nfttools – NFT firewall management and reporting tools package for Slackware 15.0

Edward McGuire

April 12, 2023

Abstract

This document defines a Slackware 15.0 package containing a set of tools to manage the NFT firewall. This document's intended audience is the developer or maintainer of the package. For the system operator who is the end-user of `nfttools`, manual pages and example configuration files are built from this document and installed as part of the package.

TODO: rename logactions.pl to logactionsd.

TODO: document postinstall syslog change needed before running rc.firewall.

TODO: Add support for Slackware 15.1. document limitation on 15.1 that /etc must be local. (see below)

TODO: Consider setting a limit on how many old snapshots are kept. Consider saving them in a directory instead of loose in /etc. Or maybe move the rule snapshots to /var/lib to keep /etc clear of backup copies. (See above)

TODO: finish the reporting scripts.

TODO: audit all file names and make them sensible.

TODO: fix the security report script to find the subscripts in the right place.

TODO: explain somewhere why we snapshot the running ruleset periodically and during multiuser shutdown, not every time we add an IP block

TODO: The uniqueness filter is broken now because the log message now also includes the rule number and elapsed time.

TODO: currently hardwired filter to ”newjersey” when generating statistics.

Contents

1 build instructions for the nfttools package container file

The file nfttools.nw is a Noweb source file. It contains all the sources and documentation for the nfttools package.

To build the package container file, your Slackware build environment must have Noweb installed. To install Noweb:

(i) install the Icon programming language package from:
https://slackbuilds.org/repository/15.0/development/icon/

(ii) install the Noweb package from: https://slackbuilds.org/repository/15.0/development/noweb/

After that, the way forward is:

```
$ mkdir nfttools

$ cp (path)/(to)/nfttools.nw nfttools

$ cd nfttools

$ noweb nfttools.nw

$ make package
```

The result, if everything goes smoothly, is the Slackware 15.0 package file nfttools-(version)-noarch-(build).tgz, suitable for input to installpkg(8), upgradepkg(8), etc.

Instead of make package, you can type make and get the Makefile usage.

For more information about Noweb, see the project's homepage: https://www.cs.tufts.edu/~nr/noweb/.

2 `rc.firewall` – Slackware 15.0 run-commands script for NFT firewall

The `rc.firewall` script contains run-commands to raise and lower shields using the NFT firewall, and to start and stop the `logactions.pl` message monitor daemon. Normally the script is run by `init` during multiuser startup and shutdown.

2.1 `rc.firewall` – outline

The script is in three parts: create the script environment (preamble); define functions; dispatch the requested function.

6a ⟨*rc.firewall* 6a⟩≡

```
  ⟨rc.firewall preamble 6b⟩
  ⟨rc.firewall functions 7c⟩
  ⟨rc.firewall dispatch 7a⟩
```

This code is written to file `rc.firewall`.

2.2 `rc.firewall` – preamble: create script environment

The preamble arranges for the script language interpreter `/bin/sh` to be loaded, prevents further execution if any unhandled exception is encountered, and defines pathnames that are used elsewhere.

6b ⟨*rc.firewall preamble* 6b⟩≡ (6a)

```
  #! /bin/sh
  # This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
  set -e
  BINNAME='/usr/sbin/nft'
  CNFNAME='/etc/nftables.conf'
```

Defines:
`BINNAME`, used in chunks 7c, 10a, and 12.
`CNFNAME`, used in chunks 7c, 8b, 10a, and 13.

2.3 rc.firewall – dispatcher: read argument and dispatch requested function

The dispatcher checks for known arguments and dispatches the appropriate function.

If no known argument is offered, the argument is treated as a request for help. In that case the script types a short help message on stderr and does nothing else.

7a ⟨*rc.firewall dispatch* 7a⟩≡ (6a)

```
case $1 in
⟨rc.firewall dispatch rules 7b⟩
*) printf "%s\n" "Usage: $0 {load|save|unload|print|test|start|stop}" >&2 ;;
esac
```

2.4 rc.firewall – functions: perform the requested operation

2.4.1 rc.firewall – nftables_test: check startup configuration for validity

The test command checks the startup configuration for errors, using the nft --check utility. A report of the results is typed on stderr. The operator can use this command to validate a new startup configuration before trying to load it.

7b ⟨*rc.firewall dispatch rules* 7b⟩≡ (7a) 8a ▷

```
test) nftables_test ;;
```

Uses nftables_test 7c.

7c ⟨*rc.firewall functions* 7c⟩≡ (6a) 8b ▷

```
nftables_test() {
"$BINNAME" --check --file "$CNFNAME" &&
        printf "%s\n" "$0: saved ruleset passed validity check: $CNFNAME" >&2
}
```

Defines:
  nftables_test, used in chunk 7b.
Uses BINNAME 6b and CNFNAME 6b.

2.4.2 rc.firewall – nftables_load: load startup configuration (raise shields)

The load command copies the startup configuration to the running configuration. Any existing running configuration is overwritten. The operator can use this command to raise shields.

8a ⟨*rc.firewall dispatch rules* 7b⟩+≡ (7a) ◁7b 8c▷

```
load) nftables_load ;;
```

Uses nftables_load 8b.

8b ⟨*rc.firewall functions* 7c⟩+≡ (6a) ◁7c 9a▷

```
nftables_load() {
if [ -e "$CNFNAME" ]
then
        { echo flush ruleset ; cat "$CNFNAME" ; } | nft --file - &&
                printf "%s\n" "$0: running ruleset loaded from: $CNFNAME" >&2
else
        printf "%s\n" "$0: no startup ruleset to load: $CNFNAME" >&2
fi
}
```

Defines:
nftables_load, used in chunks 8a and 9a.
Uses CNFNAME 6b.

2.4.3 rc.firewall – nftables_start: load startup configuration and start daemon during multiuser startup

The start command is intended to be run during multiuser startup. It runs the load command to get the ruleset in place, then starts the logactions.pl message monitor daemon. The order of events is important because the daemon expects the ruleset to already be there.

8c ⟨*rc.firewall dispatch rules* 7b⟩+≡ (7a) ◁8a 9b▷

```
start) nftables_start ;;
```

Uses nftables_start 9a.

9a ⟨*rc.firewall functions* 7c⟩+≡ (6a) ◁8b 10a▷

```
nftables_start() {
nftables_load
if [ -x /usr/sbin/logactions.pl ]
then
        /usr/sbin/logactions.pl
fi
}
```

Defines:
`nftables_start`, used in chunk 8c.
Uses `nftables_load` 8b.

2.4.4 `rc.firewall` – `nftables_save`: copy running configuration to startup configuration

The `save` command saves a copy of the running configuration. The copy is saved as the startup configuration in `/etc/nftables.conf`. The operator can use this command at any time to create a backup of the running configuration. `save` uses a utility function `nftables_rotate` that preserves all previously saved rulesets using version numbering. The numbering system is similar to that used for logfiles by `logrotate (8)`.

This is a silent operation when successful becaus4 it is run hourly by the nfttools cronjob.

9b ⟨*rc.firewall dispatch rules* 7b⟩+≡ (7a) ◁8c 10b▷

```
save) nftables_save ;;
```

Uses `nftables_save` 10a.

10a ⟨*rc.firewall functions* 7c⟩+≡ (6a) ◁9a 10c▷

```
nftables_save() {
if [ -e "$CNFNAME" ]
then
#       nftables_rotate &&
#               printf "%s\n" "$0: old ruleset(s) preserved" >&2
        nftables_rotate
fi
#"$BINNAME" --numeric list ruleset >"$CNFNAME" &&
#       printf "%s\n" "$0: running ruleset saved to: $CNFNAME" >&2
"$BINNAME" --numeric list ruleset >"$CNFNAME"
}
```

Defines:
`nftables_save`, used in chunks 9b and 10c.
Uses `BINNAME` 6b, `CNFNAME` 6b, and `nftables_rotate` 13.

2.4.5 `rc.firewall` – `nftables_stop`: copy running configuration to startup configuration and stop daemon during multiuser shutdown

The `stop` command is intended to be run during multiuser shutdown. It runs the `save` command to preserve the running configuration, and stops the message monitor daemon.

10b ⟨*rc.firewall dispatch rules* 7b⟩+≡ (7a) ◁9b 11a▷

```
stop) nftables_stop ;;
```

Uses `nftables_stop` 10c.

10c ⟨*rc.firewall functions* 7c⟩+≡ (6a) ◁10a 11b▷

```
nftables_stop() {
nftables_save
pkill logactions.pl &&
        printf "%s\n" "$0: message monitor daemon stopped"
}
```

Defines:
`nftables_stop`, used in chunk 10b.
Uses `nftables_save` 10a.

2.4.6 rc.firewall – nftables_unload: clear running configuration (lower shields)

The `unload` command clears the running configuration. After it completes, there are no firewall rules in play – shields are down! This might be useful for debugging firewall issues. The command does not save a copy of the running configuration before unloading it. If you want a backup copy, use the `save` command first.

This is not what we want to happen during system shutdown. Instead we want to save a copy of the running configuration so it can be restored at reboot. See page 9, section 2.4.4, and page 10, section 2.4.5.

11a ⟨*rc.firewall dispatch rules* 7b⟩+≡ (7a) ◁10b 11c▷

```
unload) nftables_unload ;;
```

Uses `nftables_unload` 11b.

11b ⟨*rc.firewall functions* 7c⟩+≡ (6a) ◁10c 12▷

```
nftables_unload() {
nft flush ruleset &&
        printf "%s\n" "$0: ruleset flushed: shields are DOWN" >&2
}
```

Defines:
`nftables_unload`, used in chunk 11a.

2.4.7 rc.firewall – nftables_print: type the running configuration

The `print` command types the running configuration on `stdout`. It makes no changes. It might be a useful debugging tool. When done, it announces the fact on `stderr`. This prevents the diagnostic message from interfering with the use of the command in a pipeline.

11c ⟨*rc.firewall dispatch rules* 7b⟩+≡ (7a) ◁11a

```
print) nftables_print ;;
```

Uses `nftables_print` 12.

12 ⟨*rc.firewall functions* 7c⟩+≡ (6a) ◁11b 13▷

```
nftables_print() {
"$BINNAME" --numeric list ruleset &&
        printf "%s\n" "$0: running ruleset printed" >&2
}
```

Defines:

`nftables_print`, used in chunk 11c.

Uses `BINNAME` 6b.

2.4.8 rc.firewall – nftables_rotate: preserve saved rulesets, logrotate-style

The nftables_rotate function is a utility used by nftables_save. The function renames any existing startup configuration file /etc/nftables.conf, adding a logrotate-style version number .1. If the backup file already exists, it is first renamed as .2, and so on.

Without some kind of aging, the operator would need to clean these backups up over time, as they accumulate without bounds. Instead, we delete backups that are older than 259200 seconds (three days).

The function uses head recursion until it locates its base case, which is a free version number. Then it renames and removes backups as it backtracks.

The function does not need a dispatch rule. It is not intended to be called by the operator.

13 ⟨*rc.firewall functions* 7c⟩+≡ (6a) ◁12

```
nftables_rotate() {
if [ -e "$CNFNAME".$(($1+1)) ]
then
        nftables_rotate $(($1+1))
fi
if [ $# -eq 0 ]
then
        if [ $((‘ date -r "$CNFNAME" +%s ‘+259200)) -lt ‘ date +%s ‘ ]
        then
                rm "$CNFNAME"
        else
                mv "$CNFNAME" "$CNFNAME".1
        fi
else
        if [ $((‘ date -r "$CNFNAME".$1 +%s ‘+259200)) -lt ‘ date +%s ‘ ]
        then
                rm "$CNFNAME".$1
        else
                mv "$CNFNAME".$1 "$CNFNAME".$(($1+1))
        fi
fi
```

```
  }
```

Defines:
  `nftables_rotate`, used in chunk 10a.

Uses `CNFNAME` 6b.

2.5 Relationship of `rc.firewall` to `init`

This script has an intimate relationship with other startup and shutdown scripts that are executed by `init`. What follows is a discussion of where this script is registered for execution, and why.

2.5.1 `rc.firewall` – registering the script to run at multiuser startup

The order of events during multiuser startup is critical. To protect network services, `init` should bring shields up and start the message monitor daemon before starting any other networked processes.

`init` in Slackware 15.0 already has support for raising shields at the right moment during multiuser startup. The script `rc.M` is run by `init` when the system is entering runlevel 2, 3, 4, or 5. It starts network services by calling `rc.inet2`. And the first thing `rc.inet2` does is raise shields by calling `rc.firewall`, if it exists.

That is the right order of events. This is the rationale for naming the file `rc.firewall` and installing it in `/etc/rc.d`. See page 150, section 19.3.

Like any run-commands script, this script will be passed the `start` argument when it is invoked for startup (see page 8, section 2.4.2).

`rc.firewall` – alternative startup registration method A possible alternative is to install `rc.firewall` as a System V init script in `/etc/rc.d/init.d`, then create the appropriate symbolic links to cause it to be run with the `start` argument when entering runlevel 2, 3, 4, or 5:

- `rc2.d/S01firewall` (unused by Slackware 15, treat like runlevel 3)
- `rc3.d/S01firewall` (go multi-user, with a console login prompt – default runlevel)
- `rc4.d/S01firewall` (go multi-user, with a GUI session manager)
- `rc5.d/S01firewall` (unused by Slackware 15, treat like runlevel 3)

The only advantage of this alternative is consistency. During shutdown, this script does run as a System V init script. So it would make sense to run it as a System V init script at startup, if possible. See page 16, section 2.5.2.

But there is a showstopping problem: Slackware 15.0 runs System V init scripts at startup far too late for the purpose. They are almost an afterthought. `rc.M` runs `rc.sysvinit` long after network processes are started. That is why this script cannot use the System V init script mechanism to bring shields up.

2.5.2 rc.firewall – registering it to run during multiuser shutdown

During shutdown, the requirements are different than at startup. During multiuser shutdown, the important thing is to preserve the running configuration in the startup configuration file /etc/nftables.conf. This ensures that, at the next multiuser startup, the preserved configuration is reloaded into the firewall. What we do not need to do is bring shields down. They can stay up throughout the shutdown process. This means the order of events is not as critical during shutdown as it is during startup. We can save the running configuration anytime during shutdown. At that time we will also stop the message monitor daemon.

The trouble is that Slackware 15.0 does not anticipate the need to run rc.firewall during multiuser shutdown.

One option is to require the operator to manually add a call to rc.firewall somewhere. It is not recommended to modify the stock rc scripts, because a future package update might replace them. So this approach means the operator should create or update rc.local_shutdown and add a block such as:

```
if [ -x /etc/rc.d/rc.firewall ]; then
  /etc/rc.d/rc.firewall stop
fi
```

The drawback is there is room for error in any manual process. So rather than requiring the operator to maintain a script, we take the System V init script approach, but only for shutdown. We create the following symlinks.

- rc0.d/K99firewall (halt)
- rc1.d/K99firewall (go single-user)
- rc6.d/K99firewall (reboot)

These link to rc.firewall where it is installed in rc.d. Their existence will cause rc.firewall to be run with the stop argument when entering runlevel 0, 1, or 6.

Slackware 15.1 is planned to include a shutdown hook /etc/rc.d/rc.firewall_shutdown. If that materializes, the System V init script approach can be abandoned. In its place we will create a short rc.firewall_shutdown containing the block above that runs rc.firewall
stop.

3 nftables.conf.example – a starting configuration for the NFT firewall

The stock NFT firewall configuration file illustrates firewall setup for a device that accepts outbound connections but drops inbound connections. Because this is for training purposes, most of the explanation below is in comments in the file itself.

17 ⟨*nftables.conf.example* 17⟩≡

```
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.

# This example NFT firewall configuration file illustrates
# firewall setup. Unmodified, it is suitable for a device
# such as a cloud-based server that accepts no inbound
# traffic, except a remote secure shell that lets the owner
# log in.

# To enable this file, copy it to "/etc/nftables.conf" and
# then load it using the command:
#
#       /etc/rc.d/rc.firewall start

# This table manages firewall rules for both IPv4 and IPv6
# packets.

table inet TABLE_INET_MAIN {

        ⟨nftables.conf.example inet base chain 18⟩

        ⟨nftables.conf.example ipv6 subchain 19⟩

        ⟨nftables.conf.example inet forward chain 20⟩

}
```

This code is written to file `nftables.conf.example`.
Defines:
`TABLE_INET_MAIN`, used in chunks 46 and 139.

18 ⟨*nftables.conf.example inet base chain* 18⟩≡ (17)

```
# This base chain attaches to the input packet hook. This
# means all input IPv4 and IPv6 packets must get past the
# rules in this chain. Unmodified, it drops any packet
# unless a rule specifically accepts it. Exceptions are
# documented below.

chain CHAIN_INET_MAIN_INBOUND {

        # Attach to the input packet hook. By default, drop.

        type filter hook input priority 0; policy drop;

        # Consult connection tracking. Quickly drop the
        # packet if it has invalid state, and accept the
        # packet if its state shows it belongs to an
        # established or related connection.

        ct state vmap { invalid : drop, established : accept, related : accept }

        # Otherwise, accept the packet anyway if its origin
        # was the local loopback device. We are allowed to
        # talk to ourselves.

        iifname "lo" accept

        # Otherwise, if this is an IPv6 packet, then call an
        # IPv6 chain for address family-specific decisions
        # before going on.

        meta protocol vmap { ip6 : jump CHAIN_INET_MAIN_IPV6 }

        # Otherwise, accept incoming TCP packets to the SSH
        # port to support remote secure shells.
        tcp dport { "ssh" } accept

        # Otherwise, the packet was not recognized. The
        # default policy will drop it.
}
```

Defines:
CHAIN_INET_MAIN_INBOUND, never used.
Uses CHAIN_INET_MAIN_IPV6 19.

19 ⟨*nftables.conf.example ipv6 subchain* 19⟩≡ (17)

```
# This subchain is called by the input packet base chain
# when we get get an IPv6 packet. It accepts the inbound
# ICMPV6 packet types that are necessary for IPv6
# pathfinding.

chain CHAIN_INET_MAIN_IPV6 {

        # Consult the packet type. If it is necessary to
        # IPv6 pathfinding, accept it.

        icmpv6 type { nd-router-advert, nd-neighbor-solicit, nd-neighbor-advert } accept

        # Otherwise, the packet was not recognized here.
        # Return to the base chain and keep looking.
}
```

Defines:
CHAIN_INET_MAIN_IPV6, used in chunk 18.

20 ⟨*nftables.conf.example inet forward chain* 20⟩≡ (17)

```
# This base chain attaches to the forward packet hook. The
# forward hook would be useful in a router, but a
# cloud-based server that is not a router should reject
# packet forwards.

chain CHAIN_INET_MAIN_FORWARD {

        # Attach to the forward packet hook. By default,
        # drop.

        type filter hook forward priority 0; policy drop;

        # As there are no other rules in the chain, the
        # default policy will always drop it.
}
```

Defines:
CHAIN_INET_MAIN_FORWARD, never used.

4 rc.firewall.8 – manual page for rc.firewall

This file provides the content behind the commands:

- man rc.firewall
- info rc.firewall

21 ⟨*rc.firewall.8* 21⟩≡

```
.\" This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
.
.TH rc.firewall 8
.
.SH NAME
rc.firewall \- load, save, and otherwise manage NFT firewall processes and
settings
.
.SH SYNOPSIS
.SY rc.firewall
{ \fBload\fP | \fBsave\fP | \fBunload\fP | \fBprint\fP | \fBtest\fP }
.SY rc.firewall
{ \fBstart\fP | \fBstop\fP }
.
.SH DESCRIPTION
This program defines commands that the operator can use to manage NFT firewall
settings.
The program is run automatically during multiuser startup and shutdown.
.P
When run at multiuser startup by \fCrc.inet2\fP, the program is passed the
\fBstart\fP argument.
This causes it to load the startup configuration file as the firewall running
configuration, and start the message monitor daemon \fClogactions.pl\fP.
.P
When run at multiuser shutdown by \fCrc.sysvinit\fP, the program is passed the
\fBstop\fP argument.
```

```
This causes it to save the firewall running configuration in the startup
configuration file, preserving any existing file, to be reloaded at the next
boot time, and stop the message monitor daemon.
.
.SH OPTIONS
.
.TP
.B load
Copies the startup configuration to the running configuration.
Any existing running configuration is overwritten.
This command can be run by an operator who has just made a change to the startup
configuration and wants to promote that change to the running configuration.
.
.TP
.B save
Saves a copy of the running configuration.
The copy is saved as the startup configuration.
.
.TP
.B unload
Clears the running configuration.
This command does not save a copy of the running configuration before unloading
it.
If you want a backup copy, use the \fBsave\fP mode first.
After it completes, there are no firewall rules \(en shields are down!
This might be useful for debugging firewall issues.
This command is not what you want to do during system shutdown.
See the \fBsave\fP command.
.
.TP
.B print
Types the running configuration on \fCstdout\fP.
This command makes no changes.
When complete, it announces the fact on \fCstderr\fP.
```

```
This makes the command suitable for use in a pipeline.
.
.TP
.B test
Checks the startup configuration for errors, using the \fBnft\~\-\-check\fP
command.
A report of the results is typed on \fCstderr\fP.
The operator can use this command to validate a startup configuration before
trying to load it.
.
.TP
.B start
This command is normally run at multiuser startup, as discussed above.
It runs the \fBload\fP command and starts the message monitor daemon
\fClogactions.pl\fP.
.
.TP
.B stop
This command is normally run at multiuser shutdown, as discussed above.
It runs the \fBsave\fP command and stops the message monitor daemon
\fClogactions.pl\fP.
.
.SH EXIT STATUS
.
.TP
0
Success.
.
.TP
other
A utility called by \fCrc.firewall\fP returned a non-zero exit status.
.
.SH ERRORS
Error messages are always typed on \fCstderr\fP.
```

```
.
.TP
.B Usage: ...
The command given was not recognized.
.
.TP
.B saved ruleset passed validity check
The \fBtest\fP command found no errors in the firewall startup configuration
file.
.
.TP
.B running ruleset loaded from: \fIfilename\fP
The \fBload\fP command successfully copied the firewall startup configuration
file to the running configuration.
.
.TP
.B no startup ruleset to load: \fIfilename\fP
The \fBload\fP command did not find the firewall startup configuration file.
.
.TP
.B old ruleset(s) preserved
The \fBsave\fP command found existing configuration file(s), and renamed them
rather than overwriting them.
.
.TP
.B running ruleset saved to: \fIfilename\fP
The \fBsave\fP command successfully copied the firewall running configuration to
the startup configuration file.
.
.TP
.B ruleset flushed: shields are DOWN
The \fBunload\fP command successfully cleared the firewall running
configuration.
The NFT firewall is not filtering anything.
```

```
Shields are down.
.
.TP
.B running ruleset printed
The \fBprint\fP command successfully typed the running configuration on
\fIstdout\fP.
.
.SH ENVIRONMENT
.
The environment vector is passed unchanged to the utilities called by the
command, such as \fBnft\fP.
.
.SH FILES
.
.TP
\fC/etc/nftables.conf\fP
The NFT firewall startup configuration file
.
.TP
\fC/etc/rc.d/rc0.d/K99firewall\fP
.TQ
\fC/etc/rc.d/rc1.d/K99firewall\fP
.TQ
\fC/etc/rc.d/rc6.d/K99firewall\fP
Symbolic links to \fC/etc/rc.d/rc.firewall\fP that cause it to run during
shutdown
.
.SH VERSIONS
1.0a \(en alpha-quality release by Edward McGuire.
.
.SH "CONFORMING TO"
Slackware 15.0
.
.SH NOTES
```

```
The Noweb literate programming tool by Norman Ramsey was used to develop this
package.
.
.SH BUGS
Report bugs to the author.
.
.SH EXAMPLES
.TP
# \fB/etc/rc.d/rc.firewall start\fP
.TQ
/etc/rc.d/rc.firewall: running ruleset loaded from: /etc/nftables.conf
.TP
# \fB/etc/rc.d/rc.firewall stop\fP
.TQ
/etc/rc.d/rc.firewall: old ruleset(s) preserved
.TQ
/etc/rc.d/rc.firewall: running ruleset saved to: /etc/nftables.conf
.
.SH AUTHORS
Edward McGuire
.
.SH "SEE ALSO"
.BR logactions.pl(8) ,
.BR init(8)
```

This code is written to file `rc.firewall.8`.

5 logactions.pl – daemon to monitor system log and block nuisance traffic sources in NFT firewall in real time

27a ⟨*logactions.pl* 27a⟩≡

```
⟨logactions.pl preamble 27b⟩
⟨logactions.pl forward declarations 40⟩
⟨logactions.pl main 28a⟩
⟨logactions.pl subroutines 41⟩
```

This code is written to file `logactions.pl`.

5.1 logactions.pl – Preamble

The preamble creates the runtime environment appropriate to this script. This script is run in `taintperl` mode as a precaution because we will run some system commands as super-user.

27b ⟨*logactions.pl preamble* 27b⟩≡ (27a) 27c▷

```
#! /usr/bin/perl -T
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
```

The preamble asserts Perl script language version 5.34 (or better), with warnings, and some core modules. The Syslog module is told with `:extended` to let the script use `setlogsock()` to work around a bug in the module. The `Time::HiRes` module lets the script get subsecond measurements of the runtime taken to complete a step.

27c ⟨*logactions.pl preamble* 27b⟩+≡ (27a) ◁27b 27d▷

```
use v5.34.0 ;
use warnings ;
use Carp ;
use Sys::Syslog qw {:standard :macros :extended} ;
use Time::HiRes qw {time} ;
```

Because we are running in `taintperl` mode, we are required to set `PATH`.

27d ⟨*logactions.pl preamble* 27b⟩+≡ (27a) ◁27c

```
$ENV{PATH} = ’/usr/sbin:/sbin:/usr/bin:/bin’ ;
```

5.2 logactions.pl – Main

Create variables used by the reporting mode to capture the most recent `syslog` messages.

28a ⟨*logactions.pl main* 28a⟩≡ (27a) 28b ▷

```
my @lastten ;
my $logfilepath ;
my $logmessage ;
```

Defines:
`lastten`, never used.
`logfilepath`, used in chunks 37 and 43.
`logmessage`, used in chunks 37 and 43.

Create variables used for diagnostic messages.

28b ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁ 28a 29 ▷

```
my %diag_level = (
        s => LOG_DEBUG ,
        i => LOG_INFO ,
        w => LOG_WARNING ,
        e => LOG_ERR ,
        f => LOG_CRIT ,
) ;
```

Defines:
`diag_level`, used in chunk 56a.

Create variables to keep option values and give them reasonable defaults. Accept command-line arguments that set option values.

29 ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁28b 30a▷

```
my $OPTION_PRINT = 0 ;
my $OPTION_DEBUG = 0 ;
my $OPTION_RULEQUERY = 0 ;

$diag_identification = 'argv' ;
foreach ( @ARGV )
{
        if ( $OPTION_RULEQUERY == -1 )
        {
                if ( m {^[1-9][0-9]*$} ) { $OPTION_RULEQUERY = $_ }
                else { sub_diag 'f' , qq {$0: -r (rulequery) needs a rule number\n} }
        }
        elsif ( m {^-p$} ) { $OPTION_PRINT = 1 }
        elsif ( m {^-d$} ) { $OPTION_DEBUG++ }
        elsif ( m {^-r$} ) { $OPTION_RULEQUERY = -1 }
        else
        {
                sub_diag 'f' , qq {usage: $0 [-p] [-d] [-r rulenumber]
\t-p\t(p)rint recent interesting events, do not detach
\t-d\t(d)ebug, write progress to stderr, do not detach
\t-r rulenumber\t(r)ulequery, write rule rulenumber to stderr and exit}
        }
}
$diag_identification = 'UNKNOWN' ;
```

Defines:
- `OPTION_DEBUG`, used in chunks 32, 35, 37, 38, 41, 46, 48, 50, 52, and 57.
- `OPTION_PRINT`, used in chunks 37, 43, 46, 48, 50, 52, and 53b.
- `OPTION_RULEQUERY`, used in chunk 57.

Uses `diag_identification` 55 and `sub_diag` 56a.

Predefine some regular expressions. Most are intended for use in configuration file rules.

UNIX process ID.

30a ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁29 30b▷

```
  my $re_pid = '\[[0-9]+\]' ;
```

Defines:
  re_pid, used in chunk 61.

IP addresses. IPv4 dot-decimal notation and IPv6 colon-hex notation are both supported.

30b ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁30a 30c▷

```
  my $re_ipv4 = '(?:[012])?(?:[0-9])?[0-9]\.(?:[012])?(?:[0-9])?[0-9]\.(?:[012])?(?:[0-9])?[0-9]\.(?:[012])?(?:[0-9])?[0-9]' ;
  my $re_ipv6 = '[0-9a-f]*(?::[0-9a-f]*){2,}' ;
  my $re_ip = "(?:$re_ipv4|$re_ipv6)" ;
```

Defines:
  re_ip, used in chunks 61 and 106.
  re_ipv4, used in chunks 46, 61, 122, and 139.
  re_ipv6, used in chunks 46, 52, 61, and 122.

Hostnames. This pattern matches domain names that meet the specific requirements for hostnames laid out in RFC1034 and RFC1123. A hostname consists of dot-separated labels. A label is a letter-digit optionally followed by a letter-digit-hyphen-string and a closing letter-digit. This means the pattern does NOT recognize erroneous hostnames such as a label containing an underscore, or a label that starts or ends with a hyphen. These are very rare but they do happen, notably in PTR records. For example:

```
# dig +short -x 121.58.242.164
164.242.58.121.-rev.convergeict.com.
# dig +short -x 78.198.111.128
lv929-1_migr-78-198-111-128.fbx.proxad.net.
```

30c ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁30b 31a▷

```
  my $re_hname = '(?:[a-zA-Z0-9]|[a-zA-Z0-9][a-zA-Z0-9-]*[a-zA-Z0-9])(?:\.(?:[a-zA-Z0-9]|[a-zA-Z0-9][a-zA-Z0-9-]*[a-zA-Z0-9]))*' ;
```

Defines:
  re_hname, used in chunk 61.

Dovecot session ID.

31a ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁30c 31b▷

```
my $re_dovecot_session = '[A-Za-z0-9/+]+' ;
```

Defines:
`re_dovecot_session`, used in chunk 61.

Trailing whitespace.

31b ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁31a 32▷

```
my $re_chomp = '[ \r\n]*$' ;
```

Defines:
`re_chomp`, used in chunks 37 and 38.

Define meaningful symbols for Spamhaus return codes. The first three columns of the symbol name identify the Spamhaus list on which the address or name is found. Subroutines `sub_spamhaus_zen` and `sub_spamhaus_dbl` may make a block or pass decision based on the first three columns.

If debugging, put the Spamhaus return codes in the log.

32 ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁31b 35▷

```
  my %return_codes_spamhaus = (
          '127.0.0.1' => 'NXD' ,                  # Spamhaus does not return this, used in NXDOMAIN tests
          '127.0.0.2' => 'SBL' ,                  # Spamhaus spammers blocklist
          '127.0.0.3' => 'SBL-CSS' ,              # Spamhaus spammers blocklist, composite snowshoe subset
          '127.0.0.4' => 'XBL-CBL' ,              # Spamhaus exploits blocklist, historical CBL
          '127.0.0.5' => 'XBL-NJABL' ,            # Spamhaus exploits blocklist, historical NJABNL
          '127.0.0.6' => 'XBL-OPM' ,              # Spamhaus exploits blocklist, historical OPM
          '127.0.0.7' => 'XBL-7' ,                # Spamhaus exploits blocklist, reserved
          '127.0.0.8' => 'SBL-8' ,                # Spamhaus spammers blocklist, reserved
          '127.0.0.9' => 'SBL-DROP' ,             # Spamhaus spammers blocklist, don't route or peer
          '127.0.0.10' => 'PBL-ISP' ,             # Spamhaus policy blocklist, ISP maintained
          '127.0.0.11' => 'PBL-SH' ,              # Spamhaus policy blocklist, Spamhaus maintained
          '127.0.0.20' => 'ABL' ,                 # Spamhaus authentication blocklist
          '127.0.1.2' => 'DBL-SPAM' ,             # Spamhaus domain blocklist, spam domain
          '127.0.1.3' => 'DBL-REDIR' ,            # Spamhaus domain blocklist, spam redirect domain
          '127.0.1.4' => 'DBL-PHISH' ,            # Spamhaus domain blocklist, phish domain
          '127.0.1.5' => 'DBL-MALWARE' ,          # Spamhaus domain blocklist, malware domain
          '127.0.1.6' => 'DBL-BOTNET' ,           # Spamhaus domain blocklist, botnet C&C domain
          '127.0.1.102' => 'DBL-ABUSED-SPAM' ,    # Spamhaus domain blocklist, hacked legit domain for spam
          '127.0.1.103' => 'DBL-ABUSED-REDIR' ,   # Spamhaus domain blocklist, hacked legit domain redirect for spam
          '127.0.1.104' => 'DBL-ABUSED-PHISH' ,   # Spamhaus domain blocklist, hacked legit domain for phishing
          '127.0.1.105' => 'DBL-ABUSED-MALWARE' , # Spamhaus domain blocklist, hacked legit domain for malware
          '127.0.1.106' => 'DBL-ABUSED-BOTNET' ,  # Spamhaus domain blocklist, hacked legit domain for botnet C&C
          '127.0.1.255' => 'ERR-DBL-IP' ,         # Spamhaus error code, IP query against DBL name blocklist
          '127.0.2.2' => 'ZRD-2' ,                # Spamhaus zero reputation domain seen under 2 hours ago
          '127.0.2.3' => 'ZRD-3' ,                # Spamhaus zero reputation domain seen under 3 hours ago
          '127.0.2.4' => 'ZRD-4' ,                # etc.
          '127.0.2.5' => 'ZRD-5' ,
          '127.0.2.6' => 'ZRD-6' ,
          '127.0.2.7' => 'ZRD-7' ,
          '127.0.2.8' => 'ZRD-8' ,
          '127.0.2.9' => 'ZRD-9' ,
          '127.0.2.10' => 'ZRD-10' ,
          '127.0.2.11' => 'ZRD-11' ,
          '127.0.2.12' => 'ZRD-12' ,
          '127.0.2.13' => 'ZRD-13' ,
          '127.0.2.14' => 'ZRD-14' ,
          '127.0.2.15' => 'ZRD-15' ,
          '127.0.2.16' => 'ZRD-16' ,
          '127.0.2.17' => 'ZRD-17' ,
          '127.0.2.18' => 'ZRD-18' ,
          '127.0.2.19' => 'ZRD-19' ,
          '127.0.2.20' => 'ZRD-20' ,
          '127.0.2.21' => 'ZRD-21' ,
          '127.0.2.22' => 'ZRD-22' ,
          '127.0.2.23' => 'ZRD-23' ,
          '127.0.2.24' => 'ZRD-24' ,              # Spamhaus zero reputation domain seen under 24 hours ago
          '127.0.2.255' => 'ERR-ZRD-IP' ,         # Spamhaus error code, IP query against ZRD name blocklist
          '127.255.255.252' => 'ERR-NAME' ,       # Spamhaus error code, typing error in name
          '127.255.255.254' => 'ERR-GENERIC' ,    # Spamhaus error code, query via generic resolver
          '127.255.255.255' => 'ERR-EXCESSIVE' ,  # Spamhaus error code, excessive number of queries
  ) ;

  if ( $OPTION_DEBUG )
  {
          $diag_identification = 'retcodes' ;
          sub_diag 's' , 'Spamhaus return codes defined: ' . keys %return_codes_spamhaus ;
          if ( $OPTION_DEBUG >= 2 )
          {
                  foreach ( sort keys %return_codes_spamhaus )
                  { sub_diag 's' , "$_ $return_codes_spamhaus{$_}" }
          }
```

```
          $diag_identification = 'UNKNOWN' ;
  }
```

Defines:
  `return_codes_spamhaus`, used in chunks 48 and 50.
Uses `diag_identification` 55, `OPTION_DEBUG` 29, and `sub_diag` 56a.

Similarly, define meaningful symbols for the `blocklist.de` list of return codes and put them in the debug log.

35 ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁32 36a▷

```
my %return_codes_bde = (
        ’127.0.0.2’ => ’AMAVIS’ ,
        ’127.0.0.3’ => ’APACHEDDOS’ ,
        ’127.0.0.4’ => ’ASTERISK’ ,
        ’127.0.0.5’ => ’BADBOT’ ,
        ’127.0.0.6’ => ’FTP’ ,
        ’127.0.0.7’ => ’IMAP’ ,
        ’127.0.0.8’ => ’IRCBOT’ ,
        ’127.0.0.9’ => ’MAIL’ ,
        ’127.0.0.10’ => ’POP3’ ,
        ’127.0.0.11’ => ’REGBOT’ ,
        ’127.0.0.12’ => ’RFI-ATTACK’ ,
        ’127.0.0.13’ => ’SASL’ ,
        ’127.0.0.14’ => ’SSH’ ,
        ’127.0.0.15’ => ’WOOTWOOT’ ,
        ’127.0.0.16’ => ’PORTFLOOD’ ,
        ’127.0.0.17’ => ’SQL-INJECTION’ ,
        ’127.0.0.18’ => ’WEBMIN’ ,
        ’127.0.0.19’ => ’TRIGGER-SPAM’ ,
        ’127.0.0.20’ => ’MANUALL’ ,
        ’127.0.0.21’ => ’BRUTEFORCELOGIN’ ,
        ’127.0.0.22’ => ’MYSQL’ ,
) ;

if ( $OPTION_DEBUG )
{
        $diag_identification = ’retcodes’ ;
        sub_diag ’s’ , ’blocklist.de return codes defined: ’ . keys %return_codes_bde ;
        if ( $OPTION_DEBUG >= 2 )
        {
                foreach ( sort keys %return_codes_bde )
                { sub_diag ’s’ , "$_ $return_codes_bde{$_}" }
```

```
          }
          $diag_identification = 'UNKNOWN' ;
  }
```

Defines:
`return_codes_bde`, used in chunk 52.
Uses `diag_identification` 55, `OPTION_DEBUG` 29, and `sub_diag` 56a.

Create tables to contain patterns, associated actions, and catchall flags. They are loaded by `sub_config` from the configuration file when logactions is started, and when the configuration file changes.

36a ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁35 36b▷

```
  my @patterns ;
  my @actions ;
  my @catchalls ;
```

Defines:
`actions`, used in chunks 41, 57, and 61.
`catchalls`, used in chunks 41 and 61.
`patterns`, used in chunks 41, 57, 61, and 106.

Identify the logfiles to report from.

- `$logdir` - should be a directory pathname
- `$loglist` - should be a comma-separated list of logfile names

It is possible to override `$loglist` in the configuration file using the `logs` command.

36b ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁36a 36c▷

```
  my $logdir = '/var/log' ;
  my $loglist = 'cron,maillog,secure,syslog,messages,debug' ;
```

Defines:
`logdir`, used in chunks 37 and 154.
`loglist`, used in chunks 37 and 57.

Load the configuration file.

36c ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁36b 37▷

```
  sub_config ;
```

Uses `sub_config` 57.

If reporting, review and print recent interesting syslog messages.

37 ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁36c 38▷

```
if ( $OPTION_PRINT )
{
        $diag_identification = 'report' ;
        $OPTION_DEBUG and sub_diag 's' , "loglist=$loglist" ;
        for ( glob( '{' . $loglist . '}' ) )
        {
                my $logglob = $logdir . '/' . $_ . '{,.*}' ;
                @lastten = ('','','','','','','','','','') ;
                for ( sort { -M $b <=> -M $a } glob $logglob )
                {
                        $logfilepath = $_ ;
                        $OPTION_DEBUG and sub_diag 's' , "logfilepath=$logfilepath" ;
                        open my $fh , '<' , $logfilepath or sub_diag 'f' , "open $logfilepath: $!" ;
                        while ( <$fh> ) { s {$re_chomp} {} ; $logmessage = $_ ; sub_parse $_ ; }
                }
                print "\n" , @lastten ;
        }
}
```

Uses `diag_identification` 55, `logdir` 36b 150, `logfilepath` 28a, `loglist` 36b, `logmessage` 28a, `OPTION_DEBUG` 29, `OPTION_PRINT` 29, `re_chomp` 31b, `sub_diag` 56a, and `sub_parse` 41.

But if monitoring, wait for and act on new syslog messages as a daemon.

38 ⟨*logactions.pl main* 28a⟩+≡ (27a) ◁37

```
  else
  {
          $diag_identification = 'monitor' ;

          # Connect to the message fifo. This fifo can be created using:
          #       # mkfifo /var/log/fifo_logactions
          # then declared as a message destination in the sysklogd /etc/syslog.conf:
          #       *.*     |/var/log/fifo_logactions
          my $fifopath = '/var/log/fifo_logactions' ;
          open my $fh_fifo , '<' , $fifopath or sub_diag 'f' , "open $fifopath: $!" ;

          # Unless debugging, the new message monitor forks and runs as a daemon.
          my $pid_child = $OPTION_DEBUG ? 0 : fork ;

          # Parent - announce that the fork worked and stop.
          if ( $pid_child )
          {
                  sub_diag 'i' , "Starting logactions.pl, pid=$pid_child" ;
                  exit 0 ;
          }

          # Child - process new syslog messages until killed.
          else
          {

                  # Open SYSLOG to use instead of STDERR.
                  setlogsock( { type => 'unix' } ) ;
                  #openlog $diag_facility , 'pid' , LOG_AUTHPRIV ;
                  openlog $diag_facility , 'noeol,pid' , LOG_AUTHPRIV ;
                  sub_syslog LOG_NOTICE , 'started' ;

                  if ( ! $OPTION_DEBUG ) { close STDIN ; close STDOUT ; close STDERR }
```

```
else { sub_diag ’s’ , "not detached, press C-c to interrupt" }

# Be ready for signals that should kill the daemon.
# SIGTERM - received from "kill PID"
# SIGINT - received when Ctrl+C typed
$SIG{ ’TERM’  } =
$SIG{ ’INT’  } = sub {
        my ( $signal ) = @_ ;
        sub_syslog LOG_NOTICE , "Exiting on SIG$signal" ;
        exit 1 ;
} ;

# Read the fifo forever (or until signaled).
while ()
{
        # Test for end of file. This does NOT stop our
        # processing. End of file indicates that the
        # syslog daemon has been stopped, and at that
        # point we will simply read again, which will
        # block, and we’ll pick up more records when
        # syslog is restarted.
        if ( $_ = <$fh_fifo> )
        {
                s {$re_chomp} {} ;
                my $message = $_ ;
                sub_config ; # Check the configuration file for changes.
                sub_parse $message ;
        }
        else
        {
                $diag_identification = ’monitor’ ;
                sub_diag ’w’ , "fifo disconnected, waiting" ;
                # Close and reopen the fifo; the open
                # will block until syslog restarts and
```

```
                                  # reopens the fifo for writing.
                                  close $fh_fifo ;
                                  open $fh_fifo , '<' , $fifopath or sub_diag 'f' , "reopen $fifopath: $!" ;
                                  sub_diag 'i' , "fifo reconnected, resuming" ;
                          }
                  }
          }
  }
```

Uses `diag_facility` 55, `diag_identification` 55, `OPTION_DEBUG` 29, `re_chomp` 31b, `sub_config` 57, `sub_diag` 56a, `sub_parse` 41, and `sub_syslog` 53b.

5.3 `sub_parse` – Parse a message and perform the associated action

40 ⟨*logactions.pl forward declarations* 40⟩≡ (27a) 42▷

```
  sub sub_parse ;
```

Uses `sub_parse` 41.

41 ⟨*logactions.pl subroutines* 41⟩≡ (27a) 43▷

```
sub sub_parse
{
        $diag_identification = 'monitor' ;
        ! defined $_[1] or croak 'too many arguments to sub_parse' ;
        defined $_[0] or croak 'too few arguments to sub_parse' ;
        #local $_ = $_[0] ;
        for my $i ( 0 .. $#patterns )
        {
                if ( $_[0] =~ m {$patterns[ $i ]} )
                {
                        $diag_identification = $actions[ $i ] ;
                           if ( $actions[ $i ] eq 'pass' )
                                { $OPTION_DEBUG and sub_diag 's' , qq {"$_[0]"} }
                        elsif ( $actions[ $i ] eq 'kill' )
                                { sub_block $1 , $_[0] , $i+1 }
                        elsif ( $actions[ $i ] eq 'type' )
                                { sub_print $catchalls[ $i ] ? 'catchall' : '' }
                        elsif ( $actions[ $i ] eq 'both' )
                        {
                                sub_block $1 , $_[0] , $i+1 ;
                                sub_print $catchalls[ $i ] ? "catchall/would block '$1'" : "would block '$1'" ;
                        }
                        elsif ( $actions[ $i ] eq 'zkil' )
                                { sub_spamhaus_zen $1 , $_[0] , $i+1 ; sub_block $1 , $_[0] , $i+1 }
                        elsif ( $actions[ $i ] eq 'skil' )
                        {
                                sub_spamhaus_dbl $1 , $_[0] ;
                                sub_spamhaus_zen $2 , $_[0] , $i+1 ;
                                sub_block $2 , $_[0] , $i+1 ;
                        }
                        elsif ( $actions[ $i ] eq 'styp' )
                        {
                                sub_spamhaus_dbl $1 , $_[0] ;
                                sub_spamhaus_zen $2 , $_[0] , $i+1 ;
                                sub_print $catchalls[ $i ] ? ’catchall’ : ’’ ;
                        }
                        elsif ( $actions[ $i ] eq ’blde’ )
                                { sub_blocklist_de $1 , $_[0] , $i+1 ; }
                        elsif ( $actions[ $i ] eq ’btyp’ )
                        {
                                sub_blocklist_de $1 , $_[0] , $i+1 ;
                                sub_print $catchalls[ $i ] ? ’catchall’ : ’’ ;
                        }
                        else
                                { sub_diag ’f’ , ’unknown action’ }
                        return ;
                }
        }
        $diag_identification = ’monitor’ ;
        sub_diag ’f’ , ’fell through a hole’ ;
}
```

Defines:
`sub_parse`, used in chunks 37, 38, and 40.
Uses `actions` 36a, `catchalls` 36a, `diag_identification` 55, `OPTION_DEBUG` 29, `patterns` 36a, `sub_block` 46, `sub_blocklist_de` 52, `sub_diag` 56a, `sub_print` 43, `sub_spamhaus_dbl` 50, and `sub_spamhaus_zen` 48.

5.4 `sub_print` – Add a message to the report

Used only by reporting mode.

42 ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁40 45▷

```
sub sub_print ;
```

Uses `sub_print` 43.

43 ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁41 46▷

```
sub sub_print
{
        if ( $OPTION_PRINT )
        {
                ! defined $_[1] or croak 'too many arguments to sub_print' ;
                local $_ = length $_[0] ? $_[0] : '' ;
                if ( length $_ )
                {
                        push @lastten , "$logfilepath ($_) $logmessage\n" ;
                }
                else
                {
                        push @lastten , "$logfilepath $logmessage\n" ;
                }
                @lastten = @lastten[ @lastten - 10 .. $#lastten ] ;
        }
}
```

Defines:
`sub_print`, used in chunks 41 and 42.
Uses `logfilepath` 28a, `logmessage` 28a, and `OPTION_PRINT` 29.

5.5 `sub_block` – block an IP address

Used only by monitoring mode.

The original thinking behind timing out the IP address blocks was

- IP addresses do get released and reassigned
- the blocklist should be kept clean to minimize wasted search time
- conclusion: blocked IP addresses should time out eventually

A timeout of 1 month seemed realistic until it became clear that some probe sites having a large server farm only probe once or twice a day. By the time all their IP addresses have been seen and blocked, the earliest ones are probably timing out, and they never lose track of my server. So a timeout in years is more realistic.

Though not well documented, it seems the timeout parameter is a 32-bit unsigned integer, which means 136 year timeouts are possible. But there is a balance to be found – if I set it so long that it outlives my server, or the NFT firewall, or IPV4, or me, might as well not have a timeout. So I settled for jailing IP addresses for 67108864 seconds – a nice round number that works out to about 2.128 years.

The NFT firewall should be configured as:

```
table inet TABLE_INET_MAIN {
set SET_IPV4_MAIN_TEMPBLOCK {
        type ipv4_addr
        flags timeout
}
set SET_IPV6_MAIN_TEMPBLOCK {
        type ipv6_addr
        flags timeout
}
chain CHAIN_INET_MAIN_INBOUND {
        ...
        ip saddr @SET_IPV4_MAIN_TEMPBLOCK counter drop
        ip6 saddr @SET_IPV6_MAIN_TEMPBLOCK counter drop
```

```
         ...
}
```

45 ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁42 47▷

```
  sub sub_block ;
```

Uses `sub_block` 46.

46 ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁43 48▷

```
sub sub_block
{
        if ( ! $OPTION_PRINT )
        {
                ! defined $_[3] or croak 'too many arguments to sub_block' ;
                defined $_[1] or croak 'too few arguments to sub_block' ;
                my ( $ip , $reason , $rule ) = @_ ;

                state $last_ip = '' ;
                if ( $ip eq $last_ip )
                {
                        $OPTION_DEBUG and sub_diag 's' , "duplicate block: ip=$ip reason=$reason rule=$rule" ;
                        return ;
                }
                $last_ip = $ip ;

                my $nft_set ;
                   if ( $ip =~ /^$re_ipv4$/ ) { $nft_set = 'SET_IPV4_MAIN_TEMPBLOCK' }
                elsif ( $ip =~ /^$re_ipv6$/ ) { $nft_set = 'SET_IPV6_MAIN_TEMPBLOCK' }
                else
                {
                        sub_diag 'e' , "not blocked, not IPv4 or IPv6 address: ip=$ip reason=$reason rule=$rule" ;
                        return ;
                }

                my $starttime = time() ;
                my $nft_reply = ` 2>&1 nft delete element inet TABLE_INET_MAIN $nft_set \{ $ip \} ` ;
                my $elapsedtime = time() - $starttime ;
                my $exitcode = $? >> 8 ;
                if ( $exitcode != 0 and $exitcode != 1 )
                        { sub_diag 'e' , "nft delete element returned $exitcode and reported error: $nft_reply" }
                $nft_reply = ` 2>&1 nft add element inet TABLE_INET_MAIN $nft_set \{ $ip timeout 67108864s \} ` ;
                $exitcode = $? >> 8 ;
        if ( $exitcode != 0 )
            { sub_diag 'e' , "nft add element returned $exitcode and reported error: $nft_reply" }
        else
            { sub_diag 'i' , "blocked ip=$ip rule=$rule elapsed=" . sprintf( '%.5s' , $elapsedtime ) . "s" }
    }
}
```

Defines:
`sub_block`, used in chunks 41, 45, 48, and 52.
Uses `OPTION_DEBUG` 29, `OPTION_PRINT` 29, `re_ipv4` 30b, `re_ipv6` 30b, `sub_diag` 56a, and `TABLE_INET_MAIN` 17.

5.6 sub_spamhaus_zen – run an IP by Spamhaus

Only in monitoring mode.

When using Spamhaus free access, queries take the form of:

- octet4.octet3.octet2.octet1.(list).spamhaus.org - IP lists, i.e. zen, sbl, xbl, pbl, sbl-xbl
- host.domain.name.(list).spamhaus.org - name lists, i.e. dbl

when using Spamhaus data query service (DQS, pay service):

- octet4.octet3.octet2.octet1.(key).(list).dq.spamhaus.net - IP lists, i.e. the above plus AuthBL
- host.domain.name.(list).dq.spamhaus.net - name lists, i.e. the abov plus zrd
- plus there are other queries possible, such as cryptographic hashes, see the DQS documentation

47 ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁45 49▷

```
sub sub_spamhaus_zen ;
```

Uses `sub_spamhaus_zen` 48.

48 ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁46 50▷

```
sub sub_spamhaus_zen
{
        if ( ! $OPTION_PRINT )
        {
                ! defined $_[3] or croak ’too many arguments to sub_spamhaus_zen’ ;
                defined $_[1] or croak ’too few arguments to sub_spamhaus_zen’ ;
                my ( $ip , $reason , $rule ) = @_ ;

                state $last_ip = ’’ ;
                if ( $ip eq $last_ip )
                {
                        $OPTION_DEBUG and sub_diag ’s’ , "duplicate zen query: ip=$ip reason=$reason rule=$rule" ;
                        return ;
                }
                $last_ip = $ip ;

                # Turn IP into Spamhaus ZEN query.
                my ( $octet1, $octet2, $octet3, $octet4 ) = ( $ip =˜ m {ˆ(\d+)\.(\d+)\.(\d+)\.(\d+)$} ) or do
                {
                        sub_diag ’w’ , "skipped Spamhaus ZEN, not IPv4 address: ip=$ip reason=$reason rule=$rule" ;
                        return ;
                } ;
                my @zen_replies = ‘ 2>/dev/null dig +short $octet4.$octet3.$octet2.$octet1.zen.spamhaus.org ‘ ;
                my $spamhaus_zen = join ’:’ , map { chomp ; $return_codes_spamhaus{ $_ } or $_ } @zen_replies ;
                length $spamhaus_zen && sub_diag ’i’ , "ip=$ip spamhaus_zen=$spamhaus_zen" ;

                # Block when Spamhaus opinion is source of spam, exploits, or authorization hijacks.
                if ( $spamhaus_zen =˜ m{ˆSBL|ˆXBL|ˆABL|:SBL|:XBL|:ABL} )
                        { sub_block $ip , "ip=$ip spamhaus_zen=$spamhaus_zen reason=$reason" , $rule }
        }
}
```

Defines:
`sub_spamhaus_zen`, used in chunks 41 and 47.

Uses `OPTION_DEBUG` 29, `OPTION_PRINT` 29, `return_codes_spamhaus` 32, `sub_block` 46, and `sub_diag` 56a.

5.7 `sub_spamhaus_dbl` – run hostname by the Spamhaus domain blocklist

Only in monitoring mode.

49 ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁47 51▷

```
  sub sub_spamhaus_dbl ;
```

Uses `sub_spamhaus_dbl` 50.

50 ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁48 52▷

```
sub sub_spamhaus_dbl
{
        if ( ! $OPTION_PRINT )
        {
                ! defined $_[2] or croak ’too many arguments to sub_spamhaus_dbl’ ;
                defined $_[1] or croak ’too few arguments to sub_spamhaus_dbl’ ;
                my ( $hname , $reason ) = @_ ;

                state $last_hname = ’’ ;
                if ( $hname eq $last_hname )
                {
                        $OPTION_DEBUG and sub_diag ’s’ , "duplicate dbl query: hname=$hname reason=$reason" ;
                        return ;
                }
                $last_hname = $hname ;

                my @dbl_replies = ‘ 2>/dev/null dig +short $hname.dbl.spamhaus.org ‘ ;
                my $spamhaus_dbl = join ’:’ , map { chomp ; $return_codes_spamhaus{ $_ } or $_ } @dbl_replies ;
                length $spamhaus_dbl && sub_diag ’i’ , "hname=$hname spamhaus_dbl=$spamhaus_dbl" ;
        }
}
```

Defines:
`sub_spamhaus_dbl`, used in chunks 41 and 49.
Uses `OPTION_DEBUG` 29, `OPTION_PRINT` 29, `return_codes_spamhaus` 32, and `sub_diag` 56a.

5.8 sub_blocklist_de – run an IP by blocklist.de

Only in monitoring mode.

NOTE: blocklist.de operations were taken over 28 February 2022 by Abusix Inc. There will be a gradual migration to the Abusix platform. This just means the procedures below may change eventually.

51 ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁49 53a▷

```
  sub sub_blocklist_de ;
```

Uses sub_blocklist_de 52.

52 ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁50 53b▷

```
sub sub_blocklist_de
{
        if ( ! $OPTION_PRINT )
        {
                ! defined $_[3] or croak ’too many arguments to sub_blocklist_de’ ;
                defined $_[1] or croak ’too few arguments to sub_blocklist_de’ ;
                my ( $ip , $reason , $rule ) = @_ ;
                state $last_ip = ’’ ;
                if ( $ip eq $last_ip )
                {
                        $OPTION_DEBUG and sub_diag ’s’ , "duplicate bl.de query: ip=$ip reason=$reason rule=$rule" ;
                        return ;
                }
                $last_ip = $ip ;
                my ( $octet1, $octet2, $octet3, $octet4 ) = ( $ip =~ m {^(\d+)\.(\d+)\.(\d+)\.(\d+)$} ) or do
                {
                        if ( $ip =~ /^$re_ipv6$/ )
                        {
                                $OPTION_DEBUG && sub_diag ’i’ , "skipped bl.de, IPv6 address: ip=$ip reason=$reason rule=$rule" ;
                        }
                        else
                        {
                                sub_diag ’w’ , "skipped bl.de, not IPv4 or IPv6 address: ip=$ip reason=$reason rule=$rule" ;
                        }
                        return ;
                } ;
                my @bde_replies = ‘ 2>/dev/null dig +short $octet4.$octet3.$octet2.$octet1.bl.blocklist.de ‘ ;
                my $bde = join ’:’ , map { chomp ; $return_codes_bde{ $_ } or $_ } @bde_replies ;
                length $bde && sub_diag ’i’ , "ip=$ip bde=$bde" ;
                if ( length $bde ) { sub_block $ip , "ip=$ip bde=$bde reason=$reason" , $rule }
        }
}
```

Defines:

sub_blocklist_de, used in chunks 41 and 51.
Uses OPTION_DEBUG 29, OPTION_PRINT 29, re_ipv6 30b, return_codes_bde 35, sub_block 46, and sub_diag 56a.

5.9 sub_syslog – log action taken to SYSLOG

Only in monitoring mode.

53a ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁51 53c▷

```
sub sub_syslog ;
```

Uses sub_syslog 53b.

53b ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁52 54▷

```
sub sub_syslog { if ( ! $OPTION_PRINT ) { syslog @_ } }
```

Defines:
sub_syslog, used in chunks 38, 53a, and 56a.
Uses OPTION_PRINT 29.

5.10 command_stdout – run a command and get its output

Like the backquote operator, but doesn't depend on stdout to be there.

53c ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁53a 55▷

```
sub command_stdout ;
```

Uses command_stdout 54.

54 ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁53b 56a▷

```
sub command_stdout
{
        ! defined $_[1] or croak 'too many arguments to command_stdout' ;
        defined $_[0] or croak 'too few arguments to command_stdout' ;
        my ( $command ) = @_ ;
        open my $command_fh , '-|' , $command or sub_diag 'f' , 'command_stdout: pipe' ;
        my $stdout = do { local $/ ; <$command_fh> } ;
        return $stdout ;
}
```

Defines:
  `command_stdout`, used in chunks 53c and 106.
Uses `sub_diag` 56a.

5.11 sub_diag – send a diagnostic message

The message is logged to STDERR, or when that is absent to SYSLOG. Severity 'f' does not return; it terminates the process.

Argument 1 is the severity:

s = success	operation completed	LOG_DEBUG
i = information	operation completed with information	LOG_INFO
w = warning	operation partially completed	LOG_WARNING
e = error	operation failed	LOG_ERR
f = severe (fatal) error	operation failed, program terminated	LOG_CRIT

Argument 2 is the message text.

Global value `$diag_facility` is set to identify this program.

Global value `$diag_identification` is set to identify the specific operation.

55 ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁53c 56b▷

```
  sub sub_diag ;
  my $diag_facility = 'logactions' ;
  my $diag_identification = 'UNKNOWN' ;
```

Defines:
  `diag_facility`, used in chunks 38 and 56a.
  `diag_identification`, used in chunks 29, 32, 35, 37, 38, 41, 56a, and 57.
Uses `sub_diag` 56a.

56a ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁54 57▷

```
sub sub_diag
{
        ! defined $_[2] or croak 'too many arguments to sub_diag' ;
        defined $_[1] or croak 'too few arguments to sub_diag' ;
        my ( $diag_severity , $diag_text ) = @_ ;
        exists $diag_level{ $diag_severity } or croak 'illegal severity argument to sub_diag' ;
        if ( defined fileno STDERR )
                { print STDERR "%$diag_facility-$diag_severity-$diag_identification, $diag_text\n" }
        else
                { sub_syslog $diag_level{ $diag_severity } , $diag_identification . ': ' . $diag_text }
        exit 1 if $diag_severity eq 'f' ;
}
```

Defines:
`sub_diag`, used in chunks 29, 32, 35, 37, 38, 41, 46, 48, 50, 52, 54, 55, and 57.
Uses `diag_facility` 55, `diag_identification` 55, `diag_level` 28b, and `sub_syslog` 53b.

5.12 sub_config – read, or re-read, the configuration file

Takes no arguments. When called multiple times, will only re-read the configuration file if its modification time changed.

Failure to open the configuration file is reported as a serious but non-fatal error, and any existing configuration is left in place.

Failure to parse a line of the configuration file is also reported as a serious but non-fatal error, and the line is ignored.

56b ⟨*logactions.pl forward declarations* 40⟩+≡ (27a) ◁55

```
sub sub_config ;
```

Uses `sub_config` 57.

57 ⟨*logactions.pl subroutines* 41⟩+≡ (27a) ◁56a

```
sub sub_config
{
        $diag_identification = 'config' ;
        ! defined $_[0] or croak 'too many arguments to sub_config' ;
        my $cfpath = qq {/etc/logactions.conf} ;
        state $last_cf_mtime ;
        my $cf_mtime = ( stat $cfpath )[9] ;
        if ( ! defined $last_cf_mtime or $cf_mtime > $last_cf_mtime )
        {
                $last_cf_mtime = $cf_mtime ;
                open my $fh_cf , '<' , $cfpath or do
                {
                        sub_diag 'f' , "open $cfpath: $!" ;
                        return ;
                } ;

                # Clear the pattern/action database.
                undef @actions ;
                undef @patterns ;
                undef @catchalls ;

                # Load the configuration. "eval" is used to precompile
                # patterns for decent performance.
                my @nesting ;
                my $ruleprint = 0 ;
                while ( <$fh_cf> )
                {
                        # Ignore blanklines and comments.
                        if ( m {^\s*(#|$)} )
                        {
                        }
                        # Implement the "logs" command.
                        elsif ( m {^\s*logs:(.*)$} )
```

```
{
        $loglist = $1 ;
}
# Throw error on patterns containing curly brackets.
elsif ( m /[{}]/ )
{
        sub_diag ’e’ , "curly brackets forbidden: line $. of $cfpath: $_" ;
}
# Implement the "with" command. Throw error if no pattern given.
elsif ( m {^\s*with:(.*)$} )
{
        my $re ;
        eval "\$re = qr{$1}" ;
        if ( !defined $re )
                { sub_diag ’e’ , "missing pattern: line $. in $cfpath: $_" }
        else
                { push @nesting , @nesting ? $nesting[ -1 ] . $re : $re }
}
# Implement the "done" command.
elsif ( m {^\s*done$} )
{
        pop @nesting ;
}
# Implement the action commands.
elsif ( m {^\s*(pass|kill|type|both|skil|zkil|styp|blde|btyp):(.*)$} )
{
        push @actions , $1 ;
        my $re ;
        eval "\$re = qr {" . ( @nesting ? $nesting[ -1 ] : ’’ ) . $2 . "}" ;
        push @patterns , $re ;
        push @catchalls , length $2 ? 0 : 1 ;
        $ruleprint++ ;
        if ( $ruleprint == $OPTION_RULEQUERY )
        {
```

```
                        if ( $OPTION_DEBUG )
                                { sub_diag ’s’ , qq {rule $OPTION_RULEQUERY=<$re>} }
                        else
                                { sub_diag ’s’ , qq {rule $OPTION_RULEQUERY=<$1:$2>} }
                        exit 0 ;
                }
        }
        # Throw error on unknown command.
        else
        {
                sub_diag ’e’ , "unknown command: line $. in $cfpath: $_" ;
        }
}

# Add the implicit "pass" at the end. TODO not sure
# this works. Also shouldn’t it edit @catchalls?
push @actions , ’pass’ ;
push @patterns , ’’ ;

# Warn the user if a rule query operation was requested
# and we didn’t find a matching rule.
if ( $OPTION_RULEQUERY != 0 )
{
        sub_diag ’w’ , qq {rule $OPTION_RULEQUERY not found} ;
        exit 0 ;
}

# Report result.
sub_diag ’i’ , ’running ruleset loaded from ’ . $cfpath ;
$OPTION_DEBUG and sub_diag ’s’ , ’pattern/action rules defined: ’ . @patterns ;
if ( $OPTION_DEBUG >= 2 )
{
        for ( 0 .. $#patterns )
        { sub_diag ’s’ , "pattern _=$_\t$actions[ $_ ]\t" . ( defined $patterns[ $_ ] ? $patterns[ $_ ] : ’’ ) }
```

```
                }
        }
}
```

Defines:
sub_config, used in chunks 36c, 38, and 56b.
Uses actions 36a, diag_identification 55, loglist 36b, OPTION_DEBUG 29, OPTION_RULEQUERY 29, patterns 36a, and sub_diag 56a.

6 logactions.conf – ruleset for logactions.pl

This is the file that contains the rules. It serves as an initial set of rules for the new user, so contains many comments in the file itself to guide the user.

61 ⟨*logactions.conf* 61⟩≡

```
# /etc/logactions.conf
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.

# This file configures the message monitor daemon (logactions.pl). It is read
# when logactions starts up. It is also automatically re-read if logactions is
# in monitor mode and this file's modification time changes.

# There are two basic actions to take: block and report. These actions are
# associated with the two operating modes of logactions:
#
# Live-monitoring mode watches for new log entries and blocking attackers by IP
# address.
#
#       NOTE: In live-monitoring mode, logactions will log its own activity to
#       syslog. To prevent a syslog loop, in live-monitoring mode, logactions
#       always skips log activity originating from itself.
#
#       Operational log messages:
#               - Level "warn" presents operational issues.
#               - Level "notice" adds routine operational announcements.
#       Log processing activity:
#               - Level "notice" presents catchalls so we can see new attacks.
#               - Level "info" adds blocks (see known attacks).
#               - Level "debug" adds allows (see known non-attacks).
#
# Reporting mode (-p option) reviews logfiles and shows recent "interesting"
# log entries that may require further analysis and maybe adding a new pattern.
#
```

```
#       NOTE: In reporting mode, logactions looks not only at the logfiles
#       identified by the log statement, but also at their older, logrotated,
#       versions having .1, .2, etc. suffixes.

# Here, list the logfiles to report on. Logfiles are declared using the logs
# command. The argument should be a comma-separated list of logfiles.

#logs:cron,maillog,secure,syslog,messages,debug
logs:cron,maillog,apache,secure,syslog,messages,debug

# A comment is introduced by optional whitespace at the beginning of the line,
# followed by the "#" character, followed by the comment, and terminated by the
# end of the line. A comment is ignored. A blank line, or a line containing
# only whitespace, is also ignored.
#
# Statements may be preceded by whitespace to introduce indentation.
# Indentation whitespace is ignored. Otherwise whitespace is significant, i.e.,
# part of the verb or part of the pattern.
#
# The following action statements add known signatures and identify the
# appropriate action to take.
#
#       pass:PATTERN - identifies a known signature we do not block or report
#       kill:PATTERN - identifies a known signature we block
#       type:PATTERN - identifies a known signature we report
#       both:PATTERN - identifies a known signature we both block and report
#
#       skil:PATTERN - identifies a known signature we run by Spamhaus and always block
#       zkil:PATTERN - identifies a known signature we run by Spamhaus ZEN-only and always block
#
#       styp:PATTERN - identifies a known signature we run by Spamhaus (report, might block)
#
#       blde:PATTERN - identifies a known signature we run by BlockList.de (might block)
#       btyp:PATTERN - identifies a known signature we run by BlockList.de (report, might block)
```

```
#
# The colon is a separator. The PATTERN is a Perl regular expression. Trailing
# space is part of the pattern but it might be clearer to declare it using
# an escape or a character class.
#
# The "kill" and "both" pattern must mark the IP address to be blocked using a
# () capture group. The first capture group will be used.
#
# An action statement having an empty PATTERN can be used as a catchall for
# unmatched signatures. For example:
#
#       type:
#
# If the log entry falls through the end of file without matching any action
# statements the default action is "pass".
#
# The log message being examined is matched against the patterns in this file
# from top to bottom. The first match determines the action to take, and
# processing stops.
#
# All log entries are pre-processed to remove trailing SP, CR, and NL
# characters before pattern matching begins.
#
# There are some predefined macros:
#
#       $re_pid - matches a process ID
#       $re_ipv4 - matches an IPv4 address in dot-decimal notation
#       $re_ipv6 - matches an IPv6 address in colon-hex notation
#       $re_ip - matches an IP address (v4 or v6)
#       $re_hname - matches a hostname (possibly domain-qualified)
#       $re_dovecot_session - matches a Dovecot session ID
#
# A "with:PATTERN" statement identifies a pattern that will be prepended to all
# patterns after it, unless a matching "done" statement is reached. It is legal
# to nest with/done statements. Because a () capture group is used to mark the
# IP address to block, it is a good idea to use (?:) capture groups in
# "with:PATTERN" patterns.
#
# To debug patterns, try "logactions -d -p".



##########
# LOGFILE FORMAT - Left-anchor the search, and recognize the appropriate
# logfile format. Pick one.

# RFC 3164
with:^(?:Jan|Feb|Mar|Apr|May|Jun|Jul|Aug|Sep|Oct|Nov|Dec) [ 123][0-9] [012][0-9]:[0-5][0-9]:[0-5][0-9] $re_hname[ ]

# RFC 5424
#with:^ TBD



##########
# LOGACTION MESSAGES - only pass or report (type) logaction messages. Any other
# action has the potential to send a new message to syslog and cause a loop.
# Currently we report blocklist positives, and anything we don't recognize.

with:logactions$re_pid:[ ]
        pass:started$
        pass:Exiting on SIGINT$
        pass:Exiting on SIGTERM$
        pass:....: ip=($re_ip) bde=
        pass:....: hname=$re_hname spamhaus_dbl=
        pass:....: ip=($re_ip) spamhaus_zen=
        #pass:....: blocked ($re_ip) for:
        #pass:....: blocked ip=($re_ip) rule=\d+$
        pass:....: blocked ip=($re_ip) rule=\d+
        pass:....: skipped bl\.de, not IPv4 address: ip=$re_ipv6[ ]
        type:....: skipped bl\.de, not IPv4 address:
        pass:config: configuration file loaded$
        pass:monitor: fifo reconnected, resuming$
        type:
done



##########
# PAM THROWAWAYS - throw away unnotable messages from pam that multiple
# pam-aware applications (tags) can log. It expects an alphabetic tag that
# might be divided into two parts with a slash, followed by an optional process
# ID, followed by the pam message.

with:[a-z]+(?:/[a-z]+)?(?:$re_pid)?: gkr-pam:[ ]
        pass:unable to locate daemon control file$
        pass:stashed password to try later in open session$
        pass:gnome-keyring-daemon started properly$
        pass:error looking up user information$
done



##########
# GNOME-KEYRING-DAEMON - throw away error messages that are apparently either
# bugs in the keyring-daemon or configuration bugs in Slackware 15.

with:gnome-keyring-daemon$re_pid:[ ]
        pass:couldn't create socket directory: /run/user/0/keyring-\S+?: Permission denied$
        pass:couldn't bind to control socket: /run/user/0/keyring-\S+?/control: Permission denied$
        pass:The gnome-keyring control directory cannot be accessed: /run/user/0/keyring: Permission denied$
        type:
```

```
done

##########
# CRON HEALTH - type job start messages for the monthly run, and unusual
# messages.

with:crond$re_pid:[ ]
        pass:/usr/sbin/crond 4\.5 dillon's cron daemon, started with loglevel info$
        pass:mailing cron output
        type:reading
        with:FILE /var/spool/cron/crontabs/root USER root PID \d+ /usr/bin/run-parts /etc/cron\.
                pass:hourly
                pass:daily
                pass:weekly
                type:monthly
        done
        pass:FILE /var/spool/cron/crontabs/root USER root PID \d+ : LIVE ;
        pass:FILE /var/spool/cron/crontabs/root USER root PID \d+ : PEERS ;
        pass:FILE /var/spool/cron/crontabs/root USER root PID \d+ : PLOT ;
        pass:FILE /var/spool/cron/crontabs/root USER root PID \d+ : SLACKCHECK ;
        type:
done

##########
# SSHD

with:sshd$re_pid:[ ]

        # Report possible DoS attacks, and block if possible.
        both:drop connection #\d+ from \[($re_ip)\]:\d+ on \[$re_ip\]:\d+ past MaxStartups$
        type:error: beginning MaxStartups throttling$

        # Block other attacks without reporting. Some of these might be a
        # legitimate user just mistyping something, TODO convert to using
        # bl.de.
        kill:Connection closed by ($re_ip) port \d+(?: \[preauth\])?$
        kill:Connection closed by authenticating user \S+ ($re_ip) port \d+(?: \[preauth\])?$
        kill:Connection closed by invalid user \S* ($re_ip) port \d+(?: \[preauth\])?$
        kill:Connection reset by ($re_ip) port \d+(?: \[preauth\])?$
        kill:Connection reset by invalid user \S+ ($re_ip) port \d+(?: \[preauth\])?$
        kill:Disconnected from ($re_ip) port \d+(?: \[preauth\])?$
        kill:Disconnected from invalid user \S+ ($re_ip) port \d+(?: \[preauth\])?$
        kill:Disconnected from user \S+ ($re_ip) port \d+ \[preauth\]$
        kill:Disconnecting authenticating user \S+ ($re_ip) port \d+: Change of username or service not allowed
        kill:Disconnecting authenticating user \S+ ($re_ip) port \d+: Too many authentication failures \[preauth\]$
        kill:Disconnecting invalid user \S+ ($re_ip) port \d+: Too many authentication failures \[preauth\]$
        kill:Invalid user \S* from ($re_ip) port \d+$
        kill:Received disconnect from ($re_ip) port \d+:11:  \[preauth\]$
        kill:Received disconnect from ($re_ip) port \d+:11: Bye Bye \[preauth\]$
        kill:Received disconnect from ($re_ip) port \d+:11: Client disconnecting normally \[preauth\]$
        kill:Unable to negotiate with ($re_ip) port \d+: no matching host key type found
        kill:Unable to negotiate with ($re_ip) port \d+: no matching key exchange method found
        kill:banner exchange: Connection from ($re_ip) port \d+: could not read protocol version$
        kill:banner exchange: Connection from ($re_ip) port \d+: invalid format$
        kill:error: maximum authentication attempts exceeded for(?: invalid user)? \S+ from ($re_ip) port \d+ ssh2 \[preauth\]$
        kill:ssh_dispatch_run_fatal: Connection from ($re_ip) port \d+: message authentication code incorrect \[preauth\]$
        kill:ssh_dispatch_run_fatal: Connection from ($re_ip) port \d+: Connection corrupted \[preauth\]$

        # Pass sshd status updates.
        pass:Server listening on (0\.0\.0\.0|::) port 22\.
        pass:Received SIGHUP; restarting\.
        pass:Exiting on signal 15$
        pass:Received signal 15; terminating\.

        # Report but do not block disconnects during login. This might be an
        # attacker, but might be a legitimate user. Pass the IPv4 address to
        # bl.de for a recommendation.
        blde:Disconnected from authenticating user \S+ ($re_ip) port \d+ \[preauth\]$
```

```
blde:Received disconnect from ($re_ip) port \d+:11: Closed due to user request. \[preauth\]$

# Report oddities.
type:Bad packet length \d+\. \[preauth\]$
type:Corrupted MAC on input. \[preauth\]$
type:error: Protocol major versions differ: 2 vs\. 1$
type:warning: can't get client address: Connection reset by peer$

# When the client side host goes down unexpectedly, the server side
# eventually times out.
pass:Timeout, client not responding from user root ($re_ip) port \d+$

# Now that we have UseDNS turned on, we'll watch to see whether to
# block this. Pass the IPv4 address to bl.de for a recommendation.
blde:Address ($re_ip) maps to $re_hname, but this does not map back to the address\.$
blde:reverse mapping checking getaddrinfo for $re_hname \[($re_ip)\] failed\.$
blde:Nasty PTR record "$re_hname" is set up for ($re_ip), ignoring$

# Pass messages we know about. Pass the IPv4 address to bl.de for a
# recommendation.
pass:Accepted publickey for
blde:Disconnected from user \S+ ($re_ip) port \d+$
blde:Received disconnect from ($re_ip) port \d+:11: Closed due to user request\.$
pass:error: kex protocol error: type 30 seq 1 \[preauth\]$
pass:error: kex_exchange_identification: Connection closed by remote host$
pass:error: kex_exchange_identification: banner line contains invalid characters$
pass:error: kex_exchange_identification: client sent invalid protocol identifier ".*"$
pass:error: kex_exchange_identification: read: Connection reset by peer$
pass:exited MaxStartups throttling after [\d:]+, \d+ connections dropped$
pass:pam_unix\(sshd:session\): session opened for user root\(uid=0\) by \(uid=0\)$
pass:pam_unix\(sshd:session\): session closed for user root$
blde:fatal: Timeout before authentication for ($re_ip) port \d+$
blde:error: Received disconnect from ($re_ip) port \d+:14: No supported authentication methods available \[preauth\]$
        # This one can originate from a safe host.
        # Type anything unrecognized.
        type:

done

##########
# POSTFIX

with:postfix/

        # Pass miscellaneous unnotable messages.
        pass:anvil$re_pid: statistics:
        pass:bounce$re_pid: [0-9A-F]+: sender non-delivery notification: [0-9A-F]+$
        pass:cleanup$re_pid: [0-9A-F]+: message-id=<\S+>$
        pass:local$re_pid: warning: dict_nis_init: NIS domain name not set - NIS lookups disabled$
        pass:(?:local|smtp)$re_pid: [0-9A-F]+: to=<\S+>,(?: orig_to=<\S+>,)? relay=(?:local|$re_hname\[$re_ipv4\]:[0-9]+), delay=[0-9
        pass:master$re_pid: (?:daemon started|reload) -- version [0-9.]+, configuration /etc/postfix$
        pass:pickup$re_pid: [0-9A-F]+: uid=\d+ from=<\S+>$
        pass:postfix-script$re_pid: starting the Postfix mail system$
        pass:qmgr$re_pid: [0-9A-F]+: from=<\S*>, size=\d+, nrcpt=\d+ \(queue active\)$
        pass:qmgr$re_pid: [0-9A-F]+: removed$

        # Successful opportunistic TLS encryption in transit with the receiving
        # MTA. Pass the IPv4 address to bl.de for recommendation.
        blde:smtp$re_pid: Untrusted TLS connection established to $re_hname\[($re_ip)\]:25:[ ]

        ##########
        # POSTFIX/SMTPD

        with:smtpd$re_pid:[ ]

                # Handle SMTP connections having no domain name. First we pass
                # our own old mail server.
```

```
pass:connect from unknown\[174\.143\.212\.206\]$

# Any other SMTP connections having no domain name we run past
# ZEN and always block.

zkil:connect from unknown\[($re_ip)\]$

# Handle SMTP connections having a domain name and catch and
# block various bad behaviors. With these, we run the domain
# name past Spamhaus before blocking.

# MTA tried to use us as an open relay.
skil:NOQUEUE: reject: RCPT from ($re_hname)\[($re_ip)\]: 454 4\.7\.1 <\S+>: Relay access denied; from=<\S*> to=<\S+>
skil:NOQUEUE: reject: RCPT from ($re_hname)\[($re_ip)\]: 554 5\.7\.1 <\S+>: Relay access denied; from=<\S*> to=<\S+>

# Source tried to relay with address weirdness.
skil:warning: Illegal address syntax from ($re_hname)\[($re_ip)\] in (?:RCPT|MAIL) command:

# MTA has irregular DNS or rDNS.
pass:warning: hostname metaed\.com does not resolve to address 174\.143\.212\.206
        # Pass the old mail server.
skil:warning: hostname ($re_hname) does not resolve to address ($re_ip)

# MTA is a known probe, not offering mail.
skil:connect from ($re_hname\.censys-scanner\.com)\[($re_ip)\]$
skil:connect from ($re_hname\.internet-measurement\.com)\[($re_ip)\]$
skil:connect from ($re_hname\.shadowserver\.org)\[($re_ip)\]$
skil:connect from (213-229-119-151\.static\.as29550\.net)\[($re_ip)\]$
skil:connect from ($re_hname\.shodan\.io)\[($re_ip)\]$
skil:connect from ($re_hname\.cyberresilience\.io)\[($re_ip)\]$
skil:connect from ($re_hname\.onyphe\.net)\[($re_ip)\]$
skil:connect from ($re_hname\.quadmetrics\.com)\[($re_ip)\]$
skil:connect from ($re_hname\.stretchoid\.com)\[($re_ip)\]$
```

```
        # in case they fix their rDNS
skil:connect from ($re_hname\.binaryedge\.ninja)\[($re_ip)\]$
skil:connect from ($re_hname\.alphastrike\.io)\[($re_ip)\]$

# MTA name is unknown but behaving like a probe.
skil:lost connection after CONNECT from ($re_hname)\[($re_ip)\]$

# UGFzc3dvcmQ6 is base64 for "Password", this is a standard breakin attempt
skil:warning: ($re_hname)\[($re_ip)\]: SASL (?:login|LOGIN) authentication failed: UGFzc3dvcmQ6$

# Don't know why we get unknown IPs. Keep watching.
pass:connect from $re_hname\[unknown\]$
pass:lost connection after CONNECT from $re_hname\[unknown\]$
pass:disconnect from $re_hname\[unknown\] commands=0/0$

# Shame on me.
type:warning: database /etc/aliases\.db is older than source file /etc/aliases$
type:warning: database /etc/postfix/virtual\.db is older than source file /etc/postfix/virtual$

# Possible spammer but not necessarily a reflection on the MTA.
# Could have simply misspelled the mailbox. Pass IPv4 address
# to bl.de for a recommendation. Should this be Spamhaus?
btyp:NOQUEUE: reject: RCPT from $re_hname\[($re_ip)\]: 550 5.1.1 <\S+>: Recipient address rejected: User unknown in l

type:warning: TLS library problem: error:1408F10B:SSL routines:ssl3_get_record:wrong version number:ssl/record/ssl3_r
# An oddity we can't block.

# Handle all other "postfix/smtpd" connections by looking on
# the Zen and DBL lists at Spamhaus.

# All other SMTP peers are run through bl.de to see if it can
# find bad actors.
blde:connect from $re_hname\[($re_ip)\]$
```

```
        # Handle unnotable "postfix/smtpd" events.

        # These are unnotable because they follow a connect message
        # already acted upon if notable. TODO should these be passed to
        # Spamhaus or bl.de?
        pass:SSL_accept error from $re_hname\[($re_ip)\]: Connection timed out$
        pass:SSL_accept error from unknown\[($re_ip)\]: lost connection$
        pass:SSL_accept error from unknown\[($re_ip)\]: -1$
        pass:disconnect from unknown\[($re_ip)\]
        pass:lost connection after \S+ from $re_hname\[($re_ip)\]$
        pass:timeout after DATA \([0-9]+ bytes\) from $re_hname\[($re_ip)\]$
        pass:timeout after \S+ from $re_hname\[($re_ip)\]$
        pass:warning: unknown\[($re_ip)\]: SASL LOGIN authentication failed: [A-Za-z0-9]+$

        # Who does this? Why? TODO send to Spamhaus or bl.de? Try
        # bl.de. Probably should just kill.
        kill:warning: non-SMTP command from $re_hname\[($re_ip)\]

        # These are unnotable because they are consistent with normal
        # mail receipt.
        pass:Anonymous TLS connection established from $re_hname\[($re_ip)\]: TLSv1
        pass:warning: dict_nis_init: NIS domain name not set - NIS lookups disabled$
        pass:[0-9A-F]+: client=$re_hname\[($re_ip)\](?:, sasl_method=PLAIN, sasl_username=\S+)?$
        pass:disconnect from $re_hname\[($re_ip)\]

        # Not notable enough to print.
        pass:NOQUEUE: reject: RCPT from $re_hname\[($re_ip)\]: 554 5\.7\.1 <\S+>: Recipient address rejected: Not skillful; f

        # More oddities.
        kill:warning: $re_hname\[($re_ip)\]: SASL NTLM authentication failed: Invalid authentication mechanism$
        btyp:warning: numeric hostname: ($re_ip)$

done
```

```
        # Type anything unrecognized.
        type:

done

##########
# DOVECOT

with:dovecot:[ ]

        # Block on suspicious protocol errors.
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:1408F09B:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:1408F09C:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:1408F10B:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:14094416:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:1417A0C1:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:141CF06C:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:141EC044:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:142090C1:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:142090FC:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed: SSL_accept\(\) failed: error:14209102:.*: user=<>, rip=($re_ip),
        kill:imap-login: Disconnected: Connection closed \(auth failed, \d+ attempts in \d+ secs\): user=<last\.fm(?:\@metaed\.com)?>
        kill:imap-login: Disconnected: Connection closed \(no auth attempts in \d+ secs\): user=<>, rip=($re_ip), lip=$re_ip, TLS, se
        kill:imap-login: Disconnected: Connection closed \(no auth attempts in \d+ secs\): user=<>, rip=($re_ip), lip=$re_ip, TLS: Co
        kill:imap-login: Disconnected: Connection closed \(no auth attempts in \d+ secs\): user=<>, rip=($re_ip), lip=$re_ip, TLS han
        kill:imap-login: Disconnected: Connection closed: read\(size=\d+\) failed: Connection reset by peer \(no auth attempts in \d+
        kill:imap-login: Disconnected: Too many invalid commands \(no auth attempts in \d+ secs\): user=<>, rip=($re_ip), lip=$re_ip,
        kill:imap-login: Disconnected: Aborted login by logging out \(no auth attempts in \d+ secs\): user=<>, rip=($re_ip), lip=$re_
        kill:imap-login: Disconnected: Connection closed: read\(size=\d+\) failed: Connection reset by peer \(disconnected before aut
        kill:imap-login: Disconnected: Connection closed: SSL_read failed: error:140940F5:SSL routines:ssl3_read_bytes:unexpected rec

        # Report notable administrative messages.
        type:master: Warning: Killed with signal 15 \(by pid=[0-9]+ uid=0 code=kill\)$
        # Report possible probe or breakin attempt. Use bl.de recommendation if
        # present.
        btyp:imap-login: Disconnected: Connection closed (auth failed, \d+ attempts in \d+ secs): user=<\S+>, method=PLAIN, rip=($re_
        btyp:imap-login: Disconnected: Connection closed (no auth attempts in \d+ secs): user=<>, rip=($re_ip), lip=$re_ip, TLS hands

        # Pass unnotable "dovecot" events.

        # My own logins and logouts.
        pass:imap\(metaed\)<\d+><[A-Za-z0-9/+]+>: Disconnected: (?:Connection closed|Inactivity)[ ]
        pass:imap-login: Login: user=<metaed>, method=PLAIN, rip=$re_ip, lip=$re_ip, mpid=\d+, TLS, session=<$re_dovecot_session>$

        # Unnotable administrative messages.
        pass:master: Dovecot v2\.3\.17\.1 \(476cd46418\) starting up for imap, pop3, lmtp \(core dumps disabled\)$
        pass:master: Warning: Time moved forwards by [0-9]+\.[0-9]+ seconds - adjusting timeouts\.$

        # Unnotable because probably due to a block that we just did.
        blde:imap-login: Disconnected: Inactivity \(no auth attempts in \d+ secs\): user=<>, rip=($re_ip), lip=$re_ip, TLS handshakin
        blde:imap-login: Disconnected: Inactivity \(auth failed, \d+ attempts in \d+ secs\): user=<\S*>, method=PLAIN, rip=($re_ip),

        # Possible probe or breakin attempt but there are more tracks
        # elsewhere.
        pass:imap-login: Error: Diffie-Hellman key exchange requested, but no DH parameters provided. Set ssl_dh=</path/to/dh.pem$

        # Type anything unrecognized.
        type:

done

##########
# APACHE ACCESS LOG - these are logged by a GlobalLog directive in the Apache
# configuration file, using logger(1) to hand off to the syslog service.

with:httpd_access_log:[ ]
```

```
# Expect the Apache common log file format.
#with:($re_ip) - - \[[0-9/A-Za-z: +-]+\][ ]

# This parser is configured to a customized logfile format that has a
# virtual hostname at the front, followed by the client IP address,
# followed by an Apache common logfile record (starting with client
# hostname if known, else the IP address a second time).

# Read past the virtual host, kill behavior that is not tolerated
# across the board.
with:$re_hname[ ]

        kill:(185\.191\.171\.\d+)[ ]
        # Client IP for the subnet allocated to the SEMRush crawler

        kill:(185\.173\.35\.\d+)[ ]
        # Client IP for a subnet allocated to Net Systems Research

        # Otherwise parse the client IP.
        with:($re_ip)[ ]

                # Kill httpd probes and bots we can recognize by
                # hostname. NOTE: Yandex is generally considered a
                # "good bot". However it is a Russian Internet company
                # and I am already blocking Russian IP addresses for
                # excessive intrusion attempts. So there is really no
                # point to having them index my sites.
                kill:$re_hname(?:\.webmeup\.com|\.internet-census\.org|\.ahrefs\.com|\.oliveoilsunflower\.com|\.your-server\.

                # Now read past the client hostname (or repeated IP
                # when hostname is unavailable), two identity fields,
                # and the timestamp so we can filter by client request.
                with:(?:(?:$re_hname)|(?:$re_ip)) - - \[[0-9/A-Za-z: +-]+\][ ]
```

```
# Kill SEO and other bots I don’t like by client signature
kill:".*?" \d+ \d+ ".*?help\@moz.com
kill:".*?" \d+ \d+ ".*MJ12bot
kill:".*?" \d+ \d+ "Expanse
kill:".*?" \d+ \d+ ".*Censys
kill:".*?" \d+ \d+ "IonCrawl
kill:".*?" \d+ \d+ "ZoominfoBot
kill:".*?" \d+ \d+ "test

# Block malice or probes.
kill:".*wp-config
kill:"GET .*?eval-stdin
kill:"GET /CommPilot/
kill:"GET /\.DS_Store[ ]
kill:"GET /telescope/
kill:"CONNECT
kill:"GET /.*?/\.htaccess[ ]
kill:"GET /.*?/alfacgiapi[ ]
kill:"GET /.*?/cgialfa[ ]
kill:"GET /.*?/ALFA_DATA[ ]
kill:"GET /\.local[ ]$
kill:"GET /\.production[ ]$
kill:"GET //
kill:"GET /CTUlGhc6QvL2jpx38VXImdOqoKx
kill:"GET /Public/home/js/check\.js
kill:"GET /ReportServer
kill:"GET /[pP][mM][aA]/
kill:"GET .*?/\.env[ ]
kill:"GET /\.git
kill:"GET /_ignition/
kill:"GET /_profiler
kill:"GET /ab2
kill:"GET /actuator
kill:"GET /admin
```

```
kill:"GET /autodiscover
kill:"GET /blog/wp-class.php[ ]
kill:"GET /c/
kill:"GET /cgi-bin/index2\.asp
kill:"GET /config/getuser
kill:"GET /confluence
kill:"GET /credentials
kill:"GET /db
kill:"GET /debug
kill:"GET /ecp
kill:"GET /flu/
kill:"GET /info
kill:"GET /invoker/
kill:"GET /isadmin
kill:"GET /login
kill:"GET /logon
kill:"GET /manager
kill:"GET /media/system/js/core\.js
kill:"GET /muieblackcat[ ]
kill:"GET /mysql
kill:"GET /owa
kill:"GET /pages
kill:"GET /php-?[mM]y[aA]dmin
kill:"GET /phpinfo
kill:"GET /portal
kill:"GET /pv/
kill:"GET /script[ ]
kill:"GET /setup\.cgi
kill:"GET /shell
kill:"GET /showLogin
kill:"GET /solr
kill:"GET /sql/
kill:"GET /static/cs/ntjiheng.png
kill:"GET /vendor/
```

```
kill:"GET /video/26176
kill:"GET /w00tw00t\.
kill:"GET /webfig
kill:"GET /wiki
kill:"GET /wordpress/
kill:"GET /wp-content/plugins/ninja-forms/
kill:"GET http:
kill:"GET /stalker_portal/
kill:"GET /stream/
kill:"GET /streaming/
kill:"GET /system_api\.php[ ]
kill:"GET /c/version\.js[ ]
kill:"GET /adminer
kill:"GET /jenkins/
kill:"GET .*sslvpn_websession[ ]
kill:"GET /\?%3Cplay%3Ewithme%3C/%3E[ ]
kill:"GET /__media__/js/netsoltrademark.php\?
kill:"GET /wp-admin/admin-ajax.php?action=revslider_show_image&img=../index
kill:"GET /wp-admin/admin-ajax.php?action=revslider_show_image&img=../wp-config
kill:"GET /wp-admin/admin-post.php?alg_wc_pif_download_file=../../../../../index
kill:"GET /wp-admin/admin-post.php?alg_wc_pif_download_file=../../../../../wp-config
kill:"GET /wp-content/plugins/instabuilder2/
kill:"GET /tool.php
kill:"GET /wp-content/cache.php
kill:"GET /wp-includes/js/plupload/plupload.php
kill:"GET /authorize.php
kill:"GET /wp-content/themes/classic/inc/index.php
kill:"GET /assets/global/plugins/jquery-file-upload/server/php/files/jquery.php
kill:"GET /wp-content/themes/twentythree/inc/index.php
kill:"GET /wp-content/plugins/core-engine/index.php
kill:"HEAD /y0dnie/
kill:"Gh0st
kill:"MGLNDD
kill:"POST //
```

```
kill:"POST /FD873AC4-CF86-4FED-84EC-4BD59C6F17A7
kill:"POST /HNAP1
kill:"POST /[ ]
kill:"POST /_ignition/
kill:"POST /admin
kill:"POST /ajax
kill:"POST /blog/xmlrpc
kill:"POST /boaform
kill:"POST /cgi-bin/index2.asp[ ]
kill:"POST /credentials
kill:"POST /editBlackAndWhiteList
kill:"POST /index
kill:"POST /mgmt/
kill:"POST /vendor/
kill:"SSTP_DUPLEX_POST
kill:"\\n
kill:"\\x

# There is a list of bots at Cloudflare that they
# consider relatively well-behaved. Usefully, these are
# classified into categories, which makes it easy to
# identify and block SEO bots such as Ahrefs.
#       https://radar.cloudflare.com/verified-bots
# And there is this list of "bad bots":
#       https://github.com/sminozzi/stopbadbots/blob/master/functions/aBots.php

# Unnotable transfers -- check bl.de recommendation.
blde:"GET / HTTP/1\.[01]"
blde:"GET /.well-known/security.txt HTTP/1.1"
        # RFC8615 (.well-known), RFC9116 (security.txt)
blde:"GET /ads\.txt HTTP/1\.1"
blde:"GET /favicon\.ico HTTP/1\.1"
blde:"GET /humans\.txt HTTP/1\.1"
blde:"GET /img/favicon\.ico HTTP/1\.1"
```

```
                        blde:"GET /robots\.txt HTTP/1\.1"
                        blde:"GET /sitemap(?:\.xml|\.txt)? HTTP/1\.1"
                        blde:"HEAD / HTTP/1\.[01]"
                        blde:"HEAD /robots\.txt HTTP/1\.[01]"
                        blde:"OPTIONS / HTTP/1\.0"

                        # This actually turned out to be a good probe canary so
                        # left it out.
                        #pass:"GET /PhpGedView

                        # The signature when we close connections that opened
                        # but made no timely request, and when we close cached
                        # connections that have stopped making requests. We
                        # don't do a BLDE check here because we probably
                        # already have and we're a good citizen.
                        pass:"-" 408 -

                        # We don't catchall here. We want to fall through to virtual-host-specific filtering.

                done

        done

done

# Whitelist the page that polls ntp.png.
pass:metaed\.com ($re_ip) $re_hname - - \[[0-9/A-Za-z: +-]+\] "GET /ntp\.png

# Whitelist WordPress on 13moonsgrove.com.
with:13moonsgrove\.com[ ]

        # Parse the client IP, client hostname, client identity, and
        # timestamp.
        with:($re_ip) $re_hname - - \[[0-9/A-Za-z: +-]+\][ ]
```

```
                # (Conditionally) pass WordPress.
                blde:"GET /wp-admin
                blde:"GET /wp-json
                blde:"GET /wp-login
                blde:"GET /xmlrpc
                blde:"POST /xmlrpc

        done

        # From local host only, allow WP "cron" request.
        with:($re_ip) newjersey\.metaed\.com - - \[[0-9/A-Za-z: +-]+\][ ]

                pass:"POST /wp-cron\.php[ ?]

        done

done

# And now a filter that kills WordPress on sites that we did not
# whitelist above.
with:$re_hname[ ]

        # Parse the client IP, client hostname, client identity, and
        # timestamp.
        with:($re_ip) $re_hname - - \[[0-9/A-Za-z: +-]+\][ ]

                # Kill WordPress.
                kill:"GET /wp-admin
                kill:"GET /wp-json
                kill:"GET .*?wp-login
                kill:"GET /xmlrpc
                kill:"POST /xmlrpc
                kill:"POST /wp-cron\.php[ ?]
                done

        done

done

##########
# HTTPD ERROR LOG

with:httpd$re_pid:[ ]

        # Block "origin confusion" also called "virtual host confusion" attack.
        kill:\[ssl:error\] \[pid \d+:tid \d+\] \[client ($re_ip):\d+\] AH02032: Hostname $re_hname provided via SNI and hostname $re_

        # Block SSL/TLS "renegotiation vulnerability" attack.
        kill:\[ssl:error\] \[pid \d+:tid \d+\] \[client ($re_ipv4):\d+\] AH02042: rejecting client initiated renegotiation$

        # A start on PHP errors. End user gets a 404. Most of these are brute
        # force hack attempts but can’t tell those from just a broken page so
        # get bl.de recommendation.
        btyp:\[php7:error\] \[pid \d+:tid \d+\] \[client ($re_ip):\d+\] script ’.*’ not found or unable to stat(?:, referer: )?

        # This is clearly a probe.
        kill:\[php7:error\] \[pid \d+:tid \d+\] \[client ($re_ip):\d+\] script ’/srv/www/archweaver.com/root/xmlrpc.php’ not found or

        # Trying to access a forbidden path.
        kill:\[core:error\] \[pid \d+:tid \d+\] \[client($re_ip):\d+\] AH10244: invalid URI path[ ]

done

##########
# AUTH
```

```
with:auth:[ ]

        # Pass good email addresses. Look for old compromised email addresses
        # and block them.
        type:pam_unix\(dovecot:auth\): authentication failure; logname= uid=0 euid=0 tty=dovecot ruser=root rhost=($re_ip)$
        type:pam_unix\(dovecot:auth\): authentication failure; logname= uid=0 euid=0 tty=dovecot ruser=metaed rhost=($re_ip)$
        #kill:pam_unix\(dovecot:auth\): authentication failure; logname= uid=0 euid=0 tty=dovecot ruser=(?:adm|admin|www|web|sys|smtp
        kill:pam_unix\(dovecot:auth\): authentication failure; logname= uid=0 euid=0 tty=dovecot ruser=(?:adm|admin|www|web|sys|smtp|

        # Ignore these messages associated with failed login attempts. Other
        # associated log messages handled elsewhere will deal with the
        # intrusion attempt.
        pass:pam_unix\(dovecot:auth\): check pass; user unknown$
        pass:gkr-pam: error looking up user information$

done

##########
# SYSLOGD - ignore mark messages and buffered messages. Report everything else.

with:syslogd$re_pid:[ ]
        pass:-- MARK --$
        pass:syslogd v[0-9.]+(?:-pre)?: restart\.$
        pass:exiting on signal 15$
        type:
done
pass:last message buffered \d+ times$



##########
# SU - report anything unknown.

with:su$re_pid:[ ]
```

```
        pass:Successful su for metaed by root$
        pass:\+ /dev/pts/[0-9]+ root:metaed$
        pass:pam_unix\(su:session\): session closed for user metaed$
        pass:pam_unix\(su:session\): session opened for user metaed\(uid=1000\) by root\(uid=0\)$
        type:
done



##########
# SUDO - report new sessions and anything unknown.

with:sudo:[ ]
        pass:pam_unix\(sudo:session\): session opened for user metaed\(uid=1000\) by \(uid=0\)$
        pass:pam_unix\(sudo:session\): session closed for user (?:root|metaed)$
        pass:    root : PWD=/root ; USER=metaed ; GROUP=metaed ; COMMAND=/usr/bin/dehydrated -c$
        type:
done



##########
# ELOGIND-DAEMON - report new sessions and anything unknown.

with:elogind-daemon$re_pid:[ ][ ]?
        pass:New seat seat0\.$
        pass:New session \d+ of user root\.$
        pass:Removed session \d+\.$
        pass:Watching system buttons on /dev/input/event0 \(AT Translated Set 2 keyboard\)$
        pass:Watching system buttons on /dev/input/event1 \(Power Button\)$
        type:
done

##########
```

```
# NTPD - report anything unknown.

with:ntpd$re_pid:[ ]

        # Default logging.
        pass:Soliciting pool server ($re_ip)$
        pass:($re_ip) local addr $re_ip -> <null>$
        pass:Listen (?:normally|and drop) on [0-9]+ \S+ \[?$re_ip(?:%2)?\]?:123$
        pass:Listening on routing socket on fd #[0-9]+ for interface updates$
        pass:kernel reports TIME_ERROR: 0x[0-9a-f]+: Clock Unsynchronized$
        pass:ntpd exiting on signal 1 \(Hangup\)$
        pass:ntpd .+: Starting$
        pass:Command line: /usr/sbin/ntpd -g -u ntp:ntp$
        pass:ntp-4 is maintained by Network Time Foundation,$
        pass:Inc. \(NTF\), a non-profit 501\(c\)\(3\) public-benefit$
        pass:corporation.  Support and training for ntp-4 are$
        pass:available at https://www\.nwtime\.org/support$
        pass:----------------------------------------------------$
        pass:proto: precision = [0-9.]+ usec \(-[0-9]+\)$
        pass:basedate set to [0-9-]+$
        pass:gps base set to [0-9-]+ \(week [0-9]+\)$
#       pass:receive: Unexpected origin timestamp 0x[0-9a-f]+\.[0-9a-f]+ does not match aorg 0x[0-9a-f]+\.[0-9a-f]+ from server\@($re
        pass:receive: Unexpected origin timestamp 0x[0-9a-f]+\.[0-9a-f]+
#/var/log/messages (catchall) Aug 18 11:32:06 newjersey ntpd[1438]: receive: Unexpected origin timestamp 0xe6a8e805.50b2c1fa does not
        pass:::1 config:
        pass:unpeered ($re_ip)$

        # =allall logging.
        pass:($re_ip) [0-9a-f]+ [0-9a-f]+ (?:(?:de)?mobilize assoc [0-9]+|clock_sync|clock_step [-+][0-9.]+ s|freq_set kernel [0-9.]+

        # Catchall.
        type:
done
```

```
##########
# MYSQLD (MARIADB) - SQL database service. Pass startup and shutdown messages. Log anything unknown.

pass:mysqld:$
with:mysqld:[ ]


        pass:Version: ’[0-9.]+-MariaDB’  socket: ’/var/run/mysql/mysql\.sock’  port: [0-9+]  Source distribution$

        with:[0-9-]+ [0-9:]+ 0 \[Note\][ ]
                pass:/usr/libexec/mariadbd \(initiated by: unknown\): Normal shutdown$
                pass:/usr/libexec/mariadbd \(mysqld [0-9.]+-MariaDB\) starting as process [0-9]+ \.\.\.$
                pass:/usr/libexec/mariadbd: ready for connections\.$
                pass:/usr/libexec/mariadbd: Shutdown complete$
                pass:Added new Master_info ’’ to hash table$
                pass:Event Scheduler: Purging the queue. [0-9]+ events$
                pass:InnoDB: Buffer pool\(s\) dump completed at [0-9]+ [0-9:]+$
                pass:InnoDB: Buffer pool\(s\) load completed at [0-9]+ [0-9:]+$
                pass:InnoDB: Completed initialization of buffer pool$
                pass:InnoDB: Compressed tables use zlib [0-9]+\.[0-9]+\.[0-9]+$
                pass:InnoDB: Creating shared tablespace for temporary tables$
                pass:InnoDB: Dumping buffer pool\(s\) to /var/lib/mysql/ib_buffer_pool$
                pass:InnoDB: File ’\./ibtmp1’ size is now [0-9]+ MB\.$
                pass:InnoDB: FTS optimize thread exiting\.$
                pass:InnoDB: Initializing buffer pool, total size = [0-9]+, chunk size = [0-9]+$
                pass:InnoDB: Loading buffer pool\(s\) from /var/lib/mysql/ib_buffer_pool$
                pass:InnoDB: Number of pools: [0-9]+$
                pass:InnoDB: Removed temporary tablespace data file: "ibtmp1"$
                pass:InnoDB: Setting file ’\./ibtmp1’ size to [0-9]+ MB. Physically writing the file full; Please wait \.\.\.$
                pass:InnoDB: Shutdown completed; log sequence number [0-9]+; transaction id [0-9]+$
                pass:InnoDB: Starting shutdown\.\.\.$
                pass:InnoDB: Uses event mutexes$
                pass:InnoDB: Using crc32 \+ pclmulqdq instructions$
                pass:InnoDB: Using Linux native AIO$
                pass:InnoDB: [0-9.]+ started; log sequence number [0-9]+; transaction id [0-9]+$
                pass:InnoDB: [0-9]+ rollback segments are active\.$
                pass:Plugin ’FEEDBACK’ is disabled\.$
                pass:Reading of all Master_info entries succeeded$
                type:
        done

        with:[0-9-]+ [0-9:]+ 0 \[Warning\][ ]
                type:Access denied for user ’
                        # This looks like an intrusion attempt, type it.
        done

        type:

done

with:mysqld_safe:[ ]
        pass:mysqld from pid file /var/run/mysql/mysql.pid ended$
        pass:Starting mariadbd daemon with databases from /var/lib/mysql$
        type:
done



##########
# KERNEL - messages passed from kernel to syslogd.

# For now, pass everything. We don’t expect to see anything actionable here.
pass:kernel:[ ]
#with:kernel:[ ]
#       pass:  Device   empty$
#       pass:  DMA      \[mem 0x0000000000001000-0x0000000000ffffff\]$
#       pass:  DMA32    \[mem 0x0000000001000000-0x000000003ffdbfff\]$
#       pass:  node   0: \[mem 0x0000000000001000-0x000000000009efff\]$
#       pass:  node   0: \[mem 0x0000000000100000-0x000000003ffdbfff\]$
#       pass:  Normal   empty$
#       pass:00:03: ttyS0 at I/O 0x3f8 \(irq = 4, base_baud = 115200\) is a 16550A$
#       pass:3ware (?:9000 )?Storage Controller device driver for Linux v[0-9.]+\.$
#       pass:9p: Installing v9fs 9p2000 file system support$
#       pass:9pnet: Installing 9P2000 support$
#       pass:acpi PNP0A08:00: _OSC: OS now controls \[PCIeHotplug PME AER PCIeCapability\]$
#       pass:acpi PNP0A08:00: _OSC: OS supports \[ExtendedConfig ASPM ClockPM Segments MSI EDR HPX-Type3\]$
#       pass:acpi PNP0A08:00: _OSC: platform does not support \[LTR DPC\]$
#       pass:ACPI: 1 ACPI AML tables successfully acquired and loaded$
#       pass:ACPI: Added _OSI\(3\.0 _SCP Extensions\)$
#       pass:ACPI: Added _OSI\(Linux-Dell-Video\)$
#       pass:ACPI: Added _OSI\(Linux-HPI-Hybrid-Graphics\)$
#       pass:ACPI: Added _OSI\(Linux-Lenovo-NV-HDMI-Audio\)$
#       pass:ACPI: Added _OSI\(Module Device\)$
#       pass:ACPI: Added _OSI\(Processor Aggregator Device\)$
#       pass:ACPI: Added _OSI\(Processor Device\)$
#       pass:ACPI: APIC 0x000000003FFE1FEE 000078 \(v01 BOCHS  BXPCAPIC 00000001 BXPC 00000001\)$
#       pass:ACPI: bus type PCI registered$
#       pass:ACPI: bus type USB registered$
#       pass:ACPI: button: Power Button \[PWRF\]$
#       pass:ACPI: Core revision 20210730$
#       pass:ACPI: DSDT 0x000000003FFE0040 001EBA \(v01 BOCHS  BXPCDSDT 00000001 BXPC 00000001\)$
#       pass:ACPI: Early table checksum verification disabled$
#       pass:ACPI: Enabled 1 GPEs in block 00 to 3F$
#       pass:ACPI: FACP 0x000000003FFE1EFA 0000F4 \(v03 BOCHS  BXPCFACP 00000001 BXPC 00000001\)$
#       pass:ACPI: FACS 0x000000003FFE0000 000040$
#       pass:ACPI: HPET 0x000000003FFE2066 000038 \(v01 BOCHS  BXPCHPET 00000001 BXPC 00000001\)$
#       pass:ACPI: HPET id: 0x8086a201 base: 0xfed00000$
#       pass:ACPI: Interpreter enabled$
#       pass:ACPI: INT_SRC_OVR \(bus 0 bus_irq 0 global_irq 2 dfl dfl\)$
#       pass:ACPI: INT_SRC_OVR \(bus 0 bus_irq 10 global_irq 10 high level\)$
#       pass:ACPI: INT_SRC_OVR \(bus 0 bus_irq 11 global_irq 11 high level\)$
#       pass:ACPI: INT_SRC_OVR \(bus 0 bus_irq 5 global_irq 5 high level\)$
#       pass:ACPI: INT_SRC_OVR \(bus 0 bus_irq 9 global_irq 9 high level\)$
#       pass:ACPI: LAPIC_NMI \(acpi_id\[0xff\] dfl dfl lint\[0x1\]\)$
#       pass:ACPI: MCFG 0x000000003FFE209E 00003C \(v01 BOCHS  BXPCMCFG 00000001 BXPC 00000001\)$
#       pass:ACPI: PCI Root Bridge \[PCI0\] \(domain 0000 \[bus 00-ff\]\)$
#       pass:ACPI: PCI: Interrupt link GSIA configured for IRQ 16$
#       pass:ACPI: PCI: Interrupt link GSIB configured for IRQ 17$
#       pass:ACPI: PCI: Interrupt link GSIC configured for IRQ 18$
#       pass:ACPI: PCI: Interrupt link GSID configured for IRQ 19$
#       pass:ACPI: PCI: Interrupt link GSIE configured for IRQ 20$
#       pass:ACPI: PCI: Interrupt link GSIF configured for IRQ 21$
#       pass:ACPI: PCI: Interrupt link GSIG configured for IRQ 22$
#       pass:ACPI: PCI: Interrupt link GSIH configured for IRQ 23$
#       pass:ACPI: PCI: Interrupt link LNKA configured for IRQ 10$
#       pass:ACPI: PCI: Interrupt link LNKB configured for IRQ 10$
#       pass:ACPI: PCI: Interrupt link LNKC configured for IRQ 11$
#       pass:ACPI: PCI: Interrupt link LNKD configured for IRQ 11$
#       pass:ACPI: PCI: Interrupt link LNKE configured for IRQ 10$
#       pass:ACPI: PCI: Interrupt link LNKF configured for IRQ 10$
#       pass:ACPI: PCI: Interrupt link LNKG configured for IRQ 11$
#       pass:ACPI: PCI: Interrupt link LNKH configured for IRQ 11$
#       pass:ACPI: PM-Timer IO Port: 0x608$
#       pass:ACPI: PM: \(supports S0 S3 S4 S5\)$
#       pass:ACPI: Reserving APIC table memory at \[mem 0x3ffe1fee-0x3ffe2065\]$
#       pass:ACPI: Reserving DSDT table memory at \[mem 0x3ffe0040-0x3ffe1ef9\]$
#       pass:ACPI: Reserving FACP table memory at \[mem 0x3ffe1efa-0x3ffe1fed\]$
#       pass:ACPI: Reserving FACS table memory at \[mem 0x3ffe0000-0x3ffe003f\]$
#       pass:ACPI: Reserving HPET table memory at \[mem 0x3ffe2066-0x3ffe209d\]$
#       pass:ACPI: Reserving MCFG table memory at \[mem 0x3ffe209e-0x3ffe20d9\]$
#       pass:ACPI: RSDP 0x00000000000F56D0 000014 \(v00 BOCHS \)$
#       pass:ACPI: RSDT 0x000000003FFE20DA 000034 \(v01 BOCHS  BXPCRSDT 00000001 BXPC 00000001\)$
#       pass:ACPI: Using ACPI \(MADT\) for SMP configuration information$
#       pass:ACPI: Using IOAPIC for interrupt routing$
#       pass:ACPI: \\x5c_SB_\.GSIA: Enabled at IRQ 16$
#       pass:ACPI: \\x5c_SB_\.GSIE: Enabled at IRQ 20$
#       pass:ACPI: \\x5c_SB_\.GSIG: Enabled at IRQ 22$
#       pass:ACPI: \\x5c_SB_\.GSIH: Enabled at IRQ 23$
#       pass:acpiphp: ACPI Hot Plug PCI Controller Driver version: 0\.5$
#       pass:Adaptec aacraid driver [0-9.]+\[[0-9]+\]-custom$
#       pass:Adding 524284k swap on /dev/sdb\.  Priority:-2 extents:1 across:524284k FS$
#       pass:AES CTR mode by8 optimization enabled$
#       pass:ahci 0000:00:1f\.2: AHCI 0001\.0000 32 slots 6 ports 1\.5 Gbps 0x3f impl SATA mode$
#       pass:ahci 0000:00:1f\.2: flags: 64bit ncq only$
#       pass:aic94xx: Adaptec aic94xx SAS/SATA driver version [0-9.]+ loaded$
#       pass:APIC: Switch to symmetric I/O mode setup$
#       pass:Asymmetric key parser 'x509' registered$
#       pass:async_tx: api initialized \(async\)$
#       pass:ata1: SATA link down \(SStatus 0 SControl 300\)$
#       pass:ata1: SATA max UDMA/133 abar m4096\@0xfebd4000 port 0xfebd4100 irq 32$
#       pass:ata2: SATA link down \(SStatus 0 SControl 300\)$
#       pass:ata2: SATA max UDMA/133 abar m4096\@0xfebd4000 port 0xfebd4180 irq 32$
#       pass:ata3: SATA link down \(SStatus 0 SControl 300\)$
#       pass:ata3: SATA max UDMA/133 abar m4096\@0xfebd4000 port 0xfebd4200 irq 32$
#       pass:ata4: SATA link down \(SStatus 0 SControl 300\)$
#       pass:ata4: SATA max UDMA/133 abar m4096\@0xfebd4000 port 0xfebd4280 irq 32$
#       pass:ata5: SATA link down \(SStatus 0 SControl 300\)$
#       pass:ata5: SATA max UDMA/133 abar m4096\@0xfebd4000 port 0xfebd4300 irq 32$
#       pass:ata6: SATA link down \(SStatus 0 SControl 300\)$
#       pass:ata6: SATA max UDMA/133 abar m4096\@0xfebd4000 port 0xfebd4380 irq 32$
#       pass:ata7: PATA max PIO4 cmd 0x1f0 ctl 0x3f6 irq 14$
#       pass:ata8: PATA max PIO4 cmd 0x170 ctl 0x376 irq 15$
#       pass:atomic64_test: passed for x86-64 platform with CX8 and with SSE$
#       pass:audit: initializing netlink subsys \(disabled\)$
#       pass:audit: type=2000 audit\(1660840103.628:1\): state=initialized audit_enabled=0 res=1$
#       pass:AVX2 version of gcm_enc/dec engaged\.$
#       pass:BIOS-e820: \[mem 0x0000000000000000-0x000000000009fbff\] usable$
#       pass:BIOS-e820: \[mem 0x000000000009fc00-0x000000000009ffff\] reserved$
#       pass:BIOS-e820: \[mem 0x00000000000f0000-0x00000000000fffff\] reserved$
#       pass:BIOS-e820: \[mem 0x0000000000100000-0x000000003ffdbfff\] usable$
#       pass:BIOS-e820: \[mem 0x000000003ffdc000-0x000000003fffffff\] reserved$
#       pass:BIOS-e820: \[mem 0x00000000b0000000-0x00000000bfffffff\] reserved$
#       pass:BIOS-e820: \[mem 0x00000000fed1c000-0x00000000fed1ffff\] reserved$
#       pass:BIOS-e820: \[mem 0x00000000feffc000-0x00000000feffffff\] reserved$
#       pass:BIOS-e820: \[mem 0x00000000fffc0000-0x00000000ffffffff\] reserved$
#       pass:BIOS-provided physical RAM map:$
#       pass:Block layer SCSI generic \(bsg\) driver version 0\.4 loaded \(major 244\)$
#       pass:bochs-drm 0000:00:01\.0: vgaarb: deactivate vga console$
#       pass:bochs-drm 0000:00:01\.0: \[drm\] fb0: bochs-drmdrmfb frame buffer device$
#       pass:Booting paravirtualized kernel on KVM$
#       pass:brd: module loaded$
#       pass:Btrfs loaded, crc32c=crc32c-intel, zoned=yes, fsverity=no$
#       pass:Built 1 zonelists, mobility grouping on\.  Total pages: 257756$
#       pass:Calibrating delay loop \(skipped\) preset value\.\. 3999\.99 BogoMIPS \(lpj=1999995\)$
#       pass:clocksource: acpi_pm: mask: 0xffffff max_cycles: 0xffffff, max_idle_ns: 2085701024 ns$
#       pass:clocksource: hpet: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 19112604467 ns$
#       pass:clocksource: jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 1911260446275000 ns$
#       pass:clocksource: kvm-clock: mask: 0xffffffffffffffff max_cycles: 0x1cd42e4dffb, max_idle_ns: 881590591483 ns$
#       pass:clocksource: refined-jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 1910969940391419 ns$
#       pass:clocksource: Switched to clocksource kvm-clock$
#       pass:clocksource: tsc-early: mask: 0xffffffffffffffff max_cycles: 0x39a853ae795, max_idle_ns: 881590634035 ns$
#       pass:clocksource: tsc: mask: 0xffffffffffffffff max_cycles: 0x39a853ae795, max_idle_ns: 881590634035 ns$
#       pass:Command line: BOOT_IMAGE=/boot/vmlinuz-huge-5\.15\.38 root=/dev/sda ro console=ttyS0,19200n8 net\.ifnames=0$
#       pass:Console: colour VGA\+ 80x25$
#       pass:Console: switching to colour dummy device 80x25$
#       pass:Console: switching to colour frame buffer device 128x48$
#       pass:Copyright \(c\) 1999-2008 LSI Corporation$
#       pass:cpuidle: using governor ladder$
#       pass:cpuidle: using governor menu$
#       pass:cryptd: max_cpu_qlen set to 1000$
#       pass:Dentry cache hash table entries: 131072 \(order: 8, 1048576 bytes, linear\)$
#       pass:Detecting Adaptec I2O RAID controllers\.\.\.$
#       pass:device-mapper: ioctl: [0-9.]+-ioctl \([0-9-]+\) initialised: dm-devel\@redhat\.com$
#       pass:device-mapper: uevent: version 1\.0\.3$
#       pass:devtmpfs: initialized$
#       pass:devtmpfs: mounted$
#       pass:DMA: preallocated 128 KiB GFP_KERNEL pool for atomic allocations$
#       pass:DMA: preallocated 128 KiB GFP_KERNEL\|GFP_DMA pool for atomic allocations$
#       pass:DMA: preallocated 128 KiB GFP_KERNEL\|GFP_DMA32 pool for atomic allocations$
#       pass:DMI: QEMU Standard PC \(Q35 \+ ICH9, 2009\), BIOS rel-1\.12\.0-0-ga698c8995f-prebuilt\.qemu\.org 04/01/2014$
#       pass:Early memory node ranges$
#       pass:EDAC MC: Ver: [0-9.]+$
#       pass:Error: Driver ’virtio_scsi’ is already registered, aborting\.\.\.$
#       pass:EXT4-fs \(sda\): mounted filesystem with ordered data mode\. Opts: \(null\)\. Quota mode: none\.$
#       pass:EXT4-fs \(sda\): re-mounted\. Opts: \(null\)\. Quota mode: none\.$
#       pass:failover: exports duplicate symbol failover_register \(owned by kernel\)$
#       pass:Faking a node at \[mem 0x0000000000000000-0x000000003ffdbfff\]$
#       pass:fbcon: bochs-drmdrmfb \(fb0\) is primary device$
#       pass:floppy0: floppy_shutdown: timeout handler died\.$
#       pass:floppy0: no floppy controllers found$
#       pass:fnic: Cisco FCoE HBA Driver, ver [0-9.]+$
#       pass:fnic: Successfully Initialized FC_CTLR Trace Buffer$
#       pass:fnic: Successfully Initialized Trace Buffer$
#       pass:found SMP MP-table at \[mem 0x000f58b0-0x000f58bf\]$
#       pass:Freeing SMP alternatives memory: 60K$
#       pass:Freeing unused decrypted memory: 2036K$
#       pass:Freeing unused kernel image \(initmem\) memory: 1940K$
#       pass:Freeing unused kernel image \(rodata/data gap\) memory: 1948K$
#       pass:Freeing unused kernel image \(text/rodata gap\) memory: 2024K$
#       pass:ftrace: allocated 230 pages with 5 groups$
#       pass:ftrace: allocating 58880 entries in 231 pages$
#       pass:Fusion MPT base driver [0-9.]+$
#       pass:Fusion MPT FC Host driver [0-9.]+$
#       pass:Fusion MPT LAN driver [0-9.]+$
#       pass:Fusion MPT misc device \(ioctl\) driver [0-9.]+$
#       pass:Fusion MPT SAS Host driver [0-9.]+$
#       pass:Fusion MPT SPI Host driver [0-9.]+$
#       pass:futex hash table entries: 256 \(order: 2, 16384 bytes, linear\)$
#       pass:hpet0: 3 comparators, 64-bit 100.000000 MHz counter$
#       pass:hpet0: at MMIO 0xfed00000, IRQs 2, 8, 0$
#       pass:HugeTLB registered 1\.00 GiB page size, pre-allocated 0 pages$
#       pass:HugeTLB registered 2\.00 MiB page size, pre-allocated 0 pages$
#       pass:Hypervisor detected: KVM$
#       pass:i2c i2c-0: 1/1 memory slots populated \(from DMI\)$
#       pass:i2c i2c-0: Memory type 0x07 not supported yet, not instantiating SPD$
#       pass:i801_smbus 0000:00:1f\.3: SMBus using PCI interrupt$
#       pass:i8042: PNP: PS/2 Controller \[PNP0303:KBD,PNP0f13:MOU\] at 0x60,0x64 irq 1,12$
#       pass:In-situ OAM \(IOAM\) with IPv6$
#       pass:Initialise system trusted keyrings$
#       pass:Initializing XFRM netlink socket$
#       pass:Initmem setup node 0 \[mem 0x0000000000001000-0x000000003ffdbfff\]$
#       pass:Inode-cache hash table entries: 65536 \(order: 7, 524288 bytes, linear\)$
#       pass:input: AT Translated Set 2 keyboard as /devices/platform/i8042/serio0/input/input0$
#       pass:input: Power Button as /devices/LNXSYSTM:00/LNXPWRBN:00/input/input2$
#       pass:input: VirtualPS/2 VMware VMMouse as /devices/platform/i8042/serio1/input/input3$
#       pass:input: VirtualPS/2 VMware VMMouse as /devices/platform/i8042/serio1/input/input4$
#       pass:Installing knfsd \(copyright \(C\) 1996 okir\@monad\.swb\.de\)\.$
#       pass:io scheduler mq-deadline registered$
#       pass:IOAPIC\[0\]: apic_id 0, version 17, address 0xfec00000, GSI 0-23$
#       pass:iommu: Default domain type: Translated$
#       pass:iommu: DMA domain TLB invalidation policy: lazy mode$
#       pass:IP idents hash table entries: 16384 \(order: 5, 131072 bytes, linear\)$
#       pass:IPI shorthand broadcast: enabled$
#       pass:ipr: IBM Power RAID SCSI Device Driver version: [0-9.]+ \(\S+ [0-9]+, [0-9]+\)$
#       pass:isci: Intel\(R\) C600 SAS Controller Driver - version [0-9.]+$
#       pass:JFS: nTxBlock = 7672, nTxLock = 61380$
#       pass:Kernel command line: BOOT_IMAGE=/boot/vmlinuz-huge-5\.15\.38 root=/dev/sda ro console=ttyS0,19200n8 net\.ifnames=0$
#       pass:Key type asymmetric registered$
#       pass:Key type blacklist registered$
#       pass:Key type dns_resolver registered$
#       pass:Key type encrypted registered$
#       pass:Key type fscrypt-provisioning registered$
#       pass:Key type id_legacy registered$
#       pass:Key type id_resolver registered$
#       pass:Key type \.fscrypt registered$
#       pass:Key type \._fscrypt registered$
#       pass:kvm-clock: cpu 0, msr 35c01001, primary cpu clock$
#       pass:kvm-clock: Using msrs 4b564d01 and 4b564d00$
#       pass:kvm-clock: using sched offset of 13619713807 cycles$
#       pass:kvm-guest: stealtime: cpu 0, msr 3ec31080$
#       pass:Last level dTLB entries: 4KB 512, 2MB 255, 4MB 127, 1GB 0$
#       pass:Last level iTLB entries: 4KB 512, 2MB 255, 4MB 127$
#       pass:last_pfn = 0x3ffdc max_arch_pfn = 0x400000000$
#       pass:Linux agpgart interface v0\.103$
#       pass:Linux version [0-9.]+ \(root\@z-mp\.slackware\.lan\) \(gcc \(GCC\) 11\.2\.0, GNU ld version 2\.37-slack15\) #1 SMP PREEM
#       pass:Loading Adaptec I2O RAID: Version 2\.4 Build 5go$
#       pass:Loading compiled-in X\.509 certificates$
#       pass:loop: module loaded$
#       pass:lpc_ich 0000:00:1f\.0: I/O space for GPIO uninitialized$
#       pass:LSI 3ware SAS/SATA-RAID Controller device driver for Linux v[0-9.]+\.$
#       pass:LSM: Security Framework initializing$
#       pass:md: Autodetecting RAID arrays\.$
#       pass:md: autorun \.\.\.$
#       pass:md: If you don’t use raid, use raid=noautodetect$
#       pass:md: Waiting for all devices to be available before autodetect$
#       pass:md: \.\.\. autorun DONE\.$
#       pass:megaraid cmm: [0-9.]+ \(Release Date: .*\)$
#       pass:megaraid: [0-9.]+ \(Release Date: .*\)$
#       pass:megasas: [0-9.]+-rc1$
#       pass:mem auto-init: stack:off, heap alloc:off, heap free:off$
#       pass:Memory: 981768K/1048040K available \(20501K kernel code, 3320K rwdata, 6244K rodata, 1940K init, 4728K bss, 66012K reser
#       pass:Mount-cache hash table entries: 2048 \(order: 2, 16384 bytes, linear\)$
#       pass:Mountpoint-cache hash table entries: 2048 \(order: 2, 16384 bytes, linear\)$
#       pass:mousedev: PS/2 mouse device common for all mice$
#       pass:Movable zone start for each node$
#       pass:mpt3sas version [0-9.]+ loaded$
#       pass:MPTCP token hash table entries: 1024 \(order: 2, 24576 bytes, linear\)$
#       pass:mptctl: /dev/mptctl \@ \(major,minor=10,220\)$
#       pass:mptctl: Registered with Fusion MPT base driver$
#       pass:NET: Registered PF_INET protocol family$
#       pass:NET: Registered PF_INET6 protocol family$
#       pass:NET: Registered PF_NETLINK/PF_ROUTE protocol family$
#       pass:NET: Registered PF_PACKET protocol family$
#       pass:NET: Registered PF_UNIX/PF_LOCAL protocol family$
#       pass:NET: Registered PF_XDP protocol family$
#       pass:nfs4filelayout_init: NFSv4 File Layout Driver Registering\.\.\.$
#       pass:nfs4flexfilelayout_init: NFSv4 Flexfile Layout Driver Registering\.\.\.$
#       pass:NFS: Registering the id_resolver key type$
#       pass:No NUMA configuration found$
#       pass:NODE_DATA(0) allocated [mem 0x3ffd7000-0x3ffdbfff]$
#       pass:NODE_DATA\(0\) allocated \[mem 0x3ffd7000-0x3ffdbfff\]$
#       pass:NR_IRQS: 16640, nr_irqs: 256, preallocated irqs: 16$
#       pass:ntfs: driver [0-9.]+ \[Flags: R/W\]\.$
#       pass:NX \(Execute Disable\) protection: active$
#       pass:OCFS2 User DLM kernel interface loaded$
#       pass:ocfs2: Registered cluster interface o2cb$
#       pass:On node 0, zone DMA32: 36 pages in unavailable ranges$
#       pass:On node 0, zone DMA: 1 pages in unavailable ranges$
#       pass:On node 0, zone DMA: 97 pages in unavailable ranges$
#       pass:pci 0000:00:00\.0: \[8086:29c0\] type 00 class 0x060000$
#       pass:pci 0000:00:01\.0: reg 0x10: \[mem 0xfd000000-0xfdffffff pref\]$
#       pass:pci 0000:00:01\.0: reg 0x18: \[mem 0xfebd0000-0xfebd0fff\]$
#       pass:pci 0000:00:01\.0: reg 0x30: \[mem 0xfebc0000-0xfebcffff pref\]$
#       pass:pci 0000:00:01\.0: vgaarb: bridge control possible$
#       pass:pci 0000:00:01\.0: vgaarb: setting as boot VGA device$
#       pass:pci 0000:00:01\.0: vgaarb: VGA device added: decodes=io\+mem,owns=io\+mem,locks=none$
#       pass:pci 0000:00:01\.0: Video device with shadowed ROM at \[mem 0x000c0000-0x000dffff\]$
#       pass:pci 0000:00:01\.0: \[1234:1111\] type 00 class 0x030000$
#       pass:pci 0000:00:02\.0: reg 0x10: \[io  0xc000-0xc03f\]$
#       pass:pci 0000:00:02\.0: reg 0x14: \[mem 0xfebd1000-0xfebd1fff\]$
#       pass:pci 0000:00:02\.0: reg 0x20: \[mem 0xfe000000-0xfe003fff 64bit pref\]$
#       pass:pci 0000:00:02\.0: \[1af4:1004\] type 00 class 0x010000$
#       pass:pci 0000:00:03\.0: reg 0x10: \[io  0xc040-0xc07f\]$
#       pass:pci 0000:00:03\.0: reg 0x14: \[mem 0xfebd2000-0xfebd2fff\]$
#       pass:pci 0000:00:03\.0: reg 0x20: \[mem 0xfe004000-0xfe007fff 64bit pref\]$
#       pass:pci 0000:00:03\.0: \[1af4:1004\] type 00 class 0x010000$
#       pass:pci 0000:00:04\.0: reg 0x10: \[io  0xc080-0xc0bf\]$
#       pass:pci 0000:00:04\.0: reg 0x14: \[mem 0xfebd3000-0xfebd3fff\]$
#       pass:pci 0000:00:04\.0: reg 0x20: \[mem 0xfe008000-0xfe00bfff 64bit pref\]$
#       pass:pci 0000:00:04\.0: reg 0x30: \[mem 0xfeb80000-0xfebbffff pref\]$
#       pass:pci 0000:00:04\.0: \[1af4:1000\] type 00 class 0x020000$
#       pass:pci 0000:00:1f\.0: quirk: \[io  0x0600-0x067f\] claimed by ICH6 ACPI/GPIO/TCO$
#       pass:pci 0000:00:1f\.0: \[8086:2918\] type 00 class 0x060100$
#       pass:pci 0000:00:1f\.2: reg 0x20: \[io  0xc100-0xc11f\]$
#       pass:pci 0000:00:1f\.2: reg 0x24: \[mem 0xfebd4000-0xfebd4fff\]$
#       pass:pci 0000:00:1f\.2: \[8086:2922\] type 00 class 0x010601$
#       pass:pci 0000:00:1f\.3: reg 0x20: \[io  0x0700-0x073f\]$
#       pass:pci 0000:00:1f\.3: \[8086:2930\] type 00 class 0x0c0500$
#       pass:PCI host bridge to bus 0000:00$
#       pass:PCI: CLS 0 bytes, default 64$
#       pass:PCI: MMCONFIG at \[mem 0xb0000000-0xbfffffff\] reserved in E820$
#       pass:PCI: MMCONFIG for domain 0000 \[bus 00-ff\] at \[mem 0xb0000000-0xbfffffff\] \(base 0xb0000000\)$
#       pass:PCI: Using ACPI for IRQ routing$
#       pass:PCI: Using configuration type 1 for base access$
#       pass:PCI: Using host bridge windows from ACPI; if necessary, use "pci=nocrs" and report a bug$
#       pass:pci_bus 0000:00: resource 4 \[io  0x0000-0x0cf7 window\]$
#       pass:pci_bus 0000:00: resource 5 \[io  0x0d00-0xffff window\]$
#       pass:pci_bus 0000:00: resource 6 \[mem 0x000a0000-0x000bffff window\]$
#       pass:pci_bus 0000:00: resource 7 \[mem 0xc0000000-0xfebfffff window\]$
#       pass:pci_bus 0000:00: resource 8 \[mem 0x100000000-0x8ffffffff window\]$
#       pass:pci_bus 0000:00: root bus resource \[bus 00-ff\]$
#       pass:pci_bus 0000:00: root bus resource \[io  0x0000-0x0cf7 window\]$
#       pass:pci_bus 0000:00: root bus resource \[io  0x0d00-0xffff window\]$
#       pass:pci_bus 0000:00: root bus resource \[mem 0x000a0000-0x000bffff window\]$
#       pass:pci_bus 0000:00: root bus resource \[mem 0x100000000-0x8ffffffff window\]$
#       pass:pci_bus 0000:00: root bus resource \[mem 0xc0000000-0xfebfffff window\]$
#       pass:percpu: Embedded 59 pages/cpu s204800 r8192 d28672 u2097152$
#       pass:Performance Events: Fam17h\+ core perfctr, AMD PMU driver\.$
#       pass:pid_max: default: 32768 minimum: 301$
#       pass:pinctrl core: initialized pinctrl subsystem$
#       pass:PM: hibernation: Registered nosave memory: \[mem 0x00000000-0x00000fff\]$
#       pass:PM: hibernation: Registered nosave memory: \[mem 0x0009f000-0x0009ffff\]$
#       pass:PM: hibernation: Registered nosave memory: \[mem 0x000a0000-0x000effff\]$
#       pass:PM: hibernation: Registered nosave memory: \[mem 0x000f0000-0x000fffff\]$
#       pass:pnp: PnP ACPI init$
#       pass:pnp: PnP ACPI: found 4 devices$
#       pass:Policy zone: DMA32$
#       pass:pps_core: LinuxPPS API ver\. 1 registered$
#       pass:pps_core: Software ver\. [0-9.]+ - Copyright 2005-2007 Rodolfo Giometti <giometti\@linux.it>$
#       pass:printk: console \[ttyS0\] enabled$
#       pass:Process accounting resumed$
#       pass:PTP clock support registered$
#       pass:QLogic BR-series BFA FC/FCOE SCSI driver - version: [0-9.]+$
#       pass:raid6: skip pq benchmark and using algorithm avx2x4$
#       pass:raid6: using avx2x2 recovery algorithm$
#       pass:random: crng done \(trusting CPU's manufacturer\)$
#       pass:random: get_random_u64 called from __kmem_cache_create\+0x28/0x500 with crng_init=0$
#       pass:RAS: Correctable Errors collector initialized\.$
#       pass:rcu: Adjusting geometry for rcu_fanout_leaf=16, nr_cpu_ids=1$
#       pass:rcu: Hierarchical SRCU implementation\.$
#       pass:rcu: Preemptible hierarchical RCU implementation\.$
#       pass:rcu: RCU calculated value of scheduler-enlistment delay is 100 jiffies\.$
#       pass:rcu: \\x09CONFIG_RCU_FANOUT set to non-default value of 32\.$
#       pass:rcu: \\x09Offload RCU callbacks from CPUs: \(none\)\.$
#       pass:rcu: \\x09RCU dyntick-idle grace-period acceleration is enabled\.$
#       pass:rcu: \\x09RCU restricting CPUs from NR_CPUS=256 to nr_cpu_ids=1\.$
#       pass:registered taskstats version 1$
#       pass:RocketRAID 3xxx/4xxx Controller driver v[0-9.]+$
#       pass:rodata_test: all tests were successful$
#       pass:romfs: ROMFS MTD \(C\) 2007 Red Hat, Inc\.$
#       pass:RPC: Registered named UNIX socket transport module\.$
#       pass:RPC: Registered tcp NFSv4\.1 backchannel transport module\.$
#       pass:RPC: Registered tcp transport module\.$
#       pass:RPC: Registered udp transport module\.$
#       pass:RPL Segment Routing with IPv6$
#       pass:rtc_cmos 00:00: alarms up to one day, y3k, 114 bytes nvram, hpet irqs$
#       pass:rtc_cmos 00:00: registered as rtc0$
#       pass:rtc_cmos 00:00: RTC can wake from S4$
#       pass:Run /sbin/init as init process$
#       pass:sched_clock: Marking stable \(3803829149, 140256731\)->\(4004985712, -60899832\)$
#       pass:scsi 0:0:0:0: Direct-Access     QEMU     QEMU HARDDISK    2\.5\+ PQ: 0 ANSI: 5$
#       pass:scsi 1:0:1:2: Direct-Access     QEMU     QEMU HARDDISK    2\.5\+ PQ: 0 ANSI: 5$
#       pass:scsi host0: Virtio SCSI HBA$
#       pass:scsi host1: Virtio SCSI HBA$
#       pass:scsi host2: ahci$
#       pass:scsi host3: ahci$
#       pass:scsi host4: ahci$
#       pass:scsi host5: ahci$
#       pass:scsi host6: ahci$
#       pass:scsi host7: ahci$
#       pass:scsi host8: pata_legacy$
#       pass:SCSI subsystem initialized$
#       pass:sd 0:0:0:0: \[sda\] 51380224 512-byte logical blocks: \(26\.3 GB/24\.5 GiB\)$
#       pass:sd 0:0:0:0: \[sda\] Attached SCSI disk$
#       pass:sd 0:0:0:0: \[sda\] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA$
#       pass:sd 0:0:0:0: \[sda\] Write Protect is off$
#       pass:sd 1:0:1:2: \[sdb\] 1048576 512-byte logical blocks: \(537 MB/512 MiB\)$
#       pass:sd 1:0:1:2: \[sdb\] Attached SCSI disk$
#       pass:sd 1:0:1:2: \[sdb\] Write cache: enabled, read cache: enabled, doesn't support DPO or FUA$
#       pass:sd 1:0:1:2: \[sdb\] Write Protect is off$
#       pass:sd [0-9:]+: Power-on or device reset occurred$
#       pass:Segment Routing with IPv6$
#       pass:Serial: 8250/16550 driver, 4 ports, IRQ sharing enabled$
#       pass:serio: i8042 AUX port at 0x60,0x64 irq 12$
#       pass:serio: i8042 KBD port at 0x60,0x64 irq 1$
#       pass:setup_percpu: NR_CPUS:256 nr_cpumask_bits:256 nr_cpu_ids:1 nr_node_ids:1$
#       pass:SGI XFS with ACLs, security attributes, scrub, quota, no debug enabled$
#       pass:signal: max sigframe size: 1776$
#       pass:SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=1, Nodes=1$
#       pass:SMBIOS 2\.8 present\.$
#       pass:smp: Bringing up secondary CPUs \.\.\.$
#       pass:smp: Brought up 1 node, 1 CPU$
#       pass:smpboot: Allowing 1 CPUs, 0 hotplug CPUs$
#       pass:smpboot: CPU0: AMD EPYC 7501 32-Core Processor \(family: 0x17, model: 0x1, stepping: 0x2\)$
#       pass:smpboot: Max logical packages: 1$
#       pass:smpboot: Total of 1 processors activated \(3999.99 BogoMIPS\)$
#       pass:Spectre V1 : Mitigation: usercopy/swapgs barriers and __user pointer sanitization$
#       pass:Spectre V2 : mitigation: Enabling conditional Indirect Branch Prediction Barrier$
#       pass:Spectre V2 : Mitigation: Retpolines$
#       pass:Spectre V2 : Spectre v2 / SpectreRSB mitigation: Filling RSB on context switch$
#       pass:Speculative Store Bypass: Mitigation: Speculative Store Bypass disabled via prctl and seccomp$
#       pass:squashfs: version [0-9.]+ \([0-9/]+\) Phillip Lougher$
#       pass:st: Version 20160209, fixed bufsize 32768, s/g segs 256$
#       pass:stex: Promise SuperTrak EX Driver version: [0-9.]+$
#       pass:Switched APIC routing to physical x2apic\.$
#       pass:TCP bind hash table entries: 8192 \(order: 5, 131072 bytes, linear\)$
#       pass:TCP established hash table entries: 8192 \(order: 4, 65536 bytes, linear\)$
#       pass:TCP: Hash tables configured \(established 8192 bind 8192\)$
#       pass:tcp_listen_portaddr_hash hash table entries: 512 \(order: 1, 8192 bytes, linear\)$
#       pass:thermal_sys: Registered thermal governor ’bang_bang’$
#       pass:thermal_sys: Registered thermal governor ’fair_share’$
#       pass:thermal_sys: Registered thermal governor ’step_wise’$
#       pass:thermal_sys: Registered thermal governor ’user_space’$
#       pass:TSC deadline timer available$
#       pass:tsc: Detected 1999.995 MHz processor$
#       pass:UDP hash table entries: 512 \(order: 2, 16384 bytes, linear\)$
#       pass:UDP-Lite hash table entries: 512 \(order: 2, 16384 bytes, linear\)$
#       pass:Unknown kernel command line parameters "BOOT_IMAGE=/boot/vmlinuz-huge-5\.15\.38", will be passed to user space\.$
#       pass:usbcore: registered new device driver usb$
#       pass:usbcore: registered new interface driver hub$
#       pass:usbcore: registered new interface driver usbfs$
#       pass:Using GB pages for direct mapping$
#       pass:VFS: Disk quotas dquot_6\.6\.0$
#       pass:VFS: Dquot-cache hash table entries: 512 \(order 0, 4096 bytes\)$
#       pass:VFS: Mounted root \(ext4 filesystem\) readonly on device 8:0\.$
#       pass:vgaarb: loaded$
#       pass:virtio_pci_modern_dev: exports duplicate symbol vp_modern_config_vector \(owned by kernel\)$
#       pass:work still pending$
#       pass:workingset: timestamp_bits=40 max_order=18 bucket_order=0$
#       pass:Write protecting the kernel read-only data: 30720k$
#       pass:x2apic enabled$
#       pass:x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using ’compacted’ format\.$
#       pass:x86/fpu: Supporting XSAVE feature 0x001: ’x87 floating point registers’$
#       pass:x86/fpu: Supporting XSAVE feature 0x002: ’SSE registers’$
#       pass:x86/fpu: Supporting XSAVE feature 0x004: ’AVX registers’$
#       pass:x86/fpu: xstate_offset\[2\]:  576, xstate_sizes\[2\]:  256$
#       pass:x86/mm: Memory block size: 128MB$
#       pass:x86/PAT: Configuration \[0-7\]: WB  WC  UC- UC  WB  WP  UC- WT$
#       pass:xor: automatically using best checksumming function   avx$
#       pass:zbud: loaded$
#       pass:Zone ranges:$
#       pass:zswap: loaded using pool lzo/zbud$
#       pass:\.\.TIMER: vector=0x30 apic1=0 pin1=2 apic2=-1 pin2=-1$
#       pass:\.\.\. bit width:              48$
#       pass:\.\.\. event mask:             000000000000003f$
#       pass:\.\.\. fixed-purpose events:   0$
#       pass:\.\.\. generic registers:      6$
#       pass:\.\.\. max period:             00007fffffffffff$
#       pass:\.\.\. value mask:             0000ffffffffffff$
#       pass:\.\.\. version:                0$
#       pass:\[drm\] Found bochs VGA, ID 0xb0c0\.$
#       pass:\[drm\] Framebuffer size 16384 kB \@ 0xfd000000, mmio \@ 0xfebd0000\.$
#       pass:\[drm\] Initialized bochs-drm 1\.0\.0 20130925 for 0000:00:01\.0 on minor 0$
#       pass:\[mem 0x40000000-0xafffffff\] available for PCI devices$
#       pass:\\x09Rude variant of Tasks RCU enabled\.$
#       pass:\\x09Tracing variant of Tasks RCU enabled\.$
#       pass:\\x09Trampoline variant of Tasks RCU enabled\.$
#       type:
#done



##########
# RC.INET1 - network run commands

# For now, pass everything. We don't expect to see anything actionable here.
pass:rc\.inet1$re_pid:[ ]
#with:rc\.inet1$re_pid:[ ]
#       pass:Configuring gateways$
#       pass:De-configuring gateways$
#       pass:eth0: configuring interface$
#       pass:eth0: de-configuring interface$
#       pass:eth0: enabling SLAAC$
#       pass:eth0: setting IPv4 addresses$
#       pass:eth1: configuring interface$
#       pass:eth1: de-configuring interface$
#       pass:eth2: configuring interface$
#       pass:eth2: de-configuring interface$
#       pass:eth3: configuring interface$
#       pass:eth3: de-configuring interface$
#       pass:eth4: configuring interface$
#       pass:eth4: de-configuring interface$
#       pass:eth5: configuring interface$
#       pass:eth5: de-configuring interface$
#       pass:lo: configuring interface$
#       pass:lo: de-configuring interface$
#       type:
#done



##########
# ACPID - ACPI event daemon

# For now, pass everything. We don’t expect to see anything actionable here.
pass:acpid:[ ]
#with:acpid:[ ]
#       pass:1 rule loaded$
#       pass:exiting$
#       pass:starting up with netlink and the input layer$
#       pass:waiting for events: event logging is off$
#       type:
#done



##########
# HAVEGED - random number daemon

# For now, pass everything. We don’t expect to see anything actionable here.
pass:haveged:[ ]
#with:haveged:[ ]
#       pass:haveged starting up
#       pass:haveged: cpu: \(VC\); data: 64K \(V\); inst: 64K \(V\); idx: 39/40; sz: 53875/53875$
#       pass:haveged: fills: 0, generated: 0$
#       pass:haveged: Stopping due to signal 15$
#       pass:haveged: tot tests\(BA8\): A:1/1 B:1/1 continuous tests\(B\):  last entropy estimate 7\.99581$
#       pass:haveged: ver: [0-9.]+; arch: x86; vend: ; build: \(gcc [0-9.]+ ITV\); collect: 128K$
#       type:
#done



##########
# UDEVD - uevents daemon

# For now, pass everything. We don’t expect to see anything actionable here.
pass:udevd$re_pid:[ ]
#pass:udevd$re_pid:  starting eudev-3\.2\.11$



##########
# INIT - master process

# For now, pass everything. We don’t expect to see anything actionable here.
pass:init:[ ]
#pass:init: Switching to runlevel: 6$
#pass:init: Trying to re-exec init$



##########
# SHUTDOWN - utility program to shutdown the operating system

# For now, pass everything. We don’t expect to see anything actionable here.
pass:shutdown:$re_pid:[ ]
#pass:shutdown$re_pid: shutting down for system reboot$
```

```
  ##########
  # FINAL CATCHALL - for now, report everything unknown.

  type:
```

This code is written to file `logactions.conf`.
Uses `actions` 36a, `catchalls` 36a, `patterns` 36a, `re_dovecot_session` 31a, `re_hname` 30c, `re_ip` 30b, `re_ipv4` 30b, `re_ipv6` 30b, and `re_pid` 30a.

7 syslog.logactions.conf – route copies of all syslog traffic

All syslog traffic gets routed to logactions.pl via a named pipe. This gets called by the include command in the primary file syslog.conf.

105 ⟨*syslog.logactions.conf* 105⟩≡

```
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
# This syslog rule tells syslog to send all messages to the named pipe (fifo)
# that logactions.pl will be reading.
*.*     |/var/log/fifo_logactions
```

This code is written to file syslog.logactions.conf.

8 logactions.pl.8 – manual page for logactions.pl

This file provides the content behind the commands:

- man logactions.pl
- info logactions.pl

106 ⟨*logactions.pl.8* 106⟩≡

```
.\" This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
.
.TH logactions.pl 8
.
.SH NAME
logactions.pl \- take action in firewall on certain syslog messages
.
.SH SYNOPSIS
.SY logactions.pl
[ \fIoption\fP ... ]
.
.SH DESCRIPTION
This program runs as a daemon (detached process), monitoring the system log.
Each log message is compared to a ruleset defined in \fB/etc/logactions.conf\fP.
If the message matches the rule, then the associated action is taken.
The program is started automatically during multiuser startup and stopped during
multiuser shutdown.
.
.SH OPTIONS
The options below currently cannot be combined into one argument (such as
\fB-pd\fP).
Each option should be a separate argument (such as \fB-p -d\fP).
This restriction may be lifted in a future update.
.
.TP
.B -p
```

```
Do not detach.
Instead, (p)rint recent interesting events.
The output is typed to the terminal.
.RS
.PP
The option prints the last ten interesting events from each monitored syslog
logfile in turn.
The only events printed are:
.IP (i)
unknown events that were processed by a "catchall" rule, and
.IP (ii)
known and particularly important events.
.PP
This means the operator can easily see if an important event happened recently,
and can also easily see if a new rule, or a change to an existing rule, is
needed.
.PP
The operator should run this option periodically.
It can, for example, be run by \fBcron\fP and the results emailed to the
operator.
.RE
.
.TP
.B -d
Do not detach.
Instead, enable (d)ebugging mode.
Type any diagnostic messages to \fBstderr\fP.
This is primarily for use by the script developer, but might also be useful to
the operator.
This option can be given multiple times.
Each use adds additional types of diagnostic message to the output.
.
.TP
.B -r \fIrulenumber\fP
```

```
Do not detach.
Instead, write (r)ule \fIrulenumber\fP to \fBstderr\fP and exit.
.RS
.PP
Suppose the daemon logs an action such as:
.IP
logactions kill: blocked ip=\fIip-address\fP rule=48 elapsed=1.124s
.PP
The operator who wants to know the exact pattern that rule 48 uses can enter the
command:
.IP
\fBlogactions.pl\~-r\~48\fP
.PP
With the \fB-d\fP option, the output will contain more information.
.RE
.
.SH EXIT STATUS
.TP
.B 0
Success
.TP
.B 1
Returned by the daemon when killed, or when any fatal error occurs.
Fatal errors are listed below in the ERRORS section.
.
.SH ERRORS
Error messages are always typed on \fCstderr\fP.
Error messages associated solely with the \fB-d\fP (debug) option are not
documented here.
.
.TP
.B -r (rulequery) needs a rule number
Indicates a missing argument.
A rule number should be the next argument after the \fB-r\fP option argument.
```

```
Fatal.
.
.TP
.B usage: [...]
Indicates an unrecognized option argument.
Prints the known option arguments.
Fatal.
.
.TP
.B open \fIlogfile name\fP: \fIcondition\fP
The error condition shown prevented the program from opening a syslog logfile.
Fatal.
.
.TP
.B open \fIfifo name\fP: \fIcondition\fP
The error condition shown prevented the program from opening the fifo.
Fatal.
.
.TP
.B Starting logactions.pl, pid=\fInumber\fP
The message monitor was successfully started as a daemon (detached process).
.
.TP
.B fifo disconnected, waiting
Reached the end of the fifo, which indicates that the syslog daemon has stopped.
The program waits for the syslog daemon to resume.
.
.TP
.B reopen \fIfifo name\fP: \fIcondition\fP
The error condition shown prevented the program from reopening the fifo and
waiting for the syslog daemon to resume.
Fatal.
.
.TP
```

```
.B fifo reconnected, resuming
The syslog daemon has restarted and message monitoring will resume.
.
.TP
.B unknown action
A rule in the active ruleset contains an unknown action word.
This is not supposed to be possible, as the ruleset is error-checked while being
read in. This message can be reported to the developer as a possible logic error
in the program.
Fatal.
.
.TP
.B fell through a hole
The search for a matching rule went past the end of the ruleset.
This is not supposed to be possible, as the ruleset always has an implicit
"match everything" at the end of it.
This message can be reported to the developer as a possible logic error in the
program.
Fatal.
.
.TP
.B not blocked, not IPv4 or IPv6 address
A rule calls for blocking an IP address, but in this case provided a value that
is not an IP address.
.
.TP
.B nft delete element returned \fIexitcode\fP and reported error
.B nft add element returned \fIexitcode\fP and reported error
The \fBnft\fP command returned an unexpected error condition while blocking an
IP address.
.
.TP
.B blocked
The \fBnft\fP command was used successfully to block an IP address.
```

```
.
.TP
.B skipped Spamhaus ZEN, not IPv4 address
When checking an IP address against the Spamhaus ZEN service, it must be an IPv4
address.
.
.TP
.B ip=\fIip\fP spamhaus_zen=\fIspamhaus_zen\fP
An IP match was found in the Spamhaus ZEN service.
Note that a match found in the Spamhaus ZEN service will only be blocked when
the Spamhaus opinion indicates the IP address is a source of spam, exploits, or
authorization hijacks.
.
.TP
.B hname=\fIhname\fP spamhaus_dbl=\fIspamhaus_dbl\fP
A name match was found in the Spamhaus domain blocklist.
.
.TP
.B skipped bl.de, not IPv4 or IPv6 address
A rule calls for checking an IP address against blocklist.de, but in this case
provided a value that is not an IP address.
.
.TP
.B ip=\fIip\fP bde=\fIbde\fI
An IP match was found in the blocklist.de service and will be blocked.
.TP
.B command_stdout: pipe
An error condition prevented the creation of the \fBstdout\fP pipe when needed
to execute a system command.
Fatal.
.
.TP
.B open \fIcfpath\fP: \fIcondition\fP
The error condition shown prevented the program from opening the configuration
```

```
file.
Fatal.
.
.TP
.B curly brackets forbidden
Rule patterns cannot contain curly brackets.
The restriction is necessary to let the program compile the expression using
\fBqr{}\fP.
This restriction will be lifted if a way can be found.
.
.TP
.B missing pattern
While reading a rule, its pattern was not found.
.
.TP
.B unknown command
While reading a rule, an unknown command was encountered.
.
.TP
.B rule \fInumber\fP not found
A rule having the number given as the argument to the \fB-r\fP option was not
found in the running ruleset.
.
.TP
.B running ruleset loaded from \fIcfpath\fP
Running ruleset was loaded (or reloaded) from the named configuration file.
.
.SH ENVIRONMENT
The environment vector is passed unchanged to the utilities called by the
command, such as \fBnft\fP, except that PATH is explicitly set to:
.IP
\fC/usr/sbin:/sbin:/usr/bin:/bin\fP
.
.SH FILES
```

```
.TP
\fC/etc/logactions.conf\fP
The configuration file containing the ruleset
.TP
\fC/var/log/fifo_logactions\fP
The fifo from which the program reads new syslog messages.
.TP
\fC/var/log\fP
The directory where the program finds past syslog messages in various files
named in the configuration file.
.
.SH VERSIONS
1.0a \(en alpha-quality release by Edward McGuire.
.
.SH "CONFORMING TO"
Slackware 15.0
.
.SH NOTES
The Noweb literate programming tool by Norman Ramsey was used to develop this
package.
.
.SH BUGS
Report bugs to the author.
.
.SH EXAMPLES
.TP
# \fBlogactions.pl -p\fP
.TQ
%logactions-i-config, configuration file loaded
.TQ
[recent notable syslog messages]
.TP
# \fBlogactions.pl -r 42\fP
.TQ
```

```
%logactions-s-config, rule 42=<kill:Disconnected from user \S+ ($re_ip) port \d+ \[preauth\]$>
.TP
# \fBlogactions.pl -d\fP
.TQ
%logactions-s-retcodes, Spamhaus return codes defined: 50
.TQ
%logactions-s-retcodes, blocklist.de return codes defined: 21
.TQ
%logactions-i-config, configuration file loaded
.TQ
%logactions-s-config, pattern/action rules defined: 406
.TQ
%logactions-s-monitor, not detached, press C-c to interrupt
.
.SH AUTHORS
Edward McGuire
.
.SH "SEE ALSO"
.BR rc.firewall(8)
```
This code is written to file `logactions.pl.8`.
Uses `command_stdout` 54, `patterns` 36a, and `re_ip` 30b.

9 nft-geofilter.pl – process to update geofilter rules in NFT firewall

The program nft-geofilter.pl creates filewall commands. These commands block IP addresses by geocode – basically, by country.

115a ⟨*nft-geofilter.pl* 115a⟩≡

```
  ⟨nft-geofilter.pl preamble 115b⟩
  ⟨nft-geofilter.pl itemize geocodes to block 116⟩
  ⟨nft-geofilter.pl delete existing tables 117b⟩
  ⟨nft-geofilter.pl create new empty IPv4 table 118⟩
  ⟨nft-geofilter.pl create new empty IPv6 table 119⟩
  ⟨nft-geofilter.pl grab and cache source file 120a⟩
  ⟨nft-geofilter.pl connect CSV reader 120b⟩
  ⟨nft-geofilter.pl read and process each row 121a⟩
```

This code is written to file nft-geofilter.pl.

The program is a Perl script.

115b ⟨*nft-geofilter.pl preamble* 115b⟩≡ (115a)

```
  #! /usr/bin/perl -T
  # This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
  use strict ;
  use warnings ;
  use IO::Uncompress::Gunzip qw( $GunzipError ) ;
  use Data::Dumper ;
  use Text::CSV ;
  use POSIX ;
  use HTTP::Tiny ;
```

The program takes its input from a website called `db-ip.com`. This site distributes tables that relate IP address blocks to country codes. The specific table we use is the DBIP Country Lite table. The format of the table is described on the site, at page: `https://db-ip.com/db/format/ip-to-country-lite/csv.html`. In a nutshell, it is a CSV file conforming to RFC 4180, and using UTF-8 encoding. It has three columns:

- `ip_start` first IP address in the block
- `ip_end` last IP address in the block
- `country` country code using ISO 3166-1 alpha-2 representation

The output of the program is an NFT command file. The output format is described in the manual page for the `nft` command.

Here we identify the countries to block. Based on actual experience, the most frequent country code associated with breakin attempts is CN, followed by RU, VN, US, HK, SG, and IN, and to a lesser extent KR, NL, DE, TW, BR, JP, FR, ID. These are all blocked except US and DE which would interfere with intended use. (Systems would be so much safer if we didn't have to allow users.)

For `alienBOB` I unblocked NL even though it's a common source of breakin attempts.

116 ⟨*nft-geofilter.pl itemize geocodes to block* 116⟩≡ (115a)

```
my %filtercc = (
        CN => 1 , VN => 1 ,
        RU => 1 , HK => 1 ,
        SG => 1 , IN => 1 ,
        KR => 1 ,
        TW => 1 , BR => 1 ,
        JP => 1 , FR => 1 ,
        ID => 1 ,
) ;
```

The first NFT commands flush existing geocode lists out of the firewall.

These commands are ordered by country code, just for debugging purposes. We can compare two outputs using `diff` this way.

This approach is buggy. When the table does not exist, the `flush set` commands fail. When the table does exist, the process of atomically rewriting it takes many minutes.

117a ⟨*nft-geofilter.pl flush existing tables* 117a⟩≡

```
print qq(flush table ip TABLE_GEO_IPV4\n) ;
print qq(flush table ip6 TABLE_GEO_IPV6\n) ;
foreach my $filtercc ( sort keys %filtercc )
{
        print qq(flush set ip TABLE_GEO_IPV4 SET_GEO_IPV4_$filtercc\n) ;
        print qq(flush set ip6 TABLE_GEO_IPV6 SET_GEO_IPV6_$filtercc\n) ;
}
```

The first NFT commands delete the existing geocode tables completely. To ensure this runs without error, there are two steps: (i) create the table if it does not exist; (ii) delete the table.

117b ⟨*nft-geofilter.pl delete existing tables* 117b⟩≡ (115a)

```
print qq(add    table ip TABLE_GEO_IPV4\n) ;
print qq(delete table ip TABLE_GEO_IPV4\n) ;
print qq(add    table ip6 TABLE_GEO_IPV6\n) ;
print qq(delete table ip6 TABLE_GEO_IPV6\n) ;
```

With the existing geocode tables out of the way, we create new, empty tables. Here we create the IPv4 table. This table has an initially empty list of IPv4 address blocks, and contains a rule that drops any packet originating from an IPv4 address on the list.

118 ⟨*nft-geofilter.pl create new empty IPv4 table* 118⟩≡ (115a)

```
print qq(table ip TABLE_GEO_IPV4 {\n) ;
foreach my $filtercc ( sort keys %filtercc )
{
        print qq(\tset SET_GEO_IPV4_$filtercc {\n) ;
        print qq(\t\ttype ipv4_addr;\n) ;
        print qq(\t\tflags interval;\n) ;
        print qq(\t}\n) ;
}
print qq(\tchain CHAIN_GEO_IPV4 {\n) ;
print qq(\t\ttype filter hook input priority -300; policy accept;\n) ;
foreach my $filtercc ( sort keys %filtercc )
{
        print qq(\t\tip saddr \@SET_GEO_IPV4_$filtercc counter drop\n) ;
}
print qq(\t}\n) ;
print qq(}\n) ;
print qq(\n) ;
```

And we do the same for IPv6 addresses.

119 ⟨*nft-geofilter.pl create new empty IPv6 table* 119⟩≡ (115a)

```
print qq(table ip6 TABLE_GEO_IPV6 {\n) ;
foreach my $filtercc ( sort keys %filtercc )
{
        print qq(\tset SET_GEO_IPV6_$filtercc {\n) ;
        print qq(\t\ttype ipv6_addr;\n) ;
        print qq(\t\tflags interval;\n) ;
        print qq(\t}\n) ;
}
print qq(\tchain CHAIN_GEO_IPV6 {\n) ;
print qq(\t\ttype filter hook input priority -300; policy accept;\n) ;
foreach my $filtercc ( sort keys %filtercc )
{
        print qq(\t\tip6 saddr \@SET_GEO_IPV6_$filtercc counter drop\n) ;
}
print qq(\t}\n) ;
print qq(}\n) ;
print qq(\n) ;
```

Then it’s time to populate the list of address blocks that should be dropped. First, we grab and cache the external file that relates IP address ranges to country codes.

120a ⟨*nft-geofilter.pl grab and cache source file* 120a⟩≡ (115a)

```
my $csv_gz_path = ’dbip-country-lite-’ . strftime( ’%Y-%m’ , localtime time ) . ’.csv.gz’ ;
my $http_csv_gz_path = ’https://download.db-ip.com/free/’ . $csv_gz_path ;
my $http = HTTP::Tiny->new();
my $http_response = $http->mirror( $http_csv_gz_path , $csv_gz_path ) ;
if ( $http_response->{success} )
{
        if ( $http_response->{status} eq ’304’ )
        {
                warn qq(%nftgeofilter-i, used existing "$http_csv_gz_path"\n) ;
        }
        else
        {
                warn qq(%nftgeofilter-i, downloaded "$http_csv_gz_path"\n) ;
        }
}
else
{
        die qq(%nftgeofilter-f, failed to get IP database, fail reason ") . Dumper $http_response . qq("\n) ;
}
```

Second, we uncompress the file and feed it to a CSV reader.

120b ⟨*nft-geofilter.pl connect CSV reader* 120b⟩≡ (115a)

```
my $csv_as_string ;
my $csv_gz_stream = IO::Uncompress::Gunzip->new( $csv_gz_path )
        or die qq(%nftgeofilter-f, failed to uncompress IP database, fail reason "$GunzipError"\n) ;
my $csv_object = Text::CSV->new()
        or die qq(%nftgeofilter-f, failed to create CSV stream, fail reason ") . Text::CSV->error_diag() . qq("\n) ;
```

Third, we convert each line of the CSV into NFT firewall commands that populate the IPv4 and IPv6 address drop lists.

121a ⟨*nft-geofilter.pl read and process each row* 121a⟩≡ (115a)

```
while ( my $row = $csv_object->getline( $csv_gz_stream ) )
{
        ⟨nft-geofilter.pl extract values 121b⟩
        ⟨nft-geofilter.pl filter on geocode 121c⟩
        ⟨nft-geofilter.pl create range expression 121d⟩
        ⟨nft-geofilter.pl add address range to set 122⟩
}
```

Here we extract the three values IP˙START, IP˙END, and COUNTRY.

121b ⟨*nft-geofilter.pl extract values* 121b⟩≡ (121a)

```
@$row == 3
        or die qq(wrong number of columns, record ) . $csv_object->record_number() . qq(\n) ;
my ( $IP_START , $IP_END , $COUNTRY ) = @$row ;
```

Now we validate COUNTRY, and skip this line unless it is on the list of countries to block.

121c ⟨*nft-geofilter.pl filter on geocode* 121c⟩≡ (121a)

```
$COUNTRY =~ /^[A-Z]{2}$/
        or die qq(invalid country code, record ) . $csv_object->record_number() . qq(\n) ;
next unless defined $filtercc{ $COUNTRY } ;
```

With a good address block in hand, we now create an NFT range expression from the IP˙START and IP˙END values.

121d ⟨*nft-geofilter.pl create range expression* 121d⟩≡ (121a)

```
my $range = $IP_START eq $IP_END
        ? $IP_START
        : qq($IP_START-$IP_END)
        ;
```

Depending on whether this is an IPv4 or IPv6 block, we add the address range into the appropriate set.

122 ⟨*nft-geofilter.pl add address range to set* 122⟩≡ (121a)

```
my $re_ipv4 = '([0-9]+)\.([0-9]+)\.([0-9]+)\.([0-9]+)' ;
my $re_ipv6 = '[0-9a-z:]+' ;
if ( my ( $octet1 , $octet2 , $octet3, $octet4 ) = $IP_START =~ /^$re_ipv4$/ )
{
        $octet1 < 256 && $octet2 < 256 && $octet3 < 256 && $octet4 < 256
                or die qq(bad octet in IPV4 IP_START, record ) . $csv_object->record_number() . qq(\n) ;
        ( $octet1 , $octet2 , $octet3, $octet4 ) = $IP_END =~ /^$re_ipv4$/
                or die qq(bad IPV4 IP_END address, record ) . $csv_object->record_number() . qq(\n) ;
        $octet1 < 256 && $octet2 < 256 && $octet3 < 256 && $octet4 < 256
                or die qq(bad octet in IPV4 IP_END, record ) . $csv_object->record_number() . qq(\n) ;
        print qq(add element ip TABLE_GEO_IPV4 SET_GEO_IPV4_$COUNTRY { $range }\n) ;
}
elsif ( $IP_START =~ /^$re_ipv6$/ and $IP_END =~ /^$re_ipv6$/ )
{
        print qq(add element ip6 TABLE_GEO_IPV6 SET_GEO_IPV6_$COUNTRY { $range }\n) ;
}
else
{
        die qq(bad IPV4 IP_START address, record ) . $csv_object->record_number() . qq(\n) ;
}
```

Uses `re_ipv4` 30b and `re_ipv6` 30b.

10 nfttools-crontab – crontab for nfttools

This is a crontab file that schedules nfttools maintenance operations.

123a ⟨*nfttools-crontab* 123a⟩≡

```
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
⟨nfttools-crontab hourly 123b⟩
⟨nfttools-crontab daily 123c⟩
⟨nfttools-crontab monthly 124⟩
```

This code is written to file `nfttools-crontab`.

The hourly job below snapshots the running NFT firewall ruleset. The ruleset is memory-resident in the kernel, including any real-time IP blocks that have been added since multiuser startup. This job preserves the ruleset in the event of unexpected kernel failure, such as a power-fail or a panic. The snapshot is saved as the startup ruleset, so that the ruleset will be restored after the reboot.

123b ⟨*nfttools-crontab hourly* 123b⟩≡ (123a)

```
0 * * * * /etc/rc.d/rc.firewall save
```

The daily job below writes a report of various firewall-related statistics. The report is written to the job's `stdout` which means it will be emailed.

123c ⟨*nfttools-crontab daily* 123c⟩≡ (123a)

```
0 0 * * * /usr/lib/nfttools/securityreport
```

The monthly job below refreshes the firewall geofilter using `nft-geofilter.pl`. The refresh runs just after db-ip.com drops new country-lite geotables files. The drop date is the first day of each month. During 2022, I have observed drop times ranging from 0246 to 0410, with a trend toward later. I allow 12 hours for the drop to complete, starting the download at 1200 UTC.

To run a job at a time expressed in UTC, the "Dillon's" cron daemon supplied with Slackware 15.0 is inadequate. It interprets the crontab schedule in terms of local time, not UTC. So we schedule the job early, on the 25th of the prior month. The job calculates how many minutes to wait for the correct start time, and schedules an `at` job to run at that time. The `at` job actually does the work.

The `date` command is used to get the seconds-since-epoch of the desired runtime, and the seconds-since-epoch of the current time. These are converted to minutes, and subtracted to get the number of minutes to wait. Then `at` is invoked to schedule the job that many minutes from now.

124 ⟨*nfttools-crontab monthly* 124⟩≡ (123a)

```
  0 0 25 * * export TZ=UTC0;cd /var/lib/nfttools;at -f/usr/lib/nfttools/nfttools.atjob NOW + $(($(date -d"$(date +%Y%m01) +1 month +12
```

11 `nfttools.atjob` – atjob for nfttools

This is a file containing `/bin/sh` commands suitable for feeding to `at`. It runs `nft-geofilter.pl` and passes the output to `nft -f-`, causing the geofilter to be updated. It snapshots the running firewall ruleset before and after.

125 ⟨*nfttools.atjob* 125⟩≡

```
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
/etc/rc.d/rc.firewall save
nft-geofilter.pl | nft -f-
/etc/rc.d/rc.firewall save
```

This code is written to file `nfttools.atjob`.

12 nft-geofilter.pl.8 – manual page for nft-geofilter.pl

This file provides the content behind the commands:

- man nft-geofilter.pl
- info nft-geofilter.pl

126 ⟨*nft-geofilter.pl.8* 126⟩≡

```
.\" This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
.
.TH nft-geofilter.pl 8
.
.SH NAME
nft-geofilter.pl \- produce NFT firewall commands that update the firewall’s geofilter
.
.SH SYNOPSIS
.SY nft-geofilter.pl
.
.SH DESCRIPTION
This program generates an NFT firewall script on \fBstdout\fP that
.IP (i)
flushes the geofilter;
.IP (ii)
creates an empty geofilter; and,
.IP (iii)
populates the geofilter with IP address ranges that should be dropped.
.
.SH OPTIONS
(none)
.
.SH EXIT STATUS
.IP 0
Success.
.IP non-zero
```

```
Failure.
See the error message for details.
.
.SH ERRORS
Error messages are always typed on \fBstderr\fP.
.
.TP
.B used existing "\fIfilename\fP"
The cached copy of the latest geocode file will be used.
.
.TP
.B downloaded "\fIfilename\fP"
No cached copy was found, so the file was downloaded and cached.
.
.TP
.B failed to get IP database, fail reason "\fIresponse\fP"
Download was attempted, but failed for the reason given.
Fatal.
.
.TP
.B failed to uncompress IP database, fail reason "\fIresponse\fP"
The geocode file could not be uncompressed for the reason given.
If you have a cached copy, try removing the cached copy and repeating the
command.
Fatal.
.
.TP
.B failed to create CSV stream, fail reason "\fIresponse\fP"
The library function Text::CSV->new() failed for the reason given.
Fatal.
.
.TP
.B wrong number of columns, record \fInumber\fP
Every row of the geocode CSV file is expected to have three values.
```

```
Fatal.
.
.TP
.B invalid country code, record \fInumber\fP
The third value of every row is expected to be a two-letter, capitalized country
code.
Fatal.
.
.SH ENVIRONMENT
The environment vector is not used, and is passed unchanged to any utilities
called by the program.
.
.SH FILES
.
.TP
.B https://download.db-ip.com/free/dbip-country-lite-\fIyyyy\fP-\fImm\fP.csv.gz
The remote source file that relates IP address blocks to country codes.
.
.TP
.B ./dbip-country-lite-\fIyyyy\fP-\fImm\fP.csv.gz
Cached copy of the remote source file, saved in the working directory.
.
.SH VERSIONS
1.0a \(en alpha-quality release by Edward McGuire.
.
.SH CONFORMING TO
Slackware 15.0
.
.SH NOTES
The Noweb literate programming tool by Norman Ramsey was used to develop this
package.
.
.SH BUGS
Report bugs to the author.
```

```
.
.SH EXAMPLES
.
.TP
# \fBnft-geofilter.pl > nft-geofilter.out\fP
# \fBnft -f nft-geofilter.out\fP

.
.SH AUTHORS
Edward McGuire
.
.SH SEE ALSO
.BR nft(8)
```

This code is written to file `nft-geofilter.pl.8`.

13 slack-desc – a file containing a description of this package formatted for use with Slackware 15.0 makepkg and pkgtool

The format of this file is not described in `man makepkg` or `man pkgtool`, so I document it here.

Lines starting with `|` are comment lines.

All lines should be 79 columns long (or shorter). The first line is normally a comment that forms a 79-column ruler. The ruler is there to help the maintainer of this file when changing the package description. The 79 column limit is there to avoid word wrap on an 80-column console when installing.

130a ⟨*slack-desc* 130a⟩≡ 130b ▷

```
|2345678901234567890123456789012345678901234567890123456789012345678901234567 8|
```

This code is written to file `slack-desc`.

In the second comment line, I give credit to the developer of Noweb.

130b ⟨*slack-desc* 130a⟩+≡ ◁ 130a 131 ▷

```
|Compiled from nfttools.nw using Noweb by Norman Ramsey.                      |
```

Non-comment lines contain the package description. There should be eactly 11 package description lines. Each line must be prefixed with the package's application name followed by a colon, followed optionally by one space and a description string. The application name must be an exact match to the application-name part of the package container filename, in this case `nfttools`.

Line 1 must have a description string following the prefix. The string must consist of the exact application name, followed by a single space, followed by a summary. There is no consistency to the format of the summary, but there are some common practices. Here it takes the form of the version number followed by a text in parentheses.

Line 2 must be empty (except for the prefix).

Lines 3 through 11 should give the long description of the package. The description can be padded with additional empty lines as necessary to get to 11.

It is a good idea to include a pointer to the package developer, such as an email address or project homepage URL. Here I have given that information on line 11.

131 ⟨*slack-desc* 130a⟩+≡ ◁130b

```
nfttools: nfttools 1.0a (NFT firewall management tools for Slackware)
nfttools:
nfttools: nfttools creates rc.firewall in /etc/rc.d, and creates symbolic links
nfttools: to it in /etc/rc.d/rc[016].d. At startup, rc.firewall loads the
nfttools: startup firewall configuration from /etc/nftables.conf and makes it
nfttools: the running configuration. At shutdown, rc.firewall saves the running
nfttools: configuration as the new starting configuration.
nfttools:
nfttools:
nfttools:
nfttools: nfttools, metaed, com
```

Other documents that discuss the format of `slack-desc` files:

https://docs.slackware.com/howtos:slackware_admin:building_a_package

https://www.slackwiki.com/Slack-desc.

14 securityreport – report firewall statistics

The `securityreport` script runs various reports. See the report sources below for details on what they do.

We set up a runtime environment that finds the report executables in `/usr/lib/nfttools` and keeps private data files in `/var/lib/nfttools`.

132a ⟨*securityreport* 132a⟩≡ 132b ▷

```
#!/bin/sh -x
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
PATH=/bin:/usr/bin:/sbin:/usr/sbin:/usr/lib/nfttools
cd /var/lib/nfttools
```

This code is written to file `securityreport`.

Then, in this environment, we run the reports.

132b ⟨*securityreport* 132a⟩+≡ ◁ 132a

```
blocked-by-day.sh
httpd-log-transfer-summary.sh | tail
log-geoanalyze.pl | tail
logactions.pl -p
sshd-hackers.sh
```

15 `blocked-by-day.sh` – script to summarize IP blocking activity

The report typed by `blocked-by-date.sh` shows the recent IP blocking activity. Each new IP block counts as 1 event. The report shows the total events, subtotaled by date. The count is taken from entries made in `/var/log/secure` by the IP blocking service.

133a ⟨*blocked-by-day.sh* 133a⟩≡ 133b▷

```
#! /bin/sh
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
```

This code is written to file `blocked-by-day.sh`.

The first stage filters `/var/log/secure` and returns the IP block events.

133b ⟨*blocked-by-day.sh* 133a⟩+≡ ◁133a 133c▷

```
grep -P -o -h ’^.*blocked (ip=)?[0-9.]*’ /var/log/secure{.4,.3,.2,.1,} |
```

The second stage selects the month and date and the part of the log message containing the IP address blocked. This is reduced to unique rows so that any IP blocked more than once in a day counts only once.

TODO: The uniqueness filter is broken now because the log message now also includes the rule number and elapsed time.

133c ⟨*blocked-by-day.sh* 133a⟩+≡ ◁133b 133d▷

```
        cut -c’1-7,51-’ | sort | uniq |
```

The third stage counts the blocked IP addresses, subtotaled and ordered by month and date.

133d ⟨*blocked-by-day.sh* 133a⟩+≡ ◁133c

```
        cut -c’1-6’ | sort -M | uniq -c
```

16 `httpd-log-transfer-summary.sh` – report most recent httpd activity

The report typed by `httpd-log-transfer-summary.sh` extracts all HTTP requests from the most recent system log and counts them, subtotaled by request string and result code.

134a ⟨*httpd-log-transfer-summary.sh* 134a⟩≡ 134b ▷

```
#! /bin/sh
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
```

This code is written to file `httpd-log-transfer-summary.sh`.

The first stage selects the HTTP requests from the log and extracts the request string and result code.

134b ⟨*httpd-log-transfer-summary.sh* 134a⟩+≡ ◁ 134a 134c ▷

```
< /var/log/apache sed -f /usr/lib/nfttools/httpd-log-transfer-summary.sed |
```

The second stage counts each unique request string/result code combination, and prints each combination and its count, ordered by frequency from fewest to most.

134c ⟨*httpd-log-transfer-summary.sh* 134a⟩+≡ ◁ 134b

```
        sort |
        uniq -c |
        sort
```

16.1 `httpd-log-transfer-summary.sed` – utility used by the above

This `sed` script does the job of selecting HTTP requests from an input stream and extracting the request sstring and result code.

134d ⟨*httpd-log-transfer-summary.sed* 134d⟩≡ 135a ▷

```
#! /bin/sed -f
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
```

This code is written to file `httpd-log-transfer-summary.sed`.

Parse the timestamp.

135a ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁134d 135b▷

```
# Expect a SYSLOG timestamp.
s/^[A-Z][a-z][a-z] [ 0123][0-9] [012][0-9]:[0-5][0-9]:[0-5][0-9] //
t stimeok
p
d
:stimeok
```

Parse the `syslog` hostname.

135b ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁135a 135c▷

```
# Expect the SYSLOG hostname.
s/^newjersey //
t shostok
p
d
:shostok
```

Parse the tag.

135c ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁135b 136a▷

```
# Look for access log message tags. Drop other known tags.
/^httpd\[/d
/^last message buffered [1-9][0-9]* times$/d
s/^httpd_access_log: //
t tagok
p
d
:tagok
```

Parse the Apache virtual server hostname.

136a ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁135c 136b▷

```
# Expect the Apache virtual server hostname.
s/^\([a-zA-Z0-9]\|[a-zA-Z0-9][a-zA-Z0-9-]*[a-zA-Z0-9]\)\(\.\([a-zA-Z0-9]\|[a-zA-Z0-9][a-zA-Z0-9-]*[a-zA-Z0-9]\)\)* // ;
t ahostok
p
d
:ahostok
```

Parse the client IPv4 or IPv6 address.

136b ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁136a 137a▷

```
# Expect the client IPv4 or IPv6 address.
s/^[0-9.][0-9.]* //
t caddrok
s/^[0-9a-f:][0-9a-f:][0-9a-f:]* //
t caddrok
p
d
:caddrok
```

Parse the client name (can be numeric if the name lookup failed).

137a ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁136b 137b▷

```
# Expect the client name (or address if name lookup failed).
s/^\([a-zA-Z0-9]\|[a-zA-Z0-9][a-zA-Z0-9-]*[a-zA-Z0-9]\)\(\.\([a-zA-Z0-9]\|[a-zA-Z0-9][a-zA-Z0-9-]*[a-zA-Z0-9]\)\)* // ;
t chostok
s/^[0-9.][0-9.]* //
t chostok
s/^[0-9a-f:][0-9a-f:][0-9a-f:]* //
t chostok
p
d
:chostok
```

Parse the client identity.

137b ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁137a 137c▷

```
# Expect the client identity.
s/^- - //
t identok
p
d
:identok
```

Parse the access log timestamp.

137c ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁137b 138▷

```
# Expect the access log timestamp.
s/^\[[0-9\/A-Za-z: +-][0-9\/A-Za-z: +-]*\] //
t atimeok
p
d
:atimeok
```

Parse the HTTP request and response code.

138 ⟨*httpd-log-transfer-summary.sed* 134d⟩+≡ ◁137c

```
# Expect the HTTP request and response code. Delete anything past the response
# code. This is normally the transfer size but could have been expanded to
# include other fields such as the virtual host, TLS fields, etc.
s/^\(".*" [0-9][0-9][0-9]\) .*$/\1/
```

17 log-geoanalyze.pl – report number of IP temp blocks, subtotaled by geocode

139 ⟨*log-geoanalyze.pl* 139⟩≡

```
#! /usr/bin/perl
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.

use strict ;
use warnings ;
use IO::Uncompress::Gunzip qw( $GunzipError ) ;
use POSIX ;
use Text::CSV ;

sub ipv4_to_integer ;
sub integer_to_ipv4 ;
sub geo_lookup ;

{
        my %geo_start_to_end ;
        my %geo_start_to_country = ( ) ;
        my @sorted_keys ;
        {
                my $csv_gz_path = ’dbip-country-lite-’ . strftime( ’%Y-%m’ , localtime time ) . ’.csv.gz’ ;
                -r $csv_gz_path
                        or die "%F, not readable: $csv_gz_path\n" ;
                my $csv_gz_stream = IO::Uncompress::Gunzip->new( $csv_gz_path )
                        or die "%F, failed to uncompress $csv_gz_path, fail reason: $GunzipError\n" ;
                my $csv_object = Text::CSV->new()
                        or die "%F, failed to create CSV stream from $csv_gz_path, fail reason: " . Text::CSV->error_diag() . "\n" ;
                print STDERR "%I, reading geo table from $csv_gz_path\n" ;
                while ( my $row = $csv_object->getline( $csv_gz_stream ) )
                {
                        @$row == 3
```

```
            or die qq(%F, wrong number of columns, record ) . $csv_object->record_number() . qq(\n) ;
        my ( $IP_START , $IP_END , $COUNTRY ) = @$row ;
        next if $IP_START =~ /:/ ; # ignore IPV6
        next if $IP_END =~ /:/ ; # ignore IPV6
        $COUNTRY =~ /^[A-Z]{2}$/
            or die qq(%F, invalid country code, record ) . $csv_object->record_number() . qq(\n) ;
        #$IP_START =~ m{^178\.33\.} && print STDERR "%I, IP_START=$IP_START IP_END=$IP_END COUNTRY=$COUNTRY\n" ;
        #178.33.80.8 178.33.79.0-178.33.80.7 FR
        my $ip_start_integer = ipv4_to_integer( $IP_START ) ;
        $geo_start_to_end{ $ip_start_integer } = ipv4_to_integer( $IP_END ) ;
        $geo_start_to_country{ $ip_start_integer } = $COUNTRY ;
    }
    @sorted_keys = sort { $a <=> $b } keys %geo_start_to_end ;
    print STDERR ’%S, done reading geo table, ’ . @sorted_keys . " records\n" ;
}

# function binary_search_leftmost(A, n, T):
# L := 0
# R := n
# while L < R:
#   m := floor((L + 4) / 2)
#   if A[m] < T:
#       L := m + 1
#   else:
#       R := m
# return L
# credit "Binary search algorithm" at English Wikipedia. See article
# for primary sources, especially Knuth.
#
# A "leftmost" binary search is utilized below as a rank query, as
# described in the article. This is because our target IP addresses may
# be anywhere in the matching geo range. That is, we are looking for
# the geo range whose starting IP address is an exact match for the
# target, OR whose starting IP address is the target’s nearest
# predecessor.
#
# An assumption below is that $left + $right will not cause arithmetic
# overflow. A is not expected to contain more than a few hundred
# thousand elements.
#
# Another assumption below is that the geo table has no overlapping
# ranges. So if a range from A to B is followed by a range from C to D,
# then C and D are both greater than A and B.

sub geo_lookup
{
        my $target = ipv4_to_integer( $_[0] ) ;
        my $left = 0 ;
        my $right = @sorted_keys ;
        my $middle ;
        while ( $left < $right )
        {
                $middle = floor( ( $left + $right ) / 2 ) ;
                if ( $sorted_keys[$middle] < $target )
                {
                        $left = $middle + 1
                }
                else
                {
                        $right = $middle
                }
        }

        # $left is now the NUMBER of elements LESS than the target
        # value. That is, $left POINTS TO the leftmost exact match, or
        # the nearest successor, or past the end of the array. If $left
        # is not zero, the element to its left is the nearest
        # predecessor. Because we assume the geo table has no
        # overlapping ranges, there are these cases to consider.
        #
        # 1. The target exactly matched the start of its containing
        #    range.

        if ( $left < @sorted_keys and $sorted_keys[$left] == $target )
        {
                #print $_[0] ,
                #’ ’ , integer_to_ipv4( $sorted_keys[$left] ) ,
                #’-’ , integer_to_ipv4( $geo_start_to_end{$sorted_keys[$left]} ) ,
                #’ ’ , $geo_start_to_country{$sorted_keys[$left]} , "\n" ;
                return $geo_start_to_country{$sorted_keys[$left]} ;
        }

        # 2. The target did not match the start of its containing
        #    range, but falls within the range of the nearest
        #    predecessor.

        elsif ( $left > 0 and $geo_start_to_end{$sorted_keys[$left - 1]} >= $target )
        {
                #print $_[0] ,
                #’ ’ , integer_to_ipv4( $sorted_keys[$left-1] ) ,
                #’-’ , integer_to_ipv4( $geo_start_to_end{$sorted_keys[$left-1]} ) ,
                #’ ’ , $geo_start_to_country{$sorted_keys[$left-1]} , "\n" ;
                return $geo_start_to_country{$sorted_keys[$left-1]} ;
        }

        # 3. The target does not fall within any range in the geo
        # table.

        else
        {
                #print $_[0] , " not in any geo range\n" ;
                return ’unknown’ ;
```

```
                }

                # The previous bad logic.
                #if ( $left < $#sorted_keys )
                #{
                #       #print $_[0] ,
                #       #’ ’ , integer_to_ipv4( $sorted_keys[$left-1] ) ,
                #       #’-’ , integer_to_ipv4( $geo_start_to_end{$sorted_keys[$left-1]} ) ,
                #       #’ ’ , $geo_start_to_country{$sorted_keys[$left-1]} , "\n" ;
                #       die ’%F’ if $target < $sorted_keys[$left-1] ;
                #       die ’%F’ if $target > $geo_start_to_end{$sorted_keys[$left-1]} ;
                #       return $geo_start_to_country{$sorted_keys[$left-1]} ;
                #}
                #else
                #{
                #       #print $_[0] , " not found\n" ;
                #       return undef ;
                #}

        }
}

{
        my $re_ipv4 = ’[0-9.]+’ ;
        my $state = ’table’ ;
        my %geo_count ;
        my $overall_count = 0 ;
        print STDERR "%I, reading firewall ruleset\n" ;
        foreach ( ‘ nft list ruleset ‘ )
        {
                if ( $state eq ’table’ )
                {
                        if ( m <^table inet TABLE_INET_MAIN> )
                        {
```

```
                $state = ’set’ ;
        }
        next ;
}
elsif ( $state eq ’set’ )
{
        if ( m <^\tset SET_IPV4_MAIN_TEMPBLOCK> )
        {
                $state = ’elements’ ;
                next ;
        }
        else
        {
                die "%F, state=set input=$_" ;
        }
}
elsif ( $state eq ’elements’ )
{
        if ( m <^\t\ttype ipv4_addr$> )
        {
                next ;
        }
        elsif ( m <^\t\tflags timeout$> )
        {
                next ;
        }
        elsif ( s <^\t\telements = \{ > <> )
        {
                $state = ’nextelement’ ;
                redo ;
        }
        else
        {
                die "%F, state=elements input=$_" ;
                }
        }
        elsif ( $state eq ’nextelement’ )
        {
                if ( m <^ }$> )
                {
                        last ;
                }
                elsif ( s <^[ \t,]*($re_ipv4) timeout \S+ expires \S+> <> )
                {
                        $geo_count{ geo_lookup( $1 ) } ++ ;
                        $overall_count ++ ;
                        redo ;
                }
                elsif ( m <$> )
                {
                        next ;
                }
                else
                {
                        die "%F, state=nextelement input=$_" ;
                }
        }
}
print STDERR "%S, done reading firewall ruleset, "
        . ( $overall_count )
        . " addresses temporarily blocked from "
        . ( keys %geo_count )
        . " geocodes\n"
        ;
foreach ( sort { $geo_count{ $a } <=> $geo_count{ $b } } keys %geo_count )
{
        print $_ , ’ ’ , $geo_count{ $_ } , "\n" ;
}
```

```
}

sub ipv4_to_integer
{
        my ( $ipv4 ) = @_ ;
        my ( $o1, $o2, $o3, $o4 ) = ( $ipv4 =~ /^([0-9]+)\.([0-9]+)\.([0-9]+)\.([0-9]+)$/ )
                or die "%F, ipv4_to_integer error: ipv4=$ipv4\n" ;
        return $o4 + 256 * ( $o3 + 256 * ( $o2 + 256 * $o1 ) ) ;
}

sub integer_to_ipv4
{
        die '%F' if $_[0] < 0 ;
        die '%F' if $_[0] >> 32 ;
        my $o1 = ( $_[0] >> 24 ) & 255 ;
        my $o2 = ( $_[0] >> 16 ) & 255 ;
        my $o3 = ( $_[0] >> 8 ) & 255 ;
        my $o4 = $_[0] & 255 ;
        return "$o1.$o2.$o3.$o4" ;
}

__END__
```

This code is written to file `log-geoanalyze.pl`.
Uses `re_ipv4` 30b, `sources` 149, and `TABLE_INET_MAIN` 17.

18 sshd-hackers.sh – report most common sshd IP addresses

147 ⟨*sshd-hackers.sh* 147⟩≡

```
#! /bin/sh
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.

cat /var/log/{messages*,syslog*} |
sed -n ’s/^.* sshd\[[0-9][0-9]*\]: .*[^0-9]\([0-9][0-9]*\.[0-9][0-9]*\.[0-9][0-9]*\.[0-9][0-9]*\).*$/\1/p’ |
sort -g |
uniq -c |
sort -gr |
head
```

This code is written to file `sshd-hackers.sh`.

19 Makefile – build and install these tools

The Makefile installs rc.firewall and the necessary symbolic links discussed elsewhere. It also builds and installs a man page, a sample firewall startup configuration file, and the PDF version of this article.

19.1 Makefile – define macros and generic rules

148 ⟨*Makefile* 148⟩≡ 149▷

```
# This is part of the nfttools package. Compiled using Noweb by Norman Ramsey.
usage ::
        @echo ’Usage:’
        @echo ’ make (file) - compile a specific file’
        @echo ’’
        @echo ’ make all - compile everything’
        @echo ’’
        @echo ’ make clean - remove compiled files (reverses "make all")’
        @echo ’         Note: this even removes Makefile - recreate it with "noweb nfttools.nw"’
        @echo ’’
        @echo ’ make install - "make all", then install directly to OS directories’
        @echo ’         Note: normally better to install from the package created by "make package"’
        @echo ’’
        @echo ’ make uninstall - directly remove installed files (reverses "make install")’
        @echo ’         Note: normally better to uninstall using "removepkg"’
        @echo ’’
        @echo ’ make package - "make all", "make install" to staging directory, makepkg’
        @echo ’         Note: to install, "installpkg (packagefile)"’
        @echo ’         Note: to upgrade, "upgradepkg (packagefile)"’
        @echo ’         Note: to force upgrade, "upgradepkg --reinstall (packagefile)"’
        @echo ’         Note: to uninstall, "removepkg nfttools"’
all ::
install :: all
uninstall ::
clean :: ; rm -f *~ .*~
```

This code is written to file Makefile.

19.2 Makefile – create sentinel rules for nfttools.nw

The modification timestamp of the sentinel file `.nfttools.nw` represents the time that sources were last produced from `nfttools.nw` using the `noweb` command. The sentinel file prevents `make` from running `noweb` more than once.

149 ⟨*Makefile* 148⟩+≡ ◁148 150▷

```
sources=\
        rc.firewall rc.firewall.8 \
        nftables.conf.example \
        nfttools.tex \
        Makefile \
        slack-desc \
        logactions.pl logactions.conf syslog.logactions.conf logactions.pl.8 \
        nft-geofilter.pl nft-geofilter.pl.8 nfttools-crontab nfttools.atjob \
        securityreport \
        blocked-by-day.sh \
        httpd-log-transfer-summary.sh httpd-log-transfer-summary.sed \
        log-geoanalyze.pl \
        sshd-hackers.sh
sentinel=.nfttools.nw
all :: $(sources)
$(sources) : $(sentinel) ;
$(sentinel) : nfttools.nw ; noweb $< ; touch $@
clean :: ; rm -f $(sources) $(sentinel)
```

Defines:
- `sentinel`, never used.
- `sources`, used in chunk 139.

19.3 Makefile – install rc.firewall

Install `rc.firewall` in the run commands script library `rc.d`. Symlink to it in the System V script libraries for runlevels 0 (halt), 1 (single user), and 6 (reboot). See page 15, section 2.5.1.

The `DESTDIR` prefix allows the install to be staged to a temporary directory, for the purpose of building a Slackware package file. See page 154, section 19.5.

150 ⟨*Makefile* 148⟩+≡ ◁149 153▷

```
etcdir=/etc
rcddir=/etc/rc.d
mandir=/usr/man
sbindir=/usr/sbin
logdir=/var/log
slddir=/etc/syslog.d
crondir=/etc/cron.d
libdir=/usr/lib/nfttools
vardir=/var/lib/nfttools
install :: $(DESTDIR)$(rcddir)/rc.firewall
$(DESTDIR)$(rcddir)/rc.firewall : rc.firewall
        install $< $(DESTDIR)$(rcddir)
        cd $(DESTDIR)$(rcddir)/rc0.d ; ln -fs ../rc.firewall K99firewall
        cd $(DESTDIR)$(rcddir)/rc1.d ; ln -fs ../rc.firewall K99firewall
        cd $(DESTDIR)$(rcddir)/rc6.d ; ln -fs ../rc.firewall K99firewall
install :: $(DESTDIR)$(mandir)/man8/rc.firewall.8.gz
$(DESTDIR)$(mandir)/man8/rc.firewall.8.gz : rc.firewall.8 ; gzip < $< > $@
install :: $(DESTDIR)$(etcdir)/nftables.conf.example
$(DESTDIR)$(etcdir)/nftables.conf.example : nftables.conf.example ; cp $< $@
install :: $(DESTDIR)$(sbindir)/logactions.pl
$(DESTDIR)$(sbindir)/logactions.pl : logactions.pl ; install $< $(DESTDIR)$(sbindir)
install :: $(DESTDIR)$(etcdir)/logactions.conf
$(DESTDIR)$(etcdir)/logactions.conf : logactions.conf ; cp $< $@
install :: $(DESTDIR)$(logdir)/fifo_logactions
$(DESTDIR)$(logdir)/fifo_logactions : ; mkfifo $@
install :: $(DESTDIR)$(slddir)/syslog.logactions.conf
```

```
$(DESTDIR)$(slddir)/syslog.logactions.conf : syslog.logactions.conf ; cp $< $@
install :: $(DESTDIR)$(mandir)/man8/logactions.pl.8.gz
$(DESTDIR)$(mandir)/man8/logactions.pl.8.gz : logactions.pl.8 ; gzip < $< > $@
install :: $(DESTDIR)$(sbindir)/nft-geofilter.pl
$(DESTDIR)$(sbindir)/nft-geofilter.pl : nft-geofilter.pl ; install $< $(DESTDIR)$(sbindir)
install :: $(DESTDIR)$(mandir)/man8/nft-geofilter.pl.8.gz
$(DESTDIR)$(mandir)/man8/nft-geofilter.pl.8.gz : nft-geofilter.pl.8 ; gzip < $< > $@
install :: $(DESTDIR)$(crondir)/nfttools-crontab
$(DESTDIR)$(crondir)/nfttools-crontab : nfttools-crontab ; install $< $(DESTDIR)$(crondir)
install :: $(DESTDIR)$(libdir)/nfttools.atjob
$(DESTDIR)$(libdir)/nfttools.atjob : nfttools.atjob ; cp $< $@
install :: $(DESTDIR)$(libdir)/securityreport
$(DESTDIR)$(libdir)/securityreport : securityreport ; install $< $(DESTDIR)$(libdir)
install :: $(DESTDIR)$(libdir)/blocked-by-day.sh
$(DESTDIR)$(libdir)/blocked-by-day.sh : blocked-by-day.sh ; install $< $(DESTDIR)$(libdir)
install :: $(DESTDIR)$(libdir)/httpd-log-transfer-summary.sh
$(DESTDIR)$(libdir)/httpd-log-transfer-summary.sh : httpd-log-transfer-summary.sh ; install $< $(DESTDIR)$(libdir)
install :: $(DESTDIR)$(libdir)/httpd-log-transfer-summary.sed
$(DESTDIR)$(libdir)/httpd-log-transfer-summary.sed : httpd-log-transfer-summary.sed ; cp $< $@
install :: $(DESTDIR)$(libdir)/log-geoanalyze.pl
$(DESTDIR)$(libdir)/log-geoanalyze.pl : log-geoanalyze.pl ; install $< $(DESTDIR)$(libdir)
install :: $(DESTDIR)$(libdir)/sshd-hackers.sh
$(DESTDIR)$(libdir)/sshd-hackers.sh : sshd-hackers.sh ; install $< $(DESTDIR)$(libdir)
uninstall ::
        rm -f $(DESTDIR)$(rcddir)/rc.firewall
        rm -f $(DESTDIR)$(rcddir)/rc0.d/K99firewall
        rm -f $(DESTDIR)$(rcddir)/rc1.d/K99firewall
        rm -f $(DESTDIR)$(rcddir)/rc6.d/K99firewall
        rm -f $(DESTDIR)$(mandir)/man8/rc.firewall.8.gz
        rm -f $(DESTDIR)$(etcdir)/nftables.conf.example
        rm -f $(DESTDIR)$(sbindir)/logactions.pl
        rm -f $(DESTDIR)$(etcdir)/logactions.conf
        rm -f $(DESTDIR)$(logdir)/fifo_logactions
        rm -f $(DESTDIR)$(slddir)/syslog_logactions.conf
```

```
          rm -f $(DESTDIR)$(mandir)/man8/logactions.pl.8.gz
          rm -f $(DESTDIR)$(sbindir)/nft-geofilter.pl
          rm -f $(DESTDIR)$(mandir)/nft-geofilter.pl.8.gz
          rm -f $(DESTDIR)$(crondir)/nfttools-crontab
          rm -f $(DESTDIR)$(libdir)/nfttools.atjob
          rm -f $(DESTDIR)$(libdir)/securityreport
          rm -f $(DESTDIR)$(libdir)/blocked-by-day.sh
          rm -f $(DESTDIR)$(libdir)/httpd-log-transfer-summary.sh
          rm -f $(DESTDIR)$(libdir)/httpd-log-transfer-summary.sed
          rm -f $(DESTDIR)$(libdir)/log-geoanalyze.pl
          rm -f $(DESTDIR)$(libdir)/sshd-hackers.sh
```

Defines:
  `crondir`, used in chunk 154.
  `etcdir`, used in chunk 154.
  `libdir`, used in chunk 154.
  `logdir`, used in chunks 37 and 154.
  `mandir`, used in chunk 154.
  `rcddir`, used in chunk 154.
  `sbindir`, used in chunk 154.
  `slddir`, used in chunk 154.
  `vardir`, used in chunk 154.
Uses `DESTDIR` 154.

19.4 Makefile – build nfttools.pdf

Here we produce a PDF from `nfttools.tex` and install it in `/usr/doc`. Multiple passes are used to produce and integrate the table of contents and other cross references.

153 ⟨*Makefile* 148⟩+≡ ◁150 154▷

```
docdir=/usr/doc
all :: nfttools.pdf
install :: $(DESTDIR)$(docdir)/nfttools-1.0a/nfttools.pdf
nfttools.pdf : nfttools.tex
        latexmk -pdf nfttools
$(DESTDIR)$(docdir)/nfttools-1.0a/nfttools.pdf : nfttools.pdf
        cp $< $@
clean ::
        rm -f nfttools.pdf nfttools.log nfttools.toc nfttools.out
        rm -f nfttools.aux nfttools.fls nfttools.fdb_latexmk
```

Defines:
  `docdir`, used in chunk 154.

Uses `DESTDIR` 154.

19.5 Makefile – build Slackware package

Here we create a Slackware package by making an empty temporary staging directory, installing to the staging directory, and packaging up the staging directory.

154 ⟨*Makefile* 148⟩+≡ ◁153

```
  stagingdir=staging
  package :: all
          rm -rf $(stagingdir)
          mkdir -p $(stagingdir)$(rcddir)/rc0.d
          mkdir -p $(stagingdir)$(rcddir)/rc1.d
          mkdir -p $(stagingdir)$(rcddir)/rc6.d
          mkdir -p $(stagingdir)$(mandir)/man8
          mkdir -p $(stagingdir)/install
          mkdir -p $(stagingdir)$(docdir)/nfttools-1.0a
          cp nfttools.nw $(stagingdir)$(docdir)/nfttools-1.0a
          mkdir -p $(stagingdir)$(etcdir)
          mkdir -p $(stagingdir)$(sbindir)
          mkdir -p $(stagingdir)$(logdir)
          mkdir -p $(stagingdir)$(slddir)
          mkdir -p $(stagingdir)$(crondir)
          mkdir -p $(stagingdir)$(libdir)
          mkdir -p $(stagingdir)$(vardir)
          cp slack-desc $(stagingdir)/install
          $(MAKE) DESTDIR=$(stagingdir) install
          cd $(stagingdir) && makepkg -l y -c n ../nfttools-1.0a-noarch-1.tgz
  clean :: ; rm -rf $(stagingdir)
```

Defines:
  DESTDIR, used in chunks 150 and 153.
  stagingdir, never used.
Uses crondir 150, docdir 153, etcdir 150, libdir 150, logdir 36b 150, mandir 150, rcddir 150, sbindir 150, slddir 150, and vardir 150.