July 1999

ECG515/0799

Prepared by:
Internet and E-Commerce
Solutions Business Unit

Enterprise Solutions Division

Compaq Computer Corporation

Preventing Unsolicited Bulk E-mail on Linux Systems

Contents

Abstract:

This guide provides an overview of the Unsolicited Bulk E-mail (UBE) problem. UBE is commonly referred to as spam. This guide contains information on sizing and planning a UBE filter solution for Linux systems. It also contains information on how to configure and manage that solution. Enterprise businesses and Internet Service Providers can protect their mail systems from UBE by utilizing this spam filter solution.

Notice

Preventing Unsolicited Bulk E-mail on Linux Systems
Technical Guide prepared by Internet and E-Commerce Solutions Business Unit

Enterprise Solutions Division

First Edition (July 1999)
Document Number ECG515/0799

Table of Contents

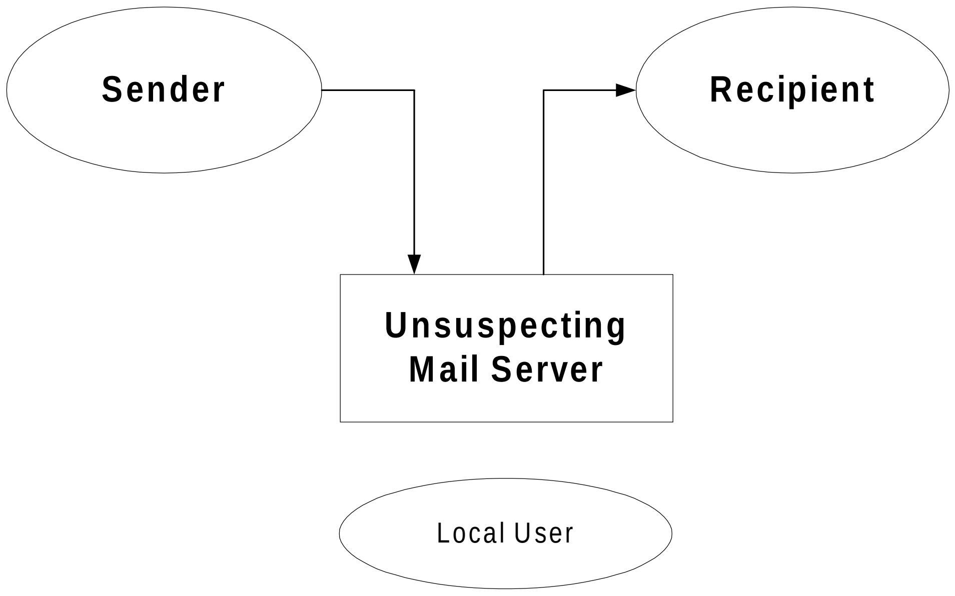
1 Introduction

Unsolicited Bulk E-mail (UBE) involves targeting large numbers of Internet mail users with unsolicited direct mail messages. These unsolicited messages are also known as junk mail or more commonly as “spam.” The process is called “spamming” and the sender is the “spammer.” The unwanted mail typically consists of commercial advertising, get rich quick schemes, and the sale of dubious products. Spamming can even be used as a denial of service attack, whose purpose is to make your mail system unavailable.

1.1 Definition of the Problem

But, you might ask yourself, why is this a problem if it consists only of advertising messages? To understand this, you need to understand the current Internet electronic mail (e-mail) model. In the traditional non-electronic mail service (for example, U.S. Post Office hand-delivered mail), the sender of a message pays for the stamp to send the message. In the e-mail model, it is the recipient of the message that pays to receive the message (via online subscription fees, phone or cable connection services). The person sending e-mail incurs no incremental cost because sending one message costs about the same as sending 100 messages. A large majority of UBE mail is sent by stealing resources from other people. These resources include computers, disks, and network bandwidth to relay the junk e-mail. Most importantly, your good name can be damaged after your systems are identified as the source of the junk e-mail. This is pictured in the following diagram, where a spammer (Sender) is using someone else’s mail server (Unsuspecting Mail Server) to send their junk mail to their targeted recipients (Recipient). The local users of the mail server are neither the senders of the mail, nor the intended recipients. Their mail server system and network resources are being stolen.

Figure 1. Model for Sending Unsolicited Bulk E-mail



Spammers typically target users by scanning addresses from Usenet newsgroups, stealing mailing lists, and scanning the web for addresses. Once they have a list of potential victims, they typically find an unsuspecting third-party mail system through which to relay their junk mail.

There are some great resources on the web detailing the spam problem:

http://maps.vix.com/tsi	MAPS Transport Security Initiative
http://spam.abuse.net	Collection of anti-spam links and resources.
http://www.ISP-Resource.com/spam/	Spam Related Information and News

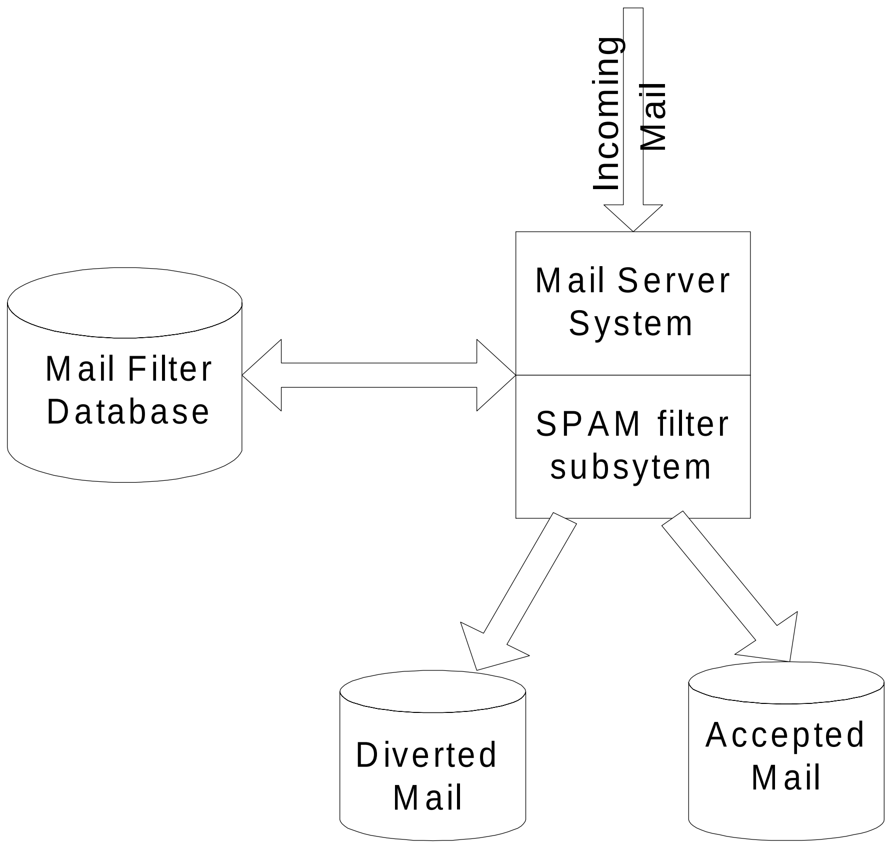
Spammers typically relay their mail because the few known sites that are dedicated to sending out junk mail are known, and at fixed locations. There are features in most mail solutions that allow you to block connections from these sites. In order to avoid being blocked, spammers are hiding behind unwitting third party mail servers. Using other people's resources also allows them to push through more junk mail than they could if they were just using their own computers and network resources.

From an ISP's perspective there are three major problems. One is protecting your customers from undesired junk e-mail. The next is protecting your computers and network from being used as a third party relay by spammers, and the last is to protect the Internet community from a potential user (or stolen account) on your network being used to send out UBE. If you don't protect your customers, you will lose them. Customers don't want to "pay" for junk mail. Many find it a waste of time having to read through it, and offensive (depending on the content). It costs you bandwidth resources, disk space, and CPU processing to handle the extra mail load. To make matters worse, spam arrival profiles are typically very demanding on your system. They tend to come in "spam storms' usually late at night when you're not looking.

1.2 Spam Handling Architecture

Figure 2 presents an example of a spam handling architecture. In this architecture, a mail server processes incoming SMTP mail. The spam filter subsystem attempts to fingerprint the spam, utilizing a database of known spammers, and then determines the disposition of the mail.

Figure 2. Spam-Handling Architecture



1.2.1 Spam Fingerprinting

Checking mail envelope information, header information, and even message content can allow you to attempt to fingerprint a mail message as spam. Since a mail server receives the envelope information before the rest of the message, a decision can be made on whether or not to immediately reject that message. This decision can be based on who the sender is (e.g. User@host or cyberporno.com). If you can reject a message quickly, you can limit the amount of resources spent processing the message.

Header information can be checked when the whole message is received. Checks can be made for subject line information, like “make money fast.” Checks can also be made for a numeric username (for example, 1234@aol.com is not a valid AOL mail address). Checks can even be made of message content, but this is hard to do and involves privacy issues.

1.2.2 Mail Filter Database Utilization

The spam filter subsystem accesses a database for spam management. The database is queried using keys such as domain names, IP addresses, or user@domain. Return values can be “ok,” “rejected,” or even “other” (for customization). These databases can be locally maintained and contain a “black list” of individuals or domains for whom mail should be rejected. They can also be remote databases that are maintained on a system in your own local area network or by some trusted third party that maintains an up-to-date “black list.” The news.admin.net-abuse.usenet.e-mail newsgroup is a great source on Usenet for tracking spammers.

1.2.3 Spam Disposition

After determining the type of message received (spam, non-spam, or just suspect), the spam filter sub-system can accept the message, reject it (return a rejection message), discard it (accept and ignore it), or divert it (save it for later human evaluation). There is no perfect solution for spam disposition. If you always accept messages, you allow in some spam. If you take the time to reject it, you're using your own resources to send a rejection message. To make matters worse, if it was sent from a bogus address, your rejection message will end up being bounced and returned to you. If you discard it, you risk losing mail that was misidentified and having an angry customer. If you divert it, you have to employ a service or your own human staff to read the messages, and decide if it's spam. This opens up privacy issues and additional costs.

1.3 An Open Source Solution to the Problem

Open Source sendmail V8.9 began shipping with several anti-spam features. The spam control features in V8.9 fall into two areas: improvements in configuring some of the existing features, and a number of new features.

Configuration Improvements:

- Promiscuous relaying is turned off by default. This means that your mail server cannot be used by a third party to send UBE.
- Disallow mail having an invalid host name in the return address by default. Invalid host names usually mean someone is using a fake hostname.
- By default, disallow acceptance of mail from any sender that does not have a fully qualified domain name (e.g. mail1 instead of mail1.isp.com).
- FEATURE(relay_entire_domain) to allow relaying within your domain. This is an example of a legitimate third party relay use.
- Optional access control database that allows rejection of mail from specific domains, user mail addresses, or IP addresses.
- FEATURE(rbl) to enable use of the Real-time Blackhole list maintained at http://maps.vix.com/rbl.

New Features:

- Regular expression matching on addresses (for example, all-numeric user names can be rejected).
- MaxRecipientsPerMessage option, to restrict messages being sent at one time to many users (a common characteristic of spam).
- Feature to extract all MX (mail exchange) records for a given domain. This allows a site to relay only hosts for which the site is a valid MX server. For example, if your server receives a recipient of user@domain.com and domain.com lists your server in its MX records, the mail will be accepted.
- Distinguish between temporary and permanent Domain Name Service lookup failures. This allows better rejection of SMTP envelope senders that have invalid host names.
- Allow message rejection on the basis of header contents. For example, messages with invalid Message-Id headers or a "To: friend@public.com" header can be rejected.

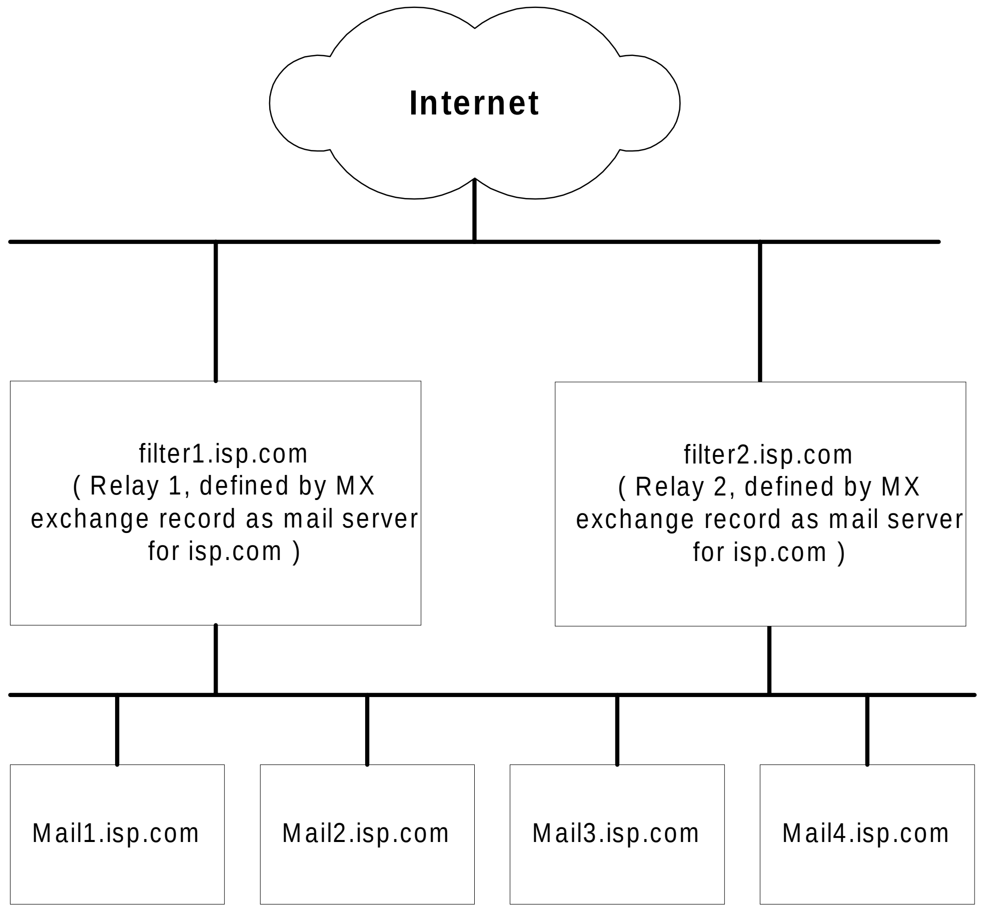
- Limit the size of HELO/EHLO parameter to prevent spammers from hiding their connection information in Received: headers.
- New built-in "discard" mailer to allow messages to be accepted and then dropped.

More information on sendmail V8 and its anti-spam features can be found at http://www.sendmail.org

By using this software on Linux systems you can create a mail relay solution that provides these anti-spam provisions in a highly available manner. This solution can be hosted on an existing Linux mail system, or added as a filtering layer to a larger mail architecture.

In the following diagram, two Linux mail filter relays (filter1.isp.com and filter2.isp.com) are placed in an existing mail architecture consisting of multiple mail server systems hosting subscriber mail (mail1.isp.com, mail2.isp.com, mail3.isp.com, mail4.isp.com). In this solution the mail filter relays are running the sendmail V8.9 open source software with anti-spam capabilities enabled. They filter both incoming and outgoing mail.

Figure 3. Configuration with Mail Filter Relays


Internet
filter1.isp.com (Relay 1, defined by MX exchange record as mail server for isp.com)
filter2.isp.com (Relay 2, defined by MX exchange record as mail server for isp.com)
Mail1.isp.com
Mail2.isp.com
Mail3.isp.com
Mail4.isp.com

2 Testing and Sizing

This chapter describes how to size a UBE filter solution using sendmail V8.9.3. To be sure that the mail filter is adequate for your needs, consider the following:

- The number of mail subscribers to be supported, and the subscriber mail load
- The need for high availability in the solution

2.1 Testing Methodology

In contrast to accepted benchmarks such as SpecWeb for web servers, there is no standard test for mail solutions. Consequently, Compaq developed its own methodology for mail relay performance testing and UBE mail filter sizing.

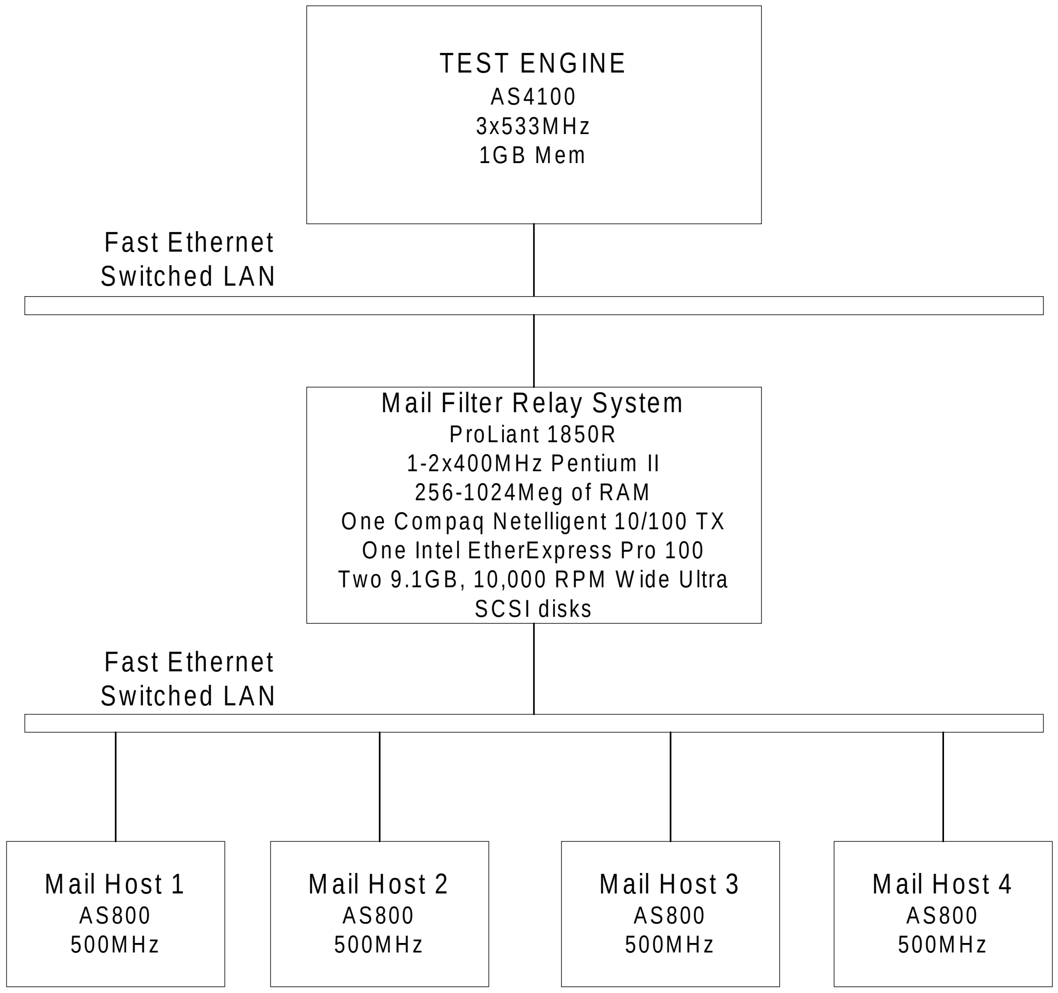
2.2 Test Description and Configuration

Tests were conducted on the configuration shown in Figure 4. The test was driven from a single Perl script. The script ran on a separate test driver system (TEST ENGINE), generating an SMTP mail load. This load was directed at the mail relay system (Mail Filter Relay System). The messages generated by the test driver system were destined for remote hosts (Mail Host 1, Mail Host 2, Mail Host 3, Mail Host 4). The script could spawn any number of simulated SMTP clients (SMTP Mail Transport Agents, or end user SMTP clients). Each client sent mail to a specified number of local mail addresses on the destination mail host systems, one message at a time.

Our testing consisted of running 50 simulated SMTP clients, each testing 200 local mail addresses on the destination mail host systems. Each SMTP message was 20 KB in size. This meant that each test run processed 10,000 SMTP messages. The test was run first with “good” messages being sent to the relay, and then with messages sent from “spammers”. The “good” messages are of course relayed on to their destination mail system. The “spam” messages are rejected during connection time, once the sender is recognized as a known spammer.

The Mail Filter Relay System was a Compaq ProLiant 1850R system. Both single and dual 400 MHz Pentium II with 512 KB cache CPU’s were tested. The operating system was Red Hat Linux 5.2, with a 2.2.9 kernel. The 2.2.9 kernel was chosen because it gave a 50% performance improvement over the 2.0.36 kernel shipped with Red Hat 5.2. This system had two Network Interface Cards (NIC) cards. One was the built-in Compaq Netelligent 10/100 TX Embedded UTP Controller (using the ThunderLAN device driver). The other was an Intel Etherexpress Pro 100 card. Only two disks were used during testing. Both were 9.1 GB 10,000 RPM ultra-wide SCSI disks. One disk contained the system software and the mail logging area (/var/log/maillog). The other disk was used as the mail spool area. This separation of mail log and mail spool areas greatly improved performance during testing. The Linux loader configuration file used on the Mail Filter Relay System is included in Appendix C.

Figure 4. Compaq Test Configuration



2.3 Test Results

The following table summarizes the test results.

Table 1. Test Results

Mail Filter Relay System RAM	"Good" Messages per second	"SPAM" Messages per second	%CPU Utilization
256 MB – 1 CPU	17.65	35.64	95 – 100 %
512 MB – 1 CPU	19.64	46.44	95 – 100 %
1024 MB – 1 CPU	20.70	46.75	95 – 100 %
256 MB – 2 CPU	21.89	51.58	95 – 100 %
512 MB – 2 CPU	29.20	56.66	95 – 100 %
1024 MB – 2 CPU	33.09	58.50	95 – 100 %

With a 50-client test load, there were never more than 100 sendmail daemons running at any one time. Each daemon took approximately 1.1 MB of space when loaded in memory. Additional memory was utilized by the system for buffer caching. The performance gains from the extra memory (more than 256 MB) were consistent but minimal.

The biggest performance gains came from the following areas:

- Adding a second CPU
- Configuration of sendmail (as detailed in Chapter 3)
- Utilizing the 2.2.9 linux kernel, instead of the 2.0.36 kernel
- Using separate fast disks for mail logging, and the mail spool area
- Using more than one NIC card

2.4 Sizing Considerations

Table 1 gives the performance that can be expected of a single ProLiant 1850R system operating as a mail relay/filter. To use this information for sizing purposes, we need to work through the arithmetic of anticipated load and size the solution to handle the anticipated load plus a safety margin. All of our solutions are sized for (N+1) systems, where N is the number of systems needed to handle the load profile. The additional system provides high availability in the solution and during normal operation, provides additional capacity (i.e. a safety margin).

Anticipated load can be broken down into three components:

- Rejecting UBE relay traffic (i.e. rejecting attempts to use your system as a relay)
- Rejecting spam targeted at your subscribers
- Relaying normal “good” message traffic to your mail servers

The most important feature of a mail filter solution is the ability to prevent spammers from utilizing your mail system as a relay for Unsolicited Bulk Email. The techniques described in Chapter 1 accomplish this goal. Spammers normally test potential relay systems before attempting to use them to send thousands of messages. Consequently, if your system is protected (cannot be used as a relay), the spammer will look elsewhere, and you need not be concerned about load implied by rejecting thousands of relay attempts.

Sizing systems to reject spam targeted at your subscribers is less concrete. The only real throttle to someone trying to “spam” all of your users at once, is your upstream network bandwidth. For example, a spam attack with 4 kilobyte messages could saturate a T1 line (1.544 Mbps) with 48.25 messages/second. A similar attack could saturate a T3 line (44.736 Mbps) with 1,398 messages/second. As you can see from the possible messages/second number, it isn’t practical to size based on your ability to send rejection messages. A “spam storm” with these kinds of loads is more like a denial of service attack. A more practical approach is to just rely on the mail relay’s ability to reject additional SMTP connections. The consequence of this can be a delay in receipt of normal mail messages, since they will temporarily be queued by the sending mail systems.

Compaq sizes the UBE mail filter solution based on normal “good” message loads. The best way to determine normal message load is to actually measure or count the number of messages received and sent by your subscribers during peak hours. If you do not have a current subscriber base or lack instrumentation on your mail servers, you can still estimate mail traffic based on subscriber profiles. For our mail server testing Compaq uses the profiles below which we feel are representative of residential and business subscribers.

Table 2. Mail Server Testing Profiles

Connection Type	**Residential Dialup**	**Business Leased**
Number of peak hours	6 (e.g. 6-12pm)	2 (e.g. 9-11am)
Average session length (min.)	30	NA
Subscriber to modem ratio	10	NA
Number of messages received per subscriber per day	5	40
Number of these messages received during peak hours	2 (i.e. 40%)	10 (i.e. 25%)
Number of messages sent per subscriber per day	1	10
Number of these messages sent during peak hours	1 (i.e. 100%)	4 (i.e. 40%)

Using these profiles, an example 10,000 subscriber residential peak load at a 10:1 modem ratio would be:

0.56 msgs/sec sent = (1 message * 1,000 active sessions * 2 sessions per hour) / 3600

0.93 msgs/sec rcvd = (10,000 subscribers * 5 msgs/day * 0.4 rcvd at peak) / (6 hours * 3600)

and an example 10,000 subscriber business peak load would be:

5.56 msgs/sec sent = (10 msgs * 10,000 subscribers * 0.4 sent at peak) / (2 hours * 3600)

13.89 msgs/sec rcvd = (10,000 subscribers * 40 msgs/day * 0.25 rcvd at peak) / (2 hours * 3600)

2.4.1 Sizing Recommendation for an ISP Residential Customer Load

The sizing information for an ISP residential subscriber mail filtering solution is included in Table 3. Our recommended sizing is based on the “good” (i.e. non-spam) message numbers. The base systems recommended for the mail filter relay solution are Compaq ProLiant 1850R systems, starting with 1 CPU, 256 MB of RAM, two NIC cards, and two 9.1 GB 10,000 RPM ultra wide SCSI disks.

Any solution requiring high availability requires a minimum of two systems. Each one is defined in mail exchange (MX) records as responsible for processing mail for our supported mail domains. This allows for an alternate system to take over the mail filtering and relaying functions from a failed system, or during scheduled down time for a mail filter system.

Table 3. Sizing Recommendations for a Residential Customer Solution

	Messages/second rates at peak			
Subscriber Numbers	**Sent**	**Received**	**Total**	**Required Number of Proliant Systems**
10,000	0.56	0.93	1.48	2 (256MB, 1-CPU)
50,000	2.78	4.63	7.41	2 (256MB, 1-CPU)
100,000	5.56	9.26	14.81	2 (256MB, 1-CPU)
150,000	8.33	13.89	22.22	2 (512MB, 2-CPU)
200,000	11.11	18.52	29.63	2 (1024MB, 2-CPU)

2.4.2 Sizing Recommendations for a Commercial Business Solution

The sizing information for a commercial business mail filtering solution is included in Table 3. Our recommended sizing is based on the “good” (i.e. non-spam) message numbers. The base systems recommended for the mail filter relay solution are Compaq ProLiant 1850R systems, starting with 1 CPU, 512 MB of RAM, two NIC cards, and two 9.1 GB 10,000 RPM ultra wide SCSI disks.

Any solution requiring high availability requires a minimum of two systems. Each one is defined in mail exchange (MX) records as responsible for processing mail for our supported mail domains. This allows for an alternate system to take over the mail filtering and relaying functions from a failed system, or during scheduled down time for a mail filter system.

Table 4. Sizing Recommendations for a Commercial Business Solution

	Messages/second rates at peak			
Subscriber Numbers	**Sent**	**Received**	**Total**	**Required Number of Proliant Systems**
10,000	5.56	13.89	19.44	2 (512MB, 1-CPU)
20,000	11.11	27.78	38.89	3 (512MB, 1-CPU)
30,000	16.67	41.67	58.33	3 (512MB, 2-CPU)
40,000	22.22	55.56	77.78	4 (512MB, 2-CPU)

3 Installation and Configuration

The first step is to build a sendmail configuration that supports the anti-spam features you will need. Most of the major Linux distributions ship with a version of sendmail. Compaq currently recommends using version 8.9.3 of sendmail.

3.1 Obtaining Sendmail V8.9.3

If you are not currently running sendmail V8.9.3, you will need to obtain the sources. Get the sendmail V8.9.3 sources and unpack them in the /usr/src directory as shown below. The central repository for the open source sendmail is on ftp://ftp.sendmail.org. Additional information on sendmail V8 and its anti-spam features can be found at http://www.sendmail.org.

```
# cd /usr/src
# wget –c ftp://ftp.sendmail.org/pub/sendmail/sendmail.8.9.3.tar.gz
```

You can also use the regular ftp client or a web browser to download the source.

The next step is to unpack the sources. If you already have a /usr/src/sendmail-8.9.3 directory, it is a good idea to rename it. The following command uncompresses and extracts the sendmail sources.

```
# tar zxvf sendmail-8.9.3.tar.gz
# cd sendmail-8.9.3
```

Review the README file to understand sendmail build options.

```
# make
```

Now go into the configuration file source directory. Configuration files are contained in the subdirectory "cf”, with a suffix of “.mc”. They must be run through “m4” (a macro-processing tool included with Linux distributions). The instructions for modifying the configuration file and running it through m4 are in the next section.

```
# cd cf/cf
```

3.2 Modifying the Configuration for Performance and Spam Filtering

Create a file and call it filter-linux.mc. It will be used to enable additional spam filtering functionality, and configuration tuning. The mc file used during our testing is included in Appendix A. The important performance and anti-spam features in this configuration are listed below. Note that these are additional features beyond the default anti-spam features provided by sendmail V8.9.3.

Features to use:

- Change the delivery mode to interactive (confDELIVERY_MODE). This is a synchronous delivery as opposed to the default of delivering asynchronously. Leaving the default results in large message queue build up during peak connection rates, and leaves final message delivery to remote hosts as a background task. Changing to interactive improves throughput, and keeps the queue area empty or small. This is a better choice for high availability. Messages will only be queued if the remote destination system is unavailable.

- Place the directory containing the queue files on its own dedicated disk (QUEUE_DIR). The default queue directory is /var/spool/mqueue. We changed it to a /data/spool/mqueue which resided on it's own dedicated disk.
- Increase the queue load average (confQUEUE_LA) from the default of 8 to 40. When the system load average exceeds LA, just queue messages (that is, don't try to send them).
- Increase the refuse load average (confREFUSE_LA) from the default of 12 to 50. When the system load average exceeds LA, refuse incoming SMTP connections.
- Decrease the maximum recipients per message (confMAX_RCPTS_PER_MESSAGE). The default is no maximum. Compaq recommends lowering this to 500 or lower. A common tactic of spammers, is to send the same message to an extremely large number of recipients. Setting it will allow no more than the specified number of recipients in an SMTP envelope. Further recipients receive a 452 error code (that is, they are deferred for the next delivery attempt).
- Utilize an access control database (access_db). This database is used to accept or reject mail from selected domains, network addresses, or e-mail addresses. For example, you can choose to reject all mail originating from known spammers. Appendix B gives an example of the access database Compaq utilized during testing.
- Only allow relaying by systems in your domain, or systems you specify in an access database (relay_entire_domain). If you are using these relays for outgoing SMTP traffic as well as incoming SMTP traffic, you will want to allow certain systems within your domain to relay messages through your spam filter system.
- Allow relaying for machines for which your mail server is a secondary mail exchanger (relay_based_on_MX). If your server receives a message with a recipient of user@domain.com and domain.com lists your server in its MX records, the mail will be accepted for relay to domain.com. This will stop spammers from using your host to relay spam. It will not stop outsiders from using your server as a relay for their site (that is, they set up an MX record pointing to your mail server, and you relay mail addressed to them without any prior arrangement).
- Decide if you would like to utilize the black list feature (blacklist_recipients). You can add entries to the access database map for local users, hosts in your domains, or addresses in your domain which should not receive mail:

Badlocaluser	550 Mailbox disabled for this username
host.mydomain.com	550 That host does not accept mail
user@otherhost.mydomain.com	550 Mailbox disabled for this recipient

  This would prevent a recipient of badlocaluser@mydomain.com, any user at host.mydomain.com, and the single address user@otherhost.mydomain.com from receiving mail. Enabling this feature will keep you from sending messages to all addresses that have an error message or REJECT as value part in the access map.
- Decide if you would like to utilize a real time blacklist database (rbl). There is a “Real-time Blackhole List” run by the MAPS project at http://maps.vix.com/. This is a database maintained in a domain name service (DNS) of spammers. This will cause sendmail to reject mail from any site in the Real-time Blackhole List database. You can specify an alternative RBL name server to contact by specifying an additional argument to the FEATURE.

3.3 Generating the Configuration File

Create a domain file in the cf/domain directory. For example, if your isp or business domain is called isp.com, use the following commands:

```
# cd cf/domain
# cp generic.m4 isp.com.m4
```

Note that this domain file name is referenced in the mc macro file (see Appendix A, argument for the DOMAIN keyword).

Now generate your configuration file to be used by sendmail.

```
# m4 ../m4/cf.m4 filter-linux.mc > linux.cf
```

You now have a configuration file called linux.cf.

3.4 Creating the Access Database

Create the access database described in Section 3.2 . Use this to specify senders whose mail you want to reject.

```
# cd /etc/mail
```

Create an access database, named access, using your favorite editor. An example access database can be found in Appendix B. You can add to this database over time, adding additional spammers, as you identify them. The following command turns the ASCII version of your access data into a quick lookup database that can be utilized by sendmail:

```
# makemap hash /etc/mail/access < /etc/mail/access
```

This will generate a hashed database from our human readable access database.

3.5 Testing the New Configuration

You can test this new sendmail configuration without stopping your running mail configuration. When you ran the make command in Section 3.1 , you generated a new sendmail image in the src/obj.Linux.2.2.0.i686 directory. You can test this new image and your new configuration file by using the following commands:

```
# cd /usr/src/sendmail-8.9.3
# ./src/obj.Linux.2.2.0.i686/sendmail -oQ/usr/tmp -C./cf/cf/linux.cf -bt
```

This puts you into test mode, and allows you to test the results of sending/receiving messages from specific users or domains. You can find further information on debugging/testing at http://www.harker.com/sendmail/sendmail-tips.html. A useful test and debug tool is the mail program with the –v option (that is, mail –v test@isp.com).

3.6 Installing the New Images and Configuration

Once you have successfully tested your new version, and its new configuration, you can install the software. Follow these steps:

1. Stop any existing running versions of sendmail.

```
# /etc/rc.d/init.d/sendmail stop
```

2. Install the new version and configuration file.

```
# cd /usr/src/sendmail-8.9.3
# make install
```

3. Copy the new configuration file to the configuration file directory.

```
# cp /usr/src/sendmail-8.9.3/cf/cf/linux.cf /etc/sendmail.cf
```

4. Restart sendmail.

```
# /etc/rc.d/init.d/sendmail start
```

You are now running the sendmail 8.9.3 image with your new anti-spam configuration.

4 Managing a Mail Filter Relay System

There are collections of management tools that allow you to monitor and manage your mail filter relay systems. They include system applications like logrotate, mailstats, ifconfig, mailq, and the sendmail command. Compaq also recommends frequent monitoring of the mail log files located in /var/log and called maillog.

4.1 Mail Statistics

The mailstats utility displays the current mail statistics. This includes the number of messages and number of bytes from the mailer, the number of messages and number of bytes to the mailer, and the number of rejected or discarded messages.

4.2 Network Interface Card Monitoring

The ifconfig utility can be used to monitor the amount of traffic passing through your NIC cards. It can also warn you of excessive errors and collisions. This lets you know if there are problems with your NIC cards or LAN configuration.

4.3 Mail Log Files

The logrotate command is usually run from a cron job and takes care of rotating log files in the /var/log directory. Compaq recommends rotating log files before they become too large. Large log files dramatically slow down performance and increase CPU utilization of the system-logging daemon (syslogd). Compaq recommends rotating log files often, and when the size is greater than 50 KB.

The maillog file is a readable file located in the /var/log directory. Compaq recommends running the sendmail program with the default level of logging (log level of 9). The log files can be perused to identify any mail server problems.

The V8 organization of logging consists of these levels:

0	No logging
1	Only serious system failures and security problems.
2	Communication failures, like lost connections, or protocol failures.
3	Malformed addresses.
4	Malformed queue control filenames.
5	A record of each message received.
6	SMTP verify attempts and messages returned to the original sender.
7	Delivery failures, excluding mail deferred because of a temporary lack of a needed resource.
8	Successful deliveries.
9	Mail deferred because of a temporary lack of a needed resource.
10	Each key as it is looked up in the database, and the result of each database lookup
11-98	Debugging information logging.

4.4 The Mail Queue

Sometimes a host cannot process a message immediately. It may be down or overloaded, causing it to refuse connections. The sending host is then expected to save this message in its mail queue and attempt to deliver it later.

Under normal conditions the mail queue will be processed transparently (default is every 15 minutes). You may find that manual intervention is necessary. For example, if a major host is down for a period of time the queue may become extremely large.

4.4.1 Printing the Queue

Print the contents of the mail queue using the mailq command.

```
# mailq
```

This will produce a listing of the queue IDs, the size of the message, the date the message entered the queue, and the sender and recipients.

4.4.2 Forcing the Queue

In some cases, you may find that a major host going offline for a couple of days may create a prohibitively large queue. This will result in sendmail spending an inordinate amount of time processing the queue. This situation can be fixed by moving the queue to a temporary place and creating a new queue. The old queue can be run later when the offending host returns to service.

To do this, it is acceptable to move the entire queue directory as follows:

```
# cd /data/spool
# mv mqueue omqueue; mkdir mqueue; chmod 700 mqueue
```

You should then kill the existing daemon (since it will still be processing in the old queue directory) and create a new daemon.

To run the old mail queue, run the following command:

```
# /usr/sbin/sendmail -oQ/data/spool/omqueue -q
```

The -oQ flag specifies an alternate queue directory and the -q flag says to just run every job in the queue. You can use the -v flag to watch what is going on.

When the queue is finally emptied, you can remove the directory as follows:

```
# rmdir /var/spool/omqueue
```

4.5 Maximum Message Size

To avoid overflowing your system with a large message, the MaxMessageSize option can be set to an absolute limit on the size in bytes of any one message. The default is infinite. Compaq recommends setting this to 10 MB or less.

4.6 Additional Pointers

Additional installation and management information can be found in the guide located at:

http://www.sendmail.org/%7Eca/email/english.html

Additional information on junk mail filtering at the Mail User Agent, or delivery agents can be found at:

http://spam.abuse.net/tools/mailblock.html

Appendix A: sendmail Macro File

The sendmail macro file (filter-linux.mc) is used to generate the sendmail configuration (sendmail.cf). The sendmail macro file contains the following:

```
divert(0)dnl
VERSIONID(`@(#)linux sendmail   8.9 (Berkeley) 5/19/98')
OSTYPE(linux)dnl
DOMAIN(isp.com)dnl
define(`LOCAL_MAILER_PATH',`/usr/bin/procmail')dnl
define(`LOCAL_MAILER_ARGS',`procmail -a $h -d $u')dnl
define(`confDELIVERY_MODE',`i')dnl
define(`confQUEUE_LA',`50')dnl
define(`confREFUSE_LA',`50')dnl
define(`confMAX_RCPTS_PER_MESSAGE',`500')dnl
define(`QUEUE_DIR',`/data/spool/mqueue')dnl
MAILER(procmail)dnl
MAILER(smtp)dnl
FEATURE(use_cw_file)dnl
FEATURE(virtusertable, `hash /etc/mail/virtusertable')dnl
FEATURE(`relay_entire_domain')
FEATURE(`relay_based_on_MX')
FEATURE(`access_db', `hash /etc/mail/access')
FEATURE(`blacklist_recipients', `hash /etc/mail/blacklist_recipients_local')
FEATURE(`rbl')
```

Appendix B: Access Control Database

The following information explains the keys and values for the access control database. The keys are e-mail addresses, domain names, and network numbers. The value part of the access database can contain one of the following:

OK	Accept mail even if other rules in the running ruleset would reject it, for example, if the domain name is unresolvable.
RELAY	Accept mail addressed to the indicated domain or received from the indicated domain for relaying through your SMTP server. RELAY also serves as an implicit OK for the other checks.
REJECT	Reject the sender/recipient with a general-purpose message.
DISCARD	Discard the message completely using the $#discard mailer. This only works for sender addresses (i.e., it indicates that you should discard anything received from the indicated domain)
### any text	where ### is an RFC 821 compliant error code and "any text" is a message to return for the command.

It is a good idea to send back a message that tells the rejected mail sender why they were rejected, and a contact e-mail address (on a non-spam filtered system) where they can request to be removed from the reject database.

The following example access database was used during testing at Compaq:

spammer@spam.com	REJECT
cyberspammer.com	550 Die evil spammer
alpha2.isp.com	REJECT
test@merrimack.isp.com	550 Mailbox disabled for local user test

In the example access database above, all mail from spammer@spam.com, and alpha2.isp.com will be immediately rejected. Mail from cyberspammer.com domain will be rejected with a message "Die evil spammer, " and mail destined for user test@merrimack.isp.com will be rejected with a message of "Mailbox disabled for local user test."

The whole access database utilized during testing (with 50 entries) is presented as follows:

```
spammer@aol.com        REJECT
cyberspammer.com       550 Die evil spammer
alpha2.ispworks.com    REJECT
test@merrimack.ispworks.com    550 Mailbox disabled for local user test
user1@cyberspammer.com  REJECT
user2@cyberspammer.com  REJECT
user3@cyberspammer.com  REJECT
user4@cyberspammer.com  REJECT
user5@cyberspammer.com  REJECT
```

```
user6@cyberspammer.com  REJECT
user7@cyberspammer.com  REJECT
user8@cyberspammer.com  REJECT
user9@cyberspammer.com  REJECT
user10@cyberspammer.com REJECT
user11@cyberspammer.com REJECT
user12@cyberspammer.com REJECT
user13@cyberspammer.com REJECT
user14@cyberspammer.com REJECT
user15@cyberspammer.com REJECT
user16@cyberspammer.com REJECT
user17@cyberspammer.com REJECT
user18@cyberspammer.com REJECT
user19@cyberspammer.com REJECT
user20@cyberspammer.com REJECT
user21@cyberspammer.com REJECT
user22@cyberspammer.com REJECT
user23@cyberspammer.com REJECT
user24@cyberspammer.com REJECT
user25@cyberspammer.com REJECT
user26@cyberspammer.com REJECT
user27@cyberspammer.com REJECT
user28@cyberspammer.com REJECT
user29@cyberspammer.com REJECT
user30@cyberspammer.com REJECT
user31@cyberspammer.com REJECT
user32@cyberspammer.com REJECT
user33@cyberspammer.com REJECT
user34@cyberspammer.com REJECT
user35@cyberspammer.com REJECT
user36@cyberspammer.com REJECT
user37@cyberspammer.com REJECT
user38@cyberspammer.com REJECT
user39@cyberspammer.com REJECT
```

```
user40@cyberspammer.com REJECT
user41@cyberspammer.com REJECT
user42@cyberspammer.com REJECT
user43@cyberspammer.com REJECT
user44@cyberspammer.com REJECT
user45@cyberspammer.com REJECT
user46@cyberspammer.com REJECT
```

Appendix C: Linux Loader Configuration File

The Linux loader configuration file, /etc/lilo.conf, was used during testing. The ether express pro NIC card was forced into full-duplex, 100 Mbps configuration. This file contained the following parameters.

```
boot=/dev/sda
map=/boot/map
install=/boot/boot.b
prompt
timeout=100

image=/boot/vmlinuz-2.2.2
        label=newlinux
        root=/dev/sda5
        append="ether=0,0,0x30,eth0"
        initrd=/boot/initrd-2.0.36-0.7.img
        read-only

image=/boot/vmlinuz-2.2.2
        label=256linux
        root=/dev/sda5
        append="mem=256M ether=0,0,0x30,eth0"
        initrd=/boot/initrd-2.0.36-0.7.img
        read-only

image=/boot/vmlinuz-2.2.2
        label=512linux
        root=/dev/sda5
        append="mem=512M ether=0,0,0x30,eth0"
        initrd=/boot/initrd-2.0.36-0.7.img
        read-only
```