CSE 543 - Computer Security

Lecture 22 - Denial of Service
November 15, 2007
URL: http://www.cse.psu.edu/~tjaeger/cse543-f07/
Denial of Service

- Intentional prevention of access to valued resource
  - CPU, memory, disk (system resources)
  - DNS, print queues, NIS (services)
  - Web server, database, media server (applications)

- This is an attack on \textit{availability (fidelity)}

- \textbf{Note}: launching DOS attacks is easy

- \textbf{Note}: preventing DOS attacks is hard
  - Mitigation the path most frequently traveled
SMURF Attacks

- This is one of the deadliest and simplest of the DOS attacks (called a *naturally amplified* attack)
- Send a large number PING packet networks on the broadcast IP addresses (e.g., 192.168.27.254)
- Set the source packet IP address to be your victim
- All hosts will reflexively respond to the ping at your victim
- … and it will be crushed under the load.
Canonical (common) DOS - Request Flood

- Attack: request flooding
  - Overwhelm some resource with legitimate requests
  - e.g., web-server, phone system

- Note: unintentional flood is called a *flash crowd*
DOS Prevention - Reverse-Turing Tests

- **Turing test**: measures whether a human can tell the difference between a human or computer (AI)

- **Reverse Turning tests**: measures whether a user on the internet is a person, a bot, whatever?

- CAPTCHA - completely automated public Turing test to tell computers and humans apart
  - contorted image humans can read, computers can’t
  - image processing pressing SOA, making these harder

- Note: often used not just for DOS prevention, but for protecting “free” services (email accounts)
DOS Prevention - Puzzles

• Make the solver present evidence of “work” done
  • If work is proven, then process request
  • Note: only useful if request processing significantly more work than

• Puzzle design
  • Must be hard to solve
  • Easy to Verify

• Canonical Example
  • Puzzle: given x-bits of output of h(r), where h is a cryptographic hash function
  • Solution: Invert h(r)
  • Q: Assume you are given 108 bits of output for 128-bit hash function, how hard would it be to solve the puzzle?
Worms
Worms

• A worm is a self-propagating program.

• As relevant to this discussion
  1. Exploits some vulnerability on a target host …
  2. (often) imbeds itself into a host …
  3. Searches for other vulnerable hosts …
  4. Goto (1)

• Q: Why do we care?
The Danger

• What makes worms so dangerous is that infection grows at an exponential rate

• A simple model:
  
  • $s$ (search) is the time it takes to find vulnerable host
  
  • $i$ (infect) is the time it takes to infect a host

• Assume that $t=0$ is the *worm outbreak*, the number of hosts at $t=j$ is

$$2^{j/(s+i)}$$

• For example, if $(s+i = 1)$, what is it at time $t=32$?
The result
The Morris Worm

- Robert Morris, a 23 doctoral student from Cornell
  - Wrote a small (99 line) program
  - November 3rd, 1988
  - Simply disabled the Internet

How it did it

- Reads /etc/password, they tries the obvious choices and dictionary, /usr/dict words
- Used local /etc/hosts.equiv, .rhosts, .forward to identify hosts that are related
  - Tries cracked passwords at related hosts (if necessary)
  - Uses whatever services are available to compromise other hosts
- Scanned local interfaces for network information
- Covered its tracks (set is own process name to sh, prevented accurate cores, re-forked itself)
Other scanning strategies

- The doomsday worm: a flash worm
  - Create a hit list of all vulnerable hosts
    - Staniford et al. argue this is feasible
    - Would contain a 48MB list
  - Do the infect and split approach
  - Use a zero-day vulnerability

- Result: saturate the Internet is less than 30 seconds!
Worms: Defense Strategies

- (Auto) patch your systems: most, if not all, large worm outbreaks have exploited known vulnerabilities (with patches)
- Heterogeneity: use more than one vendor for your networks
- Shield (Ross): provides filtering for known vulnerabilities, such that they are protected immediately (analog to virus scanning)
- Filtering: look for unnecessary or unusual communication patterns, then drop them on the floor
- This is the dominant method, getting sophisticated (Arbor Networks)
D/DOS (generalized by Mirkovic)

• Send a stream of packets/requests/whatever …
  • many PINGS, HTML requests, ...
• Send a few malformed packets
  • causing failures or expensive error handling
  • low-rate packet dropping (TCP congestion control)
  • “ping of death”
• Abuse legitimate access
  • Compromise service/host
  • Use its legitimate access rights to consume the rights for domain (e.g., local network)
  • E.g., First-year graduate student runs a recursive file operation on root of NFS partition
Distributed denial of service

- DDOS: Network oriented attacks aimed at preventing access to network, host or service
  - Saturate the target’s network with traffic
  - Consume all network resources (e.g., SYN)
  - Overload a service with requests
    - Use “expensive” requests (e.g., “sign this data”)
  - Can be extremely costly (e.g., Amazon)
- Result: service/host/network is unavailable
- Frequently distributed via other attack
- **Note**: IP is often hidden (spoofed)
The canonical DDOS attack

Internet

(master)

(zombies)

(adversary)

(router)

(LAN)

(target)
Adversary Network

(adversary) → (masters) → (zombies) → (target)
Why DDOS

• What would motivate someone DDOS?
  • An axe to grind …
  • Curiosity (script kiddies) …
  • Blackmail
  • Information warfare …

• Internet is an open system …
  • Packets not authenticated, probably can’t be
  • Would not solve the problem just move it (firewall)
  • Too many end-points can be remote controlled
Why is DDOS possible? (cont.)

- Interdependence - services dependent on each other
  - E.g., Web depends on TCP and DNS, which depends on routing and congestion control, …

- Limited resources (or rather *resource imbalances*)
  - Many times it takes few resources on the client side to consume lots of resources on the server side
  - E.g., SYN packets consume lots of internal resources

- You tell me .. (as said by Mirkovic et al.)
  - Intelligence and resources not co-located
  - No accountability
  - Control is distributed
• E2E (a simplified version): We should design the network such that all the intelligence is at the edges.
  • So that the network can be more robust and scalable
  • Many think is the main reason why the Internet works

• Downside:
  • Also, no real ability to police the traffic/content
  • So, many security solutions break this E2E by cracking open packets (e.g., application level firewalls)
  • DDOS is real because of this …
Q: An easy fix?

• How do you solve distributed denial of service?
Simple DDOS Mitigation

• Ingress/Egress Filtering
  – Helps spoofed sources, not much else

• Better Security
  – Limit availability of zombies, not feasible
  – Prevent compromise, viruses, …

• Quality of Service Guarantees (QOS)
  – Pre- or dynamically allocate bandwidth
  – E.g., diffserv, RSVP
  – Helps where such things are available …

• Content replication
  – E.g., CDS
  – Useful for static content
Pushback

• Initially, detect the DDOS
  – Use local algorithm, ID-esque processing
  – Flag the sources/types/links of DDOS traffic

• Pushback on upstream routers
  – Contact upstream routers using PB protocol
  – Indicate some filtering rules (based on observed)

• Repeat as necessary towards sources
  – Eventually, all (enough) sources will be filtered

• Q: What is the limitation here?
Traceback

• Routers forward packet data to source
  – Include packets and previous hop …
  – At low frequency (1/20,000) …

• Targets reconstruct path to source (IP unreliable)
  – Use per-hop data to look at
  – Statistics say that the path will be exposed

• Enact standard
  – Add filters at routers along the path
DDOS Reality

• None of the “protocol oriented” solutions have really seen any adoption
  – too many untrusting, ill-informed, mutually suspicious parties must play together well (*hint: human nature*)
  – solution have many remaining challenges

• Real Solution
  – Large ISP police there ingress/egress points very carefully
  – Watch for DDOS attacks and filter appropriately
    • e.g., BGP (routing) tricks, blacklisting, whitelisting
  – Products in existing that coordinate view from many points in the network to identify upswings in
  – Interestingly, this is the same way they deal with *worms* ...