

Building Firewalls with OpenBSD and PF

Building Firewalls with OpenBSD and PF

Jacek Artymiak

Second Edition

Lublin

Using spamd

In this chapter we get to know one particularly interesting anti-spam tool, spamd(8).

There are two ways to fight spam: passive and active. The passive way is to accept all mail and then filter it to remove spam. While you can certainly achieve good results with modern Bayesian spam filters, this method does not do much to deter spammers and will cost you more and more in the long run. When spammers look at logs, they will see that their mail was accepted by your host and they will assume that it is OK with you to send more spam your way. More spam means more work for filters, which will use more CPU cycles, further slowing down your communication with the outside world.

The active way of fighting spam involves keeping a list of IP addresses of hosts known to send spam and reject connections from these hosts to port 25. While very efficient, this method has one tiny fault, it is a quick way to inform spammers that your host won't accept connections from their servers and can be removed from their list. They will try to deliver their payload from a different IP, or go after other hosts (or both).

An even better way to deter spammers is to make it expensive for them to deliver mail by slowing down their mail delivery software. This is exactly what *spamd(8)* does. It uses the SMTP specification [RFC 2821] to inject spam back into the sender's mail queue by sending the 450 Requested mail action not taken: mailbox unavailable error message. This method is very effective as it uses standard communication protocol, to which all mailers must adhere.

13.1 Configuring spamd

Spamd(8) is designed to co-exist with all mail daemons, and to cause the least trouble to the system administrator. Because it never accepts mail

from spammers, the load it places on the system is negligible and because it's job is very well-defined, configuring it is a breeze.

Out of all options listed in *spamd(8)*, the two that are most important are:

- `-p port` — specifies which port should *spamd(8)* listen for connections on. This cannot be port 25, since that is where the real *sendmail(8)* or other MTA is listening on. Choose one on of the higher ports that are not used for anything else, like 8025, 8125, etc.
- `-c connections` — the maximum number of concurrent connections accepted by *spamd(8)*. The default is 200, but you can adjust it up or down, as you wish.

Other options are explained in *spamd(8)* and we are not going to dwell on them here. If you are not sure which ones you need, let *spamd(8)* use the defaults.

You can start *spamd(8)* from the command line:

```
# spamd -p 8025 -c 200
```

Or, to start *spamd(8)* automatically at system startup add the following line to */etc/rc.local*:

```
spamd -p 8025 -c 200
```

Because *spamd(8)* is not listening on port 25, *pf(4)* must redirect connections from spammers' hosts to the port defined with the `-p` option (in our case, 8025). The list of spammer's addresses will be held in the `<spamd>` table, that can be updated while *pf(4)* and *spamd(8)* are running.

The contents of */etc/pf.conf* will differ depending on where the MTA is running. If you are new to *pf(4)* and *spamd(8)* start with these simple rulesets. First, we assume that the MTA is running on the firewall host. It listens on port 25, as all MTAs do:

```
# MACROS
ext_if = "ne1"

# Tables
table <spamd> persist
```

```

# NAT rules
# redirect connections from spammers to spamd, all legitimate
# connections will not be redirected
rdr on $ext_if inet proto tcp \
  from <spamd> to ($ext_if) port 25 -> 127.0.0.1 port 8025
# block all incoming connections
block in on $ext_if all
# pass redirected connections to spamd listening on the local
# loop interface (lo0)
pass in on lo0 inet proto tcp \
  from <spamd> to 127.0.0.1 port 8025
# pass legitimate connections to port 25 on the
# external interface
pass in on $ext_if inet proto tcp \
  from any to ($ext_if) port 25 flags S/SA synproxy state

```

The ruleset will look differently if you want to redirect connections to port 25 to the MTA running on another host.

```

# MACROS
ext_if = "ne1"
# here, we assume that the MTA is running on a machine
# located in the DMZ and connected to the DMZ interface
$dmz_if = "ne2"
mta_ad = "192.168.24.63"
mta_pt = "1025"

# Tables
table <spamd> persist

# NAT rules
# redirect connections from spammers to spamd
rdr on $ext_if inet proto tcp \
  from <spamd> to ($ext_if) port 25 -> 127.0.0.1 port 8025
# redirect all legitimate connections to the real MTA
rdr on $ext_if inet proto tcp \
  from any to ($ext_if) port 25 -> $mta_ad port $mta_pt
# block all incoming connections
block in on $ext_if all
# pass redirected connections to spamd listening on the local

```

```
# loop interface (lo0)
pass in on lo0 inet proto tcp \
  from <spamd> to 127.0.0.1 port 8025
pass out on $dmz_if inet proto tcp \
  from any to $mta_ad port $mta_pt flags S/SA synproxy state
```

Copy one of the above rulesets and make modifications necessary to make it work on your machine (change addresses, port numbers, interface names, etc.), save it and reload with:

```
# pfctl -F all ; pfctl -f /etc/pf.conf
```

You are now set and can begin populating the `spamd` table, either by hand, or via a script: To test the new setup, run `spamd(8)`:

```
# spamd -p 8025 -c 200
```

Next, add the address of the host from which you will try to connect to port 25 on the firewall:

```
# pfctl -t spamd -T add 192.168.23.11
```

Then, try to connect from that host (it's address will be different from the one given above, and the address of the firewall will be different from the one given below):

```
# telnet 192.168.2.1 25
```

You should see a message appearing very slowly in the terminal window. That is a sign that `spamd(8)` is working.

Next, remove the address of the test host from `<spamd>`:

```
# pfctl -t spamd -T delete 192.168.23.11
```

Then, try to connect from that host (it's address will be different from the one given above, and the address of the firewall will be different from the one given below):

```
# telnet 192.168.2.1 25
```

You should now see a banner of the MTA waiting for delivery of mail.

Once it is running, *spamd(8)* is designed to be configurable in-flight, and comes with a configuration utility, *spamd-setup(8)* which sends configuration directives and makes changes to the `spamd` table automatically. It does its magic by parsing the *spamd.conf(5)* configuration file located in */etc/spamd.conf*, retrieving blacklists (lists of addresses known to send spam), and removing addresses from whitelists (addresses that we never want to put on a blacklist, even if they manage to get on some blacklist). Then, it sends the data in the format understood by *spamd(8)* to the port that the daemon is listening on.

Spamd-setup(8) must be run from `root` account or from another user's account as long as it has access to run it via *sudo(8)*. It's best to run *spamd-setup* at regular intervals from *cron(8)*.