-designed list servers possess the sophistication to handle such extreme loads without additional hardware or by adding additional machines, faster CPUs or more memory.

Non-English languages: Flexible list servers should be able to communicate with Non-English speaking list members in their native

language, and support posted messages with accented characters or alternate character sets.

Customizability: Customizability is important for users who desire flexibility in how they use list servers. For instance, a list administrator may wish to add functionality, integrate a mailing list with an existing database, or re-brand a list server so it appears as a product of the company. List servers need to offer customizable features to fulfill unique needs of end-users.

Compatibility and availability: All of the above issues matter little if a list server is not compatible with existing software, available on varying platforms, or priced at an affordable level.

Comparing List Servers

The following section expands the discussion of the above issues and compares the four list servers that dominate the market: Majordomo, ListProc, LISTSERV, and Lyris.

Subscribing

Methods for Subscribing

With the historical method of subscribing, a potential subscriber types an email command into a message and sends the message to the mailing list's command address. The list server checks the command address for new messages, reads the command in the message, and acts upon the command. Most list servers support this method.

The most common email command for subscribing is the word "subscribe" typed in the body of a message. Depending on the type of list server at a site, the list server's command address may be `majordomo@domain`, `listproc@domain`, `listserv@domain`, or `lyris@domain`.

List servers like Lyris are "backward compatible." This means that Lyris will also accept email at all alternate server addresses (such as `majordomo@domain`) and act on this email as if it were sent to the `lyris@domain` address. Lyris also supports the following alternate subscribe address that only Majordomo uses:

```
listname-request@domain
```

The syntax of subscribe commands varies among list servers. For Majordomo, it is:

```
subscribe listname
```

People who wish to join a list can send a message with this command to `majordomo@domain`. If people wish to omit the list name in the

email command, Majordomo allows users to type only the word "subscribe" in the body of the message and send it to an alternate list server address: listname-request@domain.

For ListProc and LISTSERV, the command syntax is:

```
subscribe listname your name
```

The Lyris list server fully supports all of the different types of syntax and addresses, a flexibility that allows the list administrator to choose a preferred syntax or address.

A newer and easier method of subscribing has been recently developed to meet the needs of less-experienced users. This technique uses a special command-address that allows users to join a list by sending any email (even a blank email) to an address such as the following: subscribe-listname@domain.

The advantage of this technique is that it does not require people to learn email subscribe commands. The Lyris list server supports this method as the preferred way of subscribing. Majordomo can also support this technique, but it requires modifications to support it.

In addition to the use of email commands and special command addresses, people can also subscribe with web forms. With web subscription forms, people type their email addresses into a form which delivers the subscribe request to a list server's command address. This technique is also easy for users and does not require knowledge of email commands. The Lyris list server has built-in support for web-based subscribing. The other three list servers do not have such built-in capability, but people can use a stand-alone web script to provide this feature for list servers that do not provide it out-of-the-box.

Subscription Confirmation

Beyond allowing people to subscribe easily, list servers also need to perform other subscription-related functions, such as sending confirmation email messages. When someone sends a list server an email subscribe request, it is wise to have the list server reply with a confirmation message. This message can confirm (1) that the person has subscribed with a valid email address (it's not uncommon for people to mistype their email addresses), and (2) that the individual is truly the person who sent the subscribe request (i.e. that the subscribe request is not forged).

Unfortunately, some people on the Internet subscribe other individuals to mailing lists without prior consent. Interestingly, the rate of prank subscriptions tends to be much higher for web form subscription methods than for email-command methods. This higher rate reflects the fact that it is easier for pranksters to type someone else's email

address into a web form than to forge an email message. Confirmation messages are an effective way to thwart Internet pranksters.

All four list servers support a subscription confirmation mechanism. Basically, these list servers have an optional confirmation feature that lets the list administrator decide whether to send a confirmation request message to new subscribers. The confirmation request message informs people that they have requested to join the mailing list. It may also request that people reply to the message in order to become a full member of the mailing list. This type of confirmation message prevents pranksters from using list server resources to send mail to an invalid address or to someone who does not wish to join the mailing list.

Unsubscribing

Methods for Unsubscribing

Once people subscribe to a mailing list, they may wish to unsubscribe at some point and stop receiving list messages. The standard unsubscribe method requires a subscriber to send an email message to the list server command address. The mailing list member must include the words "unsubscribe listname" in the body of the message.

While this method seems effective and simple, it breaks down in several ways. First, if a member of a discussion list forgets the list server command address and posts an unsubscribe request to the mailing list address, all members of the mailing list may receive a copy of the request. In addition, the member will remain subscribed to the list.

When a member sends an unsubscribe request to the mailing address of an announcement list, the member not only remains subscribed (in most cases), but may also receive a sometimes-cryptic error message that does not explain why the unsubscribe request failed. Further, if the announcement list lacks good security, the unsubscribe request might be distributed to all members of the list. Consequently, it is helpful for the list server to detect unsubscribe requests that are sent to the mailing list address and intercept them before they create problems. ListProc, LISTSERV and Lyris all support this capability.

The recent innovation of special command addresses (also used to simplify the subscribe process) simplifies the unsubscribe process for members. The special command address for unsubscribing is:

```
unsubscribe-listname@domain
```

When a person sends an email to this special command address, the list server will unsubscribe the person named in the "From:" header. The advantage of this approach is that it is simple for mailing list members, since it does not require them to learn email commands.

Problems With Unsubscribe Methods

There are potential pitfalls associated with both the email command and special command address methods for unsubscribing. For example, problems frequently occur when people change their email addresses. Why are address changes problematic? When the list server receives an unsubscribe request, it looks at the "From:" address in the header to determine the identity of the mailing list member. If a person sends an unsubscribe request from a new email address, the list server cannot recognize the person as a member of the list, since the "From" address in the membership file is the person's old email address. This lack of recognition causes the unsubscribe request to fail.

A complicated dialog can occur between a list administrator and mailing list member, when the member realizes that the unsubscribe attempt has failed and becomes frustrated. Unfortunately, the list administrator may have difficulty helping the member without knowing the exact email address the member used to subscribe to the list. It can be a tedious process for the list administrator to determine which email address to unsubscribe for the member.

Solutions to Email Address Changes

To date, there have been several attempts to solve problems caused by email address changes. For instance, Majordomo allows mailing list members to specify a different email address to unsubscribe than the address that appears in the "From:" header of a message. The syntax for this command is:

```
unsubscribe listname old@address.com
```

This command can prevent unsubscribe failures in cases where a member recalls the email addresses he or she used to subscribe to the list. However, many members frequently forget the email address that they used to subscribe to the list, and are not able to use this command effectively. In addition, this specific command increases the risk of mailing list abuse, since pranksters can use the command to unsubscribe other mailing list members who wish to remain subscribed to the list.

In the past, LISTSERV dealt with email address changes by allowing anyone to obtain their mailing member lists. Members who had trouble with unsubscribing could look at the member list, find themselves in it (perhaps at an old email address), and request that the list administrator unsubscribe the old address. Unfortunately, unscrupulous mass emailers (spammers) quickly learned that they could gather email addresses this way. LISTSERV quickly made the "obtain mailing member list" feature an option that the list administrator could turn on or off, and these days, most administrators turn it off.

ListProc deals with changes in email addresses by looking for addresses in the member mailing list that look similar to the address in the unsubscribe request message. If ListProc finds a similar address, it sends this address to the person wishing to unsubscribe, along with instructions that tell the subscriber to send an unsubscribe request from the similar address. This look-up method is helpful in cases where the email address has changed only slightly, as is often the case when companies add mail gateways. However, the look-up method is not very helpful when an email address has changed significantly.

Lyris does the best job of solving problems caused by email address changes. It uses a failsafe unsubscribe technique. Lyris does this by sending unique messages to every recipient of the mailing list, and giving each member a unique unsubscribe command address. Instead of a generic address such as `unsubscribe-listname@domain`, a Lyris unsubscribe address might appear like this:

```
unsubscribe-listname-31265P@domain
```

A different member might have this address:

```
unsubscribe-listname-53261F@domain
```

The number at the end of the email address precisely indicates which member is requesting to unsubscribe. The final character (the F or the P in these examples) serves as a "check character" to make sure that the unsubscribe address has not been tampered with or mistyped. With unique unsubscribe addresses, Lyris list members can easily unsubscribe from lists even if they have changed their email addresses significantly since they subscribed.

Headers That Facilitate Unsubscribing

Quite recently, a new Internet RFC standard for mailing list headers was developed to facilitate unsubscribing from mailing lists. The standard was written to encourage list servers to include helpful information in the headers of email messages. Mail-reading programs can detect these headers and assist members unsubscribe from lists. One such header is the "List-Unsubscribe" header, which tells a member how to unsubscribe from a mailing list. As of the writing of this article, Pegasus Mail and Eudora (for the Mac) email client programs can read this header and respond by generating a button that the user can easily click on to unsubscribe from a mailing list.

The Lyris list server uses its failsafe unsubscribe technique to include the "List-Unsubscribe" header by default. You must customize Majordomo and ListProc list servers to include this header. It is expected that in the future, more email client programs will support this feature. Consequently, it is desirable to have a list server, which supports the inclusion of helpful headers by default.

Sending Email

One of the most basic capabilities of a list server is its ability to send email to the list's membership. A multiplicative effect occurs when members send messages to a mailing list. For example, a typical discussion list has 300 members, and each member posts about 20 messages per day. Therefore, members send 6,000 email messages per day. An average list server might run 10 such mailing lists, and send 60,000 email messages per day. If a server runs 100 such lists, it might deliver 600,000 email messages per day, a number that far exceeds the capability of most email servers. Clearly, the method of mail delivery is a crucial factor when comparing list servers.

Email Delivery Methods

ListProc, Majordomo, and LISTSERV all pass their email to an external email server that delivers the mail. Consequently, these list servers can be seriously impacted by limitations in external mail server software.

Sendmail is the most common external mail server used on Unix systems. Unfortunately, Sendmail can be very delicate and often performs badly under stress. Generally, the problem with high-performance mailing is the amount of CPU time and memory used for each independent mail-sending session. Sendmail processes require more CPU time because a new copy of the program runs each time someone sends an email message. Sendmail processes also use a fair amount of memory. For example, when we received 20,000 pieces of email over a 24 hour period on our Sun UltraSparc (with 400 MB of physical memory and 500 MB of virtual memory), Sendmail spawned hundreds of copies of itself and used a combined total of 900 MB of physical and virtual memory. This brought our server to a crawl. We solved the problem by restricting Sendmail to 10 simultaneous connections, and waiting 30 hours to receive email.

While one can configure Sendmail to perform extremely well, this optimization usually requires long hours and a Sendmail guru. As a result of Sendmail's limitations, many people with high-load list servers install an alternative mail server (such as qmail or zmailer) that typically performs better under high stress and uses less memory. However, installation of these alternative packages requires a Unix system administrator.

ListProc, LISTSERV and Majordomo try to optimize the manner in which they pass mail to Sendmail or other external mail servers. These list servers save bandwidth, memory, and processor time by grouping messages that are going to the same place. For example, an announcement list with 10,000 members might have 500 members with addresses at the AOL domain. Since each of the 500 AOL members will receive the same message, the list server only needs to

transmit one copy of the message body along with the addresses of the 500 recipients. This saves bandwidth and processing time.

In addition to optimizing the way it passes email to an external server, LISTSERV also offers LSMTP, a mail server that is a replacement to Sendmail and available at an additional cost. However, LISTSERV's LSMTP version is not available on as many platforms as the non-LSMTP version.

Lyris is the only list server of the major four that uses a built-in mail server that is available at no additional cost. Lyris' built-in server solves stability and configuration issues normally associated with installing a list server and using an external mail server software. Instead of focusing on minimizing external mail server limitations, Lyris places its focus on a powerful, built-in mail server that can perform high performance mailing with processes that optimize memory and CPU usage. The built-in mail server also enables Lyris to offer unique features that require an integrated mail engine, such as failsafe unsubscribe addresses for each member, customized per-user message banners, and mail merging similar to a word processor.

The Lyris list server can also use an external mail server to deliver its email, but there is little reason to do this except in cases where a firewall restricts outbound access. Given the limited circumstances in which one might use an external mail server with Lyris, it does not offer as many features as the other three list servers do for optimizing mail delivery with Sendmail.

Multi-threading: A Solution to Memory and CPU Issues

The latest innovation to solve memory and CPU issues related to high performance mailing, is the use of a relatively new operating system feature called multi-threading. With multi-threading, one process launches separate "lightweight threads" instead of creating one separate process for each mail-sending session. Threads inside a process use fewer system resources, but are significantly more difficult for the programmer to use. By using multi-threading, the mail server can maintain a greater number of a simultaneous mail-sending sessions than is possible with the traditional process-oriented mail-sending technique.

The downside of using multi-threading is that many brands of Unix do not support this feature well. Thus, multi-threaded mail servers are not as available as mail servers that use the traditional process-oriented mail-sending technique. LISTSERV's LSMTP server uses this multi-threading technique, as does the mail server inside Lyris. However, LSMTP is only available for the Windows NT and VMS operating systems. It is not available for any Unix platform. LISTSERV without LSMTP is available for many more platforms, but it must use an external mail server for its email delivery. Lyris' built-in mail server

also uses multi-threading and is available for Windows 95, Windows NT, Sun Solaris and a number of other Unix types. As a result, Lyris is the only multi-threaded email delivery solution for a number of popular Unix platforms.

Archives

Since an archive of mailing list messages can be a valuable resource, it is useful to have a web interface for accessing archived messages.

There are two methods for making archives available through a Web interface: (1) a bulk conversion of the messages into HTML pages, or (2) a dynamic conversion of messages as people request to read them.

The advantage of the bulk conversion process is the creation of static HTML pages that popular search tools (such as AltaVista) can index. Since the Web server performs little work to display static HTML pages, the web interface for message reading is very fast. Also, one can schedule bulk conversion processes for low-load times instead of high-load, peak times. A disadvantage of static HTML pages is that unless you password protect the archives or obscure email addresses, spam companies can access the email addresses of mailing list members and send them spam.

Alternatively, with the convert-on-demand process, the list server program restricts access to the archives according to the mailing list's security policy. With this convert-on-demand process, the list server program recognizes mailing list members after they complete a login step, and then customizes output for each individual. Members can also view a subset of the information that interests them. For example, the customized output of the convert-on-demand process can display the past 15 days of messages by a single author or only a specific message thread, depending on what the member requests. This is not possible with static HTML pages created by the bulk conversion process.

One downside of the convert-on-demand process is that security measures for mailing lists may require members to authenticate themselves with an ID and password before they can view archived messages. While this login step may thwart spammers and other non-legitimate users, it can also slow down the process of reading archives for legitimate users. Still another disadvantage is that converting messages on demand requires more CPU time, memory, and disk space (often at peak times). This results in a slower web interface than the bulk conversion process method. Of course, the speed of the machine running the archive interface and its load determines the member's perception of slowness. A final disadvantage of the convert-on-demand process is that it does not allow search engines to index the contents of archives. However, these disadvantages are often

outweighed by the advantages of a security controlled web interface, custom browsing, and generally richer reading environment.

How Do List Servers Archive Messages?

The most recent versions of LISTSERV and ListProc offer a bulk conversion process for reading messages on the Web. Majordomo does not offer a built-in interface for reading web archives. However, there are a number of third-party freeware programs that offer this capability. All four list servers are compatible with third-party freeware programs for displaying web archives, since they output the standard Unix mailbox format that third-party programs read.

Lyris list server has a built-in (convert-on-demand) web interface for reading archives. Since Lyris runs a relational database system, its web interface can operate quickly and perform full-text searches with keywords, a capability offered by default. The other three list servers do not offer this capability by default. LISTSERV, ListProc and Majordomo require the installation of a freeware search program (such as Glimpse) and Web server scripts to index mailing list archives.

Lyris' archives are also optionally "open to the public" and do not require ID or password logins. Since the Lyris archives are viewed through a web interface with an interactive script, spam programs are not able to browse Lyris archives and gather email addresses, so tedious login steps are not required.

Error Mail

About 5% of all email addresses on a mailing list change or are no longer valid each month. For instance, on a mailing list with 1,000 people, 50 addresses will return an error report to the list server each month. If there are 30 posts per day to this address, this translates to 1,500 incoming error reports per day, with the quantity doubling after the second month.

Error reports come in two general types. One type indicates that a message cannot be accepted or delivered to a recipient. For example, if someone gets a HotMail® email address account and later discontinues it, HotMail® will refuse mail to that address and indicate to the sending mail server that the message cannot be delivered. The sending mail server immediately communicates the problem to the list server during the delivery process. The error report from the sending mail server is often formalized to allow list servers to read and comprehend the error. As a result, this type of error report is easiest for list servers to handle.

The second type of error report indicates that a message was accepted but cannot be delivered. This type of error report is more difficult for list servers to handle since the error message syntax varies among

servers despite the existence of a syntax standard. In addition, many mail servers return an error report that fails to specify which email address caused the error. Sometimes, when ISPs multi-host different domains, the error report may come from a different domain than the domain that caused the error. Needless to say, the list server has difficulty matching the error to the offending address.

Sometimes, a special case of error mail occurs when mailing list members have full mailboxes. The "mailbox full" error usually goes away after people read their email. Consequently, it is helpful for list servers to use an algorithm to determine if a mailbox is full instead of indiscriminately removing members who create a small number of error mail messages when their mailboxes are full.

List servers that indiscriminately remove members with full mailboxes create problems for list administrators. Administrators may become inundated with complaints from members who wished to remain on the mailing list. If a mailbox is full for several months, an advanced list server might question and possibly remove the member since it is unlikely that the person will read his mail.

How Do List Servers Handle Error Mail?

Majordomo does not attempt to identify the source of error reports, but instead forwards the reports to the list administrator who is expected to read and comprehend the error messages, and remove users responsible for the error reports. Obviously, this task takes a great deal of administrative time. Consequently, many Majordomo list administrators delete error reports. However, this is not effective because it causes the list server to work harder as it continues to deliver email to invalid email addresses and receive even more error reports.

Some ISPs, such as AOL, refuse to use their resources to send messages to inactive or invalid email accounts. They sometimes block all email from mailing lists that send a significant number of messages to these types of "dead" accounts. Consequently, users of these types of ISPs may have difficulty with participating in Majordomo mailing lists.

LISTSERV and ListProc try to understand incoming error reports by filtering the most common types of reports. Unfortunately, this is an extremely difficult task, since the error mail reports are often in different languages or formats, and come from hundreds of different servers and software packages. In cases where the list server can read the error reports, the mail may fail to mention the specific email address, which caused the delivery error. Therefore, it is difficult to pinpoint the source of the problem.

SmartBounce is a third-party program that augments error mail recognition capabilities of list servers. SmartBounce tracks error mail reports (or “bounces”) and only unsubscribes members after they exceed a user-definable number of error mail reports over a period of time. It is compatible with all four of the major list servers.

The author of SmartBounce performed an informal analysis of LISTSERV's error mail analysis rates and found that LISTSERV properly identified an offending email address for approximately 40% of the error reports. At the time of this analysis, Lyris used an augmented version of the same error mail analysis technique that LISTSERV used—namely, reading messages. However, Lyris also included the recipient's email address in the message header, and by default, the member's address in the message footer. As a result of these enhancements, Lyris identified a greater number of email addresses that caused error reports than LISTSERV did. The author of SmartBounce estimated a success rate of 80% for Lyris, compared to the success rate of 95% for SmartBounce.

Since that report, Lyris changed the way in which it processes error mail. Lyris switched to a technique where it sends a custom return error mail address with each email message. For instance, a member's error address may be: “bounce-listname-23423P@domain.” Mail servers use this custom address to return error reports. Lyris can use this technique since it is the only server that sends each message as a unique message with distinctive, identifiable components. Users do not usually see the special error mail address. Instead, mail servers, who receive the message, see and return error mail to the special error mail address. Since each mailing list member has his own unique error mail address, Lyris can easily identify a member address that causes an error report by looking at the address to which the report is sent. This feature was added to version 3 of Lyris. As a result of this feature, Lyris' error mail analysis is over 99% accurate.

The only case where this accuracy rate does not hold is when a small number of mail servers violate RFCs (Internet standards) and send error reports to the author of the message instead of to the list server. No list server is currently capable of addressing the problems caused by the small number of mail servers engaging in this practice, since the error mail message never goes to the list server. The main culprit is an old version of CC:Mail. As CC:Mail is phased out in favor of Lotus Notes, we expect this problem to become even less significant over time.

In addition to comparing how list servers handle the two basic types of error reports, we must also examine how they handle temporary error reports caused by a full mailbox. LISTSERV and Lyris put mailing list members on hold when they frequently encounter “mailbox full” error messages. These list servers also have an option to send a hold

message to remind people, over a scheduled duration of time, that their membership is on hold. The hold message also explains how the member can return to normal status. For example, if a mailbox is full for several weeks because someone has gone on vacation, the list server might place the member on hold and deliver a hold message every two weeks, up to three months. This message allows the member to reinstate himself before permanent removal takes place.

Mail Loops

Since many people use automated programs to send email, it is very easy for two automated programs to get into a mail loop. A mail loop occurs when one program sends a message to another program, which replies with a message that starts the cycle over again. This loop may cause the two programs to trade hundreds of email messages per minute in a frenzy of automation. When a mail loop occurs on a mailing list, all members can receive hundreds of messages that may cause the list server to crash or run out of disk space and memory.

Some mail loops are frequently caused by vacation messages. For instance, if a Mr. Jens goes on vacation, he may set up his email program to send an "I am on vacation" message as a reply to any message that arrives in his mailbox. When Mr. Jens receives a message from a mailing list, his email program replies with his vacation message that gets distributed to everyone on the mailing list, including himself. When Mr. Jens receives a copy of this vacation message from the mailing list, this triggers his email program to send yet another copy of the vacation message to the mailing list, and so on.

Since many mail servers allow users to write their own custom vacation messages, users may write vacation messages in different formats and languages (sometimes differing from the language of the mailing list). Consequently, list servers cannot easily detect "vacation messages" as distinct from other types of messages; list servers automatically distribute these vacation messages as they would any other message that the mailing list receives.

Needless to say, mail loops on mailing lists can create a public relations nightmare for the company that runs the list server. As members are inundated with hundreds of messages, they often complain vigorously and request to unsubscribe in large numbers. After such an experience, it can be very difficult for a company to repair its corporate respectability.

How Do List Servers Prevent Mail Loops?

There are two common techniques that list servers currently implement to prevent mail loops. First, list servers check for common text that may indicate a loop-causing email message, such as the word "vacation" appearing on the subject line. Second, list servers may

apply a generic filter to identify different kinds of mail loops and stop them in progress. For example, a list server might keep track of all the messages sent in one day, and make sure that no one person sends exactly the same email message twice in one day. The problem with such a filter is that vacation messages sent to the list often include the original message from the mailing list or text indicating the exact date and time of the message. List servers have difficulty filtering a loop-causing message because each iteration of the message is slightly different.

Highly advanced list servers have aggressive filters and proprietary techniques to address mail loop problems. All four of the major list servers have such techniques. Despite this, some of these list servers have less adequate mail loop detection compared to others. For example, ListProc version 6 has major problems with mail loops.

The Lyris list server, as a result of its integrated mail server, can automatically track the number of messages per day that mail servers send to a Lyris address. If the number of messages in one day exceeds the adjustable loop detection threshold, Lyris stops the mailings, issues a single refusal message, and then discards the rest of the messages in that 24 hour period. This feature allows failsafe loop-breaking while allowing threshold exceptions for trusted automated programs such as Web subscription forms.

Administrative Tools

There are many tasks that a list administrator might need or want to perform regularly. For instance, the administrator might want to edit moderated messages, change mailing list settings, or update the welcome or goodbye documents. LISTSERV, ListProc, Lyris and Majordomo offer email-based administrative tools for performing these and other tasks. These tools allow the administrator to send specific email commands to the list server, which processes the commands and sends a reply.

In addition to email-based administrative tools, the Lyris list server offers an extensive web interface for performing all administrative tasks. A web interface is generally more functional and interactive than an email-only interface. For instance, the web interface allows the administrator to edit a moderated message on screen and immediately see how the edits will appear in the message. A web interface also lets the administrator easily choose among different administrative tasks by clicking a button instead of typing specific email commands that may require referencing from a printed document.

LISTSERV, ListProc and Majordomo can use freeware programs to add administrative functionality to a web interface. However, the functionality provided by these freeware programs is very primitive,

and is usually just a simple form that enables the program to send email commands on the administrator's behalf.

Security

Security is a key concern for list administrators who want to protect their lists from those with unscrupulous purposes. Given the increasing problems with spam, it is essential that list servers have robust security features that enhance security measures found in common email programs. Most email programs look at the "from:" line of messages to authenticate messages. However, advanced list servers go beyond this level of message analysis and use more sophisticated methods to authenticate messages.

Special Message Posting Address

Majordomo typically uses a special message posting address, known only to the administrator. Yet, this address can be easily guessed and has been vulnerable to attack by spammers and others.

Posting Confirmation Process

ListProc, LISTSERV and Lyris offer a posting confirmation process. With the posting confirmation process, the list server holds a posted message until the administrator receives a confirmation message and copy of the posted message. Upon approval from the administrator, the list server removes the posted message from hold status and distributes it to the mailing list members. This is generally a very secure method although saboteurs can theoretically scan the administrator's email and forge an acceptance message.

Lyris uses a web interface to address the problem of potential saboteurs. The Lyris web interface is a standard CGI script that can run within a secure web server, using SSL (secure-socket layer) encryption. Lyris stores all messages that members post in a "pending moderation" area, and waits for approval by the list administrator before distributing the messages. For each message posted, Lyris sends the administrator a "request-to-approve" email message. To approve a message, the administrator replies to the "request to approve" message. This reply must contain both the message number of the post and a password that corresponds to the administrator's "From": address. Consequently, this method prevents forged administrator confirmation messages.

The Lyris web interface also allows composition of approval messages in a secure environment. As an added layer of security, one can instruct Lyris to prefer only administrator approval messages submitted through the web interface, and reject email submissions as non-authentic administrator replies.

Password-Protected Posting

In addition to the posting confirmation process, list servers may also address security issues by requiring a password in the body of the message to authenticate the identity of the users who send mail to the list. With this method, the list server recognizes and removes the password from the message before distributing it to the mailing list. The password is unique to the sender. The list server will refuse all messages that lack this password.

Both Majordomo and Lyris support this password-protected posting technique. However, Majordomo only supports it for administrators who post to an announcement mailing list. Lyris supports this technique not only for announcement mailing lists but also for discussion lists where many people can send messages. Lyris' implementation is flexible enough to allow the list administrator to determine whether to require password-protecting posting of all members or allow individual members to decide which level of security they prefer. Furthermore, Lyris allows members to change their password at will, by using email commands or its web interface.

Authenticate by Text Search

Another way list servers address security issues is to search for key text in the message. For example, an administrator who uses a specific version of Microsoft Exchange can instruct the list server to look for this version number in the headers of contributed messages, reject messages without this version number, and delete or rewrite the headers so a slightly altered number appears. Only the Lyris list server supports this kind of posting security technique, which has the great advantage of complete automation and does not require additional work from the message author once the security mechanism is set.

Scalability

Scalability refers to the ability of the list server to handle any load placed on it without additional hardware or by adding memory, a faster machine or higher network bandwidth. If a list server cannot handle demanding loads even with excellent hardware, it is considered "non-scalable."

In general, Majordomo is the least scalable of the four major list servers, since it is written in Perl (the other three are written in C or C++), and it uses an external mail server for mail delivery.

ListProc is fairly scalable, since it is written in C, and is used at some large sites. Yet, ListProc has also experienced spectacular failures such as sending the same message dozens of times when it is overstressed.

The most scalable list servers are LISTSERV and Lyris. Hundreds of highly visible sites with demanding loads use these two list servers.

LISTSERV is available on mainframe machines that run IBM VM and Digital VMS operating systems, and use highly scalable hardware. As a result, LISTSERV is capable of working in a mainframe cluster. Yet, the only version of LISTSERV that can handle high usage sites is LISTSERV LSMTP, available only for VMS and Windows NT operating systems. In contrast, Lyris has the option of using a SQL server (such as Oracle and Microsoft SQL Server) as its database engine, and clustering around the SQL Server. Another advantage of Lyris is that it is also available for a number of Unix platforms and machines that also have highly scalable hardware.

As for scalability with Windows NT operating systems, both LISTSERV LSMTP and Lyris possess multi-threaded mail engines that allow them to scale well when additional CPUs are added to Windows NT. The multi-threaded engines work well with Windows NT since this operating system distributes threads among separate processors.

Non-English Languages

List servers that can communicate with users in their native language are a great advantage when mailing list members do not read English. One difficulty of having a non-English mailing list is that some list servers cannot properly distribute email messages that have alternate encodings, necessary to preserve non-English characters.

As a result of its available Perl source code, Majordomo can be fully translated, but this requires a staff Perl programmer to accomplish. Nonetheless, there are non-English versions of Majordomo in use. Currently, ListProc, LISTSERV, and Lyris do not have the capability of full translation, although these servers can translate some documents.

Lyris can maintain multiple language versions of hello and goodbye documents and analyze the context of a command to determine which document to send as an appropriate reply. Lyris can also define a list's primary language. People who subscribe to a Lyris mailing list will automatically receive a welcome document in the language defined for the list. In cases where Lyris cannot determine the most appropriate language for list documents, Lyris looks at the country suffix of the email address and uses an internal database to match the country suffix with the appropriate language.

Customizability

Customizability allows the user to alter features and change how list servers work. Many list administrators may wish to customize the list server to add new functions. Majordomo allows users to add new commands by editing its available Perl source code. While very

powerful for Perl programmers, it can be a challenge for administrators who are not familiar with the Perl programming language. LISTSERV allows users to write their own self-standing programs that LISTSERV runs at strategic points. These user created programs are called "exits." Lyris also allows users to write extension programs to add functionality through the use of its programming toolkits for Perl or Java, or communication with Lyris via TCP/IP. ListProc is the least customizable, since it does not offer a Perl/C tool kit or run user-defined programs that address more than a few modifications.

Re-branding the list server is another area that requires customizability. With all four of the major list servers, administrators can change the hello and goodbye documents to include information that is specific to their company mailing list. Again, Majordomo is the most customizable. You can modify its source code to rewrite a document to eliminate the word "Majordomo", something that the other three programs do not allow you to do since these are commercial programs.

Lyris allows administrators to change or remove the Lyris graphics that appear on the web interface, and add custom headers or footers. Lyris also allows further customization by allowing administrators to modify the Perl source code for its web interface. While this level of customization requires skill with the Perl programming language, Lyris also provides an extensive programmer's guide to help facilitate programming with Perl.

Another important area that requires customization is importing members from existing databases. All four list servers allow administrators to write a simple program to import members. However, none of the major list servers allow live integration with an existing database. As of the writing of this article, LISTSERV and Lyris plan to release an SQL version to enable live integration. Currently, Lyris administrators can write a bridge program to read and synchronize data in the Lyris database with an existing database. This bridge program is fairly easy to write if administrators know Perl and have access to the ODBC driver for their database. In addition to writing custom programs for database integration, administrators may also use a stand-alone listserver program, Unitymail by Revnet, which integrates list server software with existing databases.

Compatibility and Availability

All of the issues we have discussed matter little if a list server is not compatible with existing software or available for platforms that the list administrator wishes to use. List server software needs to (1)

coexist with other software on the same machine, (2) work well across varying platforms, and (3) meet consumer needs for affordability.

Compatibility

Compatibility with existing software is an important issue that greatly affects installation success. On Unix, all four list servers are compatible with the major email servers (such as Sendmail, qmail, and zmailer). On Windows NT, only Lyris is compatible with external mail servers on the same machine. Lyris includes detailed instructions for coexisting with more than a dozen mail servers (e.g. Microsoft Exchange, Lotus Notes, and many others) on both Windows NT and Unix. Majordomo has not been ported to Windows NT and cannot work with this operating system, although Perl is available for Windows NT.

Platform Availability

As for platform issues, Majordomo leads the pack in availability for different types of Unix. LISTSERV and ListProc follow close behind. ListProc is available on slightly more Unix platforms than LISTSERV, but LISTSERV is uniquely available for the VM and VMS mainframe operating systems. Lyris is available for a number of Unix platforms although not as available for Unix as the other three servers.

For Windows NT, Windows 95 or Windows 98 operating systems, only two of the major four list servers are available. The multithreaded mail-delivery engines of LISTSERV and Lyris work well with Windows NT. In addition, Lyris also runs well on Windows 95 or Windows 98. None of the major list servers are available for Macintosh operating systems.

Platform issues affect which list servers are best suited for high usage sites. For example, if list administrators require a list server to handle heavy loads, they may choose LISTSERV or Lyris for Windows NT, Lyris for Unix (Solaris, HP/UX, etc.) or LISTSERV for Digital VMS. List administrators who have low to medium loads can effectively use any of the four list servers. However, administration time varies among the servers as discussed earlier, and plays a role in which server is most effective for low to medium loads.

Pricing

As with any competitive market, prices vary greatly among the four major list servers depending on the scale of installation and specific uses. Note that all four list servers have a no-cost version available. Majordomo is the only product that does not have a commercial version, so all versions are free. ListProc version 6 is free (although difficult to find), but it lacks the robustness of ListProc's commercial version. LISTSERV Lite is available free for non-commercial use.

Lyris offers a robust, free version with no restriction on commercial use for lists with 200 members or less.

Conclusion

List server software has become increasingly sophisticated to meet the demands of list administrators, who leverage email to build relationships with potential or current customers. Majordomo, LISTSERV, ListProc and Lyris continue to dominate the list server market despite the emergence of numerous, less sophisticated programs. As the role of email increases as a tool for effective e-commerce, we can expect to see further innovations in the list server industry, especially among the major four list servers.

