CSE 123
Computer Networking

Fall 2009

Network security
NAT, Firewalls, DDoS

Geoff Voelker
Network security

- The Internet is not always a friendly place
- In fact, hosts on the Internet are under constant attack
- How to deal with this is a large topic
  - Take CSE127 in the Spring!
- Today: a look at a few network layer issues
  - Address translation
  - Firewalls
  - Denial-of-service
NATs and Firewalls

- Problem: Protecting or isolating one part of the network from other parts
  - In particular: protect your network from global Internet

- 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. Web proxy
Network Address Translation

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

- NAT maintains a table of the form:
  `<client IP> <client port> <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

- Don’t need as large an external address space
  - (i.e. 10 machines can share 1 IP address)
Drawbacks of NAT

- Rewriting IP addresses isn’t always easy:
  - Must also look for IP addresses in data stream and rewrite them (may have to be protocol-aware)
  - Potentially changes sequence number information
    » E.g., Addresses listed as text in FTP session
      public address is 128.124.127.245 (16 bytes of text),
      private is 10.1.1.1 (9 bytes of text)
  - Must validate/recalculate protocol checksums

- May not work with all protocols
  - E.g., streaming media protocols
    » Common for client to invoke server and server to open
      *new* connection back to client
  - Clients may have to be aware that NAT translation
    is going on
Firewalls

- Break the invariant that all IP address/ports are equally reachable from anywhere
- Idea: classify traffic and say “yes” or “no”

- Packet Filter Based
  - Operates by filtering based on packet headers
- Network Proxy based
  - Operates at the level of the application
  - e.g. HTTP web proxy
Packet Filtering Firewalls

- Packet filtering firewalls 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”

- Best practice: default deny and only enable ports for necessary services
### 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>mailhst</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!
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.
  - Block sites based on site content
Benefits of Firewalls

- Increased security for internal hosts
- Reduced amount of effort required to counter break ins (small number of devices to configure)
- Possible added convenience of operation within firewall (with some risk)
Drawbacks of firewalls

- Costs:
  - Hardware/software purchase and maintenance
  - Administrative setup and training, and ongoing administrative costs and trouble-shooting
  - Single-point of failure on network, added management complexity

- False sense of security
  - Limited language (IP addresses, ports) so doesn’t protect against malware in Web pages, flash, e-mail, etc.
  - Inside vs outside model is fragile (once an internal host is compromised firewall does no good)
    » What about wireless laptops?
Denial-of-service

- Attack against *availability* – shut down target site
- Two kinds of attacks:
  - **Logic vulnerabilities**: exploit bugs to cause crash
    » e.g. Ping-of-Death, Land
    » Fix via filtering and patching
  - **Resource consumption**: overwhelm with spurious requests
    » e.g. SYN flood, Smurf, bandwidth overflow
    » Much tougher to fix…
- Distributed denial-of-service attacks (DDOS)
  - Lots of hosts attack a victim at once
  - Typically many hosts under centralized control (botnet)
Consequent: Victim suffers

- Server CPU/Memory resources
  - Consumes connection state (e.g. SYN flood)
  - Time to evaluate messages (interrupt livelock)
    » Some messages take “slow path” (e.g. invalid ACK)
  - Can cause new connections to be dropped and existing connections to time-out

- Network resources
  - Many routers packet-per-second limited, FIFO queuing
  - If attack is greater than forwarding capacity, good data will be dropped
Aside: UCSD analysis of DoS

Simple question: how prevalent are denial-of-service attacks?

In 2001, lots of anecdotal answers/rumors
Quantitative data?

- **Isn’t available** (i.e. no one knows)

- Inherently **hard to acquire**
  - Few content or service providers collect such data
  - If they do, its usually considered sensitive

- **Infeasible to collect** at Internet scale
  - How to monitor enough to the Internet to obtain a representative sample?
A good estimate: [Moore, Voelker, Savage01]

- Backscatter analysis
  - New technique for estimating global denial-of-service activity

- First data describing Internet-wide DoS activity
  - ~4,000 attacks per week (> 12,000 over 3 weeks)
  - Instantaneous loads above 600k pps (packets per second)
  - Characterization of attacks and victims
Key idea

- Flooding-style DoS attacks
  - e.g. SYN flood – lots of TCP SYN packets; designed to consume connection state at receiver

- Attackers spoof source address randomly
  - *No requirement to have correct source address!*
  - Hides identity of attacking host

- Victims, in turn, respond to attack packets

- Unsolicited responses (*backscatter*) equally distributed across IP address space

- Received backscatter is evidence of an attacker elsewhere
Random IP spoofing produces random backscatter
Backscatter analysis

- Monitor block of $n$ IP addresses
- Expected # of backscatter packets given an attack of $m$ packets:

$$E(X) = \frac{nm}{2^{32}}$$

- Extrapolated attack rate $R'$ is a function of measured backscatter rate $R$:

$$R \geq R' \frac{2^{32}}{n}$$
Attacks over time
Example 1: Periodic attack (1hr per 24hrs)
Example 2: Punctuated attack (1 min interval)
Attack rate distribution

![Graph showing attack rate distribution with two lines: All Attacks and Uniform Random Attacks. The x-axis represents the estimated rate in packets per second, ranging from 10 to 1e+06, and the y-axis represents the percent of attacks, ranging from 0 to 100. The graph illustrates the distribution of attacks with different rates.](image)
Victim characterization by DNS name

- Entire spectrum of commercial businesses
  - Yahoo, CNN, Amazon, etc and many smaller biz
- Evidence that minor DoS attacks used for personal vendettas
  - 10-20% of attacks to home machines
  - A few very large attacks against broadband
  - Many reverse mappings clearly compromised (e.g. is.on.the.net.illegal.ly and the.feds.cant.secure.their.shellz.ca)
- 5% of attack target infrastructure
  - Routers (e.g. core2-core1-oc48.paol.above.net)
  - Name servers (e.g. ns4.reliablehosting.com)
Victim breakdown by TLD (top level domain)

- Week 1
- Week 2
- Week 3

Top-Level Domain:
- unknown
- net
- com
- ro
- br
- org
- edu
- ca
- de
- uk
Denial-of-Service Prevalence Summary

- Lots of attacks – some very large
  - >12,000 attacks against >5,000 targets in a week
  - Most < 1,000 pps, but some over 600,000 pps
- Everyone is a potential target
  - Targets not dominated by any TLD or 2nd-level domain
  - Targets include large e-commerce sites, mid-sized business, ISPs, government, universities and end-users
  - Something weird is happening in Romania
- New attack “styles”
  - Punctuated/periodic attacks
  - Attacks against infrastructure targets & broadband
What to do?

- Defenses against address spoofing
- Filtering based on attack features or IP address
- Make attacker do work

- Those marked with * are used in real products vs hypothetical academic ideas
Address spoofing

- Filter packets with incorrect source addresses [*]
  - Network egress: filter packets on a link whose source addresses are not reached using the link as the next hop
  - Network ingress: filter packets whose source address are not in the routing table at all

- SYN Cookies [*]
  - Issue: allocating per TCP session state is expensive (that’s why the SYN flood attack works)
  - Delay allocation of state until remote host commits to three-way handshake
  - Send back SYN/ACK packet without allocating state on server; server’s initial sequence number (ISN) encodes a secret “cookie” that is function of src,dst,srcport,dstport and time.
  - Allocate state only when client sends ACK to server’s SYN/ACK (using cookie to validate)
Address spoofing (2)

- **Puzzles**
  - Don’t commit state until client has done a bunch of “work” for you (i.e. solved computationally tough problem)
  - Server provides puzzle to client
    - Hardness can be determined by load
  - Client must solve puzzle (easy to verify by server) to allocate state
  - Tricky: if validation isn’t free, bad guy can send lots of invalid puzzle solutions to server

- **CAPTCHAs (reverse turing test)**
  - Put graphical puzzle in response packet
  - Make user solve graphical puzzle before committing state
Address spoofing

- TTL filtering [*]
  - From a given host the TTL is decremented by a certain number of hops (based on network topology)
  - Standard IP implementations set the initial packet TTL value to a small set of values (32, 64, 128, 255) [can normalize because Internet diameter is mostly < 32]
  - Thus, keep track of TTLs for each source network and if attack starts, filter packets whose TTLs are inconsistent
Address spoofing(4)

- Traceback
  - Router support for tracking packets back to their source
  - Probabilistic packet marking [Savage00, etc]
    » With some probability $p$, a router encodes the identity of the link the packet will traverse. Victim uses these packet “marks” to reconstruct path back to victim
  - Packet logging [Snoeren01, etc]
    » Routers hash packet header and store in database
    » Victim queries router about whether they’ve seen a given packet and if so, from where… repeat

- Main issue: then what?
Packet filtering

- Idea, if there is a common feature to the packet (i.e. “Die, you loser” in the TCP data) then look for those packets and drop them [*]
- If no feature exists then try to find way to add a “good” feature
  - E.g. ask user to solve CAPTCHAs and if they do re-direct them to special port/IP; then filter packets not going to that special port/IP
- Instead of dropping packets, can simply rate-limit packets that are suspicious [*]
Buy more resources

- Buy more bandwidth, but more servers
- Large content distribution networks (e.g. Akamai) can handle very large attacks [*]
- Each attacker gets diverted to local Akamai server
  - Total bandwidth Akamai can handle is the product of the bandwidth to all Akamai servers
  - Akamai has weather attacks in excess of 10GB

- Issue: who pays for that? $$$
Summary

- Network address translation hides internal structure of network from outside
- Firewall limits which packets will be delivered to a host
- DDoS, some of the toughest problems to solve
  - Network service model allows unsolicited requests
  - Bad guys can leverage large # of resources
  - No simple fix