BetterDocumentation ReadmeTxt

README

The following is the text of README, based off of revision 263821.

Please feel free to edit it as much as you like to make it more useful. Periodically the version in Subversion will be updated to incorporate some of the changes.

To see the latest version in Subversion, click here

Feel free to write comments about your changes in the Comments section (at the bottom).

```
Welcome to SpamAssassin!
------------------------

What SpamAssassin Is
--------------------

SpamAssassin is a mail filter which attempts to identify spam using
a variety of mechanisms including text analysis, Bayesian filtering,
DNS blocklists, and collaborative filtering databases.

SpamAssassin is a project of the Apache Software Foundation (ASF).


What SpamAssassin Is Not
------------------------

SpamAssassin is not a program to delete spam, route spam and ham to
separate mailboxes or folders, or send bounces when you receive spam.
Those are mail routing functions, and SpamAssassin is not a mail
router.  SpamAssassin is a mail filter or classifier.  It will examine
each message presented to it, and assign a score indicating the
likelihood that the mail is spam.  An external program must then
examine this score and do any routing the user wants done.  There are
many programs that will easily perform these functions after examining
the score assigned by SpamAssassin.


How SpamAssassin Works
----------------------

SpamAssassin uses a wide range of heuristic tests on mail headers and
body text to identify "spam", also known as unsolicited commercial
email.

Once identified, the mail can then be optionally tagged as spam for
later filtering using the user's own mail user-agent application.

SpamAssassin typically differentiates successfully between spam and
non-spam in between 95% and 100% of cases, depending on what kind of
mail you get and your training of its Bayesian filter.  Specifically,
SpamAssassin has been shown to produce around 0.9% false negatives (spam
that was missed) and around 0.1% false positives (ham incorrectly marked
as spam).  See the rules/STATISTICS*.txt files for more information.

SpamAssassin also includes plugins to support reporting spam messages
automatically or manually to collaborative filtering databases such as
Pyzor, DCC, and Vipul's Razor.

The distribution provides "spamassassin", a command line tool to
perform filtering, along with the "Mail::SpamAssassin" module set
which allows SpamAssassin to be used in spam-protection proxy SMTP or
POP/IMAP server, or a variety of different spam-blocking scenarios.

In addition, "spamd", a daemonized version of SpamAssassin which
runs persistently, is available.  Using its counterpart, "spamc",
a lightweight client written in C, an MTA can process large volumes of
mail through SpamAssassin without having to fork/exec a perl interpreter
```

for each message.

Questions? Need Help?

If you have questions about SpamAssassin, please check the Wiki[1] to
see if someone has already posted an answer to your question. (The
Wiki doubles as a FAQ.) Failing that, post a message to the
spamassassin-users mailing list[2]. If you've found a bug (and you're
sure it's a bug after checking the Wiki), please file a report in our
Bugzilla[3].

        [1]: http://wiki.apache.org/spamassassin/
        [2]: http://wiki.apache.org/spamassassin/MailingLists
        [3]: http://bugzilla.spamassassin.org/

Please also be sure to read the man pages.

Upgrading SpamAssassin

IMPORTANT: If you are upgrading from a previous major version of
SpamAssassin, please be sure to read the notes in UPGRADE to find out
what has changed in a non-backwards compatible way.

Installing SpamAssassin

See the INSTALL file.

Customising SpamAssassin

These are the configuration files installed by SpamAssassin. The commands
that can be used therein are listed in the POD documentation for the
Mail::SpamAssassin::Conf class (run the following command to read it:
"perldoc Mail::SpamAssassin::Conf"). Note: The following directories are
the standard defaults that people use. There is an explanation of all the
default locations that SpamAssassin will look at the end.

  - /usr/share/spamassassin/*.cf:

        Distributed configuration files, with all defaults. Do not modify
        these, as they are overwritten when you upgrade.

  - /etc/mail/spamassassin/*.cf:

          Site config files, for system admins to create, modify, and
        add local rules and scores to. Modifications here will be
        appended to the config loaded from the above directory.

  - /etc/mail/spamassassin/*.pre:

        Plugin control files, installed from the distribution. These are
        used to control what plugins are loaded. Modifications here will
        be loaded before any configuration loaded from the above
        directories.

        You want to modify these files if you want to load additional
        plugins, or inhibit loading a plugin that is enabled by default.
        If the files exist in /etc/mail/spamassassin, they will not
        be overwritten during future installs.

Per-User Customising

```
Per-user customisation only works on some installs.
(If it doesn't work on your install, that's the fault of the integrator, not the SpamAssassin development team,
so don't bug them - see http://wiki.apache.org/spamassassin/AskingAboutIntegrations.)

(VERY incomplete, unconfirmed list: it does not work at fastmail.fm, or when part of the ECM Mac OS X package -
delete this parenthetical before updating the dox unless massively improved.)

If it does work, here's how to tweak it:

  - /usr/share/spamassassin/user_prefs.template:

        Distributed default user preferences. Do not modify this, as it is
        overwritten when you upgrade.

  - /etc/mail/spamassassin/user_prefs.template:

        Default user preferences, for system admins to create, modify, and
        set defaults for users' preferences files.  Takes precedence over
        the above prefs file, if it exists.

        Do not put system-wide settings in here; put them in a file in the
        "/etc/mail/spamassassin" directory ending in ".cf". This file is
        just a template, which will be copied to a user's home directory
        for them to change.

  - $USER_HOME/.spamassassin:

          User state directory.  Used to hold spamassassin state, such
        as a per-user automatic whitelist, and the user's preferences
        file.

  - $USER_HOME/.spamassassin/user_prefs:

          User preferences file.  If it does not exist, one of the
        default prefs file from above will be copied here for the
        user to edit later, if they wish.

        Unless you're using spamd, there is no difference in
        interpretation between the rules file and the preferences file, so
        users can add new rules for their own use in the
        "~/.spamassassin/user_prefs" file, if they like.  (spamd disables
        this for security and increased speed.)

  - $USER_HOME/.spamassassin/bayes*

        Statistics databases used for Bayesian filtering.  If they do
        not exist, they will be created by SpamAssassin.

        Spamd users may wish to create a shared set of bayes databases;
        the "bayes_path" and "bayes_file_mode" configuration settings
        can be used to do this.

        See "perldoc sa-learn" for more documentation on how
        to train this.

File Locations:

SpamAssassin will look in a number of areas to find the default
configuration files that are used.  The "__*__" text are variables
whose value you can see by looking at the first several lines of the
"spamassassin" or "spamd" scripts.

They are set on install time and can be overridden with the Makefile.PL
command line options DATADIR (for __def_rules_dir__) and CONFDIR (for
__local_rules_dir__).  If none of these options were given, FHS-compliant
locations based on the PREFIX (which becomes __prefix__) are chosen.
These are:

  __prefix__    __def_rules_dir__              __local_rules_dir__
  ------------------------------------------------------------------------
  /usr          /usr/share/spamassassin        /etc/mail/spamassassin
```

```
  /usr/local    /usr/local/share/spamassassin  /etc/mail/spamassassin
  /opt/$DIR     /opt/$DIR/share/spamassassin   /etc/opt/mail/spamassassin
  $DIR          $DIR/share/spamassassin        $DIR/etc/mail/spamassassin

The files themselves are then looked for in these paths:

  - Distributed Configuration Files
        '__def_rules_dir__'
        '__prefix__/share/spamassassin'
        '/usr/local/share/spamassassin'
        '/usr/share/spamassassin'

  - Site Configuration Files
        '__local_rules_dir__'
        '__prefix__/etc/mail/spamassassin'
        '__prefix__/etc/spamassassin'
        '/usr/local/etc/spamassassin'
        '/usr/pkg/etc/spamassassin'
        '/usr/etc/spamassassin'
        '/etc/mail/spamassassin'
        '/etc/spamassassin'

  - Default User Preferences File
        '__local_rules_dir__/user_prefs.template'
        '__prefix__/etc/mail/spamassassin/user_prefs.template'
        '__prefix__/share/spamassassin/user_prefs.template'
        '/etc/spamassassin/user_prefs.template'
        '/etc/mail/spamassassin/user_prefs.template'
        '/usr/local/share/spamassassin/user_prefs.template'
        '/usr/share/spamassassin/user_prefs.template'


After installation, try "perldoc Mail::SpamAssassin::Conf" to see what
can be set. Common first-time tweaks include:

  - required_score

        Set this higher to make SpamAssassin less sensitive.
        If you are installing SpamAssassin system-wide, this is
        **strongly** recommended!

        Statistics on how many false positives to expect at various
        different thresholds are available in the "STATISTICS.txt" file in
        the "rules" directory.

  - rewrite_header, add_header

        These options affect the way messages are tagged as spam or
        non-spam. This makes it easy to identify incoming mail.

  - ok_locales

        If you expect to receive mail in non-ISO-8859 character sets (ie.
        Chinese, Cyrillic, Japanese, Korean, or Thai) then set this.


Learning
--------

SpamAssassin includes a Bayesian learning filter, so it is worthwhile
training SpamAssassin with your collection of non-spam and spam,
if possible.  This will make it more accurate for your incoming mail.
Do this using the "sa-learn" tools, like so:

        sa-learn --spam ~/Mail/saved-spam-folder
        sa-learn --ham ~/Mail/inbox
        sa-learn --ham ~/Mail/other-nonspam-folder


If these are mail folders in mbox format, use the --mbox switch, for
Maildirs use a trailing slash, like Maildir/cur/.
```

Use as many mailboxes as you like. Note that SpamAssassin will remember
what mails it has learnt from, so you can re-run this as often as you like.

Locali[sz]ation

All text displayed to users is taken from the configuration files. This
means that you can translate messages, test descriptions, and templates
into other languages.

If you do so, we would *really* appreciate it if you could contribute
these translations, so that they can be added to the
distribution. Please file a bug in our Bugzilla[4], and attach your
translations. You will, of course, be credited for this work!

        [4]: http://bugzilla.spamassassin.org/

Disabled code

There are some tests and code in SpamAssassin that are turned off by
default: experimental code, slow code, or code that depends on
non-open-source software or services that are not always free. These
disabled tests include:

  - DCC: depends on non-open-source software (disabled in init.pre)
  - DomainKeys: experimental (disabled in init.pre)
  - MAPS: commercial service (disabled in 50_scores.cf)
  - MSExec: experimental (disabled in init.pre)
  - Razor2: depends on service that is not always free (disabled in init.pre)
  - TextCat: slow (disabled in init.pre)

To turn on tests disabled in 50_scores.cf, simply assign them a non-zero
score, e.g. by adding score lines to your ~/.spamassassin/user_prefs file.

To turn on tests disabled by commenting out the required plugin in
init.pre, you need to uncomment the loadplugin line and make sure the
prerequisites for proper operation of the plugin are present.

Automatic Whitelist System

SpamAssassin includes automatic whitelisting; The current iteration is
considerably more complex than the original version. The way it works is
by tracking for each sender address the average score of messages so far
seen from there. Then, it combines this long-term average score for the
sender with the score for the particular message being evaluated, after
all other rules have been applied.

This functionality is on by default, and is enabled or disabled with the
"use_auto_whitelist" option.

A system-wide auto-whitelist can be used, by setting the
auto_whitelist_path and auto_whitelist_file_mode configuration commands
appropriately, e.g.

    auto_whitelist_path   /var/spool/spamassassin/auto-whitelist
    auto_whitelist_file_mode   0666

The spamassassin -W and -R command line flags provide an API to add and
remove entries 'manually', if you so desire. They operate based on an
input mail message, to allow them to be set up as aliases which users can
simply forward their mails to. See the spamassassin manual page for more
details.

The default address-list implementation,
Mail::SpamAssassin::DBBasedAddrList, uses Berkeley DB files to store

```
the addresses.

(end of README)

// vim:tw=74:
```

Comments

Please enter comments here. You can type @"SIG@ to insert your signature. – DuncanFindlay <<DateTime(2005-08-22T02:40:39Z)>>