FasterPerformance

How do I get SpamAssassin to run faster?

General Advice

Examine the custom rule set files you use:

- Avoid large rule sets, those over 100k or 150k in size. The more rules you have, the slower SA will run.
- In particular, the `blacklist` and `blacklist-uri` (aka `sa-blacklist` and `sa-blacklist-uri`) rulesets are extremely heavyweight, and massively affect performance and memory usage. If you're using these, **remove them immediately**. Ensure network tests are enabled, instead; those rules have been made obsolete by the URIBL_WS_SURBL network rule. See OutOfMemoryProblems.
- Pick rule set files that are more productive. In the SARE families published by Bob Menschel, use files 0 and 1 for productivity / efficiency, and avoid files 2 and 3.
- Remove and re-add rule set files one at a time, and check performance after each change. If one rule set file causes a huge change in performance, take appropriate action.

Examine the custom rules you create, or have downloaded from third parties. Poorly-written regular expressions can use resources exponentially (see "Finding slow rules" chapter below). Avoid body, rawbody, or full rules that use + or * quantifiers.

Use spamd, or glue like Amavisd-new or Mimedefang.

Try using `sa-compile` if you're running SA 3.2.x or later.

If you are using network tests, install a local DNS server (BIND named, for example) on the same host to cache responses, and set the /etc/resolv.conf file to use that instead of one on another machine. See CachingNameserver.

If you're seeing periodic load spikes, it could be because the Bayes database is re-syncing and expiring old tokens. See BayesForceExpire for a workaround. For best performance and maintenance free operation, use the recommended Redis backend for Bayes, see SiteWideBayesSetup.

If you are memory-bound

If the spamd processes are eating up all the RAM on your machine, then you are memory-bound.

Are you experiencing high system load or possibly swapping? Look at the number of children you have spawned, and compare that to the available memory (by default each child can use tens to hundreds of megs of RAM, depending on scanned mail size). Depending on load you might find success in lowering the number of children that are spawned (see -m in the spamd documentation).

See also OutOfMemoryProblems.

If you are I/O-bound

For heavily loaded servers, you may be experiencing high iowait times depending on how hard you are hitting your disk. You can try offloading the logging and bayes disk writes to a separate disk, or even disabling Bayes rules entirely with `use_bayes 0`. For best performance and maintenance free operation, use the recommended Redis backend for Bayes, see SiteWideBayesSetup.

If the auto-whitelist is in use (user config dirs contain files named "auto-whitelist"), you should turn that off; it provides a marginal gain in accuracy for quite a bit of I/O load. Set `use_auto_whitelist 0`.

If you are CPU-bound

If your server is being limited by CPU load:

Try using `sa-compile`, though it can't do miracles. Most likely you have bad rules somewhere, see "Finding slow rules" chapter below.

See OutOfMemoryProblems. Much of the advice applies for CPU-bound machines, too.

Network related

If you are UsingNetworkTests, install a local DNS server (BIND named, for example) on the same host to cache responses, and set the /etc/resolv.conf file to use that instead of one on another machine. See CachingNameserver.

External network tests often take long time (compared to a non-network installation). See UsingNetworkTests for general hints.

- Pyzor (see UsingPyzor) may add few seconds to testing time. Starting with SA 4.0, use pyzor_fork 1 option to run it asynchronously.
- DCC (see UsingDcc) may add few seconds to testing time. Starting with SA 4.0, use dccifd daemon to run queries asynchronously.
- Razor2 (see UsingRazor) may add few seconds to testing time. Starting with SA 4.0, use razor_fork 1 option to run it asynchronously.

Consider turning off network tests, and running with "-L", if you can afford a large drop in accuracy. This is not a very good option for most people though, and while it will reduce system memory load by reducing the number of simultaneous processes, it will *increase* system CPU load, so be warned! See NetworkTestsLatency for more info.

Finding slow rules

Download http://svn.apache.org/repos/asf/spamassassin/trunk/masses/plugins/HitFreqsRuleTiming.pm somewhere.

Run this command against some badly performing messages:

```
spamassassin --cf 'loadplugin HitFreqsRuleTiming /path/to/HitFreqsRuleTiming.pm' -L messagefile
```

File named "timing.log" is created in current directory, containing the number of seconds each rule took to run, in the 2nd and 3rd columns.

You can also add the loadplugin line to your normal configuration, it might marginally slow down scanning, but should not be noticeable.

More general advice

SA should log per-message scan times to the system log. From that you should be able to determine a message recipient and message-ID for messages that take a long time to scan.

You may also be experiencing inbound traffic volume spikes that may be overloading your system.

So, if you can track when the processing spikes occur, two things to correlate are the number of messages you're processing at that time (you may have too many spamd children defined, or not enough memory - are you hitting swap?), and any specific messages received at that time that take an unusually long time to scan (they may be exercising weaknesses in the rules).

Older versions have had rules that were known to perform poorly in certain situations and that have been fixed in newer versions.