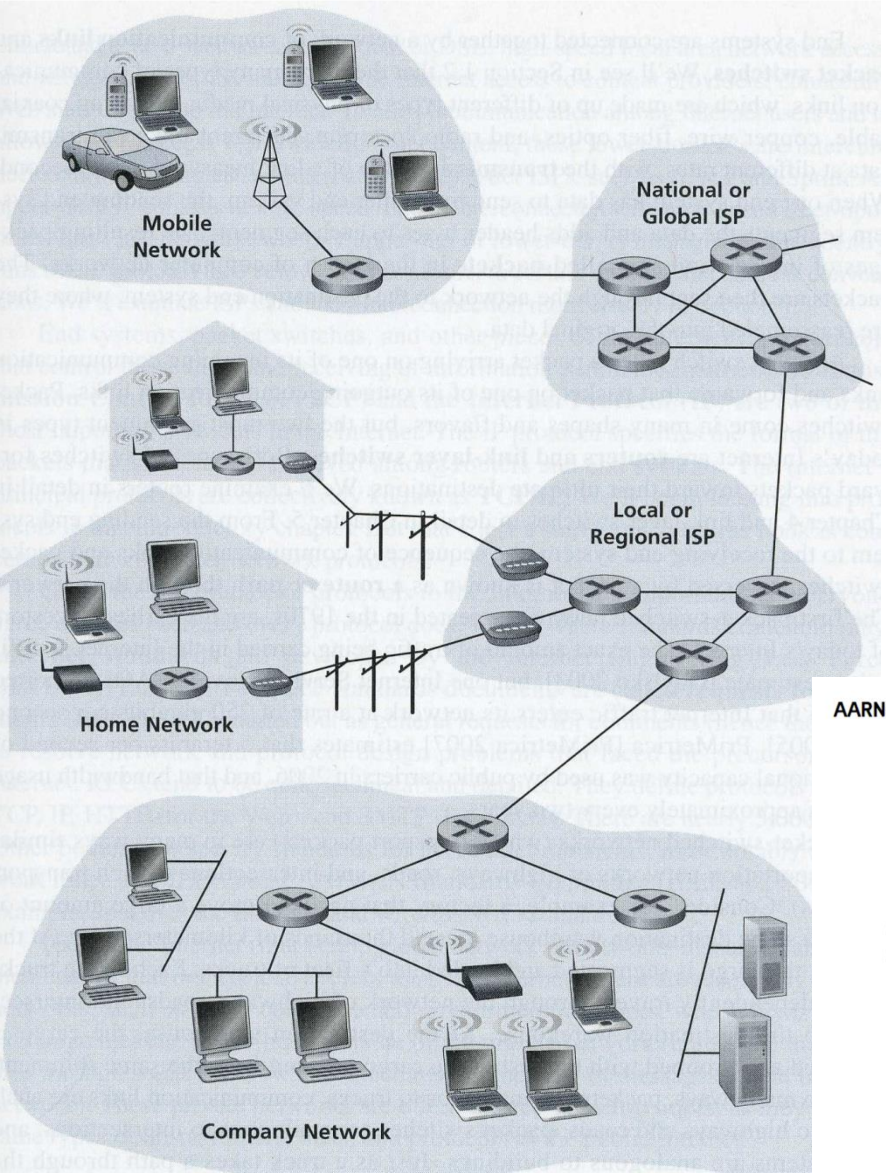
FIT5187 - Wireless Networks

Lecture 02: The Internet protocols

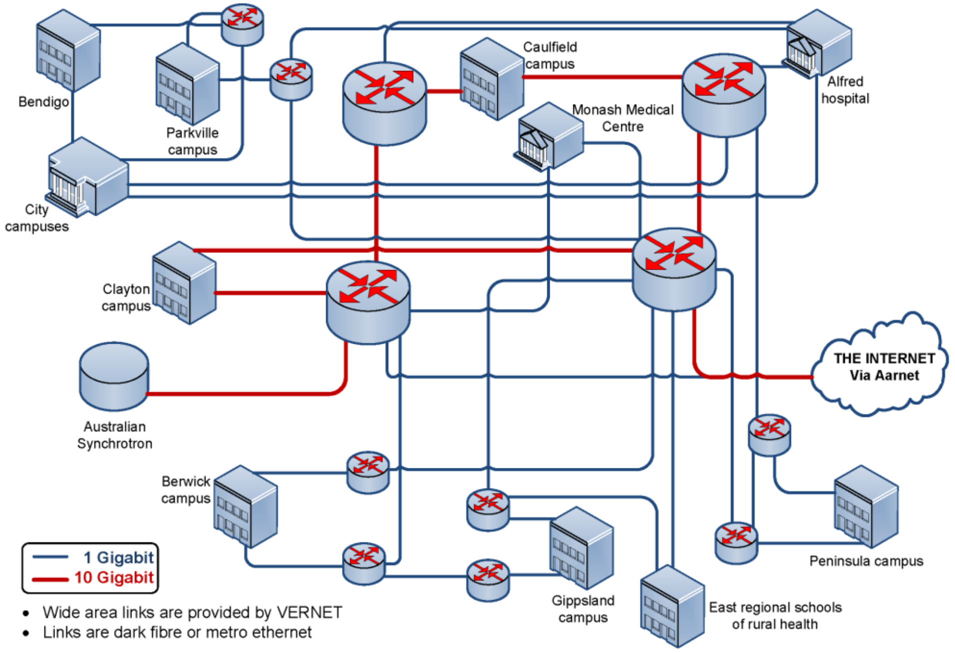
Learning outcomes:

- Understand the functions of each TCP/IP layer
- Understand how messages are moved through each layer
- Appreciate the need and how to follow protocol procedural steps
- Check the error free data transfer and data flow control
- Understand the rationale for the types of routing algorithm.

The Internet

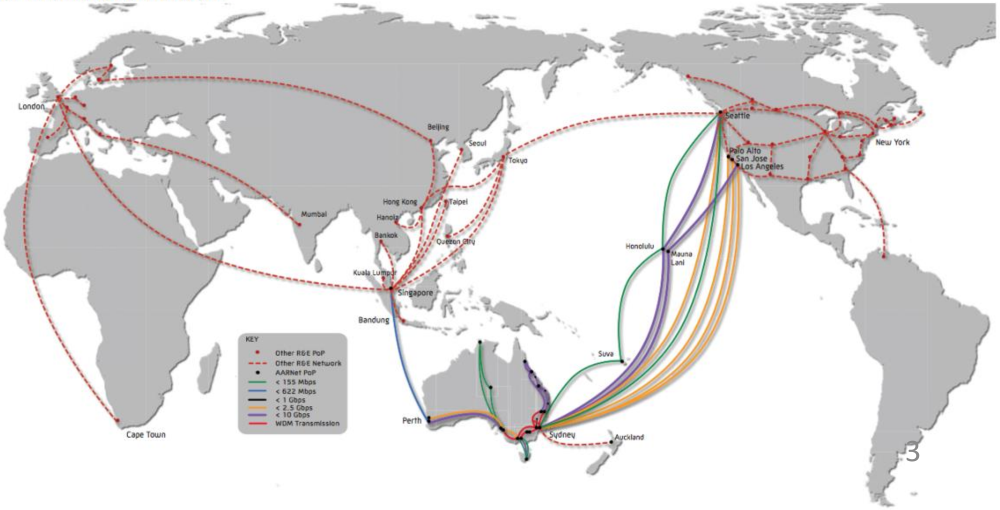


Overview of the Monash ITS Network Infrastructure



- Wide area links are provided by VERNET
- Links are dark fibre or metro ethernet

AARNet International Network



What is the Internet?

- The Internet is a global system of interconnected computer/communication networks that use the standard **Internet Protocol Suit – TCP/IP**
- More technically: the Internet is an interconnection of **routing computers** (routers)
- Three most fundamental services distributed over the Internet are:
  - World Wide Web – Interconnection of **Web servers**
  - Email – distributed by **mail servers**
  - Instant messages including VoIP – voice over IP, e.g. Skype, Google Talk, …

Network Models – the concept

- In order to manage the structural and functional complexity of the network the task of transferring the messages from the sender to the receiver is broken into a series of **layers**.
- Each layer provides a protocol – a set of basic functions and interfaces to the adjacent layers.
- The layers are, typically, dedicated software programs.
- The most important is the **Internet protocol suite** aka **TCP/IP model**
- Its communication layers are described in **RFC 1122** (Request For Comments) and related documents published by the **Internet Engineering Task Force**
- The Internet Protocols are compared with the **Open Systems Interconnection model (OSI model)**

7-Layer OSI model vs 5-layer Internet model

OSI Model	Internet Model	Functions
7. Application Layer	5. Application Layer	
6. Presentation layer		Set of utilities used by application programs
5. Session Layer		
4. Transport Layer	4. Transport Layer	Logical connection between sender and receiver, …
3. Network Layer	3. Network Layer Internet	Addressing, Routing, …
2. Data Link Layer	2. Data Link Layer	Medium Access control, …
1. Physical Layer	1. Physical Layer	Physical connection between sender and receiver

The **five-layer Internet Model** dominates current hardware and software and is the de facto standard.

Functions of each layer

Layer 5: Application Layer
(process-to-process)

- The application layer is the application software used by the network user.
- It is the user's access or interface to the network.
- By using the application software, the user defines what messages are sent over the network.
- Most common application layer software includes:
    - ➢ e-mail – SMTP protocol
    - ➢ web browsers – HTTP protocol
    - ➢ instant messages

Layer 4: The Transport Layer

Is responsible for:

- linking the application layer software to the network.
- establishing the end-to-end (host-to-host) connection between the sender and receiver when such connection is needed.
- breaking long messages into several smaller messages (**segments**) to make them easier to transmit.
- Provides the **data-flow** mechanism,
- Detects **lost and faulty** messages and requests that they be resent (**ARQ** - Automatic Repeat reQuest).

Layers 3: The Internet (aka Network) Layer

The network (internet) layer performs two functions:

1. Performs **routing**, selecting the next computer to which the message should be sent.
2. Finds the address of that computer if it doesn't already know it.

- Data is organized into **datagrams** also known as **IP packets**
- The internet layer does not guarantee the arrival of the packets
- Postal services analogy

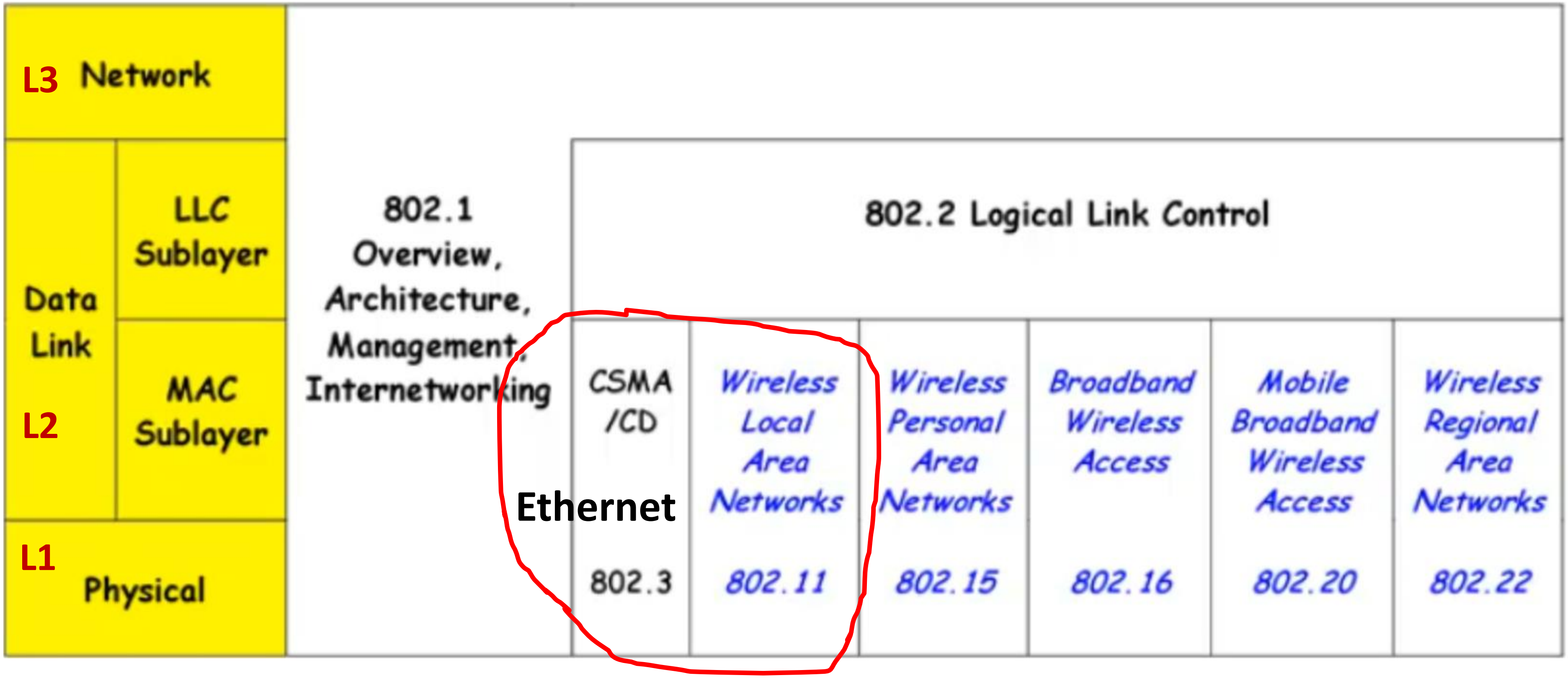
Layer 2: The Data Link Layer

- The data link layer is responsible for moving a message from one computer to the **next computer/router** in the network path from the sender to the receiver.
- The data link layer performs three basic functions:

  1. Controls the physical layer by deciding **when** to transmit messages over the media (Media Access Control)
  2. Formats the messages (vectors/strings of bits) by indicating where they start and end.
  3. Detects and corrects any errors that have occurred during transmission.

- Data Link Layer operates with **MAC/Physical addresses**
- Bits are organized into **frames** sometimes called **packets**.

Related IEEE 802 Standards



<table>
<tr><td colspan="2">L3 Network</td><td rowspan="4">802.1
Overview,
Architecture,
Management,
Internetworking</td><td colspan="6"></td></tr>
<tr><td rowspan="2">Data
Link
L2</td><td>LLC
Sublayer</td><td colspan="6">802.2 Logical Link Control</td></tr>
<tr><td>MAC
Sublayer</td><td rowspan="2">CSMA
/CD
Ethernet
802.3</td><td rowspan="2">Wireless
Local
Area
Networks
802.11</td><td rowspan="2">Wireless
Personal
Area
Networks
802.15</td><td rowspan="2">Broadband
Wireless
Access
802.16</td><td rowspan="2">Mobile
Broadband
Wireless
Access
802.20</td><td rowspan="2">Wireless
Regional
Area
Networks
802.22</td></tr>
<tr><td colspan="2">L1
Physical</td></tr>
</table>

Note that the Data Link Layer is typically divided into two sublayers:

- Logical Link Control (LLC)
- Media Access Control (MAC)

Moving messages through layers (Network)

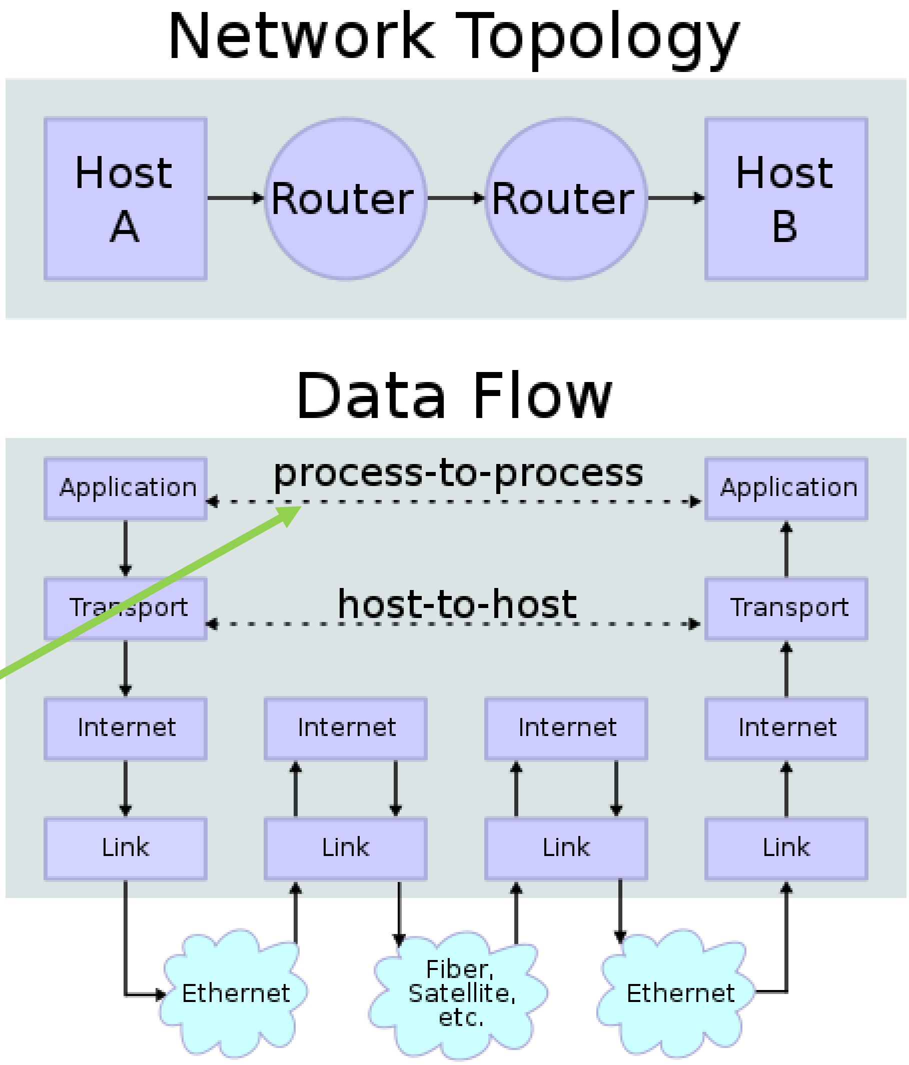
Internet Protocol Model

Layer 1: The Physical Layer

- The physical layer is the physical connection between the sender and receiver.
- Its role is to transfer a series of electrical, **radio**, or light **signals** representing **bits** of information through the circuit.
- The physical layer typically includes all the hardware devices needed to send signals through the physical media (e.g., modems, cables, antennas, …).
- We will study in some depth the physical layer (PHY) of the WiFi, IEEE 802.11.
- Modern mobile/cellular networks use similar physical layer.

Moving messages through layers (Application)

- Two Internet host computers communicate across local network boundaries constituted by their internetworking (or border) routers.
- The application on each host executes read and write operations as if the processes were directly connected to each other by some kind of data pipe.
- Detail of the communication is hidden from each application process.



Network Topology

Data Flow

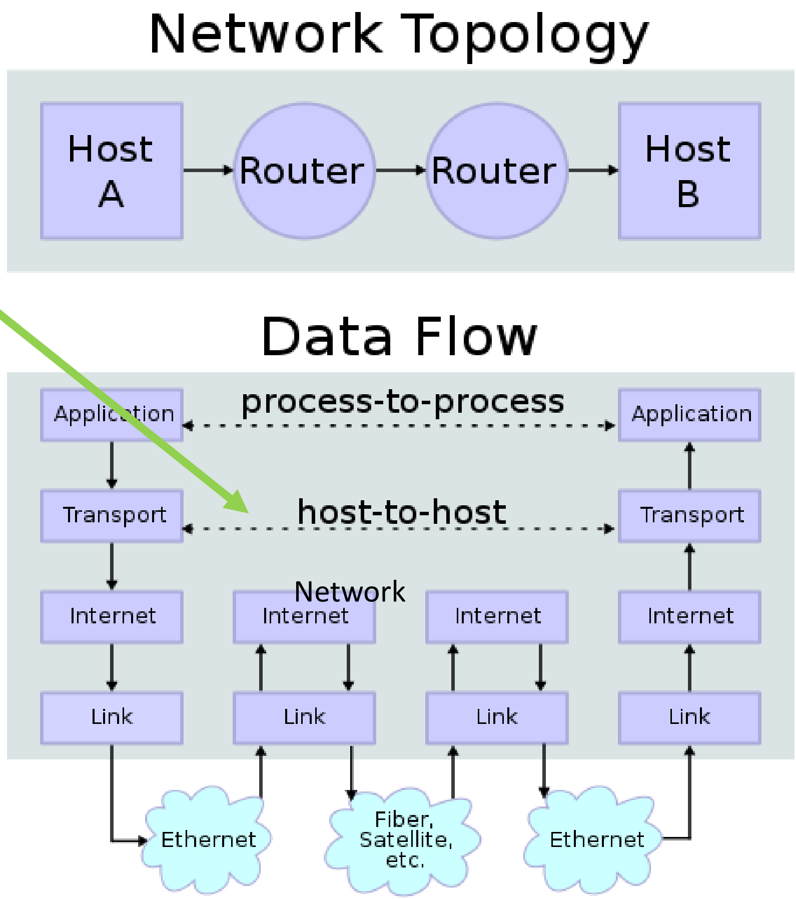
From Wikipedia

Moving messages through layers (Transport 1)

- The Transport Layer establishes host-to-host connectivity,

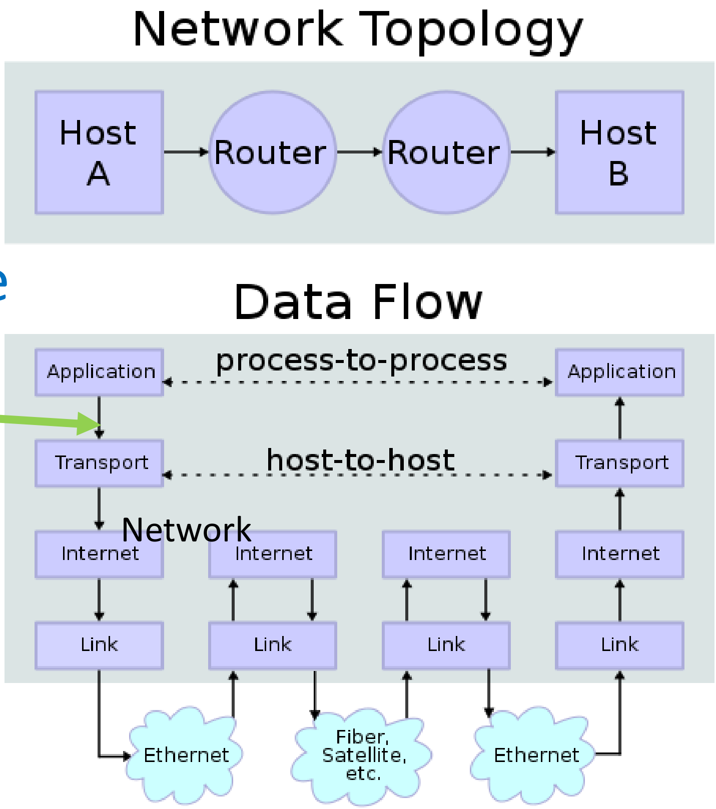
It handles

- the details of data transmission that are independent of the structure of user data (e.g. photo, text, …)
- the logistics of exchanging information for any particular specific purpose.



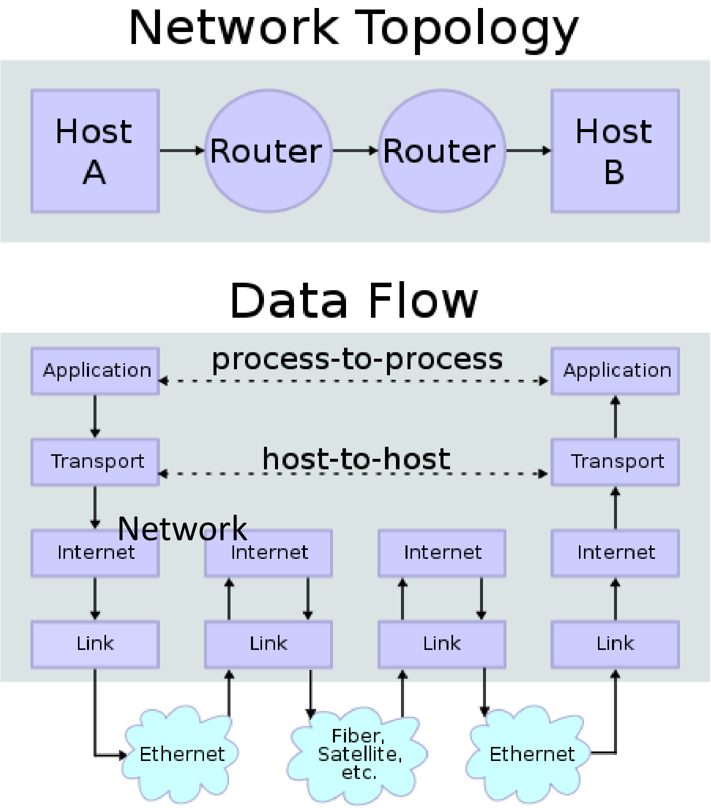
Moving messages through layers (Transport 2)

- The layer establishes a basic data channel that an application uses in its task-specific data exchange.
- For this purpose the layer establishes the concept of the ***port***, a number allocated specifically for each of the communication channels an application needs.
- For many types of services, these port numbers have been standardized (e.g. port 80 for web servers) so that client computers may address specific services of a **server computer (e.g. web server)** without the involvement of service announcements or directory services.



Moving messages through layers (Network)

- The Internet (or Network) Layer provides an unreliable datagram transmission facility between hosts located on potentially different IP networks
- It forwards the Transport Layer segments to an appropriate **next-hop** router for further relaying to its destination
- The Internet Protocol (IP) operates with two addressing systems:
    - the IP addresses
    - the physical addresses,
- Two addresses identify network hosts computers, and to locate them on the network.
- Two versions of the Internet Protocol are in use: IPv4 and IPv6

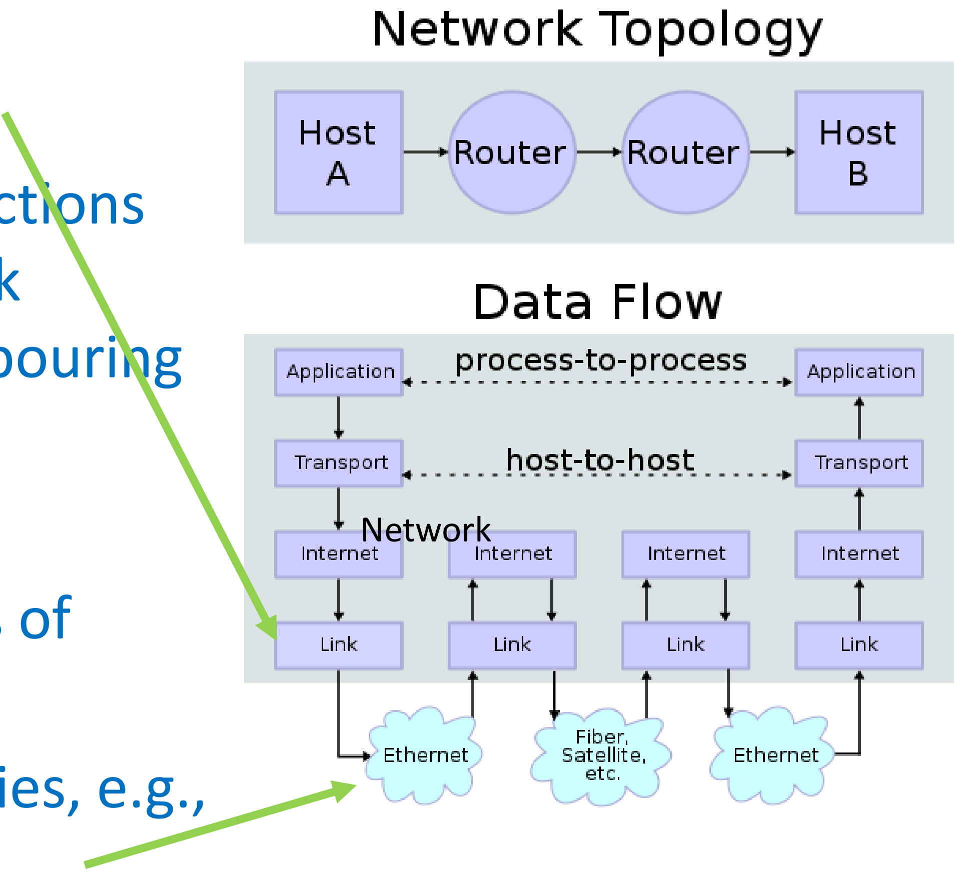


Moving messages through layers (Link)

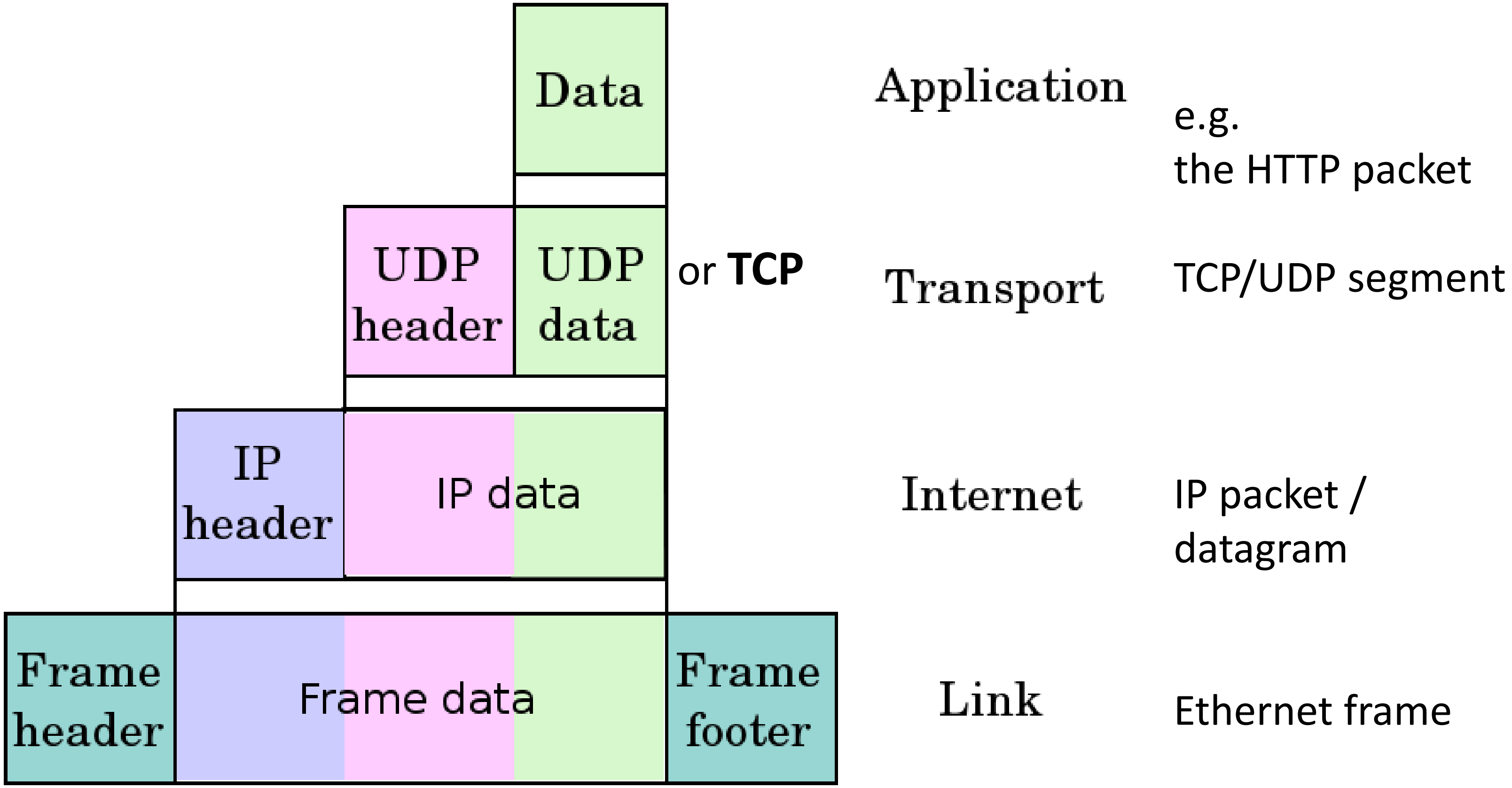
- The lowest layer in the Internet Protocol Suite is the Link Layer.
- The link layer describes the functions of the local link, i.e. the network segment connecting two neighbouring hosts.

This involves interacting with

- the hardware-specific functions of network interfaces and
- specific transmission technologies, e.g., 802.3 Ethernet, 802.11 WLAN, …



Successive Encapsulation of messages



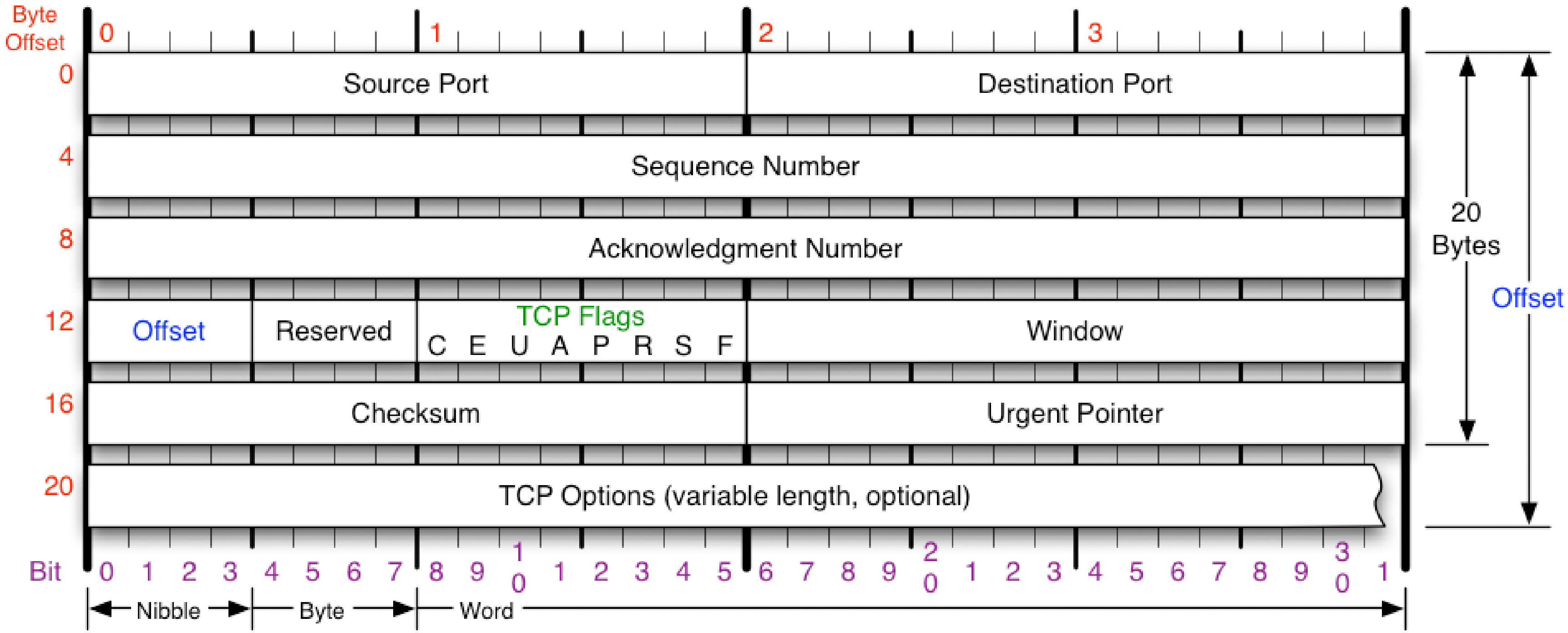
- The message created by the application layer, e.g. , is successfully encapsulated by the lower level protocols.
- Typically a header, specific for the given layer, is added

Protocols (procedural steps)

Transport layer protocols

- Transport layer protocols include:
  - **TCP**: Transmission Control Protocol. Designed to deliver reliable end-to-end transmission of data
  - **UDP**: User Datagram Protocol. A very simple protocol used in Domain Name Systems (DNS), VoIP applications, online games etc. Reliability is sacrificed for speed.
  - **SCTP**: Stream Control Transmission Protocols
  - **RSVP**: Resource reservation protocol used in multicast and unicast applications
  - others

TCP Header



- **Source** (sending) and **Destination** (receiving) ports
- **Sequence number**: the first data byte = Seq# + SYN flag
- **Acknowledgement number**: if ACK then Ack# = byte# that the receiver is expecting.
- **Data offset**: Header size in 32-bit words (5..20)
- **Checksum**: 16-bit checksum of the header and data
- 16-bit **receive Window**: # bytes that the receiver is currently willing to receive

Flags (aka Control bits):

CWR – Congestion Window Reduced
ECE – ECN-Echo (RFC 3168).
URG – the **URGent Pointer** field is significant
ACK – indicates that the ACKnowledgment field is significant
PSH – Push function
RST – Reset the connection
SYN – Synchronize sequence numbers
FIN – No more data from sender

TCP Connection Establishment – passive open

- Before a client attempts to connect with a server, the server must first bind to a **port** to open it up for connections: this is called a **passive open**.
- Once the passive open is established, a client may initiate an **active open** by exchanging frames that form a **three-way handshake** procedure.

Connection Establishment

To establish a connection, the **three-way handshake** occurs:

1. **SYN**: The active open is performed by the client sending a frame with SYN to the server. It sets the segment's sequence number to a random value A.
2. **SYN-ACK**: In response, the server replies with a SYN-ACK. The ack. number is set to one more than the received sequence number (A + 1), and the sequence number that the server chooses for the packet is another random number, B.
3. **ACK**: Finally, the client sends an ACK back to the server. The sequence number is set to the received acknowledgement value i.e. A + 1, and the ack. number is set to one more than the received sequence number i.e. B + 1.

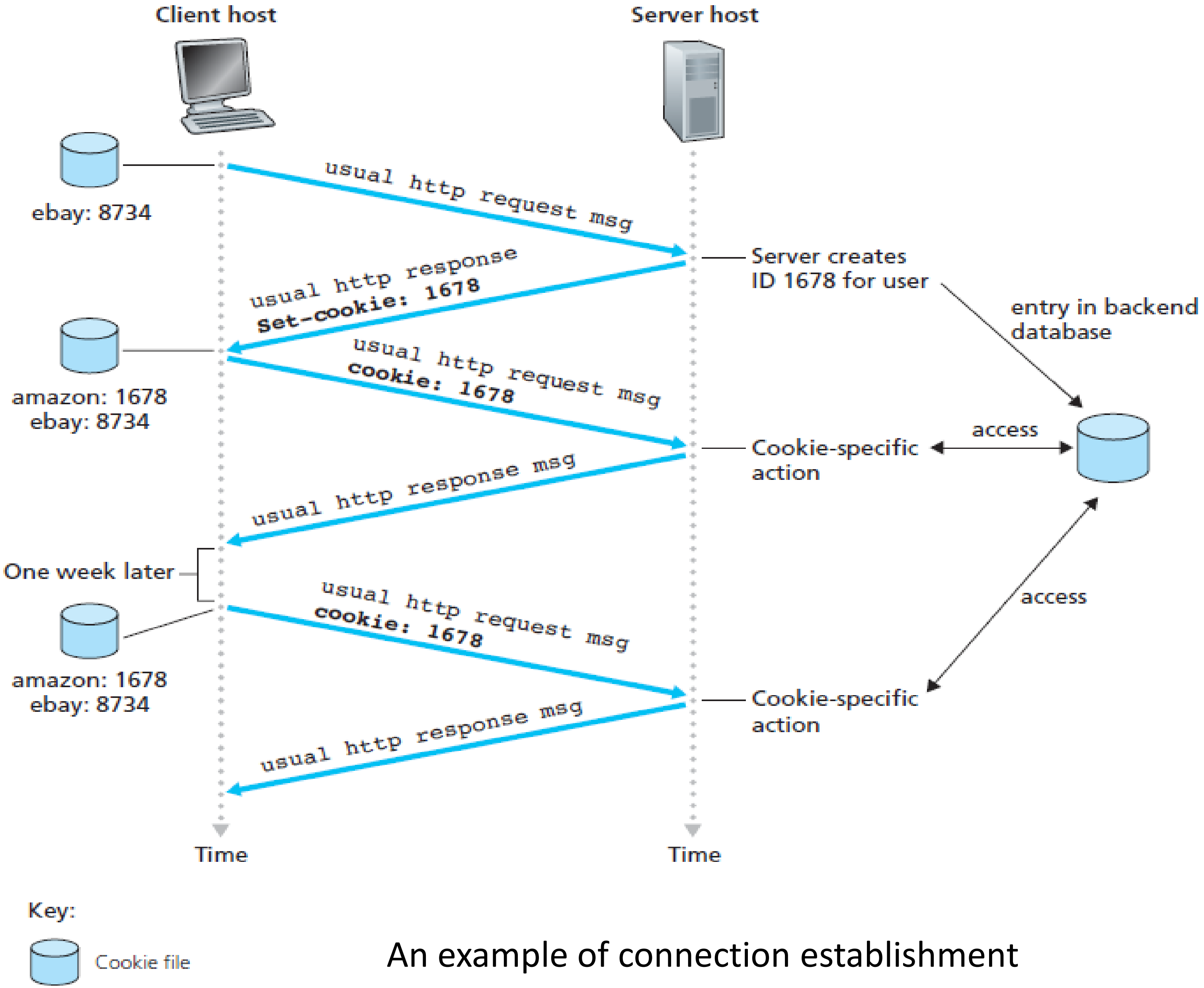
At this point, both the client and server have received an acknowledgment that the connection has been established.

TCP Data Transfer: Data and ACK frames (1)

- After the TCP connection between two hosts have been established, the hosts (Sender and receiver) exchange **data and ACKnowledgment frames.**
- The TCP software receives from the application layer data to be sent, e.g., a photo containing 1.5MB
- The data is segmented into the blocks acceptable by the data link layer. It is typically 1.5 kB for the Ethernet frame.
- The segments are continuously sent to the receiver using the sequence numbers

TCP Data Transfer: Data and ACK frames (2)

- The receiver acknowledges a group of received segments/bytes using the acknowledgment number and the **sliding window** mechanism.
- Data frames are being transmitted by the IP layer.
- Due to complexity of the Internet, there is no guarantee that the data frames will arrive to the destination in the same order that they have been sent.
- The receiver rearranges data packets according to the sequence number



An example of connection establishment

Error free data transfer

ARQ – Error-Free Date Transfer.

- The TCP layer must guarantee the error-free data transfer.
- The receiver tests the correctness of the received packets by re-calculating and comparing the checksum field in the TCP header.
- If the packet is incorrect, or lost, that is, any cumulative stream of data is not acknowledged, the hosts generate

  the **ARQ** – **A**utomatic **R**epeat re**Q**est frame
- The incorrect or lost frame is retransmitted.
- The ACK frames must arrive within a set time limit (time-out mechanism)
- Note an almost infinite number of situations with lost, incorrect and retransmitted frames.

Data Flow Control

Flow control 1

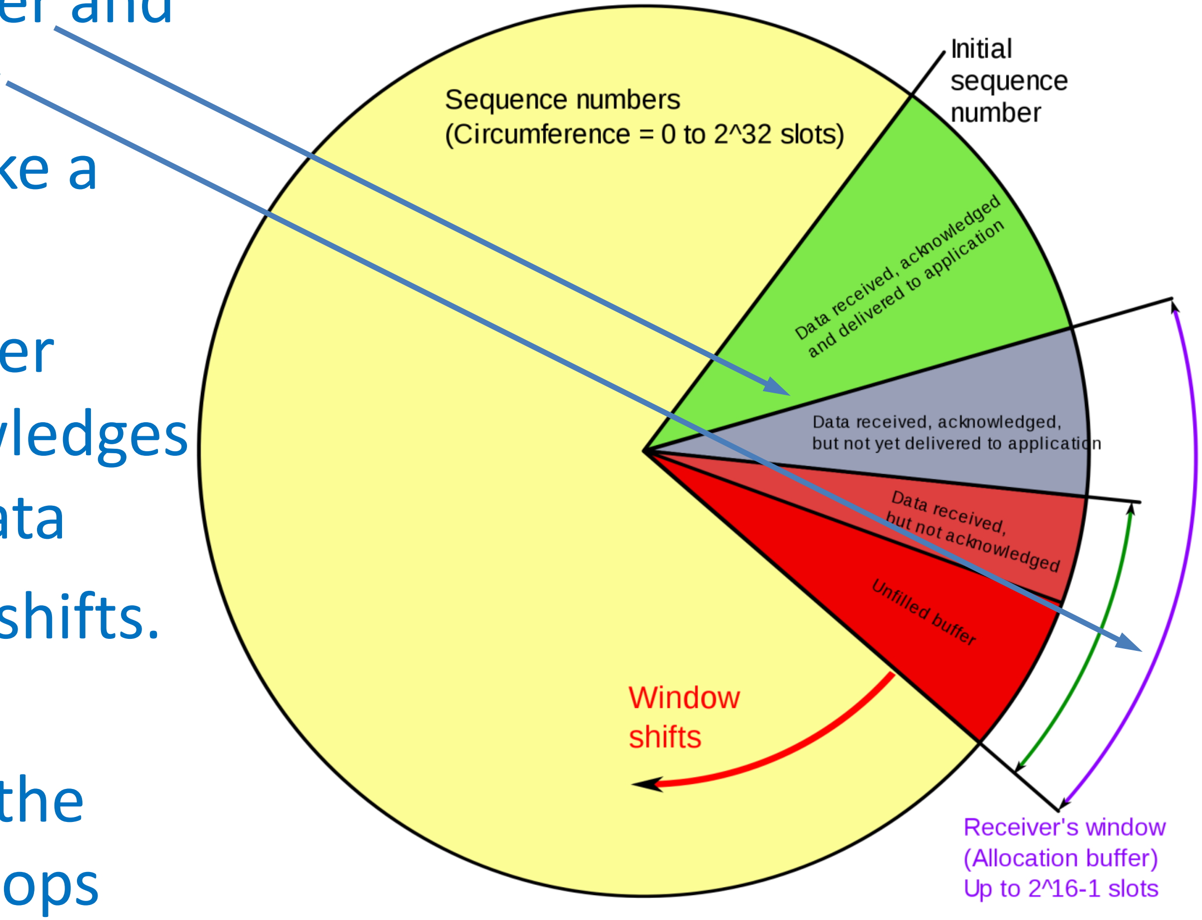
- TCP uses an end-to-end **flow control** protocol to avoid
  - having the sender send data too fast
  - for the TCP receiver to receive and process it reliably.
- Having a mechanism for flow control is essential in an environment where machines of diverse network speeds communicate.
- For example, if a fast server sends data to a mobile phone that is unable to process received data fast, the mobile phone must regulate data flow so as not to be overwhelmed.
- **Flow control** – limits the rate a sender transfers data to guarantee reliable delivery.

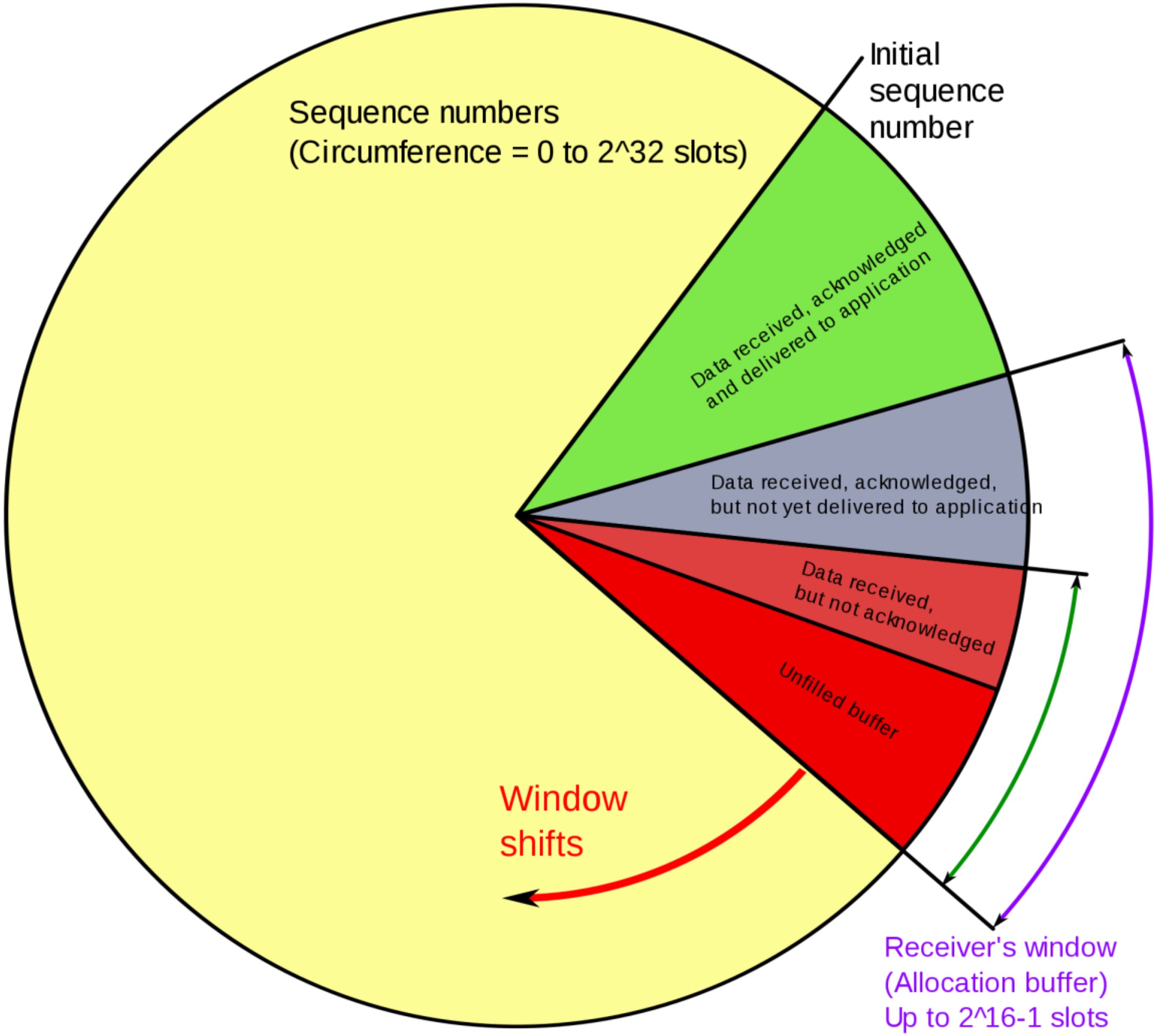
Flow control 2

- TCP uses a **sliding window** flow control protocol.
- Unlike the stop-and-wait flow control
- In each TCP segment,
  - the receiver specifies in the **receive window** field
  - the amount of additional received data (in bytes) that it is willing to buffer for the connection.
- The sending host can send only up to that amount of data
  - before it must wait for an acknowledgment
  - and for the window update from the receiving host.
- The receiver continually hints the sender on how much data can be received
- When the receiving host's buffer fills, the next acknowledgment contains a 0 in the window size, to stop transfer and allow the data in the buffer to be processed.

Flow control 3

- TCP sequence number and the receiver window behave very much like a clock.
- Each time the receiver receives and acknowledges a new segment of data
- ➢ the receive window shifts.
- Once it runs out of sequence numbers, the sequence number loops back to 0.
- Next slide:

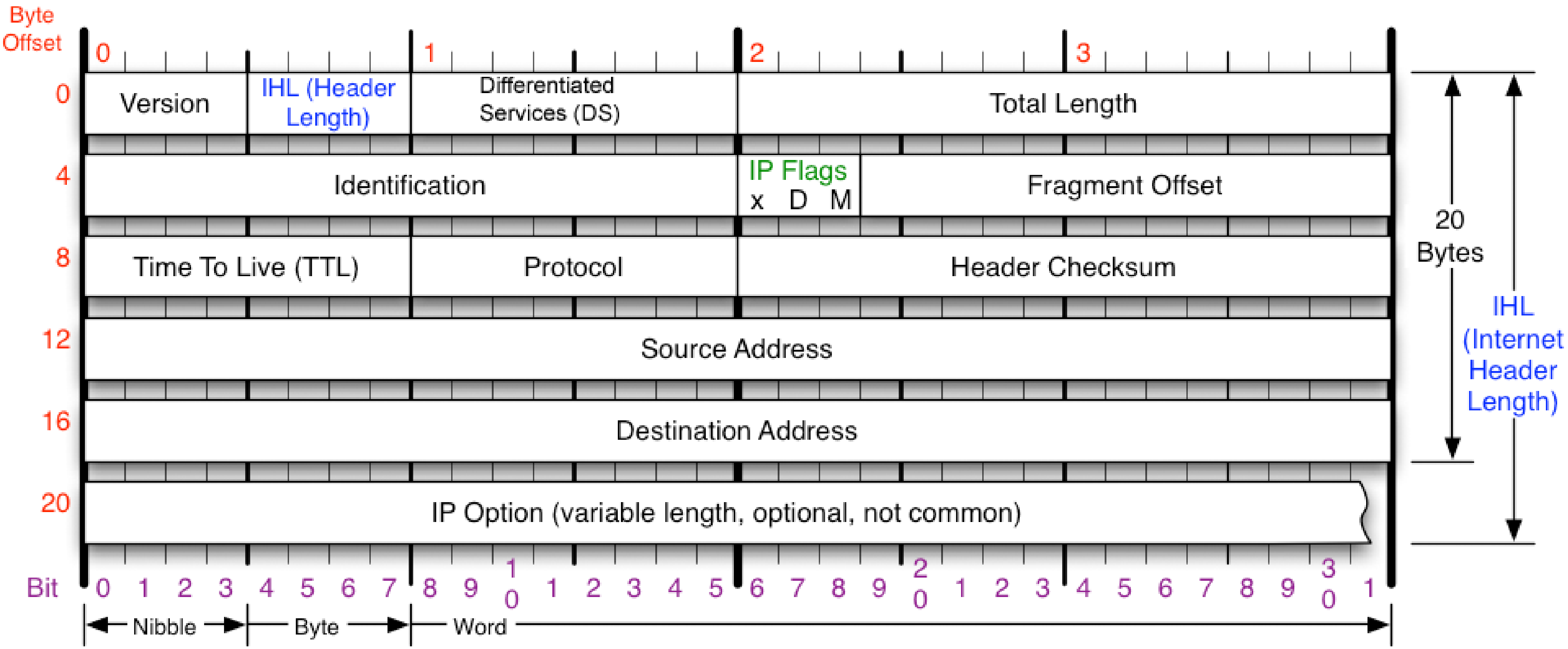

Initial sequence number
Sequence numbers (Circumference = 0 to 2^32 slots)
Data received, acknowledged and delivered to application
Data received, acknowledged, but not yet delivered to application
Data received, but not acknowledged
Unfilled buffer
Window shifts
Receiver's window (Allocation buffer) Up to 2^16-1 slots


Sequence numbers
(Circumference = 0 to 2^32 slots)
Initial
sequence
number
Data received, acknowledged
and delivered to application
Data received, acknowledged,
but not yet delivered to application
Data received,
but not acknowledged
Unfilled buffer
Window
shifts
Receiver's window
(Allocation buffer)
Up to 2^16-1 slots

Internet Protocol (IP)

- The **Internet Protocol** (**IP**) is the principal communications protocol used for relaying datagrams (packets) across the Internet.
- The Internet Protocol is responsible for
  - **addressing** hosts and
  - **routing** datagrams from a source host to the destination host across one or more IP networks.
- The IP is a connection-less protocol and its services are unreliable, and the delivery can face:
  - data corruption,
  - lost data packets,
  - duplicate arrival, and
  - out-of-order packet delivery.
- Two versions of the IP protocol exist: IPv4 and IPv6

IPv4 Header



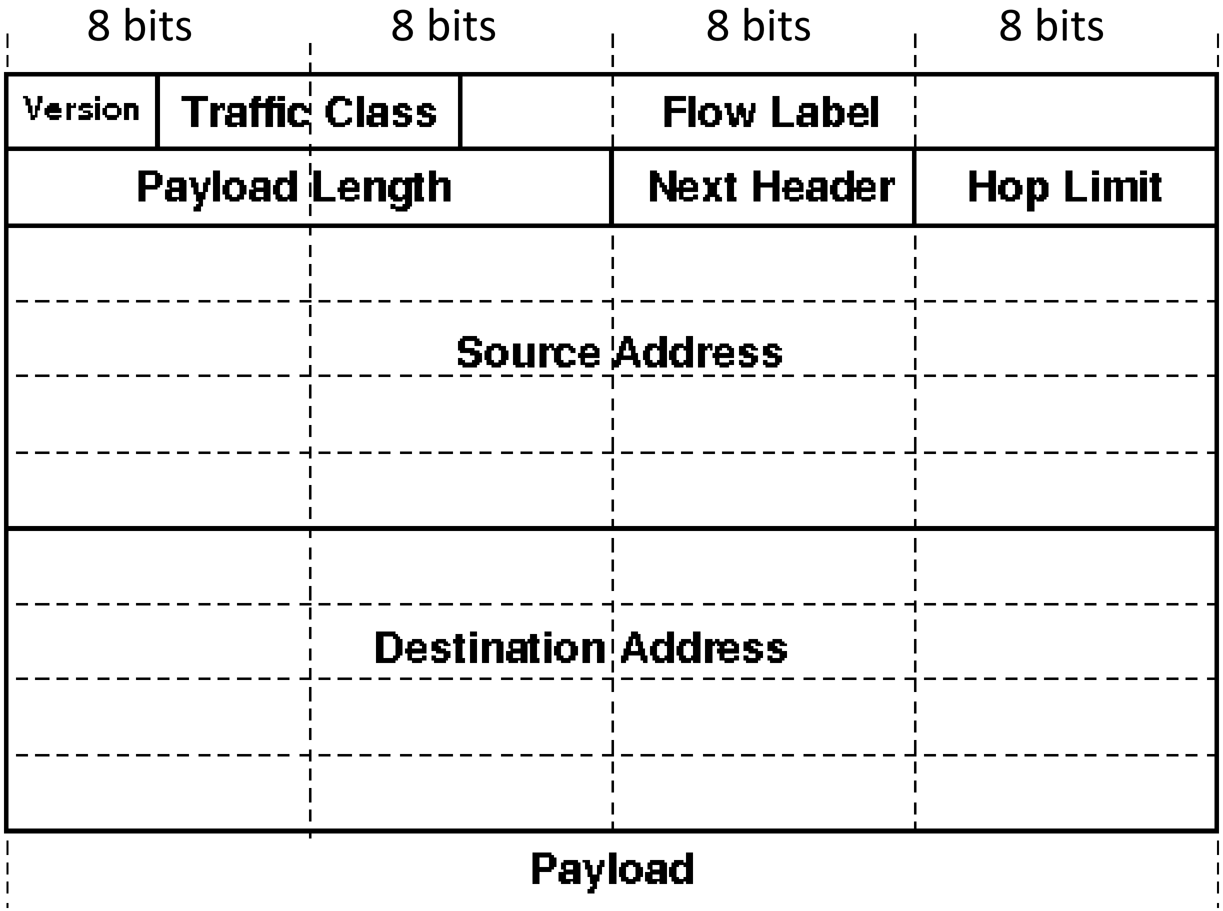
- ➢**Version:** 4 (or 6)
- ➢**DiffServ** (DS): type of service (e.g. VoIP)
- ➢**Total Length:** 16-bit, header+data,
  min 20 bytes, max 65,535 bytes

Read in Wikipedia about:

- ➢**Identification**
- ➢**IP Flags**
- ➢**Fragment Offset**
- ➢**Protocol** used in the data portion (TCP, UDP,)

- ➢**Header Checksum**
- ➢**Time to live (TTL)** aka **Hop limit:** maximum number of routers that the packet can be sent through.
- ➢**Source and Destination Addresses:** 32-bit (4-byte) IP addresses that identify the sender and receiver of the packet, e.g. 130.194.15.1

IPv6 (RFC 2460) Header


8 bits
8 bits
8 bits
8 bits
Version
Traffic Class
Flow Label
Payload Length
Next Header
Hop Limit
Source Address
Destination Address
Payload

- **Version**: 4 bits = 6 (IPv6)
- **Traffic Class**: 8-bit e,g. VoIP
- **Flow Label**: 20-bits. A sequence of packets requiring a special handling
- **Payload Length**: 16-bit length of the payload in bytes
- **Next Header**: 8-bit. Identifies the type of header immediately following the IPv6 header.
- **Hop Limit**: 8-bit. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.
- **Source and Destination addresses**: 16-byte addresses. Typically written in hexadecimal form, e.g. FEDC:BA98:7654:3210:FEDC:BA98:7654:3210. See RFC 2373 for details.

Addressing

Three levels of addressing:

- Host’s domain names, e.g. www.monash.edu

  used by the application layer
- IP addresses, e.g. 130.194.15.1, (4 bytes) or 2001:388:608c:8a3::95 (16 bytes)

  used by the network/internet layer
- MAC (Data Link, or Physical) addresses,

  e.g. 00-23-24-0B-31-F1 (6 bytes)

  a hardware-fixed address of the network card.

  Used in the next-computer delivery

Address Resolution

- **Server Name Resolution**: Translating destination host's domain name to its corresponding IP address

  Performed by Domain Name Service (DNS) servers

- **Data Link Layer Address Resolution:**
  - Identifying the MAC address of the next node (that packet must be forwarded to)
  - Uses Address Resolution Protocol (ARP)

Routing

Routing Fundamentals

- **Routing** is a process of selecting a path in the network for a packet to be sent from the source to the destination.
- It involves two basic activities:
  - **determining paths** for routing and
  - **transporting (forwarding)** packets through the networks.
- A router examines the incoming packet and use the destination IP address to find
  - the entry in the **forwarding/routing table** that gives
  - the data link address of the next router in the path.

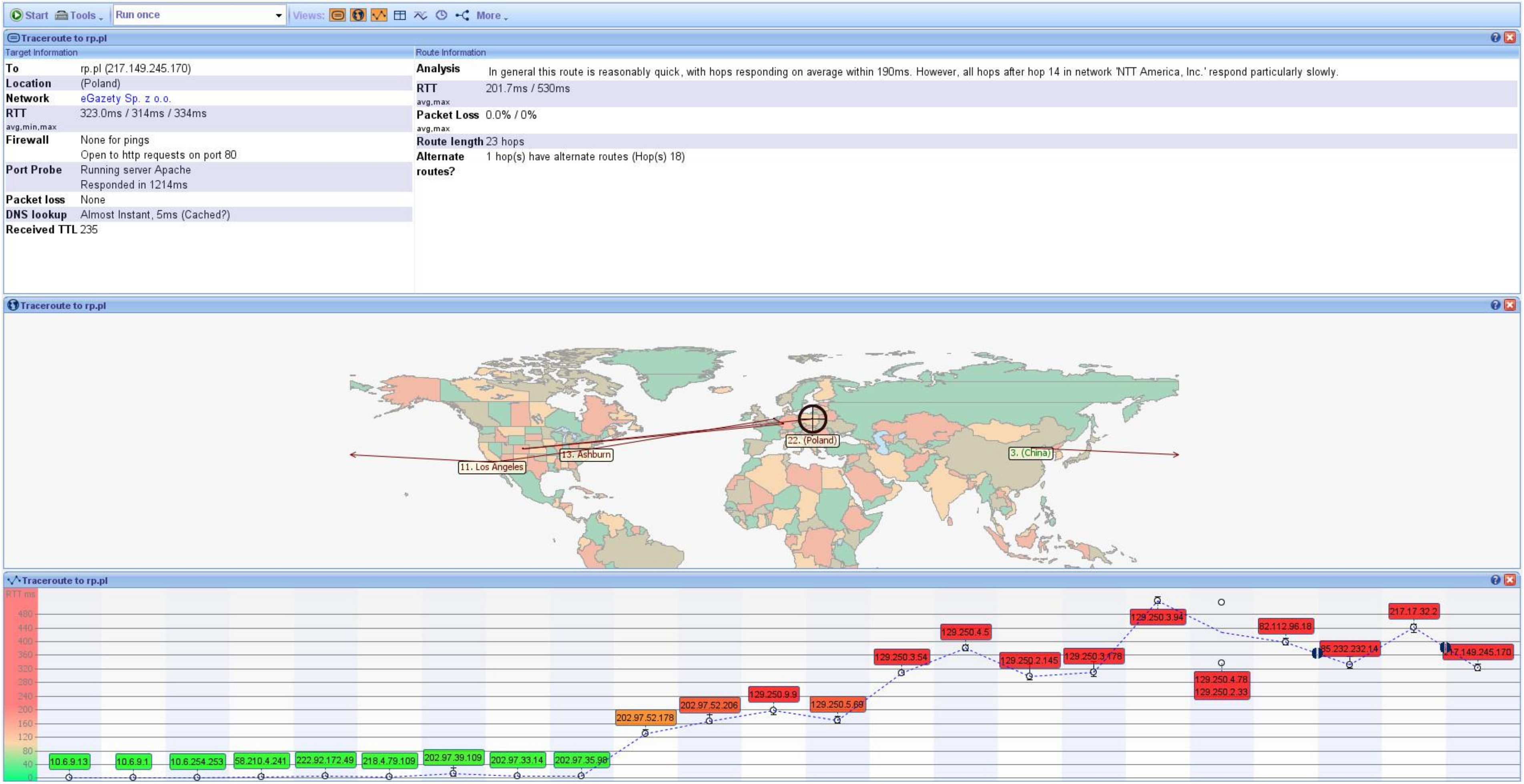
Static and Dynamic Routing

- **Static routing** implies that the forwarding tables are prepared manually.
- In a small network the choice of the next router is limited.
- **Dynamic or adaptive routing** implies that
  - the forwarding table is dynamically adjusted by
  - **routing algorithms** that use specific
  - **routing metrics** to find an optimal path through the network.

Routing algorithms

- Two basic routing metrics are based on:
  - The distance: Distance-vector routing protocol
  - The traffic reports: Link-state routing protocol
- Distance vector and link state routing are both
  - **intra-domain or interior routing protocols**.
- They are used inside an **autonomous system**, but are impractical between autonomous systems.
- Autonomous systems are linked together by
  - the **border routers** that typically use the distance vector metrics.

Internet connection from SEU to rp.pl



Traceroute to rp.pl

Target Information	
To	rp.pl (217.149.245.170)
Location	(Poland)
Network	eGazety Sp. z o.o.
RTT avg,min,max	323.0ms / 314ms / 334ms
Firewall	None for pings Open to http requests on port 80
Port Probe	Running server Apache Responded in 1214ms
Packet loss	None
DNS lookup	Almost Instant, 5ms (Cached?)
Received TTL	235

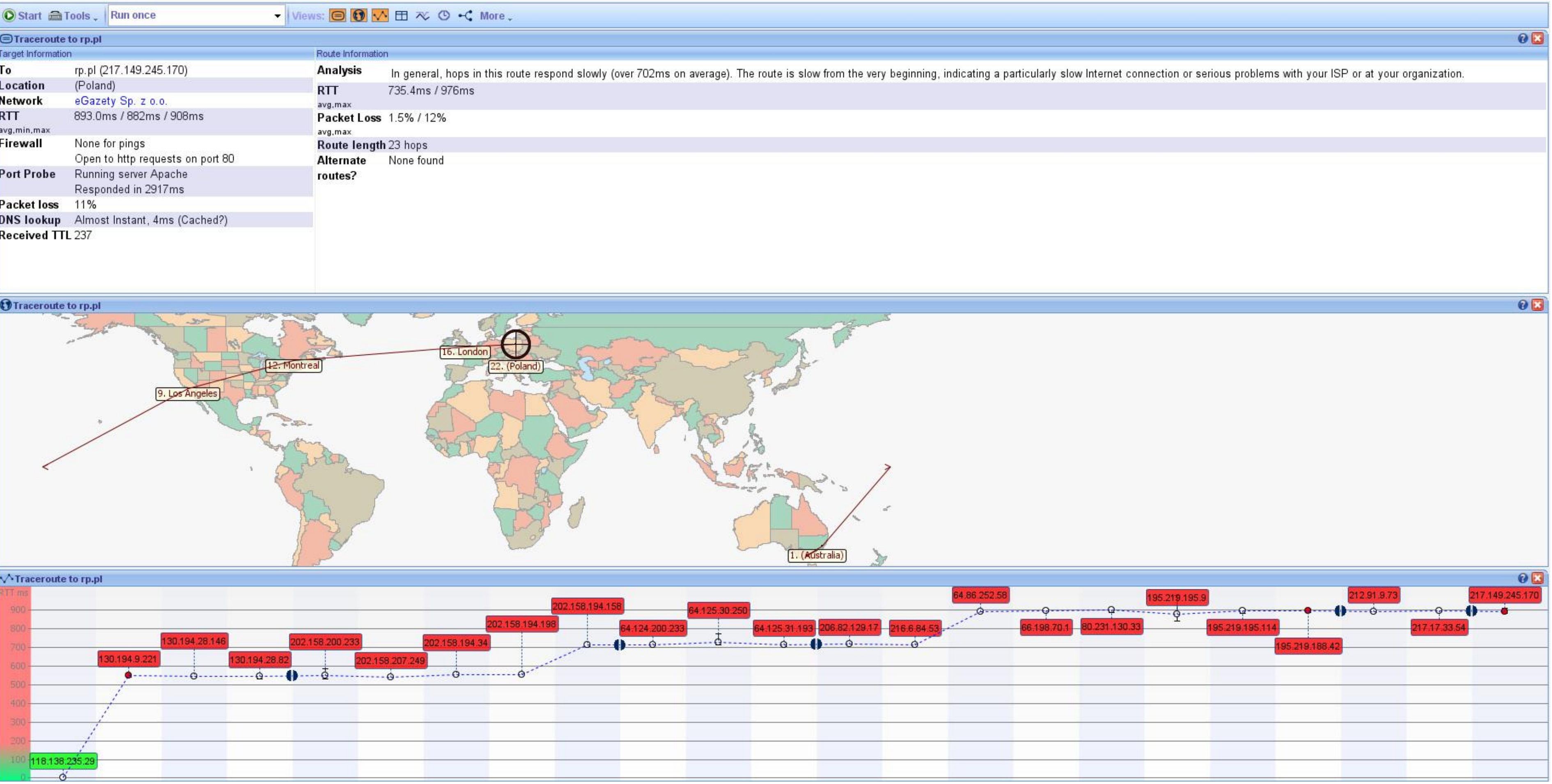
Route Information	
Analysis	In general this route is reasonably quick, with hops responding on average within 190ms. However, all hops after hop 14 in network 'NTT America, Inc.' respond particularly slowly.
RTT avg,max	201.7ms / 530ms
Packet Loss avg,max	0.0% / 0%
Route length	23 hops
Alternate routes?	1 hop(s) have alternate routes (Hop(s) 18)

Internet connection from SEU to rp.pl

```
Tracing route to rp.pl [217.149.245.170]     over a maximum of 30 hops:

  1     1 ms    <1 ms    <1 ms  10.6.9.1
  2    <1 ms    <1 ms    <1 ms  10.6.254.253
  3     2 ms     1 ms     3 ms  58.210.4.241
  4     2 ms     1 ms     2 ms  222.92.172.49
  5     7 ms     3 ms     2 ms  218.4.79.109
  6     6 ms     7 ms     6 ms  202.97.39.109
  7     5 ms     4 ms     5 ms  202.97.33.14
  8     6 ms     8 ms     6 ms  202.97.35.98
  9   129 ms   127 ms   128 ms  202.97.52.178
 10   176 ms   161 ms   163 ms  202.97.52.206
 11   190 ms   186 ms   185 ms  xe-0-1-0-9.r04.lsanca03.us.bb.gin.ntt.net [129.250.9.9]
 12   263 ms   257 ms   261 ms  ae-6.r21.lsanca03.us.bb.gin.ntt.net [129.250.5.69]
 13   376 ms   408 ms   402 ms  ae-2.r20.asbnva02.us.bb.gin.ntt.net [129.250.3.54]
 14   369 ms   375 ms   403 ms  ae-0.r21.asbnva02.us.bb.gin.ntt.net [129.250.4.5]
 15   300 ms   299 ms   288 ms  ae-2.r23.amstnl02.nl.bb.gin.ntt.net [129.250.2.145]
 16   342 ms   302 ms   305 ms  ae-1.r20.frnkge04.de.bb.gin.ntt.net [129.250.3.178]
 17   521 ms   514 ms   516 ms  ae-2.r02.frnkge04.de.bb.gin.ntt.net [129.250.3.94]
 18   521 ms   330 ms   504 ms  xe-4-1.r00.wrswpl01.pl.bb.gin.ntt.net [129.250.4.78]
 19   397 ms   392 ms   393 ms  atm-0.r00.wrswpl01.pl.bb.gin.ntt.net [82.112.96.18]
 20   377 ms   342 ms   338 ms  lt-9-3-0-31.r7.isp-r7.glo.atman.pl [85.232.232.14]
 21   443 ms   448 ms   447 ms  ge-3-0-0-3990.r2.isp-r7.isp.atman.pl [217.17.32.2]
 22   318 ms   311 ms   319 ms  host-217.149.245.170.parkiet.com [217.149.245.170]
```

Internet connection from SEU to rp.pl with VPN to monash.edu



Traceroute to rp.pl

Target Information	
To	rp.pl (217.149.245.170)
Location	(Poland)
Network	eGazety Sp. z o.o.
RTT avg,min,max	893.0ms / 882ms / 908ms
Firewall	None for pings Open to http requests on port 80
Port Probe	Running server Apache Responded in 2917ms
Packet loss	11%
DNS lookup	Almost Instant, 4ms (Cached?)
Received TTL	237

Route Information	
Analysis	In general, hops in this route respond slowly (over 702ms on average). The route is slow from the very beginning, indicating a particularly slow Internet connection or serious problems with your ISP or at your organization.
RTT avg,max	735.4ms / 976ms
Packet Loss avg,max	1.5% / 12%
Route length	23 hops
Alternate routes?	None found

Traceroute to rp.pl

Traceroute to rp.pl

Internet connection from SEU to rp.pl with VPN to monash.edu

```
1   562 ms   553 ms   551 ms  clay1-gw-v303.net.monash.edu.au [130.194.9.221]
 2   549 ms   549 ms   547 ms  clay0-gw-t2-4.net.monash.edu.au [130.194.28.146]
 3   535 ms   542 ms   539 ms  monash1-gw-v520.net.monash.edu.au [130.194.28.82]
 4   541 ms   548 ms   534 ms  gigabitethernet1.er1.monash.cpe.aarnet.net.au [202.158.200.233]
 5   520 ms   535 ms   539 ms  ge-2-1-0.bb1.a.mel.aarnet.net.au [202.158.207.249]
 6   546 ms   557 ms   553 ms  so-0-1-0.bb1.a.syd.aarnet.net.au [202.158.194.34]
 7   561 ms   547 ms   542 ms  ge-0-0-0.bb1.b.syd.aarnet.net.au [202.158.194.198]
 8   709 ms   712 ms   711 ms  so-3-0-0.bb1.a.lax.aarnet.net.au [202.158.194.158]
 9   713 ms   708 ms   715 ms  10.ge-9-3-8.er2.lax112.us.above.net [64.124.200.233]
10   711 ms   710 ms   704 ms  xe-0-1-0.cr2.lax112.us.above.net [64.125.30.250]
11   716 ms   700 ms     *     xe-2-0-0.mpr1.lax12.us.above.net [64.125.31.193]
12   715 ms   707 ms   717 ms  Vlan521.icore1.EQL-LosAngeles.as6453.net [206.82.129.17]
13   704 ms   695 ms   691 ms  if-4-28.tcore2.LVW-LosAngeles.as6453.net [216.6.84.53]
14   853 ms     *        *     if-1-3.tcore2.NJY-Newark.as6453.net [64.86.252.58]
15     *      873 ms   852 ms  if-2-2.tcore1.NJY-Newark.as6453.net [66.198.70.1]
16   857 ms   891 ms   894 ms  if-4-2.tcore1.L78-London.as6453.net [80.231.130.33]
17   856 ms   849 ms   852 ms  if-5-0-0.mcore3.LDN-London.as6453.net [195.219.195.9]
18   892 ms   893 ms   893 ms  if-8-0-0.har1.W1T-Warsaw.as6453.net [195.219.195.114]
19   870 ms   874 ms   884 ms  ix-9-2-3002.har1.W1T-Warsaw.as6453.net [195.219.188.42]
20   886 ms   887 ms   889 ms  ae1-3989.r5.glo-r7.glo.atman.pl [212.91.9.73]
21   891 ms   886 ms   887 ms  xe-1-0-0-4085.r2.isp-r5.glo.atman.pl [217.17.33.54]
22   906 ms   891 ms   905 ms  host-217.149.245.170.parkiet.com [217.149.245.170]
```

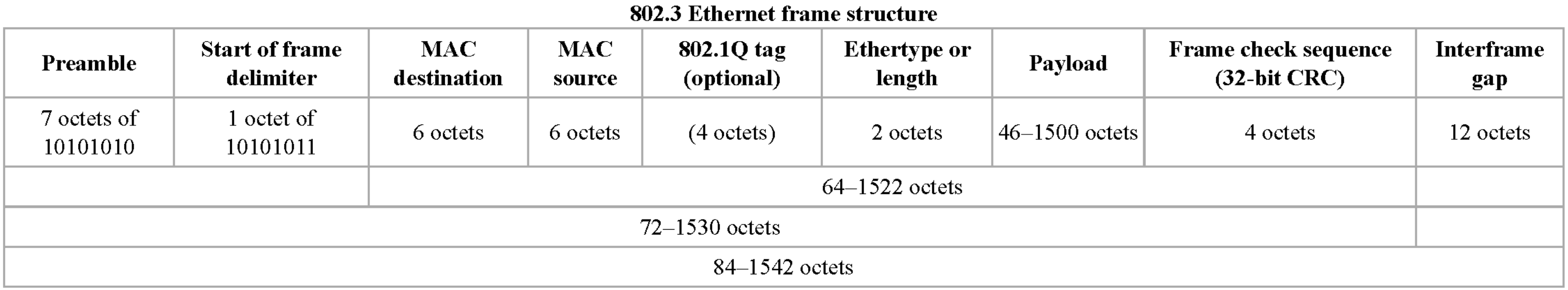
Ethernet

- **Ethernet** is a family of frame-based computer networking technologies for local area networks (LAN) standardized as IEEE 802.3.
- It defines a number of wiring and signalling standards for
  - the Physical Layer of the networking model
  - a common addressing format
  - a variety of Media Access Control procedures at the lower part of the Data Link Layer.

Most common Ethernet versions are:

- Ethernet over twisted pair to connect end systems,
- fibre optic versions for site backbones.

Ethernet II Frame



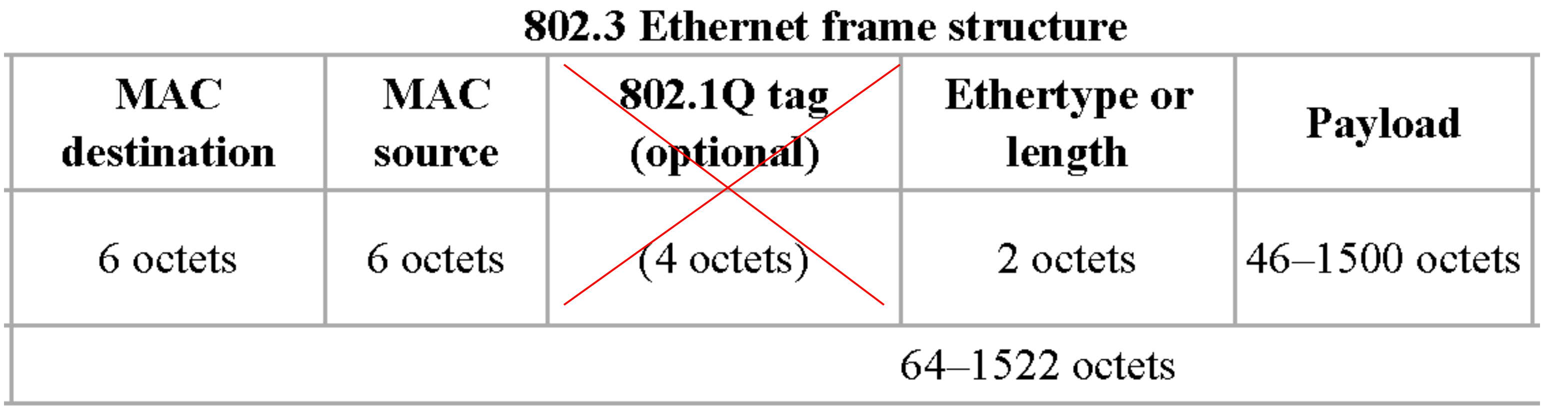
802.3 Ethernet frame structure

<table>
<tr><th>Preamble</th><th>Start of frame delimiter</th><th>MAC destination</th><th>MAC source</th><th>802.1Q tag (optional)</th><th>Ethertype or length</th><th>Payload</th><th>Frame check sequence (32-bit CRC)</th><th>Interframe gap</th></tr>
<tr><td>7 octets of 10101010</td><td>1 octet of 10101011</td><td>6 octets</td><td>6 octets</td><td>(4 octets)</td><td>2 octets</td><td>46–1500 octets</td><td>4 octets</td><td>12 octets</td></tr>
<tr><td colspan="2"></td><td colspan="6">64–1522 octets</td><td></td></tr>
<tr><td colspan="8">72–1530 octets</td><td></td></tr>
<tr><td colspan="9">84–1542 octets</td></tr>
</table>

- 7-byte preamble: repeating pattern of ones and zeros
- 1-byte start of frame delimiter (SFD): 10101011
- 6-byte destination and source MAC (physical) addresses
- Optional Virtual LAN tag. If used, first two bytes are 0x8100
- 2-byte type of frame field (EtherType) e.g.:
  - 0x0800 the frame contains IPv4 packet.
  - 0x0806 indicates an ARP frame
- Variable length data field: 46 to 1500 bytes.
- Note that 1536_{10} = 0x0600. If EtherType is < 0x0600, it indicates other Ethernet frames, e.g. 802.3ac
- 4-byte CRC-32 frame check sequence (FCS)

Ethernet II Frame with Wireshark

- In our practical experiment we will be able to capture a “non-hardware” part of the Ethernet frame:



802.3 Ethernet frame structure

MAC destination	MAC source	~~802.1Q tag (optional)~~	Ethertype or length	Payload
6 octets	6 octets	~~(4 octets)~~	2 octets	46–1500 octets
64–1522 octets				

- The 802.1Q tag will not be present.
- (octet is another word for byte)

This week tutorial session:

- Review of tutorial 1 on MatLab for wireless network signals
- Use of Wireshark for monitoring network traffic

Next lecture:

Wireless fundamentals