CSC 458/2209 – Computer Networks

Handout # 21
Network Security & Final Review

Professor Yashar Ganjali
Department of Computer Science
University of Toronto

yganjali@cs.toronto.edu
http://www.cs.toronto.edu/~yganjali
Announcements

- Programming Assignment 2
  - Deadline: Friday Dec. 11th, 2015
  - CDF downtime:
    - Thursday Dec. 10th, 9PM to 11AM on Friday Dec. 11th

- Final exam time:
  - Monday 14 Dec. 2015, 9am-12pm
  - Please check class web site for location.
  - Sample problems posted online.

- This week’s tutorial: final exam review
- Regular office hours today, no office hours tomorrow
Today

- Network security
- Final review
Connectivity: Good vs. Evil

- Network have improved significantly: in terms of bandwidth and latency
  - **Good**
    - We can communicate
    - Exchange information
    - Transfer data
    - ...
  - **Evil**
    - It’s easier to do harm
    - Harmful code can propagate faster
    - Information collection, violating privacy
    - ...
Life Just Before Slammer
Life Just After Slammer

Map source: www.caida.org

http://www.caida.org
Copyright (C) 2003 UC Regents

Sat Jan 25 06:00:00 2003 (UTC)
Number of hosts infected with Sapphire: 74855
A Lesson in Economy

- Slammer exploited connectionless UDP service, rather than connection-oriented TCP.
- Entire worm fit in a single packet! (376 bytes)
  - When scanning, worm could “fire and forget”.
  - Stateless!
- Worm infected 75,000+ hosts in 10 minutes (despite broken random number generator).
  - At its peak, doubled every 8.5 seconds.
- Progress limited by the Internet’s carrying capacity (= 55 million scans/sec)
Why Security?

- First victim at 12:45 am
- By 1:15 am, transcontinental links starting to fail
- 300,000 access points downed in Portugal
- All cell and Internet in Korea failed (27 million people)
- 5 root name servers were knocked offline
- 911 didn’t respond (Seattle)
- Flights canceled
Witty Worm
Witty Worm – Cont’d

- Attacks firewalls and security products (ISS)
- First to use vulnerabilities in security software
- ISS announced a vulnerability
  - buffer overflow problem
  - Attack in just one day!
- Attack started from a small number of compromised machines
- In 30 minutes 12,000 infected machines
  - 90 Gb/s of UDP traffic
Detecting Attacks

- How can we identify and measure attacks like Witty and Slammer?
Network Telescope

- Large piece of globally announced IP addresses
- No legitimate hosts (almost)
- Inbound traffic is almost always anomalous
- 1/256th of the all IPv4 space
  - One packet in every 256 packets if unbiased random generators used.
- Provides global view of the spread of Internet worms.

Question. Can this system identify attacks in real time?
Today

Network Security Goals

- Security vs. Internet Design
- Attacks
- Defenses
Network Security Goals

• Availability
  • Everyone can reach all network resources all the time

• Protection
  • Protect users from interactions they don’t want

• Authenticity
  • Know who you are speaking with

• Data Integrity
  • Protect data en-route

• Privacy
  • Protect private data
Today

- Network Security Goals
- Security vs. Internet Design
- Attacks
- Defenses
Internet Design

- Destination routing
- Packet based (statistical multiplexing)
- Global addressing (IP addresses)
- Simple to join (as infrastructure)
- Power in end hosts (end-to-end argument)
- “Ad hoc” naming system
Internet Design vs. Security

- Destination routing
  - Keeps forwarding tables small
  - Simple to maintain forwarding tables
  - How do we know where packets are coming from?
    - Probably simple fix to spoofing, why isn’t it in place?

- Packet based (statistical multiplexing)
- Global addressing (IP addresses)
- Simple to join (as infrastructure)
- Power in end hosts (end-to-end argument)
- “Ad hoc” naming system
Internet Design vs. Security

- Destination Routing
- Packet Based (statistical multiplexing)
  - Simple + Efficient
  - Difficult resource bound per-communication
    - How to keep someone from hogging? (remember, we can’t rely on source addresses)
- Global Addressing (IP addresses)
- Simple to join (as infrastructure)
- Power in End Hosts (end-to-end argument)
- “Ad hoc” naming system
Internet Design vs. Security

- Destination routing
- Packet based (statistical multiplexing)
- Global Addressing (IP addresses)
  - Very democratic
  - Even people who don’t necessarily want to be talked to
    - “every psychopath is your next door neighbor” – Dan Geer
- Simple to join (as infrastructure)
- Power in end hosts (end-to-end argument)
- “Ad hoc” naming system
Internet Design vs. Security

- Destination routing
- Packet based (statistical multiplexing)
- Global addressing (IP addresses)
- Simple to join (as infrastructure)
  - Very democratic
  - Misbehaving routers can do very bad things
    - No model of trust between routers
- Power in End Hosts (end-to-end argument)
- “Ad hoc” naming system
Internet Design vs. Security

- Destination routing
- Packet based (statistical multiplexing)
- Global addressing (IP addresses)
- Simple to join (as infrastructure)

- Power in end-hosts (end-to-end argument)
  - Decouple hosts and infrastructure = innovation at the edge!
  - Giving power to least trusted actors
    - How to guarantee good behavior?
- “Ad hoc” naming system
Internet Design vs. Security

- Packet Based (statistical multiplexing)
- Destination Routing
- Global Addressing (IP addresses)
- Simple to join (as infrastructure)
- Power in End Hosts (end-to-end argument)
- “Ad hoc” naming system
  - Seems to work OK
  - Fate sharing with hierarchical system
  - Off route = more trusted elements
Today

- Network Security Goals
- Security vs. Internet Design

Attacks

- How attacks leverage these weaknesses in practice
  - Denial of service
  - Indirection
  - Reconnaissance

- Defenses
DoS: Via Resource Exhaustion

- Uplink bandwidth
- Downlink bandwidth
- CPU
- User-time
- Memory (e.g. TCP TCB exhaustion)
DoS: Via Resource Exhaustion

- Uplink bandwidth
  - Saturate uplink bandwidth using legitimate requests (e.g. download large image)
  - Solution: use a CDN (Akamai)
  - Solution: admission control at the server (not a network problem??)

- CPU time similar to above

- Victim Memory
  - TCP connections require state, can try to exhaust
  - E.g. SYN Flood (next few slides)
Who Is Responsible?

- Can we rely on the attack victim to stop DoS attacks?
- If not, who can do this?
- How?
- Which resource is cheaper?
  - Bandwidth, or
  - CPU
TCP Handshake

C
SYN_C
SYN_S, ACK_C
ACK_S
S
Listening
Store data
Wait
Connected
Example: SYN Flooding

C

SYN_{C1} → SYN_{C2} → SYN_{C3} → SYN_{C4} → SYN_{C5}

S

Listening
Store data
Protection against SYN Attacks

- SYN Cookies
  - Client sends SYN
  - Server responds to Client with SYN-ACK cookie
    - sqn = f(src addr, src port, dest addr, dest port, rand)
    - Server does not save state
  - Honest client responds with ACK(sqn)
  - Server checks response
    - If matches SYN-ACK, establishes connection
- Drop Random TCB in SYN_RCVD state (likely to be attackers)

[Bernstein, Schenk]
Distributed DoS (DDoS)

- Attacker compromises multiple hosts
- Installs malicious program to do her bidding (bots)
- Bots flood (or otherwise attack) victims on command; Attack is coordinated
- Bot-networks of 80k to 100k have been seen in the wild
  - Aggregate bandwidth > 20Gbps (probably more)
- E.g. Blue Frog (by Blue Security)
Blue Frog

- Anti-spam tool:
  - Persuade spammers to remove community members’ addresses from their mailing list
- Users register: Do Not Intrude Registry, Firefox, and IE plugins
- Automatic reports: ISPs, law-enforcement, ...
- Spammers attacked
  - Intimidating e-mails
  - DDoS attack to “Blue Security” web page
  - Redirected to blogs.com → Collapse
  - Attackers identified
- Blue Security ceased its anti-spam operation.
What About Downlink? (Flooding)

- Assume attacker generates enough traffic to saturate downlink bandwidth.
- What can the server do?
- What can the network do?
  - Ideally want network to drop bad packets
  - How to tell if a packet is part of a legitimate flow? (requires per flow state?)
  - Even harder, how to tell if a SYN packet is part of a legitimate request?
- Is the phone network immune to such attacks?
DoS Aplenty

- Attacker guesses TCP seq. number for an existing connection:
  - Attacker can send Reset packet to close connection. Results in DoS.
  - Most systems allow for a large window of acceptable seq. #'s
  - Only have to a land a packet in
  - Attack is most effective against long lived connections, e.g. BGP.
- Congestion control DoS attack

- Generate TCP flow to force target to repeatedly enter retransmission timeout state
- Difficult to detect because packet rate is low
Indirection Attacks

- Rely on connecting to “end-points” to get content/access services

- Unfortunately network end-points (e.g. IPs, DNS names) are loosely bound

- Long history of problems
Example: Fetching a Web Page

Client

DHCP Request

ARP request (name server/gateway)

DNS request

HTTP Request
DNS Vulnerability

- Users/hosts typically trust the host-address mapping provided by DNS
Bellovin/Mockapetris Attack

- Trust relationships use symbolic addresses
  - /etc/hosts.equiv contains friend.stanford.edu
- Requests come with numeric source address
  - Use reverse DNS to find symbolic name
  - Decide access based on /etc/hosts.equiv, ...

- Attack
  - Spoof reverse DNS to make host trust attacker
Reverse DNS

- Given numeric IP address, find symbolic addr

- To find 222.33.44.3,
  - Query 44.33.222.in-addr.arpa
  - Get list of symbolic addresses, e.g.,
    1 IN PTR server.small.com
    2 IN PTR boss.small.com
    3 IN PTR ws1.small.com
    4 IN PTR ws2.small.com
Attack

- Gain control of DNS service for evil.org
- Select target machine in good.net
- Find trust relationships
  - SNMP, finger can help find active sessions, etc.
  - Example: target trusts host1.good.net
- Connect
  - Attempt rlogin from coyote.evil.org
  - Target contacts reverse DNS server with IP addr
  - Use modified reverse DNS to say “addr belongs to host1.good.net”
  - Target allows rlogin
DNS Rebinding Attacks

- Modern browsers implement the same-origin policy.
  - Isolate distinct origins.
- To attack:
  - Subvert the same-origin policy
  - Confuse browser to aggregate network resources.
- DNS Rebinding Attacks:
  - register a domain, e.g. attacker.com
  - Answer DNS queries for attacker.com with your IP, short TTL, serve malicious JavaScript
  - Script requests IP address of attacker.com, feed the IP of private server
  - Read private information
Solution – DNS Pinning

- Once a hostname is resolved to an IP address, cache the result for a while
  - Regardless of TTL

- Plug-ins can cause problems
TCP Connection Spoofing

- Each TCP connection has an associated state
  - Client IP and port number; same for server
  - Sequence numbers for client, server flows

Problem

- Easy to guess state
  - Port numbers are standard
  - Sequence numbers (used to be) chosen in predictable way
IP Spoofing Attack

- A, B trusted connection
  - Send packets with predictable seq numbers
- E impersonates B to A
  - Opens connection to A to get initial seq number
  - SYN-floods B’s queue
  - Sends packets to A that resemble B’s transmission
  - E cannot receive, but may execute commands on A

- Other ways to spoof source IP?
Reconnaissance/Misc

- To attack a victim, first discover available resources
- Many commonly used reconnaissance techniques
  - Port scanning
  - Host/application fingerprinting
  - Traceroute
  - DNS (reverse DNS scanning, Zone transfer)
  - SNMP
- These are meant for use by admins to diagnose network problems!
  - Trade-off between the ability to diagnose a network and reveal security sensitive information
Anecdotes ...

- Large bot networks exist that scan the Internet daily looking for vulnerable hosts (at least 16,000 participating hosts)
- Old worms still endemic on Internet (e.g. Code Red)
  - Seem to come and go in mass
  - Surreptitious scanning effort?
Today

- Network Security Goals
- Security vs. Internet Design
- Attacks

Defenses
Firewalls

- Keep out unwanted traffic
- Can be done in the network (e.g. network perimeter) or at the host
- Many mechanisms
  - Packet filters
  - Stateful packet filters
  - Proxies, gateways
Packet Filters

- Make a decision to drop a packet based on packet header
  - Protocol type
  - Transport ports
  - Source/Dest IP address
  - Etc.

- Usually done on router at perimeter of network
- And on virtually all end-hosts today
Packet Filters: Problem

- Assume firewall rule (allow from port 53 and port 80)
- Easy for an attacker to send packets from port 53 or 80
- Further attacker can forge source
- Not very effective for stopping packets from unwanted senders
Stateful Packet Filter

- Idea: Only allow traffic initiated by client
  - For each flow request (e.g. SYN or DNS req) keep a little state
  - Ensure packets received from Internet belong to an existing flow
  - To be effective must keep around sequence numbers per flow
- Very common, used in all NAT boxes today
  - Stateful NATs downside: failure → all connection state is lost!
Proxies

- Want to look “deeper” into packets
  - Application type
  - Content
- Can do by reconstructing TCP flows and “peering” in, however this is really hard
  - (Digression next slide)
Passive Reconstruction of TCP Stream

- Use passive network element to reconstruct TCP streams
- “Peer” into stream to find harmful payload (e.g. virus signatures)

- Why is this really hard?
Reconstructing Streams

- Must know the client’s view of data
- Have to know if packet reaches destination (may not if TTL is too short)
- Have to know how end-host manages overlapping TCP sequence numbers
- Have to know how end-host manages overlapping fragments
Proxies

- Full TCP termination in the network
- Often done transparently (e.g. HTTP proxies)
- Allows access to objects passed over network
  - E.g. files, streams etc.
- Does not have same problems as stream reconstruction
- Plus can do lots of other fun things
  - E.g. content caching
Proxy Discussion

- Proxies duplicate per-flow state held by clients
- How does this break end-to-end semantics of TCP?
  - E.g. what if proxy crashes right after reading from client? (lost data!)
- How to fix?
  - Lots of work in this area
Final Comments

- Internet not designed for security
- Many, many attacks
  - Defense is very difficult
  - Attackers are smart; Broken network aids them!
- Retrofitting solutions often break original design principles
  - Some of these solutions work, some of the time
  - Some make the network inflexible, brittle
- Time for new designs/principles?
Final Review

- Final exam logistics
- Review of principles
- Where next?
Final Exam Logistics

- When: Monday December 14th, 9AM-12PM
- Location: please check course web site

- Examination aids allowed:
  - Non-programmable calculators
  - 1 double-sided page of notes

- No cell phones allowed
Final Exam

- Part I – Multiple choice
  - 1 correct answer for each question
- Part II – Comparisons
  - 4-5 sentences each
- Part III – Longer Questions
  - Might need more time than Part I & II
  - Still very simple problems
  - Similar to midterm and problem sets
Final Review

- Final exam logistics
- Review of principles
- Where next?
Review of Basic Concepts

The 7-layer OSI Model

Application
Presentation
Session
Transport
Network
Link
Physical

FTP
ASCII/Binary
TCP
IP
Ethernet

The 4-layer Internet model

Application
Transport
Network
Link
Example: FTP over the Internet
Using TCP/IP and Ethernet

1. App
   "A" U of T

2. OS

3. Ethernet

4. R1

5. 6
6. 7

5. 6

6. 7

7. 8
8. 9
9. 10

8. 9

9. 10

10. R2

11. R3

12. R4

13. R5

14. 15
15. 16
16. 17
17. 18
18. 19
19. 20

"B" Stanford

20. App

Using TCP/IP and Ethernet
Review of Basic Principles

- Basic ideas:
  - Packet switching, statistical multiplexing, layering,
- Link Layer:
  - Channel capacity, encoding and clock recovery, error detection/correction, Ethernet switching
- Network Layer:
  - Fragmentation, Bellman-Ford, Dijkstra, addresses and lookups, BGP, IGP
Review of Basic Principles – Cont’d

- **Transport Layer:**
  - Flow control, congestion control, retransmissions and sliding windows, congestion avoidance (RED)

- **Miscellaneous:**
  - Queuing mechanisms, middleboxes, peer-to-peer, software-defined networking, and network security
Final Review

- Final exam logistics
- Review of principles

Where next?
Where Next?

Courses to take:

- CSC2203: Packet Switch & Network Architectures
- CSC2229: Software-Defined Networking
- CSC309: Programming on the Web
- CSC2231: Special Topics in Computer Systems
  - Online Social Networking Systems
  - Internet Systems and Services
- CSC2206: Systems Modeling and Analysis
- CSC2221: Theory of Distributed Computing
- CSC2415: Advanced Topics in Distributed Computing
- CSC2720: Systems Thinking for Global Problems

Individual study courses

- CSC494 and CSC495
Thank You!