Network Protocols

Aliaksandr Ramanovich

TRAINING CENTER

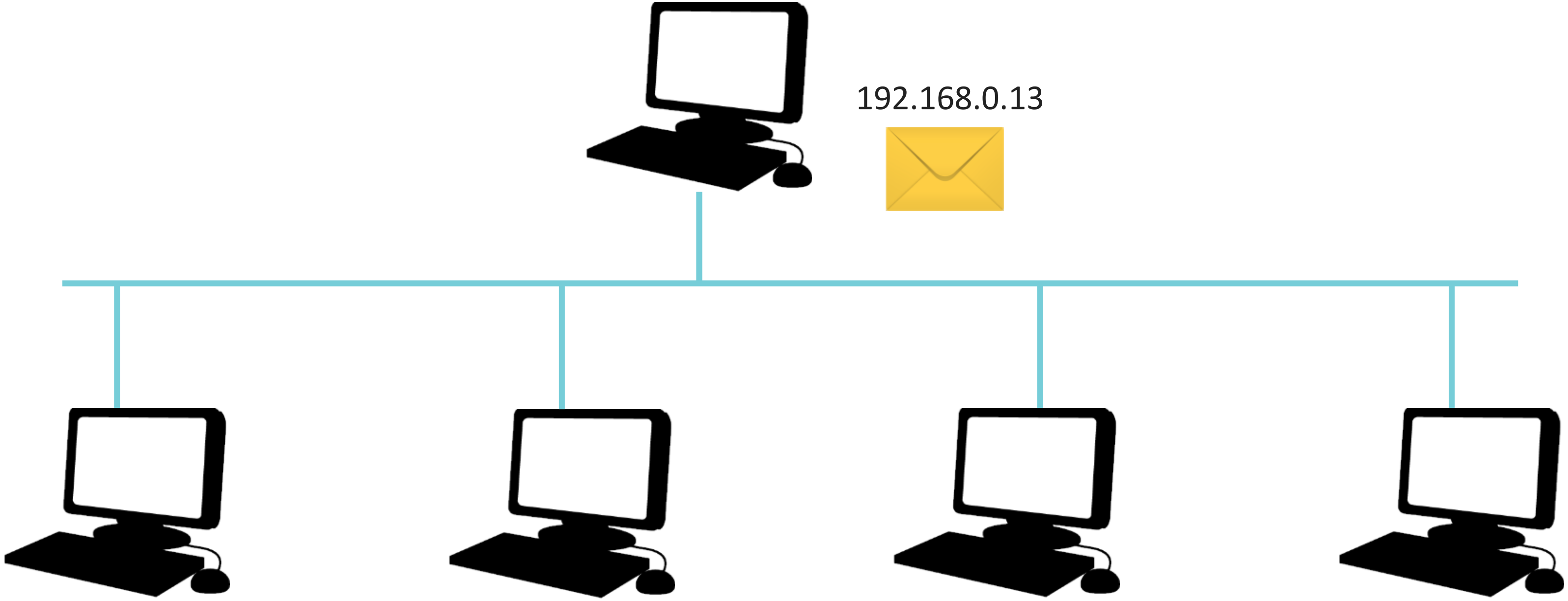
Agenda

- ARP
- TCP/UDP
- ICMP
- PORTS

ADDRESS RESOLUTION PROTOCOL

Address Resolution Protocol

IP 192.168.0.10
MAC 0D:26:57:E7:1F:02



192.168.0.13

IP 192.168.0.11
MAC ??:??:??:??:??:??

IP 192.168.0.12
MAC ??:??:??:??:??:??

IP 192.168.0.13
MAC ??:??:??:??:??:??

IP 192.168.0.14
MAC ??:??:??:??:??:??

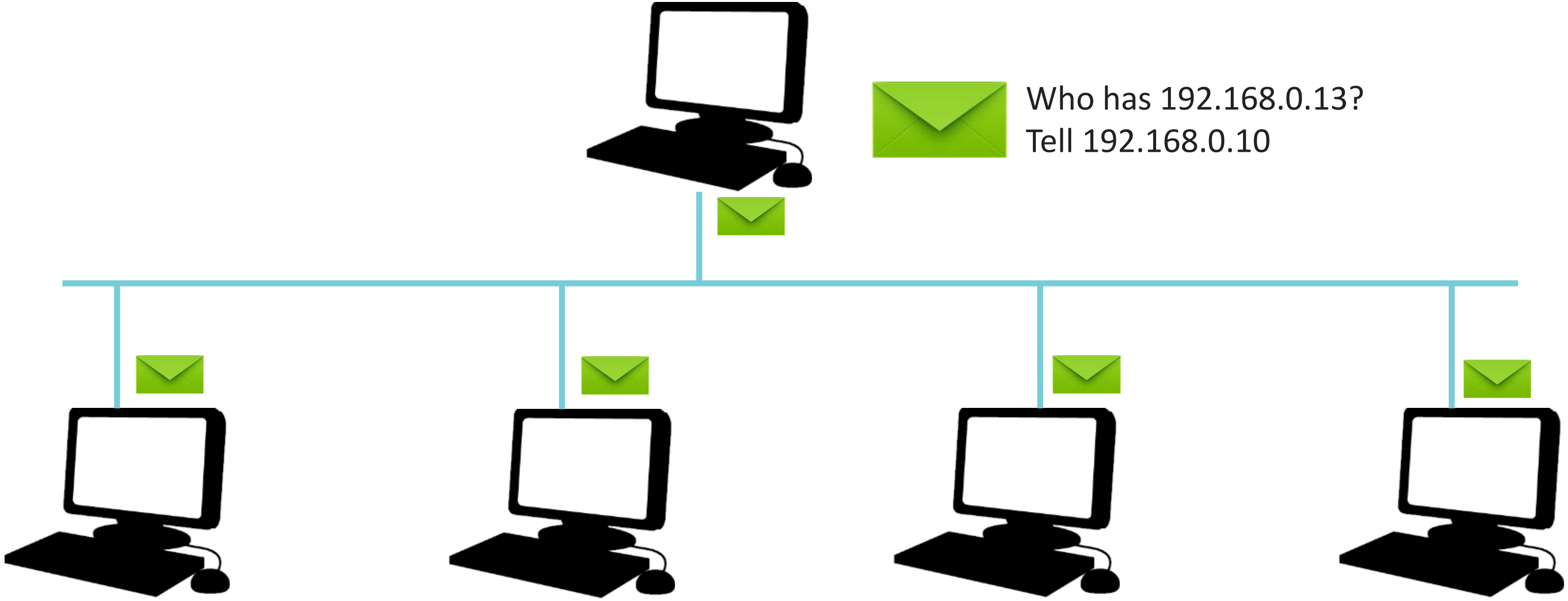
Address Resolution Protocol

November 1982

- An Ethernet Address Resolution Protocol (ARP) was described in RFC 826. It is a communication protocol used for discovering the link layer address associated with a given internet layer address. Today these addresses are MAC and IPv4, but ARP can work with a lot of combinations of network and data link layer technologies.

ARP Request

IP 192.168.0.10
MAC 0D:26:57:E7:1F:02



Who has 192.168.0.13?
Tell 192.168.0.10

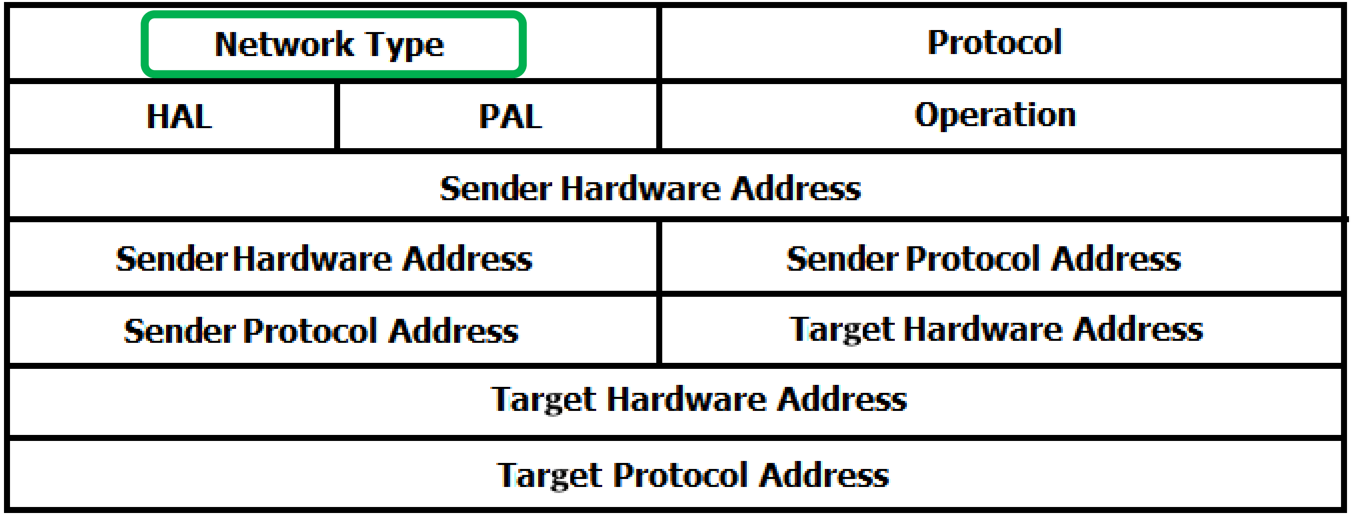
IP 192.168.0.11
MAC ??:??:??:??:??:??

IP 192.168.0.12
MAC ??:??:??:??:??:??

IP 192.168.0.13
MAC ??:??:??:??:??:??

IP 192.168.0.14
MAC ??:??:??:??:??:??

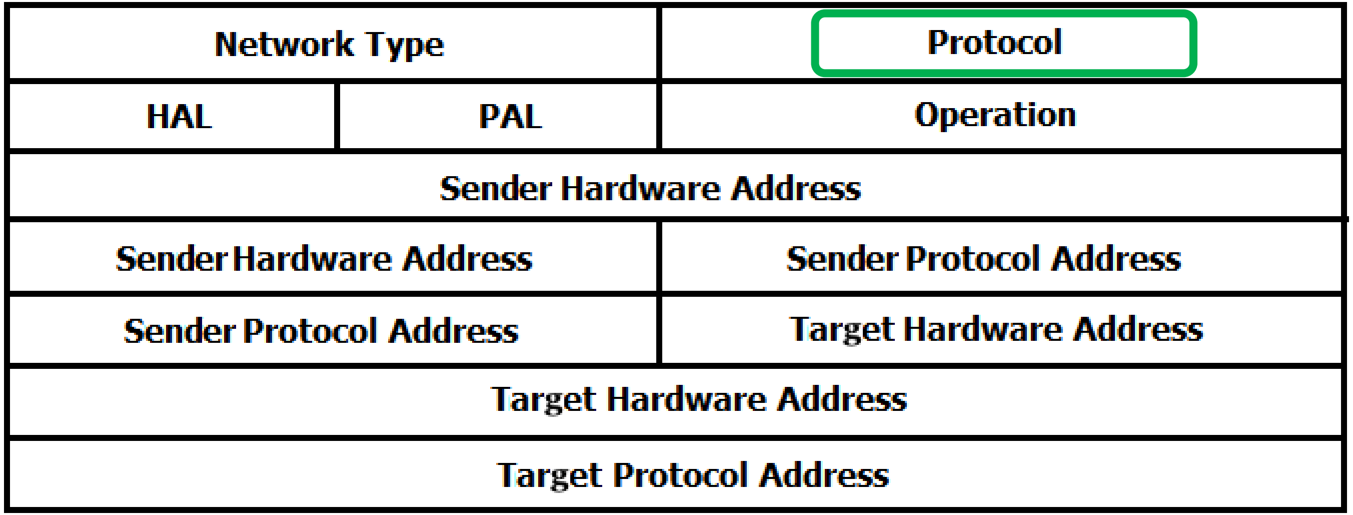
ARP Message Format



<table>
<tr><td colspan="2">Network Type</td><td>Protocol</td></tr>
<tr><td>HAL</td><td>PAL</td><td>Operation</td></tr>
<tr><td colspan="3">Sender Hardware Address</td></tr>
<tr><td colspan="2">Sender Hardware Address</td><td>Sender Protocol Address</td></tr>
<tr><td colspan="2">Sender Protocol Address</td><td>Target Hardware Address</td></tr>
<tr><td colspan="3">Target Hardware Address</td></tr>
<tr><td colspan="3">Target Protocol Address</td></tr>
</table>

Network Type – 1 for Ethernet

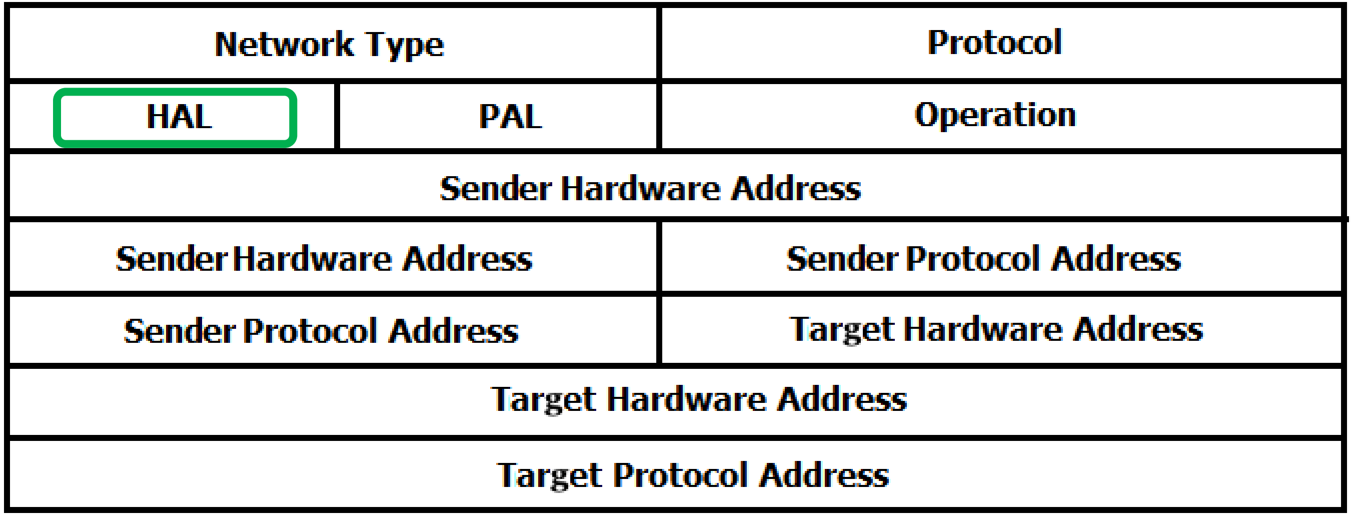
ARP Message Format



<table>
<tr><td colspan="2">Network Type</td><td colspan="2">Protocol</td></tr>
<tr><td>HAL</td><td>PAL</td><td colspan="2">Operation</td></tr>
<tr><td colspan="4">Sender Hardware Address</td></tr>
<tr><td colspan="2">Sender Hardware Address</td><td colspan="2">Sender Protocol Address</td></tr>
<tr><td colspan="2">Sender Protocol Address</td><td colspan="2">Target Hardware Address</td></tr>
<tr><td colspan="4">Target Hardware Address</td></tr>
<tr><td colspan="4">Target Protocol Address</td></tr>
</table>

Protocol – 0x0800 for IP

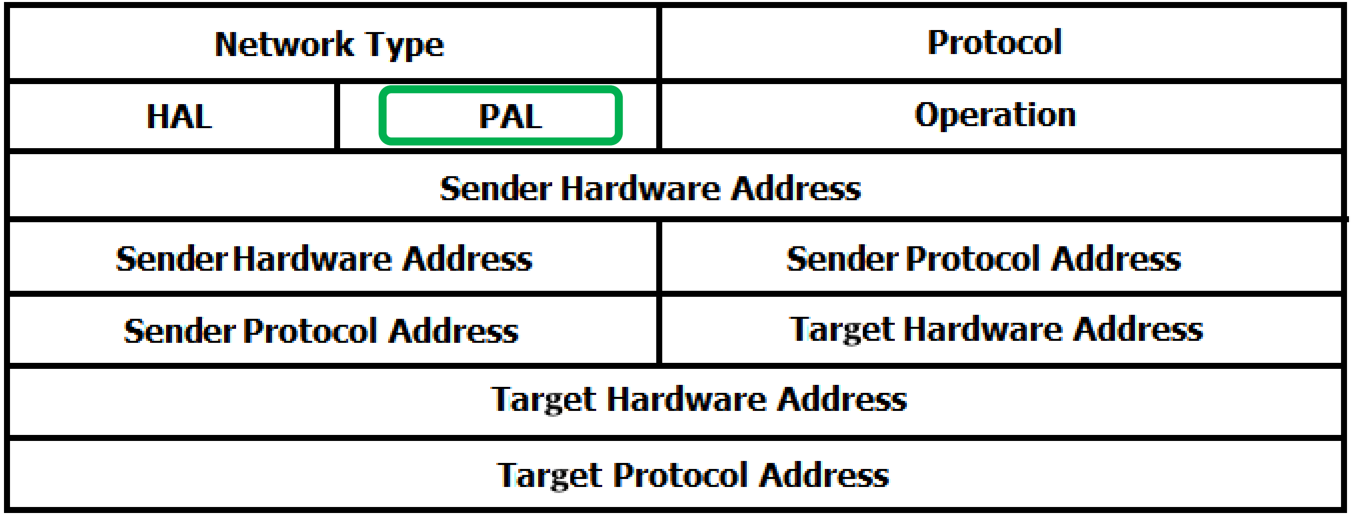
ARP Message Format



Network Type		Protocol
HAL	PAL	Operation
Sender Hardware Address		
Sender Hardware Address		Sender Protocol Address
Sender Protocol Address		Target Hardware Address
Target Hardware Address		
Target Protocol Address		

Hardware Address Length – 6 for MAC address

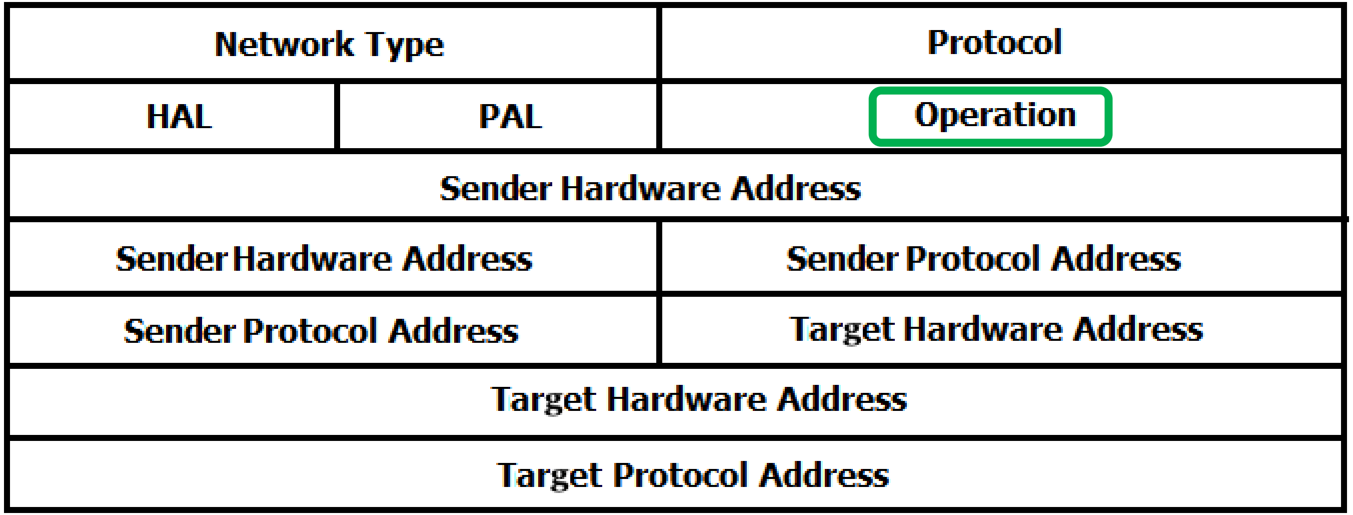
ARP Message Format



<table>
<tr><td colspan="2">Network Type</td><td>Protocol</td></tr>
<tr><td>HAL</td><td>PAL</td><td>Operation</td></tr>
<tr><td colspan="3">Sender Hardware Address</td></tr>
<tr><td colspan="2">Sender Hardware Address</td><td>Sender Protocol Address</td></tr>
<tr><td colspan="2">Sender Protocol Address</td><td>Target Hardware Address</td></tr>
<tr><td colspan="3">Target Hardware Address</td></tr>
<tr><td colspan="3">Target Protocol Address</td></tr>
</table>

Protocol Address Length – 4 for IP address

ARP Message Format



<table>
<tr><td colspan="2">Network Type</td><td>Protocol</td></tr>
<tr><td>HAL</td><td>PAL</td><td>Operation</td></tr>
<tr><td colspan="3">Sender Hardware Address</td></tr>
<tr><td colspan="2">Sender Hardware Address</td><td>Sender Protocol Address</td></tr>
<tr><td colspan="2">Sender Protocol Address</td><td>Target Hardware Address</td></tr>
<tr><td colspan="3">Target Hardware Address</td></tr>
<tr><td colspan="3">Target Protocol Address</td></tr>
</table>

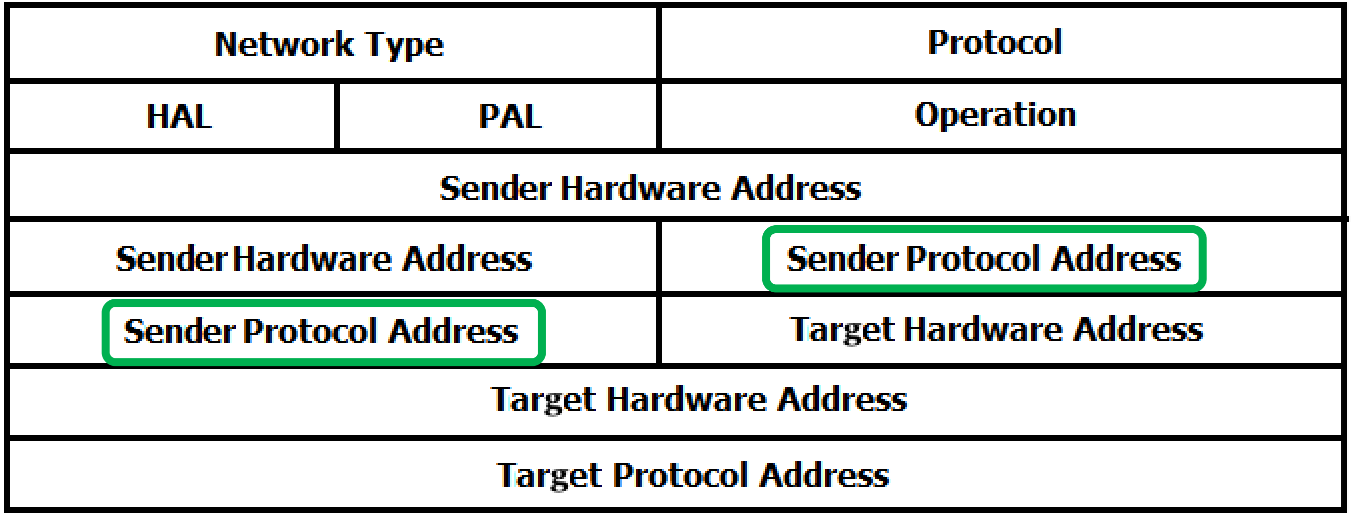
Operation: 1 – request 3 – reverse ARP request
2 – response 4 – reverse ARP response

ARP Message Format

Network Type		Protocol
HAL	PAL	Operation
Sender Hardware Address		
Sender Hardware Address		Sender Protocol Address
Sender Protocol Address		Target Hardware Address
Target Hardware Address		
Target Protocol Address		

Sender Hardware Address – source MAC address
0D:26:57:E7:1F:02 in our case

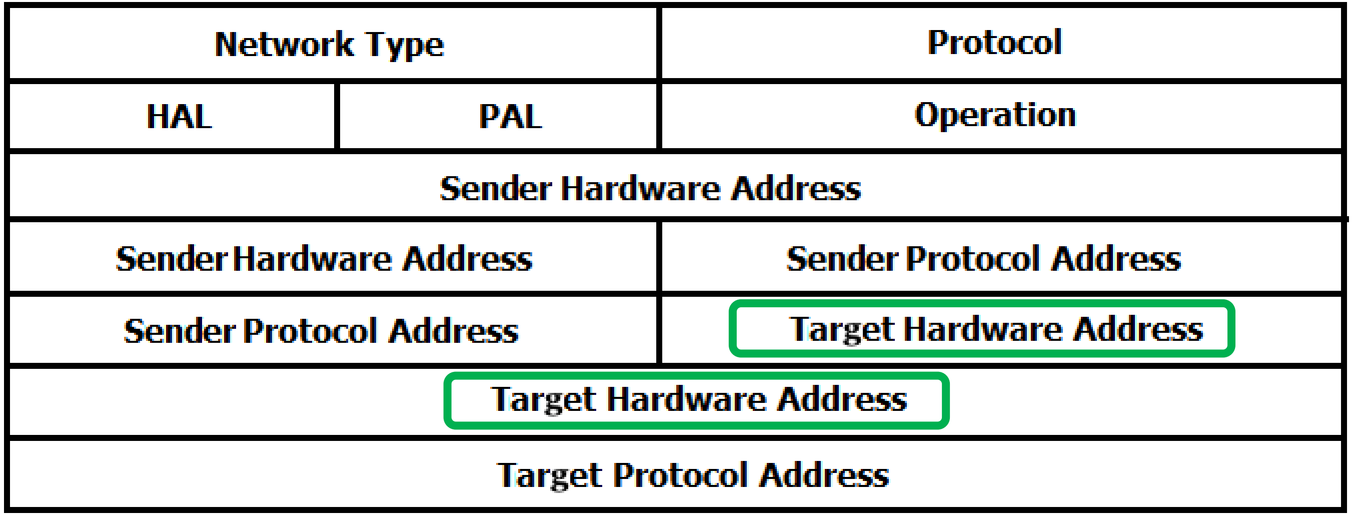
ARP Message Format



<table>
<tr><td colspan="2">Network Type</td><td>Protocol</td></tr>
<tr><td>HAL</td><td>PAL</td><td>Operation</td></tr>
<tr><td colspan="3">Sender Hardware Address</td></tr>
<tr><td colspan="2">Sender Hardware Address</td><td>Sender Protocol Address</td></tr>
<tr><td colspan="2">Sender Protocol Address</td><td>Target Hardware Address</td></tr>
<tr><td colspan="3">Target Hardware Address</td></tr>
<tr><td colspan="3">Target Protocol Address</td></tr>
</table>

Sender Protocol Address – source IP address
192.168.0.10

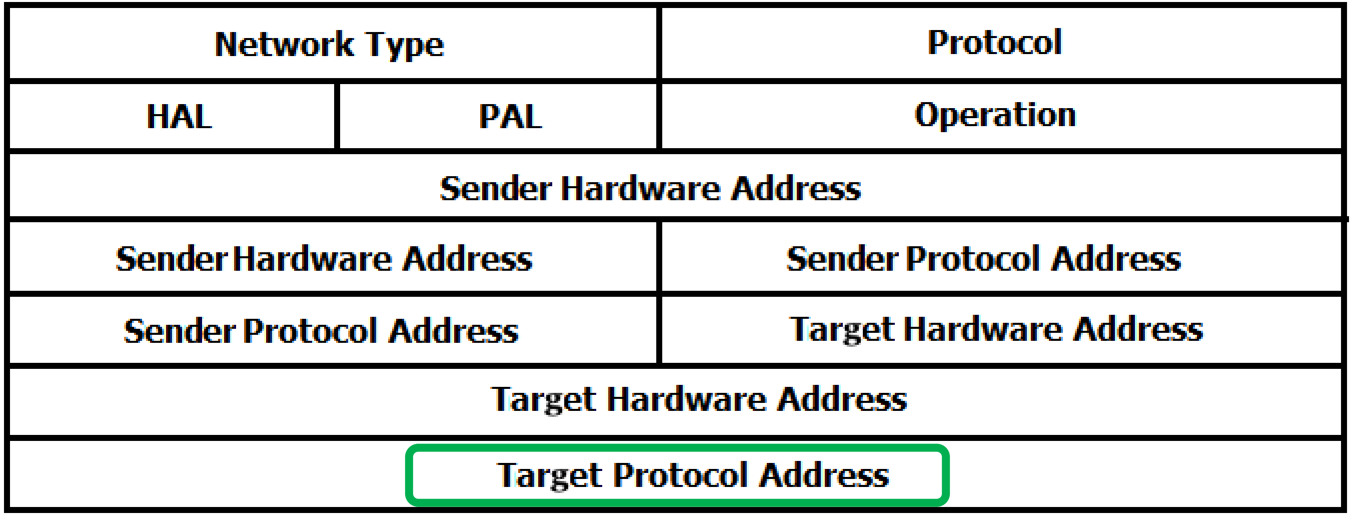
ARP Message Format



<table>
<tr><td colspan="2">Network Type</td><td>Protocol</td></tr>
<tr><td>HAL</td><td>PAL</td><td>Operation</td></tr>
<tr><td colspan="3">Sender Hardware Address</td></tr>
<tr><td colspan="2">Sender Hardware Address</td><td>Sender Protocol Address</td></tr>
<tr><td colspan="2">Sender Protocol Address</td><td>Target Hardware Address</td></tr>
<tr><td colspan="3">Target Hardware Address</td></tr>
<tr><td colspan="3">Target Protocol Address</td></tr>
</table>

Target Hardware Address – unknown MAC address in ARP request
00:00:00:00:00:00 in our case

ARP Message Format

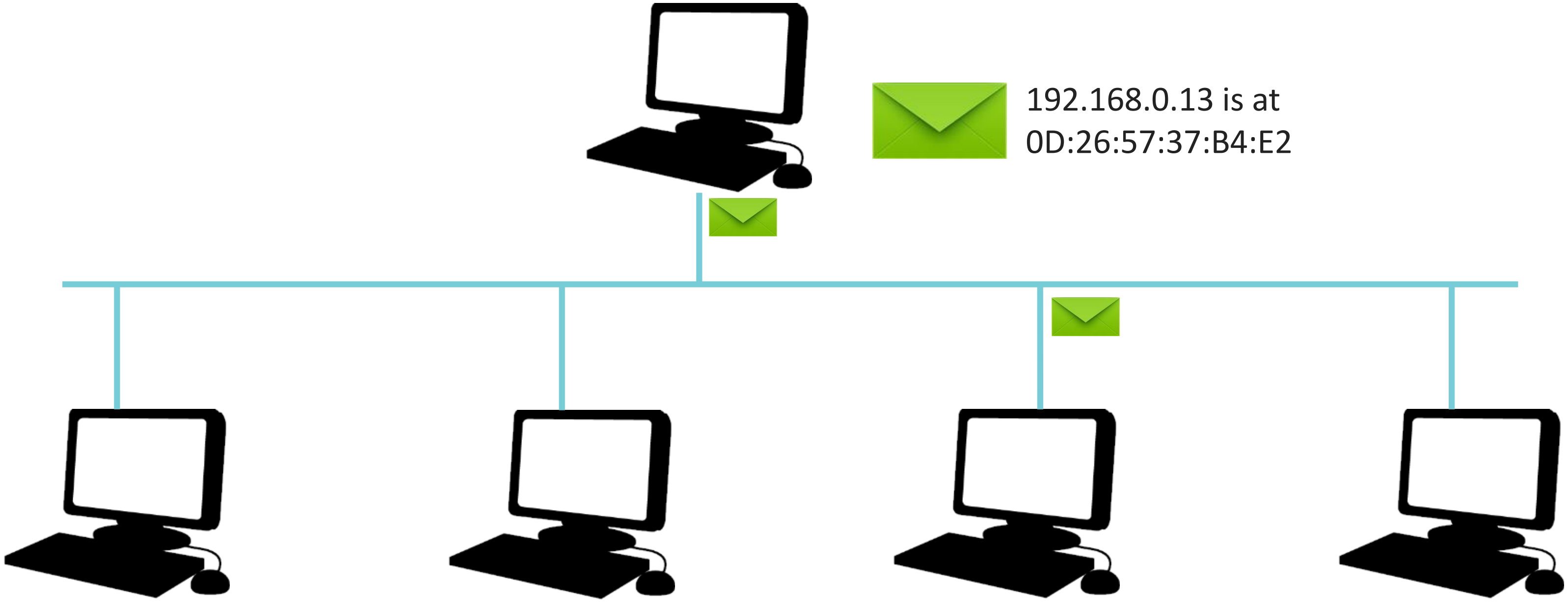


<table>
<tr><td colspan="2">Network Type</td><td>Protocol</td></tr>
<tr><td>HAL</td><td>PAL</td><td>Operation</td></tr>
<tr><td colspan="3">Sender Hardware Address</td></tr>
<tr><td colspan="2">Sender Hardware Address</td><td>Sender Protocol Address</td></tr>
<tr><td colspan="2">Sender Protocol Address</td><td>Target Hardware Address</td></tr>
<tr><td colspan="3">Target Hardware Address</td></tr>
<tr><td colspan="3">Target Protocol Address</td></tr>
</table>

Target Protocol Address – IP address of a receiver
192.168.0.13 in our case

ARP Response

IP 192.168.0.10
MAC 0D:26:57:E7:1F:02



192.168.0.13 is at
0D:26:57:37:B4:E2

IP 192.168.0.11
MAC ??:??:??:??:??:??

IP 192.168.0.12
MAC ??:??:??:??:??:??

IP 192.168.0.13
MAC 0D:26:57:37:B4:E2

IP 192.168.0.14
MAC ??:??:??:??:??:??

ARP Response Message

<table>
<tr><td colspan="2">1</td><td colspan="2">0x0800</td></tr>
<tr><td>6</td><td>4</td><td colspan="2">2</td></tr>
<tr><td colspan="4">0D:26:57:37</td></tr>
<tr><td colspan="2">B4:E2</td><td colspan="2">192.168</td></tr>
<tr><td colspan="2">0.13</td><td colspan="2">0D:26</td></tr>
<tr><td colspan="4">57:E7:1F:02</td></tr>
<tr><td colspan="4">192.168.0.10</td></tr>
</table>

Gratuitous ARP

- Gratuitous ARP request is a packet where the source and destination IP are both set to the IP of the machine issuing the packet and the destination MAC is the broadcast address FF:FF:FF:FF:FF:FF. Ordinarily, no reply packet will occur.
- It is useful to:
  - detect IP conflicts
  - update other machines' ARP tables
  - troubleshoot connection issues

ARP Table

IP address	MAC address	Type
192.168.0.1	0D-26-57-A2-FF-02	Static
192.168.0.13	0D-26-57-37-B4-E2	Dynamic

- Systems keep an ARP look-up table where they store information about what IP addresses are associated with what MAC addresses. When trying to send a packet to an IP address, the system will first consult this table to see if it already knows the MAC address. If there is a value cached, ARP is not used.

- Static type means that record was made manually with *arp* utility, Dynamic – record was discovered using ARP request. This type of record has a limited lifetime and will be deleted if there is no data transmission between hosts.

ARP

- The Reverse Address Resolution Protocol - translates Layer 2 addresses to Layer 3 addresses. It is obsolete and replaced by BOOTP, which was later superseded by the Dynamic Host Configuration Protocol (DHCP).

- ARP is vulnerable - it does not authenticate ARP requests and ARP responses. And since the network interfaces on computers support gratuitous requests/responses an ARP spoofing attack is possible.

- In IPv6 Neighbor Discovery Protocol and Secure Neighbor Discovery protocols are used instead of ARP.

TRANSMISSION CONTROL PROTOCOL

Transmission Control Protocol

1973

1981 RFC 793

2014 RFC 7323

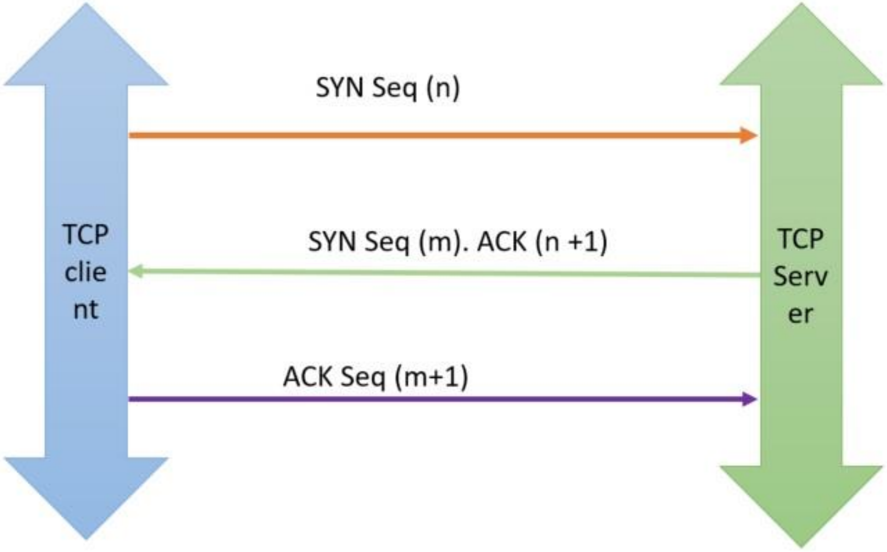
- TCP is a standard for exchanging data between different devices in a computer network. It allows two endpoints in a shared computer network to establish a connection that enables a two-way transmission of data. Any data loss is detected and automatically corrected, which is why TCP is also called a reliable protocol.

TCP Handshake

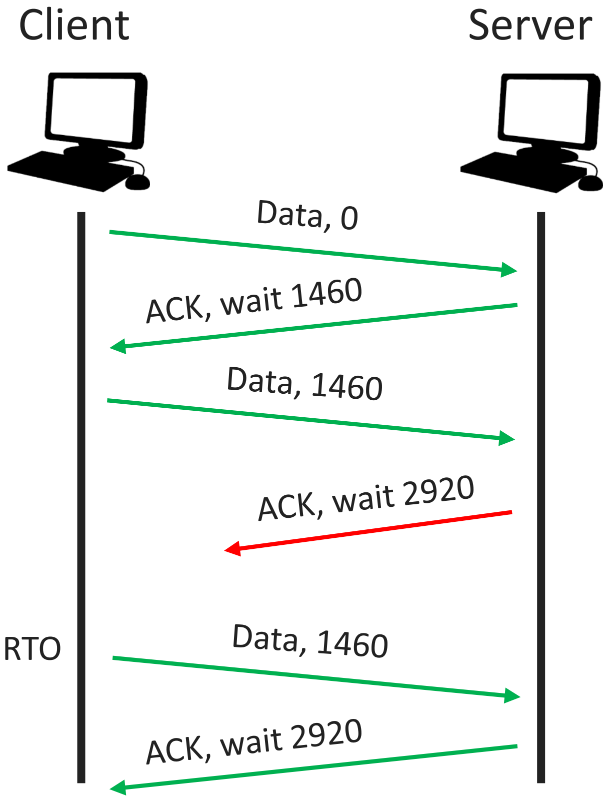
- TCP is a connection oriented protocol. Communicating hosts go through a synchronization process to establish a virtual connection.

- This synchronization process insures that both sides are ready for data transmission and allows the devices to determine the initial sequence numbers.

- Sequence numbers are reference numbers between the two devices. The sequence numbers give each host a way to ACK the SYN, so the receiver knows which connection request the sender is responding to.

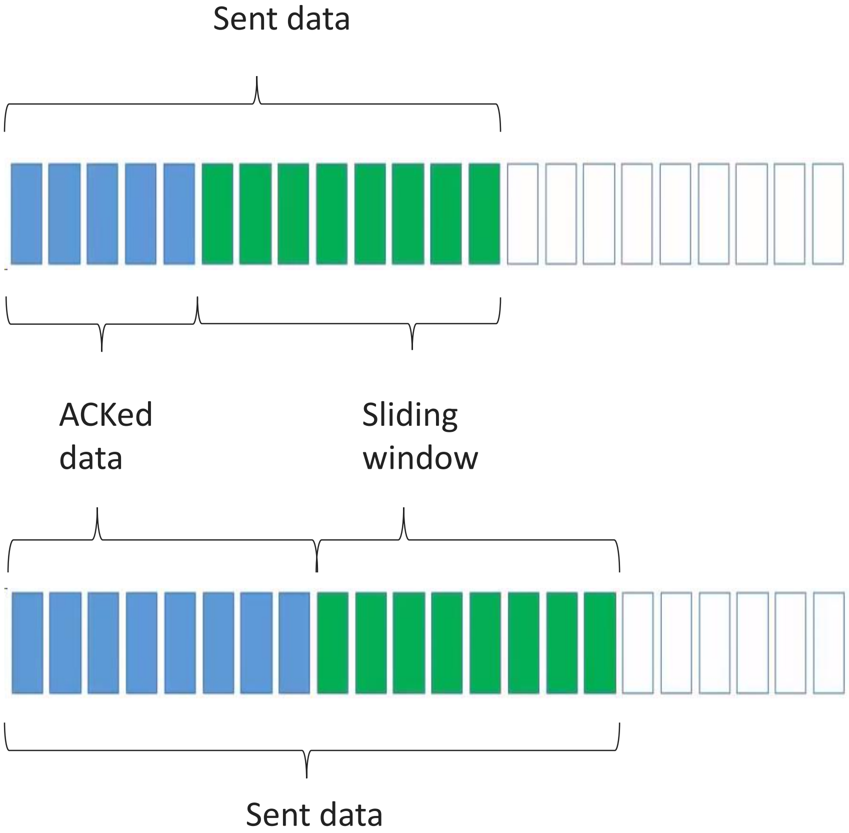


Reliable Data Transmission



- The Client and Server typically agree on the maximum size of the TCP segments to be sent (MSS). By default, up to 1,500 bytes per segment are possible, with at least 20 bytes for the TCP header and a further 20 bytes for the IP header, leaving 1,460 bytes for payload data.
- Data is divided into 1,46 kb blocks, numbered and sent to the Server.
- The Server must acknowledge the receipt of segment and can reconstruct the actual sequence based on the sequence numbers.
- If the Client does not receive acknowledgment for a transmitted segment, it resends the segment after retransmission timeout (RTO) period.

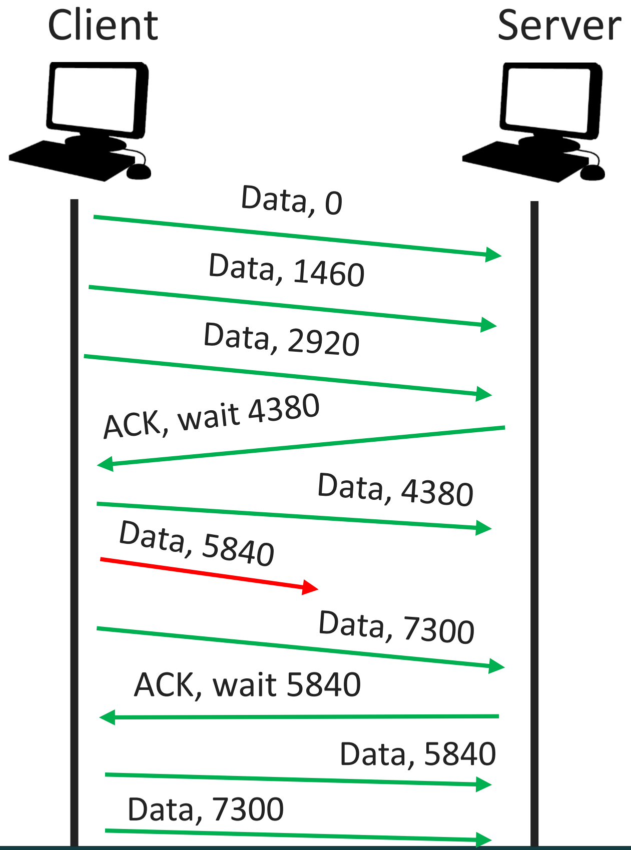
TCP Sliding Window



- A TCP sliding window provides an efficient use of network bandwidth because it enables hosts to send multiple bytes or packets before waiting for an acknowledgment.

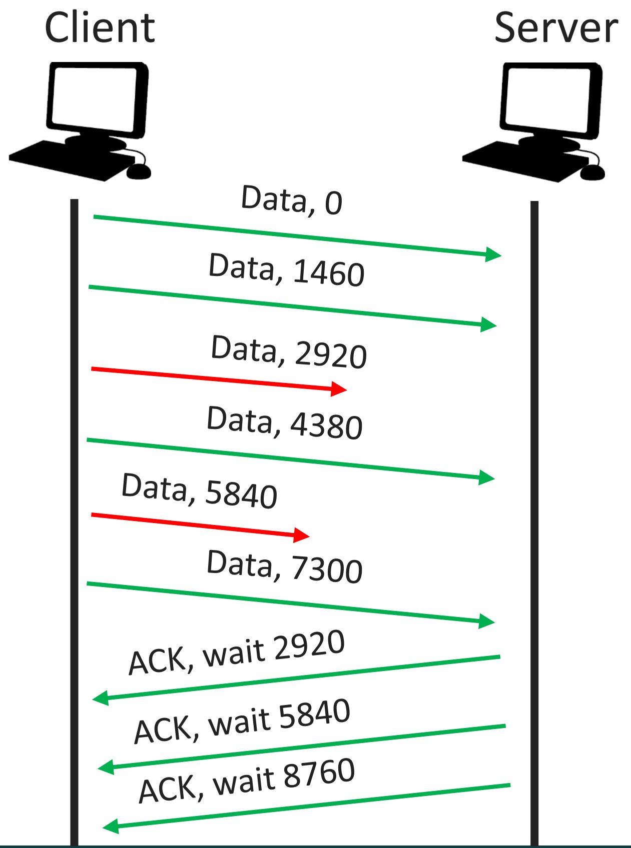
- In TCP, the receiver specifies the current window size in every packet. A window is the number of data bytes that the sender is allowed to send before waiting for an acknowledgment. Initial window sizes are indicated at connection setup, but might vary throughout the data transfer to provide flow control. A window size of zero means “Send no data.” The default TCP window size is 4128 bytes.

Cumulative ACK



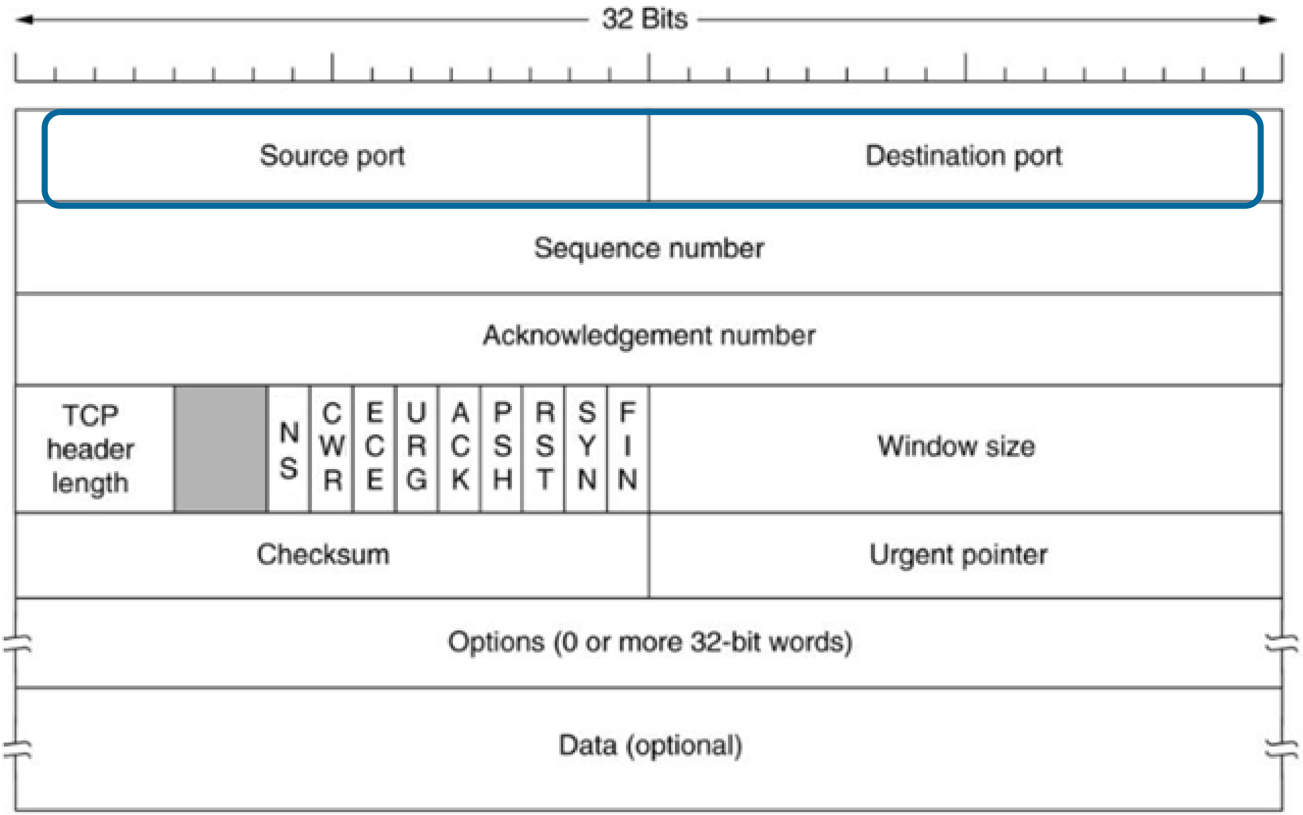
- Cumulative ACK acknowledges receipt of the entire message chain up to the specified byte. It is enabled by default.
- Example:
  - Connection is established, window size is 3 segments
  - Client sends 3 segments of data
  - Server received them and sends ACK
  - Client sends 3 segments, but the 2nd lost
  - Server received 2 segments, but they aren’t consistent. It sends ACK only for the first one
  - Client received ACK and resends segments 5840 and 7300 and adds third segment with new data

Selective ACK



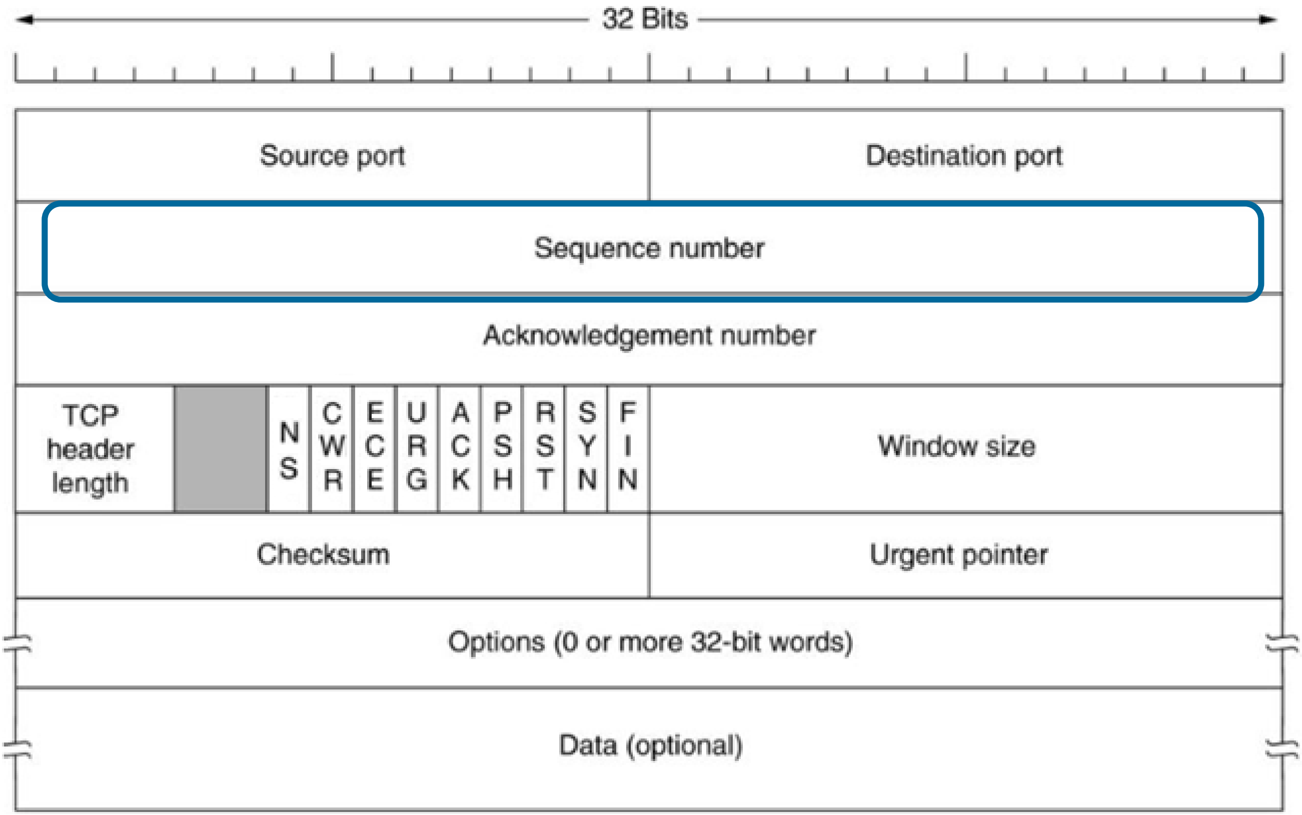
- Selective ACK confirms receipt of the range of bytes. Effective for large window sizes, but requires additional header field.
- Example:
  - Connection is established, window size is 6 segments
  - Client sends 6 segments of data, but the 3rd and 5th are lost.
  - Server received 4 segments, and sends ACKs for 1-2 segments, 4th and 6th.
  - Client received ACK and resends segments 2920 and 5840 and adds 4 segments with new data

TCP Header



- Identifies sending and receiving ports

TCP Header



32 Bits

Source port

Destination port

Sequence number

Acknowledgement number

TCP header length

NS

CWR

ECE

URG

ACK

PSH

RST

SYN

FIN

Window size

Checksum

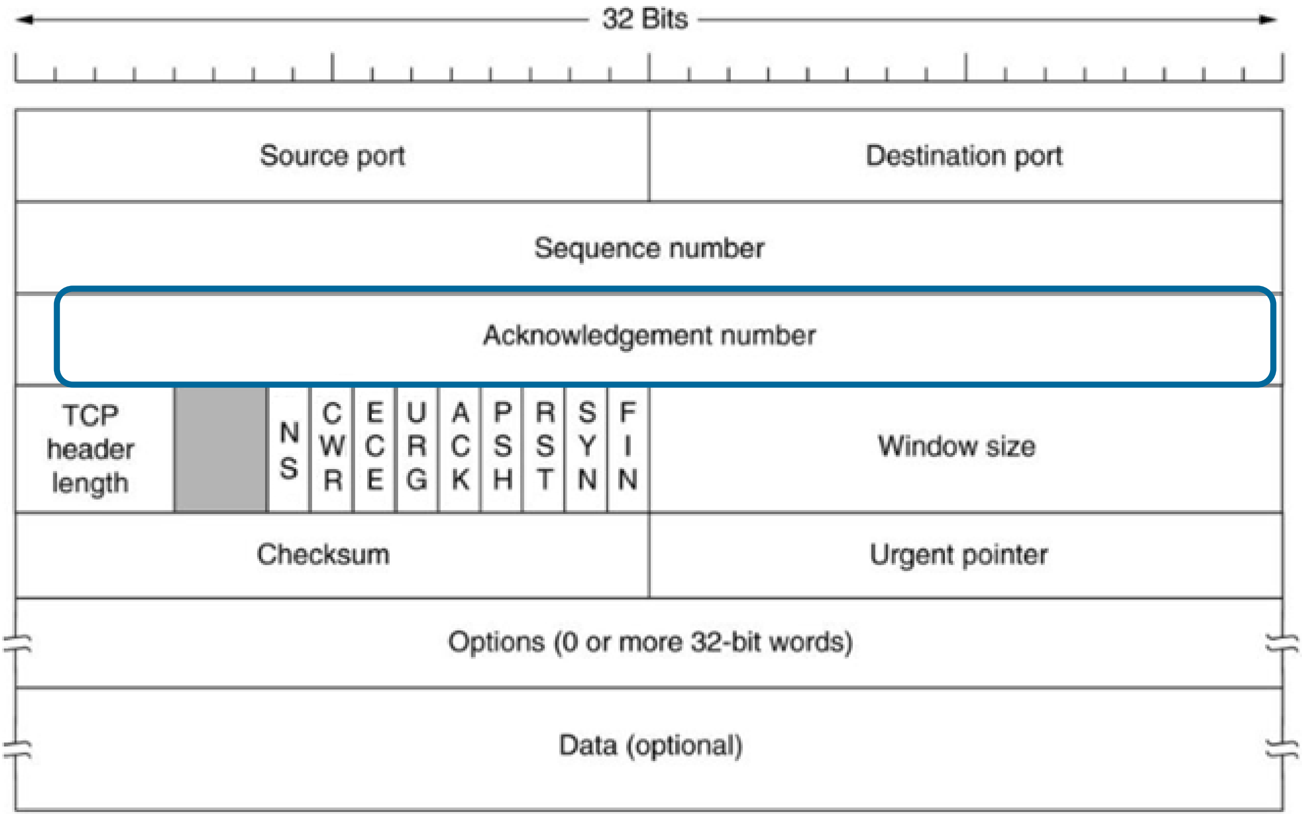
Urgent pointer

Options (0 or more 32-bit words)

Data (optional)

- If the SYN flag is set, then this is the initial sequence number.
- Otherwise is the accumulated sequence number of the first data byte of this segment for the current session.

TCP Header



32 Bits

Source port

Destination port

Sequence number

Acknowledgement number

TCP header length

NS

CWR

ECE

URG

ACK

PSH

RST

SYN

FIN

Window size

Checksum

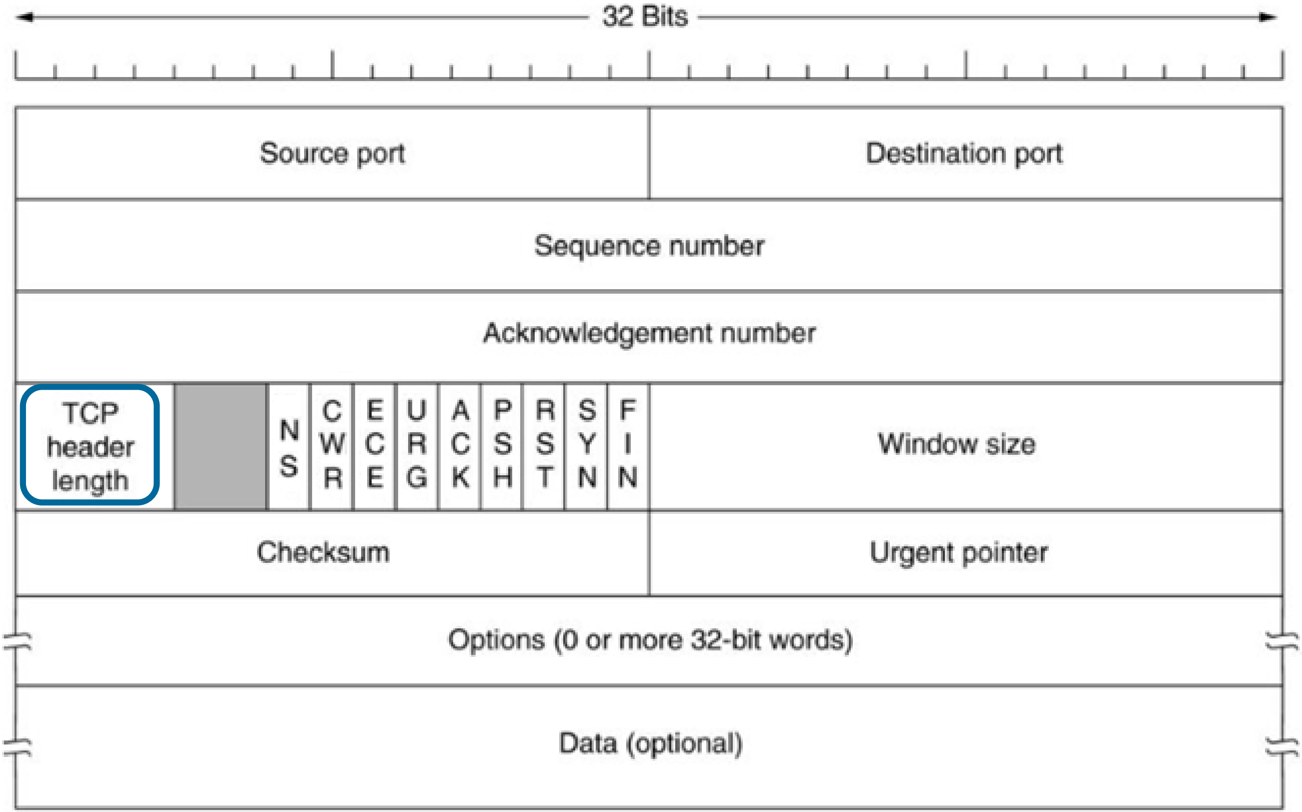
Urgent pointer

Options (0 or more 32-bit words)

Data (optional)

- If the ACK flag is set then the value of this field is the next sequence number that the sender of the ACK is expecting.

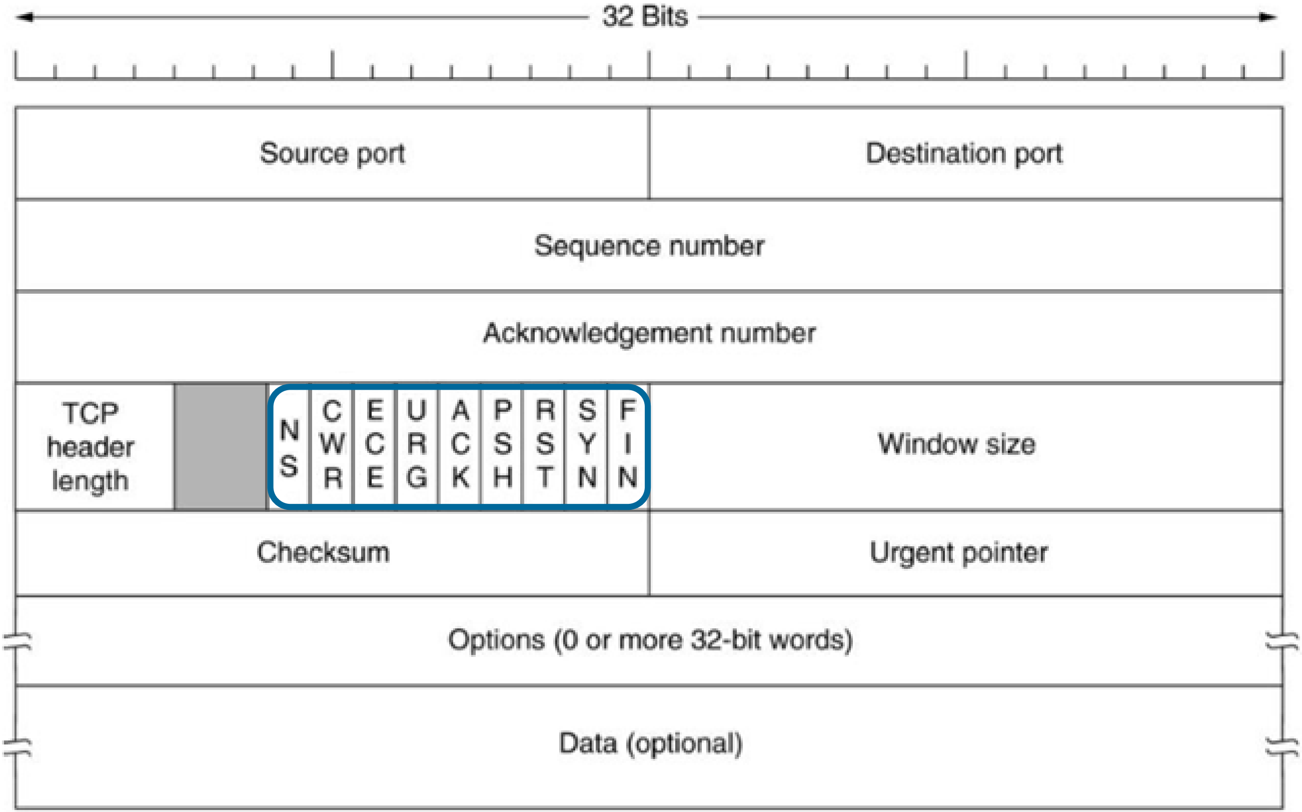
TCP Header



- Also known as Data offset - specifies the size of the TCP header in 32-bit words.

- The minimum size header is 5 words (20 bytes) and the maximum is 15 words (60 bytes).

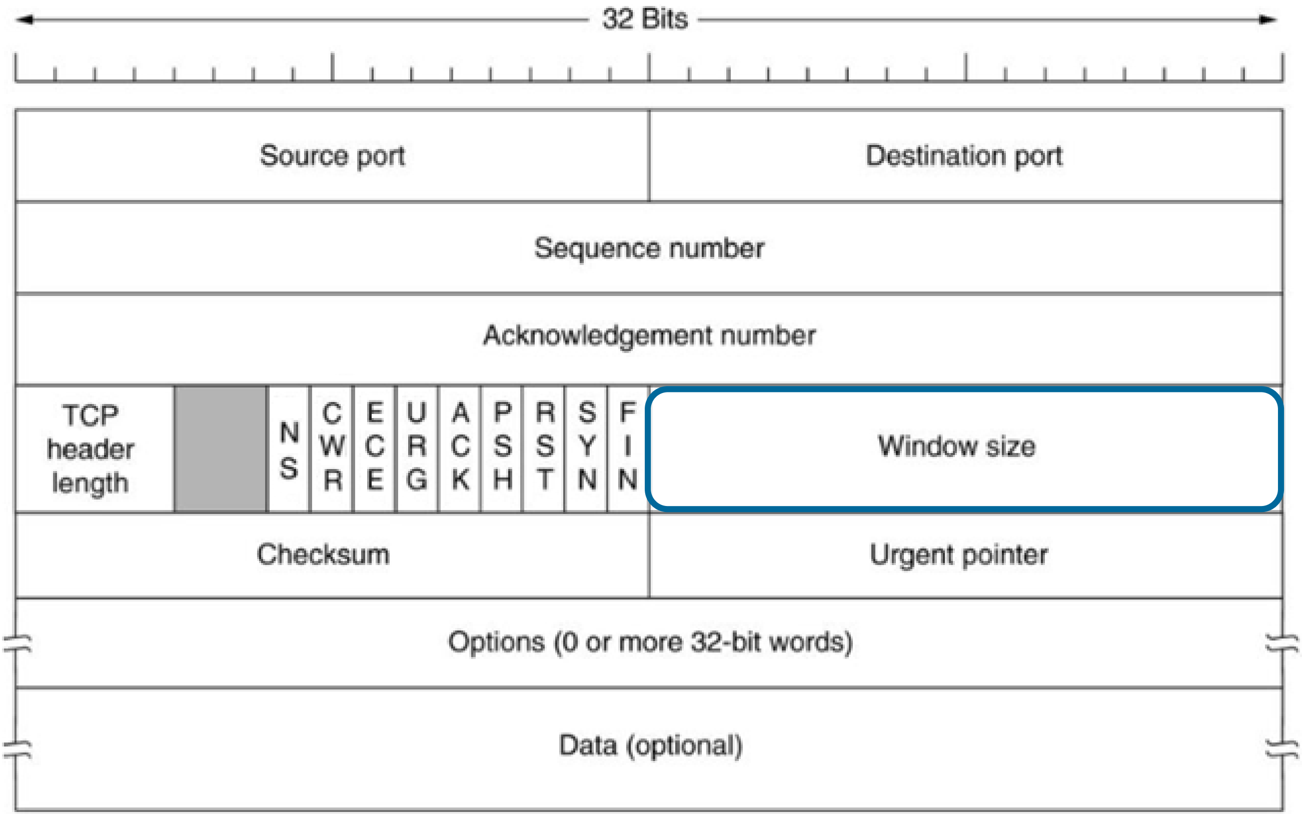
TCP Header



Flags:

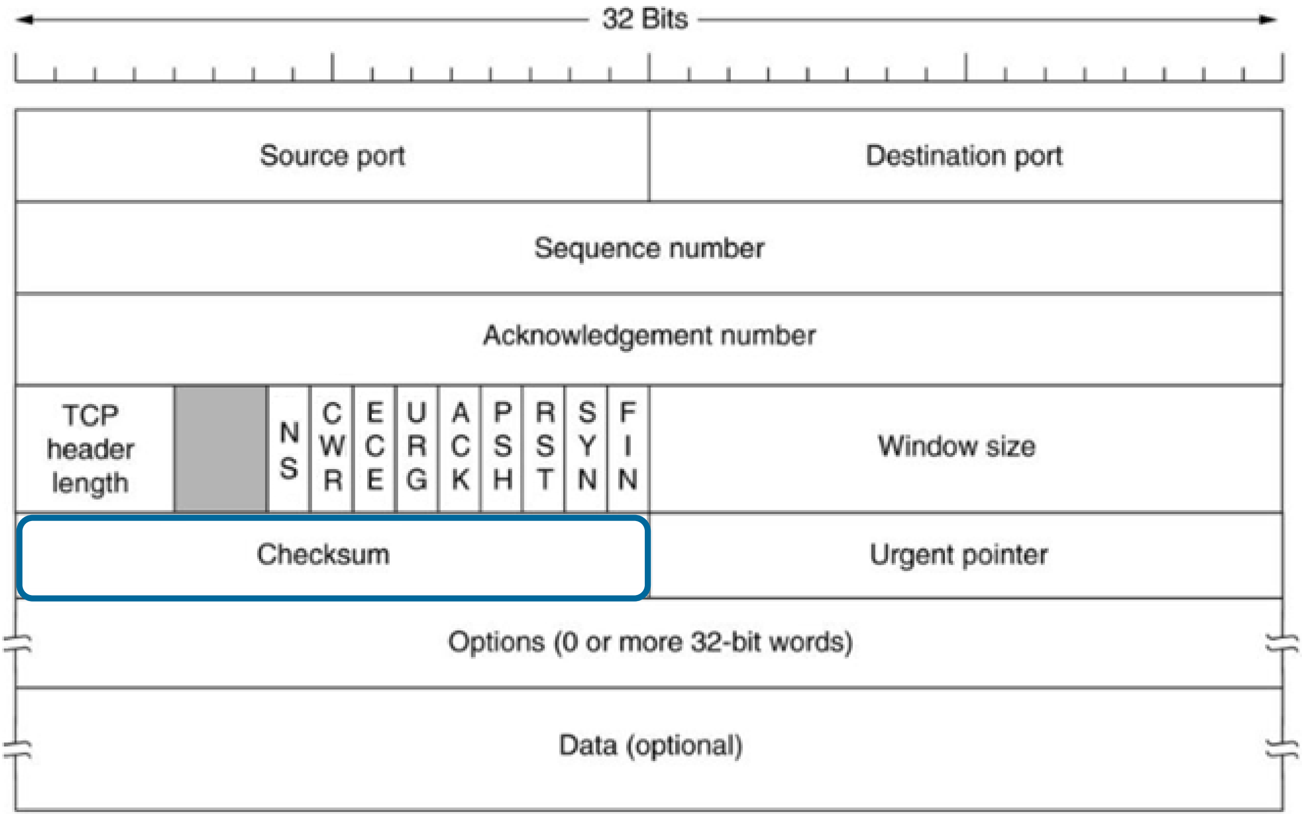
- NS, CWR, ECE – used with Explicit Congestion Notification methods
- URG - urgent pointer
- ACK - acknowledgment field
- PSH - Asks to push the buffered data to the receiving application
- RST - reset the connection
- SYN - Synchronize sequence numbers
- FIN - last packet from sender

TCP Header



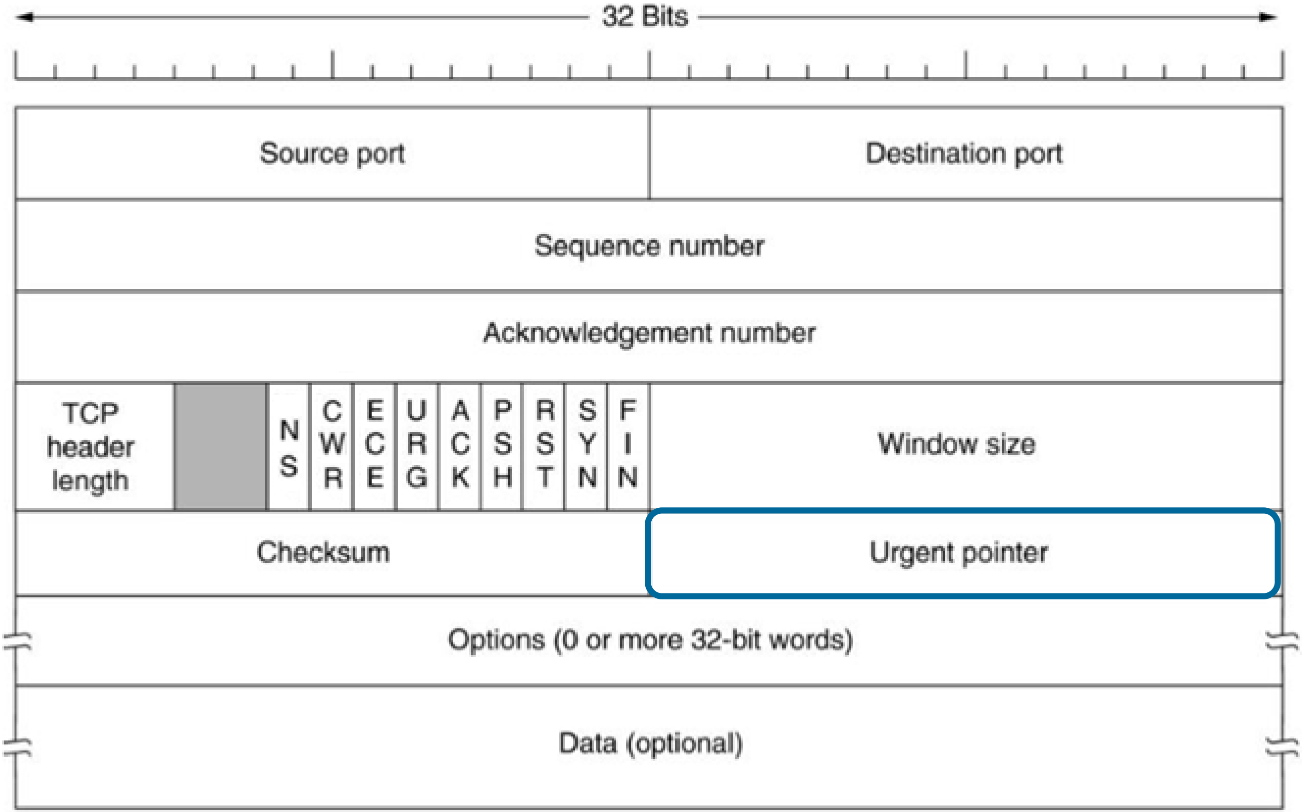
- The size of the receive window

TCP Header



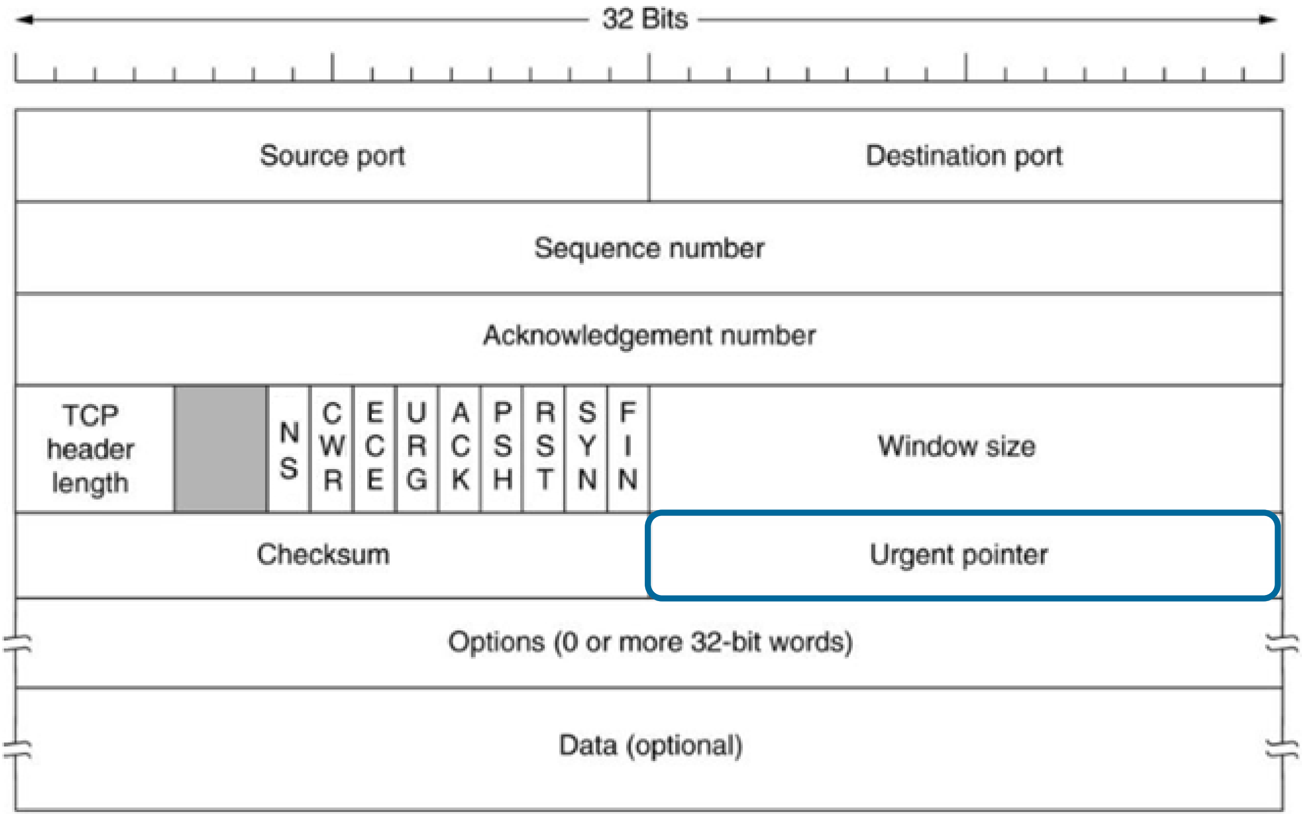
- The 16-bit checksum field is used for error-checking of the TCP header

TCP Header



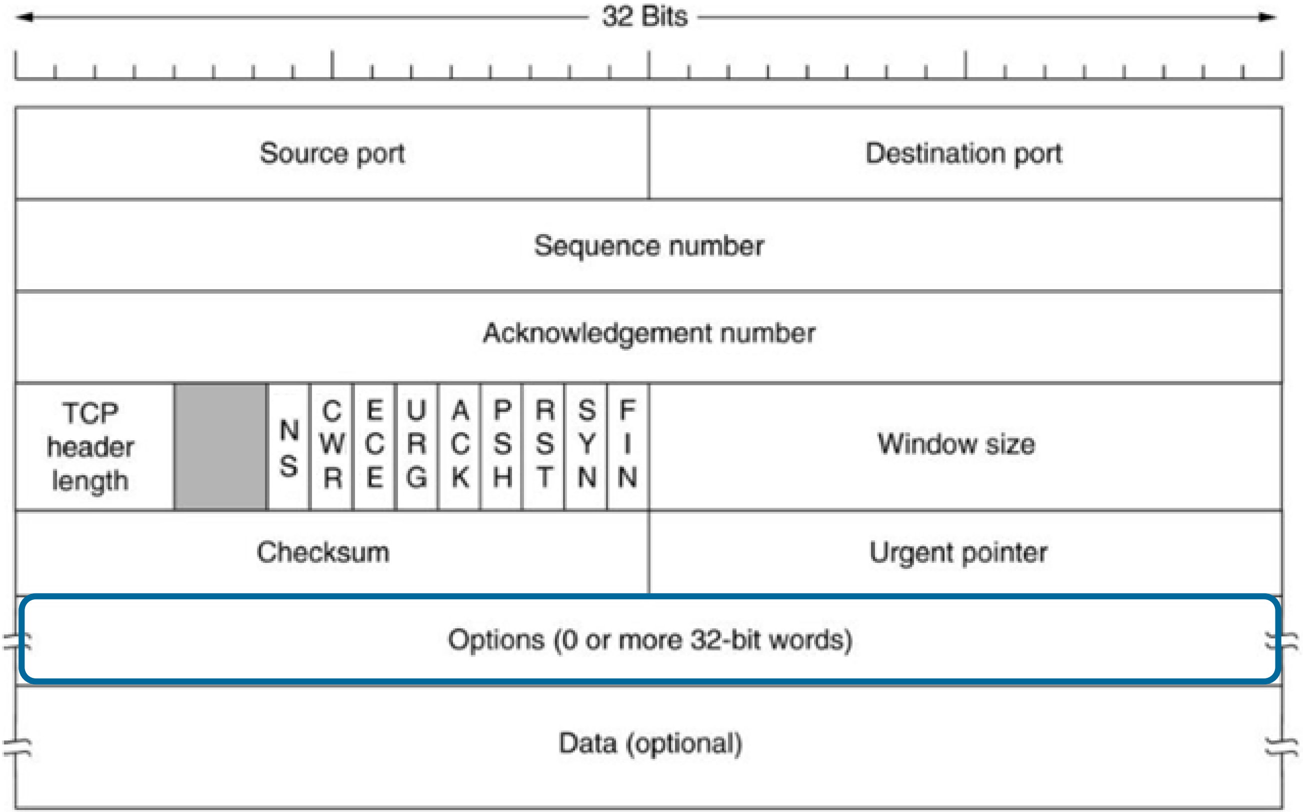
- If the URG flag is set, then this 16-bit field is an offset from the sequence number indicating the last urgent data byte

TCP Header



- If the URG flag is set, then this 16-bit field is an offset from the sequence number indicating the last urgent data byte

TCP Header



32 Bits

Source port

Destination port

Sequence number

Acknowledgement number

TCP header length

NS CWR ECE URG ACK PSH RST SYN FIN

Window size

Checksum

Urgent pointer

Options (0 or more 32-bit words)

Data (optional)

- Optional field with 0 - 40 bytes length
- Options have up to three fields: Option-Kind (1 byte), Option-Length (1 byte), Option-Data (variable). The Option-Kind field indicates the type of option, and is the only field that is not optional
- Option-Length indicates the total length of the option

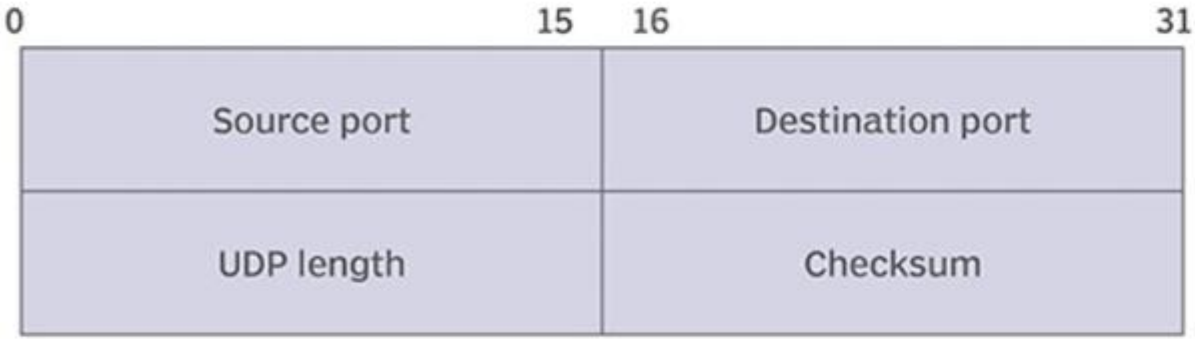
USER DATAGRAM PROTOCOL

User Datagram Protocol

1980 RFC 768

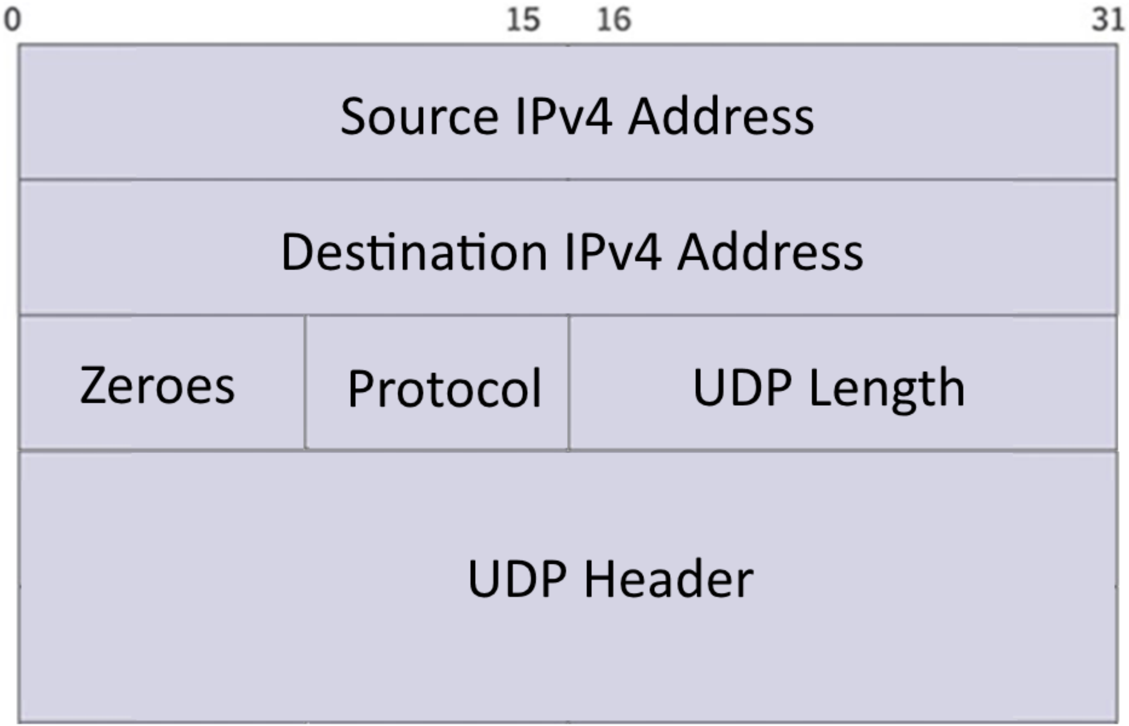
- UDP uses a simple connectionless communication model with a minimum of protocol mechanisms.
- It provides checksums for data integrity, and port numbers for addressing different functions at the source and destination of the datagram.
- UDP has no handshaking dialogues, and thus exposes the user's program to any unreliability of the underlying network; there is no guarantee of delivery, ordering, or duplicate protection.

UDP Header



- Source port identifies the sender's port, if not used should be zero
- Destination port identifies the receiver's port and is required
- Length specifies the length in bytes of the UDP header and UDP data. The minimum length is 8 bytes, maximum 65,507 bytes (20 bytes of IP header and 8 bytes of UDP)
- Checksum - field may be used for error-checking of the header and data. Optional in IPv4 and mandatory in IPv6

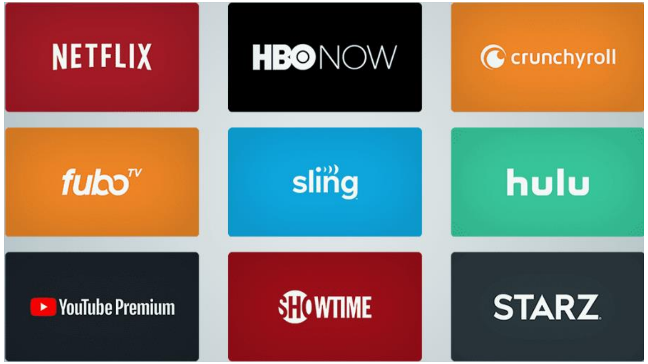
UDP Pseudo Header



0	15	16 31
Source IPv4 Address		
Destination IPv4 Address		
Zeroes	Protocol	UDP Length
UDP Header		

- When UDP runs over IPv4, the checksum is computed using a "pseudo header" that contains some of the same information from the real IPv4 header.
- The pseudo header is not the real IPv4 header used to send an IP packet, it is used only for the checksum calculation.

User Datagram Protocol



TCP and UDP

- TCP
  - Reliable
  - Ordered
  - Heavyweight
  - Congestion and Flow Control
  - Streaming
  - End-to-End

- UDP
  - Unreliable
  - Not ordered
  - Lightweight
  - No congestion control
  - Datagrams
  - Broadcast and Multicast

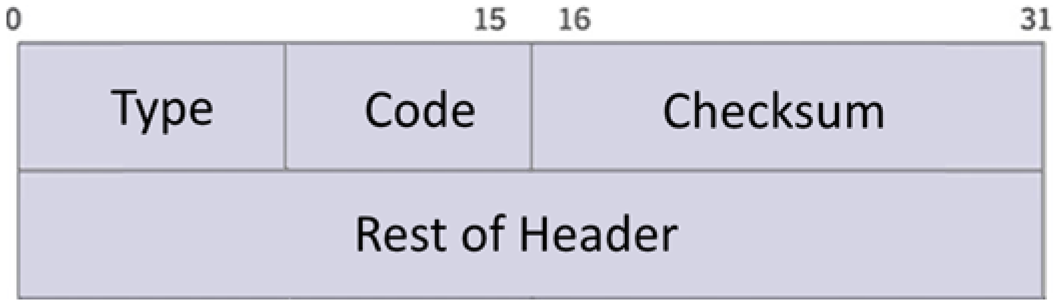
INTERNET CONTROL MESSAGE PROTOCOL

Internet Control Message Protocol

1981 RFC 792

- The ICMP is a network-layer Internet protocol that provides message packets to report errors and other information regarding IP packet processing back to the source.
- Best known for its use by the "ping" and "traceroute" programs on IP enabled hosts/devices.

ICMP Datagram



0 15 16 31

Type | Code | Checksum

Rest of Header

- Type and Code defines type and subtype of ICMP message
- Checksum is used for error checking. If it isn’t used field is full of zeros, but if checksum calculation gets 0x0000 the field value becomes 0xffff (-0 in hex)
- Rest of Header is a four-bytes field, which contents vary based on the ICMP type and code.

ICMP Types

- 0 and 8 - Echo reply and request used in ping
- 3 – Destination unreachable. Informs client that message can’t be delivered because of:
  - the physical connection to the host does not exist (distance is infinite)
  - the indicated protocol or port is not active
  - the data must be fragmented but the 'don't fragment' flag is on
  - and others
- 5 - Redirect Message requests data packets be sent on an alternative route. It is a signal for a host to update its routing information.

PORTS

Port

- Port is a communication endpoint. At the software level a port is a logical construct that identifies a specific process or a type of network service
- Ports are identified for each protocol and address combination by 16-bit unsigned numbers, commonly known as the port number.
- The most common protocols that use port numbers are TCP and UDP

Ports

- Well-known ports

  0 – 1023

- They are used by system processes that provide widely used types of network services

- Registered ports

  1024 – 49151

- They are assigned by IANA for specific service upon application by a requesting entity. On most systems, registered ports can be used without superuser privileges.

- Ephemeral ports

  49152 – 65535

- This range is used for private or customized services, for temporary purposes, and for automatic allocation

Well-known Ports

- Basic:
  - TCP:20,21 – FTP
  - 22 – SSH
  - 23 – Telnet
  - 53 – DNS
  - 67,68 – DHCP
  - 123 – NTP (Time)

- Email:
  - 25 – SMTP
  - 465 – Secure SMTP
  - 110 – POP3
  - TCP:995 – Secure POP3
  - 143 - IMAP
  - 993 – Secure IMAP

- Web:
  - 80 – HTTP
  - 443 – Secure HTTP

THANK YOU