



Lecture 7: HPC Networks





ARPANET

- On October 4, 1957, the Soviet Union (USSR) launched the first satellite ever, shocking the United States. This was a major victory for the Soviets in the cold war.
- In response the US founded ARPA (Advanced Research Projects Agency) in February of the following year.
- ARPA's mission was transformational technologies (like the first space satellite).
- Periodically US Presidents add or subtract Defense from the front (DARPA).

ARPANET

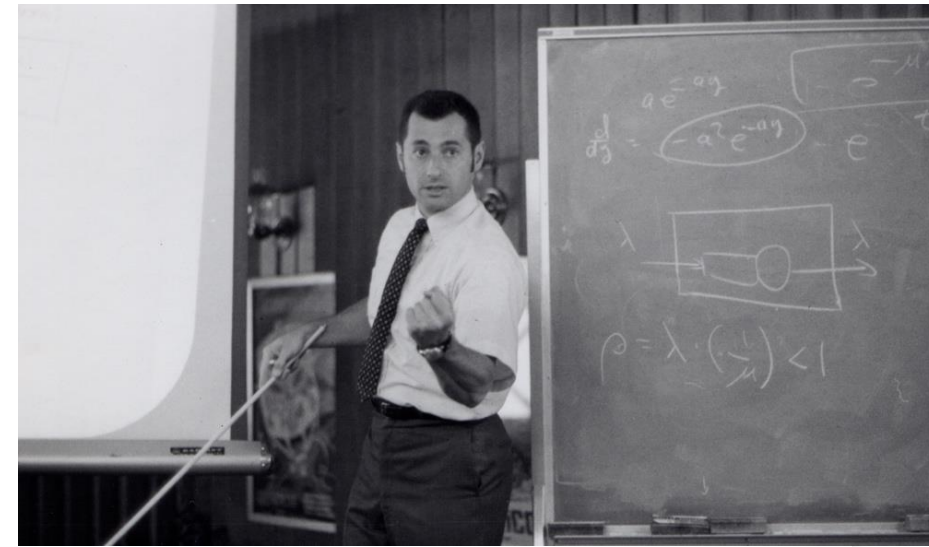
- In the early 60s Licklider's idea was that everyone could benefit from a computer assistant.

His consisted of:

1. Computer Science Departments
2. Graphics
3. Artificial Intelligence
4. Timesharing computers
5. Networking



Bob Taylor (Psychology and CS)



Leonard Kleinrock (UCLA Prof CS)



Joseph Licklider (Psychology and CS)



Larry Roberts

ARPANET

- After Sputnik a major concern for the US Military was that a circuit-switching telephone network could be interrupted by destroying the exchanges.
- Licklider's solution was what he called the Intergalactic Computer Network (ICGN) (in a 1963 memo) coupled with the packet switching network described in Leonard Kleinrock's PhD dissertation.
- The idea was that computers could communicate by breaking down messages into fragments.
- Each fragment would find its own path to the destination independently.
- This would be harder to interrupt.
- Circuit switching dedicates a hardware line between sender and receiver.



Joseph Licklider (Psychology and CS)

ARPANET

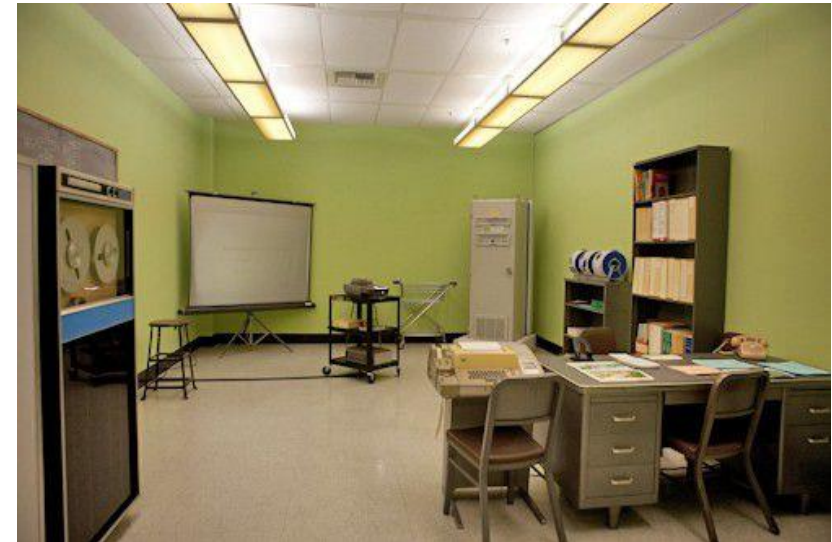
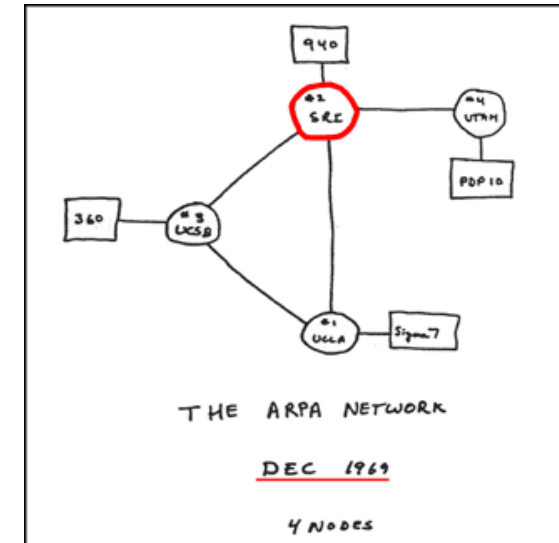
- 1969; "As of now, computer networks are still in their infancy. But as they grow up and become more sophisticated, we will probably see the spread of 'computer utilities,' which, like present electric and telephone utilities, will service individual homes and offices across the country."
- 1999: "What I did not conceive of then was that my 92-year-old mother would be using the Internet today."

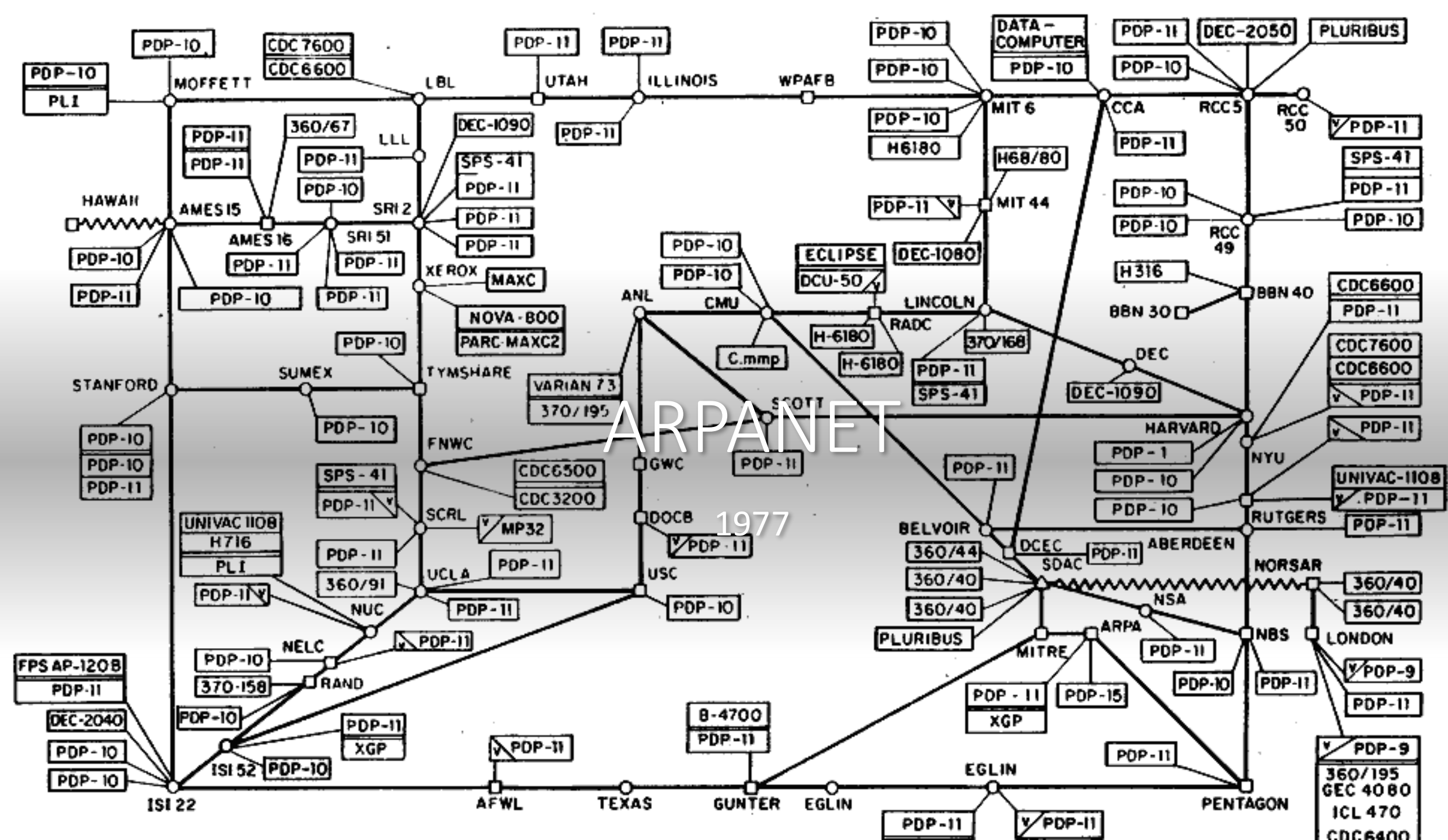


Leonard Kleinrock (UCLA Prof CS)

ARPANET

- ARPANet connected 4 computers in 1969
- University of California Los Angeles (UCLA)
 - Network Monitoring.
- Stanford Research Institute (SRI)
 - Doug Engelbart's Human Intellect Augmentation System.
- University of California Santa Barbara
 - Interactive graphics.
- University of Utah
 - Advanced 3D graphics.





Network Control Protocol (NCP)

- First Packet Switching network.
- Previously communication was don't through circuit switching.
 - Problems with circuit-switching. 1) Most of the time the channel is not used, but it is tied up. For example, waiting for the user to type the next letter in their command. Literally a mechanical circuit is made at a telephone exchange. Think modems where one computer literally called the phone number of another computer. Or if the computers were near each other, you would connect them with a serial cable (not a USB because they were not a bus).
- In packet switching a computer message is broken down into small pieces (packets). All the hardware is connected together all the time. The packets each take an independent path between sender and receiver.
- More efficient and more resistant to hardware failure.
- First packet-switching network use: Login from University of California Los Angeles Room 3420 to a computer at the Stanford Research Institute (320 miles).
- This was the first use of the term “login” and the first message was “lo” (the system crashed).

Network Control Protocol (NCP)

In a circuit-switching network where there are 5 people at one location and 5 at another if they all want to talk to one other person then there have to be 5 separate circuits. If they all want to talk at the same time there would have to be 25 dedicated hardware connections.

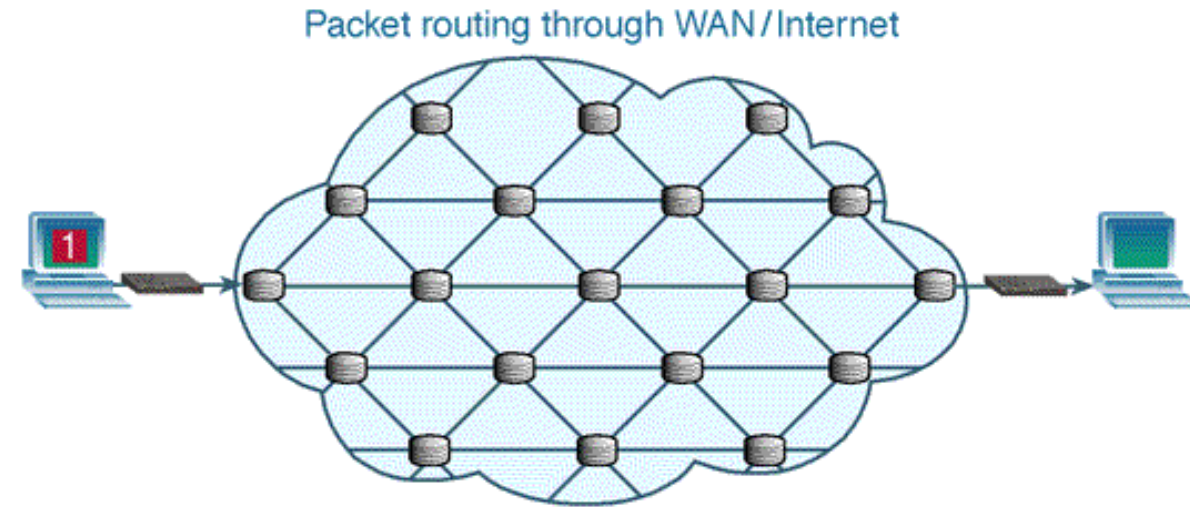
With a packet switching network there only needs to be one hardware circuit with all the computers listening to it at the same time.

Or if you want resilience then have multiple hardware connections between sites that take different physical routes. Then if one route is knocked out, or becomes congested, the packets can take the remaining, or fastest route.

A packet switching network works much like the interstate system where packets are cars.

Each computer only pays attention to packets addressed to it.

ARPANET replaced NCP with the Internet Protocol (IP)



Local Area Networks

- Everything above is considered a WAN (Wide Area Network).
- Parallel technological developments were happening with computers that were near each other.
- This might seem strange but communication mostly made sense for long distance connections.
- For computers in the same building or campus it was silly to send an email or remotely login (why not walk?)

Local Area Networks (LAN)

- But as computing devices multiplied connecting them locally made sense.
- These devices could be anything from a supercollider at a university producing millions of atomic particle products, a printer, or a VT100 terminal.
- Two approaches were developed to meet the need for an easy way to connect devices near each other.
- Token ring networks used coax cable and a round-robin approach. Each device on the LAN has a token saying it's allowed to talk. (Think of a talking stick). The token is passed from device to device around the ring of connections. That way every device gets a chance to speak and no devices try to talk at the same time.
- Ethernet is based on carrier-sensing
- CSMA/CD (Carrier-Sense Multiple Access with Collision Detection)

Local Area Networks (LAN)

- CSMA/CD (Carrier-Sense Multiple Access with Collision Detection)
- As opposed to token ring systems, CSMA devices listen to the hardware line. When they don't hear anything they can try to talk.

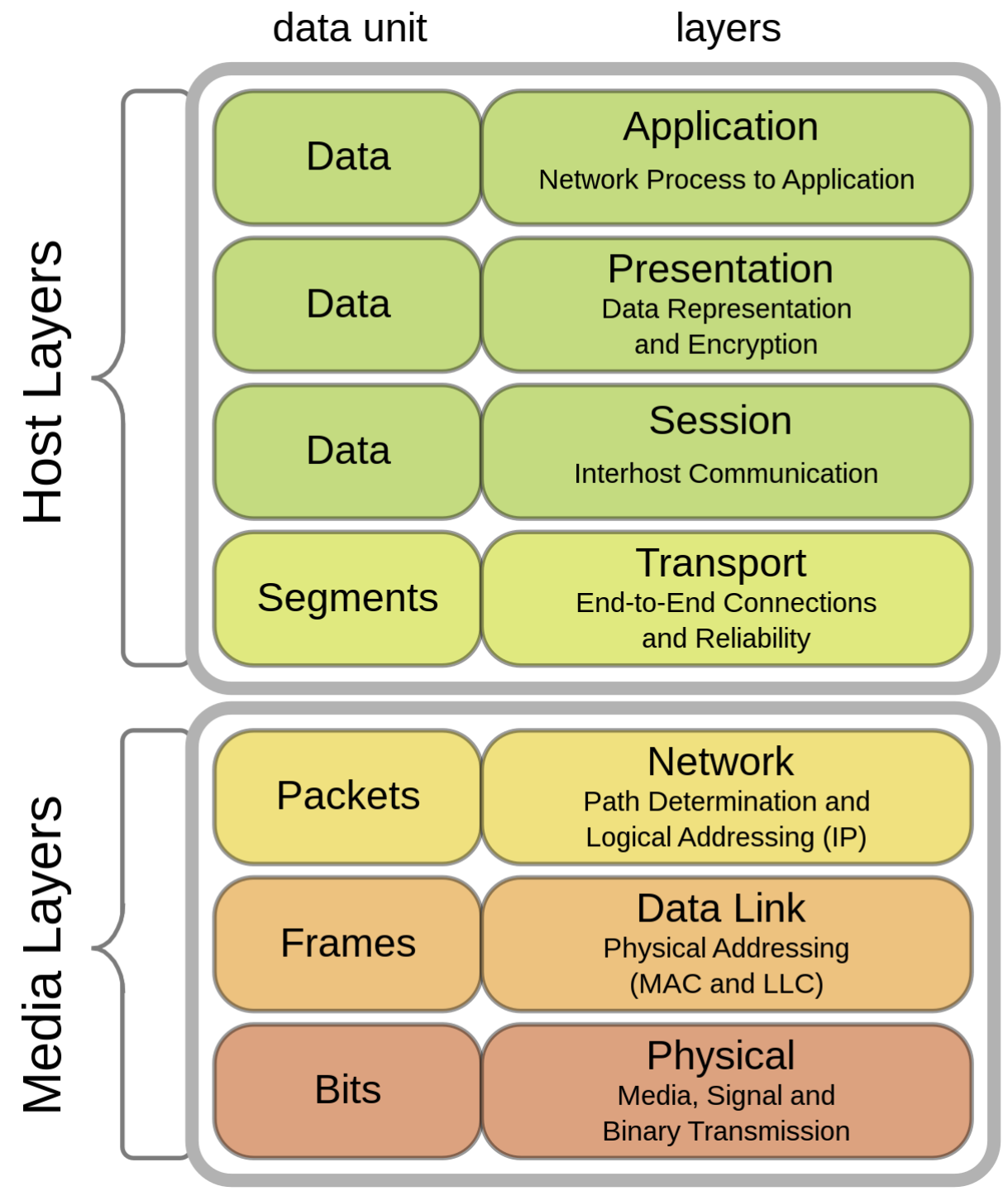
“Listen Before Talk” principle

- Sometimes two or more devices try to talk at the same time. This means there are electrical signals encoding different messages sent at the same time on the same wire, resulting in the messages being corrupted.
- The first device to detect the message collision broadcasts a “jam” message indicating that no device should transmit. All devices wait a random period of time and try again.
- Messages are divided into **frames** and those frames are addressed to MAC addresses.
- Token ring networks tried to be high quality and assure quality transmissions.
- Ethernet allowed network corruption and had a randomized recovery scheme.
- Ethernet won because it was cheaper.

Network Stack

Open Systems Interconnect (OSI) Model

- The lowest level is the actual hardware.
- At the top is the layer that talks to programs that wants to communicate (a web browser or email client for example)



Network Stack

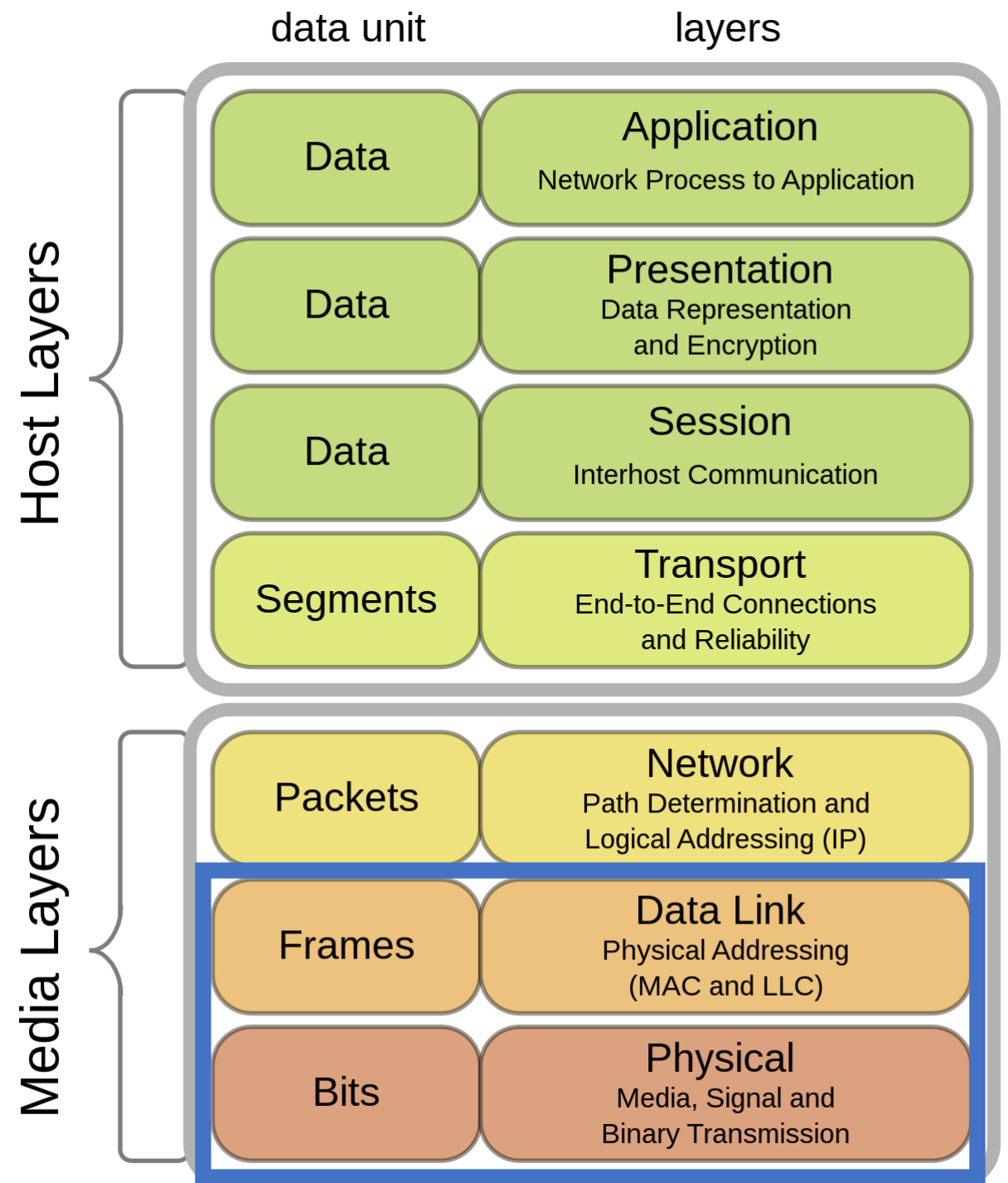
Open Systems Interconnect (OSI) Model

Ethernet

Operates at the Media Access Control sublayer between the physical and data link layers.

Each device has an ethernet address or MAC (media access control).

Format: XX:XX:XX:XX:XX:XX



Network Stack

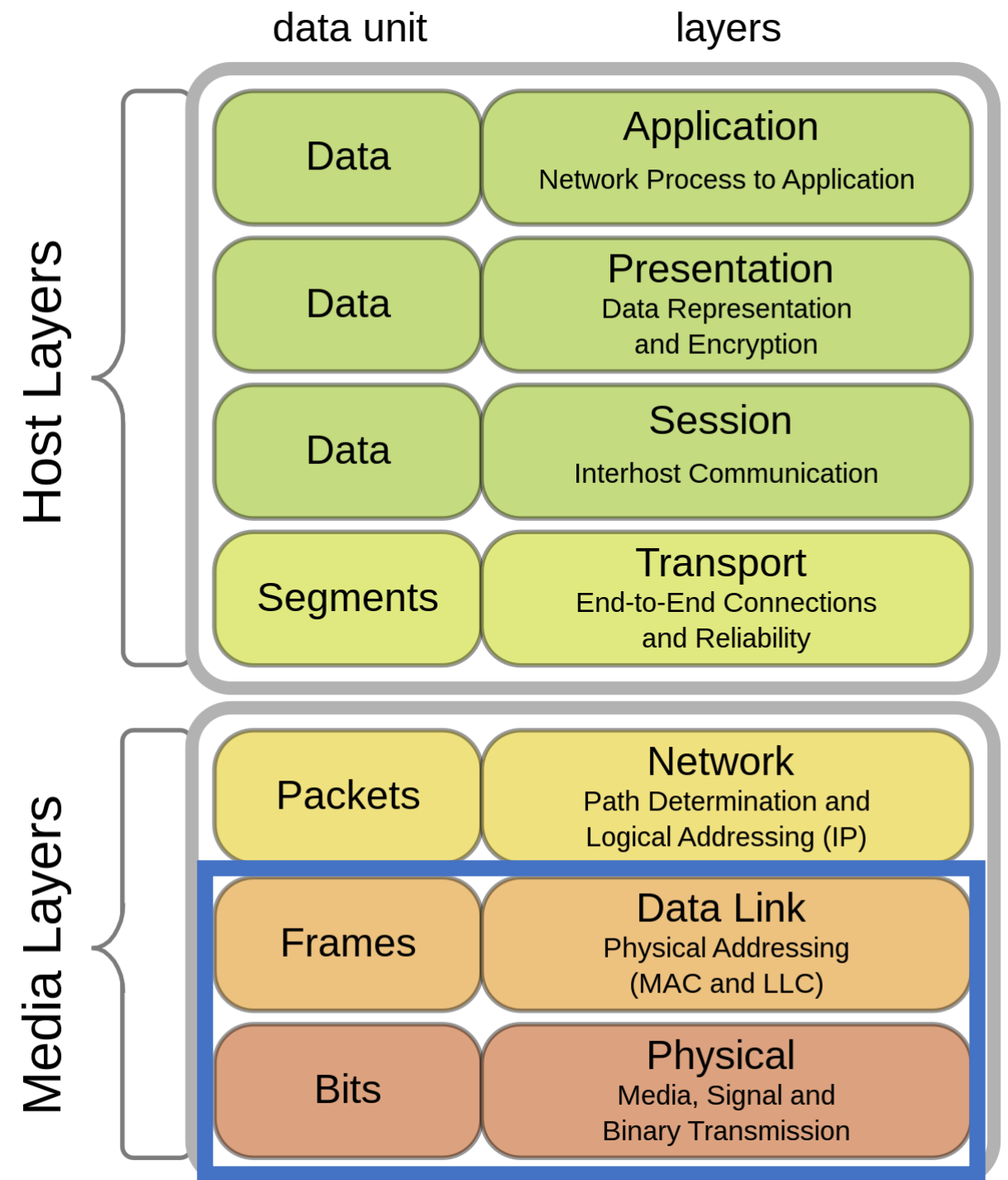
Open Systems Interconnect (OSI) Model

Ethernet

Operates at the Media Access Control (IEEE 802.2) and link (802.3) sublayers.

MAC addresses are physical addresses set by the hardware manufacturer.

Eg: d4:ae:52:8b:72:8c



[matthew@moonshine ~]\$ ip a

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether d4:ae:52:8b:72:8c brd ff:ff:ff:ff:ff:ff
    altname enp1s0f0
    inet 129.24.245.16/22 brd 129.24.247.255 scope global noprefixroute eno1
        valid_lft forever preferred_lft forever
    inet6 fe80::d6ae:52ff:fe8b:728c/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

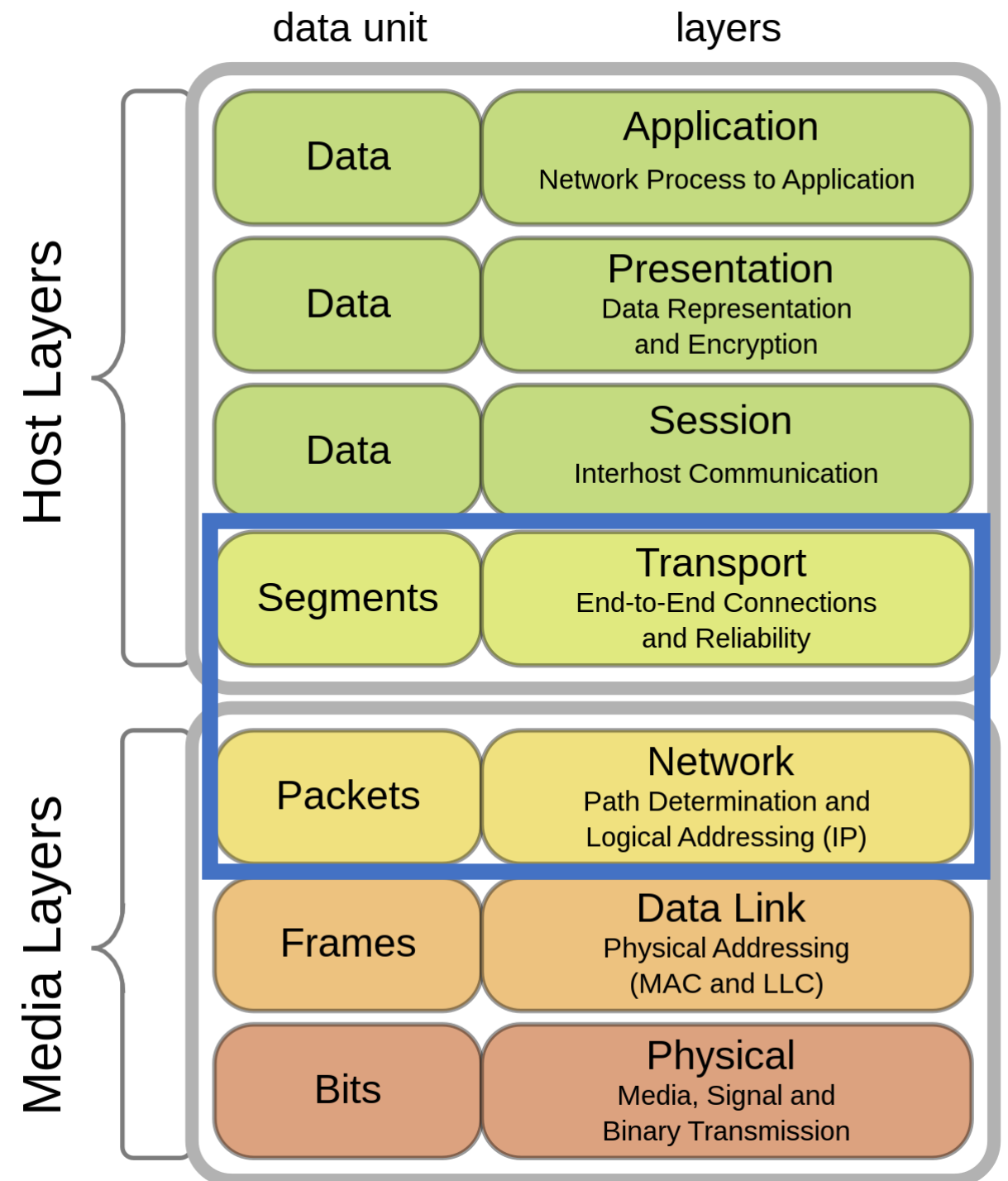
Network Stack

Open Systems Interconnect (OSI) Model

Actual messages are handled at the transport and network layers with packets.

Examples are TCP(or UDP)/IP and the now defunct SPX/IPX.

These use logical addresses (such as IP addresses) not hardware addresses.



[matthew@moonshine ~]\$ ip a

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether d4:ae:52:8b:72:8c brd ff:ff:ff:ff:ff:ff
    altname enp1s0f0
    inet 129.24.245.16/22 brd 129.24.247.255 scope global noprefixroute eno1
        valid_lft forever preferred_lft forever
    inet6 fe80::d6ae:52ff:fe8b:728c/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

Example – Reading google.com

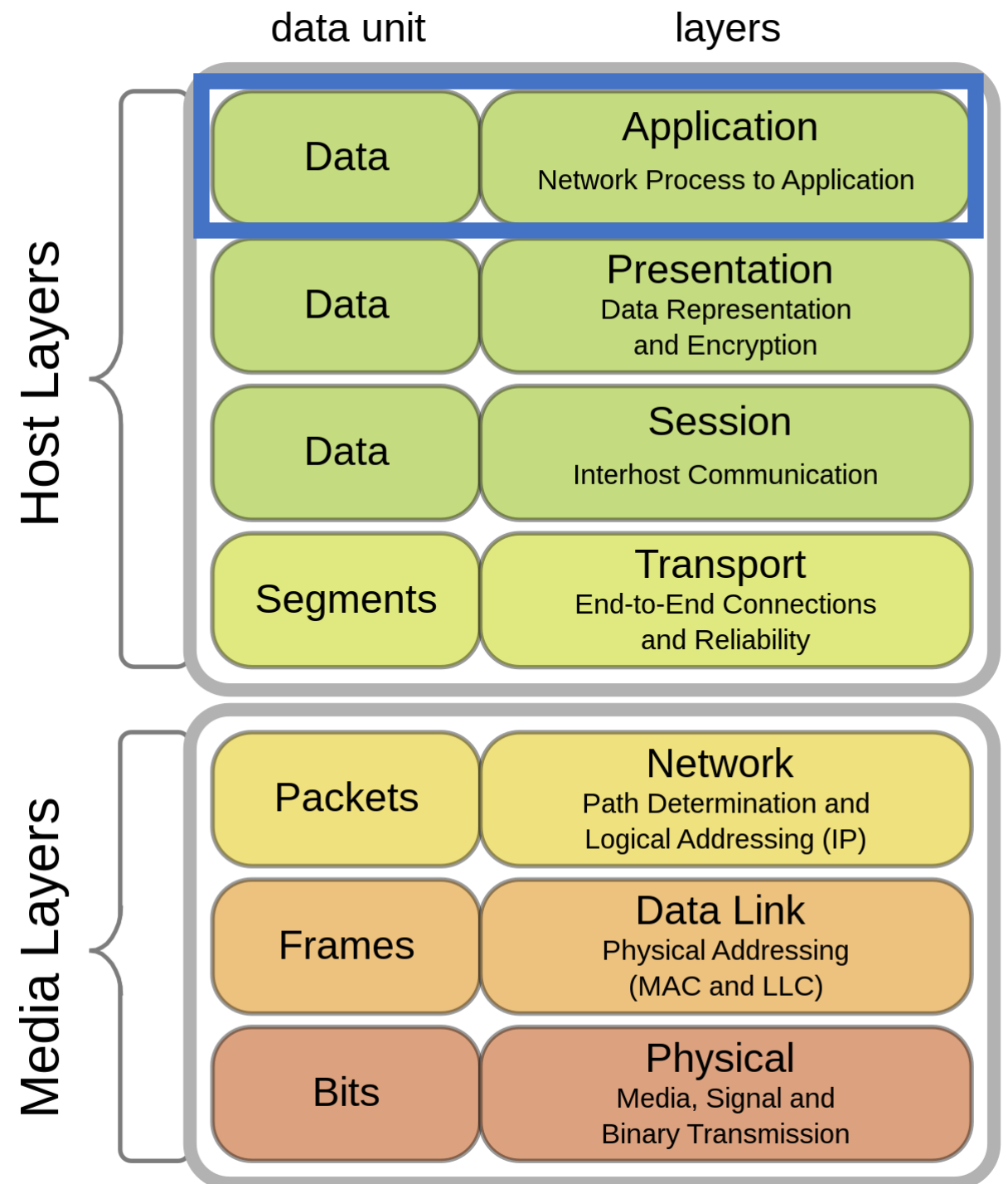
The web browser application asks a DNS (Domain Name Service) server to return the IP address of “google.com”.

DNS is just a big lookup table of all the names people have associated with domain names. Could also be a local hosts file entry.

The DNS server’s IP address is set when you configure your network interface.

This all happens in layer 7.

To be clear the web browser is not in the application layer – but it talks to the application layer of the network stack.





Register

Sign In

Home News Sport Business Innovation Culture Travel **Earth** Video Live

Earth



Climate change will bring megafloods to California

The state is beginning to experience "hydroclimate whiplash," where the climate veers wildly between extreme dryness and extreme wetness.



Stunning shot of polar bear drifting to sleep wins award



The paradox of the 'gentle protest'



Example – Reading <http://fricke.co.uk> web page.

The web browser application asks a DNS (Domain Name Service) server to return the IP address of “google.com”.

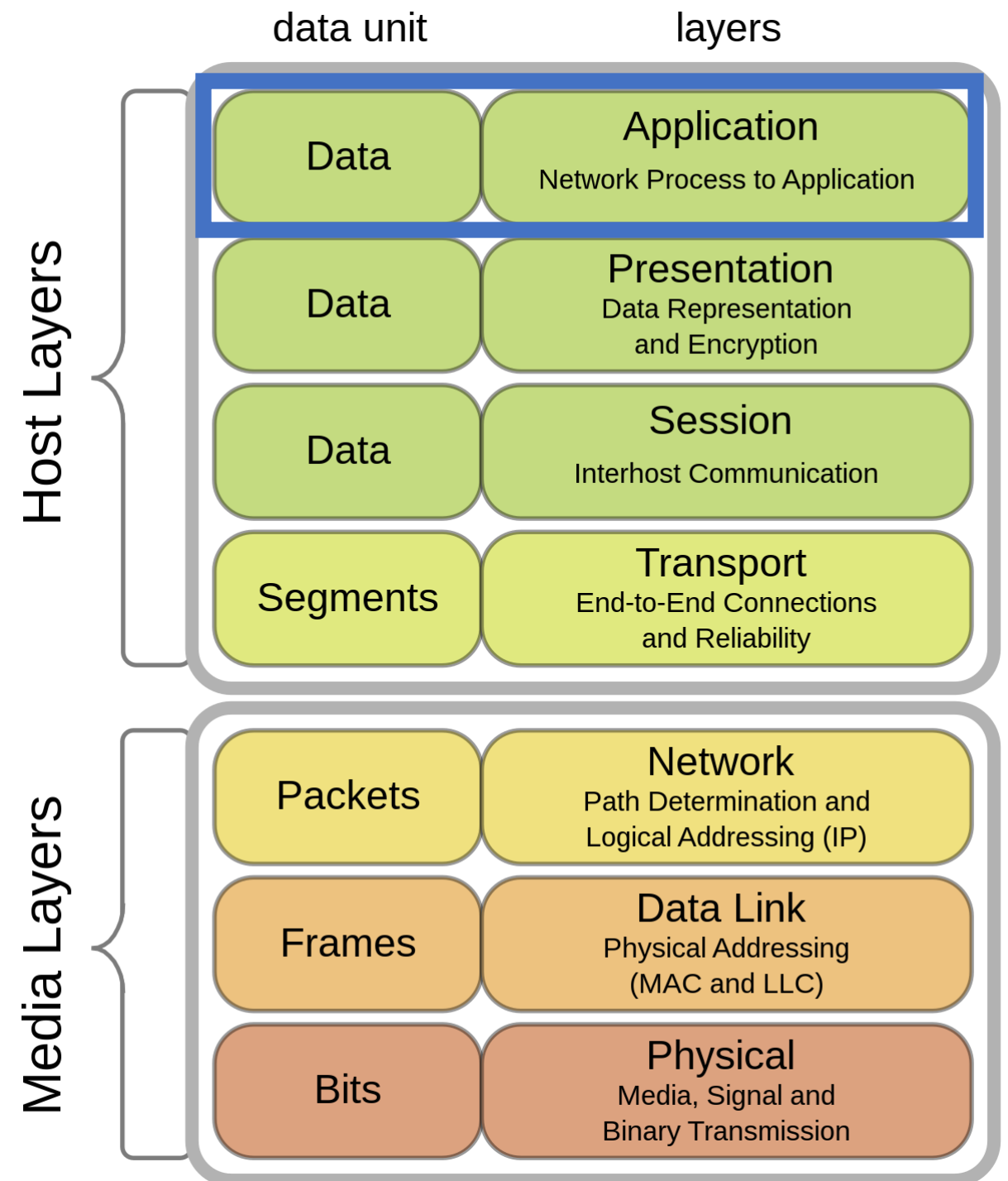
DNS is just a big lookup table of all the names people have associated with domain names. Could also be a local hosts file entry.

The DNS server’s IP address is set when you configure your network interface.

This all happens in layer 7.

Log into your cluster and run
`dig google.com`

To see the result of a DNS query.

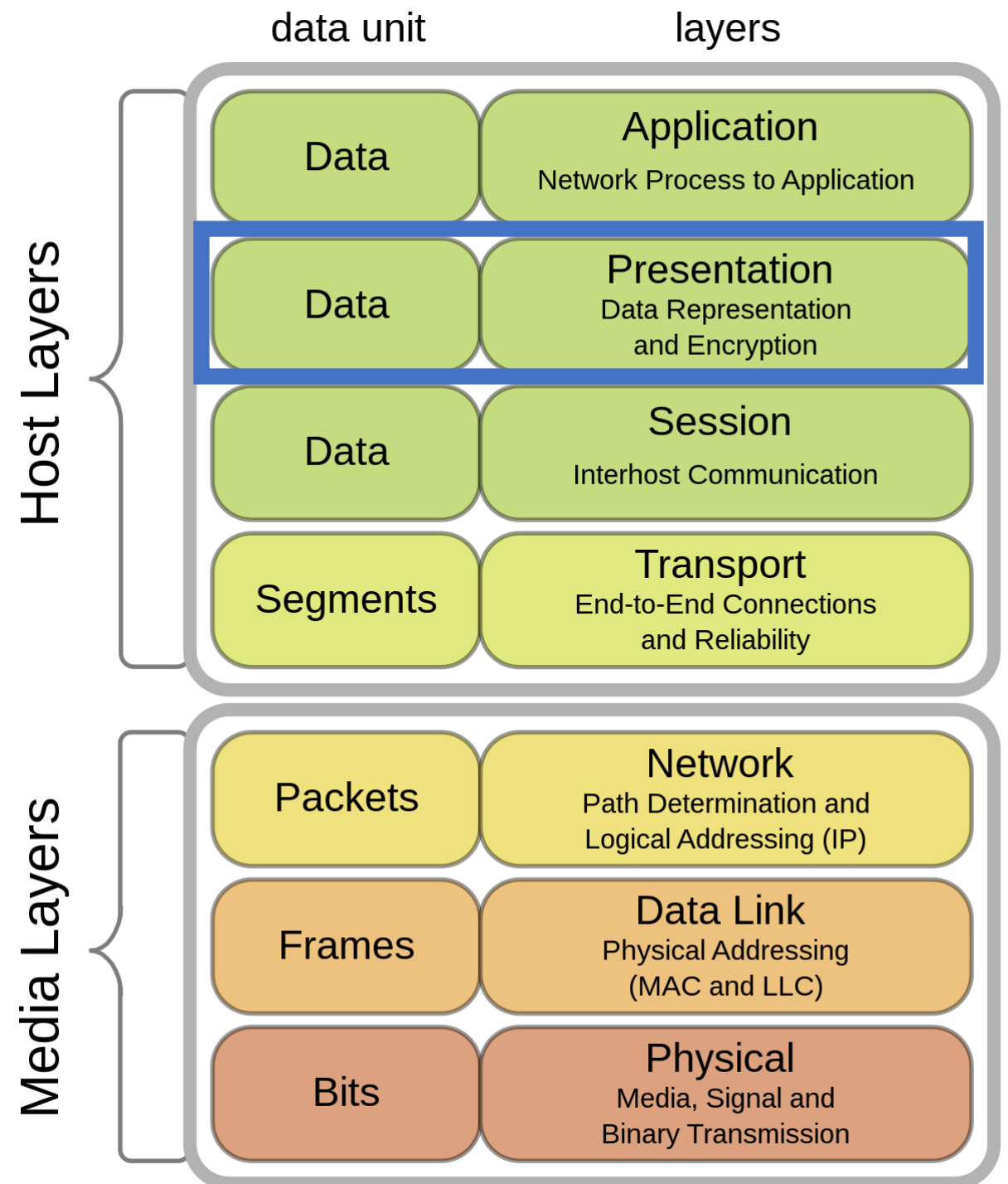


Example – Reading
<http://fricke.co.uk> web page.

The presentation layer is responsible for making sure the data encoding matches.

If it were https instead of http above then the presentation layer (6) would do the encryption/decryption.

If the data were compressed for transmission then compression and decompression would also happen in layer 6.



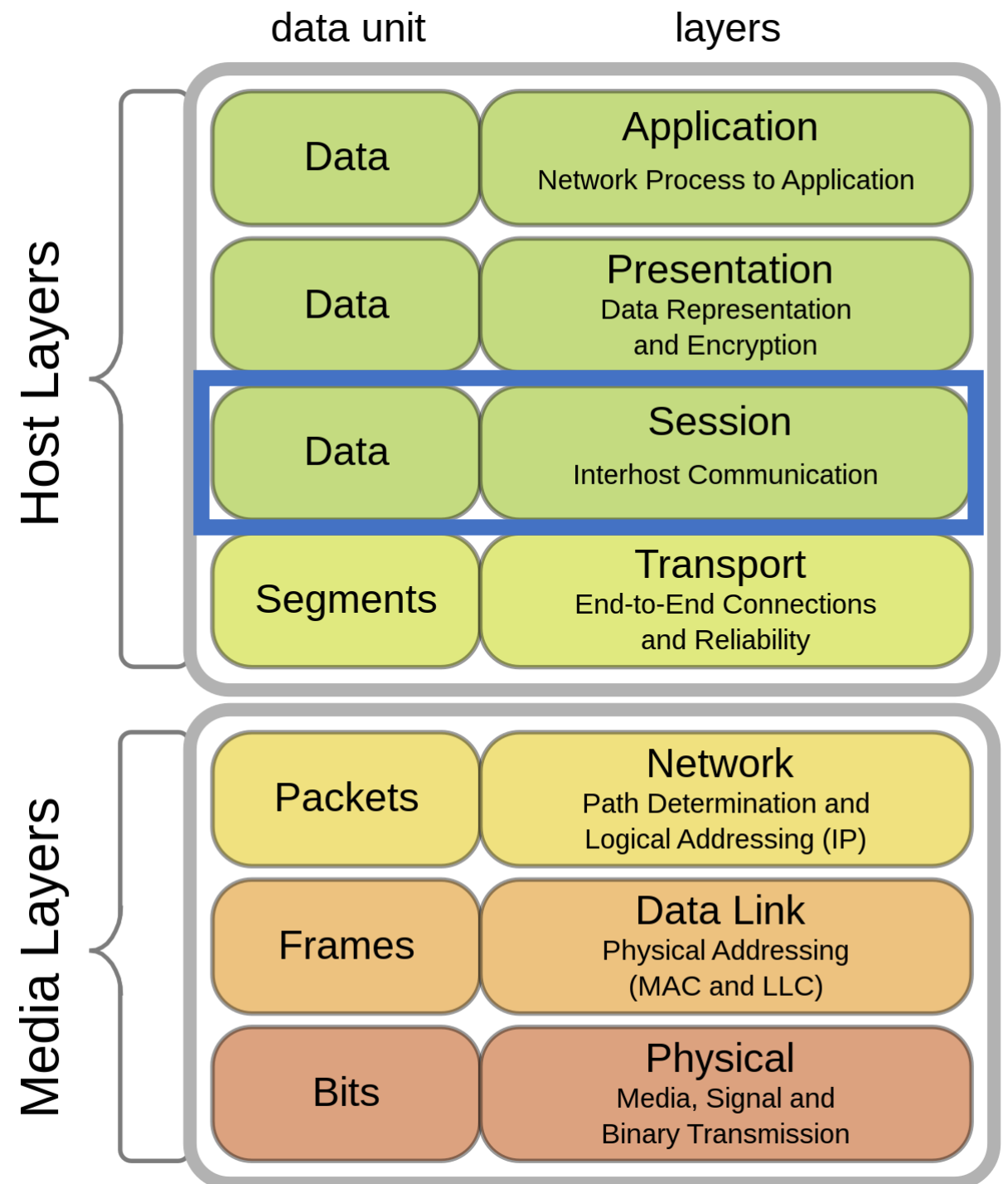
Example – Reading
<http://fricke.co.uk> web page.

The hypertext transfer protocol (HTTP) uses TCP (transmission control protocol). TCP connections include error checking to make sure the data that was sent arrived OK.

That means every time a message is sent to a web server the client waits for the web server to send a confirmation message. This kind of connection is managed in the session layer.

The user datagram protocol (UDP) is the alternative. It sends data without caring if it arrived. For example, a video stream where a few missing pixels wouldn't matter might use UDP.

(The OSI model is an idealization. In reality, the top three OSI layers are combined into one TCP/UDP application layer.)



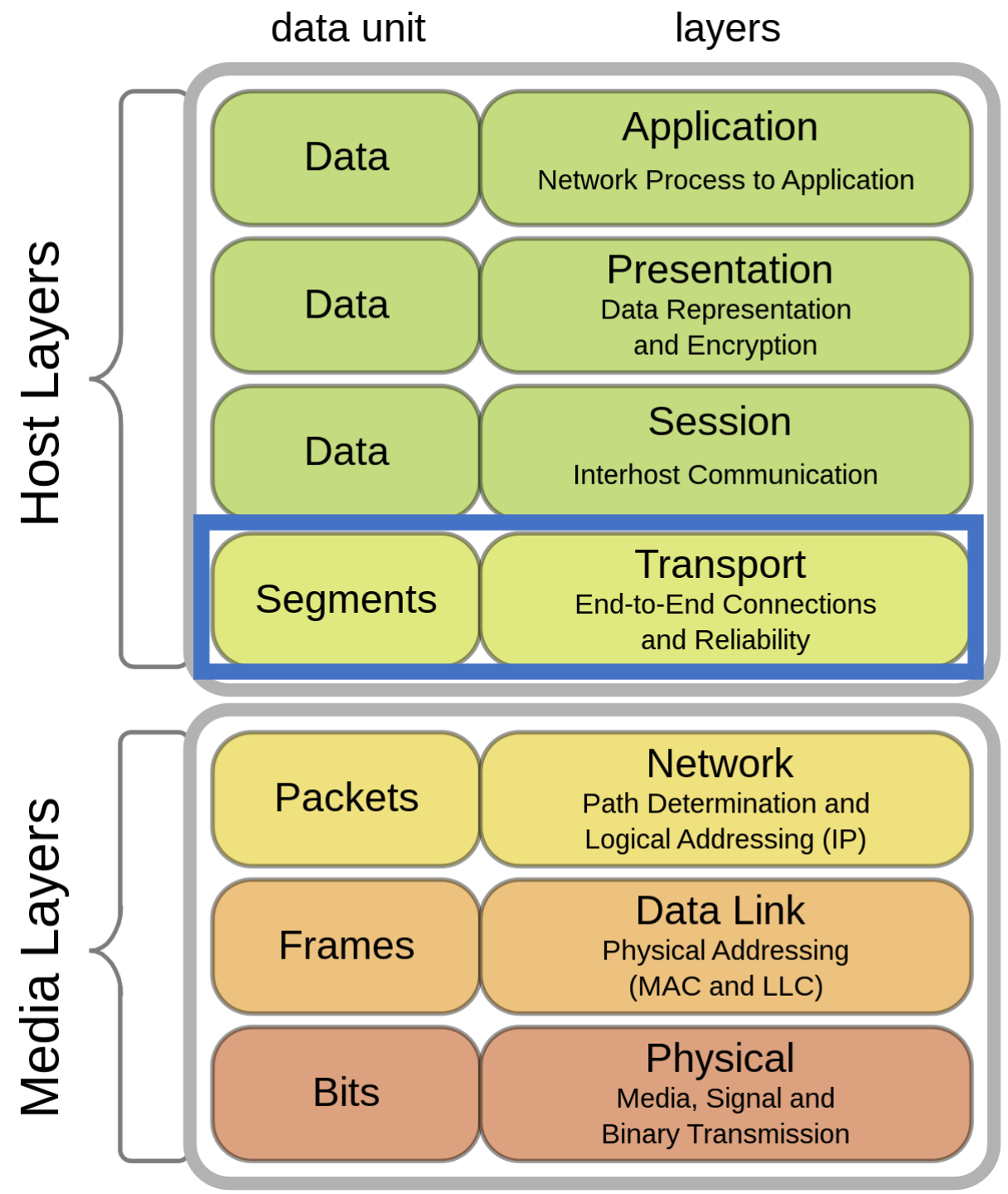
Example – Reading
<http://fricke.co.uk> web page.

The transport layer is responsible for taking the data written by the application, and encrypted/compressed, etc by the presentation layer and breaking it up into segments.

Segmentation is needed if the maximum transmission size of the network is greater than the amount of data being sent. (E.g. TCP max packet size is 64K).

Ethernet frames are only 1500 bytes. (though there are jumbo frames).

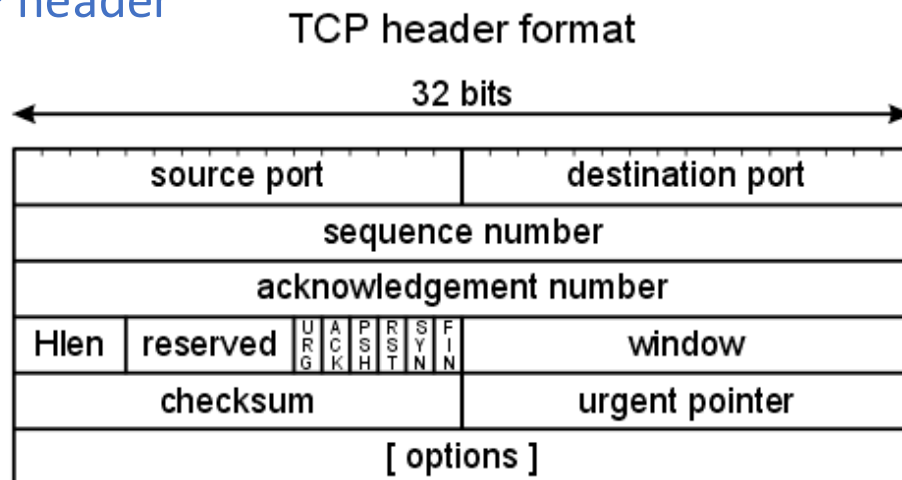
When receiving a message, the transport layer assembles segments into the right order for the application layers above. It also checks that all the expected segments arrived.



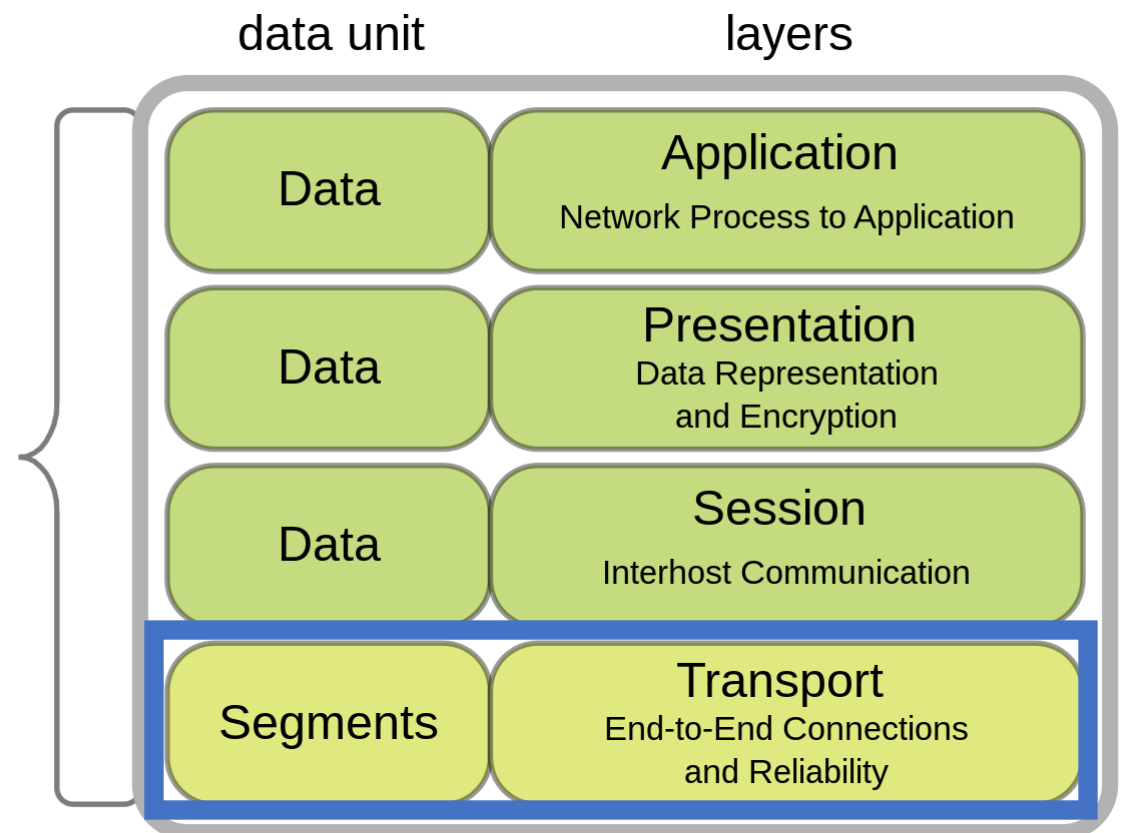
Example – Reading
<http://fricke.co.uk> web page.

The transport layer is responsible for taking the data written by the application, and encrypted/compressed, etc by the presentation layer and breaking it up into segments.

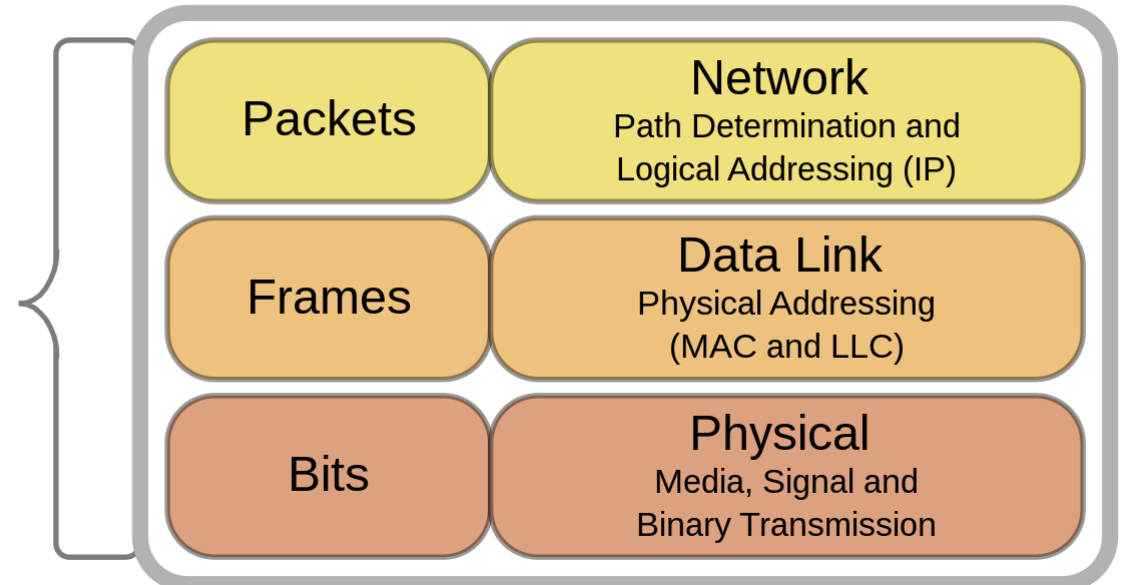
The segments have some headers added. For example, the destination port (http servers listed on port 80) and the sequence number of the segment – since they may arrive out of order. Since this is http the segment will have a TCP header



Host Layers



Media Layers



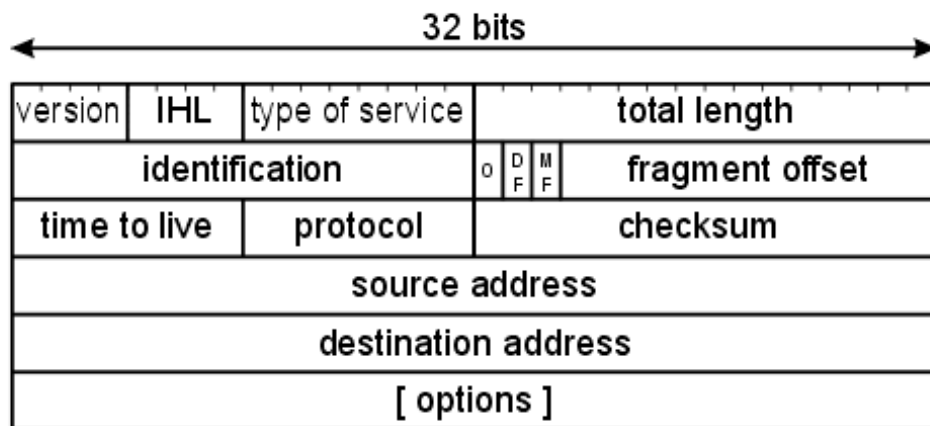
Example – Reading
<http://fricke.co.uk> web page.

Now that the data the application wants to send has been chopped up into segments, they are converted to packets.

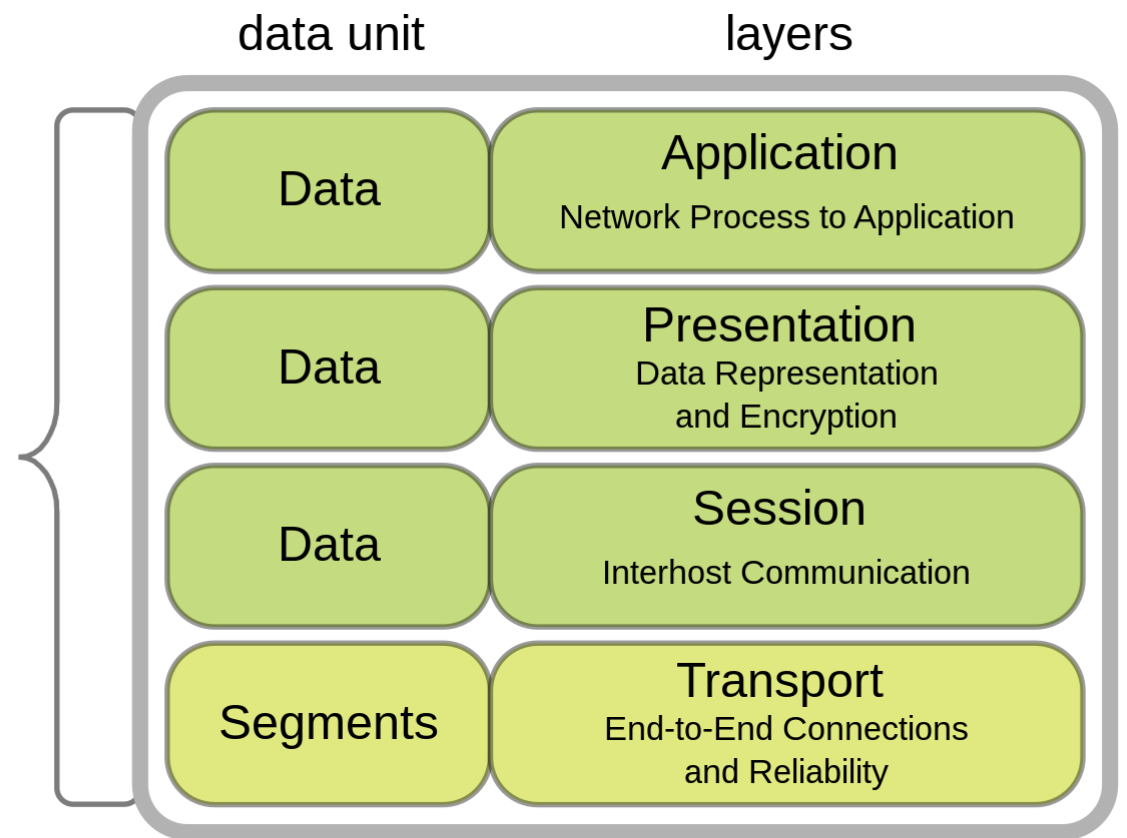
A packet is a segment of data, but with the destination logical address added.

In this case the logical address is the IP address given by the DNS lookup earlier.

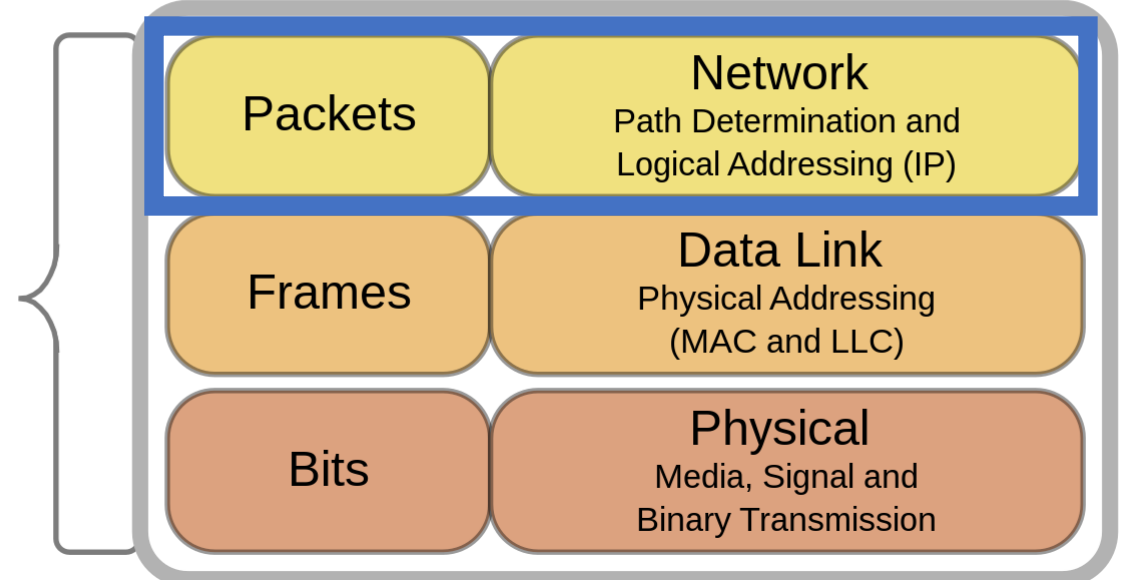
IP header format



Host Layers



Media Layers



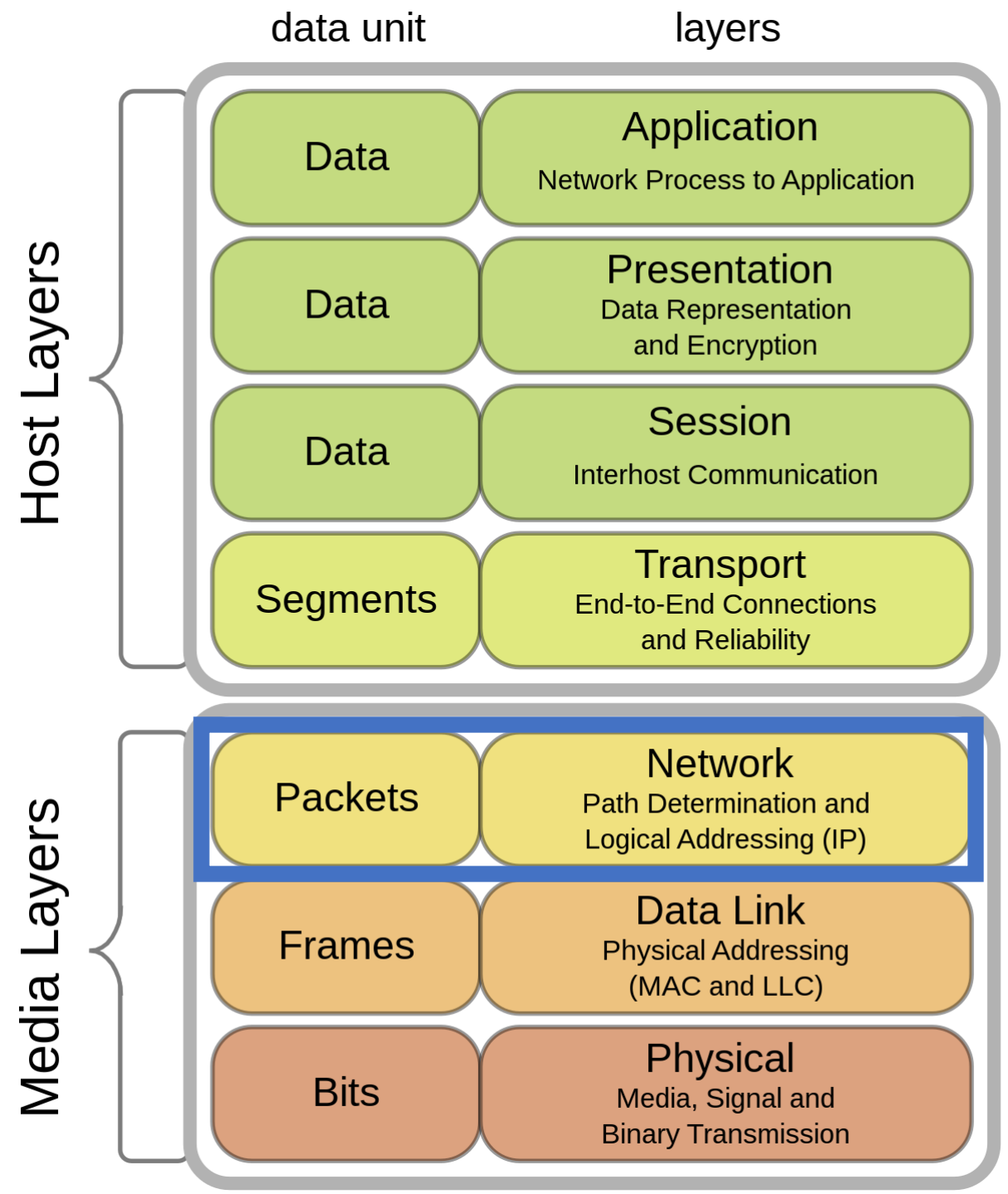
Example – Reading
<http://fricke.co.uk> web page.

Warning: the network layers are no longer being processed on the same computer. The “layer” can consist of various network devices working together.

If the destination IP address is on the same subnet as the sender, then the IP address can be translated to the destination devices MAC address for ethernet to handle immediately.

If the destination IP address is not on the same subnetwork, then the “gateway’s” MAC address is used and we begin routing.

IP addresses are translated to MAC addresses by the address resolution protocol (ARP in layer 2, data link).



ARP (Address Resolution Protocol)

- On your cluster enter this command:

```
cat /proc/net/arp
```

To view the address resolution table that maps MAC addresses to IP addresses.

The command

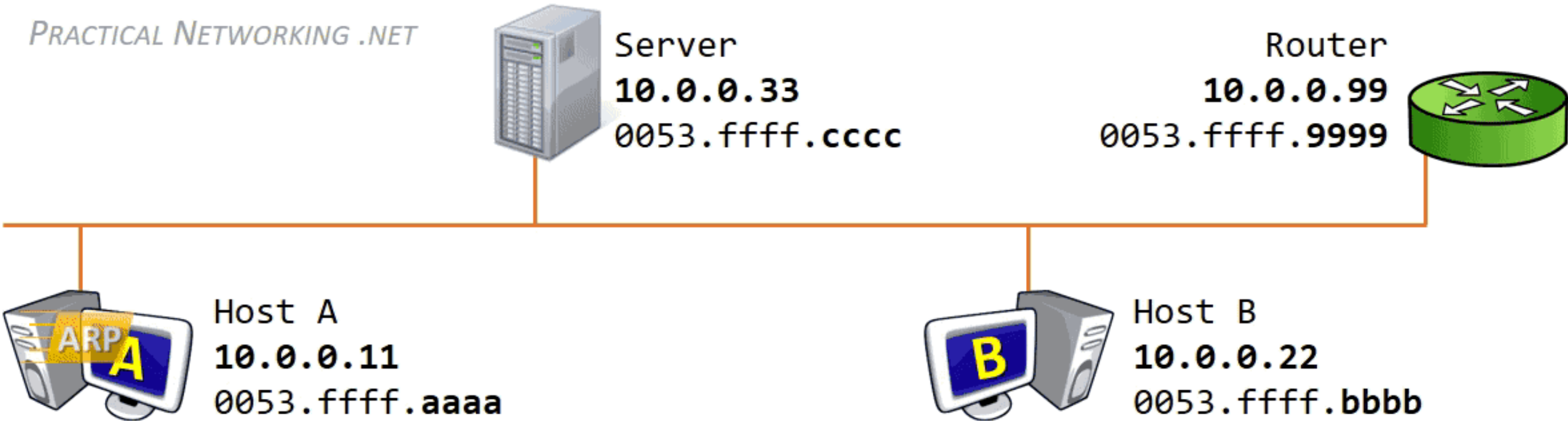
```
Arp
```

Gives you some display options and allows you to manipulate the table.

Network devices like layer 3 switches also maintain ARP tables. That is done so they only send data to the devices that have matching MAC and IP addresses. **This is not for security!** It is only for efficiency. ARP tables are often incomplete or out of date, then the switch just sends everyone the data hoping it gets to the right MAC address.

ARP (Address Resolution Protocol)

PRACTICAL NETWORKING .NET



The ARP table is not built in the same way as DNS tables.

DNS tables are usually entered manually – and you usually have to pay someone to route a domain name to your IP.

ARP just asks. That's OK because ARP only asks on the local network.

Example – Reading
<http://fricke.co.uk> web page.

Warning: the network layers are no longer being processed on the same computer. The “layer” can consist of various network devices working together.

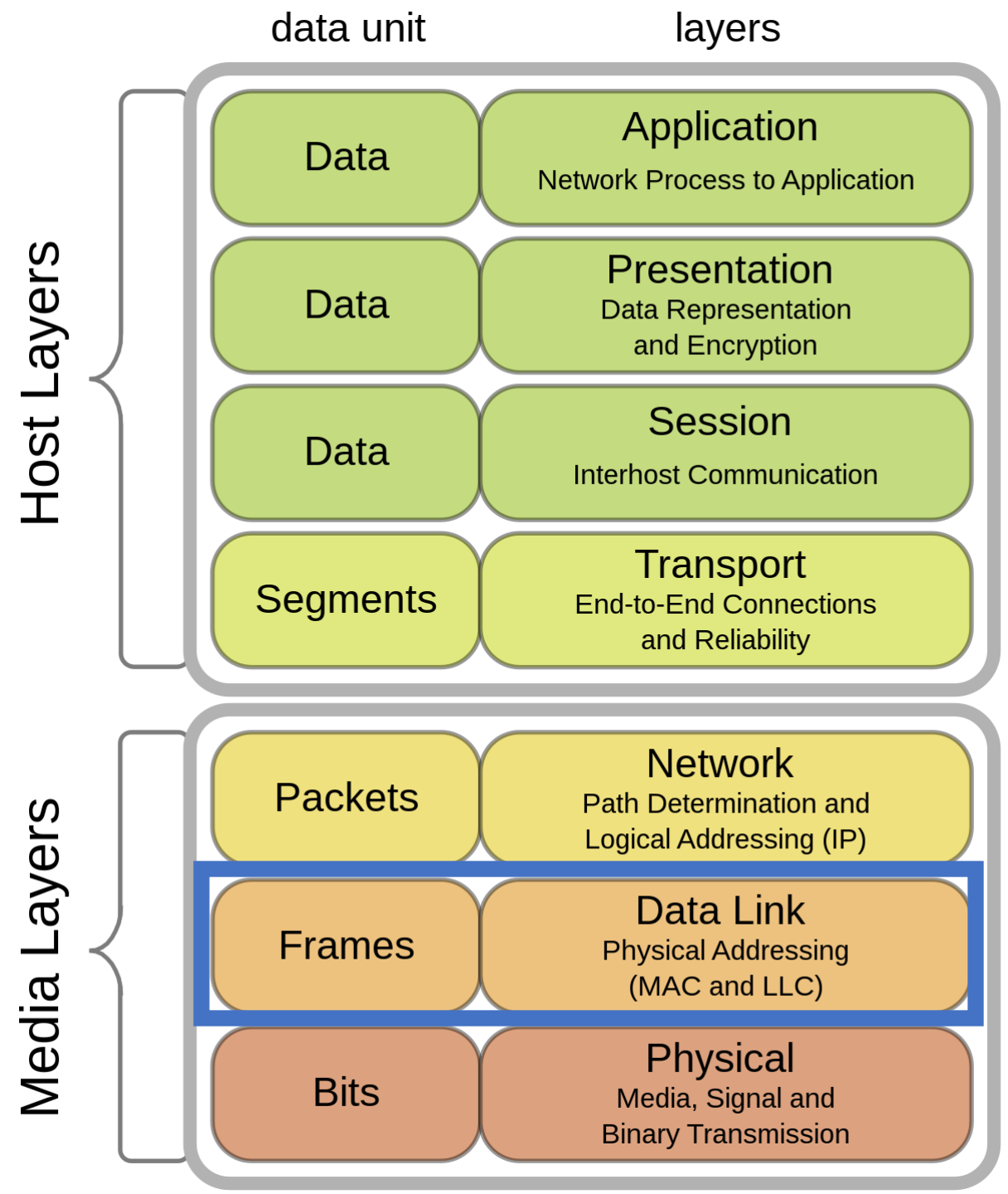
OK, so now we have a mapping from the IP (internet protocol) address to the MAC address, so we can add that MAC address to our packet header and move on to the datalink layer and the domain of the Ethernet protocol.

IEEE 802.3

7	1	6	6	2	46-1500	4
Preamble	SOF	Destin. address	Source address	length	802.2 PDU	FCS

SOF: Start of Frame

FCS: Frame Check Sequence



Why does any of this matter?

Example,

You will often hear HPC network engineers talking about layer 2 and layer 3 switches. Layer 2 switches do not know about IP addresses only MAC addresses.

There is a kind of attack called ARP poisoning*.

If I told you that “I suspect my layer 2 switch had been ARP poisoned”, would that make sense?

*ARP poisoning is when an attacker changes the ARP table so your IP address resolves to their computers MAC address so they can masquerade as your machine. It's also legitimately used when network administrators want to listen to your traffic to debug it. Sometimes it's used for seamless server rollover redundancy so a server can take traffic from a failed server.

Why does any of this matter?

Example,

Layer 2 switches don't have ARP tables
since they don't deal in IP to MAC translation.
Layer 3 switches do though.

You will often hear HPC network engineers talking about layer 2 and layer 3 switches. Layer 2 switches do not know about IP addresses only MAC addresses.

There is a kind of attack called ARP poisoning*.

If I told you that “I suspect my layer 2 switch had been ARP poisoned”, would that make sense?

*ARP poisoning is when an attacker changes the ARP table so your IP address resolves to their computers MAC address so they can masquerade as your machine. It's also legitimately used when network administrators want to listen to your traffic to debug it. Sometimes it's used for seamless server rollover redundancy so a server can take traffic from a failed server.

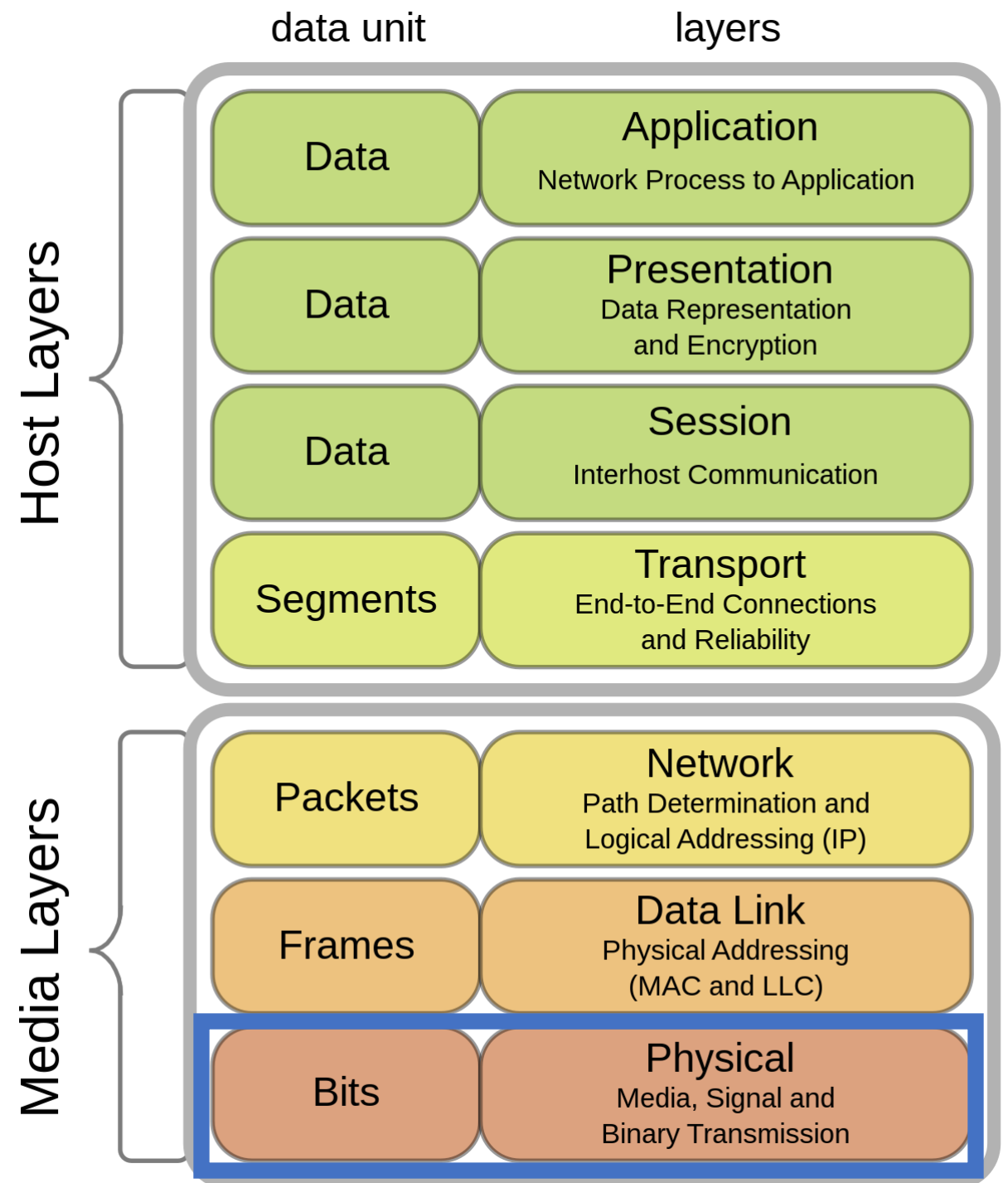
Example – Reading
<http://fricke.co.uk> web page.

Finally, we have constructed a complete packet (consisting of segments and frames) that we can send to the NIC (network interface card).

The NIC sends electrical or light signals onto the wire depending on whether we have a copper or fiber optic connection.

The NIC with a matching MAC address records those signals and we begin working up the stack to decode everything.

If the destination MAC address is a gateway then the packet is routed to the next destination on its way to the destination network.



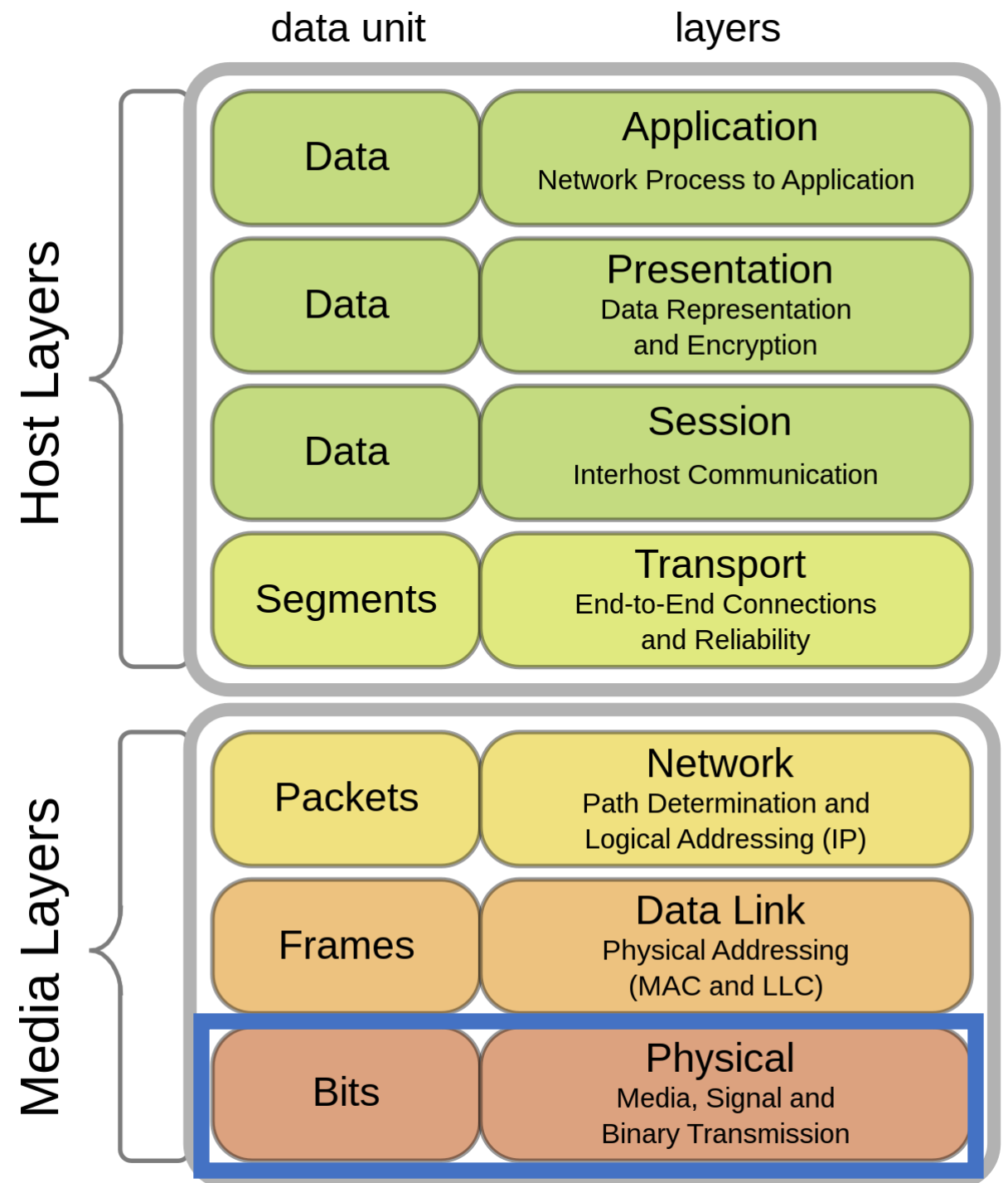
Example – Reading
<http://fricke.co.uk> web page.

There are many protocols that may be used to route the packet to it's final destination.

Each router rewrites the outermost header of your packet with the next destination.

When your packet moves from one major internet provider to another they will use BGP (Border Gateway Protocols*) to choose the next path.

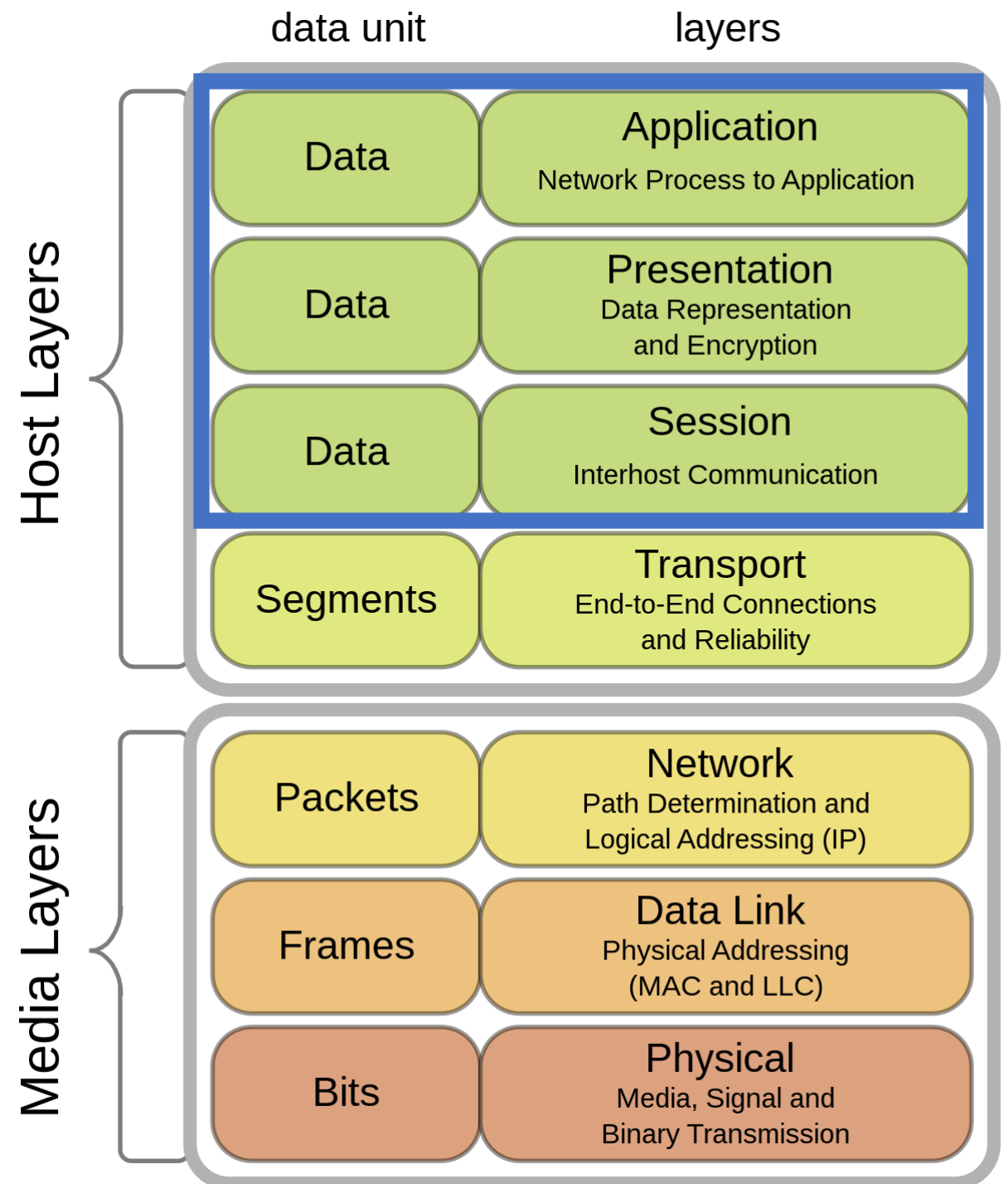
Also known as the “three-napkin protocol”



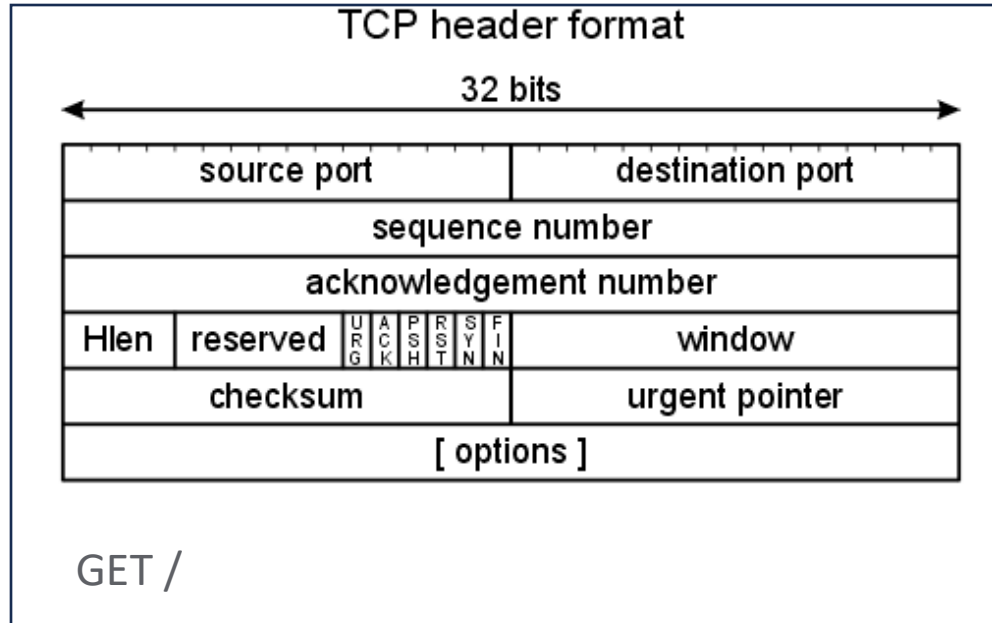
Example – Reading
<http://fricke.co.uk> web page.

GET /

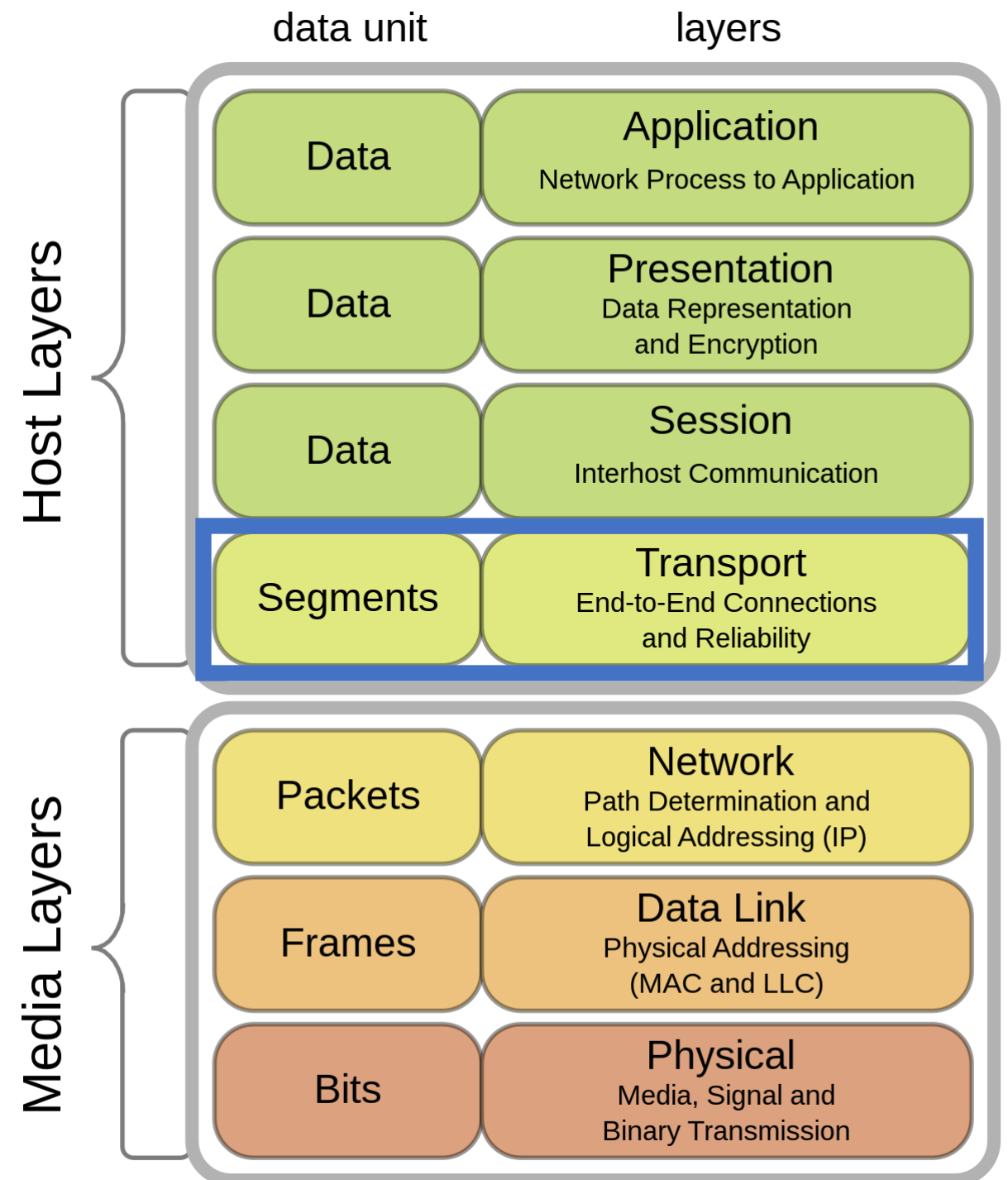
Let's go through the layers to construct a packet



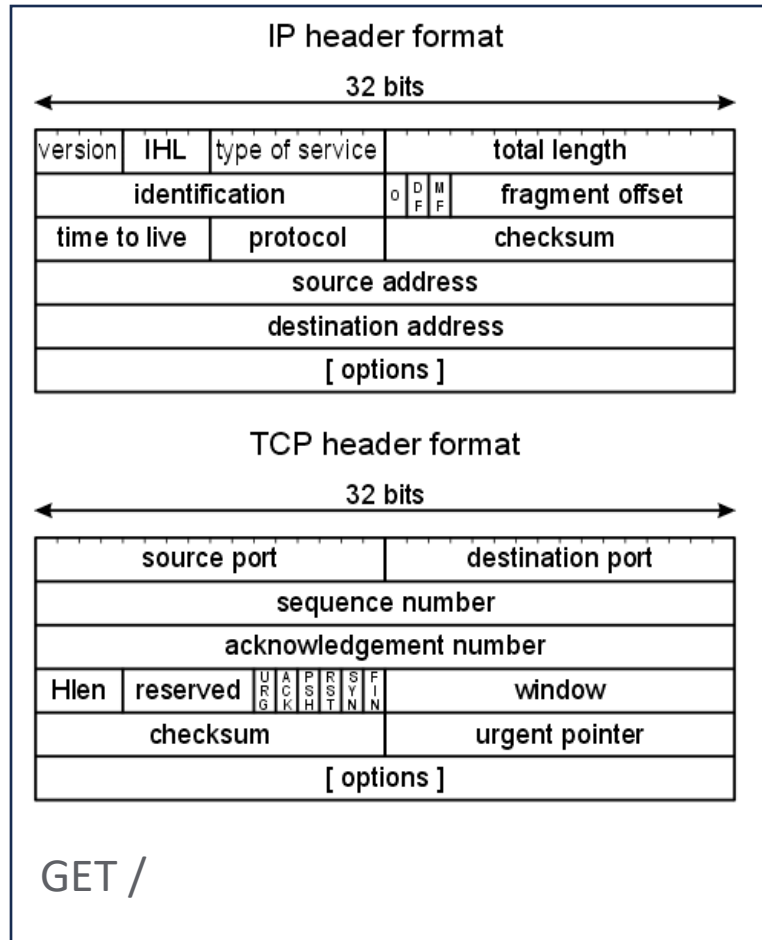
Example – Reading
<http://fricke.co.uk> web page.



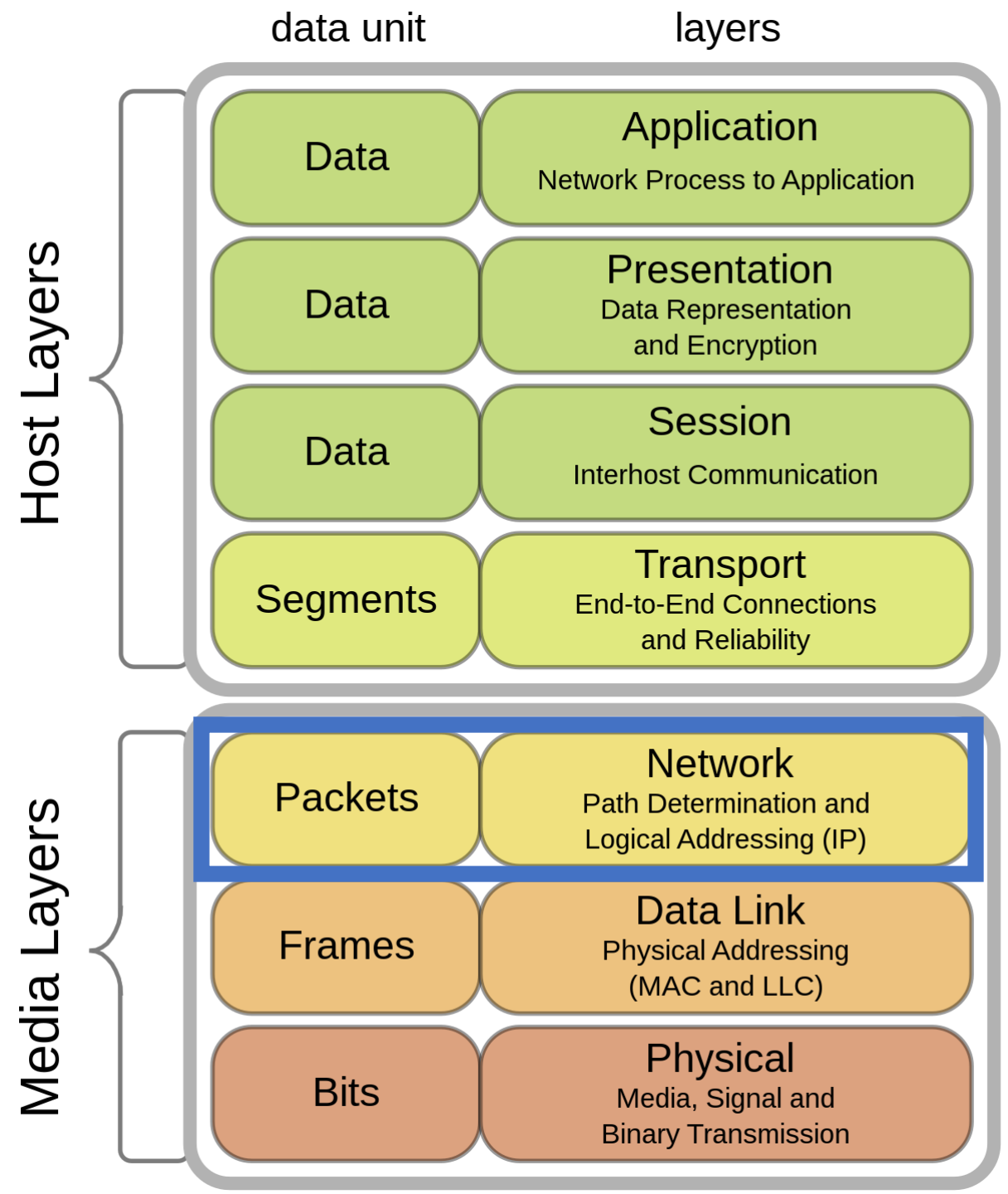
Let's go through the layers to construct a packet



Example – Reading
<http://fricke.co.uk> web page.



Let's go through the layers to construct a packet



Example – Reading
<http://fricke.co.uk> web page.

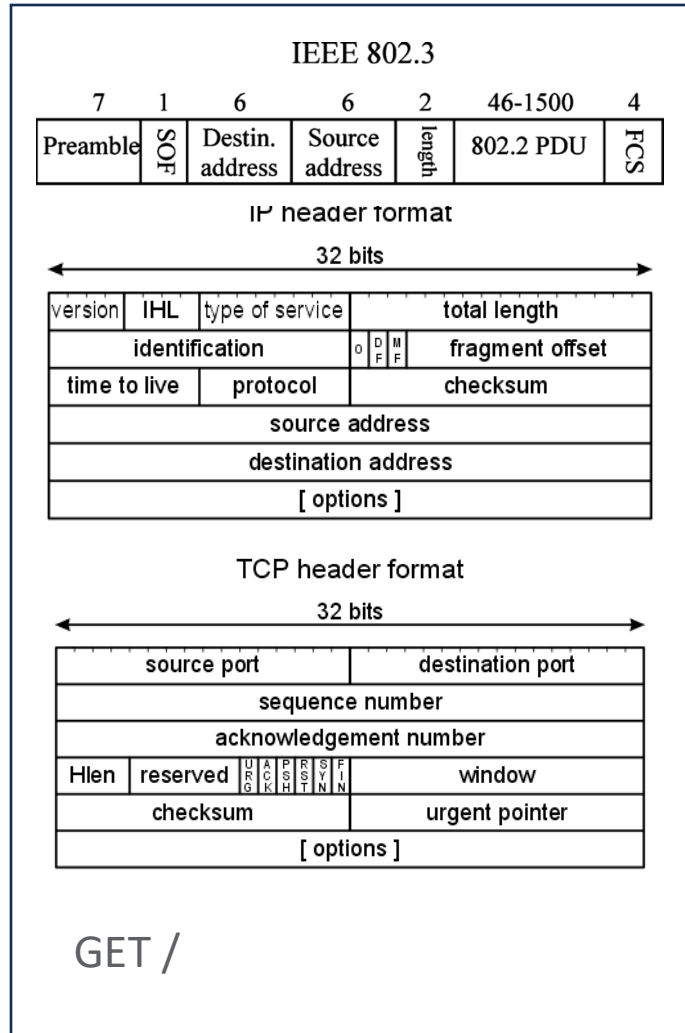
Sending Computer

data unit

layers

Host Layers

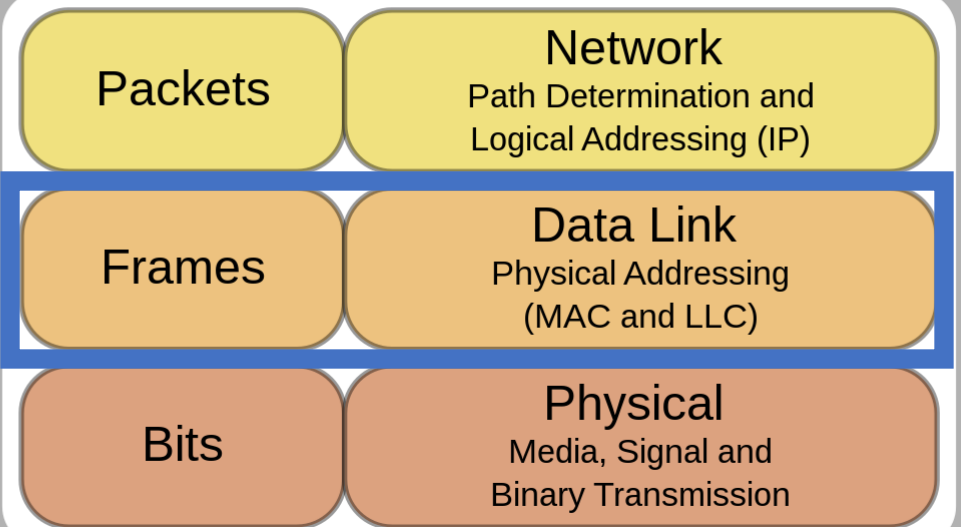
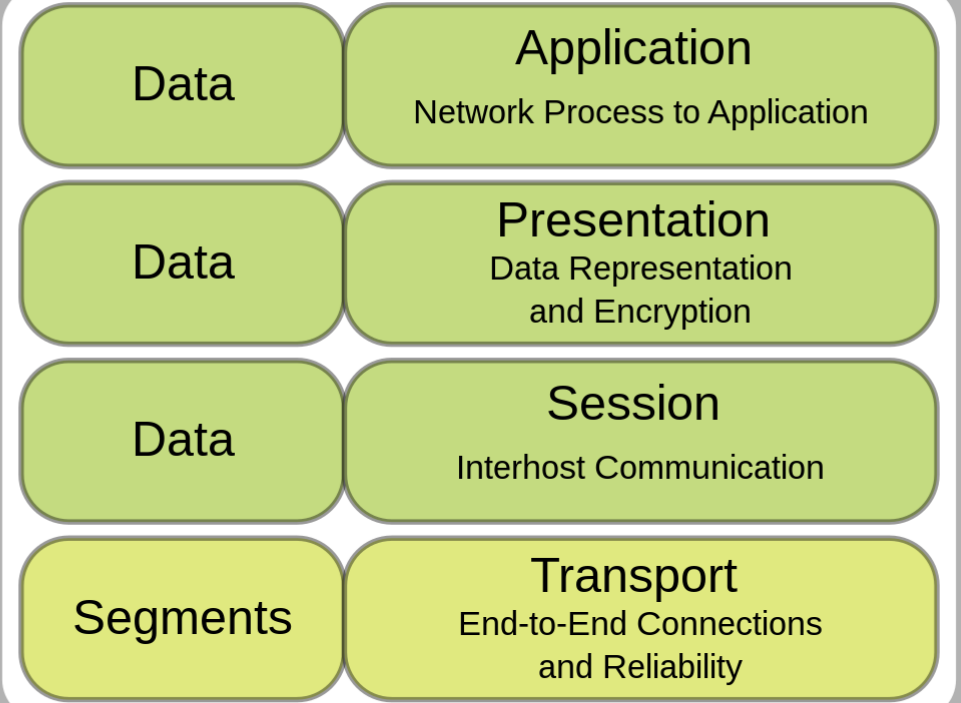
Media Layers



MAC Addresses

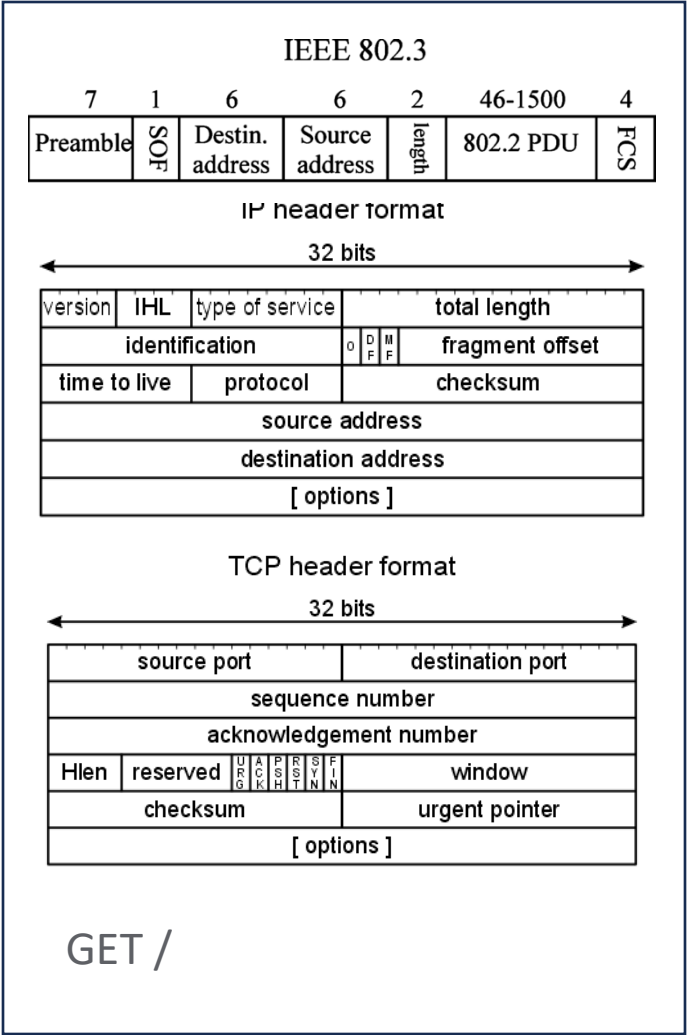
IP Addresses

Port



Let's go through the layers to construct a packet

Example – Reading
<http://fricke.co.uk> web page.



Every time the packet passes through a router the ethernet header is rewritten

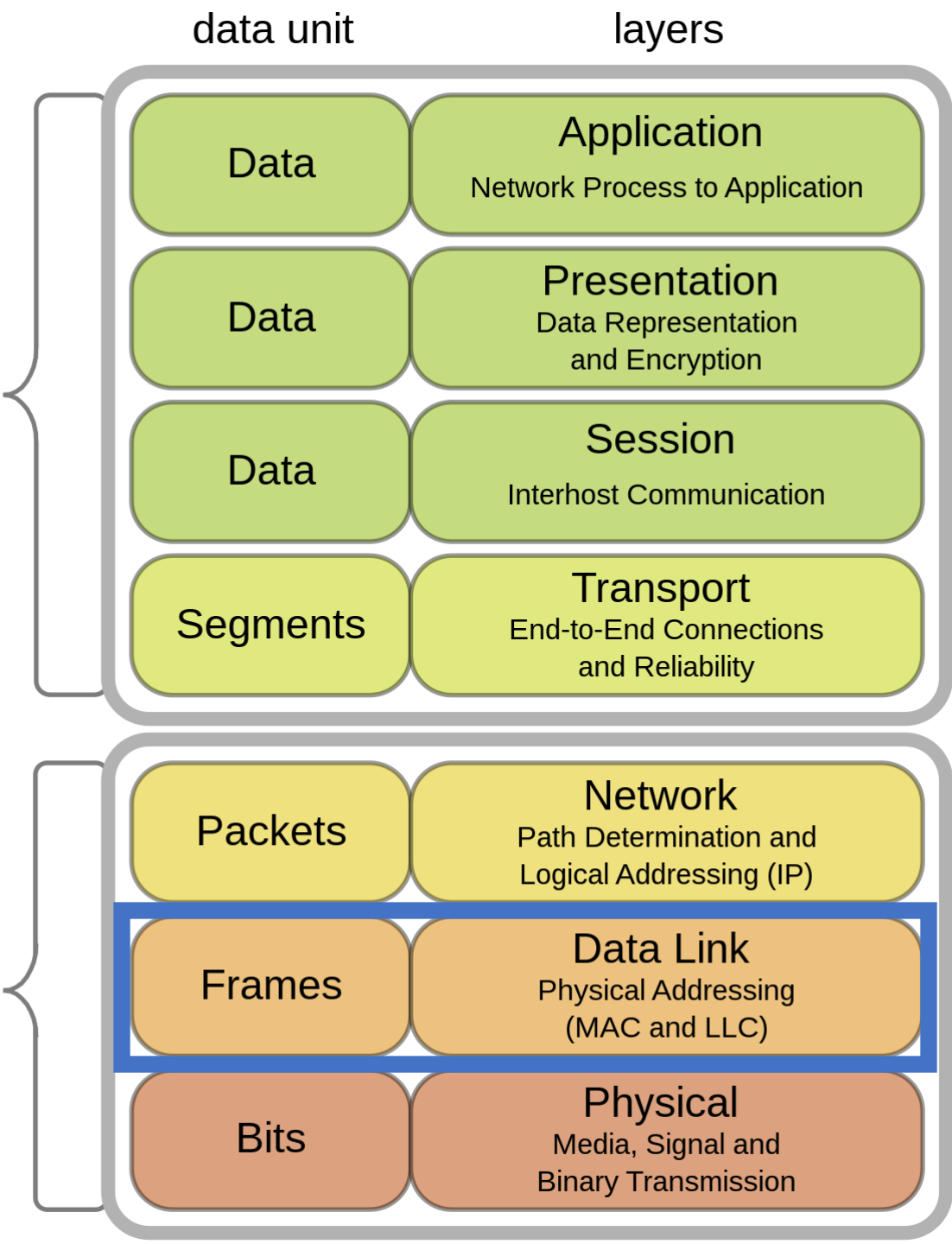
IP Addresses

Port

Let's go through the layers to construct a packet

Host Layers

Media Layers



Example – Reading
<http://fricke.co.uk> web page.

Receiving Computer

data unit

layers

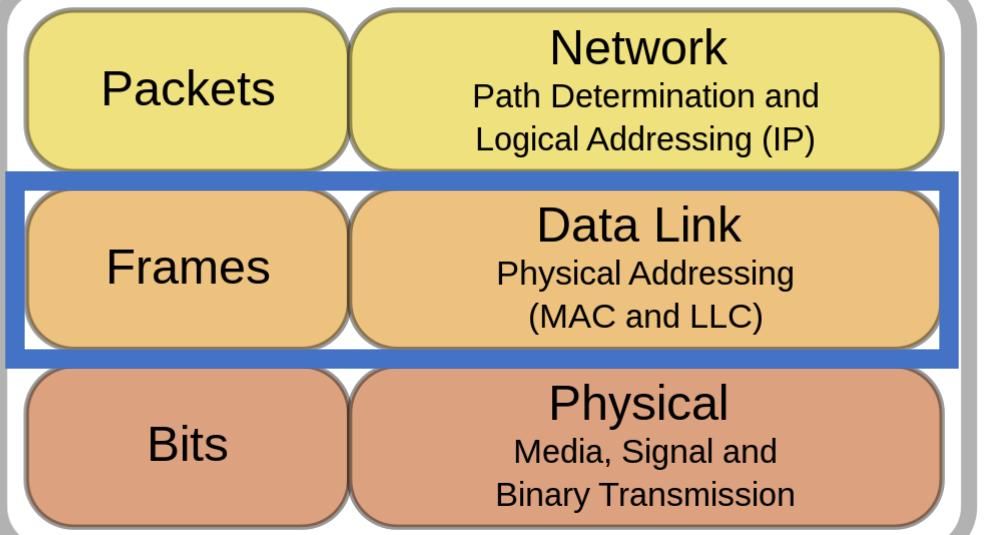
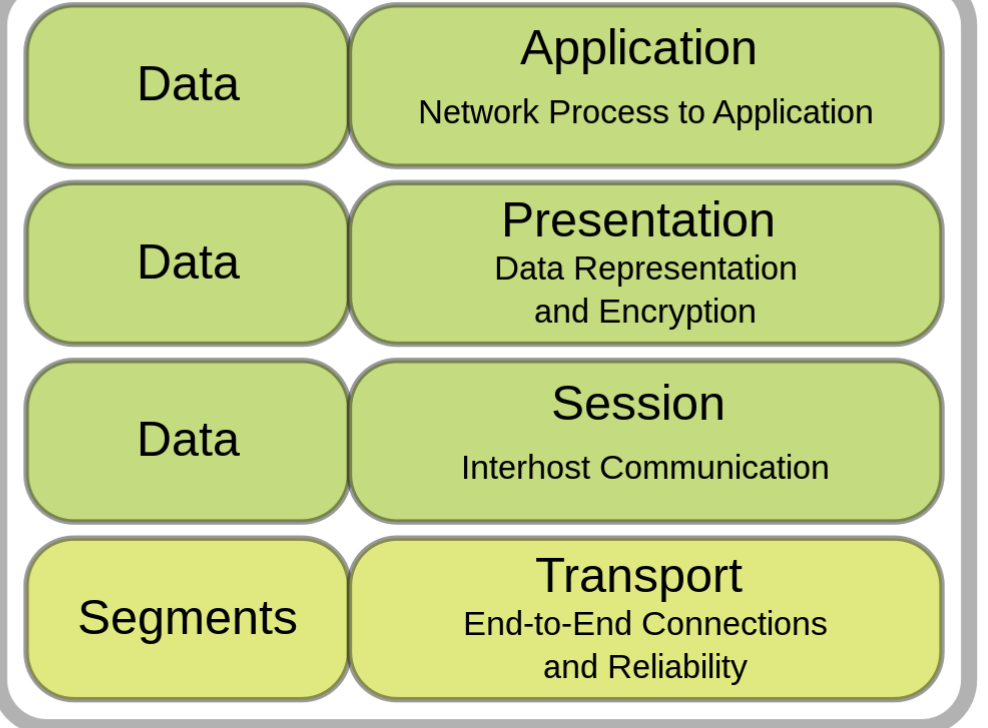
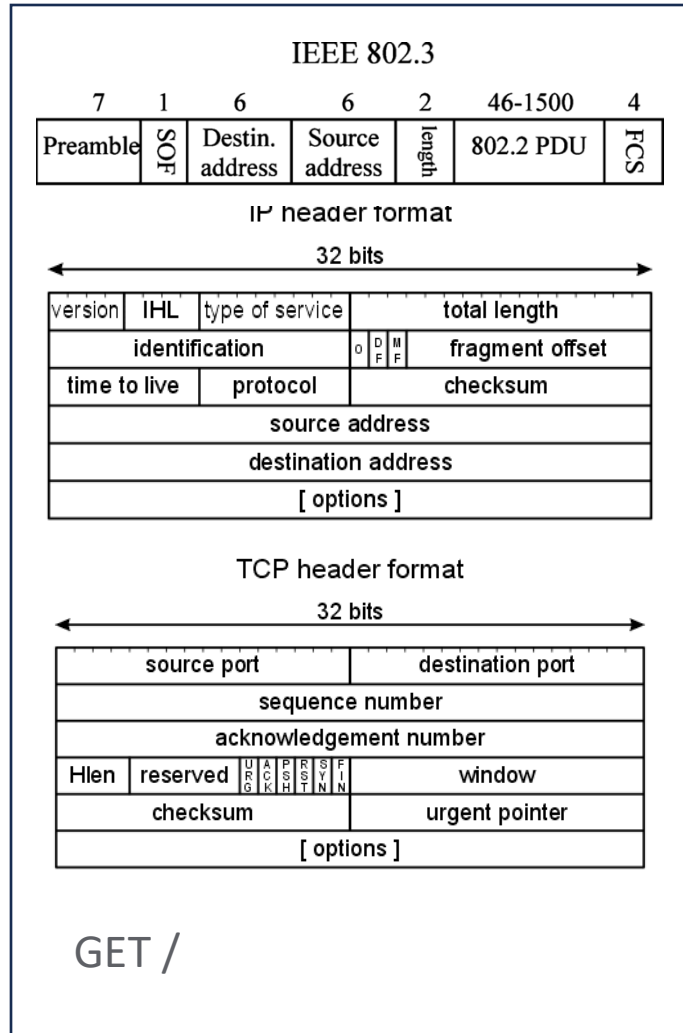
Host Layers

Media Layers

MAC Addresses

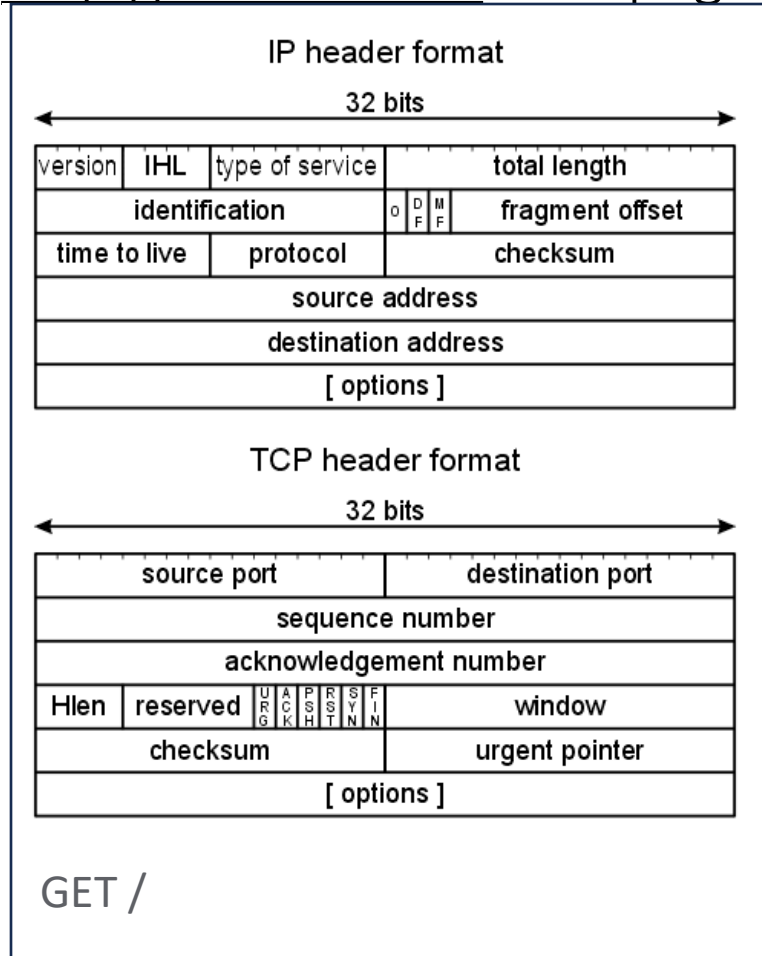
IP Addresses

Port



Let's go through the layers to construct a packet

Example – Reading
<http://fricke.co.uk> web page.



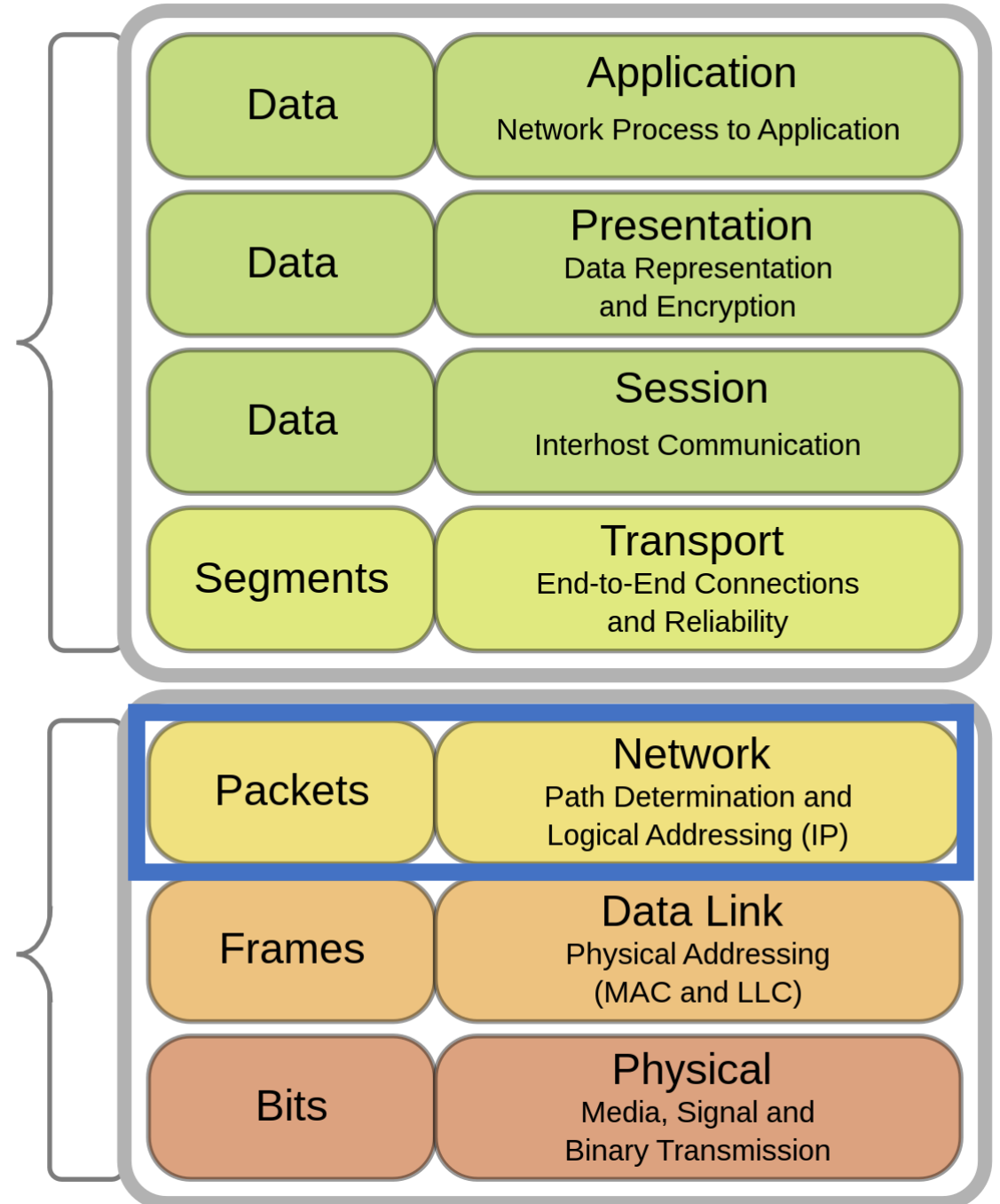
Receiving Computer

Host Layers

Media Layers

data unit

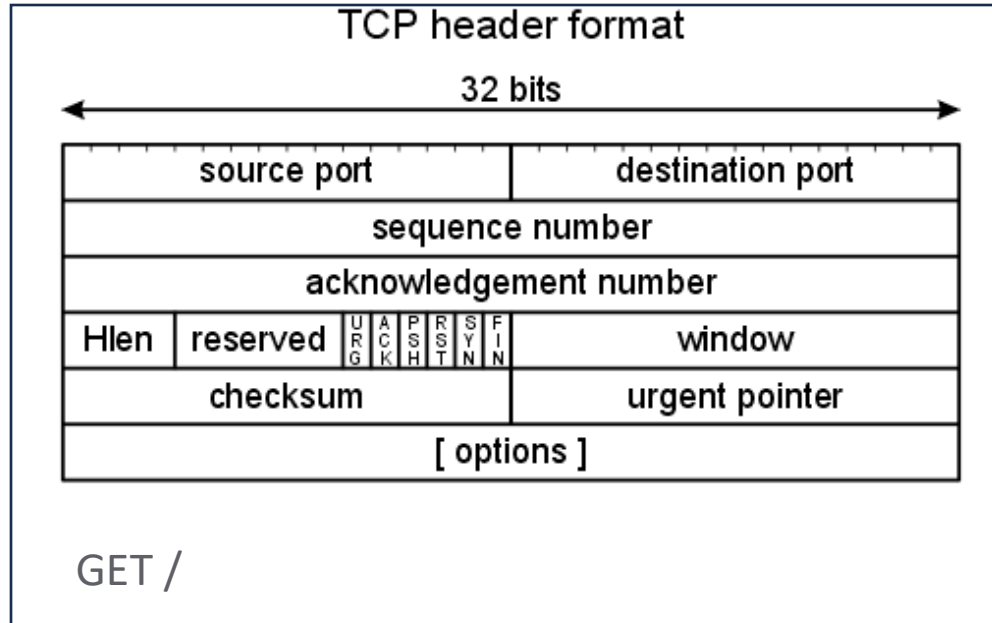
layers



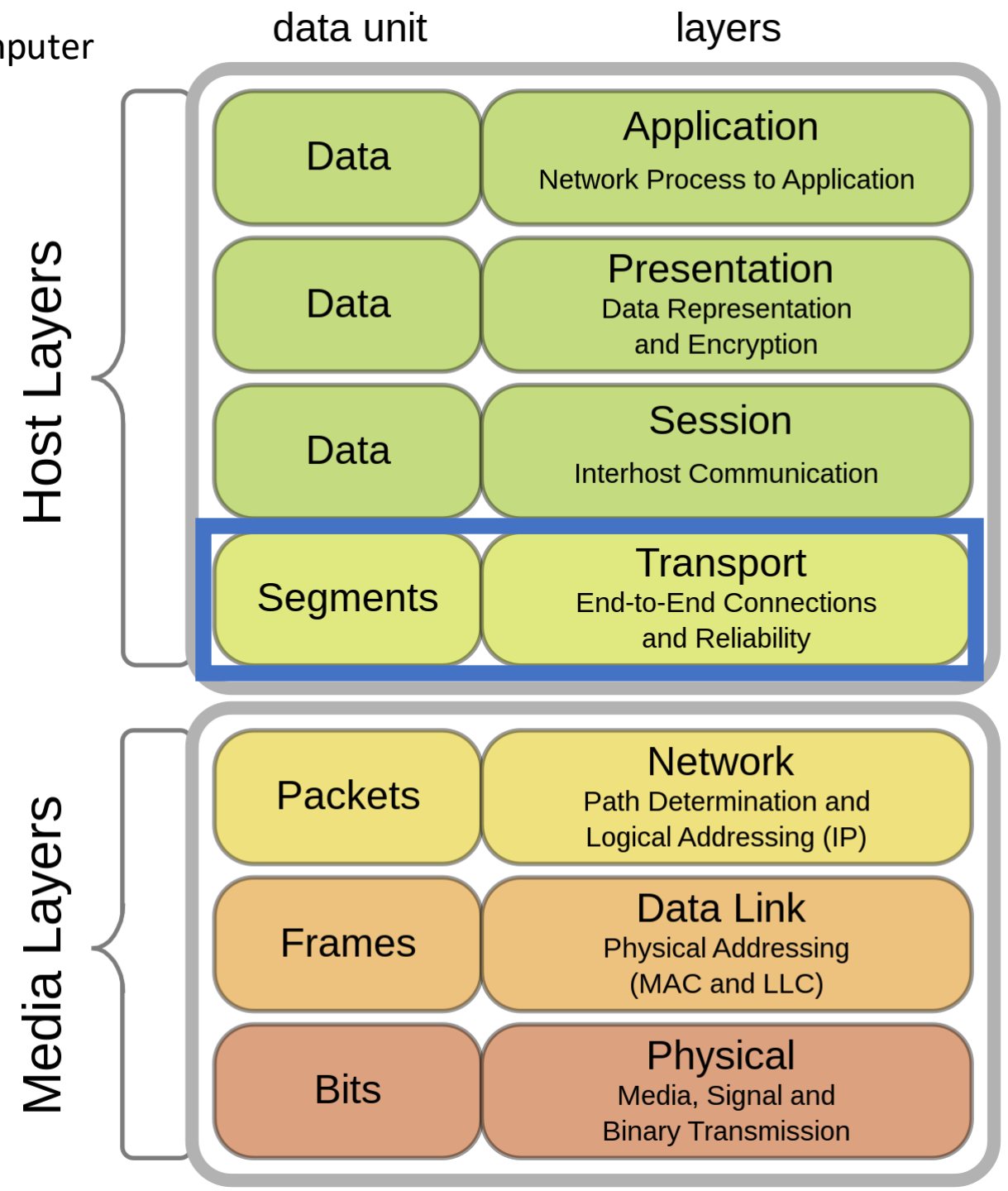
Let's go through the layers to construct a packet

Example – Reading
<http://fricke.co.uk> web page.

Receiving Computer



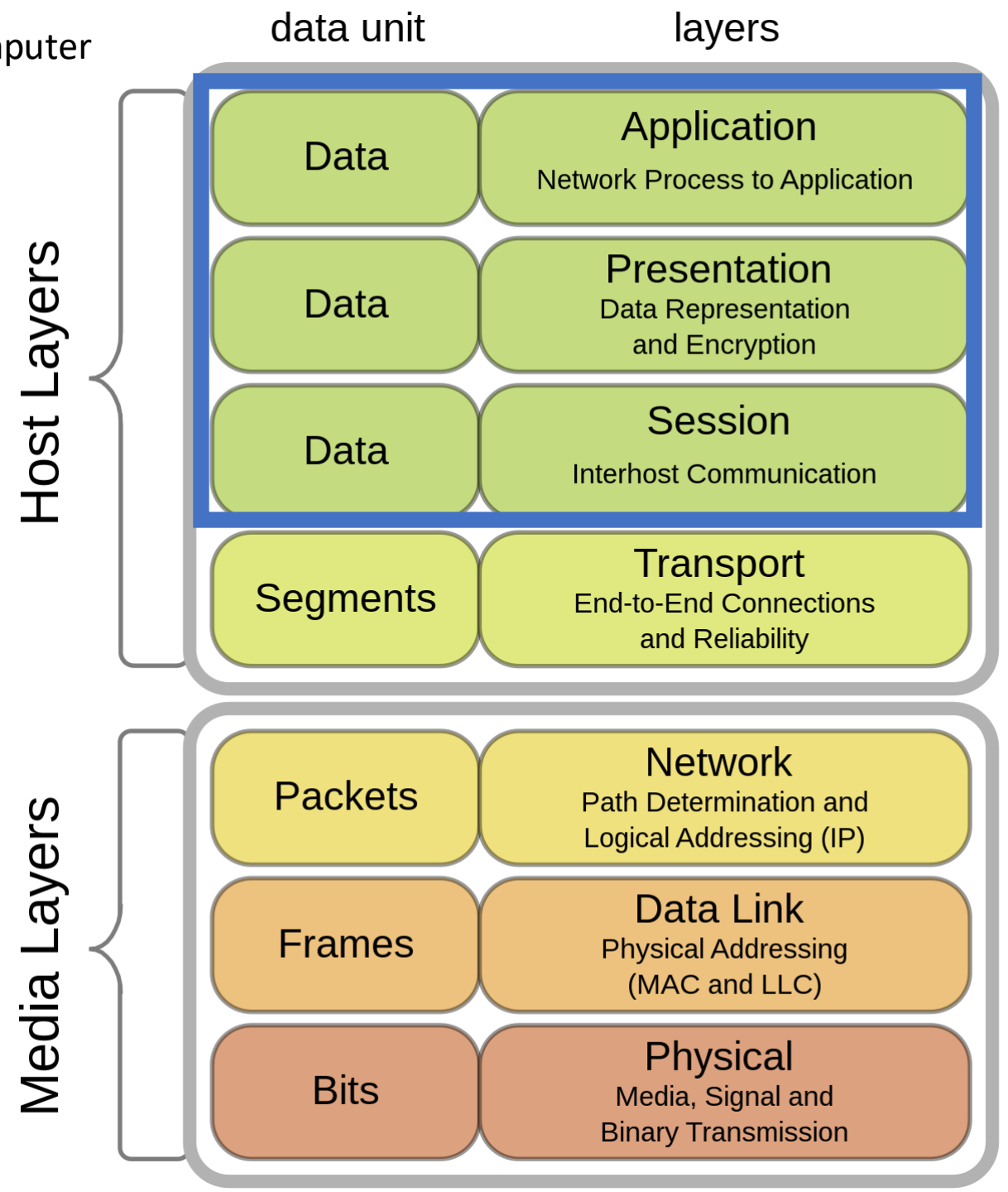
Let's go through the layers to construct a packet



Example – Reading
<http://fricke.co.uk> web page.

GET /

Receiving Computer



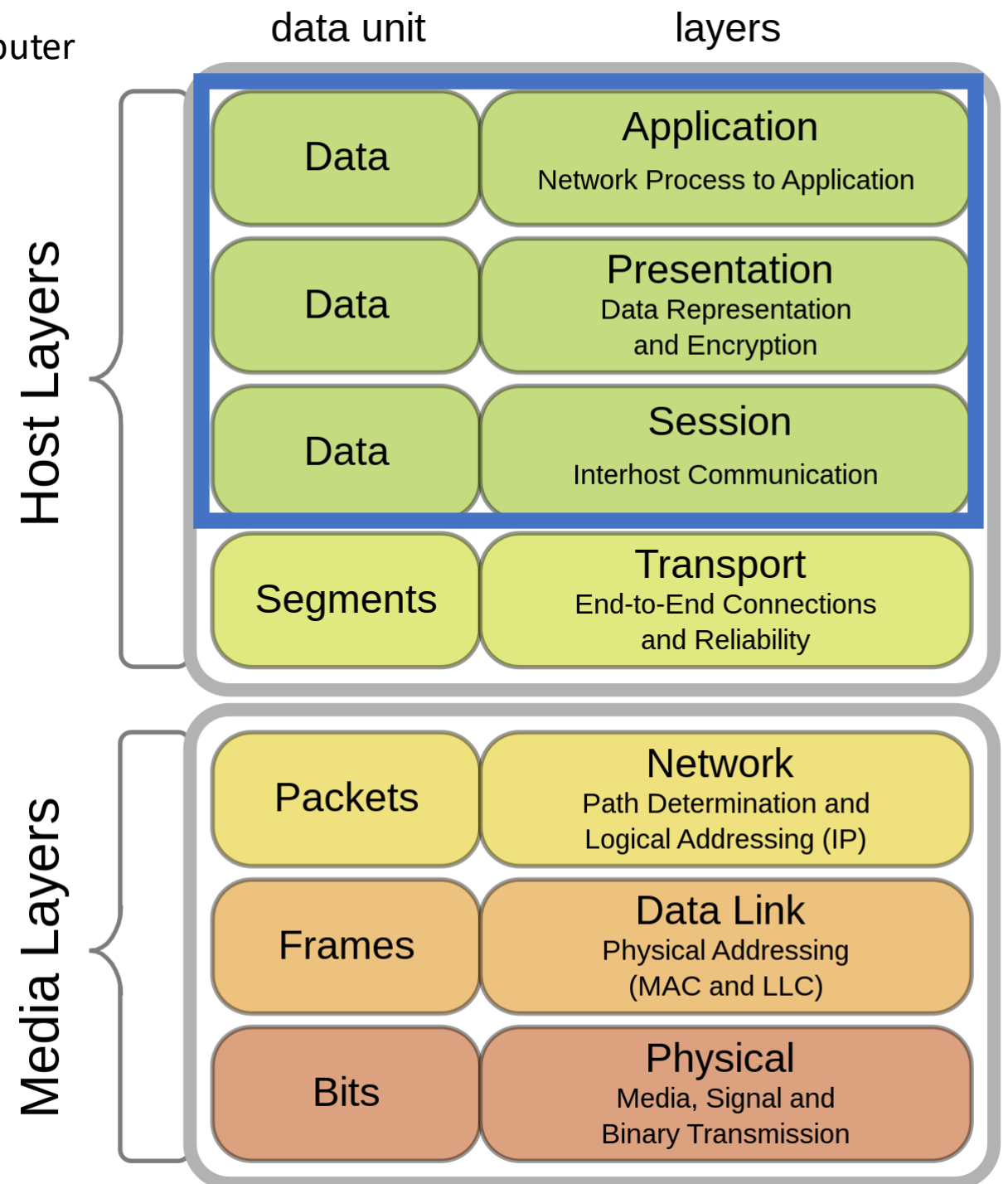
Let's go through the layers to construct a packet

Example – Reading
<http://fricke.co.uk> web page.

```
<!DOCTYPE html>
<html>
<head>
<style>office
body {
  font-family: Arial;
  font-size: 16px;
  margin: 0;
}
```

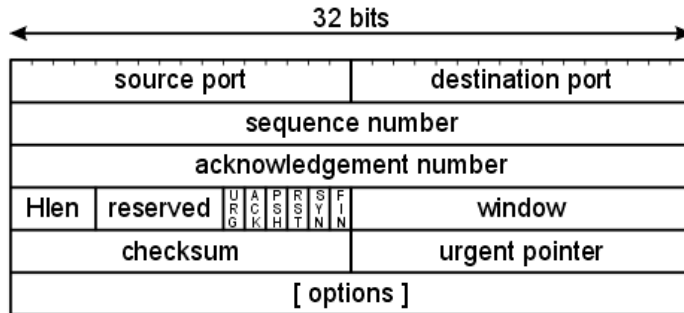
Let's go through the layers to construct a packet

Replying Computer



Example – Reading
<http://fricke.co.uk> web page.

TCP header format



```
<!DOCTYPE html>
<html>
<head>
<style>office
body {
  font-family: Arial;
  font-size: 16px;
  margin: 0;
}
```

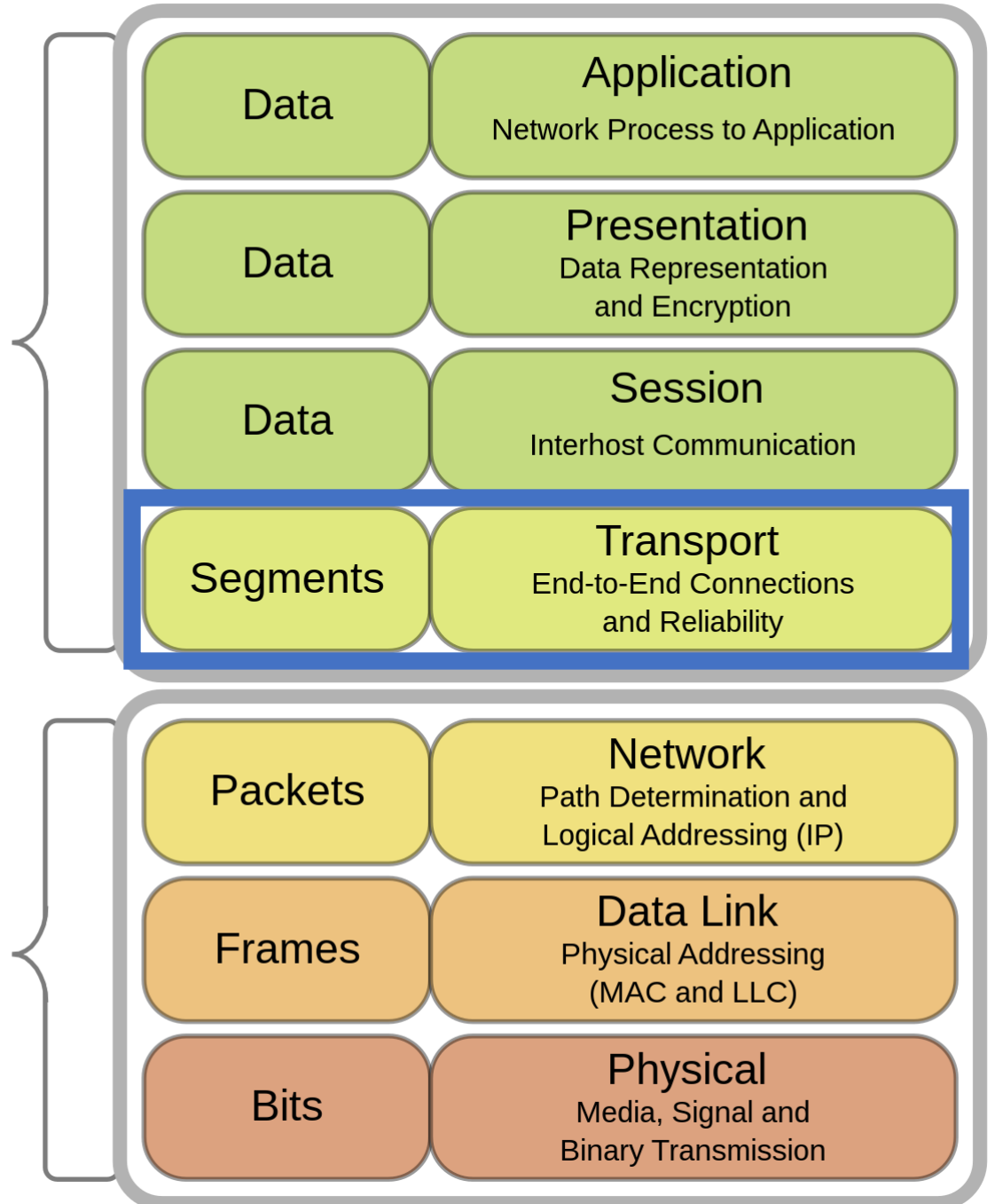
Replying Computer

Host Layers

Media Layers

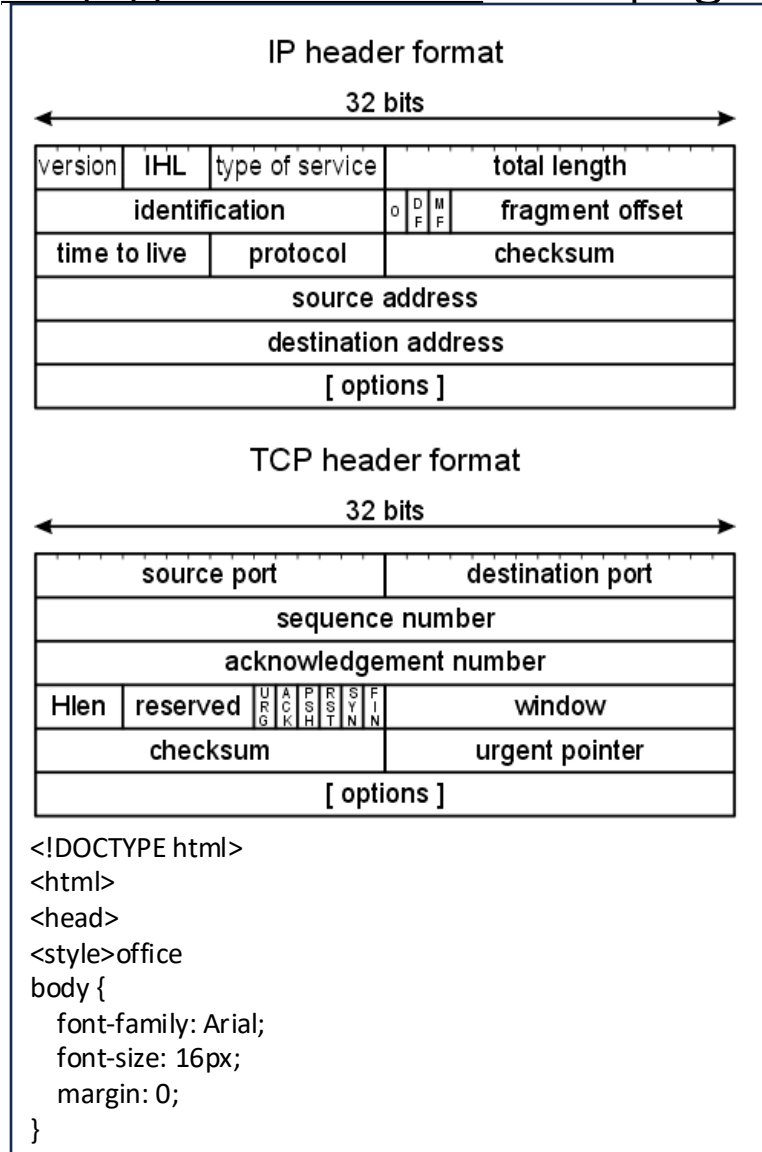
data unit

layers



Let's go through the layers to construct a packet

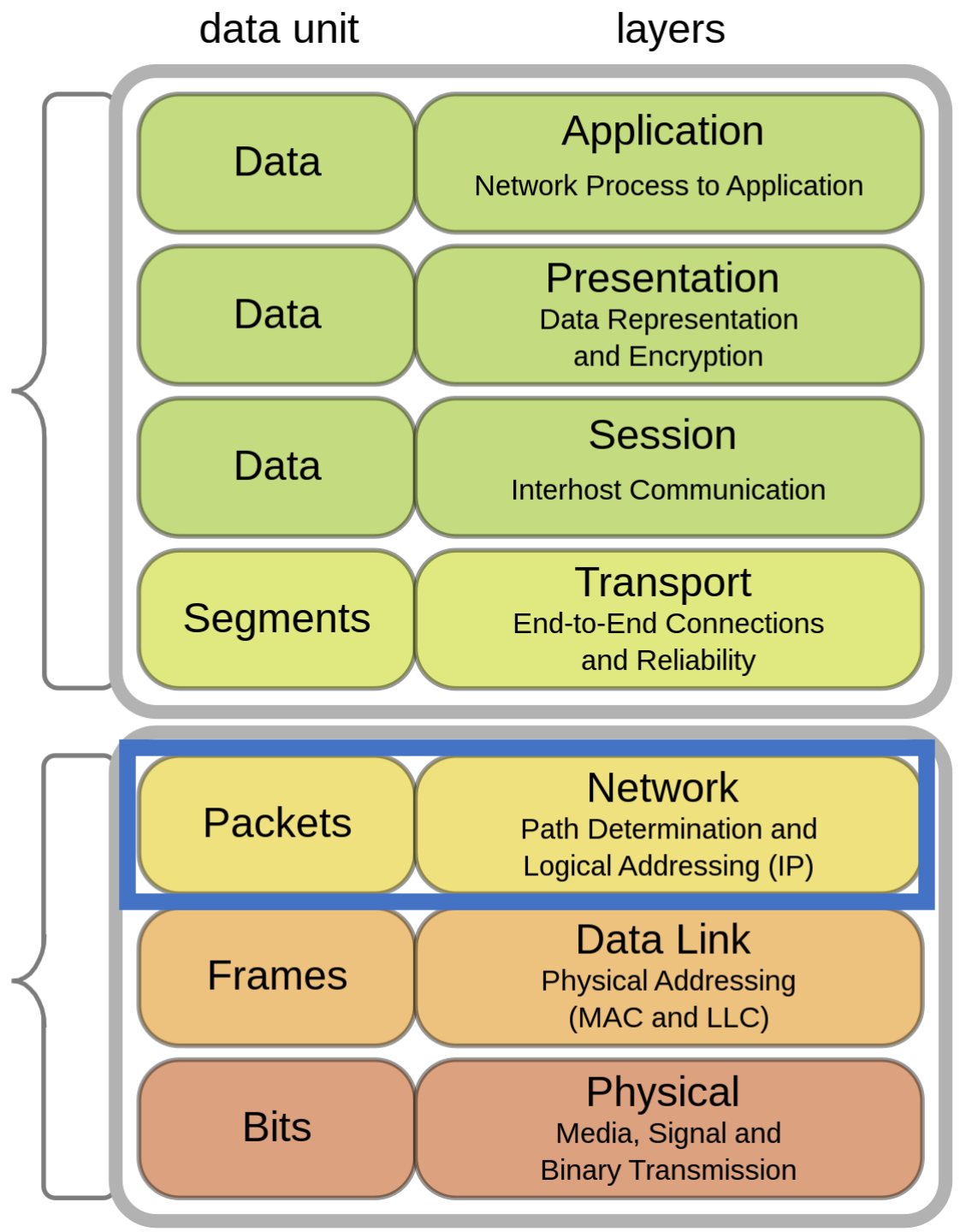
Example – Reading
<http://fricke.co.uk> web page.



Replying Computer

Host Layers

Media Layers



Example – Reading
<http://fricke.co.uk> web page.

Replying Computer

data unit

layers

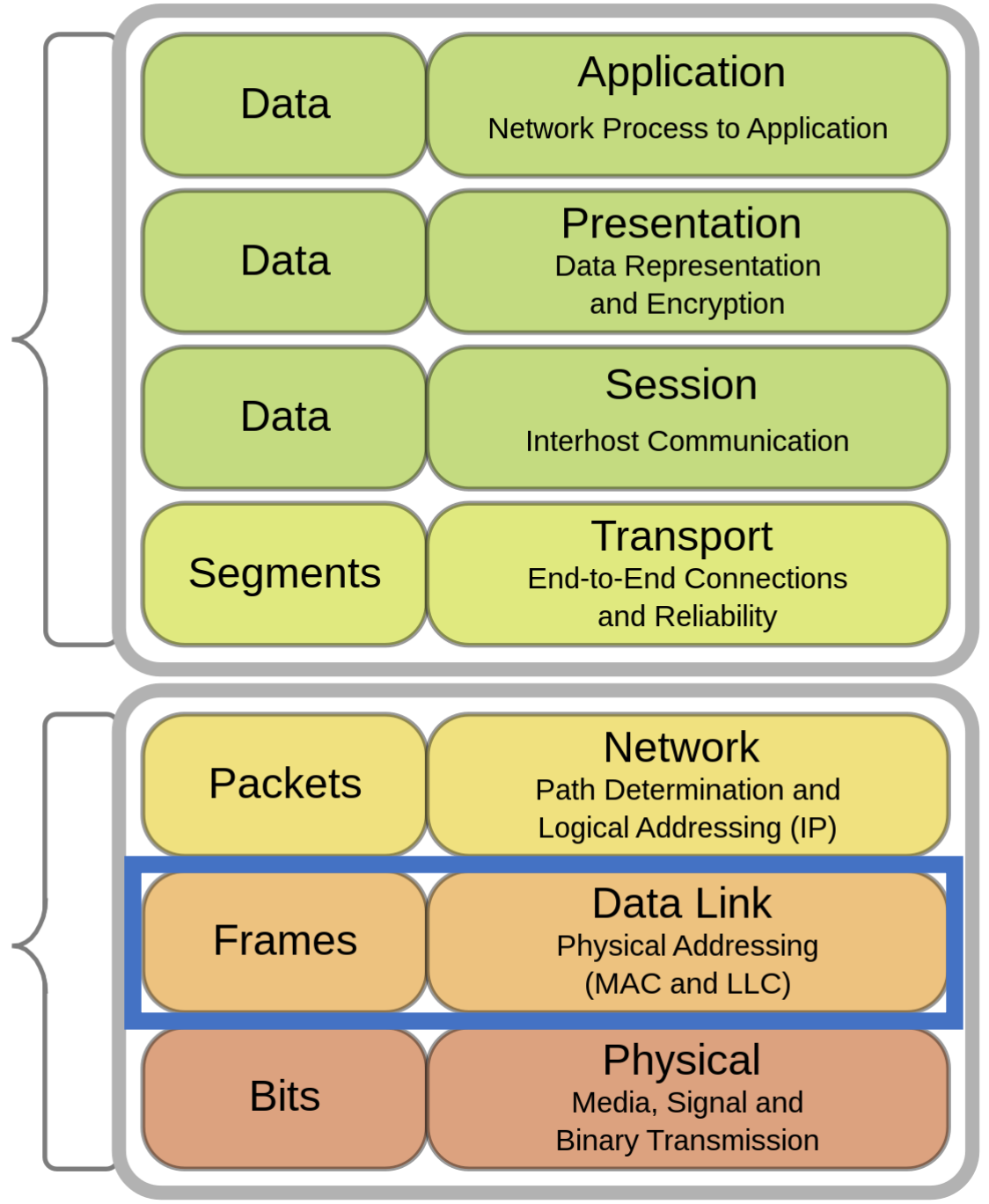
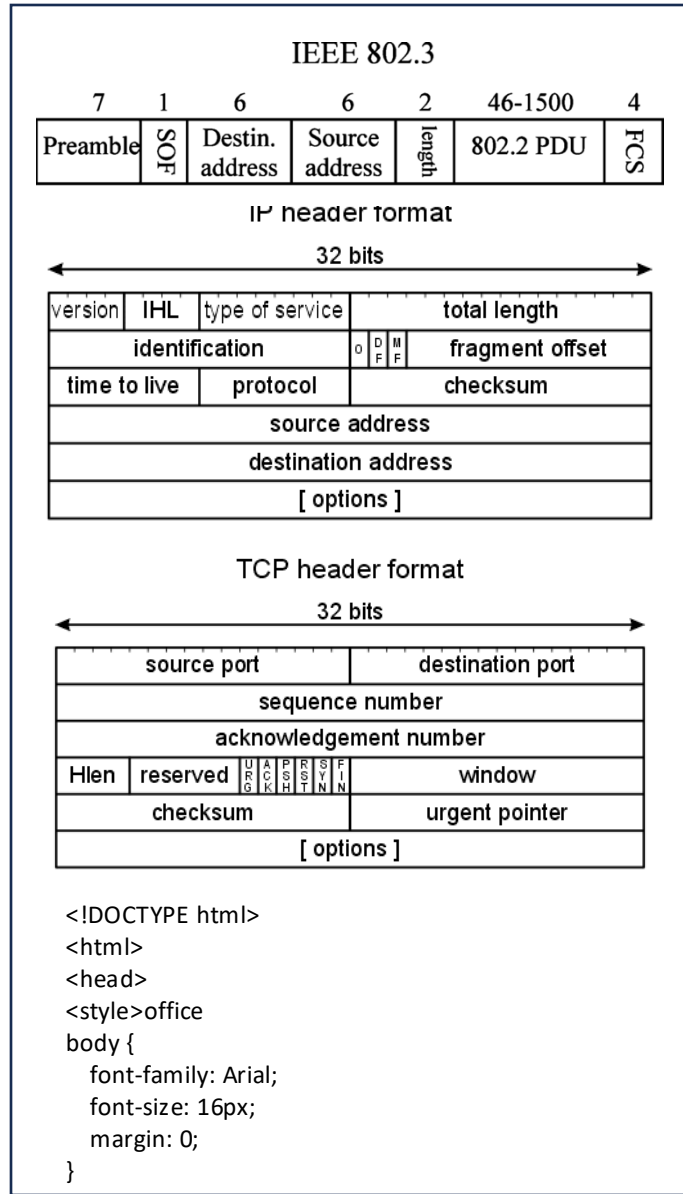
MAC Addresses

IP Addresses

Port

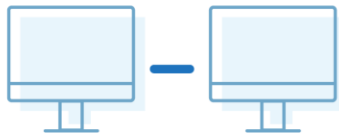
Host Layers

Media Layers

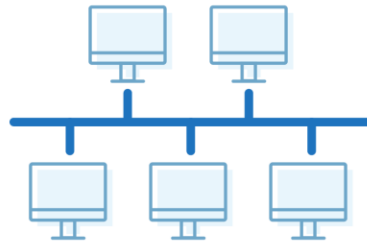


Network Topology Types

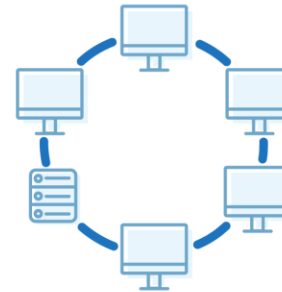
1 Point to point



2 Bus



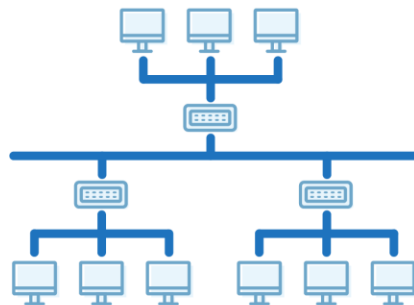
3 Ring



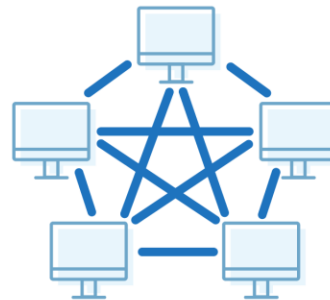
4 Star



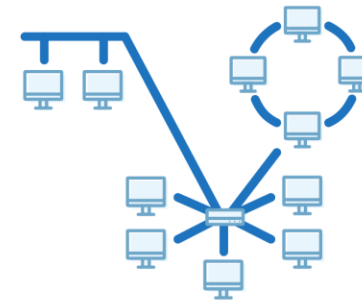
5 Tree

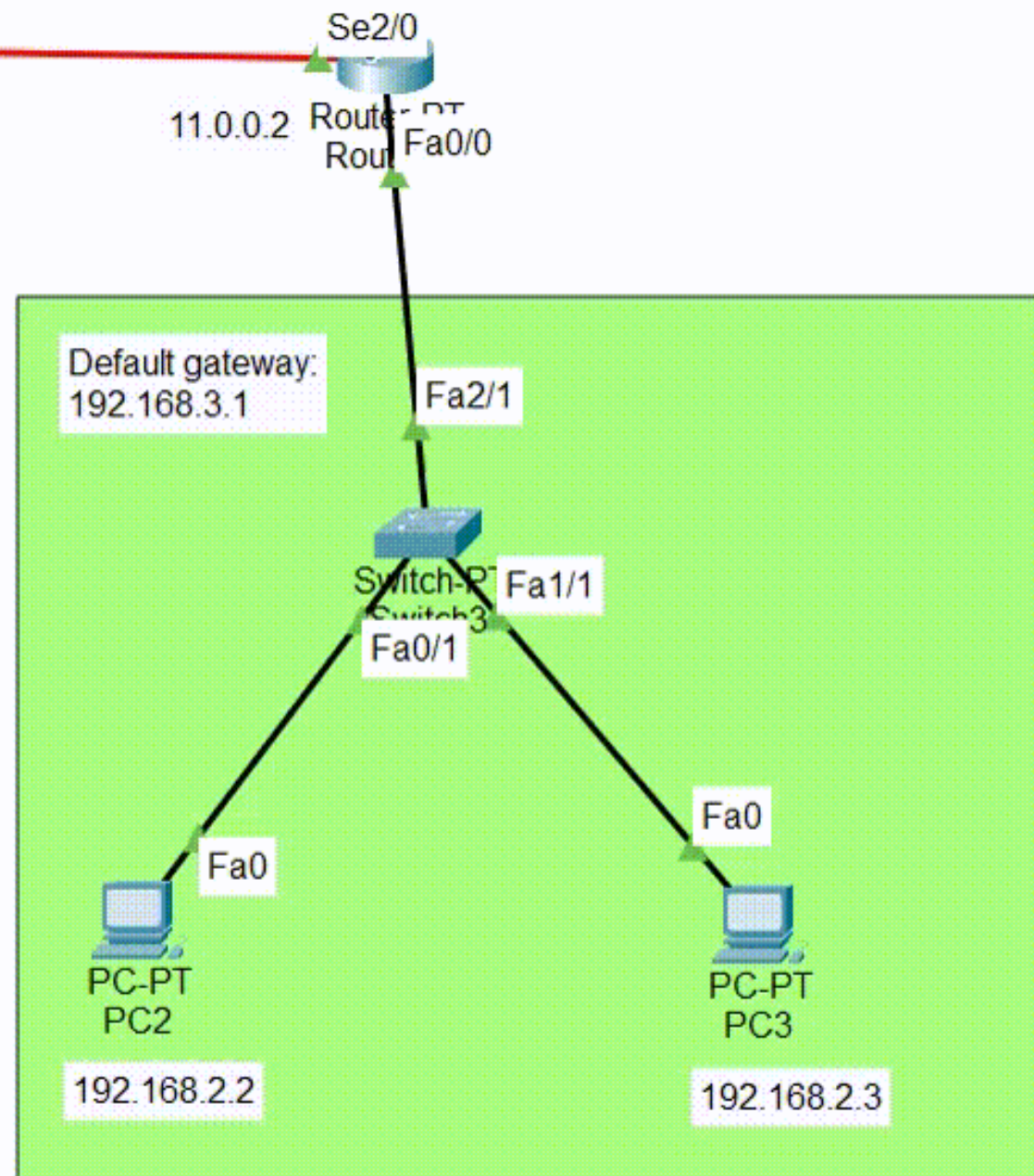
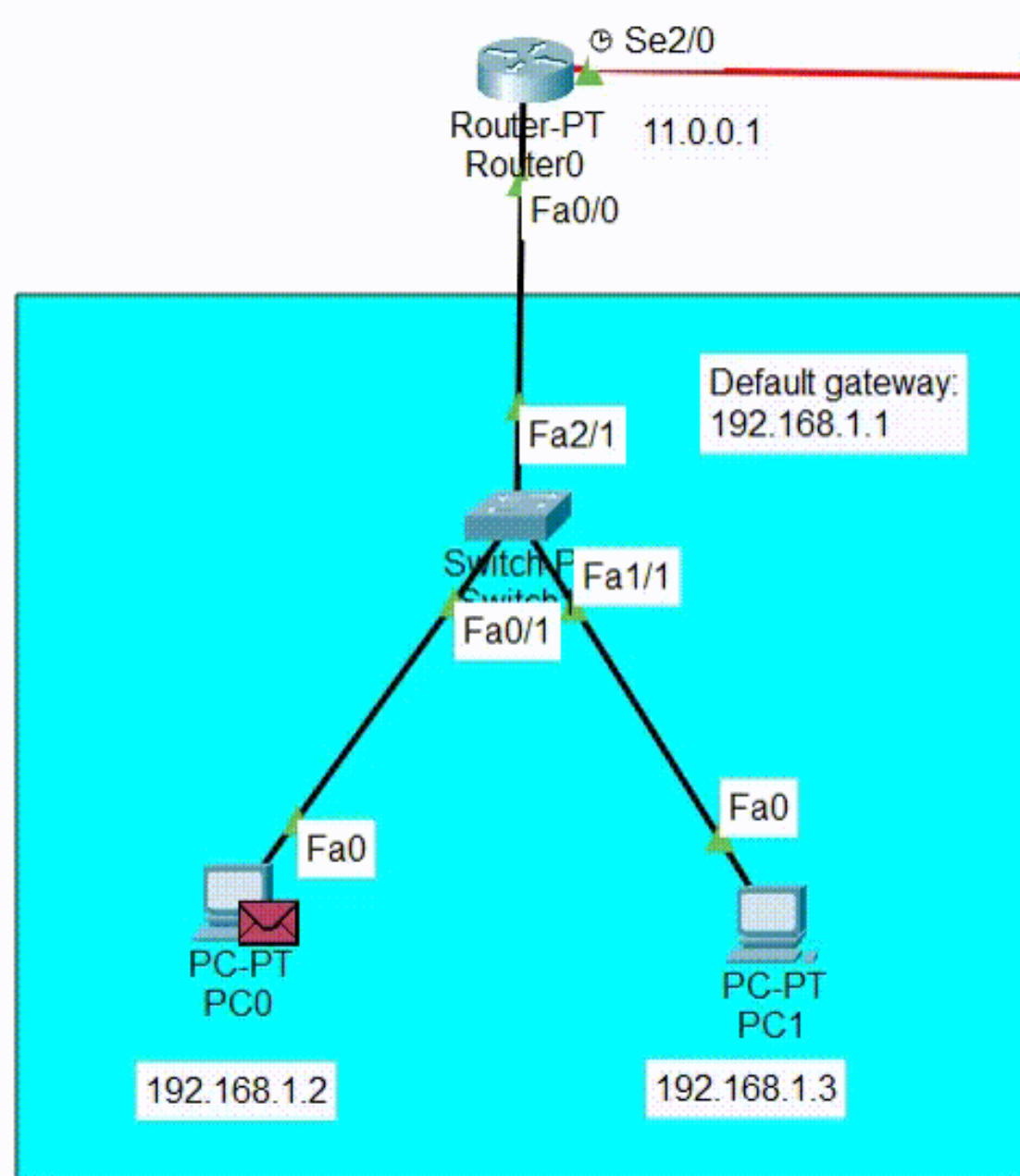


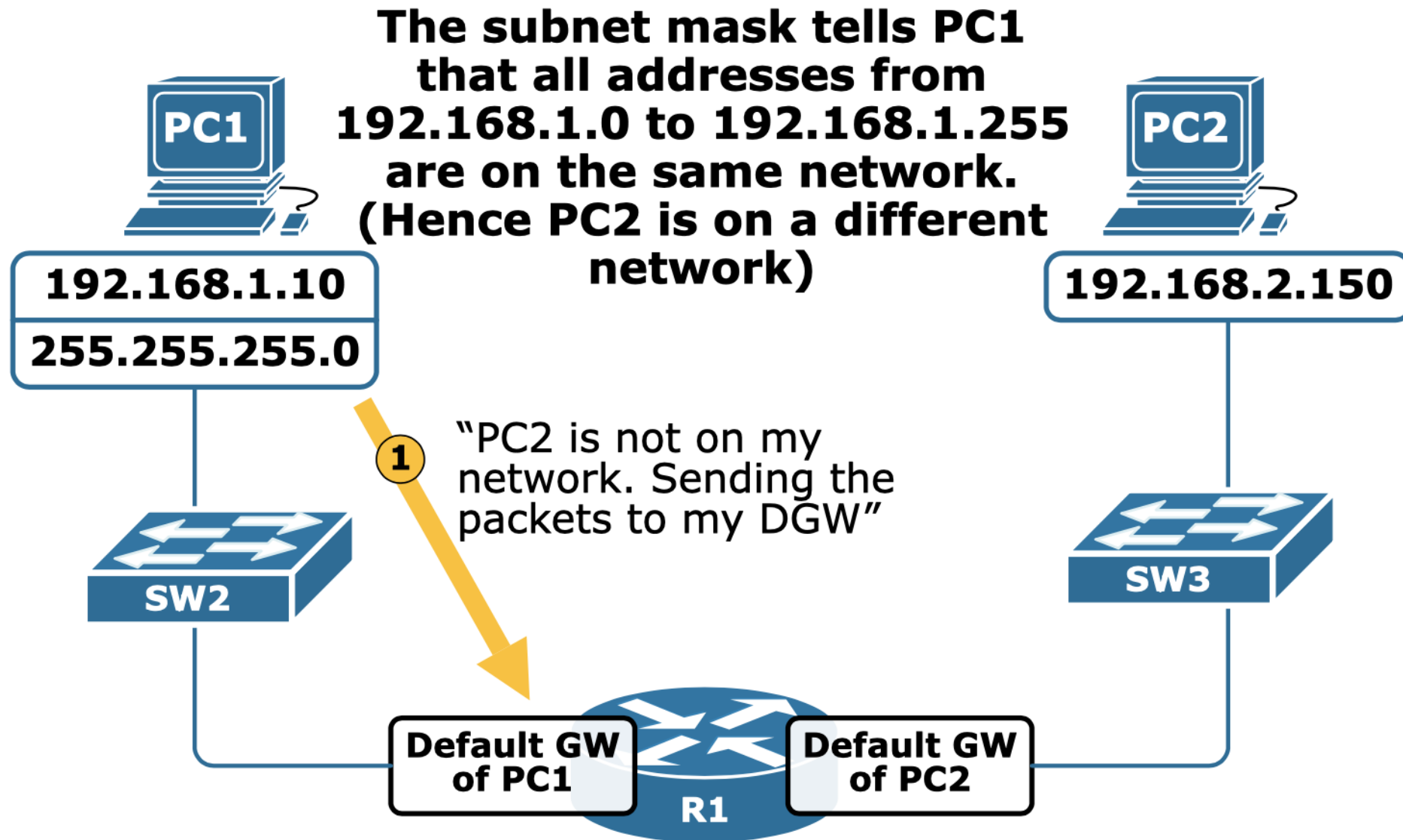
6 Mesh

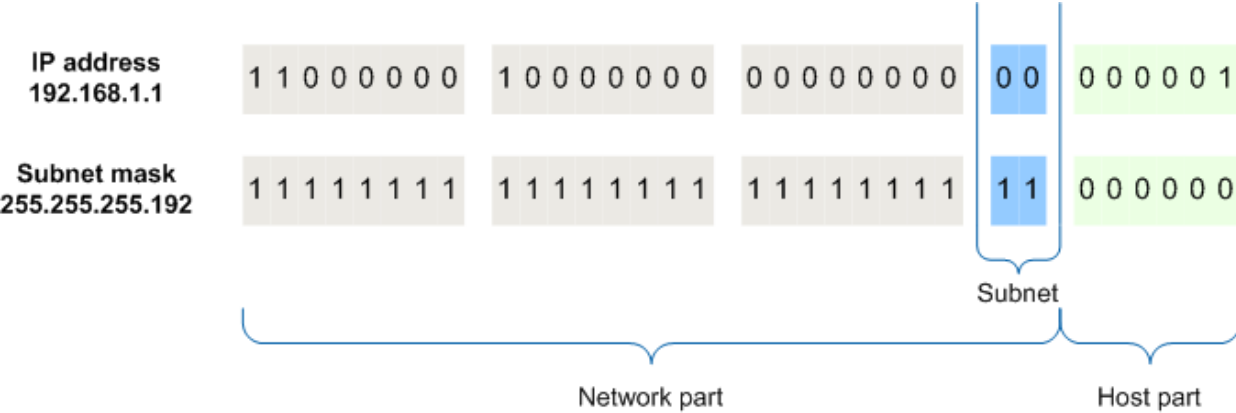


7 Hybrid





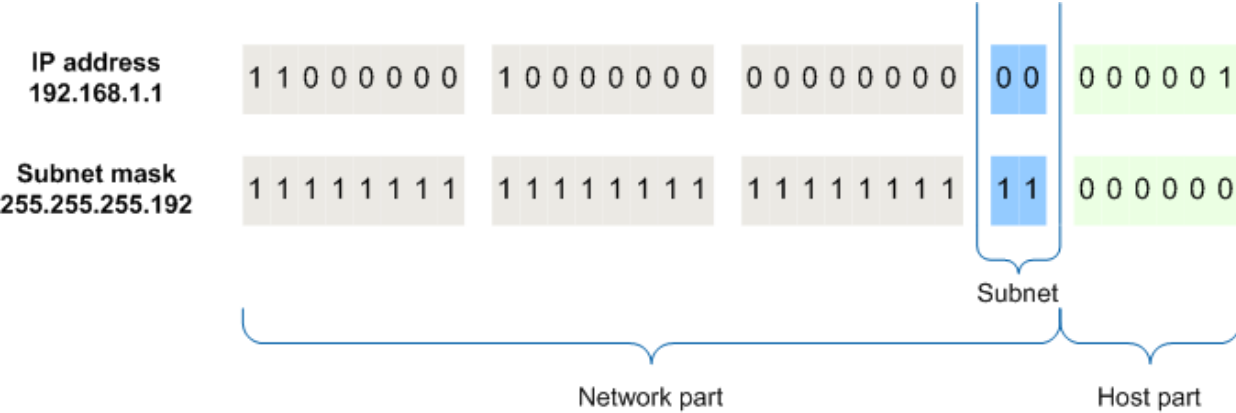




IP Subnet Masks

The mask divides addresses into addresses that need to go through a router (network address) and those that don't (host address).

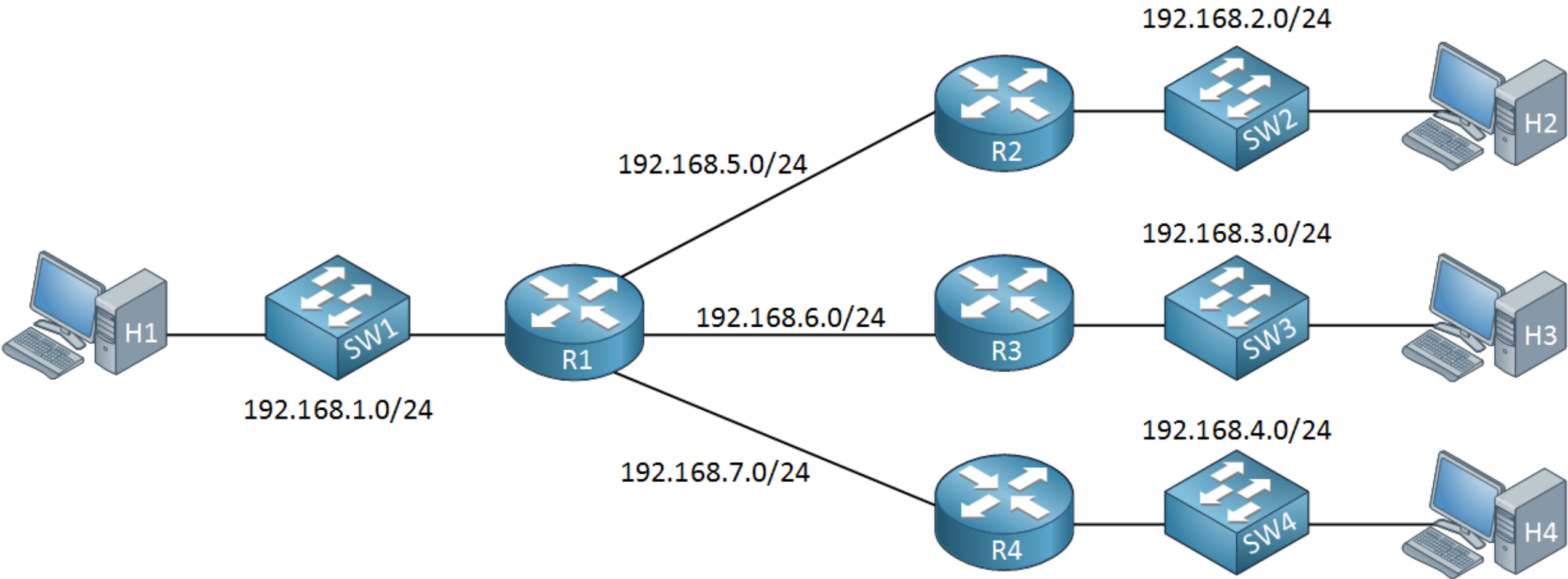
Network Bits	Subnet Mask	Bits Borrowed	Subnets	Hosts/Subnet
8	255.0.0.0	0	1	16777214
9	255.128.0.0	1	2	8388606
10	255.192.0.0	2	4	4194302
11	255.224.0.0	3	8	2097150
12	255.240.0.0	4	16	1048574
13	255.248.0.0	5	32	524286
14	255.252.0.0	6	64	262142
15	255.254.0.0	7	128	131070
16	255.255.0.0	8	256	65534
17	255.255.128.0	9	512	32766
18	255.255.192.0	10	1024	16382
19	255.255.224.0	11	2048	8190
20	255.255.240.0	12	4096	4094
21	255.255.248.0	13	8192	2046
22	255.255.252.0	14	16384	1022
23	255.255.254.0	15	32768	510
24	255.255.255.0	16	65536	254
25	255.255.255.128	17	131072	126
26	255.255.255.192	18	262144	62
27	255.255.255.224	19	524288	30
28	255.255.255.240	20	1048576	14
29	255.255.255.248	21	2097152	6
30	255.255.255.252	22	4194304	2



BINARY REPRESENTATION	DECIMAL REPRESENTATION
00000000.00000000.00000000.00000000	0.0.0.0
10000000.00000000.00000000.00000000	128.0.0.0
11000000.00000000.00000000.00000000	192.0.0.0
11100000.00000000.00000000.00000000	224.0.0.0
11110000.00000000.00000000.00000000	240.0.0.0
11111000.00000000.00000000.00000000	248.0.0.0
11111100.00000000.00000000.00000000	252.0.0.0
11111110.00000000.00000000.00000000	254.0.0.0
11111111.00000000.00000000.00000000	255.0.0.0 (Class A)
11111111.10000000.00000000.00000000	255.128.0.0
11111111.11000000.00000000.00000000	255.192.0.0
11111111.11100000.00000000.00000000	255.224.0.0
11111111.11110000.00000000.00000000	255.240.0.0
11111111.11111000.00000000.00000000	255.248.0.0
11111111.11111100.00000000.00000000	255.252.0.0
11111111.11111110.00000000.00000000	255.254.0.0
11111111.11111111.00000000.00000000	255.255.0.0 (Class B)
11111111.11111111.10000000.00000000	255.255.128.0
11111111.11111111.11000000.00000000	255.255.192.0
11111111.11111111.11100000.00000000	255.255.224.0
11111111.11111111.11110000.00000000	255.255.240.0
11111111.11111111.11111000.00000000	255.255.248.0
11111111.11111111.11111100.00000000	255.255.252.0
11111111.11111111.11111110.00000000	255.255.254.0
11111111.11111111.11111111.00000000	255.255.255.0 (Class C)
11111111.11111111.11111111.10000000	255.255.255.128
11111111.11111111.11111111.11000000	255.255.255.192
11111111.11111111.11111111.11100000	255.255.255.224
11111111.11111111.11111111.11110000	255.255.255.240
11111111.11111111.11111111.11111000	255.255.255.248
11111111.11111111.11111111.11111100	255.255.255.252
11111111.11111111.11111111.11111110	255.255.255.254
11111111.11111111.11111111.11111111	255.255.255.255

IP Subnet Masks

The mask divides addresses into addresses that need to go through a router (network address) and those that don't (host address).



Networks have a hierarchical structure

Special Addresses

A subnetwork consists of three types of addresses:

The network address (first address in the subnet)

The host addresses – can be assigned to hosts

The broadcast address – last address in the range.

Packets sent to the broadcast address go to all devices in the subnet.

The network address is the address of the network itself and is used for routing.

IP Address of your Computer (base 8 or “dot” notation):

192.168.22.187

IP Address of your Computer (base 2 or “binary” notation):

11000000 10101000 00010110 10111011

Subnet Mask (base 8 or “dot” notation):

255.255.255.0

Subnet Mask (base 2 or “binary” notation):

11111111 11111111 11111111 00000000

What is the Network IP address (the one your ISP assigns)?

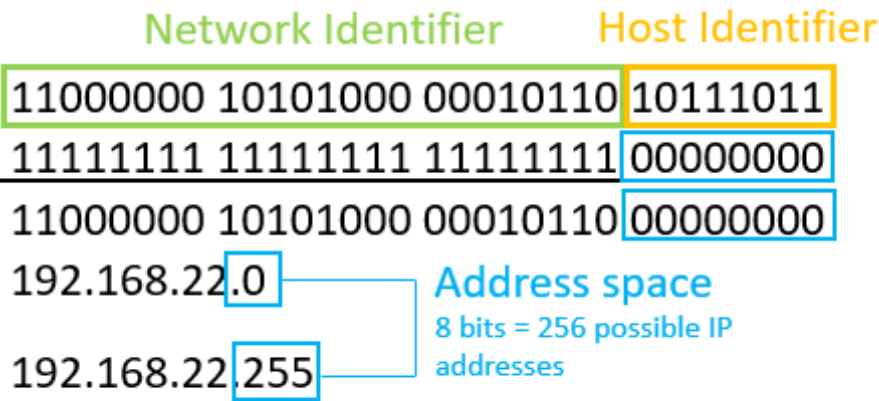
Computer IP address (could be any device on network)

Subnet mask

Network Address (binary)

Network Address (dot notation)

Broadcast Address (dot notation)



[matthew@localhost ~]\$ sudo yum install traceroute
[sudo] password for matthew:
Sorry, try again.
[sudo] password for matthew:
Last metadata expiration check: 0:46:44 ago on Wed 07 Feb 2024 12:12:07 PM CST.
Dependencies resolved.

=====

====

Package	Architecture	Version	Repository	Size
=====				

Installing:				
traceroute	x86_64	3:2.1.0-16.el9	baseos	57 k

Transaction Summary

=====Install 1 Package

Total download size: 57 k
Installed size: 108 k
Is this ok [y/N]: y
Downloading Packages:

```
[matthew@localhost ~]$ traceroute google.com
```

```
traceroute to google.com (172.253.115.101), 30 hops max, 60 byte packets
```

```
1 * * *
2 198.83.93.10 (198.83.93.10) 0.378 ms 0.358 ms 0.372 ms
3 10.2.18.65 (10.2.18.65) 0.489 ms 0.469 ms 0.449 ms
4 10.2.18.85 (10.2.18.85) 0.893 ms 0.873 ms 0.855 ms
5 10-1-3-3472.ear3.Denver1.Level3.net (4.14.121.49) 8.568 ms 8.587 ms 8.568 ms
6 Google-level3-Denver1.Level3.net (4.68.110.218) 8.580 ms 8.554 ms 8.515 ms
7 * * *
8 142.251.51.154 (142.251.51.154) 9.602 ms 142.251.51.220 (142.251.51.220) 8.535 ms 142.251.61.182 (142.251.61.182) 9.106 ms
9 142.251.49.92 (142.251.49.92) 9.026 ms 209.85.143.66 (209.85.143.66) 9.686 ms 9.652 ms
10 172.253.51.78 (172.253.51.78) 9.335 ms 172.253.51.82 (172.253.51.82) 9.611 ms 172.253.51.78 (172.253.51.78) 9.359 ms
11 172.253.74.22 (172.253.74.22) 19.065 ms 19.116 ms 18.770 ms
12 142.251.250.180 (142.251.250.180) 20.471 ms 18.762 ms 172.253.77.126 (172.253.77.126) 19.793 ms
13 192.178.72.203 (192.178.72.203) 31.754 ms 31.576 ms 192.178.72.195 (192.178.72.195) 31.549 ms
14 * 192.178.81.230 (192.178.81.230) 49.200 ms 192.178.81.232 (192.178.81.232) 50.735 ms
15 172.253.67.52 (172.253.67.52) 49.206 ms 172.253.67.50 (172.253.67.50) 48.991 ms 172.253.67.0 (172.253.67.0) 48.884 ms
16 172.253.66.159 (172.253.66.159) 43.625 ms 172.253.66.155 (172.253.66.155) 49.569 ms 172.253.66.149 (172.253.66.149) 48.383 ms^
17 * * *
18 * * *...
23 * bg-in-f101.1e100.net (172.253.115.101) 48.658 ms *
```

```
[matthew@localhost ~]$ traceroute fricke.co.uk
```

```
traceroute to fricke.co.uk (50.193.238.94), 30 hops max, 60 byte packets
```

```
1  * * *
2  198.83.93.10 (198.83.93.10)  0.336 ms  0.359 ms  0.339 ms
3  10.2.18.65 (10.2.18.65)  0.409 ms  0.429 ms  0.410 ms
4  te-0-0-0-15-3972-ssag04.albuquerque.nm.albuq.comcast.net (50.226.196.225)  2.218 ms  2.313 ms *
5  * * *
6  be-288-ar02.albuquerque.nm.albuq.comcast.net (96.108.43.153)  3.045 ms  2.794 ms  2.765 ms
7  be-501-ar02.albuquerque.nm.albuq.comcast.net (96.108.67.233)  2.772 ms  2.842 ms *
8  po-1-xar02.albuquerque.nm.albuq.comcast.net (96.108.43.106)  1.868 ms  1.855 ms  1.854 ms
9  po-1-rur202.albuquerque.nm.albuq.comcast.net (68.85.65.122)  1.811 ms  1.696 ms  1.734 ms
10 96.216.21.222 (96.216.21.222)  1.797 ms  1.821 ms  1.788 ms
11 * * *
12 * * *
13 * * *
```

```
[root@moonshine ~]# yum install net-tools
```

...

```
[root@localhost ~]# route
```

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
default	summit.carc.unm	0.0.0.0	UG	100	0	0	eno1
129.24.244.0	0.0.0.0	255.255.252.0	U	100	0	0	eno1

Dynamic Host Configuration Protocol



DHCP Server 1



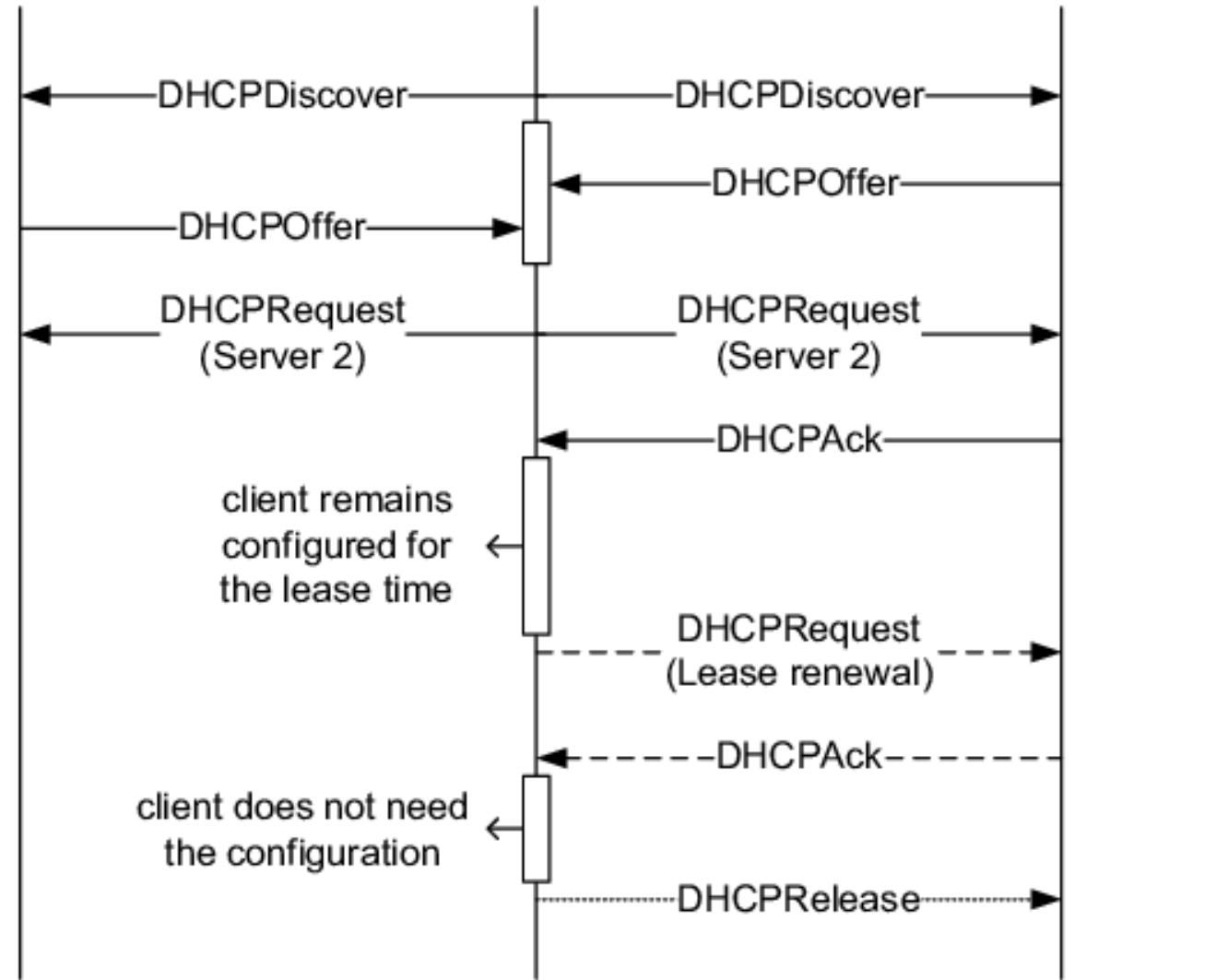
DHCP Client



DHCP Server 2

When you have many hosts to configure it is useful to configure them all in one place.

```
subnet 192.168.1.0 netmask 255.255.255.0 {  
    option routers      192.168.1.254;  
    option subnet-mask  255.255.255.0;  
    option domain-search "example.com";  
    option domain-name-servers 192.168.1.1;  
    option time-offset  -18000;  # Eastern  
Standard Time  
    range 192.168.1.10 192.168.1.100;  
}
```



Filter: dhcp Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
1	18:48:10.091515000	169.254.247.225	169.254.255.255	BROWSE	219	Request Announcement MAX-PC
2	18:48:10.091669000	169.254.247.225	169.254.255.255	BROWSE	219	Request Announcement MAX-PC
3	18:48:10.092272000	169.254.247.225	169.254.255.255	BROWSE	249	Domain/workgroup Announcement WORKGROUP, NT workstation, Domain Enum
4	18:48:11.140433000	fe80::4195:59f3:544	ff02::c	SSDP	208	M-SEARCH * HTTP/1.1
5	18:48:13.413919000	Cisco-Li_52:a6:33	Broadcast	ARP	42	who has 192.168.1.110? Tell 192.168.1.1
6	18:48:14.438379000	Cisco-Li_52:a6:33	Broadcast	ARP	42	who has 192.168.1.110? Tell 192.168.1.1
7	18:48:15.162495000	fe80::4195:59f3:544	ff02::c	SSDP	208	M-SEARCH * HTTP/1.1
8	18:48:15.359441000	Cisco-Li_52:a6:33	Broadcast	ARP	42	who has 192.168.1.110? Tell 192.168.1.1
9	18:48:16.329862000	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x6fd4f5bb
10	18:48:16.406126000	Cisco-Li_52:a6:33	Broadcast	ARP	60	who has 192.168.1.110? Tell 192.168.1.1
11	18:48:17.105542000	192.168.1.1	255.255.255.255	DHCP	590	DHCP offer - Transaction ID 0x6fd4f5bb
12	18:48:17.106124000	0.0.0.0	255.255.255.255	DHCP	350	DHCP Request - Transaction ID 0x6fd4f5bb
13	18:48:17.210337000	192.168.1.1	255.255.255.255	DHCP	590	DHCP ACK - Transaction ID 0x6fd4f5bb
14	18:48:17.227107000	fe80::4195:59f3:544	ff02::16	ICMPv6	90	Multicast Listener Report Message v2
15	18:48:17.310687000	192.168.1.110	75.75.75.75	DNS	85	Standard query 0x633f A teredo.ipv6.microsoft.com
16	18:48:17.318649000	192.168.1.110	192.168.1.255	NRNS	110	Registration NR MAX-PC<00>

Frame 11: 590 bytes on wire (4720 bits), 590 bytes captured (4720 bits) on interface 0
Ethernet II, Src: Cisco-Li_52:a6:33 (00:0f:66:52:a6:33), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 192.168.1.1 (192.168.1.1), Dst: 255.255.255.255 (255.255.255.255)
User Datagram Protocol, Src Port: bootps (67), Dst Port: bootpc (68)
Source port: bootps (67)
Destination port: bootpc (68)
Length: 556
Checksum: 0xdb3c [validation disabled]
Bootstrap Protocol

0000	ff ff ff ff ff ff 00 0f	66 52 a6 33 08 00 45 00	fR.3..E.
0010	02 40 00 00 00 00 40 11	b7 04 c0 a8 01 01 ff ff	..@.....
0020	ff ff 00 43 00 44 02 2c	db 3c 02 01 06 00 6f d4	...C.D.,.<...o.
0030	f5 bb 00 00 00 00 00 00	00 00 c0 a8 01 6e 00 00	n..
0040	00 00 00 00 00 00 cc af	78 0a de 6b 00 00 00 00	x..k....
0050	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	

Packet Sniffing

In one terminal connected to your cluster run

```
sudo yum install wireshark-cli  
watch wget http://fricke.co.uk
```

In another we will look at ARP traffic

```
sudo tshark -i <network interface> -f "arp"
```

What's this?

```
sudo tshark -i <network interface> -f "port 22"
```

Packet Sniffing

In one terminal connected to your cluster run

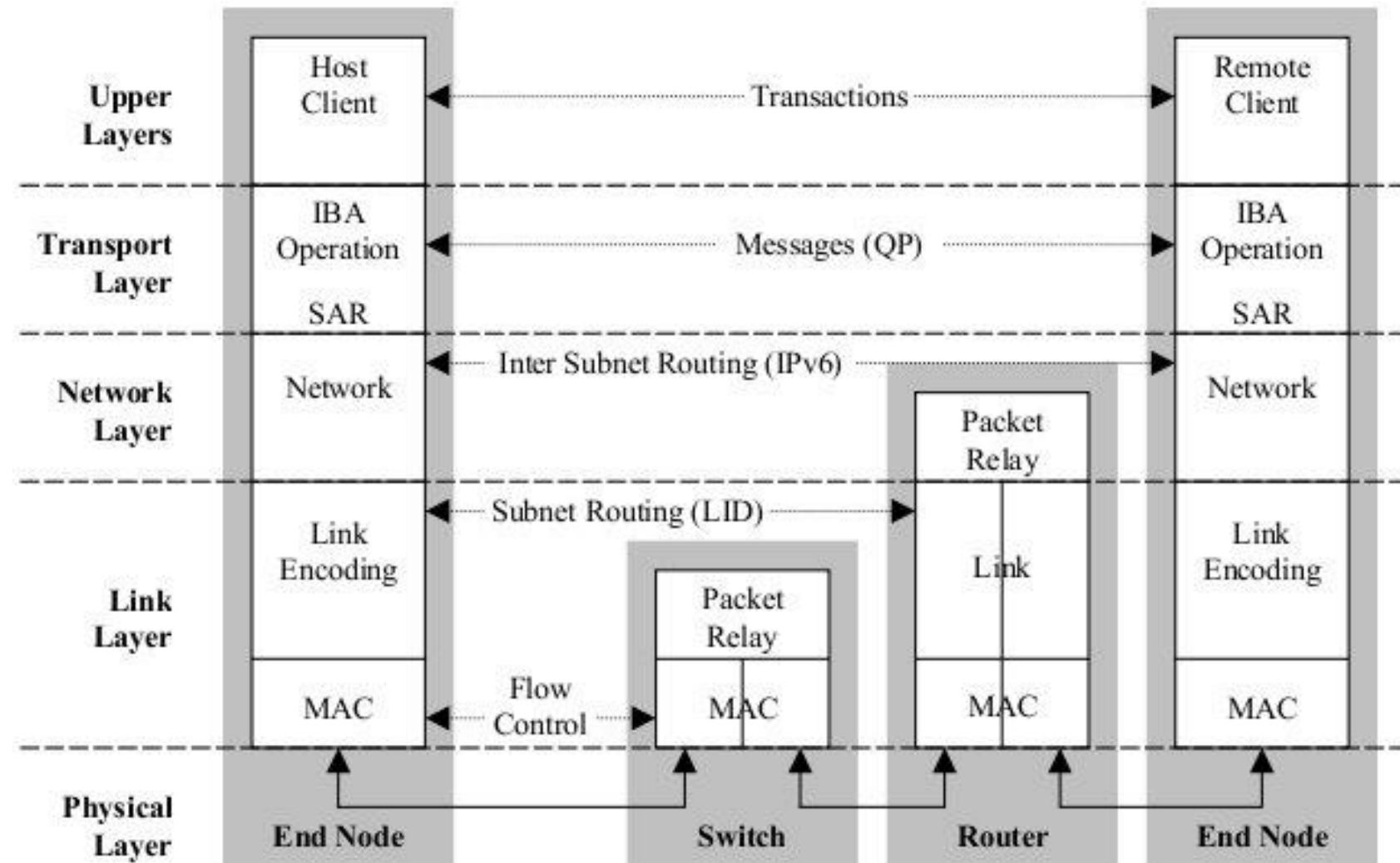
```
watch wget http://fricke.co.uk
```

In another

```
sudo tshark -V --color -i <interface name> -f "host fricke.co.uk" -f "port 80" -x
```

Infiniband uses RDMA based Communication

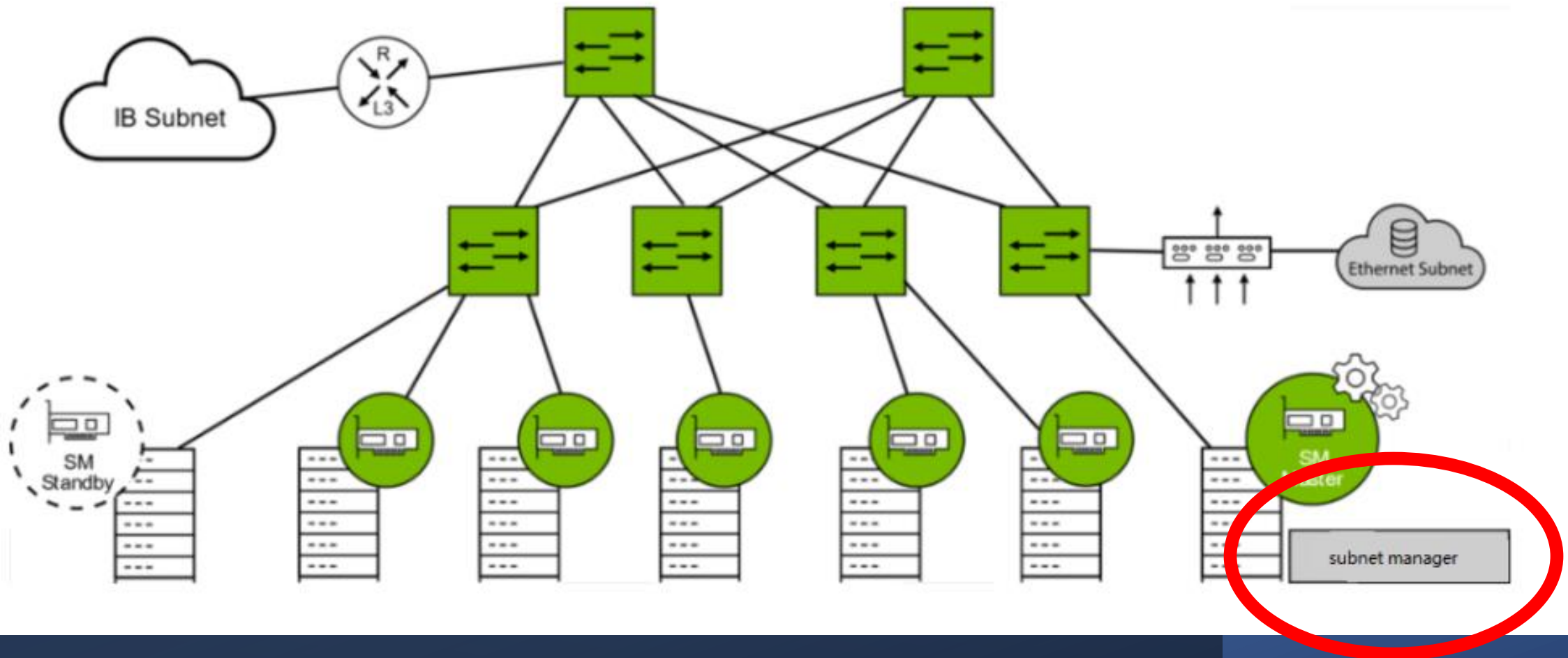
1999



Everything in the OSI model is handled by the kernel.
Infiniband tries to bypass the Kernel and even the CPU to increase performance.
DMA is Direct Memory Access. (Bypass CPU and kernel, Remote Direct Memory Access)

Ethernet broadcasts data and switches learn routes.

Infiniband uses a centralized subnet manager (opensm for example) that learns the routes and tells all the clients.



mfricke@hopper:~ \$ systemctl status opensm

- opensm.service - Starts the OpenSM InfiniBand fabric Subnet Manager
Loaded: loaded (/usr/lib/systemd/system/opensm.service; enabled; vendor pres
Active: active (running) since Wed 2023-12-20 09:07:30 MST; 1 months 20 days
Docs: man:opensm
Main PID: 1373 (opensm-launch)
Tasks: 72 (limit: 605216)
Memory: 11.9M
CGroup: /system.slice/opensm.service
└─ 1373 /bin/bash /usr/libexec/opensm-launch
└─ 10819 /usr/sbin/opensm

```
[root@moonshine ~]# dig +add +trace @8.8.8.8 fricke.uk
```

```
; <<>> DiG 9.11.36-RedHat-9.11.36-14.el8_10 <<>> +add +trace @8.8.8.8 fricke.uk
```

```
; (1 server found)
```

```
;; global options: +cmd
```

```
<snip>
```

```
. 87203 IN NS l.root-servers.net.
```

```
. 87203 IN NS g.root-servers.net.
```

```
<snip>
```

```
;; Received 525 bytes from 8.8.8.8#53(8.8.8.8) in 34 ms
```

```
<snip>
```

```
uk. 172800 IN NS dns3.nic.uk.
```

```
uk. 172800 IN NS dns1.nic.uk.
```

```
uk. 172800 IN NS dns2.nic.uk.
```

```
uk. 86400 IN DS 43876 8 2 A107ED2AC1BD14D924173BC7E827A1153582072394F9272BA37E2353 BC659603
```

```
uk. 86400 IN RRSIG DS 8 1 86400 20241004170000 20240921160000 20038 .
```

```
<snip>
```

```
;; Received 907 bytes from 192.112.36.4#53(g.root-servers.net) in 64 ms
```

```
<snip>
```

```
fricke.uk. 172800 IN NS fricke.uk.
```

```
fricke.uk. 172800 IN NS fricke.co.uk.
```

```
;; Received 639 bytes from 156.154.103.3#53(nsd.nic.uk) in 34 ms
```

```
<snip>
```

```
fricke.uk. 604800 IN A 50.193.238.94
```

```
fricke.uk. 604800 IN NS fricke.uk.
```

```
;; Received 96 bytes from 50.193.238.94#53(fricke.co.uk) in 12 ms
```

Diagnosing DNS issues by tracing
the sequence of DNS calls
DNS servers listen on port **53**

Comparison of latency

Infiniband uses a credit strategy instead of CDMA/CD.

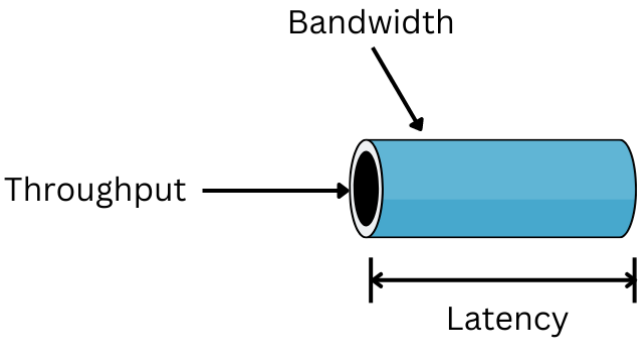
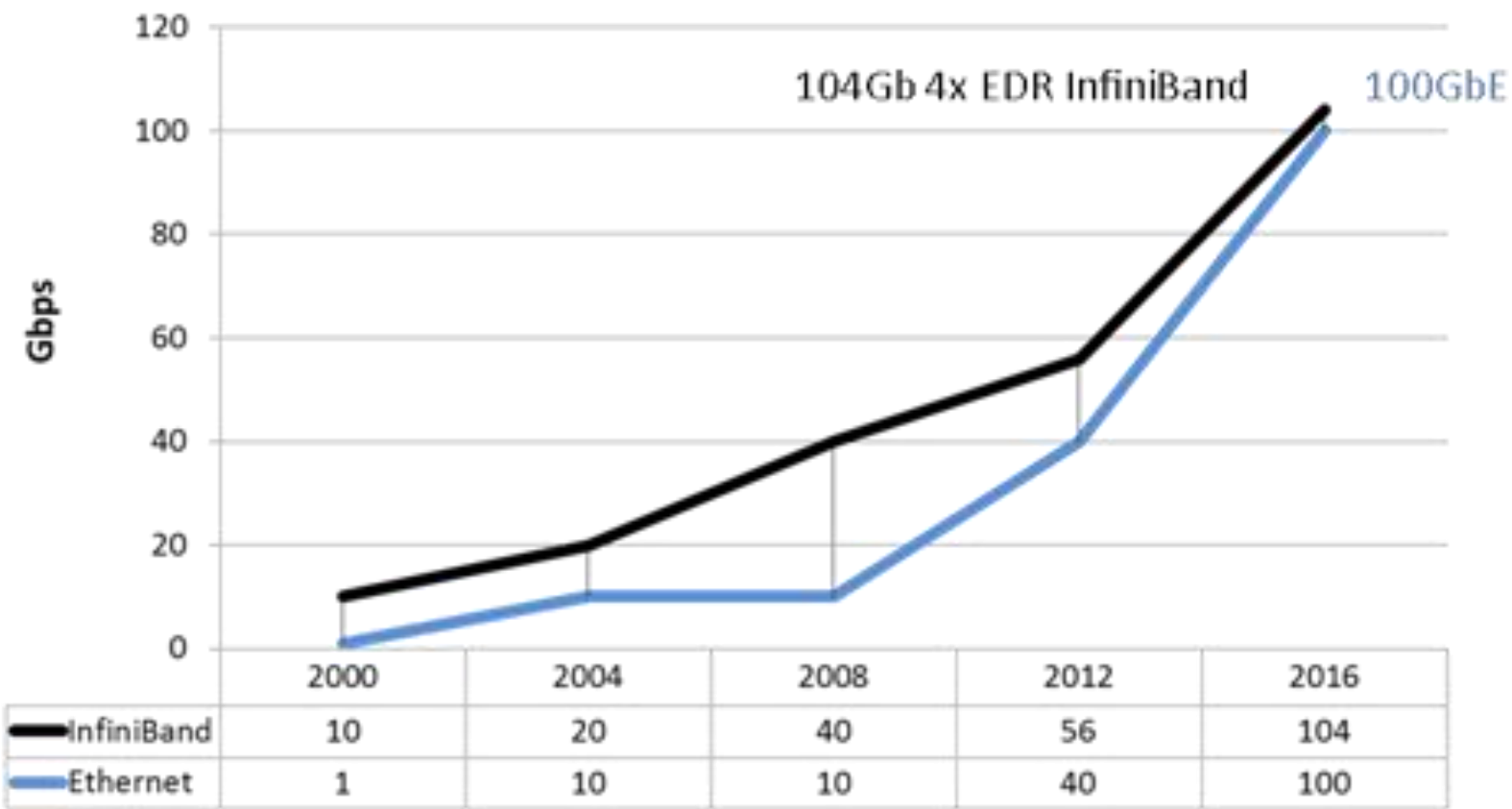
Each receivers “credit” is the amount of buffer space they have to receive data.

Only receivers who can actually receive all the data to be transmitted are allowed to receive.

With ethernet a switch stores the whole message incase it needs to retransmit.

Infiniband switches don’t need to receive the whole message before sending it on to the destination host.

Bandwidth Performance Gap



Latency is the delay between sending and receiving.

Throughput is how much data gets transmitted in a time period

Comparison of latency

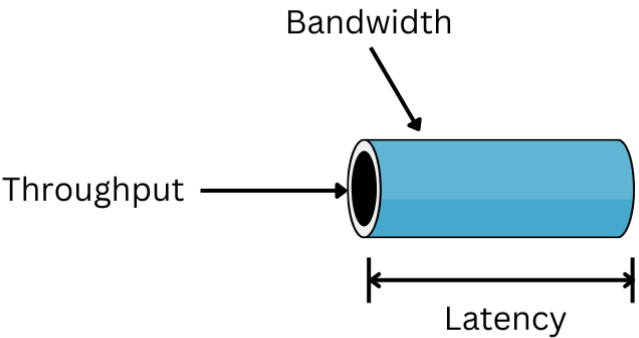
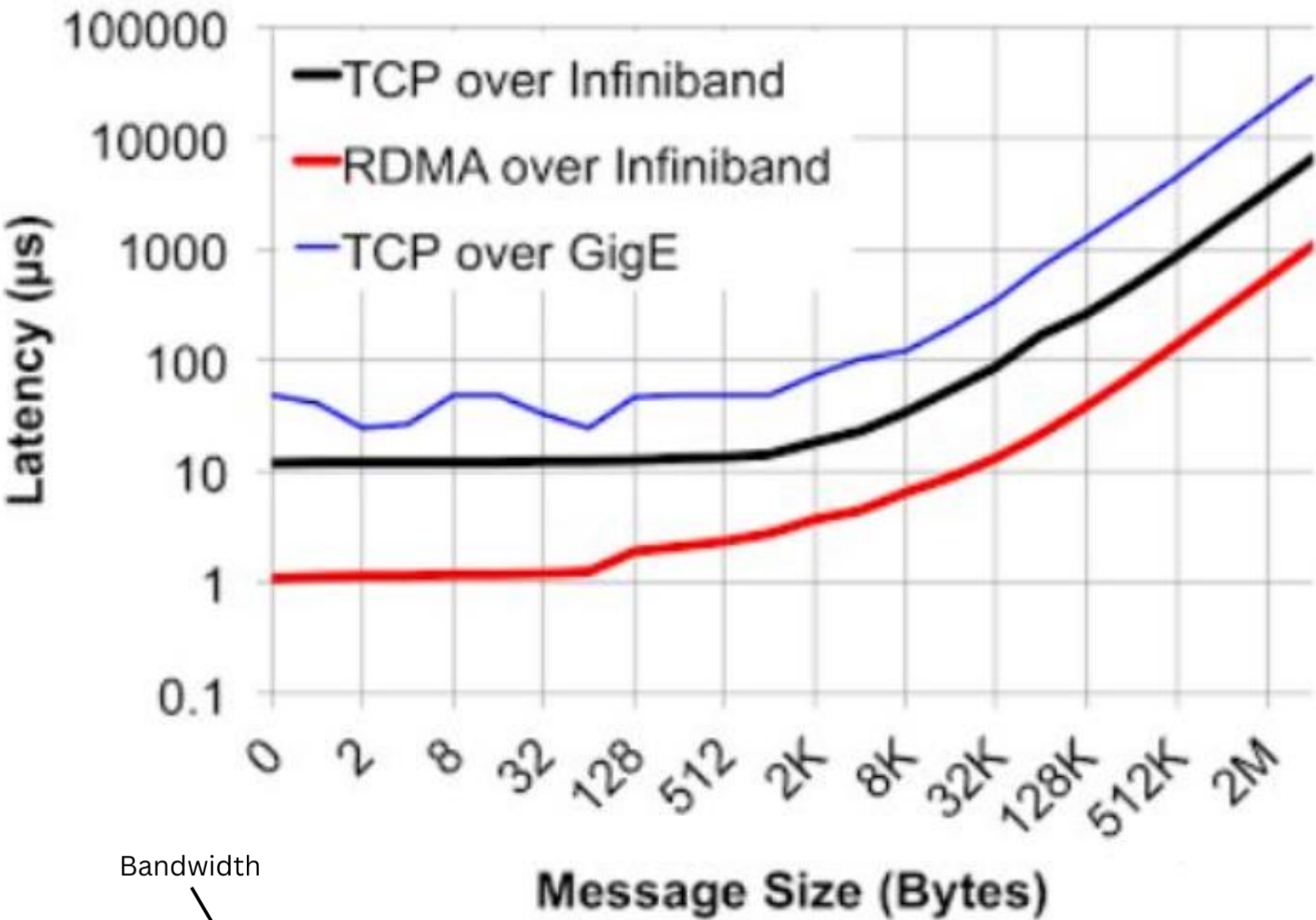
Infiniband uses a credit strategy instead of CDMA/CD.

Each receivers “credit” is the amount of buffer space they have to receive data.

Only receivers who can actually receive all the data to be transmitted are allowed to receive.

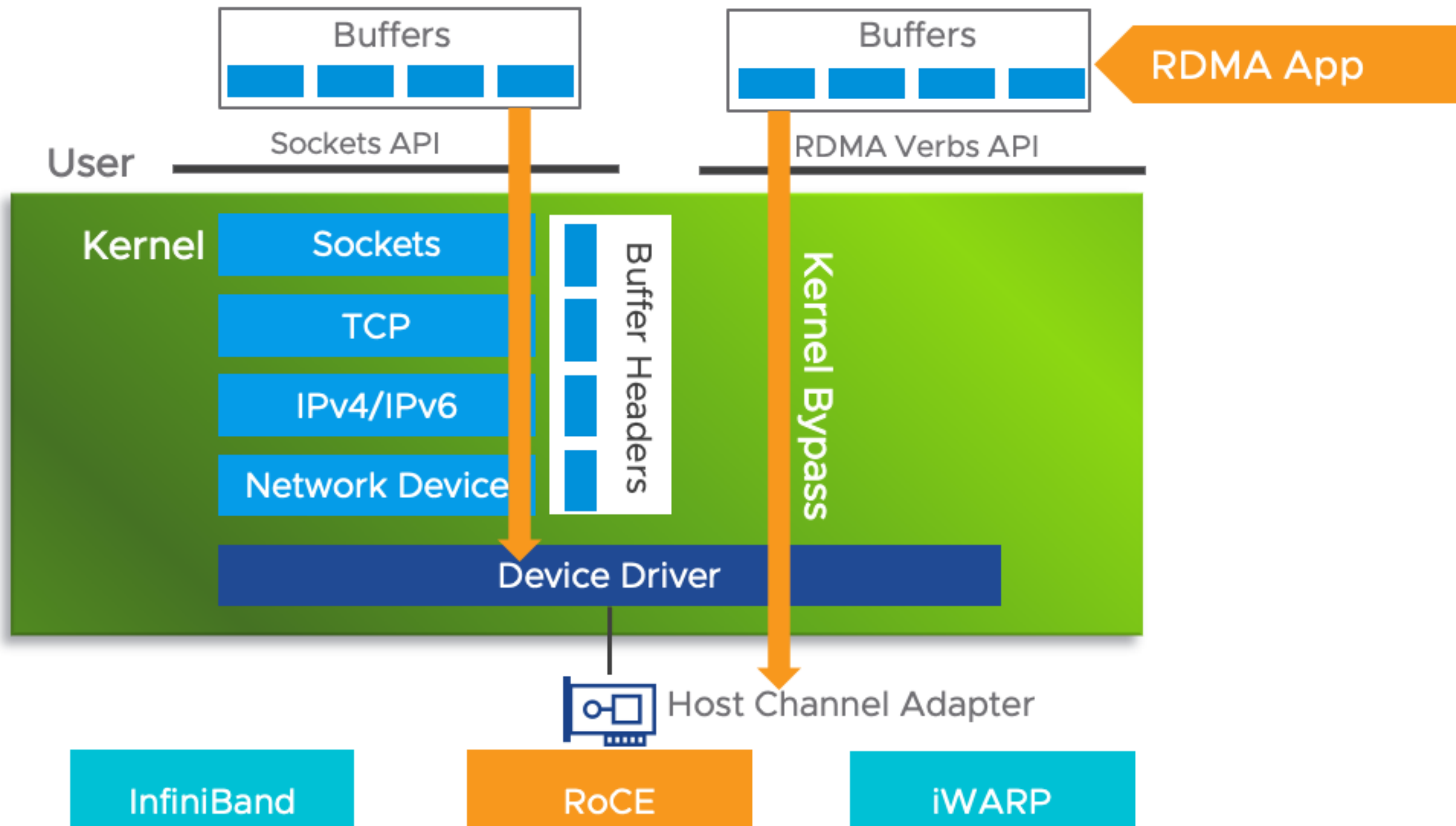
With ethernet a switch stores the whole message incase it needs to retransmit.

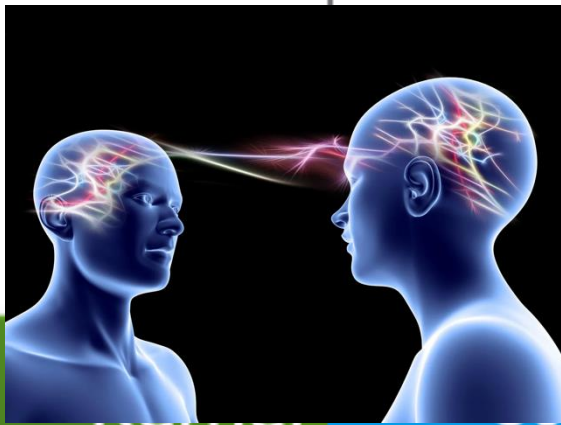
Infiniband switches don’t need to receive the whole message before sending it on to the destination host.



Latency is the delay between sending and receiving.

Throughput is how much data gets transmitted in a time period





uffers

ts API

ck

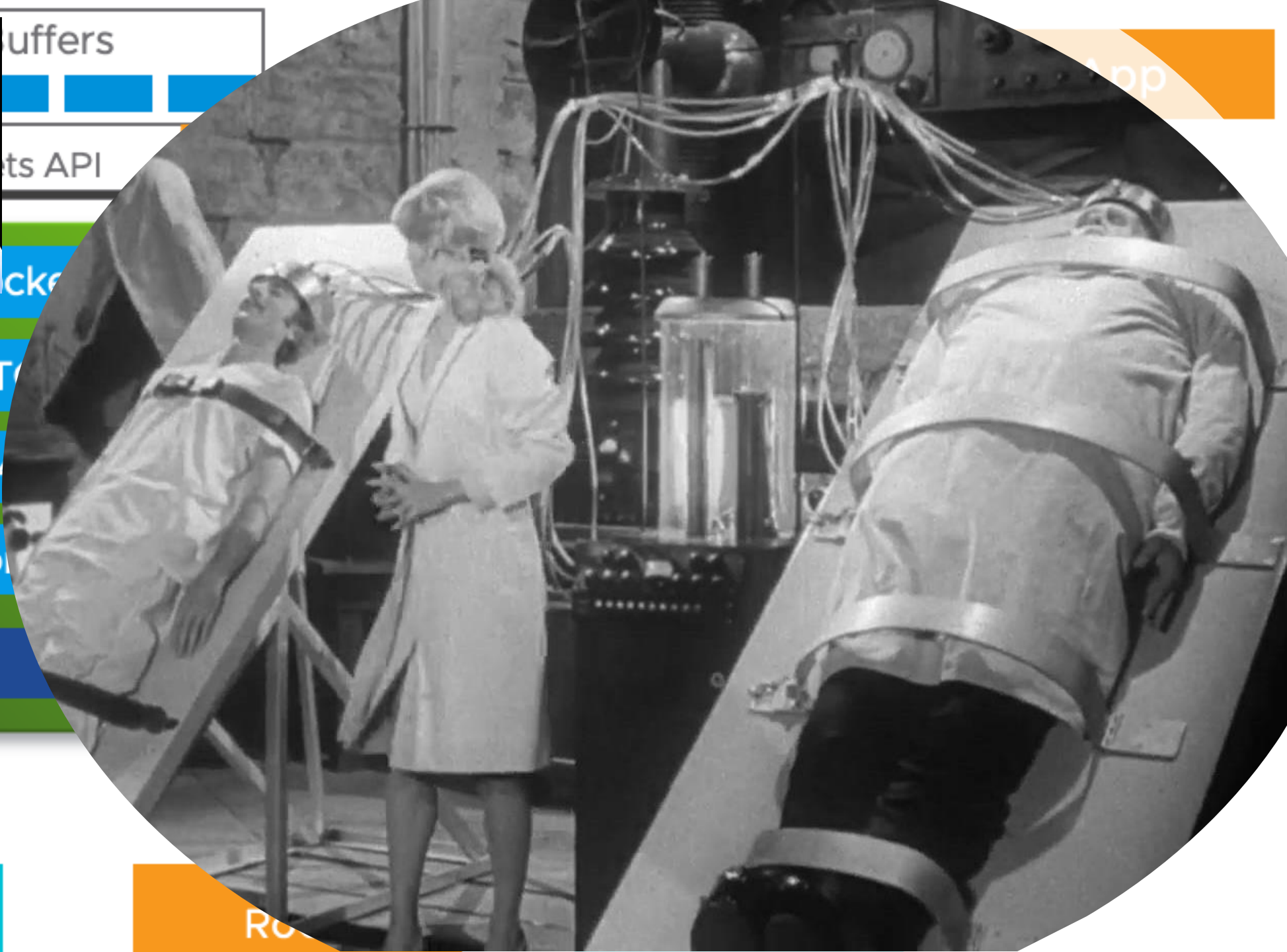
T

IPv

Netwo

InfiniBand

Ro



App

With Rocky 9.3 Connect-X 3 Inifniband Cards need the driver to be installed after install

```
[matthew@moonshine ~]# sudo yum install rdma-core libibverbs-utils librdmacm librdmacm-utils ibacm infiniband-diags opensm
```

Dependencies resolved.

=====				
Package	Architecture	Version	Repository	Size
=====				

Installing:

ibacm	x86_64	46.0-1.el9	baseos	88 k
infiniband-diags	x86_64	46.0-1.el9	appstream	310 k
libibverbs-utils	x86_64	46.0-1.el9	baseos	67 k
librdmacm	x86_64	46.0-1.el9	baseos	70 k
librdmacm-utils	x86_64	46.0-1.el9	baseos	90 k
opensm	x86_64	3.3.24-2.el9	baseos	503 k
rdma-core	x86_64	46.0-1.el9	baseos	51 k

Installing dependencies:

libibumad	x86_64	46.0-1.el9	baseos	26 k
opensm-libs	x86_64	3.3.24-2.el9	baseos	75 k
pciutils	x86_64	3.7.0-5.el9	baseos	92 k

```
[matthew@moonshine ~]$ sudo reboot now
```

```
Connection to 129.24.245.16 closed by remote host.
```

```
matthew@lycaon ~ % ssh matthew@129.24.245.16
matthew@129.24.245.16's password:
Last login: Tue Feb  6 20:07:00 2024 from 50.193.238.94
[matthew@localhost ~]$ ibstat
CA 'mlx4_0'
CA type: MT4099
Number of ports: 1
Firmware version: 2.10.700
Hardware version: 0
Node GUID: 0x0002c90300f67b70
System image GUID: 0x0002c90300f67b73
Port 1:
State: Initializing
Physical state: LinkUp
Rate: 40
Base lid: 0
LMC: 0
SM lid: 0
Capability mask: 0x02594868
```

[matthew@localhost ~]\$ ip a

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eno1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether d4:ae:52:8b:72:8c brd ff:ff:ff:ff:ff:ff
    altname enp1s0f0
    inet 129.24.245.16/22 brd 129.24.247.255 scope global noprefixroute eno1
        valid_lft forever preferred_lft forever
    inet6 fe80::d6ae:52ff:fe8b:728c/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: eno2: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group default qlen 1000
    link/ether d4:ae:52:8b:72:8d brd ff:ff:ff:ff:ff:ff
    altname enp1s0f1
4: eno3: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group default qlen 1000
    link/ether d4:ae:52:8b:72:8e brd ff:ff:ff:ff:ff:ff
    altname enp2s0f0
5: eno4: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN group default qlen 1000
    link/ether d4:ae:52:8b:72:8f brd ff:ff:ff:ff:ff:ff
    altname enp2s0f1
6: ibp65s0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 4092 qdisc fq_codel state DOWN group default qlen 256
    link/infiniband 80:00:02:08:fe:80:00:00:00:00:00:00:00:02:c9:03:00:f6:7b:71 brd 00:ff:ff:ff:ff:12:40:1b:ff:ff:00:00:00:00:00:00:ff:ff:ff:ff
```

```
[matthew@moonshine ~]$ sudo status opensm
```

- opensm.service - Starts the OpenSM InfiniBand fabric Subnet Manager
 - Loaded: loaded (/usr/lib/systemd/system/opensm.service; disabled; preset: disabled)
 - Active: active (running) since Fri 2024-10-18 21:13:31 CDT; 5s ago
 - Docs: man:opensm
 - Process: 235961 ExecStart=/usr/libexec/opensm-launch (code=exited, status=0/SUCCESS)
 - Main PID: 235962 (opensm-launch)
 - Tasks: 40 (limit: 407887)
 - Memory: 7.9M
 - CPU: 80ms
 - CGroup: /system.slice/opensm.service
 - └─235962 /usr/bin/bash /usr/libexec/opensm-launch
 - └─235963 /usr/sbin/opensm

Testing Infiniband

Make sure the
opensm Service is
running

```
un[235963]: Log File: /var/log/opensm.log
un[235963]: -----
35963]: /var/log/opensm.log log file opened
35963]: OpenSM 3.3.24
un[235963]: OpenSM 3.3.24
un[235963]: Using default GUID 0x2c90300f67b91
35963]: Entering DISCOVERING state
un[235963]: Entering DISCOVERING state
35963]: Entering MASTER state
un[235963]: Entering MASTER state
```

```
[matthew@moonshine ~]$ sudo status opensm
```

- opensm.service - Starts the OpenSM InfiniBand fabric Subnet Manager
 - Loaded: loaded (/usr/lib/systemd/system/opensm.service; disabled; preset: disabled)
 - Active: active (running) since Fri 2024-10-18 21:13:31 CDT; 5s ago
 - Docs: man:opensm
 - Process: 235961 ExecStart=/usr/libexec/opensm-launch (code=exited, status=0/SUCCESS)
 - Main PID: 235962 (opensm-launch)
 - Tasks: 40 (limit: 407887)
 - Memory: 7.9M
 - CPU: 80ms
 - CGroup: /system.slice/opensm.service
 - └─235962 /usr/bin/bash /usr/libexec/opensm-launch
 - └─235963 /usr/sbin/opensm

Testing Infiniband

Make sure the
opensm Service is
running

```
un[235963]: Log File: /var/log/opensm.log
un[235963]: -----
35963]: /var/log/opensm.log log file opened
35963]: OpenSM 3.3.24
un[235963]: OpenSM 3.3.24
un[235963]: Using default GUID 0x2c90300f67b91
35963]: Entering DISCOVERING state
un[235963]: Entering DISCOVERING state
35963]: Entering MASTER state
un[235963]: Entering MASTER state
```

```
[matthew@moonshine ~]$ ibstat  
CA 'mlx4_0'  
CA type: MT4099  
Number of ports: 1  
Firmware version: 2.10.700  
Hardware version: 0  
Node GUID: 0x0002c90300f67b90  
System image GUID: 0x0002c90300f67b93  
Port 1:  
State: Active  
Physical state: LinkUp  
Rate: 40  
Base lid: 1  
LMC: 0  
SM lid: 1  
Capability mask: 0x0259486a  
Port GUID: 0x0002c90300f67b91  
Link layer: InfiniBand
```

Testing Infiniband

Check that infiniband is
connected to the opensm
server

SM ID should be non-zero

```
[matthew@moonshine ~]$ ibstat
CA 'mlx4_0'
CA type: MT4099
Number of ports: 1
Firmware version: 2.10.700
Hardware version: 0
Node GUID: 0x0002c90300f67b90
System image GUID: 0x0002c90300f67b93
Port 1:
State: Active
Physical state: LinkUp
Rate: 40
Base lid: 1
LMC: 0
SM lid: 1
Capability mask: 0x0259486a
Port GUID: 0x0002c90300f67b91
Link layer: InfiniBand
```

Testing Infiniband

Note the GUID address of the infiniband interface.

We will ping our own connections (since we only have one server right now)

```
[matthew@moonshine ~]$ ibping -S
```

Testing Infiniband

Infiniband ping requires a server and client.

```
[matthew@moonshine ~]$ ibping -G 0x0002c90300f67b91
Pong from dhcp242.carc.unm.edu.(none) (Lid 1): time 0.027 ms
Pong from dhcp242.carc.unm.edu.(none) (Lid 1): time 0.034 ms
Pong from dhcp242.carc.unm.edu.(none) (Lid 1): time 0.034 ms
Pong from dhcp242.carc.unm.edu.(none) (Lid 1): time 0.032 ms
Pong from dhcp242.carc.unm.edu.(none) (Lid 1): time 0.032 ms
Pong from dhcp242.carc.unm.edu.(none) (Lid 1): time 0.032 ms
Pong from dhcp242.carc.unm.edu.(none) (Lid 1): time 0.034 ms
^C
--- dhcp242.carc.unm.edu.(none) (Lid 1) ibping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6955 ms
rtt min/avg/max = 0.027/0.032/0.034 ms
[root@dhcp242 ~]#
```

```
[matthew@moonshine ~]# sudo iblinkinfo
```

```
CA: Team9 mlx4_0:
```

```
0x0002c90300f67b51 4 1[ ] == ( 4X 10.0 Gbps Active/ LinkUp) ==> 2 12[ ] "MF0;switch-  
628cfa:IS5030/U1" ( )
```

```
CA: mordor mlx4_0:
```

```
0x0002c90300f67bb1 8 1[ ] == ( 4X 10.0 Gbps Active/ LinkUp) ==> 2 11[ ] "MF0;switch-  
628cfa:IS5030/U1" ( )
```

```
CA: localhost mlx4_0:
```

```
0x0002c90300f67ac1 9 1[ ] == ( 4X 10.0 Gbps Active/ LinkUp) ==> 2 10[ ] "MF0;switch-  
628cfa:IS5030/U1" ( )
```

```
CA: localhost mlx4_0:
```

```
0x0002c90300f67a21 3  
628cfa:IS5030/U1" ( )
```

```
CA: rockyrack mlx4_0:
```

```
0x0002c90300f67981 6  
628cfa:IS5030/U1" ( )
```

```
CA: localhost mlx4_0:
```

iblinkinfo reports link information in the InfiniBand “fabric”.

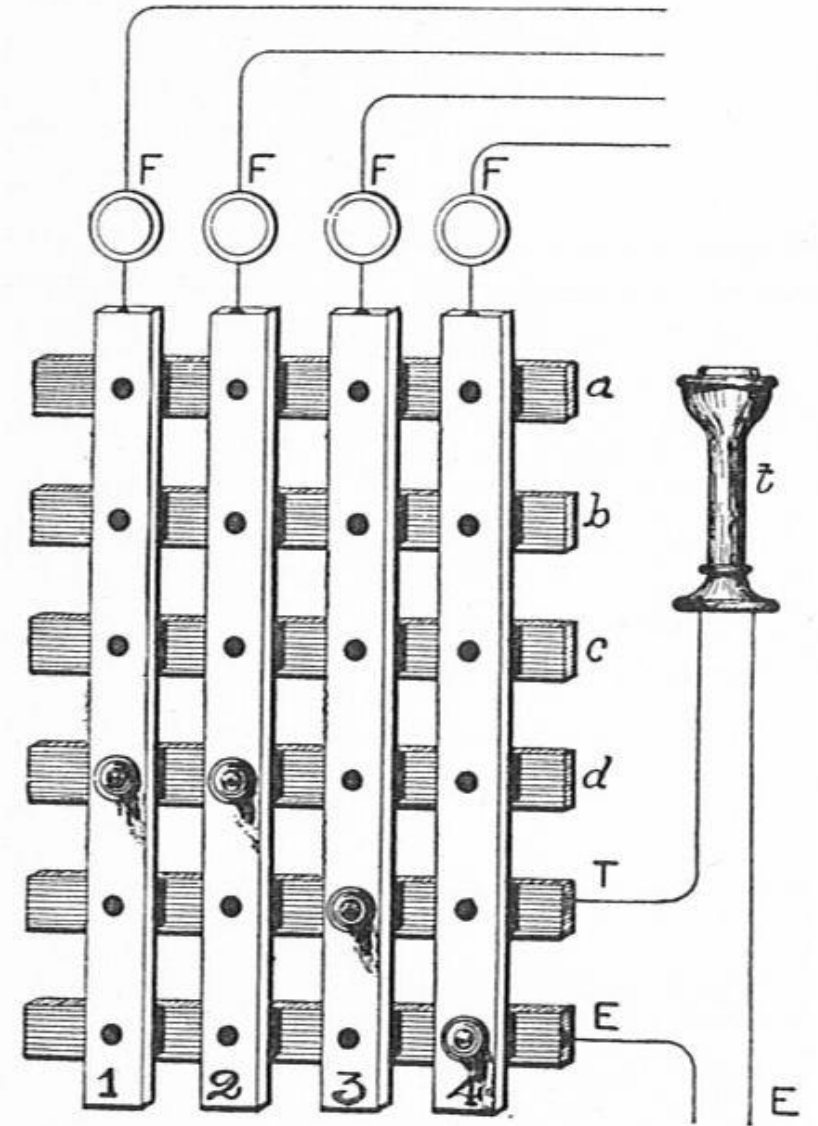
It scans the IB fabric node by node and displays link details for each port.

A fabric refers to a an HPC mesh like set of network links.

Ethernet, in contrast, is typically hierarchical (tree of switches)

Fabrics

- The term fabric is used a lot in HPC but people use it to mean different things. The hardware interconnect, the software libraries for the network, even the entire datacenter or cloud computing infrastructure.
- A fabric refers to a grid-like communication structure and has its origins in the old circuit switching mechanisms that had multiple-inputs and outputs in a grid design.
- E.g. InfiniBand connections have multiple redundant routes between hosts and are often visualized as a fabric.
- Make sure to check what is meant when someone says HPC “fabric”.



1903 Circuit Switching Fabric