

IPTables — Part 2

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

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,  
-j LOG --log-prefix "IPT: Full Xmas Flag "
```

```
iptables -A BAD_FLAGS -p tcp --tcp-flags ALL SYN,RST,ACK,  
-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`