Firewalling

David Malone

<dwmalone@maths.tcd.ie>

<David.Malone@nuim.ie>

10 June 2008

Introduction

- IPv6 Firewalling/Packet filtering.
- Ideas very similar to IPv4.
- First, cover some basics.
- Then, similarities and differences.
- Finally cover some examples.

Special address blocks

- ::1 — localhost.
  Just like 127.0.0.1.
- ::0.0.0.0/96 — (old) compatible.
  Was used to indicate tunnel.
- ::ffff:0.0.0.0/96 — mapped.
  Used in programs to represent IPv4.
- fe80::/10 — link local.
  Used extensively internally.

- fec0::/10 — (old) site local.
  RFC 1918ish, deprecated.
- fc00::/7 — Local unique unicast.
  New private addresses.
- ff00::/8 — multicast.
  No broadcast. Scoped.

Transition Mechanisms

- Dual stack
  Host speaks IPv4 and IPv6.
- (configured point2point) Tunnels
  Running IPv6 over IPv4.
- 6to4
  Automatic tunneling scheme.
- Teredo
  Tunneling through NAT over UDP.

Packet Filtering

Choose to allow or deny packets:

- TCP/UDP/ICMP.
- Port numbers/ICMP type.
- IP addresses.
- In/out which interface.
- TCP flags, sequence numbers, ...
- IP fragmentation/offset.
- Remembering state.

```
# Allow access to your DNS
add permit tcp from any to $ip6 53 setup
add permit udp from any 53 to $ip6
add permit udp from $ip6 to any 53
# Allow access to your website
add permit tcp from any to $ip6 80 setup
```

Addresses

- Use IPv6 not IPv4 address.
- Hardwiring autoconf and privacy addresses.
- (Best to filter per-subnet?)
- More special addresses: localhost, link-local, site-local, LUA, compatible, mapped.
- Multiple addresses (also 6to4/Teredo).
- Ingress/egress filtering (BCP 38).

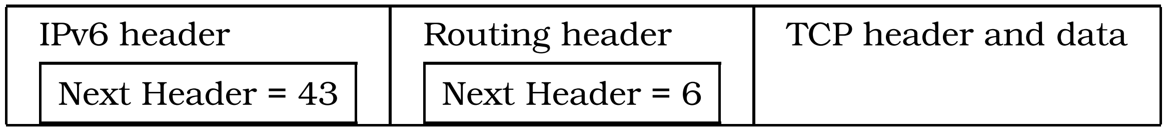
```
# Filter localhost
permit ipv6 from ::1 to any via lo0
permit ipv6 from any to ::1 via lo0
deny ipv6 from ::1 to any
deny ipv6 from any to ::1

# Block mapped addresses on the wire.
deny ipv6 from ::ffff:0.0.0.0/96 to any
deny ipv6 from any to ::ffff:0.0.0.0/96

# Denying automatically tunnelled traffic
deny ipv6 from ::0.0.0.0/96 to any
deny ipv6 from any to ::0.0.0.0/96
```

Filtering ICMP

- Remember no ARP.
- Or in-network fragmentation.
- **Allow neighbour discovery and PMTU discovery.**
- (Except on tunnels?)
- Cisco now implicitly allow ND.
- Unreachable (and other errors) for fast fallback.
- Make choice for each type.

```
# (DAD)
permit icmpv6 from ::  to ff02::/16
# for NA, NS, RA and RS messages
permit icmpv6 from fe80::/10 to fe80::/10
permit icmpv6 from fe80::/10 to ff02::/16
# allow PMTUD
permit icmpv6 from any to any icmptypes
packet-too-big
```

Extended Headers



- Headers for new features.
- Needed for mobility.
- Probably best to follow vendor advice.
- RH0 problem last summer.

Side Points

- Tunneling over IPv4? Protocol 41.
- Tunneling over other things.
- Common rules or separate?
- Portscanning much harder, discovery still possible.
- Keep an eye out for software ACLs.
- Home networks and NAT.

Cisco

- *Implementing Security for IPv6* quite good.
- Features gradually expanding.
- Numbered in 200–299 range.
- Some feature quirks.

```
ipv6 access-list telnet-vty
 permit ipv6 2001:db8:18::/48 any
 permit ipv6 2001:db8:8::/48 any
 permit ipv6 2001:db8:88::/48 any
```

Juniper

Feature set similar to IPv4.

```
family inet6 {
  filter inbound6 {
    term telnet {
      from {
        source-address {
          2001:db8:18::/48;
          2001:db8:8::/48;
          2001:db8:88::/48;
        }
        destination-port telnet;
      }
    }
  }
}
```

PF Example

```
# macros
ext_if = "{ tun0, gif0, stf0 }"
ok_if = "{ bfe0, lo0, rl0 }"
all_if = "{ tun0, gif0, stf0, bfe0, lo0, rl0 }"

tcp_services = "{ 113, 443, 80, 25, 53 }"
udp_services = "{ 53 }"

priv_nets = "{ 127.0.0.0/8, 192.168.0.0/16, 172.16.0.0/12, 10.0.0.0/8 }"
int_nets = "{ 2001:db8:ccc1::/48, 10.0.0.0/8 }"

# scrub
scrub in all fragment reassemble

# filter rules
block all

pass quick on $ok_if all keep state
pass in proto igmp all allow-opts
pass out proto igmp all allow-opts
antispoof for $ext_if
```

```
block drop in  quick on $ext_if from $priv_nets to any
block drop out quick on $ext_if from any to $priv_nets

pass out on $ext_if from any to any keep state

# Stuff going to me
pass in on $ext_if proto tcp from any to $all_if \
   port $tcp_services flags S/SA keep state
pass in on $ext_if proto tcp from any to $all_if \
   port 22 keep state
pass in on $ext_if proto udp from any to $all_if \
   port $udp_services keep state
pass in on $ext_if inet  proto icmp  from any to $all_if \
    icmp-type echoreq keep state
pass in on $ext_if inet6 proto icmp6 from any to $all_if \
    icmp6-type { echoreq, niqry } keep state

# Stuff to other hosts
pass in on $ext_if proto tcp from any to $int_nets port 22 keep state
pass in on $ext_if inet  proto icmp  from any to $int_nets \
    icmp-type echoreq keep state
pass in on $ext_if inet6 proto icmp6 from any to $int_nets \
    icmp6-type { echoreq, niqry } keep state
```

Useful Reading

- Status of Open Source and commercial IPv6 firewall implementations

  `http://www.guug.de/veranstaltungen/ecai6-2007/slides/`

  `2007-ECAI6-Status-IPv6-Firewalling-PeterBieringer-Talk.pdf`

  `http://www.bieringer.de/pb/lectures/`

  `2007-ECAI6-Status-IPv6-Firewalling-PeterBieringer-Paper.pdf`

- Survey of IPv6 Support in Commercial Firewalls

  `http://icann.org/committees/security/sac021.pdf`

- IPv6 Deployment in European Academic Networks

  `http://www.apan.net/meetings/xian2007/presentations/ipv6/`

  `apan24-deployment-chown.ppt`

- The Security Implications of IPv6

  `http://www.terena.org/events/tnc2006/programme/people/show.php?person_id=1059`

- Irish IPv6 Task Force

  `http://www.ipv6.ie/Documents.html`

Summary

- Similar to IPv4.
- A few new ideas (ND, PMTU, tunnelling, Extension headers).
- Can use similar policies.
- Somewhat subject to vendor whims.