

Fairly-Secure Anti-SPAM Gateway Using OpenBSD, Postfix, Amavisd-new, SpamAssassin, Razor and DCC

By Scott Vintinner

Last Edited 5/6/2004 9:00 AM EDT ([Changelog at bottom](#))

Latest version of this document always available at <http://www.flakshack.com/anti-spam>

Read and post comments [here](#) (note that you don't need to create an account to post).

The old after-queue based version of this doc is available [here](#) as a reference.

Index

[What is this document?](#)

[Where to find help](#)

[Hardware Requirements](#)

[Notes and conventions](#)

[Provide firewall access](#)

[Installing OpenBSD](#)

[Configuring OpenBSD](#)

[Removing Sendmail](#)

[Add User Accounts](#)

[Installing and Configuring Postfix](#)

[Installing Needed Perl Modules](#)

[Installing and Configuring SpamAssassin](#)

[Installing Razor](#)

[Installing DCC](#)

[Installing and configuring Amavisd-new](#)

[Installing Postfix Email Reporting](#)

[What if I want to disable Amavisd-new/SpamAssassin?](#)

[Performance](#)

[Performance Expectations](#)

[Configuring for Multiple Domains](#)

[Notes about running under other OS](#)

[Adding Anti-Virus Support](#)

[Download My Scripts](#)

[Changelog](#)

[Legal Stuff](#)

What is this document?

This document describes how to setup a spam-blocking email gateway based on open source and freely available software. This procedure is designed for a small to medium sized company with a single domain (multiple domains are possible...just not described here in full detail). I will describe how to setup a new computer that is meant to run on your network's DMZ in between the Internet and a corporate email server like Lotus Notes or Microsoft Exchange.

This entire procedure has been developed with security as a primary focus. The operating system is [OpenBSD \(www.openbsd.org\)](http://www.openbsd.org), which is a "Secure by Default" system with a good track record for security. The email MTA is [Postfix \(www.postfix.org\)](http://www.postfix.org) which also has a good record for security and is the easier of the 2

main competitors to the troubled [sendmail](#) program. [Amavisd-new](#) (www.ijs.si/software/amavisd) is the main filter which processes email from postfix and ensures that we don't lose any mail. Amavisd-new is an huge improvement over the original amavis which was a simple virus scanner, and I think it is the best way of implementing [SpamAssassin](#) (www.spamassassin.org). SpamAssassin is the main anti-spam component which works by comparing messages to a ruleset and by using a statistical analysis that is custom built based on your email. In addition to the SpamAssassin spam detection software, we will be using 2 online SPAM databases: [DCC](http://www.rhyolite.com/anti-spam/dcc) (www.rhyolite.com/anti-spam/dcc) and [Vipul's Razor](http://razor.sourceforge.net) (razor.sourceforge.net). These databases work by comparing hashes of our email messages with hashes of known spam. As a final security precaution, we will run all network processes in a restricted-user/chroot environment, so if an attacker were able to compromise one of the modules, the amount of damage they could do would be seriously limited.

There are many different ways to customize the handling of SPAM with amavisd-new. The following directions use Postfix's Before-Queue Content Filtering proxy to immediately reject spam while still connected to the sending server. The sender of the rejected spam will get a postmaster "undeliverable" message from their mail server which includes an error line that we can customize with directions in case of a false-positive. The sender can follow these directions to be added to a "whitelist" of approved email addresses which get to bypass the spam filter. In the end, you get an anti-spam system that requires NO work on your user's part and has a very small percentage of false positives.

Where to find help

I've been receiving an average of 5 new emails per day from people asking me to help debug their setups. In nearly all of these cases, the problem is due to a typo in the configuration files or some minor fluke caused by using a slightly different version of one of the components. Unfortunately I no longer have the time to answer everyone's questions personally.

The best place to go for help is the amavisd mailing list (<https://lists.sourceforge.net/lists/listinfo/amavis-user>). Please search through the archive (<http://marc.theaimsgroup.com/?l=amavis-user>) for answers to your problem before posting a question. Mark Martinec does a very good job of answering most questions quickly, so please don't waste his time. Do some research first and try to ask good questions. I also subscribe to the list, so when posting your question be sure to mention that you are using OpenBSD. Also please do not email huge copies of your config files or debug output unless asked to do so.

If your question is specific to SpamAssassin, DCC, Razor, or Postfix, please refer to their respective home pages (listed above) for more assistance.

If you have a suggestion, or find an error you think I should fix, please email me at scottv@rbh.com or visit the discussion board for this doc [here](#).

Hardware Requirements

Since computers with 256 MB of RAM are increasingly common, I have decided to set the configuration in these directions optimized for a computer with that much memory. If you have more or less memory, you should see the section on [Performance](#) near the end of the document before putting your server into production. This configuration will allow up to 8 simultaneous connections to your server. If you need to support more connections at a time, you will need to add more memory to the system (amavisd is a bit of a memory hog).

Notes and Conventions

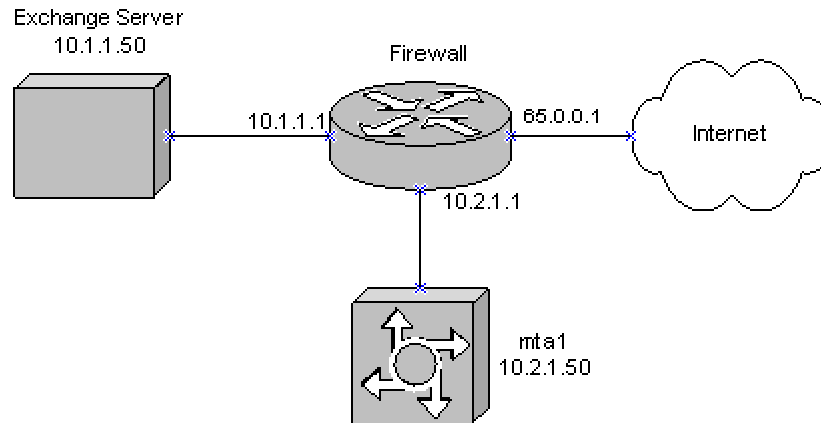
This document is not really meant for the unix newbie. If you have never worked with linux or unix before, you may experience some difficulty with these directions. Specifically, I expect that you will know how to use vi to edit files (here's a [vi cheat sheet](#)). (Actually I've been informed that you can set the EDITOR environment variable to get OpenBSD to use whatever editor you want for vipw and crontab -e, so I guess you can use whatever editor you want.)

Also, you'll want to grab a copy of [PuTTY](#) and [WinSCP](#). These 2 utilities will help you configure your server remotely without the need for a GUI running on the server itself. PuTTY is an awesome SSH client that you can use to talk to your OpenBSD server over a secure link. It has 2 great features: anything you select with your mouse is copied to the Windows clipboard, and right-clicking the mouse anywhere in the window will paste the contents of the clipboard. It works great in insert mode in vi to paste hundreds of lines. My servers are rack-mounts, so usually as soon as I get the network running on the OS, I head back to my PC and pull up PuTTY. WinSCP is a gui file browser for SSH. It will let you easily copy files back and forth between your Windows PC and the OpenBSD server.

Text listed in [blue](#) are commands that you type into the console. Text listed in [red](#) are edits in text files. Text listed in gray tables are full files. The # sign followed by a [blue command](#) indicates a shell prompt (so don't type it). The # sign in configuration files or text edits is used as a comment block, and may be included in the actual files without harming anything.

I have received many many emails from people telling me that I'm going straight to hell for not using [SUDO](#). As you look through my directions, you will see that the directions have you logged in as root during this entire procedure. The "best-practices" for OpenBSD say that you should login as an unprivileged account to do most of the work and use the sudo command whenever root access is required. I'll be the first to admit that I'm not conscientious enough to follow those guidelines 100% of the time. I certainly suggest that you read up on the [sudo command](#) if you are unfamiliar with it. Personally I think there is a time and a place for using sudo and an efficient administrator knows the difference.

One last note: obviously these directions require some customization to fit into your network. Specifically you'll want to replace the sample server names and IP addresses with your own. Here's a little diagram of what we'll be setting up:



For our examples, our internal network numbering uses the 10.0.0.0/8 private domain range. The 10.1.0.0/16 network is our main network. The 10.2.0.0/16 network is our DMZ network. Our pretend public internet class C is 65.0.0.0/24. Additionally, the server we will be setting up will be called **mta1** in these directions. You can call your server whatever you want, just replace it with your name where ever you see it. For simplicity, we'll pretend that our company's domain name is **domain.com**.

Provide Firewall Access

Hopefully you have a firewall at your company. Here's a list of what traffic you will need to allow through your firewall to make all of this work.

Source	Destination	Type	Port	Description
10.2.1.50	10.1.1.50 and any External	TCP	25	Outgoing SMTP
10.1.1.50	10.2.1.50	TCP	25	SMTP from Exchange to our MTA
any External	10.2.1.50	TCP	25	Incoming SMTP
10.2.1.50	any External	UDP	6277	Outgoing from our MTA to the DCC servers
any External	10.2.1.50	UDP	6277	Incoming from DCC Servers to our MTA
10.2.1.50	any External	TCP	2703	Outgoing from our MTA to the Razor servers
10.2.1.50	any External	TCP	7	Outgoing ping from our MTA to the Razor servers
10.2.1.50	any External	TCP	22	Outgoing SSH (used to update our source code from the OpenBSD servers using CVS)
10.2.1.50	any External	TCP	21	Outgoing FTP (so we can download files that we need)
10.2.1.50	any External	TCP	80	Outgoing HTTP (also so we can download files)

Additionally mta1 needs to have access to a DNS server. In my configuration the DNS is on the DMZ so no extra firewall rules were required.

Installing OpenBSD

The [OpenBSD FAQ Chapter 4](#) provides an excellent step by step explanation of how to install the latest version of OpenBSD, so I won't repeat it all here. I'll just give some minor tips:

- The hardest part about an initial OpenBSD install is doing the disk partitioning. Since we won't be multi-booting (between windows or some other OS and OpenBSD), our job will be much easier. Just follow the directions in the FAQ step by step and you'll make it through...trust me. When setting up your disk partitions, use these guidelines. The sizes in the FAQ are really the absolute minimums (sorta like how Microsoft says you can run NT on a 386). Disk space is cheap, so if you are running this server on an old computer, do yourself a favor and spend \$100 to get at least a 20 gig hard drive.

wd0a - / (root) 500M - This partition is where the / /etc /root /dev directories will be kept.

wd0b - (swap) - give it at least as much space as you have in RAM, so if your computer has 256 MB RAM, make the swap at least 256M.

wd0d - /tmp 300M

wd0e - /var 8G - Again disk space is cheap, so give the var directory lots of space. All of the mail queues, local mailboxes, logs and nearly everything else get stored here.

wd0f - /usr 2G - You absolutely must have at least 2 gigs in the /usr volume. If you ever need to patch your operating system (and you will) you'll need this space to store all the source code and compiled files.

wd0g - /home 1G - We won't do much with the home directories on this computer since there really won't be any local users other than the administrator. One gig of space will be plenty.

Hopefully after this you'll have plenty of space left over. If you ever need extra space somewhere else, you can create a new volume and mount it wherever you need it. As far as I know, there is no Partition Magic for OpenBSD, so make sure you give yourself enough space now.

- Do yourself a favor and configure the network through the OpenBSD install even though you will be installing from CD.
- When asked to select which filesets you want, accept the defaults. I personally don't recommend installing X on the server because it will make it needlessly more complicated. I'm not a big fan of installing X with KDE or GNOME on servers. The way I view it is that the fewer programs you have installed, the fewer programs you have to worry about updating with new versions and security patches.
- When choosing your time zone, look for the version that supports daylight savings time (if you are in North America): EST5EDT, MST7MDT, CST6CDT, etc. otherwise your server's clock will be off for half the year.

Configuring OpenBSD

1. Once OpenBSD has been installed, head back to your desk and pull up PuTTY to remotely access the system through SSH. It is much easier to do it from your desk when you can have multiple remote sessions going and can cut and paste using PuTTY.
2. (OPTIONAL) The first thing I like to do on any OpenBSD box that I have is change the default root shell from csh to ksh to get command history and tab expansion. The command history means that you can press the up-arrow key to access previously listed commands. The tab expansion allows you to press the tab key to fill out lengthy filenames. For example if you want to change directory to /root/hereisareallylongdirectoryname, you could type cd /root/her and press the TAB key. The system will then fill in the rest of the filename for you. To make this change, run vipw and change the default shell for root to /bin/ksh.

```
# vipw  
root:$asdfasdfqwerqwer$%.:0:0:daemon:0:0:MTA1 Sendmail  
&:/root:/bin/ksh
```

After making this change and saving the file, go ahead and open up another SSH session and login as root just to test. If you edited the file incorrectly, root would not be able to login, so it is important to test it before you logout of the original session. If you are unable to login, just switch back to the original session and fix the problem.

NOTE: I've also had it pointed out to me that /bin/sh also provides these same features and is also POSIX compliant.

3. Setup your system to make sure that it keeps correct time. You can do this one of 2 ways. The easiest way is to setup the rdate program to run periodically to update the time from a remote NTP server. The not-so easy, but more accurate way is to setup NTP on this computer. Personally I used NTP on my server, but we'll save the directions for setting up NTP for another day. To setup rdate to run on a schedule, run the crontab program to edit root's scheduler.

```
# crontab -e
```

Now add the following lines to the bottom of the file:

```
# update time with ntp server  
0 0 * * * /usr/sbin/rdate -ncva 128.2.136.71 |logger -t NTP
```

This line will get cron to update the time on the computer every day at midnight. The computer will contact the Carnegie Mellon public NTP server. Please take a look at this [list of public NTP servers](#) and select a server that is close to you. The |logger -t NTP line makes sure that any adjustments are added to the /var/log/messages log file. Remember that it would be impolite and inefficient for you to configure multiple servers at your site to all talk directly to one of the public NTP servers. If you find yourself doing that, it may be time to setup your own NTP server. Note: you may want to run rdate once without the -a parameter so that it adjusts your time initially. The -a

parameter tells rdate to change the time very slowly, even if it is off by a lot, so if your system clock is currently incorrect it could take hours or days before it is synchronized.

4. Create a `.forward` file in root's home directory. The email system will use this file to forward any email for the root account to your personal (exchange) account. This file should be a one line file with just a single email address in it.

```
# vi /root/.forward
```

5. Edit the hosts file to include the local address for our computer. The installs for some of the perl modules will run some network tests that won't work properly without this setting.

```
# vi /etc/hosts
```

Add the following line to the bottom of the file:

```
10.2.1.50 mta1.domain.com mta1
```

Removing Sendmail

1. Delete the old sendmail executable. Note that Postfix will install a program called sendmail in the same location for compatibility. Also note that on OpenBSD, the real sendmail executable will still be available in `/usr/libexec/sendmail/sendmail`.

```
# rm /usr/sbin/sendmail
```

2. Remove sendmail queue runner command in crontab

```
# crontab -e
```

Look for the lines that look like this and remove them or comment them out with a `#`

```
# sendmail clientmqueue runner
```

```
*/30 * * * * /usr/sbin/sendmail -L sm-msp-queue -Ac -q
```

3. Edit the `/etc/rc.conf.local` file to prevent sendmail from starting when the computer boots. Change:

```
sendmail_flags = NO
```

If this file doesn't exist, you can create it and simply include that single line in it. This file overrides changes in the `rc.conf` file. Keeping the settings in this override file makes it easier when you upgrade, since you won't have to manually merge your `rc.conf` with a new version.

4. Kill off any existing sendmail process.

```
# ps -ax |grep sendmail
```

```
# kill `head -1 /var/run/sendmail.pid`
```

NOTE: these directions assume that this is a new server that is not currently being used. If your server was previously running sendmail or some other MTA, you will want to check the postfix `INSTALL` file for better instructions on migrating to postfix.

Add User Accounts

When possible, you never want to run anything as root. The root account has unrestricted access to everything, so if there is a bug or security hole in anything you run as root, it could potentially compromise your entire system. To further tighten the security of our system, we will run different modules as different user accounts.

Use `vipw` to edit the password file. You could also use the `adduser` or `useradd` commands, but this is the easiest way for us. Go to the bottom of the file and add the following lines:

```
postfix:*:2000:2000::0:0:Postfix Mail Daemon:/var/empty:/sbin/nologin
amavisd:*:3000:3000::0:0:Amavis Mail Scanner
Daemon:/var/amavisd:/sbin/nologin
```

Next we want to add the group accounts for our new users. To do this, edit the `/etc/group` file. Add the following lines (preferably in numeric order with any other groups in the list):

```
postfix:*:2000:
postdrop:*:2001:
amavisd:*:3000:
```

Here's a quick explanation of what we'll use the new accounts for: The postfix account is the account that postfix will run under, and likewise the amavisd account is the account that the amavisd program will run under. Running these programs in non-root accounts is good for security, since if an attacker compromises either program, they are still limited by those accounts. BTW, the `*` in the first section is in place of where a normal password would be. The star tells the operating system that the user cannot login. The `/sbin/nologin` is the user's shell that is launched when they login...which in this case is another indicator to the operating system that these users shouldn't be able to login.

Install and Configure Postfix

You can install Postfix from the OpenBSD port collection using the directions included in the [Ports & Packages FAQ](#). The postfix port is located at `/usr/ports/mail/postfix`. You may want to double-check which version you are getting with that method, as it must be version 2.1.0 or higher. Below I describe how to install Postfix directly from the source.

1. Download latest version of postfix to `/root` and compile it. Check the www.postfix.org website for the latest version since it is updated frequently.

```
# cd /root
# ftp http://postfix.energybeam.com/source/official/postfix-2.1.0.tar.gz
# tar -zxvf postfix-2.1.0.tar.gz
# cd postfix-2.1.0
# make
```

2. Setup the default postfix config by running make install and answering questions.

```
# make install
```

Answers (most will be the defaults)

```
install_root: /
tempdir: [/root/postfix-2.1.0] or whichever directory the source is in
configdir: /etc/postfix
daemondir: /usr/libexec/postfix
commanddir: /usr/sbin
queuedir: /var/spool/postfix
sendmailpath: /usr/sbin/sendmail
newaliases path: /usr/bin/newaliases
mailq path: /usr/bin/mailq
mail owner: postfix
setgidgroup: postdrop
man page dir: /usr/local/man
sample dir /etc/postfix
readme no
```

3. Setup Postfix CHROOT. Here we need to copy the necessary system files to the root directory where postfix will be restricted.

```
# mkdir /var/spool/postfix/etc
# cd /etc
# cp localtime services hosts resolv.conf /var/spool/postfix/etc
```

4. Setup Aliases File. Here we want to setup the aliases file (which provides aliases for commonly used accounts). If we had one from sendmail we could use it. Otherwise we'll just copy the sample one. Basically the aliases file allows us to redirect any email for non-email receiving accounts (like www) to the root account. Root's email is forwarded to us using the /root/.forward file we setup above.

```
# cp /etc/postfix/aliases /etc/
```

Run newaliases to hash the textfile into aliases.db (which is what postfix actually uses).

```
# newaliases
```

5. Next we want to edit the postfix configuration file (/etc/postfix/main.cf) to customize it for our email system. Search for the following entries to set. Note that many will not be listed in the default file.

```
#
# >>>>>>>> You must reload postfix after editing this file
# >> NOTE >> to do this use the command:
# >>>>>>>> # postfix reload
#

queue_directory = /var/spool/postfix
daemon_directory = /usr/libexec/postfix
```

```

mail_owner = postfix

myhostname = mtal.domain.com
mydomain = domain.com
myorigin = $myhostname
mydestination = $myhostname, localhost.$mydomain, $mydomain,
mail.$mydomain
mynetworks = 10.0.0.0/8, 65.0.0.0/24, 127.0.0.0/8

local_recipient_maps =
biff = no
empty_address_recipient = MAILER-DAEMON
queue_minfree = 120000000
message_size_limit = 80000000
mailbox_size_limit = 1000000000

smtpd_banner = $myhostname ESMTX

transport_maps = hash:/etc/postfix/transport
local_transport = local

smtpd_recipient_restrictions = permit_mynetworks,
reject_unauth_destination, reject_non_fqdn_recipient
#some people also add reject_non_fqdn_sender (but I have clients that
break this rule that I need to receive from)

notify_classes = protocol,resource,software

# Install Time Configuration
sendmail_path = /usr/sbin/sendmail
newaliases_path = /usr/bin/newaliases
mailq_path = /usr/bin/mailq
setgid_group = postdrop
manpage_directory = /usr/local/man
sample_directory = /etc/postfix
readme_directory = no
html_directory = no

```

6.

Explanation of Postfix configuration settings:	
queue_directory	Postfix's work directory. Where all the mail will be temporarily stored until it is delivered.
daemon_directory	Specifies the location of all the postfix programs.
mail_owner	Specifies the user account that will own the mail queues.
myhostname	The name of this computer including the domain part. This is used when adding received by headers in email messages.
mydomain	This specifies the domain of this current computer.
myorigin	This name is added to locally

	<p>originating email. So if you sent yourself a message from root, it would appear to come from root@mtal.domain.com.</p>
mydestination	<p>This setting tells postfix what domains it will accept email for. Please note this should not be used for virtual domains, or for backup MX hosts. In our case, we set it so it would receive mail for something@mtal.domain.com, something@localhost.domain.com, something@domain.com, and something@mail.domain.com</p>
mynetworks	<p>This setting tells postfix what networks it should consider local. In other words, computers connecting from any of these networks will be able to relay mail, etc. In our case, we put 127.0.0.0 (for localhost), 65.0.0.0 (for any other computers on our external network), and 10.0.0.0 (for any internal computer).</p>
local_recipient_maps	<p>This setting tells postfix where to find the names of local users to accept mail for. We just want to leave this blank (note that removing it all together will cause errors).</p>
biff	<p>This setting tells postfix not to use the biff program to let local users know that they have new email.</p>
empty_address_recipient	<p>This setting is the destination for undeliverable mail from <></p>
queue_minfree	<p>This setting tells postfix not to accept any messages for delivery if there are less than 120 megs of disk space available. This number should be 1.5 times the message_size_limit or you will get an error message.</p>
message_size_limit	<p>This sets the maximum size of a message. Messages larger than 80 megs will be rejected. You can increase or decrease this based on your own server requirements.</p>
mailbox_size_limit	<p>This sets the maximum size of local mailbox files. We set it to 100 megs, although it should never reach this high because our only local mailboxes are spam and</p>

	notspam
smtpd_banner	This is the banner that is displayed to connecting computers. It is a good security practice to give as little information as possible. I've included just the essentials.
transport_maps	This setting tells postfix where to find the transport information. The transport file is where we tell Postfix where to route certain mail. In our case, this file is where we tell Postfix that mail for domain.com should be delivered to our exchange server.
local_transport	This setting tells postfix that all local mail should be delivered using the local delivery agent.
smtpd_helo_restrictions, smtpd_sender_restrictions, smtpd_recipient_restrictions	These settings are used to deny access to postfix based on the HELO command, the sender, or the recipient. The recipient restrictions settings are used to prevent our mail server from being used as an open relay. As configured, the helo and sender restrictions are left open. If you want to play around with these you can, just look them up in the postfix documentation. Normally these settings can be used to help block SPAM. Unfortunately they depend on the assumption that all legitimate senders have their systems correctly configured. In my experience this is never the case, so these settings are more trouble than they are worth. Like the RBL lists, I found that enabling these settings meant that I spent too much time teaching other mail administrators how to correctly configure their systems (Note that we'll use Realtime Blackhole lists (RBLs) in SpamAssassin, not in Postfix because there positive hits will simply result in higher spam scores.)
notify_classes	This setting tells postfix to send all sorts of notifications to the postmaster email account. Here's a list of the available options: bounce: Send the postmaster copies

	<p>of the headers of bounced mail. 2bounce: Send undeliverable bounced mail to the postmaster. delay: Send the postmaster copies of the headers of delayed mail. policy: Send the postmaster a transcript of the entire SMTP session when a client request was rejected because of (UCE) policy. If you enable this, you will get 1 email for every spam message that was rejected by amavisd. Good for testing, not good for production. protocol: Send the postmaster a transcript of the entire SMTP session in case of client or server protocol errors. resource: Inform the postmaster of mail not delivered due to resource problems. software: Inform the postmaster of mail not delivered due to software problems.</p>
Install Time Configuration	<p>This section holds the settings we used when we installed postfix. We keep them in the config file so that future upgrades will be easier.</p>

7. Edit `/etc/postfix/master.cf` file. Change the `CHROOT` setting for all the base postfix services from 'n' to 'y'. Adjust the `maxproc` for the `smtp` line and include the added parameters. Also add the `127.0.0.1` daemon as indicated.

```

#=====
# service          type  private  unpriv  chroot  wakeup  maxproc  command +
#                  (yes)   (yes)   (yes)   (never) (100)    args
#=====
#This is the before-filter smtpd...it passes content to amavisd on port 10024
smtp              inet   n        -       y        -        8        smtpd
                 -o smtpd_proxy_filter=127.0.0.1:10024
                 -o smtpd_client_connection_count_limit=4
pickup           fifo   n        -       y        60       1        pickup
cleanup          unix   n        -       y        -        0        cleanup
qmgr             fifo   n        -       y        300      1        qmgr
rewrite          unix   -        -       y        300      -        trivial-
                 rewrite
bounce           unix   -        -       y        -        0        bounce
defer            unix   -        -       y        -        0        bounce
flush            unix   n        -       y        1000?    0        flush
proxymap         unix   -        -       n        -        -        proxymap
smtp             unix   -        -       y        -        -        smtp
relay            unix   -        -       y        -        -        smtp
showq            unix   n        -       y        -        -        showq
error            unix   -        -       y        -        -        error
local            unix   -        n        n        -        -        local
virtual          unix   -        n        n        -        -        virtual
lmtp             unix   -        -       y        -        -        lmtp
trace            unix   -        -       y        -        0        bounce
verify           unix   -        -       y        -        1        verify

# This is the after-filter smtpd, it listens on port 10025 to receive mail from
# amavisd
127.0.0.1:10025  inet   n        -       y        -        -        smtpd
                 -o smtpd_authorized_xforward_hosts=127.0.0.0/8
                 -o smtpd_client_restrictions=
                 -o smtpd_helo_restrictions=
                 -o smtpd_sender_restrictions=
                 -o smtpd_recipient_restrictions=permit_mynetworks,reject
                 -o smtpd_data_restrictions=
                 -o mynetworks=127.0.0.0/8
                 -o receive_override_options=no_unknown_recipient_checks

```

8. I won't bother explaining this file in too much detail. Basically this file is used by the master program to figure out how to run each of the individual postfix processes. The first smtp line is the smtp daemon that listens for incoming connections. As the message is coming in, it is passed through to the proxy filter (amavisd). Note that the maxproc column specifies the maximum

number of instances of that program to run (to prevent overwhelming the memory on the PC).

The `smtpd_client_connection_count_limit` prevents the same remote server from connecting more than once to your server (and hogging all your connections). You should set this to be around half of the setting of the `smtpd_maxproc`.

The last line (with the `127.0.0.1`) describes the listener service to which `amavisd` passes back a good message for acceptance into the queue. Since this daemon is only used by `amavisd`, we add some extra config lines to make it more secure and avoid doing some checks that may already have been done (in other words, make it faster).

NOTE: using the Before-Queue filter method is not recommended for very high-traffic ISPs, since you can only have as many connections at one time as you have `amavisd` processes. For example, under a default postfix config, you could have up to 100 connections at a time. Under this configuration, you can only have 8 connections at a time. Further incoming connection attempts will be rejected, and the remote smtp will have to retry the connection later. If you experience trouble with this, you may want to switch `amavisd` into the normal filtering mode and use `D_BOUNCE` or `D_PASS` instead of `D_REJECT` (read the `README.postfix` in the `amavisd` documentation included with the source code). You can also read more about the Before-Queue filter in the `SMTPD_PROXY_README` included with the postfix source.

9. Next we want to edit the transport file (`/etc/postfix/transport`). This is a file that identifies where to route our incoming email. Since we want this server to accept all mail for our domain and route it to our exchange server, we will add that line here.

```
domain.com smtp:[10.1.1.50]
```

If you want to use a FQDN instead of an IP address, just leave out the brackets (i.e. `smtp:exchange.domain.com`). Just make sure that the server will be able to successfully lookup that DNS record.

After editing the transport file, we need to run the `postmap` command. Postfix doesn't actually read the textfile we created because that would be slow (especially if the file had many entries). Instead we convert the file into a database format using the `postmap` command.

```
# postmap /etc/postfix/transport
```

After running this command, you will see the new database file that has been created: `/etc/postfix/transport.db`.

10. Setup Postfix to launch at Startup by editing the `/etc/rc.local` file to include a line: `/usr/sbin/postfix start`. Once that is done go ahead and start postfix using the same command.
11. You should test to make sure that you can connect to the SMTP interfaces on port 25 and port 10025. To do this use the command:

```
# telnet localhost 25
```

```
The server should respond with:
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 mta1.domain.com ESMTP
press ctrl-], then type 'quit' to quit
```

```
# telnet localhost 10025
```

```
The server should respond with:
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 mta1.domain.com ESMTP
press ctrl-], then type 'quit' to quit
```

Install Needed Perl Modules

Both amavisd-new and SpamAssassin are written in perl and have a number of other perl modules as dependencies. Fortunately perl has a built-in way to download and install these modules. To start the perl command environment use the command:

```
# perl -MCPAN -e shell
```

This command will popup a little `cpan>` prompt where you can enter commands. To install a module, type *install* followed by the module name (ex. `install MIME::Words`). If you need help, type `help`.

Here is a list of the modules required. Note that some may return saying they are up to date, so just move on to the next one. Also note that you don't necessarily need to run the latest version of all these modules in order for amavisd to work correctly. If the module is already installed, it may work fine without needing to be updated (especially since some module upgrades may require a newer version of perl!).

- MD5
- LWP
- Mail::Internet
- Archive::Tar
- Archive::Zip
- IO::Wrap
- IO::Stringy
- Unix::Syslog
- MIME::Words
- MIME::Head
- MIME::Body
- MIME::Entity

- MIME::Parser
- Net::SMTP
- Net::DNS (when prompted to enable tests, choose no)
- Net::Ping
- Net::Server
- Net::Server::PreForkSimple
- Convert::TNEF
- Convert::UUlib
- MIME::Decoder::Base64
- MIME::Decoder::Binary
- MIME::Decoder::Gzip64
- MIME::Decoder::NBit
- MIME::Decoder::QuotedPrint
- MIME::Decoder::UU
- Time::HiRes
- Digest::SHA1
- Digest::Nilsimsa
- Getopt::Long
- File::Copy
- Bit::Vector
- Date::Calc

If you are feeling really impatient, you can install them all with a single install command: ([install MD5 LWP Mail::Internet Archive::Tar Archive::Zip IO::Wrap IO::Stringy Unix::Syslog MIME::Words MIME::Head MIME::Body MIME::Entity MIME::Parser Net::SMTP Net::DNS Net::Ping Net::Server Net::Server::PreForkSimple Convert::TNEF Convert::UUlib MIME::Decoder::Base64 MIME::Decoder::Binary MIME::Decoder::Gzip64 MIME::Decoder::NBit MIME::Decoder::QuotedPrint MIME::Decoder::UU Time::HiRes Digest::SHA1 Digest::Nilsimsa Getopt::Long File::Copy Bit::Vector Date::Calc](#))

Occasionally you will be prompted with the following message:

```
---- Unsatisfied dependencies detected during
[G/GA/GAAS/Somefile.tar.gz] -----
Name of Dependency
Shall I follow them and prepend them to the queue
of modules we are processing right now? [yes]
```

This is just telling you that the module you are installing needs some other module to work properly. It is asking your permission to go ahead and install it first. Go ahead and press enter to accept the default of yes.

Similarly you will be prompted with a number of other questions during this install process. Just press enter to accept the defaults on most of them.

If you run into any errors, try to read the error to figure out what you should do. In some cases you may get an error when a test that is run as part of the install fails. To install anyway use the [force install modulename](#) command. For more assistance with cpan and perl go to www.cpan.org.

Once everything is installed type 'q' to quit.

Installing and Configuring SpamAssassin

1. Run the perl shell to install it using the command:

```
# perl -MCPAN -e shell
```

then type:

```
install Mail::SpamAssassin
```

If you are asked if you want to run the Razor2 or DCC tests, just press enter to select NO. When finished type **q** and press enter to quit.

2. Edit the `/etc/mail/spamassassin/local.cf` file and make the following settings:

```
rewrite_subject 1

report_safe 0
use_terse_report 0

use_bayes 1
bayes_path /var/amavisd/.spamassassin/bayes
auto_learn 1

skip_rbl_checks 0
use_razor2 1
use_dcc 1
use_pyzor 0
dcc_add_header 1

dns_available yes

header LOCAL_RCVD Received =~ /\.*\(\S+\.domain\.com\s+\[.*\]\)/
describe LOCAL_RCVD Received from local machine
score LOCAL_RCVD -50

## Optional Score Increases
score DCC_CHECK 4.000
score RAZOR2_CHECK 2.500
score BAYES_99 4.300
score BAYES_90 3.500
score BAYES_80 3.000
```

- 3.

Explanation of SpamAssassin configuration settings:	
<code>rewrite_subject</code>	(1 0) Tells SpamAssassin to change the subject on SPAM messages to include the <code>subject_tag</code>
<code>report_safe</code>	(0 1 2) This setting configures how to handle SPAM. A setting of 0 puts the SpamAssassin report into the headers. A setting of 1 puts it in the main email and attaches the original email as an attachment. Setting 2 is similar to setting 1, plus it changes the type of attachment to text/plain (as a security

	measure). NOTE: this setting has no affect when spamassassin is run through amavisd, so just leave it set to 0.
use_terse_report	(0 1) Setting this to 0 gives the normal length explanation of why the message was considered SPAM. Setting it to 1 gives a shorter report. (Note that this report only appears if you change the report_safe setting, or if you configure blocking like we will do...in which case the sender gets this report).
use_bayes	(0 1) This setting turns the Bayesean Learning on or off. In our case we want it on.
bayes_path	(path) Location of systemwide bayes database. We need to set this so that the root account can run the sa-learn program to update the bayes database normally used by the amavisd user.
auto_learn	(0 1) If turned on (1), this will cause SA to automatically add very SPAM or very not-SPAM messages to the Baysean statistics database.
skip_rbl_checks	(0 1) Definitely turn this off (0). Unlike using the Blackhole lists from within Postfix, using them in SpamAssassin allows you to selectively use the RBLs. For example, if you have a client that has an open relay and is unwilling to change it, you can simply add their domain to your whitelist, while still being able to use the RBLs for everyone else.
use_razor2	(0 1) Tells SA that we want to use Razor version 2
use_DCC	(0 1) Tells SA that we want to use DCC (Distributed Checksum Clearinghouse)
use_pyzor	(0 1) Tells SA that we don't want to use Pyzor (since we won't be installing it).
dcc_add_header	(0 1) Tells SA to add a header from DCC containing statistics about the message
dns_available	(yes test no) Normally SA tests to see if it has access to a DNS server to do lookups. Since I know my server has access to DNS, I tell it to skip this test. It saves on initial amavis startup time.
LOCAL_RCVD rules	The last lines header, describe and score are used to prevent my outgoing mail from being tested for spam. If you're like me, your users would be upset if their mail was tagged as spam before a client read it. This rule basically checks the header for the <i>Received from:</i> lines showing the message route. You will need to customize this rule to fit your system. To do this, send a message to your test hotmail account (or some account outside

	<p>of your system). Check the headers for lines that look like this:</p> <hr/> <pre>Received: from mta1.domain.com (mta1.domain.com [64.132.107.5]) by law122.ms.hotmail.com (8.12.6/8.12.2) with ESMTP id h3EI48pL002768 for <forge@hotmail.com>; Mon, 14 Apr 2003 14:04:08 -0400 (EDT) Received: from exchange.domain.com (exchange.domain.com [10.1.1.50]) by mta1.domain.com (Postfix) with ESMTP id F3B7117EFD for <forge@hotmail.com>; Mon, 14 Apr 2003 14:04:07 -0400 (EDT) Received: by exchange.domain.com with Internet Mail Service (5.5.2653.19) id <GKFZ3XPK>; Mon, 14 Apr 2003 14:04:03 -0400</pre> <hr/> <p>The rule is a standard SpamAssassin rule and uses Regular Expression syntax. To explain it in regular terms, it looks for *.domain.com (*[*]) on the received line (where the stars are anything). When it finds a match, it gives the message a SPAM score of -50 (ensuring it is not counted as SPAM).</p>
Optional Score Increases	<p>In this section, I turn up the value of several of the rules. The default score for a spam that turns up in the DCC database is only 2.756 when we're using Bayes and network checks. This seemed a little low for me, so I upped it to 4 points. If you wanted every message listed in the DCC database to be tagged as SPAM, you'd set this to 6.3 points. You can check the default scores for everything in the file /usr/local/share/spamassassin/50_scores.cf. You may see 4 different scores listed next to some rules. The file has different scores for whether or not you are using Bayes and network checks. When there is only 1 score, that score applies all the time, otherwise the 4th score is for bayes and network checks like we are using.</p>

4. Now we want to setup the amavisd and spamassassin home directory for the amavisd user.

```
# mkdir -p /var/amavisd
# chown amavisd.amavisd /var/amavisd
# chmod 750 /var/amavisd
# cd /var/amavisd
# mkdir .spamassassin
```

```
# touch .spamassassin/user_prefs
# chown -R amavisd.amavisd .spamassassin
```

Installing Razor

Unfortunately I couldn't get Razor working correctly when *not* running as chroot. For some reason it had a hard time figuring out what its razorhome directory was (even though it was explicitly set in the config file). Fortunately it works great in chroot mode. If you are not going to run amavisd-new in chroot mode, you may want to just disable the razor checks in your /etc/mail/spamassassin/local.cf file.

Download latest v2 razor-agents Untar and run

```
# cd /root
# ftp http://unc.dl.sourceforge.net/sourceforge/razor/razor-agents-2.40.tar.gz
# tar -zxvf razor-agents-2.40.tar.gz
# cd razor-agents-2.40
# perl Makefile.PL
# make
# make test
# make install
```

Create the default configuration files in /etc/razor

```
# razor-client
# razor-admin -create
```

Register yourself with the razor network. Substitute your exchange email address for the address listed.

```
# razor-admin -register -user postmaster@domain.com
```

Copy the razor config files to the chroot directory

```
# cp -r /root/.razor /var/amavisd
# chown -R amavisd.amavisd /var/amavisd/.razor
```

You can enable or disable Razor by editing your etc/mail/spamassassin/local.cf file:

```
use_razor2 1
```

One last note, razor has its own logfile in /var/amavisd/.razor called razor-agent.log. Unchecked, this file could potentially grow to fill your entire /var volume. Therefore, once you are sure that razor is working properly, you will want to edit the razor config to turn off logging. Unfortunately you can't use newsyslog to rotate this logfile without having to stop and restart amavisd. Edit /var/amavisd/.razor/razor-agent.conf and change:

```
debuglevel = 5
```

to

```
debuglevel = 0
```

then restart amavisd.

Installing DCC

Download and extract the latest DCC (<http://www.dcc-servers.net/dcc/>)

```
# cd /root
# ftp http://www.dcc-servers.net/dcc/source/dcc-dccd.tar.Z
# tar -zxvf dcc-dccd.tar.Z
# cd dcc-dccd-1.2.36
# ./configure
# make
# make install
```

Make sure udp port 6277 is allowed out from this computer on your firewall.

```
# /usr/local/bin/cdcc 'info'
```

If everything is working, you should see a bunch of lines like:

```
dcc.rhyolite.com, - RTT+0 ms anon
# 153.19.44.233,- coral.ely.pg.gda.pl WEiAPG server-ID 1072
# 100% of 3 requests ok 1687.64+0 ms RTT 113 ms queue wait
# 192.188.61.3,- calcite.rhyolite.com Rhyolite server-ID 101
# 100% of 2 requests ok 755.52+0 ms RTT 50 ms queue wait
```

Since amavisd-new will be running in chroot mode, we need to copy DCC and all the files it needs to the chroot directory. The way DCC is called by SpamAssassin requires /bin/sh in order to work properly. You should be aware that this it reduces the security of the system. Without it, you'll receive this error when running amavisd debug: *DCC -> check failed: no response*

```
# mkdir -p /var/amavisd/var /var/amavisd/usr/bin /var/amavisd/usr/libexec
/var/amavisd/var/dcc
# mkdir -p /var/amavisd/usr/lib /var/amavisd/bin
# cp -r /var/dcc /var/amavisd/var/
# cp /usr/local/bin/dccproc /var/amavisd/usr/bin
# cp /usr/libexec/ld.so /var/amavisd/usr/libexec
# chown -R amavisd:amavisd /var/amavisd/var/dcc
# cp /bin/sh /var/amavisd/bin/
```

DCC uses several OpenBSD libraries, which we need to copy to the chroot directory. Since the version numbers on these libraries seem to change regularly, rather than just telling you which files to copy, you can run the following command:

```
# ldd /usr/local/bin/dccproc
```

This will output something like this:

```
/usr/local/bin/dccproc:
Start End Type Ref Name
00000000 00000000 exe 1 /usr/local/bin/dccproc
00008000 2000f000 rlib 1 /usr/lib/libm.so.1.0
00019000 20063000 rlib 1 /usr/lib/libc.so.30.0
00002000 00002000 rtld 1 /usr/libexec/ld.so
```

Just copy the listed files (except dccproc) to their chroot directory under /var/amavisd. So for example, you would copy **/usr/lib/libm.so.1.0** to /var/amavisd/**/usr/lib/libm.so.1.0**.

You can enable or disable DCC by editing your etc/mail/spamassassin/local.cf file:
use_dcc 1

Installing and Configuring Amavisd-new

1. Installing amavisd-new is actually pretty simple once all the PERL required modules have been installed (which we did above). To install, all we have to do is copy the perl code file where we want it, set the permissions and make it executable.

```
# cd /root
# ftp http://www.ijs.si/software/amavisd/amavisd-new-20030616-p9.tar.gz
# tar -zxvf amavisd-new-20030616-p9.tar.gz
# cd amavisd-new-20030616
# cp amavisd /usr/local/sbin/
# chown root.wheel /usr/local/sbin/amavisd
# chmod 550 /usr/local/sbin/amavisd
# cp amavisd.conf /etc/
# chown root.wheel /etc/amavisd.conf
# chmod 644 /etc/amavisd.conf
# touch /var/amavisd/amavis.log
# chown amavisd.amavisd /var/amavisd/amavis.log
```

2. Edit the Amavisd-new configuration /etc/amavisd.conf and make the following changes:

```
use strict;

$MYHOME = '/var/amavisd';
$mydomain = 'domain.com';
$daemon_user = 'amavisd';
$daemon_group = 'amavisd';
$daemon_chroot_dir = $MYHOME;

$QUARANTINEDIR = "$MYHOME/quarantine";
$TEMPBASE = "$MYHOME/tmp";
$ENV{TMPDIR} = $TEMPBASE;
$helpers_home = $MYHOME;

$max_servers=8;
```

```
$child_timeout=20*60; # we need to adjust the timeout since it
is not a localhost transfer

$forward_method = 'smtp:127.0.0.1:10025';
$notify_method = $forward_method;
$inet_socket_port = 10024;
$inet_socket_bind = '127.0.0.1';
@inet_acl = qw( 127.0.0.1 );

@bypass_virus_checks_acl = qw( . );
@local_domains_acl = ( ".$mydomain" );

$DO_SYSLOG = 1; # (1 = syslog, 0 = logfile)
$LOGFILE = "$MYHOME/amavis.log";
$log_level = 5; # (0-5)

$final_spam_destiny = D_REJECT; # Set to D_REJECT, D_PASS to
pass through

read_hash(\%whitelist_sender, '/var/amavisd/whitelist');
read_hash(\%blacklist_sender, '/var/amavisd/blacklist');
read_hash(\%spam_lovers, '/var/amavisd/spam_lovers');

#defending against mail bombs
$MAXLEVELS = 14; # Maximum recursion level for
extraction/decoding
$MAXFILES = 1500; # Maximum number of extracted files
$MIN_EXPANSION_QUOTA = 100*1024; # bytes (default undef, not
enforced)
$MAX_EXPANSION_QUOTA = 300*1024*1024; # bytes (default undef,
not enforced)
$MIN_EXPANSION_FACTOR = 5; # times original mail size (must be
specified)
$MAX_EXPANSION_FACTOR = 500; # times original mail size (must be
specified)

$path =
'/usr/local/sbin:/usr/local/bin:/usr/sbin:/sbin:/usr/bin:/bin';

# SpamAssassin settings
$sa_local_tests_only = 0;
$sa_auto_whitelist = 1; # comment this line out to turn off auto
whitelist
$sa_mail_body_size_limit = 64*1024; # 64KB

$sa_tag_level_deflt = 3.0; # controls adding the X-Spam-Status
and X-Spam-Level headers,
$sa_tag2_level_deflt = 6.3; # controls adding 'X-Spam-Flag:
YES', and editing Subject,
$sa_kill_level_deflt = $sa_tag2_level_deflt; # triggers spam
evasive actions:

$sa_spam_subject_tag = '***SPAM*** ';
```

```
$sa_debug = 1; # comment this line out to turn off debugging
1; # insure a defined return
```

3.

Explanation of Amavisd-new configuration settings:	
\$MYHOME, \$mydomain	These are really just variables used in this config file so we don't have to type the same things over and over again. MYHOME is our default work directory, mydomain is the name of our domain.
\$daemon_user, \$daemon_group	This is the user and group that amavisd and all its helper programs will run under for security reasons.
\$daemon_chroot_dir	This option tells amavisd to run in chroot mode. Chroot is a security trick that traps the program in a certain directory, in our case \$MYHOME. By turning this on, amavisd will not be able to access anything on the computer other than what is in the /var/amavisd directory.
\$QUARANTINEDIR	This is the location where amavisd would place quarantine files if you were using it for virus scanning. Even though we aren't virus scanning, it is still needed.
\$TEMPBASE	When amavisd does its thing, it creates a bunch of temp files. Normally it would just store them in \$MYHOME. By storing them in \$MYHOME/tmp, we can easily cleanup the temp directory whenever we need to by stopping amavisd and deleting everything in \$MYHOME/tmp/*
\$ENV{TMPDIR}, \$helpers_home	I found that both of these setting help SpamAssassin, Razor and DCC work better by explicitly telling these helper programs their home and tmp directories.
\$max_servers	Specifies the number of instances of amavisd child processes to spawn. You should increase or decrease this number depending on how much memory your PC has installed. Make sure that this

	number is synchronized to the smtpd maxproc setting in the postfix/master.cf file (so that you don't end up with postfix trying to connect to a non-existent queue).
\$child_timeout	Here we need to increase the timeout for the amavisd process. The default is only 8 minutes. When amavisd is used with other mail packages in a post queue structure, the message is transferred locally only (from the local mail queue to the amavisd loopback) and is therefore very fast. When we use the postfix before-queue filter, the message is being transferred directly from the remote smtp through postfix to amavisd . So for example if the remote smtp is sending a 20 megabyte message, it may take longer than 8 minutes to transmit the message (depending on network bandwidth). In my testing, amavisd would timeout before the message had been transferred and the end user would get the following message: Out: 451 Error: queue file write error. Note that you may want to increase this limit even further if you see this error message from Postfix.
\$forward_method, \$notify_method, \$inet_socket_port, \$inet_socket_bind, \$inet_acl	These settings set up the communications between amavisd and postfix. Basically the two programs communicate using different ports. Postfix sends email it wants to filter to amavisd on port 10024. Amavis processes the message and returns it to postfix on port 10025. The \$inet_acl setting makes sure that it accepts only packets from the local computer.
@bypass_virus_checks_acl	Since we won't be using the anti-virus features of amavisd, this line is used to turn them off.
@local_domains_acl	This setting is used to determine if a message is incoming or outgoing.
\$DO_SYSLOG, \$LOGFILE,	These settings describe how amavisd should do logging for

<pre>\$log_level</pre>	<p>debugging. The <code>\$log_level</code> can be set from 0-5 with 5 producing the most logging output. Even though we aren't using it, the <code>\$LOGFILE</code> setting is required. Also don't be surprised to see an empty file with this name in the <code>\$MYHOME</code> directory. You should also note that when running <code>amavisd</code> in debug mode, logging doesn't occur to the <code>syslog</code>.</p> <p>Note #1: the main reason we don't want to log to file is because when <code>chrooted</code>, <code>amavisd</code> can't be restarted with a <code>HUP</code> command. Therefore to rotate the logfile with <code>newsyslog</code>, you'd need to stop and restart <code>amavisd</code>. It's easier just to log to <code>syslog</code>.</p> <p>Note #2: if you want to have <code>amavisd</code> log to <code>/var/log/messages</code> instead of <code>/var/log/maillog</code>, you can add a config line here: <code>\$\$SYSLOG_LEVEL = 'user.info';</code></p>
<pre>\$final_spam_destiny</pre>	<p>You should set this to either <code>D_REJECT</code> or <code>D_PASS</code>. When set to <code>D_PASS</code>, the spam will be delivered to your users. They can then setup inbox assistant rules looking for the <code>X-Spam-Flag</code> header to move the messages to another folder, or delete the messages. The <code>D_REJECT</code> setting will reject the email message during the original connection. The sender will get an undeliverable bounce message from their <code>smtp</code> server (more details about this later).</p> <p>You can also configure <code>D_BOUNCE</code> mode, see the <code>amavisd</code> documentation for more info.</p> <p>At my company, we used the <code>D_PASS</code> method for the first 60 days after turning on <code>SpamAssassin</code> to make sure everything was working right (auto-whitelisting had been done, and the bayesian db had been built). Then we switched over to <code>D_BOUNCE</code> mode. Once postfix supported before-queue filtering, we switched to <code>D_REJECT</code> mode.</p>
<pre>read_hash(\%whitelist_sender, read_hash(\%blacklist_sender,</pre>	<p>These settings point to 3 files that identify our whitelist,</p>

<pre>read_hash(\%spam_lovers</pre>	<p>blacklist and spam lovers. Each file should have 1 email address (or part of an email address) per line. The whitelist identifies senders that should always be passed through even if they are identified as spam. The blacklist identifies senders that should always be marked as spam. The spam_lovers identifies our users that want to opt-out of our system and receive all their spam.</p> <p>Note that these files are read only on startup of amavisd, so if you edit them be sure to stop and restart amavisd. Also, not shown here is the ability of amavisd to support per-recipient whitelists and blacklists and MySQL based whitelists and blacklists.</p>
<pre>\$MAXLEVELS, \$MAXFILES, \$MIN_EXPAN...</pre>	<p>This whole section is part of the anti-mail bomb measures of the anti-virus software. I thought it should be kept in there just in case.</p>
<pre>\$path</pre>	<p>Tells amavisd where to look for programs it needs.</p>
<pre>\$sa_local_tests_only</pre>	<p>Set this to 1 to disable DCC and Razor for debugging. The default for this is false, so I initially didn't include it in my config file, but I have been told that leaving it out causes problems.</p>
<pre>\$sa_auto_whitelist</pre>	<p>Great feature of spamassassin that automatically whitelists people that send you lots of non-spam mail.</p>
<pre>\$sa_mail_body_size_limit</pre>	<p>Since few spammers send large attachments, we don't even bother checking messages over a certain size (in this case 64KB).</p>
<pre>\$sa_tag_level_deflt, \$sa_tag2_level_deflt, \$sa_kill_level_deflt</pre>	<p>As you know, SpamAssassin assigns each email a positive or negative score to indicate its "spamminess" (yes I know its not a word). These settings tell SA when to take anti-spam measures. At the tag level, the message's headers are modified with the spam score. At the tag2 level, in addition to the actions from the tag level, it adds a header indicating that the message is spam, and it modifies</p>

	the subject. At the <i>kill</i> level, the system will take action based on the <code>\$final_spam_destiny</code> setting. In most cases, you want <code>tag2</code> and <code>kill</code> to happen to any message that is spam, so they are usually equal.
<code>\$sa_spam_subject</code>	This text is added to the front of any spam message that receives a high enough score.
<code>\$sa_debug</code>	This setting turns on spamassassin debugging. Comment it out once everything is running smoothly for better performance.

4. Next we need to create the directories used by amavisd:

```
# mkdir /var/amavisd/tmp
# chown amavisd:amavisd /var/amavisd/tmp
# chmod 750 /var/amavisd/tmp
# mkdir /var/amavisd/quarantine
# chown amavisd:amavisd /var/amavisd/quarantine
# chmod 750 /var/amavisd/quarantine
```

5. Now we want to do everything needed to run amavisd-new in a chroot jail (you can skip this step if you want, but just make sure you comment out the `$daemon_chroot_dir` setting in the `amavisd.conf` file with a `#`). Please note that I'd advise you to do these steps since it makes your system more secure.

Note that all the commands in this numbered section assume that you are in the `/var/amavisd` directory:

```
# cd /var/amavisd
```

Make the default root folders:

```
# mkdir -p etc dev tmp var/run
# mkdir -p usr/bin usr/share/zoneinfo usr/lib usr/libexec
```

Make the spamassassin config folders:

```
# mkdir -p usr/local/share/spamassassin etc/mail/spamassassin
```

Setup a dev/null device:

```
# mknod dev/null c 2 2
```

To make this device work, you have to remove the `nodev` restriction on that filesystem and remount it. To do this, edit the `/etc/fstab` file. Look for the line where you mount the `/var` partition and remove the word `nodev` and the extra comma.

```
# vi /etc/fstab
```

```
/dev/wd0g /var ffs rw,nodev,nosuid 1 2
```

Setup a symbolic link so the chrooted process can refer to /var/amavisd and still get the files in /var/amavisd (which would then be /)

```
# ln -s / var/amavisd
```

Copy the configuration files for our system to our chroot etc directory. Please note that if you ever make any changes in the originals you will need to copy them again to this directory.

```
# cp /etc/protocols etc
# cp /etc/services etc
# cp /etc/hosts etc
# cp /etc/magic etc
# cp /etc/resolv.conf etc
# cp /etc/group etc
# cp /etc/passwd etc
```

Create a dummy pwd.db file:

```
# echo amavisd:*:3000:3000::0:0:Amavis Mail Scanner
Daemon:/var/amavisd:/sbin/nologin>/var/amavisd/etc/master.passwd
# pwd_mkdb -d /var/amavisd/etc/ -p /var/amavisd/etc/master.passwd
```

Copy the SpamAssassin files we need to our chroot directory

```
# cp /etc/mail/spamassassin/local.cf etc/mail/spamassassin/
# cp -r /usr/local/share/spamassassin usr/local/share
```

Set strict permissions. Note that amavisd must own its home directory (/var/amavisd) otherwise it will generate permissions errors.

```
# chown -R root.wheel etc dev tmp usr var
# chown -R amavisd:amavisd .spamassassin .razor quarantine var/dcc
# chmod 1777 tmp
# chmod 666 dev/null
```

*****REMINDER***** Once you are running amavisd in chroot mode, your configuration files are located in your new root directory. Be sure when you make changes that you are editing the correct files....otherwise you'll wonder why your changes aren't working. Also make sure when you copy them that the permissions are set correctly.

6. Create our whitelist, blacklist and spam_lovers files. These files are lists with 1 email address or domain per line (in lower case), of recipients and senders that we want to treat specially. Senders in the blacklist file are automatically marked as SPAM. Senders in the whitelist file are never marked as SPAM. Recipients in the spam_lovers are basically your users that want to opt-out of the spam blocking system.

```
# touch /var/amavisd/blacklist
# touch /var/amavisd/whitelist
# touch /var/amavisd/spam_lovers
```

You will want to add your postmaster account into the spam_lovers textfile. One of the RFCs for email states that the postmaster account should always

accept email.

```
# echo postmaster@domain.com >> /var/amavisd/spam_lovers
```

7. As a default, the sender of a message that has been rejected will receive a bounce message that looks something like this:

```
From "Mail Delivery Subsystem" MAILER-
DAEMON@mail.somecompany.com
Subject Returned mail: see transcript for details
Date Sat, May 1, 2004 8:57 pm
To person@somecompany.com
-----
-----
----- The following addresses had permanent fatal errors
-----
< sentto@domain.com >
(reason: 550 5.7.1 Message content rejected, UBE,
id=3452)

----- Transcript of session follows -----
... while talking to mta.domain.com.:
>>> DATA
<<< 550 5.7.1 Message content rejected, UBE, id=3452
554 5.0.0 Service unavailable
```

8. Obviously this error message is not very helpful to end users. At my company, we customized this line to include a website and phone number that the user could call:

550 5.7.1 Message content rejected:looks like SPAM. Visit <http://www.domain.com/spam> or call 704-555-1212

Here is an example of the website we refer people to:

<http://www.flakshack.com/anti-spam/whitelist.html>. Please feel free to copy this page as a template. Obviously the form on this example won't actually work without something on the back end to process the form.

To customize this SMTP error output line, you must edit the amavisd code. In the current version this is line 5587-5588:

Change:

```
: "550 5.7.1 Message content rejected, $reason")
. ", id=$am_id");
```

To:

```
: "550 5.7.1 Message content rejected:looks like SPAM. Visit
http://www.domain.com/spam or call 704-555-1212") );
```

9. There is a limit to the number of characters you can put into this error line. After editing it, you should test on your own to see what works. Just be sure to leave the 550 5.7.1 codes on that line.

10. Try running Amavis in debug mode for testing. First edit the `/etc/amavisd.conf` file to temporarily set `$log_level = 5;` and `$sa_debug = 1;`.

To make examining the output easier, click on PuTTY's control box menu (the top left) and choose Clear Scrollback.

```
# clear  
# /usr/local/sbin/amavisd debug
```

You may need to change PuTTY's default window settings to increase the size of the scrollback buffer so that you can see all of the output.

Look at the output to see if there are any error messages. Press CTRL-C to stop the process.

11. If everything checks out, go ahead and add amavisd to the local startup script. Edit the `/etc/rc.local` file and add the following lines to the bottom. Technically you will want to start this before postfix.

```
# Start amavisd spam filter  
/usr/local/sbin/amavisd
```

12. Go ahead and open up another PuTTY window, leaving the amavisd debug output running in the old window. Now we will try sending some test messages through the system to see if they get rejected or tagged correctly (Depending on your `$final_spam_destiny` setting in `amavisd.conf`).

In the amavisd tar.gz file we downloaded and extracted are some test messages to help us test our system. Here's a simple test you can do. Substitute your email address so you will receive the message.

```
# cd /root/amavisd-new-20030314/test-messages  
# cat sample-spam.txt | sendmail myaddress@domain.com  
or  
# cat sample-nonspam.txt | sendmail myaddress@domain.com
```

These sample messages don't get exceptionally high scores, but they are a starting point for your testing. To make it easier to test, you can increase the score for a phone number found in the `sample-spam.txt` file by adding the following to the `/var/amavisd/etc/mail/spamassassin/local.cf` file:

```
body LOCAL_AMAVISTEST /800-320-9895 x2068/  
describe LOCAL_AMAVISTEST Phone number in spam test message  
score LOCAL_AMAVISTEST 10
```

13. One handy thing to know is that if you run tests from a real account, it may get added to the auto-whitelist (with a positive score). If you need to use this real account, this can be bad, since all your email will start being rejected. To fix this:

```
# spamassassin -p /var/amavisd/.spamassassin/user_prefs --remove-address-from-whitelist=mytestaddress@mydomain.com
```

14. When you think it is ready to go, just CTRL-C to stop amavisd debug, then start amavisd in normal mode:

```
# /usr/local/sbin/amavisd
```

Install Postfix Email Reporting

Our last step in this process is to setup email summary reports. Once a day, right before the maillog is rotated with newsyslog, we'll run a program on the maillog to generate a nice summary report and email it to root. There are a couple different reporting scripts for postfix available on freshmeat.net. I didn't try them all because pflogsumm.pl (http://jimsun.linuxnet.com/postfix_contrib.html) worked fine and did what I wanted.

To get this running, simply download the .pl file from his website, check the MD5 signature to make sure it matches, make it executable, and add it to the crontab.

```
# cd /root
# ftp http://jimsun.linuxnet.com/downloads/pflogsumm-1.0.4.pl
# mv pflogsumm-1.0.4.pl /usr/local/sbin/
# chmod 500 /usr/local/sbin/pflogsumm-1.0.4.pl
```

Every night at midnight, the newsyslog program rotates out the maillog file. As part of its process, it renames the /var/log/maillog file to /var/log/maillog.0 then gzips it. We want our script to unzip this file, process it and generate the report, then zip it up again. To do this we'll create a new script file.

```
# vi /usr/local/sbin/my-postfix-report.sh
```

Add the following lines into this file:

```
#!/bin/sh
zcat /var/log/maillog.0.gz | /usr/local/sbin/pflogsumm-1.0.4.pl | mail
-s "Report" root
```

```
# chmod 500 /usr/local/sbin/my-postfix-report.sh
```

Now edit root's crontab:

```
# crontab -e
```

Add the following line to root's crontab to run our new script every day at 4:00 AM:

```
0 4 * * * /usr/local/sbin/my-postfix-report.sh
```

Reporting on SPAM

I have written a perl script that also generates several reports (in the same format as pflogsumm) about your spam. The reports include: 1. All spam senders listed by SA Score, 2. All spam sender domains listed by number of messages, 3. All recipients by number of messages, and 4. Total number of spam. I'm no perl expert, but this script runs fine on my system. You can download it (and all my other scripts) from the [bottom of this doc](#). If you want to use it, just copy the my-spamreport.pl and my-postfix-report.sh files over top of the ones you have already created. (I updated the my-postfix-report.sh script so that it includes the spam report in a single email message).

Update: Kris Nosack has written an updated version of my script to include percentages. You can download this new version of the script here: <http://www.flakshack.com/anti-spam/nosack-spamreport.pl>.

I've also written a script (my-summary.pl) that will count the total number of messages, spam and size per day and store it in an ongoing file. You can then download this file and load it into Excel to graph your spam. Here's an example of the output (date, number of messages sent, total bandwidth of all messages, number of blocked spam):

```
1/3/2004,18099,363151k,5894
1/4/2004,16176,106379k,6370
1/5/2004,16083,90243k,6449
1/6/2004,25380,609m,6147
1/7/2004,26420,708m,6272
```

What if I want to disable Amavisd-new/SpamAssassin for a while?

To disable amavisd, all you have to do is comment out the line in the master.cf that directs postfix to pass incoming mail through its filter. Locate the smtpd_proxy_filter line and put a comment (#) symbol in front of it.

```
#-o smtpd_proxy_filter=127.0.0.1:10024
```

Now just run `# postfix reload`

and all incoming mail will now just be routed normally.

Performance

The configuration as described above is for a PC with 256 MB of RAM and will allow your mail server to accept 8 incoming connections at one time. Once those 8 connections are in use, any further connections will be refused. In this case, the remote SMTP client should continue trying to deliver the message to your mta until it times out, usually in 5 days. If you have less than 256 MB of RAM, you must reduce the number of amavisd and postfix servers, otherwise you will overwhelm the computer (usually amavisd or postfix crashes). If you have more than 256 MB of RAM, you can increase the number of servers to allow for more connections at one time.

You can use the following directions to boost or reduce the number of connections the server can accept at one time. After you make a change, you will want to check your system memory usage using [top](#). When you run the top command, you will see a line that looks something like this:

```
Memory: Real: 113M/144M act/tot Free: 103M Swap: 0K/512M used/tot
```

The important number here is the one labeled "Free." It shows how much free memory is available on your computer. The line above was taken from one of my server running 8 processes. As you can see we are currently using 144 MB of RAM and have 103 MB free. I could probably run 10-12 servers instead of 8, but I like to play it safe.

Since there need to be enough amavisd processes to handle each smtpd process, you need to make sure that you have configured the same number in both locations.

Edit the /etc/amavisd.conf file and edit the following line:

```
$max_servers = 8;
```

The default when this line is not present is 2. Increase or decrease the number as needed.

Now edit /etc/postfix/master.cf. Look for the line that says:

```
smtp inet n - y - 8 smtpd
```

And increase or decrease the maxprocs count for the smtpd line to match the number of amavisd servers you just indicated in the amavisd.conf.

Restart amavisd and postfix. Run the top command, press the "o" key and type "res" to sort the display by memory usage. Be sure to watch the display as mail is delivered through the system, since more memory is used by a running system than by an inactive one.

Performance Expectations

Here are some quotes from people that are using these directions with notes on their performance. If you send me your experience, I'll post it here.

My Performance:

We run 2 of these systems for redundancy and load balancing. Both are Pentium III, 900 mhz Dell PowerEdge Rackmount systems with 256 MB ram and 20 GB HD. On an average week day, each server delivers 700-1000 MB of email (according to my maillog reports) which is usually around 12,000 messages each. Each server detects and blocks about 6000 SPAM messages per day. Performance with this setup seems to be great...no complaints. Load balancing and redundancy is setup by configuring both servers' MX records with the same weight, and by configuring my Exchange server's SMTP connector smart host setting with both addresses (mta1.domain.com;mta2.domain.com) .

Configuring for multiple domains

Since I get this question all the time, here are some quick directions for running with multiple domains. Please note that this is untested, but should give you some guidance. In the examples below, I've added a new domain which I called "domain2.com" to the system.

1. Edit /etc/postfix/main.cf

Modify the mydestination variable to include references to the extra domains (the example below adds a new domain called domain2.com):

```
mydestination = $myhostname, localhost.$mydomain, $mydomain,  
mail.$mydomain.com, localhost.domain2.com, domain2.com, mail.domain2.com
```

2. Edit /etc/postfix/transport

Here is where you tell postfix where to send email for that domain. So add the new domain(s) under the one you already added:

```
domain.com smtp:[10.1.1.50]  
domain2.com smtp:[10.1.1.50]
```

3. Edit /etc/amavisd.conf

Add the domain(s) to the @local_domains_acl hash variable:

```
@local_domains_acl = ( ".$mydomain", 'domain2.com' );
```

Here is an untested script that someone contributed that should add multiple domains to your setup: [add-domain.sh](#). Be sure to edit the DEFAULTTRANSPORT variable to equal the full servername before running the script.

Notes about running this setup on other Operating Systems

I've received a number of emails from users that have followed these directions to setup an anti-spam system on operating systems other than OpenBSD. This section will include their tips. If you get this system working on any other OS and want to include your OS specific changes, please send them to me.

Slackware (tips by Derek Shaw)

When creating the symbolic link in the chroot directory, he advises using the following command:

```
# cd /var/amavisd && mkdir var && ln -s ../ var/amavisd
```

Adding Anti-Virus Support

At my company, we run Symantec Mail Security on our Exchange servers, so we don't need to run anti-virus software on this gateway. However, since Amavisd was originally developed as an anti-virus scanner it works very well in this role. Kris Nosak has taken the time to write up a great set of directions for adding anti-virus support to the solution I've described. Once you've configured your system according to my directions, you can follow his additional steps to accomplish this using the open source [Clam AntiVirus](#). See his document at: <http://www.xmission.com/~kn/AddClamAV/>

Download My Scripts

Here are copies of the scripts that I've used.

[my-postfix-report.sh](#)

[my-restart.sh](#)

[my-sa-learn.sh](#)

[my-spamreport.pl](#)

[my-summary.pl](#) (Script which will track total number of messages, spam, and size per day).

[add-domain.sh](#) (Script to easily add multiple domains [1 at a time] to your setup - be sure to edit the defaulttransport variable before running the script)

Changelog

- 4/25/2003 Added changes to amavisd program to include extra perl module: Mail::SpamAssassin::PerMsgLearner.
- 4/29/2003 Added section "Keeping the mail queue clean."
Fixed syntax of razor command.
Added directions to make directories in DCC step.
Added some clarification when editing the amavisd

- perl file to include the fetch_modules.
Added note about /bin/sh as an alternative to /bin/csh
- 4/30/2003 Added a note that you can stop deleting MAILER-DAEMON messages once you've switched to D_BOUNCE mode.
Removed a line where I mistakenly said Pyzor required X.
Added the \$QUARANTINEDIR line to the /etc/amavisd.conf config file. Without it you might get error messages about a missing virusmails directory.
- 5/1/2003 Added a comment about using rdate without the -a parameter initially to set your system clock.
Added a note to add the notspam@mtal.domain.com to the spam_lovers list.
Added a comment that you can use the EDITOR environment variable to get OpenBSD to use whatever editor you prefer (if you don't like vi).
Fixed discrepancy in the my-postfix-report crontab....it should be run at 4 am, not 1 am.
- 5/2/2003 Fixed typo when adding the .forward file. It should be added to /root not /etc.
Added new directions to patch Razor2 to fix taint error messages.
Added section on improving performance.
Removed incorrect sections from log cleanup routine.
Added \$banned_filename_re to /etc/amavisd.conf as a workaround.
Added a note about turning off razor logging to the razor section.
Added Monte's enhanced cleanup script.
- 5/5/2003 Added section for Optional Score Increases to spamassassin config.
- 5/6/2003 Added note to amavisd modification not to add setuid and setgid when using OpenBSD 3.3.
- 5/7/2003 Changed recommended maximum message size in postfix from 50 megs to 20 megs.
- 5/9/2003 Added "bytes" to the amavisd module edits (for D_BOUNCE mode)
Added note about included a blank line in the notify_spam_sender.txt file.
Updated amavisd-new ftp download to patch 2 location.
Commented out \$banned_filename in amavisd.conf. It is not needed in the latest version of amavisd-new, and removing it should make spam checking faster.
Added notes of new library filenames in the DCC chroot procedure for OpenBSD 3.3.
- 5/13/2003 Added my-spamreport.pl script to the reporting section to help report on SPAM.
Corrected library filenames typo in DCC chroot section.
Added /dev/null 2>&1 to the my-queue-cleanup.sh to eliminate crontab mailings

- 5/14/2003 Updated my-spamreport.pl script to accomodate empty sender addresses.
Updated DCC section with regard to /bin/sh (apparently DCC doesn't require /bin/sh, that is probably required by the implementation by SA)
Updated firewall ports section...DCC also UDP inbound as well as outbound
Added section to bayes-learner explaining that my company ended up not using the spam/notspam addresses for learning.
- 5/15/2003 Added \$sa_local_tests_only to amavisd.conf
- 5/16/2003 Updated this Doc for OpenBSD 3.3-current. NOTE: I found a problem with large attachments on version OpenBSD 3.2 (any emails >5MB won't be delivered), so don't use any version of OpenBSD prior to 3.3.
Added directions for patching Net::Server to fix *Net::Server: Couldn't POSIX::setuid to "3000" []* error.
- 6/3/2003 Added commands to dynamically determine the libraries required by DCC and copy them to the chroot directory.
Added directions for where to find help.
- 7/16/2003 Added several revisions from Ralf Hildebrandt:
* removed smtpd_helo_restrictions and smtp_sender_restrictions (from postfix main.cf) since the setting of permit_mynetworks was the default anyway.
* added a note that some people will want to add a reject_non_fqdn_sender entry to the smtpd_recipient_restrictions.
* changed the proxymap chroot setting in postfix's master.cf from y no n. According to Ralf proxymap is better run as not chroot.
* removed -o strict_rfc821_envelopes=yes from the amavis interface section.
* updated my-postfix-report.sh to use the zcat command instead of gzip and gunzip.
Added Other OS Tips section to include tips from Derek Shaw on Slackware.
Added external->10.2.1.50:25 on the firewall table. I figured most people already knew that, but I received so many suggestions to add it that I've done so.
Added a section for performance expectations.
Added download option for Kris Nosack's updated version of my-spamreport.pl.
Added an index.
- 7/17/2003 Updated URL for WINSXP
- 7/31/2003 Updated download URL for latest amavisd-new version.
Replaced script to copy dccproc libraries with manual directions.
Modified testing command from using mail binary to using sendmail so that headers are processed correctly.
Updated Slackware tips section

- Added tips for running clamd in chroot
Added separate downloads for all my scripts
- 8/18/2003 Fixed typo in the my-postfix-report.sh
Fixed typo in the rdate cron job
- 10/4/2003 Spamassassin has changed its default rules directory
from /usr/share/spamassassin to
/usr/local/share/spamassassin.
Added directions for removing an address from the
whitelist under the Installing Amavisd section
Removed section with directions on how to edit the
amavisd file to allow for chrooting. This problem has
been fixed in the latest version, so it no longer
needs to be performed.
- 12/10/2003 Updated because of move to new website. Also updated
the forums location.
- 12/19/2003 Changed recommended setting of skip_rbl_checks to 0
instead of 1.
- 1/13/2004 Updated clamd instructions to include urandom change
from c 2 2 to c 45 2
Updated the LOCAL_RCVD regex to a more accurate
string to help avoid forgeries thx to Bojan Zdrnja.
Added my-summary.pl to create a summary text file
that can be pulled into excel.
I also updated my-spamreport.pl to eliminate some
error messages that occurred when the sender address
was blank.
- 2/19/2004 Fixed typo in master.cf (was trival-rewrite instead
of trivial-rewrite)
Fixed mydestination setting to include \$mydomain
Removed section on Bayes learning script. Now that
most spam messages include anti-bayes words, bayes is
not as critical as it used to be. Bayes already
worked well in auto_learn mode, so eliminating these
directions makes the setup easier, and still just as
effective.
Added notes about /etc/pwd.db error message.
Clarified where I mentioned whitelist that it was the
auto-whitelist (not the amavisd whitelist)
Modified directions to specify edits to the
rc.conf.local instead of rc.conf file.
Updated Razor patch directions to account for quinlan
patch
Changed smtpd_banner to remove Postfix label (for
security)
Added directions contributed by others for supporting
multiple domains.
- 3/2/2004 Added directions for pausing postfix to the disable
amavis section.
Added top command directions to performance section
Updated CLAMD directions to add the mkdir usr/sbin
step
- 3/22/2004 Updated command lines to reflect new versions of DCC
and Razor.

- Added comment to NOTES section on sudo.
Updated the section on adding anti-virus to include a link to Kris Nosack's guide.
- 5/3/2004 Major revision to include the new Postfix before-queue filtering process, and try to simplify the process a little.
Removed section on keeping the mail queue clean, since it is not needed with the new filtering process.
Removed section for adding compression packages since they are no longer needed for a spam-only solution.
Removed section describing the notify_spam_sender.txt since we are using D_REJECT instead of D_BOUNCE.
Removed section for how to patch OpenBSD to make the document easier to manage.
Removed section about Razor patch (since it is not needed anymore)
Updated link to NTP servers with a better link.
Added directions to create dummy pwd.db file.
- 5/4/2004 Removed Kris Nosack's performance comment since he is not running under these new rules.
Added section on Hardware Requirements at the beginning of the doc.
Renamed the improving performance section to Performance and tried to clear it up a little.
- 5/6/2004 Fixed dead link to Nosack's spam report.

Legal Stuff

This document is ©2004 by Scott Vintinner and is released under the OpenContent License (shown below).

Thanks to the following people who have responded back with corrections and recommendations : Monte Ohrt, Sean Lally, Jim O'Donald, Patryck, Kris Nosack, Karl Kopp, Vernon Schryver, Jens Gutzeit, Ralf Hildebrandt, Derek Shaw, Bojan Zdrnja, Kevin Roosdahl, David White, Cameron Moore, and Mark Martinec (from Amavisd-new).

OpenContent License (OPL)
Version 1.0, July 14, 1998.

This document outlines the principles underlying the OpenContent (OC) movement and may be redistributed provided it remains unaltered. For legal purposes, this document is the license under which OpenContent is made available for use.

The original version of this document may be found at
<http://opencontent.org/opl.shtml>

LICENSE

Terms and Conditions for Copying, Distributing, and Modifying

Items other than copying, distributing, and modifying the Content with which this license was distributed (such as using, etc.) are outside the scope of this license.

1. You may copy and distribute exact replicas of the OpenContent (OC) as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the OC a copy of this License along with the OC. You may at your option charge a fee for the media and/or handling involved in creating a unique copy of the OC for use offline, you may at your option offer instructional support for the OC in exchange for a fee, or you may at your option offer warranty in exchange for a fee. You may not charge a fee for the OC itself. You may not charge a fee for the sole service of providing access to and/or use of the OC via a network (e.g. the Internet), whether it be via the world wide web, FTP, or any other method.

2. You may modify your copy or copies of the OpenContent or any portion of it, thus forming works based on the Content, and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified content to carry prominent notices stating that you changed it, the exact nature and content of the changes, and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the OC or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License, unless otherwise permitted under applicable Fair Use law.

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the OC, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the OC, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it. Exceptions are made to this requirement to release modified works free of charge under this license only in compliance with Fair Use law where applicable.

3. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to copy, distribute or modify the OC. These actions are prohibited by law if you do not accept this License. Therefore, by distributing or translating the OC, or by deriving works herefrom, you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or translating the OC.

NO WARRANTY

4. BECAUSE THE OPENCONTENT (OC) IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE OC, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE OC "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK OF USE OF THE OC IS WITH YOU. SHOULD THE OC PROVE FAULTY, INACCURATE, OR OTHERWISE UNACCEPTABLE YOU ASSUME THE COST OF ALL NECESSARY REPAIR OR CORRECTION.

5. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MIRROR AND/OR REDISTRIBUTE THE OC AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE OC, EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.