CSE 127
Computer Security

Fall 2015

Network Security II
Denial of Service (DoS) and DNS

Stefan Savage
Denial-of-service

- Attack against *availability*, not confidentiality, integrity, authenticity, etc

- 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
Step 1: Attacker infiltrates machines

- Scan machines via Internet
- Exploit known bugs & vulnerabilities
- Install Zombie/Bot software
  - Code to attack target victims
  - Software to allow coordinated remote control
    » Sometimes hierarchy of handlers
- Cover tracks (e.g. rootkit)
- Repeat… (highly automated)
Step 2: Attacker sends commands to handler
Step 3: Handler sends commands to bots/zombies
Step 4: Bots/Zombies attack target

Attacker

Z Z Z H Z Z

Victim

N Gbps
Step 5: 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?
Most data is anecdotal

Press reports:

"Losses … could total more than $1.2 billion"
- Yankee Group report

Analysts:

Surveys:

"38% of security professionals surveyed reported denial of service activity in 2000"
- CSI/FBI survey

November 30, 2015
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
  - \( \sim 4,000 \) 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, ICMP flood
- Attackers spoof source address randomly
  - True of all major attack tools
- 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

SYN packets

Attacker

Victim

SYN+ACK backscatter

Attack

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 $R$:

$$R \geq R' \frac{2^{32}}{n}$$
Experimental apparatus...

Monitor (w/big disk)

Big Quiescent Network

(\(2^{24}\) addresses)

Internet
Attacks over time
Example 1: Periodic attack (1hr per 24hrs)
Example 2: Punctuated attack (1 min interval)
Attack duration distribution

% Attacks

Attack Duration

% Attacks

Attack Duration

Week 1
Week 2
Week 3
Attack rate distribution

![Graph showing the distribution of attack rates. The x-axis represents the estimated rate (packets per second), and the y-axis represents the percent of attacks. Two lines are depicted: one for all attacks and another for uniform random attacks.]
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)

<table>
<thead>
<tr>
<th>Top-Level Domain</th>
<th>Week 1</th>
<th>Week 2</th>
<th>Week 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>unknown</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>net</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ro</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>br</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>org</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>edu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ca</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>de</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>uk</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Percent of Attacks

Week 1
Week 2
Week 3
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 2\textsuperscript{nd}-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
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 ISN encodes a secret “cookie” that is function of src,dst,srcport,dstport and time.
  - Allocate state 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 (3)

- TTL filtering [*]
  - From a given host the TTL is decremented by a certain number of hops (based on network topology)
  - Std IP implementations set the 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 payload) then look for those packets and drop them [*]
- If no feature exists then find way to add a “good” feature
  - Hash for packets from well-behaved connections
  - Filter IP addresses that aren’t solving puzzles
  - Integrate with traceback so packet marks for spoofed packets can be used to filter
- Instead of dropping packets, can simply rate-limit packets that are suspicious [*]
Buy more resources

- 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 weathered attacks in excess of 100GB

- Issue: who pays for that? $$$
Modern DoS issues

- Network-focused DoS
  - Don’t attack end host, attack its router interface
    » Both bugs and resource consumption attacks
  - Yahoo attack spoofed source address of yahoo’s routers
  - Attack victim’s DNS server

- Application-focused DoS
  - Don’t need to overload network if can force application to do lots of work per request
  - E.g. Search engines use caches of common requests… uncommon requests require a full search
Reflection attacks

- Spoof source address to be that of victim
- Common example
  - Send name server request to 1000s of DNS servers *on behalf* of victim
  - All name servers send responses to victim
- Advantages
  - Amplification: frequently responses >> requests
  - Anonymity: attack doesn’t come from attacker’s machines
In general, some of the toughest problems to solve
- Network service model allows unsolicited requests
- Bad guys can leverage large # of resources
- Hard to attribute network actions
- Few systems can account for effort spent per request or isolate impact of some requests from others

DDoS-based extortion and retribution (e.g., against security companies) is not uncommon
Sitching gears: DNS

- We humans do not tend to remember 32bit numbers…

- Solution: domain names
  - Human readable identifiers (e.g., www.cs.ucsd.edu)

- Problem: how to map DNS names to IP addresses?
  - In the old days we had a big file
  - Today we use a distributed name servers called the Domain Name System (DNS)
Domain Name System (DNS)

- Hierarchical Name Space

Diagram:
- Root
  - org
  - net
  - edu
    - com
    - uk
    - ca
    - ucsd
      - cs
      - ece
      - www
    - wisc
    - ucb
    - mit
    - cmu
DNS Root Name Servers

- Hierarchical service
  - Root name servers for top-level domains
  - Authoritative name servers for subdomains
  - Local name resolvers contact authoritative servers when they do not know a name
DNS Lookup Example

DNS record types (partial list):
- **NS**: name server (points to other server)
- **A**: address record (contains IP address)
- **MX**: address in charge of handling email
- **TXT**: generic text (e.g. used to distribute site public keys (DKIM))
Caching

- **DNS responses are cached**
  - Quick response for repeated translations
  - Useful for finding servers as well as addresses
    - NS records for domains

- **DNS negative queries are cached**
  - Save time for nonexistent sites, e.g. misspelling

- **Cached data periodically times out**
  - Lifetime (TTL) of data controlled by owner of data
  - TTL passed with every record
Basic DNS Vulnerabilities

- Users/hosts trust the host-address mapping provided by DNS:
  - Used as basis for many security policies:
    - Browser “same origin” policy, URL address bar, user trust

- Obvious problems
  - Interception of requests or compromise of DNS servers can result in incorrect or malicious responses
    - e.g., if you can observe request then can spoof response
    - e.g., hijack network route to spoof DNS

- Less obvious
  - Name server can delegate name to another ns and then may also supply its IP address (trouble)
DNS Packet

- Query ID:
  - 16 bit random value
  - Links response to query

(from Steve Friedl)
# Resolver to NS request

- **src IP**: 68.94.156.1
- **dst IP**: 192.26.92.30
- **src port**: 5798
- **dst port**: 53
- **QID**: 43561
- **Question count**: 1
- **Answer count**: 0
- **Authority count**: 0
- **Additional Record count**: 0

**Qu**: What is A record for www.unixwiz.net?

- **RD**: 1 - recursion desired
- **OP**: 0 - standard query
- **QR**: 0 - this is a query

**Notes**:
- **DNSR1.SBCGLOBAL.NET**
- **C.GLTD-SERVERS.NET**
Response to resolver

Response contains IP addr of next NS server (called “glue”)

Response ignored if unrecognized QueryID
**Authoritative response to resolver**

**bailiwick checking:**
response is cached if it is within the same domain of query (i.e. a.com cannot set NS for b.com)

| IP | 
|----|---|
| src IP = 64.170.162.98 |
| dst IP = 68.94.156.1 |

| UDP | 
|-----|---|
| src port = 53 |
| dst port = 5798 |

<table>
<thead>
<tr>
<th>Question</th>
<th>Answer</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is A record for www_unixwiz_net?</td>
<td>www_unixwiz_net A = 8.7.25.94 1 hr</td>
</tr>
<tr>
<td>unixwiz_net NS = linux_unixwiz_net</td>
<td>2 dy</td>
</tr>
<tr>
<td>unixwiz_net NS = cs_unixwiz_net</td>
<td>2 dy</td>
</tr>
<tr>
<td>linux_unixwiz_net A = 64.170.162.98</td>
<td>1 hr</td>
</tr>
<tr>
<td>cs_unixwiz_net A = 8.7.25.94</td>
<td>1 hr</td>
</tr>
</tbody>
</table>

**final answer**
Victim machine visits attacker’s web site, downloads Javascript

User browser → Query: a.bank.com → local DNS resolver

Query for a.bank.com with QID=x_1 → ns.bank.com

ns.bank.com → IPaddr

256 responses: Random QID y_1, y_2, ...

NS bank.com = ns.bank.com
A ns.bank.com = attackerIP

Attacker wins if ∃j: x_1 = y_j
Response is cached and attacker owns bank.com
If at first you don’t succeed …

- Victim machine visits attacker’s web site, downloads Javascript

**Diagram:***
- User browser queries `b.bank.com`
- Local DNS resolver sends query with QID=x₂ to `ns.bank.com`
- `b.bank.com` returns 256 responses: Random QID y₁, y₂, ...
- `NS bank.com=ns.bank.com` and A `ns.bank.com=attackerIP`

**Success:***
- Attacker wins if ∃j: x₂ = yⱼ
- Response is cached and attacker owns bank.com
- Success after ~ 256 tries (few minutes)
Defenses

- Increase Query ID size. How? Some proposals
  - Randomize src port, additional 11 bits
    Now attack takes several hours
  - Ask every DNS query twice:
    » Attacker has to guess QueryID correctly twice (32 bits)
    » Doubles load on DNS system

- Try to detect poisoning
  - Ignore responses not directly necessary to query

- Authenticated requests/responses
  - Provided by DNSsec … but few domains use DNSsec
What do you do with DNS poisoning?

- One example: pharming
  - Change IP addresses to redirect URLs to fraudulent sites
  - Potentially more dangerous than phishing attacks
  - No email solicitation is required

- Lots of DNS poisoning attacks have occurred in the wild
  - January 2005, the domain name for a large New York ISP, Panix, was hijacked to a site in Australia.
  - In November 2004, Google and Amazon users were sent to Med Network Inc., an online pharmacy
  - In March 2003, a group dubbed the "Freedom Cyber Force Militia" hijacked visitors to the Al-Jazeera Web site and presented them with the message "God Bless Our Troops"
Summary

- Current DNS system does not provide strong evidence binding request to response
- Response can provide more data than was asked for
- Together allows attacker to “poison” DNS and divert traffic to their sites
Next time

- Internet crime