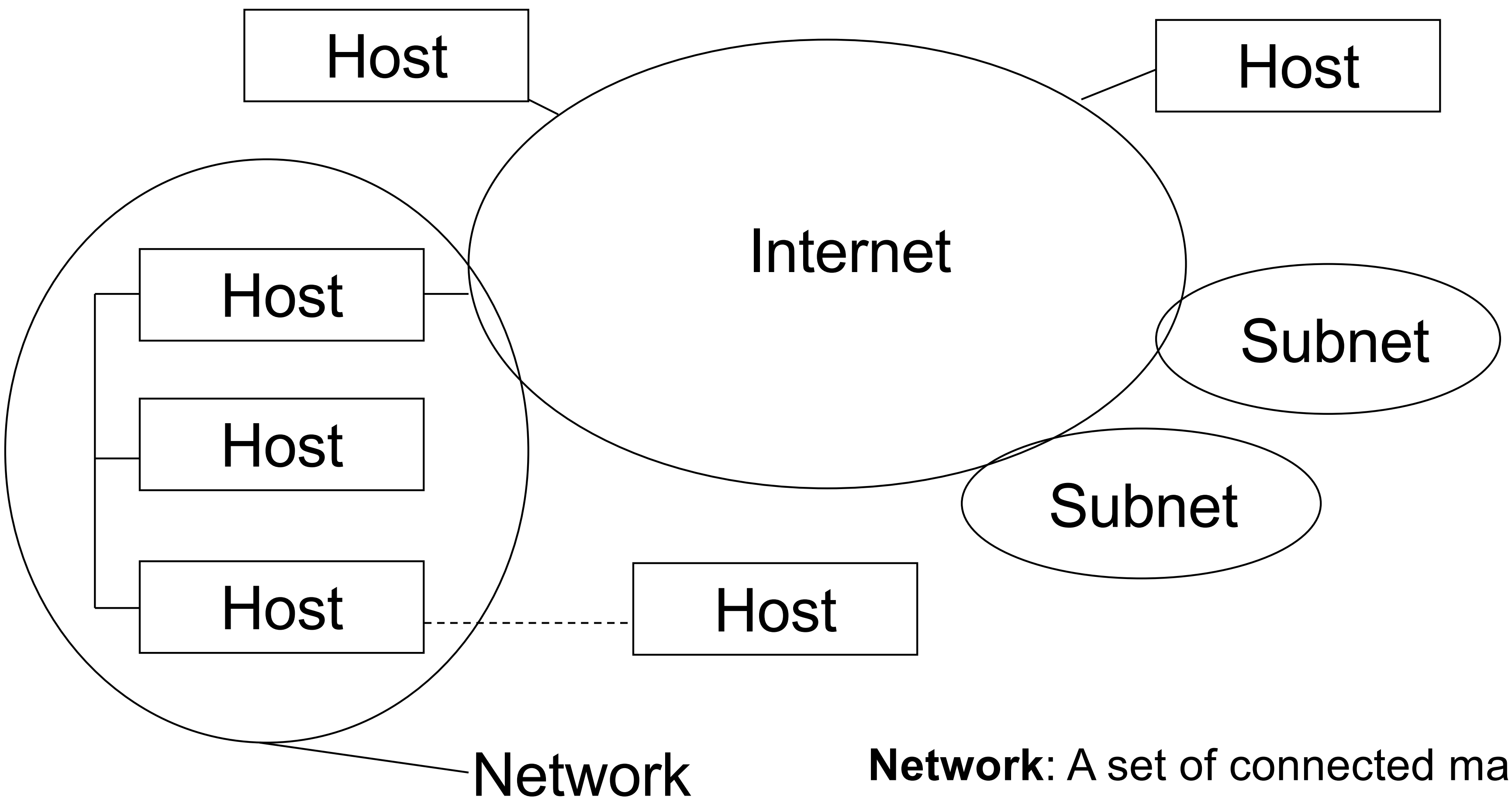
CS 177 - Computer Security

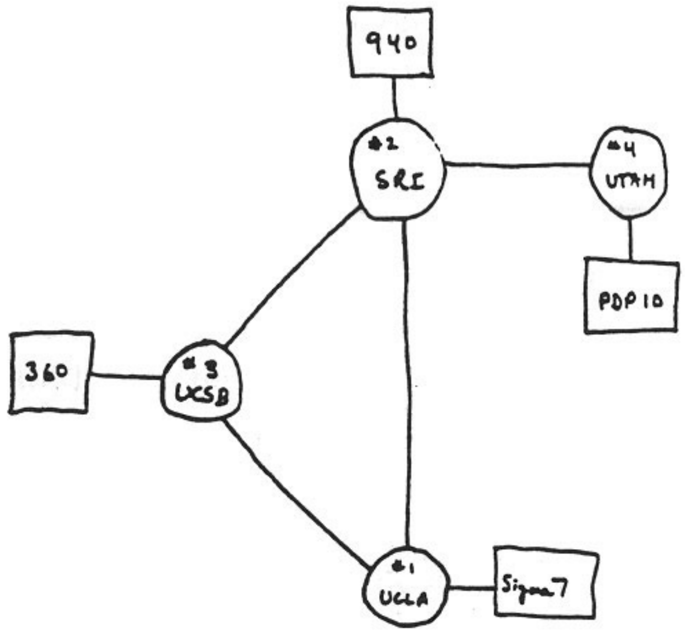
Network Security

The Internet



Network: A set of connected machines that can communicate with each other
Internet: A global network of computers

Early Internet (ARPA Network)



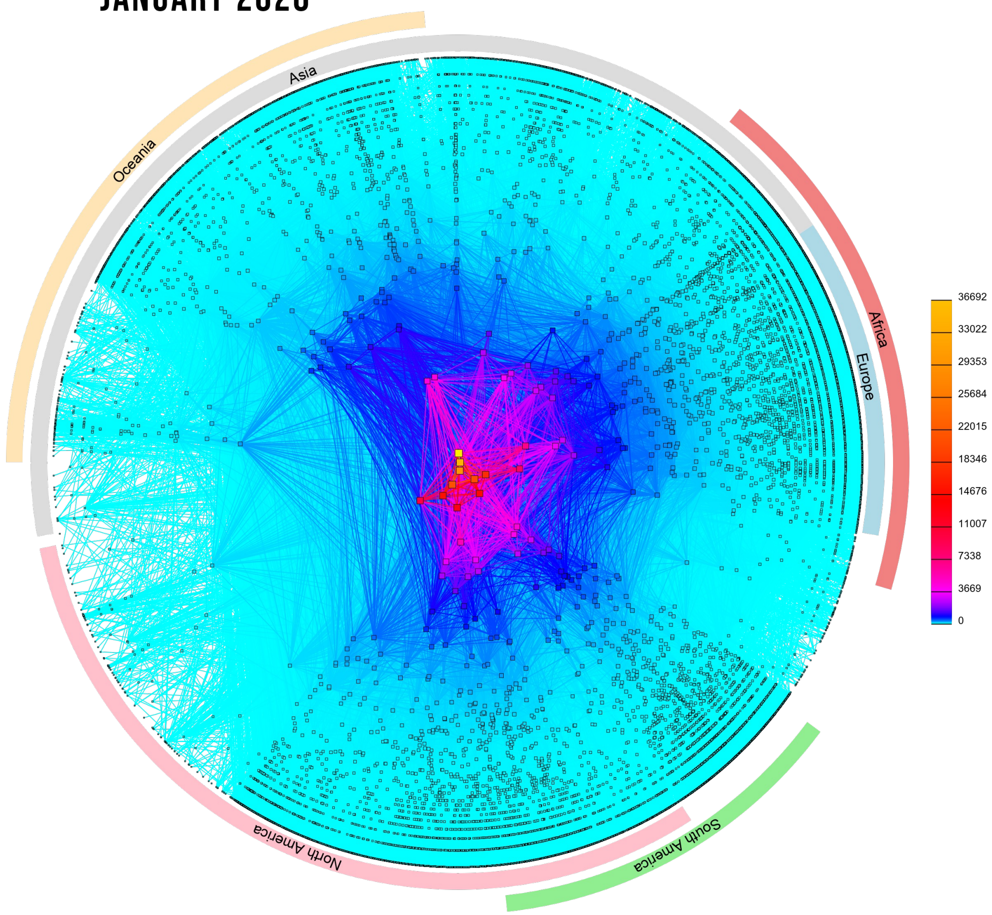
THE ARPA NETWORK

DEC 1969

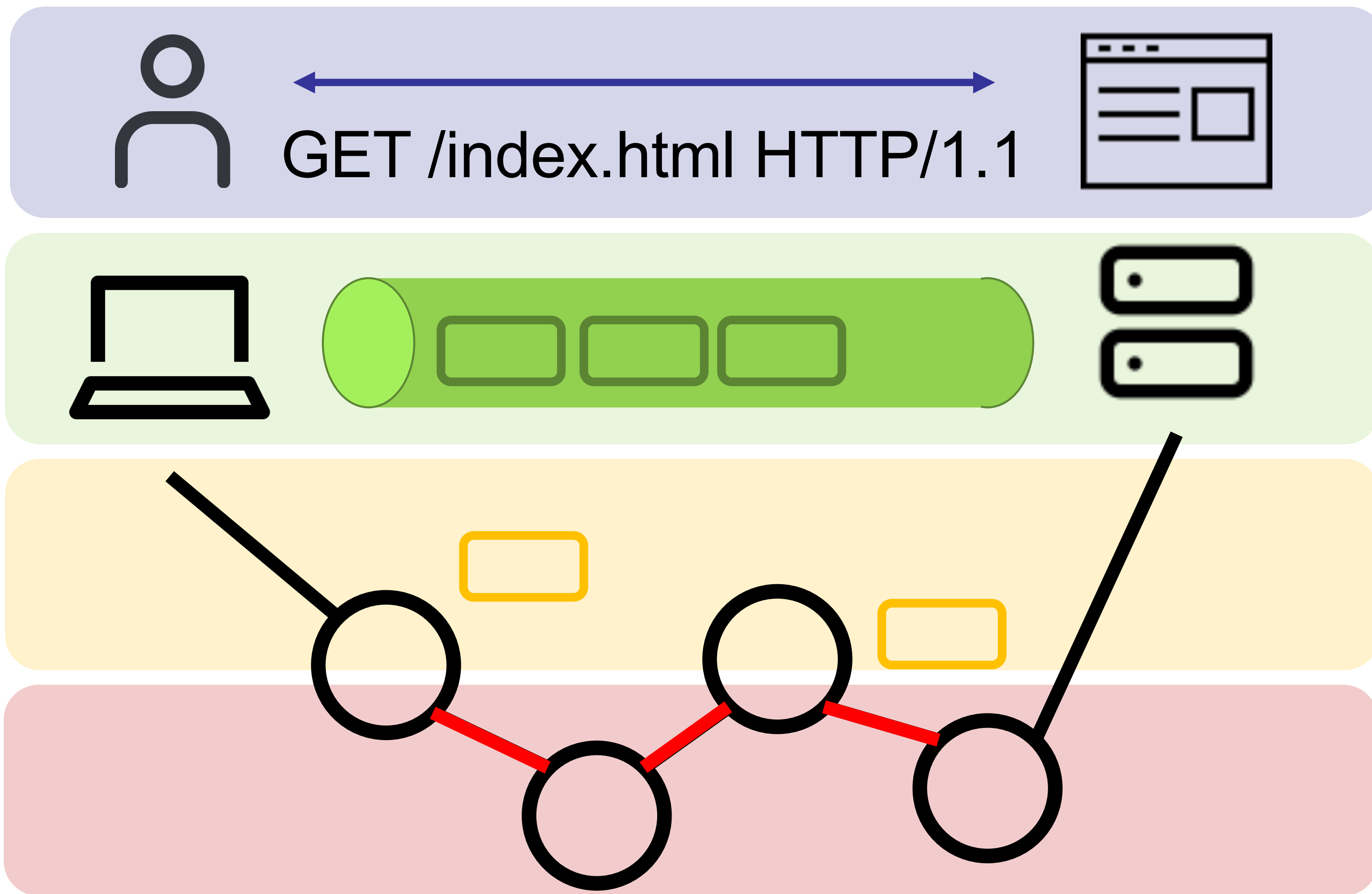
4 NODES

Internet Today

CAIDA'S IPV4 AS CORE GRAPH
JANUARY 2020



Example: Access Web Page



Layers and Protocols

- Layer
  - uses a protocol
  - relies on services provided by the layer below it
  - provides services to the layer above it

- A network protocol is an established set of rules that determine how data is transmitted between two parties
- Specifies syntax and semantics
  - syntax: How communication is specified and structured (format, order of messages)
  - semantics: What a communication means (actions taken when sending/receiving messages)

OSI Model

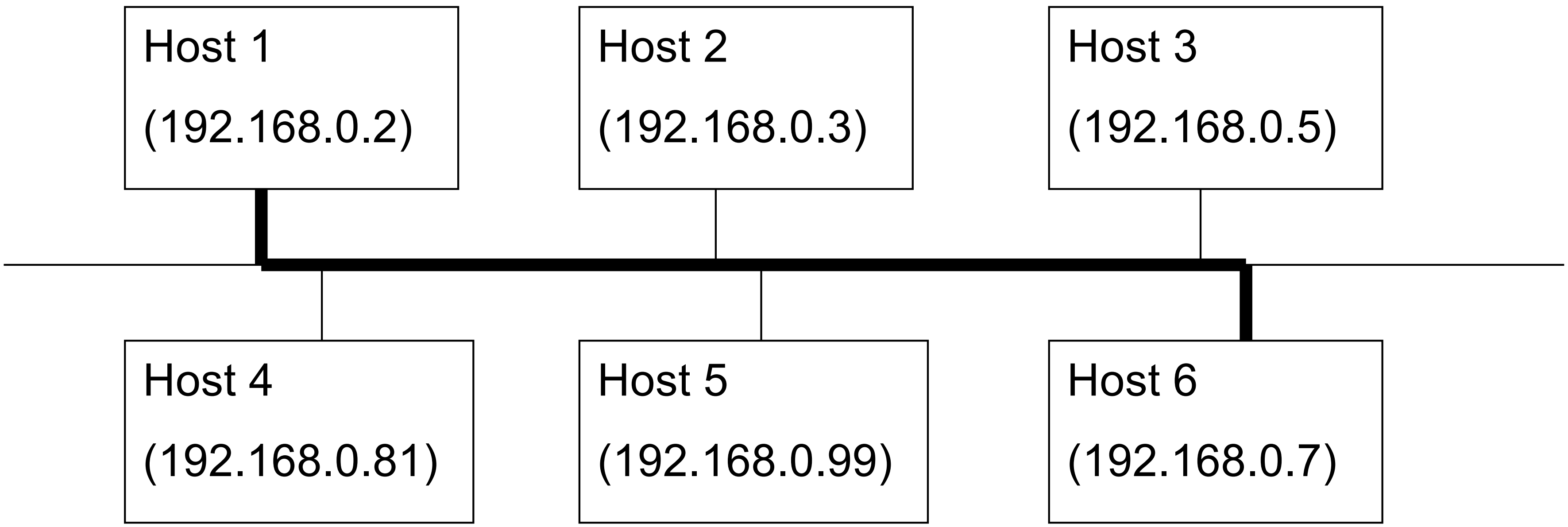
- Open Systems Interconnection (OSI) Model
  - introduced in the late 1970s

Application Layer (HTTP)
Presentation Layer
Session Layer
Transport Layer (TCP, UDP)
Network Layer (IP)
Data Link Layer (Ethernet, Wireless)
Physical Layer

DATA LINK LAYER

Direct Packet Delivery

- If two hosts are in the same physical network, the IP packet is encapsulated and delivered directly



Ethernet

dest (48 bits)	src (48 bits)	type (16)	data	CRC (32)

0x0800	IP Datagram

Ethernet

- Widely used data link layer protocol (for wired networks)
- Carrier Sense, Multiple Access, Collision Detection
- Addresses: 48 bits (e.g., 00:38:af:23:34:0f)
  – mostly hardwired by the manufacturer
- Type (2 bytes): specifies encapsulated protocol
  – IP, ARP, RARP, ..
- Data
  – min. 46 bytes payload (padding may be needed), max 1500 bytes
- CRC (4 bytes)

Direct Packet Delivery

Problem

- Ethernet uses 48-bit addresses
- IP uses 32-bit addresses
- (typically) we want to send an IP packet
- but we only can use the link layer to do this

ARP

ARP (Address Resolution Protocol)

- Service at the link-level, RFC 826
- maps network-addresses to link-level addresses
- Host A wants to know the hardware address associated with IP address of host B
- Host A broadcasts ARP message on physical link
  - including its own mapping
- Host B answers Host A with ARP answer message
- Mappings are <u>cached</u>: *arp -a* shows mapping

ARP Message

dest (6 byte)	src (6 byte)	type (2)	data	CRC (4)

0x0800	IP Datagram

0x0806	**ARP**	**PAD**

- 28 bytes - 18 bytes -

ARP Message

<table>
<tr><td colspan="2">hardware type (2 byte)</td><td>protocol type (2 byte)</td></tr>
<tr><td>hw.adr.size
(1 byte)</td><td>prot. adr. size
(1 byte)</td><td>opcode (2 byte)</td></tr>
<tr><td colspan="3">sender Ethernet address (6 byte)</td></tr>
<tr><td colspan="3">sender IP address (4 byte)</td></tr>
<tr><td colspan="3">target Ethernet address (6 byte)</td></tr>
<tr><td colspan="3">target IP address (4 byte)</td></tr>
</table>

Sending an IP Packet

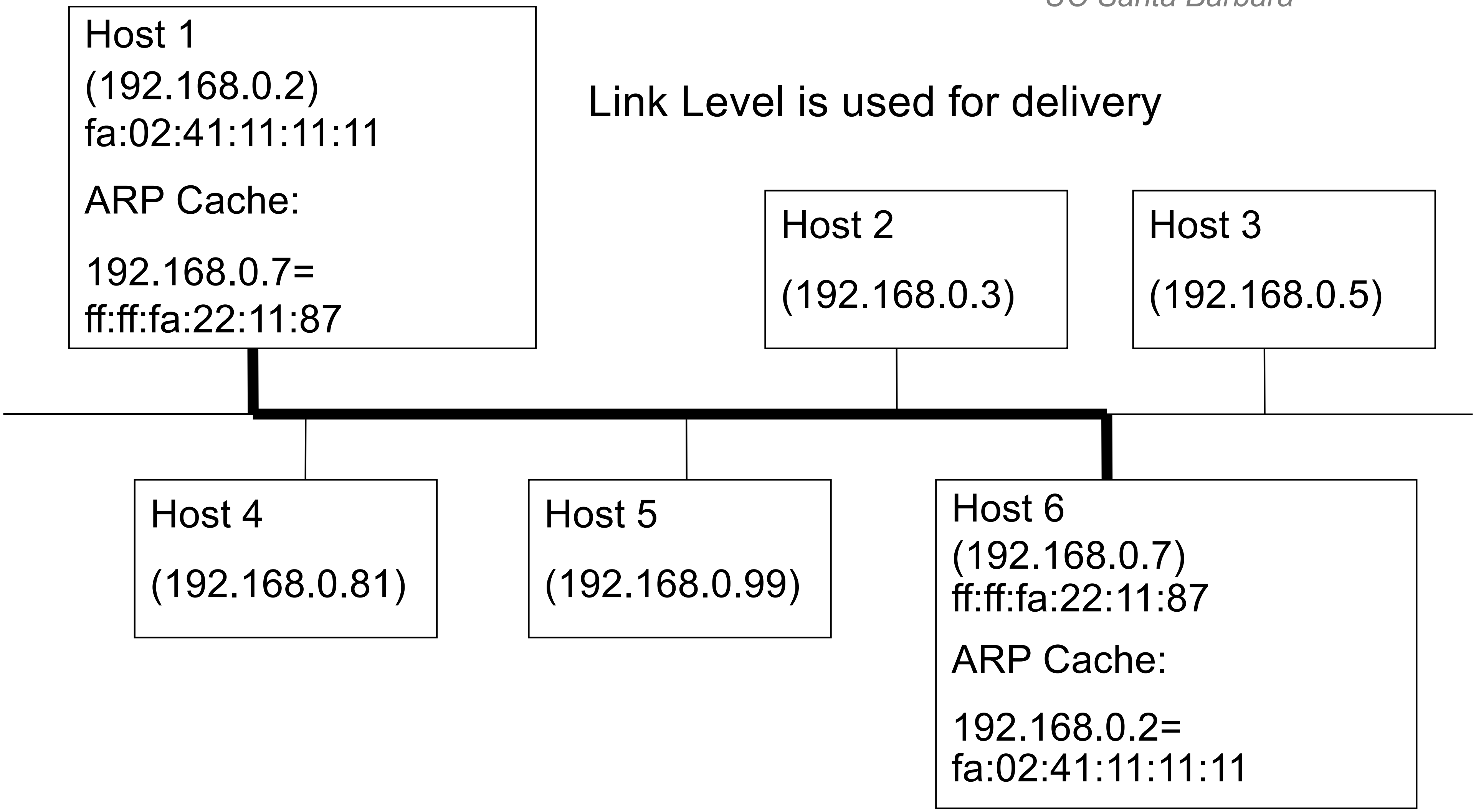
- Assume Host A wants to send an IP packet to Host B, and that all ARP caches are empty

Procedure

- Host A sends ARP request for IP-B.
- Host B sends ARP answer to Host A
- ARP caches on Hosts A and B are filled
- Host A sends encapsulated IP datagram on link level to Host B

Direct Packet Delivery

Link Level is used for delivery



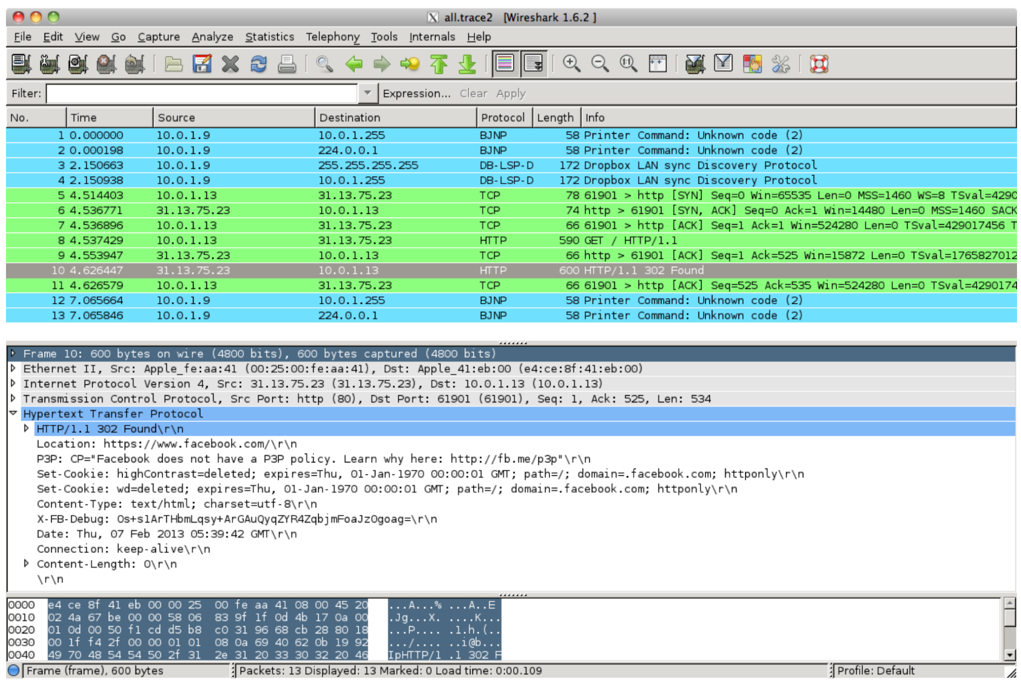
Link Layer Attacks

- Attacker Goals
  - Obtain sensitive information
  - Impersonate host
  - Disrupt data delivery

- Methods
  - Sniffing
  - ARP cache poisoning

Sniffing

- Read packets meant for other machines
  - Network card switched to *promiscuous mode*
  - May require root access on the machine
- Leverage tools such as *tcpdump* and *wireshark*



ARP Attacks

ARP provides no means of authentication (cannot trust ARP packets)

ARP Cache Poisoning

- Racing against the queried host is possible
  - provide false IP address/link-level address mapping

→ result in a redirection of traffic to the attacker

ARP messages sent continuously to keep fake entries in caches

Can also be used to impersonate the gateway and filter ALL the traffic

or map gateway IP to non-existent MAC address (denial of service)

Broadcast vs. Switched Ethernet

- Ethernet switch learns mappings between ports (of the switch) and physical address of attached devices
  - allows targeted delivery of Ethernet frames
  - avoid broadcast behavior

- Mappings contained in a content-addressable memory (CAM)

- Attacker can use MAC flooding to overflow the CAM
  - attacker pretends that tons of devices are connected to a port
  - with overflow, switch reverts back to broadcast packets

- Some switches disable a physical port when it seems that too many devices are connected to it

Virtual Local Area Networks (802.1Q)

- Mechanism for partitioning physical networks into separate broadcast domains

- If enforced at network switches, operators can isolate untrusted traffic → compartmentalization

- Contains damage of an endpoint compromise

Wireless Networks

- Wireless networks reintroduce a shared broadcast medium
  - Link-layer attacks are new again in a wireless environment
  - ARP spoofing is a major problem
  - Possible to sniff and intercept many sensitive protocols

- Idea: Introduce cryptographic controls

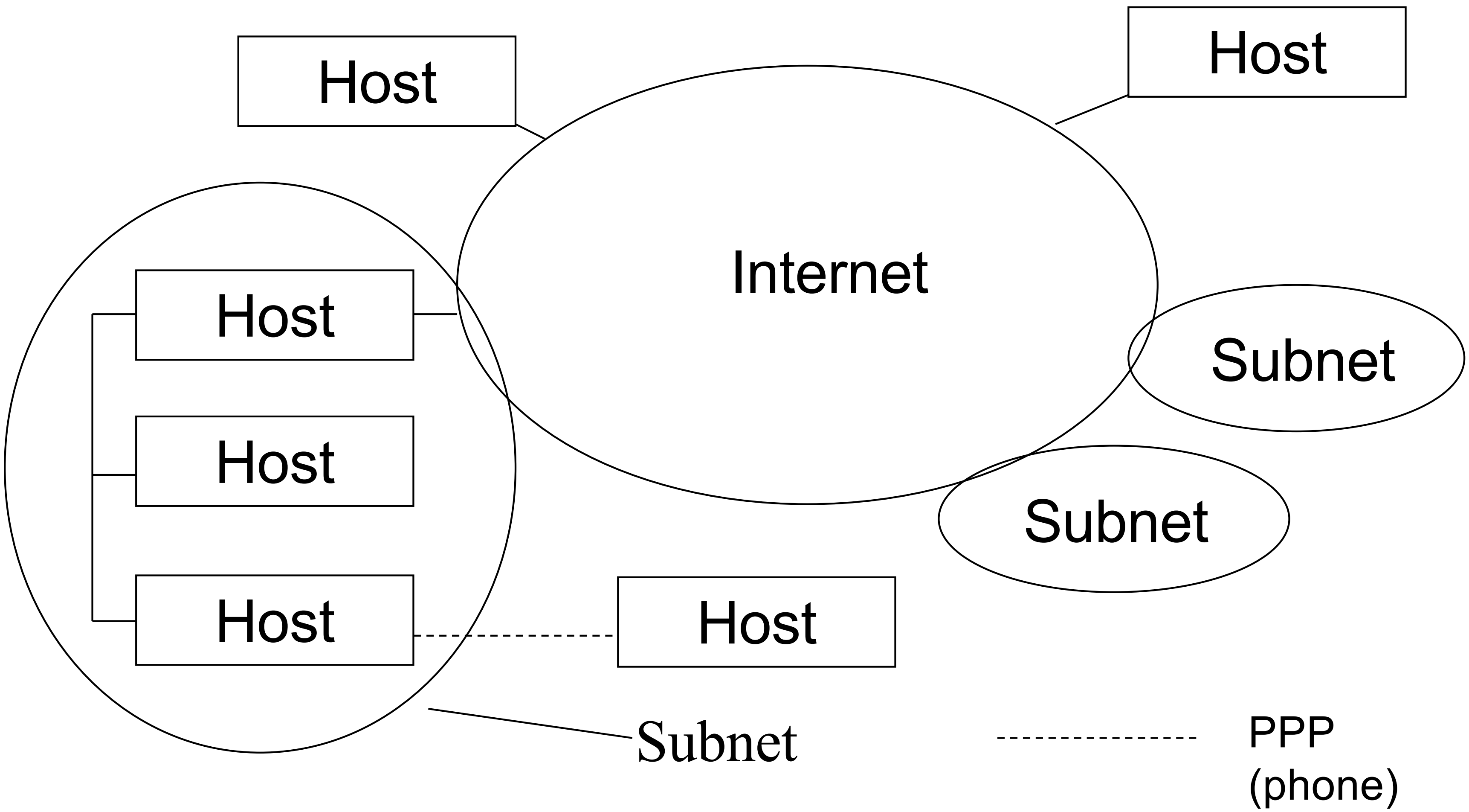
Wireless Networks

- Wired Equivalent Privacy (WEP)
  - First standard for wireless network security
  - Quickly broken: Fluhrer-Mantin-Shamir (FMS) attack

- Wi-Fi Protected Access (WPA) and successor protocols
  - WPA Personal
    - 256-bit pre-shared key (PSK) derived from password
    - Issue: Every client starts with the same password
  - WPA Enterprise
    - Introduce (trusted) authentication server
    - Have each user log in with their own username and password
    - Authentication server provides one-time keys for client and access point
    - *eduroam* uses this approach

NETWORK LAYER (IP)

The Internet



Internet Protocol

- Standard for “internetworking” → Internet
- Standardized addresses and routing
  - addresses: v4: 4 bytes, v6: 16 bytes
  - classic example: 127.0.0.1
- Packet abstraction
  - Packets can be dropped or reordered
  - Payloads can be corrupted

IPv4 Packet

<table>
<tr><td>Version</td><td>Header Length</td><td>Type of Service</td><td>Total Length (2 bytes)</td></tr>
<tr><td colspan="3">Identification (2 bytes)</td><td>Fragmentation Info (2 bytes)</td></tr>
<tr><td colspan="2">Time to Live (TTL)</td><td>Protocol</td><td>Header Checksum (2 bytes)</td></tr>
<tr><td colspan="4">Source IP Address (4 bytes)</td></tr>
<tr><td colspan="4">Destination IP Address (4 bytes)</td></tr>
<tr><td colspan="4">Optional Header and Payload</td></tr>
</table>

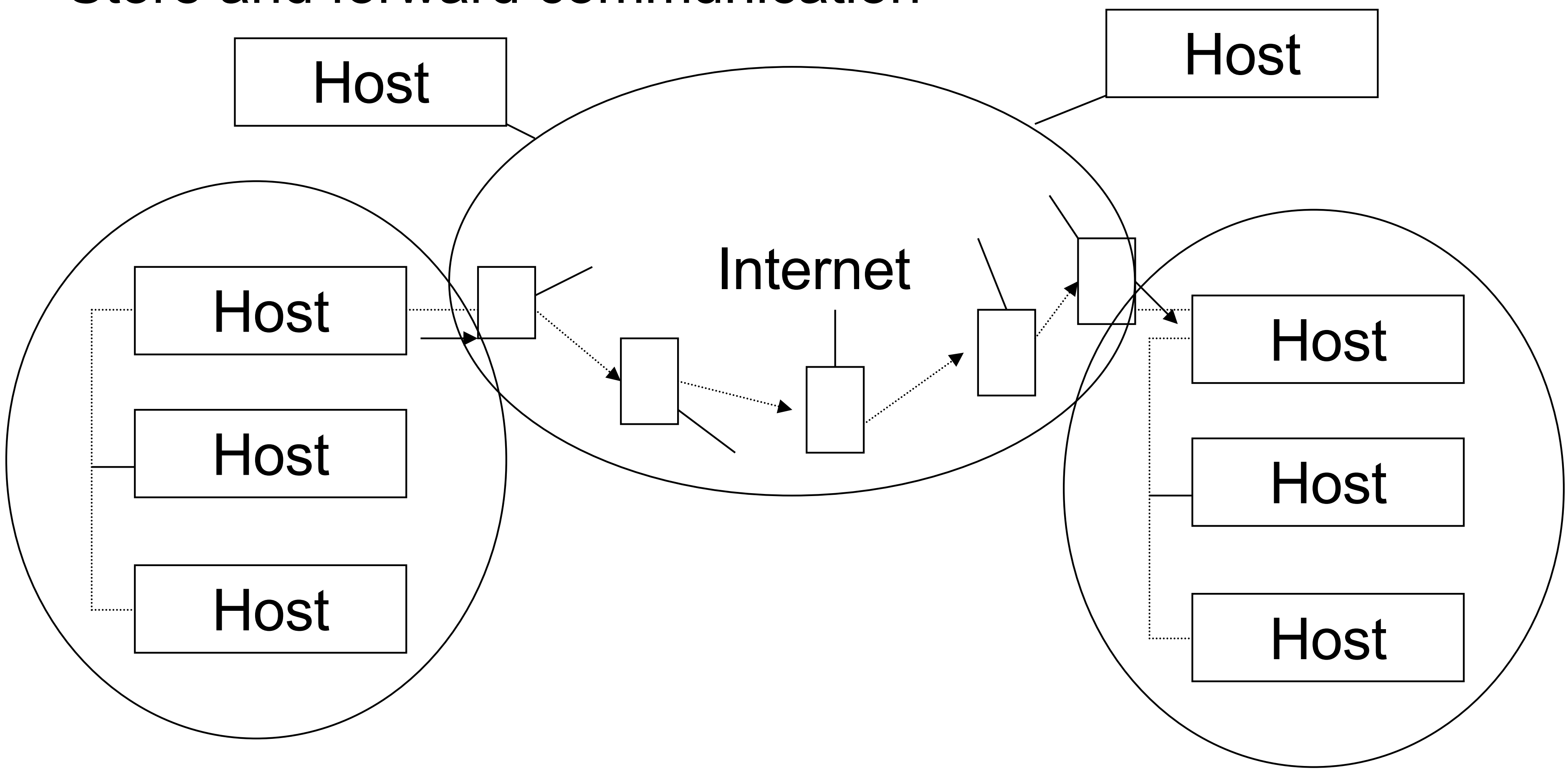
Indirect Delivery: Routing

If hosts are in different physical networks, packet cannot be delivered directly

- Packet is forwarded to a gateway / router
  - has access to other network(s)
  - decides upon destination where to send the packet next
  - this is repeated until packet arrives at network with target host
  - then direct delivery is performed
  - link level addresses change at every step

Indirect Delivery: Routing

Store and forward communication



Routing Protocols

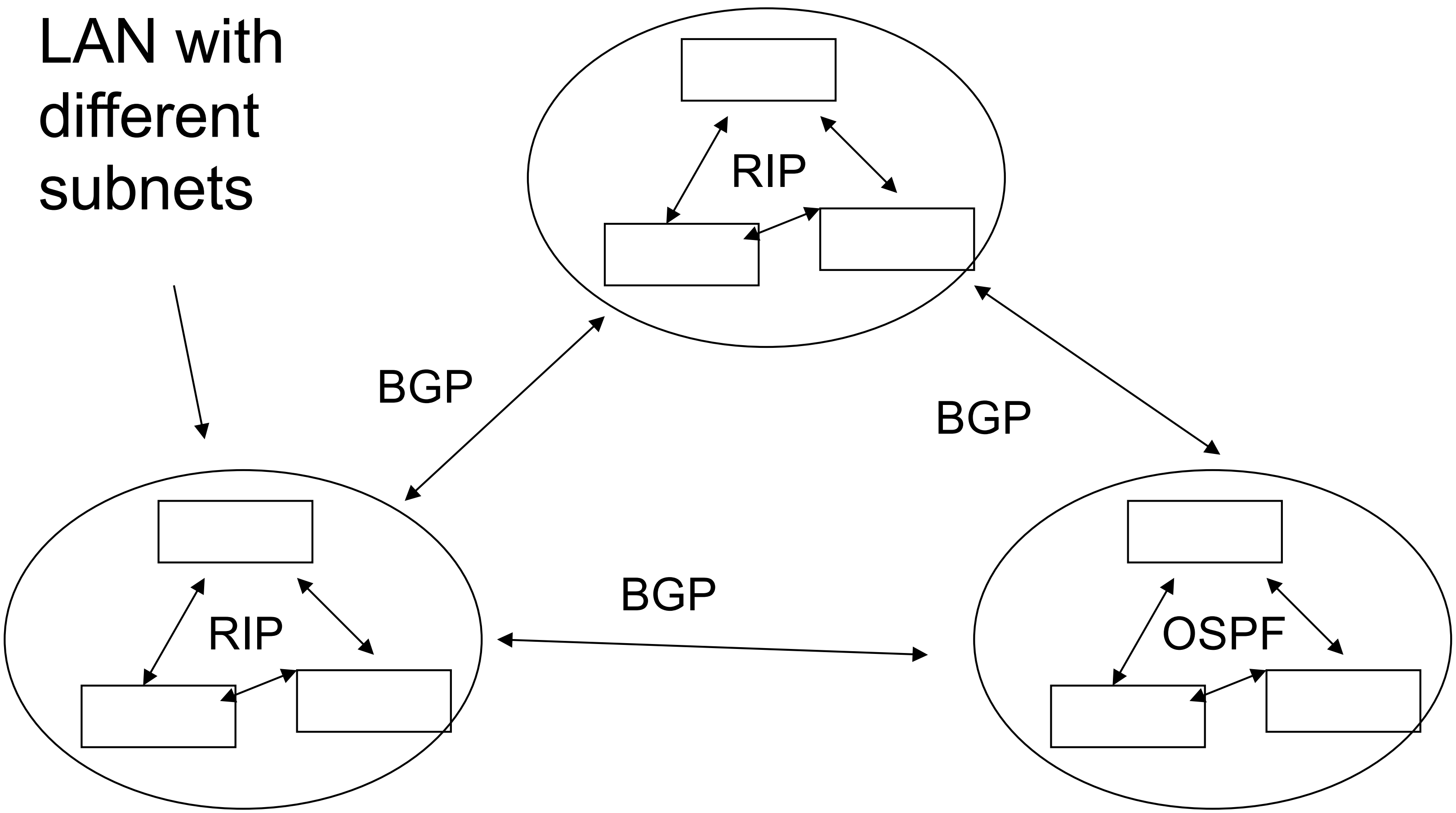
- Automatically distribute information about delivery routes
- Hierarchically organized with different scope

- Divided in
  - exterior gateway protocols (EGPs)
    - distribute information between different autonomous systems (AS)
    - e.g., Border Gateway Protocol (BGP) for Internet backbone
  - interior gateway protocols (IGP)
    - distribute information inside autonomous systems (AS)
    - e.g., Routing Information Protocol (RIP) or OSPF

- “Autonomous” means under a single administrative control

Routing Protocols

LAN with
different
subnets



IP Fragmentation

- IP datagrams must tolerate varying segment maximum transmission units (MTUs) of underlying data link layer (just as Ethernet)

- How long can an IP packet be? What is the longest Ethernet frame?

- One strategy → break up (fragment) IP packets

- Receiving endpoint (or – less often – router) reassembles fragments into the original packet

IP Fragmentation

- Involves several fields in the IP header

ID field (16 bits)
→ same value for all fragments

Bit 0: Reserved; must be zero
Bit 1: Don't Fragment (DF)
Bit 2: More Fragments (MF)
Bits 3-15: Fragment Offset into IP packet, in units of 8 bytes

<table>
<tr><td>Version</td><td>Header Length</td><td>Type of Service</td><td colspan="2">Total Length (2 bytes)</td></tr>
<tr><td colspan="3">Identification (2 bytes)</td><td colspan="2">Fragmentation Info (2 bytes)</td></tr>
<tr><td colspan="2">Time to Live (TTL)</td><td>Protocol</td><td colspan="2">Header Checksum (2 bytes)</td></tr>
<tr><td colspan="5">Source IP Address (4 bytes)</td></tr>
<tr><td colspan="5">Destination IP Address (4 bytes)</td></tr>
</table>

IP Fragmentation

- Process
  - Total length field (of IP packet) set to fragment length
  - “More fragment” bit set, unless last fragment
  - Fragment offset set to original offset, divided by 8
  - Header checksum recomputed

IP Fragmentation

UC Santa Barbara

- Maximum size of an IP packet?
- Of a reassembled packet?

$$(2^{16} - 1) - 20 = 65515 \ < \ 65528 = (2^{13} - 1) \cdot 8$$

- Ping of death
  - size discrepancy can lead to an overflow at the recipient

Network Layer Attacks

- IP packet spoofing
- Man-in-the-middle attacks
- Denial of service (DoS) attacks

IP Packet Spoofing

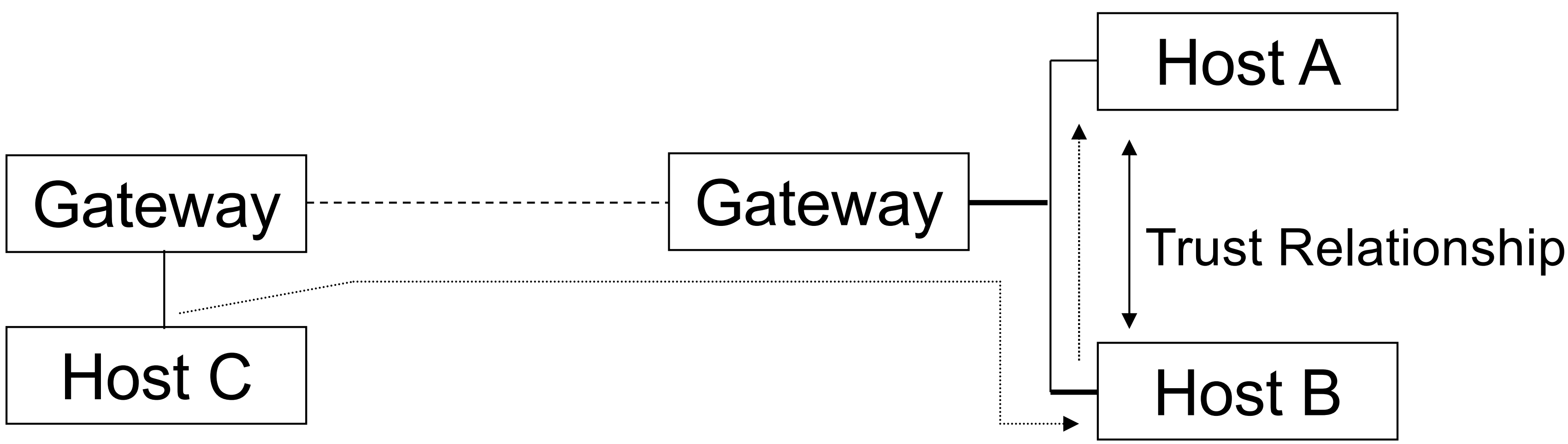
- IP addresses are often used as access control principals
  - Host-based authentication
  - Security middlebox allow lists and block lists
- But are IP addresses an appropriate authentication factor?
- Absolutely not!
  - Addresses can be spoofed
  - Addresses can be reassigned

IP Packet Spoofing

- Anybody can send their own packets through the network
- Spoofing
  - Lying about the identity of the sender
  - Example: Mallory sends a message and claims that the message is from Alice
  - Attacker controls (and hence, can lie) about the source address in the packet header
- Some spoofing attacks may be harder if the attacker cannot see responses (blind spoofing versus on-path attackers)

Blind IP Spoofing

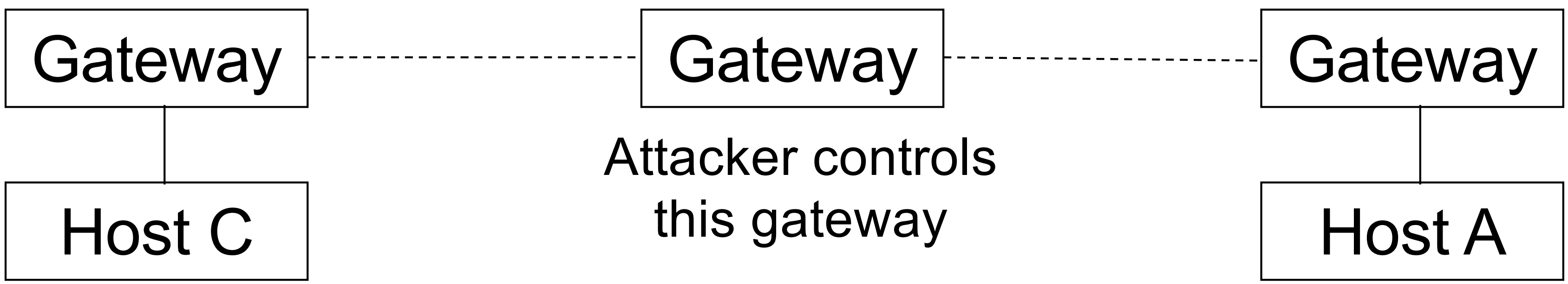
- Attacker does not have access to the reply
- Attempts to abuse trust relationship between hosts
  - Host C (attacker) sends an IP packet with the address of some other Host A as the source address to Host B
  - Victim Host B replies to the legitimate Host A



Man-in-the-Middle Attack

Attacker controls a gateway that is used in the delivery process can

- sniff the traffic
- intercept, block, and delay traffic
- modify traffic



- Difficult to do in the general case (on the Internet)
  – need to control backbone host on the Internet
  – possible in cases where attacker is closer to one of the end points

Denial-of-Service (DoS)

- IP flooding is a canonical network-based denial-of-service (DoS) attack

- Overwhelm the victim's bandwidth (amount of data it can upload/download in a certain time)
  - Example: The server can only upload/download 10 MB/s. The attacker sends the server 20 MB/s.
    - Lots of maximum-sized packets

- Overwhelm the victim's packet processing capacity
  - Example: The server can process 10 packets/second. The attacker sends the server 20 packets/second.
    - Lots of minimum-sized packets

Distributed Denial-Of-Service (DDoS)

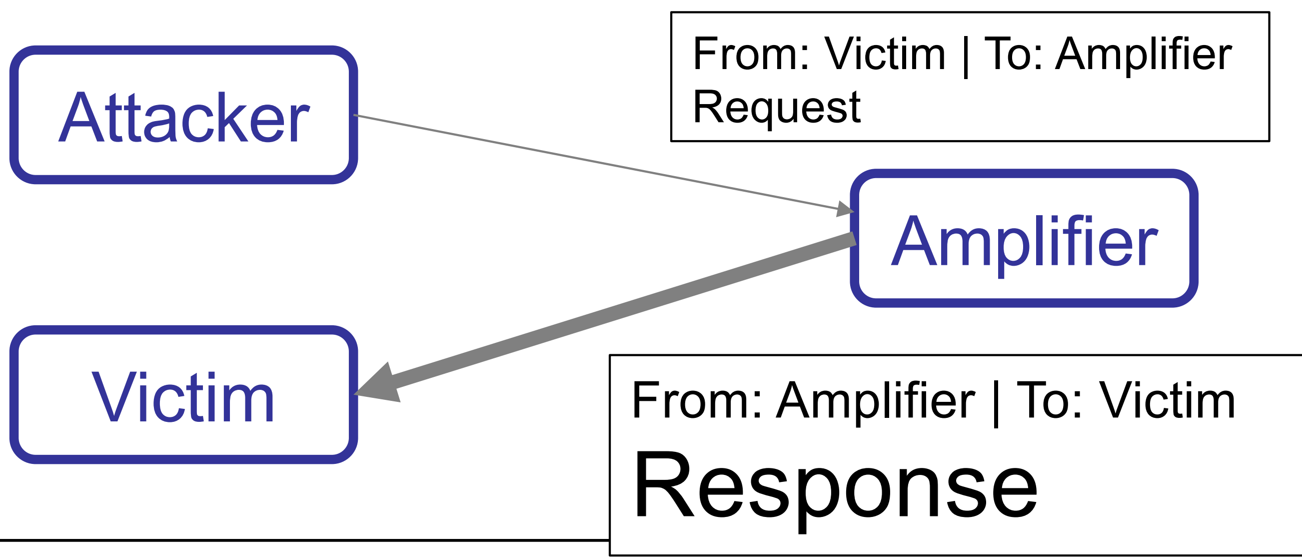
- Use multiple systems to overwhelm the target system
  - Controlling many systems gives the attacker a large amount of bandwidth
  - Sending packets from many sources makes it hard for packet filters to distinguish DDoS traffic from normal traffic
  - Botnet: A collection of compromised computers controlled by one attacker
    - Attacker can tell all the computers on the botnet to flood a given target

DDoS attack that disrupted internet was largest of its kind in history, experts say

Dyn, the victim of last week's denial of service attack, said it was orchestrated using a weapon called the Mirai botnet as the 'primary source of malicious attack'

Amplified Denial-of-Service

- Use an amplifier host to overwhelm the target more effectively
  - Idea: Some services send a large response when sent a small request (e.g., DNS, NTP, memcached)
  - Spoofing a small request that appears to come from the victim results in a large amount of data sent to the victim
  - Requires blind spoofing capability (protocols based on UDP)



TRANSPORT LAYER

Transport Layer

Many transport layer protocols use IP as the underlying network layer

- Important transport layer protocols are
  - ICMP (Internet Control Message Protocol)
  - UDP (User Datagram Protocol)
  - TCP (Transmission Control Protocol)

ICMP

ICMP (Internet Control Message Protocol)

- is used to exchange control/error messages about the delivery of IP datagrams
- ICMP messages are encapsulated inside IP datagrams

- ICMP messages can be:
  - Requests
  - Responses
  - Error messages
    - includes header and first 8 bytes of offending IP datagram

ICMP Message Format

type (1 byte)	code (1 byte)	checksum (2 bytes)
data		

Type field: specifies the class of the ICMP message

Code field: specifies the exact type of the message

ICMP Messages

Examples

- Echo request/reply
  - used to test connectivity (ping)
- Time exceeded
  - used to report expired datagrams (TTL=0)
- Destination unreachable
  - used to inform a host that it is impossible to deliver traffic to a specific destination

ICMP Echo

- Used by the ping program

type (1 byte)	code (1 byte)	checksum (2 bytes)
identifier (2 bytes) = Process ID		sequence number (2 bytes)
data		

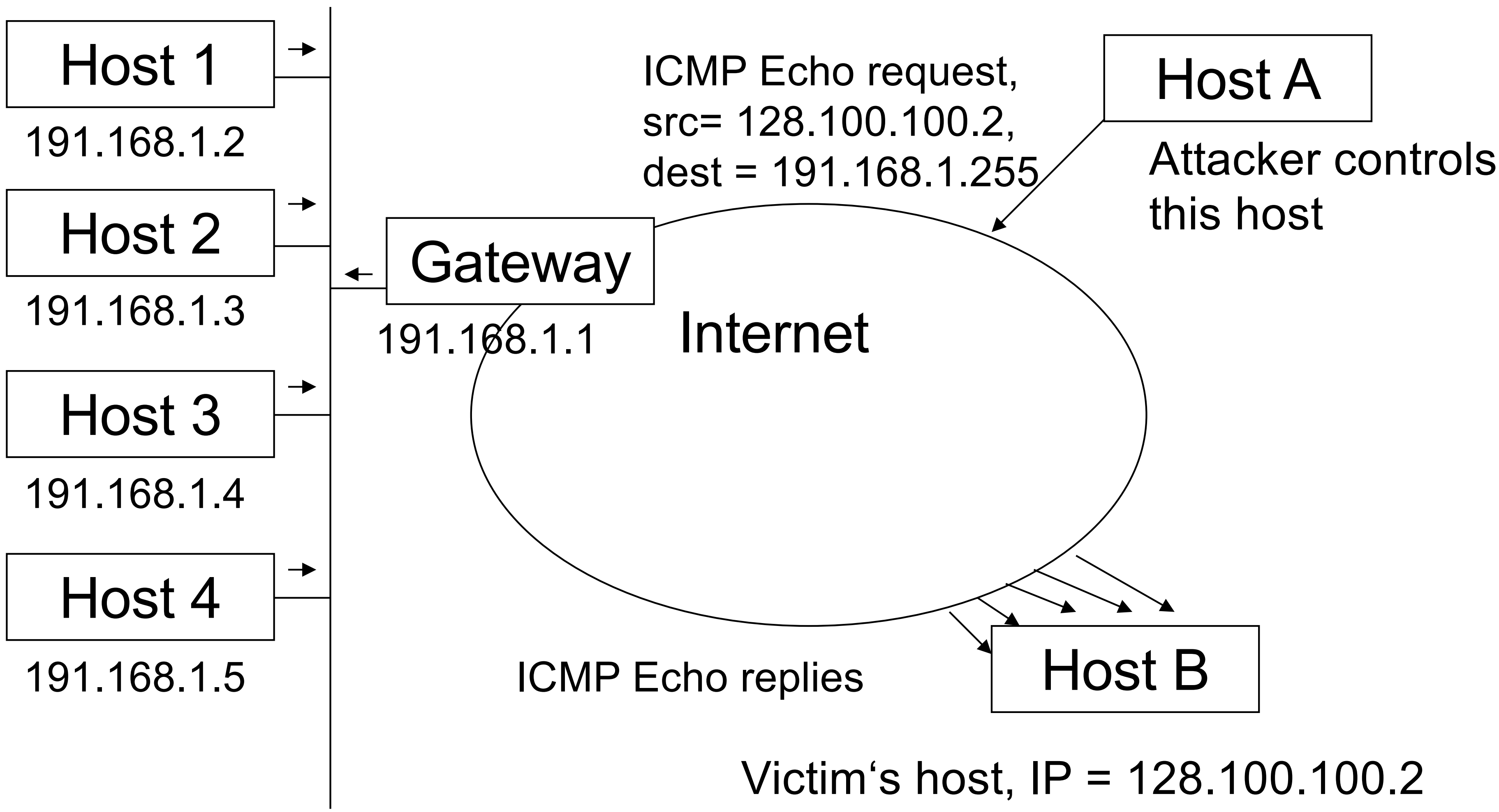
identifier is used by „ping“ to deliver back the packet to the right process (allowing more than one ping to run concurrently)

ICMP Echo Attacks

- Map the hosts of a network
  - ICMP echo datagrams are sent to all the hosts in a subnet
  - attacker collects the replies and determines which hosts are alive

- SMURF attack
  - amplified DoS attack
  - send spoofed (with victim's IP address) ICMP Echo Requests to subnets
  - victim will get ICMP Echo Replies from every machine

Smurf Attack



User Datagram Protocol

UDP (User Datagram Protocol)

- relies on IP
- connectionless
- unreliable (checksum optional)
- best-effort
- datagram delivery service

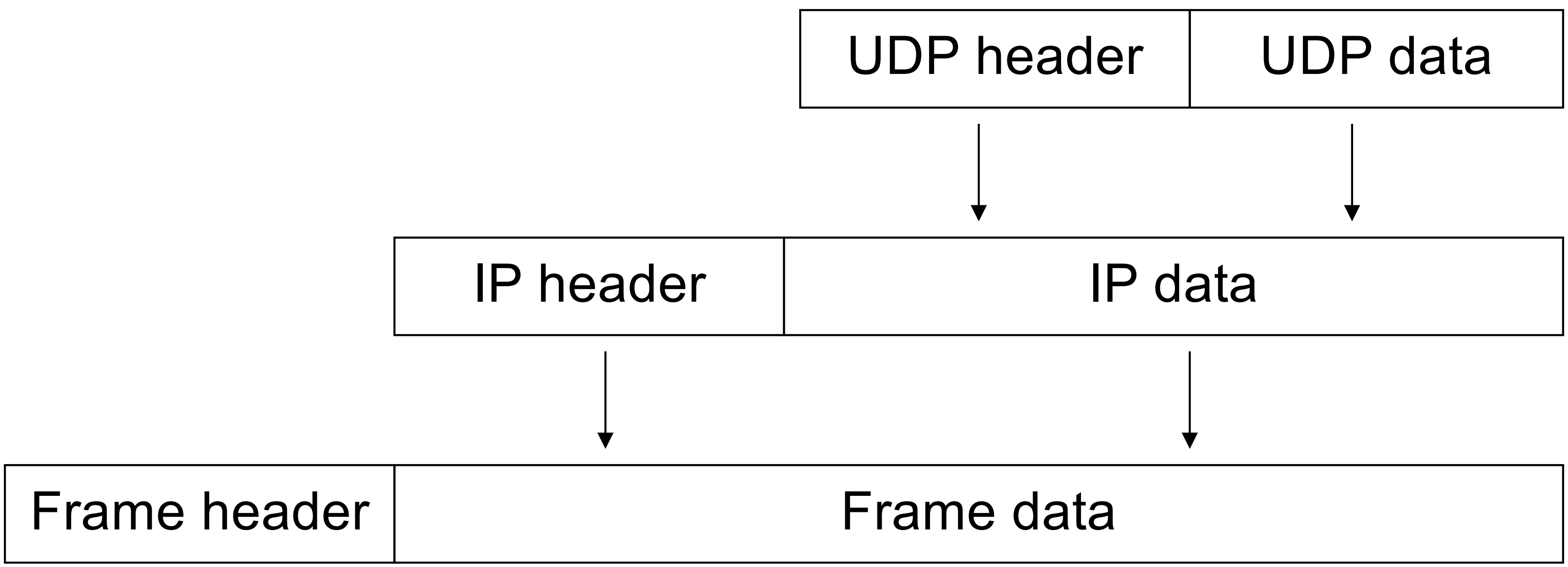
➢ delivery, integrity, non-duplication and ordering are <u>not</u> guaranteed

UDP Message

- UDP provides port abstraction
  - this allows addressing different destinations for the same IP
- UDP is often used for multimedia, games, P2P networks
  - and for services based on request/reply schema (DNS, RPC, NFS)
  - more efficient than TCP

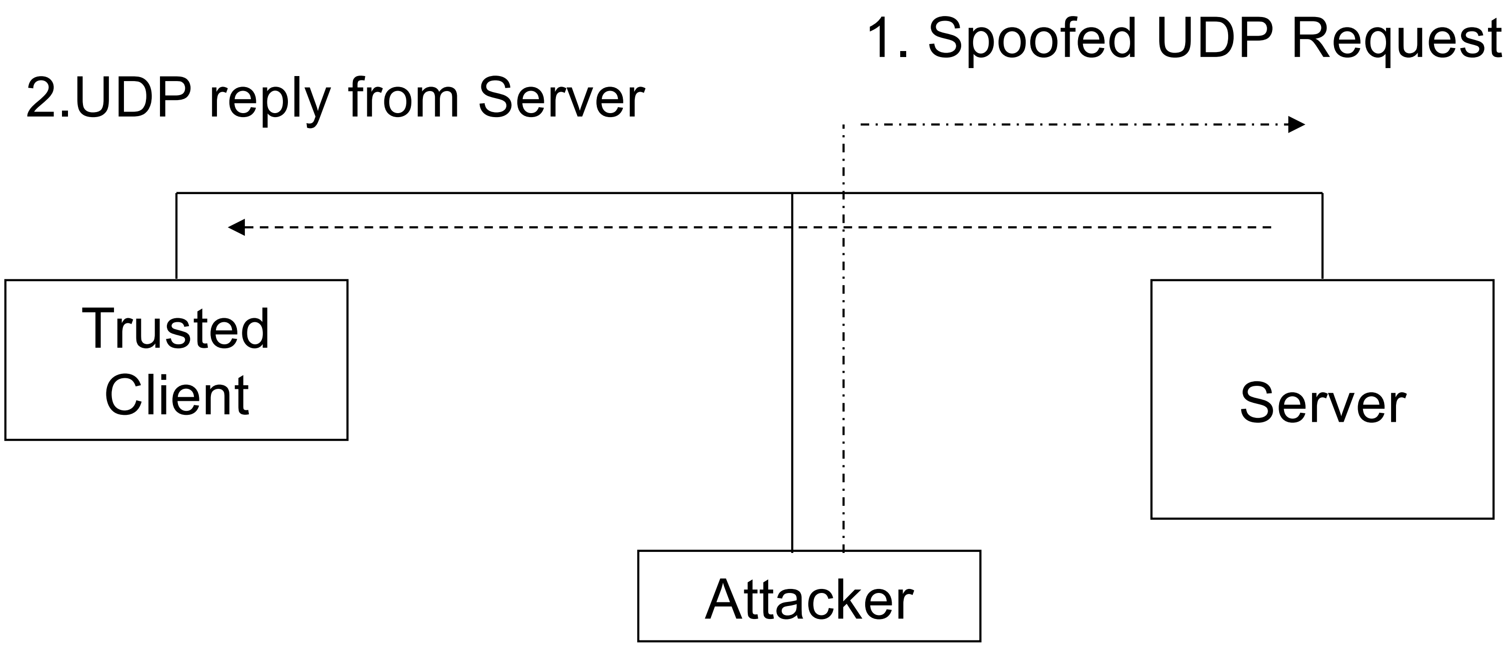
UDP source port (2 bytes)	UDP destination port (2)
UDP message length (2)	Checksum (2)
Data	

UDP Encapsulation



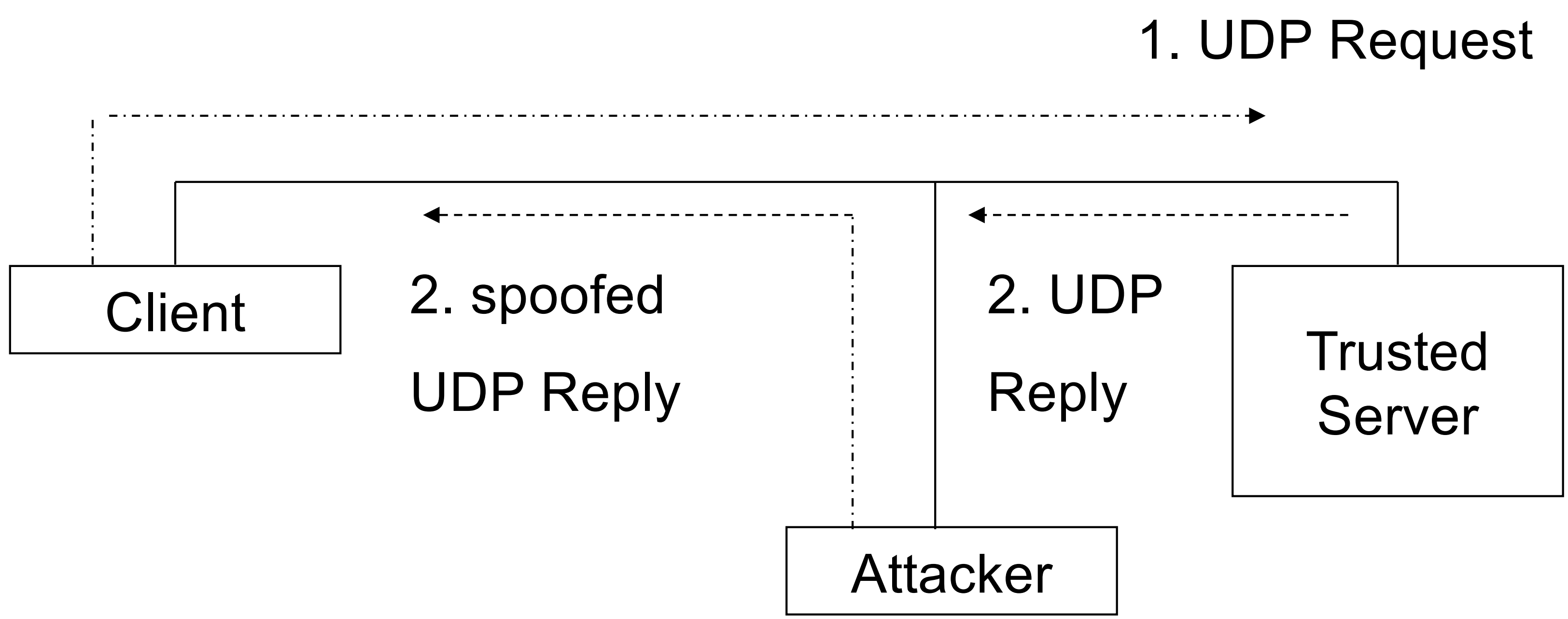
UDP Spoofing

- Basically, the same as IP spoofing, and as easy to perform



UDP Hijacking

- Variation of the UDP spoofing attack
- Racing against the legitimate server



UDP Storms

- Spoofed UDP datagram is sent to the echo service (port 7)
- Source port is set to the chargen device (port 19)
- Reply of the echo service is interpreted as a request by the chargen service
- Reply of the chargen service is interpreted as a request by the echo service
- ... etc …
- Same attack can be carried out using two echo services

TCP

TCP (Transmission Control Protocol)

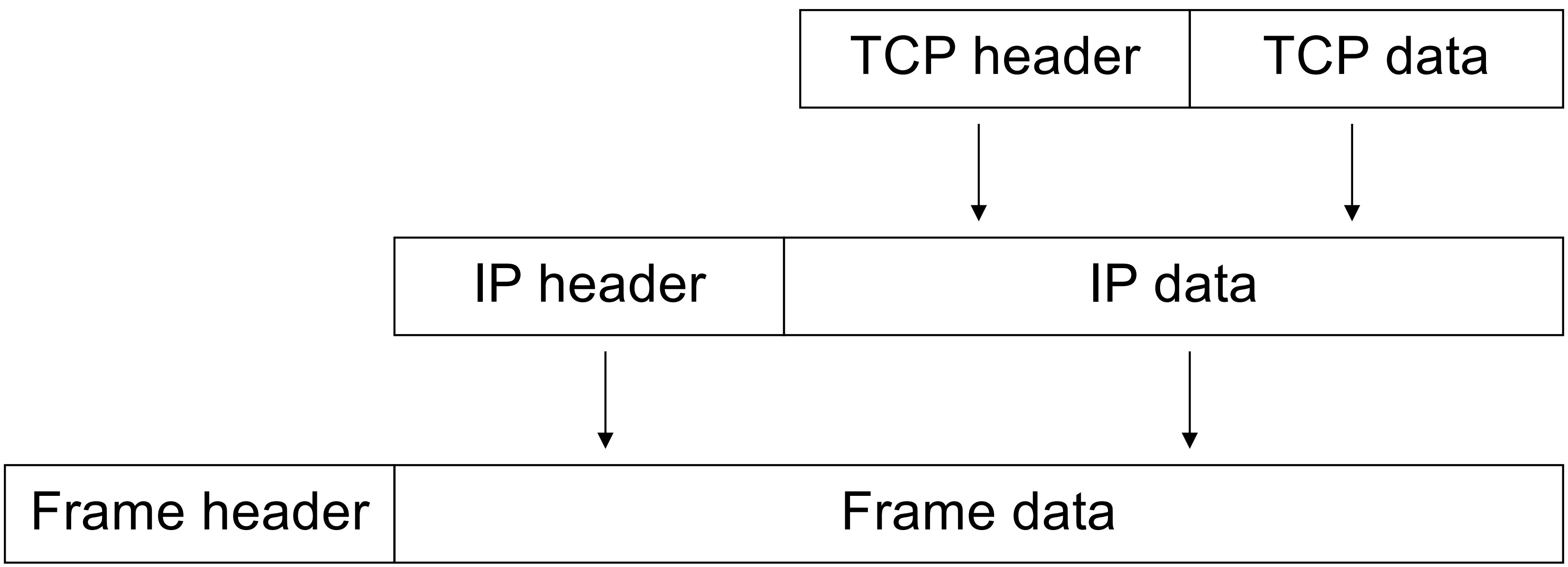
- built on top of IP and provides
  - connection-oriented
  - reliable
  - stream delivery service

➢ no loss, no duplication, no transmission errors, correct data ordering

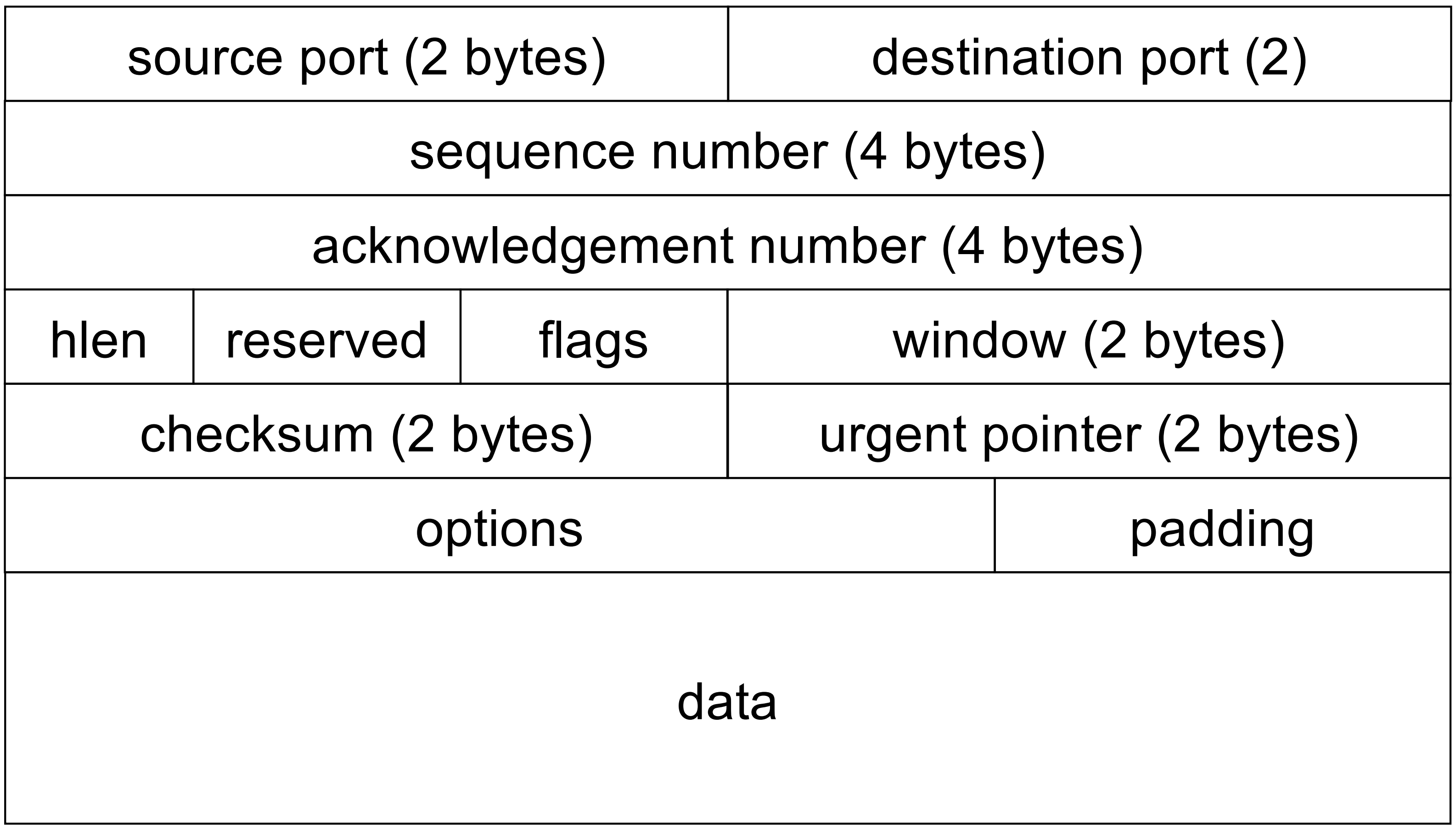
TCP

- Provides port abstraction (as with UDP)
- Allows two nodes to establish a virtual circuit
  - identified with 4-tuple: <src_ip, src_port, dst_ip, dst_port>
  - virtual circuit is composed of two streams (full duplex)
- The pair <IP address, port> is called a *socket*

TCP Encapsulation



TCP Header


source port (2 bytes)
destination port (2)
sequence number (4 bytes)
acknowledgement number (4 bytes)
hlen
reserved
flags
window (2 bytes)
checksum (2 bytes)
urgent pointer (2 bytes)
options
padding
data

TCP Seq/Ack Numbers

- Sequence number (seq)
  - specifies the position of the segment data in the communication stream
  - seq = 1234 means:
    The payload of this segment contains data starting from 1234
- Acknowledgement number (ack)
  - specifies the position of the *next expected byte* from the communication partner
  - ack = 12345 means:
    I have received the bytes correctly to 12344, I expect the next byte to be 12345
- Both are used to manage error control
  - retransmission, duplicate filtering

TCP Window

- Used to perform flow control
- Segment will be accepted only if the sequence number has a value between
  - last ack number sent and
  - last ack number sent + window size
- The window size changes dynamically to adjust the amount of information that can be sent by the sender
  - set by the receiver to announce how much it can take
  - window size = amount of data the client can handle now

TCP Flags

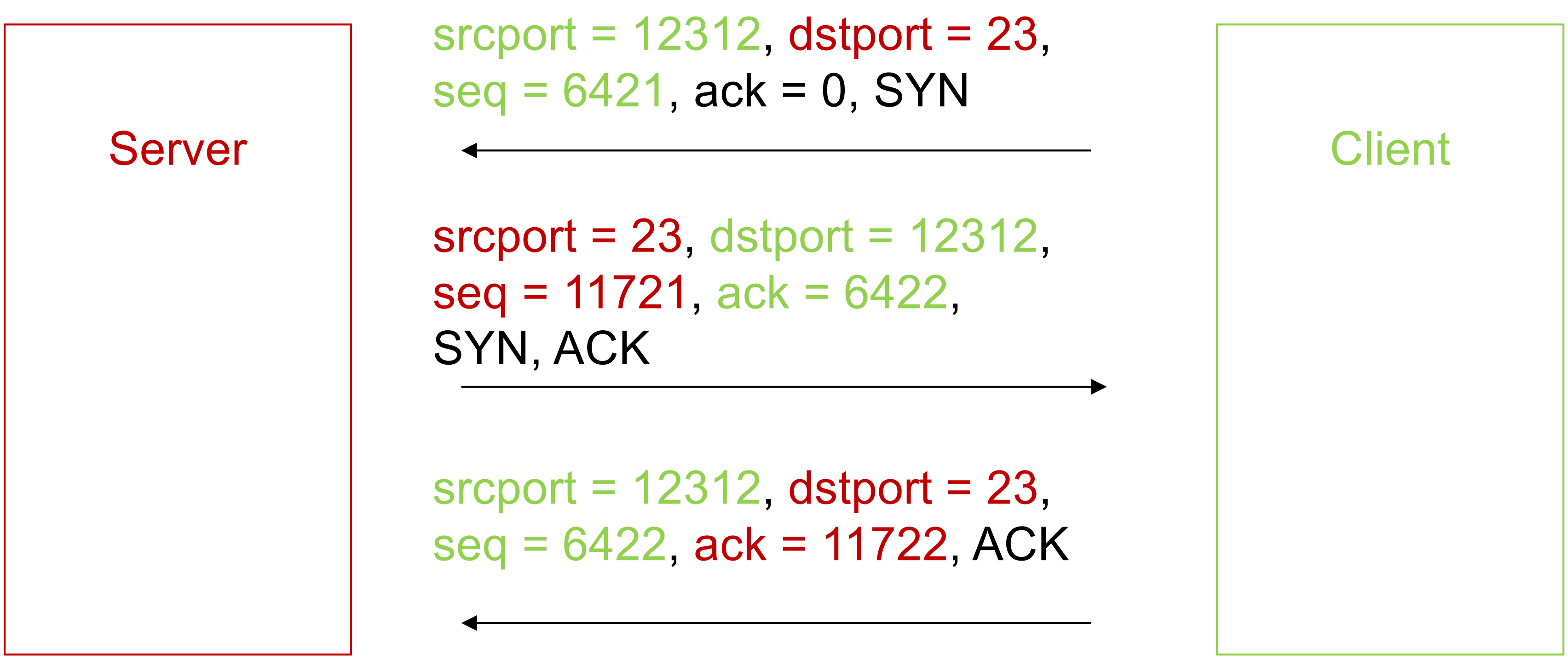
- Flags are used to manage the establishment and shutdown of a virtual circuit
  - SYN: request for synchronization of seq/ack numbers (used during connection setup)
  - ACK: the acknowledgement number is valid (all segments in a virtual circuit have this flag set, except the first)
  - FIN: request to shutdown a virtual circuit
  - RST: request to immediately reset the virtual circuit
  - URG: states that the urgent pointer is valid
  - PSH request a „push“ operation on the stream (pass the data to the application (interactive) as soon as possible)

TCP Virtual Circuit: Setup

- A server listens to a specific port
- Client sends a connection request to the server, with SYN flag set and a random initial sequence number c
- The server answers with a segment marked with both the SYN and ACK flags and containing
  - an initial random sequence number s
  - c+1 as the acknowledge number
- The client sends a segment with the ACK flag set and with sequence number c+1 and ack number s+1

Three Way Handshake

- Three way because 3 TCP segments are necessary to set up a virtual circuit



Initial Sequence Number

- The TCP standard (RFC 793) specifies that the sequence number should be incremented every 4 μs
- BSD UNIX systems initially used a number that is incremented by 64000 every half second and by 64000 each time a connection is established
- This number is important, because it adds protection against certain attacks (TCP reset, TCP spoofing, TCP hijacking, …)

TCP Data Exchange

- Each TCP segment contains
  - sequence nr = position of data in stream (often, last ack number)
  - ack nr = sequence number of last correctly received segment increased by the payload size of this segment

- A partner accepts a segment of the other partner only if the numbers are inside the transmission window

- An empty segment may be used to acknowledge the received data

- Packets with no payload and SYN or FIN flag consume one sequence number

TCP Data Exchange

Server

Client

srcport = 12312, dstport = 23, seq = 6422 (10 bytes), ack = 11722, ACK

srcport = 23, dstport = 12312, seq = 11722 (25 bytes), ack = 6432, ACK

srcport = 12312, dstport = 23, seq = 6432 (99 bytes), ack = 11747, ACK

Acknowledgement

- Not sent directly after data has been received
- Delayed ACK: if some data has been received, the receiver waits up to 200 ms in hope that some more data will arrive, which can be acknowledged at once.
- Delayed ACK is only used if no data has to be transported back to the sender

If no ACK is received at the sender (timeout), retransmission takes place

TCP Virtual Circuit: Shutdown

- One of the partners (for example, A) can terminate its stream
  - by sending a segment with the FIN flag set
- B answers with a segment with the ACK flag set
- From this point on, A will not send any data to B, it will just acknowledge data sent by B
  - with empty segments
- When B shuts its stream down, the virtual circuit is considered closed

Sample TCP Connection

From	To	S	A	F	Seq-Nr	Ack-Nr	Payload
192.168.0.1	192.168.0.2	1	0	0	4711	0	0
192.168.0.2	192.168.0.1	1	1	0	38001	4712	0
192.168.0.1	192.168.0.2	0	1	0	4712	38002	0
192.168.0.2	192.168.0.1	0	1	0	38002	4712	‚Login:\n‘ 7
192.168.0.1	192.168.0.2	0	1	0	4712	38009	‚s‘ 1
192.168.0.1	192.168.0.2	0	1	0	4713	38009	‚e‘ 1
192.168.0.1	192.168.0.2	0	1	0	4714	38009	‚c‘ 1
192.168.0.1	192.168.0.2	0	1	0	4715	38009	‚\n‘ 1
192.168.0.2	192.168.0.1	0	1	0	38009	4716	0
192.168.0.1	192.168.0.2	0	0	1	4716	38009	0
192.168.0.2	192.168.0.1	0	1	0	38009	4717	0

TCP Security

- TCP spoofing
- TCP hijacking
- SYN flood (DoS)

TCP Spoofing

Attack aimed at impersonating another host when connecting

- Node A trusts node B (e.g., login with no password)
- Node C (attacker) wants to impersonate B with respect to A in opening a TCP connection
- C suppresses B (flooding, redirecting, crashing)
- C sends A a TCP segment in a spoofed IP packet with B's address as the source IP and an initial sequence number T

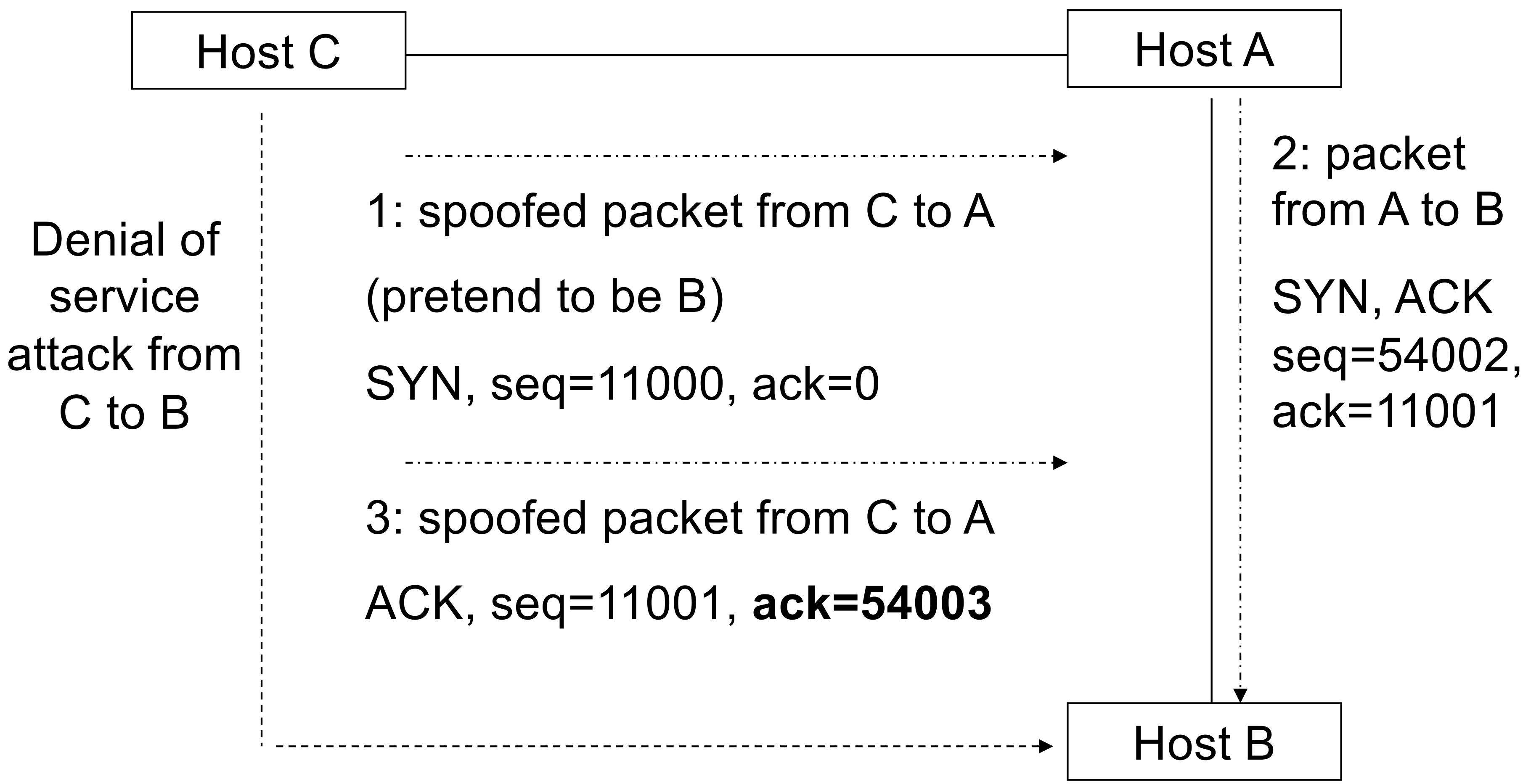
TCP Spoofing

- A replies with a TCP SYN/ACK segment to B with S as the sequence number and T+1 as acknowledge number
- C does not receive the segment from A to B, but in order to finish the handshake, it must send an ACK segment with S+1 as the acknowledge number to A

For this two possibilities exist:

- On path attacker → C eavesdrops the SYN/ACK segment and computes the number
- Blind TCP Spoofing → C guesses the correct sequence number

TCP Spoofing



TCP Hijacking

- Technique to take control of an existing TCP connection

- The attacker uses spoofed TCP segments to
  - insert data into the streams
  - reset existing connections (RST attack)

- But the correct sequence/acknowledgement numbers must be used (guessed or eavesdropped by the attacker)
  - a bit easier, since attacker only needs to hit the right window
  - still very hard to do when not eavesdropping
    assuming a strong randomness generator for sequence numbers

Syn Flood Attack

- Common denial-of-service attack
- Attacker starts handshake with SYN marked segment
- Victim replies with SYN-ACK segment
- Attacker‘s host stays silent
- A host can only keep a limited number of TCP connections in half-open state. After that limit, connections are not accepted.

- Possible solutions
  - drop half open connections in FIFO manner
  - SYN cookies

Syn Cookies

- Idea
  - Avoid the need to keep server-side state until a connection is fully opened
  - Needs to be backwards compatible

- Basic approach
  - Server encodes a “cookie” into its initial sequence number (ISN) n
  - Client response ACKs $n + 1$
  - Server verifies that the cookie is correct
  - Connection state created from the decoded cookie

Syn Cookies

- Implementation
  - Let t be the current timestamp >> 6 bits, m be a 3-bit encoding of the initial MSS, and S_k be a keyed cryptographic 24-bit hash function
  - Then, given a connection tuple $\langle a_{src}, p_{src}, a_{dst}, p_{dst} \rangle$, a SYN cookie c is

$$c = (t \text{ mod } 32) \parallel m \parallel S_k(asrc, psrc, adst, pdst, t)$$

Network Scanning

- Reconnaissance is often a prerequisite for launching attacks
- Network (or port) scanning is a technique for remotely gathering information on target networks
  - Topology, access control policy, network service availability and versions
- Relies on combination of direct and side-channel leakage

TCP Scanning

- Used to check whether a port is open on a host
- Should be done without letting monitored host know that it is scanned
- Can be used to get some extra information about the host (which applications are running, which versions, …)

In the simplest form, a TCP connection is opened to a ports

- if this succeeds a service is assumed to be available

TCP SYN Scanning

- Also known as „half open“ scanning
- The attacker sends a SYN packet (packet with SYN flag)
  - If the server answers with a SYN/ACK packet, then the port is open (or answers with a RST packet: the port is closed)
- The attacker sends a RST packet instead of an ACK

➢ Therefore, the connection is never opened, and the event might not be logged by the operating system / monitor application

TCP FIN Scanning

- The attacker sends a FIN-marked packet
- In most TCP/IP implementations (not Windows)
  - if the port is closed, a RST packet is sent back
  - if the port is open, the FIN packet is ignored
- Variations of this type of scanning technique
  - XMAS Scan: FIN + PSH + URG set
  - NULL Scan: no flags set

UDP Portscan

- Used to determine which UDP services are available
- Zero-length UDP packet is sent to each port
- If an ICMP error message „port unreachable“ is received, the service is assumed to be unavailable
- Many TCP/IP stack implementations implement a limit on the error message rate, therefore this type of scan can be slow

OS and Service Fingerprinting

- OS fingerprinting
  - allows to determine the operating system of a host by examining the reaction to carefully crafted packets
  - leverages differences in TCP/IP implementations (due to specification ambiguity and implementation flaws)
  - use of reserved flags in the TCP header
  - use of weird combination of flags in the TCP header
  - check the selection of TCP initial sequence numbers
  - analysis of responses to ICMP messages

- Service fingerprinting
  - Most services can also be trivially fingerprinted
  - Banners are very helpful

NMAP

- Is a tool for performing portscans and for OS fingerprinting

- https://nmap.org

- supports many types of scans
  - IP scans
  - TCP portscans (SYN, FIN scanning)
  - UDP portscans
  - OS fingerprinting

Firewalls

- How do you protect a set of systems against external attack?
  - More networked machines = more risk
  - More network services = more risk

- Instead of securing individual hosts, we want to secure entire network!

- Add a single point of access in and out of the network, with a monitor
  - This monitor is called a (network) firewall
  - Ensures complete mediation

- Network access is controlled by a policy
  - Defines what traffic is allowed to exit the network (outbound policy)
  - Defines what traffic is allowed to enter the network (inbound policy)
  - Firewall filters or modifies traffic according to some predicates (a ruleset)

Firewalls

- Types of firewalls
  - Packet filters
    - Operate over individual packets
  - Stateful filters
    - Operate over connection abstraction
  - Application-level
    - Operate over application features

Packet Filter

- Inspect network packets and chooses what to do with them
- Stateless
  - That is, packet filters that have no history
  - All decisions must be made using only the information in the packet
  - Can have trouble implementing complex policies that require knowledge of history
- Consider implementing a typical home network policy
  - Allow all outbound traffic
  - Allow inbound traffic in response to an outbound connection
  - Deny all other inbound traffic
- Issue: How do we know what inbound traffic is in response to an outbound connection?

Packet Filter

- Issue: How do we know what inbound traffic is in response to an outbound connection?
- Hack for TCP
  - Allow inbound segments with the ACK flag set
  - Deny inbound segments without an ACK flag set
  - Why does this (somewhat) work?
- Does not work for UDP

Stateful Filter

- Better idea: Keep state in the implementation of the packet filter
  - The filter keeps track of inbound/outbound connections
  - When a connection has been initiated from the inside, allow return packets
  - Otherwise, deny

- Can we extend state and context to application protocols?
  - Don't just block based on ports, identify the application protocol
  - What about allowing certain HTTP responses based on prior requests?

  → Application-level firewalls

Conclusions

- Internetworking
  - local delivery
    - data link layer
    - point-to-point frame exchange (Ethernet, ARP)
  - end-to-end connection
    - network layer
    - packet switching (IP)
    - routing (routing protocols)
  - additional services for end-to-end connections
    - transport layer
    - error and notification service (ICMP)
    - ports (UDP, TCP)
    - reliable transmission (TCP)