MailScanner

www.mailscanner.info

MailScanner

User Guide and Training Manual

Julian Field

First publication 2004

Printed by CafePress.com in the United States of America

ISBN: 0-85432-815-7

Contributors: Ugo Bellavance, [ugob@camo-route.com]
Michele Neylon, [michele@blacknightsolutions.com]
Denis Beauchemin [denis.beauchemin@usherbrooke.ca]
Ron Pool [amp1@nysaes.cornell.edu]

Edition 1. This book is based on the implementation of MailScanner as at Version 4.61.

I would like to gratefully acknowledge all the support and assistance provided by the following organisations:

Electronics and Computer Science

University of Southampton

FSL Fort Systems Ltd.

transtec THE EUROPEAN IT FACTORY

This book is dedicated to all the people around the world who have contributed ideas to the MailScanner project and to all those who have supported its development in so many different ways. Without you all, it would not be the world-leader it is today.

Contents

Preface

Steve Swaney

I’ve had an email address since 1987. I used to post to newsgroups and I’ve been buying things on the web since they started selling things on the web. I get a ton of spam and more than my share of viruses; a typical day will see over five hundred spams and two or three viruses hitting my inbox.

May 3rd 2003 marked the 25th anniversary of the earliest documented email spam. Calling Unsolicited Commercial Email (UCE) spam allegedly comes from the spam skit by Monty Python's Flying Circus. In the sketch, a restaurant serves all its food with lots of spam, and the waitress repeats the word several times in describing how much spam is in the items. When she does this, a group of Vikings in the corner start a song:

"Spam, spam, spam, spam, spam, spam, spam, spam, lovely spam! Wonderful spam!"

A repetitive and annoying refrain. In 1994 two lawyers, Canter and Siegel used automation software to post a message advertising their services to several thousand newsgroups. People starting referring to their posting as spam and the new meaning entered our language.

In 1995 I began receiving spam. I had just started an email hosting service and noticed that my customers I and were getting a message or two a week that we hadn’t signed up to receive. Getting even this small amount of spam annoyed me and started my crusade to eliminate spam and viruses from my user’s email. This was a very up-hill battle until a friend suggested I try MailScanner.

I installed MailScanner in the spring of 2001. Since then no email viruses have gotten through our gateways and the flood of spam has been reduced to a very small trickle. I can absolutely state that spam and viruses can be defeated simply by deploying MailScanner. Of the five hundred spams and two or three viruses I receive every day, I see 4 or 5 spams clearly marked as Spam and one or two more that the spam filters miss. I haven’t gotten a virus in years.

What makes MailScanner such a powerful tool? MailScanner is an open source application that controls a variety of additional open source applications to scan email, detecting spam and viruses. By installing and configuring MailScanner, you can also configure and coordinate the actions of:

• ClamAV Anti-virus	http://www.clamav.net
• BitDefender Anti-virus	http://www.bitdefender.com
• SpamAssassin	http://www.spamassassin.org
• Pyzor	http://pyzor.sourceforge.net
• Razor2	http://razor.sourceforge.net
• DCC	http://www.rhyolite.com/anti-spam/dcc
• Additional Antivirus Scanners	Your choices of any of the 23 supported commercial virus scanners

MailScanner's ability to coordinate all of these anti-spam and antivirus tools provides us with the most secure and highly configurable email gateway available at any price. Did I mention that as open source software, MailScanner and all of the related applications are available for download at no cost?

MailScanner is compatible with all of the most popular email transport software:

• Sendmail	http://www.sendmail.org
• Postfix	http://www.postfix.org/
• Exim	http://www.exim.org/
• Zmailer	http://www.zmailer.org/
• Qmail	http://www.qmail.org/

And MailScanner's popularity and large installed base has inspired other developers to create tools for use with MailScanner.

• MailWatch for MailScanner	http://mailwatch.sourceforge.net/
• mailscanner-mrtg	http://mailscannermrtg.sourceforge.net/
• MailScanner webmin module	http://lushsoft.dyndns.org/mailscanner-webmin/

Distributions of MailScanner and most related applications are available for many operating systems including Linux, Solaris and FreeBSD.

But the real strengths of MailScanner are its very active user community and Julian Field. The quick and accurate MailScanner support I've received by posting my problems to the user community at mailscanner@jiscmail.ac.uk beats any commercial product support I've experienced. The continual feedback from the user community has contributed to the rapid development of new MailScanner features. Still all of this feedback would go nowhere without the dedication and hard work of Julian Field.

As the inventor and principal developer of MailScanner, Julian Field has put in long hours over the last four years to make MailScanner the world's most popular email gateway software. Additionally he's always monitoring the mail list; answering questions, looking for new feature ideas and quickly responding to new email security threats. Today our email is quite a bit safer thanks to Julian's efforts.

Thank you Julian.

Part 1 User Guide

Steve Swaney

Contents

Introduction

Congratulations, your email will now be protected by the world's most widely used and respected email scanning software, MailScanner

A Brief History of MailScanner

MailScanner is a highly respected open source email security system. It is used at over 30,000 sites around the world, protecting top government departments, commercial corporations and educational institutions. This technology is fast becoming the standard email solution at many ISP sites for virus protection and spam filtering.

MailScanner scans all e-mail for viruses, spam and attacks against security vulnerabilities and plays a major part in the security of a network. To securely perform this role, it must be reliable and trustworthy. The only way to achieve the required level of trust is to be open source, an approach the commercial suppliers are not willing to take. By virtue of being open source, the technology in MailScanner has been reviewed many times over by some of the best and brightest in the field of computer security, from around the world.

MailScanner has been developed by Julian Field at a world-leading Electronics and Computer Science Department at the University of Southampton.

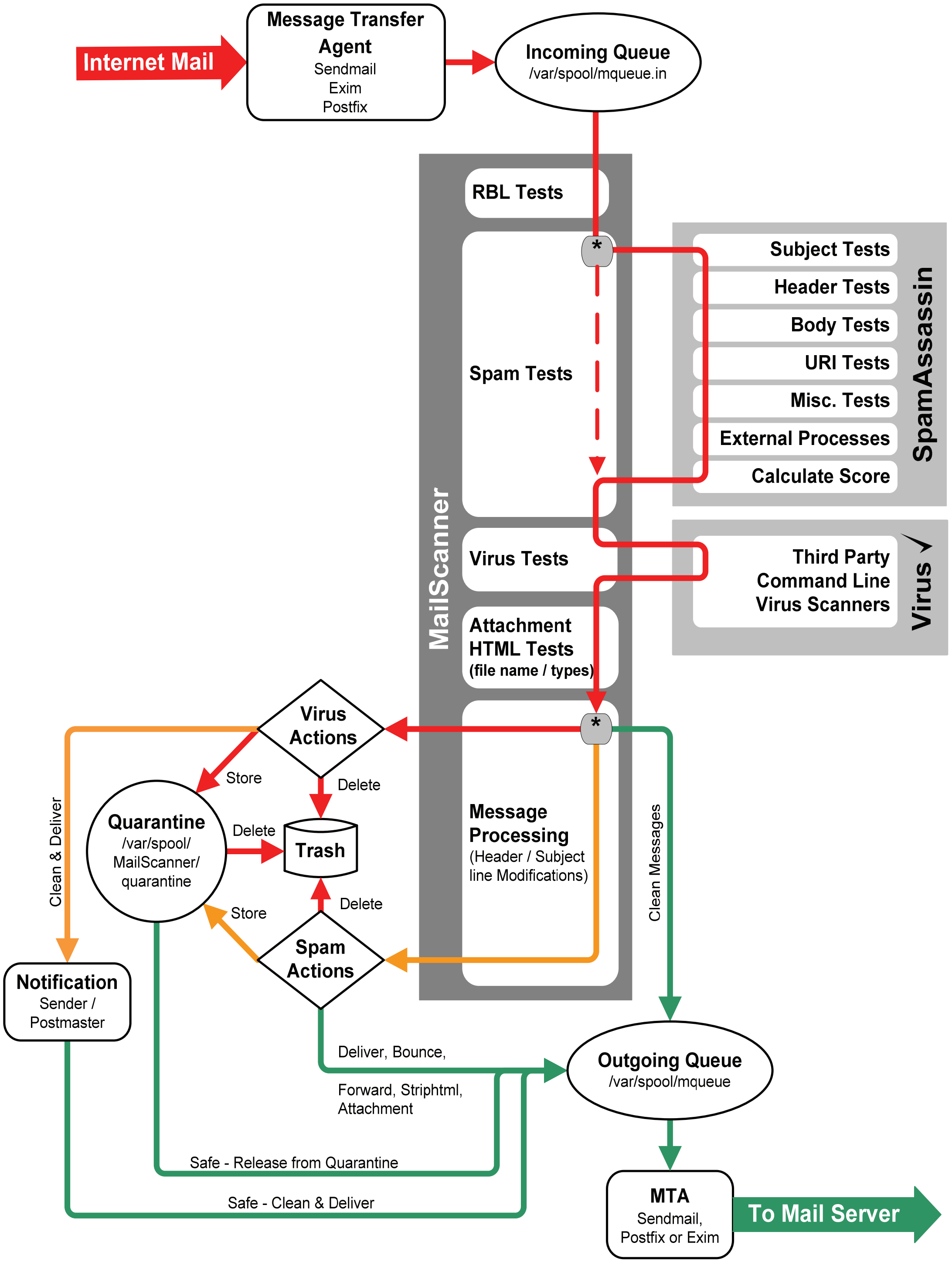
How MailScanner Works

MailScanner provides the engine used to scan incoming emails, detecting security attacks, viruses and spam.

Email is accepted and delivered to an incoming queue directory. When messages are waiting in the incoming spool directory, MailScanner processes the waiting messages and then delivers the cleaned messages to the outgoing queue directory where they are picked up and delivered normally. Only after the messages are delivered to the outgoing queue directory are they deleted from the incoming spool directory. This ensures that no mail is lost, even in the event of unexpected power loss, as the system always has an internal copy of all messages being processed.

The MailScanner engine initiates email scanning by starting, in most configurations, two instances of the Mail Transport Agent (MTA). The first MTA instance is started in daemon mode to accept incoming email. Email is accepted and simply delivered to an incoming queue directory. The second MTA instance is also started in daemon mode and watches an outgoing queue directory for scanned and processed messages that need to be delivered.

To accomplish these scanning and processing tasks, MailScanner starts a configurable number of MailScanner child processes. Typically there are five child processes which examine the incoming queue at five second intervals and select a number of the oldest messages in the queue for batch processing. The number of child processes and the time interval between them is configurable and should be set based on the gateway system’s speed, memory, number of processors and other application loading.



MailScanner Process Diagram Fig 1.

Typically, once a MailScanner child process has found a batch of emails in the incoming queue and MailScanner has been configured to use RBLs, it first runs a series of Real-time Black List (RBL) tests on each message. If the IP address of the sender's mail server or mail relay servers matches a definable number of RBLs, the message may by marked as definitely spam and no further tests are performed to save processing time.

If the message passes the MailScanner RBL tests it is passed to SpamAssassin which uses heuristic, Bayesian and other tests to determine the spam level of the message (see Figure 1.)

SpamAssassin actually assigns a numerical value to each test that is used on the message. SpamAssassin also examines the site specific white lists (not spam) and black lists (is spam). If the sender, system or domain of the message sender is on either list, a very high (black list), or a very low (negative score) is assigned to the message. SpamAssassin calculates the final spam score for each message at the end of these tests.

MailScanner may be configured to use one or more of seventeen commercial or open source virus scanners. MailScanner may be configured to scan for viruses inside of zip files. If a virus is detected at this point, the message is marked as containing a virus.

Once virus detection is complete, the MailScanner child process examines the filename and file types of any email attachments against site configurable rule sets. Virtually any type or name of attachments can be blocked or passed depending on how MailScanner has been configured. The message is also examined to see if the body contains possibly dangerous HTML content such as:

- IFrame tags
- <Form> tags
- <Object Codebase=...> tags

Configurable options allow logging, passing, deleting or disarming these HTML content tags.

After this stage of the processing, MailScanner has all the information needed to modify, deliver, reject or quarantine the message. This final message processing depends on the message content and the MailScanner configuration settings.

If a virus is detected, MailScanner can send (or not send):

- A customized message to the sender of the virus (normally not desireable)
- A customized message to the recipient of the virus
- The disarmed and sanitized message to the recipient
- The message and the virus to quarantine
- The disinfected or cleaned message to the recipient

Every message has now received a "spam score". MailScanner can be configured to discern between different levels spam cores:

- Not spam, i.e. spam score < 6

- Spam, i.e. spam score =>6 and <=10
- High scoring spam, i.e. spam score >10

For each of the not spam or spam levels listed above, MailScanner can perform any combination of the following options:

- Delete - delete the message
- Store - store the message in the quarantine
- Bounce - send a rejection message back to the sender
- Forward user@domain.com - forward a copy of the message to user@domain.com
- Strip HTML - convert all in-line HTML content to plain text.
- Attachment - Convert the original message into an attachment of the message.
- Deliver - deliver the message as normal

These options (and most other message processing options) are configurable by the To: or From: address for specific domains, senders or recipients. You can combine a From: and an To; in a single rule, for more information on Rulesets, see Chapter 5. Spam and virus detection may be turned on or off depending on the To: or From: address of specific domains, senders or recipients. This granularity is accomplished using Rulesets (For more information on Rulesets, see Chapter 5).

All mail or mail to specific recipients or domains may also be archived.

Many other alterations may be made to individual messages depending on the site’s preferences:

Various levels and types of spam scores may be added to the header of the message

Customizable “X-“style messages may be added to the header of the message

Subject: lines may be customized depending on Virus, attachment or spam score detected

Messages may be signed with site customized footers

Reports to administrators, senders and recipients may be customized (standard reports are available in fifteen different languages)

MailScanner also provides the additional features and functions required for ease of email gateway administration and maintenance:

- Simple, automated installation
- Sensible defaults for most sites
- Automated updating of virus definitions for all supported virus scanning engines
- Configurable cleaning options for quarantined messages
- Very simple application updating

Chapter 2

Planning the Installation

Taking a little time to plan out the installation of MailScanner will ensure that the process is straight forward and successful.

Gather the following information prior to installing:

root password: ____

IP address for MailScanner gateway: __________________________

Netmask for MailScanner gateway: ____________________________

Name Server IP address: ____________________________________

Domain names for which you process email: _____________________

Current mail server hostname(s): ______________________________

System Requirements

System requirements are dependent on:

- Number of email processed daily
- Number of virus scanners used
- Number of MailScanner features enabled
- Number of SpamAssassin features and rules enabled
- Number of related Applications installed

It is important to note that the number of messages per hour that the system can process is directly dependent on the type of hardware used. Larger volume sites will need to use more powerful hardware to handle their larger volume of mail.

For example, a Pentium II with 256MB of RAM running MailScanner, SpamAssassin, DCC, Pyzor, Razor, MailWatch, Vispan and MailScanner-MRTG can process approximately 5,000 messages per day.

A System with dual 2.4 GHz Xeon processors, 2 GB of RAM and 15,000 rpm SCSI drives and running only MailScanner and SpamAssassin can process approximately 1,500,000 messages per day.

Some further examples of actual system capacities may be found at:

http://www.mailscanner.biz/maq.html

Proper operation of the MailScanner software requires that it run on a server with a fixed IP address. This is typically a requirement of any mail server, and to the outside world, the MailScanner gateway appears as a mail server. For most email servers to accept email from your email gateway, it must also have a reverse name lookup entry (PTR) record.

Firewall and Network Requirements

The MailScanner gateway will need direct access to the Internet for ports:

Sendmail	tcp port 25
DNS	tcp/udp port 53 (outbound. Inbound and outbound if you are also running a DNS server on the gateway)

Related applications, if installed will also need NAT access to the internet. The most common ports that may need to be enabled on the firewall are:

Razor2	tcp ports 2703 and 7 (outbound)
Pyzor	udp port 24441 (outbound)
DCC	udp port 6277 (outbound)

Installing Red Hat Enterprise Linux

> Please note that this manual currently only covers the installation of MailScanner for Red Hat Linux (other RPM-based Linux distributions will be similar)

While MailScanner can be installed on most versions of Linux and UNIX operating systems, this first version of the MailScanner Manual includes only installation instructions for Red Hat Linux. Instruction for installing MailScanner on other operating systems may be found at:

http://www.mailscanner.info/install/

Before the MailScanner may be installed, the Linux Operating system must be installed. Step by step instructions for installing Red Hat Enterprise Linux are included in Appendix A. Installation of other Linux Operating System will be similar.

> After installing Red Hat Linux you must edit the file /etc/sysconfig/i18n to change the lines:
>
> ```
> LANG="en_US.UTF-8"
> SUPPORTED="en_US.UTF-8:en_US:en"
> ```
>
> To:
>
> ```
> LANG="en_US"
> SUPPORTED="en_US.UTF-8:en_US:en"
> ```

Note the example shown above is for US English installations. You may need to make similar edits for other languages.

Failure to make these changes may result in MailScanner and SpamAssassin installation errors.

Installing the Message Transfer Agent

Before the MailScanner may be installed, your Message Transfer Agent (MTA) must be installed, configured and tested. MailScanner supports several MTAs and the choice of which one to use is up to the user. The three most popular MTA are:

- Sendmail
- Exim
- Postfix

For other information on other supported MTAs please visit:

http://www.mailscanner.info/install/

Installing sendmail

Instructions for obtaining, installing and configuring sendmail may be found at:

http://www.sendmail.org/

Installing Exim

Instructions for obtaining, installing and configuring Exim may be found at:

http://www.exim.org/

Installing Postfix

Instructions for obtaining, installing and configuring Postfix may be found at:

http://www.postfix.org/

Installing MailScanner

Please note that this manual currently only covers the installation of MailScanner for Red Hat Linux (and other RPM-based Linux distributions)

MailScanner software may be downloaded from:

http://www.mailscanner.info/downloads.shtml

1. Login to your server as root.
2. This step is not really necessary but it is useful to keep your installation packages and installed software download in one location. Create an installation directory, i.e.:
   ```
   mkdir /home/install
   ```

cd to installation directory:

```
cd /home/install
```

Download the latest Stable version of MailScanner software for Red Hat Linux (and other RPM-based Linux distributions) from the URL listed above into the installation directory

3. Unpack the distribution:

```
mkdir build
cd build
tar zxf ../ MailScanner-<version_number>.tar.gz
cd MailScanner-<version_number>
./Update-MakeMaker.sh
./install.sh
```

4. The install.sh script should finish without major errors. This is typically all that needs to be done to install MailScanner on a Linux rpm based distribution. If you experience errors or problems at this stage, please see Chapter 7, Tips Tuning and Troubleshooting.

5. Stop the MTA from starting at boot time:
   ```
   chkconfig --level all sendmail off
   ```

6. Setup MailScanner to start at boot time:
   ```
   chkconfig --level 345 MailScanner on
   ```

7. Start MailScanner:
   ```
   service sendmail stop
   service MailScanner start
   ```

8. Check the mail logs to ensure that MailScanner has started properly with no Errors.

Installing SpamAssassin

SpamAssassin software may be downloaded from:

http://www.spamassassin.org/downloads.html

The version that should be installed with MailScanner is:

SpamAssassin(tm) in tar.gz format.

Do not install the rpm version available on the SpamAssassin Site. There have been many problems reported after installing SpamAssassin from this rpm.

Before beginning the installation, you should review the SpamAssassin installation documentation available at:

http://au2.spamassassin.org/full/2.6x/dist/INSTALL

Login to your server as root.

1. If you created the installation directory as recommended above:

   ```
   cd /home/install
   ```

2. Download the SpamAssassin in the tar.gz format. from the URL listed above into the `/home/install` directory

   ```
   cd build
   tar zxf ../ Mail-SpamAssassin-<version_number>.tar.gz
   cd Mail-SpamAssassin-<version_number>
   perl MakeFile.PL
   make
   make test
   make install
   ```

   These steps should complete without errors. This is typically all that needs to be done to install SpamAssassin for use with MailScanner. If you experience errors or problems at this stage, Please see Chapter 7, Tips Tuning and Troubleshooting.

Chapter 3

MailScanner Configuration

MailScanner ships with sensible defaults but the MailScanner default configuration should be examined in detail before placing the system into production.

MailScanner Files

MailScanner is configured and controlled by editing text files. The most important files are located it the `/etc/MailScanner` directory (Linux rpm version):

`/etc/MailScanner/MailScanner.conf` Contains the MailScanner configuration. Most of your configuration work will involve changing the values in this file to match your site's need.

/`etc/MailScanner/spam.assassin.prefs.conf` contains the SpamAssassin configuration values as:

```
Parameter <value>
```

All SpamAssassin configuration values should be placed in this file. All site SpamAssassin Rulesets should be placed in `/etc/mail/spamassassin` (default location) or the locations specified by

SpamAssassin Site Rules Dir = /etc/mail/spamassassin

In the `MailScanner.conf file`.

Please note that MailScanner ships with reasonable default values for SpamAssassin but you are advised to visit:

http://www.spamassassin.org/doc/Mail_SpamAssassin_Conf.html

And examine other configuration options.

Other configurable files (Linux rpm version) are located it the

- `/etc/MailScanner/reports/<your_language>` directories. The files located here should be edited to reflect your site name and preferences.
- /etc/MailScanner/rules directories. This directory contains the default Rulesets and your custom Rulesets. Please see Chapter 5, Advanced Configuration using Rulesets

Getting Started with MailScanner Configuration

The following steps should be followed in order to quickly configure MailScanner and place it in production:

1. Edit the `MailScanner.conf` file to reflect your sites preferences
2. Review and edit if necessary the SpamAssassin site preferences file `spam.assassin.prefs.conf`
3. Edit the files in `/etc/MailScanner/reports/<your_language>` directory and correct for your site information.

Before you start

Editing the MailScanner.conf file to reflect your sites preferences involves changing values or adding Rulesets. The format of this file is simply:

- # - Lines starting with a # are comments. While you may add comments you should note that they will be lost if you automatically upgrade MailScanner using the upgrade_MailScanner_conf script
- MailScanner configuration values may be:

```
Parameter = <value>
```

or

```
Parameter = <pointer to a ruleset>
```

or

```
Parameter = <space separated list>
```

Before editing the MailScanner.conf file please note:

- If your directories are symlinked (soft-linked) in any way, please put their *real* location as the value, not a path that includes any links. You may get some very strange error messages from some virus scanners if you don't.
- A lot of the settings can take a Ruleset as well as just simple values. These Rulesets are files containing rules which are applied to the current message to calculate the value of the configuration option. The rules are checked in the order they appear in the Ruleset. Please see Chapter 6 for additional information.

In addition to Rulesets, you can now include your own functions as values. Please locate and look at the file `MyExample.pm` located in `/usr/lib/MailScanner/MailScanner/CustomFunctions` and create your own `MyFunctions.pm` in the same directory. In this file, you can add your own "value" function and an Initvalue function to set up any global state you need such as database connections. Then for a setting below, you can put:

```
Configuration Option = &ValueFunction
```

where ValueFunction is the name of the function you have written in `MyFunctions.pm`.

MailScanner.conf Parameters

Below we will list the all of the configurable parameters in the MailScanner.conf file in the order in which they appear in the file. The format will be:

`Parameter = default value`

A description of what the rule does.

A list of the possible options and the results of specifying the specific option

General settings

`%report-dir% = /etc/MailScanner/reports/en`

Sets directory containing the language for reports used at your site.

Look in /etc/MailScanner/reports for a listing of the supported languages

An example: If you want to use French for your MailScanner reports, set:

```
%report-dir% = /etc/MailScanner/reports/fr
```

This setting may point to a Ruleset

`%etc-dir% = /etc/MailScanner`

Sets the top directory containing the MailScanner configuration files.

This should not be changed for the Linux rpm distribution. It will typically need to be changed for other distributions, i.e. Solaris, TRU64.

`%org-name% =`

Enter a short identifying name for your organization. This value will be used to create unique X-MailScanner headers which identify your organization.

Sites with multiple servers should use an identical value on all servers within the site. This will avoid adding multiple redundant headers where mail has passed through several servers within your organization.

This must be changed to identify your site.

Note: This value MUST NOT contain any white spaces or periods

`%rules-dir% = /etc/MailScanner/rules`

Sets the top directory containing the MailScanner Rulesets. Your custom Rulesets should be placed in this directory.

This should not be changed for the Linux rpm distribution. It will typically need to be changed for other distributions, i.e. Solaris, TRU64

System Settings

`Max Children = 5`

This is the number of MailScanner processes to run at a time. There is no point increasing this figure if your MailScanner server is happily keeping up with your mail traffic.

Each process will consume at least +20MB of ram and using additional SpamAssassin rule sets can increase this to +40MB. If you are running on a server with more than 1 CPU, or you have a high mail load (and/or slow DNS lookups) then you should see better performance if you increase this figure. As a very rough guide you can try 5*(number of CPUs) for multiple CPU systems.

It is important to ensure that there is enough ram for all processes. Performance will suffer greatly if the Scanner Nodes run out of ram and begin to swap.

`Run As User = <blank>`

User to run MailScanner processes as (not normally used for sendmail). If you want to change the ownership or permissions of the quarantine or temporary files created by MailScanner, please see the "Incoming Work" settings later in this document.

Other Possible values: mail postfix

`Run As Group = <blank>`

Group to run MailScanner processes as (not normally used for sendmail).

Other Possible values: mail postfix

`Queue Scan Interval = 5`

The time (in seconds) between the start up of each MailScanner child process. If you have a quiet mail server, you might want to increase this value so it causes less load on your server, at the cost of slightly increasing the time taken for an average message be processed.

Other Possible values: integers

`Incoming Queue Dir = /var/spool/mqueue.in`

Set location of incoming mail queue. This can be any one of

- A directory name
  Example: `/var/spool/mqueue.in`
- A wildcard giving directory names
  Example: `/var/spool/mqueue.in/*`
- The name of a file containing a list of directory names, which can in turn contain wildcards.
  Example: `/etc/MailScanner/mqueue.in.list.conf`

This should not be changed for the Linux rpm distribution. It may need to be changed for other distributions or with other prepackaged applications servers, i.e. Ensim

`Quarantine Dir = /var/spool/MailScanner/quarantine`

This sets where to store infected and message attachments (if they are kept).

This should not be changed for the Linux rpm distribution. It may need to be changed for other distributions

`PID file = /var/run/MailScanner.pid`

This sets where to store the process id number used to stop MailScanner processes

This should not be changed for the Linux rpm distribution. It may need to be changed for other distributions.

`Restart Every = 14400`

This setting determines how often (in seconds) MailScanner will restart the MailScanner processes. This is done to avoid resource leaks. When MailScanner processes are restarted, the configuration files are re-read. This restart will not restart the MTA, only MailScanner.

Typically this setting does not need to be changed

`MTA = sendmail`

This should be set to the MTA used on your gateway. If you are using postfix, then see the SpamAssassin User State Dir parameter later in this documentation.

Other Possible values: postfix exim or zmailer

`Sendmail2 = sendmail2`

This setting is provided for Exim users. It is the command used to attempt delivery of outgoing cleaned/disinfected messages. This is not usually required for sendmail. This can also be the filename of a ruleset. i.e. for Exim users:

```
Sendmail2 = /usr/sbin/exim -C /etc/exim/exim_send.conf
```

This setting typically only should be changed when using exim

Incoming Work Dir Settings

You should not normally need to touch Incoming Work Dir Settings unless you are using ClamAV and need to be able to use the external archive un-packers instead of ClamAV's built-in ones.

`Incoming Work User = <blank>`

`Incoming Work Group = <blank>`

These settings should be changed only if you want to create the temporary working files so they are owned by a user other than the **`Run As User`** setting discussed earlier. Note: If the **`Run As User`** setting is not "root" then you cannot change the user but may still be able to change the group, if the **`Run As`**

`User` is a member of both of the groups **`Run As Group`** and **`Incoming Work Group`**.

Permissible values are system usernames, i.e. root, postfix

Typically this setting does not need to be changed

`Incoming Work Permissions = 0600`

This settings would be changed only if you want processes running under the same *group* as MailScanner to be able to read the working files (and list what is in the directories, of course), set to 0640. If you want *all* other users to be able to read them, set to 0644. Typical use: external helper programs of virus scanners (notably ClamAV).

Permissible values are those allowed by the `chmod` command

Typically this setting does not need to be changed.

Use with care, you may well open security holes

Quarantine and Archive Settings

If you are using a web interface to allow users to manage their quarantined files, you might want to change the ownership and permissions of the quarantine files so that they can be read and/or deleted by the web server. Don't touch this unless you know what you are doing!

`Quarantine User = <blank>`

`Quarantine Group = <blank>`

These settings would be changed only if you want to create the quarantine/archive so the files are owned by a user other than the Run As User discussed earlier. Typically this is done to allow an application such as MailWatch to release messages from quarantine.

Typically this setting does not need to be changed but if it does, the typical changes are

`Quarantine User = root` and **`Quarantine Group = apache`**.

`Quarantine Permissions = 0600`

These settings would be changed only if you want processes running under the same group as MailScanner to be able to read the quarantined files and list what is in the directories. set to 0640. If you want all other users to be able to read them, set to 0644. For a detailed description, refer to `man 2 chmod`.

Typical use: let the web server have access to the files so users can download them if they really want to.

Typically this setting does not need to be changed but if it does, i.e. for MailWatch, the typical changes are 0640

Use with care, you may well open security holes

Processing Incoming Mail

`Max Unscanned Bytes Per Scan = 100000000`

This setting controls total size of un-scanned messages, in bytes, that each MailScanner child process will pick up and process from the incoming mail queue. If the Scanner Nodes have substantial unused memory, increasing this value can increase message throughput, as long as the system's CPU(s) is not overloaded.

Typically this setting does not need to be changed.

`Max Unsafe Bytes Per Scan = 50000000`

This setting controls the total size of potentially infected messages, in bytes, that each MailScanner child process will pick up and process from the incoming mail queue. On a system with plenty of unused memory, increasing this value can increase message throughput, as long as the system's CPU(s) is not overloaded.

Typically this setting does not need to be changed.

`Max Unscanned Messages Per Scan = 30`

This setting controls maximum number of un-scanned messages that each MailScanner child process will pick up and process from the incoming mail queue. On Scanner Nodes with plenty of unused memory, increasing this value can increase message throughput, as long as the system's CPU(s) is not overloaded.

Typically this setting does not need to be changed.

`Max Unsafe Messages Per Scan = 30`

This setting controls the maximum number of potentially infected messages that each MailScanner child process will pick up and process from the incoming mail queue. On Scanner Nodes with plenty of unused memory, increasing this value can increase message throughput, as long as the system's CPU(s) is not overloaded.

Typically this setting does not need to be changed

`Max Normal Queue Size = 800`

If more than this number messages are found in the incoming queue, MailScanner will switch to an "accelerated" mode of processing messages. This will cause it to stop scanning messages in strict date order, but in the order it finds them in the queue. If your queue is bigger than this size a lot of the time, then some messages could be greatly delayed. So treat this option as "in emergency only" option.

Possible values = integers

Typically this setting does not need to be changed

`Maximum Attachments Per Message = 200`

This setting controls the maximum number of attachments allowed in a message before it is considered to be an error. Some email systems, if bouncing a message between 2 addresses repeatedly, add information about each bounce as an attachment, creating a message with thousands of attachments in just a few minutes. This can slow down or even stop MailScanner as it uses all available memory to unpack these thousands of attachments.

Possible values = integers

This can also be the filename of a ruleset.

Typically this setting does not need to be changed

`Expand TNEF = yes`

This setting determines if TNEF attachments are to be expanded using an external program or a Perl module. This should be "yes" unless the scanner you are using is Sophos, McAfee or a virus scanner that has the built-in ability to expand the message. If set it to no, then the filenames within the TNEF attachment will not be checked against the filename rules.

Typically this setting does not need to be changed unless you are using the Sophos of McAfee virus scanners.

`Deliver Unparsable TNEF = no`

Some versions of Microsoft Outlook generate un-parsable Rich Text format attachments. If you want to deliver these bad attachments anyway set this value to yes. This introduces a slight risk of a virus getting through, but if you have complaints from Outlook users, you may need to set this value to yes.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`TNEF Expander = /usr/bin/tnef --maxsize=100000000`

This setting determines which MS-TNEF expander is used.

This is EITHER the full command (including maxsize option) that runs the external TNEF expander binary, OR the keyword `internal` which will cause MailScanner to use the Perl module that does the same job. They are both provided as we are unsure which one is faster and which one is capable of expanding more file formats (there are plenty!)..

The --maxsize option limits the maximum size that any expanded attachment may be. It helps protect against Denial of Service attacks in TNEF files.

If this setting is changed, it is typically set to internal.

This cannot be the filename of a Ruleset.

Typically this setting does not need to be changed.

`TNEF Timeout = 120`

This setting controls the length of time (in seconds) that the TNEF expander is allowed to run on a single message.

Permissible values = integers

Typically this setting does not need to be changed.

`File Command = #/usr/bin/file`

Where the "file" command is installed. The file command is used for checking the content type of files, regardless of their filename. The default value of `#/usr/bin/file` actually disables filename checking (note the # starts a comment)

To enable filename checking set the value to /usr/bin/file (on most systems). The location of the file command varies with different operating systems.

This setting is often changed to force file type settings.

`File Timeout = 20`

This setting controls the length of time (in seconds) that the file is allowed to run on a single message.

Permissible values = integers

Typically this setting does not need to be changed.

`Maximum Message Size = 0`

This setting controls the maximum size, in bytes, of any message including the headers. If this is set to zero, then no size checking is done. If this is set to a value, messages exceeding this value, in bytes, will be blocked.

This can also be the filename of a ruleset, so you can have different settings for different users. You might want to set this to be small for dialup users so their email applications don't time out downloading huge messages.

Permissible values = integers

Typically this setting should not to be changed.

`Maximum Attachment Size = -1`

This setting controls the maximum size, in bytes, of any attachment in a message. If this is set to zero, effectively no attachments are allowed. If this is set less than zero, then no size checking is done. Attachments that exceed this value, in bytes, will be blocked.

This can also be the filename of a ruleset, so you can have different settings for different users. You might want to set this quite small for large mailing lists so they don't get deluged by large attachments.

Typically this setting does not need to be changed.

`Maximum Archive Depth = 3`

The maximum depth to which zip archives will be unpacked, to allow for filenames and filetype checking within zip archives. To disable this feature set this to 0.

A common useful setting is to `Maximum Archive Depth = 0`, and set `Allow Password-Protected Archives = no`. This will block password-protected archives but does not do any filename or filetype checks on the files within the archive. This allows users to receive files that would normally be blocked by filename and filetype rules if they are compressed before sending. Virus scanning will still occur on files within the archive

Often this setting is changed to 0.

`Find Archives By Content = yes`

Find zip archives by filename or by file contents? Finding zip archives by content is far more reliable, but means that users cannot avoid zip file checking by renaming the file from ".zip" to "_zip.

Only set this to no (i.e. check by filename only) if you don't want to reliably check the contents of zip files. Note this does not affect virus checking, but it will affect all the other checks done on the contents of the zip file.

This can also be the filename of a ruleset.

Typically this setting does not need to be changed

Virus Scanning and Vulnerability Testing

`Virus Scanning = yes`

This setting actually switches on/off the processing of all the email messages for virus checking and MailScanner security checks. If you just want to switch off actual virus scanning, then set Virus Scanners = none (see below) instead. If you do not have a license for a commercial virus scanner you should consider installing ClamAV, an open source virus scanner.

This can also be the filename of a ruleset. If you want to be able to switch scanning on/off for different users or different domains, set this to the filename of a Ruleset and create the corresponding Ruleset.

Typically this setting does not need to be changed

`Virus Scanners = none`

If you want to use a single virus scanners, then this should be the name of the of virus scanner, for example;

```
Virus Scanners = sophos
```

If you want to use multiple virus scanners, then this should be a space-separated list of virus scanners. For example;

```
Virus Scanners = sophos f-prot mcafee
```

Make sure that you check that the base installation directory in the 3rd column of `virus.scanners.conf` file matches the location you have installed each of your virus scanners. The defaults provided in the `virus.scanners.conf` file assumes installation locations recommended by each of the virus scanner installation instructions.

Please see Appendix B, Installing Third party Virus Scanners, for instructions on configuring the many virus scanning engines supported by MailScanner.

Note for McAfee users: do not use any symlinks with McAfee at all. It is very strange but may not detect all viruses when started from a symlink or scanning a directory path that includes including symlinks.

This setting should be changed to match the virus scanner or scanners used at your site.

`File Timeout = 300`

This setting controls the length of time, in seconds, the virus scanner is allowed to run on a single message.

Permissible values = integers

Typically this setting does not need to be changed.

`Deliver Disinfected Files = no`

This setting controls whether or not to disinfect infected attachments and then deliver the cleaned attachment. "Disinfection" involves removing viruses from files (such as removing macro viruses from documents). "Cleaning" is the replacement of infected attachments with "VirusWarning.txt" text attachments.

Since less than 1% of viruses in the wild can be successfully disinfected and since macro viruses are now a rare occurrence, the default is set to no as it results in a significant performance improvement.

Typically this setting does not need to be changed.

`Silent Viruses = HTML-IFrame All-Viruses`

Strings listed here, separated by white space, will be searched for in the output of the virus scanner(s). These strings are used to list which viruses should be handled differently from other viruses. If a virus name is given here, then

- The sender will not be warned that they sent the message.
- No attempt at true disinfection will take place (but it will still be "cleaned" by removing the nasty attachments from the message)
- The recipient will not receive the message, unless the **`Still Deliver Silent Viruses`** option is set (see below).

The only words that can be put in this list are the 5 special keywords:

- **`HTML-IFrame`**: inserting this will stop senders being warned about HTML IFrame tags, when they are not allowed.
- **`HTML-Codebase`**: inserting this will stop senders being warned about HTML Object Codebase tags, when they are not allowed.
- **`HTML-Form`**: inserting this will stop senders being warned about HTML Form tags, when they are not allowed.

- `Zip-Password`: inserting this will stop senders being warned about password-protected zip files, when they are not allowed (This keyword is not needed if you include All-Viruses)
- `All-Viruses`: inserting this will stop senders being warned about any virus, while still allowing you to warn senders about HTML-based attacks. This includes Zip-Password so you don't need to include both.

The default of All`-Viruses` means that no senders of viruses will be notified (since the sender address is almost always forged), but anyone who sends a message that is blocked for other reasons will still be notified.

This setting may also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Still Deliver Silent Viruses = no`

Still deliver (after cleaning) messages that contained viruses listedin the above option ("Silent Viruses") to the recipient?

Setting this to "yes" is good when you are testing everything, and because it shows management that MailScanner is protecting them, but it is bad because they have to filter/delete all the incoming virus warnings.

Note: Once you have deployed this into "production" use, you should set this option to "no" so you don't bombard thousands of people with useless messages they don't want!

This setting may also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Non-Forging Viruses = Joke/ OF97/ WM97/ W97M/`

Strings listed here, separated by white space, will be searched for in the output of the virus scanner(s). This works to achieve the opposite effect of the `Silent Viruses` setting above. If a string here is found in the output of the virus scanners, then the message will be treated as if it were not infected with a "Silent Virus". If a message is detected as both a silent virus and a non-forging virus, then the non-forging status will override the silent status. In simple terms, you should list virus names (or parts of them) that you know do *not* forge the From address.

A good example of this is a document macro virus or a Joke program. Another word that can be put in this list is the special keyword Zip-Password. Inserting this will cause senders to be warned about password-protected zip files, when they are not allowed. This will over-ride the All-Viruses setting in the list of Silent Viruses setting described above.

Typically this setting does not need to be changed.

`Block Encrypted Messages = no`

This setting can stop encrypted messages from being sent from your site. This is useful if you do not want users to be able to send encrypted messages.

This can be a ruleset so you can block encrypted message to certain domains or from specific users.

Typically this setting does not need to be changed.

`Block Unencrypted Messages = no`

This setting will allow only encrypted messages to be set sent from your site. This is useful if you need to enforce encryption for all messages sent from your domain.

This can be a ruleset so you can force encryption to specific domains.

Typically this setting does not need to be changed.

`Allow Password-Protected Archives = no`

This setting can stop password-protected files from being received by your site. Leaving this set to "no" is a good way of protecting against all the protected zip files used by viruses.

This can be a ruleset so you can block and password-protected zip files from certain domains or to permit password-protected zip files to be sent to specific users.

Typically this setting does not need to be changed.

Options specific to Sophos Anti-Virus

`Allowed Sophos Error Messages = <blank>`

Anything on the next line that appears in brackets at the end of a line of output from Sophos will cause the error/infection to be ignored. Use of this option is dangerous, and should only be used if you are having trouble with Sophos corrupting PDF files. If you need to specify more than one string to find in the error message, then put each string in quotes and separate them with a comma. For example:

```
Allowed Sophos Error Messages = "corrupt", "format not
supported"
```

Typically this setting does not need to be changed.

`Sophos IDE Dir = /usr/local/Sophos/ide`

This sets the directory (or a link to it) containing all the Sophos *.ide files. This is only used by the "sophossavi" virus scanner, and is irrelevant for all other scanners.

Typically this setting does not need to be changed.

`Sophos Lib Dir = /usr/local/Sophos/lib`

This sets the directory (or a link to it) containing all the Sophos *. so libraries. This is only used by the "sophossavi" virus scanner, and is irrelevant for all other scanners.

Typically this setting does not need to be changed.

Options specific to ClamAV Anti-Virus

`Monitors for ClamAV Updates = /usr/local/share/clamav/*.cvd`

This sets the directory to monitor for changes in files size to detect when a ClamAV update has occurred. This setting is only used by the "clamavmodule" virus scanner, not the "clamav" virus scanner.

Typically this setting does not need to be changed.

Removing/Logging dangerous or potentially offensive content

`Dangerous Content Scanning = yes`

Do you want to scan the messages for potentially dangerous content? Setting this to "no" will disable all the content-based checks except Virus Scanning, Allow Partial Messages and Allow External Message Bodies.

This setting may also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Allow Partial Messages = no`

Do you want to allow partial messages, which only contain a part of the attachments, not the entire attachment? There is absolutely no way to scan these "partial messages" properly for viruses, since MailScanner never sees all of the attachment at the same time.

Enabling this option can allow viruses through. You have been warned.

This can also be the filename of a ruleset so you can, for example, allow them in outgoing mail but not in incoming mail.

Typically this setting does not need to be changed.

`Allow External Message Bodies = no`

Do you want to allow messages whose body is stored somewhere else on the internet, which is downloaded separately by the user's email package? There is no way to guarantee that the file fetched by the user's email package is free from viruses, as MailScanner never sees it. This feature is only currently supported by Netscape 6 anyway, and the organization using it is the IETF. Changing this setting can expose your end users to attacks which bypass MailScanner and desktop virus scanners.

Enabling this feature is dangerous as it can allow viruses to be fetched from other Internet sites by a user's email package. The user would just think it was a normal email attachment and would have been scanned by MailScanner.

This can also be the filename of a ruleset.

Typically this setting should never be changed.

`Allow IFrame Tags = no`

Do you want to allow <IFrame> tags in email messages? This can be dangerous since it leaves you unprotected against various Microsoft-specific security vulnerabilities, but since many mailing lists use <IFrame> tags, you may want to allow them to avoid end user complaints.

This can also be the filename of a ruleset, so you can allow them from known mailing lists or to be received by specific users.

Available setting values include:

yes	Allow <IFrame> tags in the message
no	remove HTML part of the message containing <IFrame> tags
disarm	Allow <IFrame> tags, but stop these tags from working.

The disarm setting will stop <IFrame> tags from working but preserve the appearance of HTML messages. This is a common setting for many sites.

This setting may also be the filename of a ruleset.

`Log IFrame Tags = no`

Banning <IFrame> tags completely is likely to break some common HTML mailing lists, like the Dilbert daily cartoon. Before you implement any restriction on <IFrame> tags, you may want log the sender of any message containing an <IFrame>, so that you can set the option above to be a ruleset allowing IFrame tags from named "From" addresses and banning all others.

This can also be the filename of a ruleset, so you log <IFrame> from unknown senders and not log them from known senders.

This setting may also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Allow Form Tags = no`

Do you want to allow <Form> tags in email messages? These are commonly use by Phishing attacks.

This can also be the filename of a ruleset, so you can allow <Form> tags from known senders but ban them from everywhere else.

Available setting values include:

yes	Allow <Form> tags in the message
no	Ban messages containing <IFrame> tags
disarm	Allow < Form > tags, but stop these tags from working.

The disarm setting will stop < Form> tags from working but preserve the appearance of HTML messages. This is a common setting for many sites.

This setting may also be the filename of a ruleset.

`Allow Script Tags = no`

Do you want to allow <Script> tags in email messages? These tags are often used to exploit vulnerabilities in email and web browsers.

This can also be the filename of a ruleset, so you can allow < Script > tags from known senders but ban them from everywhere else.

Available setting values include:

- yes Allow < Script > tags in the message
- no Ban messages containing < Script > tags
- disarm Allow < Script > tags, but stop these tags from working.

The disarm setting will stop < Script > tags from working but preserve the appearance of HTML messages. This is a common setting for many sites.

This setting may also be the filename of a ruleset.

`Allow WebBugs = disarm`

Do you want to allow <Img> tags with very small images in email messages? This is a bad idea as these are used as 'web bugs' to find out if a message has been read. It is not dangerous; it is just used to make you give away information.

Available setting values include:

- yes Allow < Img > tags in the message
- disarm Allow < Img > tags, but stop these tags from working.

Disarming can be defeated; it is not 100% safe! Also you cannot block messages containing web bugs as their detection is very vulnerable to false alarms.

This setting may also be the filename of a ruleset.

`Allow Object Codebase Tags = no`

Do you want to allow <Object Codebase=...> tags in email messages? This can be dangerous since it leaves you unprotected against various Microsoft-specific security vulnerabilities, but may be necessary to avoid end user complaints.

This can also be the filename of a ruleset, so you can allow <Object Codebase=...> tags from known senders but ban them from everywhere else.

Available setting values include:

- yes Allow < Object Codebase=...> tags in the message
- no Ban messages containing < Object Codebase=...> tags
- disarm Allow < Object Codebase=... > tags, but stop these tags from working.

The disarm setting will stop <Object Codebase=...> tags from working but preserve the appearance of HTML messages. This is a common setting for many sites.

`Convert Dangerous HTML To Text = no`

This setting interacts with the "Allow ... Tags" options above to produce the following results:

Allow (I-Frame | Codebase)Tags	Convert Dangerous HTML To Text	Action Taken on HTML Message

no	no	Blocked
no	yes	Blocked
disarm	no	Specified HTML tags disarmed
disarm	yes	Specified HTML tags disarmed
yes	no	Nothing, allowed to pass
yes	yes	All HTML tags stripped

Typically this setting does not need to be changed.

`Convert HTML To Text = no`

Do you want to convert all HTML messages into plain text? This is very useful for children or users who are offended by nasty things like pornographic spam.

This can also be the filename of a ruleset, so you can switch this feature on and off for particular users or domains.

Typically this setting does not need to be changed.

Attachment Filename Checking

`Filename Rules = %etc-dir%/filename.rules.conf`

This sets where to find the attachment filename ruleset. This Ruleset is used to accept or reject file attachments based on their name, regardless of whether they are infected or not.

The structure of this file must be:

```
# This is a comment line
# A typical entry line is below
[allow|deny]    <regular expression>   <Log Text>   <User
Report text>
```

Since the Text fields may contain spaces, all fields must be separated by tabs. All fields must exist. Use a “-“(dash) if you want to leave either of the Text fields blank.

Outlook Express allows the second (to the left) of the last extension to be the associated application used to execute the file, so to be safe, very long filenames must be denied regardless of the final extension.

This setting can also be the filename of a ruleset but the Ruleset file name must end in ".rules". Creating such a ruleset will allow you to switch this feature on and off for particular users or domains. See Appendix C, Practical Ruleset Examples, for further instructions.

This setting is often changed to allow certain domains and users to receive specific named attachments.

`Filetype Rules = %etc-dir%/filetype.rules.conf`

This sets where to find the attachment filetype ruleset. This Ruleset is used to accept or reject file attachments based on the type of file, regardless of whether they are infected or not. To disable this feature, set Filetype Rules = to a blank string.

The structure of this file must be:

```
# This is a comment line
# A typical entry line is below
[allow|deny]    <regular expression>   <Log Text>   <User
Report text>
```

Since the Text fields may contain spaces, all fields must be separated by tabs. All fields must exist. Use a “-“(dash) if you want to leave either of the Text fields blank.

This setting can also be the filename of a ruleset but the Ruleset file name must end in ".rules". Creating such a ruleset will allow you to switch this feature on and off for particular users or domains. . See Appendix C, Practical Ruleset Examples, for further instructions.

This setting is often changed to allow certain domains and users to receive specific types of attachments.

Reports and Responses

`Quarantine Infections = yes`

Do you want to store copies of the infected attachments and messages?

This can also be the filename of a ruleset, so you can switch this feature on and off for specific users or domains.

Typically this setting does not need to be changed.

# There is no point quarantining most viruses these days as the infected

`Quarantine Silent Viruses = no`

These messages contain no useful content, so if you set this to "no" then no infections listed in your "Silent Viruses" setting will be quarantined, even if you have chosen to quarantine infections in general. The default is currently set to "yes" so the behavior is the same as it was in previous versions.

This can also be the filename of a ruleset.

Typically this setting is changed to “no”.

`Quarantine Whole Message = no`

Do you want to quarantine the original *entire* message as well as just the infected attachments?

This can also be the filename of a ruleset, so you can switch this feature on and off for specific users or domains.

Typically this setting does not need to be changed.

`Quarantine Whole Messages As Queue Files = no`

When you quarantine an entire message, do you want to store it as raw mail queue files (so you can easily send them onto users) or as human-readable files (header in one file, body in another file)?

Typically this setting does not need to be changed.

`Language Strings = %report-dir%/languages.conf`

Set where to find all the language dependent strings used so they can be translated into your local language.

This may also be the filename of a ruleset so you can produce different languages for different messages.

Typically this setting does not need to be changed.

```
Deleted Bad Content Message Report =
                      %report-dir%/deleted.content.message.txt
Deleted Bad Filename Message Report =
                      %report-dir%/deleted.filename.message.txt
Deleted Virus Message Report =
                      %report-dir%/deleted.virus.message.txt
```

These should be set to the location of the message text sent to users when one of their attachments or a virus has been deleted from a message.

These can also be the filenames of Rulesets.

Typically these settings do not need to be changed.

```
Stored Bad Content Message Report =
                      %report-dir%/stored.content.message.txt
Stored Bad Filename Message Report =
                      %report-dir%/stored.filename.message.txt
Stored Virus Message Report =
                      %report-dir%/stored.virus.message.txt
```

These should be set to the location of the message text sent to users when one of their attachments has been deleted from a message and stored in the quarantine.

These can also be the filenames of Rulesets.

Typically these settings do not need to be changed.

`Disinfected Report = %report-dir%/disinfected.report.txt`

This should be set to the location of the message text sent to users explaining the attached disinfected documents.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Inline HTML Signature = %report-dir%/inline.sig.html`

`Inline Text Signature = %report-dir%/inline.sig.txt`

These should be set to the location of the HTML and text versions of the signature files that will be added to the end of all clean messages, if Sign Clean Messages is set to yes.

These can also be the filenames of Rulesets.

Typically this setting does not need to be changed.

`Inline HTML Warning = %report-dir%/inline.warning.html`

`Inline Text Warning = %report-dir%/inline.warning.txt`

These should be set to the location of the HTML and text warnings that will be inserted at the top of messages that have had viruses removed from them.

These can also be the filenames of Rulesets.

Typically these settings do not need to be changed

`Sender Content Report = %report-dir%/sender.content.report.txt`

`Sender Error Report = %report-dir%/sender.error.report.txt`

`Sender Bad Filename Report =`
`%report-dir%/sender.filename.report.txt`

`Sender Virus Report = %report-dir%/sender.virus.report.txt`

These should be set to the location of the messages that are delivered to the sender, when they sent an email containing an error, banned content, a banned filename or a virus infection.

These can also be the filenames of Rulesets.

Typically these settings do not need to be changed

`Hide Incoming Work Dir = yes`

Hide the directory path from all virus scanner reports sent to users. The extra directory paths give away information about your setup, and tend to confuse users.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed

`Include Scanner Name In Reports = yes`

Include the name of the virus scanner in each of the scanner reports. This also includes the translation of "MailScanner" in each of the report lines resulting from one of MailScanner's own checks such as filename, filetype or dangerous HTML content. To change the name "MailScanner", look in `reports/<your_language>/languages.conf`. Very useful if you use several virus scanners, but might not be desirable if you don't want to let your customers know which scanners you use.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed

Changes to Message Headers

`Mail Header = X-%org-name%-MailScanner:`

Add this extra header to all mail as it is processed.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed

> The value for this setting MUST include the colon ":" at the end and should have NO white space between the **`X-`** and the "**`:`**" at the end of the line.

`Spam Header = X-%org-name%-MailScanner-SpamCheck:`

Add this extra header to all messages found to be spam.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed

`Spam Score Header = X-%org-name%-MailScanner-SpamScore:`

Add this extra header if "Spam Score" = yes. The header will contain one character for every point of the SpamAssassin score or an integer, depending on your spam score settings.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed

`Information Header = X-%org-name%-MailScanner-Information:`

Add this extra header to all mail as it is processed. The content is set by "Information Header Value" and is intended for you to be able to insert a help URL for your users. If you don't want an information header at all, just comment out this setting or set it to be blank.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed

`Add Envelope from Header = yes`

Do you want to add the Envelope-From: header? This is very useful for tracking where spam came from as it contains the envelope sender address.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Add Envelope To Header = no`

Do you want to add the Envelope-To: header? This can be useful for tracking spam destinations, but should be used with care due to possible privacy concerns with the use of Bcc: headers by users.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

Envelope From Header = X-MailScanner-From:

This is the name of the Envelope From header controlled by the option above.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

Spam Score Character = s

The sets the character to use in the "Spam Score Header". Do not use the following characters:

x	Since users will think a score of 3 "xxx" is porn,
#	Since it will cause confusion with comments in procmail as well as MailScanner itself,
*	Since it will cause confusion with pattern matches in procmail,
.	Since it will cause confusion with pattern matches in procmail,
?	Since it will cause the users to think something went wrong.

Do use "s", is nice and safe and stands for "spam".

Typically this setting does not need to be changed.

SpamScore Number Instead Of Stars = no

If this option is set to yes, you will get a spam-score header showing only the value of the spam score, instead of the row of characters representing the score.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

Minimum Stars If On Spam List = 0

This sets the minimum number of "Spam Score Characters" which will appear if a message triggered the **Spam List =** setting (see below) but received a very low SpamAssassin score. This means that people who only filter on the Spam Stars will still be able to catch messages which receive a very low SpamAssassin score. Set this value to 0 to disable it.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

Infected Header Value = Found to be infected

Clean Header Value = Found to be clean

Disinfected Header Value = Disinfected

These values set the "Mail Header" to these values for clean, infected and disinfected messages.

These can also be the filenames of a Rulesets.

Typically these settings do not need to be changed.

Information Header Value = Please contact the ISP for more information

This sets the "Information Header" to this value.

These can also be the filenames of a Rulesets.

Typically this setting is customized for each site.

`Detailed Spam Report = yes`

Do you want the full spam report, or just a simple "spam / not spam" report?

This setting should be changed to reflect your site’s preferences.

`Always Include Scores In SpamAssassin Report = yes`

Do you want to include the numerical scores in the detailed SpamAssassin report, or just list the names of the scores?

Typically this setting does not need to be changed.

`Multiple Headers = append`

This setting determines what happens when there are multiple MailScanner headers from multiple MailScanner servers in one message.

Available setting values include:

▪ append	Append the new data to the existing header
▪ add	Add a new header
▪ replace	the old data with the new data

Typically this setting does not need to be changed.

`Hostname = the %org-name% MailScanner`

This sets the name of this host, or a name like "the MailScanner" if you want to hide the real hostname. It is used in the Help Desk note contained in the virus warnings sent to users.

This can also be the filename of a Ruleset.

Typically this setting should be changed to identify your site, for example:

```
Hostname = the %org-name% MailScanner at <site_name>
```

`Sign Clean Messages = no`

If this is "no", then (as far as possible) messages which have already been processed by another MailScanner server will not have the clean signature added to the message. This prevents messages getting many copies of the signature as they flow through your site.

This can also be the filename of a ruleset.

This setting should be changed to reflect your site’s preferences.

`Sign Messages Already Processed = no`

Add the "Inline HTML Signature" or "Inline Text Signature" to the end of uninfected messages?

This can also be the filename of a Ruleset.

Often this setting is changed to publicize the use of MailScanner at your site.

Mark Infected Messages = yes

Add the "Inline HTML Warning" or "Inline Text Warning" to the top of messages that have had attachments removed from them?

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

Mark Unscanned Messages = yes

When a message is to not be virus-scanned, which may happen depending upon the setting of "Virus Scanning =, especially if it is a ruleset, do you want to add the header a advising the users to get their email virus-scanned by their scanning system. This can be useful for advertising your MailScanning service and encouraging users to sign up for virus scanning.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed for ISP and ASP sites but other sites might consider changing this value to no.

Unscanned Header Value = Not scanned: please contact your Internet E-Mail Service Provider for details

This is the text used by the Mark Unscanned Messages option listed above.

This can also be the filename of a Ruleset.

Typically this setting is customized for each site if Mark Unscanned Messages is set to yes.

Deliver Cleaned Messages = yes

Do you want to deliver messages once they have been cleaned of any viruses? By making this a ruleset, you can re-create the "Deliver From Local" facility of previous versions of MailScanner.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

Notifications back to the senders of blocked messages

Notify Senders = yes

Do you want to notify the people who sent you messages containing viruses or badly-named filenames?

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

Notify Senders of Viruses = no

If **Notify Senders** (above) is set to yes, do you want to notify people who sent you messages containing viruses? The default value has been changed to "no" since most viruses now fake the sender addresses and therefore should be on the "Silent Viruses" list.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Notify Senders Of Blocked Filenames Or File types = yes`

If **`Notify Senders`** (above) is set to yes, do you want to notify people who sent you messages containing attachments that are blocked due to their filename or file contents?

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Notify Senders Of Other Blocked Content = yes`

If **`Notify Senders`** (above) is set to yes, do you want to notify people who sent you messages containing other blocked content, such as partial messages or messages with external bodies?

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Never Notify Senders of Precedence = list bulk`

If you supply a space-separated list of message "precedence" settings, then senders of those messages will not be warned about anything you rejected. This is particularly suitable for mailing lists, so that any MailScanner responses do not get sent to the entire list.

Typically this setting does not need to be changed.

Changes to the Subject: line

`Scanned Modify Subject = no`

When the message has been scanned but no other subject line changes have happened, do you want modify the subject line?

Available setting values include:

▪ no	Do not modify the subject line
▪ start	Add text to the start of the subject line
▪ end	Add text to the end of the subject line

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Scanned Subject Text = {Scanned}`

This is the text to add to the start/end of the subject line if the **`Scanned Modify Subject`** option (above) is set to yes.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Virus Modify Subject = yes`

If the message contained a virus, do you want to modify the subject line? This makes filtering in Outlook very easy.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Virus Subject Text = {Virus?}`

This is the text to add to the start/end of the subject line if the **`Virus Modify Subject`** option (above) is set to yes.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Filename Modify Subject = yes`

If an attachment triggered a filename check, but there was nothing else wrong with the message, do you want to modify the subject line? This makes filtering in Outlook very easy.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Filename Subject Text = {Filename?}`

This is the text added to the start of the subject if the **`Filename Modify Subject`** (above) option is set to yes. You might want to change this so your users can see at a glance whether it just was just the filename that MailScanner rejected.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Content Modify Subject = yes`

If an attachment triggered a content check, but there was nothing else wrong with the message, do you want to modify the subject line? This makes filtering in Outlook very easy.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Content Subject Text = {Blocked Content}`

This is the text to add to the start of the subject if the **`Content Modify Subject`** option (above) is set to yes. You might want to change this so your users can see at a glance whether it just was just the content that MailScanner rejected.

Typically this setting does not need to be changed.

`Spam Modify Subject = yes`

If the message is spam, do you want to modify the subject line? This makes filtering in Outlook very easy.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Spam Subject Text = {Spam?}`

This is the text to add to the start of the subject if the **`Spam Modify Subject`** option is set to yes. The exact string "_SCORE_" will be replaced by the numeric SpamAssassin score.

You might consider setting this value to **`{Spam _SCORE_}`** if you want to expose spam scores to your users.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`High Scoring Spam Modify Subject = yes`

This is just like the **`Spam Modify Subject`** option above, except that it applies when the score from SpamAssassin is higher than the High SpamAssassin Score value (see below). The exact string "_SCORE_" will be replaced by the numeric SpamAssassin score.

You might consider setting this value to **`{High Scoring Spam}`** or **`{ High Scoring Spam _SCORE_}`** if you deliver all spam to your users and want them to be able to spam and high scoring spam with different filters.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

Changes to the Message Body

`Warning Is Attachment = yes`

When a virus or attachment is replaced by a plain-text warning, should the warning be in an attachment? If "no" then it will be placed in-line.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Attachment Warning Filename = %org-name%-Attachment-Warning.txt`

When a virus or attachment is replaced by a plain-text warning, and that warning is an attachment, this is the filename of the warning text.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Attachment Encoding Charset = us-ascii`

What character set do you want to use for the attachment If your site is located outside of the US or England, you will probably want "ISO-8859-1" instead.

This can also be the filename of a Ruleset.

If your site is located outside of the US, you might need to change this setting.

Mail Archiving and Monitoring

`Archive Mail = <blank>`

This setting is used to control which messages are archived. It may be set to a space-separated list of any combination of:

- Email addresses to which mail should be forwarded
- Directory names where you want mail to be stored
- Names of local users (they must already exist!) to which mail will be appended in "mbox" format suitable for most Unix mail systems.

If you implement archiving, you should be aware of the legal implications. In many jurisdictions it may be an illegal interception of a private, privileged message unless an appropriate authority has requested you to intercept the messages.

If you set this value to a Ruleset, you can control exactly whose mail is archived or forwarded.

Typically this setting does not need to be changed.

Notices to System Administrators

`Send Notices = yes`

Notify the local system administrators ("Notices To") when any infections are found.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Notices Include Full Headers = yes`

Include the full headers of each message in the notices sent to the local system administrators?

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Notices Include Full Headers = no`

Include the full headers of each message in the notices sent to the local system administrators?

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Hide Incoming Work Dir in Notices = no`

Hide the directory path from all the system administrator notices? The extra directory paths give away information about your setup, and tend to just confuse users but are still useful for local system administrators.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Notice Signature =`
`\nMailScanner\nEmail Virus Scanner\nwww.mailscanner.info`

The signature to add to the bottom of notices to system administrators. To insert a line-break in the signature, use the sequence "\n".

This can also be the filename of a Ruleset.

Often this setting is customized for a site.

`Notices From = MailScanner`

The visible part of the email address used in the "From:" line of the notices. The <user@domain> part of the email address is set to the "Local Postmaster" setting.

This can also be the filename of a Ruleset.

Often this setting is customized for a site, i.e. MailScanner at <site_name>

`Notices to = postmaster`

Where to send MailScanner notices to system administrators.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Local Postmaster = postmaster`

Address of the local Postmaster, which is used as the "From" address in virus warnings sent to users.

This can also be the filename of a Ruleset.

Often this setting is customized for a site, i.e. helpdesk@sitename.com

Spam Detection and Virus Scanner Definitions

`Spam List Definitions = %etc-dir%/spam.lists.conf`

This is the name of the file that translates the names of the Spam List values (below) to the real DNS names of the spam blacklists. The

`%etc-dir%/spam.lists.conf` file is used only by MailScanner and rarely needs to be modified. You need to modify this file only if you want to add additional RBL sites for use by the MailScanner **`Spam List`** or **`Spam Domains`** settings that are not already listed in this file.

Typically this setting should not be changed.

`Virus Scanner Definitions = %etc-dir%/virus.scanners.conf`

This is the name of the file that translates the names of the virus scanners into the commands that have to be run to do the actual scanning.

Typically this setting should not be changed.

Spam Detection and Spam Lists (DNS blocklists)

`Spam Checks = yes`

Do you want to check messages to see if they are spam?

If you set this value to no then NO spam checks will be done at all. This includes both MailScanner's own checks and SpamAssassin. If you want to just disable MailScanner's "Spam List" feature then set `Spam List =` and `Spam Domains =` (an empty list) in the settings below.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Spam List = ORDB-RBL SBL+XBL # MAPS-RBL+ costs money (except .ac.uk)`

This is the list of spam blacklists (RBLs) which you are using in MailScanner

This value is typically customized for each site. See the `Spam List Definitions` file (see above) for more information about what values may be used here.

RBLs may be used in any combination of three methods:

1. Blocking at the MTA level: This is an MTA configuration level option. Messages blocked at the MTA level are not accepted for delivery. Blocking at this level reduces the load on you system but you assume the risk of rejecting some amount of real email.

2. MailScanner RBL checking: MailScanner checks to see if the sender or a relay of the message is listed in `Spam List =` or `Spam Domains =.` If found, the message is marked as spam. If the message is found in multiple RBL lists, the `Spam Lists To Reach High Score =` setting is used to determine if the message should be treated as High Scoring Spam.

3. SpamAssassin scoring: SpamAssassin by default checks various RBL and adds to the spam score each time sender or relay of the message is found in an RBL.

This can also be the filename of a Ruleset.

Often this setting is changed to be <blank> to enable RBL checking only by SpamAssassin.

`Spam Domain List = <blank>`

This is the list of spam domain blacklists (such as the "rfc-ignorant" domains) which are used by MailScanner.

See the "Spam List Definitions file (see above) for more information about what values may be used here.

Often this setting left <blank> to enable RBL checking only by SpamAssassin.

`Spam Lists To Reach High Score = 3`

If a message appears in at least this number of Spam Lists and/or Spam Domain Lists (as defined above), then the message will be treated as High Scoring Spam and the High Scoring Spam Actions will happen. If you use RBL checking in MailScanner you probably want to set this to 2. Setting this value to 5 is high enough that it will never happen unless you a large number of Spam Lists.

See the `Spam List Definitions file` (see above) for more information about what values may be used here.

This can also be the filename of a Ruleset.

If `Spam List =` and `Spam Domains =` are set to `<blank>` this setting is not used.

`Spam List Timeout = 10`

If an individual "Spam List" or "Spam Domain List" check takes longer than this value (in seconds), the check is abandoned and the timeout noted in the log.

Permissible values = integer

Typically this setting does not need to be changed.

`Max Spam List Timeouts = 7`

The maximum number of timeouts caused by any individual Spam List or Spam Domain List before it is marked as "unavailable". Once so marked, the marked list will be ignored until the next automatic re-start (see Restart Every above for the longest time it will wait).

Permissible values = integer

Typically this setting does not need to be changed.

`Spam List Timeouts History = 10`

The total number of Spam List attempts during which "Max Spam List Timeouts" will cause the spam list to be marked as "unavailable". See `Max Spam List Timeouts` above for more information.

The default values of 5 and 10 mean that 5 timeouts in any sequence of 10 attempts will cause the list to be marked as "unavailable" until the next periodic restart. Also see `Restart Every` =.

Permissible values = integer

Typically this setting does not need to be changed.

`Is Definitely Not Spam = %rules-dir%/spam.whitelist.rules`

Sets the location of the Spam Whitelist Ruleset. Anything in this rulese whose value is "yes" will never be marked as spam.

This is always the filename of a ruleset.

Typically this setting does not need to be changed.

`Is Definitely Spam = no`

If this value points to a Ruleset, that Ruleset will be used to determine which sites are blacklisted. See Appendix C, Practical Ruleset Examples for instructions on changing the value to a Ruleset.

This can also be the filename of a Ruleset.

This value is typically customized for each site. A Ruleset similar to `spam.whitelist.rules` is created in %rules-dir%. Sites listed in this file will be treated as defined by the Definite Spam Is High Scoring setting (see below)

`Definite Spam Is High Scoring = no`

Setting this to yes results in spam found in the blacklist being treated as High Scoring Spam in the **`High Spam Actions`** (see below). Setting the value to no means that it will be treated as "normal" spam.

This can also be the filename of a Ruleset.

Many Sites typically set this value to yes.

`Ignore Spam Whitelist If Recipients Exceed = 20`

Spammers have learned that they can get their message through by sending a message to many recipients, one of which chooses to whitelist everything coming to them, including the spammer. If a message arrives with more than this number of recipients, ignore the "Is Definitely Not Spam" whitelist.

Typically this setting does not need to be changed.

SpamAssassin

`Use SpamAssassin = no`

Do you want to find spam using the "SpamAssassin" package?

This can also be the filename of a Ruleset.

Typically this setting is changed to yes to enable SpamAssassin

Hint: point this value to a ruleset to turn off SpamAssassin checking for specific users or domains. See the configuration variable **`Spam Checks =`** to turn off all MailScanner spam checks.

`Max SpamAssassin Size = 30000`

SpamAssassin is not very fast when scanning huge messages, so messages bigger than this value will be truncated to this length for SpamAssassin testing. The original message will not be affected by this. This value is a good compromise as very few spam messages are bigger than this (it takes too long to send out large spam messages).

Permissible values = integers

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`Required SpamAssassin Score = 6`

This replaces the SpamAssassin configuration value 'required_hits'. If a message achieves a SpamAssassin score higher than this value, it is spam. Also see the High **`SpamAssassin Score`** configuration option (below).

Permissible values = nn.nn where n = integer

This can also be the filename of a Ruleset.

Typically this setting is changed at every site. The “right” value depends on the number and types (DCC, Pyzor, Razor, RBLs, etc.) of spam checking methods implemented at your site. Some general rules for setting this value:

5	Mildly Aggressive, some false positives will result
6	Normal Setting
7+	Or Above. Few false positives but a lot more spam

`High SpamAssassin Score = 10`

If a message achieves a SpamAssassin score higher than this value, it’s fate will be determined using **`High Scoring Spam Actions`** configuration option (below).

Permissible values = nn.nn where n = integer

This can also be the filename of a Ruleset.

Typically this setting is changed at every site. Again the “right” value depends on the number and types (DCC, Pyzor, Razor, RBLs, etc.) of spam checking methods implemented at your site. Some general rules for setting this value:

8	Aggressive, but end users will see less spam
10	Normal Setting
12+	More spam will be seen by the end users

`SpamAssassin Auto Whitelist = no`

Setting this option to "yes" will enable the automatic SpamAssassin white listing functions. Since some spammer have been able to abuse this function (and poison Bayes databases as a result). This value should be left set to no.

This can also be the filename of a Ruleset.

Typically this setting does not need to be changed.

`SpamAssassin Prefs File = %etc-dir%/spam.assassin.prefs.conf`

This value sets the location of the SpamAssassin user_prefs file. See Chapter 6 for configuring SpamAssassin preferences.

Typically this setting does not need to be changed.

`SpamAssassin Timeout = 40`

If SpamAssassin takes longer than this value, in seconds, the SpamAssassin check is abandoned and the timeout is noted in the logs.

Permissible values = integer

Typically this setting does not need to be changed.

“SpamAssassin timeout” messages in your logs indicates a problem and correct spam detection is probably not occurring. See Chapter 7 Tips, Tuning and Troubleshooting to diagnose this problem.

`Max SpamAssassin Timeouts = 20`

If consecutive SpamAssassin time outs exceed this value, then SpamAssassin will be marked as "unavailable" (SpamAssassin spam detection will stop, but mail will flow) until the next MailScanner re-start.

Permissible values = integer

Typically this setting does not need to be changed.

`SpamAssassin Timeouts History = 30`

The total number of SpamAssassin attempts during which "Max SpamAssassin Timeouts" will cause SpamAssassin to be marked as "unavailable". See **`Max SpamAssassin Timeouts`** above for more information. The default values of 10 and 20 mean that 10 timeouts in any sequence of 20 attempts will trigger the behavior described above, until the next periodic restart. Also see **`Restart Every =.`**

Permissible values = integer

Typically this setting does not need to be changed.

`Check SpamAssassin If On Spam List = yes`

If the message sender is on any of the Spam Lists, do you still want to do the SpamAssassin checks? Setting this to "no" will reduce the load on your server, but will stop SpamAssassin from scoring messages if the message triggers the MailScanner RBL checks.

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Always Include SpamAssassin Report = no`

Do you want to always include the Spam Report in the SpamCheck header, even if the message wasn't spam?

This can also be the filename of a ruleset.

Many sites change this setting is changed to “yes” to allow spam score checking of messages that should have been marked as spam.

`Spam Score = yes`

Do you want to include the "Spam Score" header? This shows 1 character (**`Spam Score Character`**) for every point of the SpamAssassin score. This allows users to filter their mail using whatever SpamAssassin threshold they want. For example, they just look for "sssss" for every message whose score is > 5.

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Rebuild Bayes Every = 0`

If you are using the Bayesian statistics engine on a busy server, you may well need to force a Bayesian database rebuild and expiry at regular intervals. The value 0 disables this function, the value of 86400 rebuilds the Bayes database once a day..

This setting is often changed. See Chapter 6, SpamAssassin Configuration.

`Wait During Bayes Rebuild = no`

The Bayesian database rebuilds and expiry may take a 2 or 3 minutes to complete. During this time you can either wait, or simply disable SpamAssassin checks until it has completed.

Setting this value to yes will disable MailScanner processing during the rebuild

Typically this setting does not need to be changed.

What to do with spam

`Spam Actions = deliver`

This is a list of actions to take when a message is spam. It can be white space separated list of any combination of the following:

▪ deliver	deliver the message as normal
▪ delete	delete the message
▪ store	store the message in the quarantine
▪ bounce	send a rejection message back to the sender (not recommended)
▪ forward user@domain.com	forward a copy of the message to user@domain.com
▪ striphtml	convert all in-line HTML content to plain text. You need to specify "deliver" after striphtml for the message to reach the intended recipient.
▪ attachment	Convert the original message into an attachment of the message. This means the user has to take an extra step to open the spam, and stops "web bugs" very effectively. You need to specify "deliver" after “attachment” for the message to reach the intended recipient.
▪ notify	Send the recipients a short notification that spam addressed to them was not delivered. They can then take action to request retrieval of the original message if they think it was not spam.

This can also be the filename of a ruleset

Most sites set this value to deliver or attachment deliver.

`High Scoring Spam Actions = deliver`

This is a list of actions to take when a message is high scoring spam. It can be white space separated list of any combination of the following:

▪ deliver	deliver the message as normal
▪ delete	delete the message
▪ store	store the message in the quarantine
▪ bounce	send a rejection message back to the sender (not recommended)
▪ forward user@domain.com	forward a copy of the message to user@domain.com
▪ striphtml	convert all in-line HTML content to plain text. You need to specify "deliver" after striphtml for the message to reach the intended recipient.
▪ attachment	Convert the original message into an attachment of the message. This means the user has to take an extra step to open the spam, and stops "web bugs" very effectively . You need to specify "deliver" after “attachment” for the message to reach the intended recipient.
▪ notify	Send the recipients a short notification that spam addressed to them was not delivered. They can then take action to request retrieval of the original message if they think it was not spam.

This can also be the filename of a ruleset

Many sites set this value to store or delete.

`Non Spam Actions = deliver`

This operates like the `Spam Actions` option above, except it applies messages that are not spam and would normally be delivered:

▪ deliver	deliver the message as normal
▪ delete	delete the message
▪ store	store the message in the quarantine
▪ forward user@domain.com	forward a copy of the message to user@domain.com
▪ striphtml	convert all in-line HTML content to plain text. You need to specify "deliver" after striphtml for the message to reach the intended recipient.

This can also be the filename of a ruleset

Typically this value is not changed.

`Sender Spam Report = %report-dir%/sender.spam.report.txt`

`Sender Spam List Report = %report-dir%/sender.spam.rbl.report.txt`

`Sender SpamAssassin Report = %report-dir%/sender.spam.sa.report.txt`

There are three Spam Reports:

▪ Sender Spam Report	Sent when a message triggers both a Spam List and SpamAssassin
▪ Sender Spam List Report	Sent when a message triggers a Spam List
▪ Sender SpamAssassin Report	Sent when a message triggers SpamAssassin.

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Inline Spam Warning = %report-dir%/inline.spam.warning.txt`

If you use the 'attachment' Spam Action or High Scoring Spam Action this value is the location of inline spam report that is inserted at the top of the message.

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Recipient Spam Report = %report-dir%/recipient.spam.report.txt`

If you use the 'notify' Spam Action or High Scoring Spam Action this value is the location of the notification message that is sent to the original recipients of the message.

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Enable Spam Bounce = %rules-dir%/bounce.rules`

You can use this ruleset to enable the "bounce" Spam Action. You must ONLY enable this for mail from sites with which you have agreed to bounce possible spam. Use it on low-scoring spam only (<10) and only to your regular customers for use in the rare case that a message is tagged as spam when it shouldn't have been.

Beware that many sites will automatically delete the bounce messages created by using this option unless you have agreed this with them in advance. If you enable this, be prepared to handle the irate responses from people to whom you are essentially sending more spam!

This can also be the filename of a ruleset.

Typically this setting should never be changed.

Logging

`Syslog Facility = mail`

This is the syslog "facility" name that MailScanner uses. If you don't know what a syslog facility name is, then either don't change this value or else please read "`man syslog.conf`". The default value of "mail" will cause the MailScanner logs to go into the same place as all your other mail logs.

Typically this setting does not need to be changed.

`Log Speed = no`

Do you want to log the processing speed for each section of the code for a batch? This can be very useful for diagnosing speed problems, particularly in spam checking.

Typically this setting does not need to be changed unless you are troubleshooting speed or delivery problems.

`Log Spam = no`

Do you want all spam to be logged? This can be useful if you want to gather spam statistics from your logs, but can increase the system load substantially if you receive a lot of spam.

Typically this setting does not need to be changed unless you are troubleshooting speed or delivery problems. Many sites change this value to yes to gather spam statistics from logs.

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Log Non Spam = no`

Do you want all non-spam to be logged? This can be useful if you want to gather spam statistics from your logs, but can increase the system load substantially.

Typically this setting does not need to be changed unless you are troubleshooting spam detection problems. Many sites change this value to yes to gather spam statistics from logs.

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Log Permitted Filenames = no`

Log all the filenames that are allowed by the Filename Rules, or just the filenames that are denied?

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Log Permitted Filetypes = no`

Log all the filenames that are allowed by the Filetype Rules, or just the filetypes that are denied?

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

Advanced SpamAssassin Settings

If you are using Postfix you may definitely need to use some of the settings below, since the home directory for the "postfix" user cannot be written to by the "postfix" user.

You may also need to use these settings if you have installed SpamAssassin somewhere other than the default location.

`SpamAssassin User State Dir = <blank>`

The per-user files (bayes, auto-whitelist, user_prefs) are looked for in this directory and in ~/.spamassassin/. Note the files are mutable. If this is unset (left blank) then no additional directories are searched.

If you are using Postfix, you probably want to set this value to:

SpamAssassin User State Dir = /var/spool/MailScanner/spamassassin

and then run the following commands:

```
 mkdir /var/spool/MailScanner/spamassassin
 chown postfix.postfix /var/spool/MailScanner/spamassassin
```

Typically this setting does not need to be changed.

`SpamAssassin Install Prefix = <blank>`

This setting is useful if SpamAssassin is installed in an unusual place, e.g. /opt/MailScanner. The install prefix is used to find some fallback directories if neither of the following two settings work. If this is set then it adds to the list of places that are searched; otherwise it has no effect.

Typically this setting does not need to be changed.

`SpamAssassin Site Rules Dir = /etc/mail/spamassassin`

The SpamAssassin site rules are searched for in this location.

Typically this setting does not need to be changed.

`SpamAssassin Local Rules Dir = <blank>`

The local SpamAssassin rules are searched for in this directory, and in prefix/share/spamassassin, /usr/local/share/spamassassin, /usr/share/spamassassin, and possibly other directories. If this is set it adds to the list of places that are searched; otherwise it has no effect.

Typically this setting does not need to be changed.

`SpamAssassin Default Rules Dir = <blank>`

The default SpamAssassin rules are searched for in this directory, and in prefix/share/spamassassin, /usr/local/share/spamassassin, /usr/share/spamassassin, and possibly other directories. If this is set it adds to the list of places that are searched; otherwise it has no effect.

Typically this setting does not need to be changed.

Advanced Settings

Please don't change anything below this unless you really know what you are doing, or unless MailScanner has complained about your "Minimum Code Status" setting (very unusual).

`Use Default Rules With Multiple Recipients = no`

When trying to work out the value of configuration parameters which are using a ruleset, this controls the behavior when a rule is checking the "To:" addresses.

If this option is set to yes, then the following happens when checking the ruleset:

- 1 recipient. Same behavior as normal.
- Several recipients, but all in the same domain (domain.com for example). The rules are checked for one that matches the string "*@domain.com".
- Several recipients, not all in the same domain. The rules are checked for one that matches the string "*@*".

If this option is set to "no", then some rules will use the result they get from the first matching rule for any of the recipients of a message, so the exact value cannot be predicted for messages with more than 1 recipient.

This value cannot be the filename of a ruleset.

Typically this setting does not need to be changed.

`Spam Score Number Format = %d`

When putting the value of the spam score of a message into the headers,how do you want to format it? If you don't know how to use sprintf() or printf() in C, please *do not modify* this value. A few examples for you:

```
%d     ==> 12
%5.2f  ==> 12.34
%05.1f ==> 012.3
```

This can also be the filename of a ruleset.

Typically this setting does not need to be changed.

`Debug = no`

Set Debug to "yes" to stop MailScanner from running as a daemon and just process one batch of messages and then exit.

Typically this setting does not need to be changed. It is used only for troubleshooting.

`Debug SpamAssassin = no`

Do you want to debug SpamAssassin from within MailScanner?

Typically this setting does not need to be changed. It is used only for troubleshooting.

`Run In Foreground = no`

Set Run In Foreground to "yes" if you want MailScanner to operate normally in foreground (and not as a background daemon). Use this if you are controlling the execution of MailScanner with a tool like DJB's 'supervise' (see http://cr.yp.to/daemontools.html).

Typically this setting does not need to be changed.

`#LDAP Server = localhost`

`#LDAP Base = o=fsl`

`#LDAP Site = default`

If you are using an LDAP server to read the configuration, these Configuration settings should be uncommented and the values changed to those appropriate for your site. The values provide the details required for the LDAP connection. The connection is anonymous.

Typically this setting does not need to be changed.

`Always Looked Up Last = no`

This option is intended for people who want to log more information about messages than what is written to syslog. It is intended to be used with a Custom Function which has the side-effect of logging information, perhaps to an SQL database, or any other processing you want to do after each message is processed. The default value of no disables this function. If you want to use Custom Functions, please read `MyExample.pm file` typically located in `/usr/lib/MailScanner/Mailscanner/CustomFunctions.`

Typically this setting does not need to be changed.

`Deliver In Background = yes`

When attempting delivery of outgoing messages, should we do it in the background or wait for it to complete? The danger of doing it in the background is that the machine load goes ever upwards while all the slow sendmail processes run to completion. However, running it in the foreground may cause the mail server to run too slowly.

Typically this setting does not need to be changed.

`Delivery Method = batch`

Attempt immediate delivery of messages, or just place them in the outgoing queue for the MTA to deliver when it wants to?

Permissible settings for this value are:

- batch attempt delivery of messages, in batches of up to 20 at once
- queue just place them in the queue and let the MTA find them

This can also be the filename of a ruleset. For example, you could use a ruleset to insure that messages coming to you are immediately delivered, while messages going to any other site are just placed in the queue to handle the case the remote delivery is very slow.

Typically this setting should not be changed.

`Split Exim Spool = no`

Are you using Exim with split spool directories? If you don't understand this question, the answer is probably "no". Refer to the Exim documentation for more information about split spool directories.

Typically this setting does not need to be changed.

`Lockfile Dir = /tmp`

This describes where to put the virus scanning engine lock files. These lock files are used between MailScanner and the virus signature "autoupdate" scripts, to ensure that they aren't both working at the same time (which could cause MailScanner to let a virus through).

Typically this setting should not be changed.

`Custom Functions Dir =`

`/usr/lib/MailScanner/MailScanner/CustomFunctions`

This describes where to put the code for your "Custom Functions". No code in this directory should be over-written by the installation or upgrade process. All files starting with "." or ending with ".rpmnew" will be ignored, all other files will be compiled and may be used with Custom Functions.

Typically this setting should not be changed.

`#Lock Type = flock`

How to lock spool files. Don't set (uncomment) this value unless you *know* you need to. For sendmail, it defaults to "flock". For Exim, it defaults to "posix". No other type is implemented.

Typically this setting should not be changed.

`Minimum Code Status = supported`

The value sets minimum acceptable code stability status. If we come across code that's not at least as stable as this, MailScanner complains and may become unstable. This is currently only used to check that you don't end up using untested virus scanner support code without realizing it.

Permissible settings for this value are:

▪ none	there may not even be any code.
▪ unsupported	code may be completely untested, a contributed dirty hack, anything, really.
▪ alpha	code is pretty well untested. Don't assume it will work.
▪ beta	code is tested a bit. It should work.
▪ supported	code should be reliable.

Don't even *think* about setting this to anything other than "beta" or "supported" on a system that receives real mail until you have tested it yourself and are happy that it is all working as you expect it to. Don't set it to anything other than "supported" on a system that could ever receive important mail.

Please READ and UNDERSTAND the above text BEFORE changing this value.

Typically this setting should never be changed.

Chapter 4

SpamAssassin Configuration

This chapter is intended to explain SpamAssassin configuration as it applies to running SpamAssassin from within MailScanner. How SpamAssassin scans messages and how rules are applied is a bit different than running SpamAssassin via procmail or from Milter.

If configured to use SpamAssassin, MailScanner calls SpamAssassin once for each batch of messages not once for each message. A quick look at the MailScanner Process Flow diagram in Chapter 1 shows that a MailScanner child process picks up a batch of messages from the incoming mail queue and first runs its own RBL checks on the messages in the batch. If MailScanner is configured to use SpamAssassin, it then calls SpamAssassin, by directly calling the SpamAssassin Perl modules, not the executable spamassassin or spamd, and runs the SpamAssassin rules against this batch of messages.

> Do not call the SpamAssassin modules using procmail and spamd. It is not necessary and will simply pass the messages thru SpamAssassin twice!

Each time MailScanner starts or reloads, it reads its configuration files and SpamAssassin's configuration files. These setting are retained and used to process messages until the next time MailScanner reload or restarts.

SpamAssassin configuration settings may be located in several places (See SpamAssassin Settings, Chapter 3) but the default and most common files used are:

- `/etc/MailScanner/spam.assassin.prefs.conf`
- `/etc/mail/spamassassin/*`
- `/home/root/.spamassassin/*`

SpamAssassin's own standard rules are typically located in /usr/share/spamassassin.

> Modify files in /usr/share/spamassassin at your own risk. They will be overwritten when you upgrade SpamAssassin

All of your custom SpamAssassin configuration settings should be stored in the spam.assassin.prefs.conf file. All of your custom rule sets should be stored in files which end in .cf in the `/etc/mail/spamassassin` directory

Following these simple rules for locating and storing your custom SpamAssassin configurations and rules will preserve them when SpamAssassin is upgraded

spam.assassin.prefs.conf

This file may contain SpamAssassin configuration setting and custom rule sets. For a full description of all the possible SpamAssassin configuration setting please see:

http://www.spamassassin.org/doc/Mail_SpamAssassin_Conf.html

Here we will explain only a most commonly used subset of all the possible configuration settings.

SpamAssassin uses UNIX style configuration files:

```
# this character starts a comment, which continues until
# end of line.
# blank lines are allowed
```

The most configuration settings are described by

```
<Configutation_setting_string>   <value>
```

The left hand side may be separated for the right hand side by any “white space” character.

The format for SpamAssassin scoring rule sets is discussed below.

SpamAssassin settings are most easily configured by adding configuration settings to your `spam.assassin.prefs.conf` file. All of the configuration settings described in this Section should be added to or changed in your spam.assassin.prefs.conf file.

SpamAssassin and DNS

SpamAssassin uses DNS lookups extensively. If you are on a slow network or DNS lookups are failing, you can expect to have problems. Typically these problems manifest themselves as “spamassassin timeouts” in your log files.

You can speed up SpamAssassin’s DNS lookup by simply over-riding its default behavior by adding a line to the `spam.assassin.prefs.conf` file. Each time MailScanner calls the SpamAssassin modules, SpamAssassin check to see if DNS services are available. It randomly checks one of 13 MX records. This is unnecessary (If DNS is not working on a mail gateway you have bigger problems than SpamAssassin failing) and can be turned off by adding:

```
dns_available yes
```

For additional tips on speeding up DNS please see chapter 7, Tuning and Performance.

White and Black Listing

While white and black listing of users and domains can be accomplished by adding entries in the spam.assassin.prefs.conf file, this is better done by adding these entries in your spam.whitelist.rules and spam.blacklist.rules in the MailScanner/rules directory.

Bayesian Filtering

By default, SpamAssassin uses the Bayesian engine to help identify spam. This is very CPU intensive and on a small overloaded system, you might need to disable it. The settings to use are:

`use_bayes  (0|1)`

This convention of using (0|1) to display the possible values for the configuration setting will be used throughout this Chapter. The values shown mean:

```
use_bayes  0 (turns Bayesian Filtering off)
use_bayes  1 (turns Bayesian Filtering on - default)
```

By default the Bayes database is created in the home directory of the user running SpamAssassin. When SpamAssassin is called from MailScanner, the user ID used by SpamAssassin is the user ID of the MailScanner process, most typically root or postfix. This can create problems. If the MailScanner user is root, the Bayes tokens are stored in /root/.spamassassin. If /root is located on a small partition, the Bayes database will quickly fill up the partition. The location of the Bayes is easily changed. To move the Bayes Database:

Create a new directory to store the Bayes database tokens. A typical location is `/var/local/spamassassin` (assuming you have adequate space in /var)

1. Stop MailScanner
2. Move all of the Bayes files to the new directory. These will be the files starting with “bayes_”.
3. Change the ownership and read-write permissions of the new directory and moved files to match the user of the MailScanner proceses.
4. Add the following lines to your spam.assassin.prefs.conf file:

   ```
   bayes_path /var/local/spamassassin/bayes
   bayes_file_mode 0600
   ```

This example assumes you have created the directory `/var/local/spamassassin` to store your database. Note that the configuration value must end with “`/bayes`” appended to the actual directory name. This allows SpamAssassin to identify the actual Bayes files (they all start with “bayes”) in the directory.

If you have a new installation of SpamAssassin, bayes filtering will not be used until the Bayesian database accumulates at least 200 spam and 200 ham (ham is the opposite of spam) messages. The Bayesian database, by default is set to “auto learn” from messages that pass through SpamAssassin. The SpamAssassin configuration settings that control this behavior are:

`bayes_auto_learn ( 0 | 1 ) (default: 1)`

The score threshold below which a mail has to score, to be fed into SpamAssassin's learning systems automatically as a non-spam message is set by:

`bayes_auto_learn_threshold_nonspam n.nn (default: 0.1)`

Where n = an integer

The score threshold below which a mail has to score, to be fed into SpamAssassin's learning systems automatically as a spam message.

`bayes_auto_learn_threshold_spam n.nn (default: 12.0.)`

SpamAssassin requires at least 3 points from the header, and 3 points from the body to auto-learn as spam. Therefore, the minimum working value for this option is 6.

One additional setting which is needed for SpamAssassin to interpret the information in the header of the message is to set the **`bayes_ignore_header`** to match the value you supplied in `MailScanner.conf`. For example, if you set:

```
      %org-name% = example-com
```

In your `MailScanner.conf` file, you should add the following lines to your SpamAssassin configuration file:

```
bayes_ignore_header   X-example.com-MailScanner
bayes_ignore_header   X-example.com-MailScanner-SpamCheck
bayes_ignore_header   X-example.com-MailScanner-SpamScore
bayes_ignore_header   X-example.com-MailScanner-Information
```

Another Bayes setting controls how the Bayes database expires old tokens from the database. By default the Bayes system will try to automatically expire old tokens from the database. This auto-expiry occurs when the number of tokens in the database surpasses the **`bayes_expiry_max_db_size`** value (default 150000). This occurs randomly and will cause Bayes locking problems if not controlled. While the MailScanner.conf setting:

```
      Rebuild Bayes Every = <value>
```

Will control this behavior, a safer way to accomplish the auto expiry is to set:

```
Rebuild Bayes Every = 0
```

in MailScanner.conf and then run a daily cron job at a quiet time on you system. The cron job to run is:

```
#! /bin/bash
# re-build the Bayes database daily
sa-learn -p /etc/MailScanner/spam.assassin.prefs.conf \
--rebuild --force-expire
```

While auto learning spam and ham will usually produce reasonable spam detection results, manually feeding missed spam and ham to the database will result in better Bayesian filtering.

Network Checks

By default, SpamAssassin will run network (Real-time Black Hole - RBL) checks. If you have a slow or unreliable Internet connection, you may need to turn off this feature. Network checks are enabled or disable in SpamAssassin:

```
skip_rbl_checks      (0|1) ( 1 disables the RBL checking)
```

Using additional applications such as DCC, Pyzor and Razor (see Chapter 6, Related Applications) can substantially enhance accurate spam detection. In most cases SpamAssassin will automatically find Pyzor and DCC if they are installed in the default locations or are in the directories included in the PATH variable of the effective UID of the process running MailScanner. If this fails, the following settings will enable SpamAssassin to find the applications:

```
pyzor_path /usr/bin/pyzor
dcc_path /usr/local/bin/dccproc
```

There is no corresponding SpamAssassin setting for Razor so the razor executable must be located in MailScanner's PATH variable.

While the correct DCC servers automatically selected by the application, the list of Pyzor and Razor servers periodically changes and should be updated daily. The following daily cron jobs will update these server lists:

pyzor-discover:

```
#! /bin/bash
# get a list of the Pyzor servers
/usr/bin/pyzor discover
```

razor-discover:

```
#! /bin/bash
# refresh /root/.razor/
razor-admin -discover
```

While these services are usually reliable, there have been service interruptions. If you experience such an interruption, these services may be disabled with SpamAssassin by using the following settings:

```
use_razor2      (0|1) ( 0 disables the service )
use_pyzor       (0|1) ( 0 disables the service )
use_dcc         (0|1) ( 0 disables the service )
```

The default SpamAssassin timeouts for blacklists and Razor are rather generous. Reducing these defaults on busy or heavily loaded systems will stops timeouts from removing SpamAssassin scores.

```
rbl_timeout 20
razor_timeout 10
pyzor_timeout 10
```

Adding SpamAssassin Rules

The default location for SpamAssassin's basic rule sets is /usr/share/spamassassin. If you want to augment these rules by writing you own or downloading contributed extra rule sets you should start with the excellent instructions which can be found at the SpamAssassin Custom Rules Emporium:

http://www.merchantsoverseas.com/wwwroot/gorilla/sa_rules.htm

Another excellent source for obtaining and automatically updating several of the most popular rule sets is the `rules_du_jour` script written by Chris Thielen. The current version of this script supports the following (very popular) rule sets:

- BIGEVIL
- TRIPWIRE
- CHICKENPOX
- WEEDS1
- BACKHAIR
- ANTIDRUG
- EVILNUMBERS

To obtain and install `rules_du_jour`:

```
wget http://www.fsl.com/support/Rules_Du_Jour.tar.gz
```

Please read and follow the instructions in the INSTALL. The rules_du_jour and RulesDuJour files MUST be edited to suit your sites configuration

This script is run as a daily cron job.

Changing SpamAssassin Rule Scores

SpamAssassin rule set scores go through extensive testing before release and you should seldom need to modify the basic rule set scores and you should have a very good understanding of how SpamAssassin score are used before implementing such changes. If you do need to change a rule set score you should add a line to the spam.assassin.prefs.conf similar to:

```
<Name_of_SA_Rule>     nn.nn
```

Where n = integer, for example, to change the score of the HABEAS_SWE rule from the default of -8 to -2 set add:

```
score HABEAS_SWE      -2.0
```

To disable a ruleset, set the score of the rule to 0.0

For a complete listing of standard SpamAssassin tests and scores, please visit

http://www.spamassassin.org/tests.html

SpamAssassin SURBL rules

SURBLs differ from most other RBLs in that they're used to block messages based on the domain names in message body URIs (usually web sites), for example those which have been previously reported to SpamCop as "Spamvertised sites". So SURBLs are not used to block spam senders like most other RBLs; instead they allow you to block messages based on spam domains that occur in their message bodies. Please visit:

www.surbl.org

For more information. A Kit for implementing SURBL checks may be downloaded using:

```
wget http://www.fsl.com/support/SURBL.tar.gz
```

Chapter 5

Advanced Configuration via Rulesets

Rulesets provide a very powerful way to configure many options that usually are set to "yes" or "no". Through the use of rule sets, many configuration options may be selectively applied, based on matching criteria. For example, if there is one email address or domain name for which different options need to be applied, a Ruleset can easily be created to allow this, by using the email address or domain name as the matching criteria. Matching criteria is extremely flexible and may be applied in a variety of ways. Below is a description of the format of a rule, as well as several examples.

Rule sets should be placed in the directory set by the MailScanner configuration setting:

```
%rule-dir% =
```

Typically this is /etc/MailScanner/rules.
MailScanner Rulesets must end with the extension ".rules", for example:

```
use.spamassassin.rules
```

Ruleset Formats

Rulesets are made up of individual rule lines, and each rule line of the ruleset has three parts.

- The **LEFT** side of the rule describes the direction the message is moving
- The **MIDDLE** section describes the pattern to match
- The **RIGHT** side describes the result of matching the pattern

The following is an example of a typical line in a ruleset:

Direction **pattern** (regular expression) **result**
```
From:                    john.doe@domain.com          yes
```

The **LEFT, MIDDLE** and **RIGHT** parts may be separated by spaces or tabs.

Direction

The **direction** (left side) may be any one of the following:

▪ From:	Applies the matching criteria to the email address From field.
▪ To:	Applies the matching criteria to the email address To field.
▪ FromOrTo:	Applies the matching criteria to both email address To and From fields, causing the rule to be applied when either field matches.

- FromAndTo: Applies the matching criteria to both email address To and From fields, causing the rule to be applied when both field match.
- Virus: Applies the matching criteria to any message that contains a virus, and matches when the virus report contains the matching criteria.

While using the case sensitive FromOrTo makes the rule more readable, MailScanner actually ignores case and order; toorfrom will be treated exactly the same as FromOrTo

Pattern

The pattern (middle field) contains criteria to match in the form of a regular expression. A regular expression is a string of characters that defines a set of one or more other strings.

Any string that is defined by a regular expression is said to match that expression. To get a regular expression to match more than one string you use special characters (such as * or ?) that have special meaning. A complete explanation of regular expressions is beyond the scope of this Manual, however some examples are provided below. There are many fine books and websites available on the subject of regular expressions. The Perl site is a good place to start.

http://www.perldoc.com/perl5.6/pod/perlre.html

In the simplest form, a regular expression will be exactly what should be matched, such as an email address. In addition, regular expressions allow more broad matches that offer great flexibility, such as the following examples:

```
user@domain.com
```

Matches exactly the listed user at the list domain. If the entry is joshua@email.com then the rule will match `joshua@email.com`.

```
user@*
```

Matches this user at any domain. If the entry is “`joshua@*`” then the rule will match `joshua@email.com, joshua@fsl.com, joshua@example.com`, and `joshua@` any domain.

```
*@domain.com
```

Matches any user at the listed domain. If the entry is *@example.com then the rule will match tom@example.com, dick@example.com, harry@example.com, and any other user at example.com

```
*@*.domain.com
```

Matches any user at any sub-domain of domain.com. If the entry is `*@*.example.com` then the rule will match `joshua@hr.example.com, ivan@hr.example.com, steve@it.example.com`, and any user at any sub domain of `example.com`.

```
192.168.
```

Matches any SMTP client IP address that starts with 192.168.

```
        default
```

This is the default matching rule when no other rule matches.

All rulesets must end with a default ruleset! Typically this is:

FromOrTo   default   yes (or no)

Regular expressions can also be fairly complex, for example;

```
        /^192\.168\.1[4567]\./
```

Matches any SMTP client IP address in the networks 192.168.14.0 to 192.168.17.0.

The matching criteria may also be a network address in CIDR address notion, such as:

```
        10.1.1.0/24
```

This matches any SMTP client IP address in the class c network 10.1.1.0.

Result

The result (third field contains) the value for the configuration option that is using this ruleset. Typically this is yes or no, but it may also be a filename. Please see Appendix C, Practical Ruleset Examples.

Chapter 6

Related Applications

Other applications may be installed with MailScanner to simplify administration and provide additional functionality. These applications include:

- MailWatch for MailScanner
- MailScanner Webmin Module
- Vispan
- MailScanner-mrtg
- phplistadmin

MailWatch for MailScanner

MailWatch for MailScanner is a web-based front-end to MailScanner written in PHP, MySQL and JpGraph and is available for free under the terms of the GNU Public License.

It comes with a CustomConfig module for MailScanner which causes MailScanner to log all message data (excluding body text) to a MySQL database which is then queried by MailWatch for reporting and statistics. Features include:

- Displays the inbound/outbound mail queue size (currently for Sendmail users only), load average and today's totals for Messages, Spam, Viruses and blocked content on each page header.
- Color-coded display of recently processed mail.
- Drill-down onto each message to see detailed information.
- Quarantine management allows you to release, delete or run sa-learn across any quarantined messages.
- Reports with customizable filters and graphs by JpGraph
- Tools to view Virus Scanner status (currently Sophos only), MySQL database status and to view the MailScanner configuration files.
- Utilities for Sendmail to monitor and display the mail queue sizes and to record and display message relay information.

MailWatch for MailScanner and instructions for installing are available from:

http://mailwatch.sourceforge.net/

To install MailWatch you must have a working MailScanner set-up and have running copies of MySQL, Apache, PHP (with MySQL and GD support). For MailScanner to log to MySQL you need Perl-DBI and Perl-DBD-MySQL installed (DBD-MySQL must be version <=2.1028, the later versions >=2.9 do not function correctly).

MailScanner Webmin Module

The MailScanner Webmin module was created to provide a simple front-end for administering MailScanner. Before installing the MailScanner Webmin Module, you must install Webmin. Webmin and instructions for installing are available from:

http://www.webmin.com/

Webmin and instructions for installing are available from:

http://sourceforge.net/projects/msfrontend/

Vispan

Vispan is a PERL script which analyses the mail log file to produce useful statistics. It requires MailScanner to provide the necessary log file entries. At the moment the virus list is dependent on the virus scanner you have installed.

Vispan can also use heuristics in the senders of the spam emails and can then automatically add them to the sendmail access file which will cause further mails to be rejected. After a definable period of time they will be removed from the access file and once again allowed to send mail to you.

Vispan and instructions for installing are available from:

http://www.while.homeunix.net/mailstats/

MailScanner-mrtg

mailscanner-mrtg provides configuration files, web pages, and related perl scripts for mrtg to monitor many aspects of your MailScanner machine. With it you will be able to monitor:

- Mail Relayed
- Files in incoming queue
- Spam Identified
- Files in outgoing queue
- Virii Caught
- Memory (Ram) Used
- Copies of MTA RunningLoad Average
- Copies of MailScanner Running
- CPU Utilization
- Disk Space Used in /var/spool
- Disk Space Used in /
- IP Traffic
- Files in quarantine
- Space used in ramdisk
- Spam and virus ratios

MailScanner-mrtg for MailScanner and instructions for installing are available from:

http://mailscannermrtg.sourceforge.net/

phplistadmin

phplistadmin is a php web GUI used to edit/create SQL and bydomain/byemail white and blacklists for MailScanner. For SQL black/white lists you must use the CustomConfig functions available from:

http://filelister.linuxkernel.at/?current=/tarballs/Mailscanner

phplistadmin for MailScanner and instructions for installing are available from:

http://sourceforge.net/projects/phplistadmin/

Network Spam Checks

Other applications may be installed with SpamAssassin to improve spam detection accuracy. These applications include:

- DCC
- Pyzor
- Razor

DCC

SpamAssassin uses DCC to add to the total SpamAssassin score. The use of DCC alone will not identify a spam message. DCC or Distributed Checksum Clearinghouse is a system of thousands of clients and about 200 servers collecting and counting checksums related to more than 100 million mail messages per day. The counts can be used by SMTP servers and mail user agents to detect and reject or filter spam or unsolicited bulk mail. DCC servers exchange or "flood" common checksums. The checksums include values that are constant across common variations in bulk messages, including "personalizations."

DCC is based on the concept that if mail recipients could compare the mail they receive, they could recognize unsolicited bulk mail. A DCC server totals reports of checksums of messages from clients and answers queries about the total counts for checksums of mail messages. A DCC client reports the checksums for a mail message to a server and is told the total number of recipients of mail with each checksum. If one of the totals is higher than a threshold set by the client and according to local white lists the message is unsolicited, the DCC client can log, discard, or reject the message.

Because simplistic checksums of spam would not be effective, the main DCC checksums are fuzzy and ignore aspects of messages. The fuzzy checksums are changed as spam evolves. Since the DCC started being used in late 2000, the fuzzy checksums have been modified several times.

DCC and instructions for installing are available from:

http://www.rhyolite.com/anti-spam/dcc/

Razor

SpamAssassin uses Razor to add to the total SpamAssassin score. The use of Razor alone will not identify a spam message. Vipul's Razor is a distributed, collaborative, spam detection and filtering network. Through user contribution, Razor establishes a distributed and constantly updating catalogue of spam in propagation that is consulted by email clients to filter out known spam. Detection is done with statistical and randomized signatures that efficiently spot mutating spam content. User input is validated through reputation assignments based on consensus on report and revoke assertions which in turn is used for computing confidence values associated with individual signatures.

Razor and instructions for installing are available from:

http://razor.sourceforge.net/

Pyzor

SpamAssassin uses Pyzor to add to the total SpamAssassin score. The use of Pyzor alone will not identify a spam message. Pyzor is a methodology for detecting spam similar to that used by razor but Pyzor has been completely rewritten in Python. It is extremely easy and quick to install. An excellent overview of how Pyzor works may be found at:

http://www.archeus.plus.com/colin/pydoc/

Pyzor and instructions for installing are available from:

http://pyzor.sourceforge.net/

Chapter 7

Tuning and Troubleshooting

We've attempted to list a few commonly used tuning tips and very basic trouble shooting information in this manual. For a more complete and up-to-date information, please visit:

http://www.mailscanner.biz/maq.html

While many of the techniques covered below are Linux specific, the principals may be applied to all operating systems. The commands and file locations used in the examples below are Linux specific and the actual command you may need to run may be different.

Tuning

There are a few quick steps that may be taken to improve performance. These specific instructions are for Linux distributions only, but similar techniques may be used on other operations systems.

Using a tmpfs files system: MailScanner "unpacks" messages for scanning on `/var/spool/MailScanner/incoming`. If your system has sufficient memory, mounting this directory on a tmpfs (in memory) file system will improve performance. To setup this tmpfs, modify `/etc/fstab` to add the line:

```
none   /var/spool/MailScanner/incoming   tmpfs   defaults      0 0
```

> Be sure to add this line below the point at which the /var partitions is mounted and available.

Then as root, issue the command:

```
mount -a
```

Issuing the command:

```
mount
```

Should show that `/var/spool/MailScanner/incoming` is now mounted on the tmpfs.

No email will be lost if the system crashes. MailScanner never removes a message from the incoming mail queue until it is fully written to the outgoing mail queue. If the system crashes, when MailScanner restarts, it will find the "lost" messages in the incoming mail queue and process these messages normally.

Speed Logging: /etc/syslog.conf may be modified to omit file syncing the log file after each log event is written. Note that you might lose information if the system crashes right behind a write attempt, but this will give better performance since email gateways log extensively in a very verbose manner.

Trouble shooting

Reading logs: The most effective method of locating a MailScanner problem is **reading the logs**! This should always be your starting point for identifying a problem. The exact location of the relevant logs is operating system and MTA dependent. On Linux MailScanner systems using sendmail as the MTA, all MailScanner and sendmail log information is typically written to `/var/log/maillog`. Errors or anomalies in this file will give you an indication of what is causing the problem.

Debugging MailScanner: Temporarily modify your MailScanner.conf file to set:

```
Debug = yes
Debug SpamAssassin = yes
```

Then restart MailScanner. This will cause one MailScanner process to scan one batch of messages from the incoming mail queue and print verbose output to the terminal. Carefully check this output for error messages.

Don’t forget to modify the MailScanner.conf file to turn off debugging and restart MailScanner after these modifications have been made

Debugging SppamAssassin: An excellent method to find out if Bayesian filtering, Pyzor, Razor and DCC are being used by SpamAssassin is to run the command:

```
spamassassin -D - p /etc/Mailscanner/spam.assassin.prefs.conf \
       --lint
```

This will run a test message through SpamAssassin and print verbose output to the terminal.

Getting Help

Useful detailed installation instructions for different operating systems, MTAs and specific virus scanners may be found at:

http://www.mailscanner.info/install/

Search the MailScanner FAQs. This is especially useful for help with installations and configuration problems. The FAQs are located at:

http://www.mailscanner.info/serve/cache/1.html

If the information above fails to solve your problem, first search the MailScanner List Archives located at:

http://www.jiscmail.ac.uk/cgi-bin/wa.exe?S1=mailscanner

Search the archive using short identifying terms from the error messages you found while reading the logs and using debug as described above.

If these methods fail (and they seldom do), join the MailScanner List and post your problem. This is an excellent support list and questions are almost always answered very quickly. Be prepared for a mild rebuke:

- If you failed to follow the steps listed above first.
- If you post in HTML format (please use "plain text" format)
- If you fail to include the following information in you request for help:
  - Your Operating System including version
  - The version of MailScanner you are using
  - A complete description of your problem
  - Relevant "snippets" for your logs or debug output. (Not the whole log please.)

There are 2 mailing lists for MailScanner users.

- The "announcements only" list is where all announcements will be made of new versions of MailScanner and associated software such as MailWatch and MailScanner-MRTG. You can subscribe to this list by sending an email to:

  `jiscmail@jiscmail.ac.uk`

  Containing:

  ```
  join mailscanner-announce your-first-name your-last-name
  ```

- The general discussion list is where all new features, configuration issues and Mailscanner problems are discussed. This is where to go if you need troubleshooting help. You can subscribe to this list by sending an email to;

  `jiscmail@jiscmail.ac.uk`

  Containing, in the Subject or Body:

  ```
  join mailscanner your-first-name your-last-name
  ```

Appendix A

Installing Red Hat Enterprise Linux

This appendix currently only contains instructions for installing Red Hat ES and As version 3.0, It is hoped that contributions for other operating system installations will soon follow.

This section provides step by step instructions for installing Red Hat Enterprise Linux, ES and AS version 3.0 for use with MailScanner and Related applications. For information regarding any problems encountered while installing RHEL, please contact Red Hat support.

These instructions do NOT install a graphical user interface.

Installation of Red Hat Enterprise Linux 3 (ES or AS)

1. Boot the machine to be installed with the first Red Hat CD.
2. Press Enter at the first text prompt
3. If the CDs were downloaded and burned, the installer will prompt to perform a Media Check. Perform this check.
4. Welcome: click next.
5. Language Selection: Accept the default of English and click next.
6. Keyboard: Accept the default of U. S. English and click next.
7. Mouse Configuration: Choose your mouse, and click next.
8. Disk Partitioning Setup: Accept the default of Automatically Partition and click next.
9. Automatic Partitioning: Accept the default of Remove All Partitions and click next.
10. Click Yes at the Warning confirmation.
11. Partitioning: Accept the default partition scheme and click next.
12. Boot Loader Configuration: Accept the default and click next.
13. Network Configuration: Make the following changes
    a. Click the “Edit” button under Network Devices
    b. Unselect “Configure using DHCP”
    c. Fill in the IP address and Netmask for your network
    d. Click OK

    e. Under Hostname, fill in the hostname that resolves to the IP entered for this machine.
    f. Fill in the information under Miscellaneous Settings. If you are unsure as to this information, see your Network Administrator. Note: Tertiary DNS may be left blank.
    g. Click next.

14. Firewall: Select “No Firewall” and click next.
15. Time Zone Selection: Choose your time zone, and click next.
16. Set Root Password: Enter a password for the system's root account. This password is used for system administration. It is important to keep this password in a safe place. Once the password is entered, click next.
17. Package Defaults: Select “Customize the set of packages to be installed” and click next.
18. Package Group Selection: Accept the defaults that are checked and add the following:
    a. Scroll down and select “Mail Server”
    b. Scroll down and select “MySQL Database”
    c. Click “Details” for MySQL Database
    d. Select “php-mysql” in the Details popup window
    e. Click OK
    f. Scroll down and select “Network Servers”
    g. Unselect all optional packages
    h. Select only “openldap-servers”
    i. Click OK
    j. Select “System Tools”
    k. Scroll down and select “Development”
19. Click next.
20. About to Install: Click next.
21. Insert the RHEL CDs as they are requested.
22. Graphical Interface (X) Configuration: Select your video card, or accept the default and click next.
23. Monitor Configuration: Select your monitor, or accept the default and click next.
24. Customize Graphical Configuration: Select “Text” as the login type and click next.

Click Exit and the installation is complete!

The mysql-server rpm is no longer shipped with Red Hat ES or AS. You will need this package if you plan to run MailWatch. It may be obtained from:

http://www.mysql.com/downloads/mysql-4.0.html

Appendix B

Installing Third Party Virus Scanners

MailScanner can be configured to use one or more virus scanner to scan incoming email for viruses, however installing multiple virus scanning engines will have an impact on performance.

Installing most virus scanners to work with MailScanner is as simple as

Install the virus scanning engine according to the products installation instructions

Configure MailScanner to use the installed Virus Scanner (Chapter 3 Configuring MailScanner)

Please review the table below for any additional instructions which may be required to install your specific Virus Scanner or Scanners.

MailScanner configuration name	Installation note	Product Name	Manufacturers Web Site
sophos	note 2 & 5	Linux on Intel	www.sophos.com
sophossavi (SAVI perl module)	note 3 & 5	Linux on Intel	http://www.mailscanner.info/install/SAVI.shtml
mcafee		McAfee VirusScan Unix	www.mcafee.com
command	note 1	Command AntiVirus for Linux	www.command.co.uk
kaspersky-4.5		discontinued	www.kaspersky.com
kaspersky (older versions)	note 1	Kaspersky® Anti-Virus for Linux File Server	www.kaspersky.com

kavdaemonclient	note 1	Kaspersky® Anti-Virus for Linux File Server	www.kaspersky.com
etrust	note 1		http://www3.ca.com/Solutions/Product.asp?ID=156
inoculate (CAI)		discontinued	www.cai.com
inoculan (CAI)		discontinued	www.cai.com
nod32	note 1	NOD32 for Linux Mail Server	www.nod32.com
f-secure	note 1	F-Secure Anti-Virus for Servers for Linux	www.f-secure.com
f-prot	note 1	F-Prot Antivirus for Linux Mail Servers	www.f-prot.com
panda	note 1	Panda Antivirus for Linux	www.pandasoftware.com
rav		discontinued	linux support discontinued
antivir	note 1	AntiVir for Linux	http://www.hbedv.com
clamav	note 1	ClamAV	www.clamav.net
clamavmodule (ClamAV perl module)	note 4	ClamAV	http://www.mailscanner.info/install/ClamAVModule.shtml
trend (a.k.a.TrendMicro)	note 1	InterScan VirusWall for Linux	www.trendmicro.com

Note 1: Install According to Manufacturer's directions

Note 2: Use the following steps.

1. Obtain the file `linux.intel.libc6.tar.Z` by:
   Copy from the Sophos CDROM to /tmp/Sophos

or

Edit the file `/usr/sbin/MajorSophos.sh` to add your Sophos username and password, i.e.:

```
WEBUSER="<your_username>"
WEBPASS="<your_password>"
```

2. Then run: `/usr/sbin/MajorSophos.sh -download`

This command will download linux.intel.libc6.tar.Z to `/tmp/MajorSophos.sh.xxxx` where `xxxxx` is a string dependent on the version downloaded.

3. After copying or downloading `linux.intel.libc6.tar.Z`, cd to the directory where the file was copied or downloaded

`cd /tmp/MajorSophos.sh.xxxx` (downloaded)

or

`cd /tmp/Sophos` (copied)

Uncompress and un-tar the file linux.intel.libc6.tar.Z

```
uncompress linux.intel.libc6.tar.Z
```

This will create a directory `sav-install` in the current directory

```
cd sav-install
```

Then run the command

```
/usr/sbin/Sophos.install
```

This installs Sophos in /usr/local/Sophos

Note 3: First install Sophos according to the directions in Note 2 above. Then download and install the SAVI perl module according to the instructions at:

http://www.mailscanner.info/install/SAVI.shtml

EXCEPT you will NOT need to change the MailScanner.conf variable:

Minimum Code Status = beta

Note 4: Install ClamAV first and the install the SAVI perl module according to the instructions at:

http://www.mailscanner.info/install/ClamAVModule.shtml

Note 5: While IDE files for new viruses will be updated hourly by default. You must manually update the Major Sophos virus definition file monthly using the CD supplied by Sophos or by running the following command (see instructions above).

```
/usr/sbin/MajorSophos.sh
```

Virus scanner product and pricing comparisons from the MailScanner list archives:
http://www.jiscmail.ac.uk/cgi-

bin/wa.exe?A2=ind0309&L=mailscanner&P=R145271&I=-1

Appendix C

Practical Ruleset Examples

The use of Rulesets gives you great power and flexibility in configuring MailScanner. Almost any MailScanner configuration value that can be set to yes or no can also be pointed at a Ruleset.

MailScanner provide a Ruleset as the value for white listed addresses:

```
Is Definitely Not Spam = %rules-dir%/spam.whitelist.rules
```

You can add the same function for black listing addresses or domains

Spam Black List

In MailScanner.conf set:

```
Is Definitely Spam = %rules-dir%/spam.blacklist.rules
```

In the new `spam.blacklist.rules` file, set addresses to be blacklisted using rules such as

```
# Addresses to be blacklisted.
# Rules which match below will always be marked as spam
From:           user@nasty.domain.com   yes
From:           *@spammers.com          yes
# Mark an entire network used by spammers
From:           123.231.3.              yes
ToOrFrom:       default                  no
```

Always end every ruleset with a default value. This should be the default value for anything that does not match a regular expression listed in the ruleset.

Only Sign Outgoing Messages

In MailScanner.conf set:

```
Sign Clean Messages = %rules-dir%/signing.rules
```

If your messages come from "yourdomain.com" and yourdomain.com can be identified by IP addresses that all start with 192.168., your `signing.rules` file would look like this:

```
# Addresses which should not be signed by MailScanner.
```

```
From:         192.168.                 yes
FromOrTo:     default                  no
```

Whenever possible, use IP addresses not domain names to identify systems or network blocks.

Use Different Signatures for Different Domains

In MailScanner.conf set:

```
Inline Text Signature = %rules-dir%/sig.text.rules
```

And

```
Inline HTML Signature = %rules-dir%/sig.html.rules
```

In the new `sig.text.rules` file, set addresses to receive different signatures similar to the example below:

```
# Addresses which should signed differently by MailScanner.
From:         *@domain1.com
/opt/MailScanner/etc/reports/domain1.sig.txt
From:         *@domain2.com
/opt/MailScanner/etc/reports/domain2.sig.txt
```

And add equivalent rules in the `sig.html.rules` file.

Only Virus Scan Some Domains

In MailScanner.conf set:

```
Virus Scanning = %rules-dir%/virus.scanning.rules
```

In the new **`virus.scanning.rules`** file, set addresses which should not be virus scanned similar to the example below:

```
# Addresses which should not be virus scanned by MailScanner.
FromOrTo:     user@morespam.com        yes
FromOrTo:     *@scanme.com             yes
FromOrTo:     *@scanme-too.com         yes
FromOrTo:     default                  no
```

Send System Administrator Notices to Several People

In MailScanner.conf set:

```
Notices To = %rules-dir%/notices.to.rules
```

Create the new `notices.to.rules` file, following the example below:

```
# Send notices to administrators to different lists
To:           @abc.com     postmaster@me.com george@abc.com
To:           @def.com     /etc/MailScanner/rules/techies.txt
FromOrTo:     default      postmaster@me.com
```

Note a reference to a file must include the full path and filename. It must start with a "/" and end in something other than "/". The rule will be replicated for all the entries in the file. Note that a reference to a file can contain another (nested) reference to a file. Beware of too many levels of indirection.

For the @def.com notices, create the file `/etc/MailScanner/rules/techies.txt` which should contain entries similar to:

```
# comment - MailScanner notices for def.com will be sent to
jim@def.com
frank@def.com
*@techies.def.com
hank@somewhereelse.com
/etc/MailScanner/rules/nested-filename.txt
```

Nested file format rules:

1. One pattern or address per line. The allowable patterns are the same as the normal patterns in any normal ruleset file.
2. Comments start with # and continue until the end of the line.
3. Blank lines are ignored.
4. Leading and trailing white space is ignored.
5. Further filenames can be included, allowing you to nest these files if you really need to.

Scan for spam only from certain domains

In MailScanner.conf set:

```
Use SpamAssassin = %rules-dir%/use.sa.rules
```

Create the new `use.sa.rules` file, following the example below:

```
# Don't use SpamAssassin for entries on this list
To:         *@checkme.com         yes
To:         *@dontcheck.com       no
FromOrTo:   default               no
```

Filename and Filetype Checking for Specified Domains

Create the files:

%etc-dir%/filetype.rules.allowall.conf

%etc-dir%/filename.rules.allowall.conf

Where the contents of both files is:

```
# This Ruleset will allow all attached files to pass
allow   .*      -       -
```

The four fields in these files MUST be separated by tabs

Then create the file:

%rules-dir%/filename.rules

Where the contents of this file are:

```
# File to control which domains get filename checking
# mail form or to noscan.com will not have filenames checked
FromOrTo:     noscan.com
/etc/MailScanner/filename.rules.allowall.conf

# Allow local to let MailWatch release quarantined files
From:   127.0.0.1
/etc/MailScanner/filename.rules.allowall.conf
FromOrTo: default   /etc/MailScanner/filename.rules.conf
```

Then create the file:

%rules-dir%/filetype.rules

Where the contents of this file is:

```
# File to control which domains get filetype rule checking
# mail form or to noscan.com will not have filetypes checked
FromOrTo:     noscan.com
/etc/MailScanner/filetype.rules.allowall.conf

# Allow local to let MailWatch release quarantined files
From:   127.0.0.1
/etc/MailScanner/rules/filetype.rules.allowall.conf
FromOrTo: default   /etc/MailScanner/filetype.rules.conf
```

Each rule should be typed on one line in these files

In MailScanner.conf set:

```
Filename Rules = %rules-dir%/filename.rules
Filetype Rules = %rules-dir%/filetype.rules
```

Then reload MailScanner

Appendix D

Upgrading MailScanner (rpm Version)

Upgrading the rpm vision of MailScanner is typically relatively quick and painless. The first step is to download the latest version of MailScanner from:

http://www.mailscanner.info/downloads.shtml

The Upgrade

After downloading simply unpack the upgrade and, as the root user, cd into the newly created directory, i.e.:

```
cd MailScanner-4.30.3-2
```

And then simply run the install script:

```
./install.sh
```

This will update all MailScanner files.

After the script successfully completes, you will need to:

- Update MailScanner.conf
- Check for any .rpmnew files

Upgrading Mailscanner.conf

Most often the newer version of MailScanner will include new Configuration Variables. The script `/usr/sbin/upgrade_MailScanner_conf` will automatically create a new `MailScanner.conf` file which preserves all of your current MailScanner configurations values. To use this utility:

```
cd /etc/MailScanner
```

Backup a copy of your current MailScanner.conf file:

```
cp MailScanner_conf MailScanner_conf.<old-version-id>
```

Then stop MailScanner and update `Mailscanner.conf`:

```
/usr/sbin/upgrade_MailScanner_conf \
MailScanner.conf MailScanner.conf.rpmnew > \
MailScanner.new


mv MailScanner.conf MailScanner.old
```

```
      mv MailScanner.new  MailScanner.conf
```

Installing .rpmnew files

If you have changed any of MailScanner's standard files, your changes will not be overwritten. Instead the MailScanner upgrade will leave your changed files in place and install the new version of the file with an `.rpmnew` added to the filename.

For the rpm MailScanner distribution, these files will typically be in or under the `/etc/MailScanner` and `/usr/lib/MailScanner directories`. To quickly identify any of these files:

```
      find /etc/MailScanner "*.rpmnew"
      find /usr/lib/MailScanner "*.rpmnew"
```

Once you have located the new .rpmnew files you will need to **`diff`** the existing file and the `.rpmnew` file to determine if you need to edit your existing file.

Once all of the `.rpmnew` new files have been incorporated, restart MailScanner. Your upgrade is complete.

Be sure to tail the log files to be certain that MailScanner has restarted correctly and is processing mail normally.

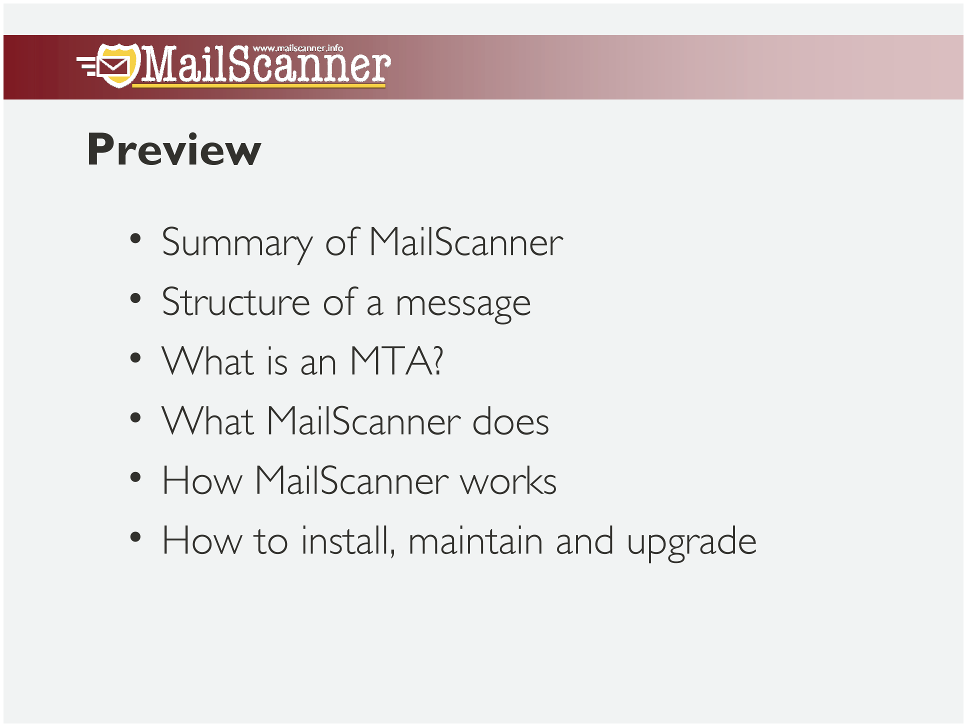
Part 2

Training Manual

Julian Field

Contents

I
Training Introduction

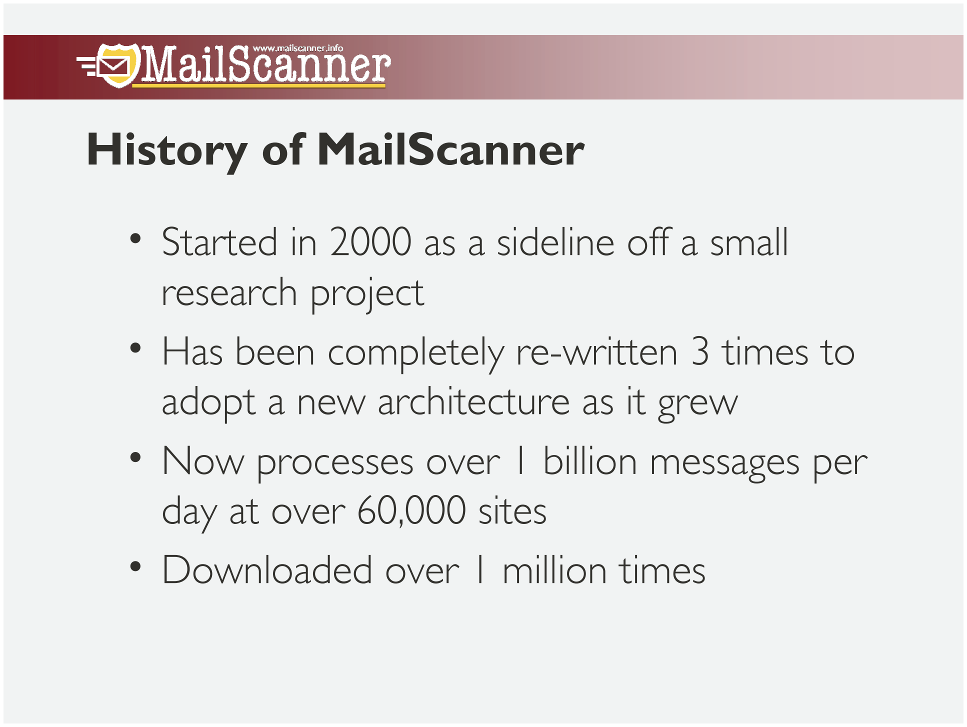


Preview

- Summary of MailScanner
- Structure of a message
- What is an MTA?
- What MailScanner does
- How MailScanner works
- How to install, maintain and upgrade

Apologies in advance if some of this is a bit heavy. It is important that we cover all the details of MailScanner and its configuration files.

If you get bored and start wanting to gnaw your own legs off, read the Appendix, it's quite amusing…



MailScanner

www.mailscanner.info

History of MailScanner

- Started in 2000 as a sideline off a small research project
- Has been completely re-written 3 times to adopt a new architecture as it grew
- Now processes over 1 billion messages per day at over 60,000 sites
- Downloaded over 1 million times

Back in 2000, the University of Southampton surveyed the marketplace for email virus scanners, as spam wasn't a problem back then. We found that all the major vendors wanted about £50k to £70k per year for virus scanning the email coming in to the University. This did not include any scanning of internal mail or outgoing mail.

This was far more than we could afford, whatever the demonstrable apparent cost benefit.

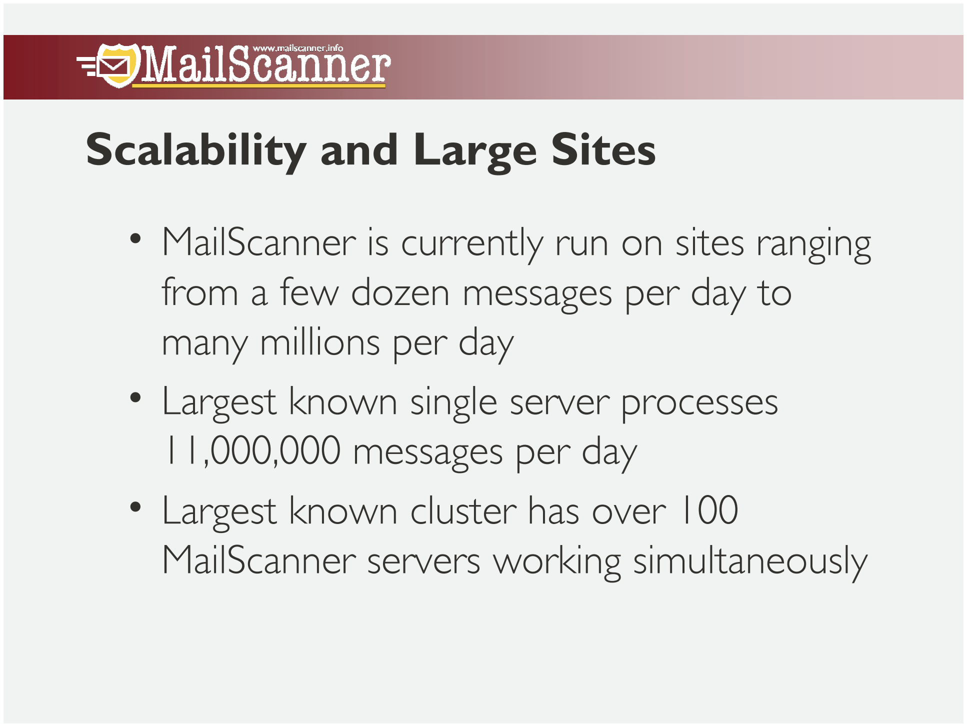
I thought that if the commercial companies could do this, then so could I. Over lunch one day the idea came to me of how to scan email for viruses very efficiently. All the other systems around scan messages one at a time, I wanted to scan batches of messages at once with the minimum I/O possible.

10 days later, I deployed version 1 (it didn't even have a name) in our department, and it worked very well. It was about 1,200 lines of code.

Six years later, I am still working on MailScanner. I have rewritten it from scratch three times to handle the new architectures needed to support multiple MTA's, multiple virus scanners, SpamAssassin and a greatly increased mail load.

It is now over 49,000 lines and has around 300 configuration options.

The mailing list includes members from well over 85 countries, including all sorts of commercial and government types who subscribe using an alias and a Hotmail address.



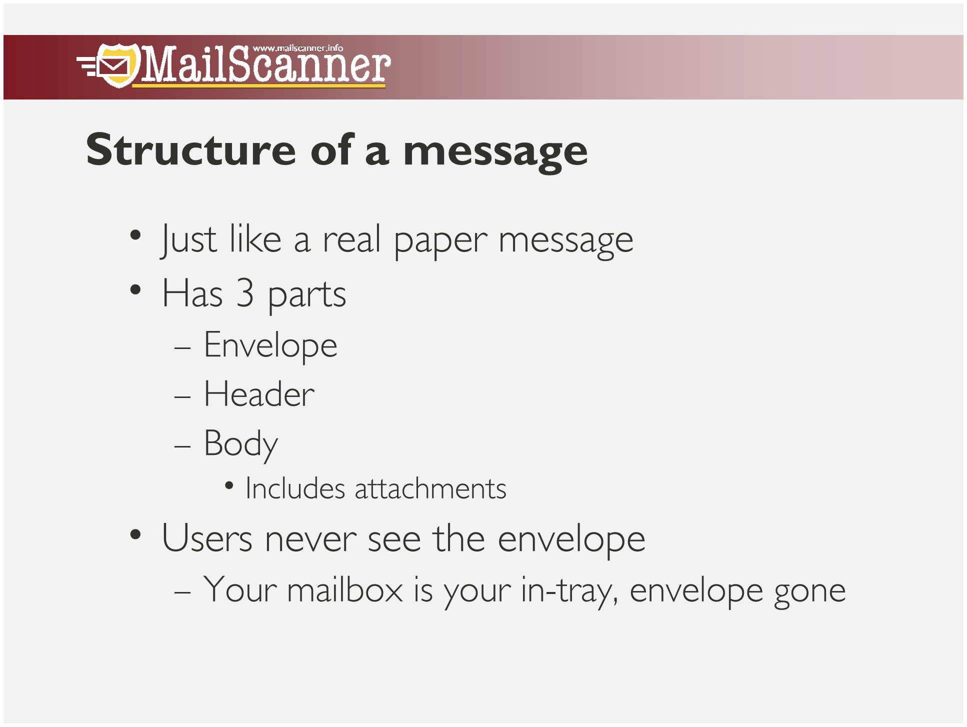
Scalability and Large Sites

- MailScanner is currently run on sites ranging from a few dozen messages per day to many millions per day
- Largest known single server processes 11,000,000 messages per day
- Largest known cluster has over 100 MailScanner servers working simultaneously

MailScanner is designed to be highly scalable and takes full advantage of all CPU, disk and network resources given to it.

MailScanner currently runs on PCs running every flavour of Unix and Linux, Apple Macintoshes, all sorts of Unix systems and mainframes up to the size of IBM z-Series systems. It works in clusters ranging in size from lone single-processor systems to clusters of over 100 multi-processor servers scanning mail in parallel.

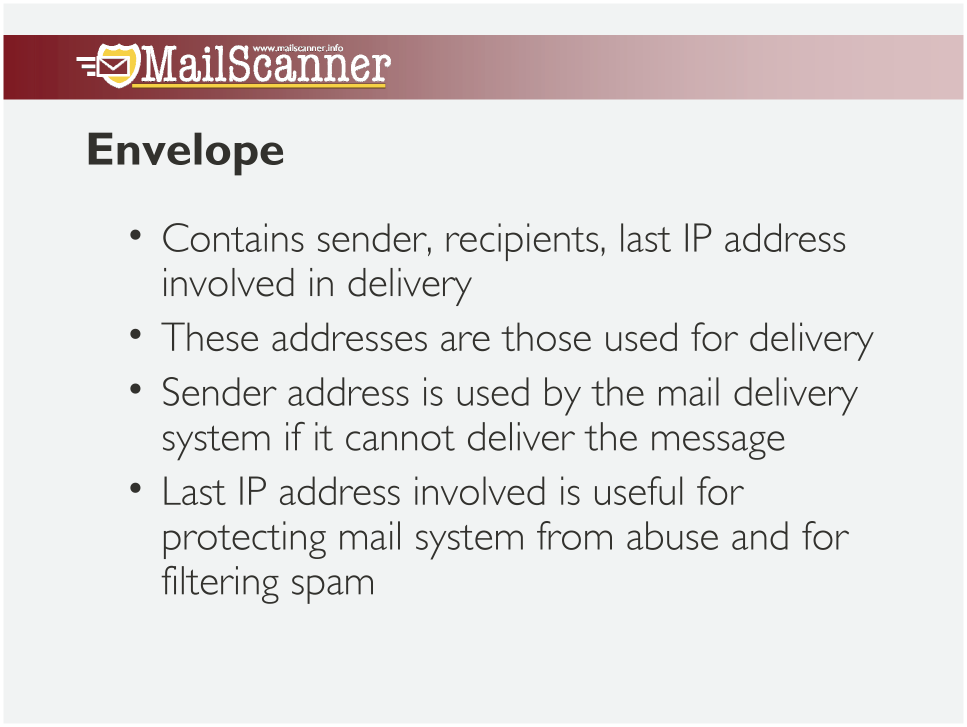
2
Email Message Structure



Structure of a message

- Just like a real paper message
- Has 3 parts
  - Envelope
  - Header
  - Body
    - Includes attachments
- Users never see the envelope
  - Your mailbox is your in-tray, envelope gone

Imagine a paper letter in an envelope. The postal system is much the same as the real physical one. To send a letter to Germany, you use your local postbox, you don't travel to Germany and post it into a German postbox. The mail system handles everything after that. The last thing to happen is that the letter is taken out of the envelope, the envelope is thrown away and the letter is put in your in-tray. Normally none of the details of the return address or the real envelope recipient address are copied to the letter in your in-tray, as you do not normally need them.

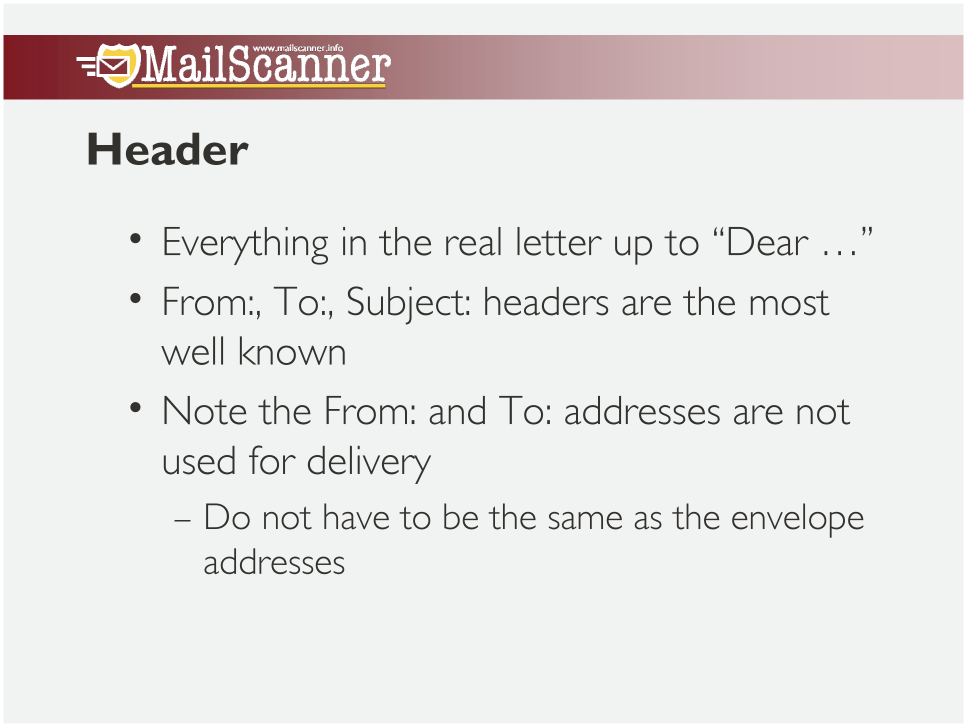


www.mailscanner.info
MailScanner

Envelope

- Contains sender, recipients, last IP address involved in delivery
- These addresses are those used for delivery
- Sender address is used by the mail delivery system if it cannot deliver the message
- Last IP address involved is useful for protecting mail system from abuse and for filtering spam

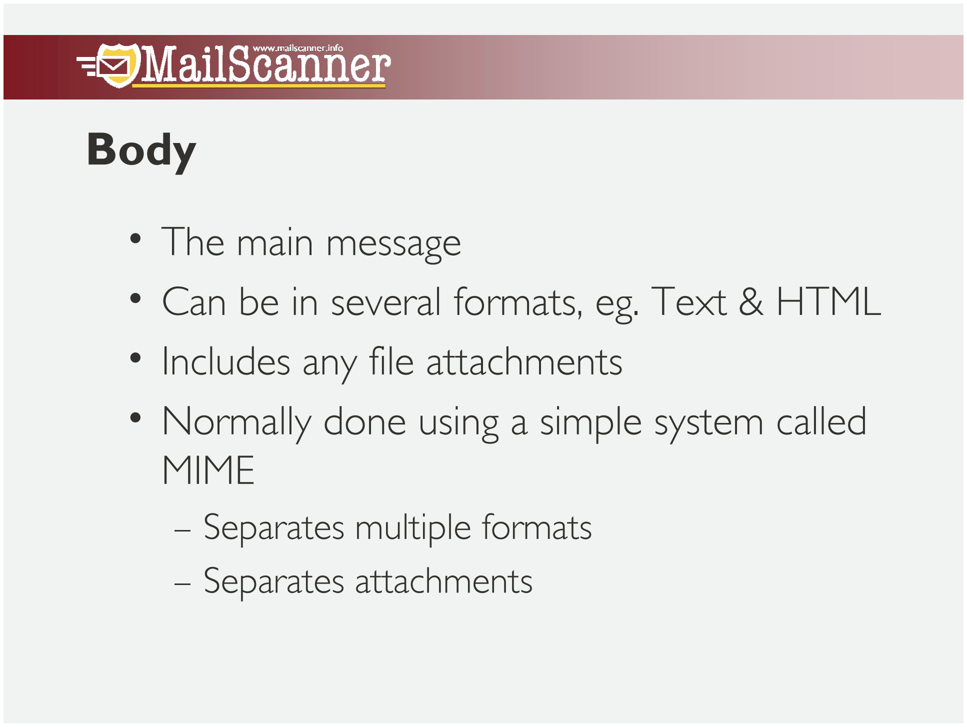
The envelope recipients are the addresses actually used to control delivery, just like the address written on the outside of a paper envelope.



Header

- Everything in the real letter up to “Dear …”
- From:, To:, Subject: headers are the most well known
- Note the From: and To: addresses are not used for delivery
  - Do not have to be the same as the envelope addresses

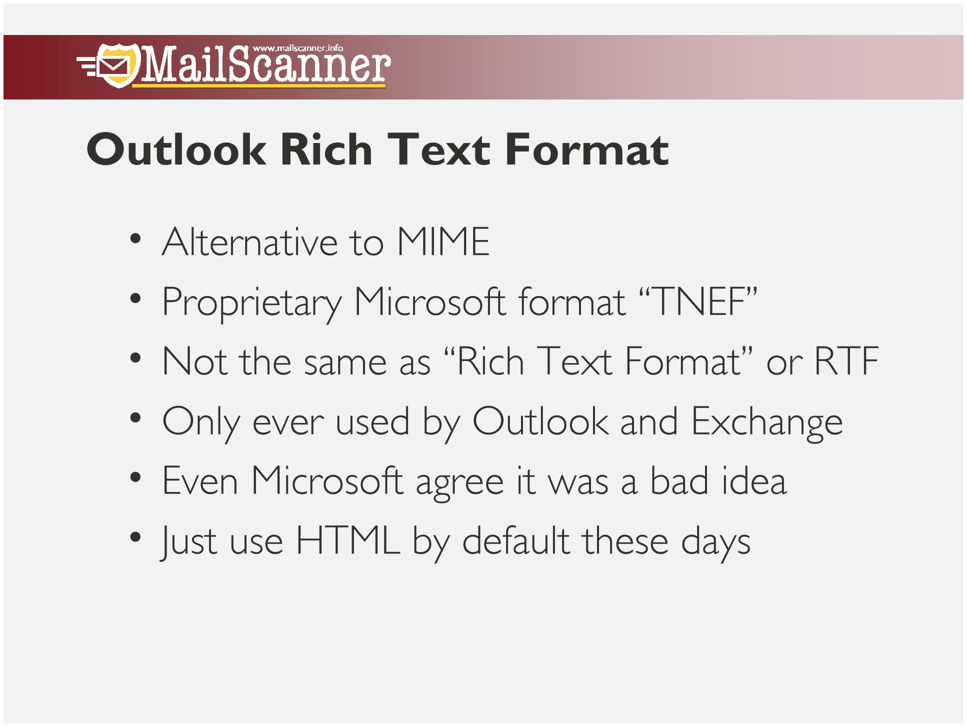
It is only a convention to put the same address on the paper letter as on the outside of the envelope. E-mail does not have envelopes with transparent windows in the front. Many mailing lists (and most spam) has different addresses on the letter and the envelope.



Body

- The main message
- Can be in several formats, eg. Text & HTML
- Includes any file attachments
- Normally done using a simple system called MIME
  - Separates multiple formats
  - Separates attachments

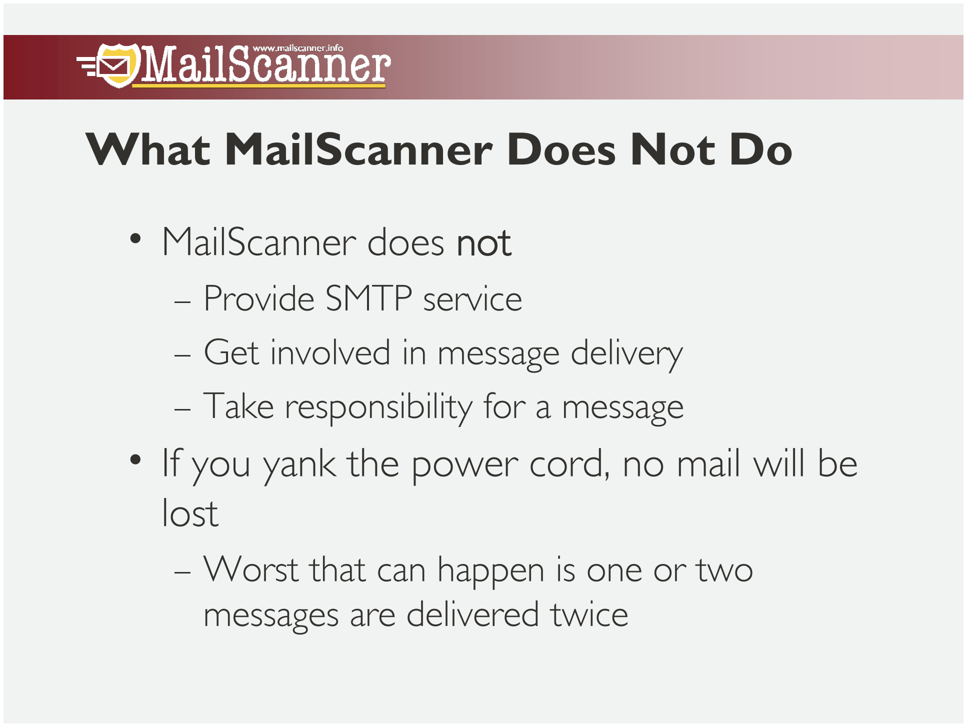
All the attachments are part of the body of the message, they are not transmitted as separate files. The MIME structure of the message tells the end user’s email application how to separate the attachments from each other.



Outlook Rich Text Format

- Alternative to MIME
- Proprietary Microsoft format “TNEF”
- Not the same as “Rich Text Format” or RTF
- Only ever used by Outlook and Exchange
- Even Microsoft agree it was a bad idea
- Just use HTML by default these days

3
The Mail Delivery Process

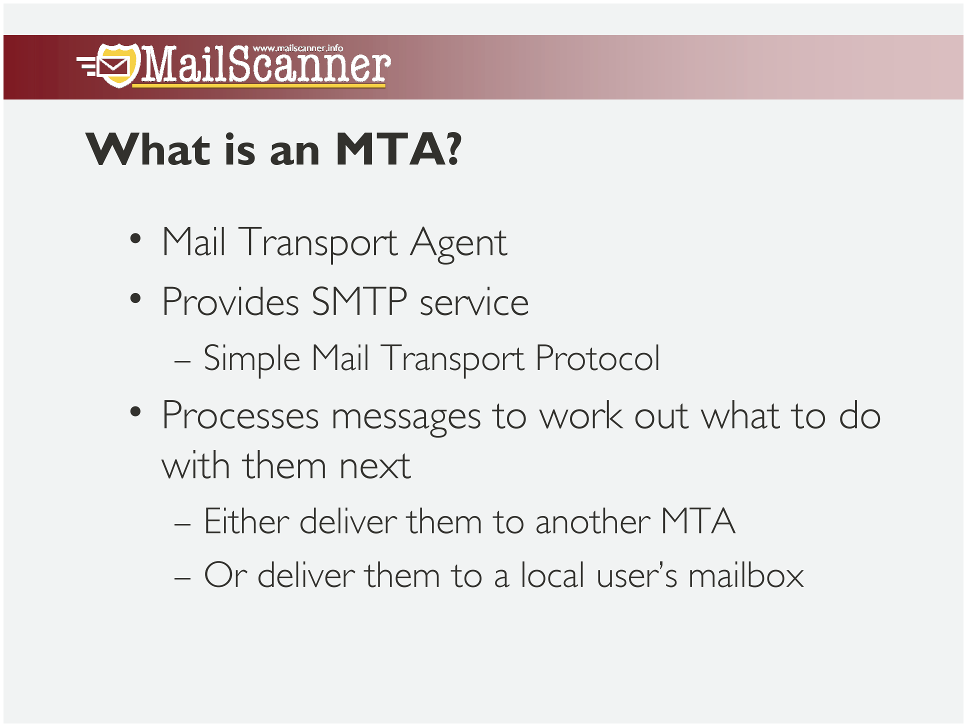


www.mailscanner.info MailScanner

What MailScanner Does Not Do

- MailScanner does **not**
  - Provide SMTP service
  - Get involved in message delivery
  - Take responsibility for a message
- If you yank the power cord, no mail will be lost
  - Worst that can happen is one or two messages are delivered twice

MailScanner never takes responsibility for a message. The message is always present in either the incoming mail queue or the outgoing mail queue or, very briefly, both. If you pull the power out no messages can be lost. The SMTP protocol demands that all incoming messages are written to disk before the server declares it has received the message.

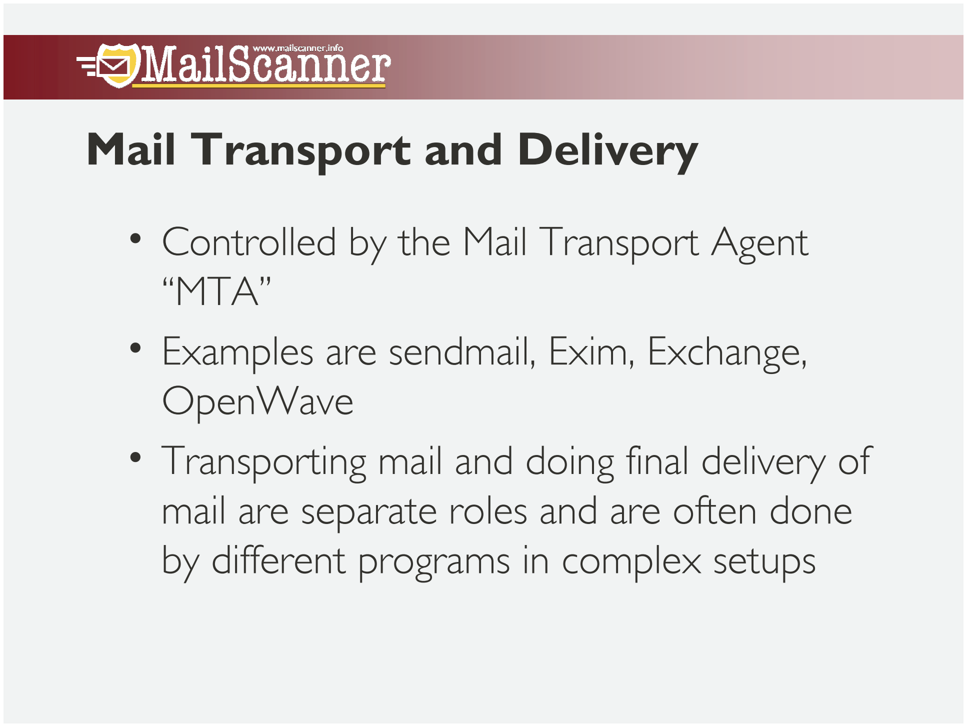


www.mailscanner.info MailScanner

What is an MTA?

- Mail Transport Agent
- Provides SMTP service
  - Simple Mail Transport Protocol
- Processes messages to work out what to do with them next
  - Either deliver them to another MTA
  - Or deliver them to a local user's mailbox

Every server responsible for carrying mail to its destination has to run an MTA. All major MTA's speak the same language, so each server can be running a totally different MTA on a different operating system on different hardware. In fact they usually are totally different, but there are a few major systems that handle the vast proportion of all the mail on the Internet. All the major ones are freely available open source packages. Relatively little mail is carried by commercial software.

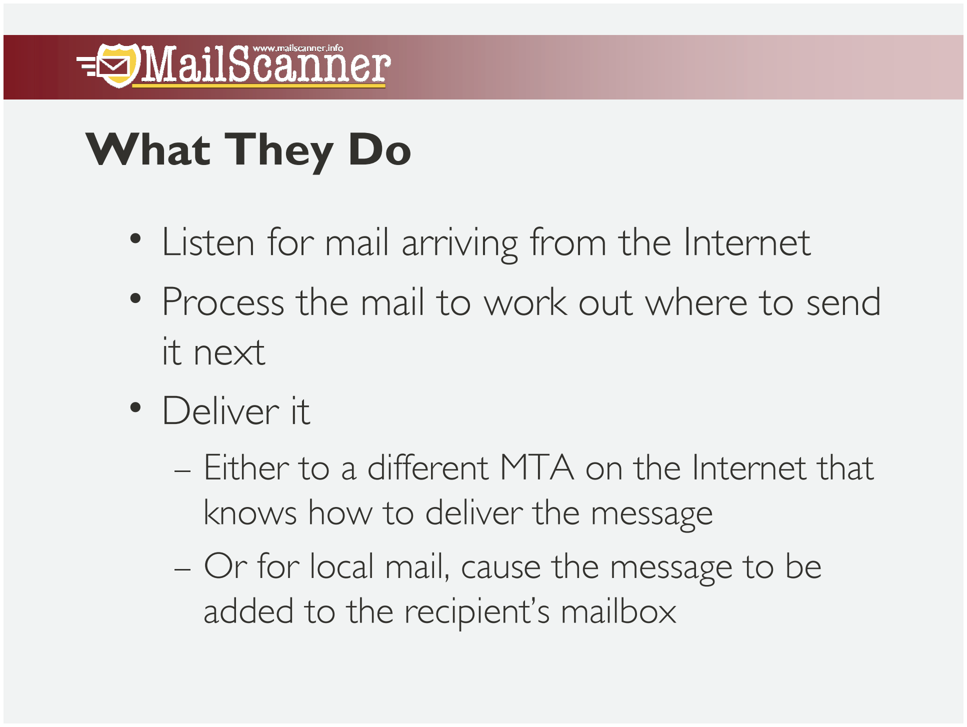


Mail Transport and Delivery

- Controlled by the Mail Transport Agent "MTA"
- Examples are sendmail, Exim, Exchange, OpenWave
- Transporting mail and doing final delivery of mail are separate roles and are often done by different programs in complex setups

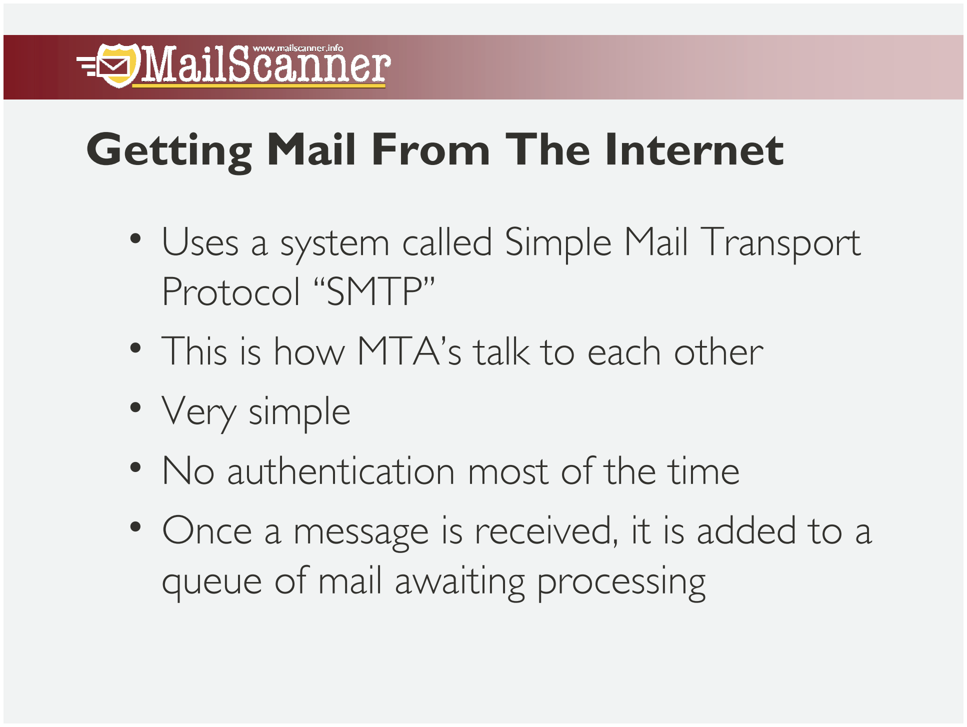
Many MTA's these days include the program used to finally deliver the messages into a user's mailbox. It is useful to realise that delivery into a mailbox does not **have** to be done by the MTA.

Mail retrieval via POP, IMAP or a web-based interface (such as that provided by Microsoft Exchange) is a totally different subject, which we will not be covering.



What They Do

- Listen for mail arriving from the Internet
- Process the mail to work out where to send it next
- Deliver it
  - Either to a different MTA on the Internet that knows how to deliver the message
  - Or for local mail, cause the message to be added to the recipient's mailbox

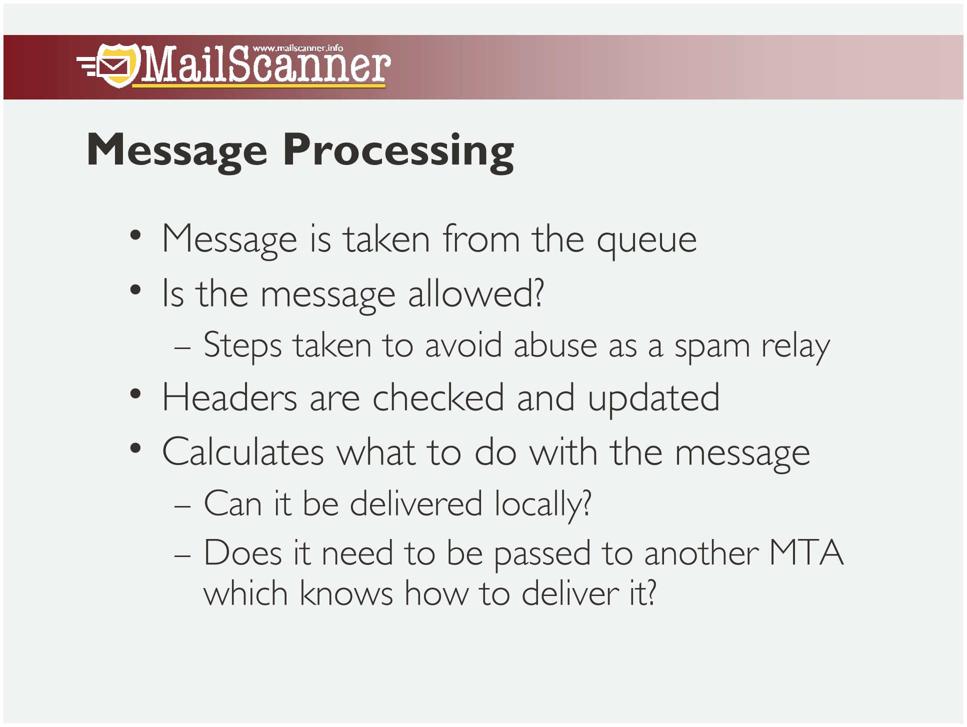

Getting Mail From The Internet

- Uses a system called Simple Mail Transport Protocol "SMTP"
- This is how MTA's talk to each other
- Very simple
- No authentication most of the time
- Once a message is received, it is added to a queue of mail awaiting processing

SMTP looks like a very simple protocol, but it is very difficult and complex to write a good SMTP server that handles all the ways that a network connection could fail, and all the other things that could potentially go wrong during the transmission of a message. I do not believe in writing things which are

1. Hard, and
2. Already done by someone else.

The same applies to delivering mail into a user's mailbox, and writing a virus scanning engine. MailScanner does not include any of these, it uses other packages to do them.


Message Processing

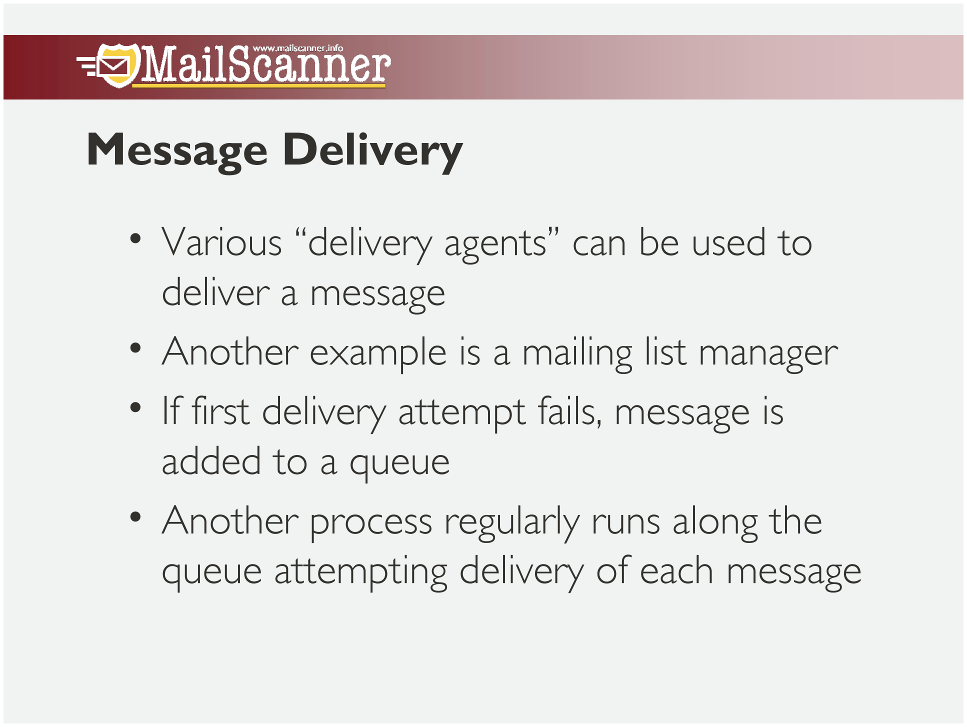
- Message is taken from the queue
- Is the message allowed?
  - Steps taken to avoid abuse as a spam relay
- Headers are checked and updated
- Calculates what to do with the message
  - Can it be delivered locally?
  - Does it need to be passed to another MTA which knows how to deliver it?

Say you are an MTA at mta.lazy.com. You should only accept mail which is either

1. from an IP address on your network, or
2. addressed to one of your users

An "open relay" is one that accepts mail from anywhere and will deliver it to any address. So if you took mail from spammer@spam.com addressed to innocent@user.com and happily delivered it to the mail system belonging to user.com, you would be relaying it. This is bad. There are huge lists of mail servers around the world which are open relays and the spammers use them to send large amounts of spam to innocent users.

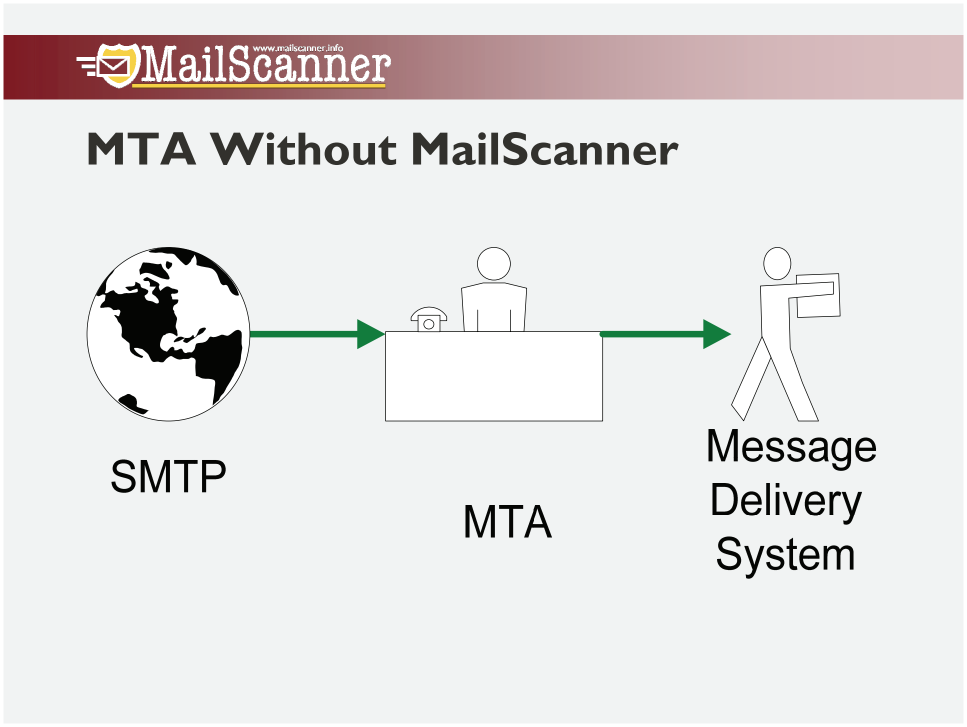
You could transmit a message to mta.lazy.com once, with 1000 recipient addresses in its envelope. mta.lazy.com would then do all the work of expanding it to 1000 different messages all going to totally different places. The spammer has been saved an awful lot of time and bandwidth, as well as the ability to hide themselves.



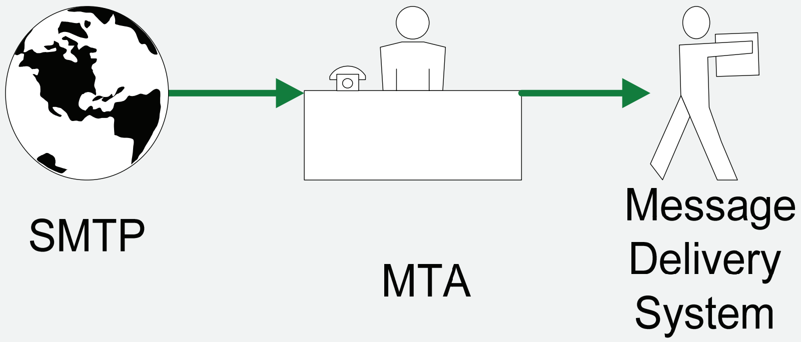
Message Delivery

- Various "delivery agents" can be used to deliver a message
- Another example is a mailing list manager
- If first delivery attempt fails, message is added to a queue
- Another process regularly runs along the queue attempting delivery of each message

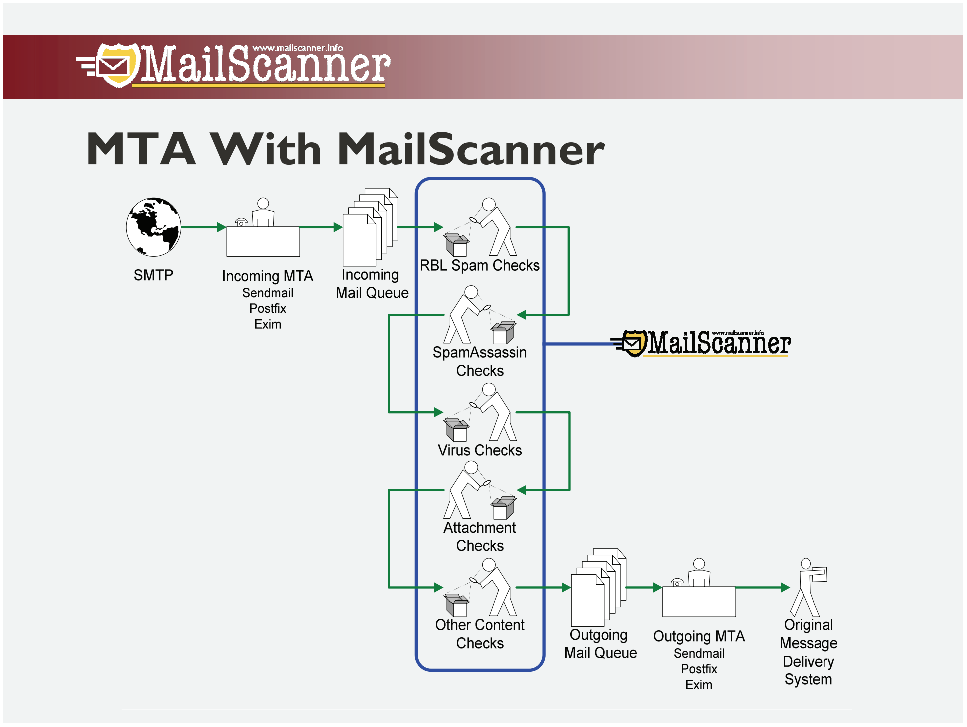
4
MailScanner: What It Does



MTA Without MailScanner

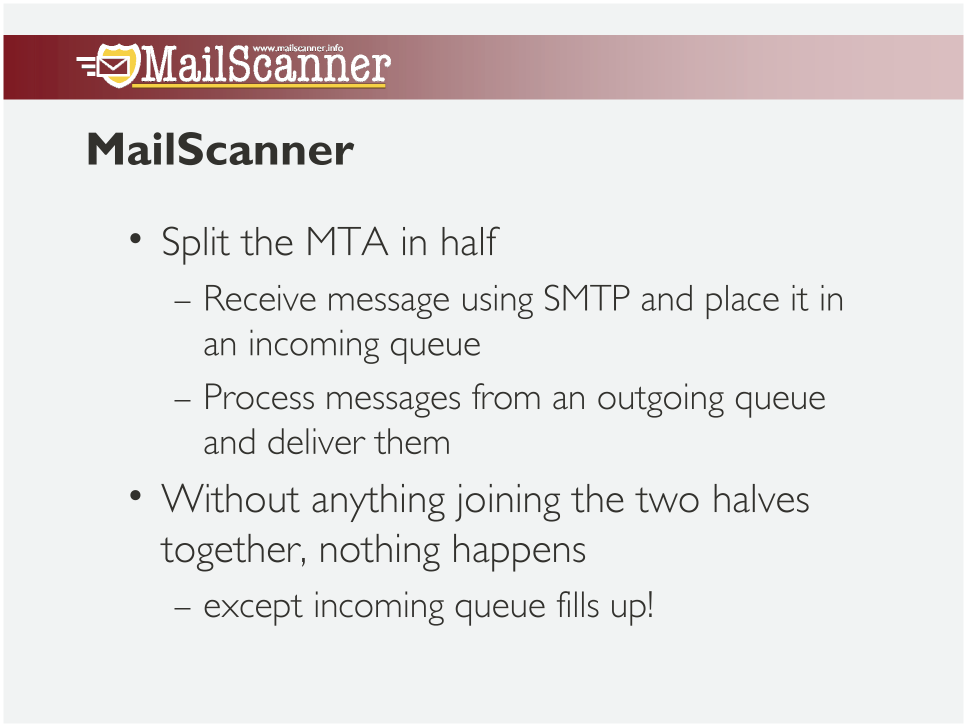


Without MailScanner, one instance of the MTA does everything.



MTA With MailScanner

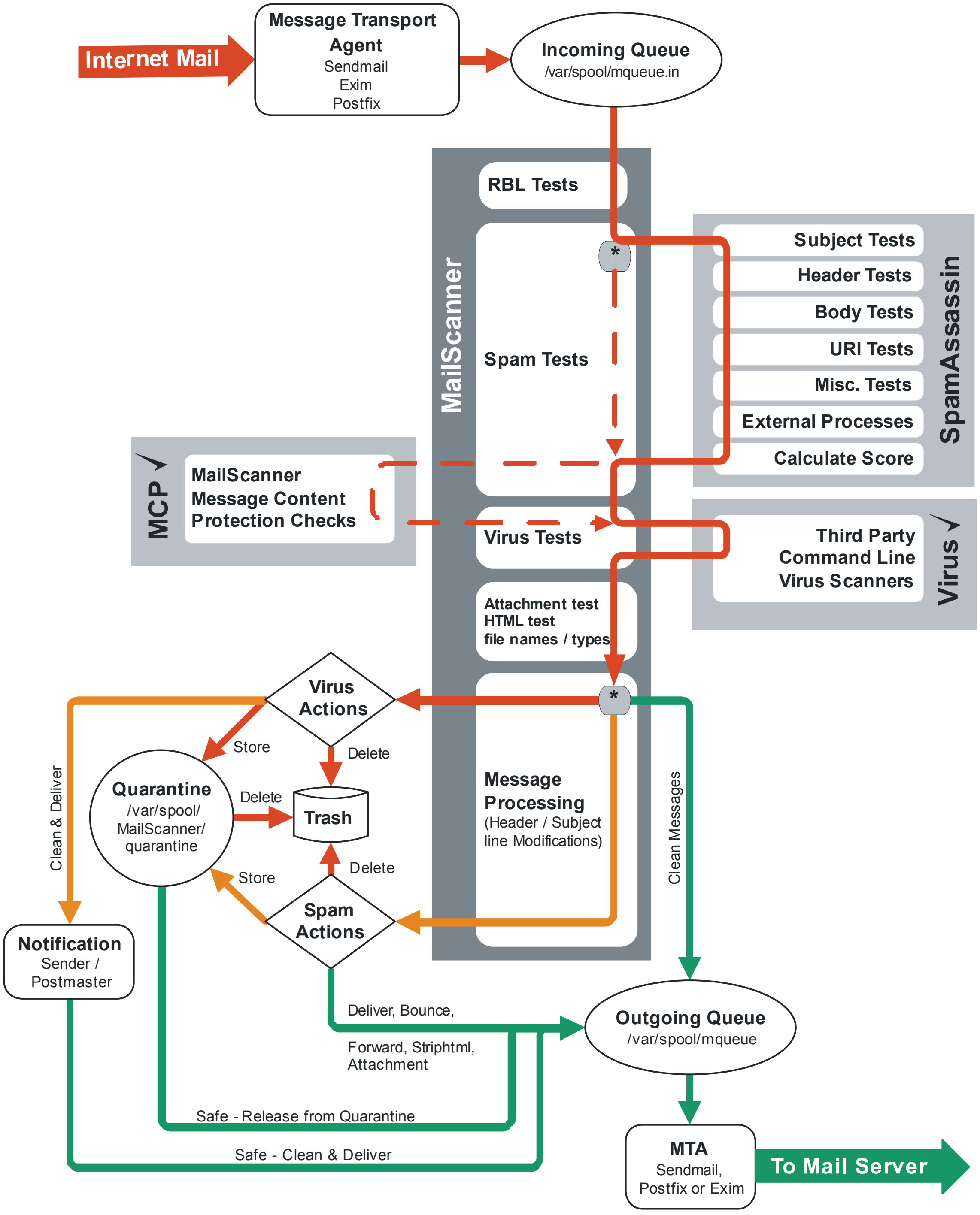
With MailScanner, there are 2 instances of the MTA, feeding from and delivering to different places.



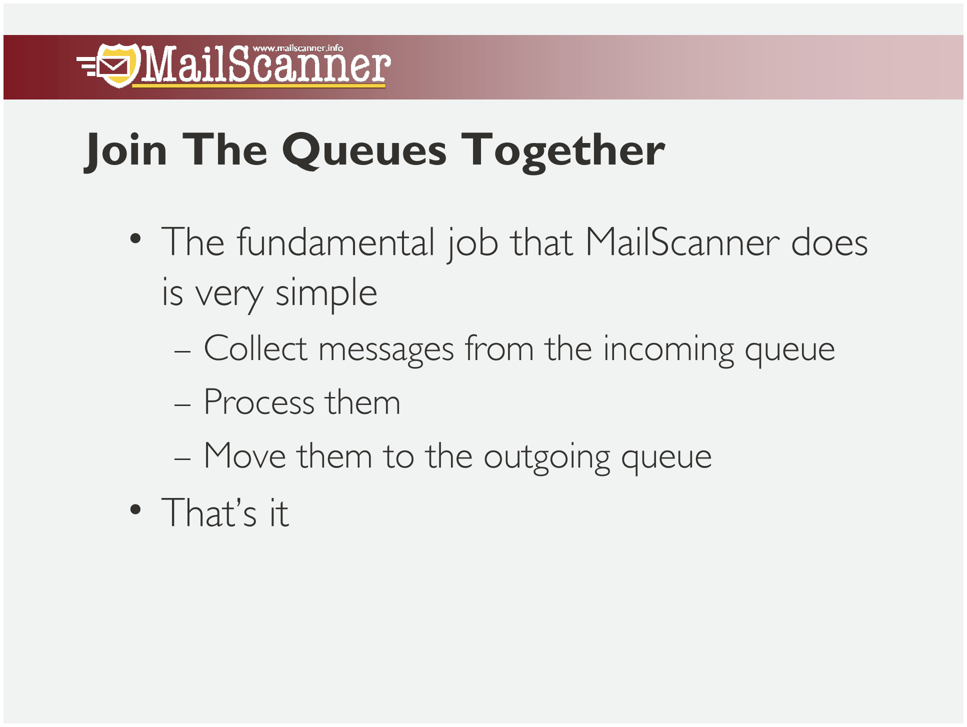
MailScanner

MailScanner

- Split the MTA in half
  - Receive message using SMTP and place it in an incoming queue
  - Process messages from an outgoing queue and deliver them
- Without anything joining the two halves together, nothing happens
  - except incoming queue fills up!



Internet Mail
Message Transport Agent
Sendmail
Exim
Postfix
Incoming Queue
/var/spool/mqueue.in
MailScanner
RBL Tests
Spam Tests
Virus Tests
Attachment test
HTML test
file names / types
Message Processing
(Header / Subject line Modifications)
SpamAssassin
Subject Tests
Header Tests
Body Tests
URI Tests
Misc. Tests
External Processes
Calculate Score
MCP
MailScanner Message Content Protection Checks
Virus
Third Party Command Line Virus Scanners
*
*
Virus Actions
Store
Delete
Quarantine
/var/spool/MailScanner/quarantine
Delete
Trash
Delete
Store
Spam Actions
Clean & Deliver
Clean Messages
Notification
Sender / Postmaster
Deliver, Bounce,
Forward, Striphtml, Attachment
Outgoing Queue
/var/spool/mqueue
Safe - Release from Quarantine
Safe - Clean & Deliver
MTA
Sendmail, Postfix or Exim
To Mail Server

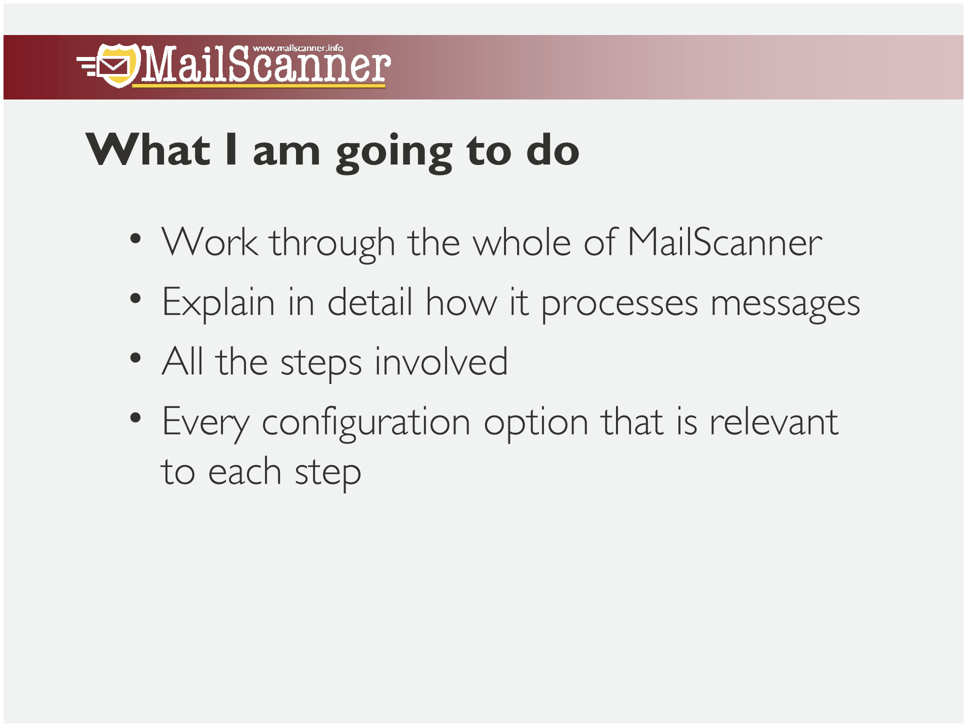


Join The Queues Together

- The fundamental job that MailScanner does is very simple
  - Collect messages from the incoming queue
  - Process them
  - Move them to the outgoing queue
- That's it

5

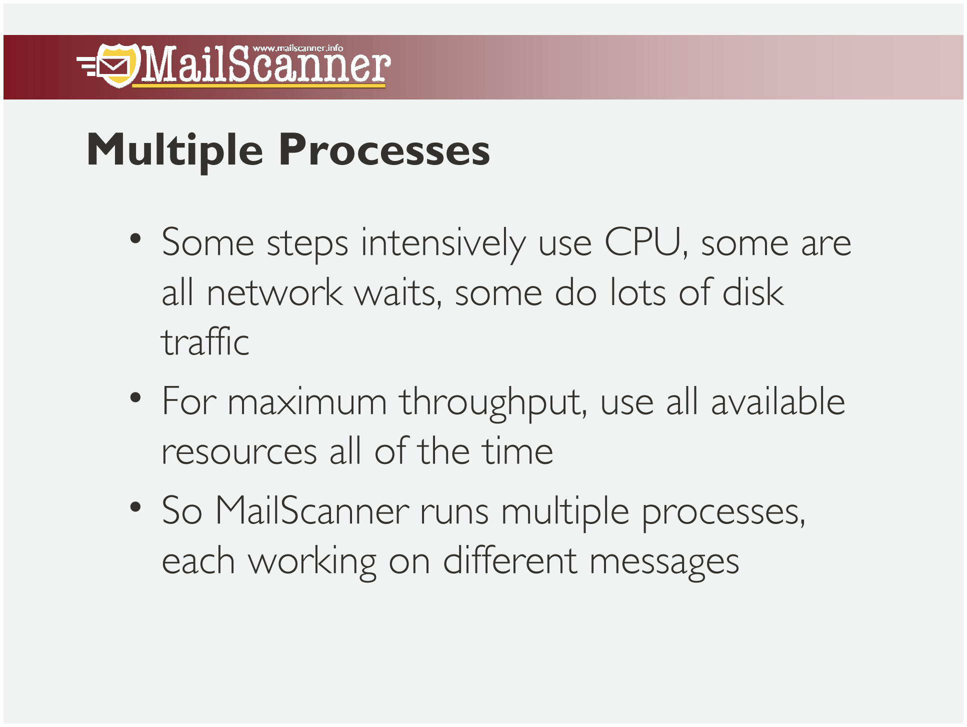
Detailed Analysis



www.mailscanner.info MailScanner

What I am going to do

- Work through the whole of MailScanner
- Explain in detail how it processes messages
- All the steps involved
- Every configuration option that is relevant to each step



Multiple Processes

- Some steps intensively use CPU, some are all network waits, some do lots of disk traffic
- For maximum throughput, use all available resources all of the time
- So MailScanner runs multiple processes, each working on different messages

On an average system, I recommend about 512MB to 1GB of RAM per CPU.

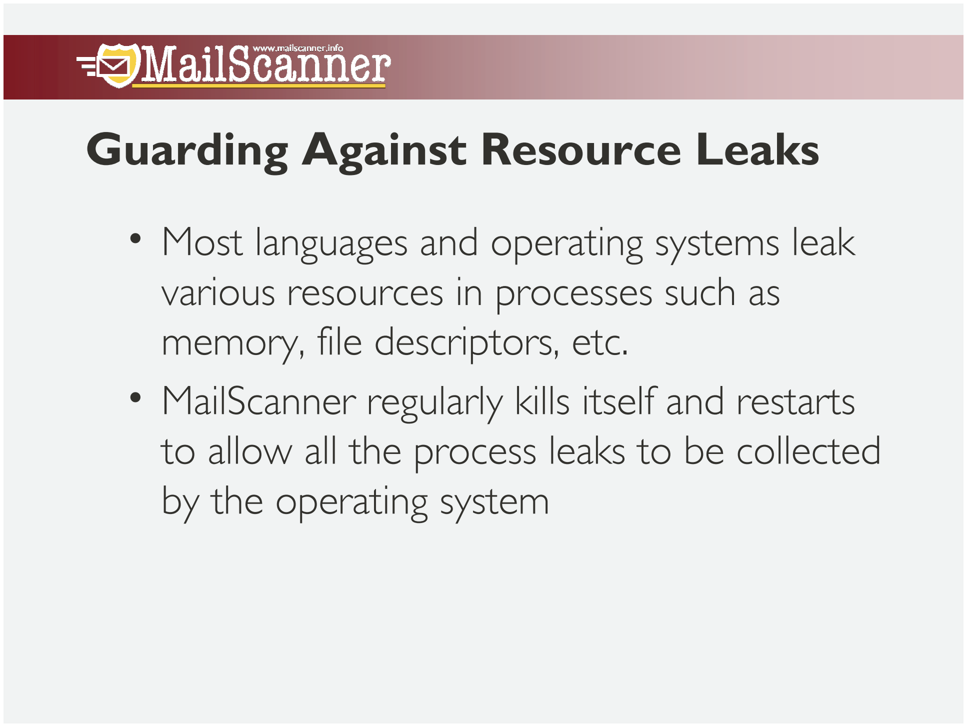
5 “worker” or “child” processes per CPU, 7 per hyper-threaded CPU.

The “parent” process is usually about 20MB, the “child” processes are about 40 to 50MB each. You also need plenty of RAM left to use for tmpfs for the temporary files and directories involved in the processing stages.

There is a problem with this called the “herd of elephants” where the processes tend to stay locked in synchronisation with each other, all demanding CPU at the same time, then all demanding disk I/O at the same time, for example. The processes are started up slowly, with a 10 second gap between each one, to ensure this problem does not arise.

`Max Children = 5`

How many MailScanner processes do you want to run at a time? There is no point increasing this figure if your MailScanner server is happily keeping up with your mail traffic. If you are running on a server with more than 1 CPU, or you have a high mail load (and/or slow DNS lookups) then you should see better performance if you increase this figure. If you are running on a small system with limited RAM, you should note that each child takes just over 20MB. As a rough guide, try 5 children per CPU. But read the notes above.

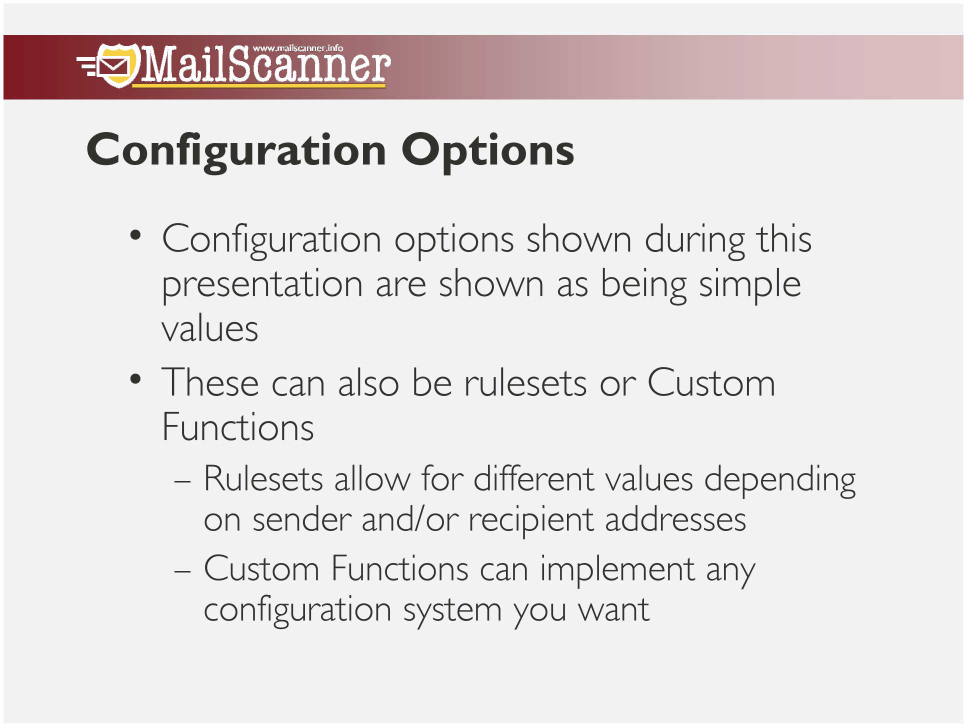

www.mailscanner.info
MailScanner

Guarding Against Resource Leaks

- Most languages and operating systems leak various resources in processes such as memory, file descriptors, etc.
- MailScanner regularly kills itself and restarts to allow all the process leaks to be collected by the operating system

```
Restart Every = 14400
```

To avoid resource leaks, re-start periodically



www.mailscanner.info

MailScanner

Configuration Options

- Configuration options shown during this presentation are shown as being simple values
- These can also be rulesets or Custom Functions
  - Rulesets allow for different values depending on sender and/or recipient addresses
  - Custom Functions can implement any configuration system you want

An example of a very simple ruleset might look like this:

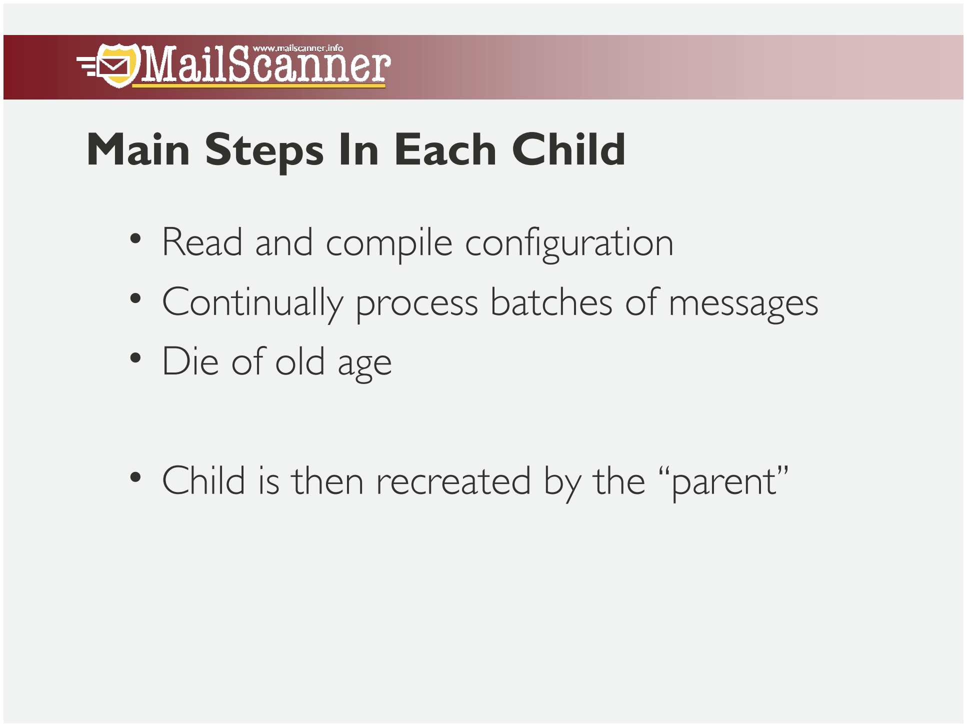
```
From: nasty@spammer.com     delete
To:         myusers@here.com deliver
FromOrTo:   /\d+@.*/                delete
```

Rulesets can be a lot more complicated than this, but they are covered in more detail later.

A simple example Custom Function could be used to deliver all a user's spam to a separate mailbox called <username>-spam. Then a user could retrieve their real mail through their normal POP account simon@ntl.com as usual, but when they have a fast network connection they could retrieve all their spam from a separate account "simon-spam@ntl.com". This incidentally has been done a few times already and is proving very popular with users.

5.1

Child Processes

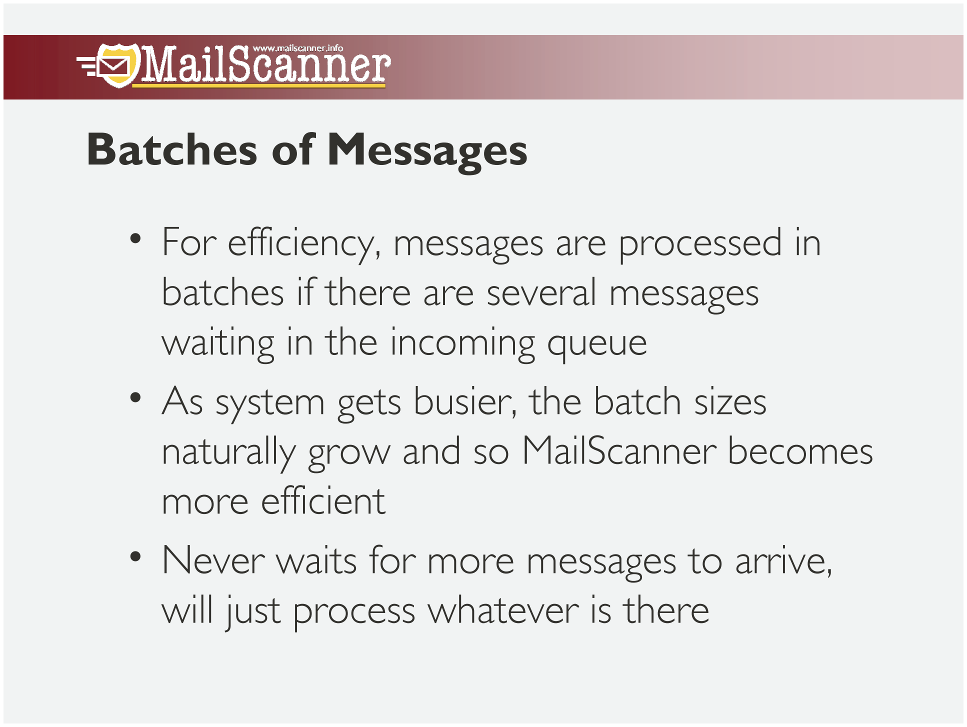


www.mailscanner.info MailScanner

Main Steps In Each Child

- Read and compile configuration
- Continually process batches of messages
- Die of old age

- Child is then recreated by the “parent”



Batches of Messages

- For efficiency, messages are processed in batches if there are several messages waiting in the incoming queue
- As system gets busier, the batch sizes naturally grow and so MailScanner becomes more efficient
- Never waits for more messages to arrive, will just process whatever is there

You may be using a ruleset to work out what email addresses to scan, and what not to scan. You would want to do this if you were charging for MailScanner on a per-domain or per-user basis. The “Unscanned” settings are useful then to allow a lot more messages through at a time when not many are being scanned anyway.

The “Unsafe” settings cover all mail that is being scanned in any way.

Setting the batch size very large results in greater RAM requirements and the resulting system can appear not as responsive as with smaller batches, as the processing cycle for each batch takes a lot longer. The overall throughput will not change much, but the apparent latency can be considerably bigger when the server is very heavily loaded.

`MTA = sendmail`

Set whether to use postfix, sendmail, exim, qmail or zmailer. If you are using postfix, then see the "SpamAssassin User State Dir" setting near the end of this file

`Queue Scan Interval = 5`

How often (in seconds) should each process check the incoming mail queue for new messages? If you have a quiet mail server, you might want to increase this value so it causes less load on your server, at the cost of slightly increasing the time taken for an average message to be processed.

`Incoming Queue Dir = /var/spool/mqueue.in`

Set location of incoming mail queue. This can be any one of

1. A directory name
   Example: /var/spool/mqueue.in

2. A wildcard giving directory names
   Example: /var/spool/mqueue.in/*

3. The name of a file containing a list of directory names, which can in turn contain wildcards.
   Example: /etc/MailScanner/mqueue.in.list.conf

   Note: In the case of sendmail, the incoming queue directories can each be split up to reduce the number of files in each sub-directory. This is very useful on systems with large queues which are also using a filesystem which does not handle very large directories well.

   Each incoming mail queue directory is automatically searched for the presence of "df" and "qf" sub-directories. If these are found, it will assume that all the "qf" files are in the "qf" sub-directory, "df" files in the "df" sub-directory and so on. Nothing special should be passed to MailScanner to ask it to use these sub-directories, it will all happen automatically.

   Sendmail can easily be configured to write the queue directories in this way, see the www.sendmail.org website or the sendmail book for more information about this.

`Outgoing Queue Dir = /var/spool/mqueue`

Set location of outgoing mail queue. This can also be the filename of a ruleset.

`Max Unscanned Bytes Per Scan = 100000000`
`Max Unsafe Bytes Per Scan = 50000000`

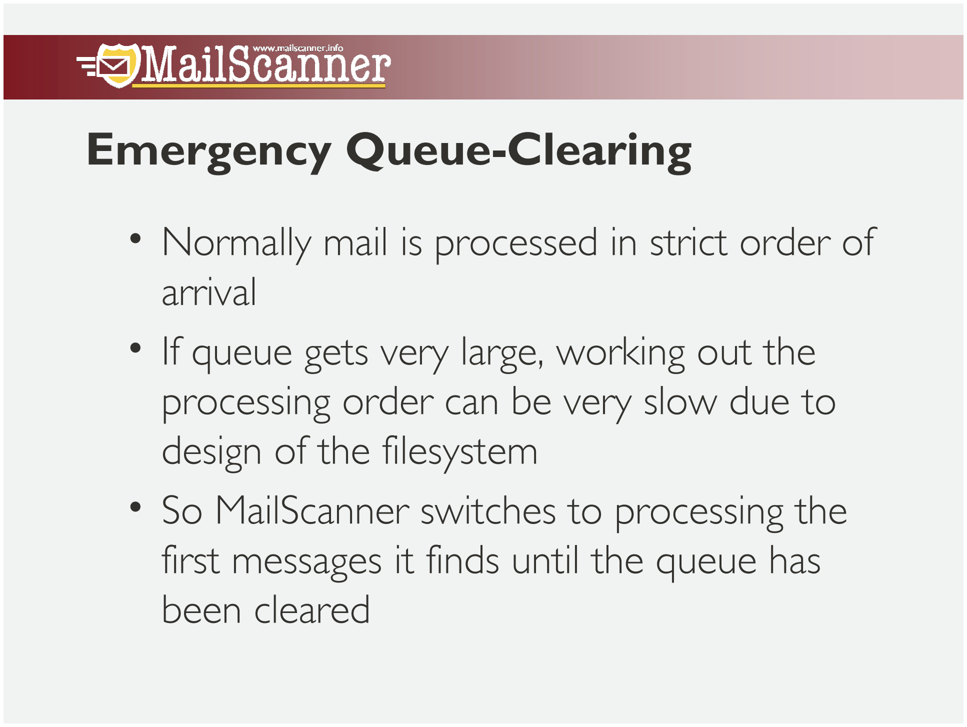
`Max Unscanned Messages Per Scan = 30`

`Max Unsafe Messages Per Scan = 30`

In every batch of virus-scanning, limit the maximum

- number of unscanned messages to deliver

- number of potentially infected messages to unpack and scan
- total size of unscanned messages to deliver
- total size of potentially infected messages to unpack and scan


www.mailscanner.info
MailScanner

Emergency Queue-Clearing

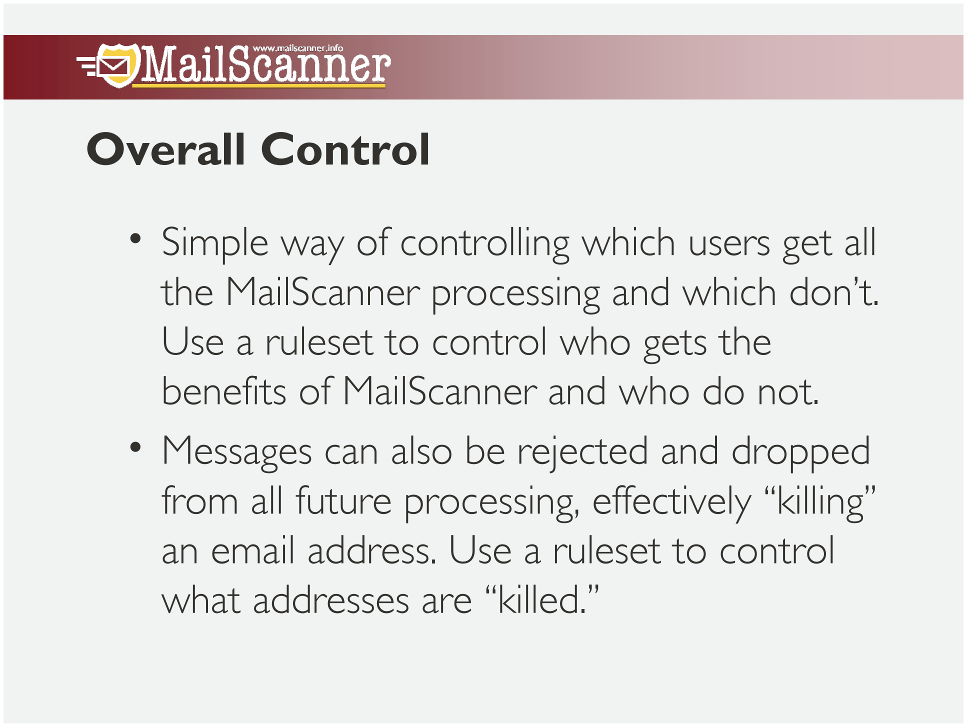
- Normally mail is processed in strict order of arrival
- If queue gets very large, working out the processing order can be very slow due to design of the filesystem
- So MailScanner switches to processing the first messages it finds until the queue has been cleared

Most filesystems implement directories as simple lists. To open the last file in the directory the operating system has to scan all the way through the list to find it. So for a directory containing `n` files, it has to scan through an average of `n/2` directory entries just to find the file before it can open it or check the datestamp.

By abandoning the strict delivery order, the first files in the directory are delivered first, and it will clear the backlog a lot faster as all these directory searches are skipped. Once the backlog has cleared, MailScanner automatically reverts to delivering mail in strict date order.

`Max Normal Queue Size = 800`

If more messages are found in the queue than this, then switch to an "accelerated" mode of processing messages. This will cause it to stop scanning messages in strict date order, but in the order it finds them in the queue. If your queue is bigger than this size a lot of the time, then some messages could be greatly delayed. So treat this option as "in emergency only".



Overall Control

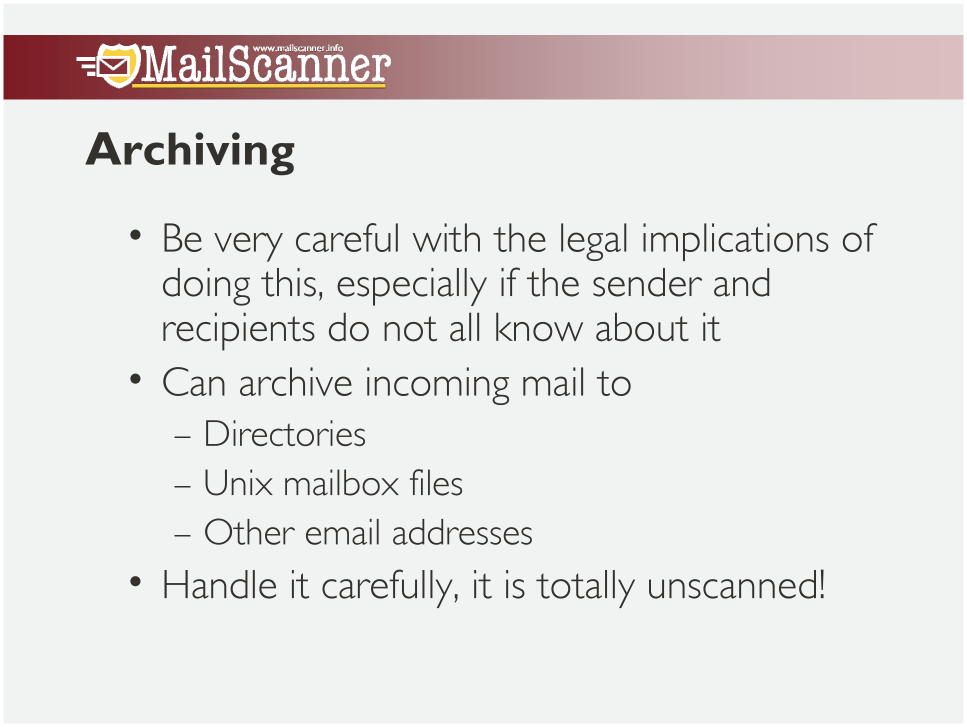
- Simple way of controlling which users get all the MailScanner processing and which don't. Use a ruleset to control who gets the benefits of MailScanner and who do not.
- Messages can also be rejected and dropped from all future processing, effectively "killing" an email address. Use a ruleset to control what addresses are "killed."

```
Scan Messages = yes
```

This setting is used to completely enable or disable all the processing done by MailScanner, except for the "Archive Mail" setting described in the next section. It is intended for use with a ruleset, so that your customers who either do not want their mail processed in any way, or who have not paid you for the service, can have their mail delivered without any processing done to it. No MailScanner headers or modifications of any sort will be applied to the messages, and it is impossible to see from the contents of the message that they have passed through MailScanner. The "Archive Mail" option is still used so that you can take copies of their mail, possibly without their knowledge if your operation or organisation requires it. This can be the filename of a ruleset, and this is how this setting would normally be used unless it is just set to "yes".

```
Reject Message = no
Rejection Report = %rules-dir%/rejection.report.txt
```

This message, in conjunction with a ruleset, will effectively disable an email address, sending a rejection message back the sender. The file sent to the original sender of the message is found in by looking in the "Rejection Report" setting.



Archiving

- Be very careful with the legal implications of doing this, especially if the sender and recipients do not all know about it
- Can archive incoming mail to
  - Directories
  - Unix mailbox files
  - Other email addresses
- Handle it carefully, it is totally unscanned!

Underneath the directory name you give it, it will create a dated directory and put the messages in there.

If you want to archive to a file rather than a directory, the file must exist as otherwise MailScanner does not know what you mean.

You can archive to multiple places and addresses.

The contents of the archive is a totally untouched copy of the original message, it has no extra headers, it has not been virus scanned or anything. So be very careful and do not blindly deliver the contents of the archive to anyone without ensuring they know exactly what they are getting.

In the EU there are privacy laws that may well make this sort of archiving illegal in some situations.

`Archive Mail =`

Space-separated list of any combination of

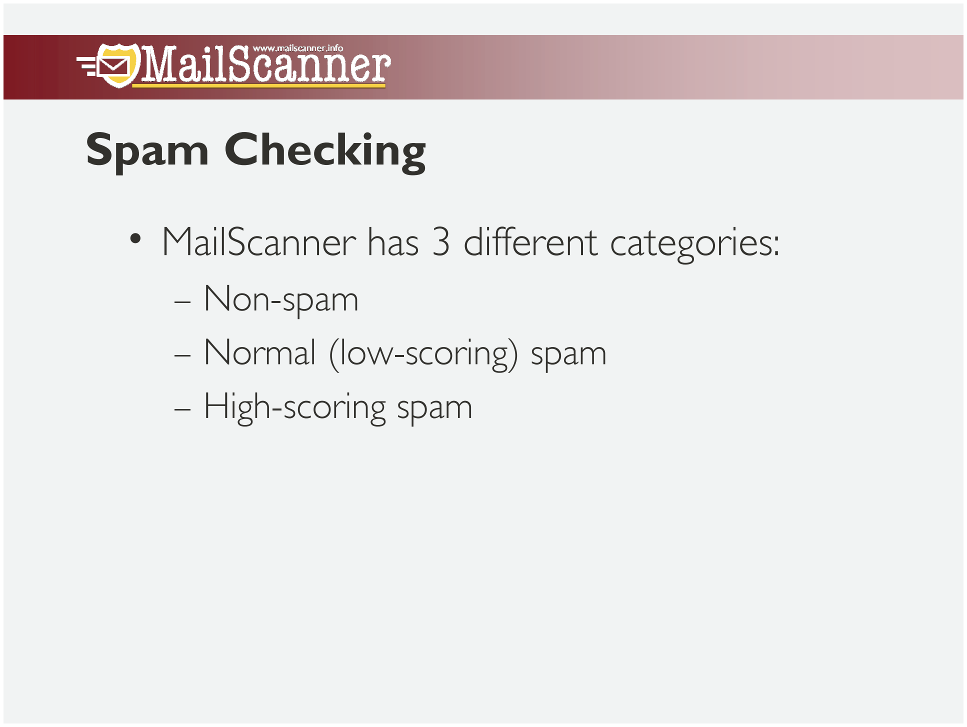
1. email addresses to which mail should be forwarded,

2. directory names where you want mail to be stored,

3. file names (they must already exist!) to which mail will be appended in "mbox" format suitable for most Unix mail systems.
   If you give this option a ruleset, you can control exactly whose mail is archived or forwarded. If you do this, beware of the legal implications as this could be deemed to be illegal interception unless the police have asked you to do this.

Any directory or file name used here may contain the magic string “_DATE”. This will be replaced by a number representing the date in the form YYYYMMDD, i.e. 4 digits representing the year, 2 digits representing the number of the month and 2 digits representing the date within that month. This format is used as it will be sorted in date order, should the entries in a list be sorted in alphabetic or numeric order. This can be used to separate the archives by date, so that previous days’ archives can be moved or deleted without altering the contents of the present day’s archive.

5.2
Spam Checking


Spam Checking

- MailScanner has 3 different categories:
  - Non-spam
  - Normal (low-scoring) spam
  - High-scoring spam

`First Check = spam`

Do you want to do MCP (Message Content Protection) checks or spam checks first? MCP is a content-blocking system that can be used to protect confidential information leaking from your institution. It can easily be configured to look for particular keywords or keyphrases and block delivery of any message containing them.For more information about MCP, please see `www.sng.ecs.soton.ac.uk/mailscanner/install/mcp/`

Checking for spam is quite network-intensive. SpamAssassin is also CPU-intensive. Fast DNS response is a necessity, consider running a local caching DNS server just for MailScanner.

Spam checking is done before virus checking. So if you choose to delete mail which is definitely spam, you can greatly reduce your system load as none of the spam will have to be analysed or scanned by anything else. You might want to leave normal low-scoring spam to be delivered, but get rid of all high-scoring spam.

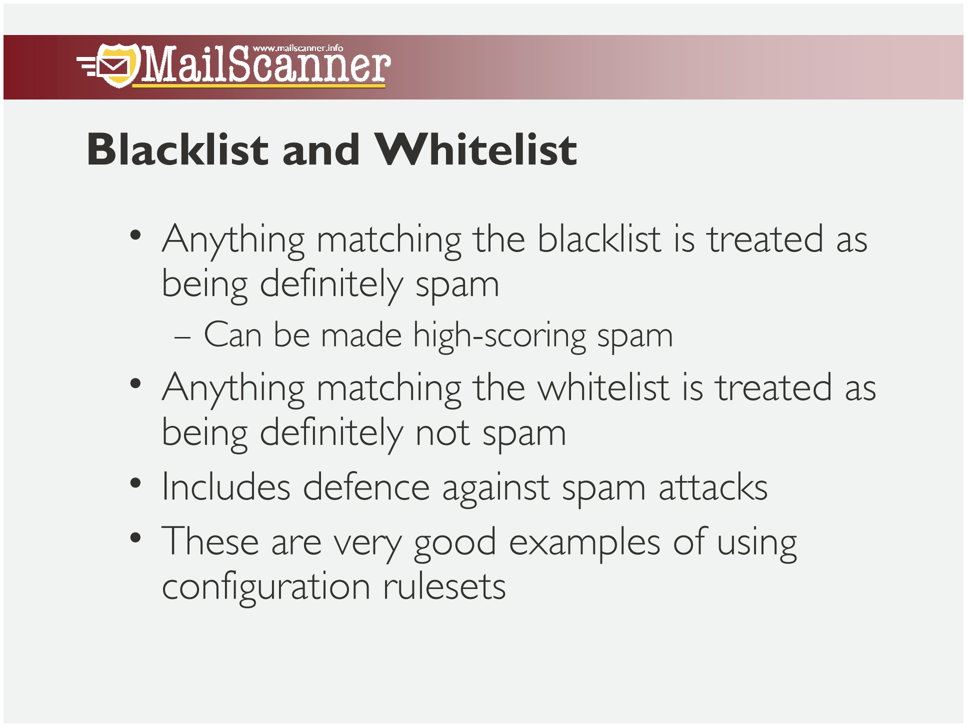
`Spam Checks = yes`

Do you want to check messages to see if they are spam? Note: If you switch this off then *no* spam checks will be done at all. This includes both MailScanner's own checks and

SpamAssassin. If you want to just disable the "Spam List" feature then set "Spam List =" (i.e. an empty list) in the setting below. This can also be the filename of a ruleset.

`Max Spam Check Size = 150000`

Messages larger than this number of bytes are not tested for spam at all. Huge messages take a long time to scan and are very unlikely to be spam, as large messages cost a spammer more to send than smaller messages. Very few spam messages are more than 100kbytes. This can also be the filename of a ruleset.



www.mailscanner.info

Blacklist and Whitelist

- Anything matching the blacklist is treated as being definitely spam
  - Can be made high-scoring spam
- Anything matching the whitelist is treated as being definitely not spam
- Includes defence against spam attacks
- These are very good examples of using configuration rulesets

The whitelist takes precedence over the blacklist. A message whose address appears on both lists will be treated as if it was whitelisted, the blacklist tag will be ignored. To put it another way, it errs on the side of caution and prefers to treat the message as non-spam.

If you are deleting high-scoring spam, the “Definite Spam Is High Scoring” control is very useful as that will cause the deletion of any blacklisted message.

`Is Definitely Spam = no`

Spam Blacklist: Make this point to a ruleset, and anything in that ruleset whose value is "yes" will *always* be marked as spam. This can also be the filename of a ruleset.

`Is Definitely Not Spam = %rules-dir%/spam.whitelist.rules`

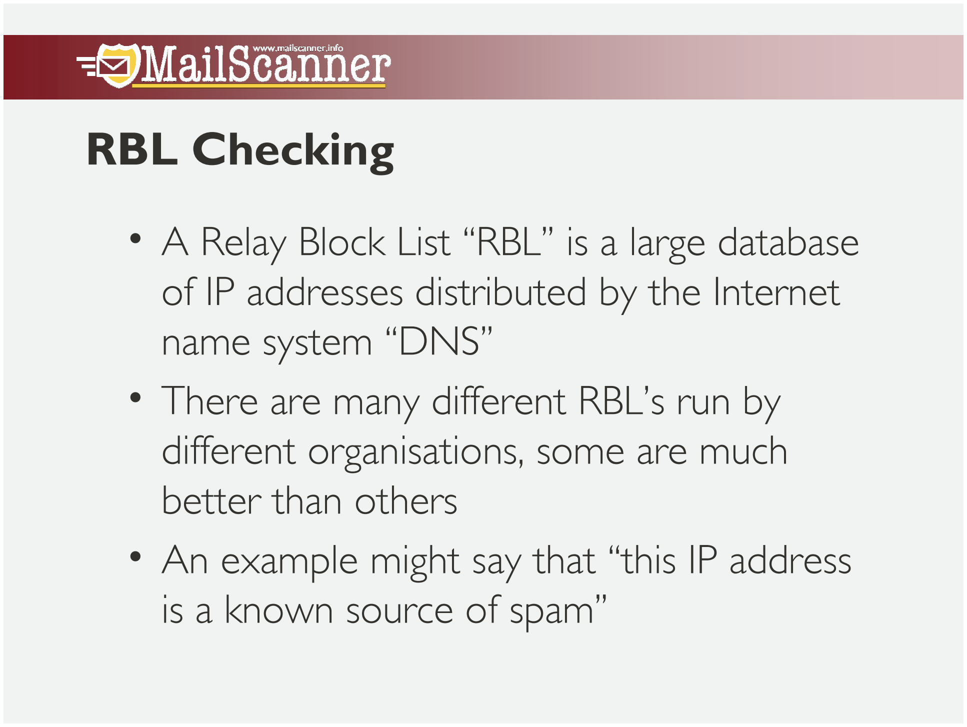
Spam Whitelist: Make this point to a ruleset, and anything in that ruleset whose value is "yes" will *never* be marked as spam. This can also be the filename of a ruleset.

`Definite Spam Is High Scoring`

Setting this to yes means that spam found in the blacklist is treated as "High Scoring Spam" in the "Spam Actions" section below. Setting it to no means that it will be treated as "normal" spam. This can also be the filename of a ruleset.

`Ignore Spam Whitelist If Recipients Exceed = 20`

Spammers have learnt that they can get their message through by sending a message with lots of recipients, one of which chooses to whitelist everything coming to them, including the spammer. So if a message arrives with more than this number of recipients, ignore the "Is Definitely Not Spam" whitelist.



RBL Checking

- A Relay Block List "RBL" is a large database of IP addresses distributed by the Internet name system "DNS"
- There are many different RBL's run by different organisations, some are much better than others
- An example might say that "this IP address is a known source of spam"

The names of the spam lists used in the "Spam List" setting are all defined in the spam.lists.conf configuration file. This file simply lists the nicknames that should be used in the "Spam List" setting alongside the real DNS domain name containing the RBL. Please do not forget to terminate every domain name with a ".".

In my opinion the best are those run by Spamhaus. Other good ones are run by ORDB and MAPS. This is entirely my personal opinion and just reflects the ones I tend to use on systems I install.

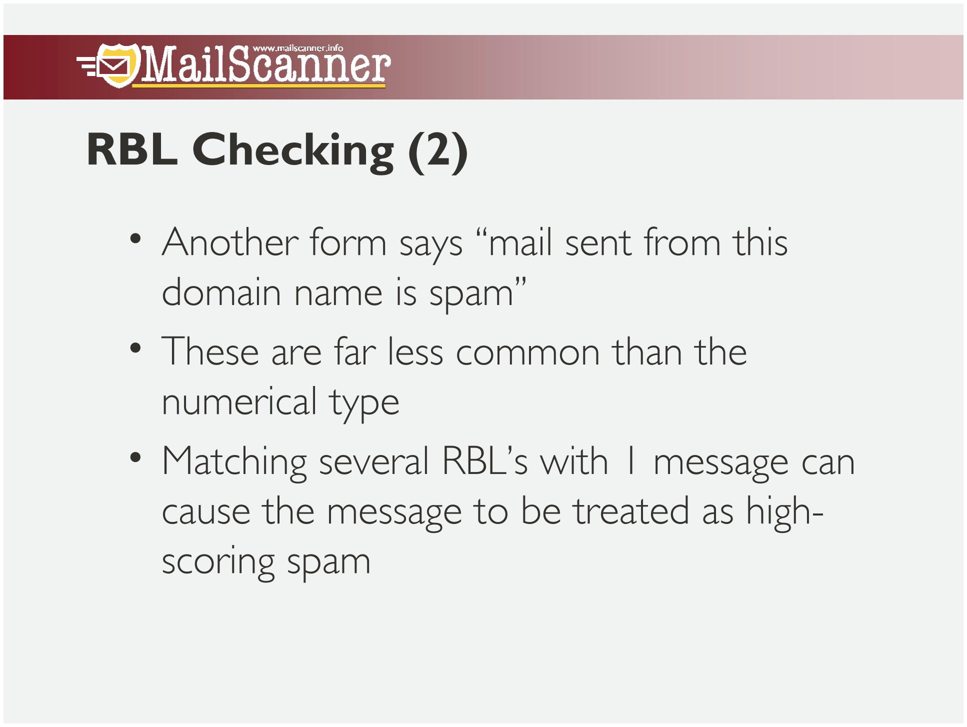
There is a very good tool for checking IP addresses against a large collection of RBL's at www.dnsstuff.com.

`Spam List Definitions = %etc-dir%/spam.lists.conf`

This is the name of the file that translates the names of the "Spam List" values to the real DNS names of the spam blacklists.

`Spam List = ORDB-RBL SBL+XBL`

This is the list of spam blacklists (RBLs) which you are using. See the "Spam List Definitions" file for more information about what you can put here. This can also be the filename of a ruleset.


MailScanner www.mailscanner.info

RBL Checking (2)

- Another form says "mail sent from this domain name is spam"
- These are far less common than the numerical type
- Matching several RBL's with 1 message can cause the message to be treated as high-scoring spam

I do not advise using more than 2 or 3 RBL's with the "Spam List" setting, as the cost in speed may be too great.

`Spam Domain List =`

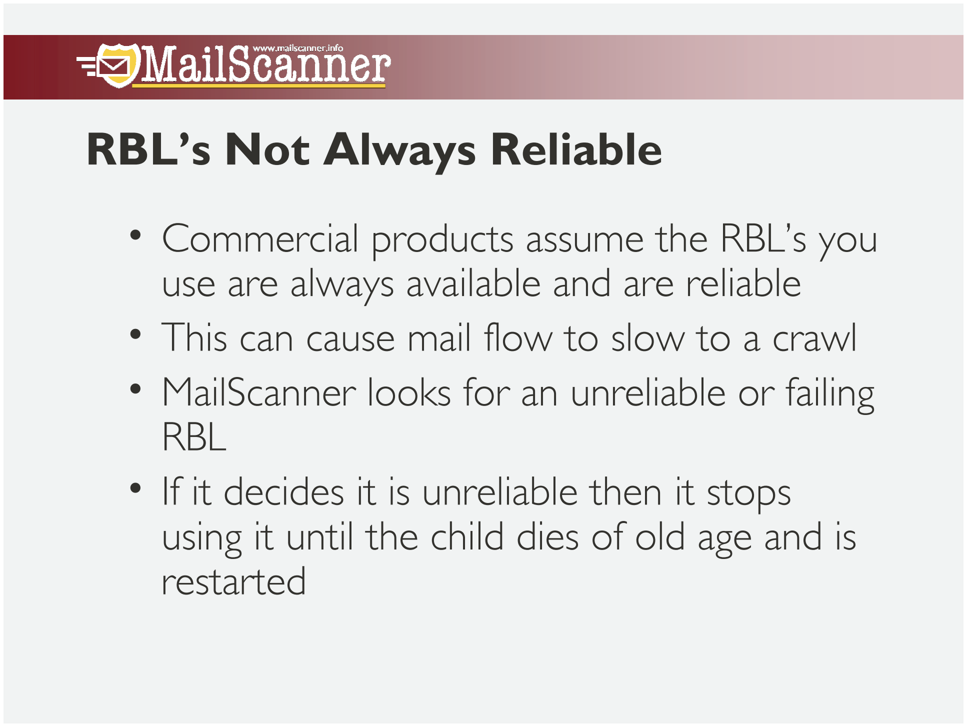
This is the list of spam domain blacklists which you are using (such as the "rfc-ignorant" domains). See the "Spam List Definitions" file for more information about what you can put here. This can also be the filename of a ruleset.

`Spam Lists To Be Spam = 1`

In order to be treated as spam, the sender of the message must appear in at least this number of "Spam Lists" to be treated as spam. Previously this setting was fixed at 1, so that a message appearing in a single "Spam List" would be treated as spam.

`Spam Lists To Reach High Score = 3`

If a message appears in at least this number of "Spam Lists" (as defined above), then the message will be treated as "High Scoring Spam" and so the "High Scoring Spam Actions" will happen. You probably want to set this to 2 if you are actually using this feature. 5 is high enough that it will never happen unless you use lots of "Spam Lists". This can also be the filename of a ruleset.


RBL’s Not Always Reliable

- Commercial products assume the RBL’s you use are always available and are reliable
- This can cause mail flow to slow to a crawl
- MailScanner looks for an unreliable or failing RBL
- If it decides it is unreliable then it stops using it until the child dies of old age and is restarted

RBL’s are all distributed using DNS zones. As such, they are not 100% reliable. Normally a DNS lookup failure results in the temporary failure of whatever system is using it. As MailScanner does not want to queue up mail while waiting for the RBL to re-appear, it detects failing RBL’s and starts ignoring them.

The two timeout “Max” settings allow for an RBL to be detected as failing even if it occasionally succeeds. One problem with an overloaded RBL is that it might start failing to respond 50% of the time. With a simple “max timeouts” setting, you are reliant on getting, say, 10 failures in a row. If the lookups are succeeding half the time, these consecutive failures may never occur. However your mail throughput will be severely slowed down as it is having to wait for “Spam List Timeout” seconds half the time for each message.

Using the approach taken by MailScanner will cause a failing RBL to be detected even if it responds occasionally.

Once an RBL has been marked as “dead”, it will not be tested by that child again until it naturally dies of old age and is restarted by the parent “controller” process. At that point all the counters are reset to zero, resulting in renewed attempts to contact the RBL which by now might well be fixed and working reliably.

`Spam List Timeout = 10`

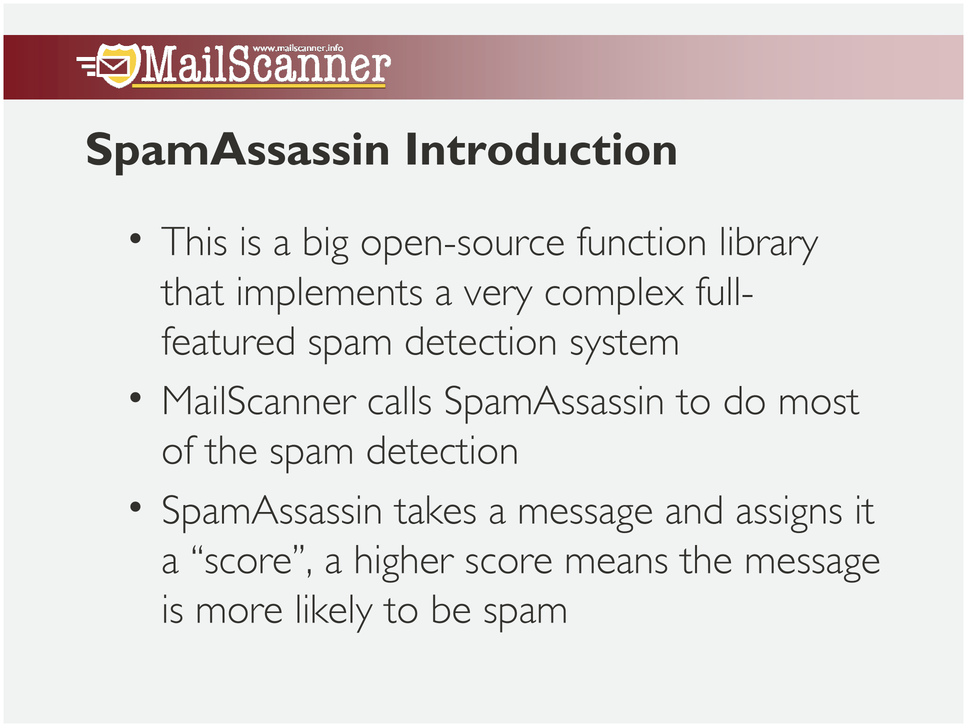
If an individual "Spam List" or "Spam Domain List" check takes longer that this (in seconds), the check is abandoned and the timeout noted.

`Max Spam List Timeouts = 7`

The maximum number of timeouts caused by any individual "Spam List" or "Spam Domain List" before it is marked as "unavailable". Once marked, the list will be ignored until the next automatic re-start (see "Restart Every" for the longest time it will wait). This can also be the filename of a ruleset.

`Spam List Timeouts History = 10`

The total number of Spam List attempts during which "Max Spam List Timeouts" will cause the spam list fo be marked as "unavailable". See the previous comment for more information. The default values of 5 and 10 mean that 5 timeouts in any sequence of 10 attempts will cause the list to be marked as "unavailable" until the next periodic restart (see "Restart Every").



SpamAssassin Introduction

- This is a big open-source function library that implements a very complex full-featured spam detection system
- MailScanner calls SpamAssassin to do most of the spam detection
- SpamAssassin takes a message and assigns it a “score”, a higher score means the message is more likely to be spam

SpamAssassin is arguably the best anti-spam tool available. It has an excellent reputation around the world and is under continuous development by a large team of developers. Like MailScanner, it is totally free to use and to incorporate into other systems.

There are 3 ways of using SpamAssassin:

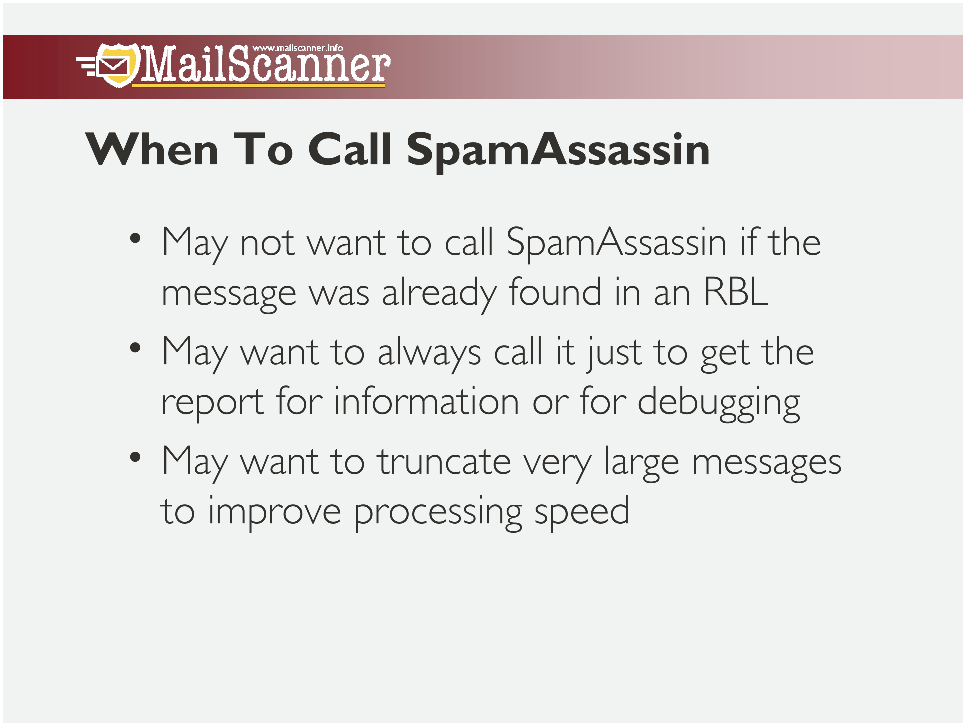
1. A command-line script called “spamassassin”.
   This is very simple to use but very slow as the entire library has to be reinitialised for every message. Useful for testing and for easy deployment on small systems.

2. A client/daemon combination of “spamc” and “spamd”.
   Most non-MailScanner systems use this. Messages are passed to the spamc client which pipes them into the spamd daemon for testing. The daemon then rewrites the message with various assorted extra headers and spam reports and passes the message back to spamc to save into a file. Faster, but involves a lot of I/O passing the message to and fro, and is reliant on the spamd daemon working reliably all the time.

3. Calling SpamAssassin’s Perl API directly.
   This is how MailScanner works. It is the fastest method as SpamAssassin is written in Perl and no extra processes or programs are involved at all. A internal copy of the message is piped straight into the spam analyser very quickly, and the resulting spam score and spam reports are read by querying the library directly. The internal copy of the message is just thrown away after analysis as there is no need to feed it back to MailScanner.

`Use SpamAssassin = yes`

Do you want to find spam using the "SpamAssassin" package? This can also be the filename of a ruleset. If SpamAssassin is not installed, this request will be ignored, and will not cause MailScanner to stop. It can therefore be safely used as the default value.



When To Call SpamAssassin

- May not want to call SpamAssassin if the message was already found in an RBL
- May want to always call it just to get the report for information or for debugging
- May want to truncate very large messages to improve processing speed

If you choose to always include the SpamAssassin report, SpamAssassin will always be called for every message regardless of other settings related to SpamAssassin.

Most spam is quite short, and longer messages can safely be truncated just for the SpamAssassin analysis. It is usually obvious to the engine that a message is spam without having to see all of a large message, the first 20 or 30Kbytes of the message is quite enough. This improves the speed of the spam analysis without any noticeable effect on the detection rate.

`Check SpamAssassin If On Spam List = yes`

If the message sender is on any of the Spam Lists, do you still want to do the SpamAssassin checks? Setting this to "no" will reduce the load on your server, but will stop the High Scoring Spam Actions from ever happening. This can also be the filename of a ruleset.

`Always Include SpamAssassin Report = no`

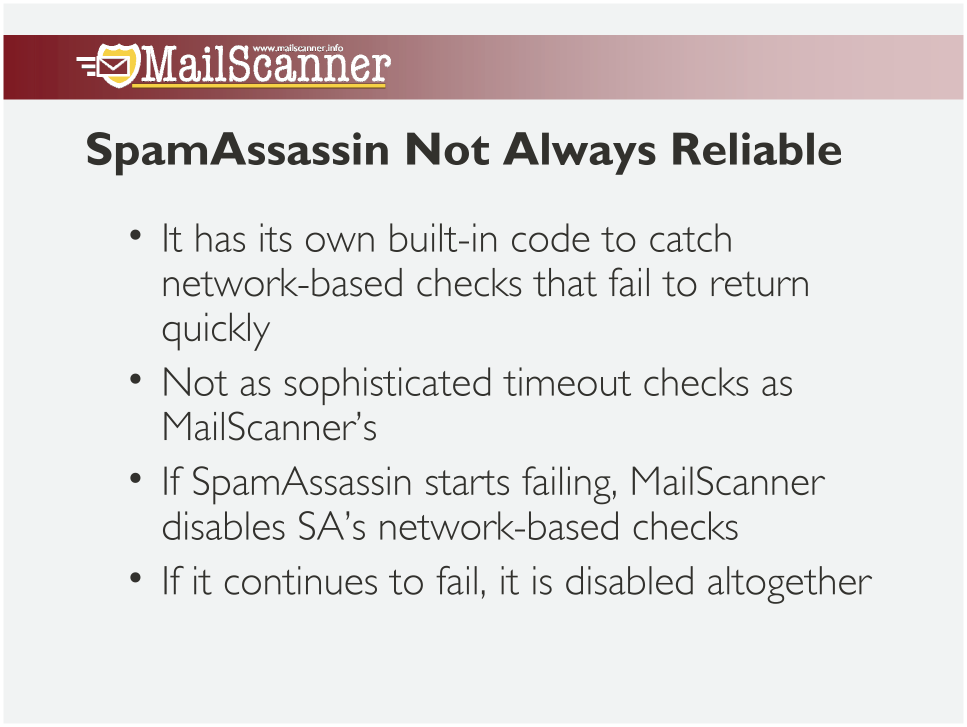
Do you want to always include the Spam Report in the SpamCheck header, even if the message wasn't spam? This can also be the filename of a ruleset.

`Max SpamAssassin Size = 30000`

SpamAssassin is not very fast when scanning huge messages, so messages bigger than this value will be truncated to this length for SpamAssassin testing. The original message will not be affected by this. This value is a good compromise as few spam messages are bigger than this.

There are three forms of this setting:

1. `<number>`
   The message is truncated at `<number>` bytes through the message.
2. `<number>` **`backtrack`**
   Go to `<number>` bytes through the message, then reverse back towards the start of the message until the start of the current attachment (if any) is found.
3. `<number n>` **`continue`** `<number m>`
   Go to `<number n>` bytes through the message, then continue forwards through the message, stopping at the end of the current attachment or after a further `<number m>` bytes, whichever occurs earlier.



SpamAssassin Not Always Reliable

- It has its own built-in code to catch network-based checks that fail to return quickly
- Not as sophisticated timeout checks as MailScanner's
- If SpamAssassin starts failing, MailScanner disables SA's network-based checks
- If it continues to fail, it is disabled altogether

These timeouts are very similar to the RBL “Spam List” timeout settings. However, instead of disabling SpamAssassin completely if it is failing, first all the network-based checks are disabled as these are the most likely culprits. Only if it continues to fail is it disabled completely. I have never seen it completely fail in 4 years.

Like disabling MailScanner's own RBL checks, the “dead” status is reset when the child process dies of old age and is re-spawned by the parent process.

`SpamAssassin Timeout = 40`

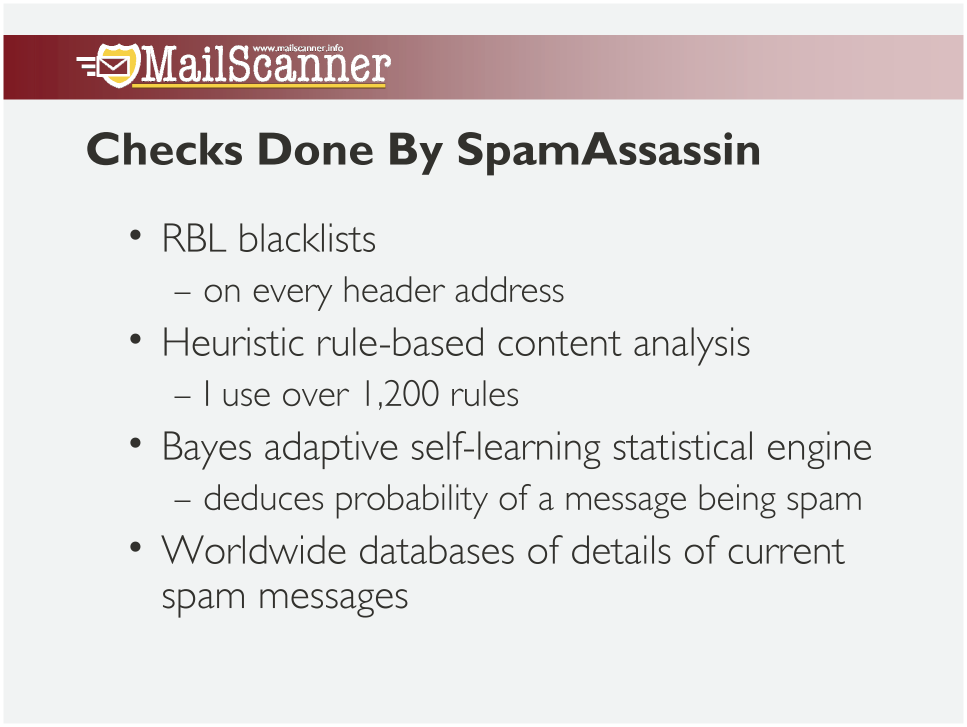
If SpamAssassin takes longer than this (in seconds), the check is abandoned and the timeout noted. This large figure takes into account the internal 30 second default timeouts built into many SpamAssassin features, thus still providing a useful score even if some internal SpamAssassin checks fail.

`Max SpamAssassin Timeouts = 20`

If SpamAssassin times out more times in a row than this, then it will be marked as "unavailable" until MailScanner next re-starts itself. This means that remote network failures causing SpamAssassin trouble will not mean your mail stops flowing.

`SpamAssassin Timeouts History = 30`

The total number of SpamAssassin attempts during which "Max SpamAssassin Timeouts" will cause SpamAssassin to be marked as "unavailable". See the previous comment for more information. The default values of 10 and 20 mean that 10 timeouts in any sequence of 20 attempts will trigger the behaviour described above, until the next periodic restart (see "Restart Every").


MailScanner www.mailscanner.info

Checks Done By SpamAssassin

- RBL blacklists
  - on every header address
- Heuristic rule-based content analysis
  - I use over 1,200 rules
- Bayes adaptive self-learning statistical engine
  - deduces probability of a message being spam
- Worldwide databases of details of current spam messages

One of the major reasons for SpamAssassin's success is that it is a "Swiss Army knife" of spam detection systems. It not only does its own RBL checks on every header in the message, but it does them in parallel for greater speed as there are many lists looked up by it.

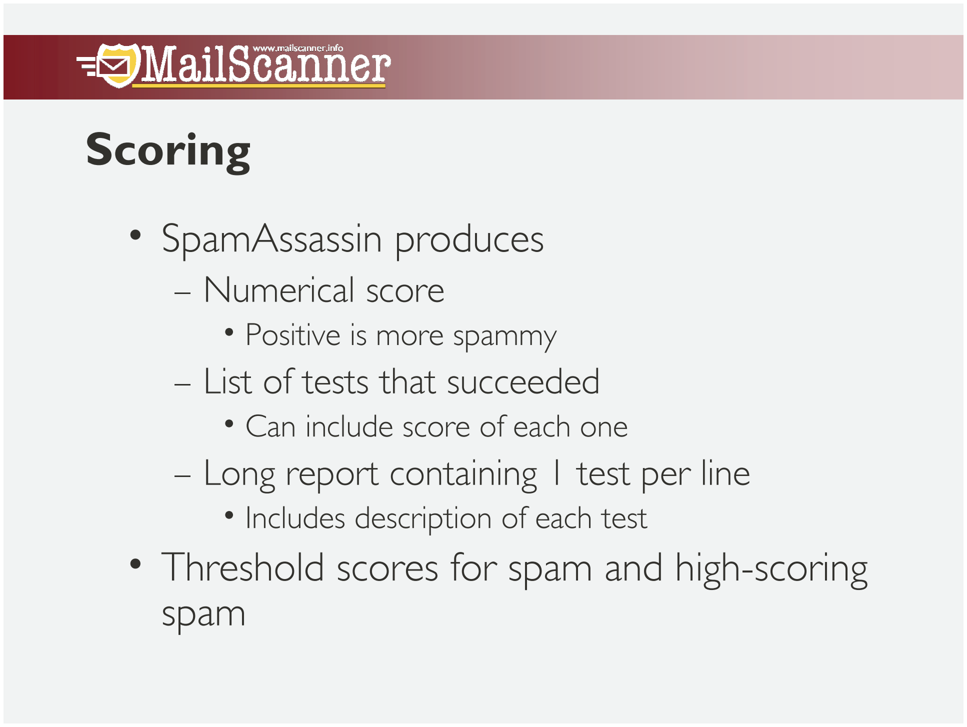
There are a wide variety of extra sets of heuristic rules available. Good places to start looking are

- www.rulesemporium.com
- wiki.apache.org/spamassassin/CustomRulesets

If you are considering writing your own rules or would like to learn more about SpamAssassin, there is also a lot of very useful documentation at wiki.apache.org/spamassassin/

The Bayes database is most useful on smaller sites, and it is not well suited to sites with millions of users. It can also be poisoned by spammers, which is why recent spam messages often contain large pieces of poetry or half of "Jane Eyre". These poisoning attempts are only partially successful, but many people now reset their Bayes database once a month or so. The Bayes database was a brilliant idea, and it worked very well when it was new and the spammers hadn't had a chance to work out how to attack it. Unfortunately the spammers have now caught up.

The databases of current messages contain checksums of messages. The 3 used by SpamAssassin are DCC, Razor and Pyzor, which are different systems but all fundamentally do a similar job. The checksums are built in such a way that the system can tell if two messages are very similar, even if they are not identical. Every time you receive a message, you calculate the checksums for the message and ask the 3 databases if they have seen similar checksums before/recently. If a very similar message has been delivered to millions of addresses, it is most likely spam. By submitting the checksums for your message, you are also contributing to the database so that it will start to detect your message as spam if lots of people have asked about the same message.


www.mailscanner.info MailScanner

Scoring

- SpamAssassin produces
  - Numerical score
    - Positive is more spammy
  - List of tests that succeeded
    - Can include score of each one
  - Long report containing 1 test per line
    - Includes description of each test
- Threshold scores for spam and high-scoring spam

SpamAssassin assigns a score to every RBL, rule, Bayes result and checksum match. It then totals these scores to produce a single number score for the whole message. If that is over the “Required SpamAssassin Score” then the message is treated as spam. It can also produce a couple of reports detailing what checks succeeded. These are very useful when trying to write your own rules or to see if a custom rule is needed for your particular situation. Custom rules are rarely needed, but can be useful if a new phishing scam appears and you quickly want to trap it.

If SpamAssassin sees no spam from an address after checking many messages, it can automatically add the address to its own internal whitelist which gives it a large negative contribution to the score. This protects future mail from this address being mistakenly tagged as spam. This whitelist is temporary and addresses do expire from it after a long time.

```
Required SpamAssassin Score = 6
```

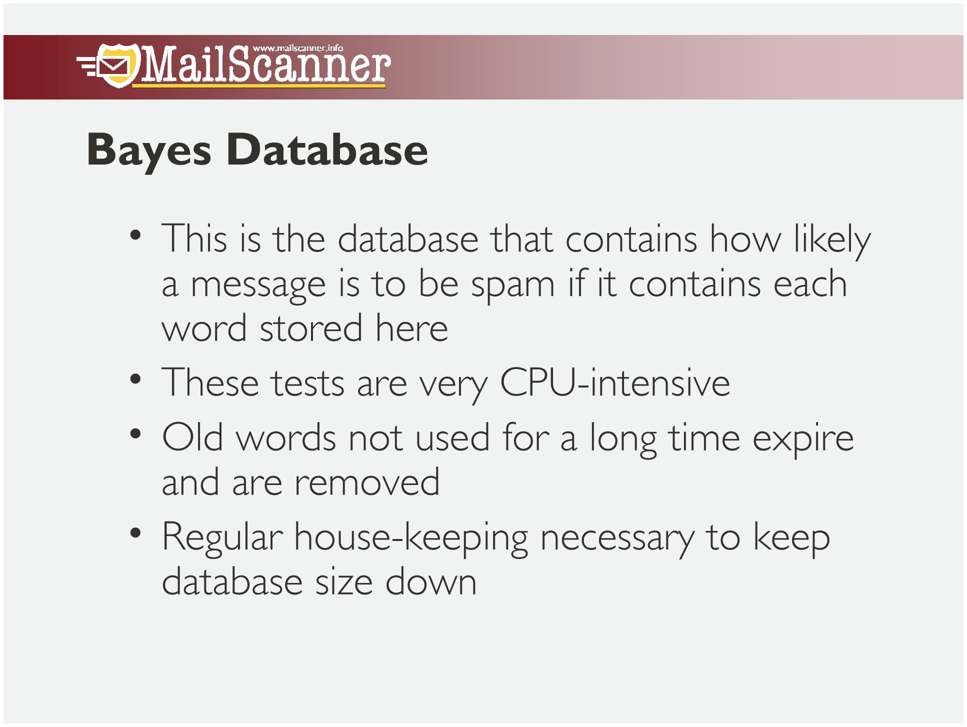
This replaces the SpamAssassin configuration value 'required_hits'. If a message achieves a SpamAssassin score higher than this value, it is spam. See also the High SpamAssassin Score configuration option. This can also be the filename of a ruleset, so the SpamAssassin required_hits value can be set to different values for different messages.

`High Scoring SpamAssassin Score = 10`

If a message achieves a SpamAssassin score higher than this value, then the "High Scoring Spam Actions" are used. You may want to use this to deliver moderate scores, while deleting very high scoring messsages. This can also be the filename of a ruleset.

`SpamAssassin Auto-Whitelist = no`

Set this option to "yes" to enable the automatic whitelisting functions available within SpamAssassin. This will cause addresses from which you get real mail, to be marked so that it will never incorrectly spam-tag messages from those addresses.



Bayes Database

- This is the database that contains how likely a message is to be spam if it contains each word stored here
- These tests are very CPU-intensive
- Old words not used for a long time expire and are removed
- Regular house-keeping necessary to keep database size down

Every word in every processed message is added to the Bayes database, so it can grow quickly. To keep it to a reasonable size, words that have not been seen for some time are expired and deleted from the database. The database is then rebuilt and compacted to optimise future searches. This rebuilding can take a few minutes and is best done with MailScanner's co-operation.

By default, SpamAssassin will automatically rebuild and compact it periodically. If you have a large Bayes database it can take several minutes for this to happen. It can do it at any time when MailScanner asks SpamAssassin to check a message. If the rebuild takes too long, MailScanner will think SpamAssassin has broken (it will trigger a timeout check) and will kill the SpamAssassin check. The result is that a temporary copy of the Bayes database will be left behind in the SpamAssassin working directory. By default this is in a .spamassassin directory under the home directory of the user MailScanner is being run as. If you are using sendmail on a Linux system, this will be in /root/.spamassassin. Every time SpamAssassin attempts to rebuild and compact the database, another temporary copy of it will be added to the .spamassassin directory. This can quickly take up a lot of space.

The best solution is to stop SpamAssassin doing the job itself and tell MailScanner to do it instead. You need to set "bayes_auto_expire 0" in spam.assassin.prefs.conf to stop SpamAssassin doing it.

The MailScanner options described below will set up MailScanner to do it. This should stop you getting lots of temporary files left lying around.

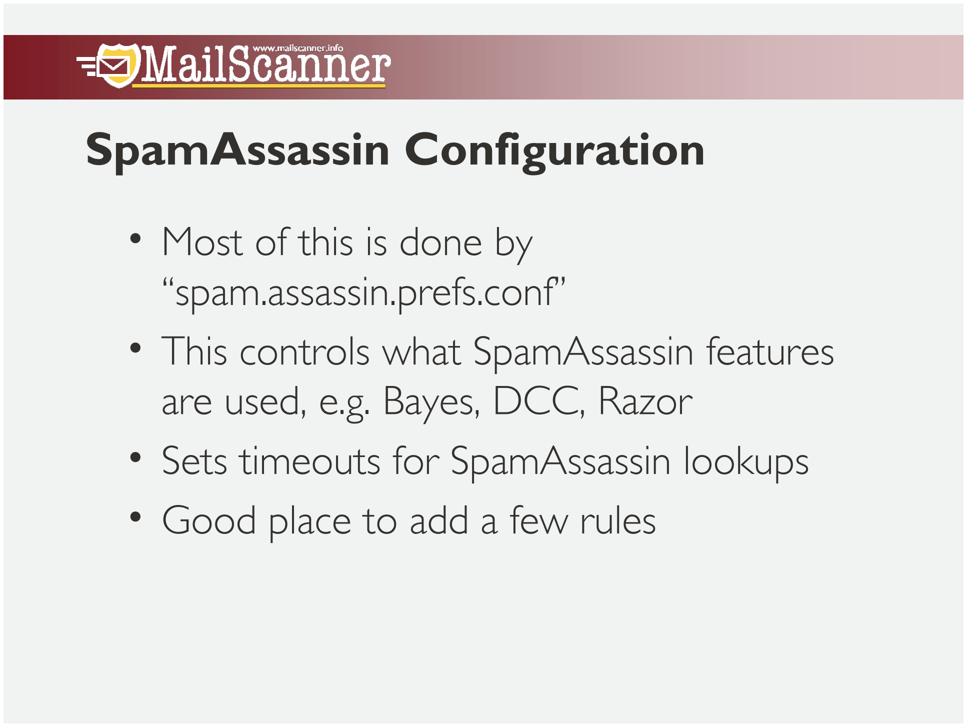
It is reasonable to rebuild it every few days on mid-sized sites. During the rebuild you have the choice of whether to suspend all mail processing until it has completed, or to keep processing mail, ignoring the Bayes database until it has completed.

`Rebuild Bayes Every = 0`

> If you are using the Bayesian statistics engine on a busy server, you may well need to force a Bayesian database rebuild and expiry at regular intervals. This is measures in seconds. 1 day = 86400 seconds. To disable this feature set this to 0.

`Wait During Bayes Rebuild = no`

> The Bayesian database rebuild and expiry may take a few minutes to complete. During this time you can either wait, or simply disable SpamAssassin checks until it has completed.


www.mailscanner.info MailScanner

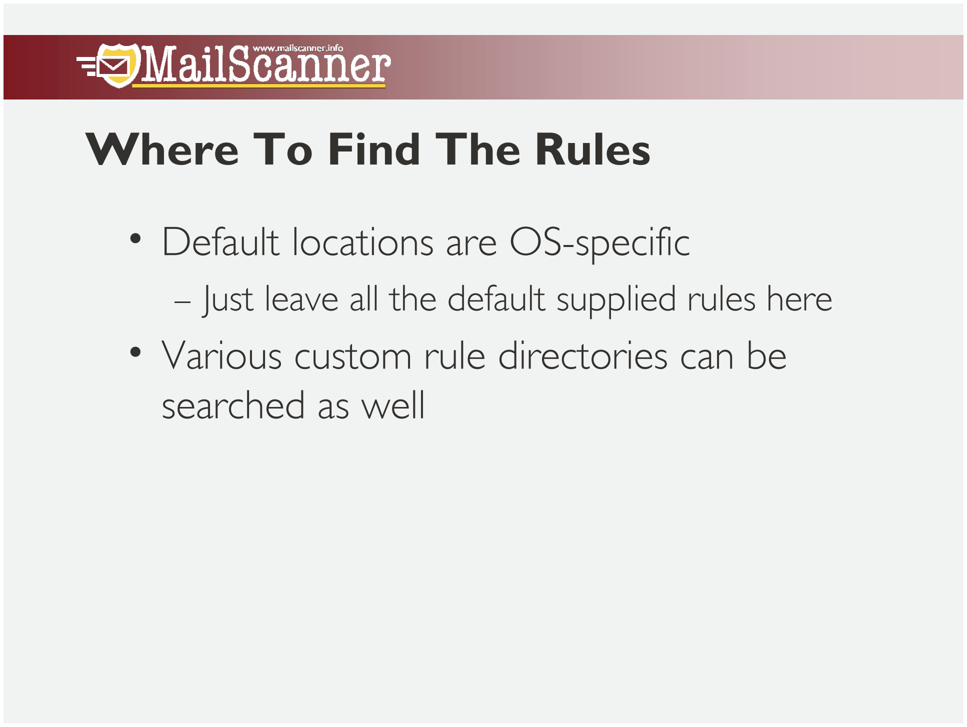
SpamAssassin Configuration

- Most of this is done by "spam.assassin.prefs.conf"
- This controls what SpamAssassin features are used, e.g. Bayes, DCC, Razor
- Sets timeouts for SpamAssassin lookups
- Good place to add a few rules

If you are only adding a few rules, this is a reasonable, easy place to add them. Syntax errors will be ignored. The full documentation of this file, including the syntax of rules, is in the man page for "Mail::SpamAssassin::Conf".

`SpamAssassin Prefs File = %etc-dir%/spam.assassin.prefs.conf`

Set the location of the SpamAssassin user_prefs file. If you want to stop SpamAssassin doing all the RBL checks again, then you can add "skip_rbl_checks = 1" to this prefs file.



Where To Find The Rules

- Default locations are OS-specific
  - Just leave all the default supplied rules here
- Various custom rule directories can be searched as well

The “SpamAssassin Site Rules Dir” is a good place to put your own rules. They have to be in “.cf” files and will be found automatically when SpamAssassin starts up. There is no central list of the names of all the “.cf” files to use.

If you are not sure that your rules are being used, shutdown and restart MailScanner and see if the “last accessed” timestamp on your file has been modified. This is a sure sign the file has been read. To look for syntax errors in your rules, use the “spamassassin” script supplied as part of the SpamAssassin package. Its man page will tell you how to use it.

`SpamAssassin User State Dir =`

The per-user files (bayes, auto-whitelist, user_prefs) are looked for here and in ~/.spamassassin/. Note the files are mutable. If this is unset then no extra places are searched for. If using Postfix, you probably want to set this as shown in the example line at the end of this comment, and do “mkdir /var/spool/MailScanner/spamassassin; chown postfix.postfix /var/spool/MailScanner/spamassassin”

`SpamAssassin Install Prefix =`

This setting is useful if SpamAssassin is installed in an unusual place, e.g. /opt/MailScanner. The install prefix is used to find some fallback directories if neither of the following two settings work. If this is set then it adds to the list of places that are searched; otherwise it has no effect.

`SpamAssassin Default Rules Dir =`

The default rules are searched for here, and in prefix/share/spamassassin, /usr/local/share/spamassassin, /usr/share/spamassassin, and maybe others. If this is set then it adds to the list of places that are searched; otherwise it has no effect.

`SpamAssassin Site Rules Dir =`

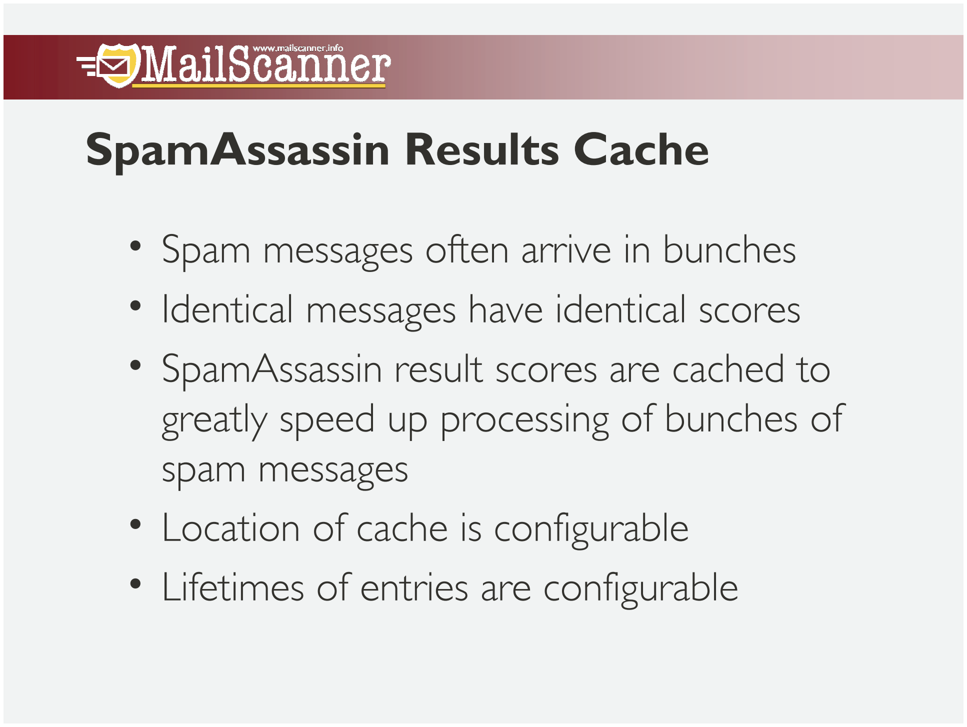
The site rules are searched for here. Normal location on most systems is /etc/mail/spamassassin.

`SpamAssassin Local Rules Dir =`

The site-local rules are searched for here, and in prefix/etc/spamassassin, prefix/etc/mail/spamassassin, /usr/local/etc/spamassassin, /etc/spamassassin, /etc/mail/spamassassin, and maybe others. If this is set then it adds to the list of places that are searched; otherwise it has no effect.

`SpamAssassin Local State Dir =`

The rules created by the SpamAssassin "sa-update" tool are searched for here. Be very careful of setting this wrong, it could cause your SpamAssassin to fail to find any rules. The default value is blank and it should be left that way unless you have changed SpamAssassin's "sa-update" to store rules elsewhere.



SpamAssassin Results Cache

- Spam messages often arrive in bunches
- Identical messages have identical scores
- SpamAssassin result scores are cached to greatly speed up processing of bunches of spam messages
- Location of cache is configurable
- Lifetimes of entries are configurable

Due to the nature of spam, identical messages often arrive in large quantities over a short period of time. By temporarily storing the scores of recent messages in a cache, large speed improvements up to 40% can be achieved. If the result of the SpamAssassin test is taken from the cache, the word "cached" is added to the SpamAssassin report in the spam header added to the message.

`Cache SpamAssassin Results = yes`

Do you want to cache the score results of SpamAssassin? This provides a large speed improvement of up to 40%.

`SpamAssassin Cache Database File = /var/spool/MailScanner/incoming/SpamAssassin.cache.db`

This sets the location of the file used to store the cached results. Its location is chosen to be in the MailScanner incoming work directory so that it is most likely to be stored using the tmpfs filesystem which greatly speeds access to the cache. Because they are temporary results, there is no loss of important data if the system is rebooted and the tmpfs data is wiped.

`SpamAssassin Cache Timings = 1800,300,10800,172800,600`

These are the 5 timing values used to set the lifetime of each type of record stored in the cache. They are all set in seconds.

1. Non-spam cache lifetime (default is 30 minutes)
2. Spam (low scoring) cache lifetime (default is 5 minutes)
3. High scoring spam cache lifetime (default is 3 hours)
4. Viruses cache lifetime (default is 2 days)
5. How frequently to expire old cache values (default is 10 minutes)

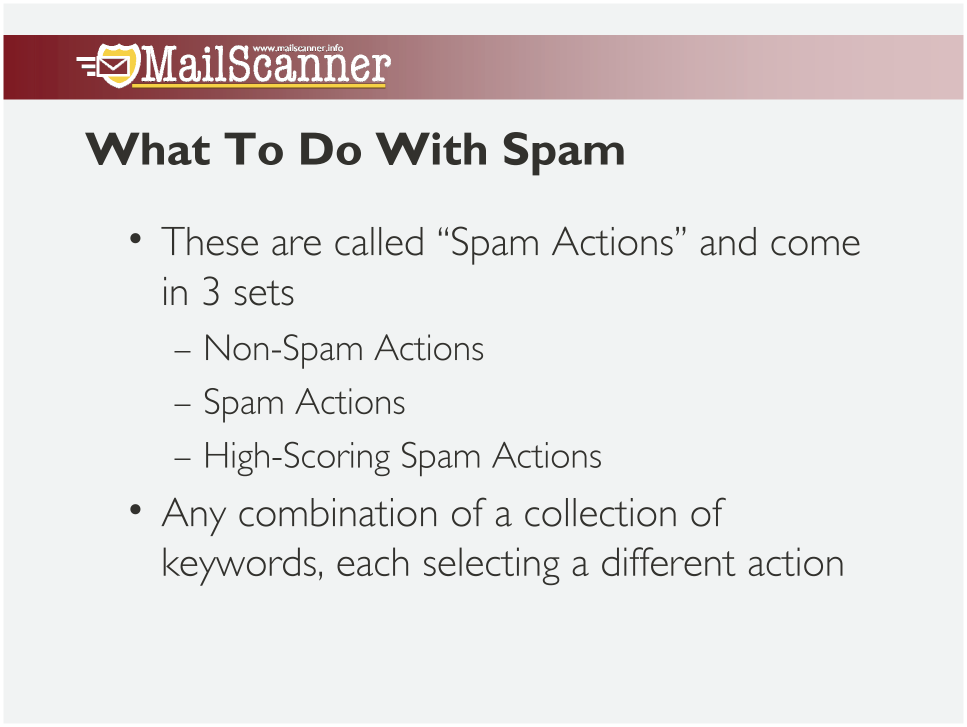
Headers and Subject: tags

- I will come back to these later

Promise!

5.3

Spam Actions



What To Do With Spam

- These are called “Spam Actions” and come in 3 sets
  - Non-Spam Actions
  - Spam Actions
  - High-Scoring Spam Actions
- Any combination of a collection of keywords, each selecting a different action

High-scoring spam only triggers the high-scoring spam actions. It does not trigger the other (low) scoring spam actions.

`Spam Actions = deliver`

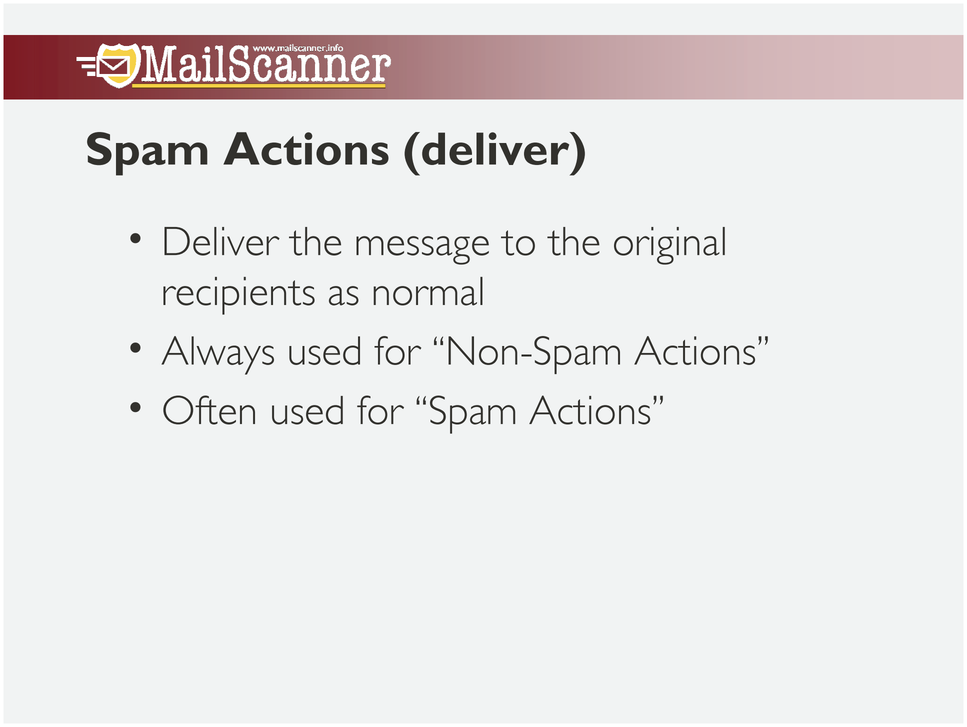
This is a list of actions to take when a message is spam. This can also be the filename of a ruleset.

`High Scoring Spam Actions = deliver`

This is just like the "Spam Actions" option above, except that it applies then the score from SpamAssassin is higher than the "High SpamAssassin Score" value. This can also be the filename of a ruleset.

`Non Spam Actions = deliver`

This is just like the "Spam Actions" option above, except that it applies to messages that are *NOT* spam. This can also be the filename of a ruleset.



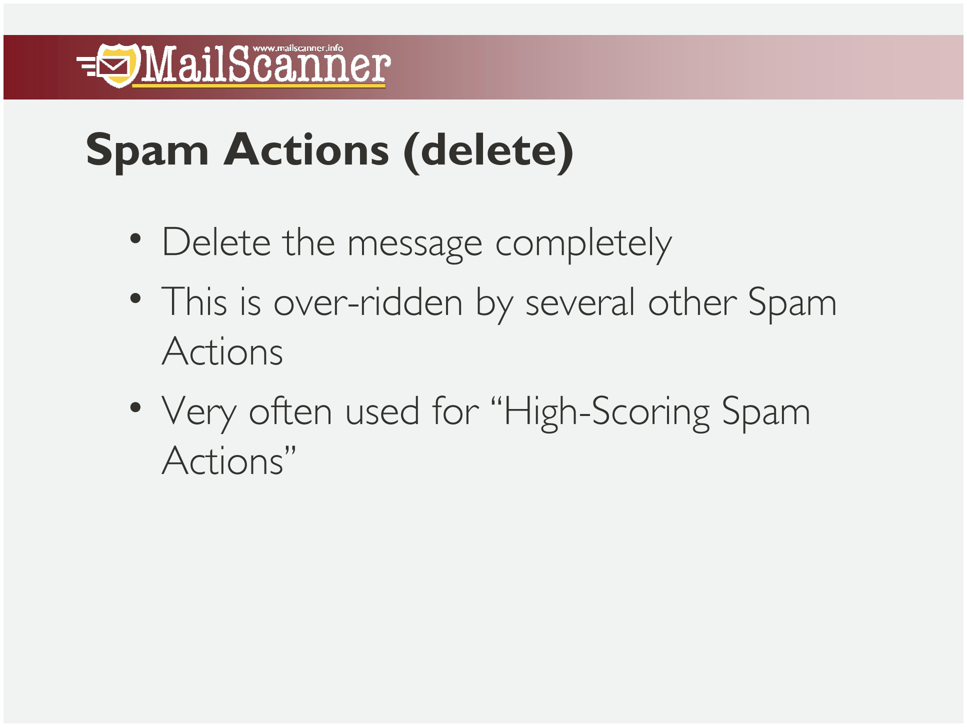
Spam Actions (deliver)

- Deliver the message to the original recipients as normal
- Always used for "Non-Spam Actions"
- Often used for "Spam Actions"

This is what is normally done with a message, it is delivered to its original recipients.

`Spam Actions = deliver`

    deliver the message as normal



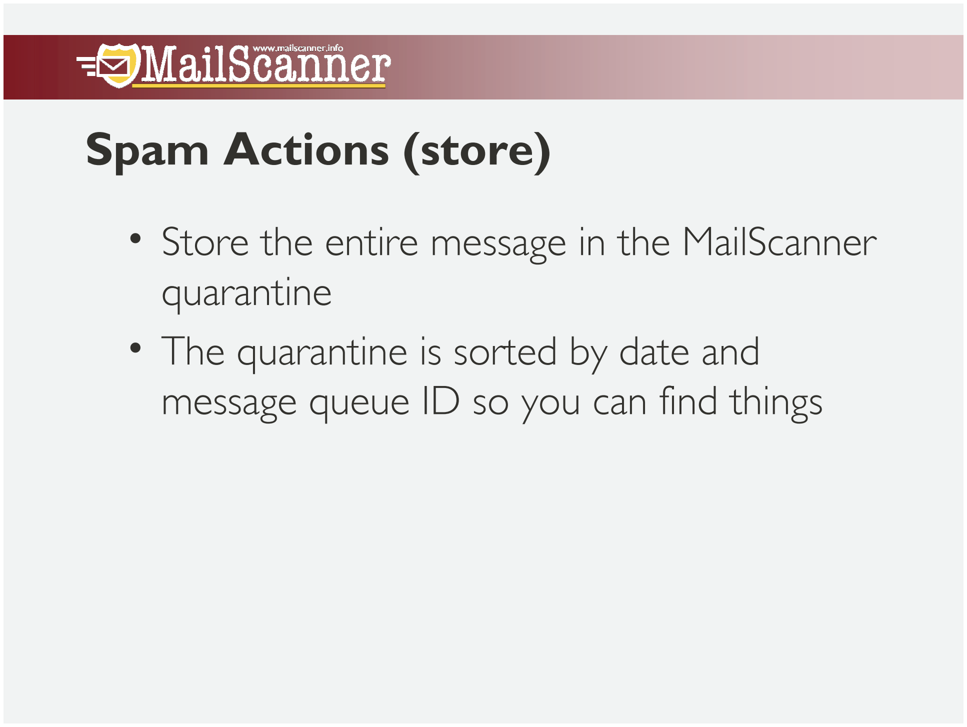
Spam Actions (delete)

- Delete the message completely
- This is over-ridden by several other Spam Actions
- Very often used for “High-Scoring Spam Actions”

This is over-ridden by most of the other actions.

`Spam Actions = delete`

delete the message



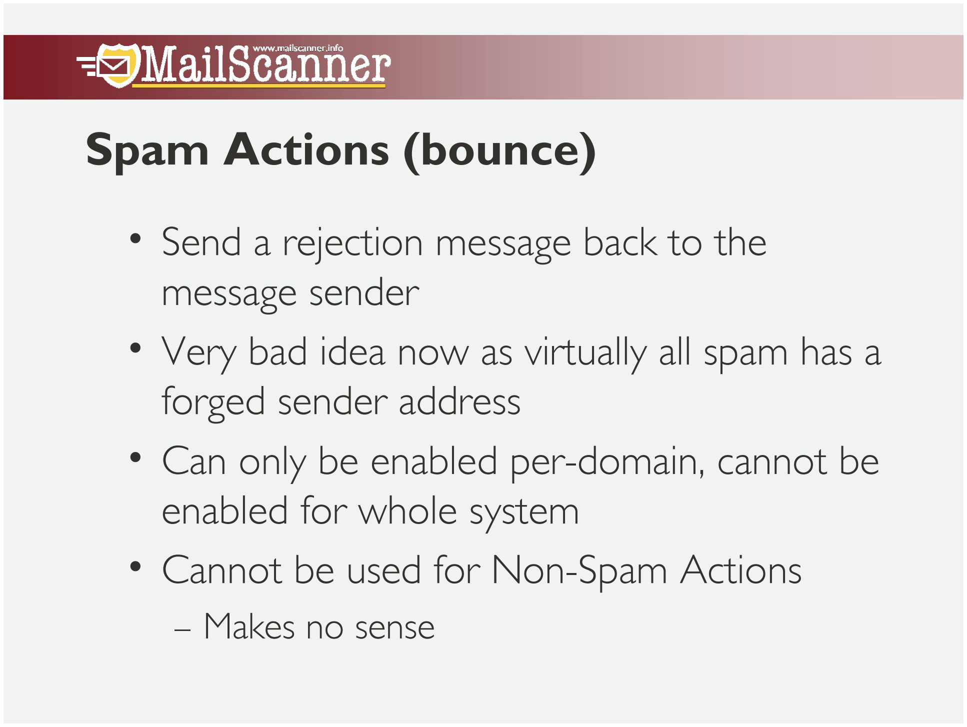
Spam Actions (store)

- Store the entire message in the MailScanner quarantine
- The quarantine is sorted by date and message queue ID so you can find things

This stores the whole message. The message is stored in the format chosen by the option that selects RFC822 message files or raw queue files.

`Spam Actions = store`

store the message in the quarantine



Spam Actions (bounce)

- Send a rejection message back to the message sender
- Very bad idea now as virtually all spam has a forged sender address
- Can only be enabled per-domain, cannot be enabled for whole system
- Cannot be used for Non-Spam Actions
  - Makes no sense

Use of this option is strongly discouraged, as virtually all spam fakes the sender's address, so this will send a message to some innocent 3rd party whose email address has been used by the spammer.

In order for this option to have any effect, you have to enable the spam bounce option for the domains you wish to bounce to. The option to enable this cannot be set to a blanket "yes" value for all addresses, it can only be enabled at the domain level. Attempts to set it to "yes" for all addresses will result in it being ignored completely.

This option will be ignored if used as a non-spam action, as bouncing normal mail makes no sense.

`Spam Actions = bounce`

send a rejection message back to the sender

`Enable Spam Bounce = %rules-dir%/bounce.rules`

You can use this ruleset to enable the "bounce" Spam Action. You must *only* enable this for mail from sites with which you have agreed to bounce possible spam. Use it on low-scoring spam only (<10) and only to your regular customers for use in the rare case that a message is mis-tagged as spam when it shouldn't have been. Beware that many sites will automatically delete the bounce messages created by using this option unless you have agreed this with them in advance. If you enable this, be prepared to handle the irate responses from people to whom you are essentially sending more spam!

`Sender Spam Report = %report-dir%/sender.spam.report.txt`

Sent when a message triggers both a Spam List and SpamAssassin.

`Sender Spam List Report = %report-dir%/sender.spam.rbl.report.txt`

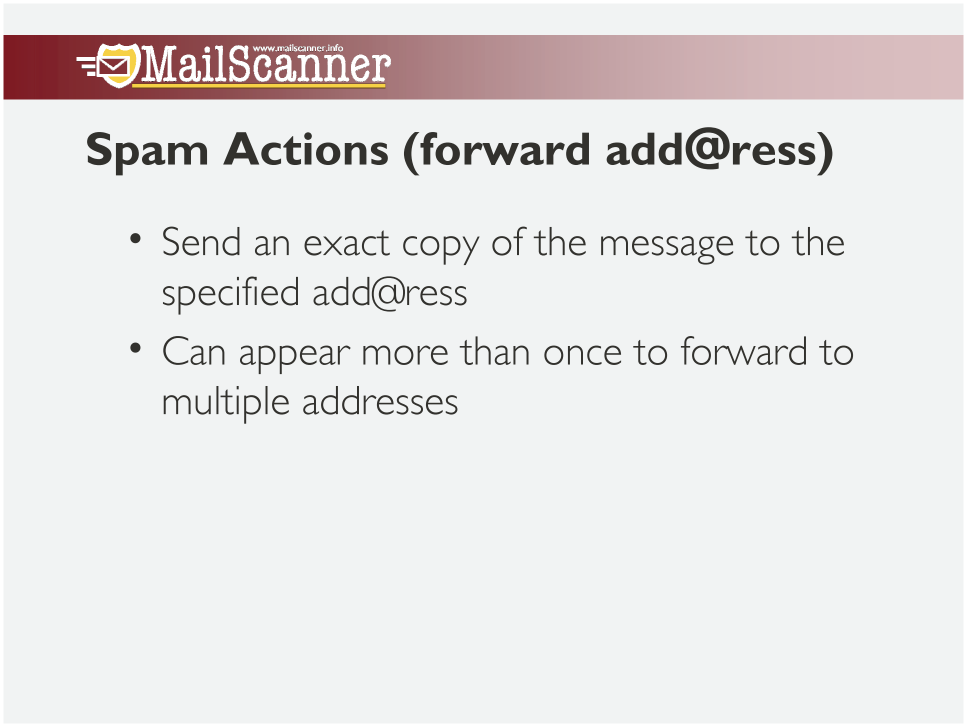
Sent when a message triggers a Spam List.

`Sender SpamAssassin Report = %report-dir%/sender.spam.sa.report.txt`

Sent when a message triggers SpamAssassin.

`Bounce Spam As Attachment = no`

If this is set to no, the original spam message is not included in the sender report. If this is set to yes, the original spam message is included in the sender report as an attachment containing the entire message. If you are bouncing spam that claims to have come from one of your important customers, this will enable them to know what message of theirs, if any, was incorrectly detected as spam.



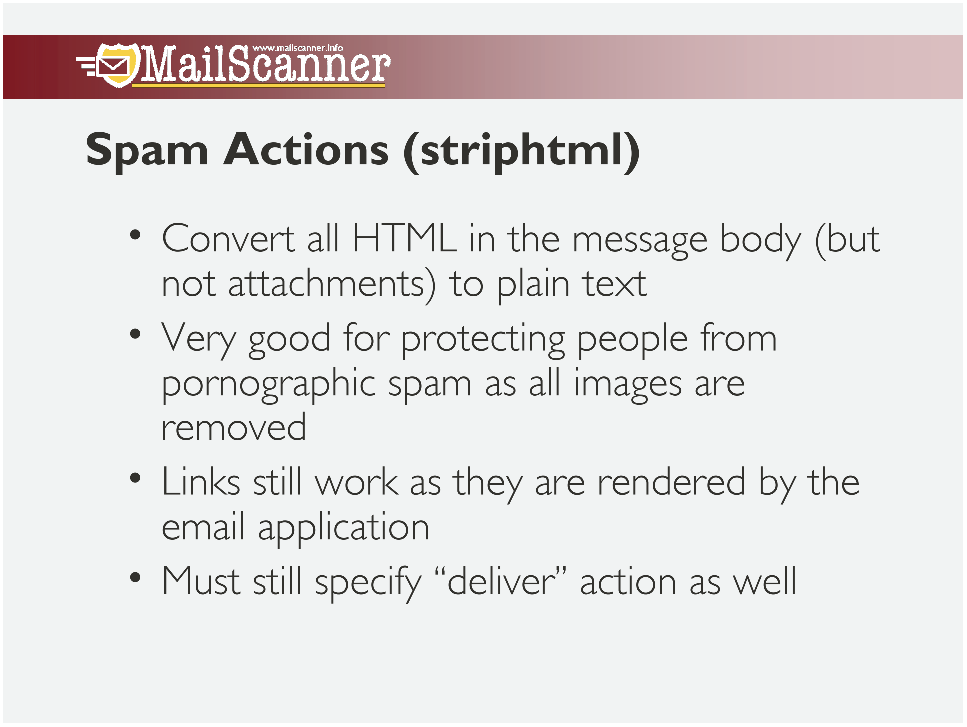
Spam Actions (forward add@ress)

- Send an exact copy of the message to the specified add@ress
- Can appear more than once to forward to multiple addresses

This sends a copy of the original message to the specified recipient. Multiple recipients can be specified if necessary, either with or without repeating the "forward" keyword itself.

`Spam Actions = forward user@domain.com`

  forward a copy of the message to user@domain.com



Spam Actions (striphtml)

- Convert all HTML in the message body (but not attachments) to plain text
- Very good for protecting people from pornographic spam as all images are removed
- Links still work as they are rendered by the email application
- Must still specify "deliver" action as well

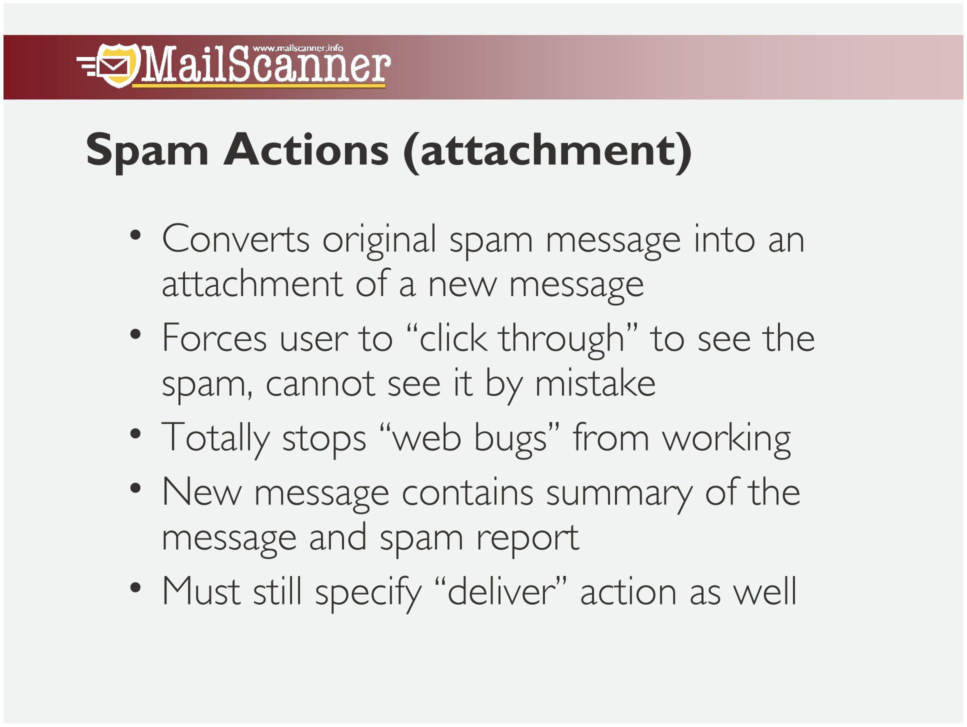
All HTML tags are removed, leaving just the raw text between the tags. <A HREF> links to web sites are replaced with the target address of the link. It is left to the user's email application to render these as active links in the message.

All HTML comments are also removed, which deletes the huge comment block containing formatting information if the message was created using Microsoft Word.

It is very useful for protecting sensitive or young users from any type of pornographic spam as it removes all graphical content. However, there is no easy way for the recipient to retrieve an unmodified copy of the original message.

`Spam Actions = striphtml`

convert all in-line HTML content to plain text. You need to specify "deliver" as well for the message to reach the original recipient.


MailScanner

Spam Actions (attachment)

- Converts original spam message into an attachment of a new message
- Forces user to “click through” to see the spam, cannot see it by mistake
- Totally stops “web bugs” from working
- New message contains summary of the message and spam report
- Must still specify “deliver” action as well

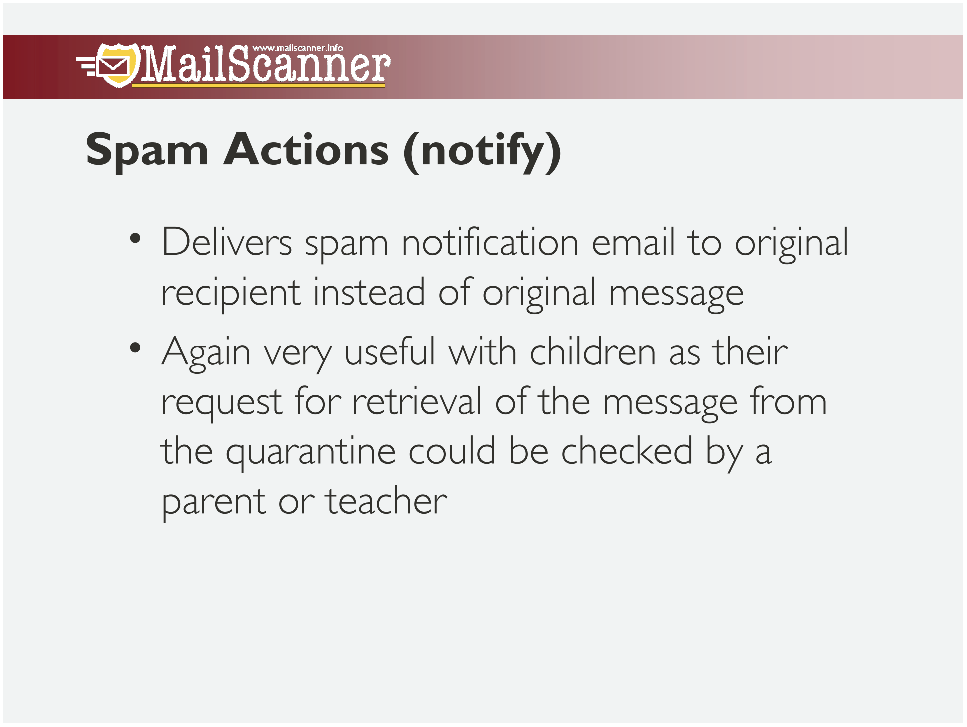
Web bugs and other spyware tricks are defeated as initially viewing the message does not cause any of the images in the message to be rendered. Also potentially dangerous scripts and other tricks are disabled unless the user, having seen the addresses and subject of the message, chooses to open the message to view it. They will then see the original fully-rendered version of the message.

`Spam Actions = attachment`

Convert the original message into an attachment of the message. This means the user has to take an extra step to open the spam, and stops "web bugs" very effectively.

`Inline Spam Warning = %report-dir%/inline.spam.warning.txt`

If you use the 'attachment' Spam Action or High Scoring Spam Action then this is the location of inline spam report that is inserted at the top of the message.



Spam Actions (notify)

- Delivers spam notification email to original recipient instead of original message
- Again very useful with children as their request for retrieval of the message from the quarantine could be checked by a parent or teacher

This option appears to be strange as it replaces a spam message with a notification that you have received the spam message.

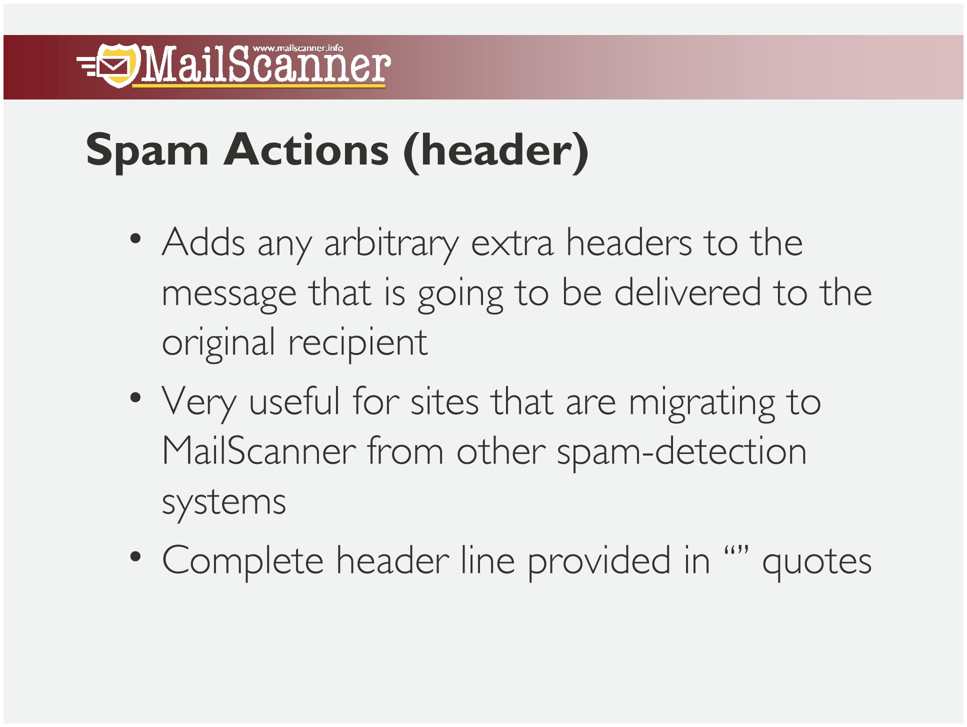
However, if your users are children, this is actually useful as the children receive the notification and can then ask a responsible adult, such as a parent or teacher, to deliver the spam message to them. This gives the adult the chance to preview the message first to ensure it is suitable for viewing by the child.

`Spam Actions = notify`

Send the recipients a short notification that spam addressed to them was not delivered. They can then take action to request retrieval of the original message if they think it was not spam.

`Recipient Spam Report = %report-dir%/recipient.spam.report.txt`

If you use the 'notify' Spam Action or High Scoring Spam Action then this is the location of the notification message that is sent to the original recipients of the message.



Spam Actions (header)

- Adds any arbitrary extra headers to the message that is going to be delivered to the original recipient
- Very useful for sites that are migrating to MailScanner from other spam-detection systems
- Complete header line provided in “” quotes

This action can be used to add any arbitrary headers, with any values, to the message that is delivered to the recipients, if the 'deliver' action is also specified.

If you are migrating from some other spam-detection system, such as a standalone SpamAssassin setup using spamc/spamd and procmail, your users will probably be used to seeing

```
X-Spam-Status: YES
```

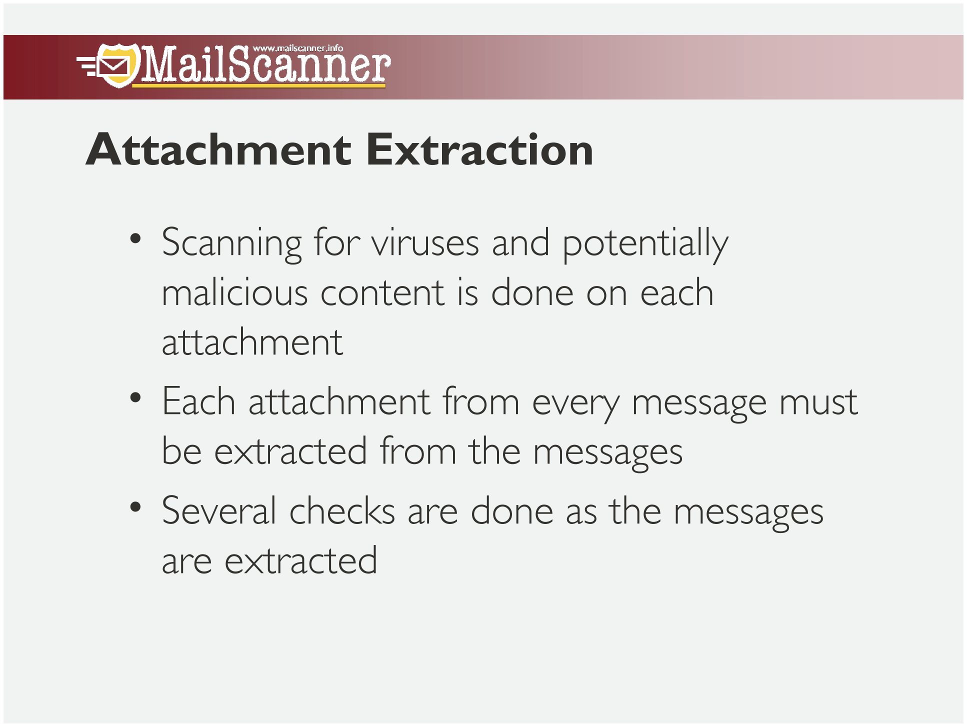
in the headers of spam messages they receive. To avoid having all your users change any automatic filters they have created, it would be very useful if you still provide this header. This can be done using, for example,

```
Spam Actions = header "X-Spam-Status: YES" deliver
```

Any number of headers may be added to the message, just place multiple headers, all in double "" quotes, in the appropriate "Actions" configuration option.

5.4

Attachment Extraction



MailScanner www.mailscanner.info

Attachment Extraction

- Scanning for viruses and potentially malicious content is done on each attachment
- Each attachment from every message must be extracted from the messages
- Several checks are done as the messages are extracted

The maximum size of a message and the maximum size of any individual attachment can be limited. Using rulesets, you could for example limit the size of a message for users who are downloading messages by POP over dialup connections, as very large attachments can cause problems for some email applications when working over slow links. Careful application of these options can cut a lot of technical support calls from users who are having trouble downloading their email.

Setting the maximum attachment size to 0 will block any message containing an attachment. Use the value of -1 to disable this check.

`Incoming Work Dir = /var/spool/MailScanner/incoming`

Set where to unpack incoming messages before scanning them This can completely safely use tmpfs or a ramdisk, which will give you a significant performance improvement.
NOTE: The path given here must not include any links at all, but must be the absolute path to the directory. If it is not the absolute path, MailScanner will continue to work correctly, but it will log a warning message about this.

`Incoming Work User =`
`Incoming Work Group =`

You should not normally need to touch these settings at all, unless you are using ClamAV and need to be able to use the external archive unpackers instead of ClamAV's built-in ones.If you want to create the temporary working files so they are owned by a user other than the "Run As User" setting at the top of this file, you can change that here.
Note: If the "Run As User" is not "root" then you cannot change the user but may still be able to change the group, if the "Run As User" is a member of both of the groups "Run As Group" and "Incoming Work Group".

`Incoming Work Permissions = 0600`
You should not normally need to touch these settings at all, unless you are using ClamAV and need to be able to use the external archive unpackers instead of ClamAV's built-in ones. If you want processes running under the same *group* as MailScanner to be able to read the working files (and list what is in the directories, of course), set to 0640. If you want *all* other users to be able to read them, set to 0644. For a detailed description, if you're not already familiar with it, refer to `man 2 chmod`. Typical use: external helper programs of virus scanners (notably ClamAV), like unpackers. Use with care, you may well open security holes.

`Maximum Message Size = 0`
The maximum size, in bytes, of any message including the headers. If this is set to zero, then no size checking is done. This can also be the filename of a ruleset, so you can have different settings for different users. You might want to set this quite small for dialup users so their email applications don't time out downloading huge messages.

`Maximum Attachment Size = -1`
The maximum size, in bytes, of any attachment in a message. If this is set to zero, effectively no attachments are allowed. If this is set less than zero, then no size checking is done. This can also be the filename of a ruleset, so you can have different settings for different users. You might want to set this quite small for large mailing lists so they don't get deluged by large attachments.

`Sender Size Report =`
The filename of the report sent back to the message sender if the message is outside the bounds of the “Maximum Message Size” and “Maximum Attachment Size” settings. This can also be the filename of a ruleset, so you can have different settings for different users.

`Stored Size Message Report =`
This is the filename of the report message sent back to the sender when the message size exceeded the allowable limits and the message was stored in the quarantine. This can also be the filename of a ruleset.

`Deleted Size Message Report =`

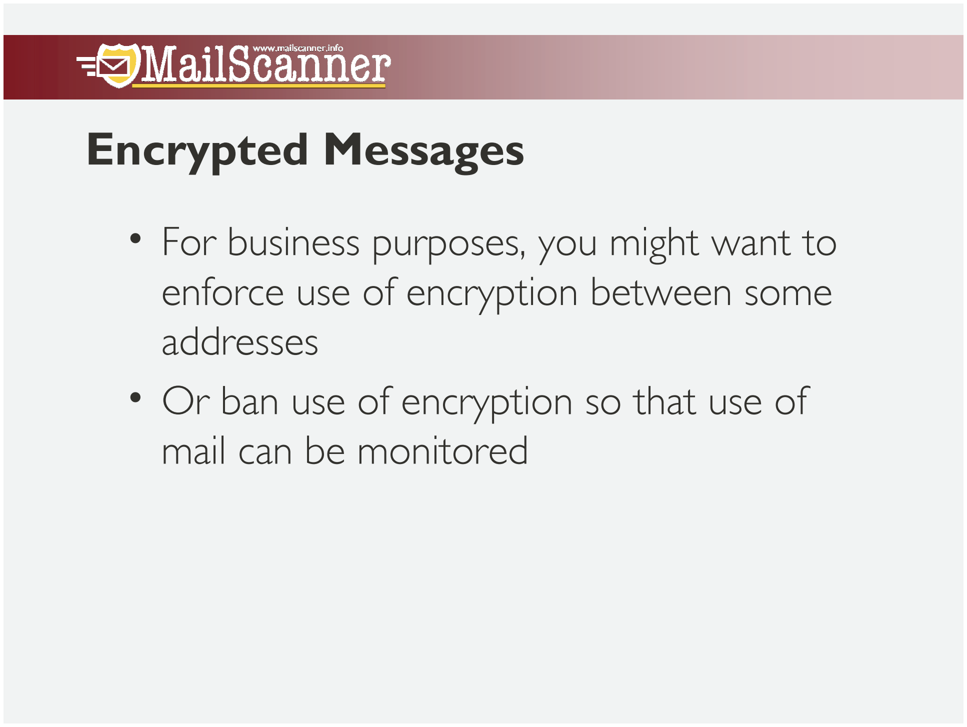
This is the filename of the report message sent back to the sender when the message size exceeded the allowable limits and the message was deleted from the server. This can also be the filename of a ruleset.

`Size Modify Subject =`

If the size of the message exceeded the allowable limits, and nothing else was wrong with the message, should the Subject line of the message be modified? This can be “no”, “yes” (or“start”) or “end”. The possible values reflect the position in the Subject line where the defined text will be added.

`Size Subject Text =`

If the size of the message exceeded the allowable limits, what text should be added to the Subject line if required?



Encrypted Messages

- For business purposes, you might want to enforce use of encryption between some addresses
- Or ban use of encryption so that use of mail can be monitored

In some corporate installations, it is useful to be able to force the use of encrypted or unencrypted email. MailScanner can also extract the public keys from the messages passing through it, in order that they could be saved for future signature checking. This is not covered here as it was written for only one user and is not fully supported.

Many corporations disallow the use of encrypted email by their employees as it removes the ability to monitor the employees' use of email. Any breaches of the policy are logged as attempts to use encrypted email may indicate an employee trying to give away company confidential information.

In the reverse scenario, you may want to force the use of encrypted email to certain other sites, such as business partners or lawyers where it is essential that the contents of the email messages are private and cannot be read by anyone else.

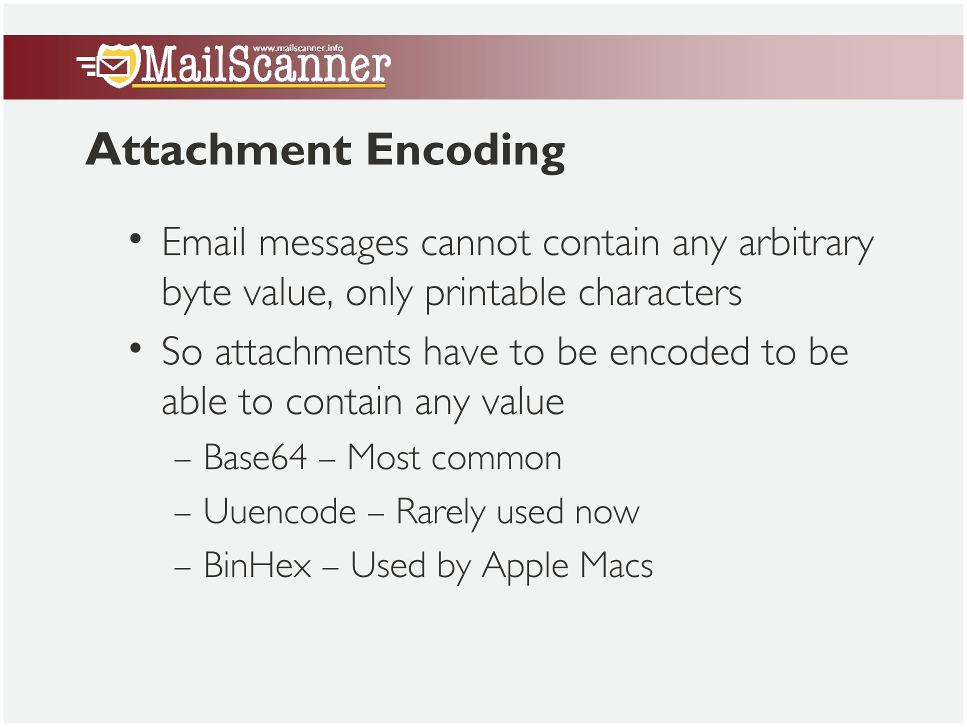
Note that the signatures are added to clean messages in a way that is compatible with most email encryption systems, such as PGP.

`Block Encrypted Messages = no`

Should encrypted messages be blocked? This is useful if you are wary about your users sending encrypted messages to your competition. This can be a ruleset so you can block encrypted message to certain domains.

`Block Unencrypted Messages = no`

Should unencrypted messages be blocked? This could be used to ensure all your users send messages outside your company encrypted to avoid snooping of mail to your business partners. This can be a ruleset so you can just check mail to certain users/domains.

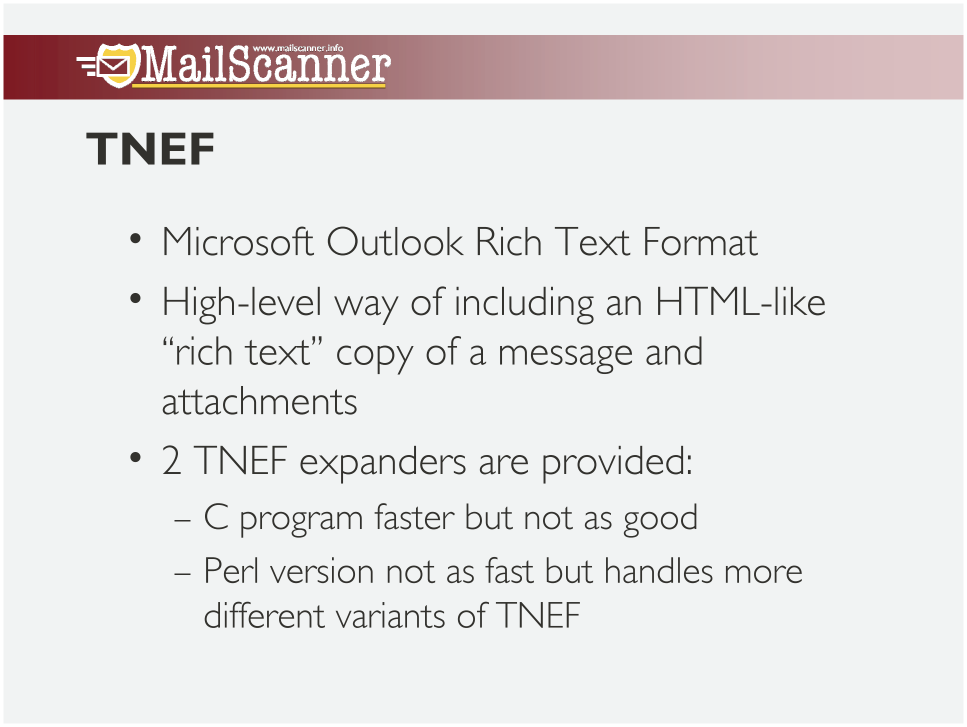


Attachment Encoding

- Email messages cannot contain any arbitrary byte value, only printable characters
- So attachments have to be encoded to be able to contain any value
  - Base64 – Most common
  - Uuencode – Rarely used now
  - BinHex – Used by Apple Macs

The attachments are all part of the body of the message, they are not transmitted separately in any way. There are various encoding standards as email messages can only contain printable ASCII characters, whereas file attachments need to use any value from 0-255 in each byte.

The most common encoding format by far is "Base64". The Base64 decoder and encoder used in MailScanner is written entirely in C for maximum efficiency and speed.


TNEF

- Microsoft Outlook Rich Text Format
- High-level way of including an HTML-like "rich text" copy of a message and attachments
- 2 TNEF expanders are provided:
  - C program faster but not as good
  - Perl version not as fast but handles more different variants of TNEF

This has been mentioned earlier. The message body contains a plain-text rendition of the body text, and one attachment called "winmail.dat". This attachment contains a rich-text rendition of the body text along with all the attachments in the message.

Fortunately this format is now uncommon, and users are discouraged from setting up systems that use it.

Like other external programs used by MailScanner, the TNEF extraction process is protected by a timeout to defend against attacks against the TNEF format and the decoders used by MailScanner.

`Expand TNEF = yes`

Expand TNEF attachments using an external program (or a Perl module)? This should be "yes" unless the scanner you are using (Sophos, McAfee) has the facility built-in. However, if you set it to "no", then the filenames within the TNEF attachment will not be checked against the filename rules.

`TNEF Expander = /usr/bin/tnef --maxsize=100000000`

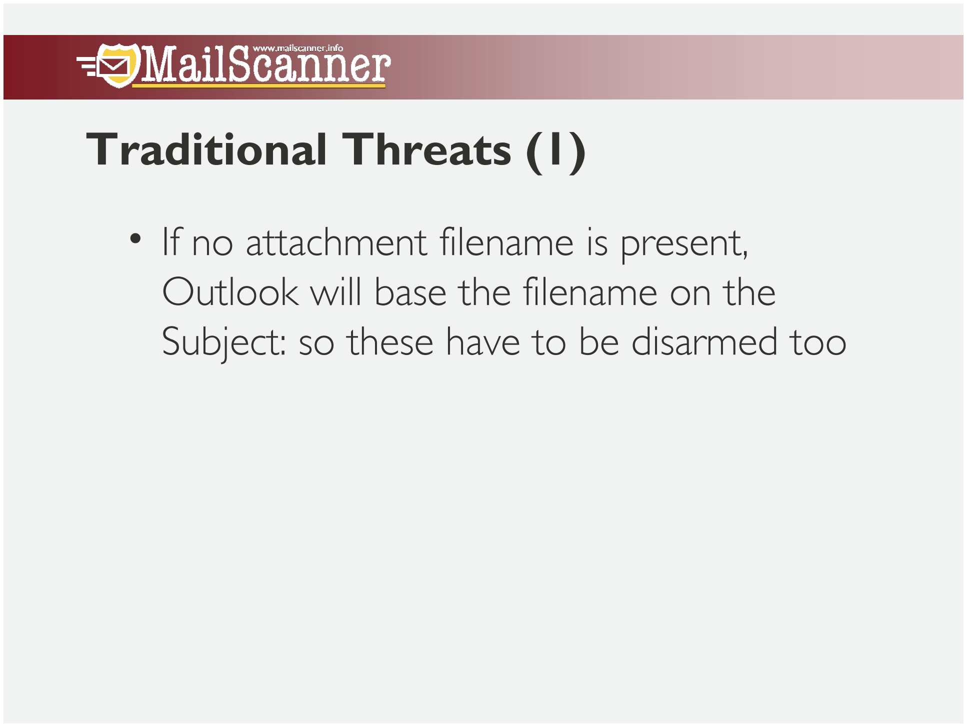
Where the MS-TNEF expander is installed. This is EITHER the full command (including maxsize option) that runs the external TNEF expander binary, OR the keyword "internal" which will make MailScanner use the Perl module that does the same job. They are both provided as I am unsure which one is faster and which one is capable of expanding more file formats (there are plenty!). The --maxsize option limits the maximum size that any expanded attachment may be. It helps protect against Denial Of Service attacks in TNEF files. A valid setting is “TNEF Expander = internal”. This can also be the filename of a ruleset.

`Deliver Unparsable TNEF = no`

Some versions of Microsoft Outlook generate unparsable Rich Text format attachments. Do we want to deliver these bad attachments anyway? Setting this to yes introduces the slight risk of a virus getting through, but if you have a lot of troubled Outlook users you might need to do this. We are working on a replacement for the TNEF decoder. This can also be the filename of a ruleset.

`TNEF Timeout = 120`

The maximum length of time the TNEF Expander is allowed to run for 1 message. (in seconds)



Traditional Threats (1)

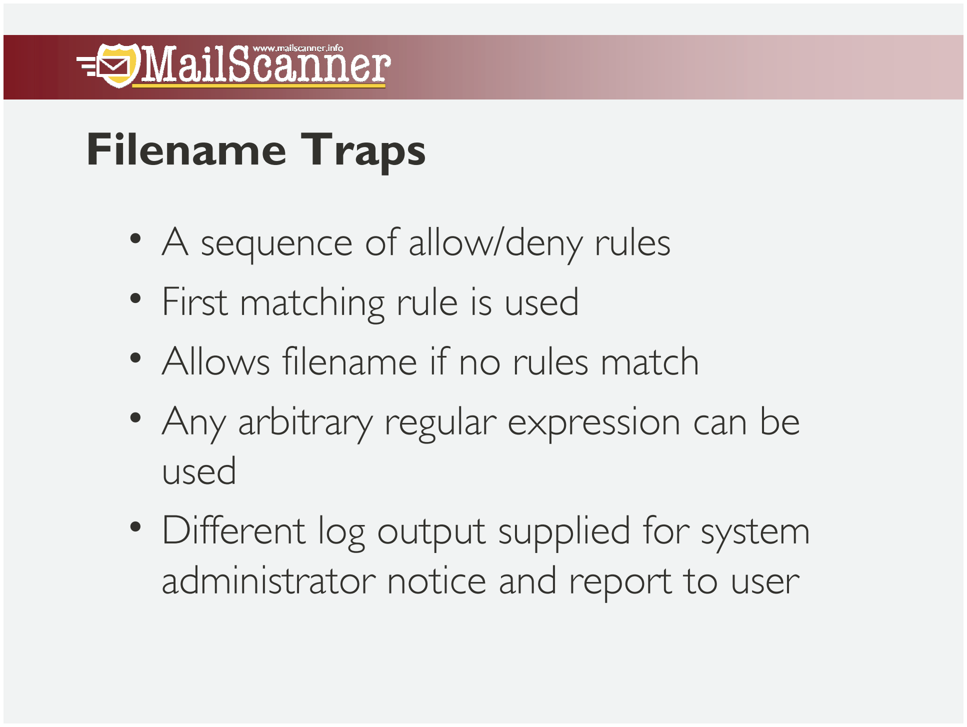
- If no attachment filename is present, Outlook will base the filename on the Subject: so these have to be disarmed too

Other email applications tend to call the file “Untitled” or some other sensible safe name. Outlook filename exploits are possible by just including the attachment with no filename, as there are limits to the changes that can be made to the subject without your users complaining about their messages being edited. MailScanner attempts to sanitise the subject line by removing any large sequence of whitespace and very long subject lines will be truncated. Unfortunately it is not reasonable to apply all the filename restrictions to the subject line.

Sometimes multiple Subject: headers are put in the message to confuse email scanning systems so that the Subject: seen by the user does not contain anything indicating that the message might be spam. MailScanner keeps the 1st Subject: header and discards all others.

5.5

Attachment Checks



Filename Traps

- A sequence of allow/deny rules
- First matching rule is used
- Allows filename if no rules match
- Any arbitrary regular expression can be used
- Different log output supplied for system administrator notice and report to user

Each line of filename.rules.conf can either be a comment or can contain 4 fields. These fields must be separated by tab characters and not just spaces. The files filename.rules.conf and filetype.rules.conf are the only files in which tabs must be used.

Each line contains 4 fields:

1. “allow” or “deny”.

2. the regular expression that is to be matched against the filename. This regular expression should not have “/” characters around it.

3. The text that is placed in the notice to the system administrators.

4. The text that is placed in the report contained in the attachment warning file that is delivered to the recipient in place of the relevant attachment

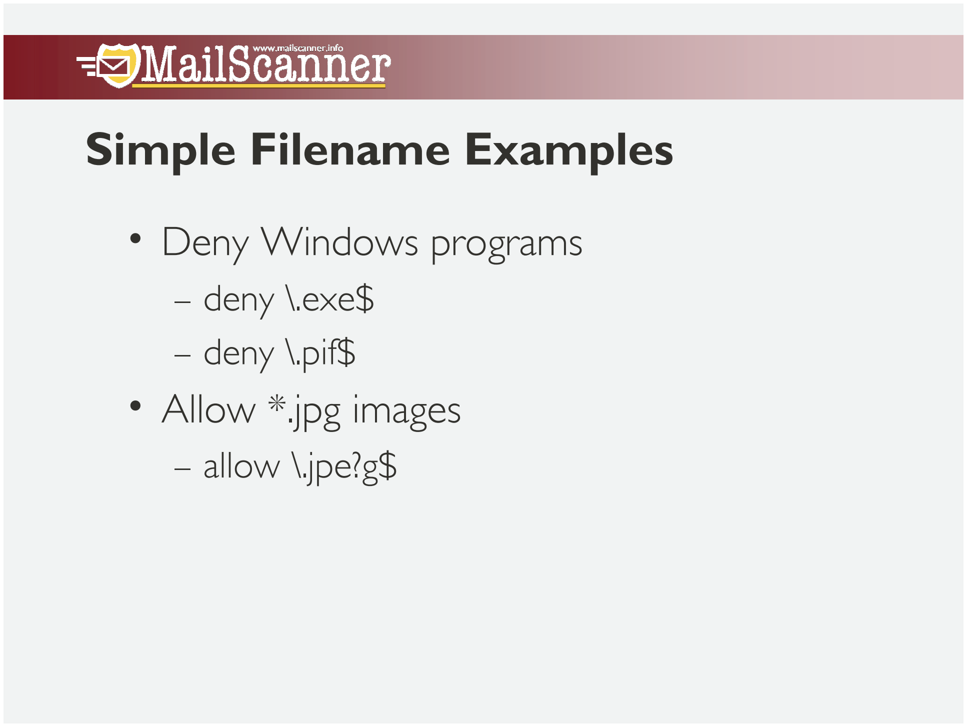
The rules are applied in order from the top to the bottom of the file. The first rule that matches the filename is the one that is used. Processing the file stops at that point and no further rules are checked.

If no rules match then the filename is allowed.

A ruleset could be used with this option to choose different filename checks files for different users as required.

`Filename Rules = %etc-dir%/filename.rules.conf`

Set where to find the attachment filename ruleset. The structure of this file is explained elsewhere, but it is used to accept or reject file attachments based on their name, regardless of whether they are infected or not. This can also point to a ruleset, but the ruleset filename must end in ".rules" so that MailScanner can determine if the filename given is a ruleset or not!



Simple Filename Examples

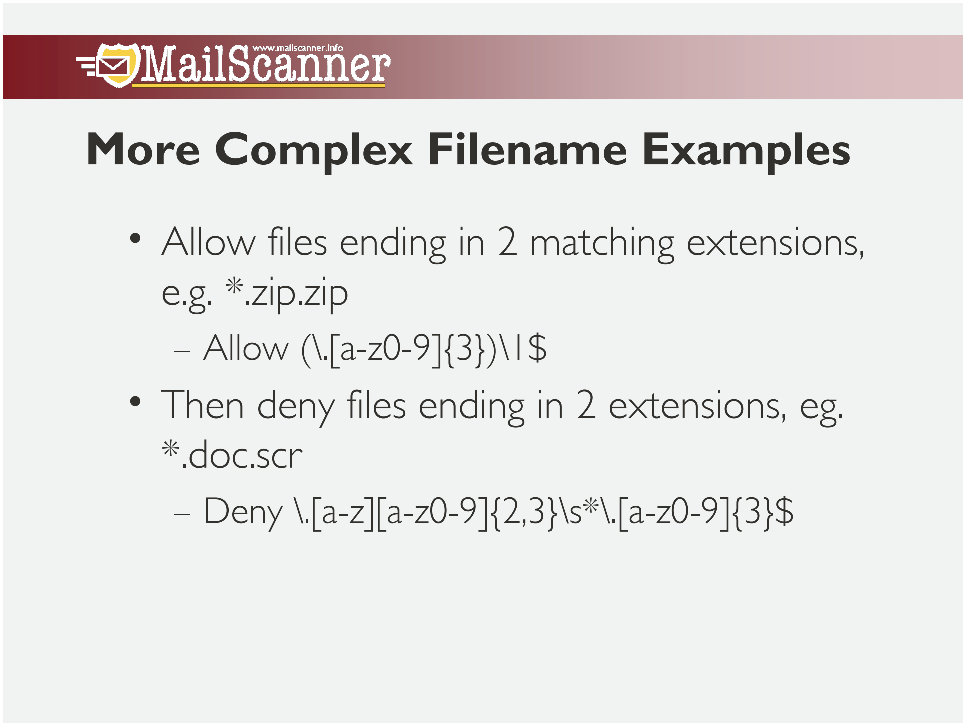
- Deny Windows programs
  - deny \.exe$
  - deny \.pif$
- Allow *.jpg images
  - allow \.jpe?g$

Note that the "." characters in the regular expression are escaped with a "\" so that they match a real "." and not just any character.

The "?" matches 0 or 1 occurrences of the preceding character, so that both ".jpg" and ".jpeg" are matched by the expression.

Every filename expression is checked with and without a '?' symbol on the end (but before any '$' end-of-string match) so that filenames encoded in other character sets are still detected.

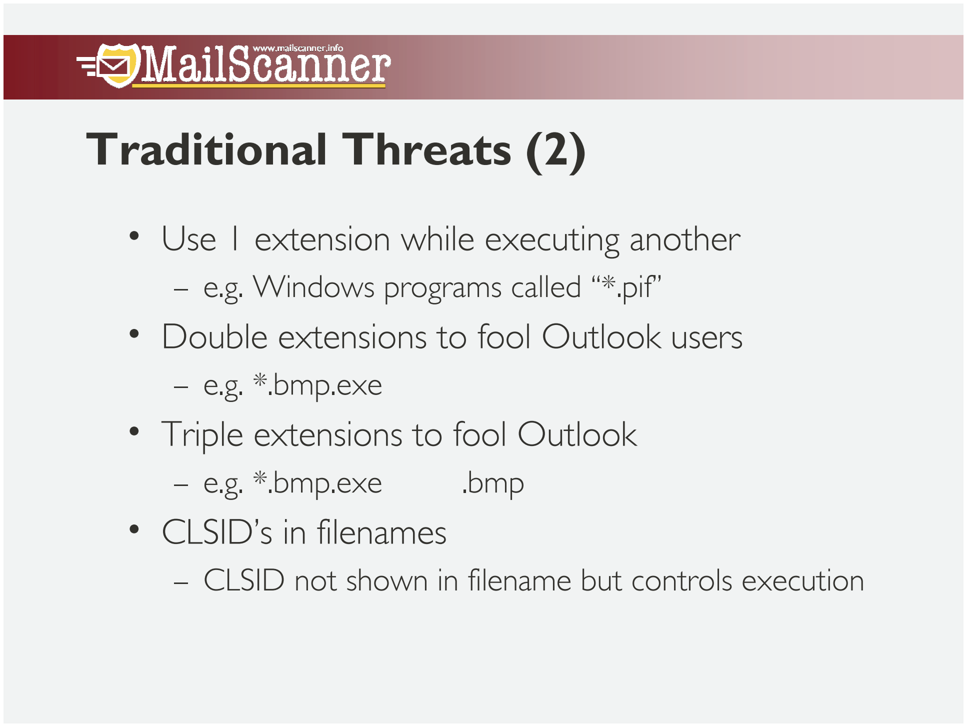
It is faster to build more complex expressions than to use a sequence of several simple expressions that achieve the same effect.



More Complex Filename Examples

- Allow files ending in 2 matching extensions, e.g. *.zip.zip
  - Allow (\.[a-z0-9]{3})\1$
- Then deny files ending in 2 extensions, eg. *.doc.scr
  - Deny \.[a-z][a-z0-9]{2,3}\s*\.[a-z0-9]{3}$

Both of these rules are enabled in the supplied filename.rules.conf file. The point of blocking files whose names appear to end in 2 filename extensions is that Windows by default will not display the last extension, despite that being the one normally used to control how the file is executed.



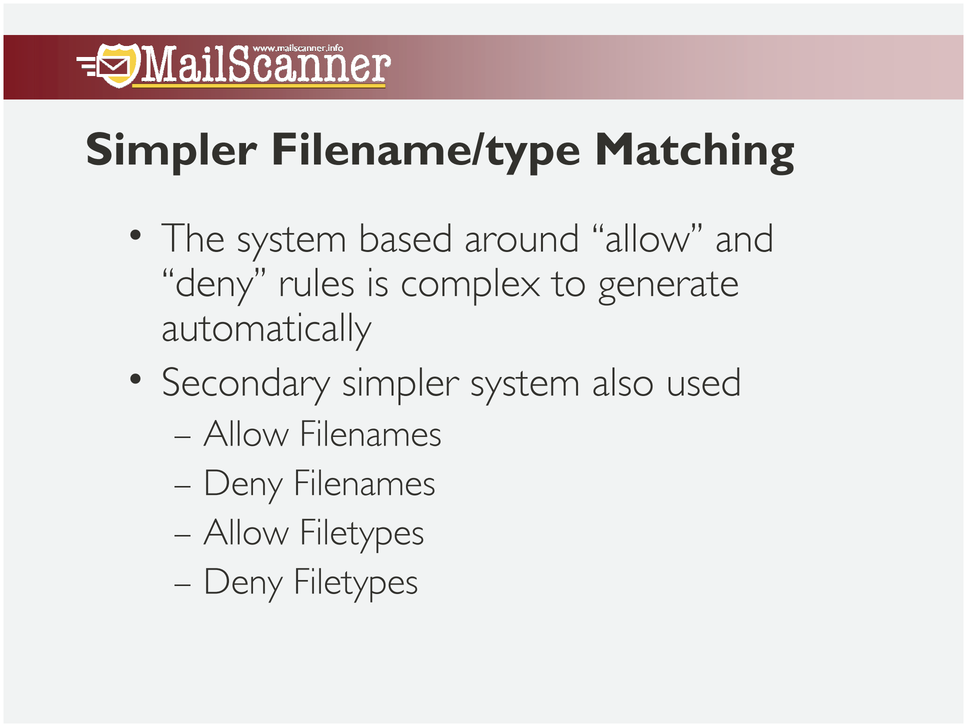
Traditional Threats (2)

- Use 1 extension while executing another
  - e.g. Windows programs called "*.pif"
- Double extensions to fool Outlook users
  - e.g. *.bmp.exe
- Triple extensions to fool Outlook
  - e.g. *.bmp.exe   .bmp
- CLSID's in filenames
  - CLSID not shown in filename but controls execution

There is a security vulnerability in Microsoft Outlook where, if there are enough spaces in the filename to place the last extension at the 256th character of the filename, then the extension used to determine the icon displayed by the file is not the last one, but the second to last. This can be used to make the file attachment not only appear to end in a harmless extension, but the icon displayed by the file will also be that of a harmless extension.

CLSID's are the internal registry keys used to determine the execution of different filetypes. Any CLSID appearing in a filename (they look like lots of hex digits surrounded with "{}" braces) will not be displayed at all by Outlook, but will still be used to determine execution of the file.

It is accepted that the system previously described based on filename.rules.conf and filetype.rules.conf is very difficult to use from a web-based user interface. In order to make it simpler to implement a web-based configuration application, there are 4 configuration options which can implement many of the common rules required.



Simpler Filename/type Matching

- The system based around "allow" and "deny" rules is complex to generate automatically
- Secondary simpler system also used
  - Allow Filenames
  - Deny Filenames
  - Allow Filetypes
  - Deny Filetypes

`Allow Filenames =`

This contains a space-separated list of regular expressions used as rules which are applied to the original fileames of attachments. If any of these rules matches, then the filename is accepted. This can also be the filename of a ruleset.

`Deny Filenames =`

This contains a space-separated list of regular expressions used as rules which are applied to the original filenames of attachments. If any of these rules matches, then the filename is not accepted, and the attachment is blocked. This can also be the filename of a ruleset.

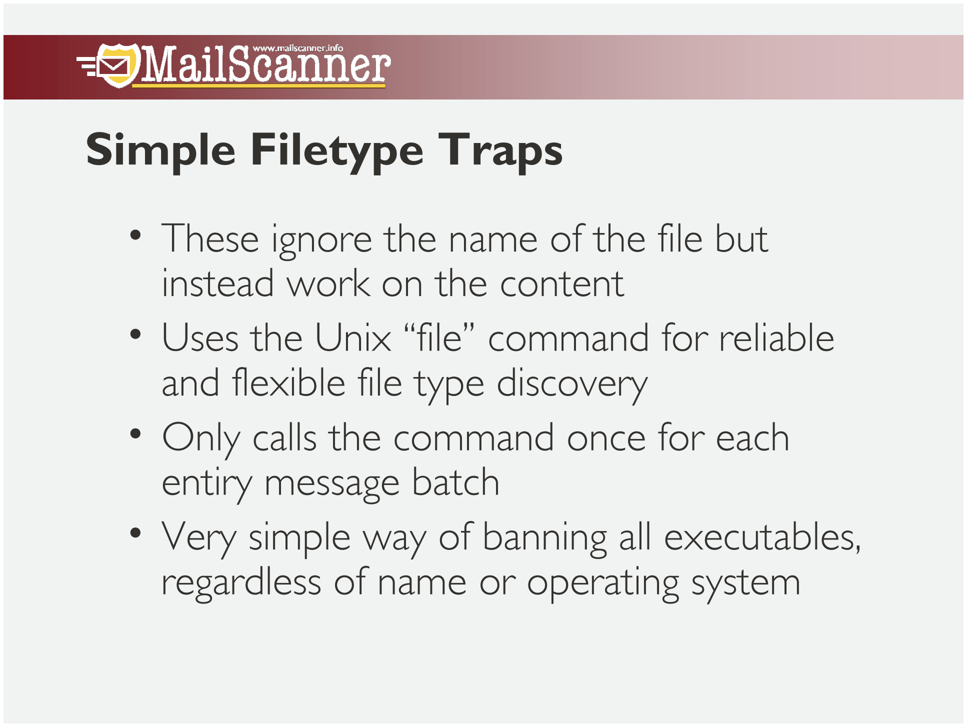
`Allow Filetypes =`

This contains a space-separated list of regular expressions. These expressions are matched against the output of the "file" command. If any of the expressions match, then the attachment is accepted and allowed to remain in the message. This can also be the filename of a ruleset.

`Deny Filetypes =`

> This contains a space-separated list of regular expressions, and is used similarly to the “Allow Filetypes” option above. If and of the expressions match, then the attachment is blocked and removed from the message. This can also be the filename of a ruleset.

The attachment must pass all four tests before it is allowed to remain in the message. If none of the regular expressions match at all, then the previous system based around “filename.rules.conf” and “filetype.rules.conf” is applied to the attachment instead, and all of those tests must pass for it to remain in the message.



Simple Filetype Traps

- These ignore the name of the file but instead work on the content
- Uses the Unix "file" command for reliable and flexible file type discovery
- Only calls the command once for each entiry message batch
- Very simple way of banning all executables, regardless of name or operating system

These are stored in filetype.rules.conf and follow a similar format as the filename.rules.conf file just described. Instead of containing a regular expression matched against the filename, the regular expression is matched against the output of the Unix "file" command. Usually these can just be substrings of the output of the "file" command.

By providing a pattern as simple as "executable", you can ban all executable files for any operating system identified by the "file" command. The GNU version of "file" has a very large database of file types and can recognise executable files for every operating system you are likely to encounter.

`Filetype Rules = %etc-dir%/filetype.rules.conf`

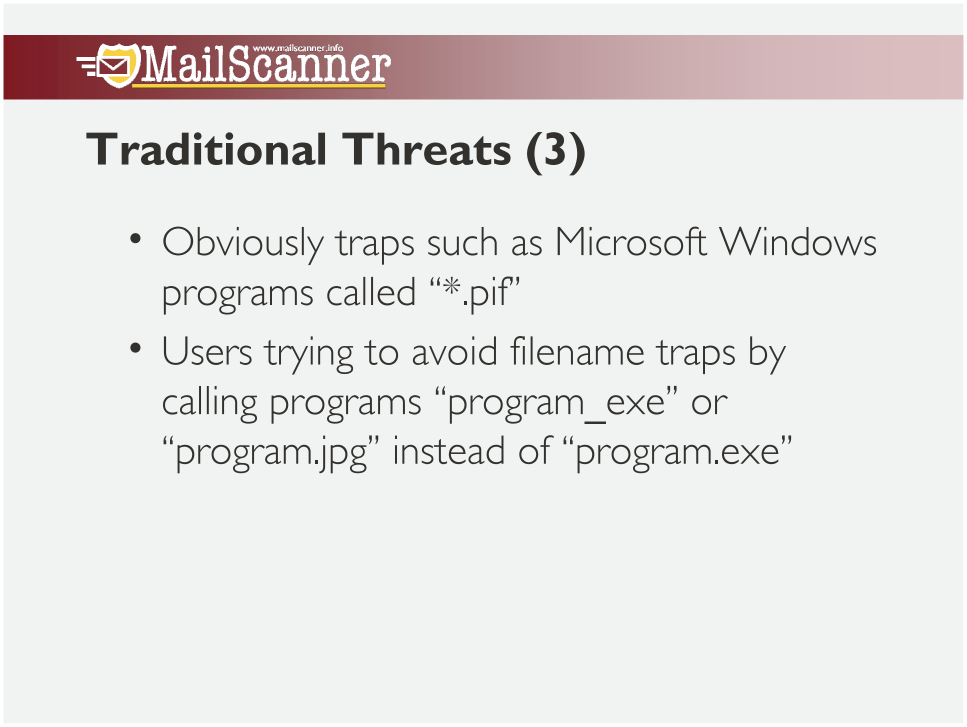
Set where to find the attachment filetype ruleset. The structure of this file is explained elsewhere, but it is used to accept or reject file attachments based on their content as determined by the "file" command, regardless of whether they are infected or not. This can also point to a ruleset, but the ruleset filename must end in ".rules" so that MailScanner can determine if the filename given is a ruleset or not! To disable this feature, set this to just "Filetype Rules =" or set the location of the file command to a blank string.

`File Timeout = 20`

The maximum length of time the "file" command is allowed to run for 1 batch of messages (in seconds)

`File Command = /usr/bin/file`

Where the "file" command is installed. This is used for checking the content type of files, regardless of their filename. To disable Filetype checking, set this value to blank.

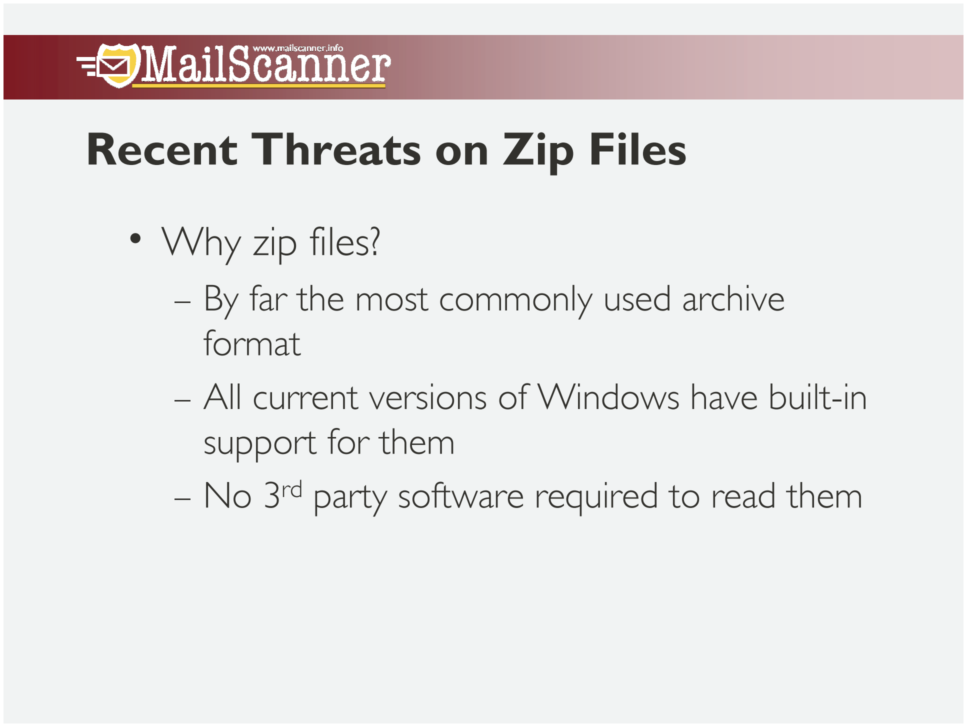


Traditional Threats (3)

- Obviously traps such as Microsoft Windows programs called “*.pif”
- Users trying to avoid filename traps by calling programs “program_exe” or “program.jpg” instead of “program.exe”

The most common exploit attempted is the use of “.pif” files that actually contain Windows executable programs. Real “.pif” files are useless without the associated executable program, but Windows recognises them and they can contain any program.

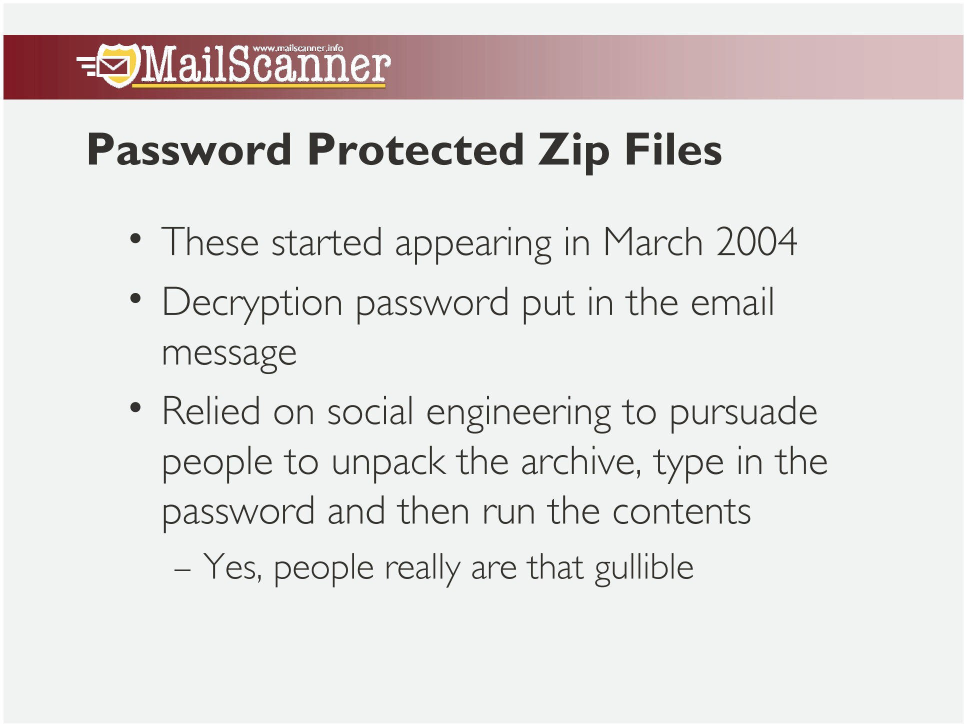
Because the check is done on the file contents and not the file name, this traps users attempting to circumvent your email policy by renaming files so they look innocent.



Recent Threats on Zip Files

- Why zip files?
  - By far the most commonly used archive format
  - All current versions of Windows have built-in support for them
  - No 3rd party software required to read them

Viruses spreading by means of zip files are one of the biggest current threats. The format is ubiquitous and Windows has had direct support for reading them since about '99. So no extra software is needed to read them and people appear to trust things put into zip files more than they trust other attachments, presumably because they assume that a human being will have been involved in putting together the zip file.


Password Protected Zip Files

- These started appearing in March 2004
- Decryption password put in the email message
- Relied on social engineering to pursuade people to unpack the archive, type in the password and then run the contents
  - Yes, people really are that gullible

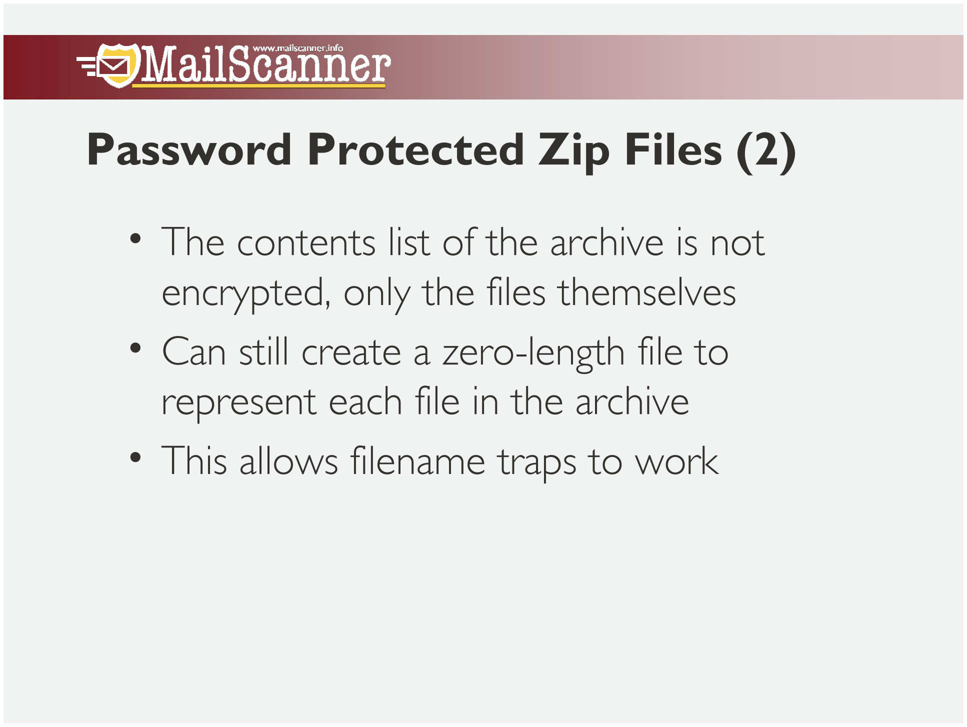
The presence of a password-protected zip file is now a very good indicator of a virus. Most people know how to create zip files, as programs like Winzip make this very easy. But few people are capable of creating a password-protected zip file.

Reliance on social engineering is an important factor in the spread of viruses and all types of malware. It appears that some people will blindly follow instructions contained in an email message, however ridiculous they may seem to someone who is familiar with these tactics. This not only results in viruses spreading but also every other type of scam relies on it, such as the bank phishing scams, Nigerian 419 fraud and many other deceptions.

Despite all the technological solutions that can be put in place, there will always be people who blindly do what they are told without any apparent common sense. This is a problem that will never go away.

`Allow Password-Protected Archives = no`

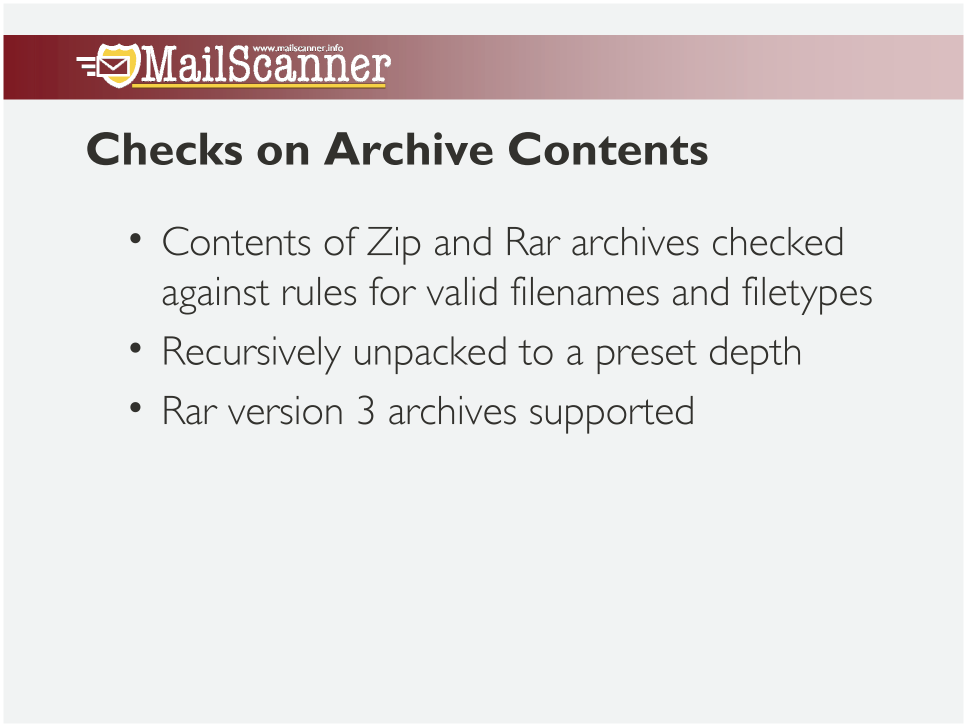
Should archives which contain any password-protected files be allowed? Leaving this set to "no" is a good way of protecting against all the protected zip files used by viruses at the moment. This can also be the filename of a ruleset.



Password Protected Zip Files (2)

- The contents list of the archive is not encrypted, only the files themselves
- Can still create a zero-length file to represent each file in the archive
- This allows filename traps to work

This is a very useful feature of zip files, which has made work considerably easier for software, though few packages actually take advantage of it. Even without the password, you can still find the names of all the files and directories within the zip file, so you can still find signs of executable programs and any potentially dangerous filenames present in the zip file.


Checks on Archive Contents

- Contents of Zip and Rar archives checked against rules for valid filenames and filetypes
- Recursively unpacked to a preset depth
- Rar version 3 archives supported

In order to apply filename and filetype checks on the contents of Zip, Rar and UU-encoded archives, they are all unpacked. The Rar archives are unpacked using an external "unrar" program, while Zip and UU-encoded archives are handled internally. Any archive found nested deeper than the “Maximum Archive Depth” will make MailScanner reject the message.

In order to stop MailScanner checking the filenames and filetypes of the contents of Zip and Rar archives, the "Maximum Archive Depth" option must be set to 0. Setting this to 0 stops MailScanner unpacking Zip and Rar archives itself. Virus scanners do their own archive unpacking, and so setting this to 0 does **not** stop MailScanner finding viruses in archives.

There are 2 common settings for this option:

- 0 stops MailScanner unpacking Zip and Rar archives. Viruses inside Zip and Rar archives will still be found. No filename checking or filetype checking will be done on the contents of Zip and Rar archives.

- 3 makes MailScanner check the filenames and filetypes of files within archives within archives within the message, which is as deeply nested as any average user will create. Any archive nested deeper than this will cause MailScanner to reject the attachment as a potential denial-of-service attack and it will be replaced by a warning message.

`Unrar Command = /usr/bin/unrar`

If this is set, and the unrar program exists, it will be automatically passed to the ClamAV scanner. No wrapper script adjustments are necessary for unrar to work.

`Unrar Timeout = 50`

The unrar command above will be used when unpacking archives to expand all RAR archives into their original contents. This specifies the maximum amount of time that can be used for the process, after which the unrar command will be terminated.

`Gunzip Command = /usr/bin/gunzip`

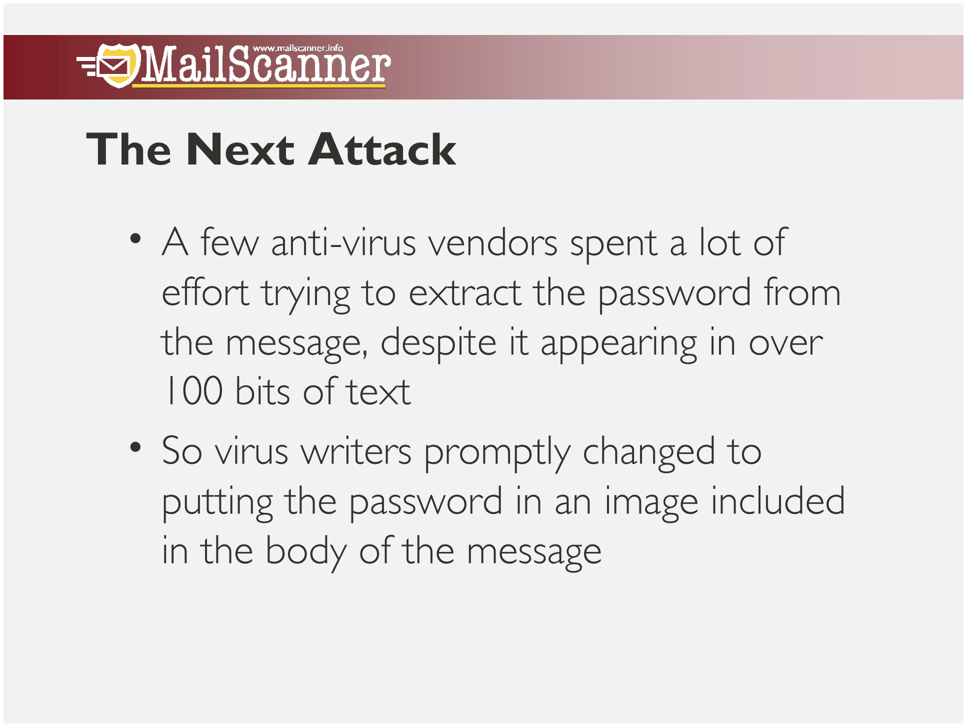
If this is set, and the gunzip program exists, it will be used to decompress gzip-ed files so that the filetype and file content checks can be applied to the contents.

`Gunzip Timeout = 50`

This specifies the maximum amount of time that can be used by any call to the "gunzip" program, after which it will be terminated.

`Find UU-Encoded Files = no`

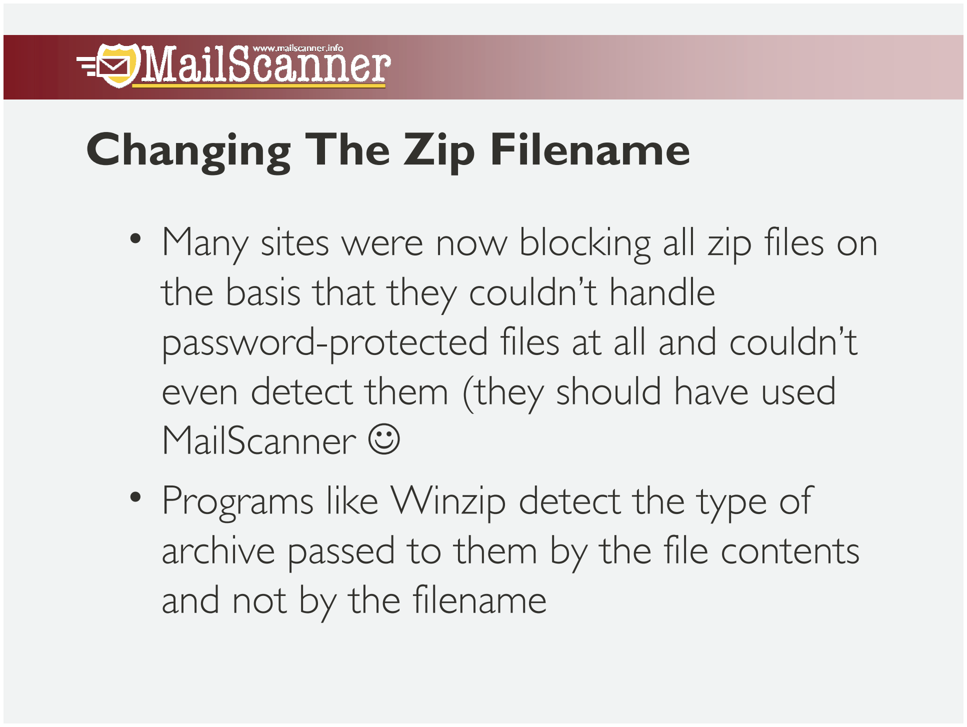
Occasionally files are stored in a form known as UU-encoded, and this format can be unpacked by some of the common Zip-file extractor applications.



The Next Attack

- A few anti-virus vendors spent a lot of effort trying to extract the password from the message, despite it appearing in over 100 bits of text
- So virus writers promptly changed to putting the password in an image included in the body of the message

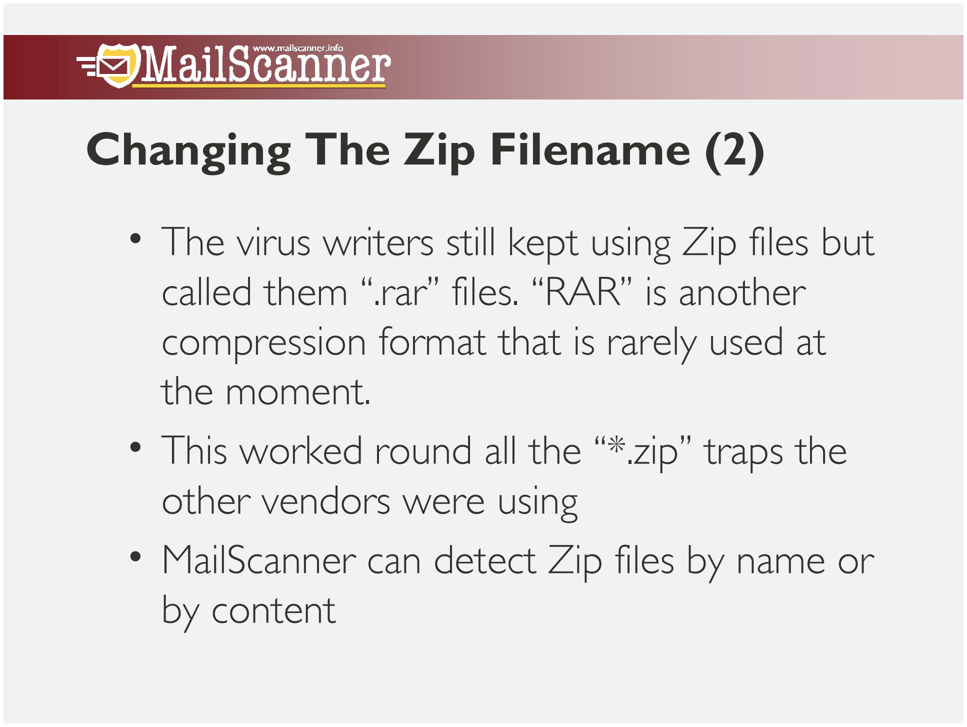
There are signs that Kaspersky went as far as producing an OCR system to read the passwords stored in images inserted into the email message. They are one of the few vendors that have attempted to extract the password from the message. Unfortunately this is not a long-term solution as the virus writers can easily add a thousand new ways of hiding the password in every new variant they produce.



Changing The Zip Filename

- Many sites were now blocking all zip files on the basis that they couldn't handle password-protected files at all and couldn't even detect them (they should have used MailScanner ☺
- Programs like Winzip detect the type of archive passed to them by the file contents and not by the filename

There was a week in March when virtually all sites were blocking all zip files completely while the anti-virus vendors fought to keep up with the virus writers. Many commercial systems still rely on the filename to find zip files. MailScanner looks for the archives by checking the contents of the files as well. It cannot be easily fooled by renaming the file from "nasty-virus.exe" to "nasty-virus_exe", unlike most commercial systems.



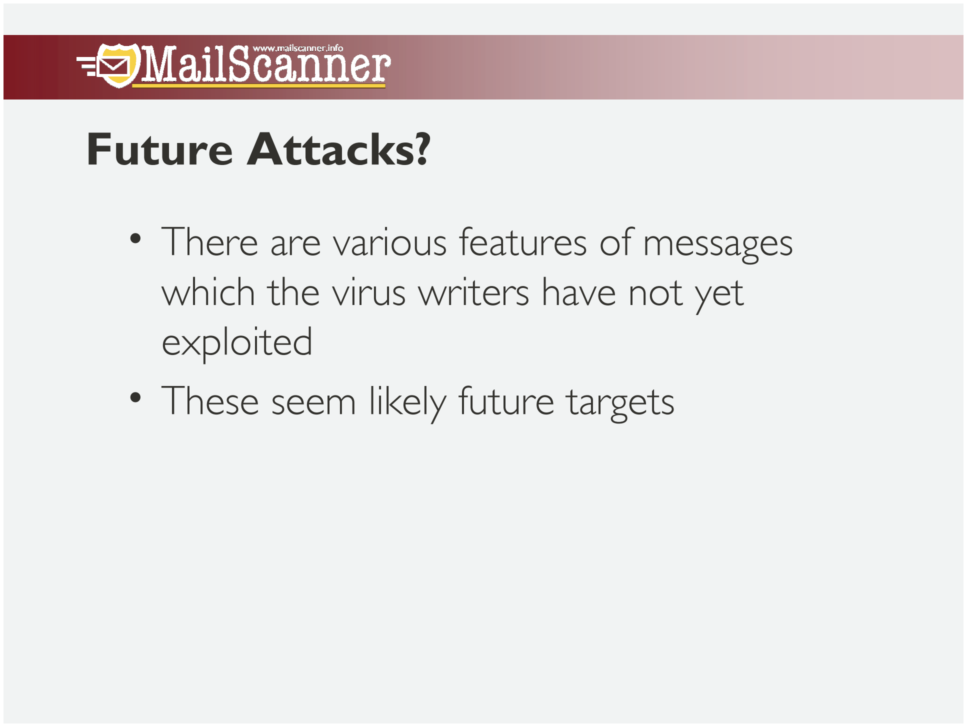
Changing The Zip Filename (2)

- The virus writers still kept using Zip files but called them ".rar" files. "RAR" is another compression format that is rarely used at the moment.
- This worked round all the "*.zip" traps the other vendors were using
- MailScanner can detect Zip files by name or by content

If you open a file whose filename is tied to Winzip or WinRar, it will attempt to open it regardless of what the filename says. But by changing the filename extension from ".zip" to ".rar" any system relying on the filename will fail to open it. This is a tactic used a lot at the moment, which is strong evidence that the tactic is still working.

`Find Archives By Content = yes`

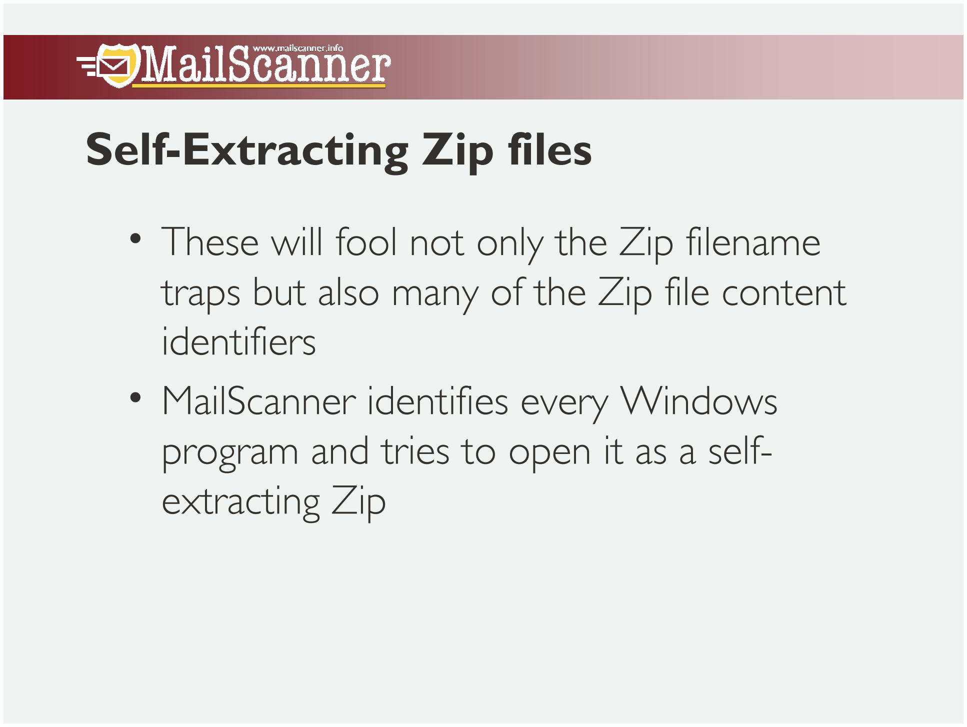
Find zip archives by filename or by file contents? Finding them by content is a far more reliable way of finding them, but it does mean that you cannot tell your users to avoid zip file checking by renaming the file from ".zip" to "_zip" and tricks like that. Only set this to no (i.e. check by filename only) if you don't want to reliably check the contents of zip files. Note this does not affect virus checking, but it will affect all the other checks done on the contents of the zip file. This can also be the filename of a ruleset.



www.mailscanner.info
MailScanner

Future Attacks?

- There are various features of messages which the virus writers have not yet exploited
- These seem likely future targets



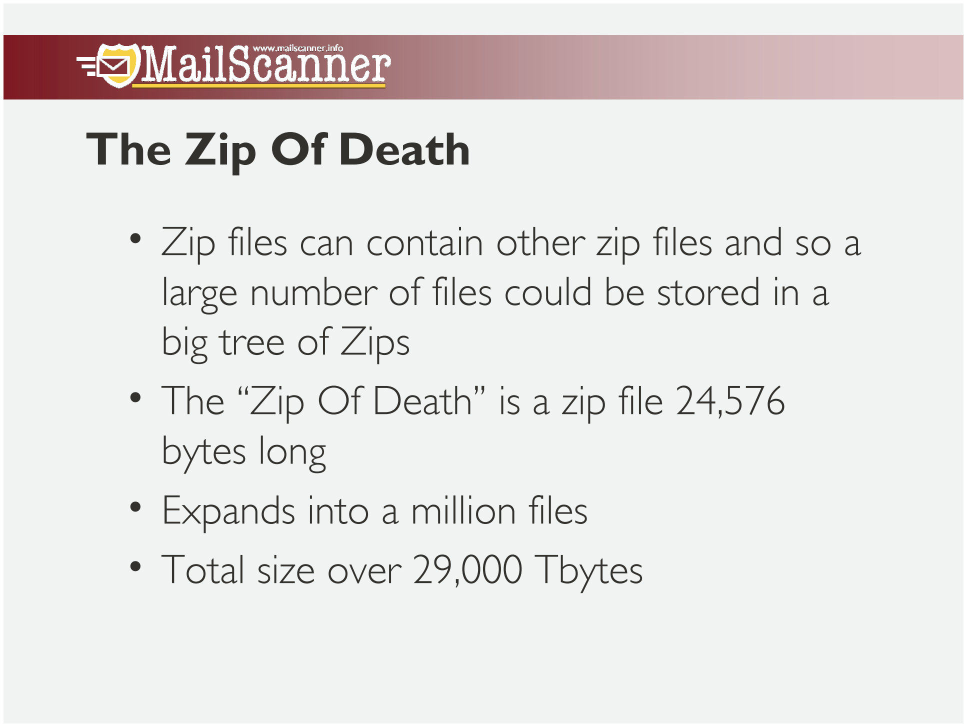
Self-Extracting Zip files

- These will fool not only the Zip filename traps but also many of the Zip file content identifiers
- MailScanner identifies every Windows program and tries to open it as a self-extracting Zip

Any system that is identifying zip files by content will tend to look at the first few bytes of the file and check the file "signature" of a zip file.

Using self-extracting zip files will defeat most zip file detectors as they look more like Windows/DOS executables than zip files.

This will be a very easy way for the virus writers to avoid detection, but will only work against sites and users that allow the transmission of Windows executables by email. Most corporations do not allow this, but most home user service providers do not enforce such restrictions.



MailScanner www.mailscanner.info

The Zip Of Death

- Zip files can contain other zip files and so a large number of files could be stored in a big tree of Zips
- The “Zip Of Death” is a zip file 24,576 bytes long
- Expands into a million files
- Total size over 29,000 Tbytes

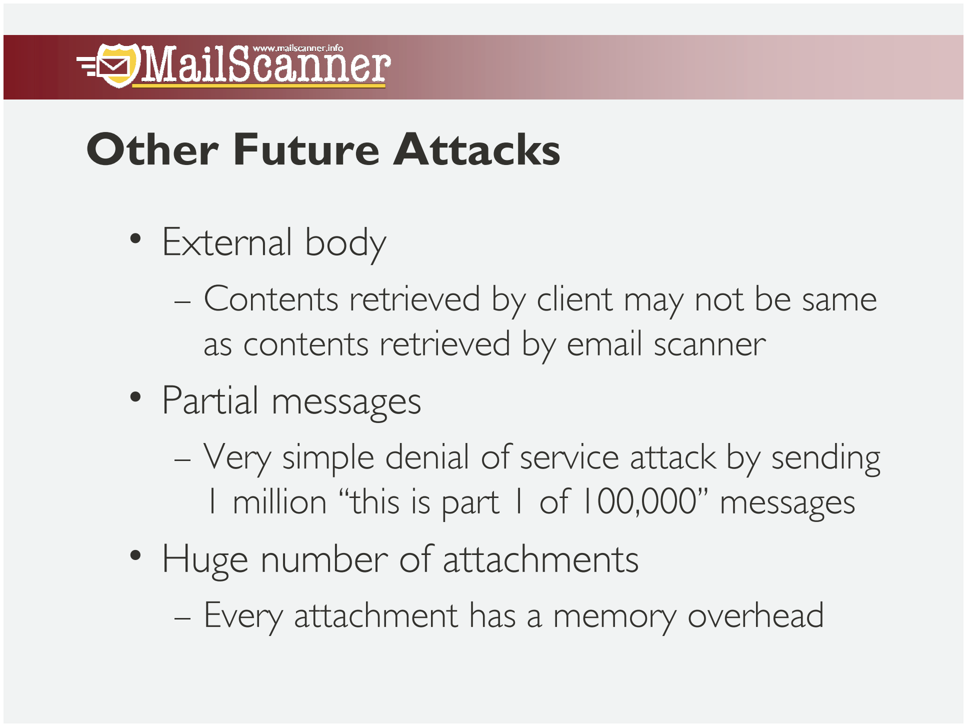
This is highly effective as a Denial of Service attack against many mail systems. MailScanner has configuration options to limit the effectiveness of this attack.

`Maximum Archive Depth = 2`

The maximum depth to which zip archives will be unpacked, to allow for checking filenames and filetypes within zip archives. To disable this feature set this to 0. A common useful setting is this option = 0, and Allow Password-Protected Archives = no. That block password-protected archives but does not do any filename/filetype checks on the files within the archive.

`Virus Scanner Timeout = 300`

The maximum length of time the commercial virus scanner is allowed to run for 1 batch of messages (in seconds).


Other Future Attacks

- External body
  - Contents retrieved by client may not be same as contents retrieved by email scanner
- Partial messages
  - Very simple denial of service attack by sending 1 million "this is part 1 of 100,000" messages
- Huge number of attachments
  - Every attachment has a memory overhead

The only people using external message bodies at the moment is the IETF (Internet Engineering Task Force). They send messages which have an external body rather than just including a link to a web page which is how the rest of the world achieve the same result. The external body is retrieved by a few different methods such as anonymous FTP or even by email. This standard pre-dates the web and will hopefully be deprecated in a few years time.

Some ISP's have a very small maximum message size, far smaller than the mail storage space they give to each user. I have never figured out quite why. The result is that people split up attachments into several messages, as this can automatically be done by the most popular email applications.

No normal message should need more than 200 attachments. I have yet to see a genuine use which the user could not achieve better by using zip files to create a single archive file.

```
Allow External Message Bodies = no
```

Do you want to allow messages whose body is stored somewhere else on the internet, which is downloaded separately by the user's email package? There is no way to guarantee that the file fetched by the user's email package is free from viruses, as MailScanner never sees it. This feature is dangerous as it can allow viruses to be fetched from other Internet sites by a user's email package. The user would just think it was a normal email attachment and would have been scanned by MailScanner. It is only currently supported by Netscape 6 anyway, and the only people who it are the IETF. So I would strongly advise leaving this switched off. This can also be the filename of a ruleset.

`Allow Partial Messages = no`

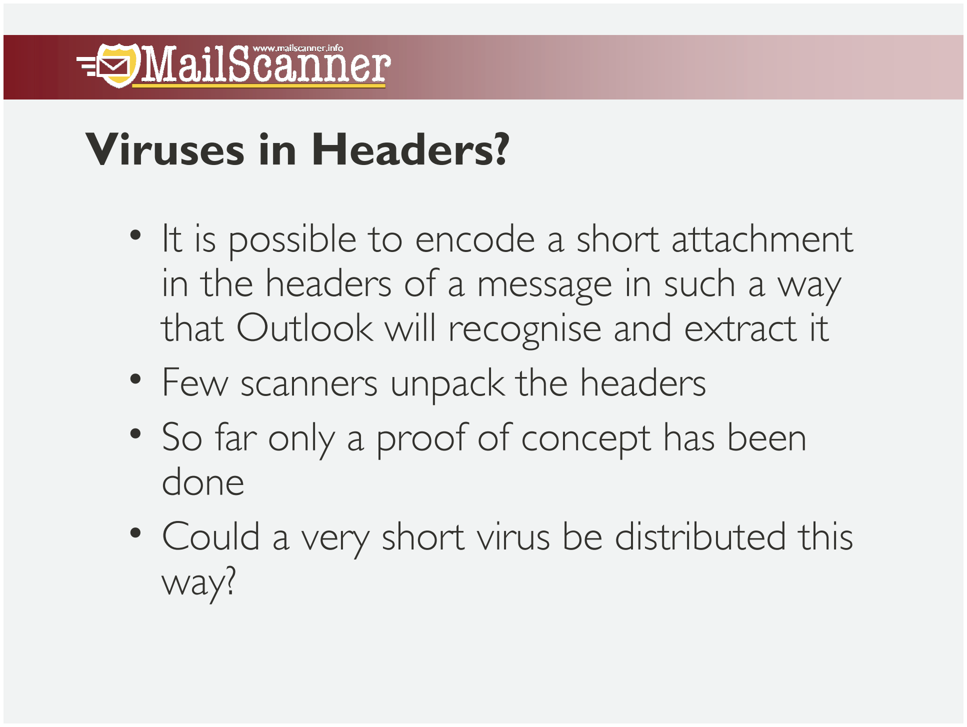
Do you want to allow partial messages, which only contain a fraction of the attachments, not the whole thing? There is absolutely no way to scan these "partial messages" properly for viruses, as MailScanner never sees all of the attachment at the same time. Enabling this option can allow viruses through. You have been warned. This can also be the filename of a ruleset so you can, for example, allow them in outgoing mail but not in incoming mail.

`Maximum Attachments = 200`

The maximum number of attachments allowed in a message before it is considered to be an error. Some email systems, if bouncing a message between 2 addresses repeatedly, add information about each bounce as an attachment, creating a message with thousands of attachments in just a few minutes. This can slow down or even stop MailScanner as it uses all available memory to unpack these thousands of attachments. This can also be the filename of a ruleset.

5.6

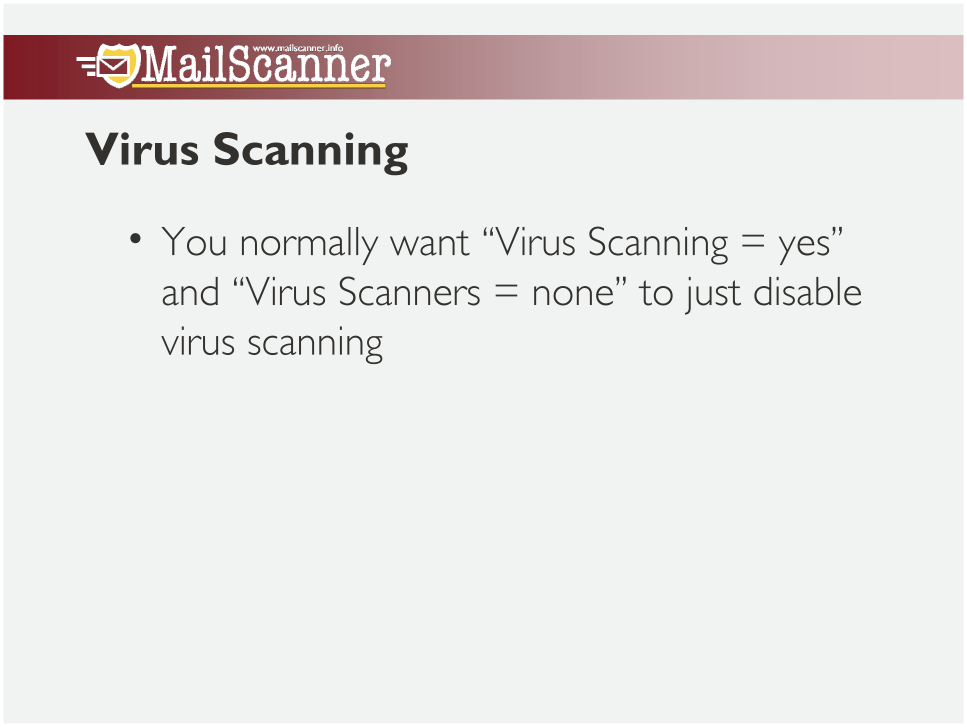
Virus Scanning



Viruses in Headers?

- It is possible to encode a short attachment in the headers of a message in such a way that Outlook will recognise and extract it
- Few scanners unpack the headers
- So far only a proof of concept has been done
- Could a very short virus be distributed this way?

This is all to do with end-of-line sequences. You can make a header contain multiple lines by using ^M as the line separator. It is not a valid end-of-line sequence in a header, and so MTA’s pass it intact. However, Outlook will treat the ^M as a valid end-of-line sequence, creating a multi-line block of text. This can then be used to embed an attachment.



Virus Scanning

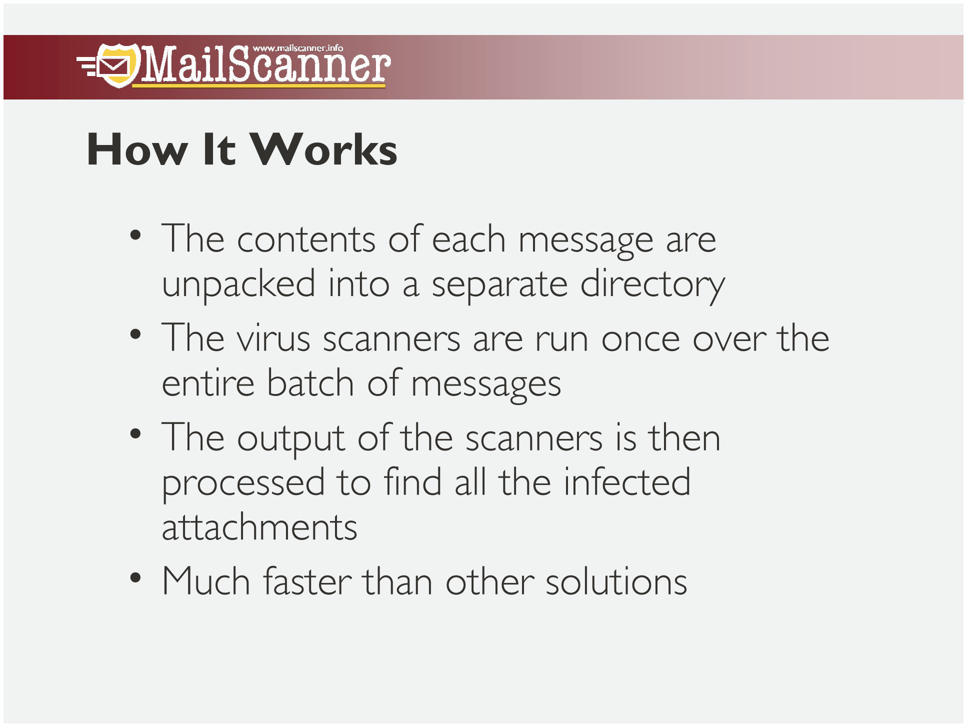
- You normally want “Virus Scanning = yes” and “Virus Scanners = none” to just disable virus scanning

Sorry for this confusion, but it is for good historical reasons and I cannot change it without forcing configuration changes onto thousands of sites, which will confuse them more than leaving it in its present state.

The fact is well documented in the MailScanner.conf file.

`Virus Scanning = yes`

Do you want to scan email for viruses? A few people don't have a virus scanner licence and so want to disable all the virus scanning. NOTE: This switch actually switches on/off all processing of the email messages. If you just want to switch off actual virus scanning, then set "Virus Scanners = none" instead. If you want to be able to switch scanning on/off for different users or different domains, set this to the filename of a ruleset. This can also be the filename of a ruleset.



How It Works

- The contents of each message are unpacked into a separate directory
- The virus scanners are run once over the entire batch of messages
- The output of the scanners is then processed to find all the infected attachments
- Much faster than other solutions

As the scanners are run over a whole batch of messages at once, the per-message overhead of running the scanner decreases with larger batches of messages. Heavily loaded servers will run with larger batches than lightly loaded servers, so the efficiency of the scanning process increases with system load.

Most other systems scan each message individually, which is a lot slower than my approach as the scanners' overheads are a much bigger factor.

`Virus Scanners = auto`

Which Virus Scanning packages to use:

auto	Use all available installed virus scanners
sophos	From www.sophos.com
sophossavi	(also from www.sophos.com, using the SAVI perl module)
mcafee	From www.mcafee.com

command	From www.command.co.uk
bitdefender	From www.bitdefender.com
drweb	From www.dials.ru/english/dsav_toolkit/drwebunix.htm
kaspersky-4.5	From www.kaspersky.com
kaspersky	From www.kaspersky.com
kavdaemonclient	From www.kaspersky.com
etrust	From http://www3.ca.com/Solutions/Product.asp?ID=156
inoculate	From www.cai.com/products/inoculateit.htm
inoculan	From ftp.ca.com/pub/getbbs/linux.eng/inoctar.LINUX.Z
nod32	From www.nod32.com
nod32-1.99	From www.nod32.com
f-secure	From www.f-secure.com
f-prot	From www.f-prot.com
panda	From www.pandasoftware.com
rav	From www.ravantivirus.com
antivir	From www.antivir.de
clamav	From www.clamav.net
clamavmodule	(also from www.clamav.net using the ClamAV perl module)
clamd	(also from www.clamav.net using the clamd daemon)
trend	From www.trendmicro.com
norman	From www.norman.de
css	From www.symantec.com

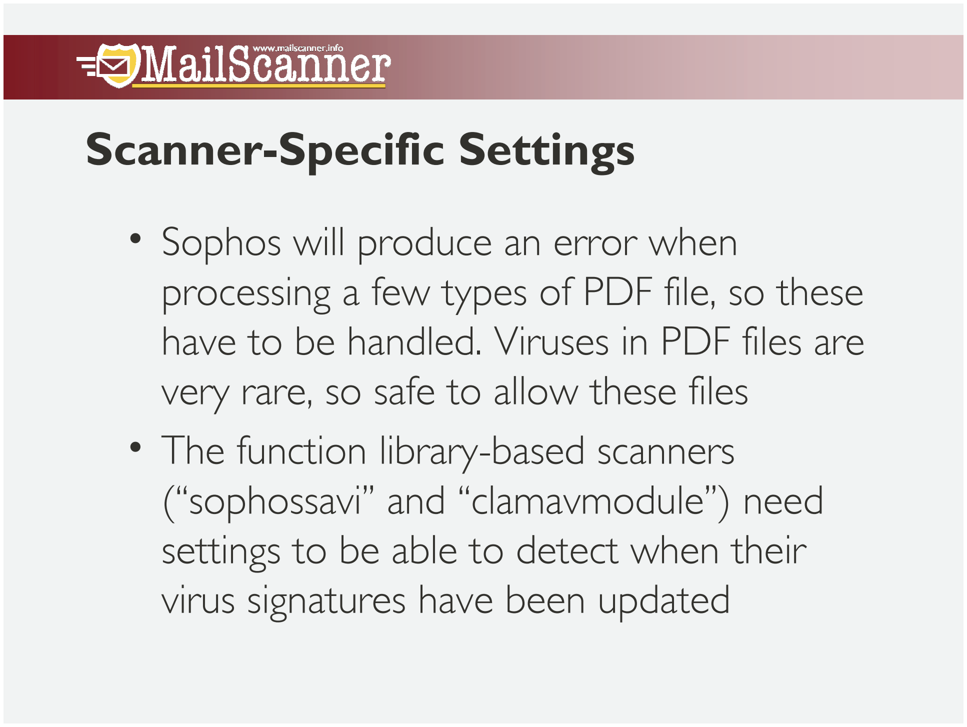
avg	From www.grisoft.com
vexira	From www.centralcommand.com
symscanengine	From www.symantec.com (Symantec Scan Engine)
avast	From www.avast.com
avastd	(also from www.avast.com using the avast daemon)
generic	Write your own, see generic-wrapper for instructions
None	(no virus scanning at all)

Note for McAfee users: do not use any symlinks with McAfee at all. It is very strange but may not detect all viruses when started from a symlink or scanning a directory path including symlinks.

Note: If you want to use multiple virus scanners, then this should be a space-separated list of virus scanners. For example:

Virus Scanners = sophos f-prot mcafee

Note: Make sure that you check that the base installation directory in the 3rd column of virus.scanners.conf matches the location you have installed each of your virus scanners. The supplied virus.scanners.conf file assumes the default installation locations recommended by each of the virus scanner installation guides.



Scanner-Specific Settings

- Sophos will produce an error when processing a few types of PDF file, so these have to be handled. Viruses in PDF files are very rare, so safe to allow these files
- The function library-based scanners ("sophossavi" and "clamavmodule") need settings to be able to detect when their virus signatures have been updated

Sophos have been working on handling PDF files for a long time. They still have occasional problems with PDF files generated by programs other than Acrobat. PDF viruses are almost unheard of, so simply allowing them to pass is quite safe.

MailScanner does not normally support daemon-based scanners as it adds a reliance on the daemon working properly. However, library-based scanners are much more reliable and are a very good way of avoiding the startup overhead of a command-line scanner. This improves performance on lightly-loaded servers but does not really affect heavily-loaded servers as much.

`Allowed Sophos Error Messages =`

Anything on the next line that appears in brackets at the end of a line of output from Sophos will cause the error/infection to be ignored. Use of this option is dangerous, and should only be used if you are having trouble with lots of corrupt PDF files, for example. If you need to specify more than 1 string to find in the error message, then put each string in quotes and separate them with a comma. For example:

`Allowed Sophos Error Messages = "corrupt", "format not supported"`

Sophos IDE Dir = /usr/local/Sophos/ide

The directory (or a link to it) containing all the Sophos *.ide files. This is used by the "sophossavi" and "sophos" virus scanners, and is irrelevant for all other scanners.

`Sophos Lib Dir = /usr/local/Sophos/lib`

The directory (or a link to it) containing all the Sophos *.so libraries. This is only used by the "sophossavi" virus scanner, and is irrelevant for all other scanners.

`Monitors for Sophos Updates = /usr/local/Sophos/ide/*ides.zip`

SophosSAVI only: monitor each of these files for changes in size to detect when a Sophos update has happened. The date of the Sophos Lib Dir is also monitored. This is only used by the "sophossavi" virus scanner, not the "sophos" scanner setting.

`Monitors for ClamAV Updates = /usr/local/share/clamav/*.cvd`

ClamAVModule only: monitor each of these files for changes in size to detect when a ClamAV update has happened. This is only used by the "clamavmodule" virus scanner, not the "clamav" scanner setting.

`ClamAVModule Maximum Recursion Level = 5`

ClamAVModule only: the maximum depth to which archives will be unpacked. If you set this too high you are inviting denial-of-service attacks, but you should set it deep enough that normal users will not pack archives deeper than this setting.

`ClamAVModule Maximum Files = 1000`

ClamAVModule only: the max number of files that can be scanned at once.

`ClamAVModule Maximum File Size = 10000000`

ClamAVModule only: the maximum size of any file that can be scanned.

`ClamAVModule Maximum Compression Ratio = 250`

ClamAVModule only: the maximum compression ratio of any archive scanned. Any archives which expand more than this amount will be flagged as errors and will be quarantined.

`Unrar Command = /usr/bin/unrar`

If this is set, and the unrar program exists, it will be automatically passed to the ClamAV scanner. No wrapper script adjustments are necessary for unrar to work.

`Unrar Timeout = 50`

The unrar command above will be used when unpacking archives to expand all RAR archives into their original contents. This specifies the maximum amount of time that can be used for the process, after which the unrar command will be terminated.

`Clamd Port = 3310`

This is the TCP/IP port number of the socket used by the clamd virus-scanning daemon when communicating via TCP. It should match the value set in /etc/clamd.conf.

`Clamd Socket = /tmp/clamd`

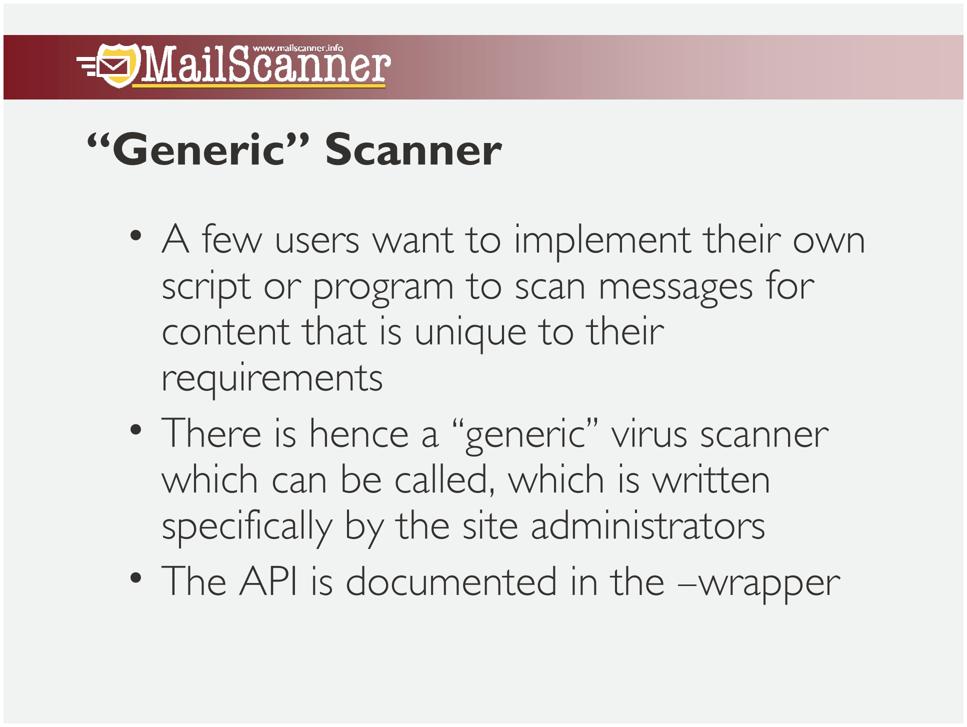
This is the filename of the Unix domain socket used by clamd, or the IP address which the daemon is listening on. It should match the value set in /etc/clamd.conf.

`Clamd Lock File =`

This is the filename of the lock file created by the Linux clamd init.d script used to start and stop the clamd daemon.

`Clamd Use Threads = no`

If you are running on a system with more than 1 CPU, or more than 1 CPU core, then this value should be set to “yes” so that files are scanned in parallel. If you are only using a single CPU core then it should be left at “no”.



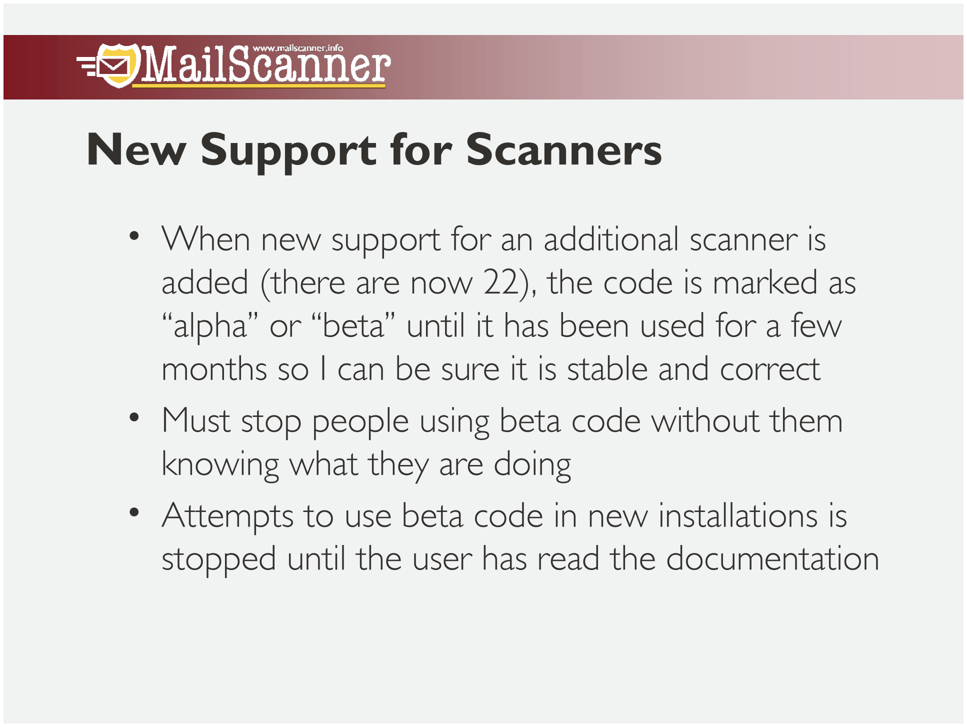
"Generic" Scanner

- A few users want to implement their own script or program to scan messages for content that is unique to their requirements
- There is hence a "generic" virus scanner which can be called, which is written specifically by the site administrators
- The API is documented in the –wrapper

For the "generic" scanner, I supply

1. output parser
2. –wrapper script
3. –autoupdate script

The output format required from your scanner program is very simple, and is documented in the –wrapper script. You may well not need to use the –autoupdate script at all. This will enable you to provide reports about particular attachments or the whole message and label the message as being "infected" in some way.



New Support for Scanners

- When new support for an additional scanner is added (there are now 22), the code is marked as “alpha” or “beta” until it has been used for a few months so I can be sure it is stable and correct
- Must stop people using beta code without them knowing what they are doing
- Attempts to use beta code in new installations is stopped until the user has read the documentation

This has to be the most confusing feature ever added to MailScanner (it wasn’t my idea, and I strongly voted against it!). If you break the “Minimum Code Status” setting, the error message produced is very easy to understand, and specifically asks you to read a web page that explains all about it. But I still get site administrators contacting me saying “I got this error message, what should I do?” My usual response is “Did you do what the error message told you to do?”

`Minimum Code Status = supported`

Minimum acceptable code stability status -- if we come across code that's not at least as stable as this, we barf. This is currently only used to check that you don't end up using untested virus scanner support code without realising it.
Levels used are:

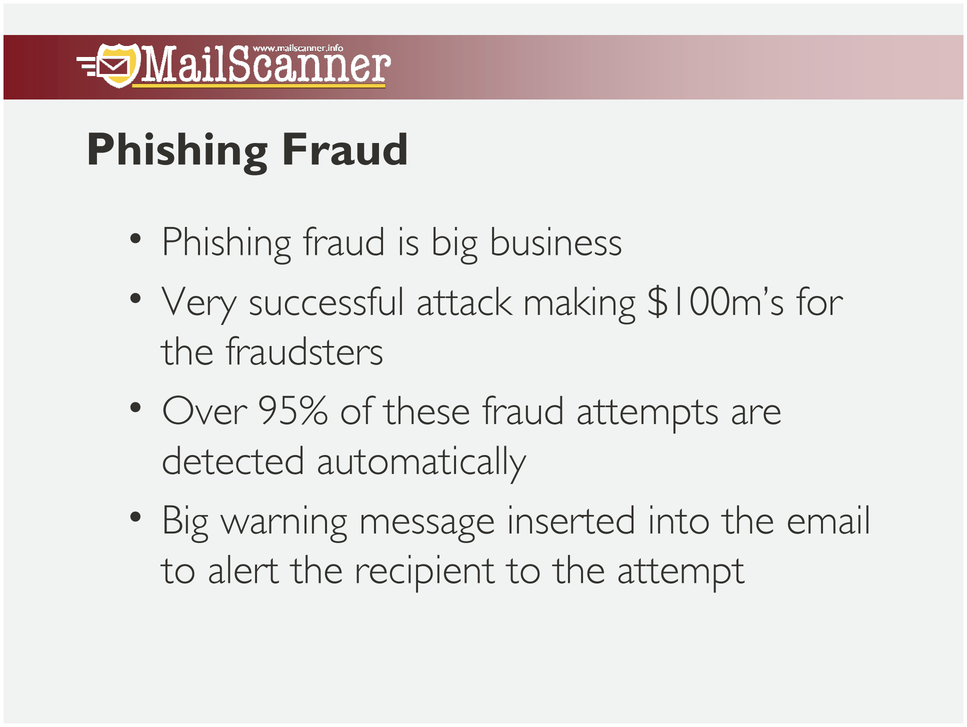
none	there may not even be any code.
unsupported	code may be completely untested, a contributed dirty hack, anything, really.
alpha	code is pretty well untested. Don't assume it will work.

beta	code is tested a bit. It should work.
supported	code should be reliable.

Do not consider setting this to anything other than "beta" or "supported" on a system that receives real mail until you have tested it yourself and are happy that it is all working as you expect it to. Do not set it to anything other than "supported" on a system that could ever receive important mail.

5.7

HTML Checks


Phishing Fraud

- Phishing fraud is big business
- Very successful attack making $100m’s for the fraudsters
- Over 95% of these fraud attempts are detected automatically
- Big warning message inserted into the email to alert the recipient to the attempt

"Phishing" is a term coined by hackers. It is the use of email messages in which there is a link you are encouraged to click which appears to link to your bank or credit card company’s web site; in reality it takes you to a fraudulent realistic copy of the site. The aim is to lure you into entering confidential information, including items such as

- Bank account numbers
- Internet banking passwords
- PIN numbers
- Account details
- Your mother's maiden name

This information is then used by the fraudsters to empty your bank account, duplicate your credit card or acquire large loans in your name.

It is estimated to have costed US banks and credit card companies well over $100m in 2004. By using viruses to infect PC's, the fraudsters then gain control over a large number of PC's and use these "zombie" machines to send out huge numbers of the phishing emails, making them impossible to trace. Fortunately the global internet community, in particular the banks, have so far been quite efficient at tracing the fake copies of their websites and quickly having them shut down.

`Find Phishing Fraud = yes`

MailScanner can detect most of the phishing fraud attempts, and will place a large warning in the email message, right next to the potentially fake link.

The text added into the message is set in 2 settings in the languages.conf file, one placed before the real address of the link, and the other placed before the displayed address of the link:

```
PossibleFraudStart = <font color="red"><b>MailScanner has detected a
possible fraud attempt from
PossibleFraudEnd = claiming to be</b></font>
```

The actions that can be taken are intentionally quite minor in their effect, as some companies persist in issuing email messages to their customers containing links that look like phishing fraud attacks. As a result this trap does occasionally trigger with a false alarm, but that cannot be totally avoided. Steps have been taken to adapt to all the cases of false alarms that have been reported, and the heuristic rules used to detect these are now very complex and reliable.

I am not going to attempt to describe here exactly how the heuristics work, they are far too complex to easily explain. But if you see some messages which are mistakenly triggering the "phishing net" then please send them to support@mailscanner.info for analysis.

The heuristics used by the phishing net are given online at http://www.phishingnet.info.

`Use Stricter Phishing Net = yes`

If this is set to “no”, then the website is checked only to ensure it belongs to the same company or organisation. So “lists.company.com” and “www.company.com” will be seen to match, along with international variations such as “lists.company.co.uk” and “www.company.co.uk”. This is done using a comprehensive list of all the international top level and second level domains (and a few third level domains), so that the name of the organisation can be extracted from the middle of the hostname in the URL.

`Country Sub-Domains List = %etc-dir%/country.domains.conf`

This is the list of country domain names used by the less strict phishing net. It should never need modifying. If you find changes are needed, please ensure the authors of MailScanner are notified of the changes.

`Highlight Phishing Fraud = yes`

This setting causes the warning, described above, to be added to the message. The message is checked to determine if the warning is already present, and it is not added a second time. This is very useful when you have multiple MailScanner servers in the path of a message.

`Phishing Modify Subject = no`
`Phishing Subject Text = {Fraud?}`

This setting causes the Subject: line of the message to be modified when a phishing trap is found. The text is placed at the start of the Subject: line, followed by a space, followed by the original Subject: line. The value of "Phishing Modify Subject" can be "no", "yes", "start" or "end" in the same way as the other "Modify Subject" parameters. "yes" is equivalent to "start" and causes the "Phishing Subject Text" to be added to the start of the Subject line.

`Also Find Numeric Phishing = yes`

Many phishing attacks rely on the use of numeric IP addresses, as the servers used to host the fake sites are often run on systems that have been automatically hacked into and had the fake web site copied onto them. In order not to give away the name, the fraudsters just use the numeric IP address of the server. It is rare for a real website to use a numeric IP address in a valid link, and so all uses of numeric IP addresses in links are flagged as possible phishing attacks.

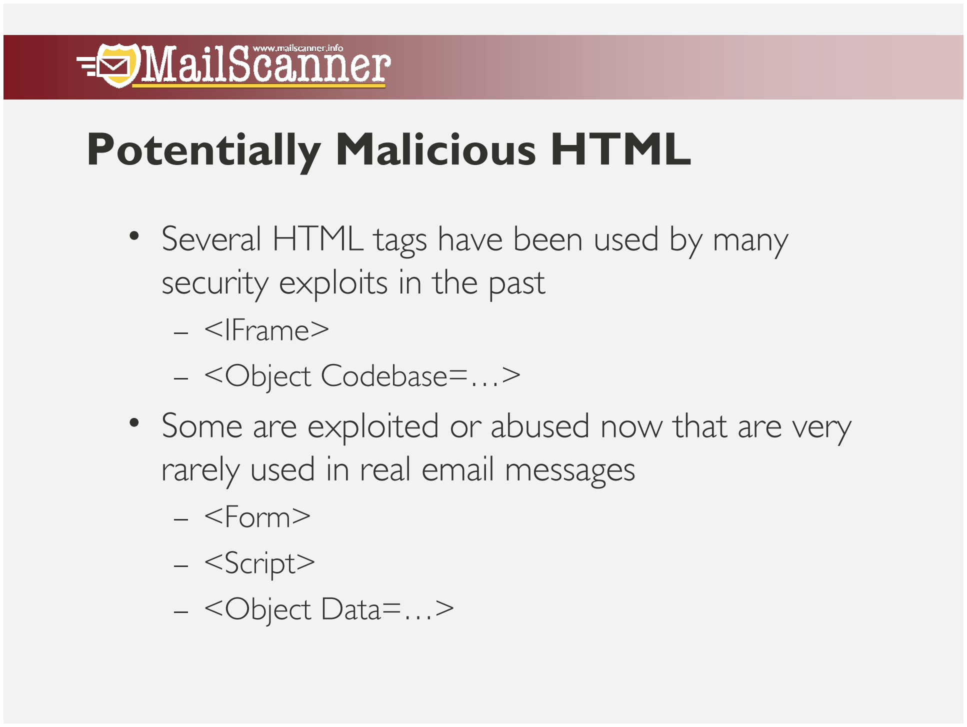
`Phishing Safe Sites File = %etc-dir%/phishing.safe.sites.conf`

This file contains the list of web sites which are known to be safe and whose owners send out newsletters and other email messages containing links that look like phishing attacks. The hosts listed in this file are the real hostnames of the sites mentioned in the email message. A leading "*" wildcard character may be used in the entry to represent any hostname in the given domain. For example, "*.example.com" would match both "www.example.com" and "mail.example.com". The entries must each be put on a separate line. Comments start with a "#" character and continue to the end of the line. Blank lines are ignored.

This file is automatically updated every night on Linux RPM-based systems. The contents of the master copy of the file is merged with your local copy of the file in such a way that your own additions and changes are kept during the merge. It is not a simple over-write of the file. The master copy of the file is held at

http://www.mailscanner.info/phishing.safe.sites.conf.master

The update is done by executing the script update_phishing_sites, which on RPM-based Linux installations is place in the /etc/cron.daily directory so that it is executed once per day.



Potentially Malicious HTML

- Several HTML tags have been used by many security exploits in the past
  - <IFrame>
  - <Object Codebase=…>
- Some are exploited or abused now that are very rarely used in real email messages
  - <Form>
  - <Script>
  - <Object Data=…>

A few HTML tags are recognised as they have been used in so many attacks over the years. They are for features that would never normally be used in email as they are more suited to complex web pages.

About the only one that is legitimately used is <IFrame> as the Dilbert daily cartoon uses it. As a result, the sources of <IFrame> tags can be logged so that you can construct a whitelist for the tags so you allow them from a few addresses but ban them from all others.

`Log Dangerous HTML Tags = no`

Banning some HTML tags completely is likely to break some common HTML mailing lists, such as Dilbert and other important things like that. So before you implement any restrictions on them, you can log the sender of any message containing a dangerous HTML tag, so that you can set the “dangerous HTML tags” options to be rulesets allowing some of the tags from named "From" addresses and banning all others. This can also be the filename of a ruleset.

`Allow IFrame Tags = disarm`

Do you want to allow <IFrame> tags in email messages? This is not a good idea as it allows various Microsoft Outlook security vulnerabilities to remain unprotected, but if you have a load of mailing lists sending them, then you will want to allow them to keep your users happy.

yes	Allow these tags to be in the message
No	Ban messages containing these tags
disarm	Allow these tags, but stop these tags from working

This can also be the filename of a ruleset, so you can allow them from known mailing lists but ban them from everywhere else.

`Allow Form Tags = disarm`

Do you want to allow <Form> tags in email messages? This is a bad idea as these are used as scams to pursuade people to part with credit card information and other personal data.

yes	Allow these tags to be in the message
no	Ban messages containing these tags
disarm	Allow these tags, but stop these tags from working

Note: Disarming can be defeated, it is not 100% safe! This can also be the filename of a ruleset.

`Allow Script Tags = disarm`

Do you want to allow <Script> tags in email messages? This is a bad idea as these are used to exploit vulnerabilities in email applications and web browsers.

yes	Allow these tags to be in the message
no	Ban messages containing these tags
disarm	Allow these tags, but stop these tags from working

Note: Disarming can be defeated, it is not 100% safe! This can also be the filename of a ruleset.

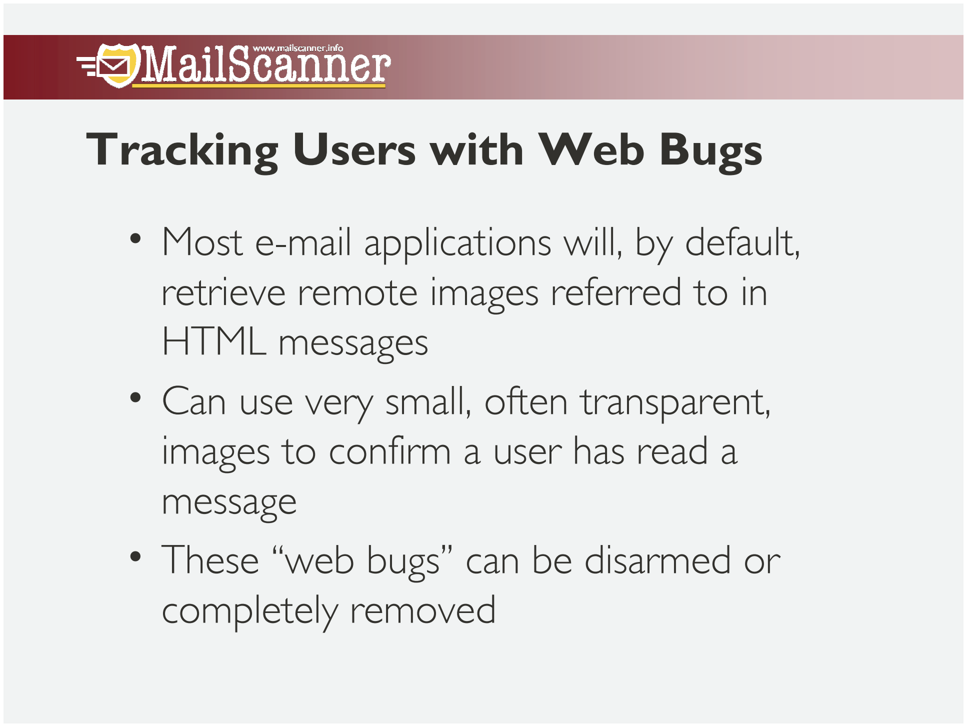
`Allow Object Codebase Tags = disarm`

Do you want to allow <Object Codebase=...> or <Object Data=...> tags in email messages? This is a bad idea as it leaves you unprotected against various Microsoft-specific security vulnerabilities. But if your users demand it, you can do it.

yes	Allow these tags to be in the message

No	Ban messages containing these tags
disarm	Allow these tags, but stop these tags from working

This can also be the filename of a ruleset, so you can allow them just for specific users or domains.


Tracking Users with Web Bugs

- Most e-mail applications will, by default, retrieve remote images referred to in HTML messages
- Can use very small, often transparent, images to confirm a user has read a message
- These “web bugs” can be disarmed or completely removed

Fortunately this problem has been helped by Windows XP Service Pack 2, which changes Outlook’s default to stop it retrieving web bugs automatically.

However, in all other ways this is becoming more of a problem as this is being increasingly used by spammers. Not only can they get a confirmation that the message has been read, but they get confirmation that it has been rendered, along with large amounts of information about the recipient’s system setup, even including their time zone and hence their approximate location. This is all revealed in the HTTP headers sent to the web server which is serving up the web bug.

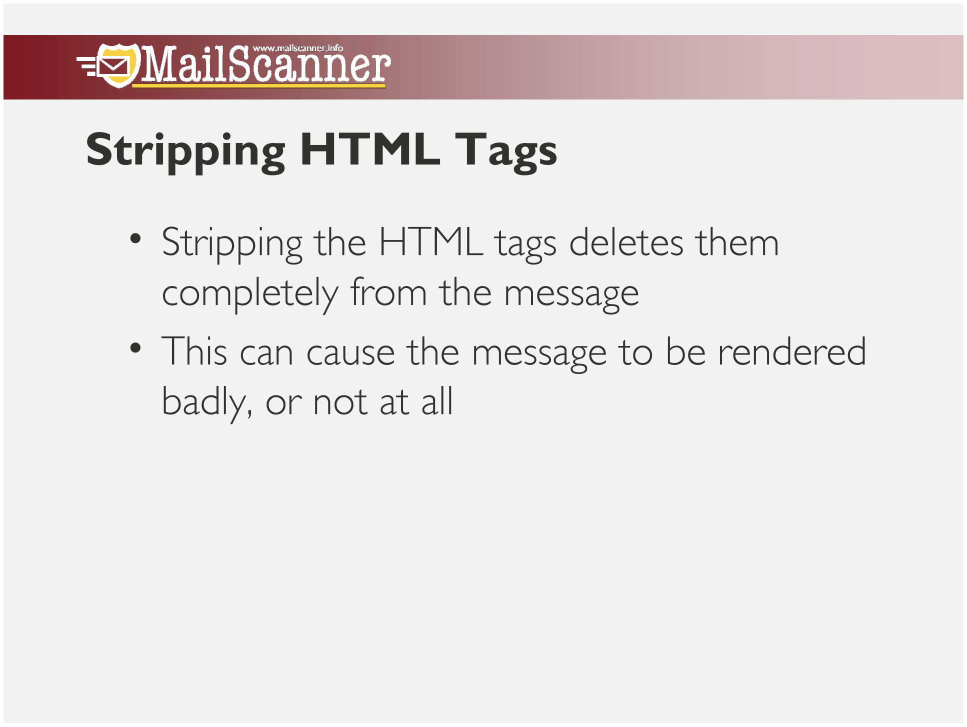
```
Web Bug Replacement =
        http://www.mailscanner.info/images/1x1spacer.gif
```

Removing web bugs altogether often destroys the HTML layout of the message, so the web bug can be replaced with a 1x1 pixel transparent image. This allows the web bug removal process to leave the HTML message formatting mostly intact.

`Ignored Web Bug Filenames =`

This option is a space-separated list of words. If any of these words appear in the filename portion of the URL of the web bug, the web bug is ignored as it is assumed to be a harmless spacer image. A suitable setting for this might be the following:

```
Ignored Web Bug Filenames = spacer pixel.gif pixel.png
```

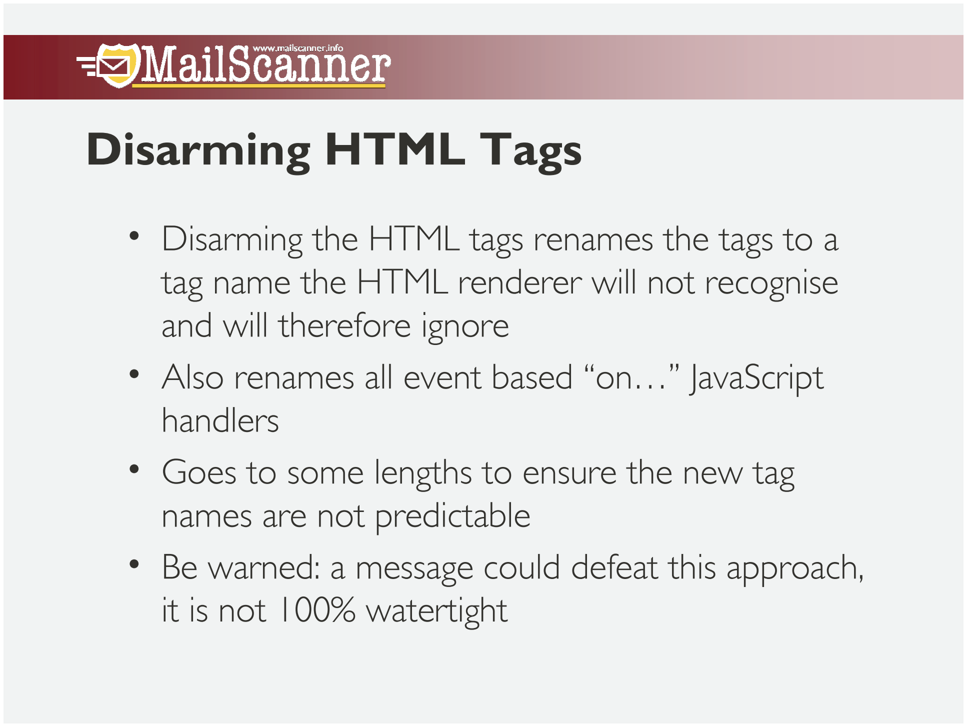


Stripping HTML Tags

- Stripping the HTML tags deletes them completely from the message
- This can cause the message to be rendered badly, or not at all

This is useful for <Object>, <Script> and <IFrame> tags but bad for forms.

In the case of the <Script> and <Style> tags, all the text between the starting element and the corresponding closing element (</Script> or </Style>) is removed from the message, to make it easier to read.

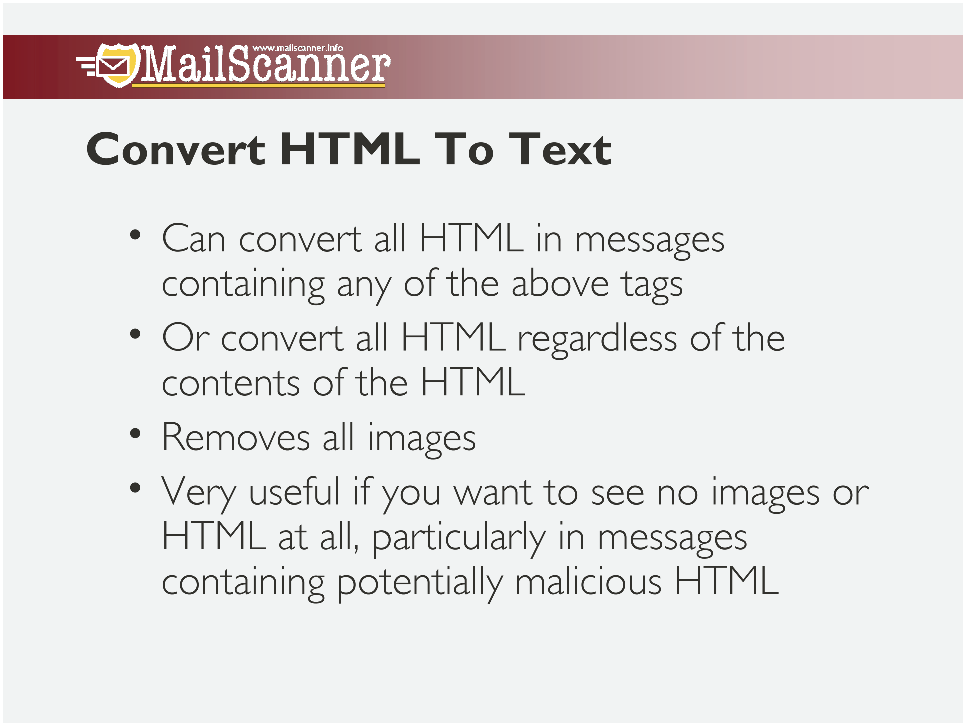


Disarming HTML Tags

- Disarming the HTML tags renames the tags to a tag name the HTML renderer will not recognise and will therefore ignore
- Also renames all event based "on…" JavaScript handlers
- Goes to some lengths to ensure the new tag names are not predictable
- Be warned: a message could defeat this approach, it is not 100% watertight

The new tag names include the MailScanner process id so that they cannot be predicted by a malicious message.

Using some clever XML, it is possible to define new tags that would have the same effect as, for example, the <Script> tag but it would have a different name to avoid detection. There has not been any sign of this being attacked yet, so it is currently a useful defence against HTML attacks.



Convert HTML To Text

- Can convert all HTML in messages containing any of the above tags
- Or convert all HTML regardless of the contents of the HTML
- Removes all images
- Very useful if you want to see no images or HTML at all, particularly in messages containing potentially malicious HTML

This will remove all HTML formatting from the message, but will leave the destinations of any links intact, so they can still be used if the user's email application recognises links and displays them as "active" links in the message.

`Convert Dangerous HTML To Text = no`

This option interacts with the "Allow ... Tags" options above like this:

Allow...Tags	Convert Danger...	Action Taken on HTML Message
No	no	Blocked
No	yes	Blocked
Disarm	no	Specified HTML tags disarmed
Disarm	yes	Specified HTML tags disarmed

Yes	no	Nothing, allowed to pass
Yes	yes	All HTML tags stripped

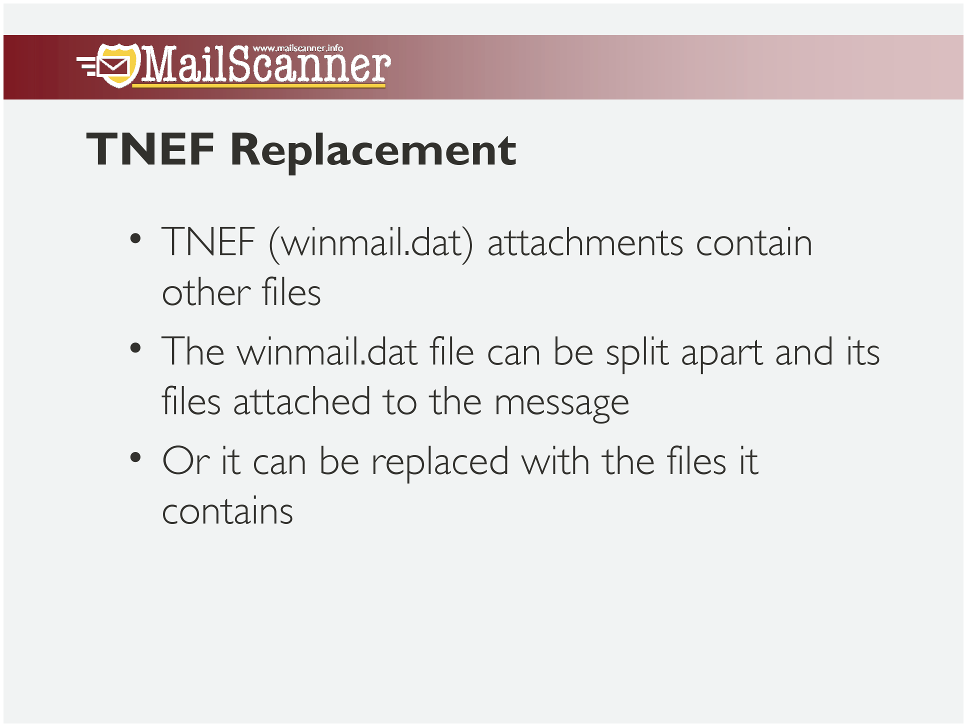
If an "Allow ... Tags = yes" is triggered by a message, and this "Convert Dangerous HTML To Text" is set to "yes", then the HTML message will be converted to plain text. This makes the HTML harmless, while still allowing your users to see the text content of the messages. Note that all graphical content will be removed. This can also be the filename of a ruleset, so you can make this apply only to specific users or domains.

`Convert HTML To Text = no`

Do you want to convert all HTML messages into plain text? This is very useful for users who are children or are easily offended by nasty things like pornographic spam. This can also be the filename of a ruleset, so you can switch this feature on and off for particular users or domains.

Note that when converting HTML to plain text, all the contents between <script> and </script> tags, and <style> and </style> tags, are completely removed from the message. This makes messages much easier to read after they have been stripped.

5.8 Message Attachments Manipulation


TNEF Replacement

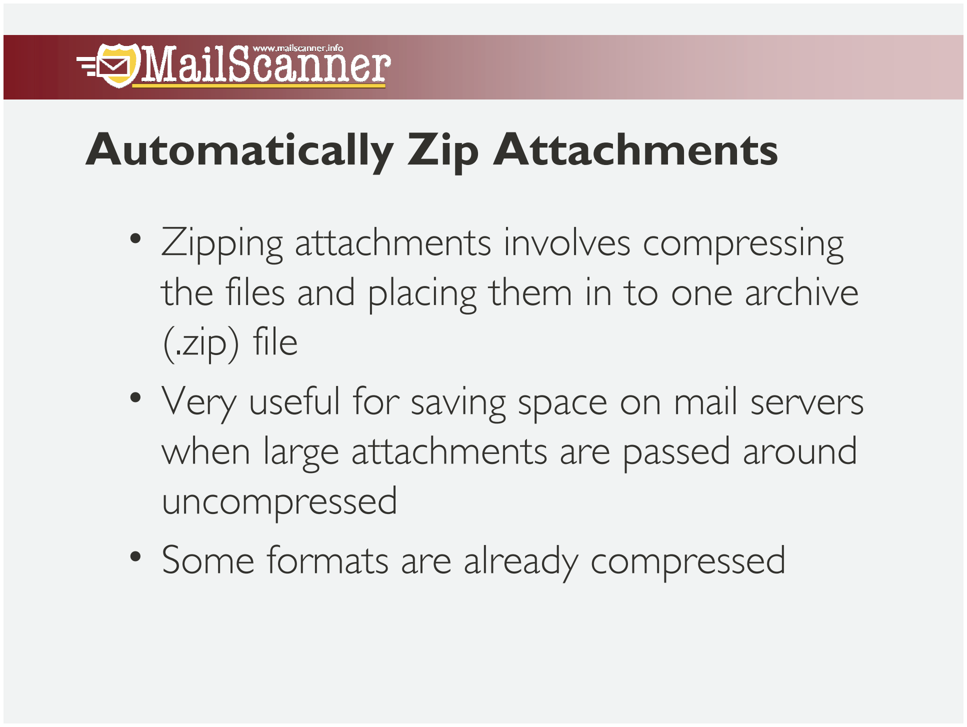
- TNEF (winmail.dat) attachments contain other files
- The winmail.dat file can be split apart and its files attached to the message
- Or it can be replaced with the files it contains

TNEF files can usually be split apart into their member files. The winmail.dat file can either have its contents added to the message, or the file can be replaced by its contents.

`Use TNEF Contents = replace`

This can have one of three values:

no	Leave the winmail.dat file alone, do not add its contents to the message.
add	Leave the winmail.dat file in the message, but add attachments to the message containing the files within the winmail.dat file.
replace	Replace the winmail.dat file with attachments containing the files within it.


Automatically Zip Attachments

- Zipping attachments involves compressing the files and placing them in to one archive (.zip) file
- Very useful for saving space on mail servers when large attachments are passed around uncompressed
- Some formats are already compressed

Some file formats, particularly Microsoft Word and Excel files, compress very well. When large files are passed around a lot within an organisation, lots of disk space can be saved by compressing these attachments and placing them in a Zip file.

Clearly, some file formats compress better than others, and there are some formats which already include advanced compression techniques, so some files should not be compressed.

Compressing files adds another step for the recipient to be able to access the files, as they have to extract the files from the zip file, so it is not worth doing unless a large amount of space is likely to be saved.

`Zip Attachments = no`

Should the attachments in a message ever be compressed into a zip file. This will most likely be used as a ruleset so that, for example, only mail leaving a company will be compressed.

`Attachments Zip Filename = MessageAttachments.zip`

This sets the filename used within the message as the name of the zip file containing some or all of the original attachments. This can also be the filename of a ruleset.

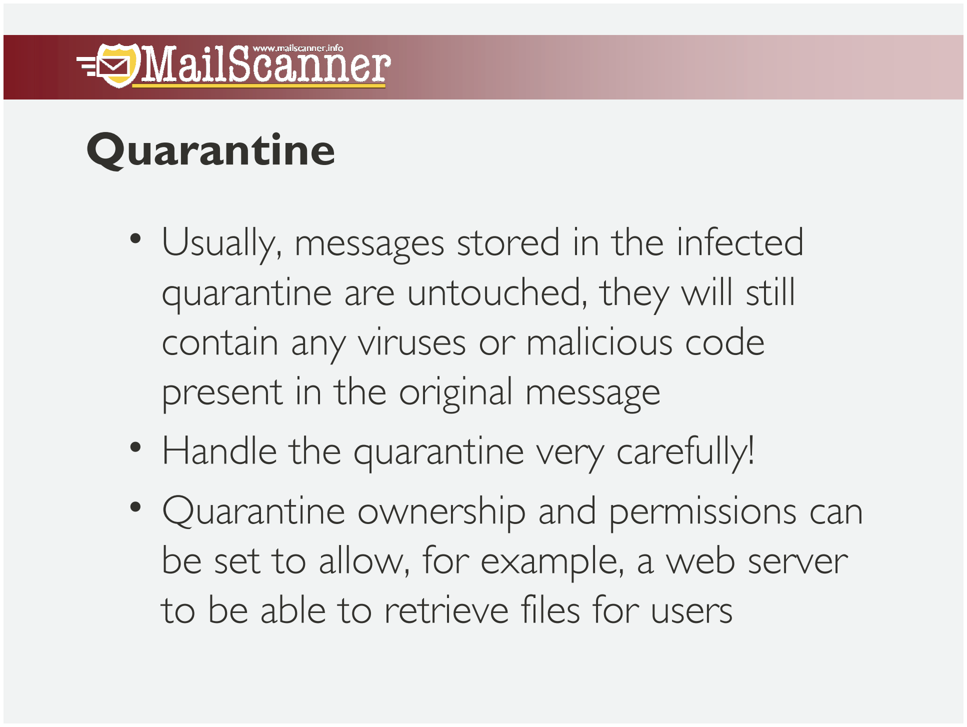
`Attachments Min Total Size To Zip = 100k`

The attachments are only compressed if the total size of the attachments that can be put in the zip file is more than this number of bytes.

`Attachment Extensions Not To Zip = .zip .rar .gz .tgz .jpg .jpeg .mpg .mpe .mpeg .mp3 .rpm`

Some types of files use formats which already contain advanced compression. Clearly there is no point attempting to compress these attachments any further. This setting gives a list of filename extensions which will not be considered for adding to the zip file.

5.9
Quarantine & Modifying Messages



Quarantine

- Usually, messages stored in the infected quarantine are untouched, they will still contain any viruses or malicious code present in the original message
- Handle the quarantine very carefully!
- Quarantine ownership and permissions can be set to allow, for example, a web server to be able to retrieve files for users

Any message placed into the main quarantine should be checked carefully by other means before it is delivered to a user. Spam stored in the quarantine is put into separate directories, so it is easy to differentiate between spam and potentially dangerous messages.

Several people have written simple systems where spam messages are replaced with a notification (see “Spam Actions = notify”) containing an HTML link to an automatic retrieval system. That results in large spam messages being replaced by very small notifications, but the recipients can still recover their spam messages from the quarantine if they want to read any individual messages.

`Quarantine Infections = yes`

Do you want to store copies of the infected attachments and messages? This can also be the filename of a ruleset.

`Quarantine Dir = /var/spool/MailScanner/quarantine`

Set where to store infected and message attachments (if they are kept) This can also be the filename of a ruleset.

`Quarantine Modified Body = no`

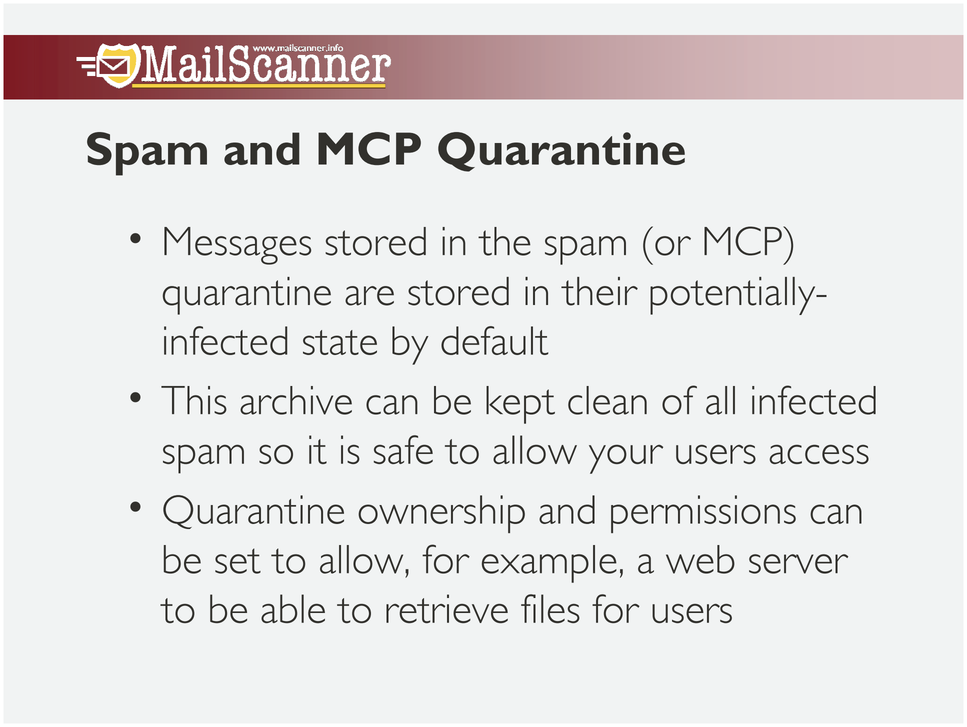
Do you want to save modified messages in the quarantine? This will save all messages that have been modified, including any changes to the HTML of the message. This can also be the name of a ruleset.

`Quarantine User =`
`Quarantine Group =`

If, for example, you are using a web interface so that users can manage their quarantined files, you might want to change the ownership and permissions of the quarantined so that they can be read and/or deleted by the web server.
Don't touch this unless you know what you are doing!If you want to create the quarantine/archive so the files are owned by a user other than the "Run As User" setting at the top of this file, you can change that here. Note: If the "Run As User" is not "root" then you cannot change the user but may still be able to change the group, if the "Run As User" is a member of both of the groups "Run As Group" and "Quarantine Group".

`Quarantine Permissions = 0600`

If you want processes running under the same *group* as MailScanner to be able to read the quarantined files (and list what is in the directories, of course), set to 0640. If you want *all* other users to be able to read them, set to 0644. For a detailed description, if you're not already familiar with it, refer to `man 2 chmod`. Typical use: let the webserver have access to the files so users can download them if they really want to. Use with care, you may well open security holes.



Spam and MCP Quarantine

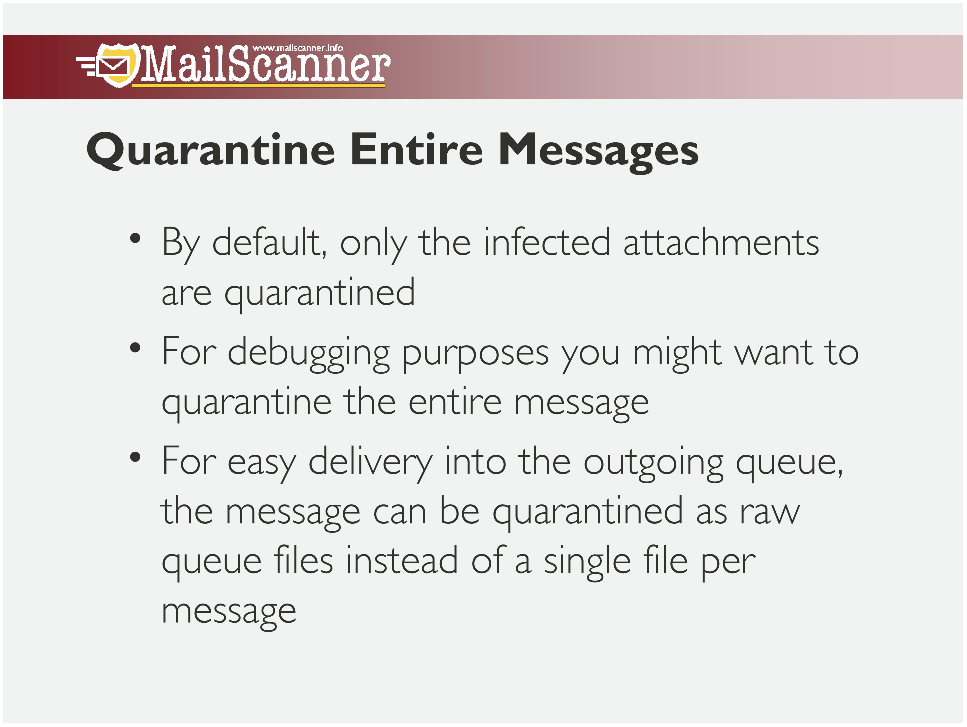
- Messages stored in the spam (or MCP) quarantine are stored in their potentially-infected state by default
- This archive can be kept clean of all infected spam so it is safe to allow your users access
- Quarantine ownership and permissions can be set to allow, for example, a web server to be able to retrieve files for users

The location of the Spam and MCP quarantines is determined automatically from the location of the main infected quarantine, by adding "spam" or "mcp" directories to the end of the path to each one.

As you may well want to provide some simple interface for your users to be able to access the spam quarantine and retrieve incorrectly-tagged messages from it, there is a considerable danger of allowing them to access spam that has virus infections in it.

`Keep Spam And MCP Archive Clean = yes`

Setting this option to yes will remove all messages from the spam archive that were found to be infected or dangerous in any way, for example by viruses or having filenames that are not allowed. If your users require access to infected or dangerous files, they will not be able to retrieve them from the spam archive, and will have to contact the administrators in some other way for access to these files. This can also be the filename of a ruleset.



Quarantine Entire Messages

- By default, only the infected attachments are quarantined
- For debugging purposes you might want to quarantine the entire message
- For easy delivery into the outgoing queue, the message can be quarantined as raw queue files instead of a single file per message

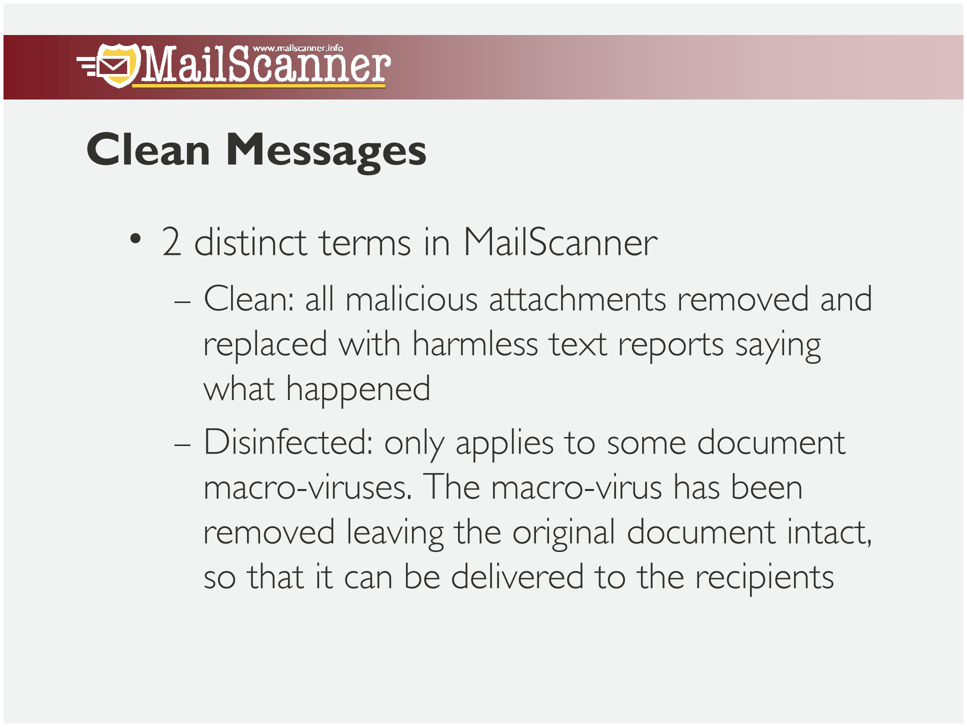
If you intend to deliver potentially malicious files to users, it is useful to recover them from the quarantine as files as they are easier to scan before delivery. In the case of quarantined spam, the whole message is usually what is needed.

`Quarantine Whole Message = no`

> Do you want to quarantine the original *entire* message as well as just the infected attachments? This can also be the filename of a ruleset.

`Quarantine Whole Message As Queue Files = no`

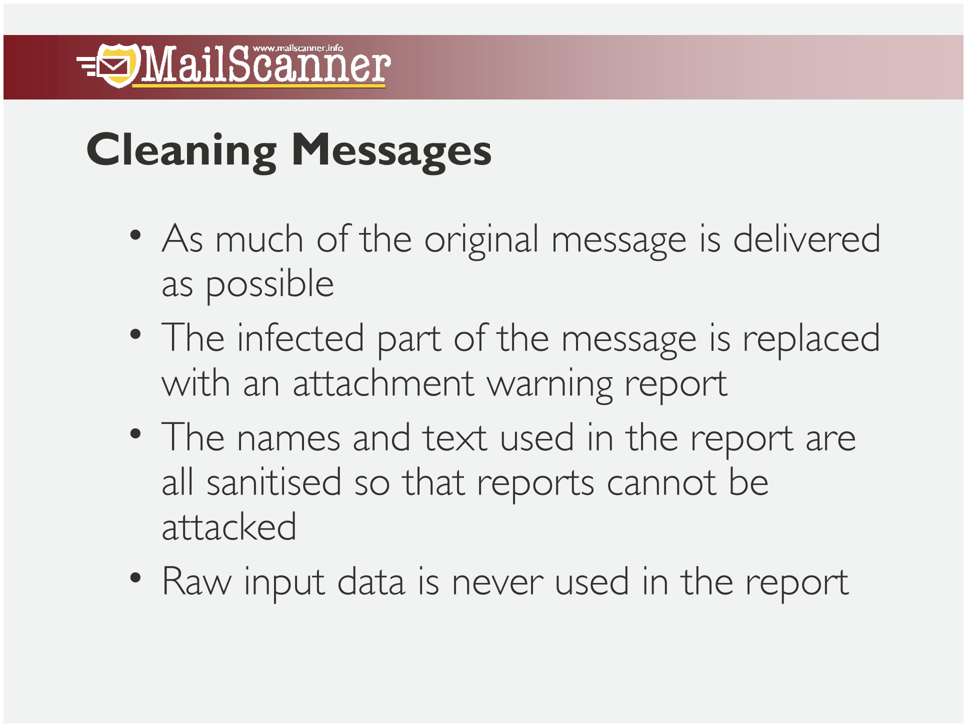
> When you quarantine an entire message, do you want to store it as raw mail queue files (so you can easily send them onto users) or as human-readable files (header then body in 1 file)?


Clean Messages

- 2 distinct terms in MailScanner
  - Clean: all malicious attachments removed and replaced with harmless text reports saying what happened
  - Disinfected: only applies to some document macro-viruses. The macro-virus has been removed leaving the original document intact, so that it can be delivered to the recipients

All messages are cleaned.

Only macro-viruses can be disinfected, leaving the original document contents intact.


www.mailscanner.info
MailScanner

Cleaning Messages

- As much of the original message is delivered as possible
- The infected part of the message is replaced with an attachment warning report
- The names and text used in the report are all sanitised so that reports cannot be attacked
- Raw input data is never used in the report

An example might be a cleverly coded filename attachment that itself contained a complete short attachment, or that confused the MIME parser into not parsing the message. Never using real input data in a system's output is very good security practice, and underpins several design features in MailScanner.

`Mark Infected Messages = yes`

Add the "Inline HTML Warning" or "Inline Text Warning" to the top of messages that have had attachments removed from them? This can also be the filename of a ruleset.

```
Inline HTML Warning = %report-dir%/inline.warning.html
Inline Text Warning = %report-dir%/inline.warning.txt
```

Set where to find the HTML and text versions that will be inserted at the top of messages that have had viruses removed from them. These can also be the filenames of rulesets.

`Attachment Warning Filename = %org-name%-Attachment-Warning.txt`

When a virus or attachment is replaced by a plain-text warning, and that warning is an attachment, this is the filename of the new attachment. This can also be the filename of a ruleset.

`Attachment Encoding Charset = us-ascii`

What character set do you want to use for the attachment that replaces viruses (VirusWarning.txt)? The default is "us-ascii" but if you speak anything other than English, you will probably want "ISO-8859-1" instead. This can also be the filename of a ruleset.

`Warning Is Attachment = yes`

When a virus or attachment is replaced by a plain-text warning, should the warning be in an attachment? If "no" then it will be placed in-line. This can also be the filename of a ruleset.

`Hide Incoming Work Dir = yes`

Hide the directory path from all virus scanner reports sent to users. The extra directory paths give away information about your setup, and tend to just confuse users. This can also be the filename of a ruleset.

`Include Scanner Name In Reports = yes`

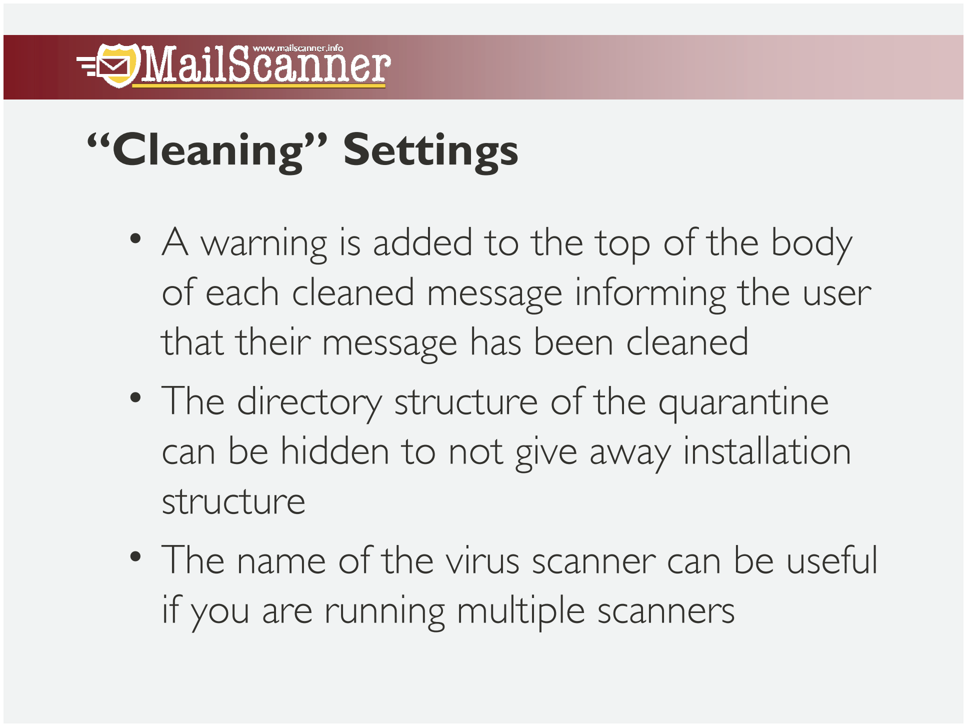
Include the name of the virus scanner in each of the scanner reports. This also includes the translation of "MailScanner" in each of the report lines resulting from one of MailScanner's own checks such as filename, filetype or dangerous HTML content. To change the name "MailScanner", look in reports/...../languages.conf. Very useful if you use several virus scanners, but a bad idea if you don't want to let your customers know which scanners you use.

```
Deleted Bad Content Message Report  = %report-
      dir%/deleted.content.message.txt
Deleted Bad Filename Message Report = %report-
      dir%/deleted.filename.message.txt

Deleted Virus Message Report = %report-
      dir%/deleted.virus.message.txt
```

Set where to find the message text sent to users when one of their attachments has been deleted from a message. These can also be the filenames of rulesets.

```
Stored Bad Content Message Report = %report-
      dir%/stored.content.message.txt
Stored Bad Filename Message Report = %report-
      dir%/stored.filename.message.txt

Stored Virus Message Report = %report-dir%/stored.virus.message.txt
```

Set where to find the message text sent to users when one of their attachments has been deleted from a message and stored in the quarantine. These can also be the filenames of rulesets.

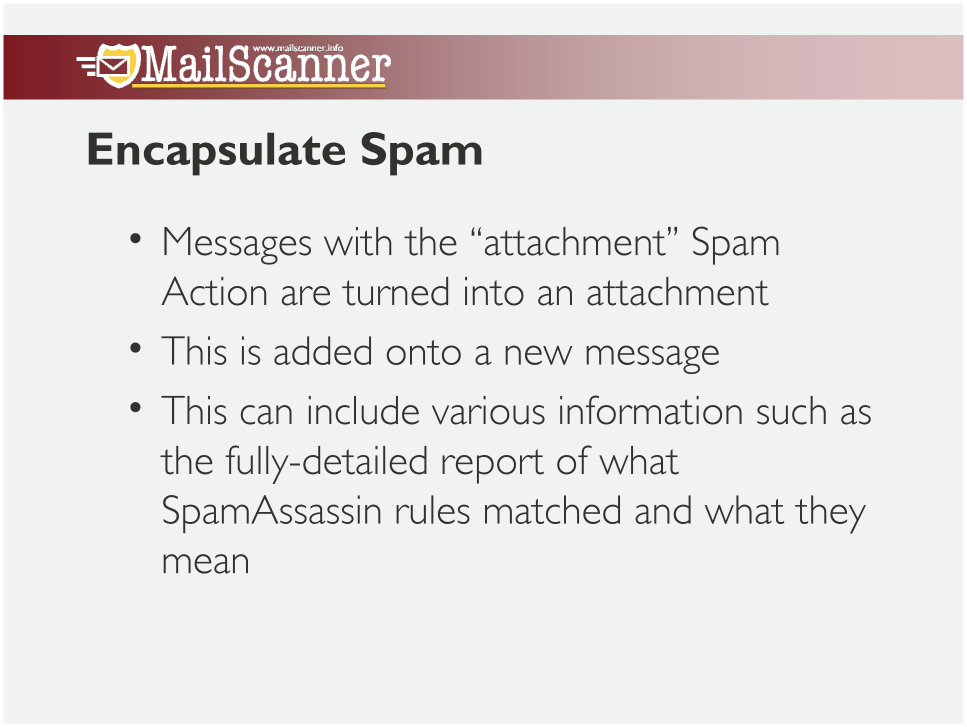


“Cleaning” Settings

- A warning is added to the top of the body of each cleaned message informing the user that their message has been cleaned
- The directory structure of the quarantine can be hidden to not give away installation structure
- The name of the virus scanner can be useful if you are running multiple scanners

The name of the virus scanner can be MailScanner itself, when it has found potentially malicious content that it has removed from a message. It is very common for one attachment to be removed for several different reasons, so there can easily be 2 or 3 reports per attachment.

As with other modifications and reports produced by MailScanner, it is easy to customise them so that a company’s own branding can be applied to all output see by an end user.



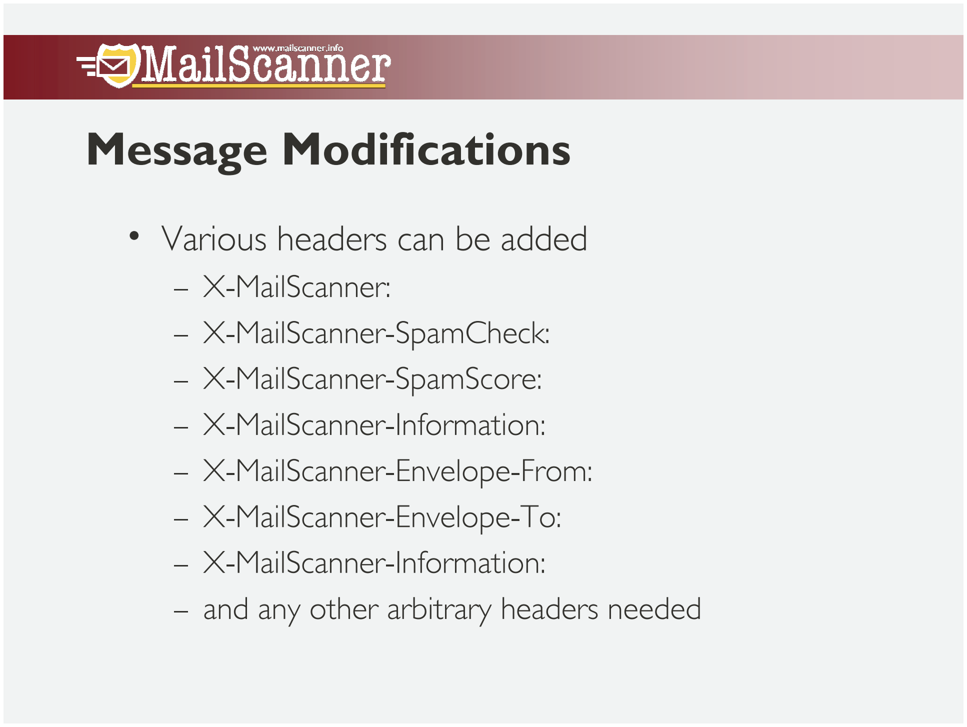
Encapsulate Spam

- Messages with the “attachment” Spam Action are turned into an attachment
- This is added onto a new message
- This can include various information such as the fully-detailed report of what SpamAssassin rules matched and what they mean

The most-detailed SpamAssassin report, consisting of a table with the details of each rule on a separate line, cannot be produced in all the 15 languages supported by MailScanner as the text is generated by SpamAssassin itself, which only supports a few languages.

`Inline Spam Warning = %report-dir%/inline.spam.warning.txt`

This is the filename of the text put into the new message to explain the encapsulation. Like the others, it can contain various variable names which are expanded in the report. This can also be the filename of a ruleset.

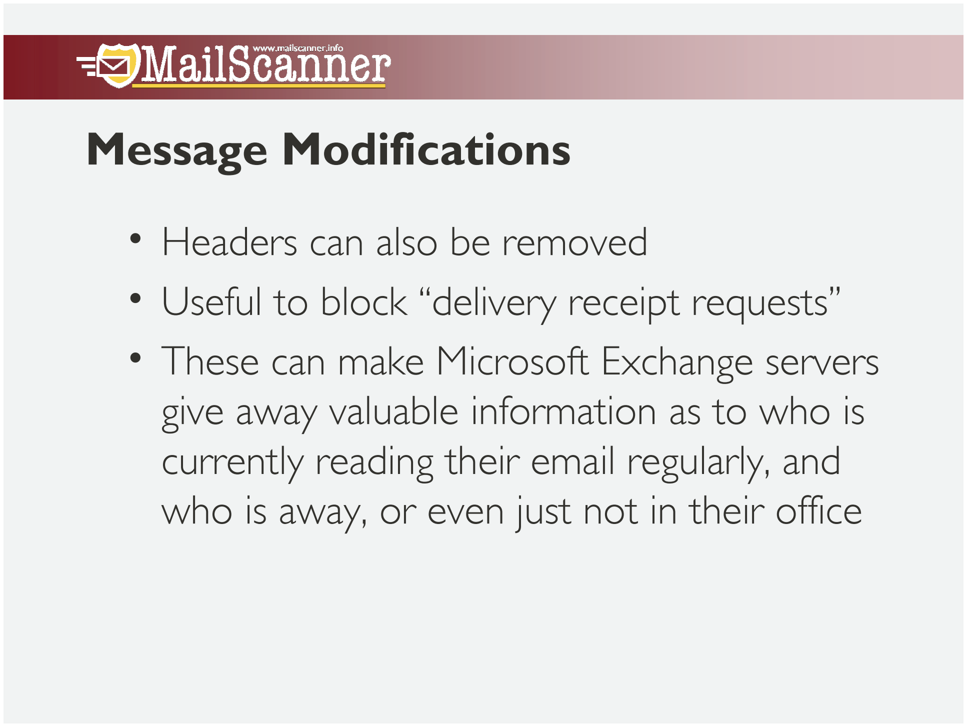


Message Modifications

- Various headers can be added
  - X-MailScanner:
  - X-MailScanner-SpamCheck:
  - X-MailScanner-SpamScore:
  - X-MailScanner-Information:
  - X-MailScanner-Envelope-From:
  - X-MailScanner-Envelope-To:
  - X-MailScanner-Information:
  - and any other arbitrary headers needed

All of these headers can be renamed to provide your own branding.

For more information on adding arbitrary headers to the resulting message, please see the index entry for "Spam Actions".


MailScanner www.mailscanner.info

Message Modifications

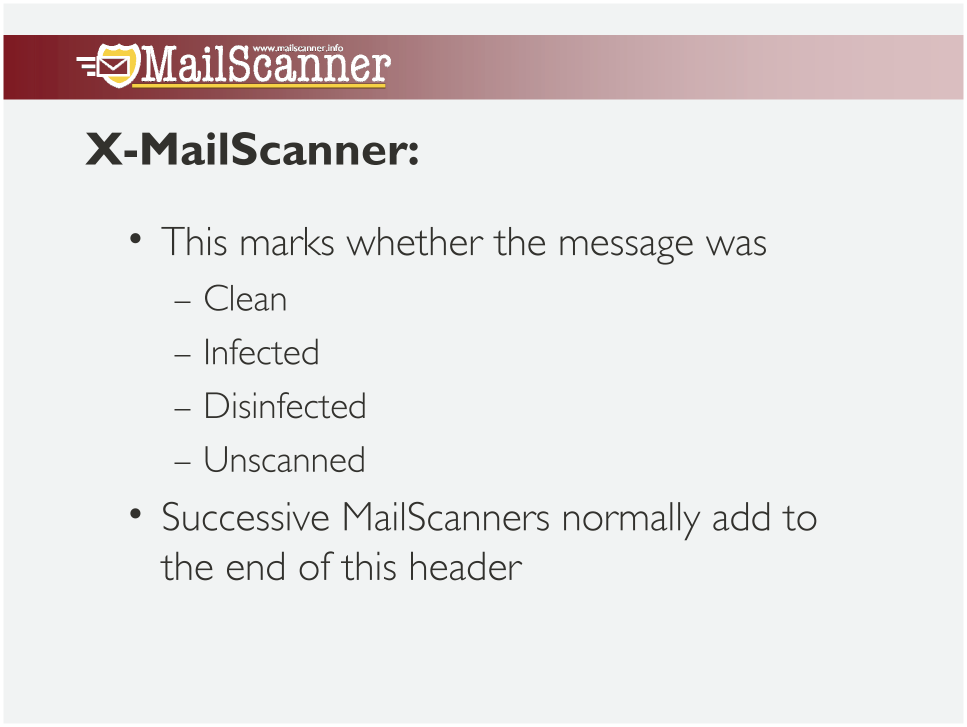
- Headers can also be removed
- Useful to block “delivery receipt requests”
- These can make Microsoft Exchange servers give away valuable information as to who is currently reading their email regularly, and who is away, or even just not in their office

`Remove These Headers =`

Here you can specify a space-separated list of header names which you would like to delete from outgoing messages. For example, to remove requests for delivery receipts, you can set

Remove These Headers = Disposition-Notification-To   Return-Receipt-To

The ":" character on the end of each header name are optional. They will be added automatically if you do not specify them.



X-MailScanner:

- This marks whether the message was
  - Clean
  - Infected
  - Disinfected
  - Unscanned
- Successive MailScanners normally add to the end of this header

The other options available are to over-write the value of the header with the results from the last MailScanner through which the message passed, or to add a new header onto the bottom of the message headers containing the new result. Mail on many sites passes through more than one MailScanner server, resulting in a series of reports.

`Mail Header = X-%org-name%-MailScanner:`

Add this extra header to all mail as it is processed. This *must* include the colon ":" at the end. This can also be the filename of a ruleset.

`Clean Header Value = Found to be clean`
`Infected Header Value = Found to be infected`

`Disinfected Header Value = Disinfected`

Set the "Mail Header" to these values for clean/infected/disinfected messages. This can also be the filename of a ruleset.

`Mark Unscanned Messages = yes`

When a message is to not be virus-scanned (which may happen depending upon the setting of "Virus Scanning", especially if it is a ruleset), do you want to add the header advising the users to get their email virus-scanned by you? Very good for advertising your MailScanning service and encouraging users to give you some more money and sign up to virus scanning. This can also be the filename of a ruleset.

`Unscanned Header Value = Not scanned: please contact your Internet E-Mail Service Provider for details`

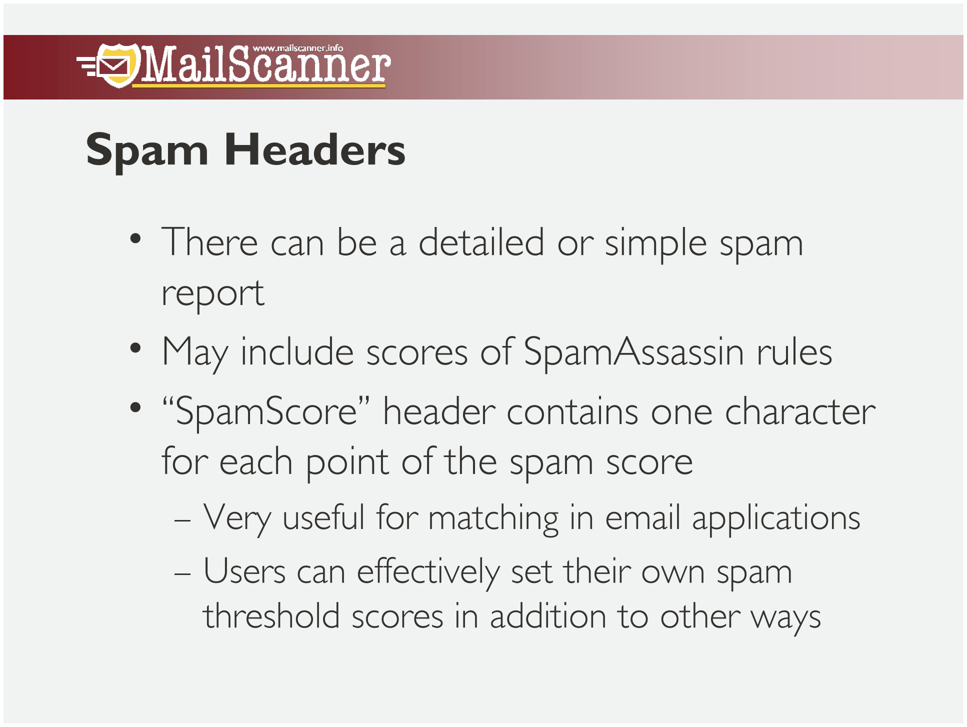
This is the text used by the "Mark Unscanned Messages" option above. This can also be the filename of a ruleset.

`Multiple Headers = append`

What to do when you get several MailScanner headers in one message, from multiple MailScanner servers. Values are

append	Append the new data to the existing header
add	Add a new header
replace	Replace the old data with the new data

The default is "append". This can also be the filename of a ruleset.



Spam Headers

- There can be a detailed or simple spam report
- May include scores of SpamAssassin rules
- "SpamScore" header contains one character for each point of the spam score
  - Very useful for matching in email applications
  - Users can effectively set their own spam threshold scores in addition to other ways

The simple spam report basically just says "yes it is spam" but does not give any details of the spam detection system in use.

The most common character for the Spam Score header is "s" as it will not get in the way of any automatic pattern-matching being used by any of your users. Using "*" may appear a more obvious choice, but any of your users running procmail on their incoming mail will find this difficult to handle in their rules. The problem with using "s" is that it is the default character and so is used in most MailScanner installations. Its presence is a strong indication that your site is running MailScanner, a fact you may wish to disguise.

`Spam Header = X-%org-name%-MailScanner-SpamCheck:`

Add this extra header to all messages found to be spam. This can also be the filename of a ruleset.

`Detailed Spam Report = yes`

Do you want the full spam report, or just a simple "spam / not spam" report?

`Include Scores In SpamAssassin Report = yes`

Do you want to include the numerical scores in the detailed SpamAssassin report, or just list the names of the scores

`Spam Score Header = X-%org-name%-MailScanner-SpamScore:`

Add this extra header if "Spam Score" = yes. The header will contain 1 character for every point of the SpamAssassin score.

`Spam Score = yes`

Do you want to include the "Spam Score" header. This shows 1 character (Spam Score Character) for every point of the SpamAssassin score. This makes it very easy for users to be able to filter their mail using whatever SpamAssassin threshold they want. For example, they just look for "sssss" for every message whose score is > 5, for example. This can also be the filename of a ruleset.

`Spam Score Character = s`

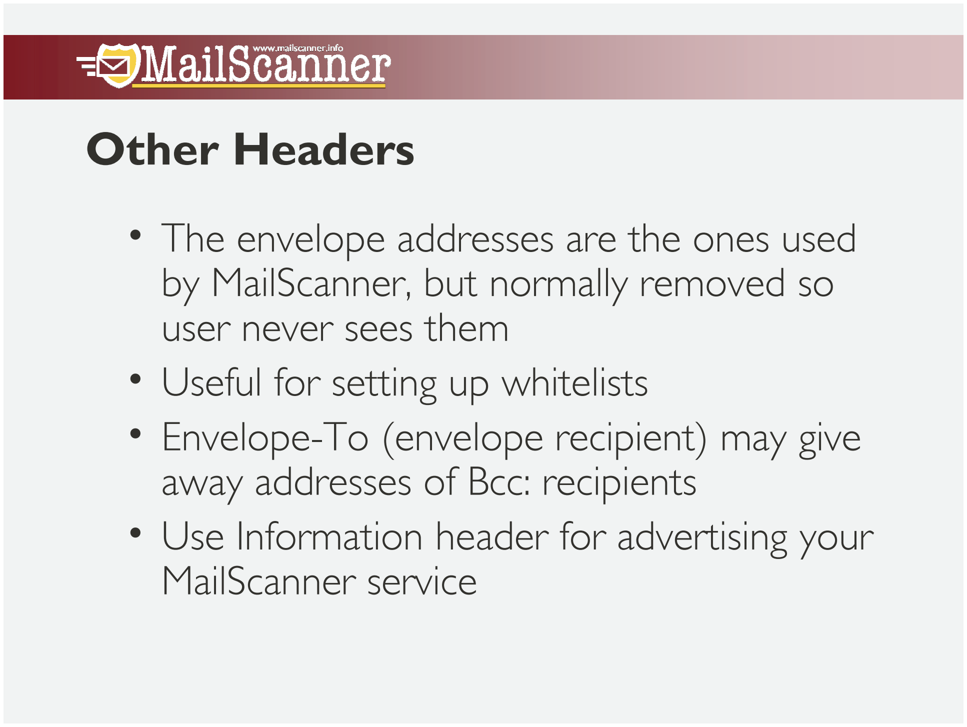
The character to use in the "Spam Score Header". Don't use: x as a score of 3 is "xxx" which the users will think is porn, as it will cause confusion with comments in procmail as well as MailScanner itself,
* as it will cause confusion with pattern matches in procmail,
. as it will cause confusion with pattern matches in procmail,
? as it will cause the users to think something went wrong.
"s" is nice and safe and stands for "spam".

`Spam Score Number Instead Of Stars = no`

If this option is set to yes, you will get a spam-score header saying just the value of the spam score, instead of the row of characters representing the score. This can also be the filename of a ruleset.

`Minimum Stars If On Spam List = 0`

This sets the minimum number of "Spam Score Characters" which will appear if a message triggered the "Spam List" setting but received a very low SpamAssassin score. This means that people who only filter on the "Spam Stars" will still be able to catch messages which receive a very low SpamAssassin score. Set this value to 0 to disable it. This can also be the filename of a ruleset.


MailScanner

Other Headers

- The envelope addresses are the ones used by MailScanner, but normally removed so user never sees them
- Useful for setting up whitelists
- Envelope-To (envelope recipient) may give away addresses of Bcc: recipients
- Use Information header for advertising your MailScanner service

As a message can have many recipients, some of these recipients may reside on the same mail server. The message will therefore still have multiple recipients when it passes through a site's MailScanner server. The "Envelope-To:" header will give away the addresses of the other recipients of the message. This is a very bad idea if you are trying to use "Bcc:" headers to quietly deliver copies of a message to several people without them being able to discover the addresses of the other recipients.

However, the "Envelope-From:" header does not give away any private information and is an essential tool when building whitelists, as these have to be based on the envelope sender of the messages.

`Add Envelope From Header = yes`

Do you want to add the Envelope-From: header? This is very useful for tracking where spam came from as it contains the envelope sender address. This can also be the filename of a ruleset.

`Envelope From Header = X-%org-name%-MailScanner-From:`

This is the name of the Envelope From header controlled by the option above. This can also be the filename of a ruleset.

`Add Envelope To Header = no`

Do you want to add the Envelope-To: header?
This can be useful for tracking span destinations, but should be used with care due to possible privacy concerns with the use of Bcc: headers by users. This can also be the filename of a ruleset.

`Envelope To Header = X-%org-name%-MailScanner-To:`

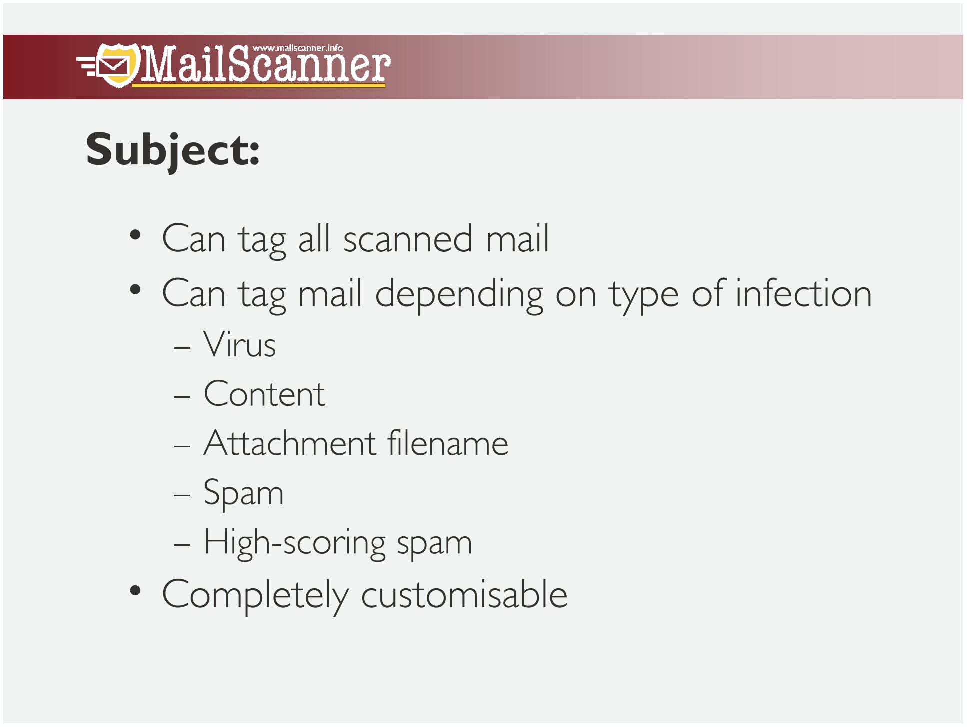
This is the name of the Envelope To header controlled by the option above. This can also be the filename of a ruleset.

`Information Header = X-%org-name%-MailScanner-Information:`

Add this extra header to all mail as it is processed.The contents is set by "Information Header Value" and is intended for you to be able to insert a help URL for your users. If you don't want an information header at all, just comment out this setting or set it to be blank. This can also be the filename of a ruleset.

`Information Header Value = Please contact the ISP for more information`

Set the "Information Header" to this value. This can also be the filename of a ruleset.


www.mailscanner.info
MailScanner

Subject:

- Can tag all scanned mail
- Can tag mail depending on type of infection
  - Virus
  - Content
  - Attachment filename
  - Spam
  - High-scoring spam
- Completely customisable

This is by far the best header change for your users. Some email applications make it very difficult to view other mail headers, and so users can only easily create their own automatic filters using the "Subject:" header.

`Scanned Modify Subject = no`

When the message has been scanned but no other subject line changes have happened, do you want modify the subject line? This can be 1 of 4 values:

no	Do not modify the subject line, or
start or yes	Add text to the start of the subject line, or
end	Add text to the end of the subject line.

This makes very good advertising of your MailScanning service. This can also be the filename of a ruleset.

`Scanned Subject Text = {Scanned}`

This is the text to add to the start/end of the subject line if the "Scanned Modify Subject" option is set. This can also be the filename of a ruleset.

`Virus Modify Subject = yes`

If the message contained a virus, do you want to modify the subject line? This makes filtering in Outlook very easy. This can be any of the 4 values described above. This can also be the filename of a ruleset.

`Virus Subject Text = {Virus?}`

This is the text to add to the start of the subject if the "Virus Modify Subject" option is set. This can also be the filename of a ruleset.

`Content Modify Subject = yes`

If an attachment triggered a content check, but there was nothing else wrong with the message, do you want to modify the subject line? This makes filtering in Outlook very easy. This can be any of the 4 values described above. This can also be the filename of a ruleset.

`Content Subject Text = {Blocked Content}`

This is the text to add to the start of the subject if the "Content Modify Subject" option is set. You might want to change this so your users can see at a glance whether it just was just the content that MailScanner rejected. This can also be the filename of a ruleset.

`Filename Modify Subject = yes`

If an attachment triggered a filename check, but there was nothing else wrong with the message, do you want to modify the subject line? This makes filtering in Outlook very easy. This can be any of the 4 values described above. This can also be the filename of a ruleset.

`Filename Subject Text = {Filename?}`

This is the text to add to the start of the subject if the "Filename Modify Subject" option is set. You might want to change this so your users can see at a glance whether it just was just the filename that MailScanner rejected. This can also be the filename of a ruleset.

`Spam Modify Subject = yes`

If the message is spam, do you want to modify the subject line? This makes filtering in Outlook very easy. This can be any of the 4 values described above. This can also be the filename of a ruleset.

`Spam Subject Text = {Spam?}`

This is the text to add to the start of the subject if the "Spam Modify Subject" option is set. The exact string "_SCORE_" will be replaced by the numeric SpamAssassin score. This can also be the filename of a ruleset.

`High Scoring Spam Modify Subject = yes`

This is just like the "Spam Modify Subject" option above, except that it applies then the score from SpamAssassin is higher than the "High SpamAssassin Score" value. This can be any of the 4 values described above. This can also be the filename of a ruleset.

`High Scoring Spam Subject Text = {Spam?}`

This is just like the "Spam Subject Text" option above, except that it applies then the score from SpamAssassin is higher than the "High SpamAssassin Score" value. The exact string "_SCORE_" will be replaced by the numeric SpamAssassin score. This can also be the filename of a ruleset.

`Spam Score Number Format = %d`

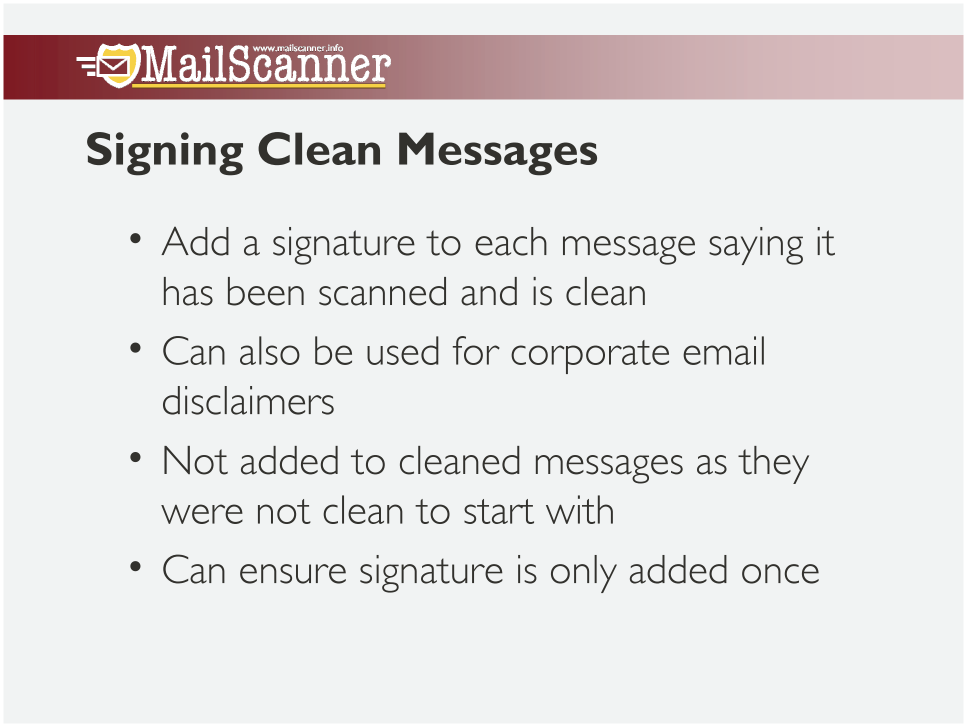
This sets the format of the spam score used when the string "_SCORE_" is encountered. It allows use of all the standard printf() formatting mechanisms. The default value prints a simple integer. This can also be the filename of a ruleset.

`Disarmed Modify Subject = yes`

If the message has had any dangerous HTML tags disarmed, do you want to modify the subject line? Users find this very helpful to avoid them wasting time discovering that a form in their email message has been disarmed and hence will not work. Note that this will not be triggered by a phishing attack on its own, but only if there are disarmed dangerous HTML tags present. This can be any of the 4 values described above. This can also be the filename of a ruleset.

`Disarmed Modify Text = {Disarmed}`

This is the text to add to the start of the subject if the "Disarmed Modify Subject" option is set. This can also be the filename of a ruleset.



Signing Clean Messages

- Add a signature to each message saying it has been scanned and is clean
- Can also be used for corporate email disclaimers
- Not added to cleaned messages as they were not clean to start with
- Can ensure signature is only added once

The signature is added outside the MIME structure of the message, so it does not break any of the means of PGP-signing messages. The signature is placed in the MIME "epilogue" and no main MIME sections of the message are affected.

When signing plain text messages, the signature is simply added at the end of the message text. If the message is PGP-signed in plain text mode, then the MailScanner signature is added after the end of the PGP signature and so the verification of the message is not affected.

To be able to personalise the signature for different users, or to be able to place a personalised link in the inline signature, any of the strings "$from", "$to" and "$subject" may be used in the inline signature files. These will be replaced with the message sender's address, the message recipients' addresses and the message subject line respectively.

The check to see if the message has already been signed is the only place where MailScanner makes use of the headers of an incoming message. If there is already an "X-MailScanner:" header (or whatever you have customised it to be for your site), then the signature is not added. Only the first server to process the message will add the signature.

`Sign Clean Messages = no`

Add the "Inline HTML Signature" or "Inline Text Signature" to the end of uninfected messages? This can also be the filename of a ruleset.

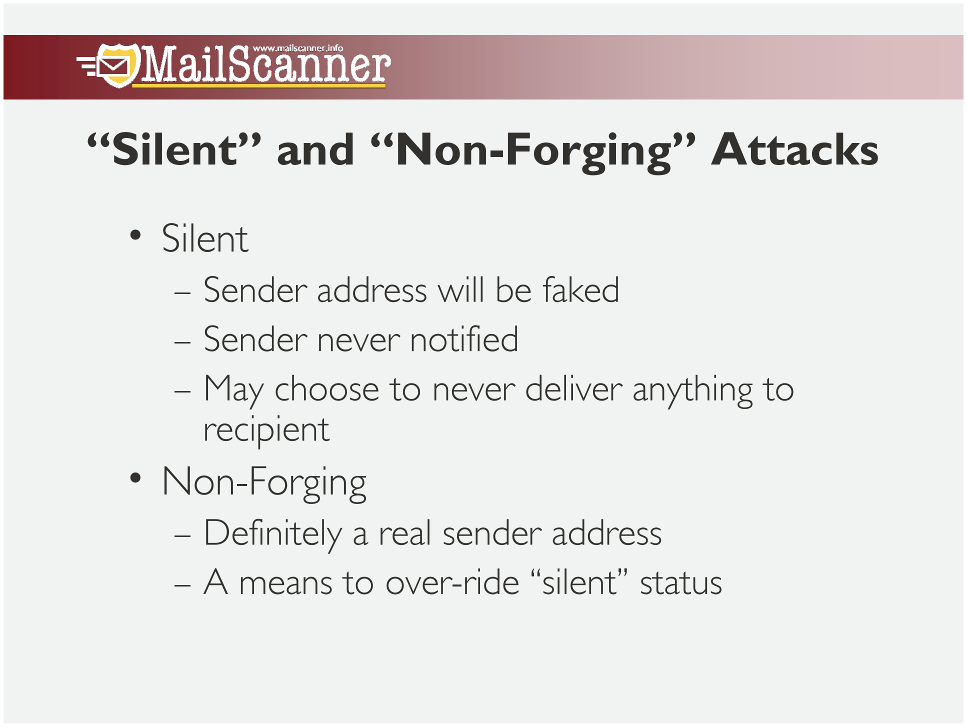
`Sign Messages Already Processed = no`

If this is "no", then (as far as possible) messages which have already been processed by another MailScanner server will not have the clean signature added to the message. This prevents messages getting many copies of the signature as they flow through your site. This can also be the filename of a ruleset.

`Inline HTML Signature = %report-dir%/inline.sig.html`
`Inline Text Signature = %report-dir%/inline.sig.txt`

Set where to find the HTML and text versions that will be added to the end of all clean messages, if "Sign Clean Messages" is set. These can also be the filenames of rulesets.

5.10 Silent & Non-Forging Viruses



“Silent” and “Non-Forging” Attacks

- Silent
  - Sender address will be faked
  - Sender never notified
  - May choose to never deliver anything to recipient
- Non-Forging
  - Definitely a real sender address
  - A means to over-ride “silent” status

These options have slightly curious names for historical reasons.

Most viruses now fake the sender address (and the From: header) in email they send to propagate themselves. Therefore, at least, the apparent senders of these messages should not be sent a warning as they are not the real sender. It may also be desirable to not deliver anything to the recipient either as the message contains no useful content, but just silently drop the message.

However, there are some viruses and attacks in which the sender address is not faked or forged, but is the correct email address of the sender. Examples of these are the “joke” programs that circulate from time to time, which are harmless but are really a waste of everyone’s time and mail storage. Other examples are most of the potentially malicious HTML, as these are sometimes generated by mailing lists.

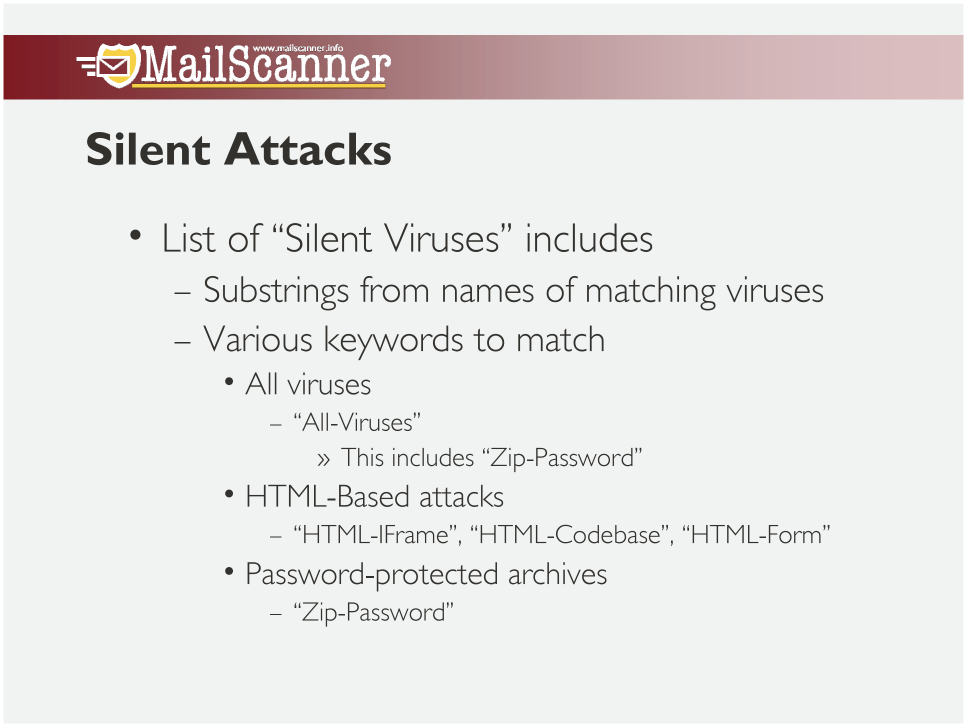
Including the name of a virus or attack in both lists causes the “Non-Forging” list to take precedence over the “Silent Viruses” list.

`Still Deliver Silent Viruses = yes`

Still deliver (after cleaning) messages that contained viruses listed in the above option ("Silent Viruses") to the recipient? Setting this to "yes" is good when you are testing everything, and because it shows management that MailScanner is protecting them, but it is bad because they have to filter/delete all the incoming virus warnings.
Note: Once you have deployed this into "production" use, you should set this option to "no" so you don't bombard thousands of people with useless messages they don't want! This can also be the filename of a ruleset.

`Log Silent Viruses = yes`

Log the detection of silent viruses in the syslog. Setting this to "yes" simplifies the automatic report generation from the logs, as they will include all the silent viruses as well as the non-forging viruses and other attacks that are detected.



Silent Attacks

- List of “Silent Viruses” includes
  - Substrings from names of matching viruses
  - Various keywords to match
    - All viruses
      - “All-Viruses”
        - » This includes “Zip-Password”
    - HTML-Based attacks
      - “HTML-IFrame”, “HTML-Codebase”, “HTML-Form”
    - Password-protected archives
      - “Zip-Password”

The words included in this setting are searched for in the output reports from the virus scanners. They are just searched as substrings, so you do not have to state the exact name of a virus, part of the name will do, and will match variants of the virus as well as the names used by other virus scanning engines you might be running. For example, instead of writing “W32/Klez-H”, you would be better just writing “Klez” as this will cover all variants of Klez.

The various keywords can be included so that you can selectively not warn the senders of different specific types of blocked content.

`Silent Viruses = HTML-IFrame All-Viruses`

Strings listed here will be searched for in the output of the virus scanners. It is used to list which viruses should be handled differently from other viruses. If a virus name is given here, then

1. The sender will not be warned that he sent it

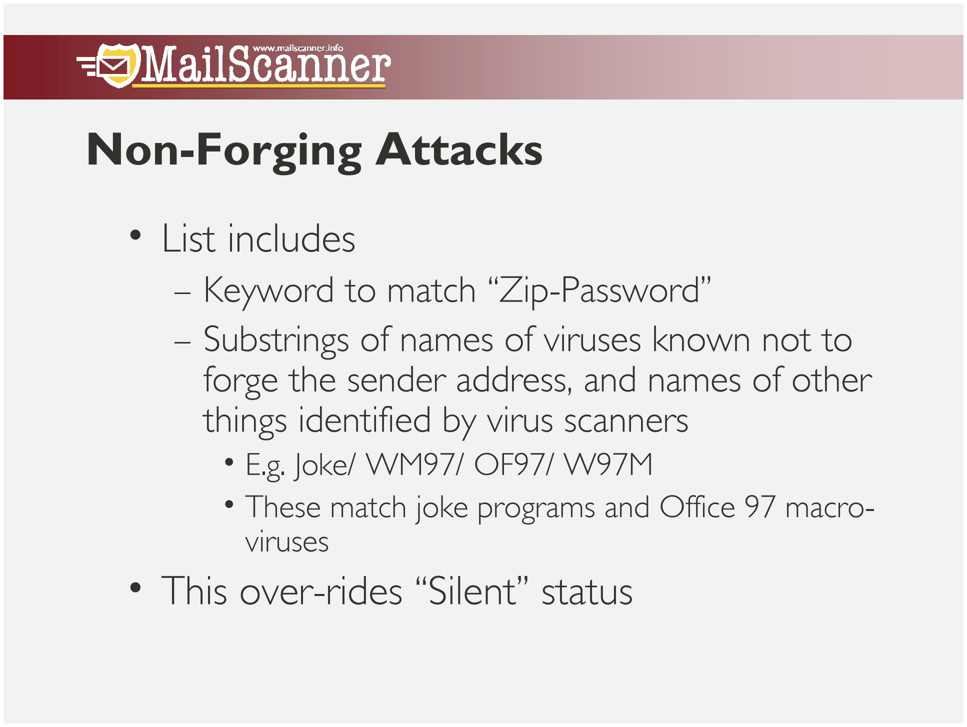
2. No attempt at true disinfection will take place (but it will still be "cleaned" by removing the nasty attachments from the message)

3. The recipient will not receive the message, unless the "Still Deliver Silent Viruses" option is set

Other words that can be put in this list are the 5 special keywords

HTML-IFrame	inserting this will stop senders being warned about HTML Iframe tags, when they are not allowed.
HTML-Codebase	inserting this will stop senders being warned about HTML Object Codebase/Data tags, when they are not allowed.
HTML-Form	inserting this will stop senders being warned about HTML Form tags, when they are not allowed.
Zip-Password	inserting this will stop senders being warned about password-protected zip files, when they are not allowed. This keyword is not needed if you include All-Viruses.
All-Viruses	inserting this will stop senders being warned about any virus, while still allowing you to warn senders about HTML-based attacks. This includes Zip-Password so you don't need to include both.

The default of "All-Viruses" means that no senders of viruses will be notified (as the sender address is always forged these days anyway), but anyone who sends a message that is blocked for other reasons will still be notified. This can also be the filename of a ruleset.


www.mailscanner.info MailScanner

Non-Forging Attacks

- List includes
  - Keyword to match “Zip-Password”
  - Substrings of names of viruses known not to forge the sender address, and names of other things identified by virus scanners
    - E.g. Joke/ WM97/ OF97/ W97M
    - These match joke programs and Office 97 macro-viruses
- This over-rides “Silent” status

In much the same way as the “Silent Viruses” list, words listed here are searched for in the output of the virus scanning engines. As an example, if you are using Sophos, all joke programs can easily be included with the string “Joke/” as Sophos’s naming scheme is “type/name-variant”. The default setting below also includes all Microsoft Office macro viruses.

The keyword “Zip-Password” can be included here to over-ride its setting in “Silent Viruses”, where the “All-Viruses” keyword encompasses “Zip-Password” as well.

Viruses matched by this list will be treated as if they were not in the “Silent Viruses” list, as “Non-Forging Viruses” over-rides “Silent Viruses”.

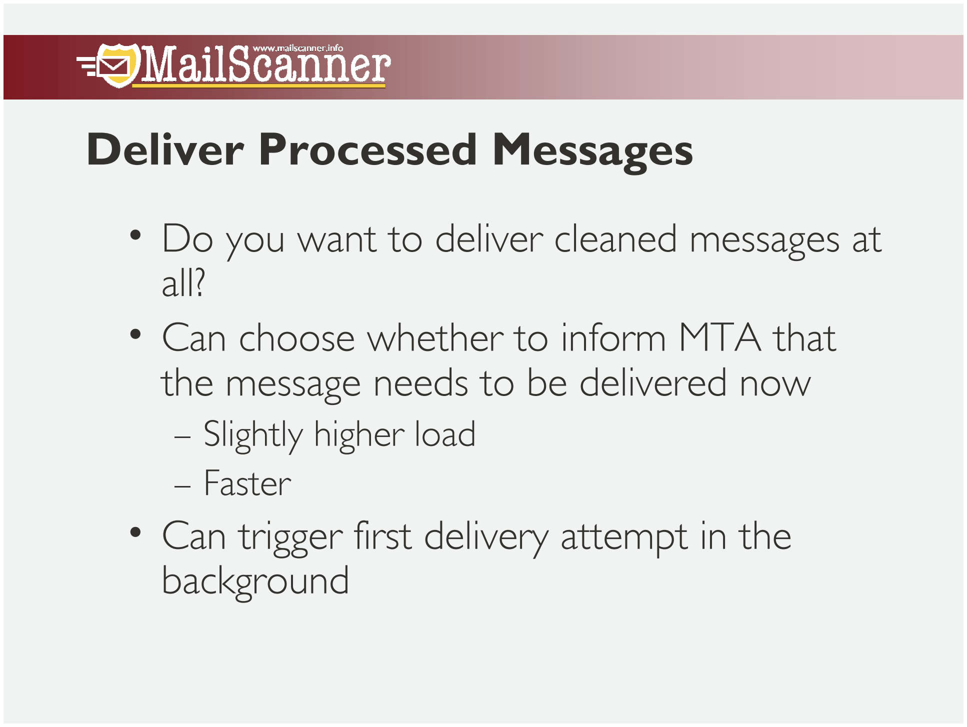
`Non-Forging Viruses = Joke/ OF97/ WM97/ W97M/`

Strings listed here will be searched for in the output of the virus scanners. It works to achieve the opposite effect of the "Silent Viruses" listed above. If a string here is found in the output of the virus scanners, then the message will be treated as if it were not infected with a "Silent Virus". If a message is detected as both a silent virus and a non-forging virus, then the **non-**

forging status will override the silent status. In simple terms, you should list virus names (or parts of them) that you know do *not* forge the From address.
A good example of this is a document macro virus or a Joke program. Another word that can be put in this list is the special keyword Zip-Password : inserting this will cause senders to be warned about password-protected zip files, when they are not allowed. This will over-ride the All-Viruses setting in the list of "Silent Viruses" above.

5.11

Message Responses


Deliver Processed Messages

- Do you want to deliver cleaned messages at all?
- Can choose whether to inform MTA that the message needs to be delivered now
  - Slightly higher load
  - Faster
- Can trigger first delivery attempt in the background

You may not want your users to be bothered by viruses at all. This can be very useful for users downloading their email over slow dial-up connections, especially if the bulk of your viruses are mass-mailing worms which create messages containing no meaningful content.

There is little reason, except for testing, to set MailScanner into the “queue” delivery mode. The first delivery attempt, if done in the background, is very fast and adds little extra load. In “queue” mode the latency of the MailScanner will appear to be a lot greater as the messages will only be delivered by the periodic queue runner processes, which often only run once an hour on large systems.

Attempting to use the “batch” method in conjunction with delivering the messages in the foreground will greatly drop the throughput of your MailScanner, as each of the MailScanner processes will spend most its time waiting for these initial delivery attempts to succeed or timeout. The delivery attempt should always be made in the background.

`Deliver Cleaned Messages = yes`

Do you want to deliver messages once they have been cleaned of any viruses? By making this a ruleset, you can re-create the "Deliver From Local" facility of previous versions.

`Delivery Method = batch`

Attempt immediate delivery of messages, or just place them in the outgoing queue for the MTA to deliver when it wants to?

batch	attempt delivery of messages, in batches of up to 20 at once.
queue	just place them in the queue and let the MTA find them.

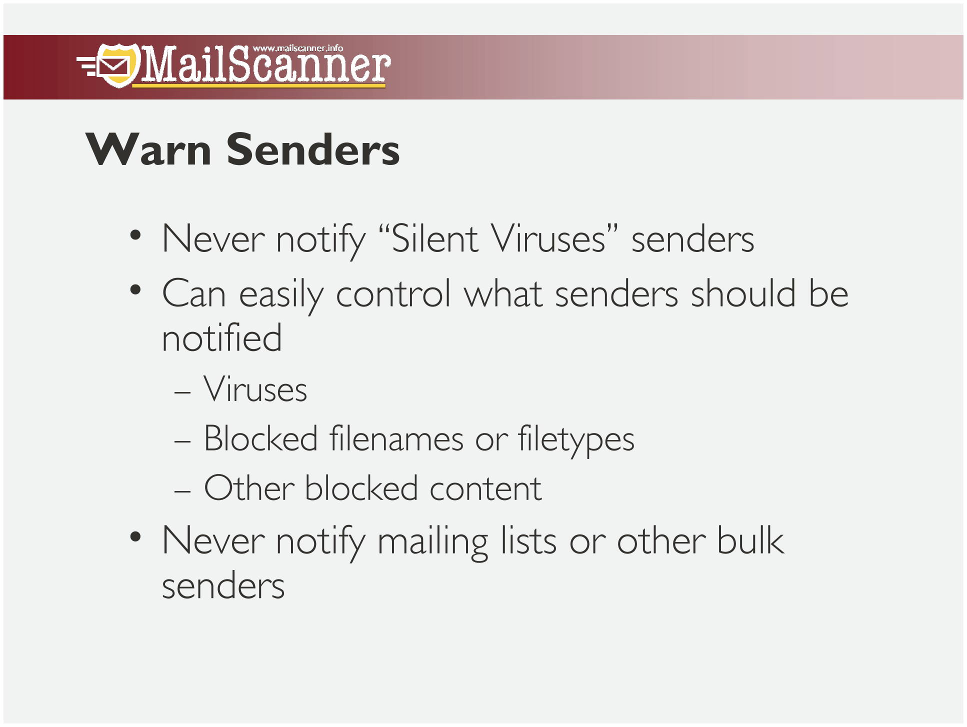
This can also be the filename of a ruleset. For example, you could use a ruleset here so that messages coming to you are immediately delivered, while messages going to any other site are just placed in the queue in case the remote delivery is very slow.

`Deliver In Background = yes`

When attempting delivery of outgoing messages, should we do it in the background or wait for it to complete? The danger of doing it in the background is that the machine load goes ever upwards while all the slow sendmail processes run to completion. However, running it in the foreground may cause the mail server to run too slowly.

`Sendmail2 = /usr/sbin/sendmail`

Sendmail2 is provided for Exim and other non-sendmail users. It is the command used to attempt delivery of outgoing cleaned/disinfected messages. This is not usually required for sendmail. This can also be the filename of a ruleset.
For Exim users: Sendmail2 = /usr/sbin/exim -C /etc/exim/exim_send.conf
For sendmail users: Sendmail2 = /usr/sbin/sendmail



Warn Senders

- Never notify "Silent Viruses" senders
- Can easily control what senders should be notified
  - Viruses
  - Blocked filenames or filetypes
  - Other blocked content
- Never notify mailing lists or other bulk senders

These rules may appear to overlap with the "Silent Viruses" settings. However a warning to a sender will only be produced if all these options agree that the sender should be warned. If a virus or attack is present in the "Silent Viruses" list, and/or the relevant setting here is set to "no", no warning message will be sent to the sender.

It is normal and advisable to warn senders of problems such as blocked content, as this may help them to repackage their file attachments in a suitable safe way. Another user might be to advise the sender of your e-mail use policy so that they understand why their message was not delivered.

Mailing lists are commonly configured so that the sender address is the mailing list manager (either automated or human). Sending rejection warning messages to the sender address will either annoy the human or may even cause your email address to be removed from the mailing list completely. The owners of mailing lists tend not to care if you cannot receive your copy of the list postings, that is your problem and not theirs. Fortunately mailing list management systems usually label their email messages with a "Precedence:" header. MailScanner looks for this header and will not send warnings to mailing lists.

`Notify Senders = yes`

Do you want to notify the people who sent you messages containing viruses or badly-named filenames? This can also be the filename of a ruleset.

`Notify Senders Of Viruses = no`

If "Notify Senders" is set to yes, do you want to notify people who sent you messages containing viruses? The default value has been changed to "no" as most viruses now fake sender addresses and therefore should be on the "Silent Viruses" list. This can also be the filename of a ruleset.

`Notify Senders Of Blocked Filenames Or Filetypes = yes`

I "Notify Senders" is set to yes, do you want to notify people who sent you messages containing attachments that are blocked due to their filename or file contents? This can also be the filename of a ruleset.

`Notify Senders of Blocked Size Attachments = no`

If "Notify Senders" is set to yes, do you want to notify people who sent you messages containing attachments which are outside the size limits set in “Maximum Attachment Size” or “Minimum Attachment Size”. This can also be the filename of a ruleset.

`Notify Senders of Other Blocked Content = yes`

If "Notify Senders" is set to yes, do you want to notify people who sent you messages containing other blocked content, such as partial messages or messages with external bodies? This can also be the filename of a ruleset.

`Never Notify Senders Of Precedence = list bulk`

If you supply a space-separated list of message "precedence" settings, then senders of those messages will not be warned about anything you rejected. This is particularly suitable for mailing lists, so that any MailScanner responses do not get sent to the entire list.

`Sender Content Report = %report-dir%/sender.content.report.txt`
`Sender Error Report = %report-dir%/sender.error.report.txt`

`Sender Bad Filename Report = %report-dir%/sender.filename.report.txt`

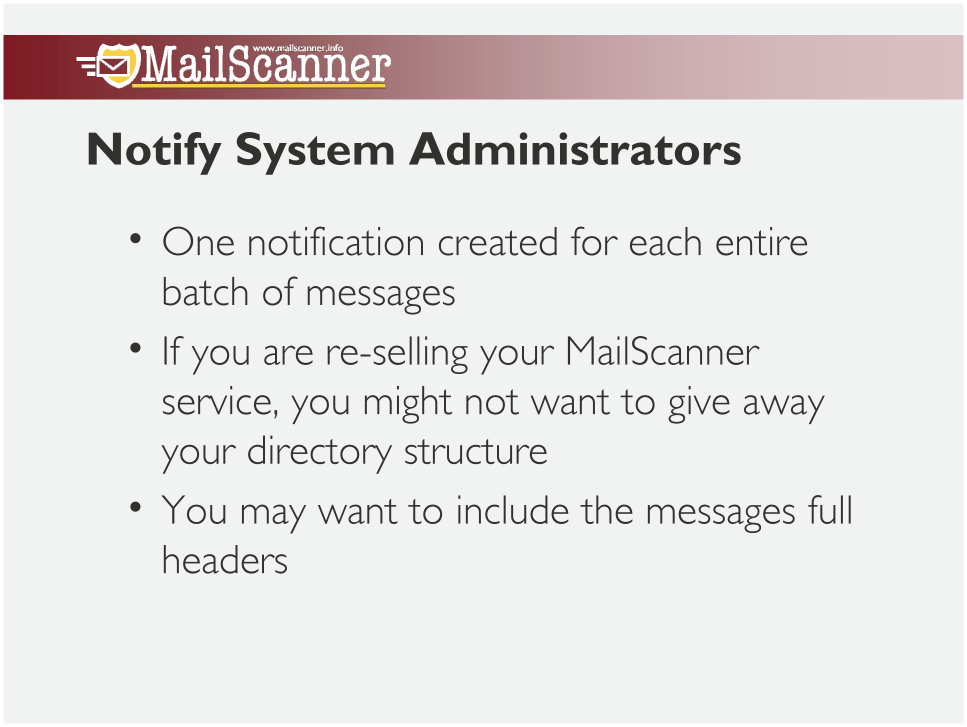
`Sender Virus Report = %report-dir%/sender.virus.report.txt`

Set where to find the messages that are delivered to the sender, when they sent an email containing either an error, banned content, a banned filename or a virus infection. These can also be the filenames of rulesets.

`Sendmail = /usr/sbin/sendmail`

Set how to invoke MTA when sending messages MailScanner has created

(e.g. to sender/recipient saying "found a virus in your message")
This can also be the filename of a ruleset.



Notify System Administrators

- One notification created for each entire batch of messages
- If you are re-selling your MailScanner service, you might not want to give away your directory structure
- You may want to include the messages full headers

In addition to the syslog output produced by MailScanner, it can also generate 1 message per batch containing a list of all the virus scanner reports generated by that batch. This can be very useful when generating statistics of the number of different types of viruses that were detected.

On busy systems it is advisable to keep the size of these messages down to a minimum, to reduce network traffic overhead. All the messages have exactly the same “Subject:”, regardless of the type of attack encountered, just to make automatic processing easier. It is unlikely that a human being would ever want to read all of these messages.

`Send Notices = yes`

> Notify the local system administrators ("Notices To") when any infections are found? This can also be the filename of a ruleset.

`Notices Include Full Headers = no`

> Include the full headers of each message in the notices sent to the local system administrators? This can also be the filename of a ruleset.

`Hide Incoming Work Dir In Notices = no`

Hide the directory path from all the system administrator notices. The extra directory paths give away information about your setup, and tend to just confuse users but are still useful for local sys admins. This can also be the filename of a ruleset.

`Notice Signature = -- \nMailScanner\nEmail Virus Scanner\nwww.mailscanner.info`

What signature to add to the bottom of the notices. To insert a line-break in there, use the sequence "\n".

`Notices From = MailScanner`

The visible part of the email address used in the "From:" line of the notices. The <user@domain> part of the email address is set to the "Local Postmaster" setting.

`Notices To = postmaster`

Where to send the notices. This can also be the filename of a ruleset.

`Local Postmaster = postmaster`

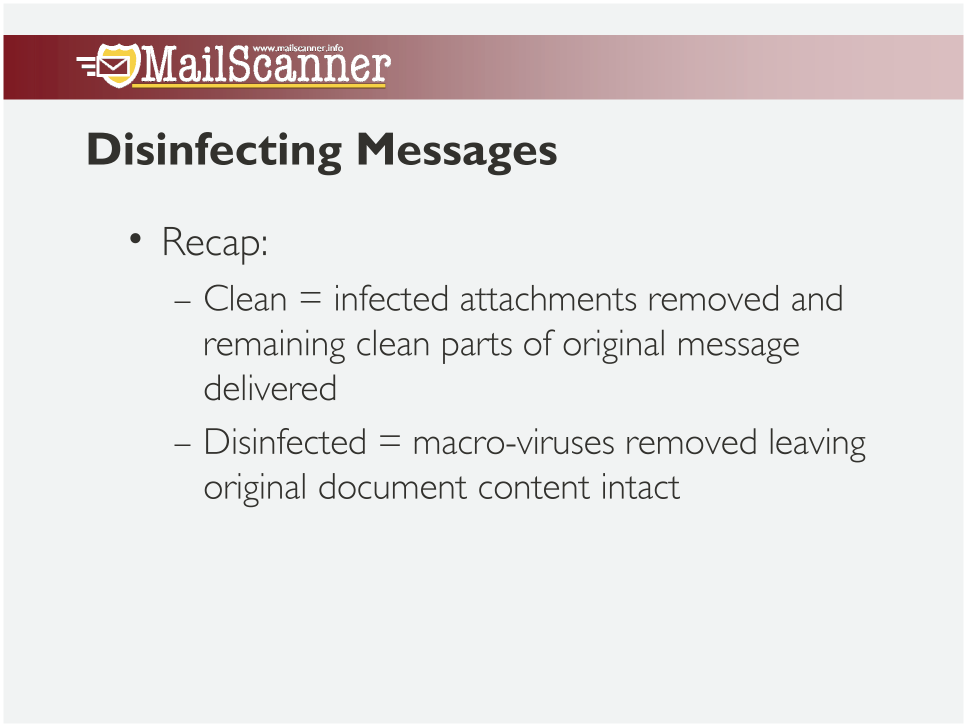
Address of the local Postmaster, which is used as the "From" address in virus warnings sent to users. This can also be the filename of a ruleset.

`Sendmail = /usr/sbin/sendmail`

Set how to invoke MTA when sending messages MailScanner has created
(e.g. to sender/recipient saying "found a virus in your message")
This can also be the filename of a ruleset.

5.12

Macro-virus Disinfection

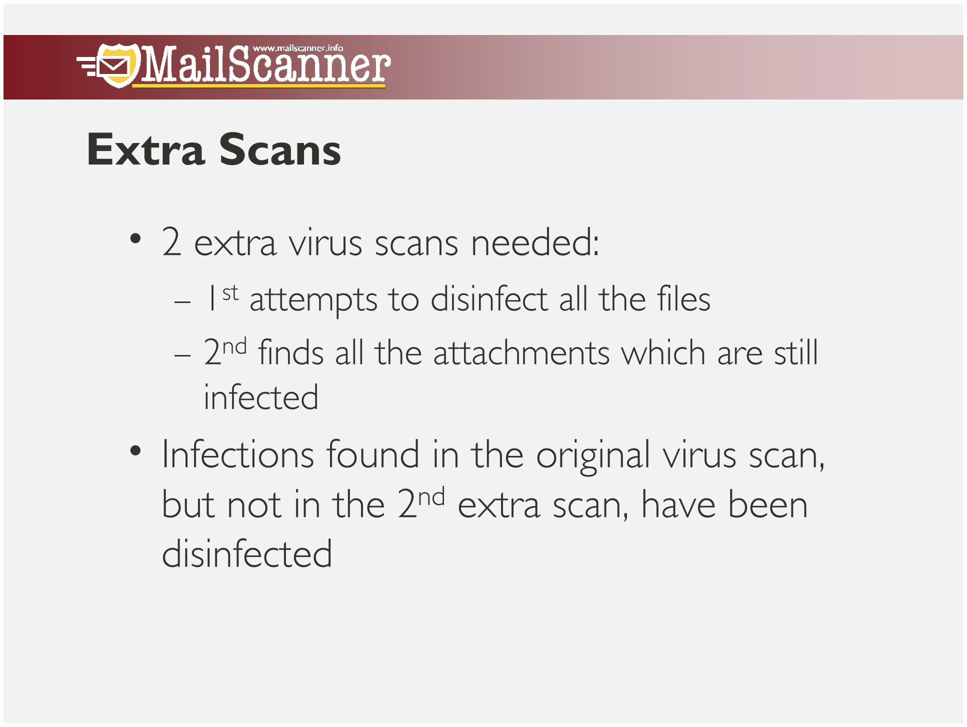


Disinfecting Messages

- Recap:
  - Clean = infected attachments removed and remaining clean parts of original message delivered
  - Disinfected = macro-viruses removed leaving original document content intact

The disinfection process adds a significant overhead to the processing of each batch, as the pass done by the virus scanning engine to actually disinfect the messages is usually very slow in comparison to a simple scanning pass. As macro-viruses are quite uncommon, and are the only viruses which can be usefully disinfected, this feature is now rarely used.

Note that some virus scanning engines do not support disinfection at all, so enabling this feature would have no effect other than to waste time on extra scanning passes. ClamAV is a good example.



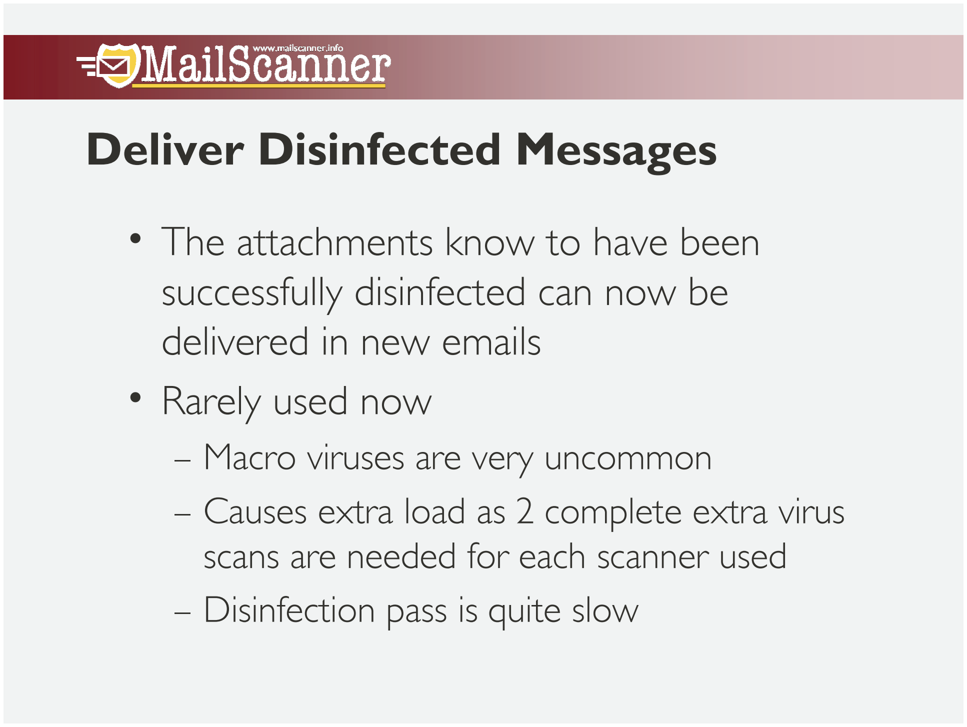
Extra Scans

- 2 extra virus scans needed:
  - 1st attempts to disinfect all the files
  - 2nd finds all the attachments which are still infected
- Infections found in the original virus scan, but not in the 2nd extra scan, have been disinfected

During the disinfection pass, the virus scanning engines produce output saying which files they have successfully disinfected. MailScanner does not trust (or even use) this information at all. After the disinfection pass, another scanning pass is done to find the new list of virus infections. Files which were included in the initial virus scanning pass, but which are missing from the final scanning pass, are the only files that have been successfully disinfected.

`Deliver Disinfected Files = no`

Should I attempt to disinfect infected attachments and then deliver the clean ones. "Disinfection" involves removing viruses from files (such as removing macro viruses from documents). "Cleaning" is the replacement of infected attachments with "VirusWarning.txt" text attachments. Less than 1% of viruses in the wild can be successfully disinfected, as macro viruses are now a rare occurrence. So the default has been changed to "no" as it gives a significant performance improvement. This can also be the filename of a ruleset.



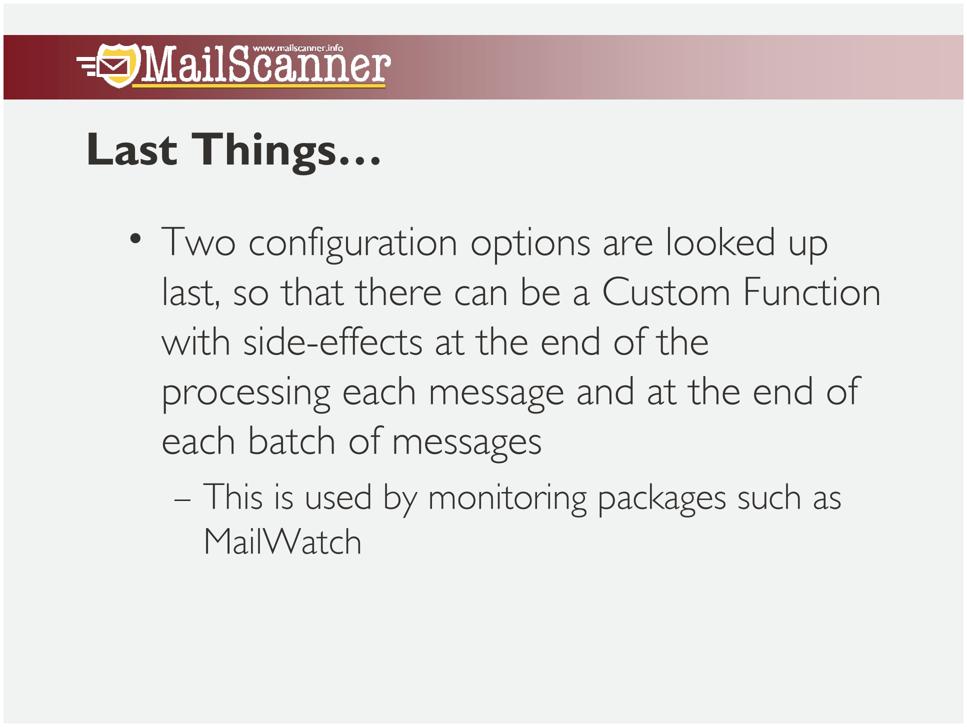
Deliver Disinfected Messages

- The attachments know to have been successfully disinfected can now be delivered in new emails
- Rarely used now
  - Macro viruses are very uncommon
  - Causes extra load as 2 complete extra virus scans are needed for each scanner used
  - Disinfection pass is quite slow

Each disinfected file is put into a completely new message generated by MailScanner, which is sent to each of the original recipients of the infected message. The contents of the message can be completely customised as normal.

`Disinfected Report = %report-dir%/disinfected.report.txt`

> Set where to find the message text sent to users explaining about the attached disinfected documents. This can also be the filename of a ruleset.



Last Things…

- Two configuration options are looked up last, so that there can be a Custom Function with side-effects at the end of the processing each message and at the end of each batch of messages
  - This is used by monitoring packages such as MailWatch

A major feature of Custom Functions in the configuration is that they can have all sorts of intentional side-effects. One of the best examples of this is the "MailWatch" package which provides a graphical interface to MailScanner when running on a lightly-loaded single email gateway. It is driven by a Custom Function which logs data about each message to a SQL database.

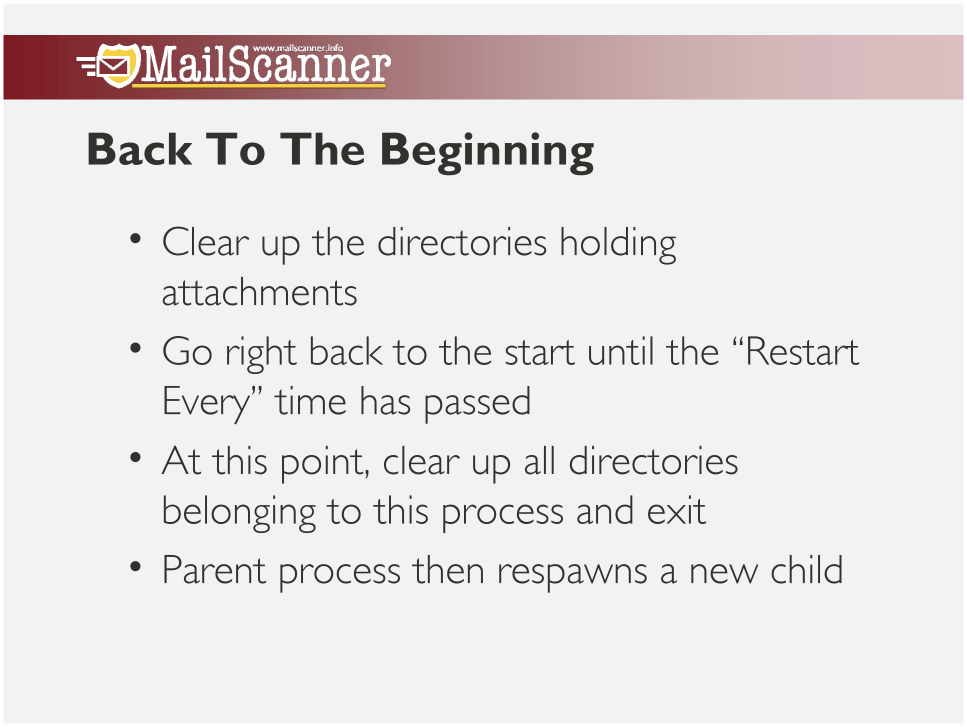
The result of this configuration option is never used, but it is guaranteed that this will be the last configuration option that is evaluated in the processing of any batch of messages.

`Always Looked Up Last = no`

This option is intended for people who want to log more information about messages than what is put in syslog. It is intended to be used with a Custom Function which has the side-effect of logging information, perhaps to an SQL database, or any other processing you want to do after each message is processed. Its value is completely ignored, it is purely there to have side effects. If you want to use it, read CustomConfig.pm and any files in the …/MailScanner/CustomFunctions directory.

`Always Looked Up Last After Batch = no`

This is intended for users who want to apply a Custom Function at the end of an entire batch of messages.



Back To The Beginning

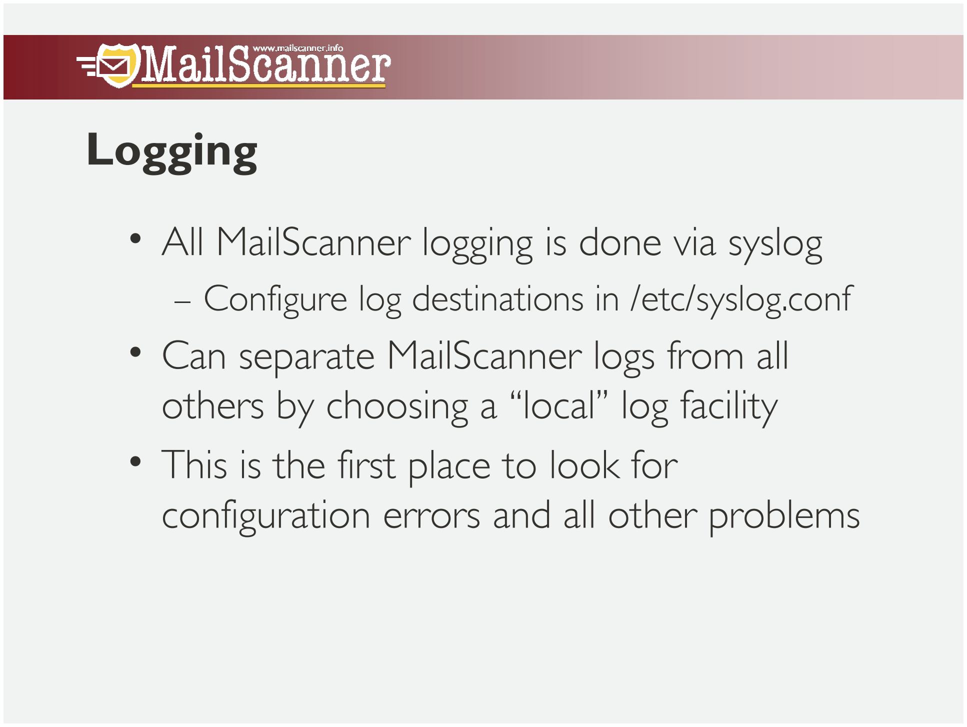
- Clear up the directories holding attachments
- Go right back to the start until the “Restart Every” time has passed
- At this point, clear up all directories belonging to this process and exit
- Parent process then respawns a new child

In the “Incoming Work Dir” there is both a directory and a header-file for every message in the batch, containing all the necessary attachments and sections of each message. These are all deleted before the start of the next message batch.

When the process has been alive for more than “Restart Every” seconds, it will naturally and gently die of old age. Before it dies it is careful to delete all the other directories that will have been created for the process. It is worth noting at this point that the MailScanner processes should always be killed off gently if stopped manually. They will take a few seconds to clean up, but otherwise potentially large directories of temporary files may be left on your system, particularly inside the “Incoming Work Dir”.

When a child process dies, this is detected by the parent process which automatically starts up a new replacement. Because the old process was completely removed, any resource leaks contained in it will have been recovered by the operating system. This makes MailScanner systems very reliable, and they can easily run for several months without any intervention whatsoever, other than to install any virus scanner upgrades that cannot be done automatically. This usually only happens when the vendor releases a new version of the scanning engine, as opposed to the continuous upgrading of virus signature files.

6 Administration



Logging

- All MailScanner logging is done via syslog
  - Configure log destinations in /etc/syslog.conf
- Can separate MailScanner logs from all others by choosing a “local” log facility
- This is the first place to look for configuration errors and all other problems

Logging is all done via syslog so that the logging can be easily configured without adding any extra complexity to MailScanner, such as the ability to log events of different importance to different places, generate email alerts, log across the network etc. All the logging is done with the same syslog “facility” name, but various different precedence values are used for information of different types, including debug, info, notice, warning and error.

The destination of any individual facility/precedence pair can be configured in /etc/syslog.conf. It is trivial to, for example, log a large cluster of systems to a single management host running syslogd.

`Syslog Facility = mail`

> This is the syslog "facility" name that MailScanner uses. If you don't know what a syslog facility name is, then either don't change this value or else go and read "man syslog.conf". The default value of "mail" will cause the MailScanner logs to go into the same place as all your other mail logs.

`Log Silent Viruses = yes`

> Do you want to log all silent viruses as well as all other viruses? Useful for gathering statistics automatically from your logs.

`Log Spam = no`

Do you want all spam to be logged? Useful if you want to gather spam statistics from your logs, but can increase the system load quite a bit if you get a lot of spam.

`Log Non Spam = no`

Do you want all non-spam to be logged? Useful if you want to see all the SpamAssassin reports of mail that was marked as non-spam. Note: It will generate a lot of log traffic.

`Log Permitted Filenames = no`

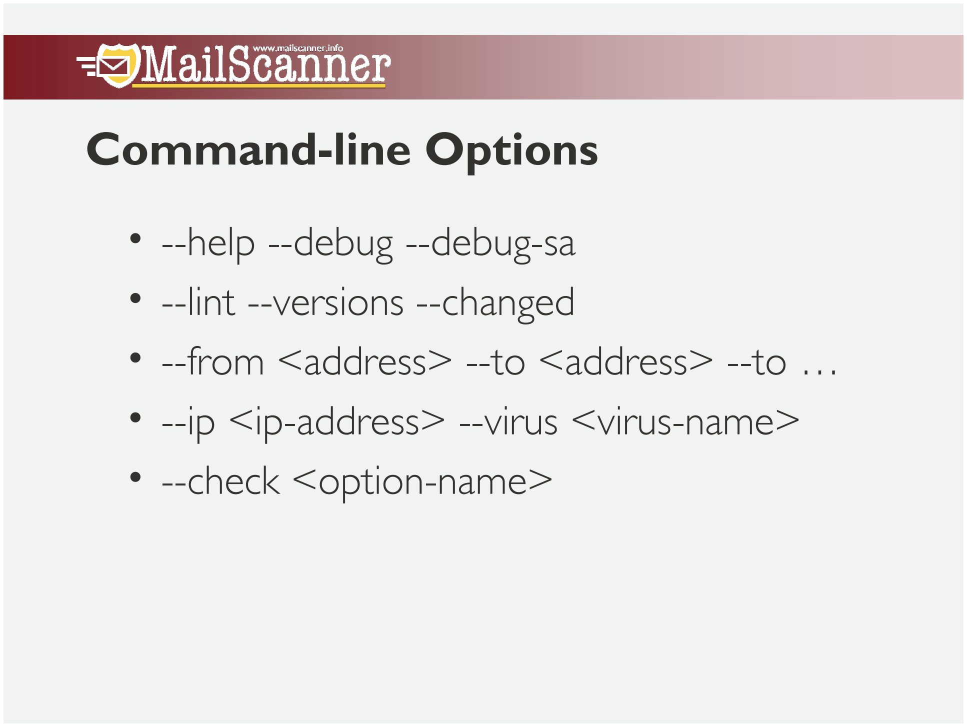
Log all the filenames that are allowed by the Filename Rules, or just the filenames that are denied? This can also be the filename of a ruleset.

`Log Permitted Filetypes = no`

Log all the filenames that are allowed by the Filetype Rules, or just the filetypes that are denied? This can also be the filename of a ruleset.

`Log Speed = no`

Do you want to log the processing speed for each section of the code for a batch? This can be very useful for diagnosing speed problems, particularly in spam checking.



www.mailscanner.info
MailScanner

Command-line Options

- --help --debug --debug-sa
- --lint --versions --changed
- --from <address> --to <address> --to …
- --ip <ip-address> --virus <virus-name>
- --check <option-name>

The MailScanner executable can take several command-line options, which are listed below

`--help`
: Print the command-line usage and exit

`--debug`
: Stop MailScanner forking into the background, process one batch of messages and then exit. Equivalent to the MailScanner.conf setting “Debug = yes”

`--debug-sa`
: When used in conjunction with --debug print out all the SpamAssassin debug information as well. Equivalent to the MailScanner.conf setting “Debug SpamAssassin = yes”

`--lint`
: Check the MailScanner.conf file, test connection to the SpamAssassin cache database, list all the installed virus scanners and exit.

--versions

Print out the version numbers of the operating system, Perl, MailScanner and all the Perl modules that are used by MailScanner and SpamAssassin. It is very useful diagnostic output if you are posting a problem to the MailScanner mailing list.

--changed

Print out all the configuration option values that have been changed from the default supplied values hard-wired into MailScanner. Very useful when diagnosing problems as it instantly shows what configuration options have been changed by the user.

--from

When using the --check option, this sets the sender's address being used for the test.

--to

When using the --check option, this sets the recipient's address being used for the test. This option may be supplied more than once to build a list of recipients.

--ip

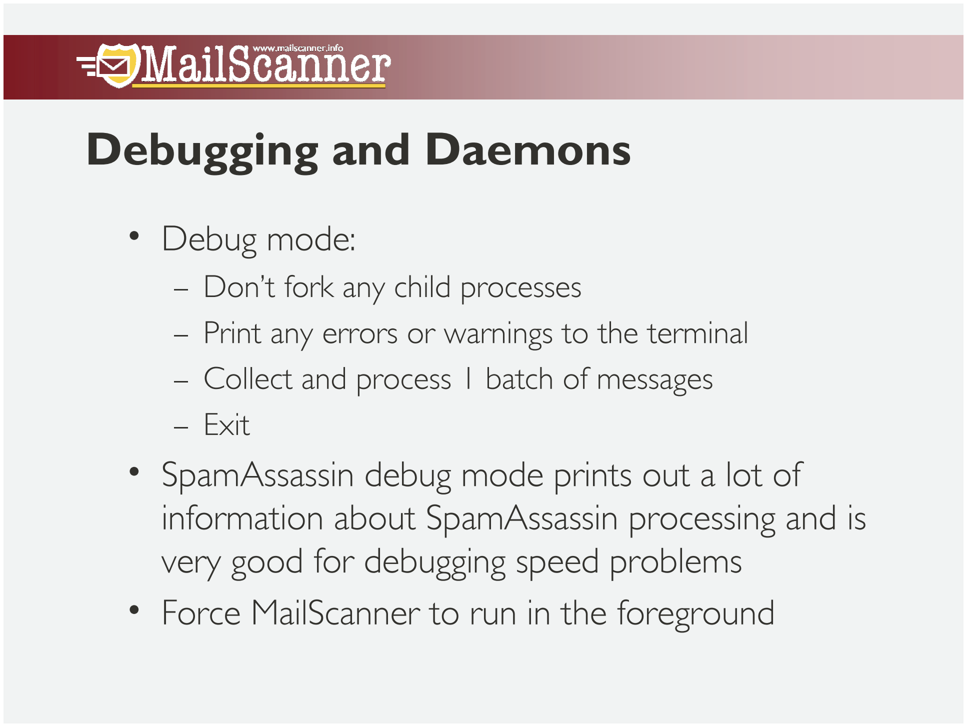
When using the --check option, this sets the IP address of the origin of the message being used for the test.

--virus

When using the --check option, this sets the virus name being used for the test.

--check

This is supplied together with the name of a configuration setting in MailScanner.conf, with all the spaces removed from the name. It takes the values of the other command-line options described above and prints out the result of the configuration setting for a message described by them. It is the easiest way of testing a ruleset or Custom Function.


Debugging and Daemons

- Debug mode:
  - Don't fork any child processes
  - Print any errors or warnings to the terminal
  - Collect and process 1 batch of messages
  - Exit
- SpamAssassin debug mode prints out a lot of information about SpamAssassin processing and is very good for debugging speed problems
- Force MailScanner to run in the foreground

If you enable Debug mode, do not expect to see large amounts of debugging information to be printed. One slight optimisation is that the debugging output code is all commented out in the source code to save dozens of needless system calls during the processing of each batch. If all the debug statements were enabled, the output for a single batch of messages would cover several pages, making it almost impossible to locate the information needed.

Setting “Debug SpamAssassin”, on the other hand, will cause all the internal SpamAssassin debug output to be generated, of which there is quite a lot for every message. It is usually only worth checking the first page or two of this output, as any potential errors should be visible there. Note that any reference to “cloudmark.com” is a reference to the name of the globally central servers used by Razor. It is not possible to run your own mirrors of these servers due to the architecture they have used.

Setting “Debug SpamAssassin = yes” without also setting “Debug = yes” will produce no output, both options must be set to produce any readable SpamAssassin output.

If you are running MailScanner from a daemon monitor utility, you may find you need to stop the daemon from backgrounding itself for the monitoring to work correctly.

`Debug = no`

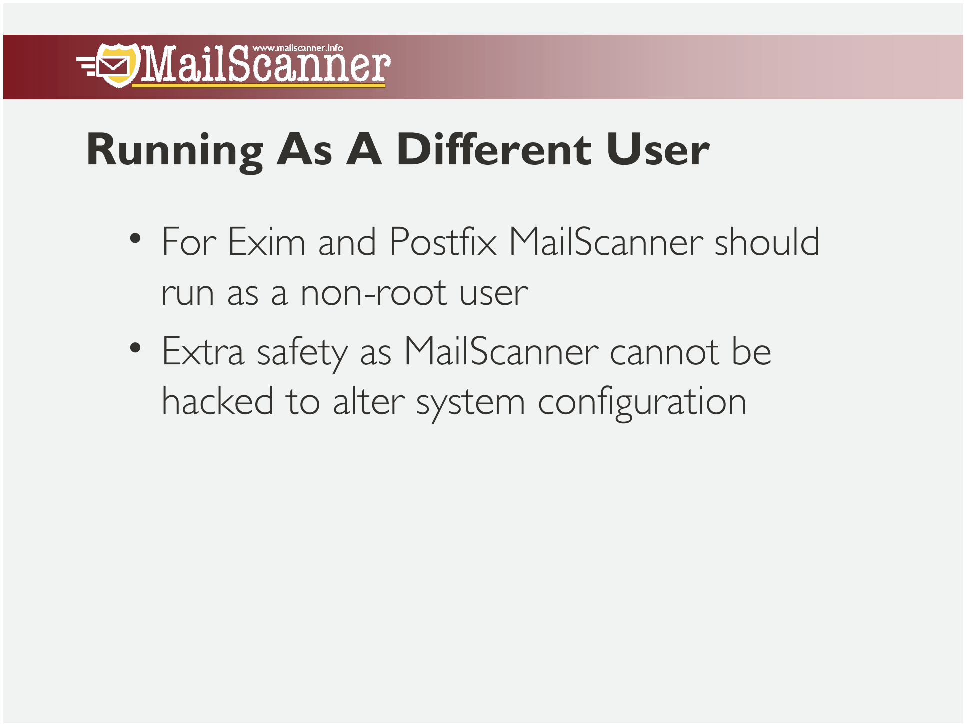
Set Debug to "yes" to stop it running as a daemon and just process one batch of messages and then exit.

`Debug SpamAssassin = no`

Do you want to debug SpamAssassin from within MailScanner?

`Run In Foreground = no`

Do you want MailScanner to run in the foreground and not daemonise itself at startup? This option is automatically set if you set Debug to "yes".



MailScanner www.mailscanner.info

Running As A Different User

- For Exim and Postfix MailScanner should run as a non-root user
- Extra safety as MailScanner cannot be hacked to alter system configuration

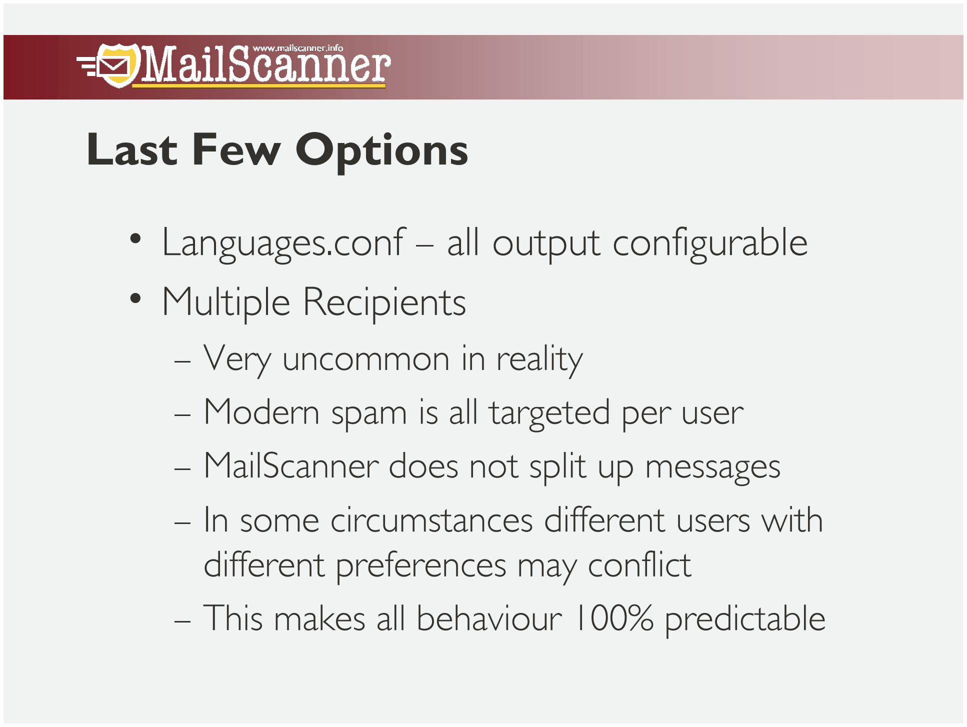
When using sendmail, MailScanner should be run as root. But when running for any other MTA, it should run as the same user as the MTA. When using this feature, you must check that the user can read and write both the “Incoming Work Dir” and the “Quarantine Dir”. Also, beware of the home directory of this user, as this will be the default location for all SpamAssassin and Razor configuration files. If you do not want the SpamAssassin and Razor files to go in the “.spamassassin” and “.razor” directories, you should see the SpamAssassin documentation to discover how to move them.

`Run As User =`

User to run as (not normally used for sendmail) If you want to change the ownership or permissions of the quarantine or temporary files created by MailScanner, please see the "Incoming Work" settings later in this file.

`Run As Group =`

Group to run as (not normally used for sendmail)


MailScanner www.mailscanner.info

Last Few Options

- Languages.conf – all output configurable
- Multiple Recipients
  - Very uncommon in reality
  - Modern spam is all targeted per user
  - MailScanner does not split up messages
  - In some circumstances different users with different preferences may conflict
  - This makes all behaviour 100% predictable

The languages.conf file is very important as it contains the text strings such as "not spam" translated into each language. If you need to totally re-brand MailScanner, you should edit this file, along with each of the reports.

Multiple recipients are mostly used when sending legitimate mail, they are not often used by spammers or viruses. As MailScanner does not split an incoming message into several outgoing messages, there is an inevitable debate about what to do with messages addressed to several users who have conflicting MailScanner settings. In this situation MailScanner normally makes a sensible decision for each configuration option, erring on the side of caution. It tends to deliver a message unless there is a good reason not to. However, the option can be set to make the behaviour totally predictable, by applying per-domain rules in the event of several different per-user rules for a message. This is useful on corporate sites which have many closely related users; ISP's do not need this message as most of their users have no direct relation or association with each other.

See the notes below for the configuration option for an exact specification of the behaviour of this option.

`Hostname = the %org-name% MailScanner`

Name of this host, or a name like "the MailScanner" if you want to hide the real hostname. It is used in the Help Desk note contained in the virus warnings sent to users. This can also be the filename of a ruleset.

`Lockfile Dir = /tmp`

Where to put the virus scanning engine lock files. These lock files are used between MailScanner and the virus signature "-autoupdate" scripts, to ensure that they aren't both working at the same time (which could cause MailScanner to let a virus through).

`Lock Type = flock`

How to lock spool files. Don't set this unless you know you need to. For sendmail, it defaults to "flock". For Exim, it defaults to "posix". No other type is implemented.

`PID File = /var/run/MailScanner.pid`

Set where to store the process id number so you can stop MailScanner

`Language Strings = %report-dir%/languages.conf`

Set where to find all the strings used so they can be translated into your local language. This can also be the filename of a ruleset so you can produce different languages for different messages.

`Split Exim Spool = no`

Are you using Exim with split spool directories? If you don't understand this, the answer is probably "no". Refer to the Exim documentation for more information about split spool directories.

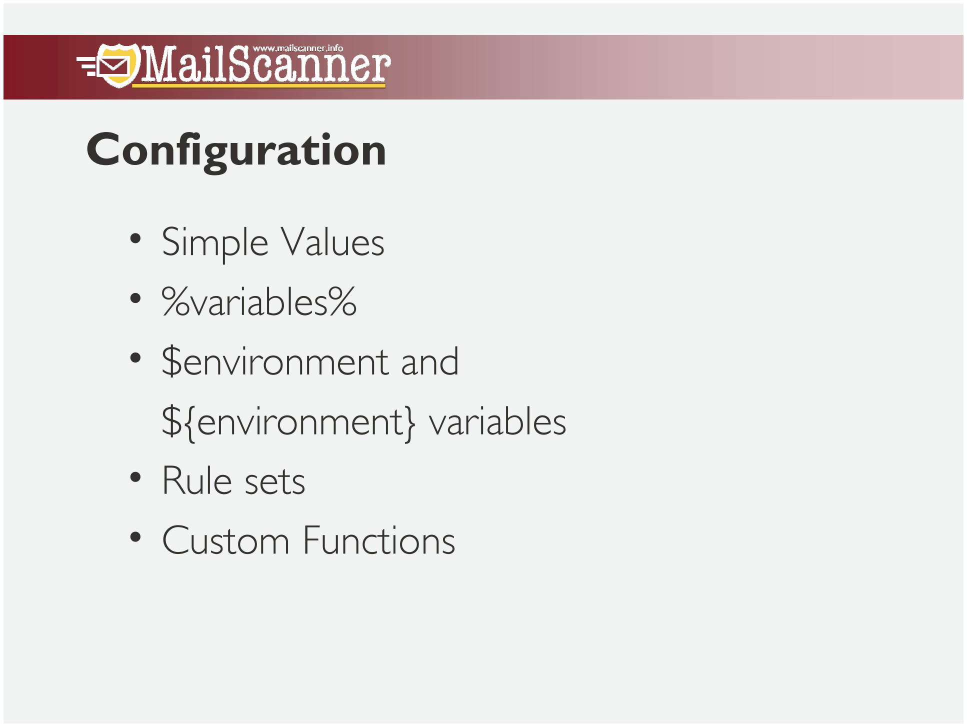
`Use Default Rules With Multiple Recipients = no`

When trying to work out the value of configuration parameters which are using a ruleset, this controls the behaviour when a rule is checking the "To:" addresses. If this option is set to "yes", then the following happens when checking the ruleset:

1. 1 recipient. Same behaviour as normal.
2. Several recipients, but all in the same domain (domain.com for example). The rules are checked for one that matches the string "*@domain.com".
3. Several recipients, not all in the same domain. The rules are checked for one that matches the string "*@*".

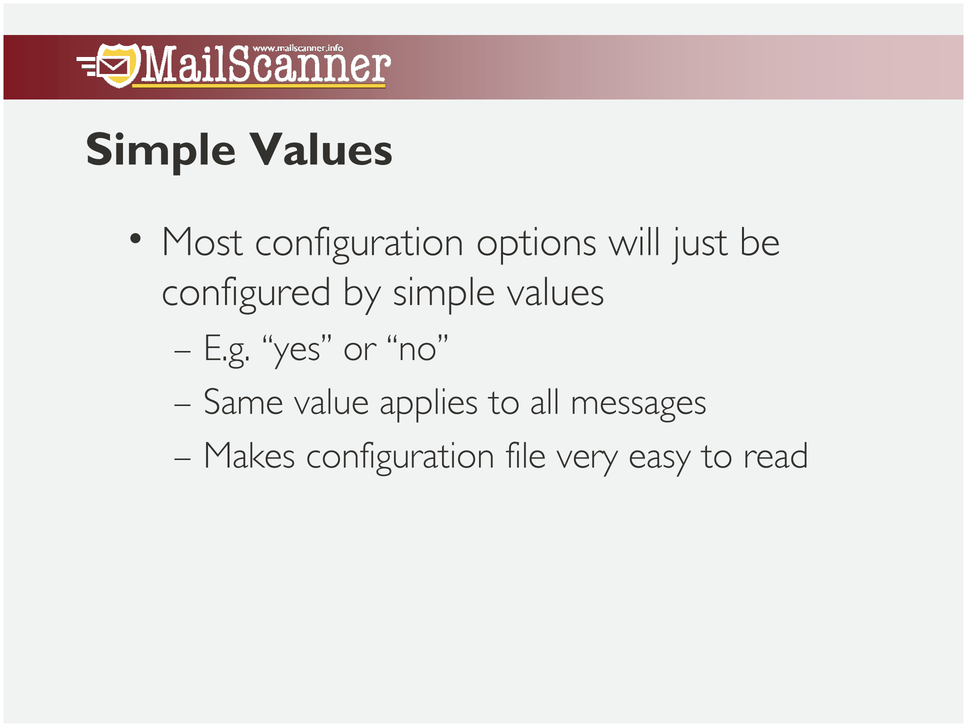
If this option is set to "no", then some rules will use the result they get from the first matching rule for any of the recipients of a message, so the exact value cannot be predicted for messages with more than 1 recipient. This value **cannot** be the filename of a ruleset.

7
Configuration



Configuration

- Simple Values
- %variables%
- $environment and ${environment} variables
- Rule sets
- Custom Functions


Simple Values

- Most configuration options will just be configured by simple values
  - E.g. “yes” or “no”
  - Same value applies to all messages
  - Makes configuration file very easy to read

To help you understand how things work in the results of Custom Functions, here is a description of the different types of values:

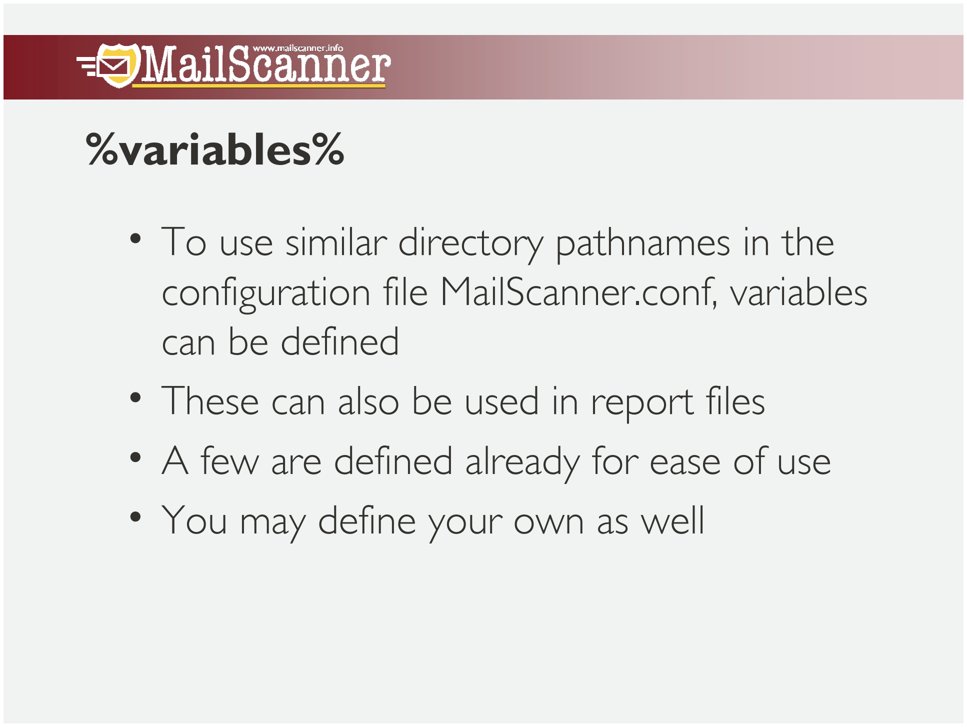
1) Yes, No or a keyword
   These have a numeric or string equivalent internally. 0 is always “No”, 1 is always “Yes”, other keywords are just stored as the string name in lower case.
   This information will be useful if you write your own Custom Functions.

2) Number
   This does not have to be an integer. Thousands and millions can be separated by “_” if you so desire, similarly to the way they can in Perl. So “150_000” is a legitimate number.
   The suffixes “k”, “m” or “g” can be added to represent thousands, millions, and billions respectively. So “150000000” may be more clearly written as “150m”.

3) Filename
   The file must exist, an error will be produced if it does not. No other checks are made on the file.

4) Directory
   The directory must exist, an error will be produced if it does not. No checks are done to ensure the files in the directory are readable or writable.

5) Other
   These can be any text string, including being empty (i.e. blank). No checks are made on the values at all.



MailScanner www.mailscanner.info

%variables%

- To use similar directory pathnames in the configuration file MailScanner.conf, variables can be defined
- These can also be used in report files
- A few are defined already for ease of use
- You may define your own as well

These reduce the number of places where you need to configure the location and language of your MailScanner installation. They can be used in MailScanner.conf and in the report files contained in the %report-dir% directories. A few are pre-defined for you in the supplied configuration to ease installation, but you may define as many as you need.

The pre-defined variables are:

`%report-dir% = /etc/MailScanner/reports/en`
Set the directory containing all the reports in the required language

`%etc-dir% = /etc/MailScanner`
Configuration directory containing this file

`%rules-dir% = /etc/MailScanner/rules`
Rulesets directory containing your ".rules" files

`%org-name% = yoursite`

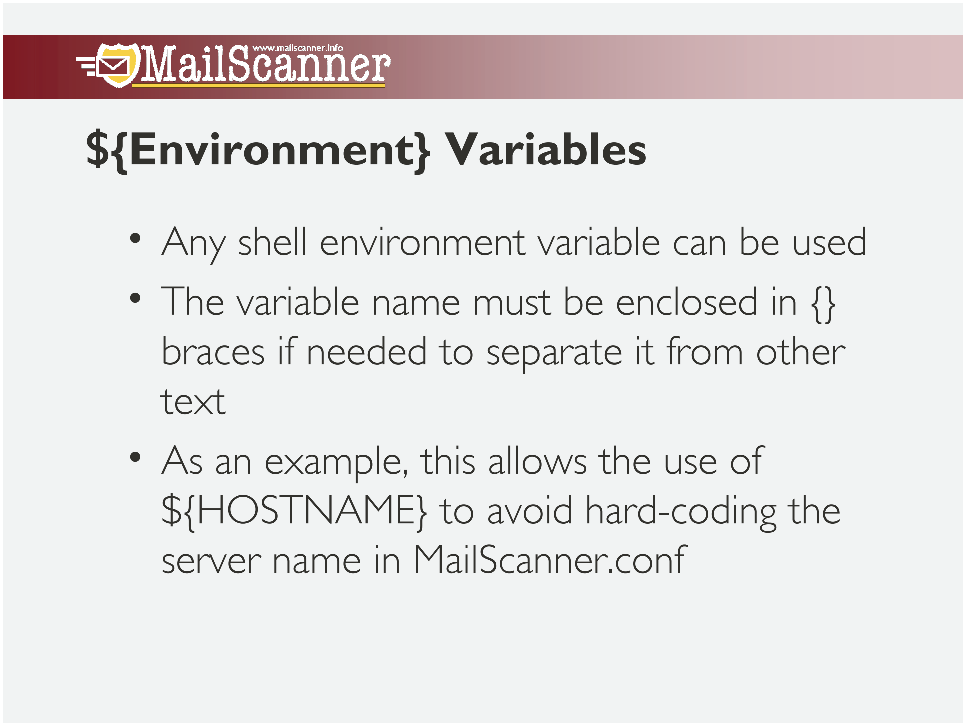
Enter a short identifying name for your organisation below, this is used to make the X-MailScanner headers unique for your organisation. Multiple servers within one site should use an identical value here to avoid adding multiple redundant headers where mail has passed through several servers within your organisation.
Rule: It must not contain any spaces!
Note: Some Symantec scanners complain (incorrectly) about "." characters appearing in the names of headers.

`%org-long-name% = Your Organisation Name Here`

Enter the full name of your organisation, this is used in the signatures at the end of email reports. To split this text over several lines, mark each linebreak by including the 2-character sequence "\n". This text can include spaces and any other punctuation. This allows simple central customisation of all the report message signatures, directing questions to your site and not to me! If it is not changed from the default value supplied, your full domain name is used.

`%web-site% = www.your-organisation.com`

Enter the web address of your organisation's internet presence. This is also used in the signatures at the end of email reports. You may want to set it to the address of a page containing information about your MailScanner service. If it is not changed from the default value supplied, then it is constructed from "www." followed by your full domain name.



${Environment} Variables

- Any shell environment variable can be used
- The variable name must be enclosed in {} braces if needed to separate it from other text
- As an example, this allows the use of ${HOSTNAME} to avoid hard-coding the server name in MailScanner.conf

Normally, shell environment variable names are all upper-case. They can be of the form $HOSTNAME or ${HOSTNAME}. The latter format is used when it is not obvious how to separate the name of the environment variable from any text following it. It is usually safer to use the ${} form, and cannot do any harm.

Any configuration option can also contain the string "\n" which will be replaced by a newline character in the result.

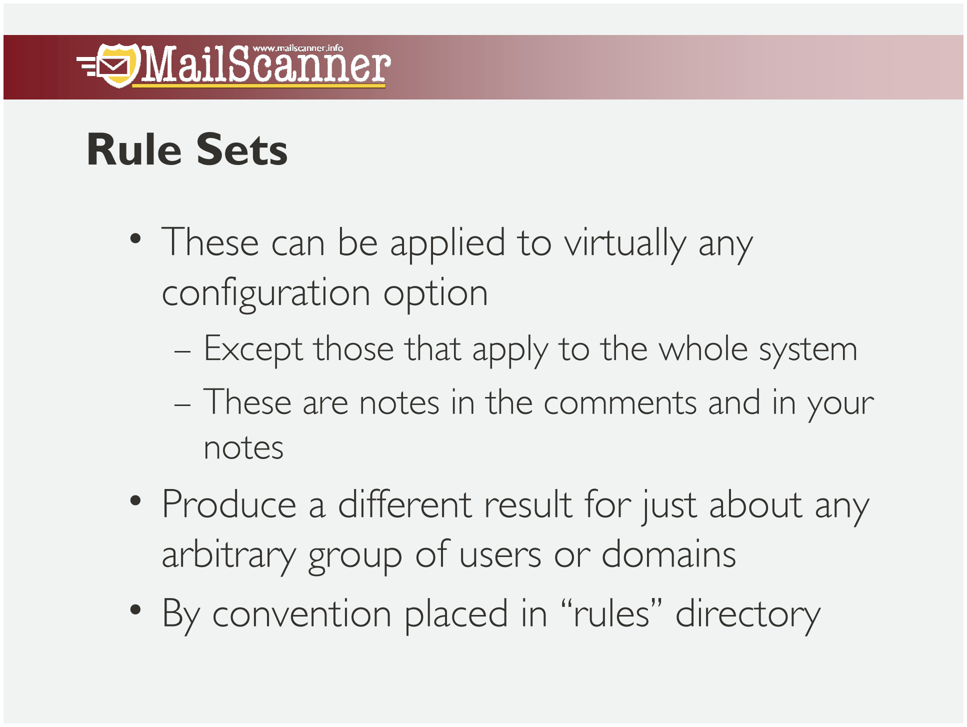
The most common use for this is in a setting such as

`Hostname = The %org-long-name% (${HOSTNAME}) MailScanner`

This nicely demonstrates the use of both %variables% and shell environment ${variables} in order to produce a neat but informative server name in MailScanner reports.

7.1

Rulesets



MailScanner

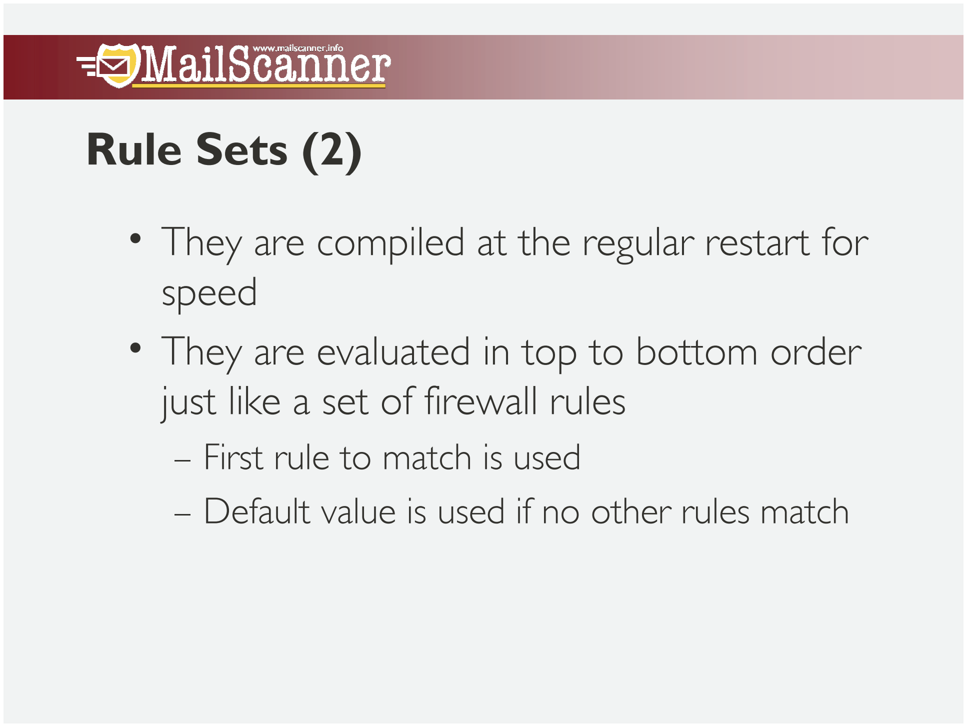
www.mailscanner.info

Rule Sets

- These can be applied to virtually any configuration option
  - Except those that apply to the whole system
  - These are notes in the comments and in your notes
- Produce a different result for just about any arbitrary group of users or domains
- By convention placed in “rules” directory

Rule Sets are one of the most powerful configuration methods in MailScanner. They work rather like firewall rulesets in that they are evaluated from top to bottom, producing the result from the first rule that matches the message. They can be used, for example, to produce the same result for any arbitrary group of users, domains or networks. But they are not restricted to those groups, anything that can be encoded in a regular expression can be matched.

The usual location for them is the “rules” subdirectory within the main MailScanner configuration directory.

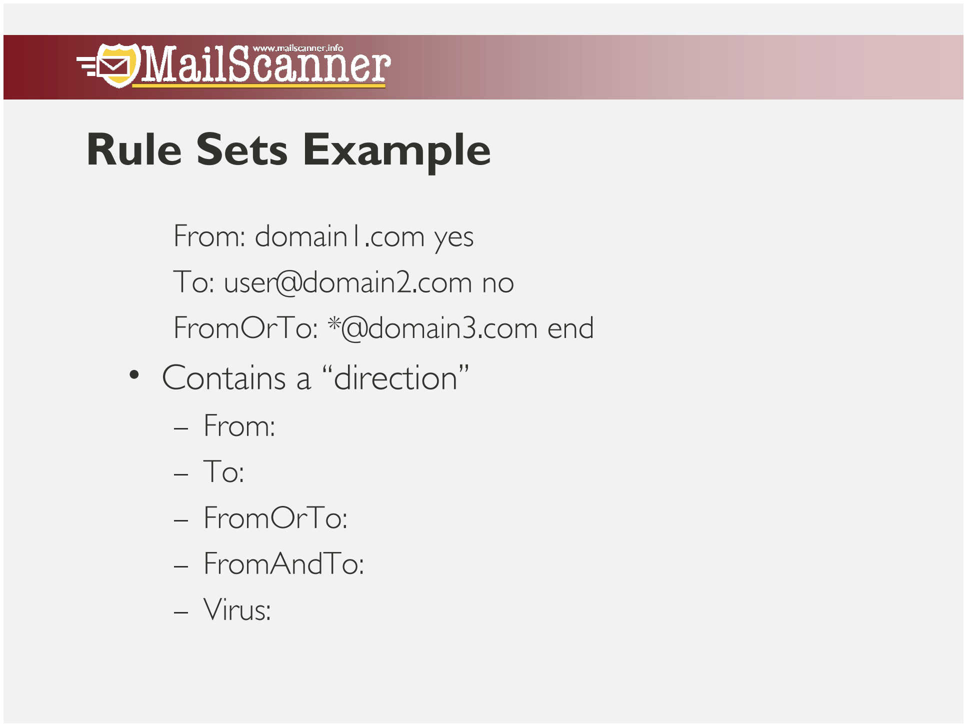


Rule Sets (2)

- They are compiled at the regular restart for speed
- They are evaluated in top to bottom order just like a set of firewall rules
  - First rule to match is used
  - Default value is used if no other rules match

It should be noted that very large rulesets can lower performance as it may need to scan every rule for several configurations options for every message. Very large rulesets are better implemented as simple Custom Functions, optimised to use hash arrays where possible designed around a-priori knowledge of your particular configuration.

The default value can be stored anywhere within the ruleset, it is only matched if no other rules match. It is common for the default value to be set near the top of the file, as this makes it easier to automatically re-generate the rest of the file.



Rule Sets Example

From: domain1.com yes
To: user@domain2.com no
FromOrTo: *@domain3.com end

- Contains a "direction"
  - From:
  - To:
  - FromOrTo:
  - FromAndTo:
  - Virus:

The directions are:

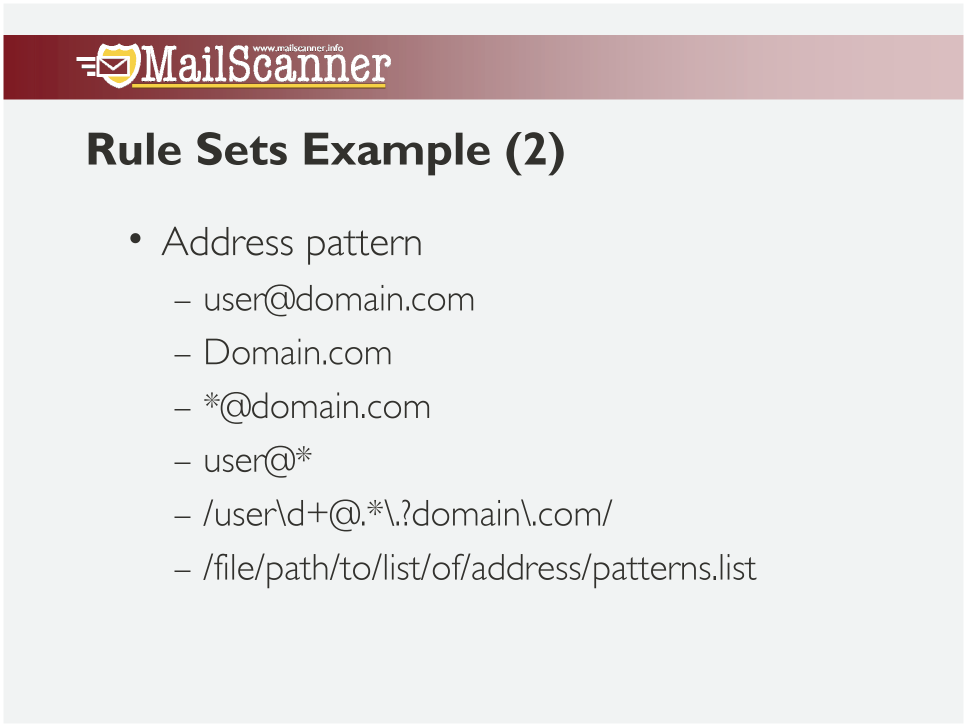
1) From:
   This will match against the sender address

2) To:
   This will match against any of the recipient addresses

3) FromOrTo:
   This is the most commonly used direction as it applies to mail whose recipient or sender addresses match. It is the easiest way of matching all email to or from a particular user or domain.

4) FromAndTo:
   This is rarely used. With a full email address it is no use at all as it would only match email from a user sending to her/himself. However, with a domain name, it will match all internal email within the domain. This is very useful for applying different results to internal company email.

5) Virus:
   This can only be used for rulesets that are evaluated after the virus scanning has been done. It is used by a few people who wish to handle particular viruses or types of virus differently. It is matched against the output of the virus scanning engines, in much the same way as the "Silent Viruses" list.


Rule Sets Example (2)

- Address pattern
  - user@domain.com
  - Domain.com
  - *@domain.com
  - user@*
  - /user\d+@.*\.?domain\.com/
  - /file/path/to/list/of/address/patterns.list

The address pattern can take many forms, including

1) user@domain.com
   Full email address

2) *@domain.com
   All users within a domain.
   * matches any sequence of zero or more characters

3) user@*
   The same username within any domain
   Again, * matches any sequence of zero or more characters

4) domain.com
   The string "*@" is automatically added to the front of this to produce the same result as *@domain.com

5) default
   The literal “default” matches when no other rule matches in the entire ruleset

6) *@*
   This is translated automatically into “default”

7) /regular-expression/
   This is used as given, with no translation

8) 152.78
   Translated into 152.78.0.0-152.78.255.255 and matches email which was delivered to MailScanner from a matching IP address

9) 152.78.
   As above

10) 152.78.0.0/16
    As above

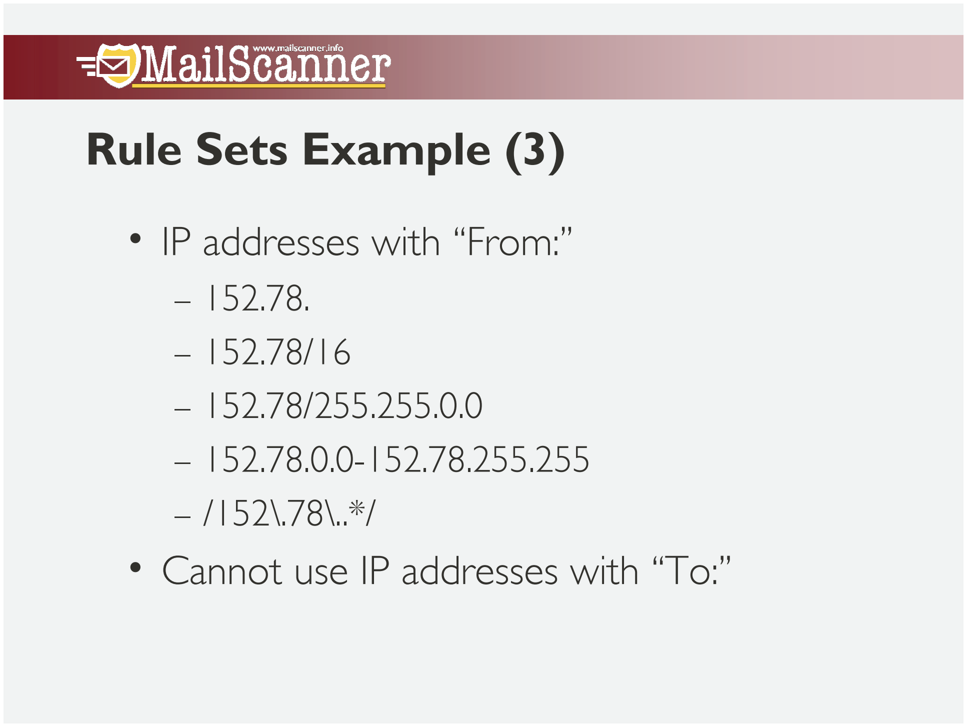
11) 152.78.0.0/255.255.0.0
    As above

12) 152.78.0.0-152.78.255.255
    As above

13) 192.168.2.1
    The exact IP address given, not just any IP address starting with the given number. It will not match 192.168.21.11 or 192.168.21.123, for example

14) /numerical-regular-expression/
    Similar to the regular expression example earlier, except this is matched agains the originating IP address

15) /pathname/filename
    This gives the location of a file containing 1 address pattern per line, using any of the pattern formats given above. It includes the ability to nest these files so that a file can effectively include the contents of another file. Checks are performed to detect loops in the nesting. If they are nested more than 4 levels deep, the deeper files will be ignored and a warning will be logged

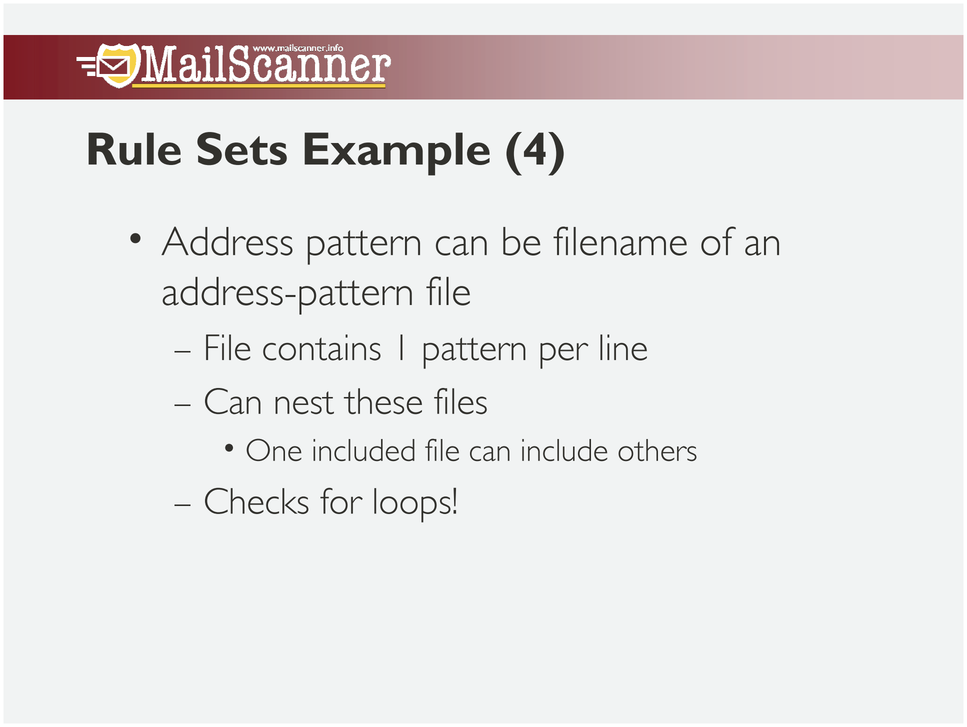


Rule Sets Example (3)

- IP addresses with “From:”
  - 152.78.
  - 152.78/16
  - 152.78/255.255.0.0
  - 152.78.0.0-152.78.255.255
  - /152\.78\..*/
- Cannot use IP addresses with “To:”

It is important to remember that the IP address to which the mail is going to be delivered cannot be determined until the mail delivery process is actually in progress. This is due to the fault-tolerant design of internet email.

So the IP address cannot be used in “FromOrTo:”, “FromAndTo:” or “To:”, but only “From:”.


MailScanner www.mailscanner.info

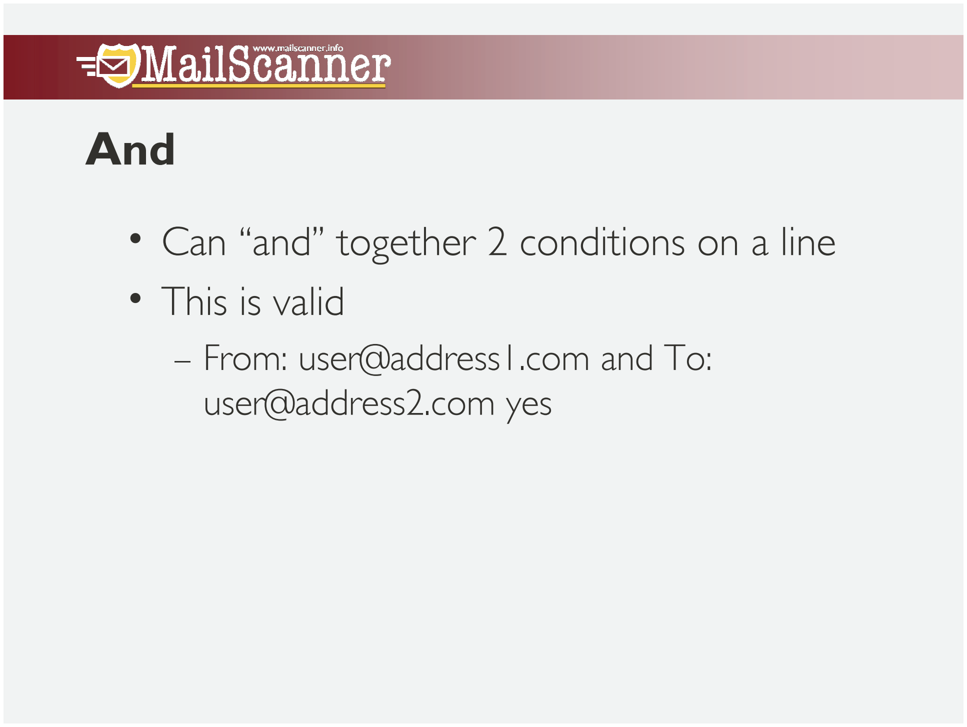
Rule Sets Example (4)

- Address pattern can be filename of an address-pattern file
  - File contains 1 pattern per line
  - Can nest these files
    - One included file can include others
  - Checks for loops!

If you have a large number of similar rules with the same right-hand sides, ie the same result, you can list all the address patterns in another file and refer to it by putting the filename in instead of the address pattern itself. So for the "Spam Checks" option, if you wanted to have a file which listed all the domains for which you provide a spam checking service, you could write a ruleset like this:

```
To:           /etc/MailScanner/spam.check.domains   yes
From Or To:   default                                   no
```

The /etc/MailScanner/spam.check.domains could just contain a list of all the domain names for which you want the "yes" result, one domain per line. This is just a short-cut for having to manually write a rule for each domain who pays you for spam checking. This short-cut may help quite a lot if you automatically generate the list of domain names.

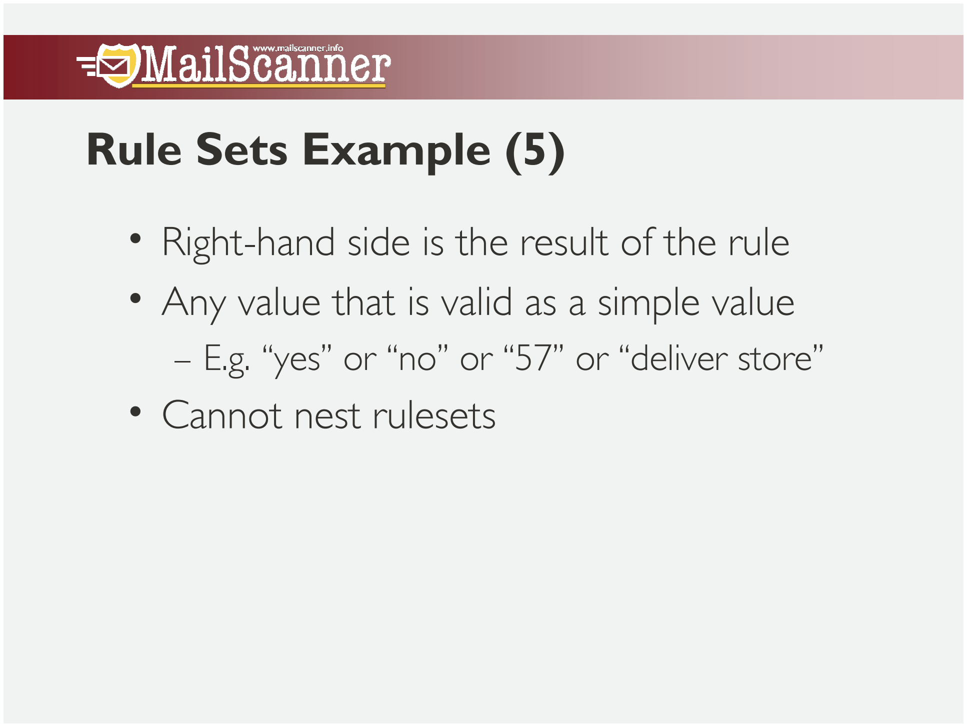


And

- Can “and” together 2 conditions on a line
- This is valid
  - From: user@address1.com and To: user@address2.com yes

The logical “Or” condition can be implemented simply by using 2 matching rules with the same result. Only 2 conditions per rule can be used, the “And” operator may only occur once per rule.

The address patterns either side of the “And” can each be of any type, so you can detect mail associated with a given domain which also originated from one of their IP addresses.



Rule Sets Example (5)

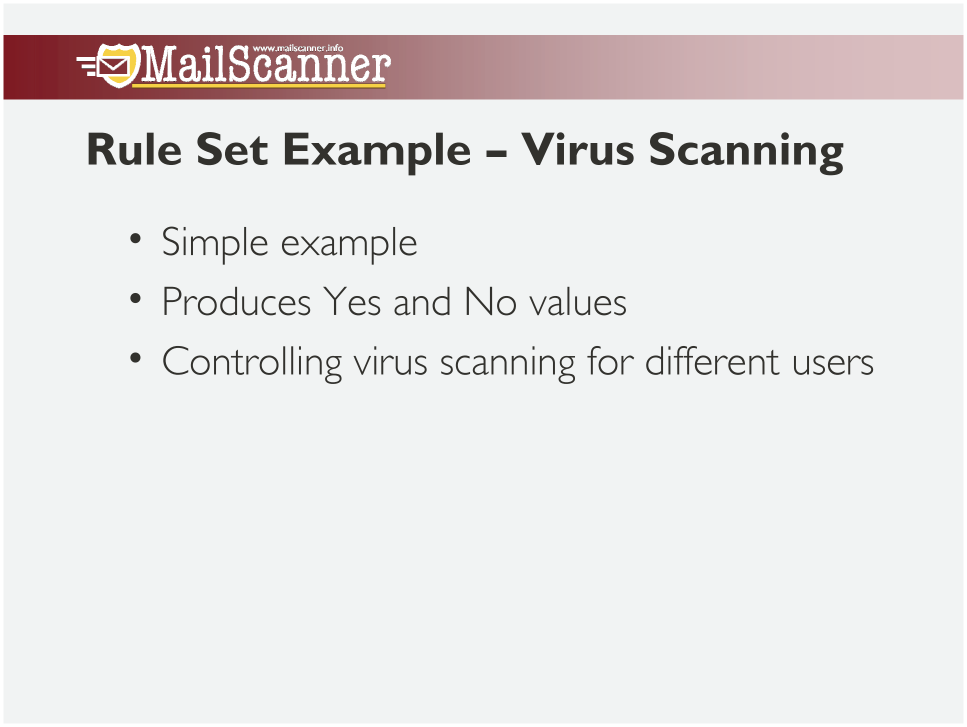
- Right-hand side is the result of the rule
- Any value that is valid as a simple value
  - E.g. "yes" or "no" or "57" or "deliver store"
- Cannot nest rulesets

If you are in doubt as to what can be the result of a rule, the answer is very simple. Any value which could be put in the setting directly in MailScanner.conf can be used as a result, including an empty string.

For example, if using a ruleset for the "Spam Actions" configuration option, a ruleset might containg "store forward user@domain.com" on the right-hand side of a rule. If you were using a ruleset for the "SpamAssassin Required Score" option, then the number "8" would be suitable for the right-hand side of the rule. If the ruleset was for the "Spam Checks" option, then the only valid right-hand sides would be "yes" or "no".

The contents of the right-hand side is simply the setting that you want to apply to the configuration option, when the condition in the rule matches the message.

Rulesets cannot be nested. The only nesting allowed is when using files full of address patterns, as described a few slides ago.



MailScanner www.mailscanner.info

Rule Set Example – Virus Scanning

- Simple example
- Produces Yes and No values
- Controlling virus scanning for different users

In this example, the aim is to do virus scanning for all the domains of your customers, except for a few who have specifically requested that they do not want it for mail going to or coming from them. Also, you do not want to do virus scanning for mail sent to your support staff. You want to scan all other mail for viruses.

The domains who do not want it are nocheck1.com and nocheck2.com. The addresses of all the support staff at your customers' sites are "support" at each site.

In MailScanner.conf, set this:

```
Virus Scanning = %rules-dir%/virus.scanning.rules
```

It is worth mentioning a couple of points about this line. Firstly it uses the %rules-dir% variable which is set at the top of MailScanner.conf, so that if you move it all then changing all the directory names is a lot easier. Secondly, it is good practice to make the name of ruleset files end in ".rules". This is usually not absolutely necessary, but will avoid a few problems in specific situations.

Assuming %rules-dir% is set to /etc/MailScanner/rules, then the following ruleset will be stored in /etc/MailScanner/rules/virus.scanning.rules. In it, put these lines:

```
FromOrTo:       nocheck1.com          no
FromOrTo:       nocheck2.com          no
To:             support@*             no
FromOrTo:       default               yes
```

The last line sets the default value used for all mail that does not have any of the addresses given in any other rule. Note that the default rule does not have to be at the end of the ruleset file, but it is often more clear if you do.

After changing this configuration, you will need to restart or reload MailScanner. On RedHat systems, this can be done most easily with the “service” command:

```
service MailScanner reload
```

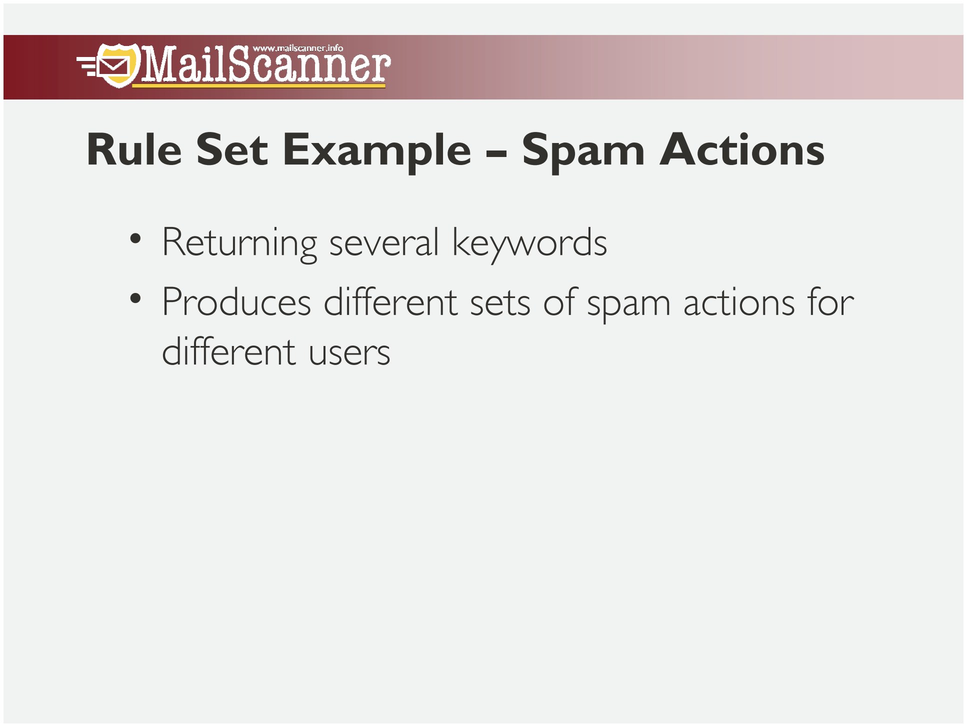
On other Linux systems you can do it like this:

```
Kill -HUP `cat /var/run/MailScanner.pid`
```

On non-Linux systems where MailScanner is installed under /opt you can do it like this:

```
Kill -HUP `cat /opt/MailScanner/var/MailScanner.pid`
```

If you do not do this, the configuration will automatically updated at the next regular MailScanner restart (governed by the “Restart Every” configuration setting).


Rule Set Example – Spam Actions

- Returning several keywords
- Produces different sets of spam actions for different users

This example is a bit more complex than the previous one, in that it returns a string of Spam Actions instead of a simple yes or no value. This example uses the “Spam Actions” setting. This does not affect the actions required for high-scoring spam at all, that is outside the scope of this example.

- Some of your customers want all their spam forwarded to a single account so the support staff there can review it and deliver any messages that were incorrectly tagged as spam. The customers wanting this are forwarding1.com and forwarding2.com.

- One of your customers want all their spam to be delivered but also stored in an archive for later review. The customer wanting this is archive3.com.

- Another customer wants to turn each spam message into an attachment and deliver the resulting message, so that they do not have to look at the real spam message, but still receive it all as they are new users of your MailScanner service and are not yet convinced by the accuracy of MailScanner’s spam detection. The customer wanting this is attach4.com.

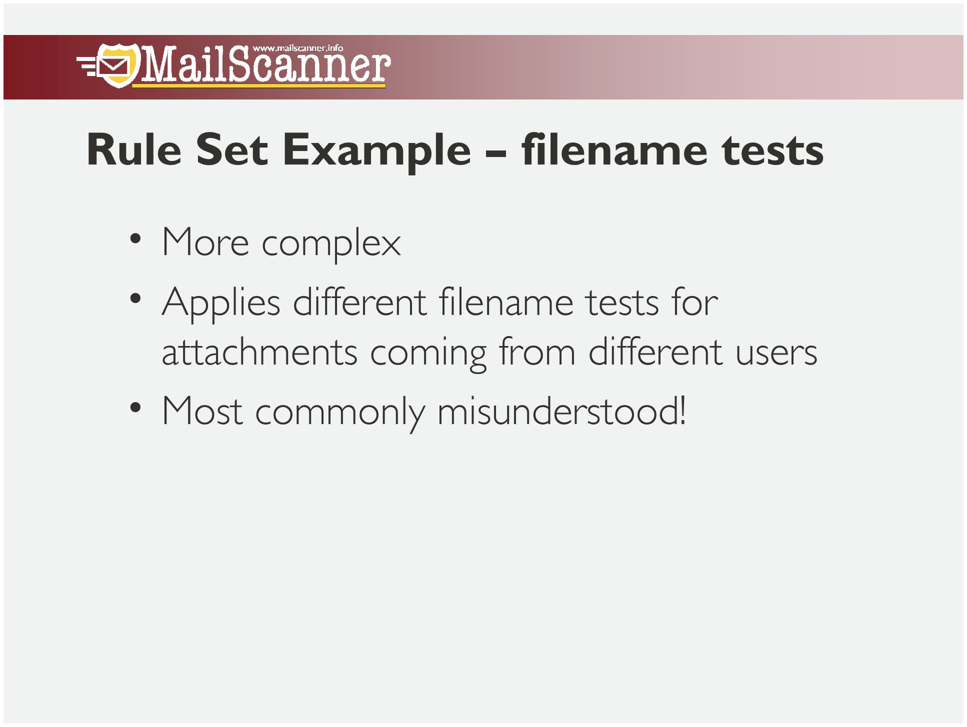
- You want to simply delete all other spam for other customers.

In MailScanner.conf you need to set

Spam Actions = %rules-dir%/spamactions.rules

In /etc/MailScanner/rules/spamactions.rules put these lines:

```
To:         forwarding1.com  forward reviewme@forwarding1.com
To:         forwarding2.com  forward reviewme@forwarding1.com
To:         archive3.com          store deliver
To:         attach4.com           deliver attachment
FromOrTo:   default               delete
```



Rule Set Example – filename tests

- More complex
- Applies different filename tests for attachments coming from different users
- Most commonly misunderstood!

This example is rather more complex, but is one of the most common requests from the MailScanner user community.

Most of your customers are happy with the filename checking provided by the default allow/deny tests supplied with MailScanner.

However, 2 of your customers have slightly different requirements.

- One of them wants to ban its staff from sending out Zip archives ending in ".zip", but allow all Microsoft Word ".doc" documents, regardless of the rest of the filename. Their domain name is domain1.com.

- Another customer wants to allow its support staff to send out ".ini" files, which are used by the software they sell and they need to be able to send these files to their own clients to assist them. Their domain name is domain2.com.

What you need to do first is to use a ruleset which points the “Filename Rules” setting to different combinations of “filename.rules.conf” files. Point the setting at your “.rules” ruleset that will determine which files to use. Put this in MailScanner.conf:

```
Filename Rules = %rules-dir%/filenames.rules
```

In addition to the original filename.rules.conf file supplied with MailScanner, you need to create 2 new “.rules.conf” files containing the differences required by your customers. These new “.rules.conf” files **must** use **tab** characters to separate the 4 parts of each line, and not just spaces.

Create the 2 new “.rules.conf” files. Create a “filename.zip.doc.rules.conf” file containing this:

```
deny       \.zip$     No zip files!    Zip files are blocked
allow \.doc$     -              -
```

And create a “filename.ini.rules.conf” file containing this:

```
allow -     \.ini$     -     -
```

The last part is to tell MailScanner which combination of “.rules.conf” files are needed for the different email addresses. Create /etc/MailScanner/rules/filenames.rules containing this:

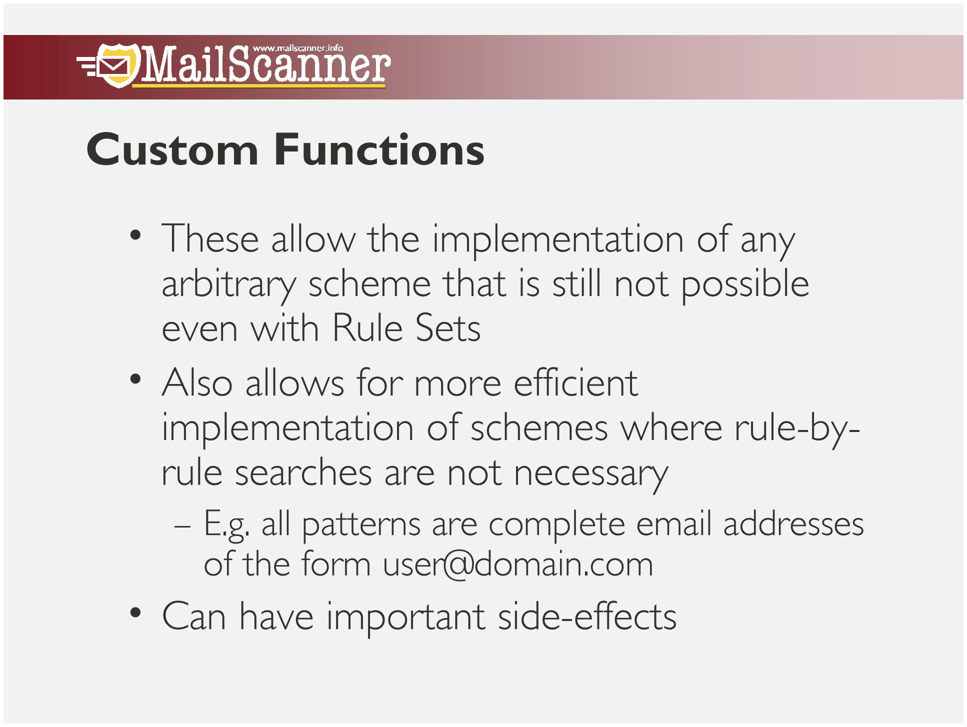
```
From: domain1.com      /etc/MailScanner/filename.zip.doc.rules.conf
      /etc/MailScanner/filename.rules.conf
From: domain2.com      /etc/MailScanner/filename.ini.rules.conf
      /etc/MailScanner/filename.rules.conf
FromOrTo:  default    /etc/MailScanner/filename.rules.conf
```

Unfortunately, the 3 lines above are too long to easily fit on one line of this book. The first and second lines start with “From:”, and the third line starts with “FromOrTo:”.

The result of this ruleset is that mail from domain1.com will have its attachments checked according to the allow/deny tests in filename.zip.doc.rules.conf followed by the allow/deny tests in filename.rules.conf. Mail from domain2.com will have its attachments checked according to the allow/deny tests in filename.ini.rules.conf followed by the allow/deny tests in filename.rules.conf. All other mail is just checked against filename.rules.conf on its own.

7.2

Custom Functions



Custom Functions

- These allow the implementation of any arbitrary scheme that is still not possible even with Rule Sets
- Also allows for more efficient implementation of schemes where rule-by-rule searches are not necessary
  - E.g. all patterns are complete email addresses of the form user@domain.com
- Can have important side-effects

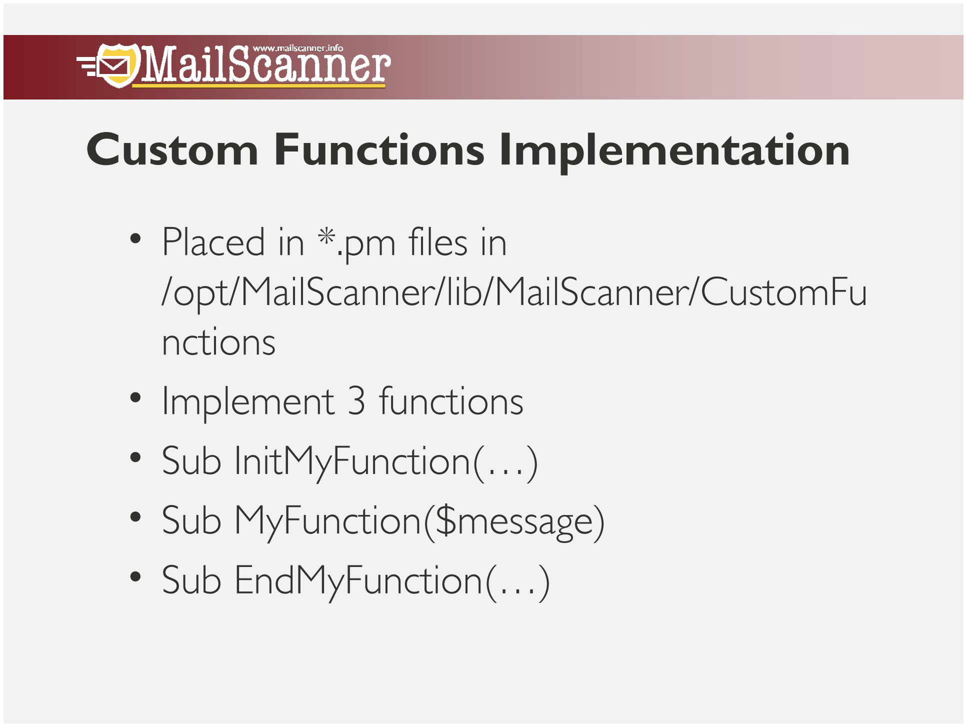
If you have a million possible email addresses to match, and you know that they are all complete email addresses, then a directory service lookup will be considerably faster than using a ruleset with a million entries. It will also be a lot smaller. They return a value in the same way as a ruleset, with the exception that "Yes" and "No" should be returned as 1 and 0 respectively.

They can use variables declared outside of the function itself, to preserve state while MailScanner executes, but please prefix variable names with some indication of your code so as to guarantee their uniqueness.

These are either placed in CustomConfig.pm in the mail MailScanner code directory, or else in the Custom Functions directory which is usually a subdirectory of it.

```
Custom Functions Dir =
      /usr/lib/MailScanner/MailScanner/CustomFunctions
```

Where to put the code for your "Custom Functions". No code in this directory should be over-written by the installation or upgrade process. All files starting with "." or ending with ".rpmnew" will be ignored, all other files will be compiled and may be used with Custom Functions.



Custom Functions Implementation

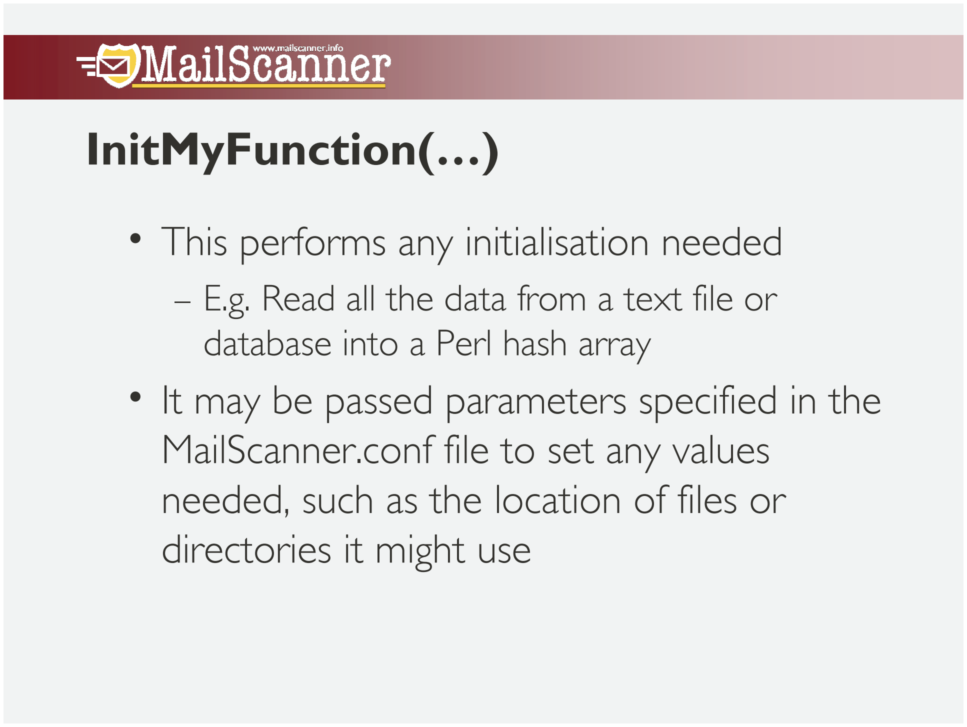
- Placed in *.pm files in /opt/MailScanner/lib/MailScanner/CustomFunctions
- Implement 3 functions
- Sub InitMyFunction(…)
- Sub MyFunction($message)
- Sub EndMyFunction(…)

Note that several processes will be running independent instantiations of your code, so great care must be taken with file locking if you are writing to or updating the content of any file. This locking is entirely your own responsibility.

The “Init” function will be called by each MailScanner child process as it starts up. It will be passed any parameters specified in the call to the Custom Function in MailScanner.conf. This removes the need to modify the code of the Custom Function just to specify simple settings just as directory names used by the Custom Function itself.

The “End” function will be called when the MailScanner child process dies. This includes the child dieing of old age as well as MailScanner being killed with the “kill” command, but not if it is killed impolitely with “kill -9” as this event cannot be trapped. This will also be passed any parameters specified in the call to the Custom Function in MailScanner.conf.

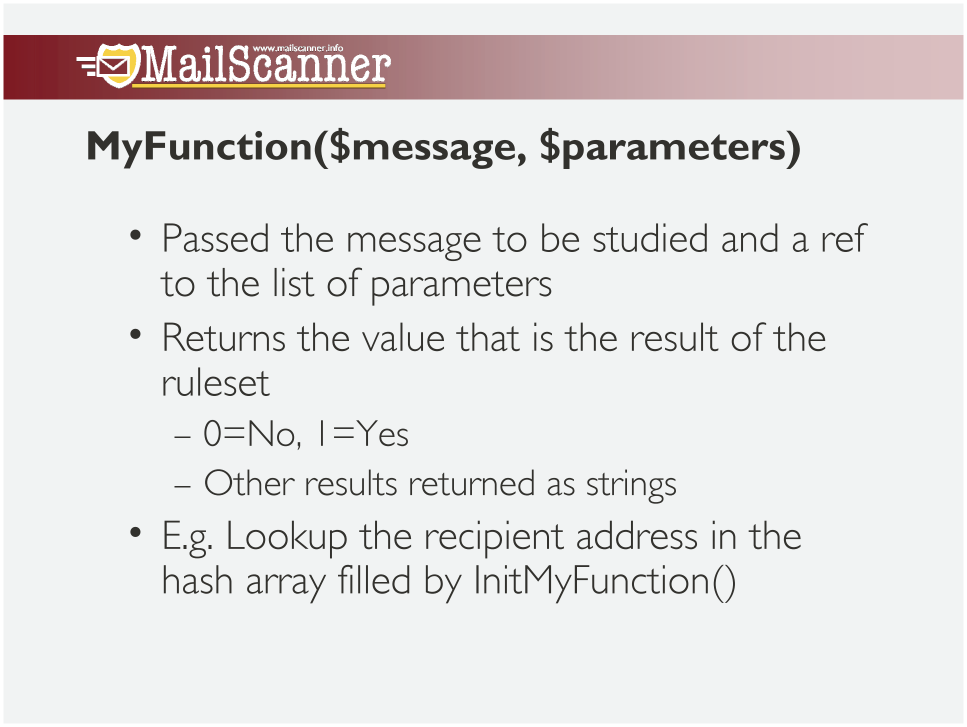
The main code is passed an object representing the message, followed by a list of any parameters specified in the call to the Custom Function in MailScanner.conf. Any of the message properties can be accessed, a list of which is at the top of Message.pm. This is the function that actually calculates the result of the configuration option to which it has been assigned.



InitMyFunction(...)

- This performs any initialisation needed
  - E.g. Read all the data from a text file or database into a Perl hash array
- It may be passed parameters specified in the MailScanner.conf file to set any values needed, such as the location of files or directories it might use

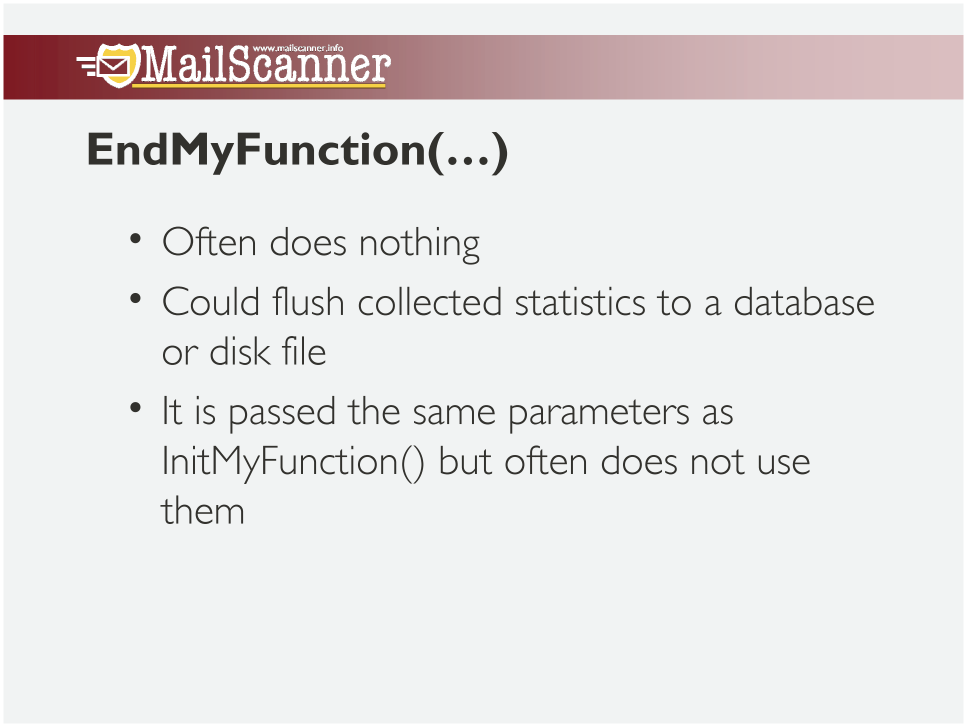
Be careful with file locking if you are opening any files for writing. One copy of the Init function will be called inside each child process, and they can be made to communicate via file contents or named pipes if necessary.



MyFunction($message, $parameters)

- Passed the message to be studied and a ref to the list of parameters
- Returns the value that is the result of the ruleset
  - 0=No, 1=Yes
  - Other results returned as strings
- E.g. Lookup the recipient address in the hash array filled by InitMyFunction()

It is quite common for this function to use a database handle or directory service, whose connection was created in the Init function.

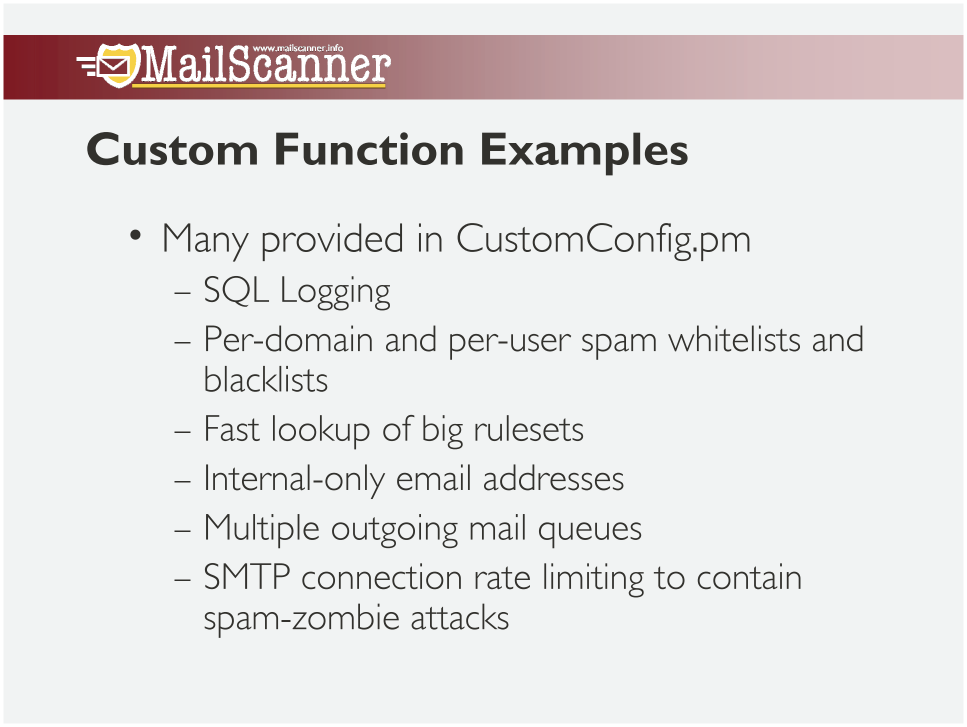


EndMyFunction(...)

- Often does nothing
- Could flush collected statistics to a database or disk file
- It is passed the same parameters as InitMyFunction() but often does not use them

This is called whenever the child process dies. This can either be naturally through old age, or because MailScanner is stopped or HUP-ed for some reason.

Note: "kill -9" cannot be trapped by MailScanner, so stopping MailScanner in this way will not allow the "End" functions to be called, or any other cleanup to be done. Please do not kill MailScanner this way.



Custom Function Examples

- Many provided in CustomConfig.pm
  - SQL Logging
  - Per-domain and per-user spam whitelists and blacklists
  - Fast lookup of big rulesets
  - Internal-only email addresses
  - Multiple outgoing mail queues
  - SMTP connection rate limiting to contain spam-zombie attacks

The large ruleset example is a very good simple demonstration of how the Init, Custom Function and End functions interact.

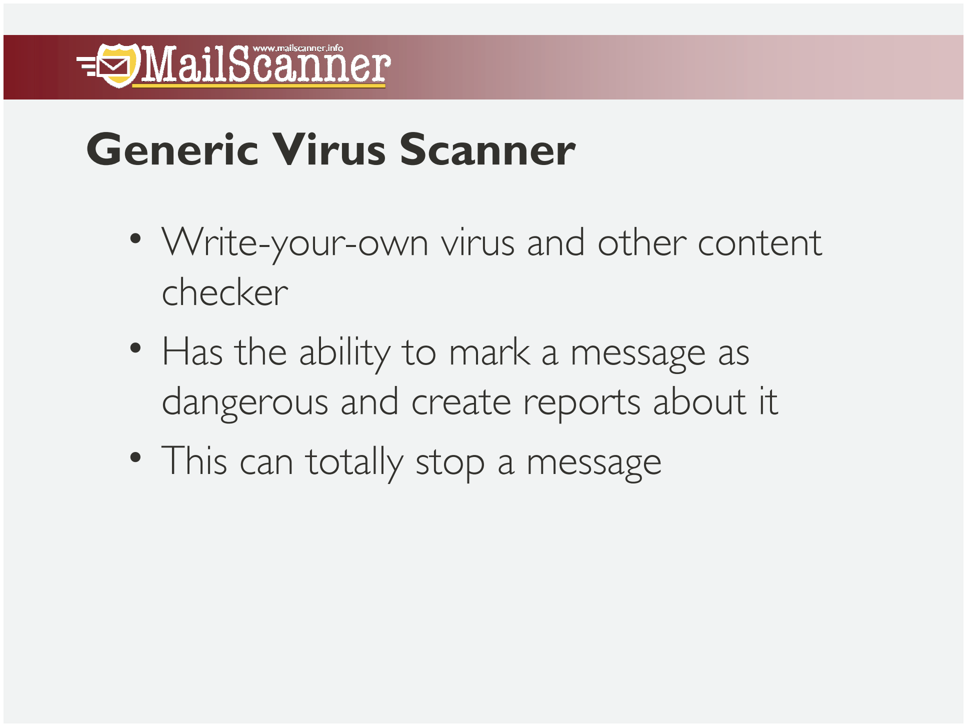
The SMTP connection rate limiting is used by several sites. It is implemented as a Custom Function as it is only currently supported on sendmail. Expanding it to other MTA's should be a simple process. It enables use of a configuration file that declares IP addresses and IP networks, together with the per-host-per-hour limit on the number of messages that will be accepted from a host on that network before its SMTP access is blocked at the MTA level. The block is left in place for up to an hour, after which an hourly cron job resets all the blocks. It is primarily used to defeat spammers and mass-mailing worms among your users and their computers.

It interacts with the MTA by use of the sendmail "access" database, and is therefore a good example of how to maintain contention and synchronisation problems between multiple child processes all accessing and modifying a single resource.

7.3

Generic Virus Scanner

Generic Spam Scanner



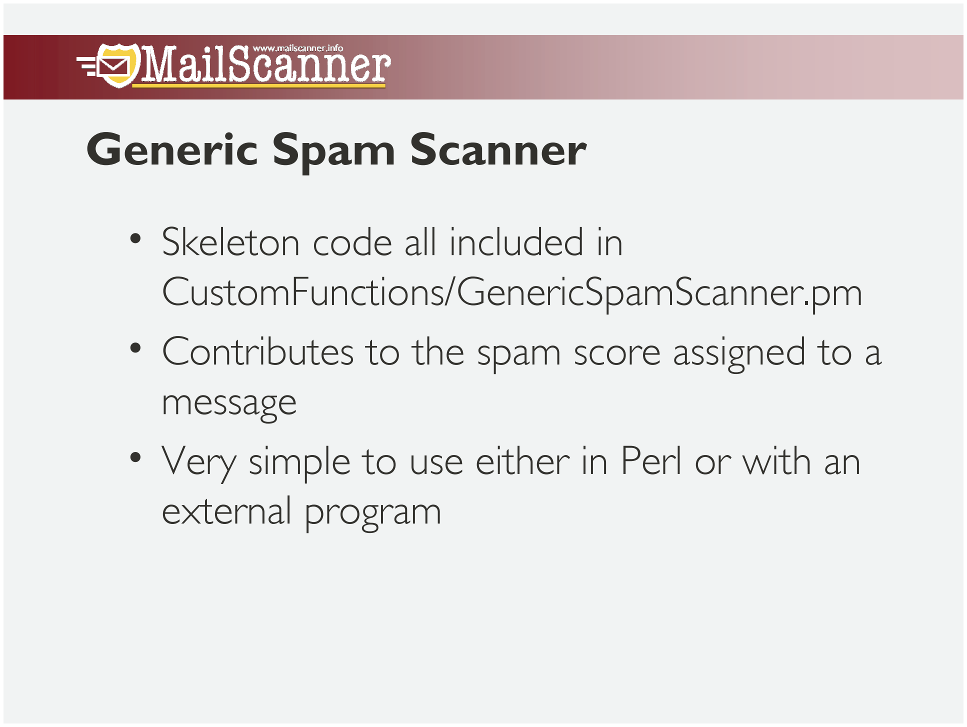
Generic Virus Scanner

- Write-your-own virus and other content checker
- Has the ability to mark a message as dangerous and create reports about it
- This can totally stop a message

There are situations in which you will want to user a content checker you have written yourself, or you want to use some other content checker that is not directly supported by MailScanner.

The “Generic Virus Scanner” provides a means whereby you can do your own content checks. There is a “generic-wrapper” script provided for you to start and run your content checker. You will have to write your own parser, which should go into MailScanner/ SweepViruses.pm, to process all the output from your checker.

If your checker needs any form of regular updates, such as new virus signatures, then you should customise the “generic-autoupdate” script as needed.



Generic Spam Scanner

- Skeleton code all included in CustomFunctions/GenericSpamScanner.pm
- Contributes to the spam score assigned to a message
- Very simple to use either in Perl or with an external program

The “Generic Spam Scanner” gives you the ability to easily implement support for unsupported spam scanners such as CRM114 and dspam. All the code you need to get started is in MailScanner/CustomFunctions/GenericSpamScanner.pm.

You can either implement it using a Perl function, or as an external program which is run for each message. There is skeleton code in the file already where you can simply slot in your Perl function, and skeleton code for running an external program.

You do not need to check that your function or program does not take too long or otherwise fails. Full timeout support is already provided by MailScanner itself.

If you choose to do it as an external program, the suggested specification of your program is that it will accept a few lines of envelope information followed by the message. Once it has been processed, it should print the message’s score and a 1-line report which will be included in the MailScanner headers:

Input:

1. SMTP connection IP address
2. Envelope sender
3. List of envelope recipients, one per line
4. 1 blank line
5. The whole message, including all the headers and the body in rfc822 format

Output:

1. Floating point or integer giving the spam score of the message
2. 1-line report giving a brief summary of the results of the program

`Use Custom Spam Scanner = no`

If this is set to yes, then the custom (or "generic") spam scanner will be applied to the message. Base your code on the GenericSpamScanner.pm file in the CustomFunctions directory. A timeout wrapper is already included, so you do not need to worry about your program or function not terminating properly.
This can also be the filename of a ruleset.

`Max Custom Spam Scanner Size = 20000`

This is the maximum amount of the message that is passed to the custom spam scanner. The message passed to the scanner will be truncated at this point. This can greatly speed up the scanning of a message, and most spam detection techniques do not require the whole body of a very long message.
This can also be the filename of a ruleset.

`Custom Spam Scanner Timeout = 20`

This is the maximum time in seconds allowed for your spam scanner to run. If it does not complete within this time, it will be automatically killed and will effectively generate a score of 0.
This cannot be the filename of a ruleset.

`Max Custom Spam Scanner Timeouts = 10`

If your spam scanner fails to complete more than this number of times in any sequence of messages, it is considered to have failed permanently. It will not be tried again until the next periodic restart of MailScanner. See the "Restart Every" setting for more details about this.
This cannot be the filename of a ruleset.

`Custom Spam Scanner Timeout History = 20`

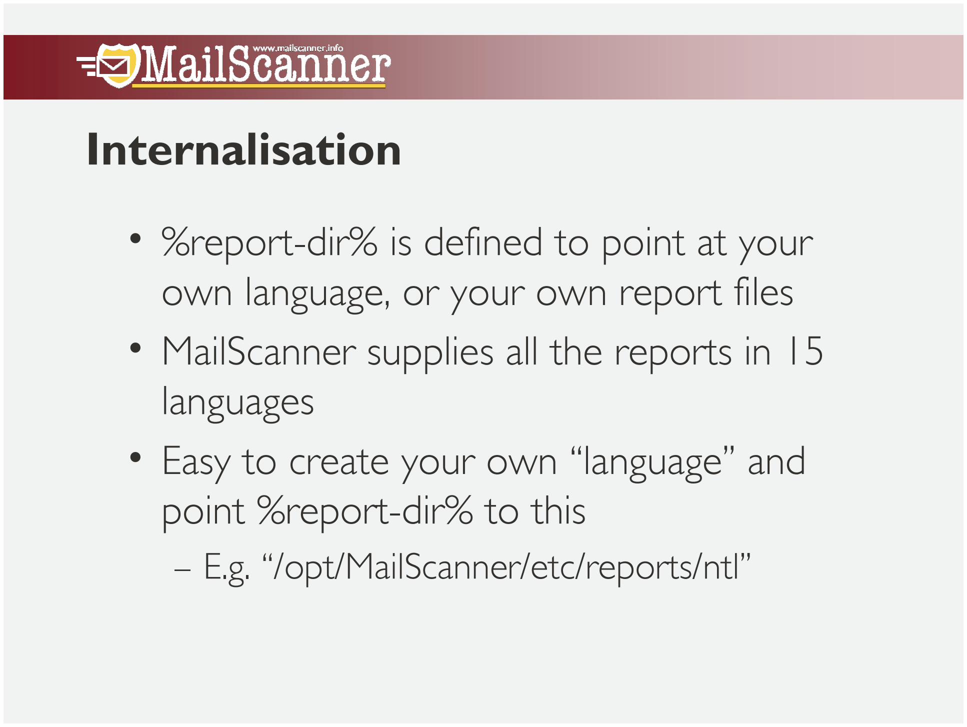
This is the length of the history of timeout occurrences. More than “Max Custom Spam Scanner Timeouts” in any sequence of messages of this length will cause the spam scanner to be considered to have failed permanently. It will not be tried again until the next periodic restart of MailScanner.

A good example is provided by the default values. 10 failures in any sequence of 20 messages will cause it to be considered dead. So over a sample of 20 messages, a 50% failure rate will stop it being used.

This cannot be the filename of a ruleset.

7.4

Internationalisation & Reports



Internalisation

- %report-dir% is defined to point at your own language, or your own report files
- MailScanner supplies all the reports in 15 languages
- Easy to create your own "language" and point %report-dir% to this
  - E.g. "/opt/MailScanner/etc/reports/ntl"

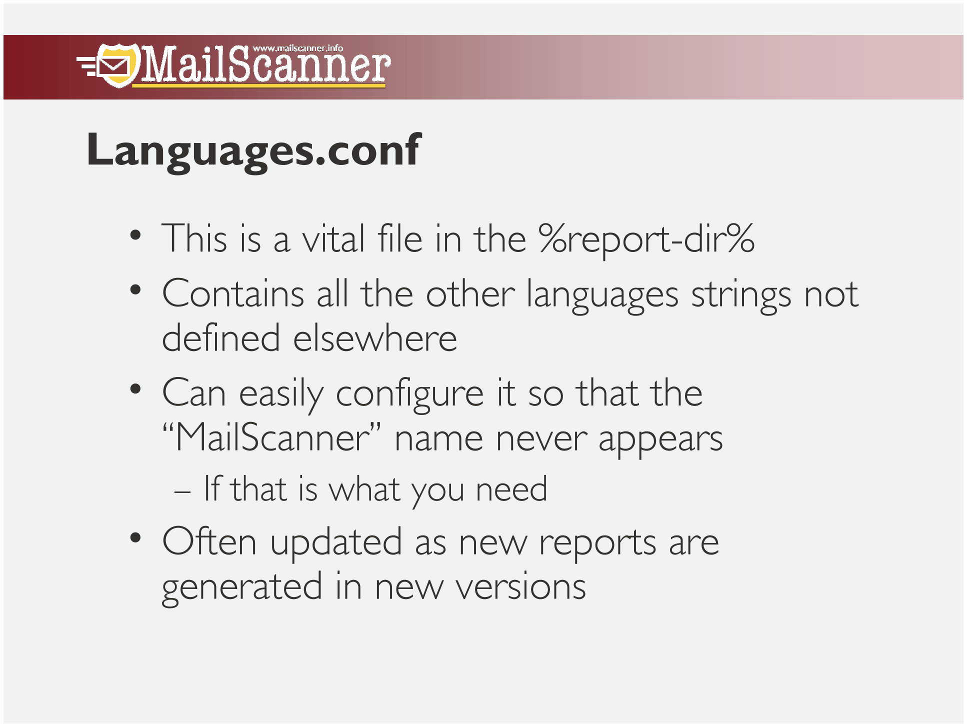
Sites wishing to totally re-brand MailScanner are encouraged to make a copy of an existing language directory (en=English) and use that as a starting point for their own re-branding. This is better than merely modifying the existing report files as it will make upgrading easier as there will not be any confusion between modified and unmodified files. Just remember to save a copy of them all before upgrading, as a safeguard.

The languages supported are currently:

Brazilian Portuguese
Catalan
Czech
Danish
Dutch
English
French
German
Hungarian
Italian
Romanian

Slovak
Spanish
Swedish
Welsh

Translations into any other languages would be most welcome, particularly languages from the Far East.



Languages.conf

- This is a vital file in the %report-dir%
- Contains all the other languages strings not defined elsewhere
- Can easily configure it so that the "MailScanner" name never appears
  - If that is what you need
- Often updated as new reports are generated in new versions

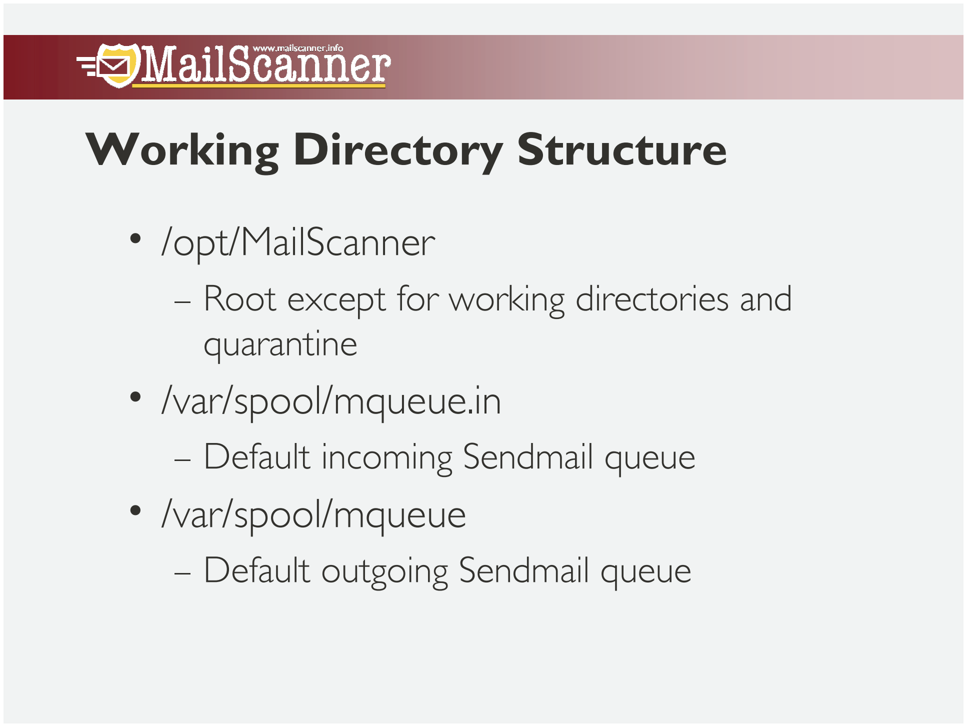
If your languages.conf file becomes out of date, and MailScanner attempts to look up a string which is not present in the file, 2 things happen:

1. A warning message is logged
2. The value returned is the internal (left-hand side) value of the string.
   This will hopefully be intelligible in English but will be all lower case.

When upgrading it is worth checking that your languages.conf file is not missing any new strings, or your reports to users may be untidy and/or give away MailScanner's real name. New items are added to the end of the file to make this check as easy as possible. There is a script upgrade_languages_conf provided which works in exactly the same way as upgrade_MailScanner_conf. Just run the script and it will print instructions describing how to use it.

7.5

Directory Structure



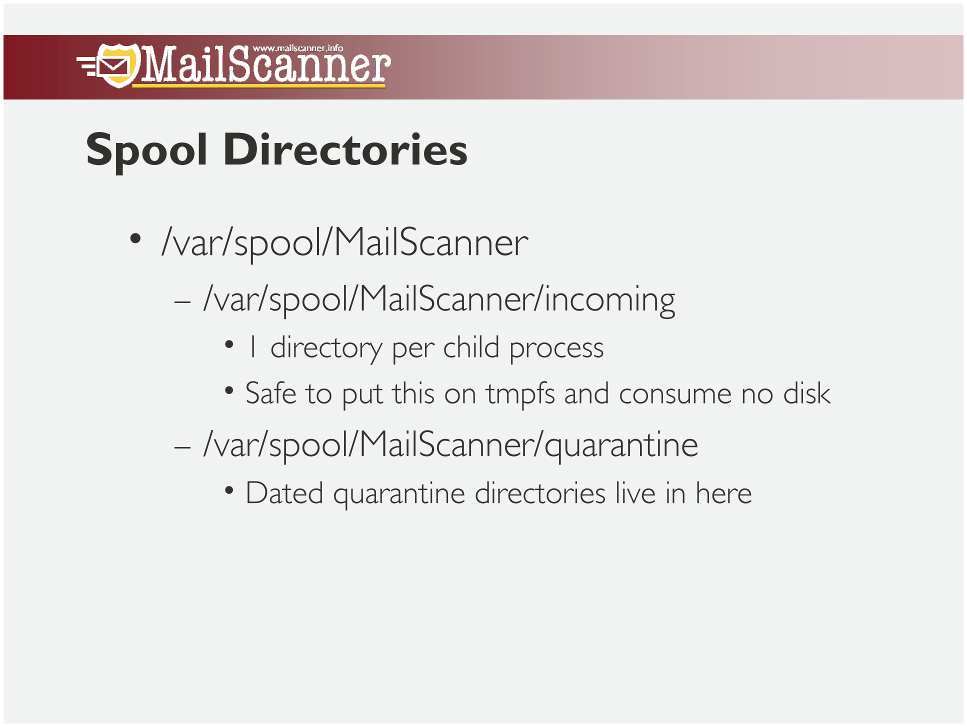
Working Directory Structure

- /opt/MailScanner
  - Root except for working directories and quarantine
- /var/spool/mqueue.in
  - Default incoming Sendmail queue
- /var/spool/mqueue
  - Default outgoing Sendmail queue

On non-RPM systems, MailScanner can be easily unpacked directly into /opt/MailScanner-version.minor by unpacking the tar archive file from within /opt. Then just make a soft-link from /opt/MailScanner-version.minor to /opt/MailScanner. Doing this results in you never having to totally remove any previous versions of MailScanner, so backing off to a previous version is a trivial task of moving 1 link.

For performance reasons, the incoming and outgoing queue directories must be contained in the same partition or logical volume. The result of this restriction is that large message bodies can be easily moved from one queue to the other by just hard-linking the file into the new directory. This completely avoids doing any direct file I/O. The only modifications are to the incoming and outgoing directories.

When using MTA's other than MailScanner, ensure that the "Run As User" can write to the incoming and outgoing queue directories.

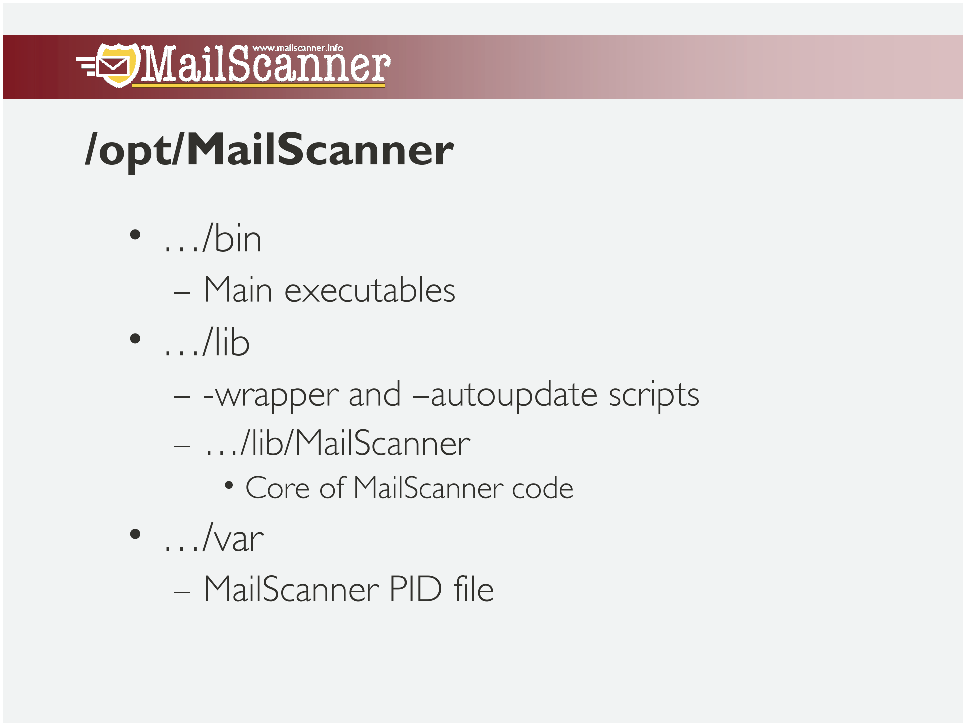


Spool Directories

- /var/spool/MailScanner
  - /var/spool/MailScanner/incoming
    - 1 directory per child process
    - Safe to put this on tmpfs and consume no disk
  - /var/spool/MailScanner/quarantine
    - Dated quarantine directories live in here

It is completely safe to put the “incoming” directory (not the incoming queue) on a RAM-based filesystem. Tmpfs is preferred over a simple ram-disk as its size expands and contracts as necessary, resulting in much better use of available RAM. If a ram-disk must be used, or the directory has to be placed on disk, then enable “soft-updates” on the filesystem if it is supported by your operating system.

The quarantine is used for long-term storage of messages and attachments that were not delivered. It should therefore be on a large hard disk partition, and regular house-keeping should be done to stop it growing too large. The RPM-based distributions include a “clean_quarantine” cron job which, if enabled, will trim the quarantine to files that have been quarantined in the past month. This is a very simple process and it is easy to customise.



/opt/MailScanner

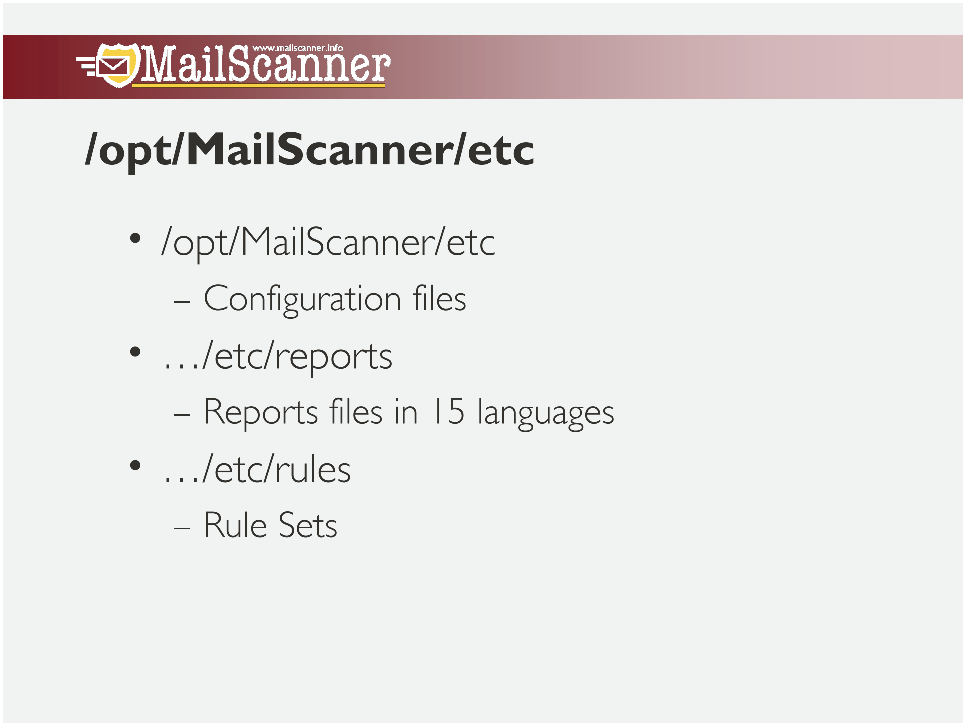
- …/bin
  - Main executables
- …/lib
  - -wrapper and –autoupdate scripts
  - …/lib/MailScanner
    - Core of MailScanner code
- …/var
  - MailScanner PID file

The directory containing the main executables also contains the entire distribution of the "tnef" program, which is used as the default TNEF expander.

To start MailScanner, use the "check_mailscanner" command instead of attempting to run the "MailScanner" program directly.

The –wrapper scripts are the interface between MailScanner and the virus-scanning engines. If you wish to test one by hand, you must put the scanner installation directory as the first command-line parameter, followed by any command-line options and the name of the directory to be scanned. The scanner installation directories can be found in the "virus.scanners.conf" file in the main MailScanner configuration directory.

When installing a virus scanner, MailScanner assumes it is installed in its default directory unless the "virus.scanners.conf" file is changed to contain the new installation directory. The format of this file is fairly obvious.



/opt/MailScanner/etc

- /opt/MailScanner/etc
  - Configuration files
- …/etc/reports
  - Reports files in 15 languages
- …/etc/rules
  - Rule Sets

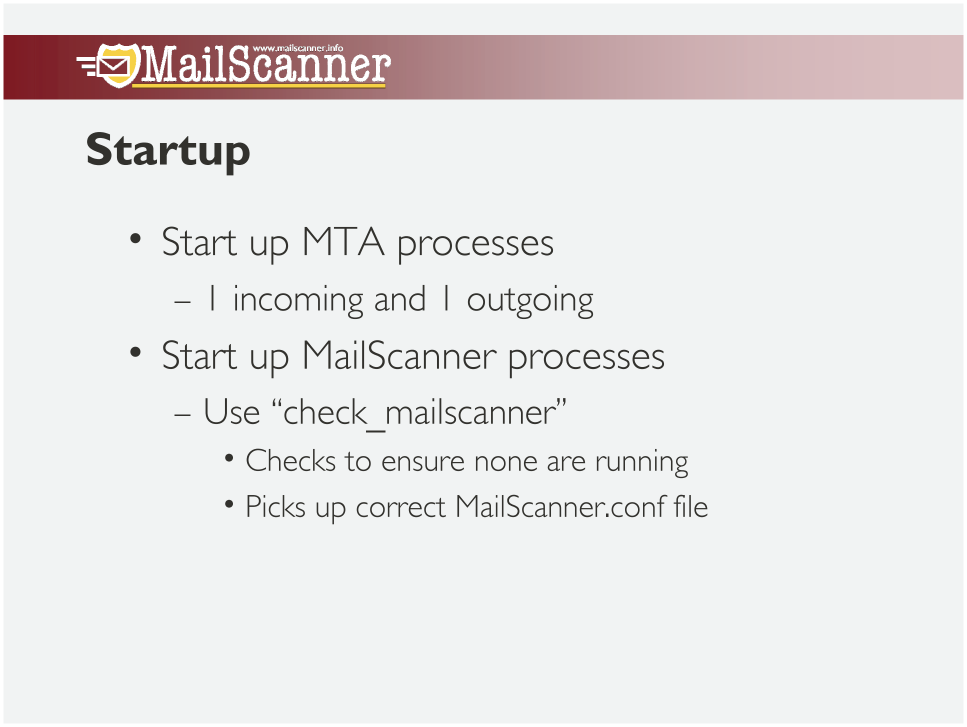
These are all merely the default locations, you can edit “MailScanner.conf” to move them. To move the main directory containing “MailScanner.conf” itself, a very simple change to “check_MailScanner” is required. The alternative is to simply create a link from /opt/MailScanner to the directory under which you have installed MailScanner.

7.6 Startup and Shutdown

Starting and Stopping

Starting the whole of MailScanner can involve up to 3 MTA processes, in addition to MailScanner itself. If using a non-RPM distribution, please check the documentation on the website for the gory details on starting up all the MTA processes.

MailScanner will shut down cleanly unless it is killed with “kill -9”. This shut down process may take up to 20 seconds. If you use the “kill” command to stop MailScanner for some reason, then wait 20 seconds before using “ps” to ensure it has shut down completely. The shut down may involve flushing large database tables if SQL servers are involved in Custom Functions.



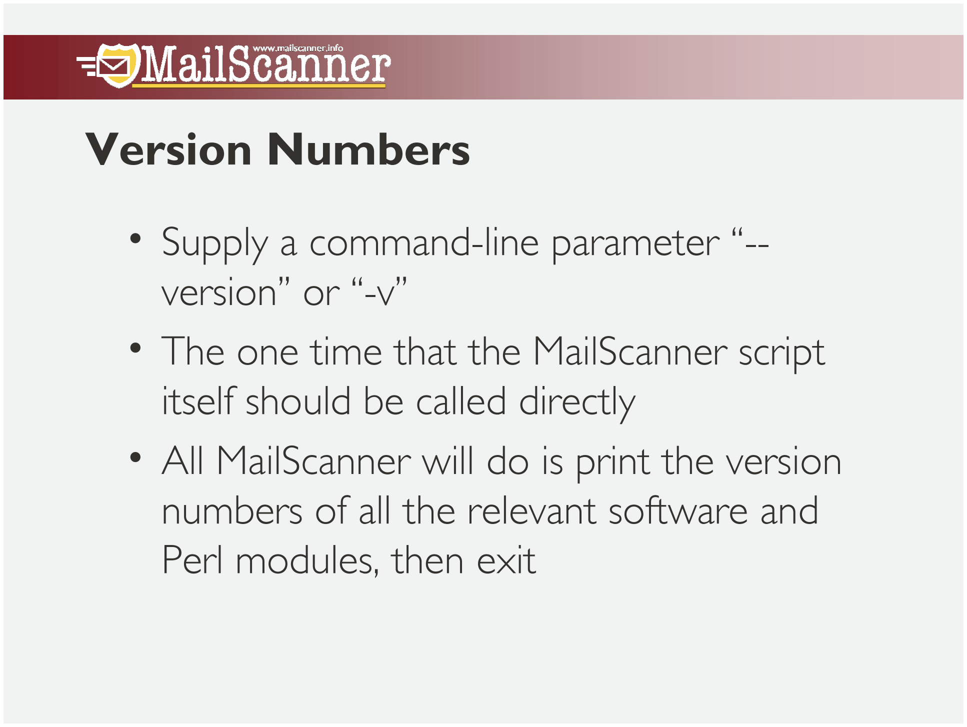
Startup

- Start up MTA processes
  - 1 incoming and 1 outgoing
- Start up MailScanner processes
  - Use “check_mailscanner”
    - Checks to ensure none are running
    - Picks up correct MailScanner.conf file

In the case of sendmail, there is an additional MTA process to handle the “clientmqueue” queue which is used for messages submitted by running the “sendmail” command directly.

A useful feature of “check_mailscanner” is that when run, it will print all the process ids (PIDs) of the MailScanner processes that are running. If MailScanner is not running, it will start it. Note that it does not start the MTA processes, they must be started separately.

If using the RPM distributions, the init.d script for MailScanner starts up all the required MTA processes, and then runs “check_mailscanner”. To select the MTA that is used, it should be selected in /etc/sysconfig/MailScanner. This is also where to put customisations to the init.d script such as MTA queue runner frequencies and other MTA-specific settings.



Version Numbers

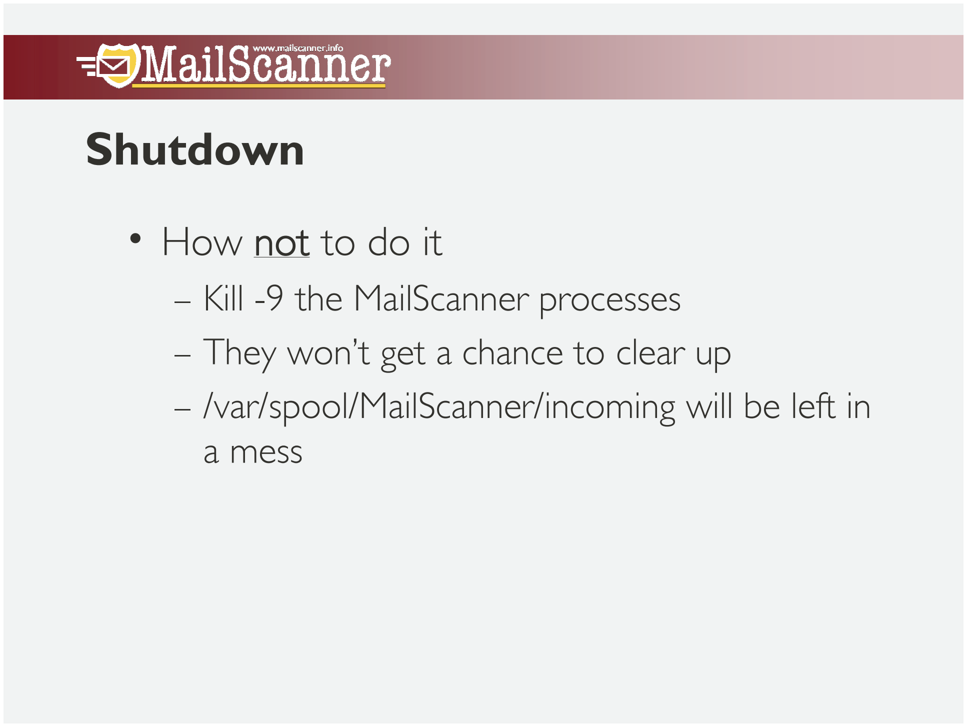
- Supply a command-line parameter "--version" or "-v"
- The one time that the MailScanner script itself should be called directly
- All MailScanner will do is print the version numbers of all the relevant software and Perl modules, then exit

Any of these may be specified, they all have the same effect:

- -v in upper and lower case
- --version in any combination of upper and lower case

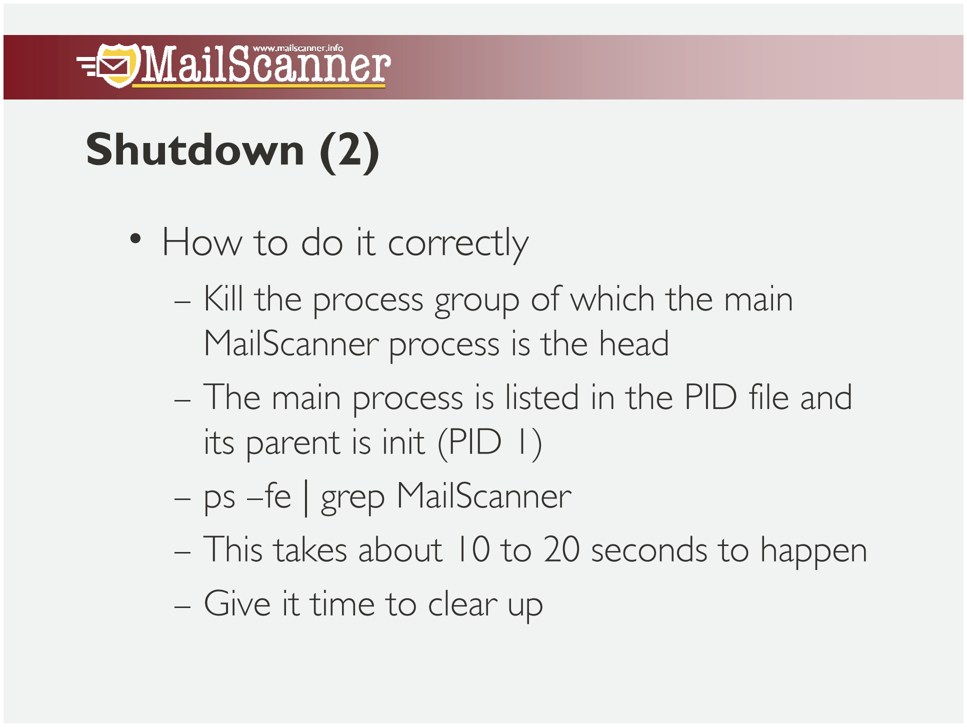
It will print the version numbers of

- Perl
- MailScanner
- All required Perl modules
- Optional items such as SpamAssassin and the ClamAV Perl module



Shutdown

- How not to do it
  - Kill -9 the MailScanner processes
  - They won't get a chance to clear up
  - /var/spool/MailScanner/incoming will be left in a mess



Shutdown (2)

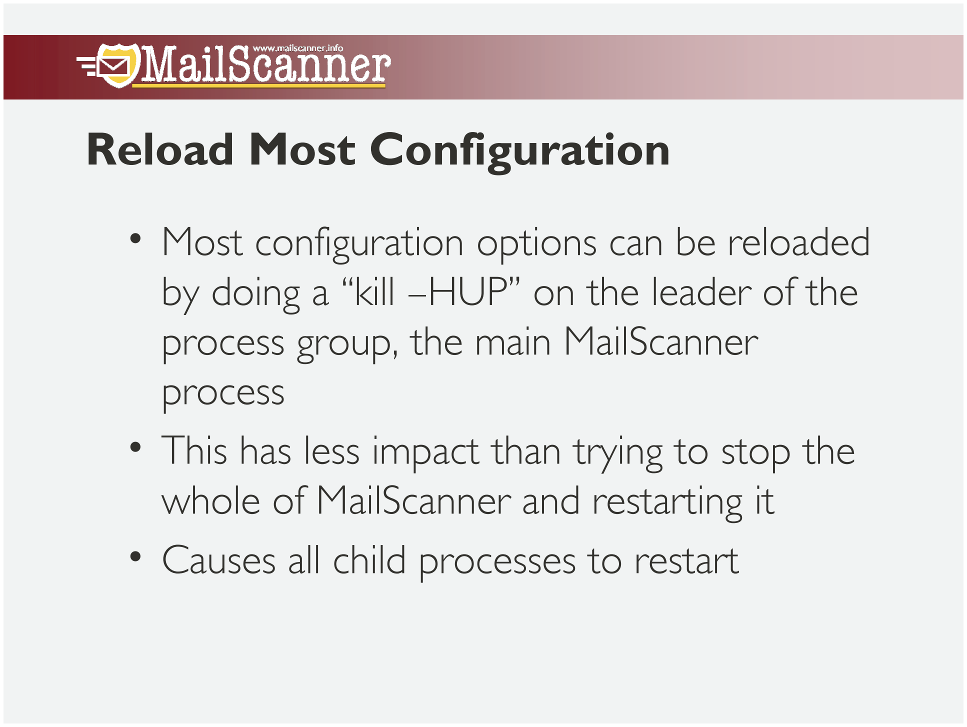
- How to do it correctly
  - Kill the process group of which the main MailScanner process is the head
  - The main process is listed in the PID file and its parent is init (PID 1)
  - ps –fe | grep MailScanner
  - This takes about 10 to 20 seconds to happen
  - Give it time to clear up

The “kill” command can be used:

- “Kill –HUP” sent to the MailScanner parent process, or to all MailScanner processes, will force the child processes to exit and clean up. They will then be re-spawned by the parent, at which point they will re-read their configuration.

- “kill” or “kill –TERM” sent to the MailScanner parent process will tell MailScanner to shut down cleanly. This is the action taken by the init.d script when supplied with the “stop” or “stopms” parameter.

The process ID or PID of the main MailScanner parent process can be found in the pid file as defined in MailScanner.conf.

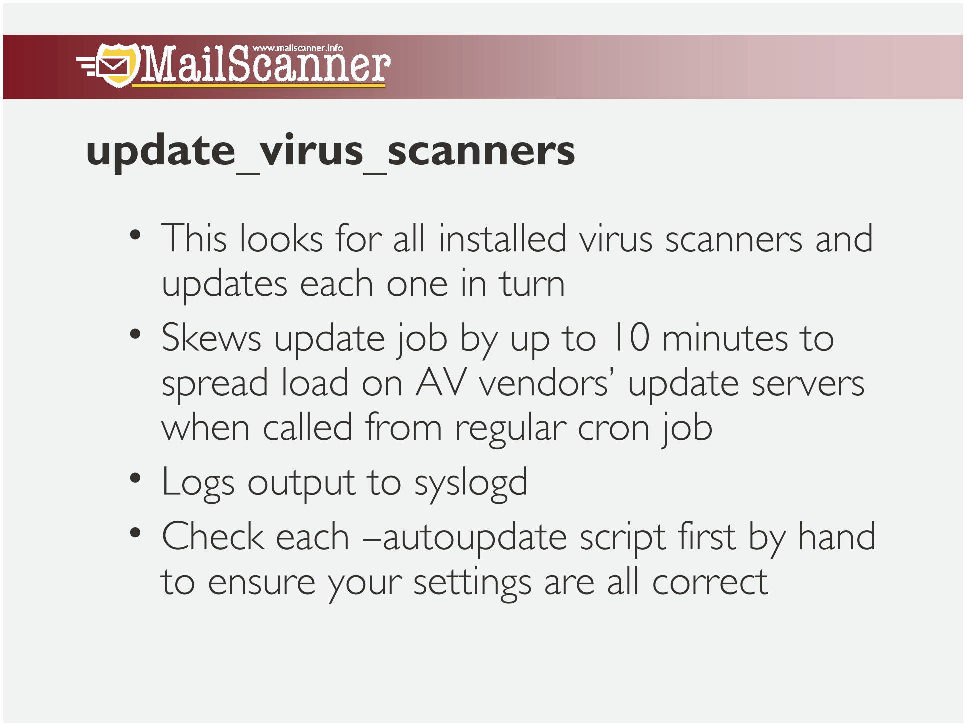


Reload Most Configuration

- Most configuration options can be reloaded by doing a “kill –HUP” on the leader of the process group, the main MailScanner process
- This has less impact than trying to stop the whole of MailScanner and restarting it
- Causes all child processes to restart

Unfortunately some of the options have to be read very early in the execution of MailScanner, and so this cannot work for every configuration option.

It is better than totally stopping and restarting MailScanner as the 20 second delay is not needed. The incoming queue is therefore left for much less time without be processed, which can be important on very heavily-loaded servers.



www.mailscanner.info

MailScanner

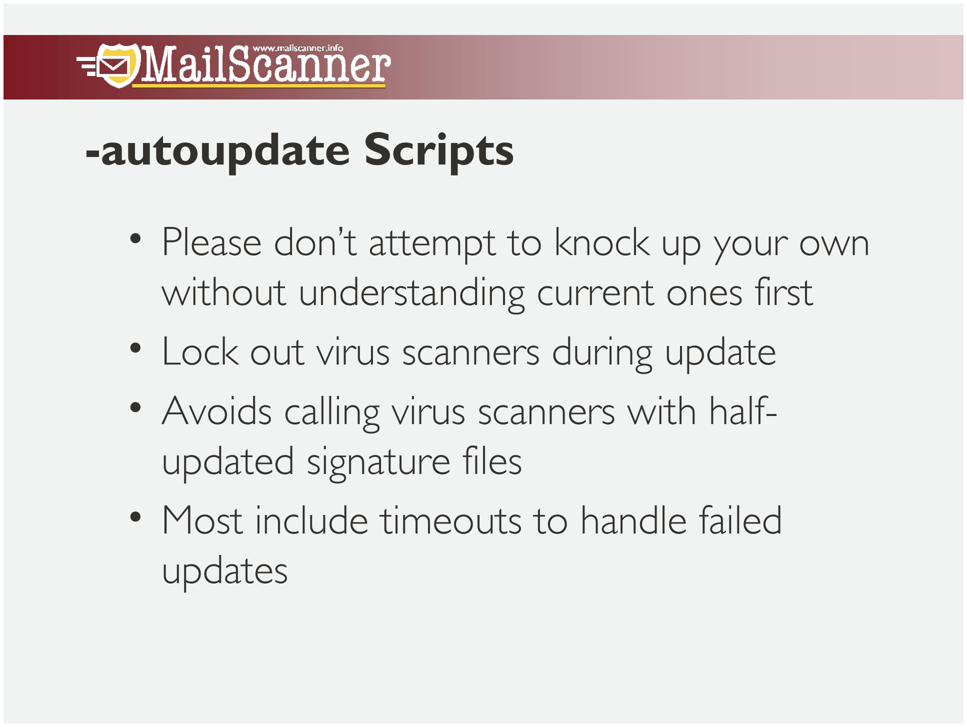
update_virus_scanners

- This looks for all installed virus scanners and updates each one in turn
- Skews update job by up to 10 minutes to spread load on AV vendors' update servers when called from regular cron job
- Logs output to syslogd
- Check each –autoupdate script first by hand to ensure your settings are all correct

Each virus scanner is located via its entry in “virus.scanners.conf”. The –wrapper script for the scanner is passed the “-IsItInstalled” command-line parameter to which it responds with an appropriate exit code. If found, the corresponding –autoupdate script is called.

After installing a new scanner, a test that should be done is to call the –autoupdate script for it to ensure that new virus signatures will be automatically downloaded and installed.

You should also check the path provided to your cron jobs, to ensure that everything necessary can be found which you have installed in non-standard locations such as “/usr/local/bin”. This path may be set in “/etc/default/cron” on some operating systems.

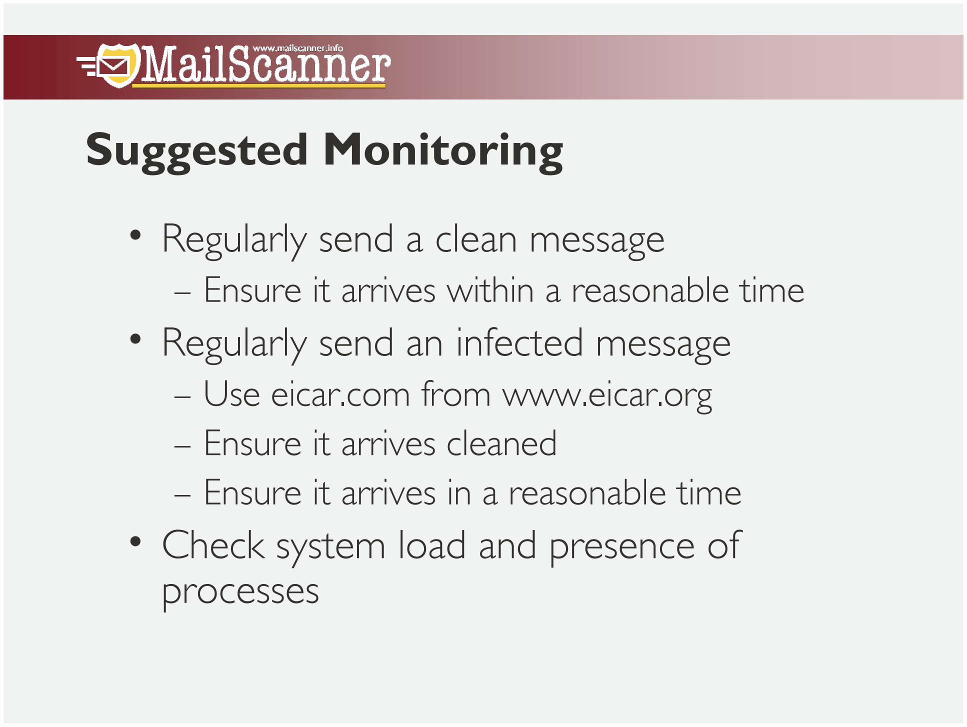


-autoupdate Scripts

- Please don't attempt to knock up your own without understanding current ones first
- Lock out virus scanners during update
- Avoids calling virus scanners with half-updated signature files
- Most include timeouts to handle failed updates

It is vital that the virus scanning engines are locked out while the update is in progress, to avoid calling a scanner with no signature files.

Ideally all –autoupdate scripts should implement a timeout limit on the update, and download all the new files to a separate directory, from where they are moved into the active signature directory in an atomic operation.



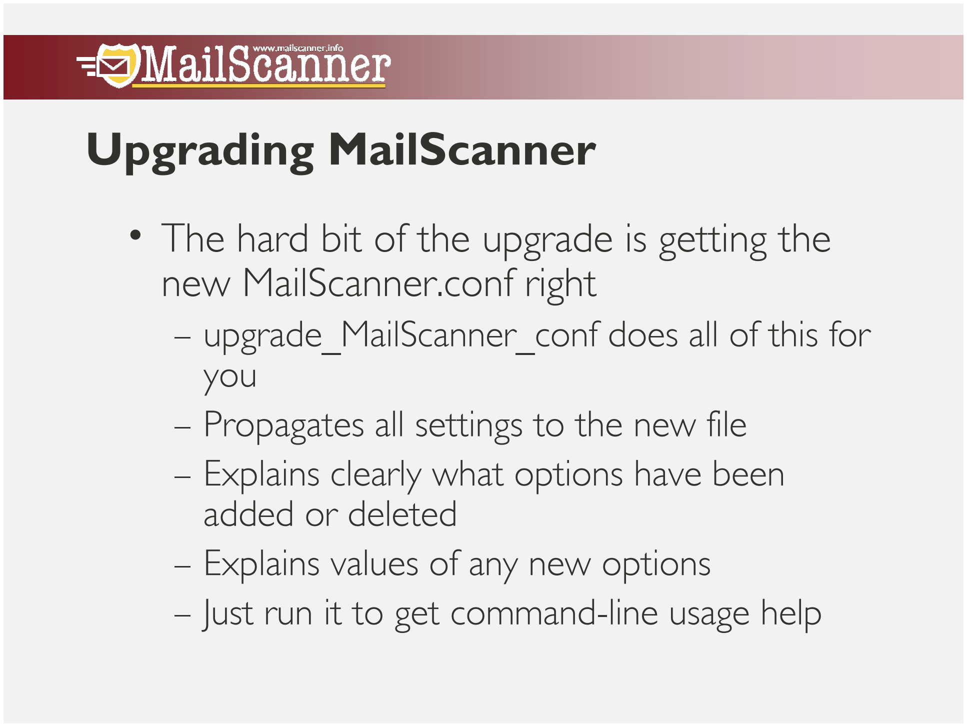
Suggested Monitoring

- Regularly send a clean message
  - Ensure it arrives within a reasonable time
- Regularly send an infected message
  - Use eicar.com from www.eicar.org
  - Ensure it arrives cleaned
  - Ensure it arrives in a reasonable time
- Check system load and presence of processes

These checks are not currently implemented in the MailScanner distribution, except for a call to "check_MailScanner" from a cron job to ensure MailScanner has not completely died. Due to the differences between mail routing on different installations, it is very difficult to implement this in a general-purpose way.

Various system monitoring tools and packages such as nagios, nocol and bigbrother exist which can be used as a basis for writing such checks.

Busy MailScanner systems tend to have high system load averages as this figure is not only an indication of CPU load, but also network traffic and disk I/O. MailScanner works the system very hard, and double-digit load averages are normal.



Upgrading MailScanner

- The hard bit of the upgrade is getting the new MailScanner.conf right
  - upgrade_MailScanner_conf does all of this for you
  - Propagates all settings to the new file
  - Explains clearly what options have been added or deleted
  - Explains values of any new options
  - Just run it to get command-line usage help

If you run "upgrade_MailScanner_conf" on its own, it will do nothing apart from explain how to use it.

It takes 2 main command-line parameters:

1. the path of your current MailScanner.conf file
2. the path of the new MailScanner.conf file into which you wish to import your settings

The STDOUT output of the script is a new MailScanner.conf file, with all your settings installed. So a common command is

```
cd /opt/MailScanner/etc
../bin/upgrade_MailScanner_conf MailScanner.conf \
/opt/MailScanner.4.30.2/etc/MailScanner.conf > MailScanner.new
```

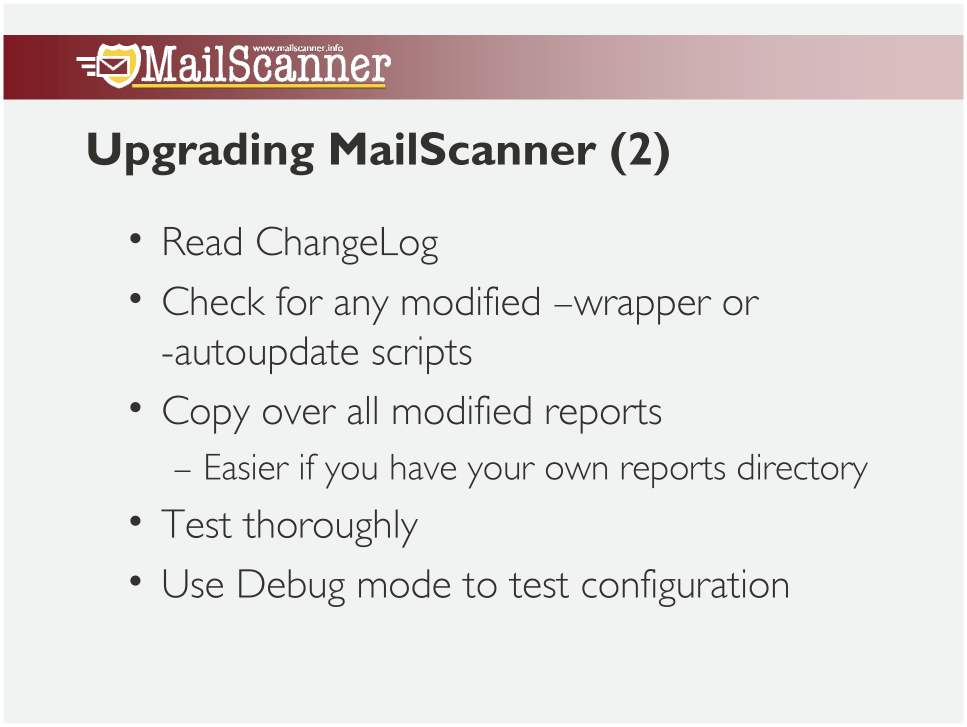
Or, for an RPM-based system

```
cd /etc/MailScanner
```

```
upgrade_MailScanner_conf MailScanner.conf MailScanner.conf.rpmnew >
MailScanner.new
```

When used in this way, it will still output (to the terminal) information about the number of options copied, along with details of new options that have been added and any old options that have been removed.

If you also wish to preserve all your comments that were added to the old file, it takes an optional "--keep-comments" command-line option before the filenames to work on. Bear in mind that you will not get the new comments, which may well indicate new features that are available in existing configuration options.



Upgrading MailScanner (2)

- Read ChangeLog
- Check for any modified –wrapper or -autoupdate scripts
- Copy over all modified reports
  - Easier if you have your own reports directory
- Test thoroughly
- Use Debug mode to test configuration

If you start up MailScanner without first checking for configuration errors, then it will occupy all the CPU time it can get, and will syslog the errors repeatedly, while processing no mail. If run in Debug mode, it will output detected errors in the configuration and then exit. The syslog should also be checked for errors, as it may give more information than the terminal output.

Do not forget to check the "languages.conf" file for new strings added to the end of it. The upgrade_languages_conf script can be used to upgrade this file in exactly the same way as the upgrade_MailScanner_conf script. Just run it, and it will print out instructions on how to use it.

8

Charity Support

www.mailscanner.info MailScanner

www.mailscanner.info MailScanner

Is Proud To Support

Please consider making a donation to this charity. The address to which you should post donations is on their website, www.ellenmacarthurtrust.org. If you with to donate by credit card or by PayPal, see the donations page on www.mailscanner.info. Just let me know that you would like your donation to go to the Trust, and I will forward it to them immediately.

They take children who have been in hospital for a long time, notably cancer and leukaemia patients, and take them sailing for a few days on the best sailing water in the country around the Solent and the South Coast of England. This gives them a fantastic opportunity to get out of the hospital for a couple of days, get the sun on their faces and the wind in their hair. This is all done on well-equipped yachts with all the necessary safety equipement and full nursing care, just as good as if they were in hospital for the weekend. This costs UKP 850 per child for 1 weekend, as of 2004.

Those of you who know me will be aware that this is a subject very close to my heart as I have spent a year in hospital myself and know very well what it is like to get out for a while, however short it must be.

The kids deserve any donation you can make, and will remember the weekend for the rest of their life; that I can guarantee.

Thankyou.

Appendix – How Electronic Mail Works For Laymen

From a posting to the MailScanner mailing list…

How Electronic Mail Works For Laymen

-Or-

How I learned to stop worrying and love acronyms.

Every time you type a message, small multi-headed trailer- park demons (or SMTP's for short) take that message and run inside the network cables (or phone lines, but they're narrower, that's why they're slower) to the destination where it's handed to the incorrigible mail altered-state postal demons (or IMAPD) to then be handed out by the positronic otter people (or POP) to the users (aka idiots.) Incidentally, this is also why telephone lines sag from pole to pole.

If MailScanner is running on your system, then there are a few more steps involved, but mostly it's just small portly aristocratic men making sure that you don't have the word "penis" in your post.

Not to be confused with the Kraft Foods Post[tm] brand cereal products. That would be gross.

P.T.O…

Still here?

Ok, here's how it really works, but keep it secret so us admin/priests still have jobs:

[note to the pedants: yes, this is simplified -- it's meant to be. A lot of documentation out there seems to be needlessly complicated, especially in the context of someone just starting out. I realize it's fun to abuse our PFY's, but they've got to know something if we're going to be able to sneak out to a pub at 10AM. Further note: I wrote this in a hurry.]

There are three parts to a mail system:

Mail User Agent (MUA).

Mail Transfer Agent (MTA).

Mail Delivery Agent (MDA).

The MUA is what the user uses to create (and usually read) mail.

It's what's usually referred to as the "client". This can be Outlook, Outlook Express, Evolution, a web browser connecting to a web-based email system (Hotmail) etc.

The MTA is the meat and potatoes (or beans and franks, if you prefer). It, surprisingly enough, transfers mail; usually between an MUA and a MDA. Some packages that do this are Sendmail, Exim, and Postfix. This list isn't exhaustive, don't get your panties in a bunch if I don't name your favorite.

The MDA takes the message from the MTA and delivers it to the user. This is some murky water right here, because, technically, all MTA's have MDA components, but usually when someone is talking about a MDA they're talking about the software at the end of the chain on the server, the part that hands the message out, like a pop3 server or an imap server. Some big packages seem to be both an MTA and an MDA (like Exchange) but internally they are separate components.

Here is (a very simplified) chain of events where we follow a message from a sender to the destination.

bob@fromhere.com is sending a message to tom@tohere.com:

1) Outlook [MUA] (Bob) (Sender)

2) Sendmail [MTA] (mail.fromhere.com)

3) Sendmail [MTA] (mail.tohere.com)

4) imapd [MDA] (mail.tohere.com)

5) Outlook [MUA] (Tom) (Recipient)

1) Bob fires up Outlook, types his message to Tom

2) outlook connects to the "outgoing" mail server in its configuration, in this case let's say it's mail.fromhere.com. mail.fromhere.com sees that the mail is to go to mail.tohere.com so it in turn connects to

3) mail.tohere.com. mail.tohere.com realizes that the mail is destined for local delivery, so it puts it in toms spool that

4) imap knows about. When Tom fires up his client and it connects to the imap server that lives on mail.tohere.com it hands the message over to

5) Toms Outlook, that he then reads:

"Hey, Tom, do you ever get that unfresh feeling?

Love,

Bob."

So, now you're probably asking yourself "why the hell am I reading this garbage?"

And I'd have to answer "because you want to know how MailScanner fits into this, punk!"

And then I'd be like "You been shown."

and then you'd be all "Oh, yeah? It's ON!"

and I'd be like "Bring it, bitch!"

and then I'm all:

To simplify things we're going to just look at the receiving end of the mail conversation I described above, so Bob's sent his message and it's now coming into mail.tohere.com:

1) Sendmail incoming queue mail.tohere.com (MTA)
2) Mailscanner on mail.tohere.com (sort-of-MTA)
3) Still Mailscanner: Checks for viruses and spam (usually using 3rd party utilities)
4) Sendmail delivery queue (mail.tohere.com) (MTA)
5) imapd (mail.tohere.com) (MDA)
6) Outlook (Tom) (MUA)

1) sendmail takes the message and immediately puts it in a queue, in this example, that's it.
2) MailScanner picks it up from the queue and
3) beats the crap out of it, if it survives it's put in another queue that
4) Sendmail picks up and delivers to the users spool that
5) imap knows about and delivers to the user when
6) he fires up Outlook.

[note: there's a really cool diagram available at http://www2.essex.ac.uk/cs/services/email/anti-spam/mailscanner.html that covers this part in more detail. If you're into that sort of thing.]

As you learn more about the mail process you'll learn that 1) most of what I've written here is incredibly simplified and 2) doing this crap day in and day out is a sure-fire way to end up in the nuthouse.

There is much more to all of this, and explaining it might (and I'm guessing here) almost fill up a book.

Or two hundred.

If you're seriously interested you need to get your grubby lil' paws on a box and install all this software on it. Play around. Get a couple of boxes and try getting them to send mail to each other.

Get yourself a dyndns account and set up a test mail server at home. Play around and learn. You could go take classes, but that costs money and the reason you're into this is because you think you might be able to get a job doing it because right now you work at a fast food joint and don't make jack and can't afford the classes.

Well, that's why I said to use the free stuff. Then you can be a highly paid and mentally stable systems administrator like me! Twitch.

--

PS: Slow day, what can I say. If you don't like it, it's not my fault you read it all the way to the end.

Index