5. Network Security Basics
ENEE 757 | CMSC 818V

Prof. Tudor Dumitraș
Assistant Professor, ECE
University of Maryland, College Park

http://ter.ps/757
https://www.facebook.com/SDSAtUMD

Today’s Lecture

• Where we’ve been
  – Crypto basics
  – OS security basics

• Where we’re going today
  – Network security
  – TCP/IP, BGP
  – Intrusion detection

• Where we’re going next
  – Trustworthy computing
Internet Is a Network of Networks

- TCP/IP for packet routing and connections
- Border Gateway Protocol (BGP) for route discovery
- Domain Name System (DNS) for IP address discovery

Autonomous system (AS) is a collection of IP networks under control of a single administrator (e.g., ISP)

OSI Protocol Stack

- Application
- Presentation
- Session
- Transport
- Network
- Data Link
- Physical

email, Web, NFS

RPC

TCP

IP

Ethernet
Data Formats

- Application data
  - TCP header
  - Data
  - TCP header
  - Data
  - TCP header
  - Data

IP (Internet Protocol)

- Connectionless
  - Unreliable, “best-effort” protocol
- Uses numeric addresses for routing
- Typically several hops in the route
**TCP (Transmission Control Protocol)**

- **Sender:** break data into packets
  - Sequence number is attached to every packet
- **Receiver:** reassemble packets in correct order
  - Acknowledge receipt; lost packets are re-sent
- **Connection state maintained on both sides**

---

**Threat #1: Eavesdropping on Network Connections**

- **Goal:** extract information from network packets
- Many applications send data unencrypted
  - ftp, telnet send passwords in the clear
- Network interface card (NIC) in “promiscuous mode” reads all passing data
  - Attacker sniffs packets to eavesdrop passively

**Solution:** encryption (e.g., IPsec, HTTPS), improved routing
Threat #2: Denial of Service (DoS)

- **Goal:** take out a large site with little computing work
- DoS can happen at any layer
  - Link
  - TCP/UDP
  - Application

- DoS solutions for one layer cannot always be replicated at other layers
  - This means that DoS cannot be solved with end-to-end solutions
  - Need cooperation from the network

---

### IP and TCP Headers

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>4</td>
</tr>
<tr>
<td>Header Length</td>
<td>1</td>
</tr>
<tr>
<td>Type of Service</td>
<td>8</td>
</tr>
<tr>
<td>Total Length</td>
<td>16</td>
</tr>
<tr>
<td>Identification</td>
<td>12</td>
</tr>
<tr>
<td>Flags</td>
<td>1</td>
</tr>
<tr>
<td>Fragment Offset</td>
<td>1</td>
</tr>
<tr>
<td>Time to Live</td>
<td>16</td>
</tr>
<tr>
<td>Protocol</td>
<td>8</td>
</tr>
<tr>
<td>Header Checksum</td>
<td>16</td>
</tr>
<tr>
<td>Source Address of Originating Host</td>
<td>16</td>
</tr>
<tr>
<td>Destination Address of Target Host</td>
<td>16</td>
</tr>
<tr>
<td>Options</td>
<td>0-60</td>
</tr>
<tr>
<td>Padding</td>
<td>0-60</td>
</tr>
<tr>
<td>IP Data</td>
<td>Variable</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source Port</td>
<td>16</td>
</tr>
<tr>
<td>Dest port</td>
<td>16</td>
</tr>
<tr>
<td>SEQ Number</td>
<td>32</td>
</tr>
<tr>
<td>ACK Number (if ACK flag is set)</td>
<td>32</td>
</tr>
<tr>
<td>URG</td>
<td>1</td>
</tr>
<tr>
<td>ACK</td>
<td>1</td>
</tr>
<tr>
<td>PSH</td>
<td>1</td>
</tr>
<tr>
<td>SYN</td>
<td>1</td>
</tr>
<tr>
<td>FIN</td>
<td>1</td>
</tr>
<tr>
<td>Other headers</td>
<td>Variable</td>
</tr>
<tr>
<td>Data</td>
<td>Variable</td>
</tr>
</tbody>
</table>
**TCP Handshake**

C

\[ \text{SYN, seq\_no = x} \]

\[ \text{Listening…} \]

\[ \text{Wait} \]

S

\[ \text{SYN, seq\_no = y, ACK, ack\_no = x+1} \]

\[ \text{Connected} \]

\[ \text{seq\_no = x+1, ACK, ack\_no = y+1} \]

Delayed segment from previous connection with seq\_no=x+2

Recommendation: Increment the initial sequence number every 4 ms (why?)

**TCP Flow Control**

- TCP uses a sliding window mechanism
- Receiver advertises a window of size W
- Sender can send up W unacknowledged bytes
  - Can be split among multiple segments, if data is not yet available
- Receiver can delay sending ACKs until it has data to transmit
  - ACKs will be piggybacked on the data packets
  - ACK will correspond to the next byte it expects to receive => this may acknowledge multiple packets received previously

[11]

[12]
DoS Attack #1: TCP SYN Flood

MS Blaster (August 16, 2003): every infected machine sent 50 packets per second to port 80 on windowsupdate.com

SYN Flooding Explained

- Attacker sends many connection requests with spoofed source addresses
- Victim allocates resources for each request
  - New thread, connection state maintained until timeout
  - Fixed bound on half-open connections (backlog)
- Once resources exhausted, requests from legitimate clients are denied
- This is a classic denial of service pattern
  - It costs nothing to TCP initiator to send a connection request, but TCP responder must spawn a thread for each request - asymmetry!
Preventing Denial of Service

• DoS is caused by asymmetric state allocation
  – If responder opens new state for each connection attempt, attacker can initiate thousands of connections from bogus or forged IP addresses

• Cookies ensure that the responder is stateless until initiator produced at least two messages
  – Responder’s state (IP addresses and ports of the connection) is stored in a cookie and sent to initiator
  – After initiator responds, cookie is regenerated and compared with the cookie returned by the initiator

SYN Flooding Defense: SYN Cookies
[Bernstein and Schenk]

More info: http://cr.yp.to/syncookies.html
Anti-Spoofing Cookies: Basic Pattern

- Client sends request (message #1) to server
- Typical protocol:
  - Server sets up connection, responds with message #2
  - Client may complete session or not - potential DoS!
- Cookie version:
  - Server responds with hashed connection data in message #2
  - Client confirms by returning hashed data
    - If source IP address is bogus, attacker can’t confirm
  - Need an extra step to send postponed message #2, except in TCP (can piggyback on SYN-ACK in TCP)

Domain Name Service (DNS)

DNS maps symbolic names to numeric IP addresses
(for example, www.umd.edu ↔ 54.83.56.209)
DoS Attack #2: DNS Amplification Attack

- DNS runs over UDP (rather than TCP) => can spoof source IP
- **Open DNS resolvers**: answer queries from any host
  - 2006: 0.58M open resolvers on Internet (Kaminsky-Shiffman)
  - 2013: 21.7M open resolvers (openresolverproject.org)
- March 2013: **300 Gbps DDoS** attack on Spamhaus
- There are other protocols that amplify traffic (more on this later)

Other DNS Vulnerabilities

- DNS servers can be DDoS'ed
  - Oct ‘02: ICMP flood took out 9 root servers for 1 hour
- Kaminski attack: poison DNS caches
  - Attacker guesses transaction ID used to match queries with replies
  - Solution: randomize ports and transaction IDs
- DNS implementations have vulnerabilities
  - Reverse query buffer overrun in old releases of BIND
  - MS DNS for NT 4.0 crashes on chargen stream
- Can use “zone transfer” requests to download DNS database and map out the network
  - Solution: block port 53 on corporate name servers

See [http://cr.yp.to/djbdns/notes.html](http://cr.yp.to/djbdns/notes.html)
Threat #3: Impersonate Other Hosts

• Goal 1: Defeat authentication that relies on IP-source address
  – Must spoof the source address

• Goal 2: Draw packets destined to other hosts
  – Allows conducting man-in-the-middle attacks (more on this later)
  – Must target the destination address

TCP Connection Spoofing

• Each TCP connection has associated state
  – Sequence number, port number

• TCP state is easy to guess
  – Port numbers standard, seq numbers predictable

• Can inject packets into existing connections
  – If attacker knows initial sequence number and amount of traffic, can guess likely current number
  – How do you guess a 32-bit sequence number?
“Blind” IP Spoofing Attack

- Can’t receive packets sent to Bob, but can bypass Alice’s IP address-based authentication
  - rlogin and other remote access tools, SPF defense against spam

Intrusion Detection Systems (IDS)

- Hard to prevent all network attacks; can we detect them?
  - Host-based / Network-based intrusion detection system (HIDS/NIDS)
- Misuse detection
  - Use attack “signatures” (need a model of the attack)
    - Sequences of system calls, patterns of network traffic, etc.
    - Must know in advance what attacker will do
    - Can only detect known attacks
- Anomaly detection
  - Using a model of normal system behavior, try to detect deviations and abnormalities
    - E.g., raise an alarm when a statistically rare event(s) occurs
    - Can potentially detect unknown attacks
Intrusion Detection Errors

- **False negatives**: attack is not detected
  - Big problem in signature-based misuse detection
- **False positives**: harmless behavior is classified as an attack
  - Big problem in statistical anomaly detection
- All intrusion detection systems (IDS) suffer from errors of both types
- Which is a bigger problem?
  - Attacks are fairly rare events
  - Thus IDS often suffer from the base-rate fallacy

Conditional Probability

- Suppose two events A and B occur with probability $\Pr(A)$ and $\Pr(B)$, respectively
- Let $\Pr(AB)$ be probability that both A and B occur
- What is the conditional probability that A occurs assuming B has occurred?

$$\Pr(A \mid B) = \frac{\Pr(AB)}{\Pr(B)}$$
Bayes’s Theorem

- Suppose mutually exclusive events $E_1, \ldots, E_n$ together cover the entire set of possibilities.
- Then the probability of any event $A$ occurring is
  \[ \Pr(A) = \sum_{1 \leq i \leq n} \Pr(A \mid E_i) \cdot \Pr(E_i) \]
  - Intuition: since $E_1, \ldots, E_n$ cover the entire probability space, whenever $A$ occurs, some event $E_i$ must have occurred.
- Can rewrite this formula as
  \[ \Pr(E_i \mid A) = \frac{\Pr(A \mid E_i) \cdot \Pr(E_i)}{\Pr(A)} \]

Base-Rate Fallacy

- 1% of traffic is SYN floods; IDS accuracy is 90%
  - IDS classifies a SYN flood as attack with prob. 90%, classifies a valid connection as attack with prob. 10%
- What is the probability that a connection flagged by IDS as a SYN flood is actually valid?

\[
\Pr(\text{valid} \mid \text{alarm}) = \frac{\Pr(\text{alarm} \mid \text{valid}) \cdot \Pr(\text{valid})}{\Pr(\text{alarm})} \\
= \frac{\Pr(\text{alarm} \mid \text{valid}) \cdot \Pr(\text{valid})}{\Pr(\text{alarm} \mid \text{valid}) \cdot \Pr(\text{valid}) + \Pr(\text{alarm} \mid \text{SYN flood}) \cdot \Pr(\text{SYN flood})} \\
= \frac{0.10 \cdot 0.99}{0.10 \cdot 0.99 + 0.90 \cdot 0.01} = 92\% \text{ chance raised alarm is false!!!} \]
Review of Lecture

• What did we learn?
  – IP spoofing
  – TCP handshake and flow control
  – TCP cookies
  – Various eavesdropping and denial-of-service attacks
  – Base rate fallacy

• Sources
  – Vitaly Shmatikov

• What’s next?
  – Trustworthy computing