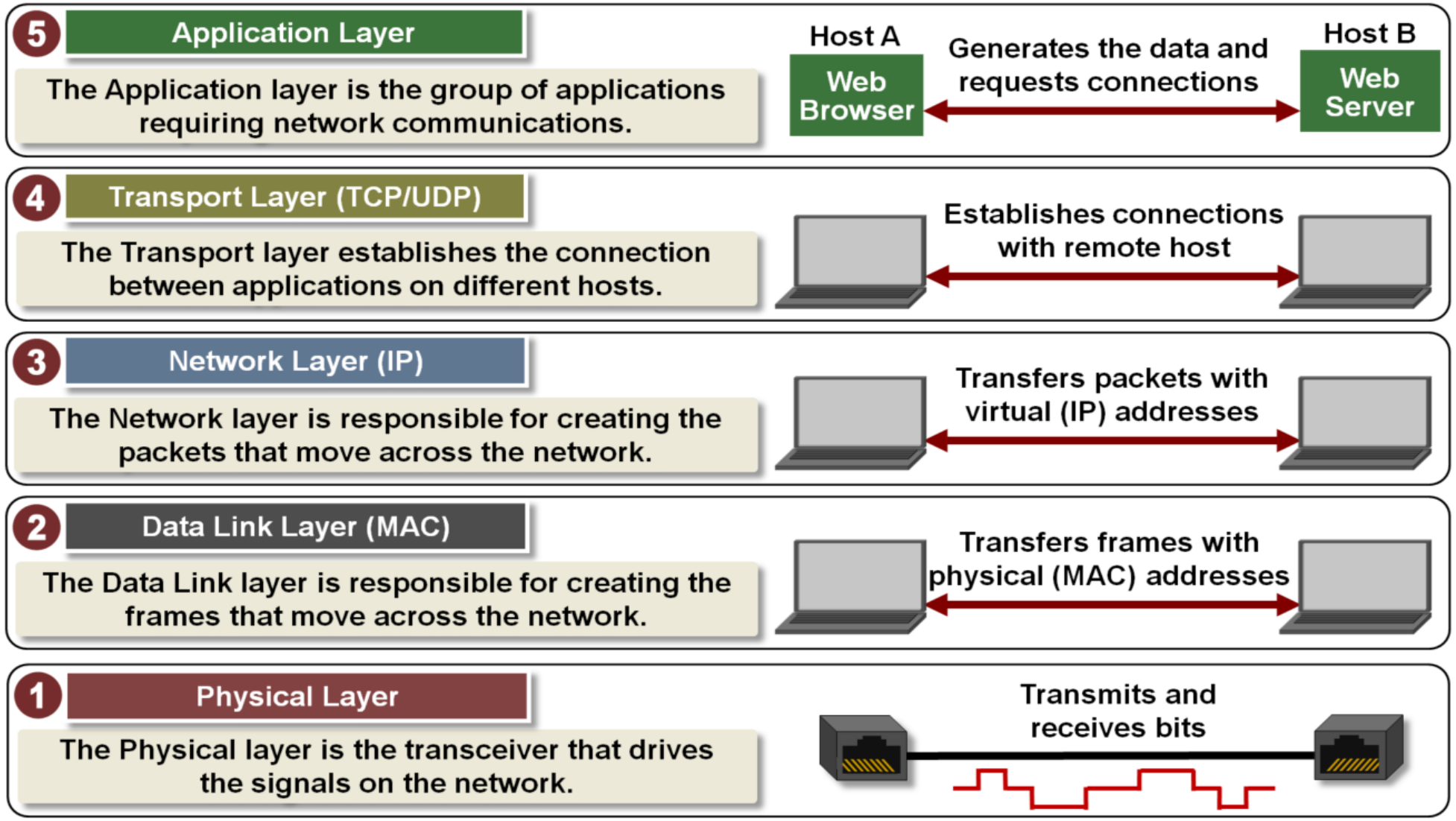
# Introduction to IP Networks

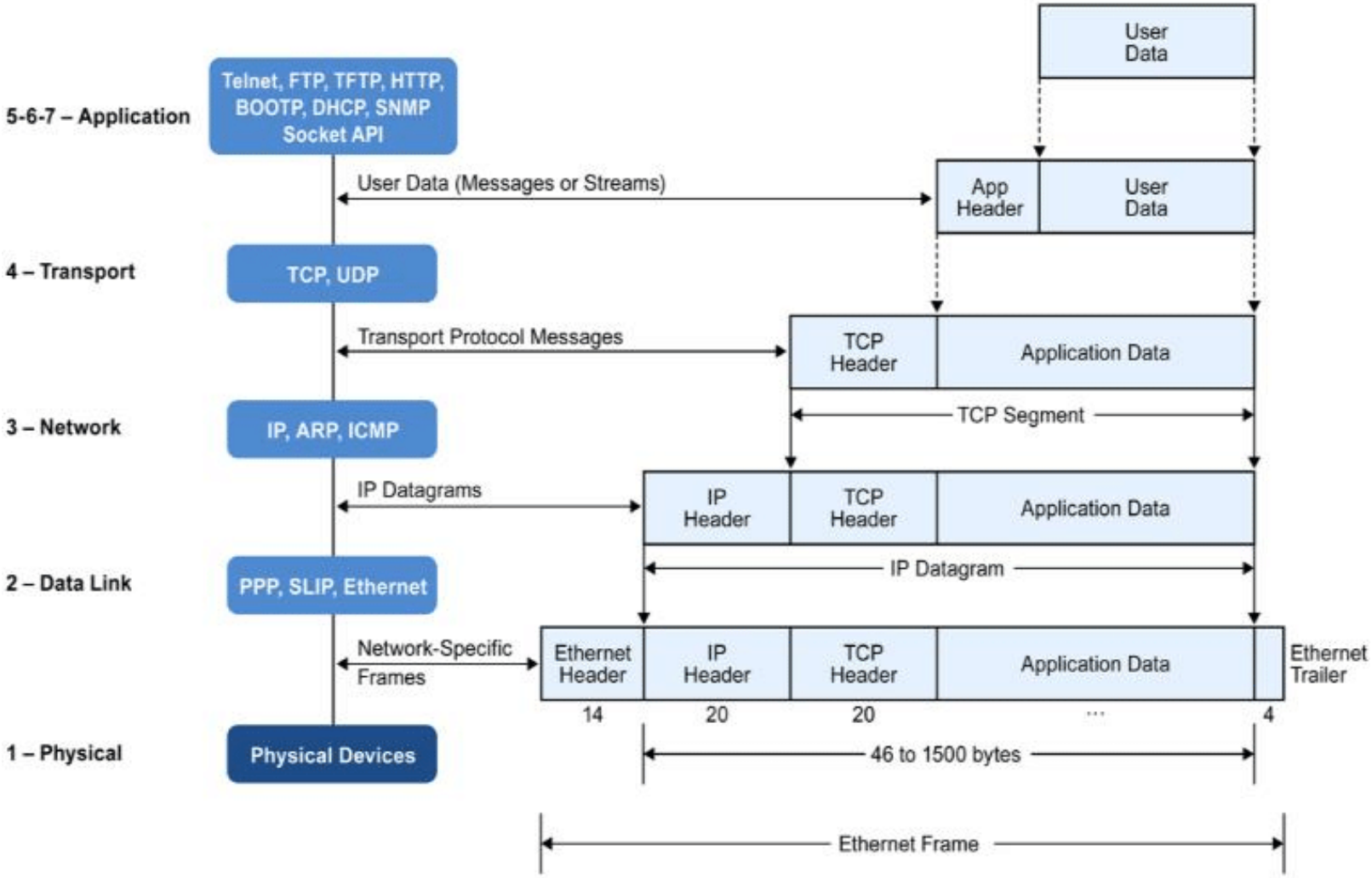
17TH FEB, 2020

SAHIL SEMICONDUCTOR

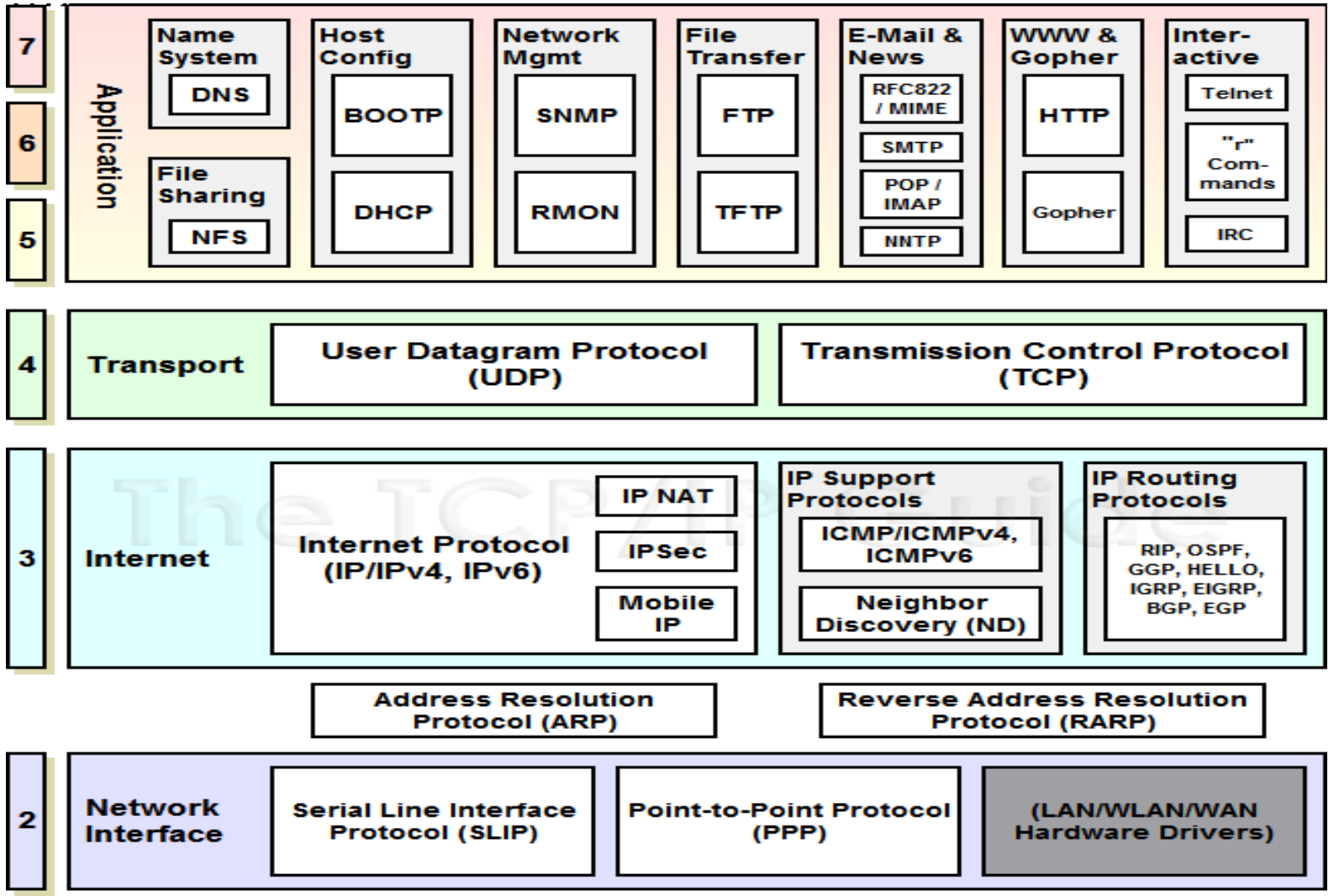
# TCP/IP Layers



# TCP/IP Layers – Packet Encapsulation



# TCP/IP Applications



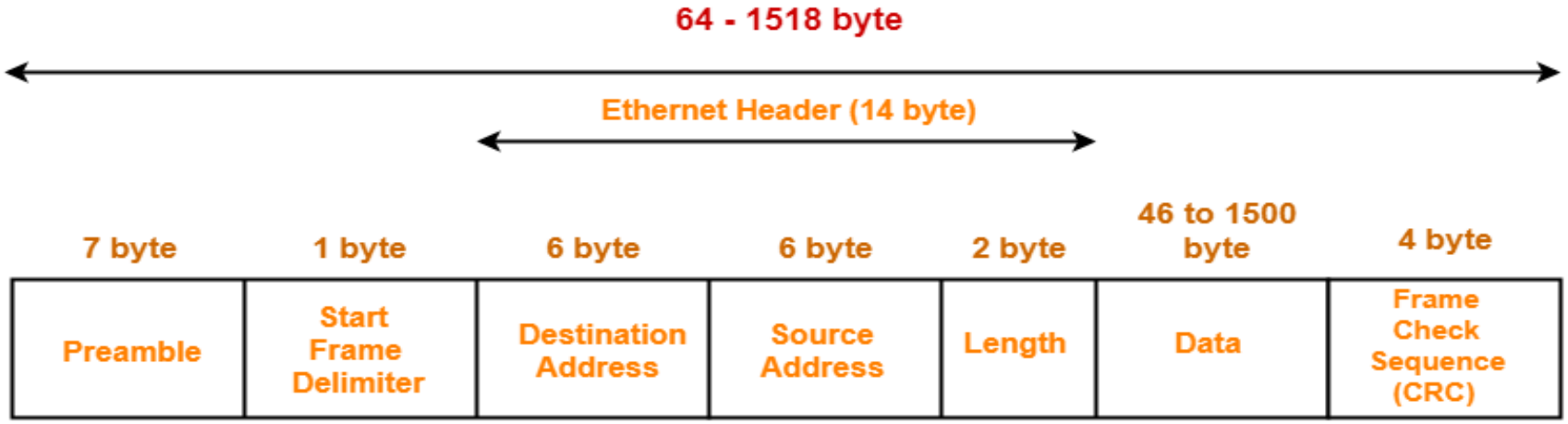
# Data Link Layer (MAC)

# Data Link Layer

- Ethernet is the most popular physical layer LAN technology.
- Standardized as IEEE 802.3.
- It supports auto-negotiation and duplex modes.
- Physical supported mediums include Coaxial cable, twisted pair and optical fiber.
- Ethernet speeds now can reach around 400Gbps.

Click to add text

# Ethernet Frame



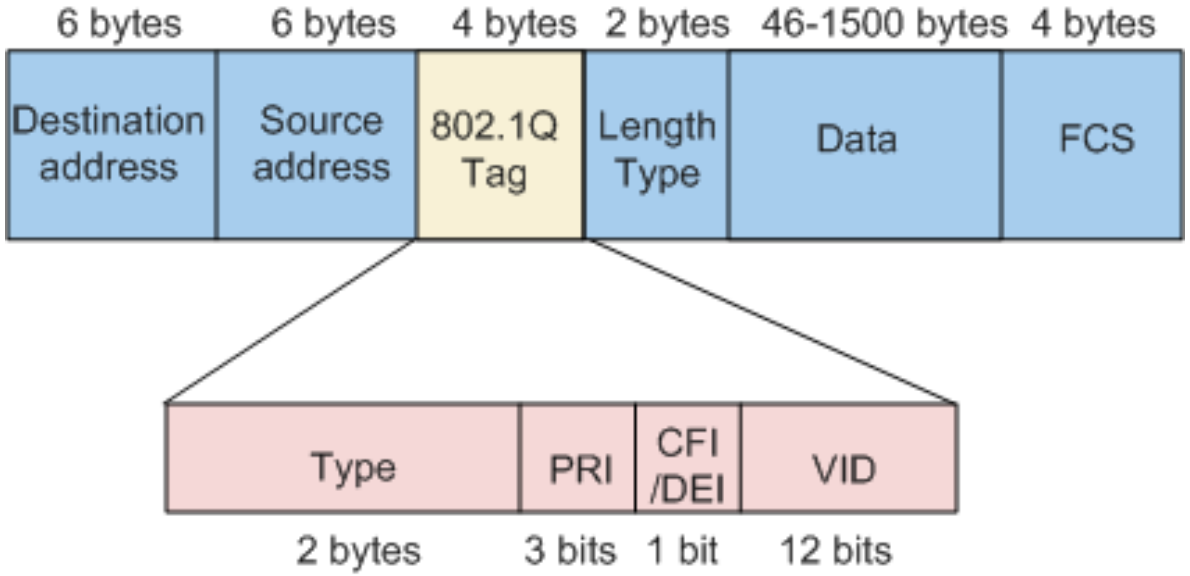

IEEE 802.3 Ethernet Frame Format

- **Preamble:** Sequence of alternate 0's and 1's indicating start of the frame.
- **Start Frame Delimiter (SFD):** One-byte field which is always set to 10101011.
- **Destination and Source Addresses:** 48-bit MAC addresses
- **Length:** Length of the entire frame in bytes (Lengths > 1536 indicates Ether Type)
- **Data:** Payload
- **Frame Check Sequence (CRC):** 32-bit hash code of the data (Dst/Src addr, Length, Data)

# VLAN (Virtual LAN)

- VLAN is defined as IEEE 802.1Q. It defines a system of VLAN tagging for Ethernet frames and procedures used by switches to handle such frames.
- When a frame enters the VLAN-aware portion of the network, a tag is added to represent the VLAN membership.
- Each switch uses the tag to keep each VLAN's traffic separate from other VLANs.
- Switches use MAC based learning to forward VLAN packets to their respective ports.
- Both destination MAC address and VLAN ID are used for routing.

# Ethernet Frame Format for VLAN

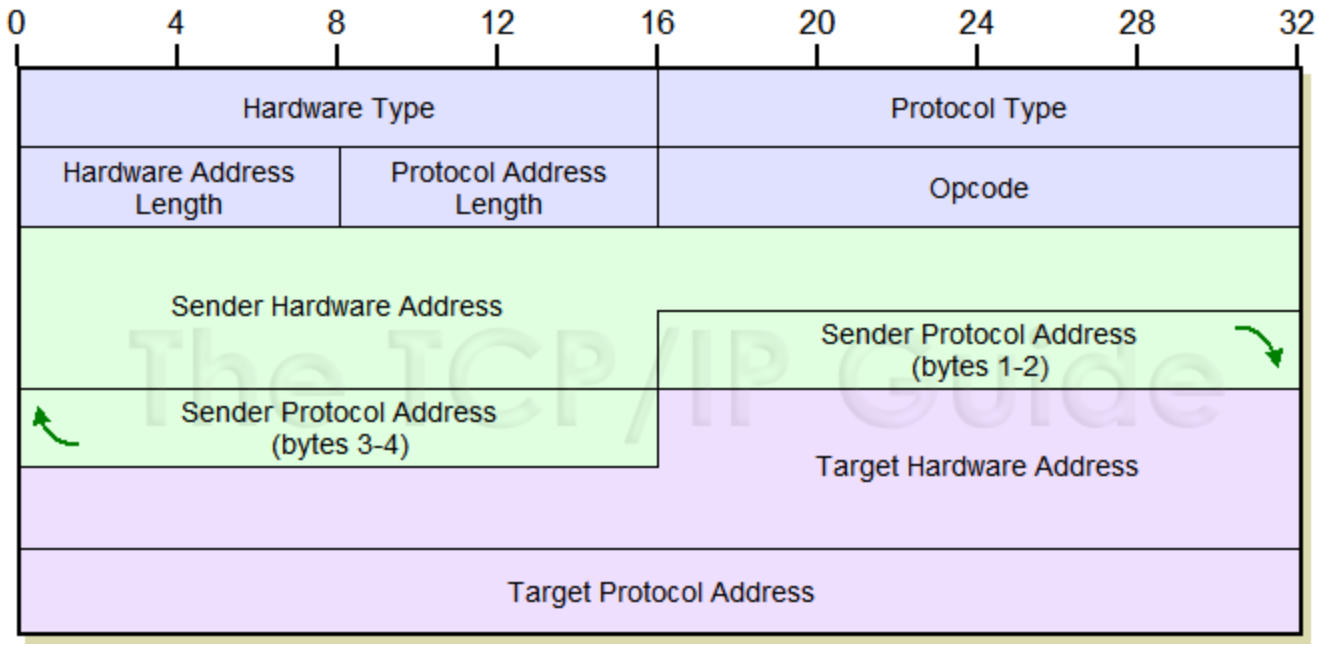



- **Type:** Ether type of 0x8100 to indicate that this is a VLAN frame.
- **PRI (Priority Code Point):** Frame priority level.
- **CFI/DEI (Drop Eligible Indicator):** Indicates if the frame can be dropped in case of congestion.
- **VID (VLAN Identifier):** 12-bit field specifying VLAN ID (4096 VLANs IDs, some of them are reserved).

# ARP (Address Resolution Protocol)

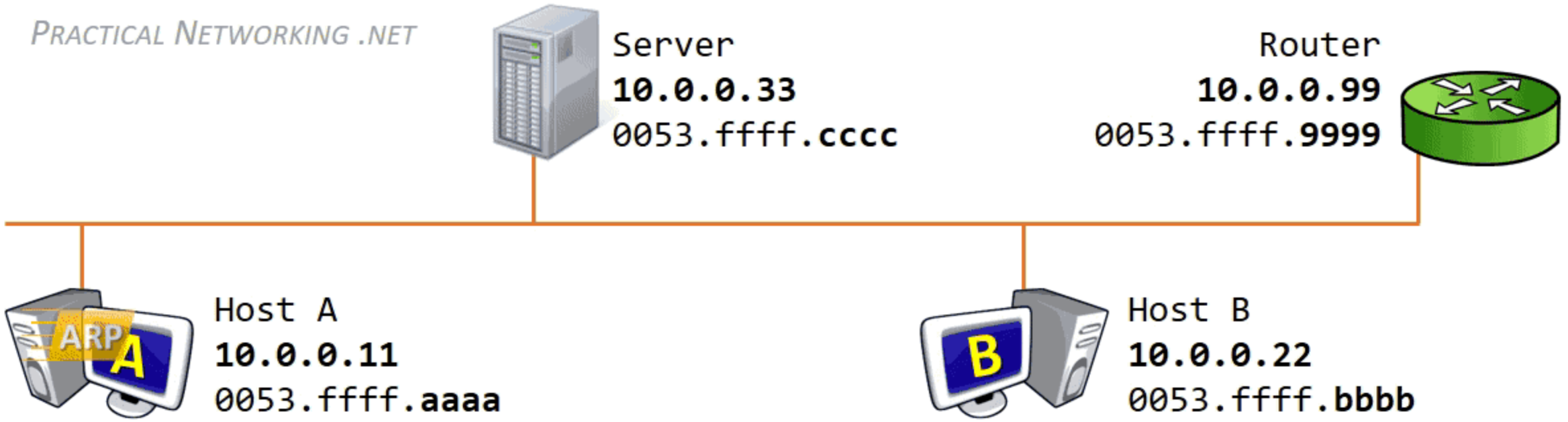
- How to find out destination MAC address?
- We can maintain a table with all MAC-IP mappings, but it is not practical. There will be millions of entries and keeping them up to date will be nearly impossible.
- ARP protocol (RFC 826) defines how to discover a MAC address associated to an IP address.
- Once the MAC address is resolved through ARP, the host keeps them in its local cache for future references. A timeout will invalidate this entry if it is not used for certain amount of time.
- IPv6 provides a similar protocol (Neighbor Discovery Protocol) for MAC address resolution.
- In the following video the working of ARP is explained:
- ARP Explained - Address Resolution Protocol

# ARP (Address Resolution Protocol)

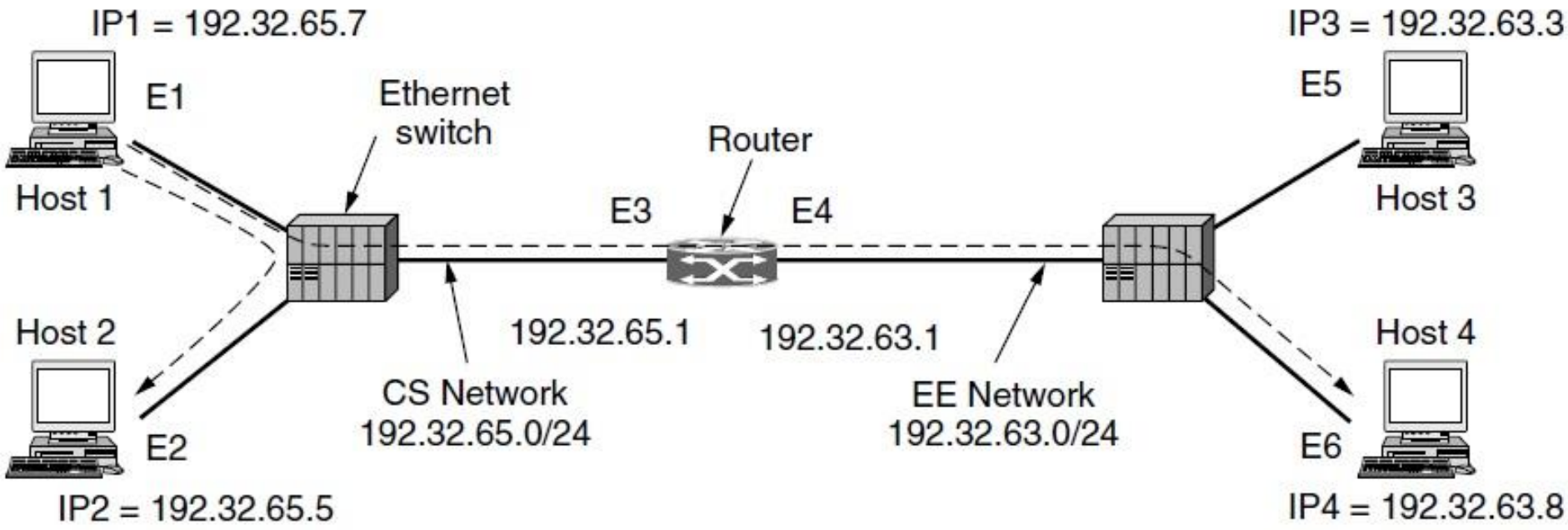



- **Hardware Type:** 6 for 802 networks, 18 for Fiber Channel
- **Protocol Type:** 0x0800 for IPv4 (we discussed this in an earlier slide)
- **Hardware Address Length:** MAC address length for ethernet (6)
- **Protocol Address Length:** IPv4 address length for ethernet (4)
- **Opcode:** Type of ARP message (1 – request, 2 – reply etc.)
- **Addresses:** MAC and IP addresses

# ARP – how it works



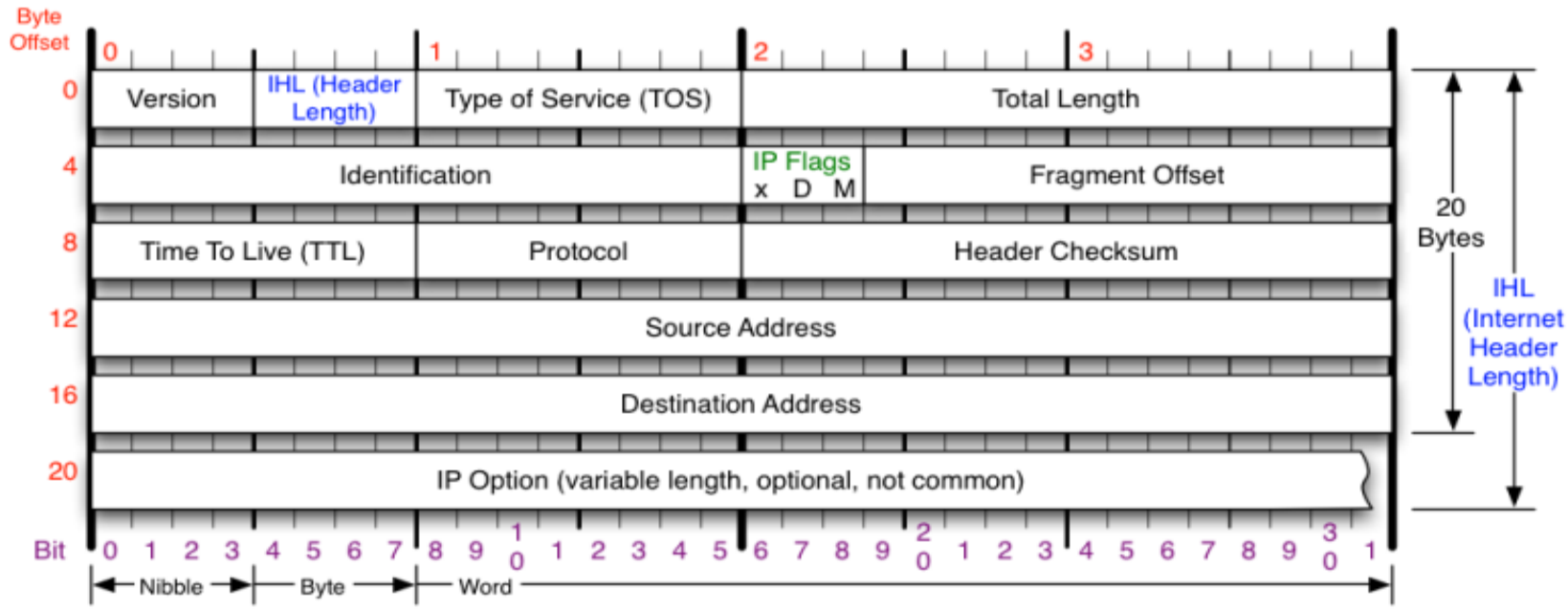
# ARP – Addresses to be used




| Frame | Source IP | Source Eth. | Destination IP | Destination Eth. |
|---|---|---|---|---|
| Host 1 to 2, on CS net | IP1 | E1 | IP2 | E2 |
| Host 1 to 4, on CS net | IP1 | E1 | IP4 | E3 |
| Host 1 to 4, on EE net | IP1 | E4 | IP4 | E6 |

# Network Layer (IP)

# IP – Internet Protocol




**Version**

Version of IP Protocol. 4 and 6 are valid. This diagram represents version 4 structure only.

**Header Length**

Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.

**Protocol**

IP Protocol ID. Including (but not limited to):

| | | |
|---|---|---|
| 1 ICMP | 17 UDP | 57 SKIP |
| 2 IGMP | 47 GRE | 88 EIGRP |
| 6 TCP | 50 ESP | 89 OSPF |
| 9 IGRP | 51 AH | 115 L2TP |

**Total Length**

Total length of IP datagram, or IP fragment if fragmented. Measured in Bytes.

**Fragment Offset**

Fragment offset from start of IP datagram. Measured in 8 byte (2 words, 64 bits) increments. If IP datagram is fragmented, fragment size (Total Length) must be a multiple of 8 bytes.

**Header Checksum**

Checksum of entire IP header

**IP Flags**

x D M

x 0x80 reserved (evil bit)
D 0x40 Do Not Fragment
M 0x20 More Fragments follow

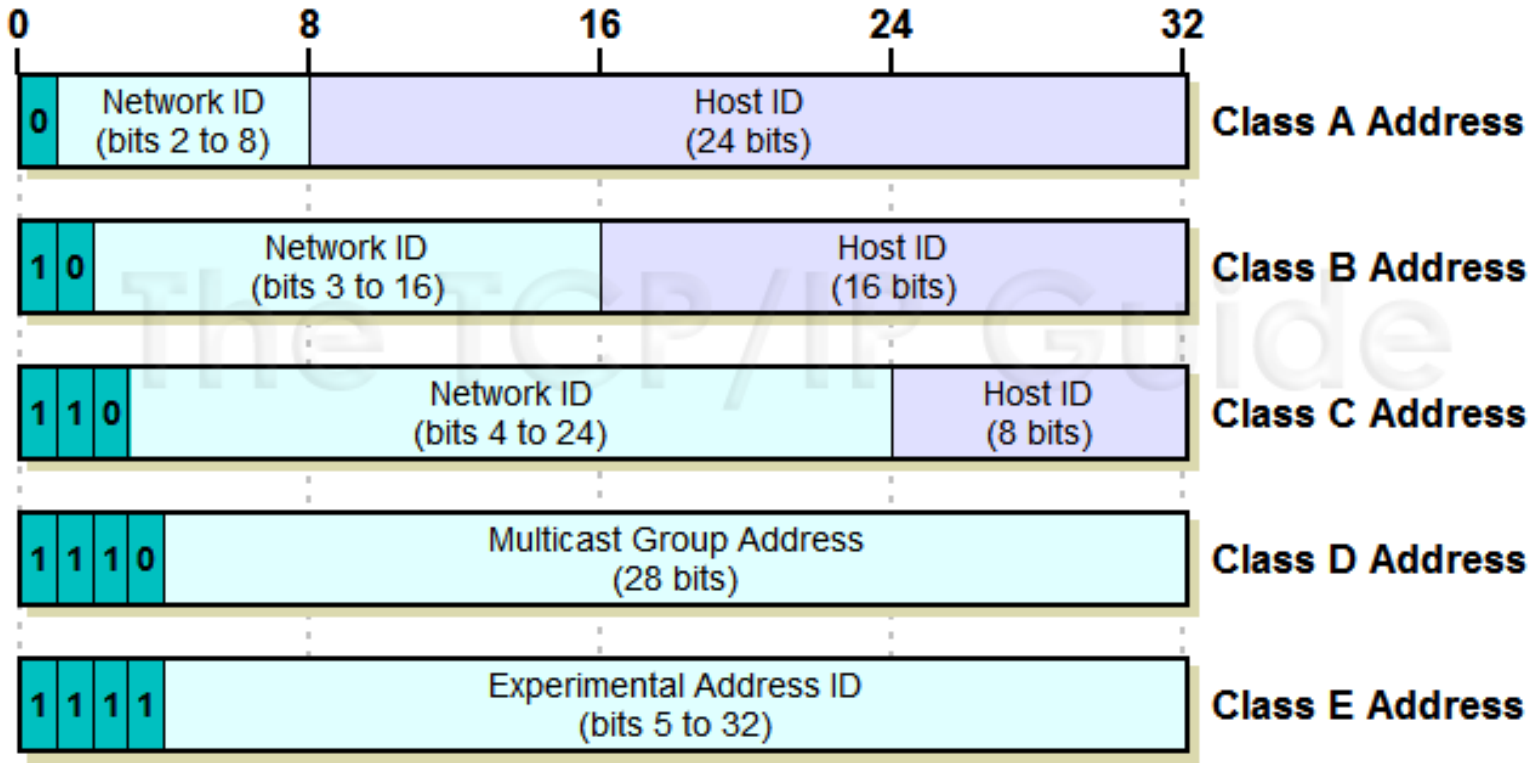
**RFC 791**

Please refer to RFC 791 for the complete Internet Protocol (IP) Specification.

# IPv4 Address Subnetting

- Routing by prefix requires all the hosts in the network to have the same network address (for example, for class B address 128.208.x.x, network address of all hosts will remain the same. There can be ~64k host addresses).
- This can cause problems as number of networks grows.
- A university started with a class B address for CS department with 64k potential hosts on the same network.
- Later, EE department wants to go on the internet and Art department follows soon.
- Getting further blocks of class B address for each department will be difficult (we are running out of IPv4 addresses). In addition, CS department has enough addresses for new hosts, but we cannot use these hosts for other departments (they have to be on the same network).
- IPv4 address subnetting allows splitting of a larger block of addresses into several parts for internal use as multiple networks.

# IPv4 Address Classes



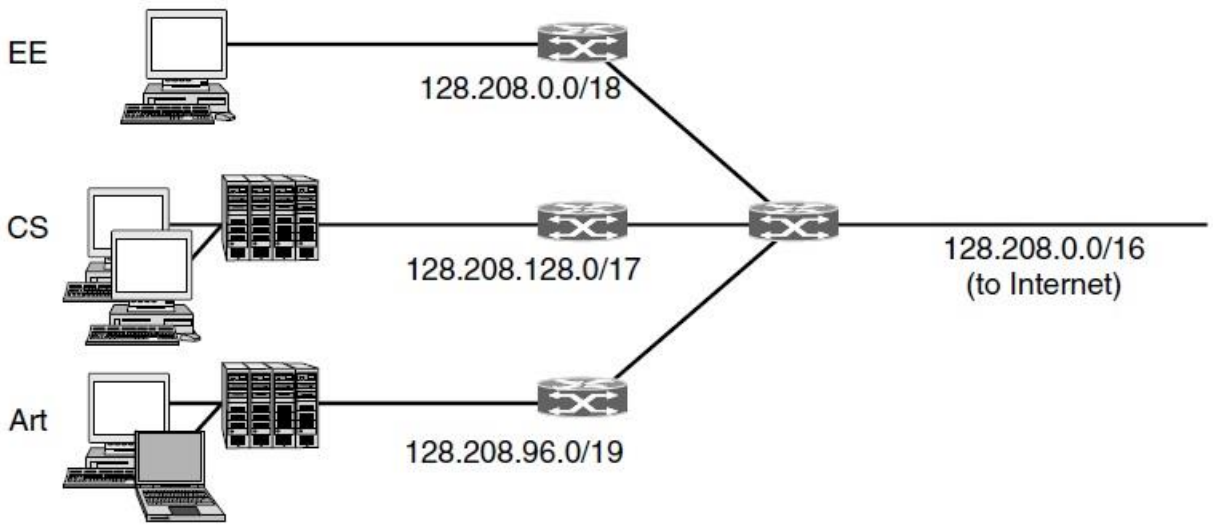

- Historically, IPv4 addresses are divided into 5 classes with pre-defined netmasks.
- For example, netmask of Class A IPv4 address is 255.0.0.0 (/8).
- 10.50.120.7 is a class A IP address (netmask=255.0.0.0 or /8)
- 172.16.55.13 is a class B IP address (netmask=255.255.0.0 or /16)

# IPv4 Address Subnetting

- University has a class B IP address allocated (128.208.x.x) (64k hosts)
- We can allocate half of the hosts to CS, quarter to EE and one eighth to Arts department.
- Netmasks can be represented as CIDR (Classless Interdomain Routing) notation. For example, 192.168.0.15/24 represents first 24 bits of the address are for network routing).

| Department | IP Address | CIDR/ Network Prefix | Range | Total number of hosts |
|---|---|---|---|---|
| CS | 1000 0000 – 1101 0000 – 1 \| xxx xxxx – xxxx xxxx | /17 128.208.128.0 | 128.208.128.0 – 128.208.255.255 | 32,768 |
| EE | 1000 0000 – 1101 0000 – 00 \| xx xxxx – xxxx xxxx | /18 128.208.0.0 | 128.208.0.0 – 128.208.63.255 | 16,384 |
| Arts | 1000 0000 – 1101 0000 – 011 \| x xxxx – xxxx xxxx | /19 128.208.96.0 | 128.208.96.0 – 128.208.127.255 | 8,192 |

# IPv4 Address Subnetting




- When a packet arrives from the internet, the router checks the destination IP address of the packet and finds out which subnet it belongs to.
- This is done by ANDing the address with the subnet mask and then checking to see if the result is the corresponding prefix.

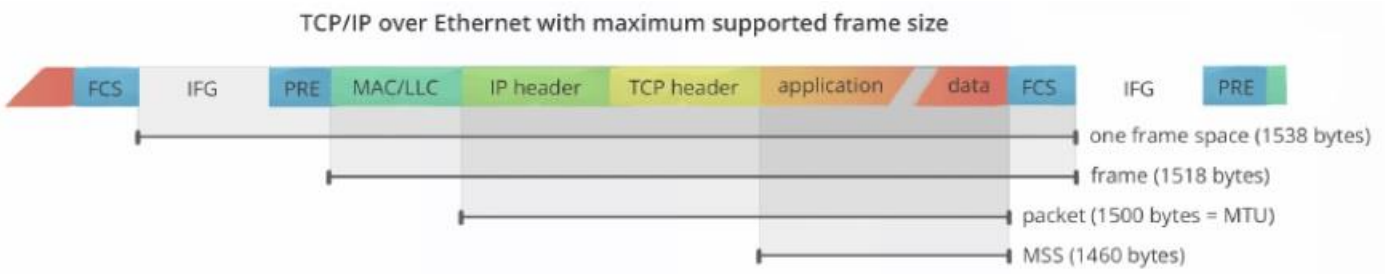
# IPv4 Address Subnetting

- For example, a packet arrives with destination IP address of 128.208.97.151.
- In order to find the subnet, AND operations is done using all three subnets.

- CS (/17):
  - (128.208.97.151) AND (255.255.128.0) = 128.208.0.0
  - This is not a CS prefix which is 128.208.128.0 – packet is not for CS network.
- EE (/18):
  - (128.208.97.151) AND (255.255.192.0) = 128.64.0.0
  - This is not an EE prefix which is 128.208.0.0 – packet is not for EE network.
- Arts (/19):
  - (128.208.97.151) AND (255.255.224.0) = 128.208.96.0
  - This results is Arts network prefix – packet is forwarded to Arts network.

# IPv4 MTU (Maximum Transmission Unit)

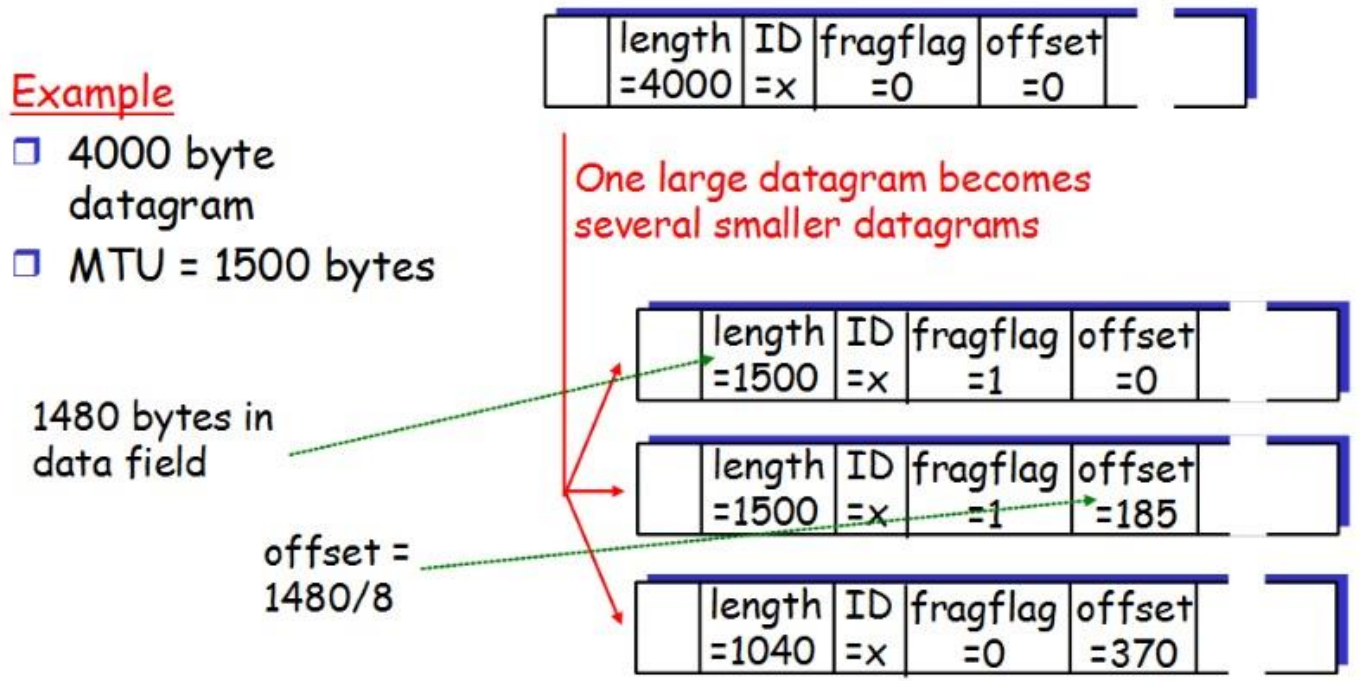
- The Maximum Transmission Unit (MTU) is the maximum length of data that can be transmitted by a protocol in one instance.
- MTU is usually associated with the Ethernet protocol, where a 1500-byte packet is the largest allowed.




- Packets needs to be fragmented for the case where higher-level protocols and applications try to send data which is greater than the MTU size.
- Path MTU Discovery protocol is a technique to determine the MTU on the network between to IP hosts (different hops between two hosts can have different MTU). Finding the MTU before transmitting a packet will avoid IP fragmentation by the intermediate routers on the way.

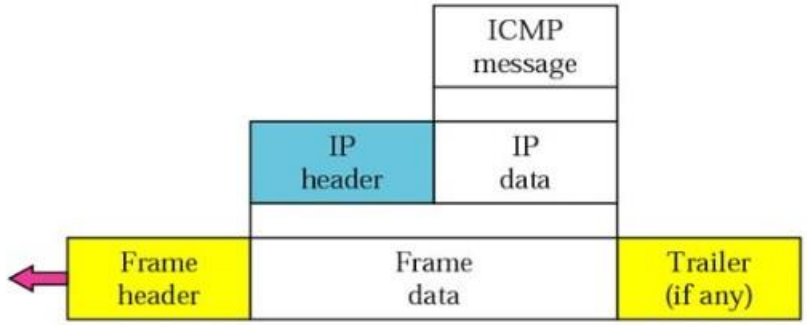
# IPv4 Fragmentation and Reassembly

- **IP fragmentation** is a process that breaks packets into smaller pieces (fragments), so that the resulting data can pass through a link with a smaller MTU than the original packet size.
- The fragments are reassembled by the receiving host.
- IP identification, frag offset and  M (more fragments) flags are used to handle IP reassembly.



# ICMP (Internet Control Messaging Protocol)

- IP protocol used to send error messages and operational information including success or failure when communicating with another IP device.
- ICMP packets are encapsulated within the IP packet.

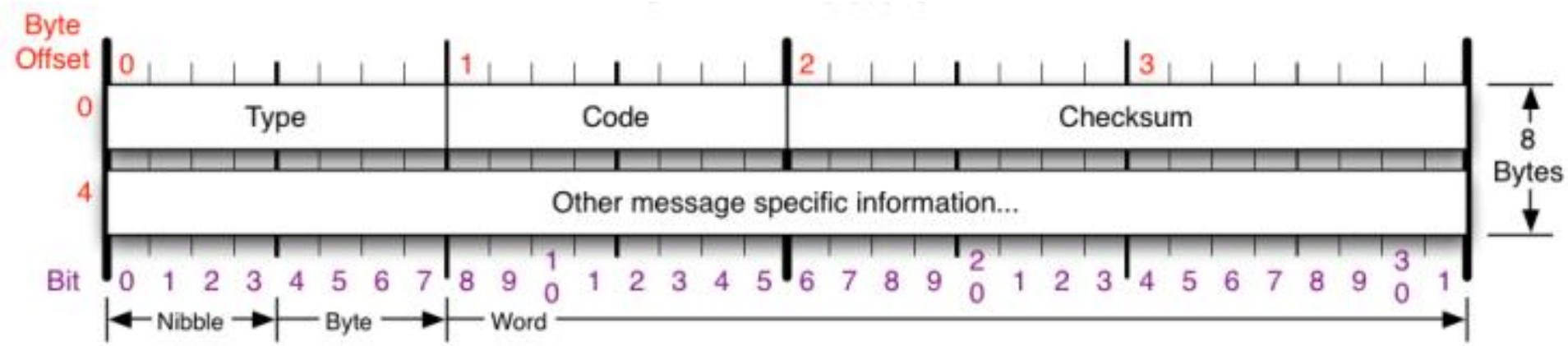



- Most common use of ICMP is with ping which is used to verify the reachability of a remote IP host.
- Some of the ICMP control messages are:
  - Destination unreachable.
  - TTL expired in transit.
  - Echo request and reply.

# Traceroute

- Traceroute is an application which uses ICMP protocol to display the route to a remote IP host. It also measures transit delays of packets within the IP network.
- The TTL value in IP header (also known as hop-limit) is used to determine intermediate routes being traversed towards the destination.
- Traceroute sends ICMP echo request packets with TTL value that gradually increase for each subsequent packet, starting from TTL of 1.
- Routers decrement the value of TTL, discarding the packet if it reaches 0. At this point, router sends back an ICMP error message (ICMP Time Exceeded) back to the sender.
- By getting an ICMP message back, the sender knows the IP address of each router within the route.
- The timestamp value is used to measure the latency.
- Eventually, the packet reaches the destination and sender received an ICMP echo reply message back.

# ICMP – header format




## ICMP Message Types

| Type | Code/Name |
|---|---|
| 0 | Echo Reply |
| 3 | Destination Unreachable |
| | 0 Net Unreachable |
| | 1 Host Unreachable |
| | 2 Protocol Unreachable |
| | 3 Port Unreachable |
| | 4 Fragmentation required, and DF set |
| | 5 Source Route Failed |
| | 6 Destination Network Unknown |
| | 7 Destination Host Unknown |
| | 8 Source Host Isolated |
| | 9 Network Administratively Prohibited |
| | 10 Host Administratively Prohibited |
| | 11 Network Unreachable for TOS |
| 3 | Destination Unreachable (continued) |
| | 12 Host Unreachable for TOS |
| | 13 Communication Administratively Prohibited |
| 4 | Source Quench |
| 5 | Redirect |
| | 0 Redirect Datagram for the Network |
| | 1 Redirect Datagram for the Host |
| | 2 Redirect Datagram for the TOS & Network |
| | 3 Redirect Datagram for the TOS & Host |
| 8 | Echo |
| 9 | Router Advertisement |
| 10 | Router Selection |
| 11 | Time Exceded |
| | 0 TTL Exceeded |
| | 1 Fragment Reassembly Time Exceeded |
| 12 | Parameter Problem |
| | 0 Pointer Problem |
| | 1 Missing a Required Operand |
| | 2 Bad Length |
| 13 | Timestamp |
| 14 | Timestamp Reply |
| 15 | Information Request |
| 16 | Information Reply |
| 17 | Address Mask Request |
| 18 | Address Mask Reply |
| 30 | Traceroute |

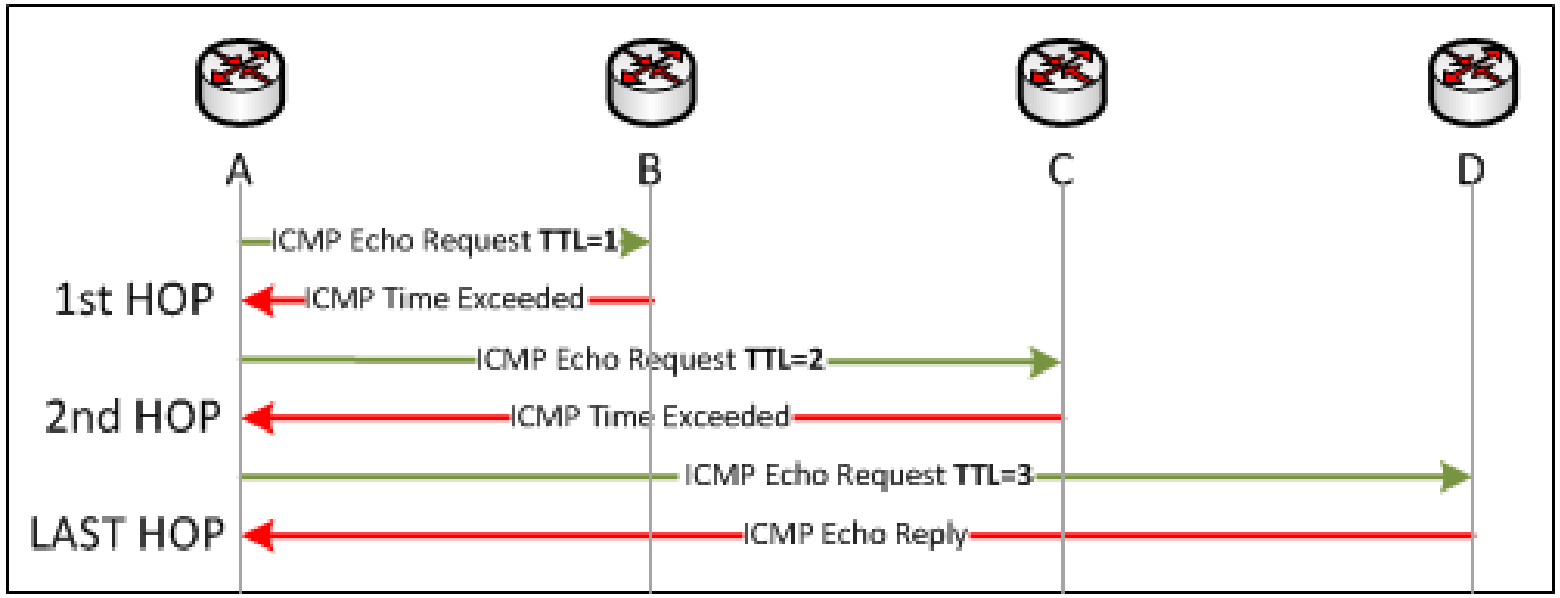
## Checksum

Checksum of ICMP header

## RFC 792

Please refer to RFC 792 for the Internet Control Message protocol (ICMP) specification.

# Traceroute

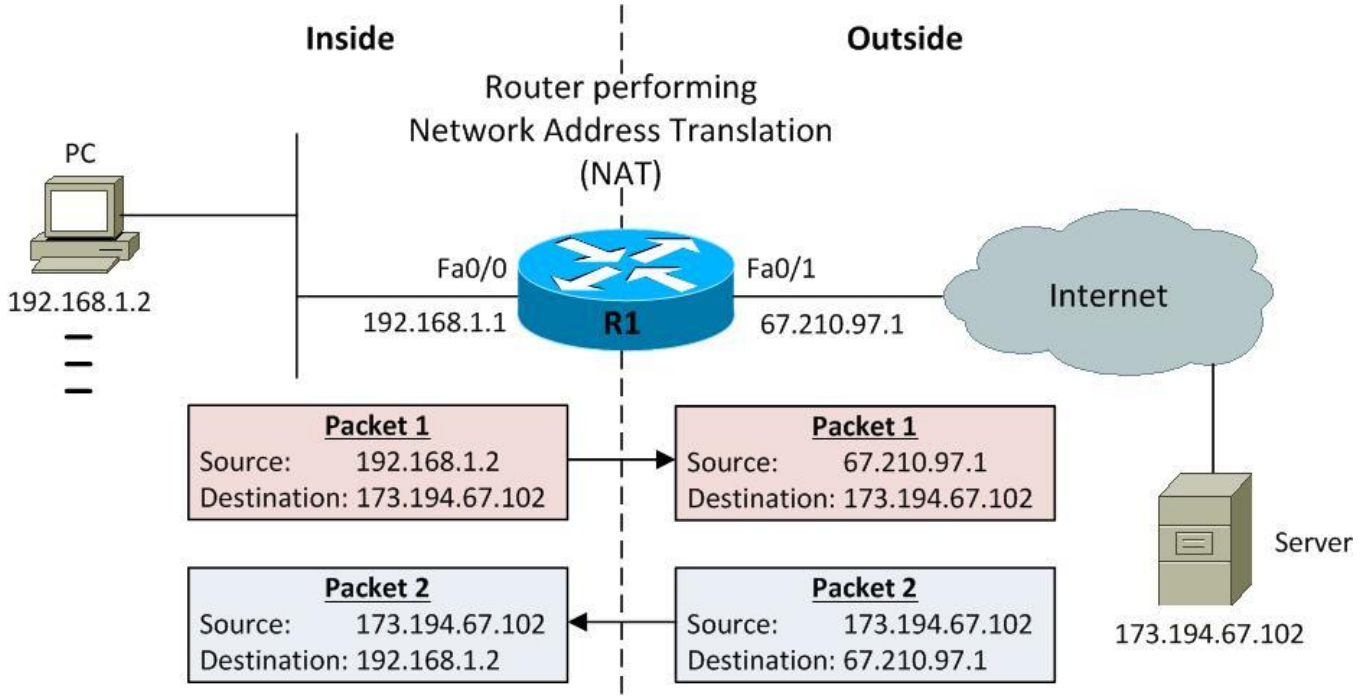

A
B
C
D
ICMP Echo Request TTL=1
1st HOP
ICMP Time Exceeded
ICMP Echo Request TTL=2
2nd HOP
ICMP Time Exceeded
ICMP Echo Request TTL=3
LAST HOP
ICMP Echo Reply

# NAT – Network Address Translation

- **Network Address Translation (NAT)** is a process in which one or more local IP addresses are translated into one or more Global IP addresses and vice versa in order to provide Internet access to the local hosts.
- Main use of NAT is to limit the number of public IP addresses an organization uses. IP addresses are limited resource and not every device can have a unique IP address.
- NAT also performs the translation of port numbers i.e. masks the port number of the host with another port number, in the packet that will be routed to the destination.
- It then makes the corresponding entries of IP address and port number in the NAT table.
- NAT usually operates on router or firewall.
- The working of the NAT is explained in the following video:
- NAT Explained - Network Address Translation

# How NAT Works

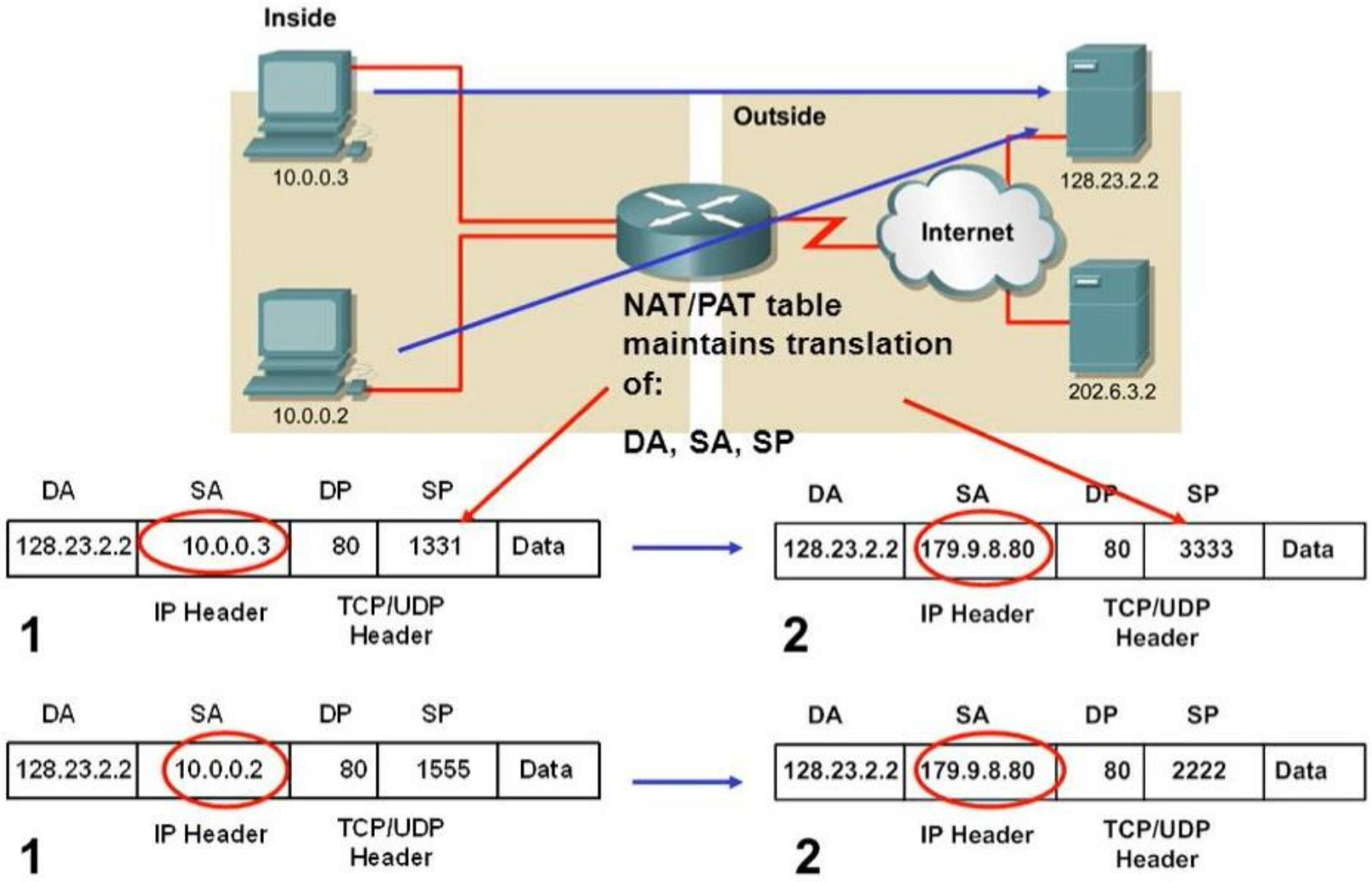
- When a packet traverse outside the local (inside) network, NAT converts that local (private) IP address to a global (public) IP address.
- When a packet enters the local network, the global (public) IP address is converted to a local (private) IP address.
- NAT look up for outside to inside packets is done using local port number.



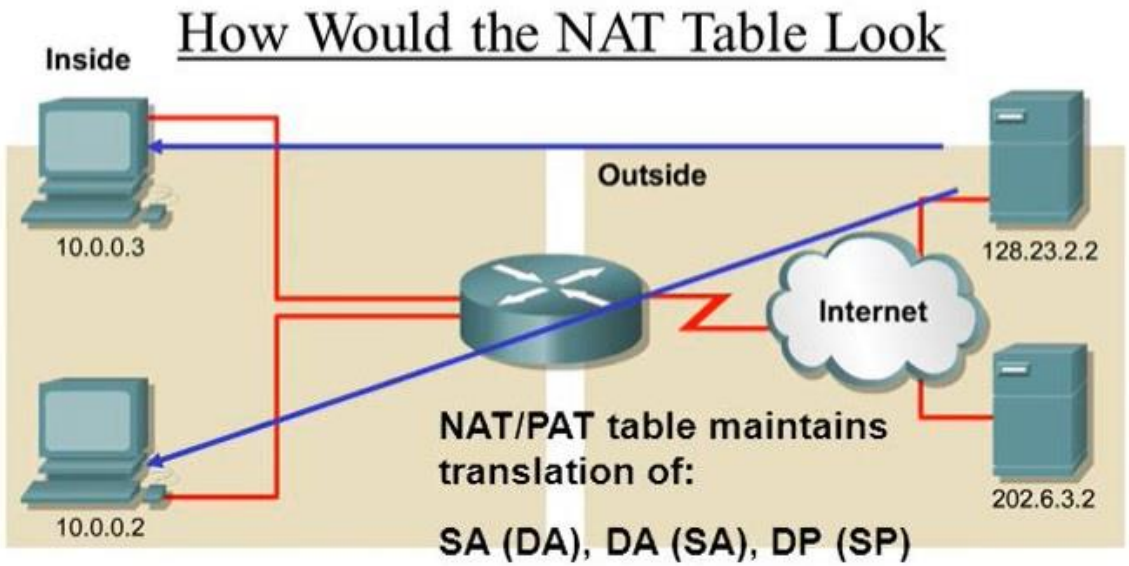
# PAT – Port Address Translation

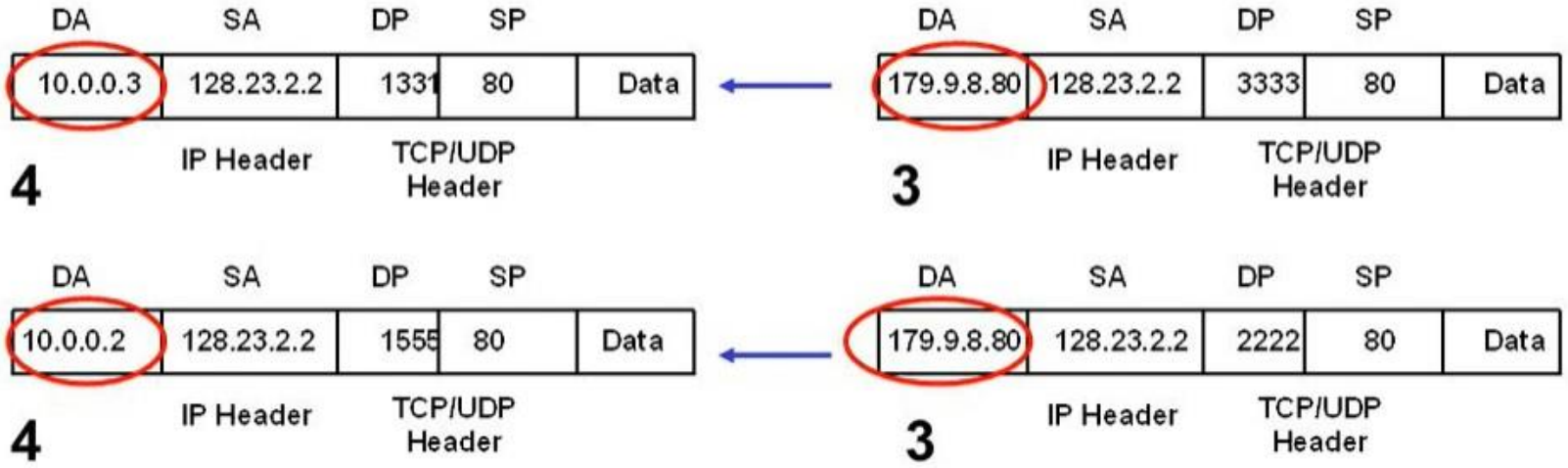
- **Port Address Translation (PAT)** is an extension of NAT where both Address and port numbers are translated.
- Here is a scenario where PAT is needed:
  - There are two hosts A and B connected to a router.
  - Both request for the same destination, on the same port number, say 1000, on the host side, at the same time.
  - If NAT does only IP address translation, then when their packets will arrive at the NAT, both of their IP addresses would be translated to the public IP address of the network and sent to the destination.
  - Destination will send replies on the public IP address of the router.
  - On receiving a reply, it will be unclear to NAT as to which reply belongs to which host (because source port numbers for both A and B are same). Hence, to avoid such a problem, NAT masks the source port number as well and makes an entry in the NAT table.

# PAT – Port Address Translation



# PAT – Port Address Translation

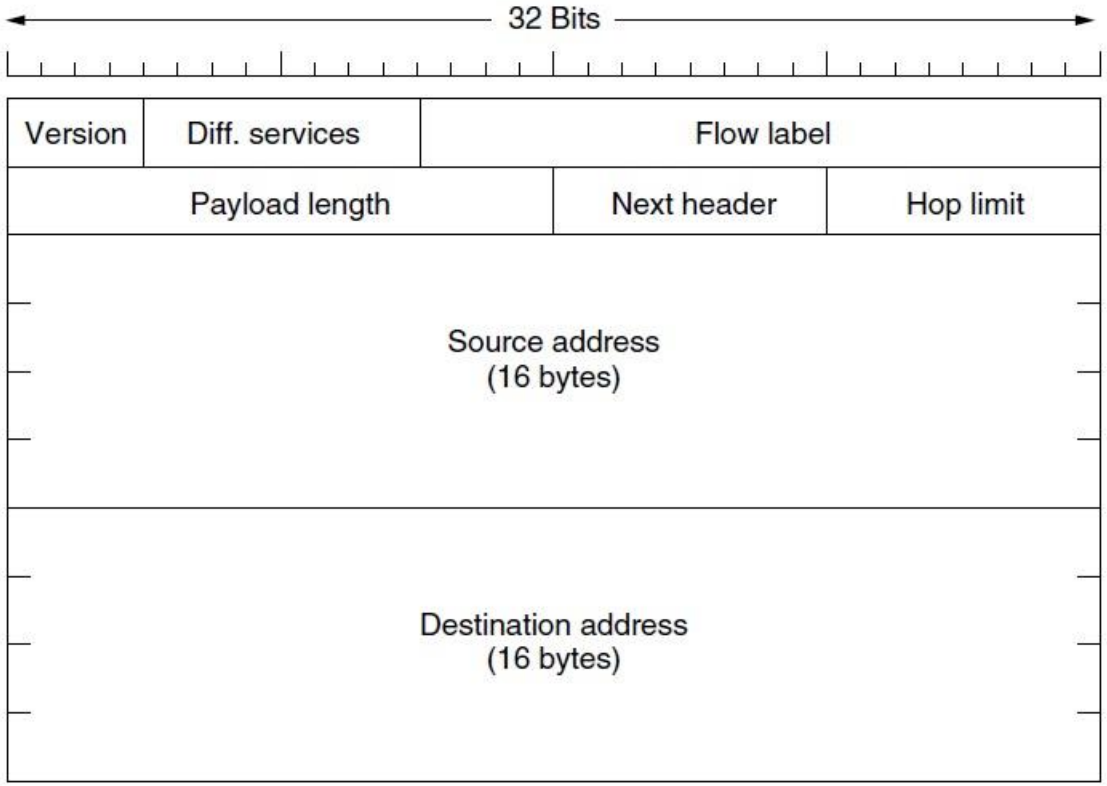


# IPv6

- We are running out of IPv4 addresses (addresses are only 32-bit).
- A new format was needed which has much more addresses than IPv4. IPv6 IP addresses are 128 bit wide.
- IPv6 header has only 7 fields compared to 13 in IPv4. This allows routers to process packets faster.

# IPv6 – Header Format




- **Version:** Indicates the version of IP packet (6 for IPv6)
- **Diff services (also known as class of service):** Distinguishes the class of service for packet with different real time delivery requirements.

# IPv6 – Header Format

- **Flow Label:** Used to identify all packets in a single flow (packets sent from a source device to one of more destination devices). Routers use the flow label to ensure that all packets are treated the same.
- **Payload Length:** Number of bytes follow the IPv6 header. It is different from IPv4 and doesn't include the header length.
- **Next Header:** Specifies the next header. When there are extension headers, this field identifies the first extension header. In case of no extension headers, this field indicates the transport protocol being used.
- **Hop Limit:** Same as TTL in IPv4.
- **Source Address:** 128-bit source IP address.
- **Destination Address:** 128-bit destination IP address.

# IPv6 – Address Notation

- IPv6 address is written as eight groups of four hex digits with colons between the groups.
- Example of IPv6 address:

  1029:9183:0000:0000:0000:0AC1:2143:019B

- Since IPv6 addresses and long and have many zeroes, few techniques are used to make them easier to read. These are called Zero Suppression and Zero Compression.
  - Zero Suppression: we can remove leading zeros from a 16-bit number. Each 16-bit boundary must have at-least one number.

    1029:9183:0:0:0:0AC1:2143:019B

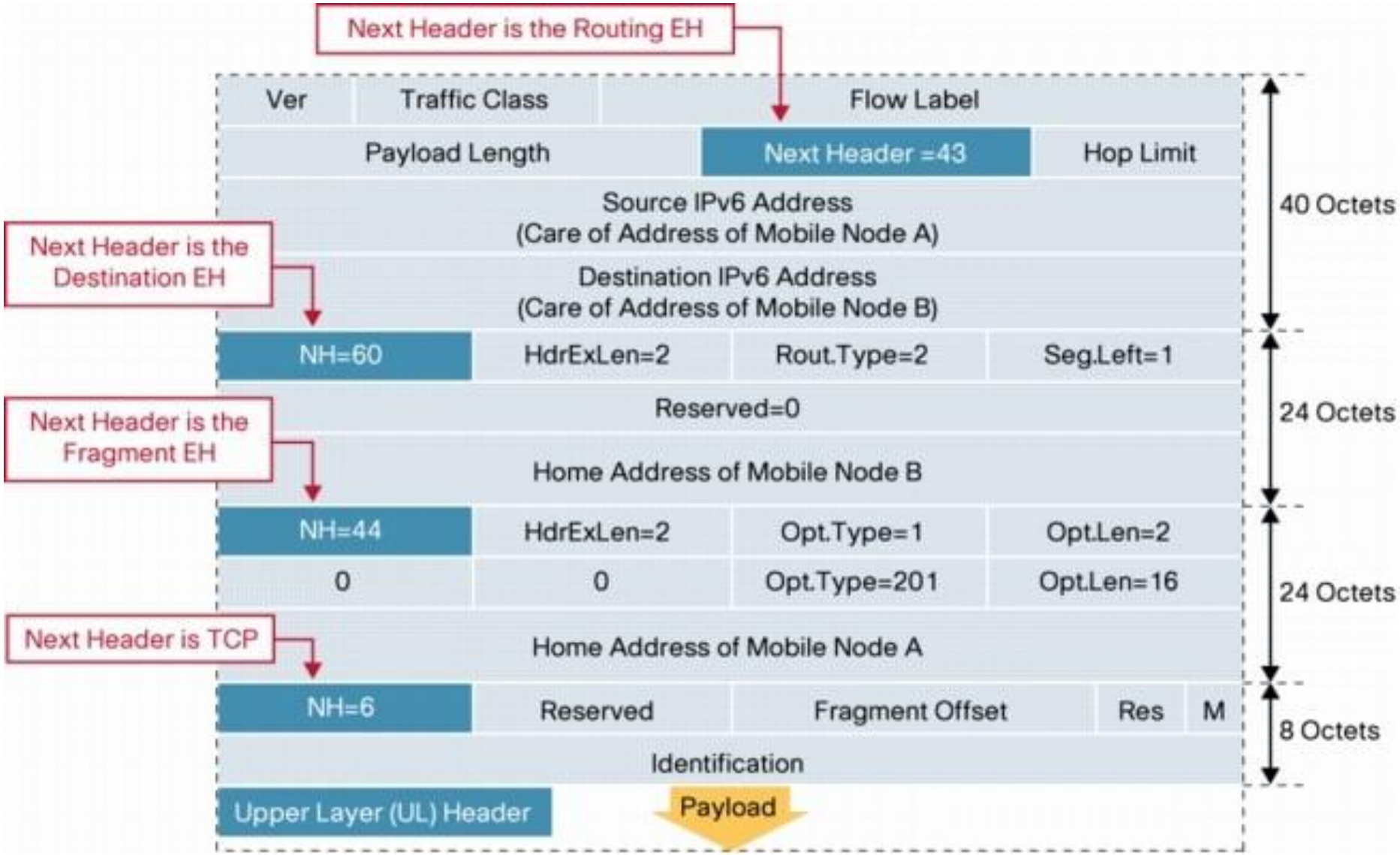
  - Zero Compression: contiguous sequence of 16-bit blocks with a value of zero can be compressed to "::" but this process can only be used once for an address:

    1029:9183::0AC1:2143:019B

# IPv6 – Extension Headers

| Next Header Value (decimal) | Extension Header Name | Length (bytes) | Description | Defining RFC |
|---|---|---|---|---|
| **0** | ***Hop-By-Hop Options*** | Variable | Defines an arbitrary set of options that are intended to be examined by all devices on the path from the source to destination device(s).<br><br>This is one of two extension headers used to define variable-format options. | 2460 |
| **43** | ***Routing*** | Variable | Defines a method for allowing a source device to specify the route for a datagram. This header type actually allows the definition of multiple routing types. The IPv6 standard defines the Type 0 *Routing* extension header, which is equivalent to the "loose" source routing option in IPv4 and used in a similar way.<br><br>See below for the format of this extension header. | 2460 |
| **44** | ***Fragment*** | 8 | When a datagram contains only a fragment of the original message, this extension header is included. It contains the *Fragment Offset*, *Identification* and *More Fragment* fields that were removed from the main header.<br><br>See below for the format of this extension header, and the topic on fragmentation and reassembly for details on how the fields are used. | 2460 |
| **50** | ***Encapsulating Security Payload (ESP)*** | Variable | Carries encrypted data for secure communications. This header is described in detail in the section on IPSec. | 2406 |
| **51** | ***Authentication Header (AH)*** | Variable | Contains information used to verify the authenticity of encrypted data. This header is described in detail in the section on IPSec. | 2402 |
| **60** | ***Destination Options*** | Variable | Defines an arbitrary set of options that are intended to be examined only by the destination(s) of the datagram.<br><br>This is one of two extension headers used to define variable-format options. | 2460 |

# IPv6 – Extension Headers



# IPv6 – Neighbor Discovery

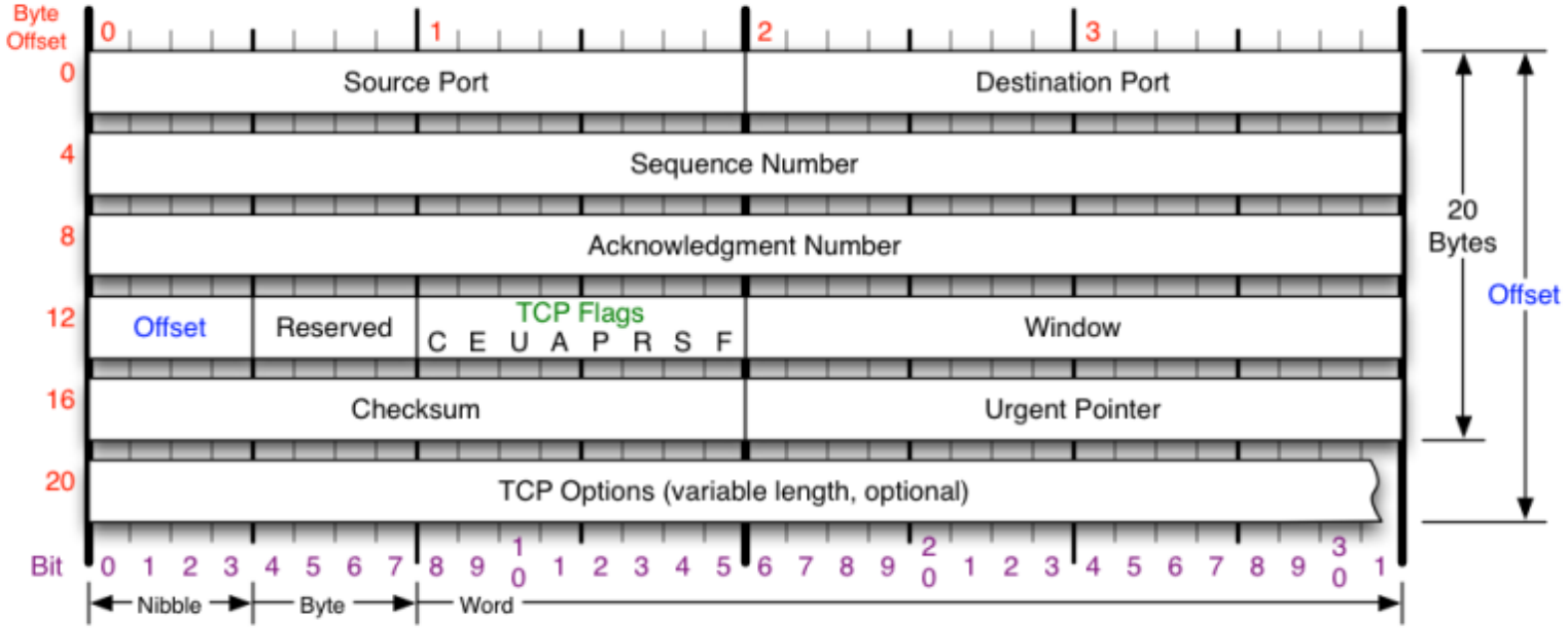
- Set of messages and processes that determine relationships between neighboring nodes.
  - Replaces ARP, ICMPv4, Router Discovery.
- ND is used for:
  - Address Resolution.
  - To determine link layer address changes.
  - To determine neighbor reachability.
- ICMPv6 and multicast is used for ND messages.
- There are five type of ND messages:
  1. Router Solicitation
  2. Router Advertisement
  3. Neighbor Solicitation
  4. Neighbor Advertisement
  5. Redirect

# Transport Layer (TCP/UDP)

# TCP - Transmission Control Protocol (RFC 793)

- TCP provides reliable, ordered and error-checked streams of bytes between two IP hosts.
- It is a connection-oriented protocol – a connection between a server and client needs to be established before transferring any data.
- Sending and receiving entities exchange data in the form of segments.
- Each segment has a unique sequence number.
- A TCP segment consists of fixed 20-byte header (plus an optional part) followed by zero or more data bytes.
- Two limits restrict the TCP segment size:
  1. Each segment including TCP header must fit in 65,515 bytes IP payload.
  2. Each link has an MTU (Maximum Transmit Unit). Each segment must fit in the MTU at the sender and receiver.
- A sliding window protocol is used for flow control between a sender and a receiver.
- A TCP connection is identified by 5-tuple (protocol, src IP, src port, dst IP, dst port).

# TCP – Header Format




### TCP Flags

C E U A P R S F

Congestion Window

C 0x80 Reduced (CWR)
E 0x40 ECN Echo (ECE)
U 0x20 Urgent
A 0x10 Ack
P 0x08 Push
R 0x04 Reset
S 0x02 Syn
F 0x01 Fin

### Congestion Notification

ECN (Explicit Congestion Notification). See RFC 3168 for full details, valid states below.

| Packet State | DSB | ECN bits |
|---|---|---|
| Syn | 0 0 | 1 1 |
| Syn-Ack | 0 0 | 0 1 |
| Ack | 0 1 | 0 0 |
| No Congestion | 0 1 | 0 0 |
| No Congestion | 1 0 | 0 0 |
| Congestion | 1 1 | 0 0 |
| Receiver Response | 1 1 | 0 1 |
| Sender Response | 1 1 | 1 1 |

### TCP Options

0 End of Options List
1 No Operation (NOP, Pad)
2 Maximum segment size
3 Window Scale
4 Selective ACK ok
8 Timestamp

### Checksum

Checksum of entire TCP segment and pseudo header (parts of IP header)

### Offset

Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.

### RFC 793

Please refer to RFC 793 for the complete Transmission Control Protocol (TCP) Specification.

# TCP – Header Format

- **Source and Destination ports:** Define the endpoints of a connection.
- **Sequence and Acknowledge numbers:** Indicates the sequence number of data being sent and how much data has been received. Both of these are cumulative.
- **Offset/TCP header length:** Number of 32-bit words in TCP header. The header size can be variable if TCP options are used.
- **TCP flags:**
  - ACK – indicates that the Acknowledgement number is valid.
  - PSH – the receiver is requesting to deliver to data to the application without waiting for the receive to become full.
  - RST – reset a TCP connection.
  - SYN – used during connection establishment.
  - FIN – used during connection teardown.
- **Windows:** Used for flow control. Indicates how many bytes a remote entity send starting at the Acknowledgment number. Window size of 0 indicates that the receiver cannot receive any data.

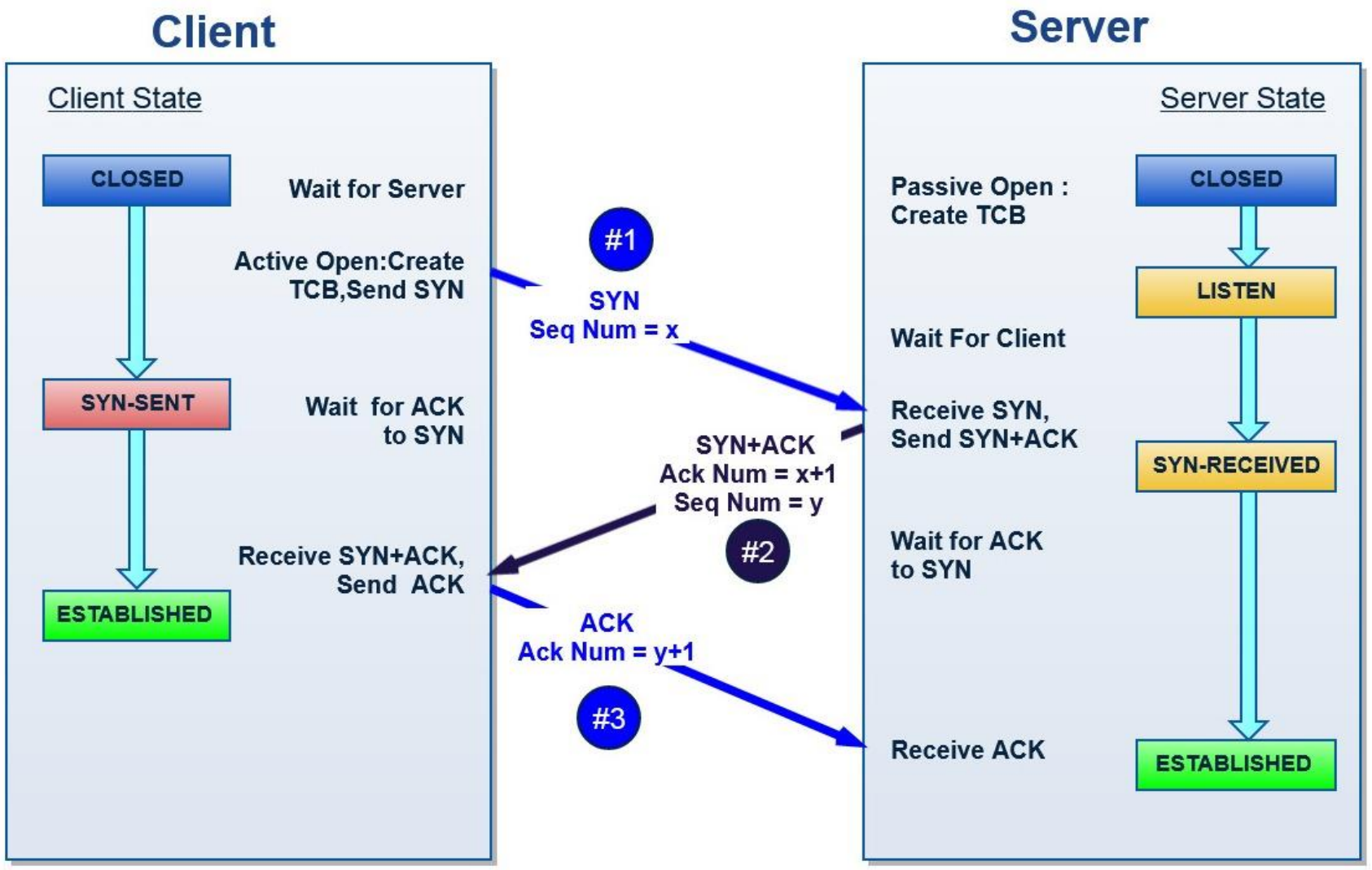
# TCP – Header Format

- **Checksum:** Checksum value covering both header and data. Only source address, destination address, protocol and TCP length are included from the TCP header (also called Pseudo Header).
- **Urgent Pointer:** Indicates a byte offset from the current sequence number at which urgent data can be found.
- **TCP Options (optional):**
  1. MSS – indicates what is the Maximum Segment Size a host is willing to accept.
  2. Window Scale – allows both sender and receiver to negotiate a scale factor to be used for a larger window size (by default, max window size is 64k).
  3. Timestamp – carries a timestamp sent by the sender and echoed by the receiver. It is used to compute roundtrip time samples.
  4. PAWS (Protection Against Wrapped Sequence Numbers) - on a faster link, sequence numbers could wrap quickly, leading to possible confusion between old and new data. PAWS discards arriving segments with old timestamps.
  5. SACK (Selective Acknowledgment) – allows Ack a range of received bytes.

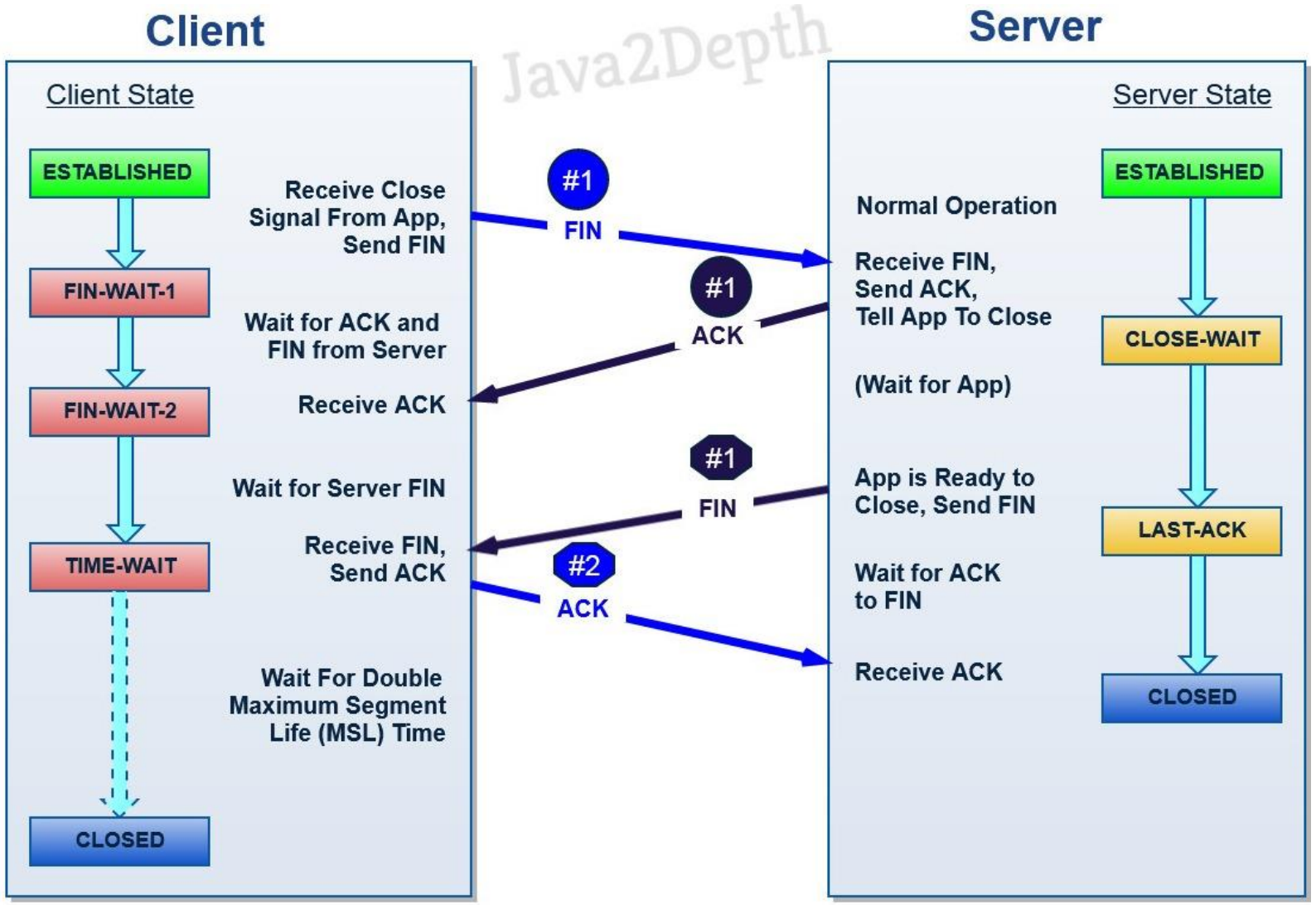
# TCP – Connection Establishment / Teardown

- The connection establishment and teardown process is explained in the following video:
- **TCP-Connection establishment and Teardown**

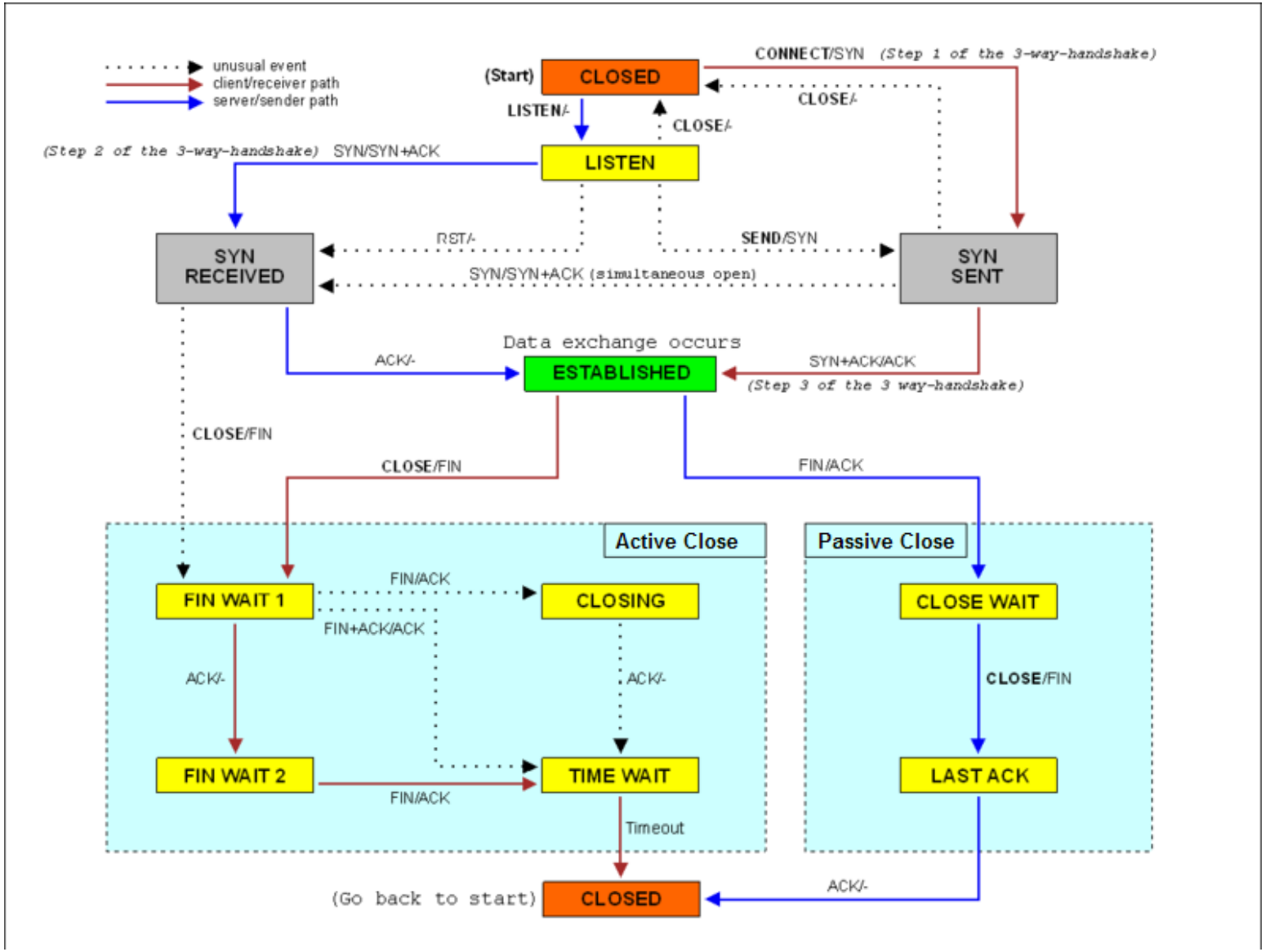
# TCP – Connection Establishment



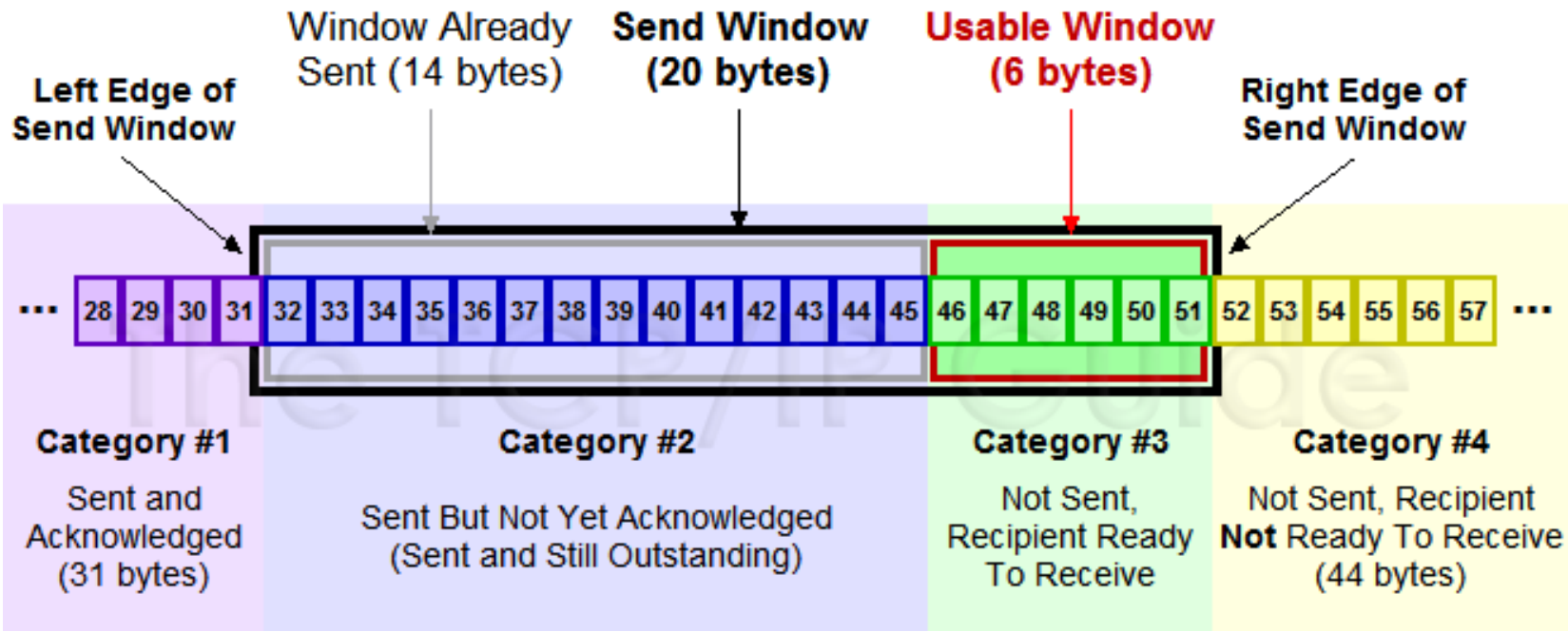
# TCP – Connection Teardown



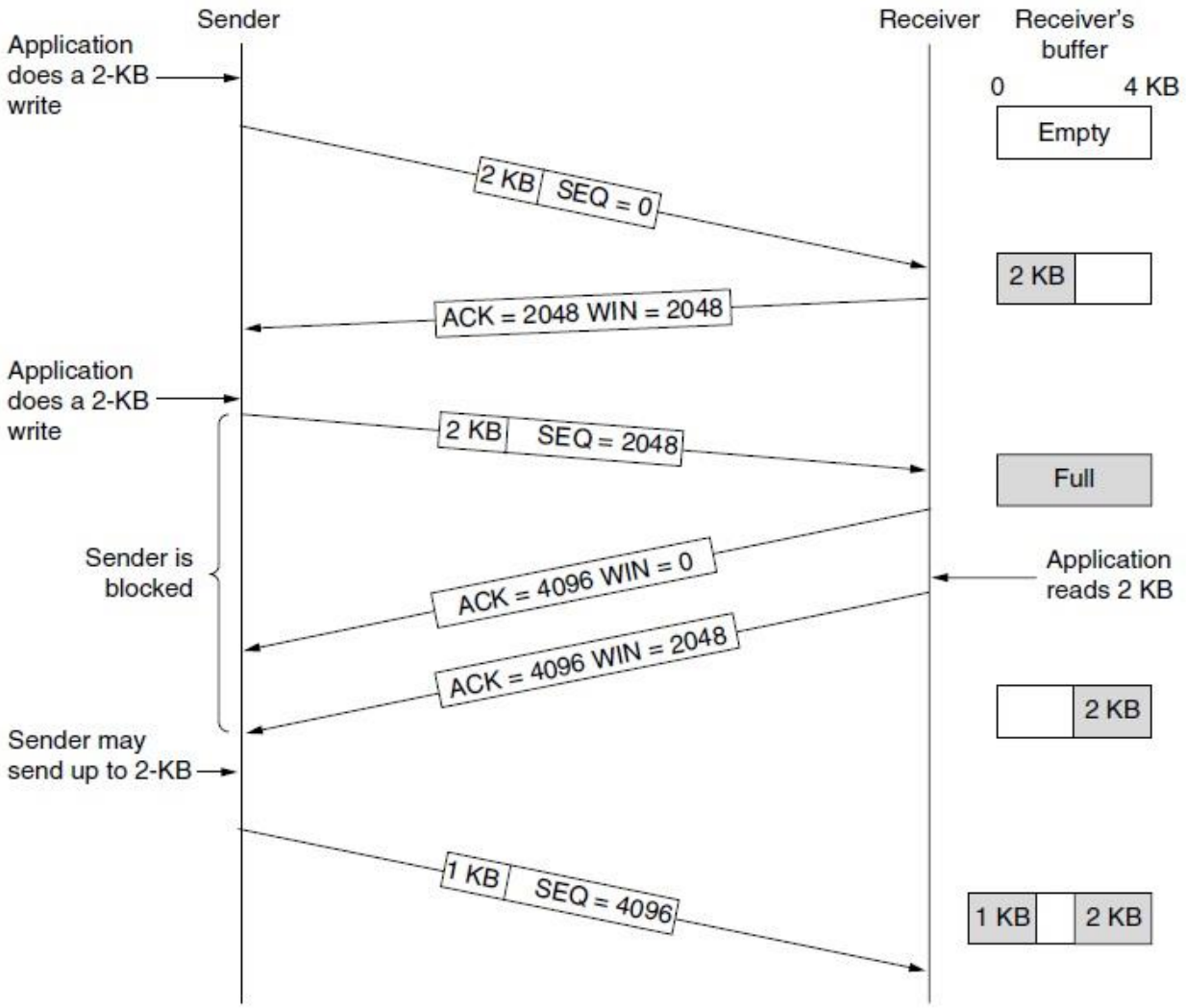
# TCP – State Diagram



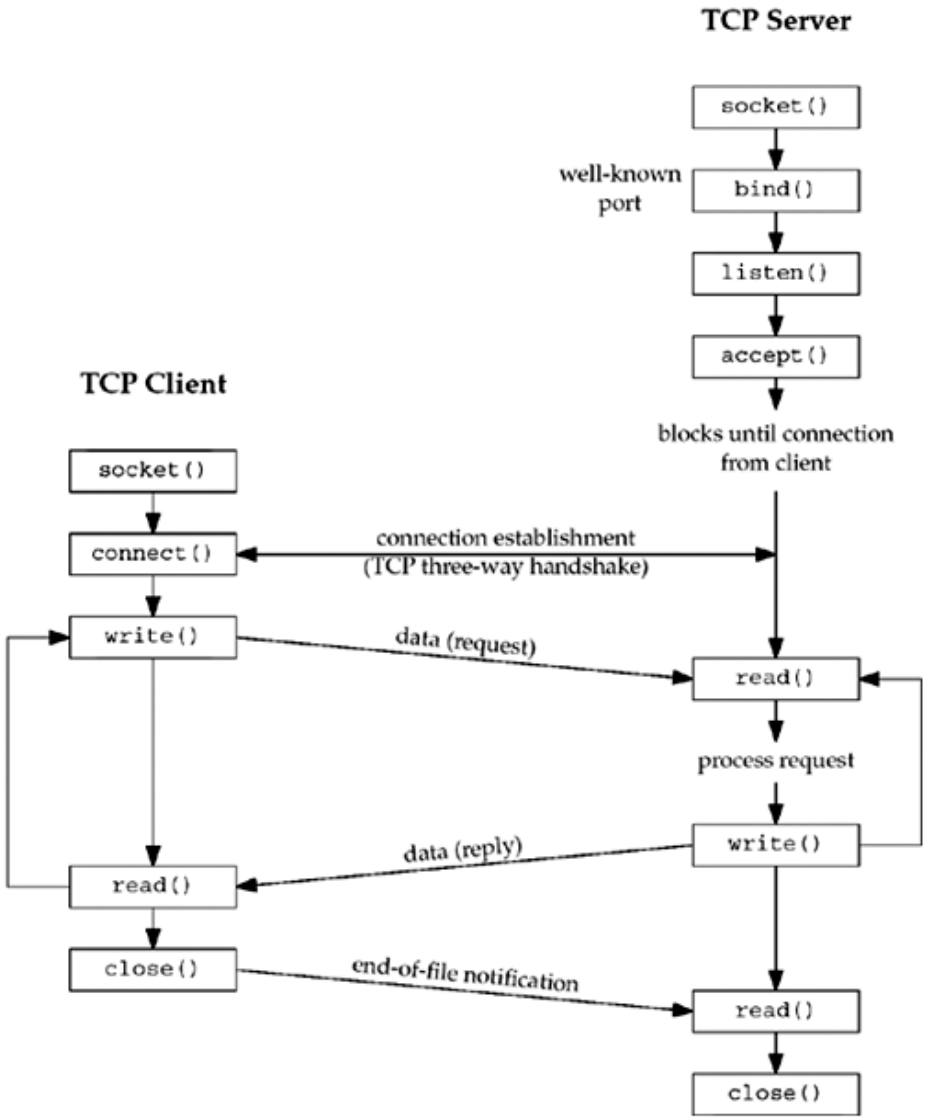
# TCP – Sliding Window



# TCP – Sliding Window Example



# TCP – Socket Programing



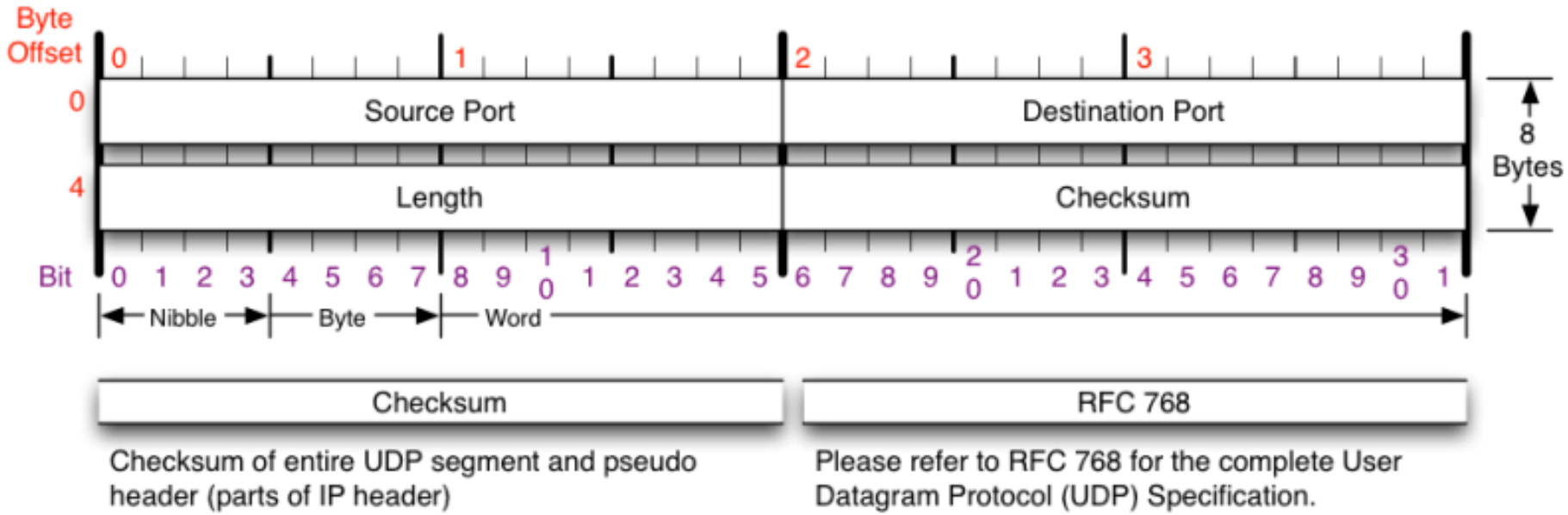
# TCP – Advanced Topics

- TCP timers
- Delayed Acknowledgement and Nagle's Algorithm
- TCP congestion control (slow start, fast recovery)
- Selective ACK

# UDP – User Datagram Protocol (RFC 768)

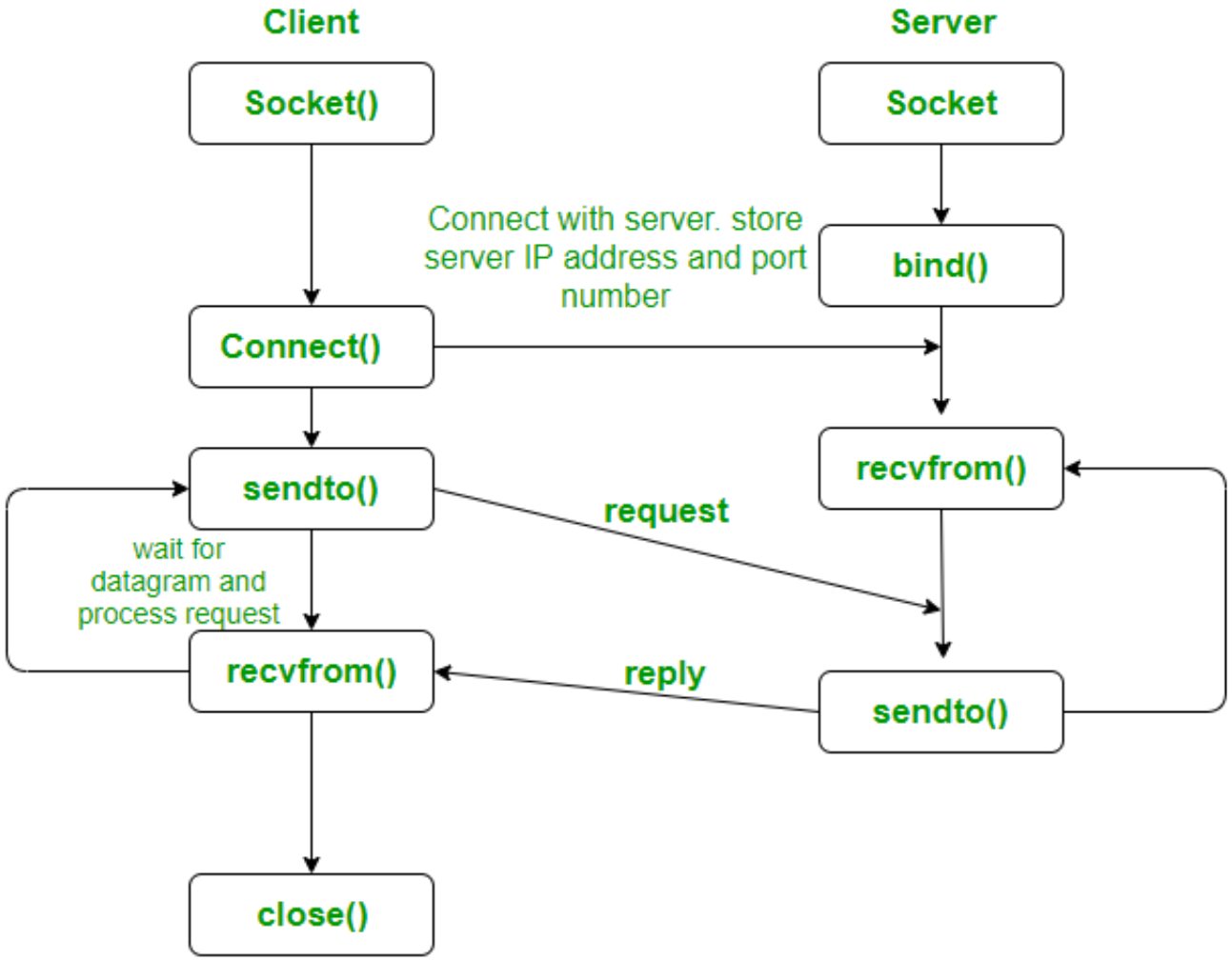
- UDP is a connectionless protocol – no additional handshake for connection establishment.
- Ordering and guaranteed data delivery is not supported by UDP. It is the responsibility of the application using the UDP to implement additional logic to handle packet loss.
- UDP is used for application which are time sensitive but can tolerate data loss, for example audio and video.
- UDP segments consists of 8 bytes of header followed by the payload.

# UDP – Header Format




- Source and destination ports.
- Length – length in bytes of UDP header and payload.
- Checksum (optional) – checksum of UDP header, payload and IP pseudo header (src IP, dst IP, protocol and length).

# UDP – Socket Programing



# TCP vs UDP

| TCP | UDP |
|---|---|
| TCP is a connection-oriented protocol | UDP is a connectionless datagram protocol |
| TCP is reliable as it guarantees delivery of data to the destination. | The delivery of data to the destination cannot be guaranteed in UDP. |
| TCP provides extensive error checking mechanisms through flow control and acknowledgment of data. | UDP has only the basic error checking mechanism using checksums. |
| Sequencing of data is a feature of TCP. | There is no sequencing of data in UDP. If ordering is required, it has to be managed by the application layer. |
| TCP is comparatively slower than UDP. | UDP is faster, simpler and more efficient than TCP. |
| Retransmission of lost packets is possible. | No retransmission of lost packets. |
| TCP is heavy-weight. | UDP is lightweight. |
| TCP doesn’t support Broadcasting. | UDP supports Broadcasting. |

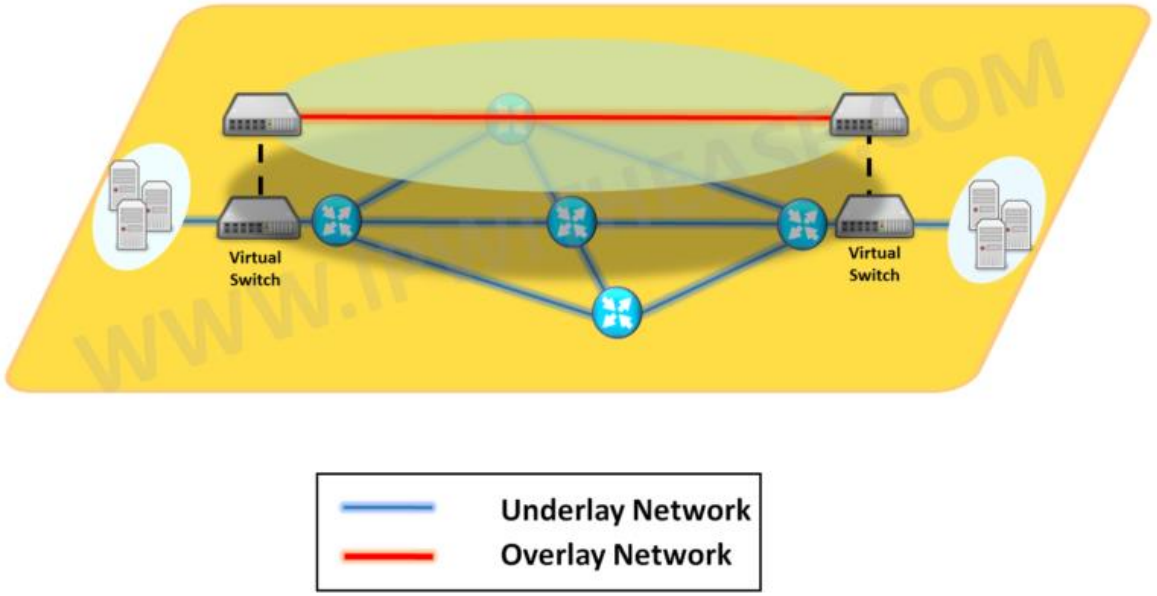
# Application Layer

# Application Layer

- DNS (Domain Name System):
  - Phonebook of internet. Translates domain names to IP addresses.
  - Uses both TCP and UDP
- DHCP (Dynamic Host Configuration Protocol):
  - Dynamically assigns IP address and other network configuration to a device.
  - It uses UDP.
- FTP and TFTP (Trivial/File Transfer Protocol):
  - Used for transferring files between a client and server.
  - FTP uses TCP while TFTP uses UDP.
- Telnet:
  - Provides bi-directional interactive text-oriented communication facility using a terminal connection.
  - It uses TCP.

# Overlay Networks

- A virtual network which is built on top of an existing physical network
- Physical Network is the underlay, and the Virtual Network on top of the physical network is the overlay.
- Overlays are useful because we can hide the complexities of the underlay and created a simplified network for our service.
- An example of a service that operates on an overlay is Voice-Over-IP (VoIP). VoIP uses the infrastructure of the Internet as the underlay, while the overlay is the virtual network of phone numbers used to address each phone.



# Why Overlay Networks?

- Add missing functionality without a complete network redesign.
- Enables separation of the virtual network configuration and topologies from the physical networks underneath. Data centers have rapidly increased their server virtualization over the past decade, resulting in dramatic increases in agility and flexibility. Virtualization of the network and decoupling the virtual network from the physical network makes it easier to manage, automate, and orchestrate. Overlay networks allows this to happen.
- Used for experimental protocols and services without changing the underlying network infrastructure.
- Intelligent software entities are implemented at each end of an overlay network tunnel which are responsible for higher layer traffic policing while the underlying network is responsible for moving packets as fast as possible.
- Packets move hop-by-hop between software entities.

# VLAN – A native Layer 2 Overlay

- VLANs have been around for years!
- A LAN is defined as a single broadcast domain.
- 12-bit VLAN ID is used to identify a VLAN (total 4096 VLANs).
- Each VLAN has its own set of forwarding rules, which separate it from other VLANs and provides the control over who can connect to the VM.

- Major problem with VLAN is that it can only support up to 4096 VLANs. In data centers, a lot more than 4096 VLANs are needed.

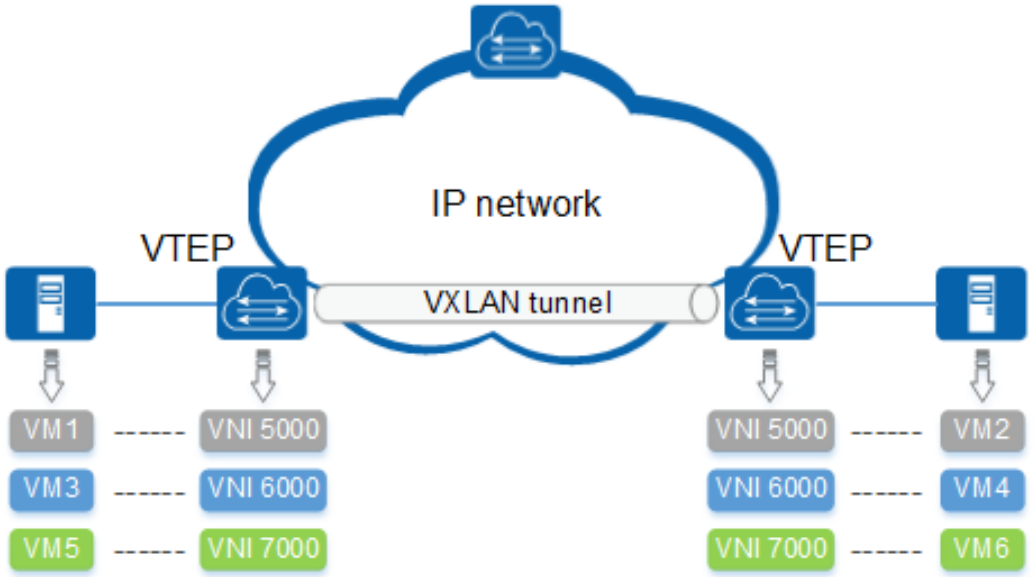
# Modern Overlay Protocols

- There are number of Overlay Protocols used to overcome VLAN deficiencies.
- More notable are:

1. VXLAN (Virtual Extensible LAN)
2. NVGRE (Network Virtualization using Generic Routing Encapsulation)
3. GENEVE (Generic Network Virtualization Encapsulation)

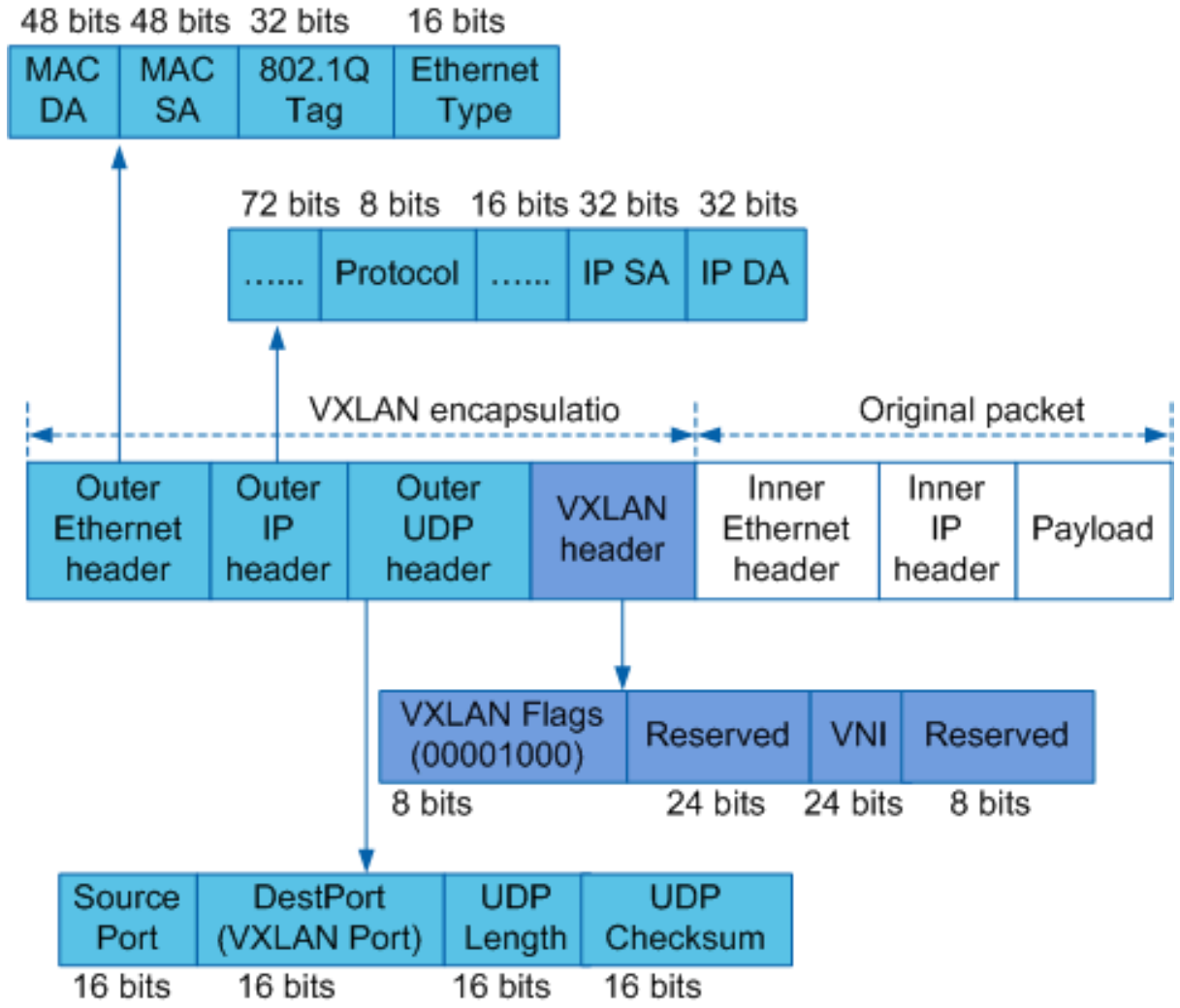
# VXLAN (RFC 7348)

- VXLAN tunnels layer 2 frames inside of L4 UDP datagrams.
- Each Layer 2 subnet is uniquely identified by a VXLAN network identifier (VNI) that segments traffic.
- VNI is 24-bit long which enables up to 16 million VXLAN segments (a lot more than VLAN)
- The entity that performs the encapsulation and decapsulation of packets is called a VXLAN tunnel endpoint (VTEP) and typically resides in hypervisor hosts.

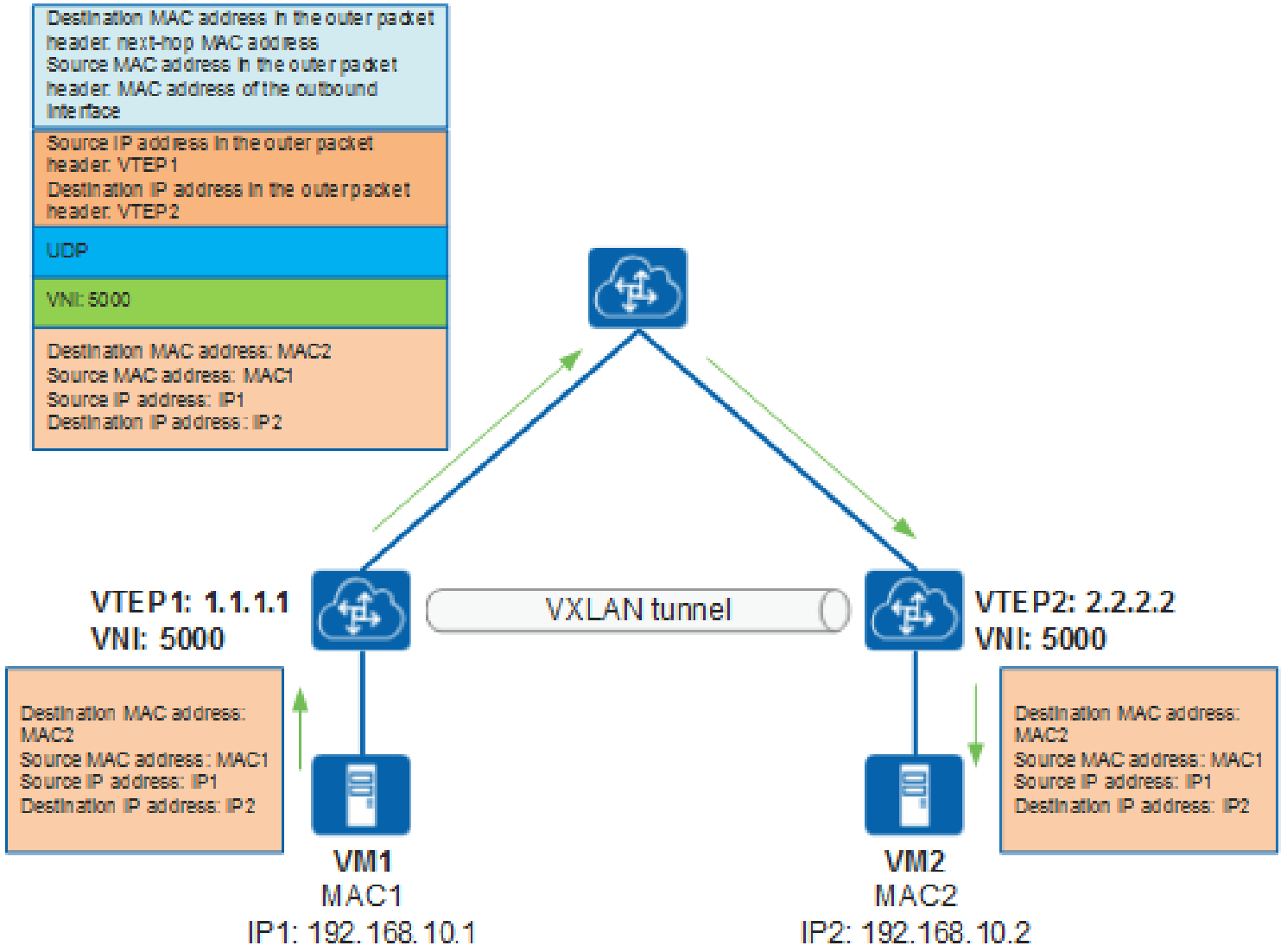


# VXLAN Encapsulation

- Original packet (Ethernet header/IP header and payload) is encapsulated inside a UDP packet.
- Outer Ethernet and IP headers are used for packet routing.
- VTEP at source adds encapsulation while another VTEP at destination removes this encapsulation.



# VXLAN Packet Forwarding

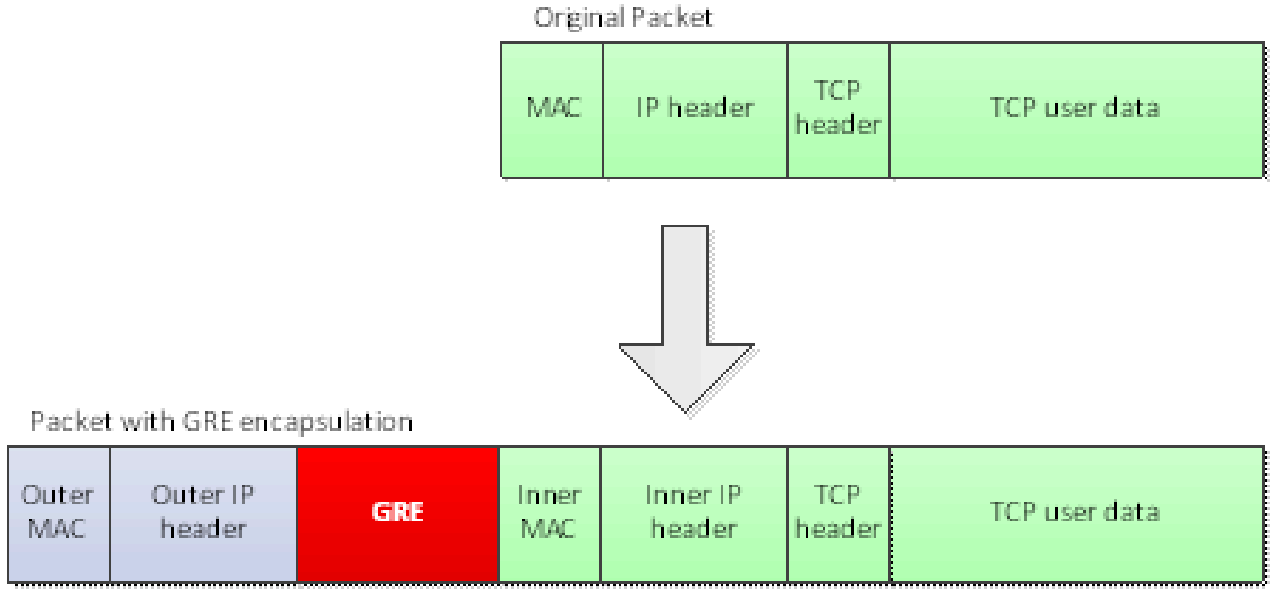


# VXLAN Packet Forwarding

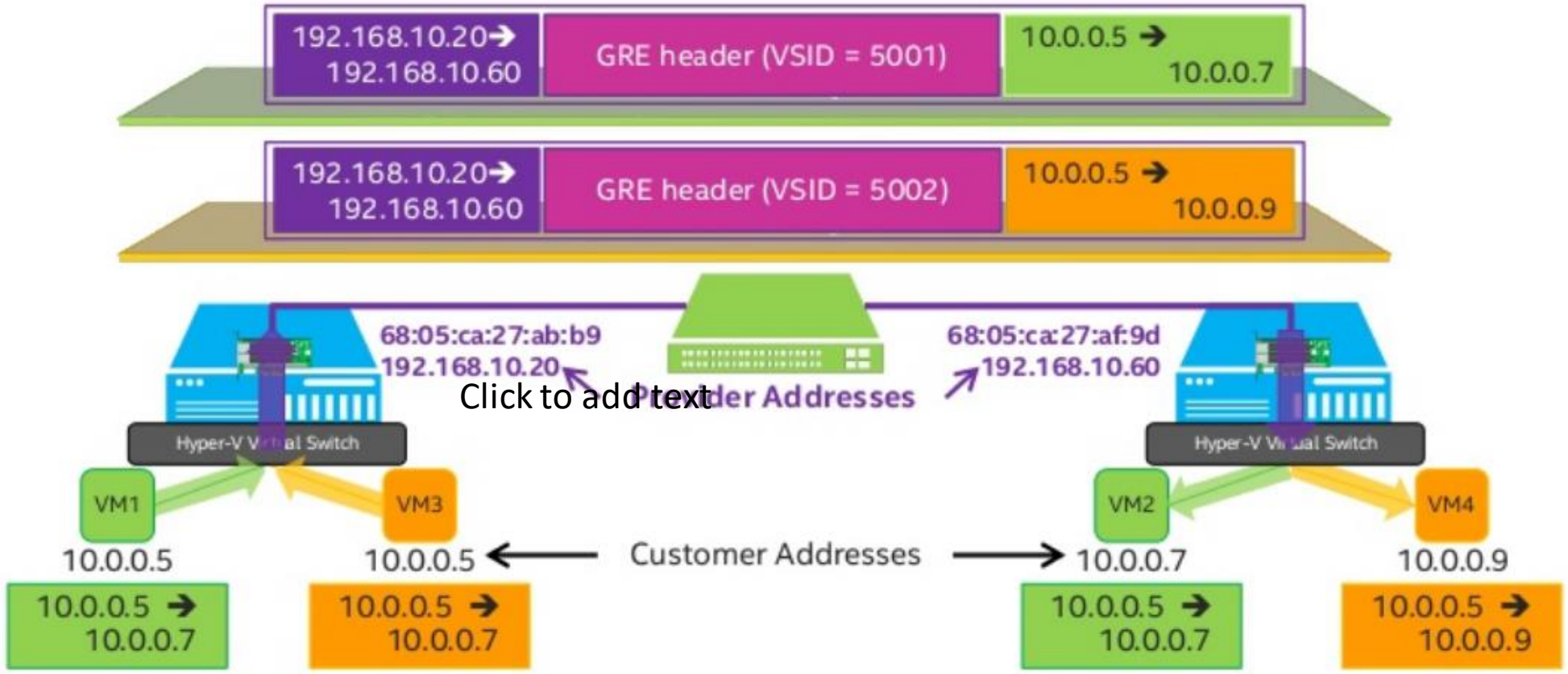
1. VM1 sends a packet destined for VM2.
2. After receiving the packet, VTEP1 performs VXLAN encapsulation. The IP address of VTEP2 is the destination IP address in the outer IP header added to the packet. VTEP1 transmits the encapsulated packet to VTEP2 through the IP network based on the outer MAC address and IP address of the packet.
3. VTEP2 decapsulates the received packet, obtains the original packet sent by VM1, and forwards the packet to VM2.

# NVGRE (RFC 7637)

- Driven by Microsoft.
- Uses GRE (Generic Routing Encapsulation) to tunnel L2 packets across and IP network.
- A 24-Bit VSID (Virtual Subnet ID) is stored within the lower bits of GRE header of the new packet. This provides 16 million logical networks.
- First 32 bits are the same for GRE and NVGRE headers.
- Protocol type for NVGRE in GRE header is 0x6558.



# NVGRE



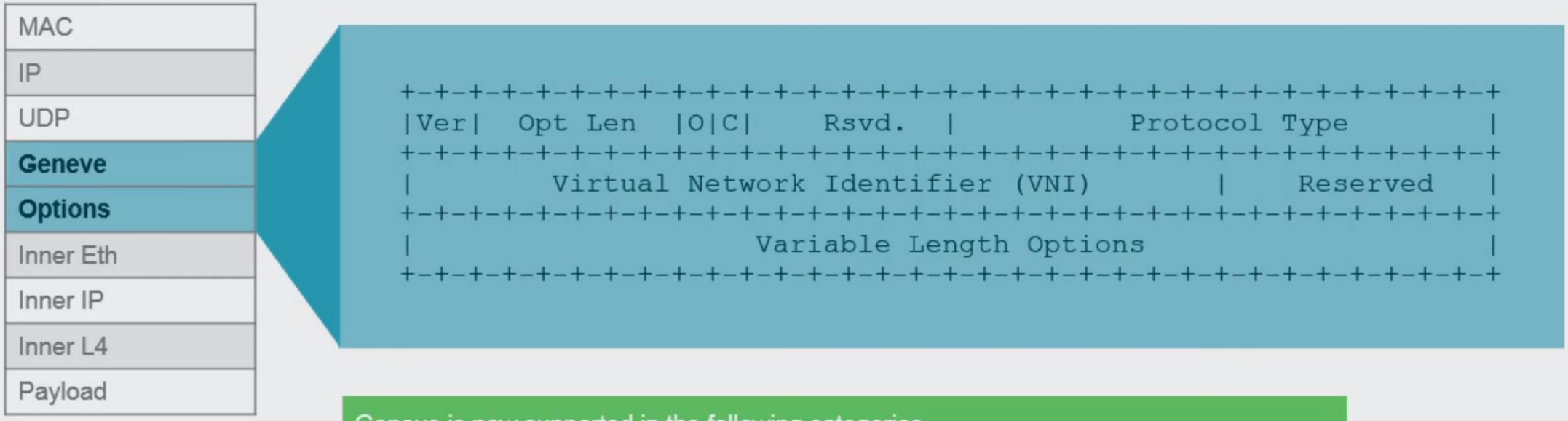

Customer #1 – Green GRE Key - 5001
Customer #2 – Orange GRE Key - 5002

# GENEVE

- GENEVE (Generic Network Virtualization Encapsulation) is a UDP encapsulation for overlay networks.
- Tries to unify other overlay protocols (VXLAN, NVGRE).
- Extensible to support future control planes.
- Adds “options” for flexibility and future extensibility.
- Options use TLV format (Type, Length, Value)

# GENEVE

## Geneve Update




| Geneve is now supported in the following categories | |
|---|---|
| Software | Linux, OVS, OVN |
| NICs | Intel, Broadcom, Mellanox, Netronome |
| Switch ASICS | Broadcom, Mellanox, Cavium, Centec |
| Monitoring Tools | wireshark, tcpdump, libcap |

# Reference Material

# Reference Material

- Computer Networks, 5th Edition by Andrew S. Tanenbaum, David J. Wetherall
  - Sixth edition will be coming soon.

- RFC Source Book:
  - http://networksorcery.com/enp/default.htm
  - Details of RFCs, protocols, header formats etc.

- Linux admin guide for network configuration and management:
  - https://www.tecmint.com/linux-networking-commands/