OutOfMemoryProblems

Memory problems with SpamAssassin

If you are seeing mailserver meltdowns due to load imposed by SpamAssassin, here's a checklist of items you should run through.

Heavyweight custom rules

You should also avoid using very large custom rules files. The larger custom rules files available from the SA community can double or triple your memory usage.

In particular, the following CustomRulesets *will* kill your server:

- blacklist.cf (aka. sa-blacklist.cf)
- blacklist-uri.cf (aka. sa-blacklist-uri.cf)
- bigevil.cf

They have been deprecated *years* ago, and should not be used.

Try running without custom rulesets, measure memory usage, and re-add them gradually. By doing this you can determine which rulesets are worth the memory/accuracy trade-off. Note also that old rulesets probably won't catch any spam anymore, so you may be adding load for no good reason. Only use rulesets that are current and updated.

Spamd

If you have enough load to run into memory problems, and you're not using *spamd*, MailScanner, Amavisd-new, Mimedefang or another system that keeps the `Mail::SpamAssassin` modules loaded persistently, then that should be your first priority.

The overhead of starting the perl interpreter, compiling the code, compiling the SpamAssassin ruleset, and so on is quite high; *spamd* was designed to get around that problem. It greatly reduces the per-message scan time.

Simultaneous scans

If you're using spamassassin to filter all mail on your domain, you *will* run out of memory, unless you impose a limit on simultaneous scans.

Many MTAs (mail transfer agents) will allow unlimited simultaneous deliveries to local mail accounts. If you then start a process to handle each of those deliveries – as you will if you insert SpamAssassin into the delivery process – those processes will chew up RAM and bog down the system.

Many spammers do not impose any kind of rate-throttling on their sending side; they just want to send as much spam as quickly as possible. If the receiving system breaks, that's not their problem.

SpamAssassin is quite a heavyweight filter, so it's more likely to happen with SA running; but would probably happen with any filter if you're not careful.

To avoid this, you should (a) use "spamd", which is more suited for heavy use than the "spamassassin" script, and (b) impose a limit on concurrent "spamd" processes using the "-m" switch.

Finally, if you are using an MTA like Postfix that allows control over how many concurrent local deliveries to allow, set that to a sane number; for example, Postfix allows you to specify a limit of 5 local deliveries at once by adding this line to "main.cf":

```
local_destination_concurrency_limit = 5
```

This helps throttle down the load further upstream, and is very beneficial.

In extremis, if you're using procmail, you can set it to scan just one mail at a time using this stanza in the procmailrc:

```
:0fw: spamassassin.lock
* < 512000
| spamc
```

Very large mails

Spamassassin will attempt to scan anything you throw at it. However, the time taken to scan messages and memory usage rise with the message size. The best way to deal with this problem is to limit the size of messages that get scanned by SpamAssassin. Tests show that larger messages are overwhelmingly likely to be non-spam, given the economics of spamming.

For procmail, the following recipe works:

```
:0fw
* < 512000
| spamc
```

(The "* < 512000" line prevents messages over 512kB from being passed to SpamAssassin.)

If you're using spamd, you should be aware that spamc will only pass messages under 512kB in size to spamd by default. This is a built-in limit that you can override using the "-s" option on the command-line. Refer to the spamc man page for more details.

Starting with SpamAssassin 3.4.3, it is much more safe to scan larger messages. New body_part_scan_size/rawbody_part_scan_size options limit scanned data size so CPU usage won't be affected much. In 2019, spam up to 5MB in size has been seen, so it is recommended to use a large limit. Note that larger messages use much more memory per children, parsing a 5MB message can currently use 200MB memory, and 20MB message over 600MB, so keep the limit sane.

Network tests

If your scan times are very high (greater than 10 seconds per message) and you have network tests enabled, there's a good chance you're running into latency issues caused by network slowness, as described in NetworkTestsLatency.

Ensure you are running a local caching DNS server on the mailserver itself. This greatly improves latency of DNS requests, which SpamAssassin uses heavily.

Consider turning off Razor, Pyzor, or DCC, if you have those enabled. They can be very slow at times, and will typically increase latency by a second or two in the best case. Note that these limitations are removed starting from SpamAssassin 4.0, which has razor_fork, pyzor_fork options for asynchronous operation, and also using dccifd will be asynchronous.

AWL

If you're doing the above and still running out of memory intermittently, check the size of your AWL databases. Version 3.0.3 fixes an AWL specific bug that can cause memory bloat from large AWL db files.

You can also use the --clean switch of the tools/check_whitelist utility to remove entries with < n hits, providing a way to clean out the db. This should resolve this specific problem.