Lecture 5
Large-Scale Network Attacks

ENGG5105/CSCI5470 Computer and Network Security

Spring 2014

Patrick P. C. Lee

Review of Last Lecture

- Many network attacks exploit the weaknesses of the fundamental design features of network protocols

Topics to Cover

- Port scanning
  - A technique that quickly searches for vulnerabilities on many hosts
- Worms
  - large-scale scanning for compromised hosts
- DDoS attacks
  - denial of service attacks from many compromised hosts
- Botnets
  - An army of compromised hosts

Our Goals

- To understand existing attack strategies of large-scale attacks
- To identify novel attacks that are potentially possible
- To help us develop defense strategies for existing and new attacks

Roadmap

- Port Scanning
- DoS
- DDoS
- Worms
- Botnets

Port Scanning

- Each network process on a single host is associated with a **port number**, so that incoming packets can identify which network process they are destined for.
  - It's like a door.
- Port numbers ranging from 0 – 1023 are called well-known port numbers and reserved for use by well-known applications
  - Examples:
    - HTTP: TCP 80
    - FTP: 20 (data), 21 (control)
    - DNS: 53
- Check: http://www.iana.org/assignments/port-numbers

Port Scanning

- Port scanning is to scan through a range of ports to identify any active network process
  - **Horizontal portscans**: scan a range of hosts on a specific port
  - **Vertical portscans**: scan a range of ports on a specific host
- An attacker scans for open ports to find vulnerable network processes to compromise

Nmap

- Nmap is a full-featured port scanning tool
  - used by operators to debug network services
- Nmap provides different scan types, each sends a specific type of packets to a target.
- Some scan types could cause the target system to be flooded or even crash under the load of strange and unusual types

Scan Types in Nmap

- **TCP connect() scan**:
  - completes the 3-way handshake with each scan port
  - not stealthy (i.e., source IP easily gets logged when a connection is open)
  - command line option: -sT

Scan Types in Nmap

- **TCP SYN scan**:
  - only sends the initial SYN and awaits the SYN-ACK response to determine if a port is open. If the port is closed, the destination will send a RST or nothing.
  - Stealthier than TCP connect() since only two packets are involved for open ports (while TCP connect() involves three packets)
  - command line option: -sS

Scan Types in Nmap

- **TCP FIN scan**:
  - sends a TCP FIN to each port. The FIN is unexpected as no connection is created before. A RESET indicates the port is closed, and no response may mean the port is open.
  - may be broken for systems that send RESET anyway
  - Even stealthier than TCP SYN (why?)
  - command line option: -sF

Scan Types in Nmap

- **OS Fingerprinting**:
  - Instead of scanning open ports, Nmap uses it to determine the OS type of the target based on their responses to illegal combinations of TCP control bits
  - command line option: -O

Defenses Against Port Scanning

- Close all unused ports
  - Use Nmap:
    - **`nmap localhost`**
  - Check for listening ports:
    - **`netstat –na | grep “LISTEN”`**
- Apply firewalls to block port accesses
  - iptables
- Use network intrusion detection system
  - Not blocking scanners, but detecting the presence of scanners

Roadmap

- Port Scanning
- DoS
- DDoS
- Worms
- Botnets

Denial-of-Service (DoS)

- In the last lecture, we focus on network attacks that gain access
- **Denial-of-service (DoS)** attacks deny access of legitimate users or stop critical system processes
- Launching DoS attacks has a business incentive
  - e.g., you want your competitor companies on their e-commerce websites

Ways of Launching DoS Attacks

- DoS attacks can be launched locally or remotely.
  - Locally: install a local account on a compromised machine
    - e.g., kill/crash a process, reconfigure systems
  - Remotely: attackers generate attack traffic to a target machine across the network
- Remote DoS attack is more prevalent

Types of DoS Attacks

- Stopping a service
  - Crash or shutting off a specific program or machine that users want to access
- Resource exhaustion
  - While the service is still running, attackers consume computer or network resources to prevent users from reaching the service
    - e.g., by flooding a large amount of packets
- In this lecture, we focus on network-based (remote) DoS attacks based on resource exhaustion

SYN Flood

- All TCP connections begin with 3-way handshake, in which a client first sends a SYN packet to an open port of a server
- When the server receives SYN, it allocates state to track the status of the half-open connection, and replies SYN-ACK
- In **SYN flood**, an attacker sends a large number of SYN packets to a target server to consume its resources

SYN Flood

- Attack SYN packets usually have spoofed IP addresses that are not existing (unresponsive)
  - Forcing the server to keep the state for a long while
  - By choosing a large pool of spoofed addresses, the detection will be difficult
- Two ways of resource exhaustion
  - Fill up the connection queue of the server
  - Fill up the communication link attached to the server

SYN Flood Defenses

- Provision enough resources
  - May not be feasible
- SYN Cookies
  - only need to be implemented on the TCP/IP stack of the server
  - carefully construct sequence numbers in the SYN-ACK field sent by the server
  - trade CPU for memory

SYN Cookies

- How traditional 3-way handshake works?
  - A → B: SYN(A, ISN_A)
  - B → A: SYN(B, ISN_B), ACK(A, ISN_A+1)
  - A → B: ACK(B, ISN_B+1)
- ISN_B is a cryptographic one-way function value of source/destination IP addresses, port numbers, time, and a secret seed. ISN_B is called a SYN cookie.
- When Bob replies SYN-ACK, he forgets the half-open state, i.e., no connection queue is used.

SYN Cookies

- When the ACK(B, ISN_B+1) arrives, Bob applies the same function to validate if ISN_B is legitimate. If so, the connection is established.
  - Need additional computation on hash
- Trudy sends spoofed SYN packets, but Bob doesn't store any state.
- Of course, Trudy may send a flood of ACKs to consume CPU resources of Bob. That's the tradeoff.

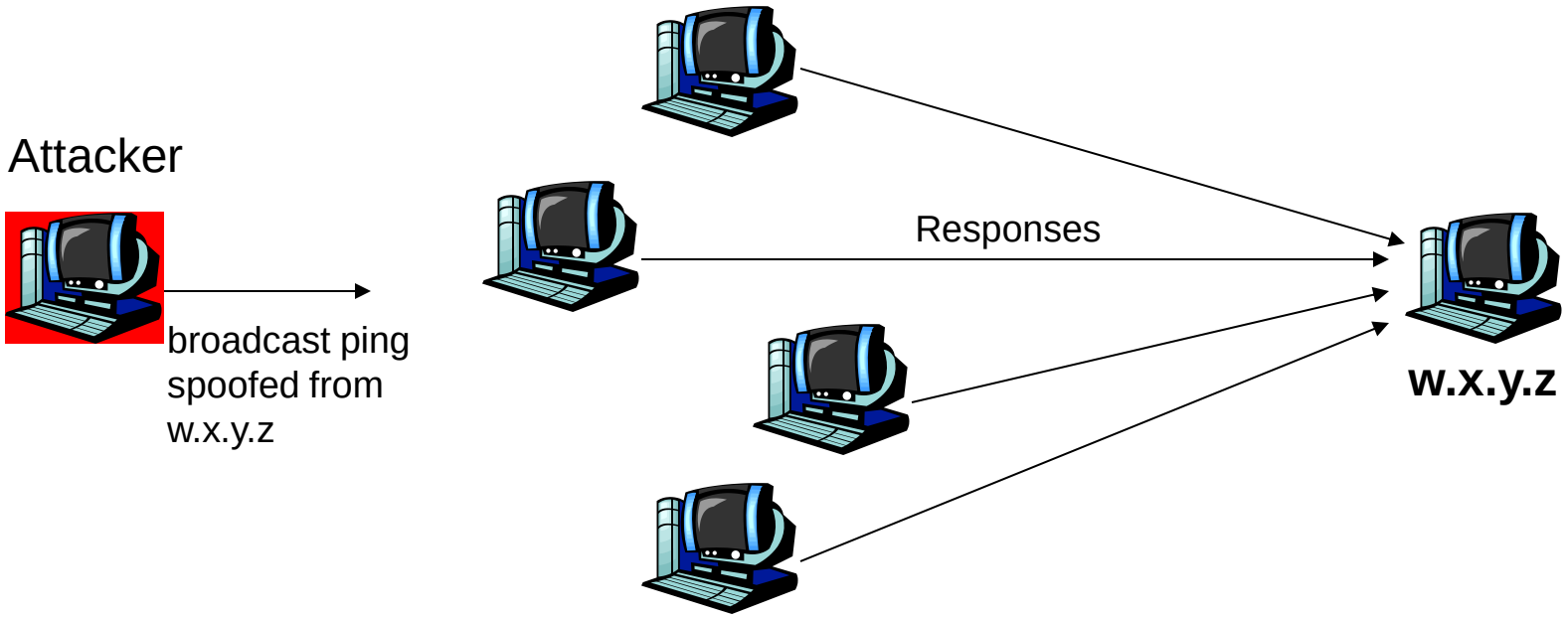
SYN Cookies

- In Linux, you can enable SYN cookies:
  - echo “1” > /proc/sys/net/ipv4/tcp_syncookies
- Other defenses against SYN flooding
  - traffic shaping: apply rate-limiting on routers to upper-bound the number of incoming SYN packets

Smurf Attacks

- Smurf attacks use direct broadcast to create a flood of traffic for a victim
- How broadcast works?
  - If a packet has IP address *.*.*.255, then it's a a broadcast packet
  - Directed broadcast: host part of an address is all 1's (e.g., 10.1.255.255 for 10.1.0.0/16)
  - When a switch/router receives a broadcast packet for a LAN, it sends a packet to every system on the LAN

Smurf Attacks



- An attacker sends a ping request to a broadcast address (ICMP Echo Request)
- Every host on the target network makes a replies to a victim.
- The victim is overloaded with many ping replies.
- **Defense:** disable directed broadcasts in routers

Roadmap

- Port Scanning
- DoS
- DDoS
- Worms
- Botnets

Distributed DoS

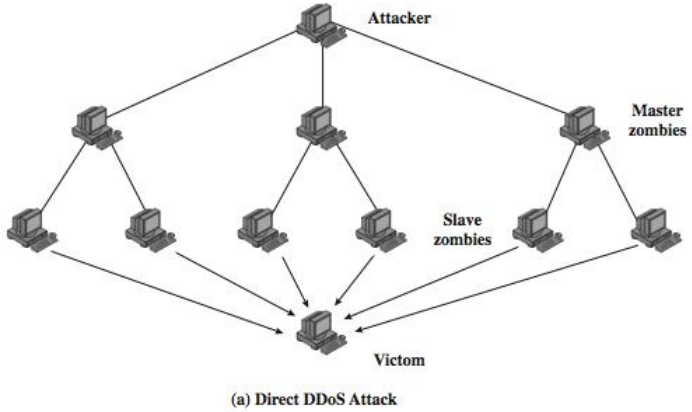
- DoS attacks usually refer to point-to-point attacks:
  - one attack source, one target
- **Distributed DoS (DDoS)** attacks usually refer to multipoint-to-point attacks:
  - an attacker controls multiple attack sources to attack a target
- But you need to control many attack sources first before launching DDoS attacks.
  - How???

DDoS

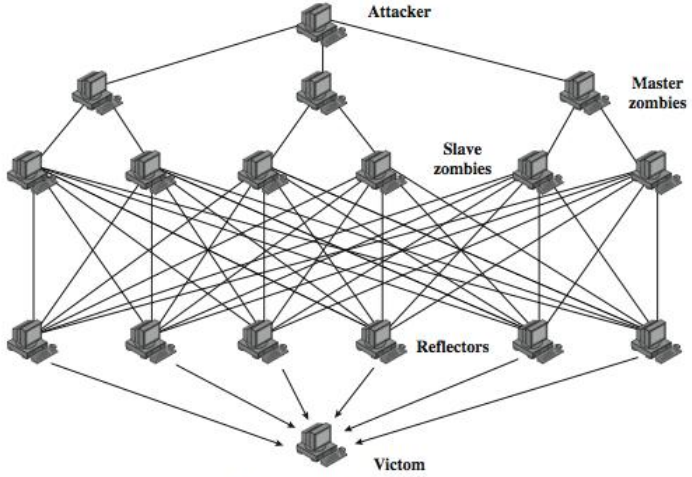
- DDoS is difficult to detect
  - There are many attack sources, so finding all of them is difficult
  - Each attack source only needs to send a small amount of attack traffic, but the aggregate can be overwhelming

DDoS Flood Types

- Direct DoS attacks:
  - instruct zombies to send flood traffic
- Reflector DoS attacks
  - Send request packets to uninfected machines, while packets have spoofed source IP address set to the target
  - All uninfected machines make replies to the target
  - e.g., send DNS queries to DNS servers



(a) Direct DDoS Attack



(b) Reflector DDoS Attack

DDoS Countermeasures

- three broad lines of defense:
  1. attack prevention & preemption (before)
  2. attack detection & filtering (during)
  3. attack source traceback & ident (after)
- huge range of attack possibilities
- hence evolving countermeasures

Roadmap

- Port Scanning
- DoS
- DDoS
- Worms
- Botnets

Viruses

- piece of software that infects programs
  - modifying them to include a copy of the virus
  - so it executes secretly when host program is run
- specific to operating system and hardware
  - taking advantage of their details and weaknesses

Worms

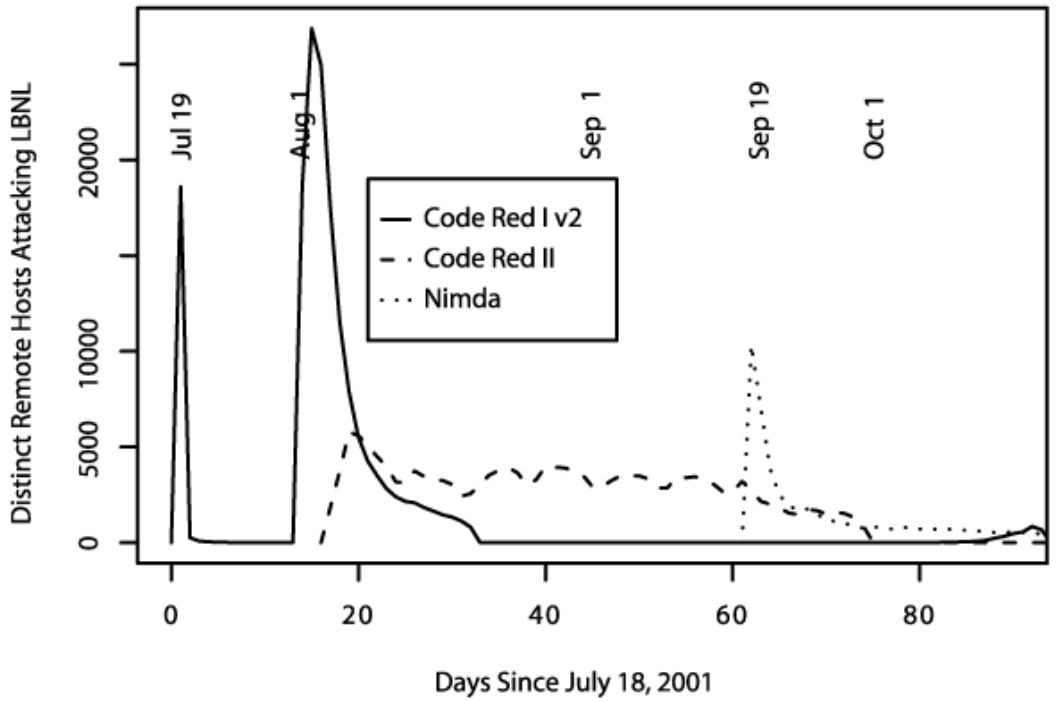
- How do you control many attack sources first before launching DDoS attacks?
- **Worms** are programs that self-propagate across the Internet by exploiting security flaws in widely-used services
  - E.g., by port scanning
- Worms are different from viruses, where the latter require some sort of user actions to abet their propagation
  - but some worms also requires user to trigger, but their propagation are done by themselves
  - won't cover viruses in details in this course.

Morris Worm

- one of best known worms
- released by Robert Morris in 1988
- various attacks on UNIX systems
  - cracking password file to use login/password to logon to other systems
  - exploiting a bug in the finger protocol
  - exploiting a bug in sendmail
- if succeed have remote shell access
  - sent bootstrap program to copy worm over

How Serious are Worms?

From [Staniford et al., 2002]



- Onset of Code Red I v2, code Red II and Nimda: Number of remote hosts launching confirmed attacks corresponding to different worms, as seen at the Lawerence Berkeley National Laboratory.

Code Red I

- Launched on July 13, 2001 (version 2 launched on July 19, 2001).
- Once Code Red I infected a host, it spreads by launching 99 threads that generated random IP addresses.
- v2 launches payload to www.whitehouse.gov

Worm Propagation Model

- Propagation model of Code Red I is based on **Random Constant Spread (RCS)**
- How to provide a mathematical model for it?
- Let
  - **N** = number of vulnerable hosts in the Internet
  - **K** = compromise rate: number of vulnerable hosts which an infected host can find and compromise per unit time
  - **a** = proportion of vulnerable machines that have been compromised

Worm Propagation Model

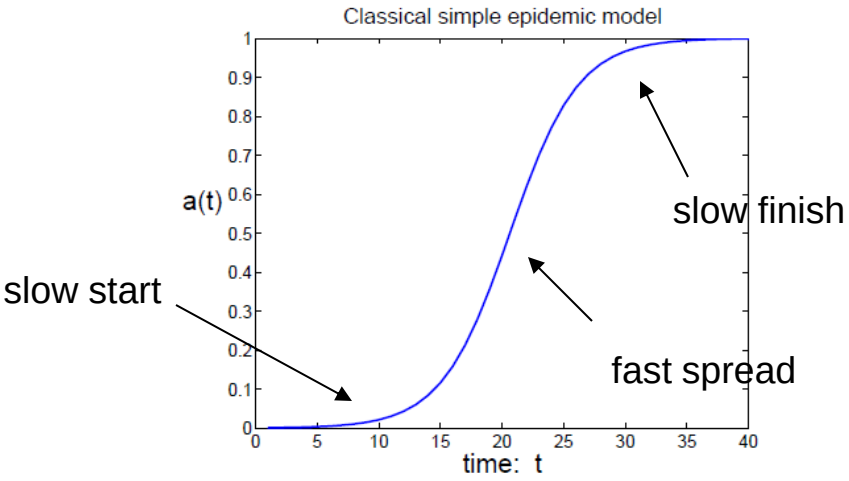
- We can construct a differential equation:

$$N\,da = (Na)\,K(1 - a)dt$$

Number of machines compromised in the next increment of time (N da) is equal to the number of machines already compromised (Na) times the number of machines each compromised machine can compromise (K(1-a)dt).

Worm Propagation Model

- **da/dt = Ka(1-a)**
- Three phases:
  - Slow-start phase:
    - if a is small, da/dt ~ Ka (exponential growth)
  - Fast-spread phase:
    - da/dt is max when a=0.5
  - Slow-finish phase:
    - if a → 1, da/dt ~ 0 (all vulnerable machines are compromised and no leftover)



K = 1.8

From "Code Red Worm Propagation Modeling and Analysis", CCS 2002

Code Red II – Better Worm

- Code Red I chose a random IP address from the Internet and attempted to infect it
- Code Red II (released on Aug 4, 2001) used a localized scanning strategy, which scans the addresses that close to it
  - probability 3/8 to choose an IP from the /16 network of the infected machine itself
  - probability 1/2 to choose an IP from its own /8
  - probability 1/8 to choose a random address from the Internet

Nimda – Better Worm

- Nimda (released on Sep 18, 2001) used different methods for its propagation
  - probing for a Microsoft IIS vulnerability
  - bulk emailing of itself as an attachment
  - copying itself across open network shares
  - adding exploit code to Web pages
  - scanning the backdoors left by Code Red II

Recent Worm Attacks

- SQL Slammer
  - early 2003, attacks MS SQL Server
- Mydoom
  - mass-mailing e-mail worm that appeared in 2004
  - installed remote access backdoor in infected systems
- Warezov family of worms
  - scan for e-mail addresses, send in attachment

Mobile Phone Worms

- first appeared on mobile phones in 2004
  - target smartphone
- they communicate via Bluetooth or MMS
- to disable phone, delete data on phone, or send premium-priced messages
- CommWarrior, launched in 2005
  - replicates using Bluetooth to nearby phones
  - and via MMS using address-book numbers

Worm Countermeasures

- overlaps with anti-virus techniques
- once worm on system A/V can detect
- worms also cause significant net activity
- worm defense approaches include:
  - signature-based worm scan filtering
  - filter-based worm containment
  - payload-classification-based worm containment
  - threshold random walk scan detection
  - rate limiting and rate halting

Roadmap

- Port Scanning
- DoS
- DDoS
- Worms
- Botnets

Botnets

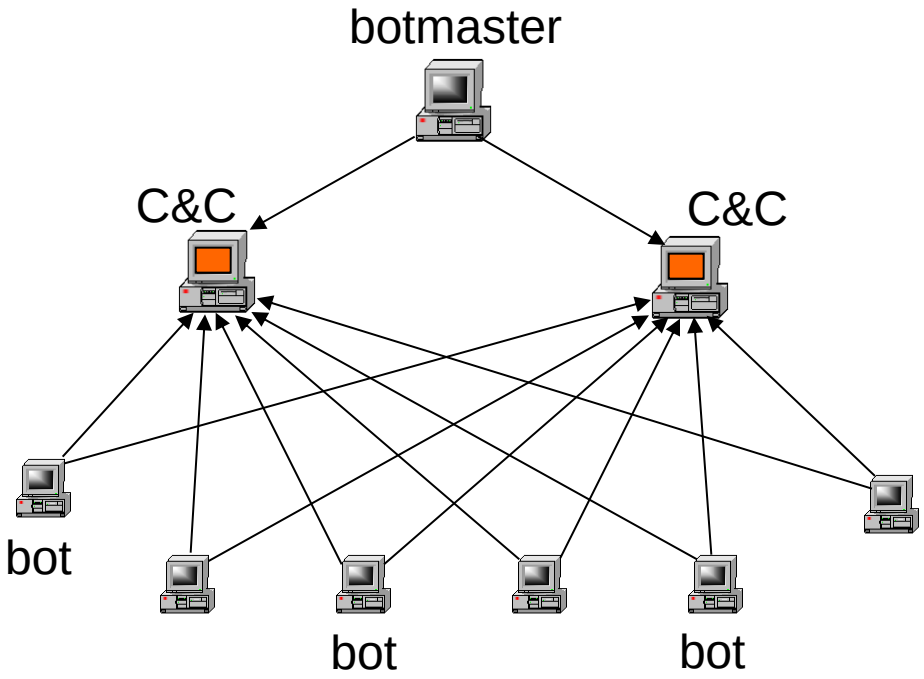
- Using worms, an attacker can compromise millions of hosts, and form a botnet
- A bot is a compromised host that can be remotely controlled by an attacker (called botmaster)
- A botnet is a network of bots (or bot army)

Centralized Botnets

- Most existing botnets use a centralized architecture
- A botmaster sends commands to Command and Control (C&C) servers, which relay the commands to bots
  - e.g., the command could be “send junk packets to info.gov.hk”
  - using Internet Relay Chat (IRC) protocol – send commands, receive replies
- Weaknesses:
  - Easier to detect
  - Single point of failure (from an attacker’s view)

Centralized Botnets

- Centralized C&C-based Botnet



A defender can fix one of the C&C servers and find out where the botmaster is

Decentralized Botnets

- **Peer-to-peer (P2P):** a botmaster sends commands to its peer bot through an existing P2P network
  - More resilient connectivity
  - Difficult to be detected
- **Unstructured**: pass along encrypted commands to a random peer (may or may not be bot). If the peer is a bot, it decrypts and executes the commands.

Attacks and Theft

- Why creating a botnet?
  - Launch DDoS attacks to enemies
  - Collect personal information (e.g., credit card numbers) and make business
  - Send spam (most important use)
  - Create phishing websites

Honeypots

- A honeypot is a decoy system to lure attackers
  - unpatched system so as to be attacked
  - away from accessing critical systems
  - to collect information of attackers' activities
  - to encourage attackers to stay on system so administrator can respond
- Usually deployed as a virtual machine
- A **honeynet** is a network of honeypots

References

- “Counter Hack Reloaded”, Ch. 7, 9
- Stallings, Ch. 21
- Staniford et al., “How to 0wn the Internet in your Spare Time”, Security ’02
- Bailey et al., “A Survey of Botnet Technology and Defenses”, 2009