Building Advanced Firewall

Overview

From everything we have learned so far, let's try to build an advanced firewall. In this firewall building example, we will try to use as many firewall features as we can to illustrate how they work and when they should be used the right way.

Most of the filtering will be done in the RAW firewall, a regular firewall will contain just a basic rule set to accept `established, related`, and untracked connections as well as dropping everything else not coming from LAN to fully protect the router.

Interface Lists

Two interface lists will be used **WAN** and **LAN** for easier future management purposes. Interfaces connected to the global internet should be added to the WAN list, in this case, it is *ether1*!

```
/interface list
  add comment=defconf name=WAN
  add comment=defconf name=LAN
/interface list member
  add comment=defconf interface=bridge list=LAN
  add comment=defconf interface=ether1 list=WAN
```

Protect the Device

The main goal here is to allow access to the router only from LAN and drop everything else.

Notice that ICMP is accepted here as well, it is used to accept ICMP packets that passed RAW rules.

```
/ip firewall filter
  add action=accept chain=input comment="defconf: accept ICMP after RAW" protocol=icmp
  add action=accept chain=input comment="defconf: accept established,related,untracked" connection-
state=established,related,untracked
  add action=drop chain=input comment="defconf: drop all not coming from LAN" in-interface-list=!LAN
```

IPv6 part is a bit more complicated, in addition, UDP traceroute, DHCPv6 client PD, and IPSec (IKE, AH, ESP) are accepted as per RFC recommendations.

```
/ipv6 firewall filter
add action=accept chain=input comment="defconf: accept ICMPv6 after RAW" protocol=icmpv6
add action=accept chain=input comment="defconf: accept established,related,untracked" connection-
state=established,related,untracked
add action=accept chain=input comment="defconf: accept UDP traceroute" dst-port=33434-33534 protocol=udp
add action=accept chain=input comment="defconf: accept DHCPv6-Client prefix delegation." dst-port=546
protocol=udp src-address=fe80::/10
add action=accept chain=input comment="defconf: accept IKE" dst-port=500,4500 protocol=udp
add action=accept chain=input comment="defconf: accept IPSec AH" protocol=ipsec-ah
add action=accept chain=input comment="defconf: accept IPSec ESP" protocol=ipsec-esp
add action=drop chain=input comment="defconf: drop all not coming from LAN" in-interface-list=!LAN
```

⚠ In certain setups where the DHCPv6 relay is used, the src address of the packets may not be from the link-local range. In that case, the src-address parameter of rule #4 must be removed or adjusted to accept the relay address.

Protect the Clients

Before the actual set of rules, let's create a necessary `address-list` that contains all IPv4/6 addresses that cannot be forwarded.

Notice that in this list multicast address range is added. It is there because in most cases multicast is not used. If you intend to use multicast forwarding, then this address list entry should be disabled.

```
/ip firewall address-list
  add address=0.0.0.0/8 comment="defconf: RFC6890" list=no_forward_ipv4
  add address=169.254.0.0/16 comment="defconf: RFC6890" list=no_forward_ipv4
  add address=224.0.0.0/4 comment="defconf: multicast" list=no_forward_ipv4
  add address=255.255.255.255/32 comment="defconf: RFC6890" list=no_forward_ipv4
```

In the same case for IPv6, if multicast forwarding is used then the multicast entry should be disabled from the *address-list.*

```
/ipv6 firewall address-list
  add address=fe80::/10  comment="defconf: RFC6890 Linked-Scoped Unicast" list=no_forward_ipv6
  add address=ff00::/8  comment="defconf: multicast" list=no_forward_ipv6
```

`Forward` chain will have a bit more rules than input:

- accept *established, related* and *untracked* connections;
- FastTrack *established* and *related* connections (currently only IPv4);
- drop *invalid* connections;
- drop bad forward IPs, since we cannot reliably determine in RAW chains which packets are forwarded
- drop connections initiated from the internet (from the WAN side which is not destination NAT`ed);
- drop bogon IPs that should not be forwarded.

We are dropping all non-dstnated IPv4 packets to protect direct attacks on the clients if the attacker knows the internal LAN network. Typically this rule would not be necessary since RAW filters will drop such packets, however, the rule is there for double security in case RAW rules are accidentally messed up.

```
/ip firewall filter
  add action=accept chain=forward comment="defconf: accept all that matches IPSec policy" ipsec-policy=in,ipsec
disabled=yes
  add action=fasttrack-connection chain=forward comment="defconf: fasttrack" connection-state=established,
related
  add action=accept chain=forward comment="defconf: accept established,related, untracked" connection-
state=established,related,untracked
  add action=drop chain=forward comment="defconf: drop invalid" connection-state=invalid
  add action=drop chain=forward comment="defconf:  drop all from WAN not DSTNATed" connection-nat-state=!dstnat
connection-state=new in-interface-list=WAN
  add action=drop chain=forward src-address-list=no_forward_ipv4 comment="defconf: drop bad forward IPs"
  add action=drop chain=forward dst-address-list=no_forward_ipv4 comment="defconf: drop bad forward IPs"
```

IPv6 `forward` chain is very similar, except that IPsec and HIP are accepted as per RFC recommendations, and ICMPv6 with `hop-limit=1` is dropped.

```
/ipv6 firewall filter
add action=accept chain=forward comment="defconf: accept established,related,untracked" connection-
state=established,related,untracked
add action=drop chain=forward comment="defconf: drop invalid" connection-state=invalid
add action=drop chain=forward src-address-list=no_forward_ipv6 comment="defconf: drop bad forward IPs"
add action=drop chain=forward dst-address-list=no_forward_ipv6 comment="defconf: drop bad forward IPs"
add action=drop chain=forward comment="defconf: rfc4890 drop hop-limit=1" hop-limit=equal:1 protocol=icmpv6
add action=accept chain=forward comment="defconf: accept ICMPv6 after RAW" protocol=icmpv6
add action=accept chain=forward comment="defconf: accept HIP" protocol=139
add action=accept chain=forward comment="defconf: accept IKE" protocol=udp dst-port=500,4500
add action=accept chain=forward comment="defconf: accept AH" protocol=ipsec-ah
add action=accept chain=forward comment="defconf: accept ESP" protocol=ipsec-esp
add action=accept chain=forward comment="defconf: accept all that matches IPSec policy" ipsec-policy=in,ipsec
add action=drop chain=forward comment="defconf: drop everything else not coming from LAN" in-interface-list=!LAN
```

Notice the IPsec policy matcher rules. It is very important that IPsec encapsulated traffic bypass fast-track. That is why as an illustration we have added a disabled rule to accept traffic matching IPsec policies. Whenever IPsec tunnels are used on the router this rule should be enabled. For IPv6 it is much more simple since it does not have fast-track support.

Another approach to solving the IPsec problem is to add RAW rules, we will talk about this method later in the RAW section

Masquerade Local Network

For local devices behind the router to be able to access the internet, local networks must be masqueraded. In most cases, it is advised to use src-nat instead of masquerade, however in this case when the WAN address is dynamic it is the only option.

```
/ip firewall nat
  add action=accept chain=srcnat comment="defconf: accept all that matches IPSec policy" ipsec-policy=out,ipsec
disabled=yes
  add action=masquerade chain=srcnat comment="defconf: masquerade" out-interface-list=WAN
```

Notice the disabled policy matcher rule, the same as in firewall filters IPSec traffic must be excluded from being NATed (except in specific scenarios where IPsec policy is configured to match NAT`ed address). So whenever IPsec tunnels are used on the router this rule must be enabled.

RAW Filtering

IPv4 Address Lists

Before setting RAW rules, let's create some address lists necessary for our filtering policy. RFC 6890 will be used as a reference.

First, *address-list* contains all IPv4 addresses that cannot be used as src/dst/forwarded, etc. (will be dropped immediately if such address is seen)

```
/ip firewall address-list
  add address=127.0.0.0/8 comment="defconf: RFC6890" list=bad_ipv4
  add address=192.0.0.0/24 comment="defconf: RFC6890" list=bad_ipv4
  add address=192.0.2.0/24 comment="defconf: RFC6890 documentation" list=bad_ipv4
  add address=198.51.100.0/24 comment="defconf: RFC6890 documentation" list=bad_ipv4
  add address=203.0.113.0/24 comment="defconf: RFC6890 documentation" list=bad_ipv4
  add address=240.0.0.0/4 comment="defconf: RFC6890 reserved" list=bad_ipv4
```

Another address list contains all IPv4 addresses that cannot be routed globally.

```
/ip firewall address-list
  add address=0.0.0.0/8 comment="defconf: RFC6890" list=not_global_ipv4
  add address=10.0.0.0/8 comment="defconf: RFC6890" list=not_global_ipv4
  add address=100.64.0.0/10 comment="defconf: RFC6890" list=not_global_ipv4
  add address=169.254.0.0/16 comment="defconf: RFC6890" list=not_global_ipv4
  add address=172.16.0.0/12 comment="defconf: RFC6890" list=not_global_ipv4
  add address=192.0.0.0/29 comment="defconf: RFC6890" list=not_global_ipv4
  add address=192.168.0.0/16 comment="defconf: RFC6890" list=not_global_ipv4
  add address=198.18.0.0/15 comment="defconf: RFC6890 benchmark" list=not_global_ipv4
  add address=255.255.255.255/32 comment="defconf: RFC6890" list=not_global_ipv4
```

And last two address lists for addresses that cannot be as destination or source address.

```
/ip firewall address-list
  add address=224.0.0.0/4 comment="defconf: multicast" list=bad_src_ipv4
  add address=255.255.255.255/32 comment="defconf: RFC6890" list=bad_src_ipv4
add address=0.0.0.0/8 comment="defconf: RFC6890" list=bad_dst_ipv4
  add address=224.0.0.0/4 comment="defconf: RFC6890" list=bad_dst_ipv4
```

IPv4 RAW Rules

Raw IPv4 rules will perform the following actions:

- **add disabled "accept" rule** - can be used to quickly disable RAW filtering without disabling all RAW rules;
- **accept** DHCP discovery - most of the DHCP packets are not seen by an IP firewall, but some of them are, so make sure that they are accepted;
- **drop** packets that use bogon IPs;
- **drop** from invalid SRC and DST IPs;
- **drop** globally unroutable IPs coming from WAN;
- **drop** packets with source-address not equal to 192.168.88.0/24 (default IP range) coming from LAN;
- **drop** packets coming from WAN to be forwarded to 192.168.88.0/24 network, this will protect from attacks if the attacker knows the internal network;
- **drop** bad ICMP, UDP, and TCP;
- **accept** everything else coming from WAN and LAN;
- **accept** local traffic between router interfaces;
- **drop** everything else, to make sure that any newly added interface (like PPPoE connection to service provider) is protected against accidental misconfiguration.

```
/ip firewall raw
add action=accept chain=prerouting comment="defconf: enable for transparent firewall" disabled=yes
add action=accept chain=prerouting comment="defconf: accept DHCP discover" dst-address=255.255.255.255 dst-
port=67 in-interface-list=LAN protocol=udp src-address=0.0.0.0 src-port=68
add action=drop chain=prerouting comment="defconf: drop bogon IP's" src-address-list=bad_ipv4
add action=drop chain=prerouting comment="defconf: drop bogon IP's" dst-address-list=bad_ipv4
add action=drop chain=prerouting comment="defconf: drop bogon IP's" src-address-list=bad_src_ipv4
add action=drop chain=prerouting comment="defconf: drop bogon IP's" dst-address-list=bad_dst_ipv4
add action=drop chain=prerouting comment="defconf: drop non global from WAN" src-address-list=not_global_ipv4
in-interface-list=WAN
add action=drop chain=prerouting comment="defconf: drop forward to local lan from WAN" in-interface-list=WAN
dst-address=192.168.88.0/24
add action=drop chain=prerouting comment="defconf: drop local if not from default IP range" in-interface-
list=LAN src-address=!192.168.88.0/24
add action=drop chain=prerouting comment="defconf: drop bad UDP" port=0 protocol=udp
add action=jump chain=prerouting comment="defconf: jump to ICMP chain" jump-target=icmp4 protocol=icmp
add action=jump chain=prerouting comment="defconf: jump to TCP chain" jump-target=bad_tcp protocol=tcp
add action=accept chain=prerouting comment="defconf: accept everything else from LAN" in-interface-list=LAN
add action=accept chain=prerouting comment="defconf: accept everything else from WAN" in-interface-list=WAN
add action=accept chain=prerouting comment="defconf: accept local traffic between router interfaces" src-
address-type=local
add action=drop chain=prerouting comment="defconf: drop the rest"
```

Notice that we used some optional chains, the first **TCP** chain to drop **TCP** packets known to be *invalid*.

```
/ip firewall raw
add action=drop chain=bad_tcp comment="defconf: TCP flag filter" protocol=tcp tcp-flags=!fin,!syn,!rst,!ack
add action=drop chain=bad_tcp comment=defconf protocol=tcp tcp-flags=fin,syn
add action=drop chain=bad_tcp comment=defconf protocol=tcp tcp-flags=fin,rst
add action=drop chain=bad_tcp comment=defconf protocol=tcp tcp-flags=fin,!ack
add action=drop chain=bad_tcp comment=defconf protocol=tcp tcp-flags=fin,urg
add action=drop chain=bad_tcp comment=defconf protocol=tcp tcp-flags=syn,rst
add action=drop chain=bad_tcp comment=defconf protocol=tcp tcp-flags=rst,urg
add action=drop chain=bad_tcp comment="defconf: TCP port 0 drop" port=0 protocol=tcp
```

And another chain for **ICMP**. Note that if you want a very strict firewall then such strict **ICMP** filtering can be used, but in most cases, it is not necessary and simply adds more load on the router's CPU. ICMP rate limit in most cases is also unnecessary since the Linux kernel is already limiting ICMP packets to 100pps.

```
/ip firewall raw
add action=accept chain=icmp4 comment="defconf: echo reply" icmp-options=0:0 limit=5,10:packet protocol=icmp
add action=accept chain=icmp4 comment="defconf: net unreachable" icmp-options=3:0 protocol=icmp
add action=accept chain=icmp4 comment="defconf: host unreachable" icmp-options=3:1 protocol=icmp
add action=accept chain=icmp4 comment="defconf: protocol unreachable" icmp-options=3:2 protocol=icmp
add action=accept chain=icmp4 comment="defconf: port unreachable" icmp-options=3:3 protocol=icmp
add action=accept chain=icmp4 comment="defconf: fragmentation needed" icmp-options=3:4 protocol=icmp
add action=accept chain=icmp4 comment="defconf: echo" icmp-options=8:0 limit=5,10:packet protocol=icmp
add action=accept chain=icmp4 comment="defconf: time exceeded " icmp-options=11:0-255 protocol=icmp
add action=drop chain=icmp4 comment="defconf: drop other icmp" protocol=icmp
```

IPv6 Address Lists

List of IPv6 addresses that should be dropped instantly

```
/ipv6 firewall address-list
add address=::1/128 comment="defconf: RFC6890 lo" list=bad_ipv6
add address=::ffff:0:0/96 comment="defconf: RFC6890 IPv4 mapped" list=bad_ipv6
add address=2001::/23 comment="defconf: RFC6890" list=bad_ipv6
add address=2001:db8::/32 comment="defconf: RFC6890 documentation" list=bad_ipv6
add address=2001:10::/28 comment="defconf: RFC6890 orchid" list=bad_ipv6
add address=::/96 comment="defconf: ipv4 compat" list=bad_ipv6
```

List of IPv6 addresses that are not globally routable

```
/ipv6 firewall address-list
add address=100::/64 comment="defconf: RFC6890 Discard-only" list=not_global_ipv6
add address=2001::/32 comment="defconf: RFC6890 TEREDO" list=not_global_ipv6
add address=2001:2::/48 comment="defconf: RFC6890 Benchmark" list=not_global_ipv6
add address=fc00::/7 comment="defconf: RFC6890 Unique-Local" list=not_global_ipv6
```

List of addresses as an invalid destination address

```
/ipv6 firewall address-list add address=::/128 comment="defconf: unspecified" list=bad_dst_ipv6
```

List of addresses as an invalid source address

```
/ipv6 firewall address-list
  add address=::/128 comment="defconf: unspecified" list=bad_src_ipv6
  add address=ff00::/8  comment="defconf: multicast" list=bad_src_ipv6
```

IPv6 RAW Rules

Raw IPv6 rules will perform the following actions:

- **add disabled accept rule** - can be used to quickly disable RAW filtering without disabling all RAW rules;
- **drop** packets that use bogon IPs;
- **drop** from invalid SRC and DST IPs;
- **drop** globally unroutable IPs coming from WAN;
- **drop** bad ICMP;
- **accept** everything else coming from WAN and LAN;
- **drop** everything else, to make sure that any newly added interface (like PPPoE connection to service provider) is protected against accidental misconfiguration.

```
/ipv6 firewall raw
add action=accept chain=prerouting comment="defconf: enable for transparent firewall" disabled=yes
add action=accept chain=prerouting comment="defconf: RFC4291, section 2.7.1" src-address=::/128 dst-
address=ff02:0:0:0:0:1:ff00::/104 icmp-options=135 protocol=icmpv6
add action=drop chain=prerouting comment="defconf: drop bogon IP's" src-address-list=bad_ipv6
add action=drop chain=prerouting comment="defconf: drop bogon IP's" dst-address-list=bad_ipv6
add action=drop chain=prerouting comment="defconf: drop packets with bad SRC ipv6" src-address-list=bad_src_ipv6
add action=drop chain=prerouting comment="defconf: drop packets with bad dst ipv6" dst-address-list=bad_dst_ipv6
add action=drop chain=prerouting comment="defconf: drop non global from WAN" src-address-list=not_global_ipv6
in-interface-list=WAN
add action=jump chain=prerouting comment="defconf: jump to ICMPv6 chain" jump-target=icmp6 protocol=icmpv6
add action=accept chain=prerouting comment="defconf: accept local multicast scope" dst-address=ff02::/16
add action=drop chain=prerouting comment="defconf: drop other multicast destinations" dst-address=ff00::/8
add action=accept chain=prerouting comment="defconf: accept everything else from WAN" in-interface-list=WAN
add action=accept chain=prerouting comment="defconf: accept everything else from LAN" in-interface-list=LAN
add action=drop chain=prerouting comment="defconf: drop the rest"
```

Notice that the optional **ICMP** chain was used. If you want a very strict firewall then such strict **ICMP** filtering can be used, but in most cases, it is not necessary and simply adds more load on the router's CPU. ICMP rate limit in most cases is also unnecessary since the Linux kernel is already limiting ICMP packets to 100pps

```
/ipv6 firewall raw
# Be aware that different operating systems originate packets with different default TTL values
add action=drop chain=icmp6 comment="defconf: rfc4890 drop ll if hop-limit!=255" dst-address=fe80::/10 hop-
limit=not-equal:255 protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: dst unreachable" icmp-options=1:0-255 protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: packet too big" icmp-options=2:0-255 protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: limit exceeded" icmp-options=3:0-1 protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: bad header" icmp-options=4:0-2 protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: Mobile home agent address discovery" icmp-options=144:0-255
protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: Mobile home agent address discovery" icmp-options=145:0-255
protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: Mobile prefix solic" icmp-options=146:0-255 protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: Mobile prefix advert" icmp-options=147:0-255 protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: echo request limit 5,10" icmp-options=128:0-255 limit=5,10:
packet protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: echo reply limit 5,10" icmp-options=129:0-255 limit=5,10:packet
protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: rfc4890 router solic limit 5,10 only LAN" hop-limit=equal:255
icmp-options=133:0-255 in-interface-list=LAN limit=5,10:packet protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: rfc4890 router advert limit 5,10 only LAN" hop-limit=equal:255
icmp-options=134:0-255 in-interface-list=LAN limit=5,10:packet protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: rfc4890 neighbor solic limit 5,10 only LAN" hop-limit=equal:255
icmp-options=135:0-255 in-interface-list=LAN limit=5,10:packet protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: rfc4890 neighbor advert limit 5,10 only LAN" hop-limit=equal:
255 icmp-options=136:0-255 in-interface-list=LAN limit=5,10:packet protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: rfc4890 inverse ND solic limit 5,10 only LAN" hop-limit=equal:
255 icmp-options=141:0-255 in-interface-list=LAN limit=5,10:packet protocol=icmpv6
add action=accept chain=icmp6 comment="defconf: rfc4890 inverse ND advert limit 5,10 only LAN" hop-limit=equal:
255 icmp-options=142:0-255 in-interface-list=LAN limit=5,10:packet protocol=icmpv6
add action=drop chain=icmp6 comment="defconf: drop other icmp" protocol=icmpv6
```