CS 4910: Intro to Computer Security

Network Security

Instructor: Xi Tan

Updates

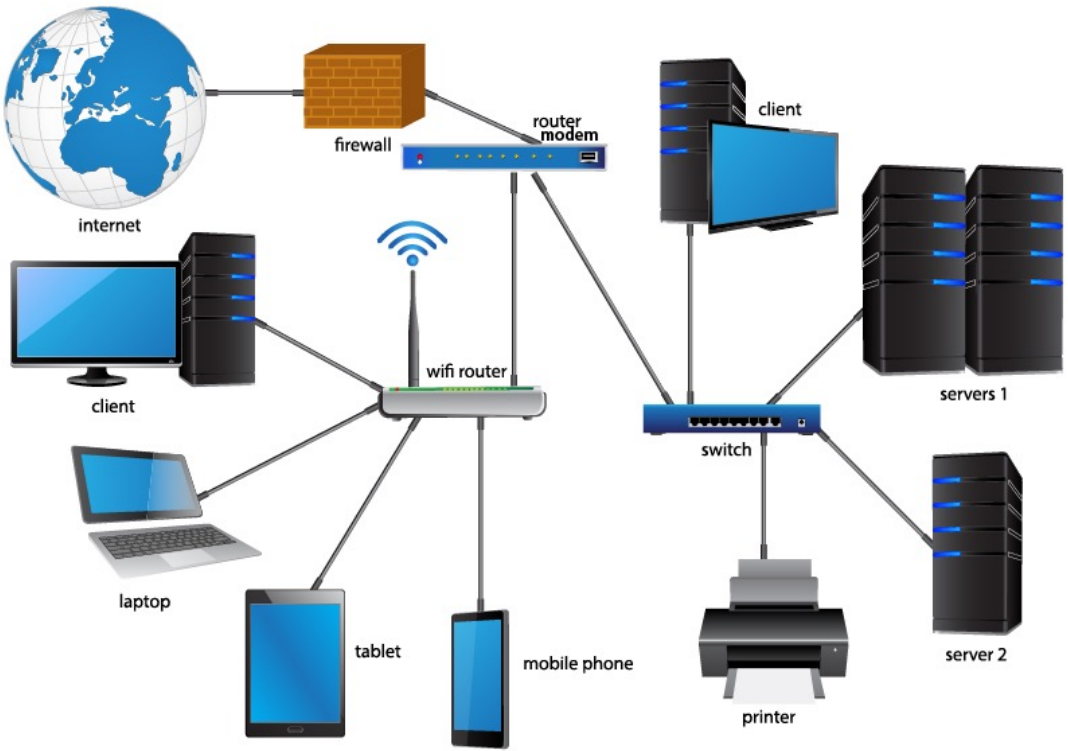
- Research Paper:
  - Research Paper
  - Deadline: 03/23

- Lab 2:
  - Task 1: Packet sniffing and spoofing
  - Task 2: Not required
  - Deadline: 4/08

Network Security

- Computer Network Concepts
- Network Attacks
- Network Security

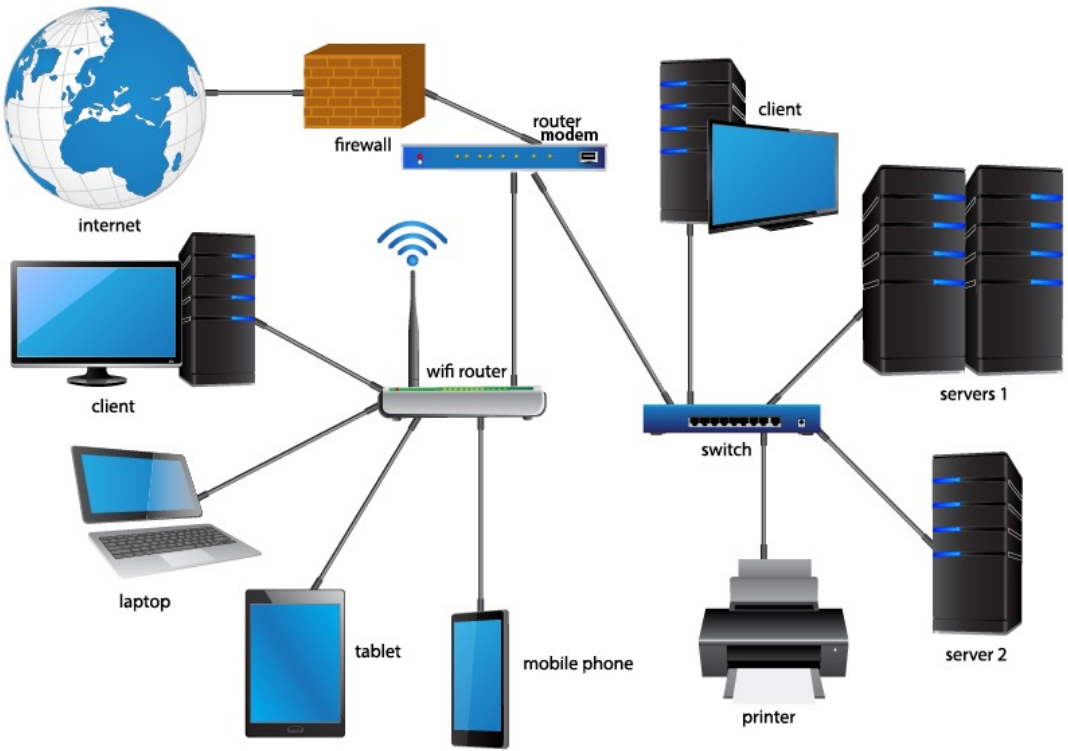
What is A Computer Network?



A computer network is a collection of computers and other devices connected together to communicate and share resources.

Figure is from: https://integrinetit.com/what-is-a-computer-network/

Important Components for A Network



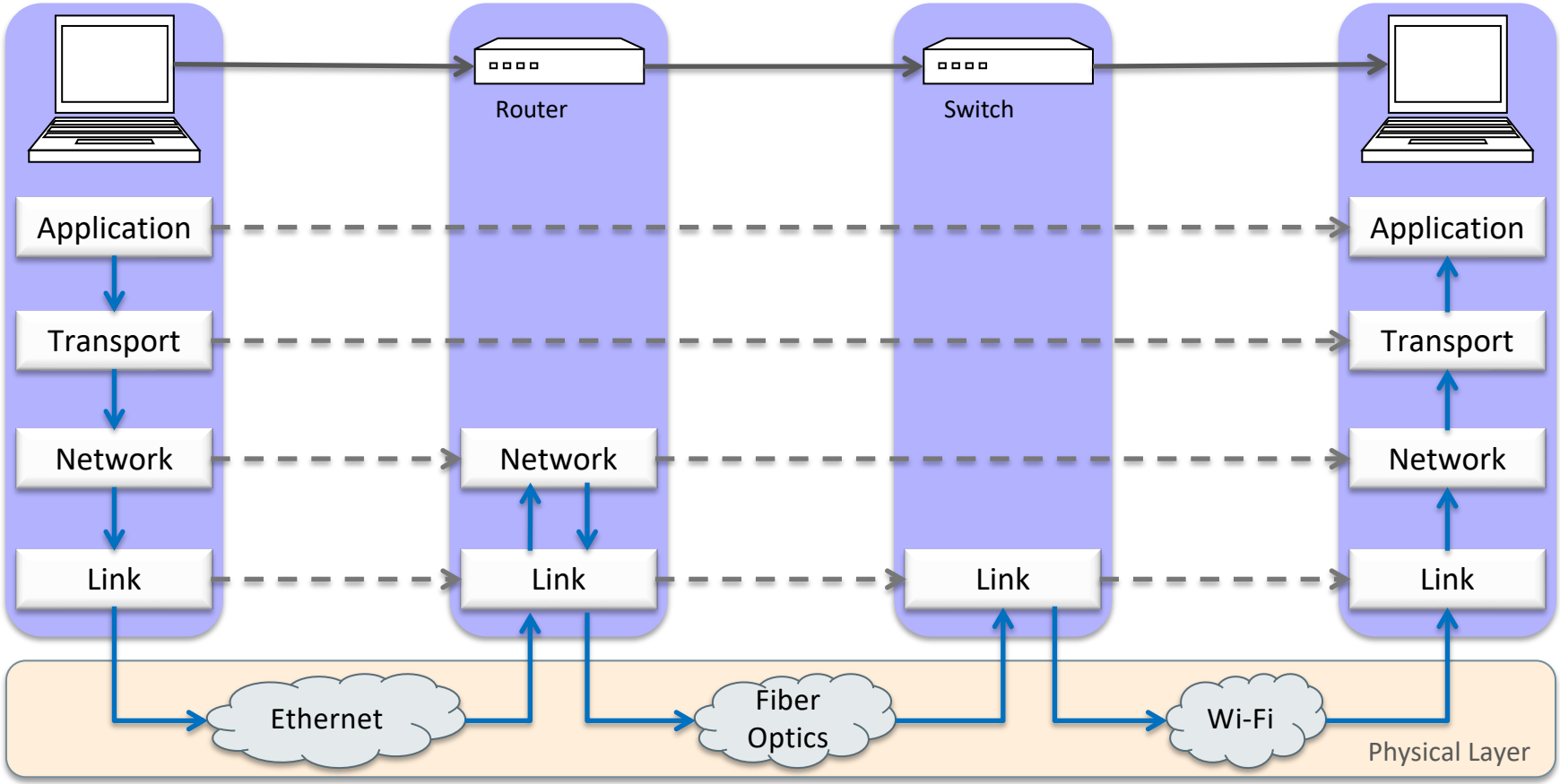
How are those devices connected?

How is the data (frame/packet) transmitted through the connection?

How do those devices interpret the data?

Figure is from: https://integrinetit.com/what-is-a-computer-network/

How are those devices connected? -- Network Layers



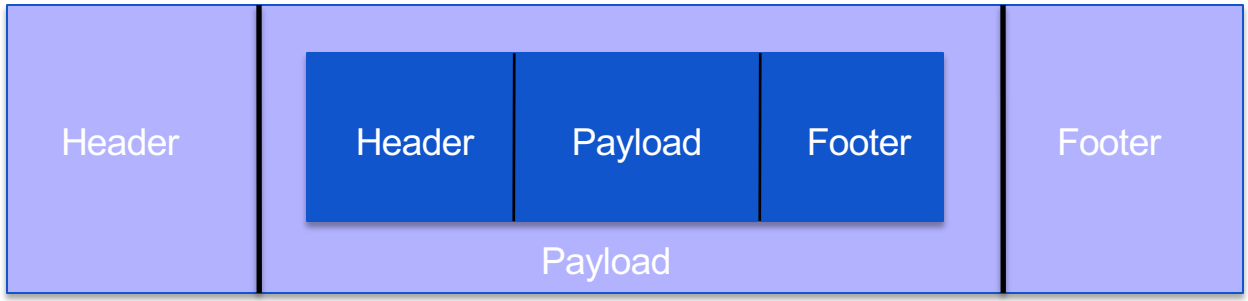
How do those devices interpret the data?

Protocols

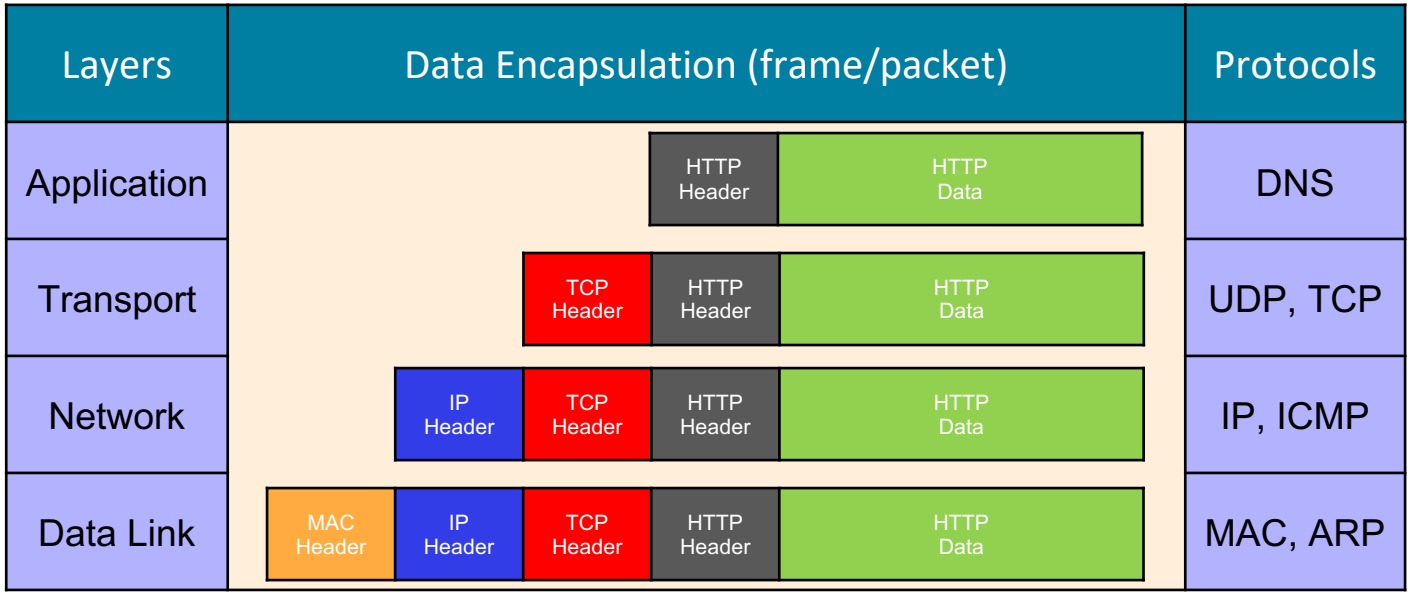
- A protocol defines the rules for communication between computers
- Protocols are broadly classified as connectionless and connection oriented
  - Connectionless protocol
    - Sends data out as soon as there is enough data to be transmitted
    - E.g., user datagram protocol (UDP)
  - Connection-oriented protocol
    - Provides a reliable connection stream between two nodes
    - Consists of set up, transmission, and tear down phases
    - Creates virtual circuit-switched network
    - E.g., transmission control protocol (TCP)

Encapsulation

- A packet typically consists of
  - Control information for addressing the packet: header and footer
  - Data: payload
- A network protocol N1 can use the services of another network protocol N2
  - A packet p1 of N1 is encapsulated into a packet p2 of N2
  - The payload of p2 is p1
  - The control information of p2 is derived from that of p1



Internet Communication



Network Interfaces

- Network interface: device connecting a computer to a network
  - Ethernet card
  - WiFi adapter
- A computer may have multiple network interfaces
- Packets transmitted between network interfaces
- Most local area networks, (including Ethernet and WiFi) broadcast frames
- In regular mode, each network interface gets the frames intended for it
- Traffic sniffing can be accomplished by configuring the network interface to read all frames

Packet Sniffers

- Packet sniffers “read” information traversing a network
  - Packet sniffers intercept network packets, possibly using ARP cache poisoning
  - Can be used as legitimate tools to analyze a network
    - Monitor network usage
    - Filter network traffic
    - Analyze network problems
  - Can also be used maliciously
    - Steal information (i.e. passwords, conversations, etc.)
    - Analyze network information to prepare an attack
- Packet sniffers can be either software or hardware based
  - Sniffers are dependent on network setup

Packet Sniffers

- What can we get from packet sniffers?
  - Packet header
  - Payload data
  - Unencrypted sensitive data
  - Protocols in use
- Tools?
  - Wireshark, tcpdump, etc.

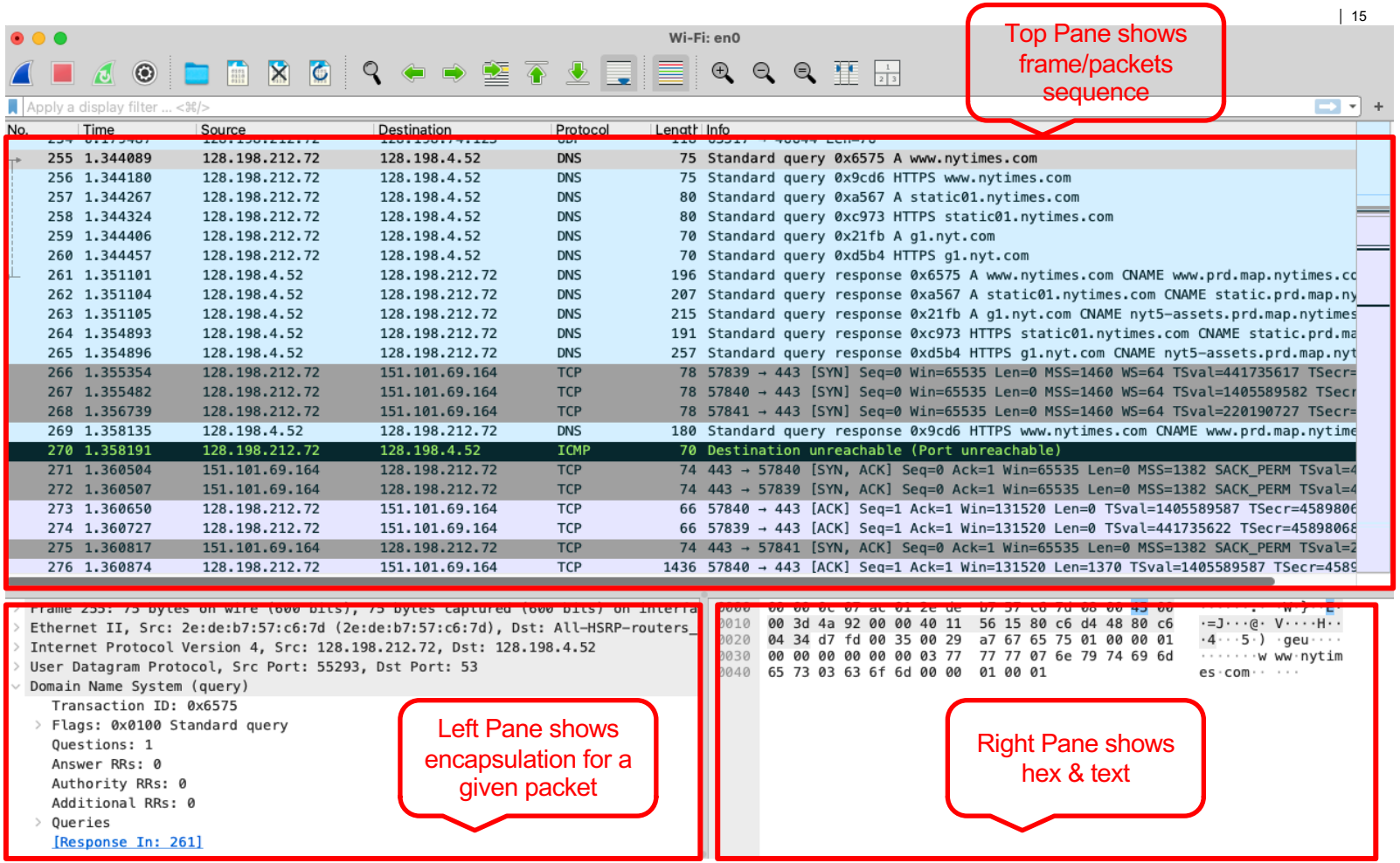
Detecting Sniffers

- Sniffers are almost always passive
  - They simply collect data
  - They do not attempt “entry” to “steal” data
- This can make them extremely hard to detect
- To reduce the impact of packet sniffing, encryption mechanisms should be utilized in higher-level protocols to prevent attackers from recovering sensitive data

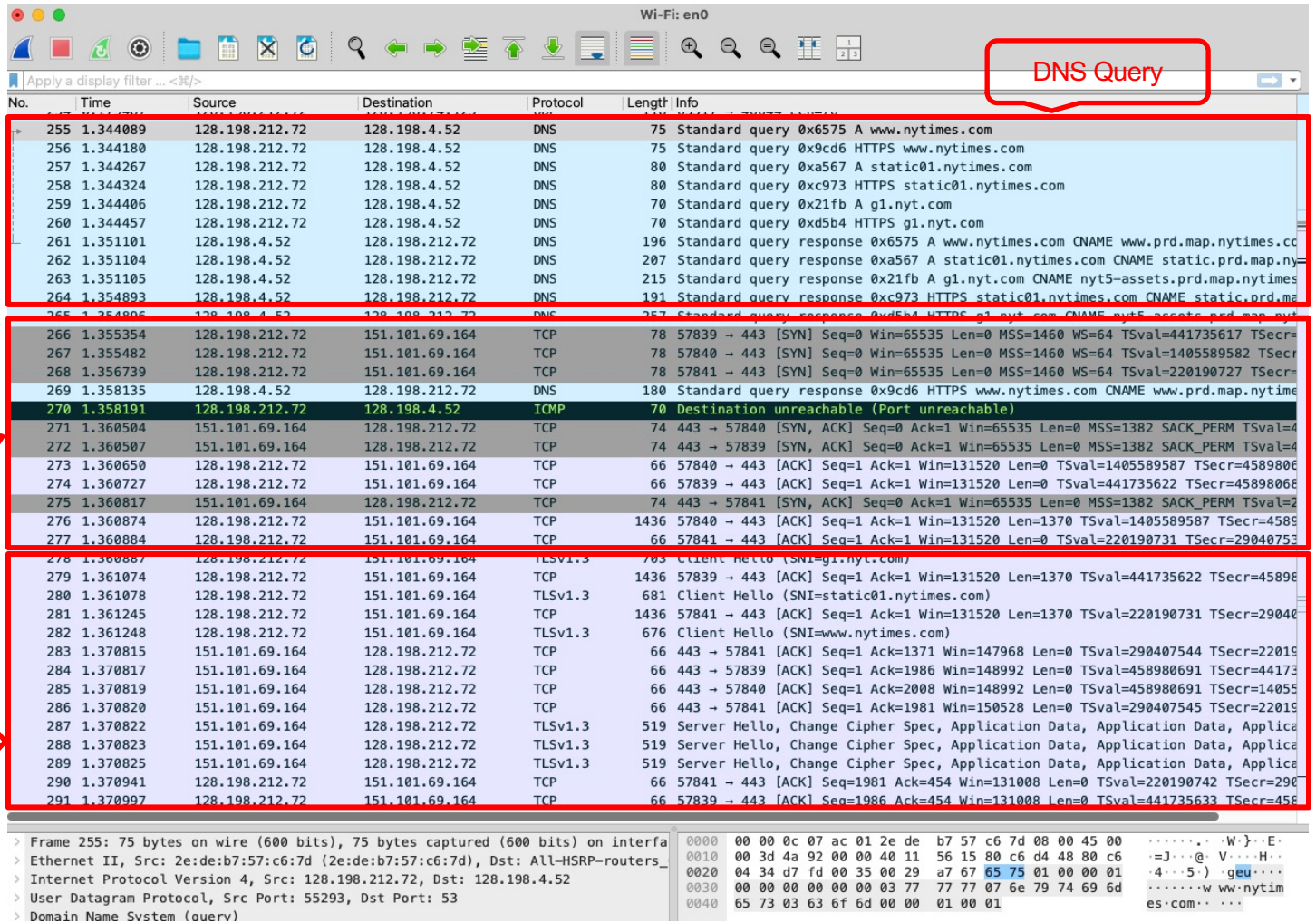
Network Analyzer -- Wireshark

- User clicks on http://www.nytimes.com/
- Network analyzer captures all frames observed by its NIC
- Sequence of frames and contents of frame can be examined in detail down to individual bytes

Wireshark Windows



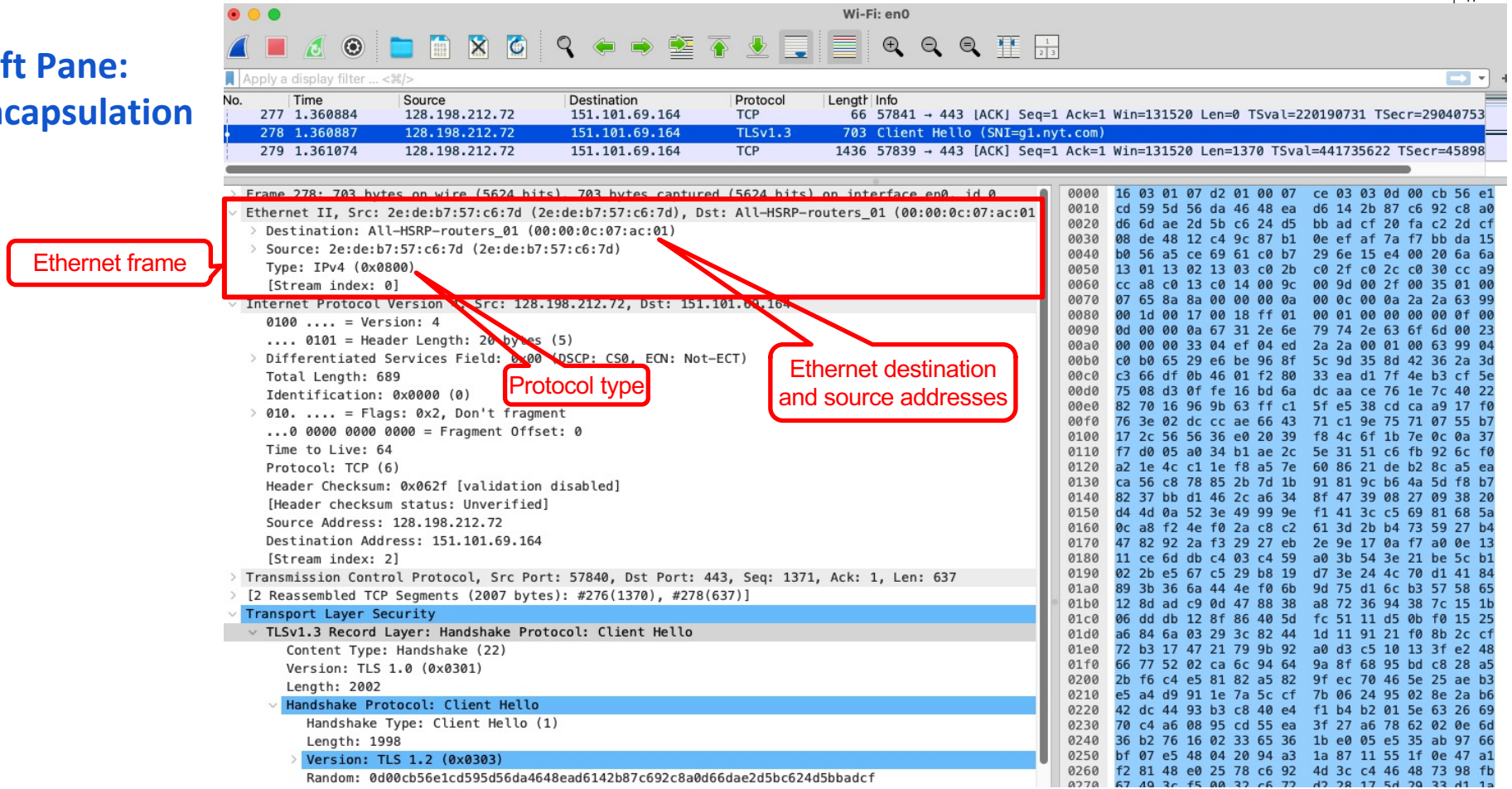
Top pane: Frame Sequence



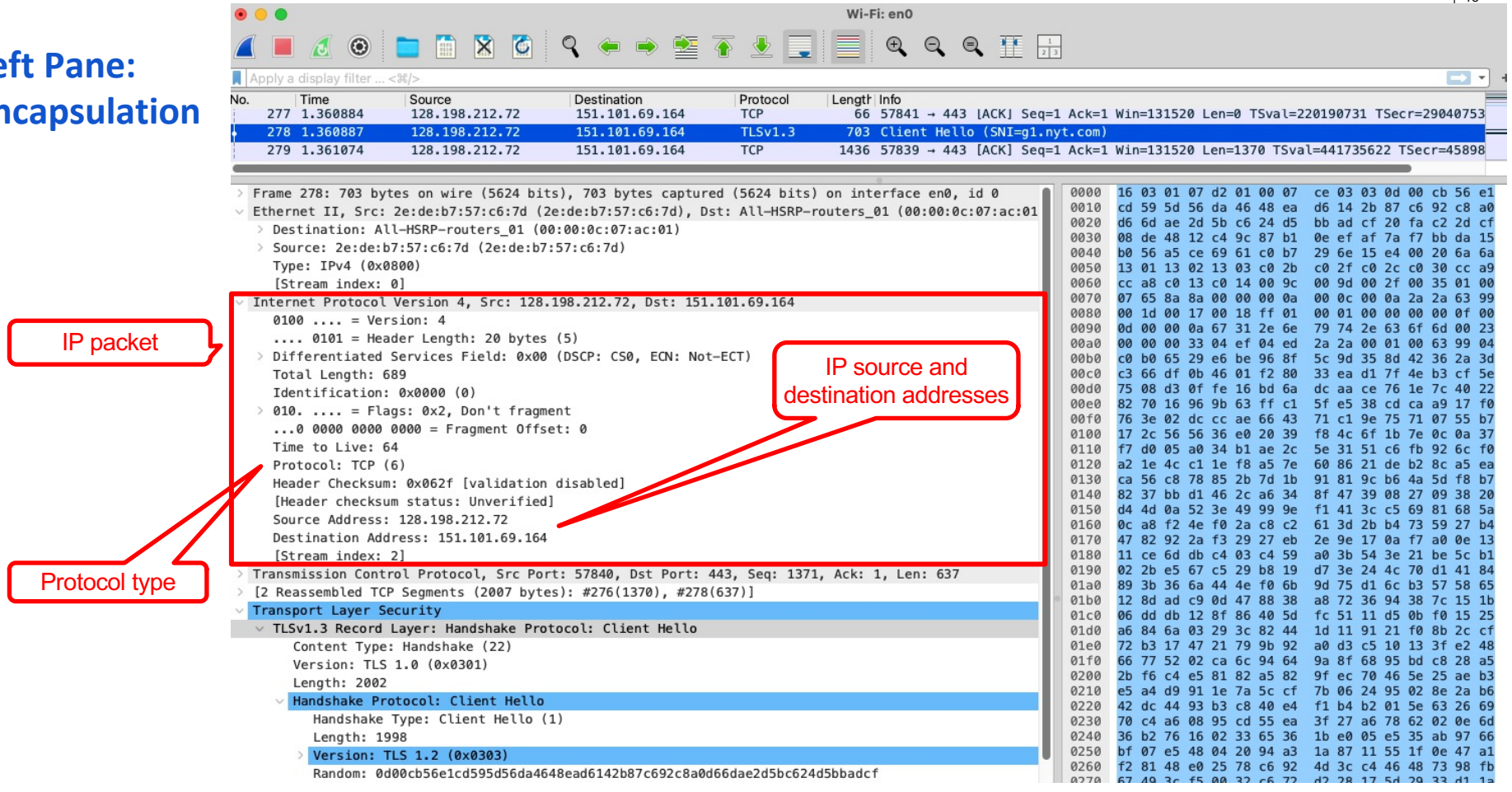
TCP connection establishment: SYN, SYN-ACK, ACK

TLS handshake: Client hello, server hello, key exchange and final handshake (Required when using HTTPS)

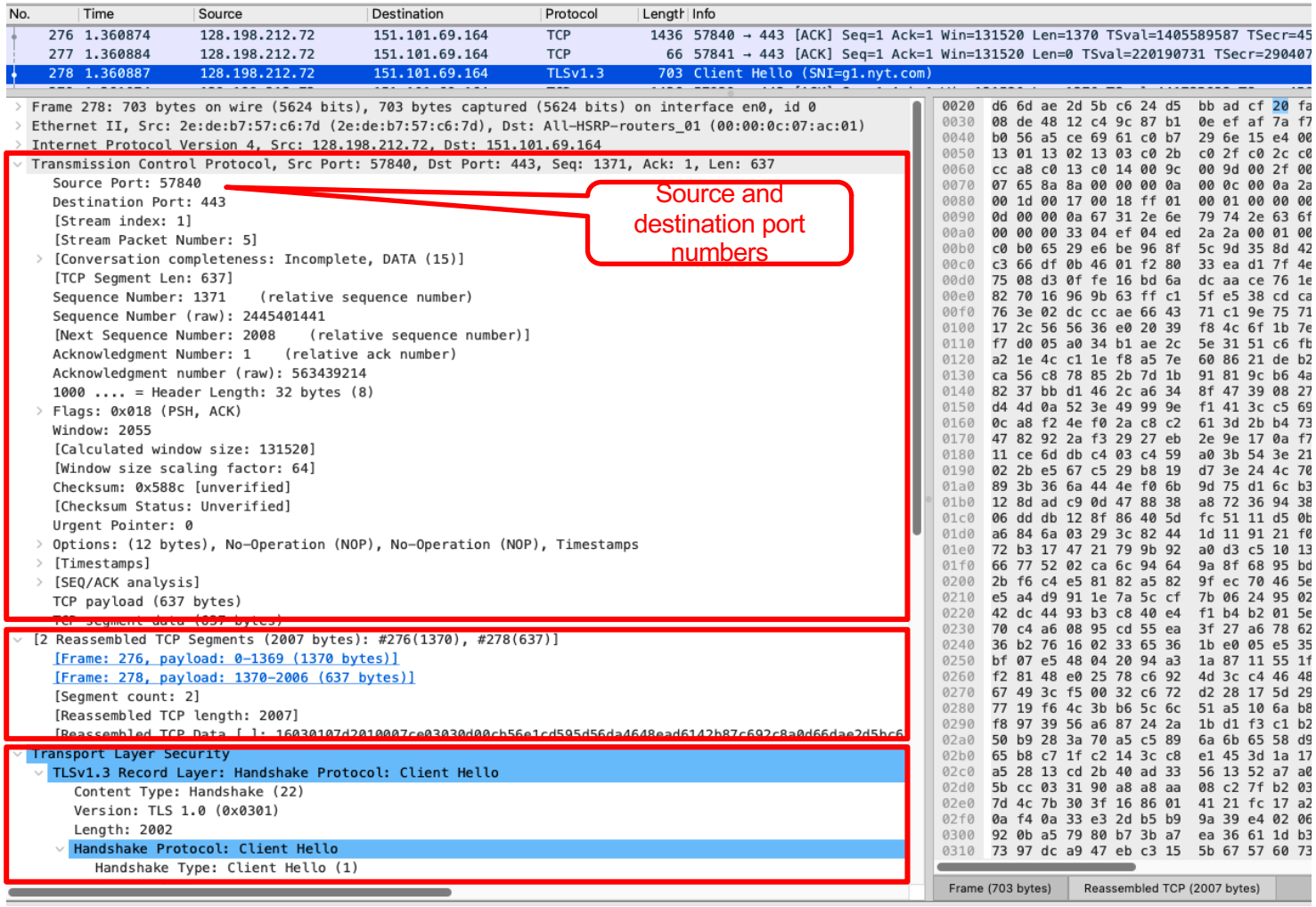
Left Pane: Encapsulation



Left Pane: Encapsulation



Left Pane: Encapsulation



Source and destination port numbers

TCP segment

Reassemble

TLS handshake for HTTPS

Network Attacks

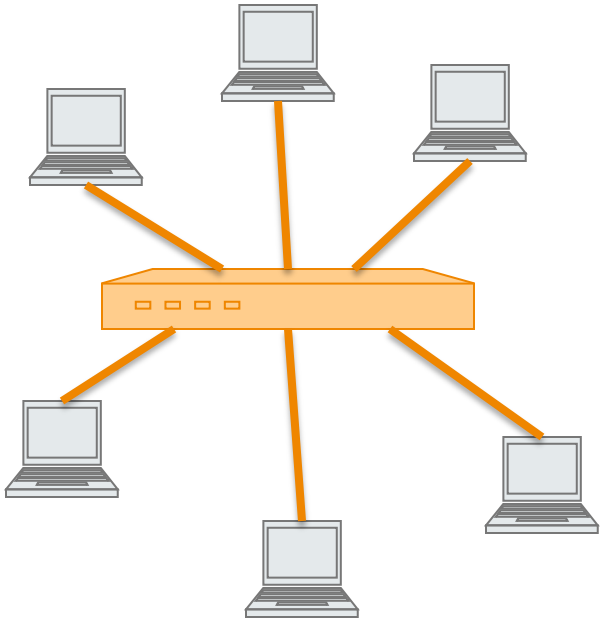
- MAC Spoofing, ARP Spoofing
- IP Spoofing
- Denial of Service
- DNS Cache Poisoning

MAC Addresses

- Most network interfaces come with a predefined MAC (Media Access Control) address
- A MAC address is a 48-bit number usually represented in hex
  - E.g., 00-1A-92-D4-BF-86
- The first three octets of any MAC address are IEEE-assigned Organizationally Unique Identifiers
  - E.g., Cisco 00-1A-A1, D-Link 00-1B-11, ASUSTek 00-1A-92
- The next three can be assigned by organizations as they please, with uniqueness being the only constraint
  - Organizations can utilize MAC addresses to identify computers on their network
  - MAC address can be reconfigured by network interface driver software

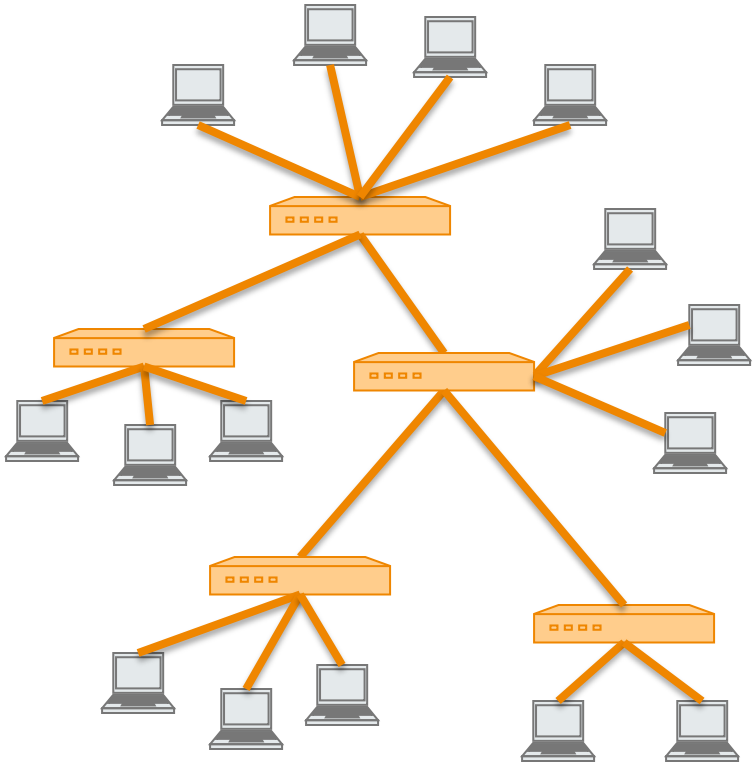
Switch

- A switch is a common network device
  - Operates at the link layer
  - Has multiple ports, each connected to a computer
  - Connects computers in an organization's internal Lan (local area network)
- Operation of a switch
  - Learn the MAC address of each computer connected to it
  - Forward frames only to the destination computer



Combining Switches

- Switches can be arranged into a tree
- Each port learns the MAC addresses of the machines in the segment (subtree) connected to it
- Fragments to unknown MAC addresses are broadcast
- Frames to MAC addresses in the same segment as the sender are ignored



MAC Spoofing

- A switch can be configured to provide service only to machines with specific MAC addresses
- Allowed MAC addresses need to be registered with a network administrator
- A MAC spoofing attack impersonates another machine
  - Find out MAC address of target machine
  - Reconfigure MAC address of rogue machine
  - Turn off or unplug target machine
- Countermeasures
  - Block port of switch when machine is turned off or unplugged
  - Disable duplicate MAC addresses

Viewing and Changing MAC Addresses

- Viewing the MAC addresses of the interfaces of a machine
  - Linux: ifconfig
  - Windows: ipconfig /all
- Changing a MAC address in Linux
  - Stop the networking service: /etc/init.d/network stop
  - Change the MAC address: ifconfig eth0 hw ether <MAC-address>
  - Start the networking service: /etc/init.d/network start
- Changing a MAC address in Windows
  - Open the Network Connections applet
  - Access the properties for the network interface
  - Click “Configure …”
  - In the advanced tab, change the network address to the desired value
- Changing a MAC address requires administrator privileges

ARP

- The address resolution protocol (ARP) is a link-layer protocol that connects the network layer to the link 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>
- Then the machine with <IP address1> responds the requestor with an ARP reply as

  <IP address1> is <MAC address>
- The Linux and Windows command arp - a displays the ARP table

```
Internet Address      Physical Address      Type
128.148.31.1          00-00-0c-07-ac-00     dynamic
128.148.31.15         00-0c-76-b2-d7-1d     dynamic
128.148.31.71         00-0c-76-b2-d0-d2     dynamic
128.148.31.75         00-0c-76-b2-d7-1d     dynamic
128.148.31.102        00-22-0c-a3-e4-00     dynamic
128.148.31.137        00-1d-92-b6-f1-a9     dynamic
```

ARP Spoofing

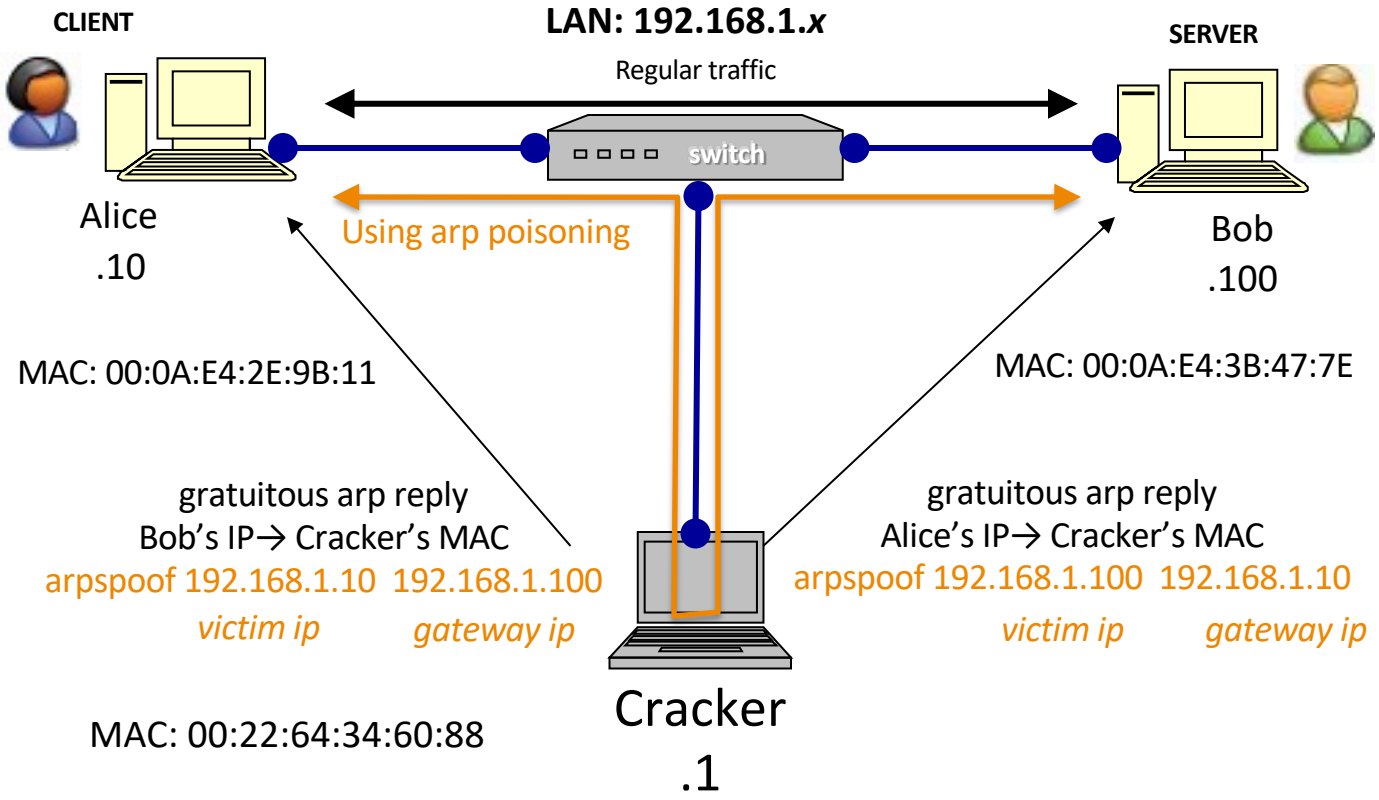
- The ARP table is updated whenever an ARP response is received
- Requests are not tracked
- ARP announcements are not authenticated
- Machines trust each other

- A rogue machine can spoof other machines

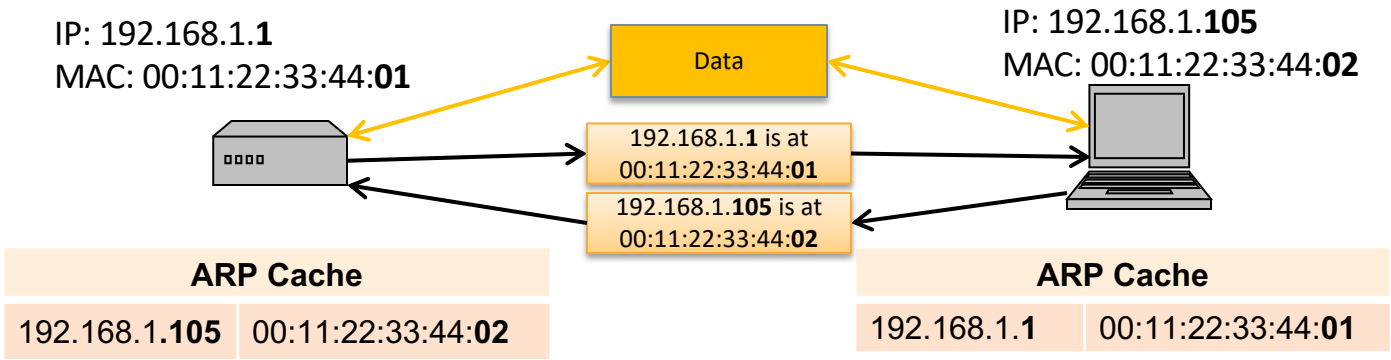
ARP Spoofing (ARP Poisoning)

- 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!

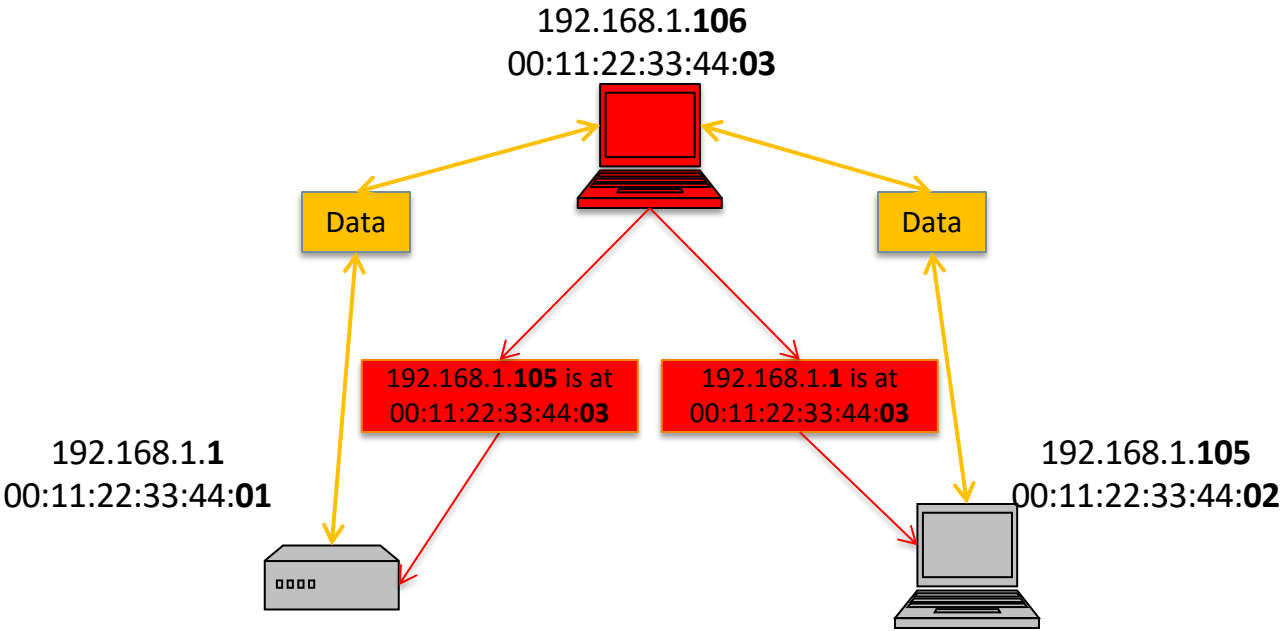
Example 1: ARP Spoofing



ARP Caches


IP: 192.168.1.1
MAC: 00:11:22:33:44:01
Data
IP: 192.168.1.105
MAC: 00:11:22:33:44:02
192.168.1.1 is at 00:11:22:33:44:01
192.168.1.105 is at 00:11:22:33:44:02
ARP Cache
192.168.1.105
00:11:22:33:44:02
ARP Cache
192.168.1.1
00:11:22:33:44:01

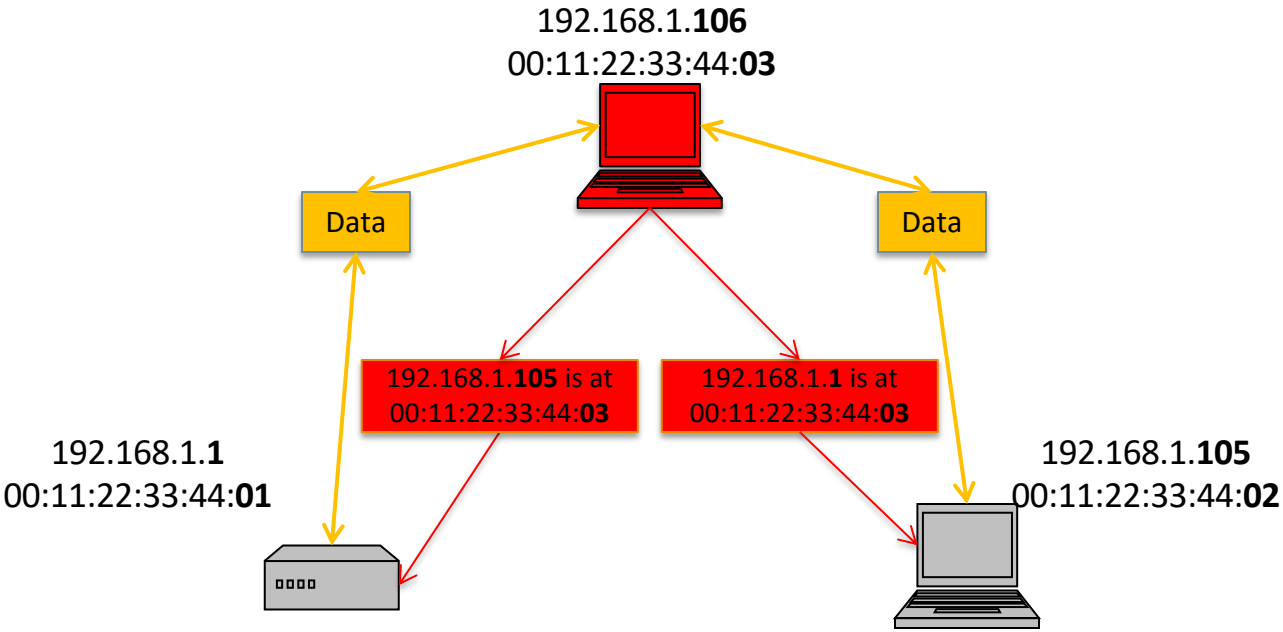
Example 2: Poisoned ARP Caches



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

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

Example 2: Poisoned ARP Caches



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

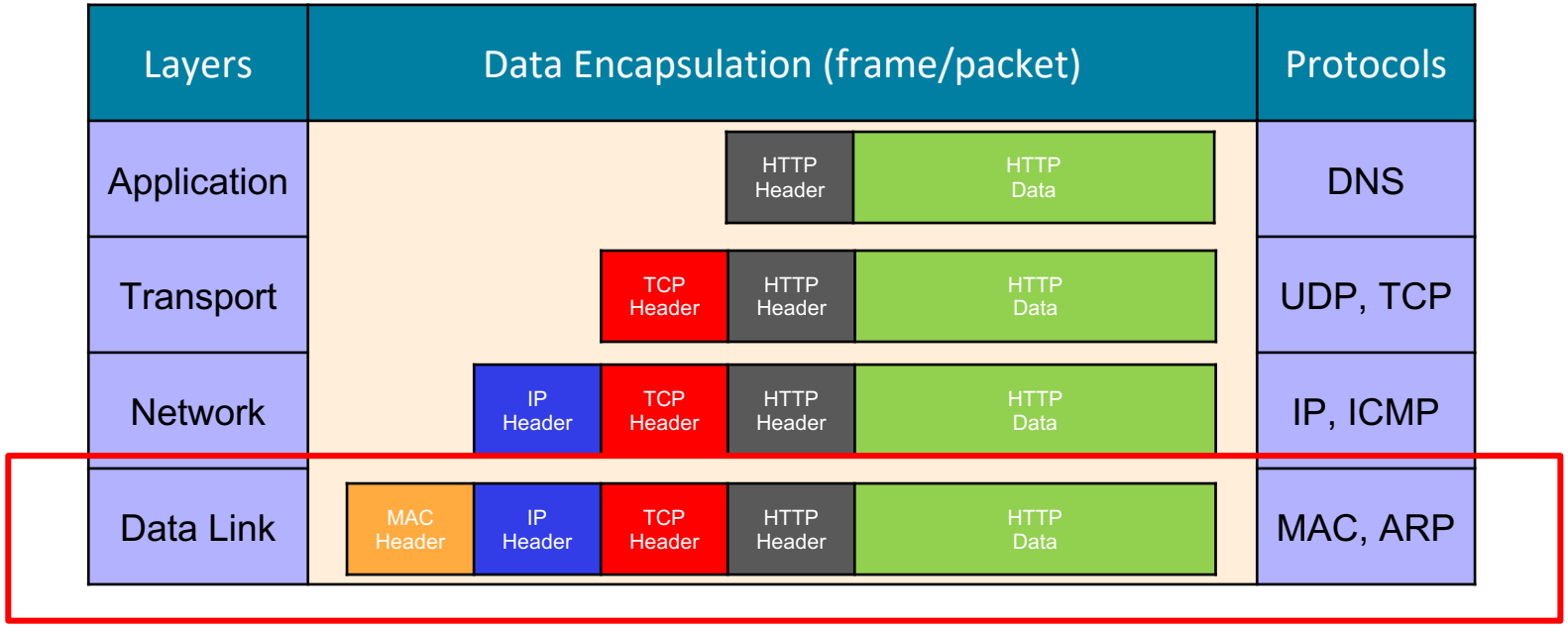
Poisoned ARP Cache	
192.168.1.**1**	00:11:22:33:44:**03**

ARP Spoofing (or ARP poisoning)

- Send fake ARP messages to an Ethernet LAN (no authentication)
  - this causes other machines to associate IP addresses with attacker's MAC

- Defenses
  - static ARP table
  - DHCP snooping (access control based on IP, MAC, and port)
  - detection: Arpwatch, reverse ARP

Recall: Internet Communication



MAC spoofing, ARP spoofing

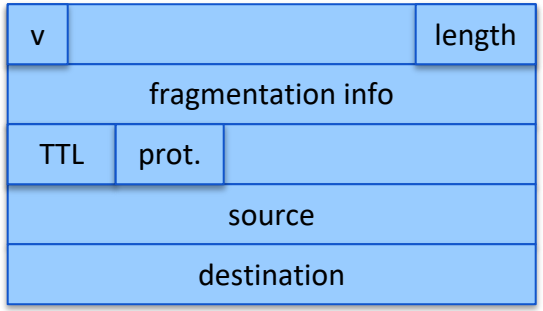
Network Attacks

- MAC Spoofing, ARP Spoofing
- IP spoofing
- Denial of service
- DNS cache poisoning

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

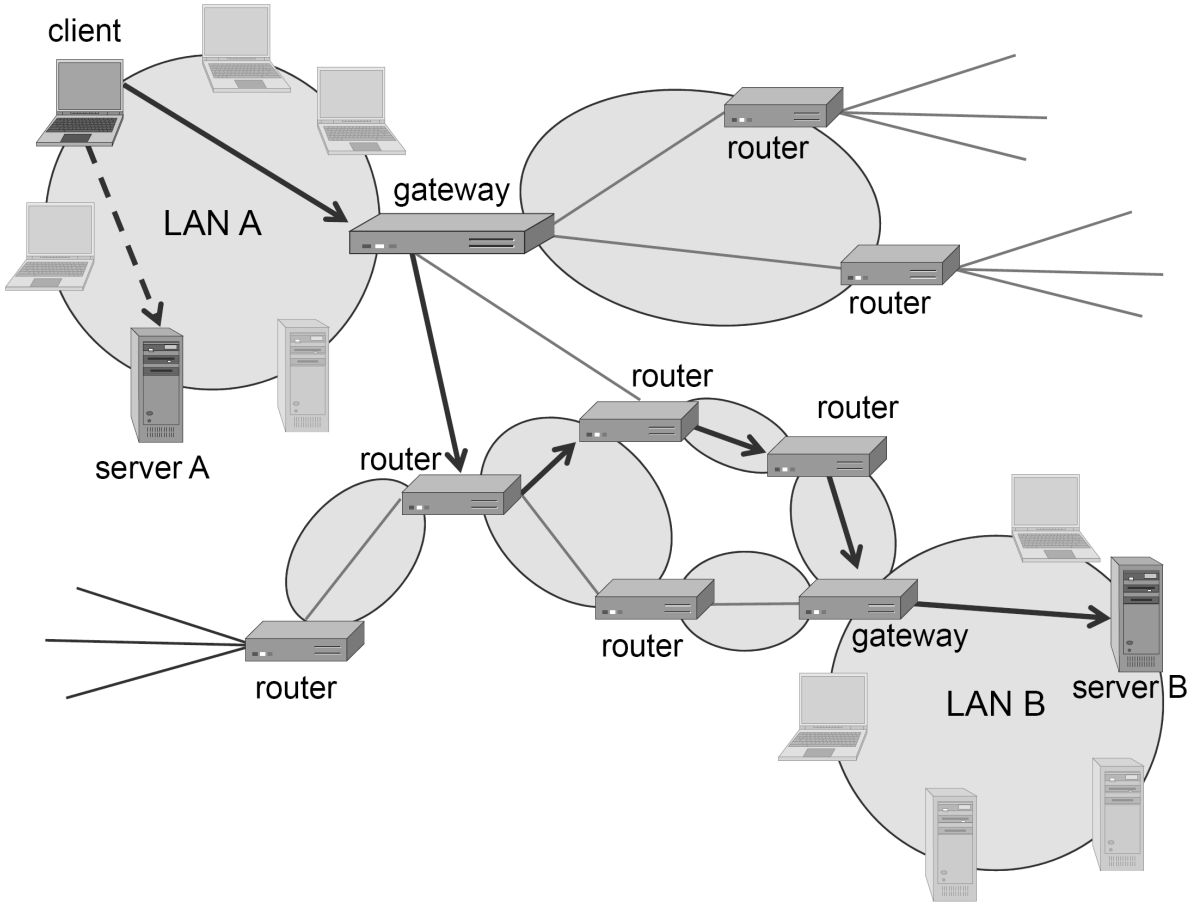
- IP header includes
  - Source address
  - Destination address
  - Packet length (up to 64KB)
  - Time to live (up to 255)
    - Hop limit
  - IP protocol version
  - Fragmentation information
  - Transport layer protocol information (e.g., TCP)



IP Routing

- A router bridges two or more networks
  - Operates at the network layer
    - Drop: if the packet is expired
    - Deliver: on one of the LANs connected
    - Forward: on different LANs
  - 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 on the Internet

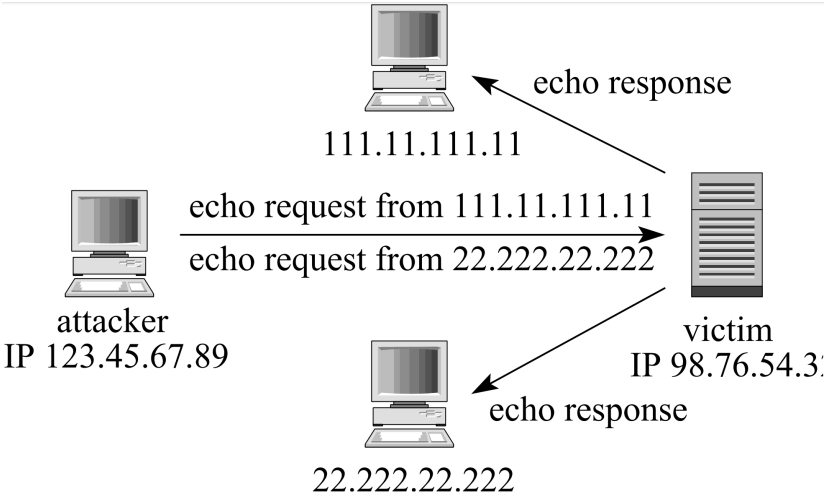


IP Vulnerabilities

- 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 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

IP Spoofing

- With sufficient privileges to a machine, the source address in IP packets can be set to anything
- The source address is set to a randomly chosen address
- Replies from the victim machine are scattered across the internet



IP Spoofing Prevention

- Ingress and Egress Filtering
  - Ingress Filtering: Configure routers or firewalls to block packets with source IP addresses that are outside the range of valid IP addresses for incoming traffic.
  - Egress Filtering: Prevent outgoing packets from leaving the network with a spoofed source IP address.

- Reverse Path Forwarding (uRPF)
  - Enables routers to verify the reachability of the source IP address by checking the routing table.

Network Attacks

- MAC Spoofing, ARP Spoofing
- IP spoofing
- Denial of service
- DNS cache poisoning

Denial of Service (DoS) Attacks

- Denial of service attacks target at denying availability of some service or resource, including
  - Network bandwidth
    - relates to the capacity of the network links connecting a server to the Internet
    - for most organizations, this is their connection to their Internet Service Provider (ISP)
  - System resources
    - aims to overload or crash the network handling software
  - Application resources
    - typically involves a number of valid requests, each of which consumes significant resources and thus limiting the ability of the server to respond to requests from other users

DoS Attacks

- Types of DoS attacks

	stopping services	exhausting resources
local	process crashing process killing system reconfiguration	spawning processes to fill process table filling up file system saturating bandwidth
remote	malformed packets to crash buggy services	packet floods

DoS Attacks

- Basic form of DoS
  - Attacker sends a large number of packets through a link or to a particular service
  - The goal is to saturate the network or overload the server
  - Most requests from legitimate users will be dropped

- Example:
  - Flooding attack: ICMP Flood, SYN Flood, etc.
  - Distributed DoS (DDoS)
  - Other DoS Attacks

DoS Attacks – Flooding Attack

- Flooding attack: attackers send **a very high volume of traffic** to a system so that it cannot examine and allow permitted network traffic

- Flooding attacks in general can use any type of packets
  - e.g., ICMP flood, TCP SYN flood, UDP flood
- In any attack with spoofed addresses, it is hard to find attacker

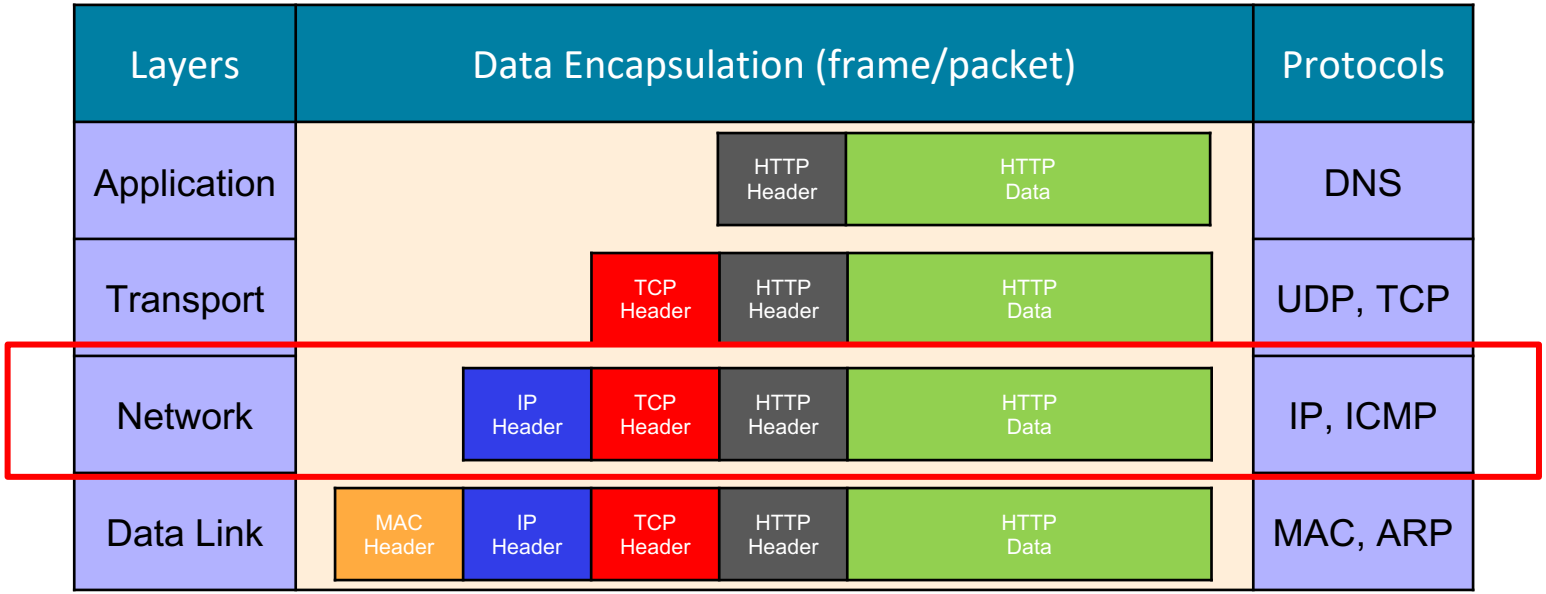
Internet Routes

- Internet Control Message Protocol (ICMP)
  - Used for network testing and debugging
  - Simple messages encapsulated in single IP packets
    - Echo request (sender); echo response (reciever)
  - 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

DoS Attacks – ICMP Flood

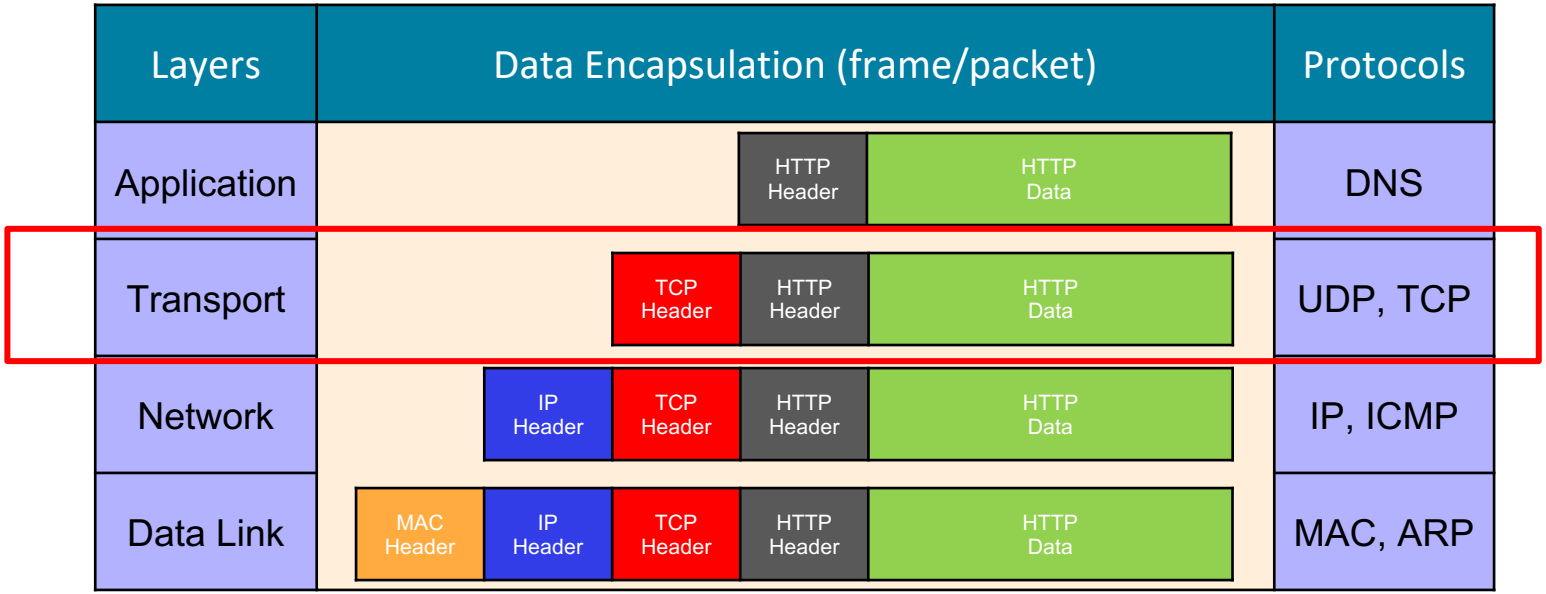
- Ping of death
  - ICMP specifies messages must fit a single IP packet (64KB)
  - Send a ping packet that exceeds maximum size using IP fragmentation
  - Reassembled packet caused several operating systems to crash due to a buffer overflow
- Ping Flood
  - Send a massive amounts of echo requests to a single victim server

So far …



IP spoofing, ICMP flood

Next


Layers
Data Encapsulation (frame/packet)
Protocols
Application
HTTP Header
HTTP Data
DNS
Transport
TCP Header
HTTP Header
HTTP Data
UDP, TCP
Network
IP Header
TCP Header
HTTP Header
HTTP Data
IP, ICMP
Data Link
MAC Header
IP Header
TCP Header
HTTP Header
HTTP Data
MAC, ARP

UDP and TCP

- UDP: User Datagram Protocol
  - transport protocol with minimal guarantees
    - no acknowledgment, no flow control, no message continuation
  - traffic is separated by port number
- TCP: Transmission Control Protocol
  - connection-oriented transport protocol
  - partitions data into packets and reassembles them in correct order at the destination
  - transmission is reliable
    - packets are acknowledged and retransmitted if necessary
  - port numbers are used for different services as well

UDP

- UDP is a stateless, unreliable datagram protocol built on top of IP
- It does not provide delivery guarantees, or acknowledgments, but is significantly faster
- Distinguish data for multiple concurrent applications on a single host via ports

- 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.

DoS Attack – UDP Flood

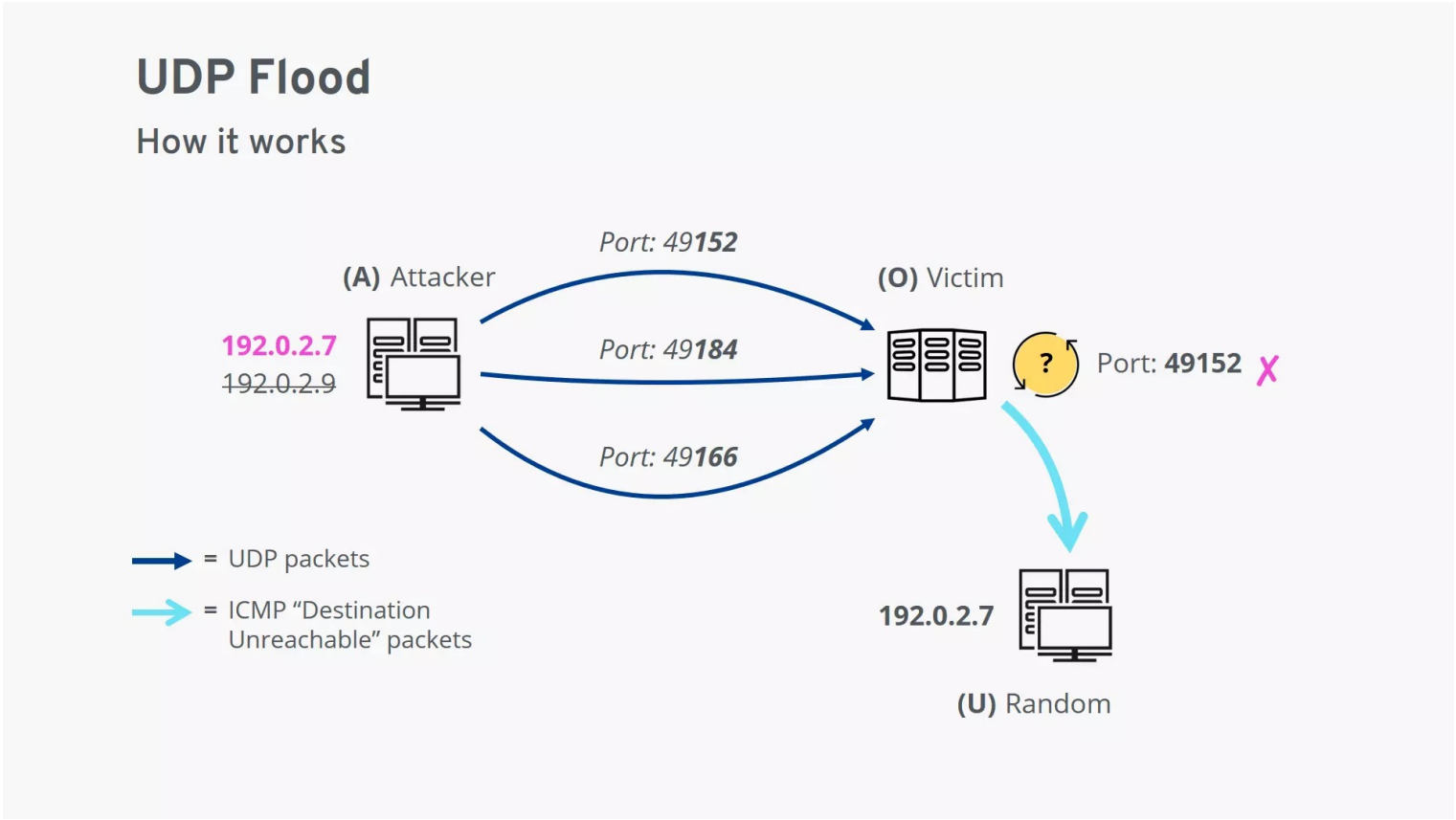


Figure from: https://www.ionos.com/digitalguide/server/security/udp-flood/

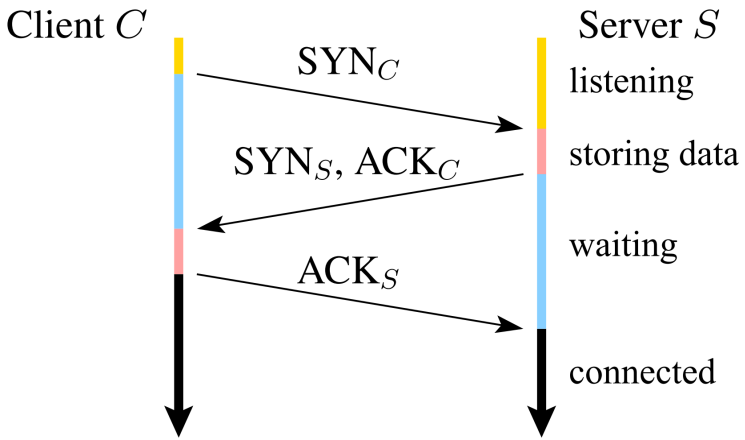
TCP

- 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

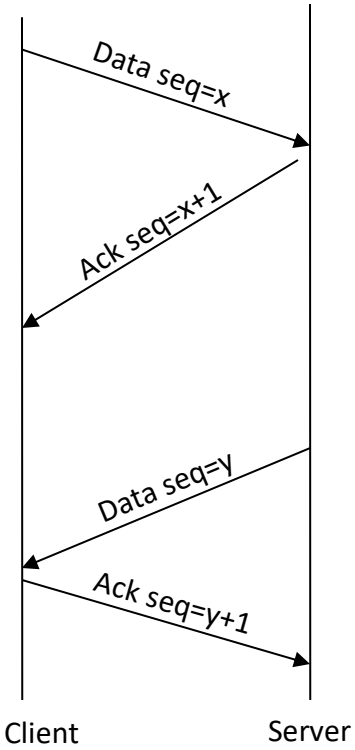
- TCP takes a stream of 8-bit byte data, packages it into appropriately sized segment and calls on IP to transmit these packets
- Delivery order is maintained by marking each packet with a sequence number
- Every time TCP receives a packet, it sends out an ACK to indicate successful receipt of the packet.
- TCP generally checks data transmitted by comparing a checksum of the data with a checksum encoded in the packet

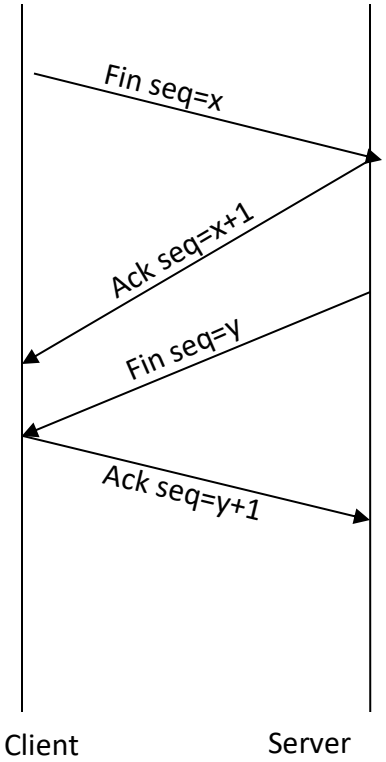
Establishing TCP Connections

- 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 (synchronization) packet
  - The server responds by sending a SYN/ACK packet, indicating an acknowledgment for the connection
  - The client responds by sending a **concluding** ACK to the server thus establishing connection



TCP Data Transfer and Teardown



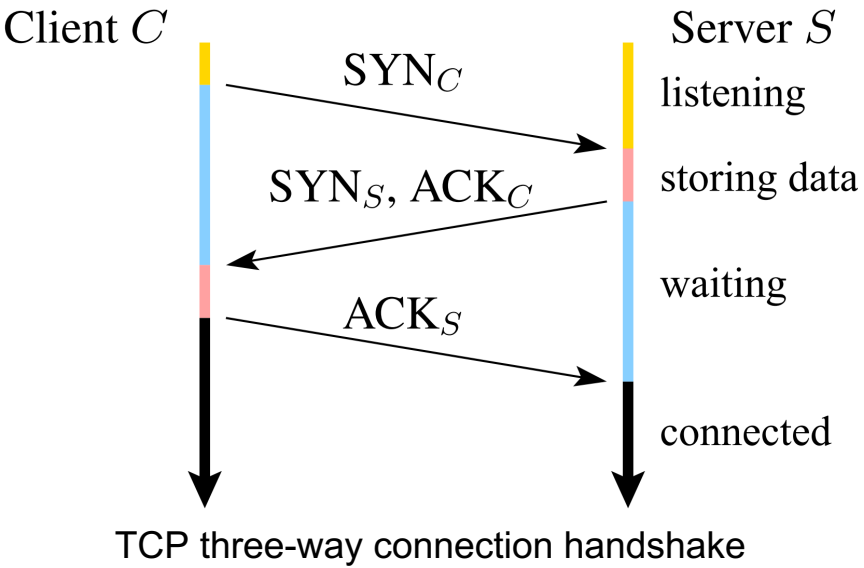


Ports

- Identify multiple concurrent applications on the same server
- 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
      - 20/21: file transfer protocol (FTP)
      - 22: SSH secure login
      - 23: telnet remote login
      - 80: http used in WWW
      - …
    - 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

DoS Attacks – SYN Flood

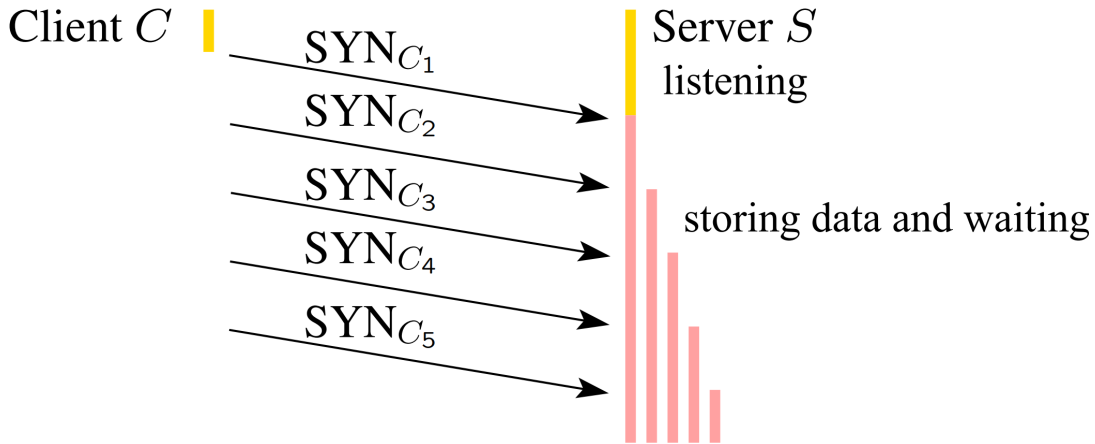
- TCP SYN flooding
  - uses the fact that a machine has a limit on the number of open connections
  - allows attacker to deny availability with much less traffic



TCP three-way connection handshake

DoS Attacks – SYN Flood

- TCP SYN flooding attack exploits the fact that server waits for ACKs
  - attacker sends many SYN requests with spoofed source addresses
  - victim allocates resources for each request
    - connection requests exist until timeout
    - there is a fixed bound on half-open connections



DoS Attacks – SYN Flood

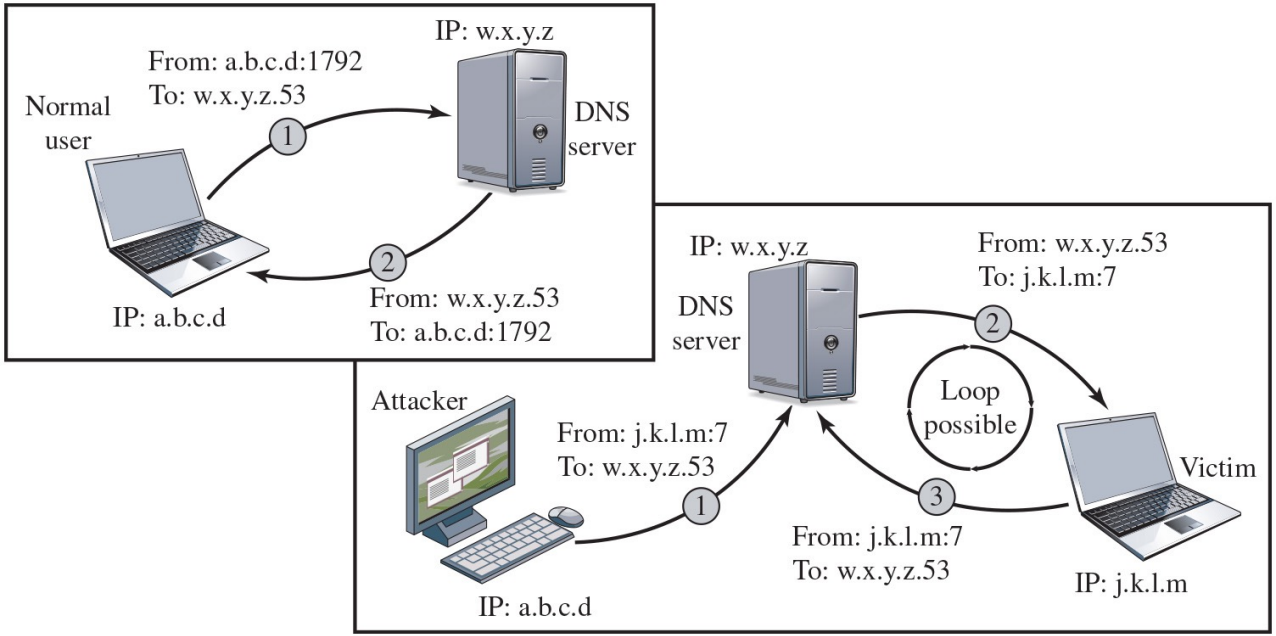
- TCP SYN flooding attack (cont.)
  - resources exhausted ⇒ legitimate requests rejected
  - the attack relies on the fact that many SYN-ACK packets will be unanswered
    - an existing host replies to a SYN-ACK packet
    - many IP addresses are not in use
  - the attacker needs to keep sending new SYN packets to keep the table full

Other DoS Attacks

- Other variants of DoS attacks that use additional machines
  - Reflection
    - Attacker sends packets to a known service on the intermediary with a spoofed source address of the actual target system
    - When intermediary responds, the response is sent to the target
    - "Reflects" the attack off the intermediary (reflector)
    - Goal is to generate enough volumes of packets to flood the link to the target system without alerting the intermediary
    - The basic defense against these attacks is blocking spoofed-source packets

Other DoS Attacks

- Other variants of DoS attacks that use additional machines
  - Reflection example

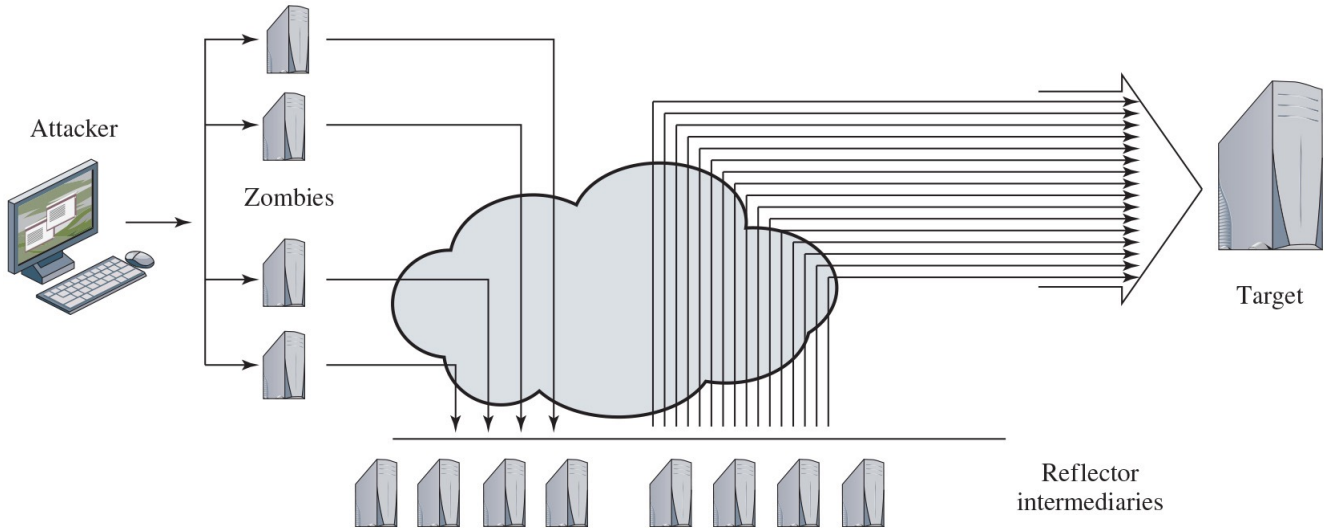


Other DoS Attacks

- Other variants of DoS attacks that use additional machines
  - Amplification
    - Also sends packets with spoofed addresses to intermediaries
    - Now one original packet generates many response packets
    - Target is flooded with responses
    - Basic defense against this attack is to prevent the use of spoofed source addresses

Other DoS Attacks

- Other variants of DoS attacks that use additional machines
  - Amplification example



Use a misconfigured network to amplify traffic intended to overwhelm the bandwidth of a target

Other DoS Attacks

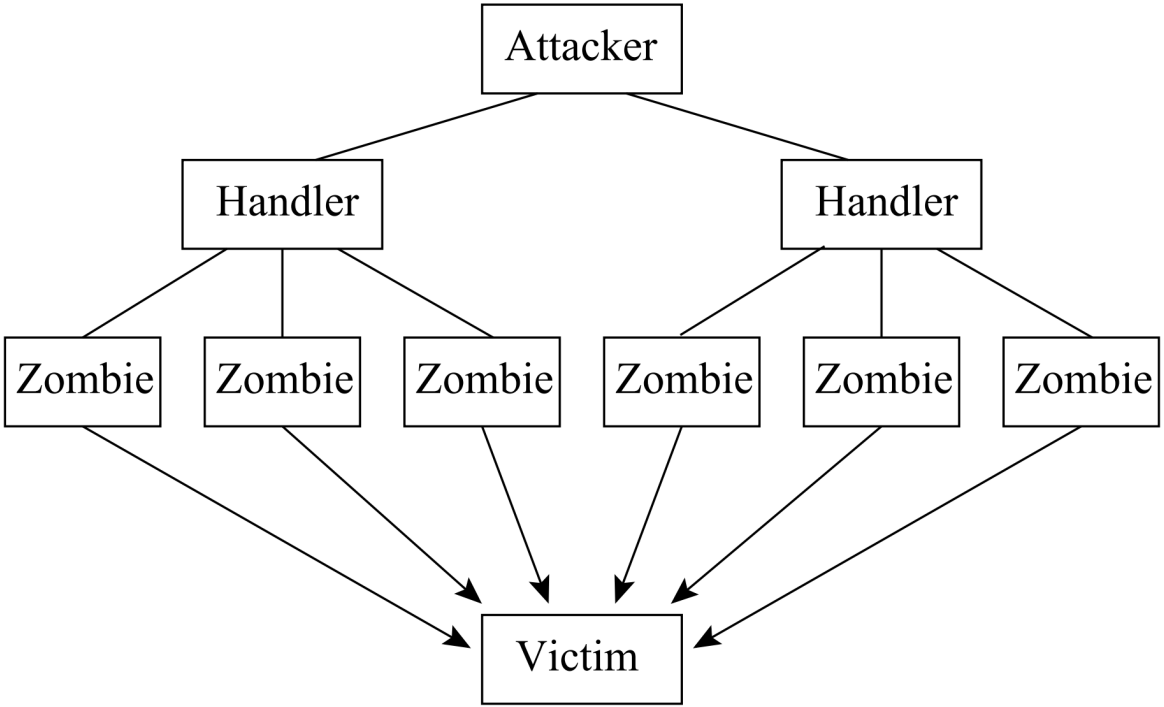
- Other variants of DoS attacks (cont.)
  - Pulsing zombie floods
    - each zombie is active briefly and then goes dormant
    - zombies take turns in attacking
    - this makes tracing difficult

DDoS Attacks

- In all of the above attacks, attacker needs to have substantial resources
  - thus attacks are more effective if carried out from many sources
  - they are called distributed DoS (DDoS) attacks
- DDoS attacks often use compromised computers (zombies)
  - attacker compromised machines and builds a botnet
  - attacker instructs the bots to attack the target machine
  - all communication is often encrypted, can be authenticated
  - zombie machines flood the victim
  - spoofing IP addresses is not necessary since it is hard to trace the attacker from the zombie machines

DDoS Attacks

- DDoS attack illustrated



Defenses Against DoS Attacks

- A significant challenge in defending against DoS attacks is that spoofed addresses are used
- What can be done
  - Ingress filtering
    - basic recommendation to check that packets coming from a network have source address within the network's range
    - ISPs (Internet service providers) are best suited to perform such filtering
    - despite its simplicity and effectiveness, this recommendation is not implemented by many ISPs

Defenses Against DoS Attacks

- DoS defenses (cont.)
  - SYN cookies
    - this technique is used to defend against TCP SYN floods
    - after receiving a SYN, information about it is not stored the server
    - instead it is encoded in the SYN-ACK packet
    - upon receiving ACK, server can reconstruct all information
    - disadvantages: increased server computation
  - Blocking certain packets
    - many systems block ICMP echo requests from outside of network
    - often IP broadcasts are also blocked from outside

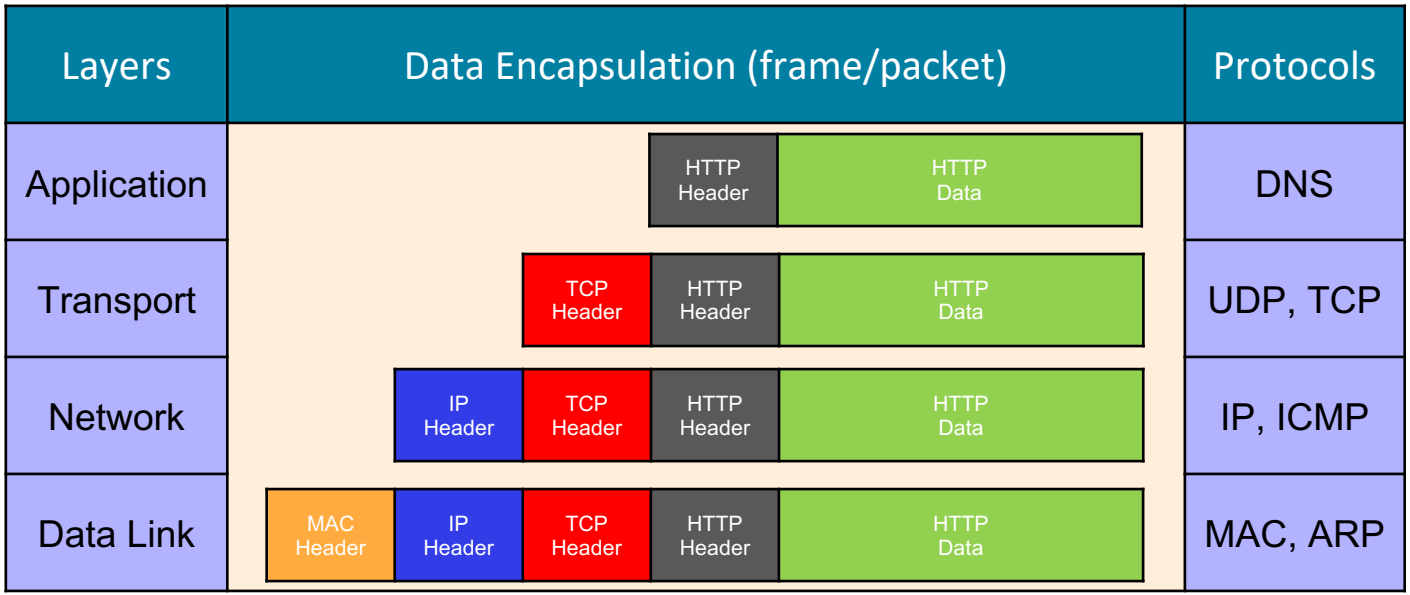
Defenses Against DoS Attacks

- DoS defenses (cont.)
  - Limiting packet rates
    - certain types of packets such as ICMP are rather rare in normal network operation
    - limiting their rate can help mitigate attacks
  - Packet marking
    - a router marks a small number of packets with its ID
    - for high volume traffic, packets will be marked by most servers on their path to the victim
    - path to the attacker can be reconstructed
    - effectiveness of this technique depends on its wide usage
  - General good security practices

Network Attacks

- MAC Spoofing, ARP Spoofing
- IP Spoofing
- Denial of Service
- DNS Cache Poisoning

So far …



?

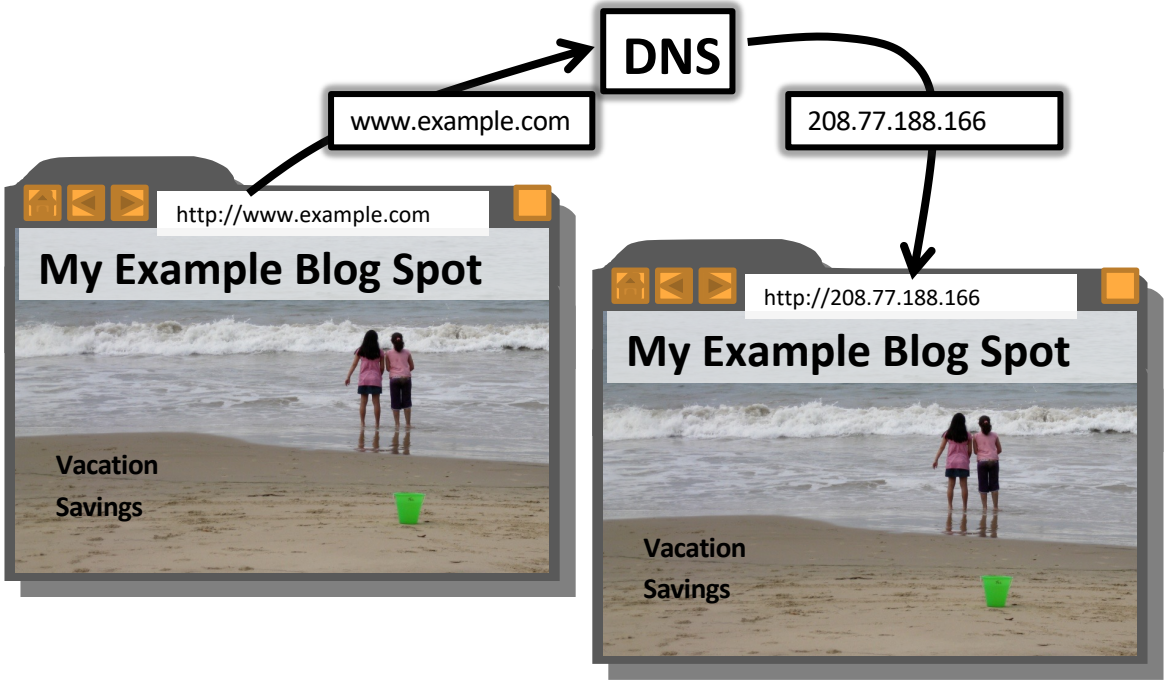
TCP flood

IP spoofing,
ICMP flood

MAC spoofing,
ARP spoofing

Domain Name System (DNS)

- The domain name system (DNS) is an application-layer protocol for mapping domain names to IP addresses



DNS

- DNS provides a distributed database over the internet that stores various resource records, including:
  - Address (A) record: IP address associated with a host name
  - Mail exchange (MX) record: mail server of a domain
  - Name server (NS) record: authoritative server for a domain

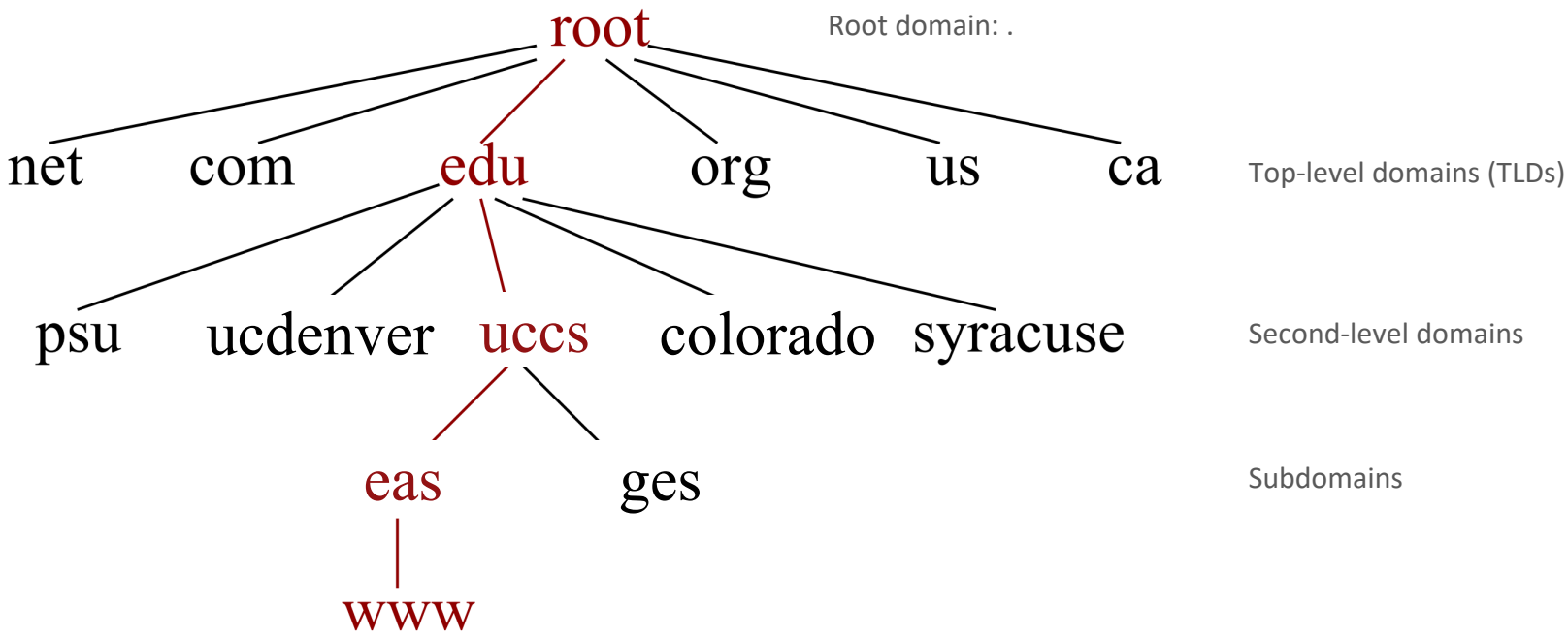
Resource records [edit]

Type	Type id. (decimal)	Defining RFC	Description	Function
A	1	RFC 1035[1]	Address record	Returns a 32-bit IPv4 address, most commonly used to map hostnames to an IP address of the host, but it is also used for DNSBLs, storing subnet masks in RFC 1101, etc.
AAAA	28	RFC 3596[2]	IPv6 address record	Returns a 128-bit IPv6 address, most commonly used to map hostnames to an IP address of the host.
AFSDB	18	RFC 1183	AFS database record	Location of database servers of an AFS cell. This record is commonly used by AFS clients to contact AFS cells outside their local domain. A subtype of this record is used by the obsolete DCE/DFS file system.
APL	42	RFC 3123	Address Prefix List	Specify lists of address ranges, e.g. in CIDR format, for various address families. Experimental.

List of DNS records from https://en.wikipedia.org/wiki/List_of_DNS_record_types

DNS

- DNS tree
  - the name space is hierarchical

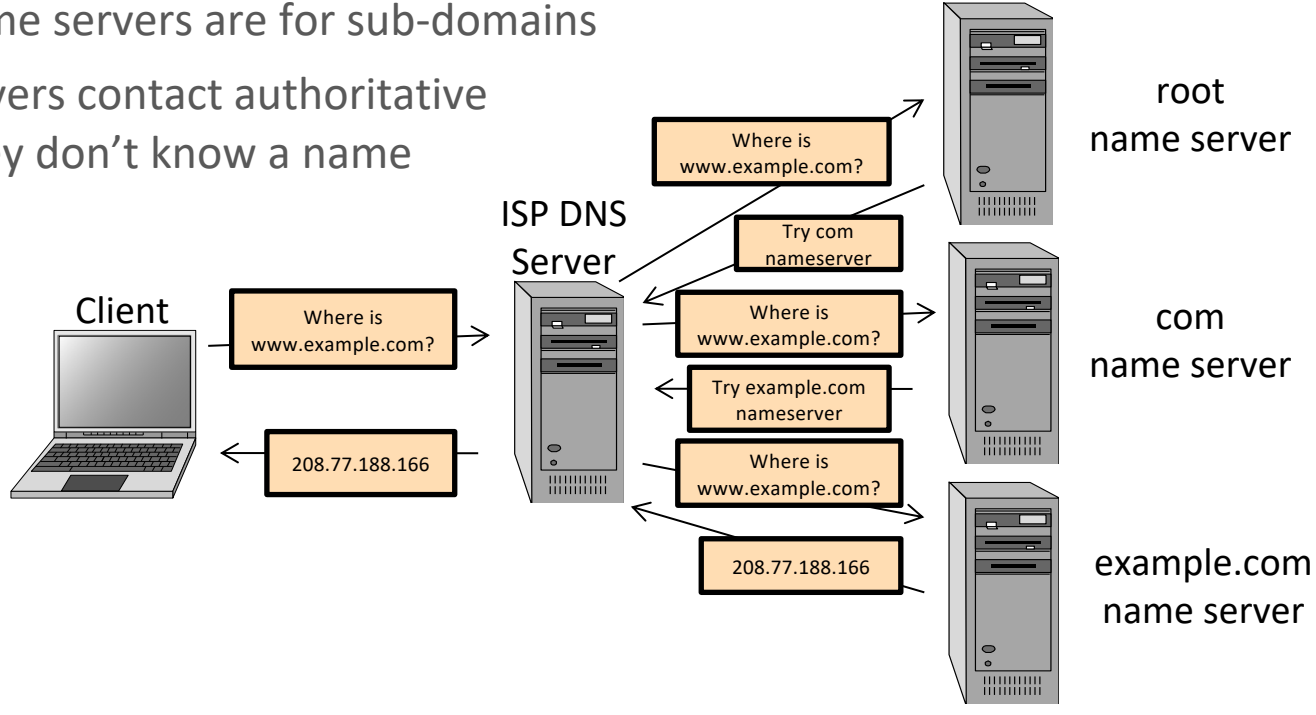


Top Level Domains

- Started in 1984
- Originally supposed to be named by function
    - .com for commercial websites, .mil for military
- Eventually agreed upon unrestricted top-level domain (TLD) for .com, .net, .org, .info
- In 1994 started allowing **country** TLDs such as .it, .us
- Tried to move back to **hierarchy** of purpose in 2000 with creation of .aero, .museum, etc.
- Two primary types of TLD:
    - Generic top-level domains: .com, .net, .edu, .org
    - Country-code top-level domains: .au(Australia), .cn(China),. It (Italy)

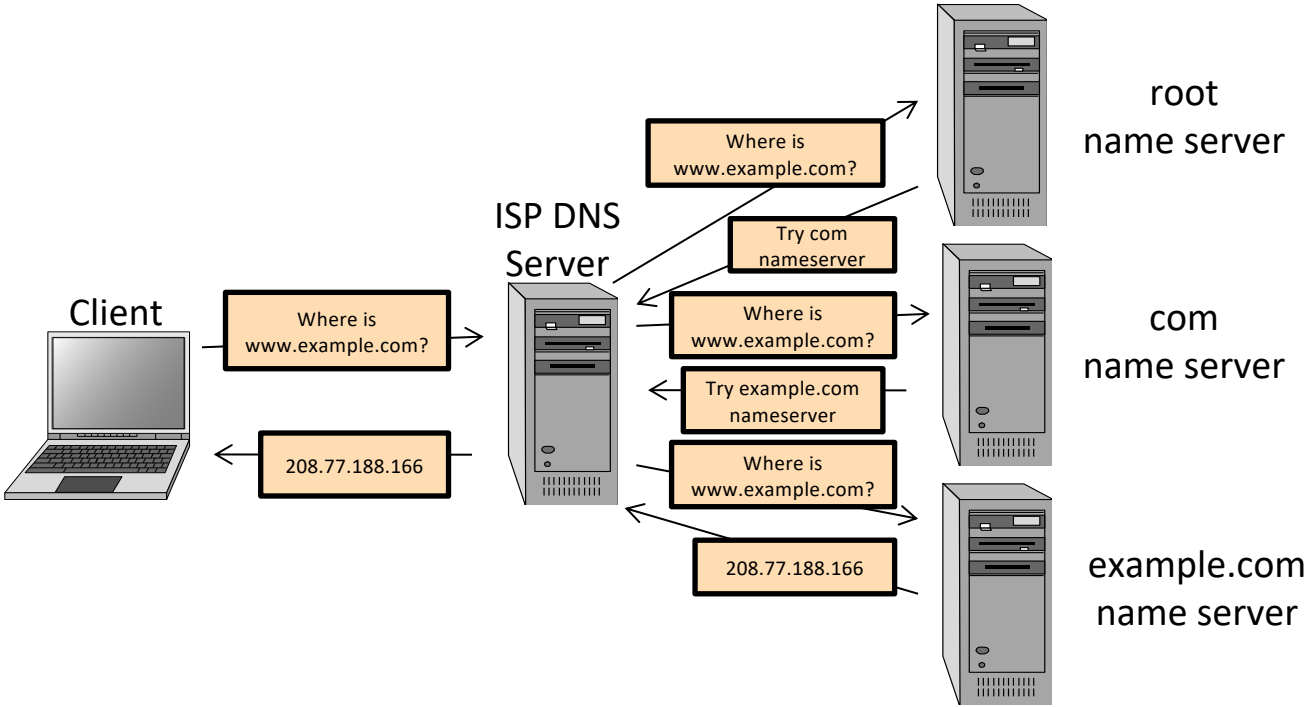
DNS

- Hierarchical service
  - root name servers are for top-level domains
  - authoritative name servers are for sub-domains
  - local name resolvers contact authoritative servers when they don't know a name

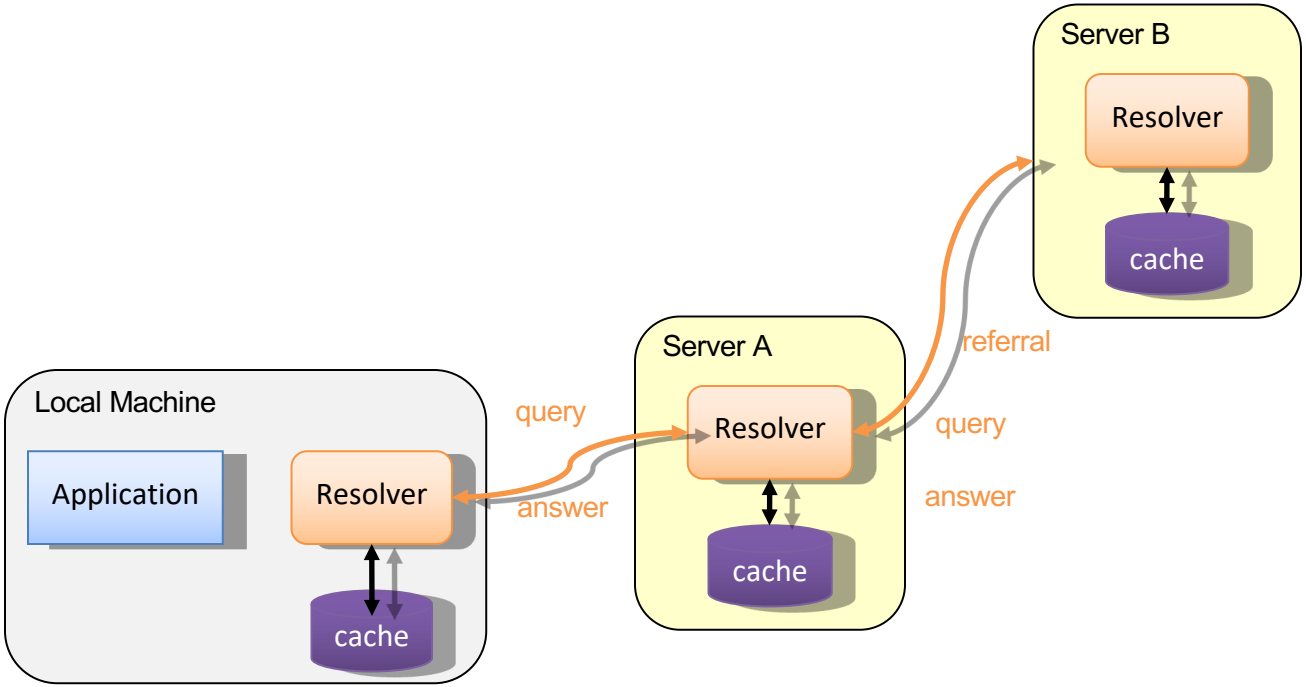


Name Resolution

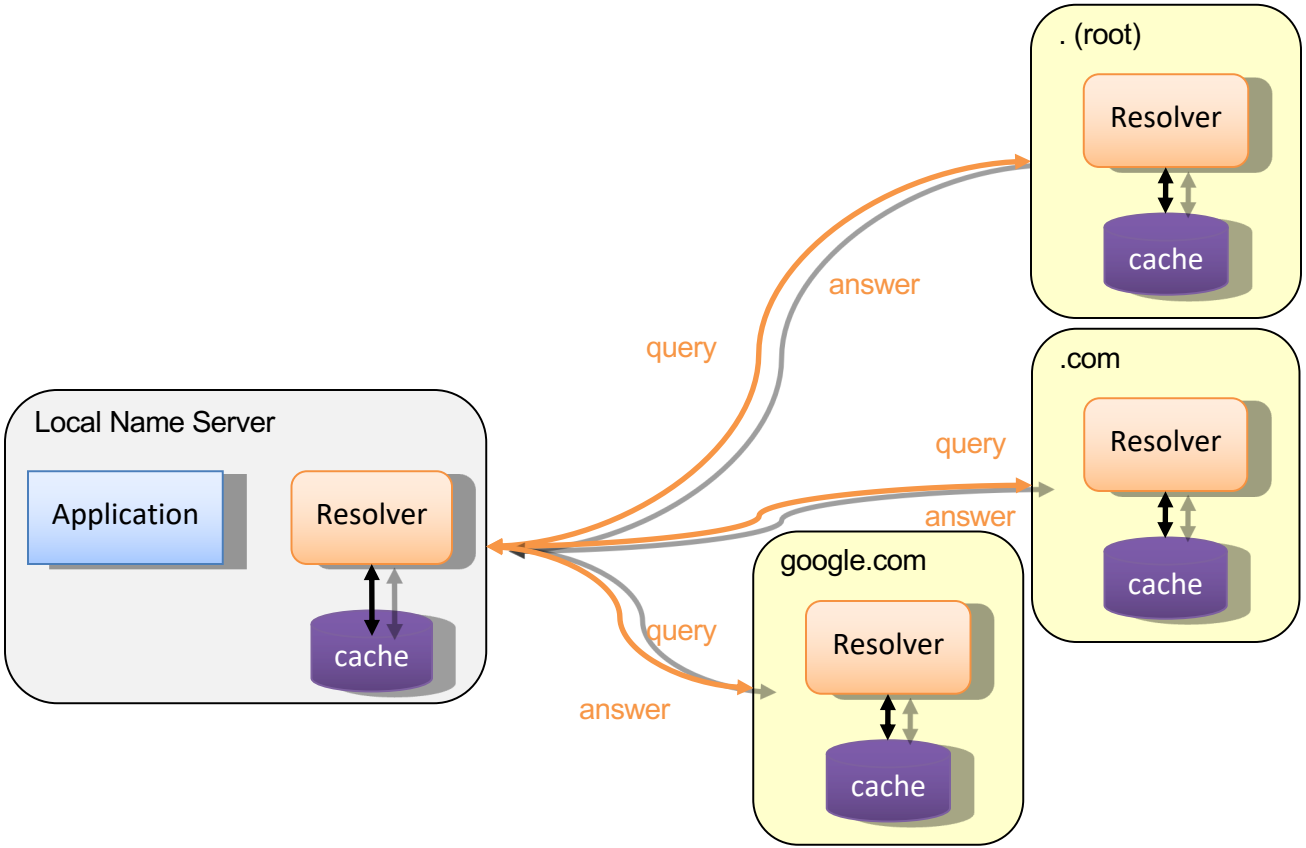
- Zone: collection of connected nodes with the same authoritative DNS server
- Resolution method when answer not in cache.



Recursive Name Resolution



Iterative Name Resolution

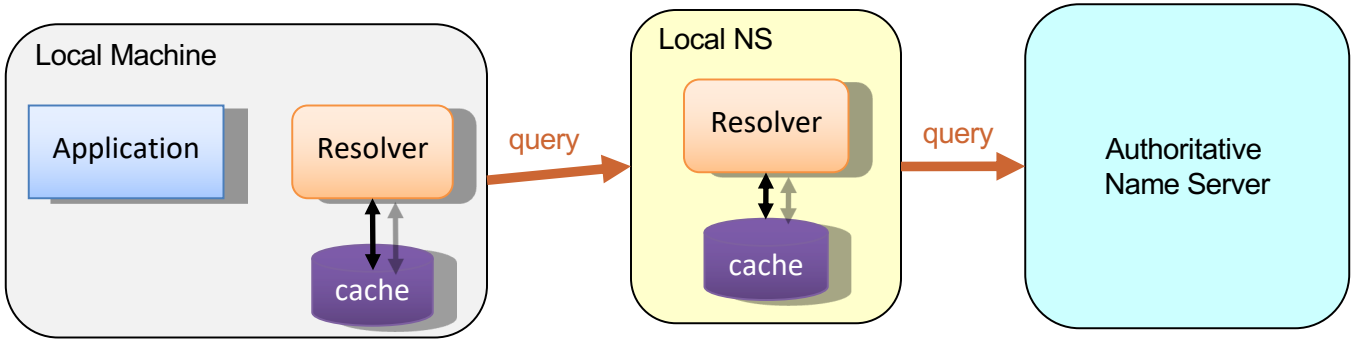


DNS Caching

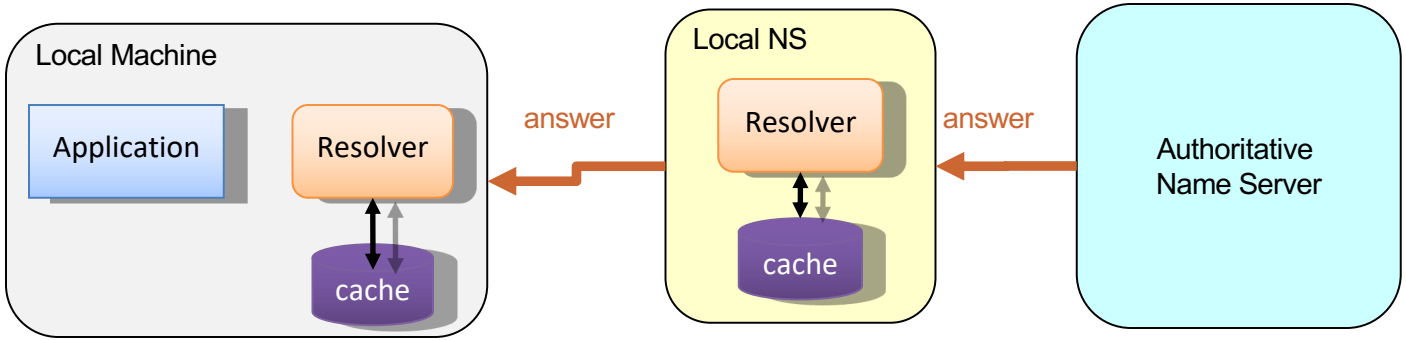
- DNS responses are cached
  - quick response for repeated translations
  - useful for finding servers as well as addresses
- negative results are cached
  - save time for nonexistent sites, e.g., misspelling
- cashed data periodically time out

DNS Caching

Step 1: query yourdomain.org

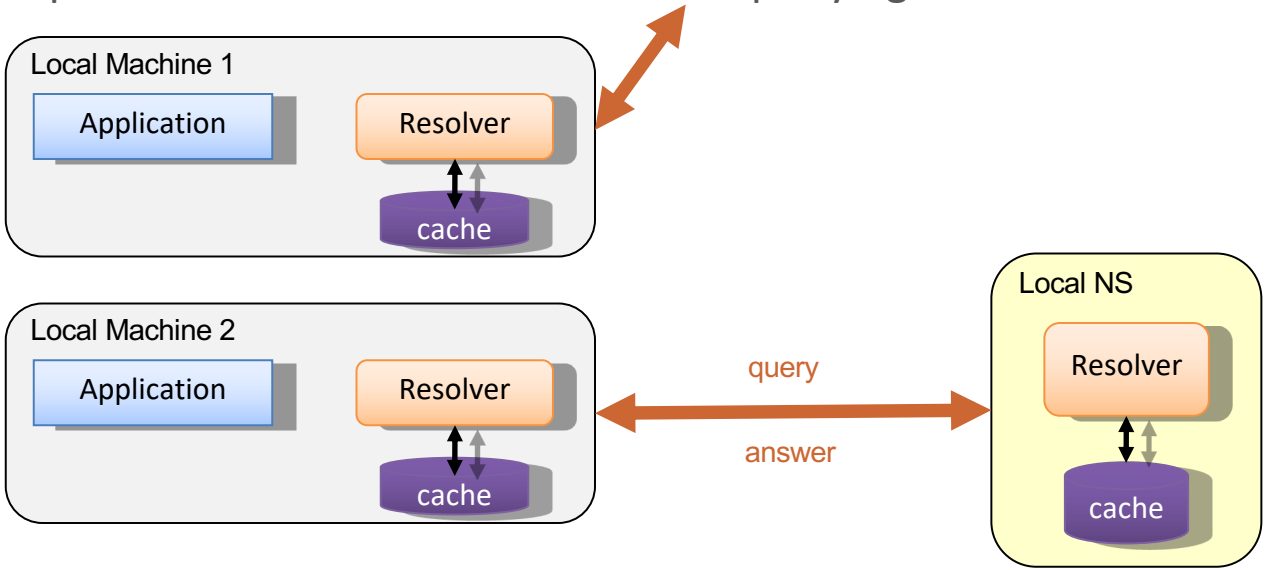


Step 2: receive reply and cache at local name server and host



DNS Caching (con'd)

Step 3: use cached results rather than querying the ANS



Step 4: Evict cache entries upon Time-To-Live (TTL) expiration

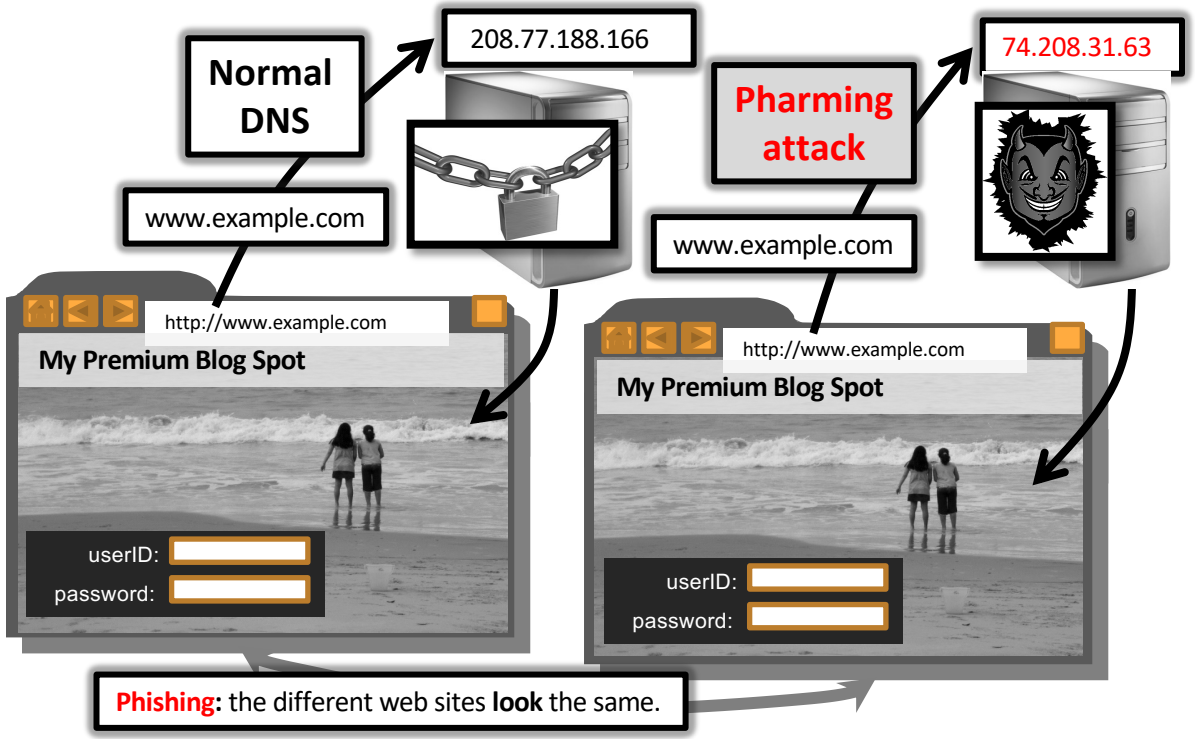
- Common TTL values might be 3600 (1 hour), 86400 (1 day), or even as low as 300 (5 minutes).
- Once the TTL expires, the cached result is **discarded**, and any subsequent request for the same domain will trigger a **new** DNS query to get the updated information.

DNS Cache Poisoning

- DNS is susceptible to cache poisoning attacks
  - Basic idea: change IP address in cache to redirect URLs to fraudulent sites
    - this attack is called pharming/DNS hijacking
    - example
      - www.yahoo.com NS ns.evil.org (delegate to evil.org)
      - ns.evil.org A 1.2.3.4 (address for evil.org)
      - if resolver looks up www.yahoo.com, the address 1.2.3.4 will be returned
  - root cause: DNS uses a 16-bit request identifier to pair queries with answers
  - Cache may be poisoned when a name server:
    - Disregards identifiers
    - Has predictable IDs
    - In 2002, most major DNS software used sequential numbers of query IDs
    - Accepts unsolicited DNS records

DNS Cache Poisoning

- Changing IP associated with a server maliciously:



DNS Cache Poisoning

- DNS cache poisoning
  - the problem is DNS messages are **NOT** authenticated
  - some DNS poisoning attacks in the past
    - in January 2005, the address of a large ISP Panix was redirected to a site in Australia
    - in November 2004, Google and Amazon users were sent to Med Network Inc., an online pharmacy
- There are also attacks on DNS reverse address lookup and DNS implementations
  - example: reverse query buffer overrun in BIND releases 4.9 and 8
    - could gain root access, abort DNS service

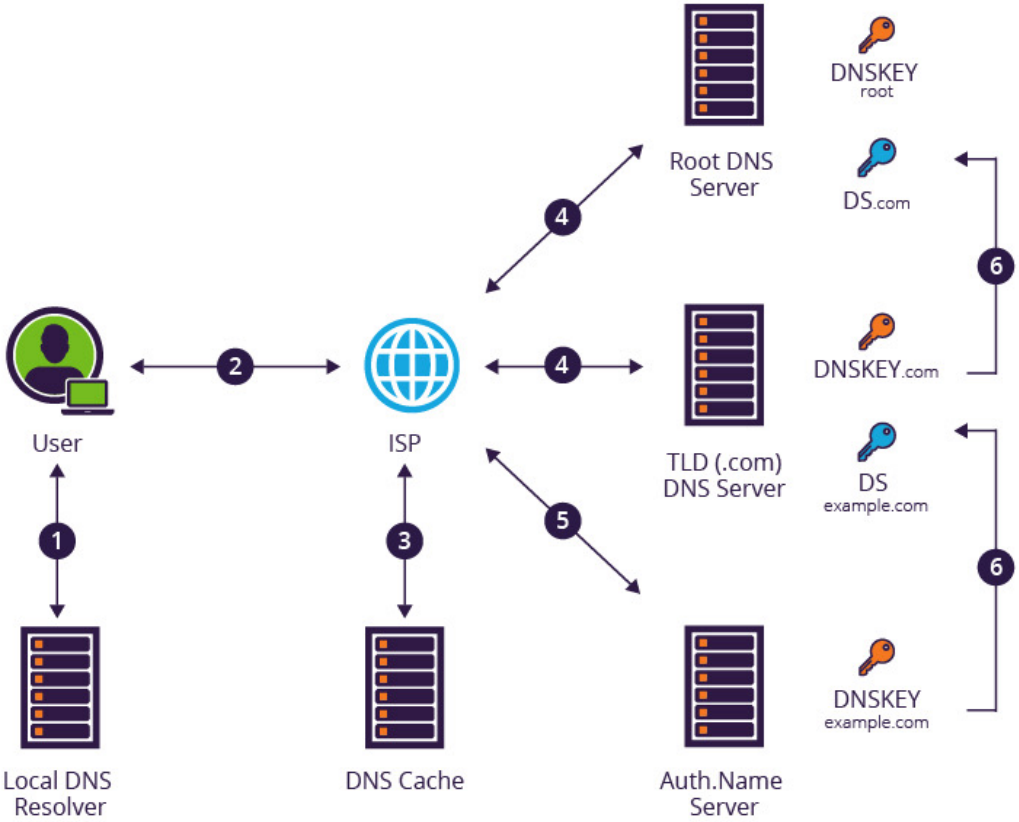
DNS Cache Poisoning Prevention

- Difficult to prevent
  - Relying on a 16-bit number to verify the DNS response
- Possible solutions:
  - Use random identifiers for queries
  - Always check identifiers
  - Port randomization for DNS requests
  - Deploy DNSSEC

DNS Cache Poisoning Prevention - DNSSEC

- Domain Name System Security Extensions (DNSSEC) was developed to protect integrity of DNS records
  - Guarantees:
    - Authenticity of DNS answer origin
    - Integrity of reply
    - Authenticity of denial of existence
  - Accomplishes this by signing DNS replies at each step of the way
    - Uses public-key cryptography to sign responses
  - Typically use trust anchors, entries in the OS to bootstrap the process

DNSSEC



Example of DNSSEC validation process from https://www.imperva.com/learn/application-security/dnssec/

Other Attacks

- Session hijacking attacks
  - host-based session hijacking
    - with root privileges can read and write to local terminal devices
  - network-based session hijacking
    - often performed against TCP
- What harm can be done
  - data injection into unencrypted server-to-server traffic such as email exchange, DNS zone transfers, etc.
  - data injection into unencrypted client-to-server traffic such as ftp file downloads and http responses
  - denial of service attacks such as resetting a connection

Other Attacks

- TCP session hijacking
  - each TCP connection has an associated state
    - client and server IP and port numbers, sequence numbers
  - the problems is that it is not difficult to guess state
    - port numbers can be standard
    - sequence numbers are often chosen in a predictable way
- TCP sequence numbers
  - need high degree of unpredictability
    - attacker who knows initial sequence numbers and amount of traffic sent can estimate likely current values
    - send a flood of packets with likely sequence numbers

Other Attacks

- TCP sequence numbers (cont.)
  - packets can be injected into existing connection
  - some implementations are vulnerable
  - DoS vulnerability
    - if attacker can guess sequence numbers for an existing connection, it can send a RST packet to close connection (DoS)
    - naively, success probability is $1/2^{32}$ (32-bit numbers)
    - most systems allow for a large window of acceptable sequence numbers resulting in much higher success probability
    - attack is most effective against long lived connections such as BGP

Defenses

- Cryptographic network protection
  - protocol level solutions
    - adding authentication to protocols would solve many problems (various types of spoofing and poisoning)
    - perceived as too expensive for current internet speeds/volumes
  - solutions at network layer
    - use cryptographically random initial sequence numbers, IPsec
    - can protect against session hijacking/data injection and DoS using session resets
  - solutions above transport layer
    - tools such as TLS and SSH
    - protect against session hijacking, but not against RST-based DoS

Network Attacks: Summary

Layers	Data Encapsulation (frame/packet)	Protocols	Attacks
Application	HTTP Header, HTTP Data	DNS	DNS cache poisoning
Transport	TCP Header, HTTP Header, HTTP Data	TCP	TCP flood
Network	IP Header, TCP Header, HTTP Header, HTTP Data	IP, ICMP	IP spoofing, ICMP flood
Data Link	MAC Header, IP Header, TCP Header, HTTP Header, HTTP Data	MAC, ARP	MAC spoofing, ARP spoofing

Next

- Network Security
  - Network Firewalls
  - Intrusion Detection Systems

Firewalls

- A **firewall** is an integrated collection of security measures designed to prevent unauthorized access to a networked computer system.
- A network firewall is similar to firewalls in building construction, because in both cases they are intended to isolate one "network" or "compartment" from another.

Firewalls

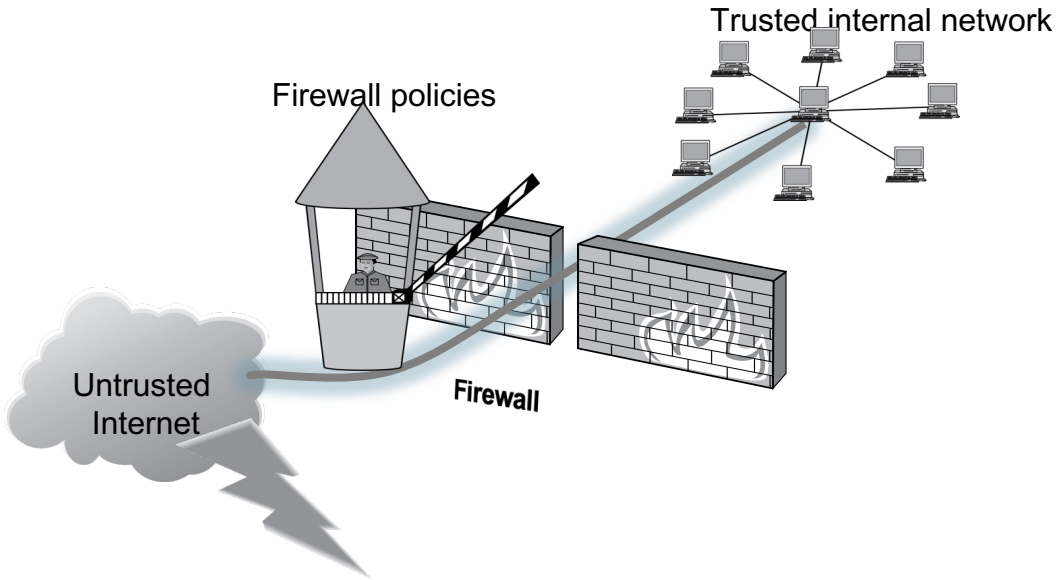
- What can we expect from a firewall?
  - single point that blocks unauthorized users from the protected network and simplifies security management
  - monitoring and reporting of security-related events
  - implementation of virtual private networks by means of IPsec, tunneling
  - convenient place for integration of other functions for network management
- A firewall does not protect against attacks that don’t go through the firewall
  - e.g., wireless connections, internal attacks, external devices connected directly to the internal machines/network

Firewalls

- Where can a firewall reside?
  - on a router
  - on a dedicated machine
  - personal firewall on a host
    - software that protects a single host rather than a network
    - e.g., Windows firewall, iptables in Linux, etc.
    - typically is configured to block most incoming traffic, but some applications can be let through
    - can be bypassed/disabled if host is compromised
- A firewall must be immune to penetration
  - ideally, it should run on a hardened system with a secured OS

Firewall Policies

- To protect private networks and individual machines from the dangers of the greater Internet, a firewall can be employed to filter incoming or outgoing traffic based on a predefined set of rules called **firewall policies**.



Policy Actions

- Packets flowing through a firewall can have one of three outcomes:
  - **Accepted:** permitted through the firewall
  - **Dropped:** not allowed through with no indication of failure
  - **Rejected:** not allowed through, accompanied by an attempt to inform the source that the packet was rejected
- Policies used by the firewall to handle packets are based on several properties of the packets being inspected, including the protocol used, such as:
  - TCP or UDP
  - the source and destination IP addresses
  - the source and destination ports
  - the application-level payload of the packet (e.g., whether it contains a virus).

Blacklists and WhiteLists

- **Blacklist** approach
  - All packets are allowed through except those that fit the rules defined specifically in a blacklist.
  - Flexible in ensuring that service to the internal network is not disrupted by the firewall
  - But naïve from a security perspective in that it assumes the network administrator can enumerate all of the properties of malicious traffic.
- **Whitelist** approach
  - A safer approach to defining a firewall ruleset is the default-deny policy, in which packets are dropped or rejected unless they are specifically allowed by the firewall.

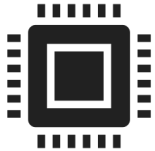
Firewall Types

Packet filters (stateless)

If a packet matches the packet filter's set of rules, the packet filter will drop or accept it

"Stateful" firewalls

It maintains records of all connections passing through it and can determine if a packet is either the start of a new connection, a part of an existing connection, or is an invalid packet.



Application layer

It works like a **proxy** it can “understand” certain applications and protocols.

It may inspect the contents of the traffic, blocking what it views as inappropriate content (i.e. websites, viruses, vulnerabilities, ...)

Packet Filtering (stateless)

- simplest kind of firewall
- router has a list of access control rules
- router checks each received packet against security rules to decide whether to forward or drop it
- each rule specifies which packets it applies to based on a packet's header fields
- can specify source and destination IP addresses, port numbers, protocol names, or wild cards
- actions are `ALLOW` or `DROP`
  - `<ACTION> <PTRCL> <SRC:PT> → <DEST:PT>`

Packet Filtering

- list of rules is examined one-by-one
- first matching rules determines how packet will be handled
- if no match is found, the default option can be to allow or drop
  - if the default option is drop, it is more noticeable to users
  - additional rules are added over time
  - this option, however, is preferred from security management point of view

Packet Filtering

- Policies based on IP header fields
  - a TCP or UDP service is specified by machine's IP address and port number
    - e.g., web server `buffalo.edu` is at `128.205.201.56` port `80`
  - identify each service with triplet (`addr, prot, port`)
    - `addr` is machine's IP address (a.b.c.d/[mask])
    - `prot` is TCP/UDP protocol identifier
    - `port` is the port number
  - example: all official web servers are located on subnet `12.34.56.x`
    - add (`12.34.56.0/24, TCP, 80`) to allowed list

Packet Filtering

- Let's examine a sample ruleset
  - `drop TCP *:* -> *:23`

    `allow * *:* -> *:*`
  - what does it do?
    1. discard any TCP packet that is direct to port "23" (regardless of its source)
    2. allow any packet from any source to any destination
    3. firewall processes rules top-down
    4. thus, any TCP traffic directed to port 23 will be dropped, but all other traffic will be allowed
  - is this ruleset satisfactory?
    - there is no notion of a connection, inbound vs outbound connections
    - inbound and output packets to port 23 are dropped
    - default allow policy is undesirable

Packet Filtering

- Another example
  - assume that we want to allow
    - inbound connections to web server `12.34.56.78` on port `80`
    - all outbound connections
    - nothing else
  - we create the following ruleset
    - ```
      allow TCP *:* -> 12.34.56.78:80
      allow TCP (our-hosts):* -> *:*
      drop * *:* -> *:*
      ```
  - there are problems with it
    - TCP connections are bidirectional, data have to be able to go both ways
      ```

Packet Filtering

- Recall that TCP handshake is 3-way
  - send SYN, receive SYN-ACK, send ACK, then send data with ACK
- Suppose an inside host connects to an external machine on port 25 (mail)
  - initial packets get through (using rule 2)
  - SYN-ACK is dropped (fails the first two rules, matches the last)
- We need to distinguish between two types of inbound packets
  - allow inbound packets associated with an outbound connection
  - disallow inbound packets associated with an inbound connection

Packet Filtering

- We use TCP feature to make this distinction
  - ACK bit is set on all packets except the first one
  - recipients discard any TCP packet with ACK bit if it is not associated with an existing TCP connection
- Revised ruleset
  - ```
    allow TCP *:* -> 12.34.56.78:80
    allow TCP (our-hosts):* -> *:*
    allow TCP *:* -> (our-hosts):* (if ACK bit set)
    drop * *:* -> *:*
    ```
  - rules 1 and 2 permit inbound connections to 12.34.56.78 port 80
  - rules 2 and 3 allow outbound connections to any port
    ```

Packet Filtering

- Let's see how our firewall stops packets
  - attacker wants to exploit finger service vulnerability (TCP port 79)
  - attempt 1: attacker sends SYN packet to internal machine
    - packet doesn't have ACK bit set, so firewall rule drops it
  - attempt 2: attacker sends SYN-ACK packet to internal machine
    - firewall permits the packets, but then it is dropped by the TCP stack (i.e., ACK bit set, but it is not part of an existing connection)
- We can customize the ruleset to let any types of packets through according to the policy
- Does it mean we done now? how about spoofed addresses?

Packet Filtering

- Suppose an attacker can spoof source IP address and performs the following attack
  - let `12.34.56.77` be an internal host
  - attacker sends a spoofed TCP SYN packet from address `12.34.56.77` to another internal machine on port `79`
    - rule 2 in the ruleset allows the packet
  - target machine replies with SYN-ACK packet to `12.34.56.77` and waits for ACK (to finish handshake)
  - attacker sends spoofed ACK packet
  - attacker sends data packet(s)

Packet Filtering

- The attack above permits connections to internal hosts
  - it violates our security policy
  - it allows an attacker to exploit security vulnerabilities in internal machines
  - one difficulty: the attacker has to guess initial sequence number set by target in SYN-ACK packet to `12.34.56.77`
    - the attacker doesn't see the response packet, but guessing might not be difficult
- What do we do now?
  - solve this by taking the interface a packet is coming from into consideration
  - mark a packet with interface id and incorporate ids into the rules

Packet Filtering

- New ruleset
  - internal interface is in, external interface out
  - ```
    allow TCP *:*/out -> 12.34.56.78:80/in
    allow TCP *:*/in -> *:*/out
    allow TCP *:*/out -> *:*/in (if ACK bit set)
    drop * *:* -> *:*
    ```
  - this allows inbound packets only to 12.34.56.78:80 (rule 1) or if ACK bit set (rule 3)
  - all other inbound packets are dropped
- Simple modification cleanly defeats IP spoofing threat
  - it simplifies ruleset administration (no need to hardcode internal hosts)
    ```

Stateless Firewalls

- Stateless packet filtering has its limitations
  - It does not remember the TCP connection states or any other information from any packet
  - It only looks at each packet
  - It cannot correlate packets.

Firewall Types

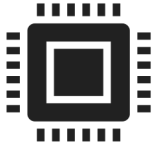
Packet filters (stateless)

Stateless packet filtering has its limitations

- It does not remember the TCP connection states or any other information from any packet
- It only looks at each packet
- It cannot correlate packets.

"Stateful" filters

It maintains records of all connections passing through it and can determine if a packet is either the start of a new connection, a part of an existing connection, or is an invalid packet.



Application layer

It works like a **proxy** it can "understand" certain applications and protocols.

It may inspect the contents of the traffic, blocking what it views as inappropriate content (i.e. websites, viruses, vulnerabilities, ...)

Stateful Firewalls

- Why need stateful: a stateless firewall doesn't know whether a packet belong to an acceptable connection

- **Stateful:**
  - Packet decision made in the context of a connection
  - If packet is a new connection, check against security policy
  - If packet is part of an existing connection, match it up in the state table & update table
    - Can be viewed as packet filtering with rules dynamically updated

Stateful Firewalls

- **Stateful firewalls** can tell when packets are part of legitimate sessions originating within a trusted network.
- Stateful firewalls maintain tables containing information on each active connection, including the IP addresses, ports, and sequence numbers of packets.
- Using these tables, stateful firewalls can allow only inbound TCP packets that are in response to a connection initiated from the internal network.

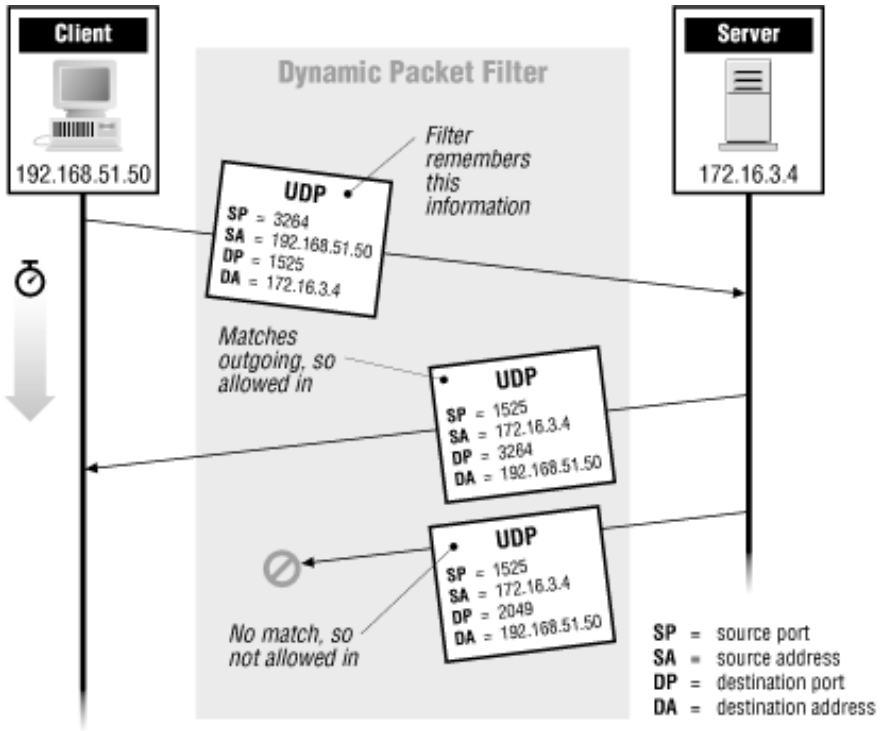
Stateful Firewalls

- Stateful packet inspection
  - packet decision is made in the context of a connection
  - if a packet is a new connection, check against security policy
  - if a packet is part of an existing connection, find it in the state table and update the table
    - this can be viewed as packet filtering with rules dynamically updated
- Example connection state stable

source address	source port	dest address	dest port	conn state
219.22.123.32	2112	124.33.44.5	80	established
124.33.44.129	1030	132.65.89.2	80	established
124.33.44.7	1035	190.3.15.4	25	established

Stateful Firewall Example

- Allow only requested UDP response:



Example from Packet Filtering docstore.mik.ua/orelly/networking_2ndEd/fire/ch08_01.htm

Firewall Types

Packet filters (stateless)

Stateless packet filtering has its limitations

- It does not remember the TCP connection states or any other information from any packet
- It only looks at each packet
- It cannot correlate packets.

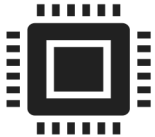
"Stateful" firewalls

Advantages

- Can do everything a packet filter can do plus...
- Keep track of ongoing connections

Disadvantages

- Cannot see application data
- Slower than packet filtering



Application layer

It works like a **proxy** it can "understand" certain applications and protocols.

It may inspect the contents of the traffic, blocking what it views as inappropriate content (i.e. websites, viruses, vulnerabilities, ...)

Application Layer Firewalls

- Application layer firewalls (or proxy firewalls)
  - is used as a relay for connections: Client ↔ Proxy ↔ Server
  - understands specific applications
    - limited versions of applications are available
    - proxy “impersonates” both sides of a connection
    - tends to be more secure than simple packet filters (can block application-specific attacks, can support authentication)
  - is resource-intensive (i.e., one process per connection)
  - certain proxies (e.g., HTTP) may cache data (e.g., web pages)

Firewall Types

Packet filters (stateless)

Stateless packet filtering has its limitations

- It does not remember the TCP connection states or any other information from any packet
- It only looks at each packet
- It cannot correlate packets.

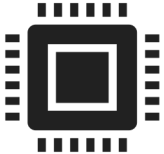
"Stateful" firewalls

Advantages

- Can do everything a packet filter can do plus...
- Keep track of ongoing connections

Disadvantages

- Cannot see application data
- Slower than packet filtering



Application layer

Advantages

- Complete view of connections and applications data
- Filter bad data at application layer (viruses, Word macros)
  - Email system – disable attachment

Disadvantage

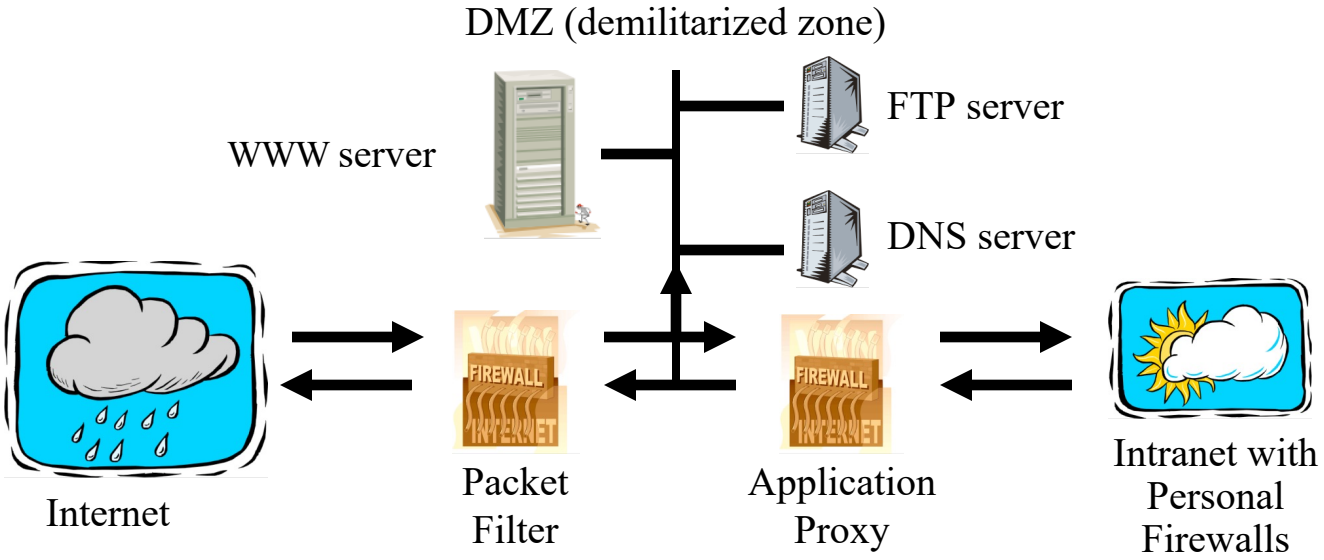
- Speed

Personal Firewalls

- Running on one PC, controlling network access
  - Windows firewall, iptables (Linux), ZoneAlarm, etc.
- Typically determines network access based on application programs
- Typically block most incoming traffic, harder to define policies for outgoing traffic
- Can be bypassed/disabled if host is compromised

Firewalls and Defense in Depth

- Example security architecture



Other tools

Tunnels: A technique that encapsulates one network protocol within another, creating a secure pathway for data.

- **Purpose:** To protect data as it travels across untrusted networks by isolating the payload from potential threats.
- **Key Features:** Encapsulation, secure pathway, protocol layering
- **Use Cases:** Forming the basis for VPN connections, Secure communication channels

SSH (Secure Shell): A protocol for secure remote access that encrypts data exchanged between the client and server.

- **Purpose:** To provide secure remote command-line access, file transfers, and even secure tunneling for other applications.
- **Key Features:** Encryption, authentication, port forwarding, file transfer (SCP/SFTP)
- **Use Cases:** Remote system administration, Secure file transfer, Creating encrypted tunnels

IPSec (Internet Protocol Security): A suite of protocols designed to secure IP communications by authenticating and encrypting each packet.

- **Purpose:** To protect data flows between devices over IP networks by ensuring confidentiality, integrity, and authentication.
- **Key Features:** Packet encryption, Authentication, Integrity verification, Network layer security
- **Use Cases:** Establishing secure VPN tunnels, Protecting data transmissions, Securing corporate networks

VPN (Virtual Private Network): A service that creates an encrypted connection (tunnel) over a public network, effectively extending a private network.

- **Purpose:** To allow remote users and branch offices to securely access a corporate network over the Internet.
- **Key Features:** Encrypted tunnel, Remote access, IP masking, Secure connectivity
- **Use Cases:** Secure remote work, Protecting data on public Wi-Fi, Connecting multiple office locations

Next

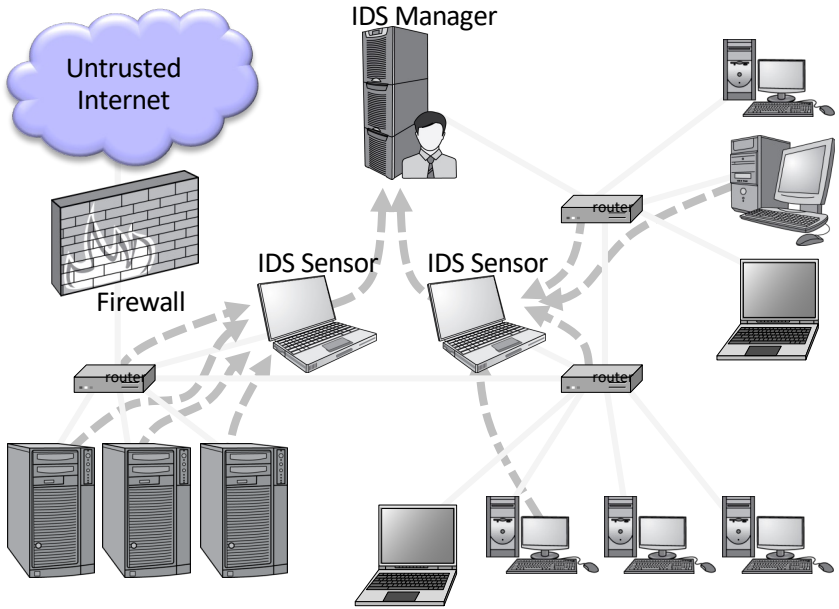
- Network Security
  - Network Firewalls
  - Intrusion Detection Systems

Intrusion Detection

- Intrusion
  - Actions aimed at compromising the security of the target (confidentiality, integrity, availability of computing/networking resources)
- Intrusion detection
  - The identification of intrusions and report of intrusion activities
- Intrusion prevention
  - The process of both detecting intrusion activities and managing automatic responsive actions throughout the network

Intrusion Detection System (IDS) Components

- The **IDS manager** compiles data from the IDS sensors to determine if an intrusion has occurred.
- This determination is based on a set of **site policies,** which are rules and conditions that define probable intrusions.
- If an IDS manager detects an intrusion, then it sounds an **alarm**.



Intrusion Detection Systems

- Who is likely intruder?
  - May be outsider who got thru firewall
  - May be evil insider
- What do intruders do?
  - Launch well-known attacks
  - Launch variations on well-known attacks
  - Launch new or un-known attacks
  - Use a system to attack other systems
  - Etc.

Intrusions

- An IDS is designed to detect automated attacks and threats, including the following:
  - **Port scans:** information gathering intended to determine which ports on a host are open for TCP connections
  - **Denial-of-service attacks:** network attacks meant to overwhelm a host and shut out legitimate accesses
  - **Malware attacks:** replicating malicious software attacks, such as Trojan horses, computer worms, viruses, etc.
  - **ARP spoofing:** an attempt to redirect IP traffic in a local-area network
  - **DNS cache poisoning:** a pharming attack directed at changing a host's DNS cache to create a falsified domain-name/IP-address association

Intrusion Detection Systems

- Intrusion detection is not perfect, two types of errors are
  - false positives: legitimate behavior of authorized users is classified as an intrusion
  - false negatives: an intrusion is not recognized as suspicious activity
- False negatives result in higher losses than false positives
  - thus a higher rate of false positives is normally tolerated than the rate of false negatives
  - if an error rate is very high, warnings tend to get ignored
  - proper tuning of the system is important
- The earlier intrusion is detected, the better
  - it is easier to recover while the damage is small

Intrusion Detection Systems (IDS)

- Intrusion detection system (IDS) is a security service that monitors and analyzes system events
- IDS classification
  - host-based IDS
    - monitors events and characteristics of a single host for suspicious activity
  - network-based IDS
    - monitors data on the network for traces of suspicious activity
    - often a single monitor scans data sent to/from many machines on the network
  - hybrid IDS
    - combines information gathered from hosts and network

Host-based IDS

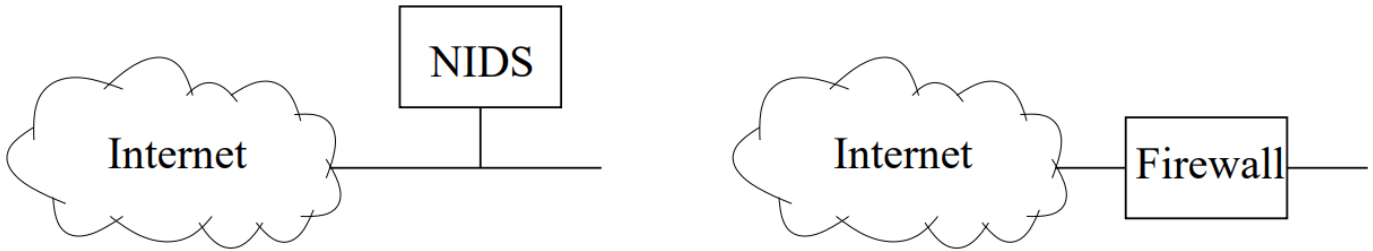
- A host-based IDS runs on a single host
  - it is best positioned to evaluate the state of the machine
- It can monitor events and activity such as
  - login and session activity
    - frequency and location, time since last login, failed login attempts
    - events of security importance can include break-in into a dead account, logins from unusual locations or unusual hours, password guessing, etc.
  - program execution activity
    - monitored activity can include execution denials, resource utilization and execution frequency

Host-based IDS

- Monitored events and activity
  - file access activity
    - record frequency of different types of file access, denial of access
    - look for abnormal usage patterns, suspicious activity such as copying system programs or opening devices directly
  - some combination of the above
    - e.g., users who login after hours often access the same files they used earlier
- If a host-based IDS runs on each host, information from different machines can be collected and managed at a central facility
  - the central manager receives aggregate information and distributes updates to all machines running the IDS

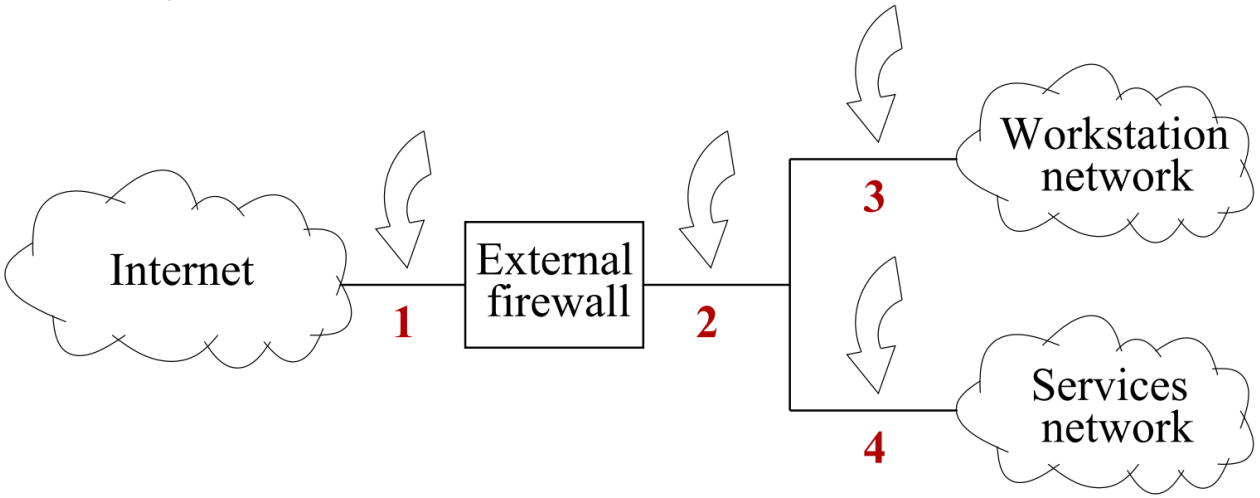
Network-based IDS

- A network-based IDS monitors traffic corresponding to many machines on a network
  - often such a monitor is passive
    - NIDS receives a copy of the traffic
  - a **firewall**, on the other hand, performs active filtering
    - all traffic goes directly through it
  - active filtering adds overhead and normally needs to be minimized



Network-based IDS

- Where NIDS is positioned matters



  - point 1: complete picture of traffic, lots of data
  - point 2: can recognize problems with firewall, see outgoing attacks
  - points 3 and 4: increased visibility of attacks on the local network, can see internal attacks

Network-based IDS

- A NIDS is often stateful and performs deep packet inspection
  - full stream reassembly
  - analysis at network, transport and/or application layers
    - **network layer**: IP, ICMP protocols, illegal header values, spoofed addresses
    - **transport layer**: analysis of TCP and UDP headers, detection of unusual packet fragmentation, floods, scans
    - **application layer**: understanding of DHCP, DNS, HTTP, Network File System (NSF), remote login and many other protocols; detection of buffer overflow attacks, malware propagation, etc.
  - detection of DoS attacks, scanning, malware (worms)

Network-based IDS

- Example systems
  - Snort
    - can be host-based or network-based
    - can monitor traffic inline (supports intrusion prevention) or passively
    - intrusion detection/prevention is rule-based
  - Bro
    - provides passive monitoring of network traffic
    - suitable for high-speed high-volume detection
  - commercial appliances

Intrusion Detection Systems

- IDSs can be classified based on how they recognize suspicious activity
  - misuse detection (signature or heuristic based)
    - define what constitutes an intrusion attempt through a set of rules
    - e.g., specific patterns in network traffic, a combination of events
    - can detect only known/encoded intrusion attempts
  - anomaly detection
    - train the system on clean data to understand behavior of legitimate users
    - use it to monitor real data and detect anomalous behavior
    - advantages: more flexible, can detect unknown misuses
    - disadvantages: higher error rate, difficult to tune

Signature or Heuristic Detection

- Signature approaches
  - Match a large collection of known patterns of malicious data against data stored on a system or in transit over a network
  - The signatures need to be large enough to minimize the false alarm rate while still detecting a sufficiently large fraction of malicious data
  - Widely used in anti-virus products, network traffic scanning proxies, and NIDS
- Rule-based heuristic identification
  - Involves the use of rules for identifying known penetrations or penetrations that would exploit known weaknesses
  - Rules that identify suspicious behavior can also be defined, even when the behavior is within the bounds of established patterns of usage
  - Typically rules used are specific
  - SNORT is an example of a rule-based NIDS

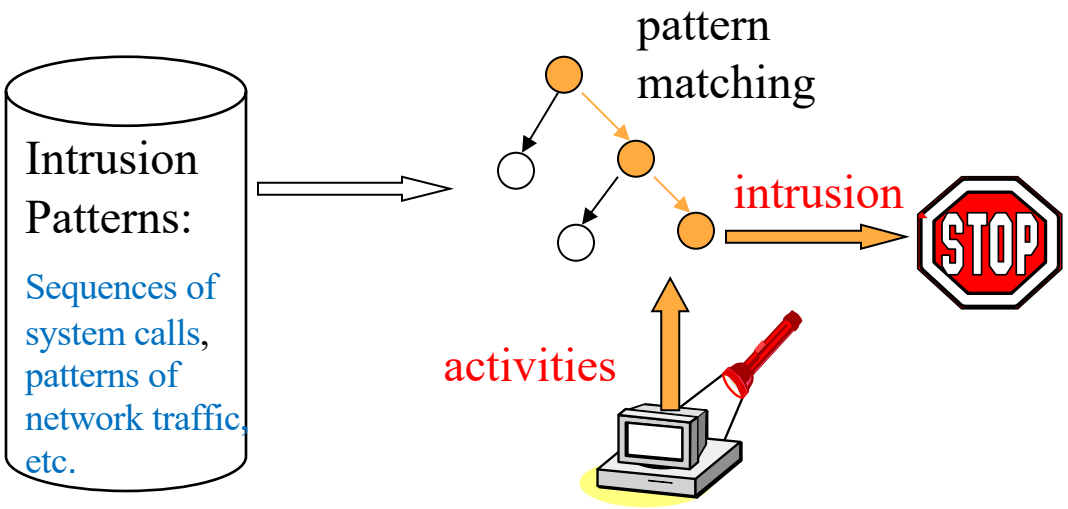
Signature Detection Example

- Failed login attempts may indicate password cracking attack
- IDS could use the rule "N failed login attempts in M seconds" as **signature**
- If N or more failed login attempts in M seconds, IDS warns of attack
- Note that the warning is specific
  - Admin knows what attack is suspected
  - Admin can verify attack (or false alarm)

Signature Detection

- But if attacker knows the signature, he can try $N-1$ logins every M seconds!
- In this case, signature detection slows the attacker, but might not stop him

Signature Detection



Example: ***if*** (traffic contains "x90+de[^\r\n]{30}") ***then*** "attack detected"

Advantage: Mostly accurate. But problems?

Can't detect new attacks

Signature Detection

- Advantages of signature detection
  - Simple
  - Detect known attacks
  - Know which attack at time of detection
  - Efficient (if reasonable number of signatures)
- Disadvantages of signature detection
  - Signature files must be kept up to date
  - Number of signatures may become large
  - Can only detect known attacks
  - Variation on known attack may not be detected

Anomaly Detection

- A variety of classification approaches are used:
  - Statistical
    - Analysis of the observed behavior using univariate, multivariate, or time-series models of observed metrics
  - Knowledge based
    - Approaches use an expert system that classifies observed behavior according to a set of rules that model legitimate behavior
  - Machine-learning
    - Approaches automatically determine a suitable classification model from the training data using data mining techniques

Anomaly Detection

- Anomaly detection systems look for unusual or abnormal behaviors
- There are (at least) two challenges
  - What is normal for this system?
  - How “far” from normal is abnormal?
- Statistics is obviously required here!
  - The **mean** defines normal
  - The **variance** indicates how far abnormal lives from normal

Anomaly Detection

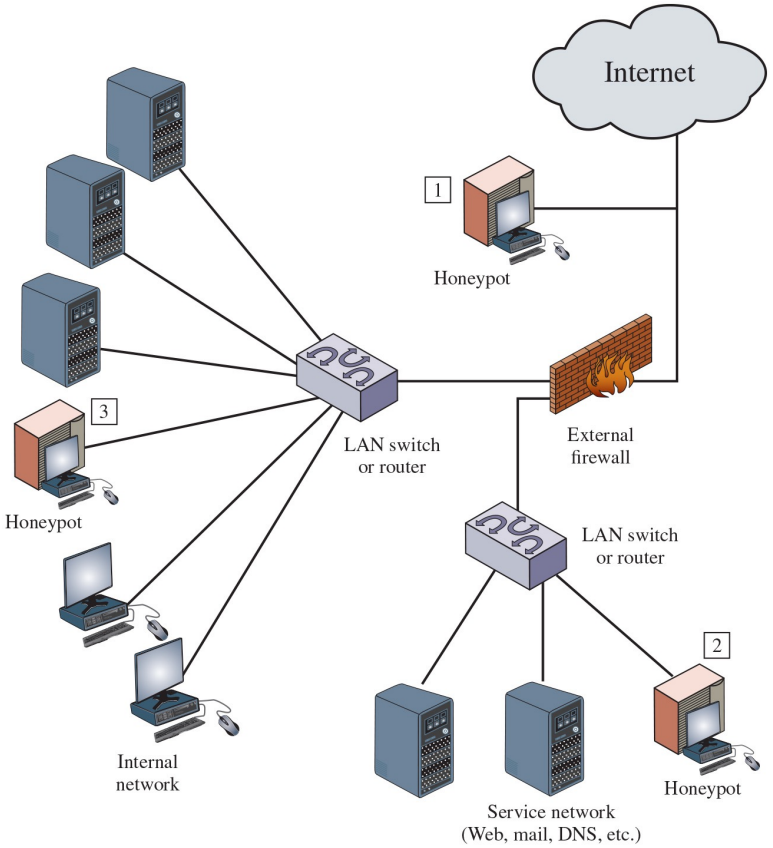
- Advantages
  - Chance of detecting unknown attacks
  - May be more efficient (since no signatures)
- Disadvantages
  - Reliability is unclear
    - High false positive/false negative
  - Anomaly detection indicates something unusual, but lack of specific info on possible attack!
  - Today, cannot be used alone
  - Must be used with a signature detection system

Honeypots

- A further component of intrusion detection technology is the honeypot
- Decoy systems designed to:
  - Lure a potential attacker away from critical systems.
  - Collect information about the attacker's activity.
  - Encourage the attacker to stay on the system long enough for administrators to respond.
- Systems are filled with fabricated information that a legitimate user of the system wouldn't access
- Resources that have no production value
  - Therefore incoming communication is most likely a probe, scan, or attack
  - Initiated outbound communication suggests that the system has probably been compromised

Honeypots

Example of Honeypot Deployment



Firewall and IDS: Summary

- Firewalls: first line of defense
- Intrusion detection systems
  - Based on deploy position:
    - host-based: best positioned to detect attacks on a machine
    - network-based: monitors traffic of the entire network
    - hybrid
  - Based on detection method:
    - signature-based: effective, but don’t recognize new attacks
    - anomaly-based: can find novel attacks, but often result in many false positives
- Effort must be applied to protect the IDS itself from attacks

Summary

- Computer network concepts: layering, encapsulation, etc.

- Network attacks
  - MAC Spoofing, ARP Spoofing
  - IP Spoofing
  - Denial of Service Attacks
  - DNS cache poisoning

- Network security
  - Network Firewalls
  - IDS