

Open Source Security Tools for Information Technology Professionals

School of Professional Studies (SPS)

The City University of New York (CUNY)

Aron Trauring
Adjunct Professor
CEO, Zoteca

Class 6 November 2nd, 2005

Class Agenda

6:30-6:45 Article of the Week

6:45-7:45 IPTables — Part 1

7:45-8:00 Break

8:00-9:30 IPTables — Part 2

Instructor

Aron Trauring – Adjunct Professor, CUNY SPS / CEO, Zoteca

Email: atrauring@zoteca.com

Personal Website: <http://aronst.org/>

Zoteca Corporate Website: <http://www.zoteca.com/>

IPTables — Part 1

Netfilter

- Linux kernel built-in stateful packet-filtering firewall
- Controlled in user space by `iptables` command

Netfilter Components

- Tables which contain
- Chains which contain
- Rules — criteria to match traffic and apply specific set of actions

Standard Tables

- `filter` — default table for filtering traffic
- `nat` — network address translation rules
- `mangle` — packet alteration functions

Standard Chains (for `filter`)

- First evaluates which chain a packet is destined for
- INPUT — packet coming into host on a network interface
- OUTPUT — packet generated by host going out to network
- FORWARD — entered host but ultimately destined for elsewhere (e.g. on perimeter firewalls)

Rules

- `iptables [command] [rule specifications] [extensions]`

Commands

- `-A [chain]` — add rule to the end of the chain
- `-I [chain rulenum]` — add rule to position `rulenum` in chain
- `-R [chain rulenum]` — replaces rule
- `-D [chain [rulenum]]` — replaces rule
- `-L [chain]` — list all rules
- `-F` — flush all rules; good to start
- `-P [chain]` — set default policy for chain

Rule Specifications

- `-i / -o` — ethernet interface
- `-p` — protocol (tcp, udp, icmp, number)
- socket
- state inspection
- policy

Socket

- `-s` — source IP address
- `--sport` — source port
- `-d` — destination IP address
- `--dport` — destination port
- IP address can be any valid address or range including masks
- Port can be number or name (as defined in `/etc/services`)

State Inspection

- `-m state` — enable state module
- `--state [state]`
- NEW — new connection
- ESTABLISHED — existing connection in the process of sending data
- RELATED — connection used to facilitate another connection
- INVALID — connection having problems
- refines rules on connections to further restrict traffic

Policies

- `-j policy`
- ACCEPT — allow
- DROP — discards without notification to sender
- REJECT — discards but sends ICMP notification
- DROP is harsh (remote device needs to timeout) but more secure (less information provided)
- LOG — log traffic

Example Commands

- `iptables -A INPUT -i eth0 -p tcp --dport http -d 192.168.0.1 -j ACCEPT`
- `iptables -A OUTPUT -o eth0 -p tcp --dport http -j ACCEPT`
- `iptables -F INPUT`
- `iptables -L`

Example Basic Firewall — Initial Setup

- Flush and set basic policies
- Allow loopback to be freely useable
- Setup logging (log rules must come before others or will never be executed)

```
iptables -F
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT
iptables -A INPUT -i eth0 -j LOG --log-prefix "IPT_INPUT "
--log-level warning
iptables -A OUTPUT -o eth0 -j LOG --log-prefix "IPT_OUTPUT "
--log-level warning
```

Example Basic Firewall — HTTP Server

- allow all incoming HTTP
- restrict outgoing to established connections

```
iptables -A INPUT -i eth0 -p tcp --dport http -d 192.168.0.1 -m state
--state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport http -d 192.168.0.1 -m state
--state ESTABLISHED -j ACCEPT
```

Example Basic Firewall — DNS Queries

- allow establishing new connections for output only
- restrict to specific DNS servers using `-s/-d` flags

```
iptables -A INPUT -i eth0 -p udp --sport domain -s 192.168.0/24 -m state
--state ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p udp --dport http -d 192.168.0/24 -m state
--state NEW,ESTABLISHED -j ACCEPT
```

Example Basic Firewall — Remote SSH administration

- allow establishing new connections for output only
- restrict to specific DNS servers using `-s/-d` flags

```
iptables -A INPUT -i eth0 -p tcp -d 192.168.2.11 --sport ssh
-m state --state ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp -s 192.168.2.11 --dport ssh
-m state --state NEW,ESTABLISHED -j ACCEPT
```

ICMP (Internet Control Message Protocol) Traffic

- Various message types
- `ping — echo request (8)` followed by an `echo reply (0)`
- `traceroute — destination unreachable (3)` — host down or declining ICMP
- `traceroute — time exceeded (11)` — used for mapping

ICMP Attacks

- Flood attacks — storm of pings overwhelm system resulting in Denial of Service (DoS)
- Smurf attacks — forged ICMP packets sent to network broadcast addresses; results in DoS of forged recipient
- Ping of Death — ICMP packet larger than maximum IP packet size causes system to crash
- Nuke attack — ICMP packet contains information system can't handle causing system to crash

ICMP Rules

- Allow outbound `echo request` and inbound `echo reply` for ping from host.
- Allow destination unreachable and time exceeded inbound for traceroute.

ICMP Rules — Set Up

- Create new chains and redirect traffic to those chains

```
iptables -N ICMP_IN
iptables -N ICMP_OUT
iptables -A INPUT -p icmp -j ICMP_IN
iptables -A OUTPUT -p icmp -j ICMP_OUT
```

ICMP Rules — Inbound

- Drop inbound ping request
- Allow inbound echo reply, destination unreachable **and** time exceeded

```
iptables -A ICMP_IN -p icmp --icmp-type echo-request -j DROP
iptables -A ICMP_IN -i eth0 -p icmp --icmp-type 0 -m state
--state ESTABLISHED,RELATED -j ACCEPT
iptables -A ICMP_IN -i eth0 -p icmp --icmp-type 3 -m state
--state ESTABLISHED,RELATED -j ACCEPT
iptables -A ICMP_IN -i eth0 -p icmp --icmp-type 11 -m state
--state ESTABLISHED,RELATED -j ACCEPT
```

ICMP Rules — Inbound

- Create a new log chain
- Redirect rests of incoming icmp traffic to log chain
- Drop and log this traffic

```
iptables -N LOG_DROP
iptables -A ICMP_IN -i eth0 -p icmp -j LOG_DROP
iptables -A LOG_DROP -i eth0 -p icmp -j LOG --log-prefix "IPT_ICMP_IN "
iptables -A LOG_DROP -i eth0 -p icmp -j DROP
```

ICMP Rules — Outbound

- Allow outbound echo request
- Log and drop all other ICMP outbound

```
iptables -A ICMP_OUT -o eth0 -p icmp --icmp-type 8 -m state
--state NEW -j ACCEPT
iptables -A ICMP_OUT -o eth0 -p icmp -j LOG_DROP
iptables -A LOG_DROP -o eth0 -p icmp -j LOG --log-prefix "IPT_ICMP_OUT
"
iptables -A LOG_DROP -o eth0 -p icmp -j DROP
```

IPTables — Part 2

Inbound Attacks — Spoofing

- Man in the middle — A and B in legitimate session, C intercepts packets headed to B through packet sniffing, and then pretends to be B
- Blind spoofing — C makes A believe its B through forged TCP packet sequences guessed at through sampling (from outside)
- Non-blind spoofing — C can see sequence and forge them (from inside)
- C connects to A or sends A malicious information to compromise or penetrate A

Inbound Attacks — Hijacking

- Attacker C redirects routing information for A using ICMP `redirect` or manipulating ARP tables
- Traffic for A gets to C instead
- Information can be used to exploit A or sending hosts B

Inbound Attacks — Denial of Service

- “SYN Flood” — begins the process of establishing a connection in such a way as to prevent the ultimate completion; use up limited data structures
- “Own resources” — uses debugging ports `echo` and `chargen` (random character generator) between two machines on internal network
- “Fraggle” — uses UDP instead of ICMP to flood network using `echo`, `chargen`, `discard`, `qotd` and `daytime`

Using Firewall to Reduce Inbound Attacks

- Explicitly deny traffic from hosts, networks and sources you should not or cannot be receiving traffic from

Incoming Traffic from Own Host IP Address

- If `eth0` has internal IP address `192.168.0.20` and external `209.219.88.52` then can't have incoming from those IP

```
iptables -A INPUT -i eth0 -s 192.168.0.20 -j DROP
iptables -A INPUT -i eth1 -s 209.219.88.52 -j DROP
```

Outgoing Traffic Must Be from Own Host IP Address

- If from another IP must be incorrect or spoofing
- `!` is used to negate e.g. not this IP address; can be used on most iptables flags

```
iptables -A OUTPUT -o eth0 -s ! 192.168.0.20 -j DROP
iptables -A OUTPUT -o eth1 -s ! 209.219.88.52 -j DROP
```

Block Private IP Addresses on Internet-Facing Interfaces

- Private IPs can't go out to the Internet
- TEST-NET should also be blocked

```
iptables -A INPUT -i eth1 -s 10.0.0.0/8 -j DROP
iptables -A INPUT -i eth1 -s 172.16.0.0/12 -j DROP
iptables -A INPUT -i eth1 -s 192.168.0.0/16 -j DROP
iptables -A INPUT -i eth1 -s 192.0.2.0/24 -j DROP
```

Block Incoming from the Zeroconf Address

- Zeroconf range of addresses assigned when DHCP fails
- Can't ever be incoming from Internet

```
iptables -A INPUT -s 168.254.0.0/16 -j DROP
```

Block Incoming from Reserved Classes

- Block class D, E and unallocated

```
iptables -A INPUT -i eth1 -s 224.0.0.0/4 -j DROP
iptables -A INPUT -i eth1 -s 240.0.0.0/5 -j DROP
iptables -A INPUT -i eth1 -s 248.0.0.0/5 -j DROP
```

Block Incoming from Reserved Addresses

- Block localhost, broadcast and zero addresses

```
iptables -A INPUT -i eth1 -s 127.0.0.0/8 -j DROP
iptables -A INPUT -i eth1 -s 255.255.255.255/32 -j DROP
iptables -A INPUT -i eth1 -s 0.0.0.0/8 -j DROP
```

TCP Flags

- ACK — Acknowledge (used in 3-way handshake)
- RST — Reset connection
- SYN — Start of a new connection
- FIN — Close and complete a connection
- URG — Urgent pointer points to urgent data
- PSH — PUSH data directly to target port instead of buffering (eliminate lag)

iptables and TCP Flags

- `--tcp-flags [flags to check] [flags to match]`

Set up a bad flags chain

- Put redirect near top of INPUT chain so all traffic first tested for bad flags
- Anything that passes then goes to rest of INPUT chain rules

```
iptables -N BAD_FLAGS
iptables -A INPUT -p tcp -j BAD_FLAGS
```

Block SYN/FIN combination

- No way these two can validly be in same packet
- Often used to perform OS detection
- Log first and be specific so can trace origin of attack

```
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,FIN SYN,FIN
-j LOG --log-prefix "IPT: Bad SF Flag "
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,FIN SYN,FIN
-j DROP
```

Block Other Bad Flag Combinations

```
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,RST SYN,RST
-j LOG --log-prefix "IPT: Bad SR Flag "
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,RST SYN,RST
-j DROP
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,FIN,PSH SYN,FIN,PSH
-j LOG --log-prefix "IPT: Bad SFP Flag "
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,FIN,PSH SYN,FIN,PSH
-j DROP
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,FIN,RST SYN,FIN,RST
-j LOG --log-prefix "IPT: Bad SFR Flag "
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,FIN,RST SYN,FIN,RST
-j DROP
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,FIN,RST,PSH SYN,FIN,RST,PSH
-j LOG --log-prefix "IPT: Bad SFRP Flag "
iptables -A BAD_FLAGS -p tcp --tcp-flags SYN,FIN,RST,PSH SYN,FIN,RST,PSH
-j DROP
```

Block Single FIN Packets

- Used for port scans and network probes

```
iptables -A BAD_FLAGS -p tcp --tcp-flags FIN FIN
-j LOG --log-prefix "IPT: Bad F Flag "

iptables -A BAD_FLAGS -p tcp --tcp-flags FIN FIN
-j DROP
```

Block NULL and ALL Packets

- All flags present and either all not set or all set
- Used for network probing

```
iptables -A BAD_FLAGS -p tcp --tcp-flags ALL NONE
-j LOG --log-prefix "IPT: Null Flag "

iptables -A BAD_FLAGS -p tcp --tcp-flags ALL NONE
-j DROP

iptables -A BAD_FLAGS -p tcp --tcp-flags ALL ALL
-j LOG --log-prefix "IPT: All Flag "

iptables -A BAD_FLAGS -p tcp --tcp-flags ALL ALL
-j DROP
```

Block XMAS Packets

- All flags present and some are set
- Used for network probing

```
iptables -A BAD_FLAGS -p tcp --tcp-flags ALL FIN,URG,PSH
-j LOG --log-prefix "IPT: Xmas Flags "

iptables -A BAD_FLAGS -p tcp --tcp-flags ALL FIN,URG,PSH
-j DROP

iptables -A BAD_FLAGS -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG
-j LOG --log-prefix "IPT: Full Xmas Flag "

iptables -A BAD_FLAGS -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG
-j DROP
```

IPTables Limit Module

- Limits rate and volume at which packets are matched to rules

```
iptables -A INPUT -p tcp -m limit --limit 10/second -j LOG
```

- Burst function sets a threshold, that if exceeded creates a new limit
- Burst function is “recharged” only if new packets come in below the new limit

```
iptables -A INPUT -p tcp -m limit --limit-burst 100 --limit 10/minute -j
LOG
```

- Burst limit is recharged one packet for every minute rate is below 10/minute

SYN Flooding

- Send SYN packet with source unreachable or non-existent
- Connection fails and eventually resets — but meanwhile memory data structures fill up
- Attacker sends bad SYN connections until data structure overflows and no new connections possible

```
iptables -A INPUT -i eth0 -p tcp --syn -m limit --limit 5/second -j ACCEPT
```

- `--syn == --tcp-flags SYN,ACK,RST SYN`
- On heavily used servers such limits might block legitimate traffic

Kernel Modules

- `-m` loads a module
- `iprange` — range of source or destination IP addresses: `--src-range --dst-range`
- `multiport` — allows multiple ports e.g. `--dport 80,443`
- `comment` — allows adding comments up to 256 characters: `--comment "<comment>"`
- Debian automatically loads if you use `-m <module-name>`

Kernel Parameters

- `sysctl -a` — Display all parameters
- `sysctl -w <parameter>="1"/"0"`
- `sysctl -w net/ipv4/tcp_syncookies = "1"`

Saving and Restoring IPTables

- `iptables-save -t [<tablename>] [> <filename>]`
- `iptables-restore [< <filename>]`

Testing with tcpdump

- Shows packet headers
- `-i <interface>`
- `-v -vv -vvv` — level of verbosity

tcpdump Selectors

- host <hostname>
- net <IP Network>
- port <TCP port>
- **Traffic Direction:** src dst
- **Protocol:** tcp udp icmp
- **Boolean:** and or not

GUI IPTables and Smoothwall

Install and Configure Webmin

- webmin-firewall
- “Block all incoming traffic on external interface”
- “Enable firewall at boot time”

Install and Configure Shorewall

- shorewall, webmin-shorewall
- Debian does not install default configuration files
- These may be found in `/usr/share/doc/shorewall/default-config/`

Install and Configure Smoothwall

- Standalone security appliance
- Red and Green Zone