

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 5 October 24th, 2005

Class Agenda

6:30-6:45 Article of the Week

6:45-7:15 Review / SSH/VNC Lab

7:15-7:45 Introduction to Firewalls

7:45-8:00 Break

8:00-9:30 IPTables — Part 1

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/>

FOSS Resources: <http://www.fourm.info/>

SSH — Secure Shell

Stages of Establishing the SSH connection

1. Client needs to establish that it is talking to the machine you asked it to, and not another machine that's spoofing it (or sitting in the middle accepting data and passing it on transparently)
2. Server on the remote machine may want to establish that you are connecting from the machine you appear to be, and not another machine that's spoofing it
3. Client and server exchange keys for encrypting all future traffic between them
4. Client needs to convince the server that you are who you claim to be, and are authorized to do things on the server

SSH Uses

- Remote login
- Port forwarding

SSH Lab

SSH Agent

- Allows you to start a session and load keys only once
- Add key to two remotes
- `ssh-agent`
- `ssh-add`
- Now `ssh` to each of the remote servers
- Tied to particular bash shell. Will disappear when you log out (`exit`).
- `ps -T`

SSH Keychain

- Like `ssh-agent` except keeps keys in memory even if you logout
- Install `keychain`
- Add to `~/.bash_profile`:

```
keychain id_rsa
. ~/.keychain/$HOSTNAME-sh
```

- Null passphrase allows unattended reboots but allows easier exploitation of private key
- Add line to cron jobs for passwordless cron:

```
source ~/.keychain/$HOSTNAME-sh
```

Tunneling X Over SSH

- Enable in Webmin
- `ssh -X`
- X is a hog
- VNC better alternative

Port Forwarding with SSH

```
ssh -f -N -C -T -l [username] -L[localport]:localhost:[remoteport] [server
hostname/IP address]
```

- `-L` option forwards stuff from `localport` on the client box to `remoteport` on `hostname`
- `-f` option allows `ssh` to run in background after it prompts for the login password
- `-N` means don't execute any remote command (like a shell initiation script). This is used because all we want is port forwarding.

- -C means use compression.
- -T disables a pseudo-tty allocation (again because all we want to do is port forwarding, not open a remote shell).
- -l is login as username
- On Macintosh use 127.0.0.1 instead of localhost

Port Forward Webmin

```
ssh -f -N -C -T -l [username] -L10001:localhost:10000 [server hostname/IP address]
```

Port Forward VNC

- Install vncserver and vncclient
- On server:

```
vncserver -geometry 800x600 -depth 16 :1
```

- This starts VNC on port 5901
- On client:

```
ssh -f -N -C -T -l [username] -L5902:localhost:5901 [server hostname/IP address]
vncviewer [-shared] localhost:2
```

SSH File Permissions

- For user accounts in ~/.ssh, use the following permissions:

```
~/.ssh mode 700
~/.ssh/id_dsa and other private keys mode 400
~/.ssh/id_dsa.pub and other public keys mode 644
~/.ssh/ssh_config mode 644
~/.ssh/known_hosts mode 644
~/.ssh/authorized_keys mode 644
```

- Files in /etc/ssh should have these permissions:

```
/etc/ssh mode 755
/etc/ssh/sshd_config mode 644
/etc/ssh/ssh_config mode 644
/etc/ssh/ssh_host_dsa_key and other private keys mode 400
/etc/ssh/ssh_host_dsa_key.pub and other public keys mode 644
/etc/ssh/moduli mode 644
```

Introduction to Firewalls

What is a Firewall?

- Most important element of host defense against attack
- Filters all network traffic by examining all TCP/IP packets
- Decides whether to allow packet to continue on network or to drop packet

Three Functions of a Firewall

1. Deal with incoming traffic
2. Deal with outgoing traffic
3. Log suspicious or malicious traffic

Minimalism and Vigilance

- Start: deny everything from everywhere and to everywhere
- Defining “bad behavior” puts you in an endless treadmill
- Access to host should be exception not the rule
- Create a wall and remove “bricks” for access one at a time

Firewall Types

- Perimeter firewalls — Cisco PIX, Check Point, Smoothwall
- Perimeter firewalls are usually associated with routers
- Host firewalls — iptables, Zone Alarm

TCP Three Way Handshake

- Machine wanting to communicate sends a SYN packet (“ready”)
- Receiving machines sends a SYN/ACK (“got it”)
- Sending machine sends ACK back (“I’m going to start sending”)

TCP/IP Session Communication

- If three way handshake successful communication begins
- Data packets tagged with sequence numbers
- Acknowledges receipt of packets

TCP/IP Session Close

- One side (can be either depending on circumstances) sends a FIN (“got everything”)
- Other machine sends a FIN/ACK (“ready to end”)
- Other side sends ACK back (“Let’s finish this”)

Stateless Firewall

- Stateless firewall looks at packet headers
- Sees each packet in isolation not in context of a session
- Filters packets based on header only
- Prevents/allows connection from specific IP address or network
- Prevents/allows connection based on TCP/UDP port or type of application
- Prevents/allows connection based on category of packet (e.g. ICMP)

Stateful Firewall

- Keeps track of state of TCP/IP session
- Can do more sophisticated packet filtering
- Block SYN packets

Perimeter Firewalls

- Two or more ethernet connections
- Trusted network — internal LAN
- Public network — outside WAN
- DMZ — servers that need to be exposed to the outside but may serve internal functions
- Commercial boxes — many have Linux built in
- Built your own — use multiple NICs for ethernet connections
- Smoothwall — Linux based software distribution

Firewall Setup Business Process

1. Develop a network use policy — IM? P2P?
2. Map out inbound and outbound services
3. Convert 1 and 2 into firewall rules
4. Implement firewall rules and test
5. Review and test periodically

Host Firewalls

- Securing perimeter is not enough
- Particularly important on bastion hosts and trusted computer
- Bastion host — internet-facing server
- Problematic on client machines

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
```