Computer Networks Security: ARP Attacks and Session Hijacking

CS 1951e - Computer Systems Security: Principles and Practice
IP Addresses and Packets

- **IP addresses**
  - IPv4: 32-bit addresses
  - IPv6: 128-bit addresses
- **Address subdivided into**
  - network, subnet, and host
    - E.g., 128.148.32.110
- **Broadcast addresses**
  - E.g., 128.148.32.255
- **Private networks**
  - not routed outside of a LAN
  - 10.0.0.0/8
  - 172.16.0.0/12
  - 192.168.0.0/16

- **IP header includes**
  - Source address
  - Destination address
  - Packet length (up to 64KB)
  - Time to live (up to 255)
  - IP protocol version
  - Fragmentation information
  - Transport layer protocol information (e.g., TCP)

<table>
<thead>
<tr>
<th>v</th>
<th>fragmentation info</th>
<th>length</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTL</td>
<td>prot.</td>
<td>source</td>
</tr>
</tbody>
</table>
IP Vulnerabilities

- Unencrypted transmission
  - Eavesdropping possible at any intermediate host during routing

- No source authentication
  - Sender can spoof source address, making it difficult to trace packet back to attacker

- No integrity checking
  - Entire packet, header and payload, can be modified while en route to destination, enabling content forgeries, redirections, and man-in-the-middle attacks

- No bandwidth constraints
  - Large number of packets can be injected into network to launch a denial-of-service attack
  - Broadcast addresses provide additional leverage

Subnetting

- Subnetting allows breaking of a larger single network into small subnetworks

- Has benefits at present too, since it reduces network overhead by reducing the number of nodes receiving broadcasts, and reduces the size of routing tables

- A subnet mask is a bit mask which tells how many bits in an IP address identify subnets, and how many provide for host addressing

- Subnet masks are represented as IP addresses, that is as 4 dotted decimals representing octets in the mask.

- Subnetting is the process of allocating bits from the host portion to the network portion of the address.
**IP Routing**

- A router bridges two or more networks
  - Operates at the network layer
  - Maintains tables to forward packets to the appropriate network
  - Forwarding decisions based solely on the destination address

- Routing table
  - Maps ranges of addresses to LANs or other gateway routers

---

**Routing Examples**

[Diagram showing a network with routers and gateways connecting LAN A and LAN B with clients, servers, and routers.]
ARP

- The **address resolution protocol** (ARP) connects the network layer to the data layer by converting IP addresses to MAC addresses.
- ARP works by **broadcasting** requests and caching responses for future use.
- The protocol begins with a computer broadcasting a message of the form `who has <IP address1> tell <IP address2>`
- When the machine with `<IP address1>` or an ARP server receives this message, it broadcasts the response `<IP address1> is <MAC address>`
- The requestor’s IP address `<IP address2>` is contained in the link header.
- The Linux and Windows 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>
<tr>
<td>128.148.31.137</td>
<td>00-1d-92-b6-f1-a9</td>
<td>dynamic</td>
</tr>
</tbody>
</table>

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.
Example

- Normal operation
  – Alice communicates with Bob

<table>
<thead>
<tr>
<th>IP</th>
<th>MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.1.1</td>
<td>00:11:22:33:44:01</td>
</tr>
<tr>
<td>192.168.1.105</td>
<td>00:11:22:33:44:02</td>
</tr>
</tbody>
</table>

Data

Spoofing Using ARP Poisoning

DEMO 1
Example (cont.)

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

ARP Poisoning &ARP Spoofing

• According to the standard, 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!
• It is possible to “poison” an arp cache by sending gratuitous arp replies
• Using static entries solves the problem but it is almost impossible to manage!
Telnet Protocol (RFC 854)

• Telnet is a protocol that provides a general, bi-directional, not encrypted communication
• **telnet** is a generic TCP client
  – Allows a computer to connect to another one
  – Provides remote login capabilities to computers on the Internet
  – Sends whatever you type
  – Prints whatever comes back
  – Useful for testing TCP servers (ASCII based protocols)

Ettercap

• Ettercap is a suite for man in the middle attacks on LAN
• In this demo we use:
  – Unified sniffing (promiscuous mode)
  – MiTM attack (arp poisoning)
  – Protocol dissection active and passive (telnet password retrieval) no working in ettercap NG 0.8, you can use Wireshark
ARP Spoofing

• Rewriting the ARP tables and providing bad ARP responses allows attacking computers to receive information not intended for them.

---

Denial of Service Using ARP

DEMO 2
DEMO 2: network DOS using ARP

Ping 192.168.1.101

192.168.1.101

192.168.1.102

Cable Loop

How can it be solved?

Spanning Tree Protocol
(ISO 802.1D)

A Meshed Network

Four spanning trees of the Meshed Network

• Suppose you have a Meshed Network with bidirectional links that make loops/cycles...
• ...then a spanning tree of the Meshed Network is the same network and no loops/cycles
Transmission Control Protocol

- TCP is a transport layer protocol guaranteeing reliable data transfer, in-order delivery of messages and the ability to distinguish data for multiple concurrent applications on the same host.
- Most popular application protocols, including WWW, FTP and SSH are built on top of TCP.
- TCP takes a stream of 8-bit byte data, packages it into appropriately sized segment and calls on IP to transmit these packets.
- Delivery order is maintained by marking each packet with a 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 multiple concurrent applications on the same server.
- Accomplishes this by having ports, 16 bit numbers identifying where data is directed.
- The TCP header includes space for both a source and a destination port, thus allowing TCP to route all data.
- In most cases, both TCP and UDP use the same port numbers for the same applications.
- Ports 0 through 1023 are reserved for use by known protocols.
- Ports 1024 through 49151 are known as user ports, and should be used by most user programs for listening to connections and the like.
- Ports 49152 through 65535 are private ports used for dynamic allocation by socket libraries.
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>Source Port</td>
<td>Destination Port</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>Sequence Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>Acknowledgment Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>Offset</td>
<td>Reserved</td>
<td>Flags</td>
<td>Window Size</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>Checksum</td>
<td>Urgent Pointer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>Options</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&gt;= 160</td>
<td>Payload</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 has 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, indicating an acknowledgment for 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

Fin seq=x
Ack seq=x+1

Data seq=y
Ack seq=y+1

Fin seq=y
Ack seq=y+1

Client
Server

Client
Server
Session Hijacking

• Also commonly known as TCP Session Hijacking
• A security attack over a protected network
• Attempt to take control of a network session
• Sessions are server keeping state of a client’s connection
• Servers need to keep track of messages sent between client and the server and their respective actions
• Most networks follow the TCP/IP protocol
• IP Spoofing is one type of hijacking on large networks

IP Spoofing

• IP Spoofing is an attempt by an intruder to send packets from one IP address that appear to originate at another
• If the server thinks it is receiving messages from the real source after authenticating a session, it could inadvertently behave maliciously
• There are two basic forms of IP Spoofing
  • Blind Spoofing
    – Attack from any source
  • Non-Blind Spoofing
    – Attack from the same subnet
Blind IP Spoofing

- The TCP/IP protocol requires that “acknowledgement” numbers be sent across sessions
- Makes sure that the client is getting the server’s packets and vice versa
- Need to have the right sequence of acknowledgment numbers to spoof an IP identity
  - Attacker must guess correctly or know the sequence a priori

Non-Blind IP Spoofing

- IP Spoofing without inherently knowing the acknowledgment sequence pattern
  - Done on the same subnet
  - Use a packet sniffer to analyze the sequence pattern
    - Packet sniffers intercept network packets
    - Eventually decodes and analyzes the packets sent across the network
    - Determine the acknowledgment sequence pattern from the packets
    - Send messages to server with actual client's IP address and with validly sequenced acknowledgment number
User Datagram Protocol

- UDP is a stateless, unreliable datagram protocol built on top of IP, that is it lies on level 4
- It does not provide delivery guarantees, or acknowledgments, but is significantly faster (i.e. dns uses on port 53)
- 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 error packages and data loss.
  - Most applications used on UDP will suffer if they have reliability. VoIP, Streaming Video and Streaming Audio all use UDP.
IPv6

- IPv6 or IPng were adopted by IETF in 1994 and is supposed to be the next generation of IP
- Supports 128-bit addressing, however still supports IPv4 and most application can run with no changes
- Supports Stateless address autoconfiguration (SLAAC)
- of hosts connected to IPv6 networks, can be disabled, and stateful configuration in the form of DHCPv6 can be used
- Not all networks support dual-stack, tunneling is used for IPv4 networks to talk to IPv6 networks (and vice-versa)
- Does not have broadcast messages but provides support for multicast, (multicasts can be extended to support broadcast)
- Allows for very large packet sizes (jumbograms)
- The package header structure is simplified to facilitate routing
- Utilizes IPsec to provide for network level authentication and security
- Is currently not yet used… (IPv6 traffic share was reported to be approaching 1%)

Network Address Translation

- Introduced in the early 90s to alleviate IPv4 address space congestion
- Relies on translating addresses in an internal network, to an external address that is used for communication to and from the outside world
- NAT is usually implemented by placing a router in between the internal private network and the public network.
- Saves IP address space since not every terminal needs a globally unique IP address, only an organizationally unique one
- While NAT should really be transparent to all high level services, this is sadly not true because a lot of high level communication uses things on IP
Translation

- Router has a pool of private addresses 192.168.10.0/24

<table>
<thead>
<tr>
<th>Source IP Address</th>
<th>Destination IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.10.237</td>
<td>128.148.36.11</td>
</tr>
<tr>
<td>128.148.36.11</td>
<td>192.168.10.237</td>
</tr>
</tbody>
</table>

IP Packet Modifications

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vers</td>
<td>Version</td>
</tr>
<tr>
<td>len</td>
<td>Length</td>
</tr>
<tr>
<td>type of service</td>
<td>Type of service</td>
</tr>
<tr>
<td>total length</td>
<td>Total length</td>
</tr>
<tr>
<td>ident</td>
<td>Identification</td>
</tr>
<tr>
<td>flags</td>
<td>Flags</td>
</tr>
<tr>
<td>fragment offset</td>
<td>Fragment Offset</td>
</tr>
<tr>
<td>header checksum</td>
<td>Header Checksum</td>
</tr>
<tr>
<td>source IP address</td>
<td>Source IP Address</td>
</tr>
<tr>
<td>destination IP address</td>
<td>Destination IP Address</td>
</tr>
<tr>
<td>options</td>
<td>Options</td>
</tr>
<tr>
<td>padding</td>
<td>Padding</td>
</tr>
<tr>
<td>data</td>
<td>Data</td>
</tr>
</tbody>
</table>

- Computed
- Modified on output
- Modified on input
- ????