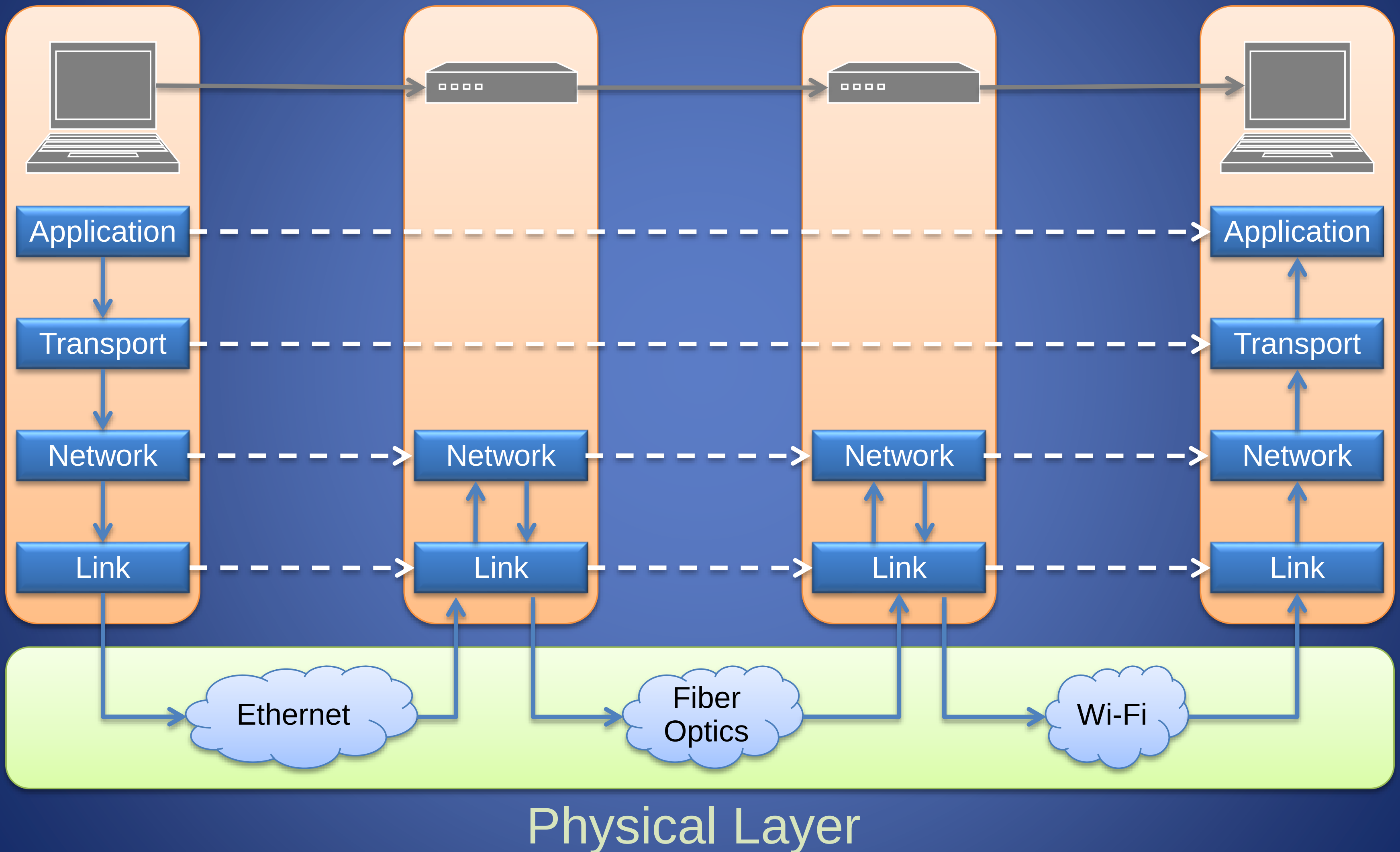
Internet Layers

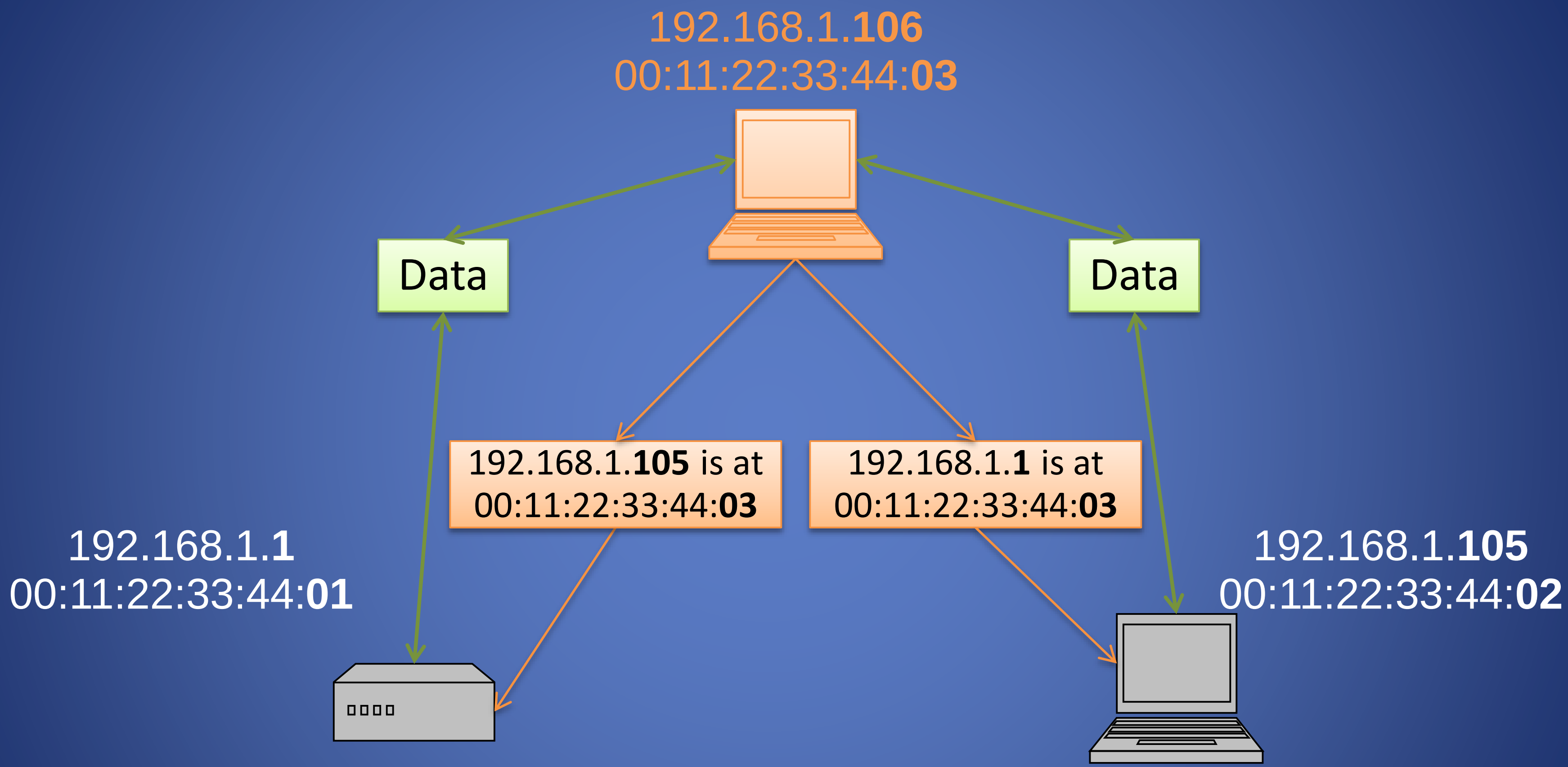


ARP requests and responses

ARP Cache	
192.168.1.**105**	00:11:22:33:44:**02**

ARP Cache	
192.168.1.**1**	00:11:22:33:44:**01**

Poisoned ARP Caches



Poisoned ARP Cache	
192.168.1.**105**	00:11:22:33:44:**03**

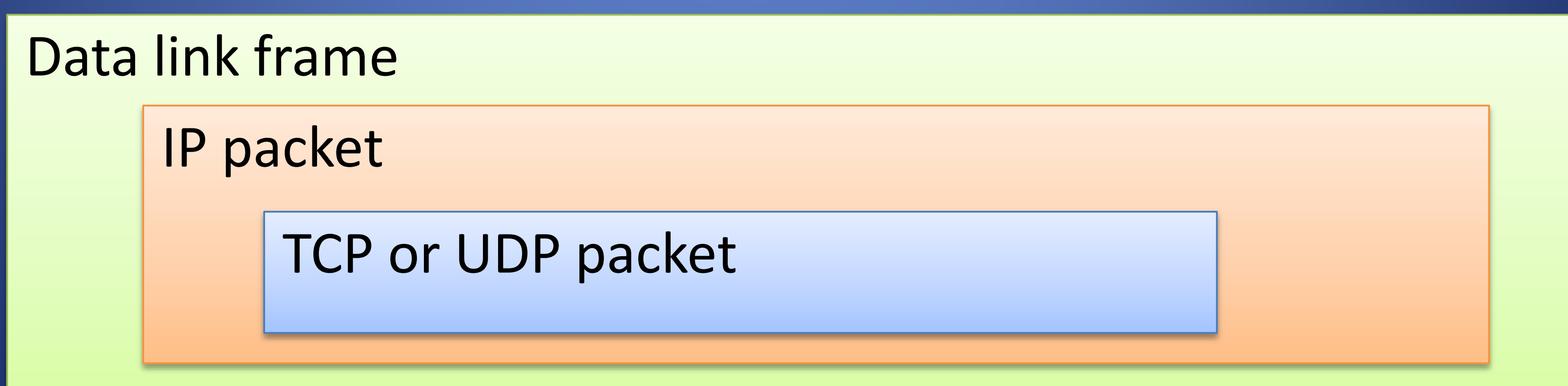
Poisoned ARP Cache	
192.168.1.**1**	00:11:22:33:44:**03**

How to prevent ARP poisoning

IP

Internet Protocol

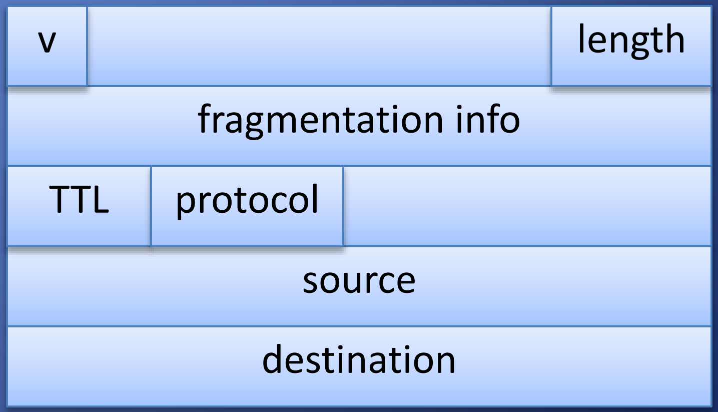
- Connectionless
  - Each packet is transported independently from other packets
- Unreliable
  - Delivery on a best effort basis
  - No acknowledgments
  - Packets may be lost, reordered, corrupted, or duplicated
- IP packets
  - Encapsulate TCP and UDP packets
  - Encapsulated into link-layer frames



IP Addresses and Packets

- IP addresses
  - IPv4: 32-bit addresses
  - IPv6: 128-bit addresses
- E.g., 128.148.32.110

- 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)



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

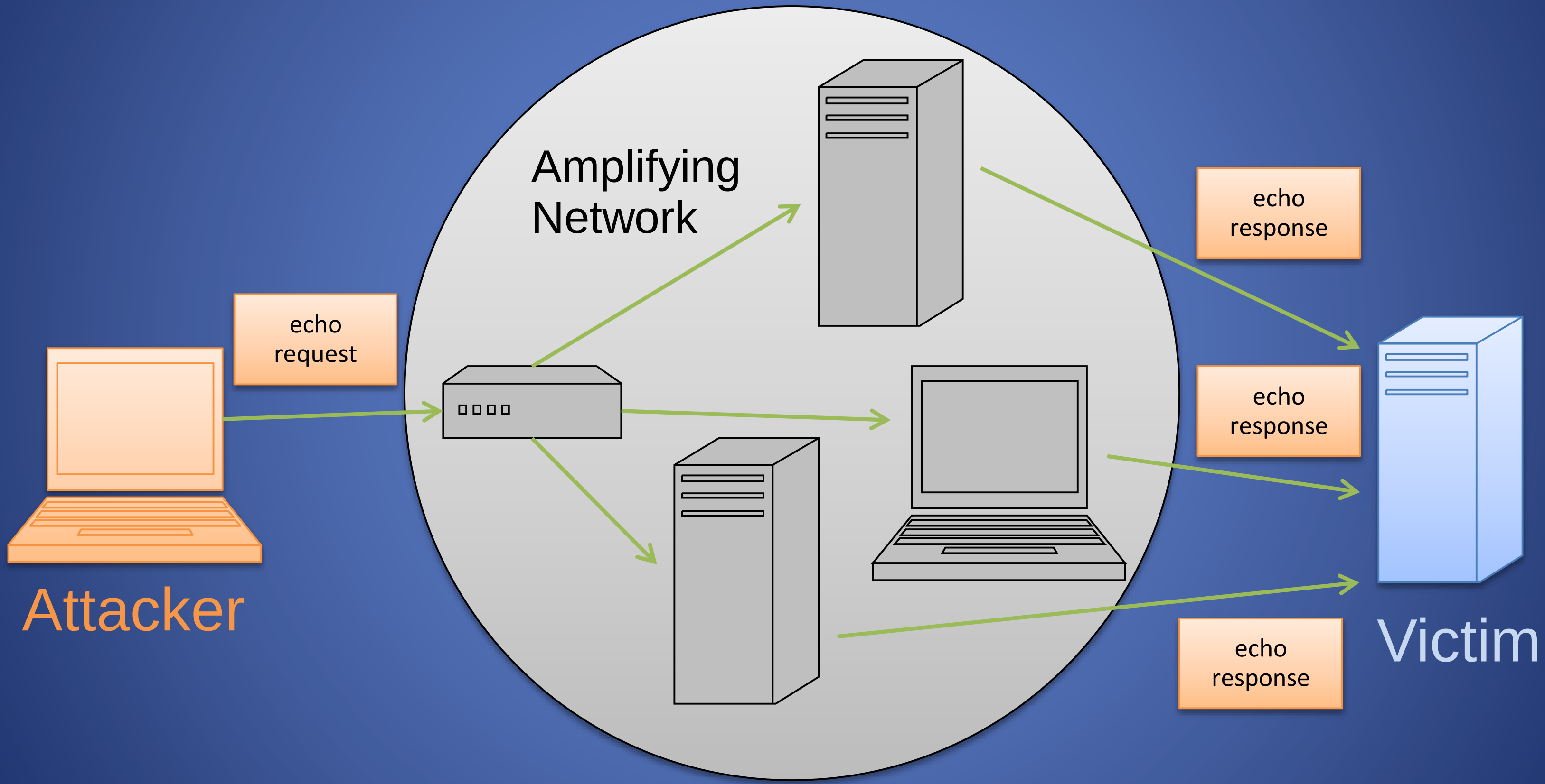
Internet Routes

- Internet Control Message Protocol (ICMP)
  - Used for network testing and debugging
  - Considered a network layer protocol
- Tools based on ICMP
  - Ping: sends series of echo request messages and provides statistics on roundtrip times and packet loss
  - Traceroute: sends series ICMP packets with increasing TTL value to discover routes

ICMP Attacks

- Ping of death
  - ICMP specifies messages must fit a single IP packet (64KB)
  - Send a ping packet that exceeds maximum size using IP fragmentation
  - Reassembled packet caused several operating systems to crash due to a buffer overflow
- Smurf
  - Ping a broadcast address using a spoofed source address

Smurf Attack



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

TCP

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

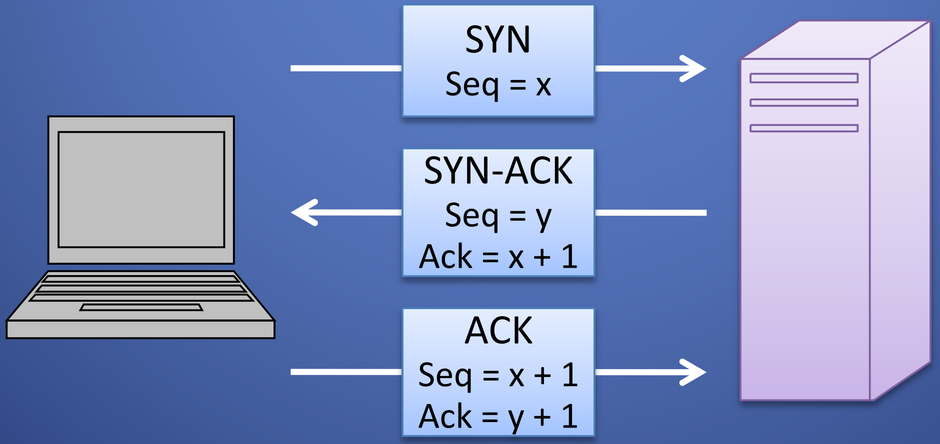
- TCP supports multiple concurrent applications on the same server
- Accomplishes this by having ports, 16 bit numbers identifying where data is directed
- The TCP header includes space for both a source and a destination port, thus allowing TCP to route all data
- In most cases, both TCP and UDP use 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

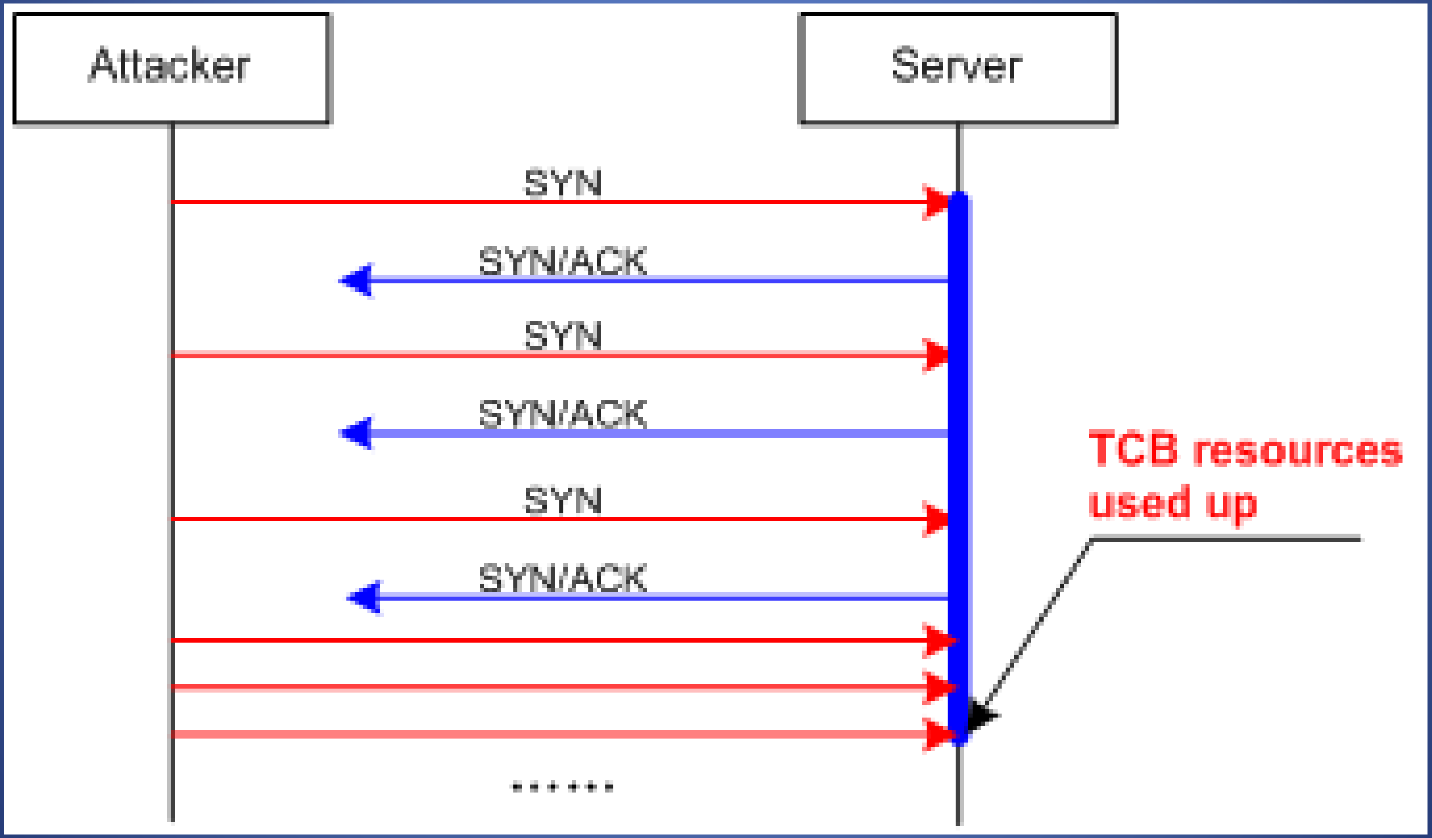
Bit Offset	0-3	4-7	8-15	16-18	19-31
0	**Source Port**			**Destination Port**	
32	**Sequence Number**				
64	**Acknowledgment Number**				
96	Offset	Reserved	Flags	**Window Size**	
128	Checksum			Urgent Pointer	
160	Options				
>= 160	Payload				

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 Flood



Attacker
Server
SYN
SYN/ACK
SYN
SYN/ACK
SYN
SYN/ACK
TCB resources used up
......

SYN Cookies

- A SYN flood leaves half-open connections
  - The "SYN queue" is a data structure which keeps track of these half-open connections
  - We track the source IP and port of client, server IP and port, seq# of client, seq# of server
  - Idea: we don't really need to keep all of this
    - We just need enough to recognize the ACK of the client
    - Can we get away without storing *anything* locally?

SYN Cookies: The Idea

- Store nothing locally
  - ISN: Initial sequence number
  - Encode all we need to remember in the ISN we send back to the client
  - t: a 32-bit counter which increments every 64 seconds
  - K: a secret key selected by server for uptime of server
  - Limitations: MSS limited to 8 values

Server ISN

t mod 32	***MSS***	***hash(client IP and port || server IP and port || t || K)***
5	*3*	*24*

SYN Cookies: Details

- MSS: Maximum Segment Size
  - Suggested by client, server then computes best value
    - Details depend on whether they are on the same network, MTU on network, etc
    - Server can have only 8 values to encode here
- What happens when client replies with ACK?
  - Client will reply with ISN+1 of server in the ACK
  - Server then subtracts 1 and checks against hash of client IP and port, server IP and port, t which matches in the lowest 5 bits, and K
    - If match, put in SYN queue
    - If not, ignore

SYN Cookies: Limitations

- Note that this will NOT prevent bandwidth-saturation attacks
  - This technique seeks only to prevent SYN queue overflows

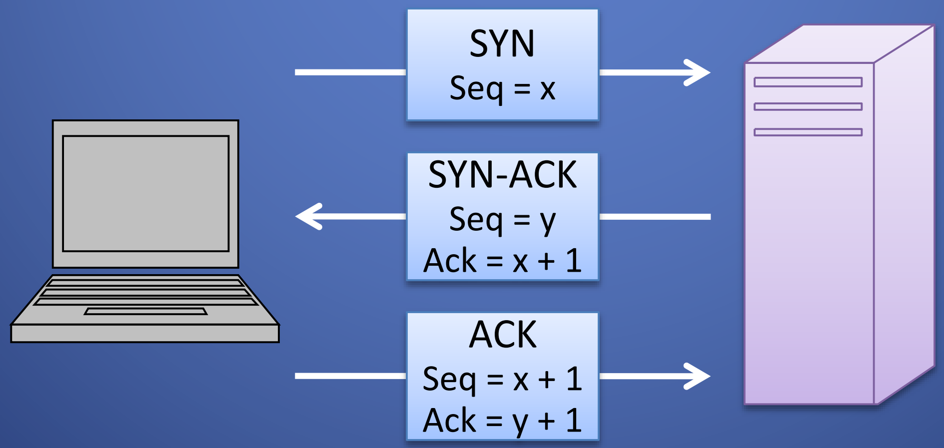
SYN Cookies: Implementation

- Standard in Linux and FreeBSD

echo 1 > proc/sys/net/ipv4/tcp_syncookies

Session Hijacking

- Also commonly known as TCP Session Hijacking
- A security attack over a protected network
- Attempt to take control of a network session
- Guess sequence numbers x and y and take over
- Make sure the victim does not send SYN-ACK by launching DoS



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