Announcements

• Project 3
  – Due Date: April 21st (Last day of classes)
  – Updated project description (clarifying some things)
  – Group project: you must work in groups of 2 or 3 people.
    • Mail groups to cis551staff@seas.upenn.edu
    • If you have trouble finding a group, post on the class news group

• Final Exam has been Scheduled:
  – Friday, May 5th
  – 9-11 a.m.
  – Moore 216
Plan for today

• Briefly talk about application level protocols

• Talk about NATs and Firewalls

• Excellent reference:
  – "Firewalls and Internet Security" by Cheswick, Bellovin, and Rubin
Protocol Stack Revisited

Application

Presentation

Session

Transport

Network

Data Link

Physical

SMTP, HTTP, SNMP, FTP, ...

So far…
Common Features

- SMTP, HTTP, SNMP, FTP…
  - Request/Reply protocols built on TCP or UDP
  - Designed to handle a fixed set of messages
  - Companion *data format*
  - Many applications

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Data Format</th>
<th>Programs</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMTP</td>
<td>RFC 822 and MIME</td>
<td>Pine, NSMail, Eudora, Outlook,…</td>
</tr>
<tr>
<td>HTTP</td>
<td>HTML</td>
<td>Explorer, Netscape, Opera,…</td>
</tr>
<tr>
<td>SNMP</td>
<td>MIB</td>
<td>snmpget, snmpset,…</td>
</tr>
</tbody>
</table>
SMTP: Simple Mail Transfer Protocol

- Data format RFC822
  - ASCII text
  - Header and Body

- MIME: Multipurpose Internet Mail Extensions
  - Mail systems assume ASCII
    - Only 64 valid characters A-Z, a-z, 0-9, +, /
  - Some datatypes include arbitrary binary data (e.g. JPEG)
  - Base64 encoding
    - 3 bytes of data map to 4 ASCII Characters
    - A=0,B=1,...
SMTP

- **Mail Reader**
  - User edits/reads/search e-mail

- **Mail Daemon**
  - Process running on each host (port 25)
  - Uses SMTP/TCP to transmit mail to daemons on other machines
  - Most daemons based on Berkley’s **sendmail**

- **Mail Gateways**
  - Store and forward e-mail (much like IP router)
  - Buffers on disk
  - Attempts to resend
RFC822 Headers

- <CRLF>-terminated lines containing pairs of form **type**: value
- Many valid Header types
- Some headers filled out by client
  - To: stevez@cis.upenn.edu
  - Subject: CSE551
- Others filled out by mail delivery system
  - Date:
  - Received:
  - From:
From: Steve Zdancewic <stevez@cis.upenn.edu>
MIME-Version: 1.0
To: stevez@cis.upenn.edu
Subject: Example Mail
Content-Type: multipart/mixed; boundary="---------020307000708030506070607"

This is a multi-part message in MIME format.
---------020307000708030506070607
Content-Type: text/plain; charset=us-ascii; format=flowed
Content-Transfer-Encoding: 7bit

This is the body.

---------020307000708030506070607
Content-Type: text/plain; name="example.txt"
Content-Transfer-Encoding: 7bit
Content-Disposition: inline; filename="example.txt"

Hello

---------020307000708030506070607
Content-Type: image/jpeg; name="doc.jpg"
Content-Transfer-Encoding: base64
Content-Disposition: inline; filename="doc.jpg"

/9j/4AAQSkZJRgABAQEASABIAAD//gAXQ3JlYXRlZCB3aXRoIFRoZSBHSU1Q/9sAQwAIBgYHBgUIBwcHCQkICgwUDQwLCwwZEhMPFB0aHx4dGhwcICQuJyAiLCMcHCg3KSwwMTQ0NB8nOT04...
SMTP security

- SMTP provides no authentication
  - Easy to spoof sending address
  - Very familiar problem found in Spam
- Sendmail program is a notorious source of vulnerabilities
  - Complicated, concurrent program
  - Needs privileges to write to all mail files
  - See www.sendmail.org

- Sendmail hit by data interception
  Thursday 23 March 2006
  "Internet security researchers have discovered a serious flaw in Sendmail. The flaw could allow remote attackers to take control of users' PCs."
MIME security

- Mime allows ability to mail executable content
  - Primary transmission vector for worms and viruses
- MIME allows external references to files:

```plaintext
Content-Type: Message/External-body;
    name="foo.txt";
    site="ftp.cis.upenn.edu";
    access-type="anon-ftp";
    directory="bar"
Content-Type: text/plain
```
NATs and Firewalls

- Problem: Protecting or isolating one part of the network from other parts

- Need to filter or otherwise limit network traffic
  - How to configure this information?

- Questions:
  - What information do you use to filter?
  - Where do you do the filtering?
Kinds of Firewalls

• Personal firewalls
  – Run at the end hosts
  – e.g. Norton, Windows, etc.
  – Benefit: has more application/user specific information

• Network Address Translators
  – Rewrites packet address information

• Filter Based
  – Operates by filtering based on packet headers

• Proxy based
  – Operates at the level of the application
  – e.g. HTTP web proxy
Network Address Translation

- Idea: Break the invariant that IP addresses are globally unique

![Diagram showing NAT and IP addresses]
NAT Behavior

- NAT maintains a table of the form:
  \(<\text{client IP}> \ <\text{client port}> \ <\text{NAT ID}>\)

- Outgoing packets (on non-NAT port):
  - Look for client IP address, client port in the mapping table
  - If found, replace client port with previously allocated NAT ID (same size as PORT #)
  - If not found, allocate a new unique NAT ID and replace source port with NAT ID
  - Replace source address with NAT address
NAT Behavior

• Incoming Packets (on NAT port)
  – Look up destination port number as NAT ID in port mapping table
  – If found, replace destination address and port with client entries from the mapping table
  – If not found, the packet is not for us and should be rejected

• Table entries expire after 2-3 minutes to allow them to be garbage collected
Benefits of NAT

- Only allows connections to the outside that are established from inside.
  - Hosts from outside can only contact internal hosts that appear in the mapping table, and they’re only added when they establish the connection
  - Some NATs support firewall-like configurability

- Can simplify network administration
  - Divide network into smaller chunks
  - Consolidate configuration data

- Traffic logging
Drawbacks of NAT

• Rewriting IP addresses isn’t so easy:
  – Must also look for IP addresses in other locations and rewrite them (may have to be protocol-aware)
  – Potentially changes sequence number information
  – Must validate/recalculate checksums

• Hinder throughput

• May not work with all protocols
  – Clients may have to be aware that NAT translation is going on

• Slow the adoption of IPv6?

• Limited filtering of packets / change packet semantics
  – For example, NATs may not work well with encryption schemes that include IP address information
Firewalls

- Filters protect against “bad” packets.
- Protect services offered internally from outside access.
- Provide outside services to hosts located inside.
Filtering Firewalls

- Filtering can take advantage of the following information from network and transport layer headers:
  - Source
  - Destination
  - Source Port
  - Destination Port
  - Flags (e.g. ACK)

- Some firewalls keep state about open TCP connections
  - Allows conditional filtering rules of the form “if internal machine has established the TCP connection, permit inbound reply packets”
Three-Way Handshake

Active participant (client) <-> Passive participant (server)

- SYN, SequenceNum = x
- SYN + ACK, SequenceNum = y,
  Acknowledgment = x + 1
- ACK, Acknowledgment = y + 1
Ports

- Ports are used to distinguish applications and services on a machine.
- Low numbered ports are often reserved for server listening.
- High numbered ports are often assigned for client requests.

- Port 7 (UDP,TCP): echo server
- Port 13 (UDP,TCP): daytime
- Port 20 (TCP): FTP data
- Port 21 (TCP): FTP control
- Port 23 (TCP): telnet
- Port 25 (TCP): SMTP
- Port 79 (TCP): finger
- Port 80 (TCP): HTTP
- Port 123 (UDP): NTP
- Port 2049 (UDP): NFS
- Ports 6000 to 6xxx (TCP): X11
Filter Example

<table>
<thead>
<tr>
<th>Action</th>
<th>ourhost</th>
<th>port</th>
<th>theirhost</th>
<th>port</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>*</td>
<td>*</td>
<td>BAD</td>
<td>*</td>
<td>untrusted host</td>
</tr>
<tr>
<td>allow</td>
<td>GW</td>
<td>25</td>
<td>*</td>
<td>*</td>
<td>allow our SMTP port</td>
</tr>
</tbody>
</table>

Apply rules from top to bottom with assumed *default* entry:

<table>
<thead>
<tr>
<th>Action</th>
<th>ourhost</th>
<th>port</th>
<th>theirhost</th>
<th>port</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>default</td>
</tr>
</tbody>
</table>

Bad entry intended to allow connections to SMTP from inside:

<table>
<thead>
<tr>
<th>Action</th>
<th>ourhost</th>
<th>port</th>
<th>theirhost</th>
<th>port</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>25</td>
<td>connect to their SMTP</td>
</tr>
</tbody>
</table>

This allows all connections from port 25, but an outside machine can run *anything* on its port 25!
Filter Example Continued

Permit * \textit{outgoing} calls to port 25.

<table>
<thead>
<tr>
<th>Action</th>
<th>src</th>
<th>port</th>
<th>dest</th>
<th>port</th>
<th>flags</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow</td>
<td>123.45.6.*</td>
<td>*</td>
<td>*</td>
<td>25</td>
<td>*</td>
<td>their SMTP</td>
</tr>
<tr>
<td>allow</td>
<td>*</td>
<td>25</td>
<td>*</td>
<td>*</td>
<td>ACK</td>
<td>their replies</td>
</tr>
</tbody>
</table>

This filter doesn’t protect against IP address spoofing. The bad hosts can “pretend” to be one of the hosts with addresses 123.45.6.*.
When to Filter?

Router

Inside

Outside
On Input or Output

• Filtering on output can be more efficient since it can be combined with table lookup of the route.

• However, some information is lost at the output stage
  – e.g. the physical input port on which the packet arrived.
  – Can be useful information to prevent address spoofing.

• Filtering on input can protect the router itself.
Recommend: Filter ASAP

<table>
<thead>
<tr>
<th>Action</th>
<th>src</th>
<th>port</th>
<th>dest</th>
<th>port</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>BAD</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>we don’t trust them</td>
</tr>
<tr>
<td>allow</td>
<td>*</td>
<td>*</td>
<td>GW</td>
<td>25</td>
<td>connect to our SMTP</td>
</tr>
<tr>
<td>allow</td>
<td>GW</td>
<td>25</td>
<td>*</td>
<td>*</td>
<td>our reply packets</td>
</tr>
</tbody>
</table>

Is preferred over:

<table>
<thead>
<tr>
<th>Action</th>
<th>src</th>
<th>port</th>
<th>dest</th>
<th>port</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>block</td>
<td>*</td>
<td>*</td>
<td>BAD</td>
<td>*</td>
<td>subtle difference</td>
</tr>
<tr>
<td>allow</td>
<td>*</td>
<td>*</td>
<td>GW</td>
<td>25</td>
<td>connect to our SMTP</td>
</tr>
<tr>
<td>allow</td>
<td>GW</td>
<td>25</td>
<td>*</td>
<td>*</td>
<td>our reply packets</td>
</tr>
</tbody>
</table>
Example of a Pitfall

- Filter output to allow incoming and outgoing mail, but prohibit all else.

<table>
<thead>
<tr>
<th>Action</th>
<th>dest</th>
<th>port</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>allow</td>
<td>*</td>
<td>25</td>
<td>incoming mail</td>
</tr>
<tr>
<td>allow</td>
<td>*</td>
<td>&gt;= 1024</td>
<td>outgoing responses</td>
</tr>
<tr>
<td>block</td>
<td>*</td>
<td>*</td>
<td>nothing else</td>
</tr>
</tbody>
</table>

- Apply this output filter set to both interfaces of the router. Does it work?
- Unintended consequence: allows all communication on high numbered ports!
Principles for Firewall Configuration

- Least Privileges:
  - Turn off everything that is unnecessary (e.g. Web Servers should disable SMTP port 25)

- Failsafe Defaults:
  - By default should reject
  - (Note that this could cause usability problems…)

- Egress Filtering:
  - Filter outgoing packets too!
  - You know the valid IP addresses for machines internal to the network, so drop those that aren’t valid.
  - This can help prevent DoS attacks in the Internet.
Example “real” firewall config script

# FreeBSD Firewall configuration.
# Single-machine custom firewall setup. Protects somewhat
# against the outside world.

# Set this to your ip address.
ip="192.100.666.1"
setup_loopback

# Allow anything outbound from this address.
${fwcmd} add allow all from ${ip} to any out

# Deny anything outbound from other addresses.
${fwcmd} add deny log all from any to any out

# Allow inbound ftp, ssh, email, tcp-dns, http, https, imap, imaps,
# pop3, pop3s.
${fwcmd} add allow tcp from any to ${ip} 21 setup
${fwcmd} add allow tcp from any to ${ip} 22 setup
${fwcmd} add allow tcp from any to ${ip} 25 setup
${fwcmd} add allow tcp from any to ${ip} 53 setup
${fwcmd} add allow tcp from any to ${ip} 80 setup
${fwcmd} add allow tcp from any to ${ip} 443 setup

...
Another problem with Filtering

• Handling IP Fragments
  – Possible for ACK and SYN flag bits in a TCP packet could end up in a different IP fragment than the port number
  – There are malicious tools that intentionally break up traffic in this way
  – Fix: Problem is "tiny" initial IP fragment, so require that initial IP fragment be > 16 bytes (or better yet, large enough for whole TCP header).
Proxy-based Firewalls

- Proxy acts like *both* a client and a server.
- Able to filter using application-level info
  - For example, permit some URLs to be visible outside and prevent others from being visible.
- Proxies can provide other services too
  - Caching, load balancing, etc.
  - FTP and Telnet proxies are common too
Benefits of Firewalls

- Increased security for internal hosts.
- Reduced amount of effort required to counter break ins.
- Possible added convenience of operation within firewall (with some risk).
- Reduced legal and other costs associated with hacker activities.
Drawbacks of Firewalls

• Costs:
  – Hardware purchase and maintenance
  – Software development or purchase, and update costs
  – Administrative setup and training, and ongoing administrative costs and trouble-shooting
  – Lost business or inconvenience from broken gateway
  – Loss of some services that an open connection would supply.

• False sense of security
  – Firewalls don’t protect against viruses…
  – Can almost always "tunnel" one protocol on top of another: e.g. mail protocol on top of HTTP