Firewall

(1) **Firewall Basics**

- **Firewall**
  - A filter that will let through only desirable interactions.
  - The model is like a defensive medieval castle: these castles had strong and solid walls with slits through which archers could shoot arrows. These slits were so narrow that it was almost impossible to shoot an arrow through it from the outside.

- **What is a Firewall**
  - A process that filters all traffic between a protected or “inside” network and a less trustworthy or “outside” network.
  - Firewalls implement a *security policy*, which distinguish “good” traffic from “bad” traffic. Part of the challenge of protecting a network with a firewall is determining the security policy that meets the needs of the installation.

- **Design of Firewalls**
  - By careful positioning of a firewall within a network, we can ensure that all network access that we want to control must pass through it.
  - A firewall is typically well isolated, making it highly immune to modification. Usually a firewall is implemented on a separate computer, with direct connections generally just to the outside and inside networks.

- **Types of Firewalls**
  - **Screening router (also called packet filter)**
    - Look at the headers of packets.
    - The simplest and, in some situations, the most effective type of firewall.
  - **Proxy gateway (also called bastion host)**
    - Look at the data inside the packets.
    - Simulates the (proper) effects of an application so that the application will receive only requests to act properly.
    - An Example:
      - A company wants to set up an on-line price list so that outsiders can see the products and prices offered. It wants to be sure that no outside can change the prices or product list and that outsiders can access only the price list, not any of the more sensitive files stored inside.

- **What firewalls Can-and Cannot-Block**
  - Can protect an environment only if the firewalls control the entire perimeter.
  - Do not protect data outside the perimeter.
  - Are the most visible part of an installation to the outside and are the most attractive target for attack.
  - Must be correctly configured.
  - Cannot protect against inside attacks.
Personal Firewalls
- Protect personal machines.
- Software
  - tcpwrapper
  - iptables

TCP Wrapper
- inetd daemon: listen to incoming network connections --> invoke server program.
- inetd is the "Internet Super Servier"
- telnet stream tcp nowait root /usr/bin/in.telnetd in.telnet
- telnet stream tcp nowait root /usr/bin/tcpd in.telnet
- Beauty: generality
- TCP Wrapper Configuration File: /etc/hosts.allow (and /etc/hosts.deny)

```
in.telnetd: 10.0.2.15
in.ftpd: 10.0.2.15
```

Inetd.conf

```
ftp stream tcp nowait root /usr/sbin/tcpd in.ftpd
telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd
shell stream tcp nowait root /usr/sbin/tcpd in.rshd
login stream tcp nowait root /usr/sbin/tcpd in.rlogind
finger stream tcp nowait nobody /usr/sbin/tcpd in.fingerd
```

iptables
- Support both stateless and stateful packet filtering
- You need a kernel which has the netfilter infrastructure in it: netfilter is a general framework inside the Linux kernel which other things (such as the iptables module) can plug into. This means you need kernel 2.3.15 or beyond, and answer `Y' to CONFIG_NETFILTER in the kernel configuration.
- The iptables tool inserts and deletes rules from the kernel's packet filtering table.
- How packets traverse the filters

```
Incoming / \  Outgoing
---[Routing ]---[FORWARD]------>
    [Decision] \______/ ^
       | \    |
   ^ \___/ ^
  | \INPUT\ |OUTPUT|
    \____/ ^
          |OUT/ |
             |
       -----> Local Process ----
```
When a packet reaches a circle in the diagram, that chain is examined to decide the fate of the packet. If the chain says to DROP the packet, it is killed there, but if the chain says to ACCEPT the packet, it continues traversing the diagram.

An example of firewall rules

```
# iptables
iptables
iptables -A INPUT -p tcp --sport 80 -d 10.1.1.2
      --dport 1024:65536 -j ACCEPT
```

(2) Bypassing Firewalls

Motivation:

- If the system administrator deliberately filters out all traffic except port 22 (ssh), to a single server, it is very likely that you can still gain access other computers behind the firewall.

```
ssh -L [localhost:]port:host:hostport
```

The given port on the local (client) host is forwarded to the given host and port on the remote side. This allocates a listener port on the local side. Whenever a connection is made to this listener, the connection is forwarded over the secure channel and a connection is made to host:hostport from the remote machine (this latter connection will not be secure, it is a normal TCP connection). Port forwarding can also be specified in the configuration file.

Use ssh to communicate across a firewall: SSH Tunneling

```
# set up the tunneling (gate is the ssh server)
% ssh -l wedu -L 7777:work:22 gate.ecs.syr.edu

# use the tunneling to login to work.ecs.syr.edu
% ssh -p 7777 localhost

# For telnet
% ssh -l wedu -L 7777:apollo:23 gate.ecs.syr.edu
telnet localhost 7777
```
The GNU httptunnel

- GNU httptunnel creates a bidirectional virtual data connection tunneled in HTTP requests.
- Example 1: I want to telnet to a remote host, but my company’s firewall blocks all the outgoing telnet traffic.
  - On the server you must run hts. If I wanted to have port 80 (http) redirect all traffic to port 23 (telnet) then it would go something like:
    ```
    % hts -F server.test.com:23 80
    ```
  - On the client you would run htc. If you are going through a proxy, the -P option is needed, otherwise omit it.
    ```
    % htc -P proxy.corp.com:80 -F 23 server.test.com:80
    ```
  - Then telnet localhost and it will redirect the traffic out to port 80 on the proxy server and on to port 80 of the server, then to port 23.

- Example 2: I want to ssh to myown.ecs.syr.edu, but ECS firewall forbids that.
  - On myown.ecs: forward 80 to 22
  - On home.rr.com: forward 22 to myown.ecs.syr.edu:80
  - Run ssh localhost -p 22

IP Fragment Attacks on Firewalls

- When the filtering rule is based on TCP header, but the TCP header is fragmented, the rule will fail.
  - TCP header is at the beginning of the data area of an IP packet.
  - Firewalls often check TCP header: for example, SYN packet for connection request.

- Tiny Fragment Attack
- Filters that attempt to drop connection requests (TCP datagrams having \textit{SYN}=1 and \textit{ACK}=0) will be unable to test these flags in the first octet, and will typically ignore them in subsequent fragments.

- Fragment 1

\begin{verbatim}
IP HEADER
  ... Fragment Offset 0 ...
TCP HEADERS

Source Port | Destination Port

| Sequence Number |
\end{verbatim}

- Fragment 2

\begin{verbatim}
IP HEADERS
  ... Fragment Offset 1 ...
TCP HEADERS

Data Offset | Reserved | U | A | P | R | S | F | R | S | S | V | N |

| Acknowledgment Number | Window |
\end{verbatim}

- Protection against the tiny-fragment attack: require a minimum length for the zero-offset fragment.

- Overlapping Fragment Attack
  - Assumption: firewalls only check the packets with offset=0.
  - Fragment 1: The first fragment contains values, e.g., \textit{SYN}=0, \textit{ACK}=1, that enable it to pass through the filter unharmed.
Fragment 2: The second fragment, with a fragment offset of eight octets, contains TCP Flags that differ from those given in the first fragment, e.g., \( \text{SYN} = 1, \ \text{ACK} = 0 \). Since this second fragment is not a 0-offset fragment, it will not be checked, and it, too will pass through the filter.

Firewalking

- **Firewall protocol scan**: determine what ports/protocols a firewall will let traffic through on from the attacking host.
- **Approach**: send IP packets with small TTL; if you get a TTL-exceeded error, the port can pass through.
- **Tools**:
  - traceroute
  - firewalk: http://www.packetfactory.net/projects/firewalk/
Reference

1. Security Considerations for IP Fragment Filtering
   http://www.scit.wlv.ac.uk/rfc/rfc18xx/RFC1858.html