Week 10: Lecture B
Network Denial of Service
Thursday, November 2, 2023
Announcements

- **Project 3: WebSec** released
  - **Deadline:** November 9th by 11:59PM (**next Thursday**)
Project 3 progress

- Working on Part 1: 0%
- Finished Part 1, working on Part 2: 0%
- Finished Part 2, working on Part 3: 0%
- Finished with everything!: 0%
- Haven't started yet :( : 0%
Announcements

See Discord for meeting info!

www.utahsec.com
Announcements

See Discord for meeting info!

www.utahsec.com
Questions?
Last time on CS 4440...

Application Layer Attacks
HTTP Content Injection
SMTP Header Spoofing
DNS Hijacking
Application Layer Attacks

- **Application Layer:**
  - ???
Application Layer Attacks

- **Application Layer**: where network-facing apps send/receive message
  - Application-specific protocols (message semantics, structure, processing rules, etc.)

- **Attacking the application layer**:
  - ???
Application Layer Attacks

- **Application Layer**: where network-facing apps send/receive messages
  - Application-specific protocols (message semantics, structure, processing rules, etc.)

- **Attacking the application layer:**
  - **Command Injection**
    - SQL injection, CSRF, XSS
  - **Denial of Service**
    - Crash a remote application
    - Prevent others from using it
  - **Message Tampering / Sniffing**
    - Injecting data into messages
    - Capturing unencrypted data
  - **Other protocol-specific attacks**
The HTTP Protocol

- What is HTTP?
The HTTP Protocol

- **What is HTTP?**
  - Protocol for transmitting hypermedia documents (e.g., web pages)

- **HTTP’s Characteristics:**
  - Widely used
  - Simple
  - Unencrypted
HTTP Tampering

- Attack: ???

Request → Alter

CS 4440: Introduction to Computer Security
This course teaches the security mindset and introduces the principles and practices of computer security as applied to software, host systems, and networks. It covers the foundations of building, using, and managing secure systems. Topics include standard cryptographic functions and protocols, threats and defenses for real-world systems, incident response, and computer forensics.

Professor: Stefan Nagy
HTTP Tampering

- **Attack:** exploit HTTP’s insecurity
  - Nothing is encrypted!

- **Attacker intercepts** requested webpage and **modifies it**
  - User receives modified webpage

- **Attacker capabilities?**
  -Inject malicious **content**
  -Inject malicious **code**
Thwarting HTTP Injection

- Defenses: ???
Thwarting HTTP Injection

- **Defenses:** encrypt everything all the time!

**Answer:** *completely ditch HTTP!*

- As web and app developers, enforce strict HTTPS compliance
  - Necessary to prevent HTTPS→HTTP **downgrade** attacks
The SMTP Protocol

- **SMTP**: Simple Mail Transfer Protocol
  - Implemented in the **application** layer

- **Characteristics:**
  - Text-based
  - Connection-oriented
  - Uses TCP ports 25/587

- **Security guarantees:**
  - ???
The SMTP Protocol

- **SMTP**: Simple Mail Transfer Protocol
  - Implemented in the *application* layer

- **Characteristics:**
  - Text-based
  - Connection-oriented
  - Uses TCP ports 25/587

- **Security guarantees:**
  - Message integrity—**no!**
  - Confidentiality—**no!**
  - Authentication—**no!**
The SMTP Protocol

- **No message integrity**
  - ???

- **No confidentiality**
  - ???

- **No authentication**
  - ???
The SMTP Protocol

- **No message integrity**
  - Tamper with messages
  - Block messages

- **No confidentiality**
  - Find sender/recipient
  - Read message contents

- **No authentication**
  - Spoof sender identity
SMTP Header Spoofing

- **Attack:** ???

```plaintext
S: 220 attacker.com SMTP Exim
C: HELO attacker.com
S: 250 Hello attacker.com
C: MAIL FROM: <ceo@company.com>
S: 250 0k
C: RCPT TO: <bob@company.com>
S: 250 Accepted
C: DATA
S: 354 Enter a message, ending with "." on a line by itself
C: Subject: Download this urgently
C: From: ceo@company.com
C: To: bob@company.com
C:
C: Hi Bob,
C: Please download this urgently: https://some-malicious-link.com
C: Regards
C: .
S: 250 OK
C: QUIT
S: 221 attacker.com closing connection
```

```plaintext
To: robertbateman@email.com
Subject: Hi There
From: "Mickey Mouse" <m.mouse@disney.com>
X-Priority: 3 (Normal)
Importance: Normal
Errors-To: m.mouse@disney.com
Reply-To: m.mouse@disney.com
Content-Type: text/plain
```
SMTP Header Spoofing

- **Attack**: spoof SMTP header to **mislead recipient** about sender of the email

```plaintext
S: 220 attacker.com SMTP Exim
C: HELO attacker.com
S: 250 Hello attacker.com
C: MAIL FROM: <ceo@company.com>
S: 250 0k
C: RCPT TO: <bob@company.com>
S: 250 Accepted
C: DATA
S: 354 Enter a message, ending with “.” on a line by itself
C: Subject: Download this urgently
C: From: ceo@company.com
C: To: bob@company.com
C:
C: Hi Bob,
C: Please download this urgently
C: Regards
C: .
S: 250 OK
C: QUIT
S: 221 attacker.com closing connection
```

Fake Sender

Victim

Malicious Link

```plaintext
To: robertbateman@email.com
Subject: Hi There
From: "Mickey Mouse" <m.mouse@disney.com>
X-Priority: 3 (Normal)
Importance: Normal
Errors-To: m.mouse@disney.com
Reply-To: m.mouse@disney.com
Content-Type: text/plain

https://some-malicious-link.com
```
Thwarting Email Spoofing

- Defenses: ???
Thwarting Email Spoofing

- **Checking email bodies**
  - Included links
  - Attached files
  - Text analysis (e.g., known spam campaigns)

- **Checking email headers**
  - Egress server domain registration
    - Check that sender is who it says it is
  - Pretty Good Privacy (PGP)
    - Sender and Receiver authentication
    - Confidentiality
    - Integrity
Identification on the Web

- How do we identify people?
  - Social security numbers
  - Passports, drivers licenses
  - Their unique fingerprints

- How can we identify internet hosts?
  - ???
Identification on the Web

- **How do we identify people?**
  - Social security numbers
  - Passports, drivers licenses
  - Their unique fingerprints

- **How can we identify internet hosts?**
  - **Network layer**: location via **IP addresses**
    - A logical addressing system
    - 32-bit (IPV4) addressing datagrams
  - **What you care about**: ???
Identification on the Web

- How do we identify **people**?
  - Social security numbers
  - Passports, drivers licenses
  - Their unique fingerprints

- How can we identify **internet hosts**?
  - **Network layer**: location via **IP addresses**
    - A logical addressing system
    - 32-bit (IPV4) addressing datagrams
  - **What you care about**: the domain name
    - E.g., www.wikipedia.org
The Domain Name System

- **Distributed database** implemented in hierarchy of many name servers

- **Application-layer protocol:**
  - Hosts and domain name servers communicate to resolve **domain names**
    - Address–name translation

- **Result:** user requests ???
  - But their host really gets ???
The Domain Name System

- **Distributed database** implemented in hierarchy of many name servers

- **Application-layer protocol:**
  - Hosts and domain name servers communicate to resolve **domain names**
    - Address–name translation

- **Result:** user requests **domain name**
  - But their host really gets its **IP address**
  - Convenient!
How can we optimize DNS resolution?

- ???
How can we optimize DNS resolution?
- Cache look-ups to amortize initial look-up, reduce system load

Optimization 1: ???

Optimization 2: ???

www.google-analytics.com
Record Name: www.google-analytics.com
Record Type: 5
Time To Live: 104
Data Length: 4
Section: Answer
CNAME Record: www.google-analytics.l.google.com
The Domain Name System

How can we optimize DNS resolution?
- Cache look-ups to amortize initial look-up, reduce system load

Optimization 1: temporal locality
- www.espn.com/page1
- www.espn.com/page2

Optimization 2: popular domains
- google.com
- Facebook.com
The Domain Name System

First Lookup (non-cached)

```
$ time nslookup facebook.com
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: facebook.com
Address: 31.13.70.36
Name: facebook.com
Address: 2a03:2880:f10d:83:face:b00c:0:25de

real 0m0.474s
user 0m0.000s
sys 0m0.015s
```

Second Lookup (cached)

```
$ time nslookup facebook.com
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: facebook.com
Address: 31.13.70.36
Name: facebook.com
Address: 2a03:2880:f10d:83:face:b00c:0:25de

real 0m0.023s
user 0m0.000s
sys 0m0.011s
```
The Domain Name System

- What can an attacker do if they control a DNS server?
The Domain Name System

What can an attacker do if they control a DNS server?
- Control how users of that DNS server view the internet!
  - Assuming they use domain names
DNS Cache Poisoning

- **Attack:** ???

Diagram:
- User
- DNS server
- example.com
  - IP address: 192.0.0.16
- Malicious website
  - IP address: 192.0.0.17

User queries DNS server: "What's the IP for example.com?"
- Server returns cached IP: 192.0.0.17
- User uses this IP, accessing a malicious website.
**DNS Cache Poisoning**

- **Attack**: pre-empt DNS lookup by injecting malicious cache contents
  - Exploits DNS lookup optimization!

- Victim performs cache lookup, instead gets malicious domain IP
  - Attacker can redirect the victim’s browser to the malicious website

Diagram:
- User queries "What's the IP for example.com?"
- DNS server returns "192.0.0.17" (Cached)
- User gets IP 192.0.0.16
- Example.com IP address: 192.0.0.16
- Malicious website IP address: 192.0.0.17
- User is directed to malicious website
Thwarting DNS Hijacking

- Defenses: ???
Thwarting DNS Hijacking

- **DNS-level** authentication
  - DNSSec
  - Public-key crypto to “sign” DNS records

- **Endpoint** authentication
  - Certify that what I am seeing really is bank.com
  - Transport Layer Security (TLS)
Questions?
This time on CS 4440...

Network Availability
Denial of Service (DoS) Attacks
Transport, Link, Network, and Physical DoS
Basic Security Properties

- Confidentiality: ???
- Authenticity: ???
- Integrity: ???
- Access Control: ???
Basic Security Properties

- **Confidentiality**: Concealment of information or resources
  - Attacks: ???

- **Authenticity**: Identification and assurance of info origin
  - Attacks: ???

- **Integrity**: Preventing improper and unauthorized changes
  - Attacks: ???

- **Access Control**: Enforce who is allowed access to what
  - Attacks: ???
Basic Security Properties

- **Confidentiality:** Concealment of information or resources
  - Attacks: intercept credentials, info

- **Authenticity:** Identification and assurance of info origin
  - Attacks: SMTP header spoofing

- **Integrity:** Preventing improper and unauthorized changes
  - Attacks: tampering HTML over HTTP

- **Access Control:** Enforce who is allowed access to what
  - Attacks: web app code injection
Basic Security Properties

- **Confidentiality**: Concealment of information or resources
  - Attacks: intercept credentials, info

- **Authenticity**: Identification and assurance of info origin
  - Attacks: SMTP header spoofing

- **Integrity**: Preventing improper and unauthorized changes
  - Attacks: tampering HTML over HTTP

- **Access Control**: Enforce who is allowed access to what
  - Attacks: web app code injection

- **Availability**: Ability to use desired information or resource
  - Attacks: ???
Basic Security Properties

- **Confidentiality:** Concealment of information or resources
  - Attacks: intercept credentials, info

- **Authenticity:** Identification and assurance of info origin
  - Attacks: SMTP header spoofing

- **Integrity:** Preventing improper and unauthorized changes
  - Attacks: tampering HTML over HTTP

- **Access Control:** Enforce who is allowed access to what
  - Attacks: web app code injection

- **Availability:** Ability to use desired information or resource
  - Attacks: denial of service
Denial of Service Attacks
DoS: Denial of Service

- Goal: ???
DoS: Denial of Service

- **Goal:** make a service unusable, usually by overloading the server or network

- **How?**
DoS: Denial of Service

- **Goal:** make a service unusable, usually by overloading the server or network

- **How?**
  - Trigger the host to **crash**
    - Application-based DoS
    - Memory corruption
  - Consume host’s **resources**
    - TCP SYN floods
    - ICMP ECHO (ping) floods
  - Consume host’s **bandwidth**
    - UDP floods
    - ICMP floods

Amazon loses **$66,000 per minute** of downtime

Higher security makes DoS attacks **more likely**
Common DoS Attacks

- Locally-induced crash
  - Exploit host’s OS or server software bug

- Local resource consumption
  - fork() bomb, fill disk, deep directory nesting

- Deny service to individual hosts
  - Force crash or outage of critical services

- Remotely-induced crash
  - “Magic” packets—ping of death, teardrop

- Remote resource consumption
  - Syslog, SYN, fragment flood, UDP storm
Common DoS Attacks (cont.)

- Deny service to an entire network
  - Target vulnerable links, critical network infrastructure

- Remotely-induced network outage
  - Attacks against routers, DNS servers
  - Redirected routes—forged routing information

- Remote network congestion
  - Remote control of compromised hosts (“zombies”)
    - Allows for coordinated flooding
    - Distributed Denial of Service (DDoS)
Simple DoS Attacks

- Attacker spoofs their source address to hide origin

- Defenses:
  - ???
Simple DoS Attacks

- Attacker spoofs their source address to hide origin

**Defenses:**
- Block source IP address
- Firewall
- ISP-level blocking
- Ignore requests
Coordinated DoS Attacks

- Multiple willing attackers coordinate an attack on victim(s)
  - Same source-spoofing techniques as before
  - Harder to deal with
Distributed DoS Attacks (DDoS)
Distributed DoS Attacks (DDoS)
Distributed DoS Attacks (DDoS)
Distributed DoS Attacks (DDoS)

- Attacker
- Handler
- Handler
- Agent
- Agent
- Agent
- Agent
- Agent
- Victim

Rootkit-owned victim systems
Timeline of a DDoS Attack

- **Goal:** compromise a large number of machines to form a botnet
  1. Attacker identifies **exploitable hosts** with scanners or other techniques
  2. Attacker gains control over systems via exploits, password cracking, etc.
  3. Attacker installs **rootkit**
  4. Attacker **remotely instructs** compromised machines to **attack the target**
Timeline of a DDoS Attack

- **Goal**: compromise a large number of machines to form a botnet
  1. Attacker identifies **exploitable hosts** with scanners or other techniques
  2. Attacker **gains control** over systems via exploits, password cracking, etc.
  3. Attacker installs **rootkit**
  4. Attacker **remotely instructs** compromised machines to **attack the target**
Timeline of a DDoS Attack

- **Goal:** compromise a large number of machines to form a botnet
  1. Attacker identifies *exploitable hosts* with scanners or other techniques
  2. Attacker *gains control* over systems via exploits, password cracking, etc.
  3. Attacker installs *rootkit*
  4. Attacker *remotely instructs* compromised machines to **attack the target**

**Layers of Obfuscation**

- Victim never sees the true **attacker**
The Mirai botnet explained: How teen scammers and CCTV cameras almost brought down the internet

Mirai took advantage of insecure IoT devices in a simple but clever way. It scanned big blocks of the internet for open Telnet ports, then attempted to log in default passwords. In this way, it was able to amass a botnet army.

A tiny botnet launched the largest DDoS attack on record

A small but powerful army of just 5,000 devices generated a record-breaking web attack.

Storm: the largest botnet in the world?

Timely spam blasts help spread highly aggressive malware
DDoS or legitimate traffic?
DDoS or legitimate traffic?

u/TheRealAndyReid

“OMG... Joe’s in KC serves the BEST brisket sandwich”
DDoS or legitimate traffic?

u/TheRealAndyReid

“OMG... Joe’s in KC serves the BEST brisket sandwich”

https://www.joeskc.com/
DDoS or legitimate traffic?

u/TheRealAndyReid

“OMG... Joe’s in KC serves the BEST brisket sandwich”

“Click!”

“Order now!”

https://www.joeskc.com/
DDoS or legitimate traffic?

u/TheRealAndyReid

“OMG... Joe’s in KC serves the BEST brisket sandwich”

“Ooooh!”
“Click!”
“Order now!”
“Hungry!”

“Yum!”

“I’m from Cali and am clueless about BBQ!”

https://www.joeskc.com/

“I must try!”

“OMG… Joe’s in KC serves the BEST brisket sandwich”

https://www.joeskc.com/
DDoS or legitimate traffic?

u/TheRealAndyReid

“OMG... Joe’s in KC serves the BEST brisket sandwich”

“Ooooh!”

“Click!”

“Order now!”

“Hungry!”

“Yum!”

“I must try!”

“I’m from Cali and am clueless about BBQ!”

https://www.joeskc.com/

This webpage is not available

Reload
DDoS or legitimate traffic?

- How can we differentiate between Flash Mob traffic and DDoS traffic?
How can we differentiate between **Flash Mob traffic** and **DDoS traffic**?

**Flash Mob traffic**
- Many clients using service legitimately
- “Slashdot Effect”, “Reddit Hug of Death”
  - Traffic dies down when the network is flooded
- Sources in flash crowd are clustered
  - Usually by location (e.g., USA)

**What Does Slashdot Effect Mean?**
The slashdot effect refers to a temporary surge in traffic to a website, which can occur when a high-traffic website posts a link to smaller site or blog, thus directing an unprecedented surge in traffic. If the traffic increase is very large, it slow the site down or make it unreachable. The site is then considered to have been "slashdotted."

It's when someone posts a link to a website saying "Everyone, look at this website!" and everyone does. This puts so much traffic on the site in question's servers that they get overloaded and crash, causing the site to be inaccessible until the amount of traffic dies down a bit.
DDoS or legitimate traffic?

How can we differentiate between Flash Mob traffic and DDoS traffic?

- **Flash Mob traffic**
  - Many clients using service legitimately
  - “Slashdot Effect”, “Reddit Hug of Death”
    - Traffic dies down when the network is flooded
  - Sources in flash crowd are clustered
    - Usually by location (e.g., USA)

- **DDoS traffic**
  - Attack does not end when host crashes
  - Scattered locations (e.g., entire world)
Questions?
Transport Layer DoS
Recap: The Transport Layer

- What does it facilitate?
  - ???

- Key protocols?
  - Protocol 1: ???
    - Characteristics: ???
  - Protocol 2: ????
    - Characteristics: ???
Recap: The Transport Layer

- **What does it facilitate?**
  - Communication between apps on different hosts

- **Key protocols?**
  - **Protocol 1: TCP** (Transmission Control Protocol)
    - Characteristics: slow/complex but reliable
  - **Protocol 2: UDP** (User Datagram Protocol)
    - Characteristics: fast/simple but unreliable
The TCP Three-way Handshake

- **Recall:** TCP is a connection-oriented protocol
  - Initiate with three-way “handshake”: SYN, SYN-ACK, ACK
**Recall:** TCP is a \textit{connection-oriented} protocol
- Initiate with three-way “handshake”: SYN, SYN-ACK, ACK
The TCP Three-way Handshake

- **Recall**: TCP is a **connection-oriented** protocol
  - Initiate with three-way “handshake”: **SYN**, **SYN-ACK**, **ACK**

Client: **SYN**

**Server**: **SYN-ACK**

Client: **ACK**
Recall: TCP is a connection-oriented protocol

- Initiate with three-way “handshake”: SYN, SYN-ACK, ACK
- Server waits until client responds with ACK

Client: SYN

Server: SYN-ACK

Server: Hurry up!

Client: ACK
### SYN Flooding Attack

- **Attack:** spam SYN packets to server, with spoofed origin address

![Diagram of SYN Flooding Attack]

**Attacker:** SYN

**Attacker:** SYN

**Attacker:** SYN

**Attacker:** SYN
**SYN Flooding Attack**

- **Attack:** spam SYN packets to server, with spoofed origin address

  - **Attacker:** SYN
  - **Server:** SYN-ACK
  - **Attacker:** SYN
  - **Server:** SYN-ACK
  - **Attacker:** SYN
**SYN Flooding Attack**

- **Attack:** spam SYN packets to server, with spoofed origin address
  - Server’s resources completely reserved—now can’t serve legitimate clients
Thwarting SYN Flooding

- **Attack**: spam SYN packets to server, with spoofed origin address
  - Server’s resources completely reserved—now can’t serve legitimate clients

How can we **prevent** SYN flooding?
Thwarting SYN Flooding

- **Attack:** spam SYN packets to server, with spoofed origin address
  - Server's resources completely reserved—now can't serve legitimate clients

How can we **prevent** SYN flooding?

Incorporate **state**—use **SYN cookies**!
Network Layer DoS
Recap: The Network Layer

- **What does it facilitate?**
  - ???

- **Key functions?**
  - Function1: ???
  - Function1: ???

- **Addressing?**
  - ???
Recap: The Network Layer

- **What does it facilitate?**
  - Sending of data from host on one network to another

- **Key functions?**
  - **Function1: Routing:** (find the shortest path for a packet)
  - **Function1: Forwarding** (send packet on to the next hop)

- **Addressing?**
  - **IP addressing** (logical addressing)
ICMP: Internet Control Message Protocol

- **ICMP**: pings to determine whether a system is connected to the Internet
  - Analogous to “Hello, are you still there?”

**Diagram:**
- **Client**: ECHO REQUEST
- **Server**: ECHO REPLY
ICMP Smurf Attacks

- **Attack**: takes advantage of broadcast-enabled hosts to amplify attack
ICMP Smurf Attacks

- **Attack**: takes advantage of broadcast-enabled hosts to **amplify** attack

![Diagram showing ECHO REQUEST to an Amplifier, which then sends requests to multiple broadcast-enabled hosts.](image-url)
ICMP Smurf Attacks

- **Attack:** takes advantage of broadcast-enabled hosts to amplify attack
- Attacker spams **spoofed-source** ICMP requests, reflected to victim’s IP

![Diagram showing ICMP Smurf Attack]

- **Amplifier**
  - **ECHO REQUEST**
    - Spoofed source IP
  - **ECHO REPLY**
    - Reflected to Victim
Advanced DoS Strategies

- **Reflection:**
  - IP spoofing to redirect response to a victim

- **Amplification:**
  - Technique that increases the amount of traffic or packet size that the victim sees versus what the attacker originally sent

- **Common in real-world DDoS attacks**
  - Harder to detect (source obfuscation)
  - Harder to thwart (changing sources)
ICMP Ping of Death Attack

- **Internet Protocol**: IPv4 packets should be **less than 65,536 bytes**
  - Packets can be sent in **fragments** and **reassembled** by receiver

<table>
<thead>
<tr>
<th></th>
<th>IP Header</th>
<th>ICMP Header</th>
<th>ICMP Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length</td>
<td>20 bytes</td>
<td>8 bytes</td>
<td>65,508 bytes</td>
</tr>
</tbody>
</table>
ICMP Ping of Death Attack

- **Internet Protocol:** IPv4 packets should be **less than 65,536 bytes**
  - Packets can be sent in **fragments** and **reassembled** by receiver

- **Attack:** send packet in fragments that **reassemble to 64K+ bytes**
  - Many historical computer systems could not **handle larger packets**
ICMP Ping of Death Attack

- **Internet Protocol:** IPv4 packets should be less than **65,536 bytes**
  - Packets can be sent in **fragments** and **reassembled** by receiver

- **Attack:** send packet in fragments that reassemble to **64K+ bytes**
  - Many historical computer systems could not handle **larger packets**

- **Result:** crash by **buffer overflow**
  - Can’t serve clients until restart!
Thwarting ICMP-based DoS

- **Internet Protocol:** IPv4 packets should be less than 65,536 bytes
  - Packets can be sent in **fragments** and **reassembled** by receiver

- **Attack:** send packets in fragments that **reassemble** to 64K+ bytes
  - Many historical computer systems **could not handle larger packets**

- **Result:** crash by **buffer overflow**
  - Can’t serve clients until restart!

How can we **prevent** ICMP attacks?
Thwarting ICMP-based DoS

- **Internet Protocol**: IPv4 packets should be less than \(65,536\) bytes.
  - Packets can be sent in fragments and reassembled by the receiver.
- **Attack**: Send packets in fragments that reassemble to more than \(64K\) bytes.
  - Many historical computer systems could not handle larger packets.
- **Result**: Crash by buffer overflow.
  - Can’t serve clients until restart!

**How can we prevent ICMP attacks?**

- **Secure** any open amplifiers, and **sanitize** network input.
Link Layer DoS
Recap: The Data Link Layer

- What does it facilitate?
  - ???

- Addressing?
  - ???

- Authenticity?
  - ???
Recap: The Data Link Layer

■ What does it facilitate?
  ▪ Responsible for the node-to-node delivery of data

■ Addressing?
  ▪ MAC addresses
    ▪ Physical identifier for hardware

■ Authenticity?
  ▪ No—MAC addresses can be changed!
ARP: Address Resolution Protocol

- **ARP**: query to *resolve the MAC address* given a desired host IP
  - How we know which *physical* address to transmit data to from its logical address

**Diagram:**

- **Client:** REQUEST MAC for IP 192.168.1.1
- **Server:** SENDING MAC 00:B0:D0:63:C2:26
**ARP Flooding Attack**

- **Attack**: same idea as **SYN flood**; **spoof source to victim** and spam away!
  - Victim gets overwhelmed by ARP replies and bandwidth crashes
Thwarting ARP Flooding

- **Attack**: same idea as **SYN flood**; spoof source to victim and spam away!
  - Victim gets overwhelmed by ARP replies and bandwidth crashes

How can we **prevent** ARP flooding?
Thwarting ARP Flooding

- **Attack**: same idea as SYN flood; *spoof source to victim* and spam away!
  - Victim gets overwhelmed by ARP replies and bandwidth crashes

**How can we prevent ARP flooding?**

**Limit rate**—allow *only so many* reqs!
Physical Layer DoS
Recap: Physical Layer

- What is it?
Recap: Physical Layer

- What is it?
Physical Layer DoS

Russian Spy Submarines Are Tampering with Undersea Cables That Make the Internet Work. Should We Be Worried?

A massive cable attack is probably an over-hyped scenario, at least for a country with as many redundant cables as the United States pitted against a limited number of Russian special-operations submarines.

CNN Exclusive: FBI investigation determined Chinese-made Huawei equipment could disrupt US nuclear arsenal communications
Iran blocks capital's internet access as Amini protests grow

Social media platforms have also been cut off in areas of Tehran and Kurdistan as videos of dissent go viral
Thwarting Physical Layer Attacks
Questions?
Analyzing Network Packets
Recap: Internet Packet Encapsulation

- How packets are generated and sent

What you care about

Application Message

Segment Header  Segment Data

Packet Header  Packet Data

Frame Header  Frame Data  Frame Footer

App Layer

Transport Layer

Network Layer

Link Layer
Recap: Internet Packet Encapsulation

- How packets are generated and sent

What really gets sent

Application Message
Segment Header
Segment Data
Packet Header
Packet Data
Frame Header
Frame Data
Frame Footer
App Layer
Transport Layer
Network Layer
Link Layer
Recap: Internet Packet Encapsulation

- How packets are generated and sent

How can we **detect and thwart** different **attacks** on network layers?
Tools of the Trade

- **Packet Analyzers:**
  - Tools for dissecting network packets

- **Packet Analyzers allow you to:**
  - Identify unusual packets
  - Characterize network activity
  - Pinpoint **malicious traffic**

- The basis of modern-day network security (e.g., firewalls, antivirus)
Familiarity with packet analysis tools?

1. I eat NetSec CTF challenges like a kid eats candy on Halloween. 0%
2. Some (e.g., Wireshark, DPKT, Scapy, or something else) 0%
3. None (but that's totally okay!) 0%
Tools of the Trade: Wireshark

- A “graphical interface” for manual packet analysis
  - Completely open-source and free

General workflow:
- Load up a PCAP (packet capture)
- Wireshark will display each packet
- Inspect particular fields of interest
### Tools of the Trade: Wireshark

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.000000</td>
<td>10.0.0.2</td>
<td>10.128.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
<tr>
<td>2</td>
<td>0.000001</td>
<td>10.128.0.2</td>
<td>10.0.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
<tr>
<td>3-7</td>
<td>0.000020</td>
<td>10.128.0.2</td>
<td>10.0.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
<tr>
<td>8</td>
<td>0.000022</td>
<td>10.128.0.2</td>
<td>10.0.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
<tr>
<td>9</td>
<td>0.000049</td>
<td>10.128.0.2</td>
<td>10.0.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
<tr>
<td>10</td>
<td>0.000054</td>
<td>10.128.0.2</td>
<td>10.0.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
<tr>
<td>11</td>
<td>0.000072</td>
<td>10.128.0.2</td>
<td>10.0.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
<tr>
<td>12</td>
<td>0.000088</td>
<td>10.128.0.2</td>
<td>10.0.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
<tr>
<td>13</td>
<td>0.000123</td>
<td>10.128.0.2</td>
<td>10.0.0.2</td>
<td>TCP</td>
<td>54 3341 – 80 [SYN] Seq=0 Win=512 Len=8</td>
</tr>
</tbody>
</table>

*Frame 2: 58 bytes on wire (484 bits), 58 bytes captured (484 bits)*

- Ethernet II, Src: 42:01:0a:f0:00:01 (42:01:0a:f0:00:01), Dst: 42:01:0a:f0:00:17 (42:01:0a:f0:00:17)

- Internet Protocol Version 4, Src: 10.128.0.2, Dst: 10.0.0.2

- Transmission Control Protocol, Src Port: 80, Dst Port: 3222, Seq: 0, Ack: 1, Len: 0

Source Port: 80

Destination Port: 3222

- [Stream index: 1]
- [TCP Segment Len: 0]
- Sequence number: 0 (relative sequence number)
- Next sequence number: 0 (relative sequence number)
- Acknowledgment number: 1 (relative ack number)
- Window size value: 29260
- [Calculated window size: 29260]
- Checksum: 0x0280 [unverified]
- Checksum Status: Unverified
- Urgent pointer: 0
- Options: [4 bytes], Maximum segment size
  - [Timestamps]
Tools of the Trade: Scapy

- Python API for programmatic packet capture and analysis
  - Think of it as “Wireshark in API form”
  - **Project 4:** you will use Scapy to write your own packet analysis scripts
Tools of the Trade: Scapy

- Python API for programmatic packet capture and analysis
  - Think of it as “Wireshark in API form”
  - **Project 4:** you will use Scapy to write your own packet analysis scripts

- We’ll provide the PCAP traces...
  - You’ll write code to analyze them!
  - **Examples:**
    - Detecting attacks on a network
    - Finding user credentials
    - Sniffing a user’s **browsing history**
Questions?
Next time on CS 4440...

Passwords and Secure Authentication