A look at the *ferm* firewall

NFWS 2017 - Faro, Portugal

Pablo Neira Ayuso <pablo@netfilter.org>

Introduction

- Project started in 2001.
- Written in Perl.
- People seem to ♥ this...
- Wrapper for iptables...
  - Better syntax, no more dash dash
  - “Interactive” mode
    - Try this ruleset, if no confirmation after 30s, revert it.
  - Scripting capabilities
- Source of inspiration for nft...
  - Disclaimer: Not proposing to add all features there into nft! ;-)

Syntax

- Well-structured syntax. Example:

```
chain INPUT {
    policy DROP;
    mod state state (RELATED ESTABLISHED) ACCEPT;
    proto tcp dport (http ftp ssh) ACCEPT;
}
```

- Parens, list of space-separated items, for pseudo-sets.
  - Expand in several iptables rules!
- Map 1:1 to iptables native extensions
  - Iptables matches start by 'mod'
  - Also integrates with ipset, via explicit 'mod set'.
- Targets are expressed in uppercase.
- Lines end by semicolon.

“Interactive mode”

- Test the rules without fearing to lock yourself out.

  --interactive … --timeout

  Apply the firewall rules and ask the user for confirmation. Reverts to the previous ruleset if there is no valid user response within 30 seconds (see –timeout).

  --[Extracted from ferm documentation]

Variable definitions

- $ identifies variable.

```
@def $DEV_INTERNET = eth0;
@def $PORTS = (http ftp);
@def $MORE_PORTS = ($PORTS 8080);
```

- Define variable from command line call:

```
ferm --def '$name=value' …
```

  - Define some 'extern' keyword for variables in nft?

Scripting capabilities (2)

- Allows rule grouping by match selector (aka. “block of rules”).

```
@def $DEV_INTERNET = eth1;

chain INPUT {
        proto tcp {
            @def $DEV_INTERNET = ppp0;
            interface $DEV_INTERNET dport http ACCEPT;
        }
        # for Ipsec
        interface $DEV_WORLD {
            proto udp dport 500 ACCEPT;
            proto (esp ah) ACCEPT;
        }
        interface $DEV_INTERNET DROP;
}
```

Functions

- Actually work like macro? Name starts by &.
- “Functions” take parameters.
- Example 1:

  @def &FOO() = proto (tcp udp) dport domain;

  &FOO() ACCEPT;
- Example 2:

```
@def &TCP_TUNNEL($port, $dest) = {
        table filter chain FORWARD interface ppp0 proto tcp dport $port daddr
$dest outerface eth0 ACCEPT;
        table nat chain PREROUTING interface ppp0 proto tcp dport $port daddr
1.2.3.4 DNAT to $dest;
    }
```

&TCP_TUNNEL((ssh smtp), 192.168.1.2);

External command invocations

- Aka. Backticks.
- Plug output of external command.
  - Uses /bin/sh as default shell.

```
@def $DNSSERVERS = `grep nameserver /etc/resolv.conf |
awk '{print $2}'`;

chain INPUT proto tcp saddr $DNSSERVER ACCEPT;
```

Include files

- Include another file, eg.

  @include 'vars.ferm';

- Include directory
  - sorted alphabetically
- Globbing:

  @include @glob('*.include');

- “Trailing pipe”, fails is exit code non-zero.
  - @include "/root/generate_ferm_rules.sh $HOSTNAME|"

Conditionals

- Conditional macro-like expansion... ☠?

```
@if $condition {
   MARK set-mark 2;
   RETURN;
} @else {
    MARK set-mark 3;
}
```

Built-in functions

- @defined($name), @defined($name)

  @def $a = 'foo';

  @if @defined($a) good;

- @eq(a,b), @ne(a,b), @not(x)

  @if @eq($DOMAIN, ip6) DROP;

- @resolve((hostname1 hostname2 ...), [type]) ☠

  saddr @resolve(my.host.foo) proto tcp dport ssh ACCEPT;

- String handling:
  - @cat(a, b, …), @substr(expression, offset, length), @length(expression)
  - Paths: @basename(path), @dirname(path), @glob(path)
- Filter out things that don't match current domain:
  - @ipfilter(list)

A look at the *ferm* firewall

NFWS 2017 - Faro, Portugal

Pablo Neira Ayuso <pablo@netfilter.org>