ARP, IP, TCP, UDP

CS 166: Introduction to Computer Systems Security
IP and MAC Addresses

• Devices on a local area network have
  – IP addresses (network layer)
  – MAC addresses (data link layer)
• IP addresses are used for high level protocols
• MAC addresses are used for low level protocols
• How to translate IP Addresses into MAC addresses?
Address Resolution Protocol (ARP)

- Connects the network layer to the data link layer
- Maps IP addresses to MAC addresses
- Based on broadcast messages and local caching
- Does not support confidentiality, integrity, or authentication
- Defined as a part of RFC 826
ARP Messages

- ARP broadcasts requests of type
  **who has** <IP addressC>
  **tell** <IP addressA>
- Machine with <IP addressC> responds
  <IP addressC> is at <MAC address>
- Requesting machine caches response
- Network administrator configures IP address and subnet on each machine

4/7/18 ARP, IP, TCP, UDP
ARP Cache

• The Linux, Windows and OSX command `arp -a` displays the ARP table

<table>
<thead>
<tr>
<th>Internet Address</th>
<th>Physical Address</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.148.31.1</td>
<td>00-00-0c-07-ac-00</td>
<td>dynamic</td>
</tr>
<tr>
<td>128.148.31.15</td>
<td>00-0c-76-b2-d7-1d</td>
<td>dynamic</td>
</tr>
<tr>
<td>128.148.31.71</td>
<td>00-0c-76-b2-d0-d2</td>
<td>dynamic</td>
</tr>
<tr>
<td>128.148.31.75</td>
<td>00-0c-76-b2-d7-1d</td>
<td>dynamic</td>
</tr>
<tr>
<td>128.148.31.102</td>
<td>00-22-0c-a3-e4-00</td>
<td>dynamic</td>
</tr>
</tbody>
</table>

• Command `arp –a –d` flushes the ARP cache

• ARP cache entries are stored for a configurable amount of time
ARP Spoofing

• The ARP table is updated whenever an ARP response is received
• Requests are not tracked
• ARP announcements are not authenticated
• Machines trust each other
• A rogue machine can spoof other machines
ARP Normal Operation

- Normal operation
  - Alice communicates with Bob

**ARP Cache**

<table>
<thead>
<tr>
<th>IP</th>
<th>MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.101</td>
<td>00:11:22:33:44:01</td>
</tr>
<tr>
<td>192.168.1.102</td>
<td>00:11:22:33:44:02</td>
</tr>
</tbody>
</table>

**Alice**

IP: 192.168.1.102
MAC: 00:11:22:33:44:02

**Bob**

IP: 192.168.1.101
MAC: 00:11:22:33:44:01

Data

192.168.1.101 is at 00:11:22:33:44:01
192.168.1.102 is at 00:11:22:33:44:02
ARP Poisoning Attack

• Man-in-the-middle attack
  – ARP cache poisoning leads to eavesdropping

192.168.1.102 is at 00:11:22:33:44:03
192.168.1.101 is at 00:11:22:33:44:03

Poisoned ARP Cache

<table>
<thead>
<tr>
<th>IP</th>
<th>MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.102</td>
<td>00:11:22:33:44:03</td>
</tr>
</tbody>
</table>

IP: 192.168.1.103
MAC: 00:11:22:33:44:03

IP: …102
MAC: …02

Poisoned ARP Cache

<table>
<thead>
<tr>
<th>IP</th>
<th>MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.101</td>
<td>00:11:22:33:44:03</td>
</tr>
</tbody>
</table>

Data

Bob

Alice

Eve
ARP Poisoning & ARP Spoofing

• Almost all ARP implementations are stateless
• An ARP cache updates every time that it receives an ARP reply
  — ... even if it did not send any ARP request!
• Can “poison” ARP cache with gratuitous ARP replies
• Using static entries solves the problem but it is almost impossible to manage!
From the LAN to the Internet

- LAN
  - 192.168.0.5
    - GTW 192.168.0.1
    - SN 255.255.255.0

- 192.168.0.6
  - GTW 192.168.0.1
  - SN 255.255.255.0

- Internet
  - 138.16.160.252
Brown's IP Space

• Brown separates the network connecting dorms and the network connecting offices and academic buildings

• Dorms
  – Class B network 138.16.0.0/16 (64K addresses)

• Academic buildings and offices
  – Class B network 128.148.0.0/16 (64K addresses)

• CS department
  – Several class C (/24) networks, each with 254 addresses
    – Tstaff supported machines: 128.148.31.0/24, 128.148.33.0/24, 128.148.38.0/24
  – Unsupported machines: 128.148.36.0/24
User Datagram Protocol

• UDP is a stateless, unreliable datagram protocol built on top of IP, that is it lies at the transport layer.
• UDP does not provide delivery guarantees or acknowledgments, which makes it efficient.
• Can however distinguish data for multiple concurrent applications on a single host.
• A lack of reliability implies applications using UDP must be ready to accept a fair amount of corrupted and lost data.
  – Most applications built on UDP will suffer if they require reliability.
  – VoIP, streaming video, and streaming audio all use UDP.
Transmission Control Protocol

- Transport layer protocol for reliable data transfer, in-order delivery of messages and ability to distinguish multiple applications on same host
  - HTTP and SSH are built on top of TCP
- TCP packages a data stream it into segments transported by IP
  - Order maintained by marking each packet with sequence number
  - Every time TCP receives a packet, it sends out an ACK to indicate successful receipt of the packet
- TCP generally checks data transmitted by comparing a checksum of the data with a checksum encoded in the packet
Ports

• TCP supports concurrent applications on the same server
• Ports are 16 bit numbers identifying where data is directed
• The TCP header includes both a source and a destination port
• Ports 0 through 1023 are reserved for use by known protocols
  • E.g., HTTPS uses 443 and SSH uses 22
• Ports 1024 through 49151 are known as user ports, and are used for listening to connections
## TCP Packet Format

<table>
<thead>
<tr>
<th>Bit Offset</th>
<th>0-3</th>
<th>4-7</th>
<th>8-15</th>
<th>16-18</th>
<th>19-31</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Source Port</strong></td>
<td></td>
<td><strong>Destination Port</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Sequence Number</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Acknowledgment Number</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>Offset</td>
<td>Reserved</td>
<td>Flags</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Window Size</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>160</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Options</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&gt;= 160</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Payload</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Establishing TCP Connections

- TCP connections are established through a three-way handshake.
- The server generally is a passive listener, waiting for a connection request.
- The client requests a connection by sending out a SYN packet.
- The server responds by sending a SYN/ACK packet, acknowledging the connection.
- The client responds by sending an ACK to the server, thus establishing connection.

**.syn**
Seq = x

**Syn-Ack**
Seq = y, Ack = x + 1

**Ack**
Seq = x + 1, Ack = y + 1
TCP Data Transfer

- During connection initialization using the three way handshake, initial sequence numbers are exchanged.
- The TCP header includes a 16 bit checksum of the data and parts of the header, including the source and destination.
- Acknowledgment or lack thereof is used by TCP to keep track of network congestion and control flow and such.
- TCP connections are cleanly terminated with a 4-way handshake:
  - The client which wishes to terminate the connection sends a FIN message to the other client.
  - The other client responds by sending an ACK.
  - The other client sends a FIN.
  - The original client now sends an ACK, and the connection is terminated.
TCP Data Transfer and Teardown

Data seq=x

Ack seq=x+1

Data seq=y

Ack seq=y+1

Client

Server

Fin seq=x

Ack seq=x+1

Fin seq=y

Ack seq=y+1

Client

Server
What We Have Learned

• ARP protocol
• ARP poisoning attack
  – MitM attack on a LAN
• Transport layer protocols
  – TCP for reliable transmission
  – UDP when packet loss/corruption is tolerated
• Lack of built-in security in network protocols
  – Security can be incorporated into application layer (SSL)