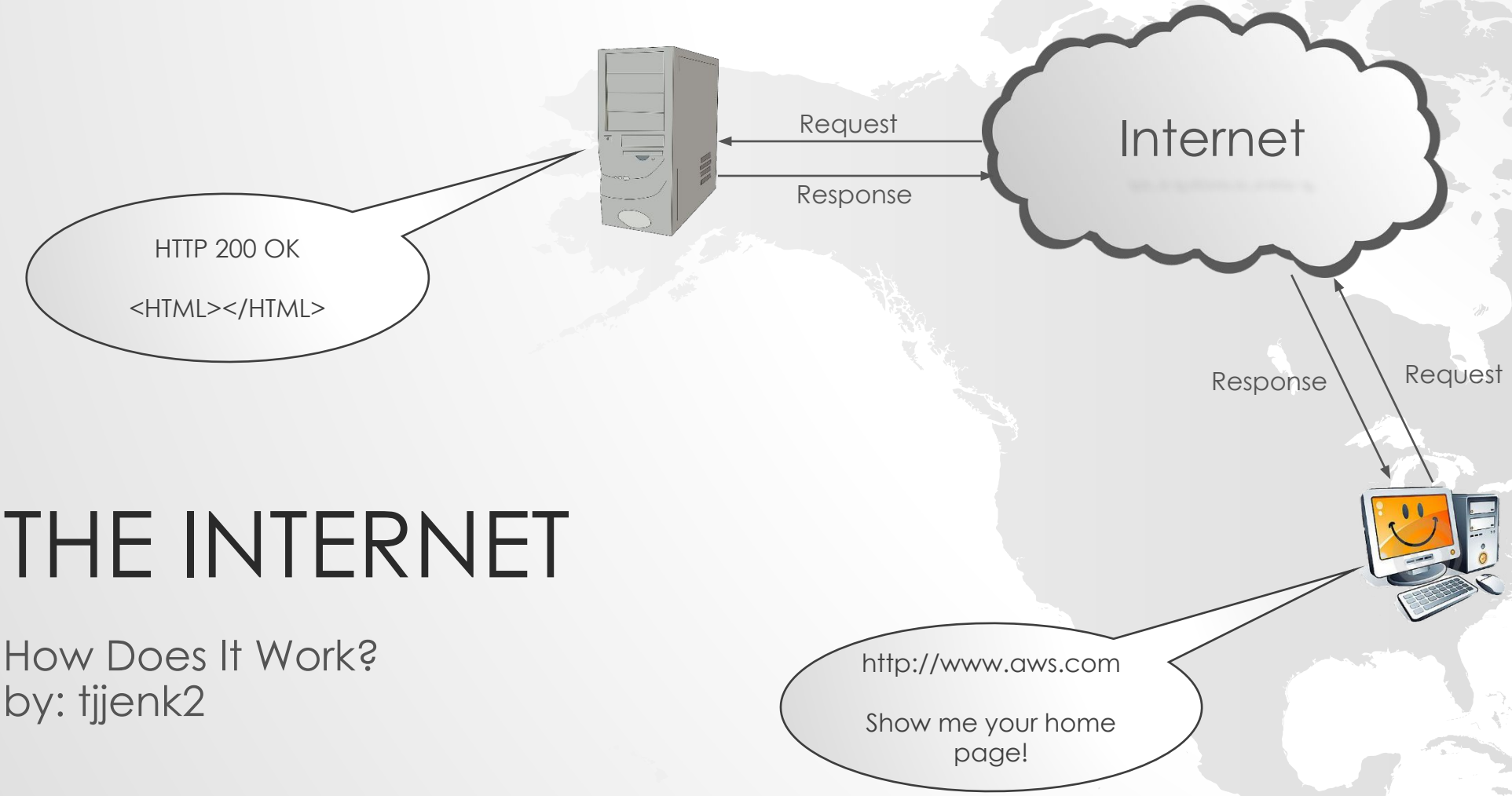
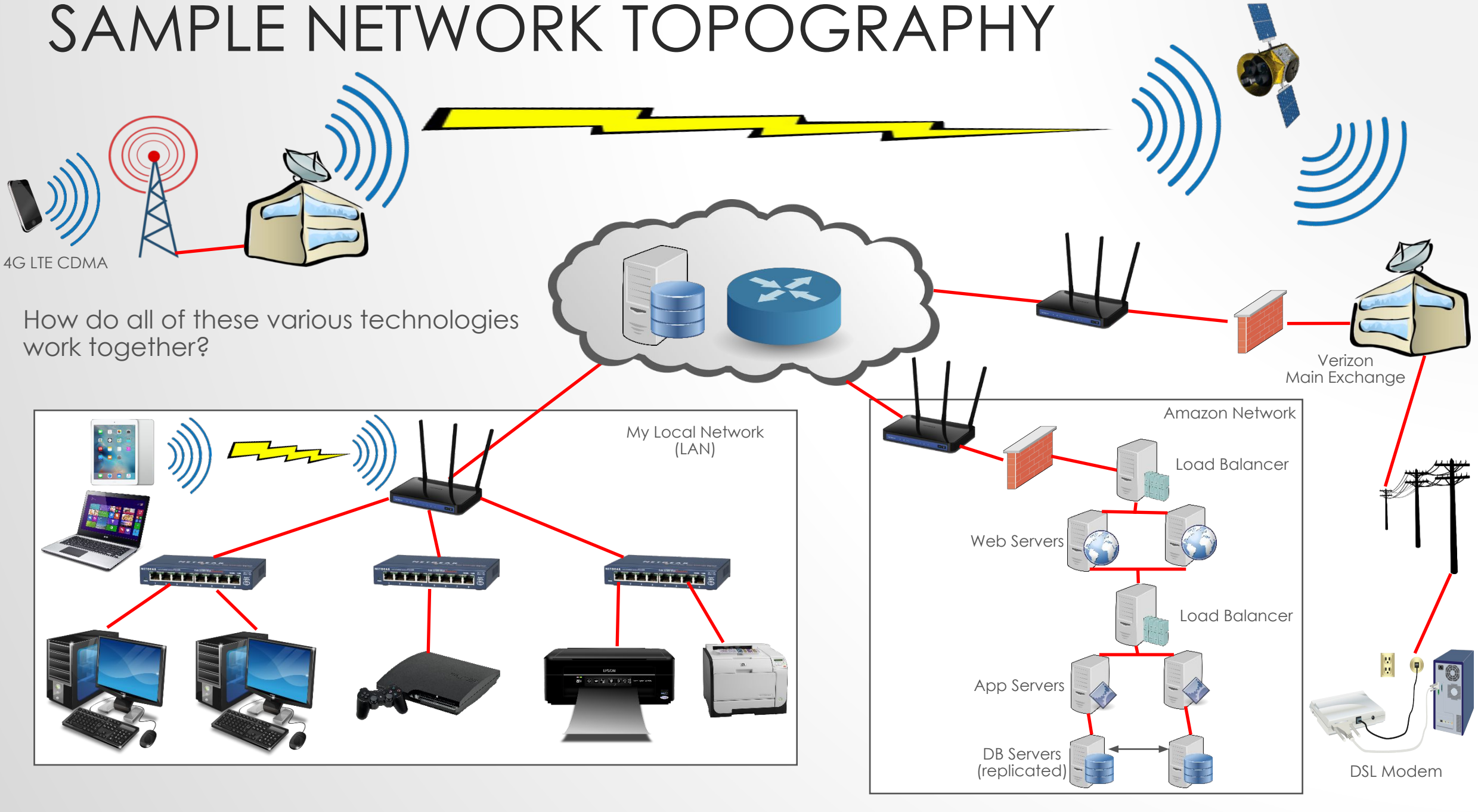
THE INTERNET

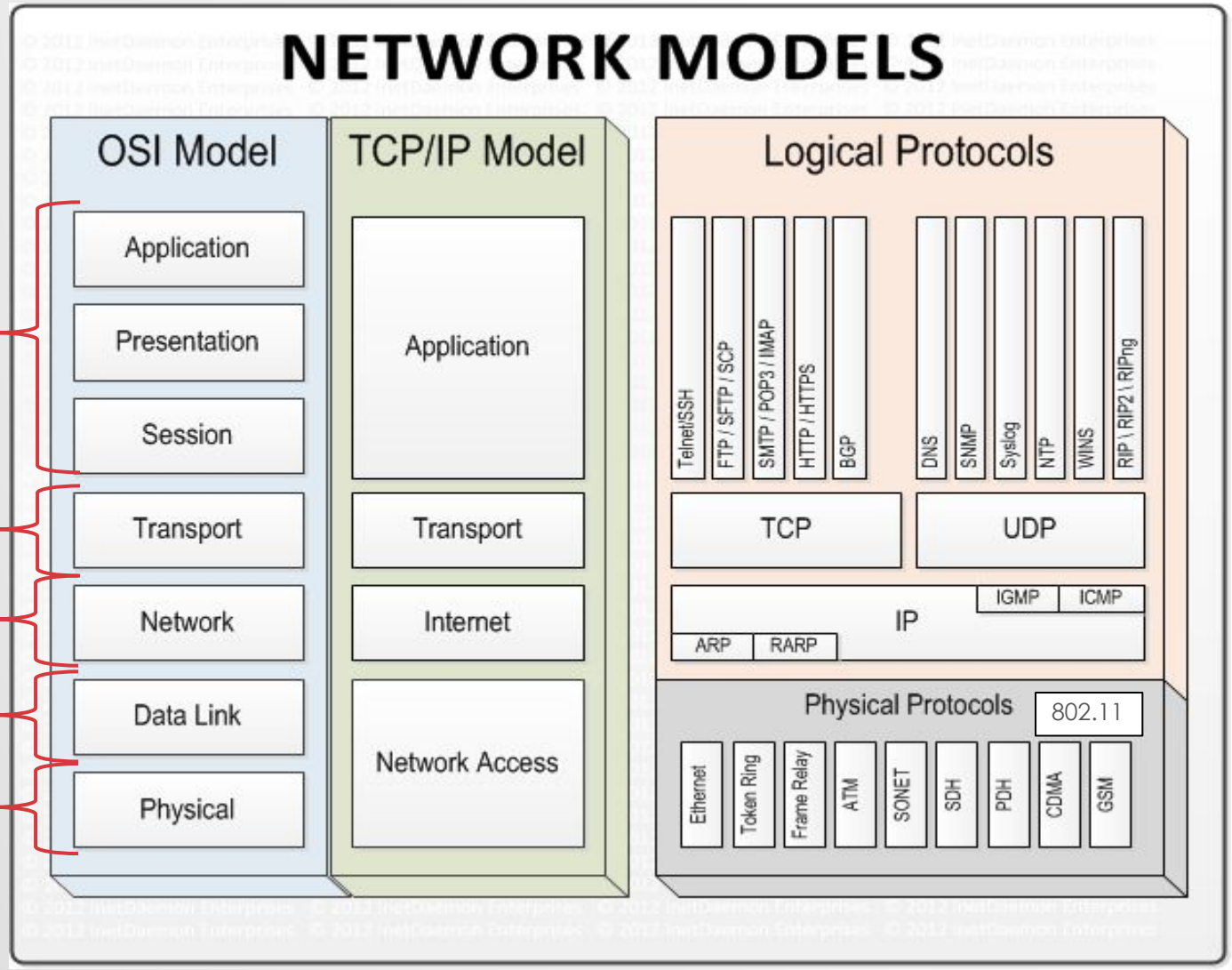
How Does It Work?
by: tjjenk2

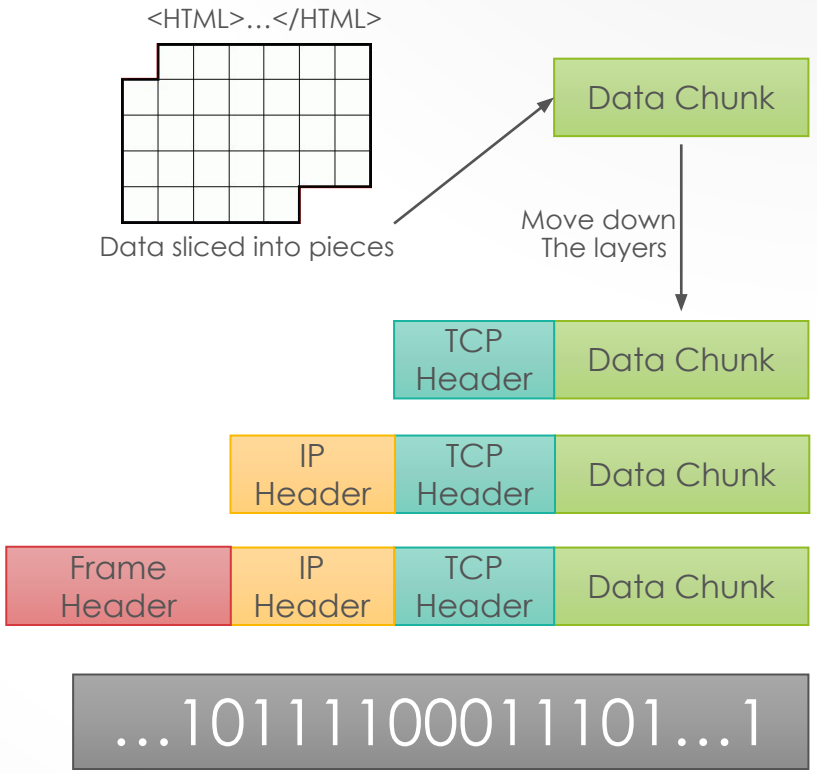


SAMPLE NETWORK TOPOGRAPHY



THE OSI MODEL

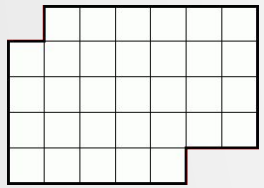




CONNECTING DIFFERENT SYSTEMS

Browser

HTTP GET /blah/



Request sliced into chunks

HTTP Request

HTTP Request | TCP Header

HTTP Request | TCP Header | IP Header

HTTP Request | TCP Header | IP Header | 802.11 Header

Move Down

Each Frame

...10111...

Layer	Protocol
App. Present. Session	HTTP
Transport	TCP
Network	IP
Data Link	802.11
Physical	WAP

IP	IP
802.11	Ethernet

Move Up

Move Down

HTTP Request | TCP Header | IP Header | Ethernet Header

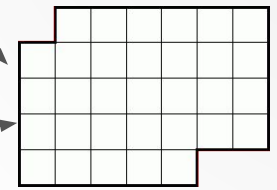
Layer	Protocol
App. Present. Session	HTTP
Transport	TCP
Network	IP
Data Link	Ethernet
Physical	Cat 5

HTTP Request | TCP Header | IP Header | Ethernet Header

...10111...

Put back together before handing to application

HTTP GET /blah/



Chunks appear in order

Web Server

HTTP Request

HTTP Request | TCP Header

HTTP Request | TCP Header | IP Header

HTTP Request | TCP Header | IP Header | Frame Header

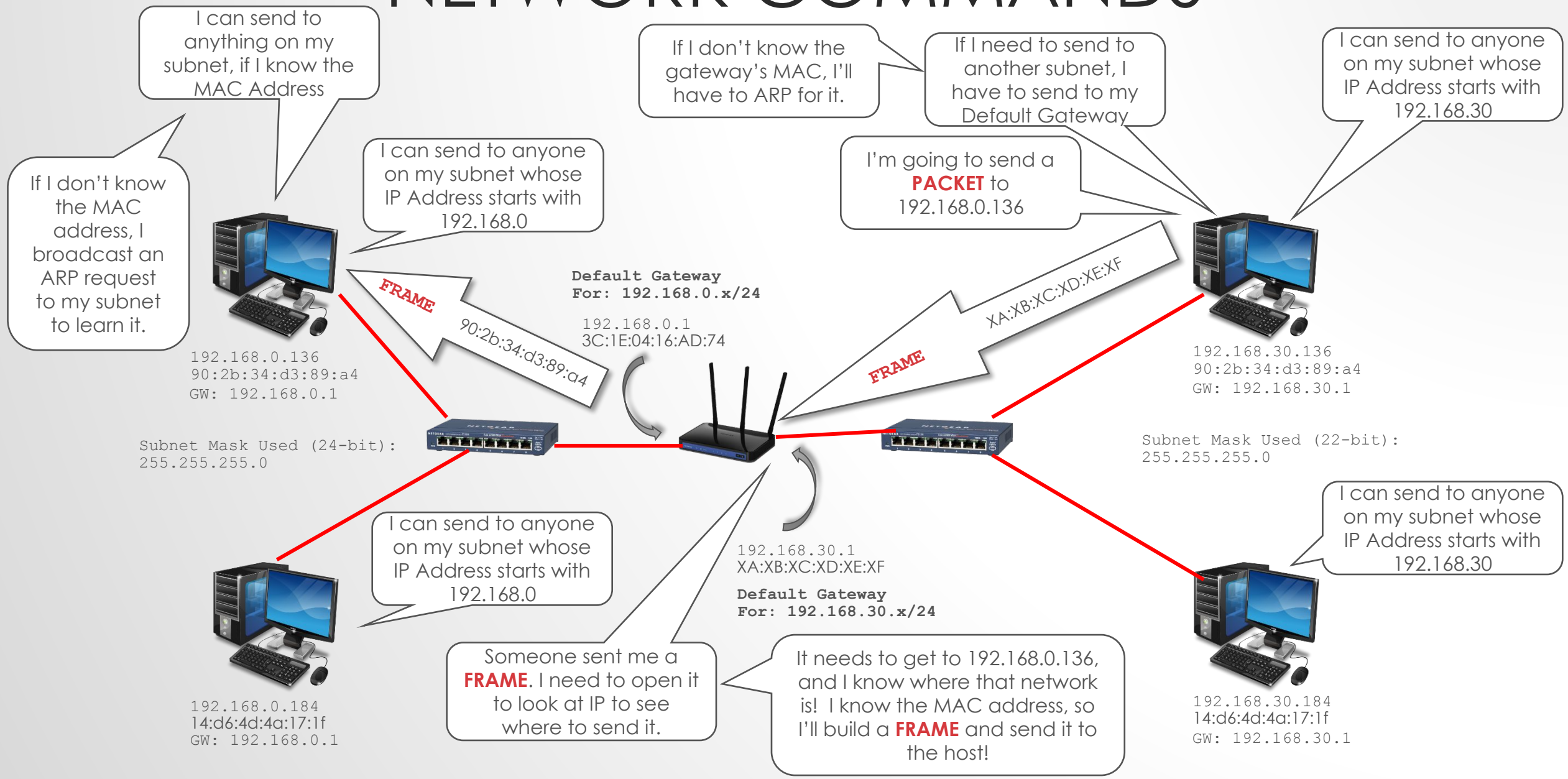
Move Up

TCP is reliable and guarantees ordered delivery

NETWORK COMMANDS

- Configure a network interface(or get information about one)
  - `Win:  ipconfig`
  - `Lin:  ifconfig –a (also shows your mac address)`
- Send ICMP ECHO_REQUEST to network hosts
  - `Win:  ping www.comcast.net`
  - `Lin:  ping -4 cdns01.comcast.net`
- Query Internet name servers interactively
  - `Win:  nslookup www.comcast.net`
  - `Lin:  nslookup www.comcast.net`
- Print the route packets trace to network host
  - `Win:  tracert www.comcast.net`
  - `Lin:  sudo traceroute -I www.comcast.net`
- Manipulate the system ARP cache (or get information about it)
  - `Win:  arp –a  (to get your mac:  getmac)`
  - `Lin:  arp     (to get your mac:  ifconfig –a; ethtool -P enp0s3)`
- Print network connections, routing tables, interface stats, etc.
  - `Win:  netstat`
  - `Lin:  netstat -an | grep LISTEN | grep 8081`

NETWORK COMMANDS

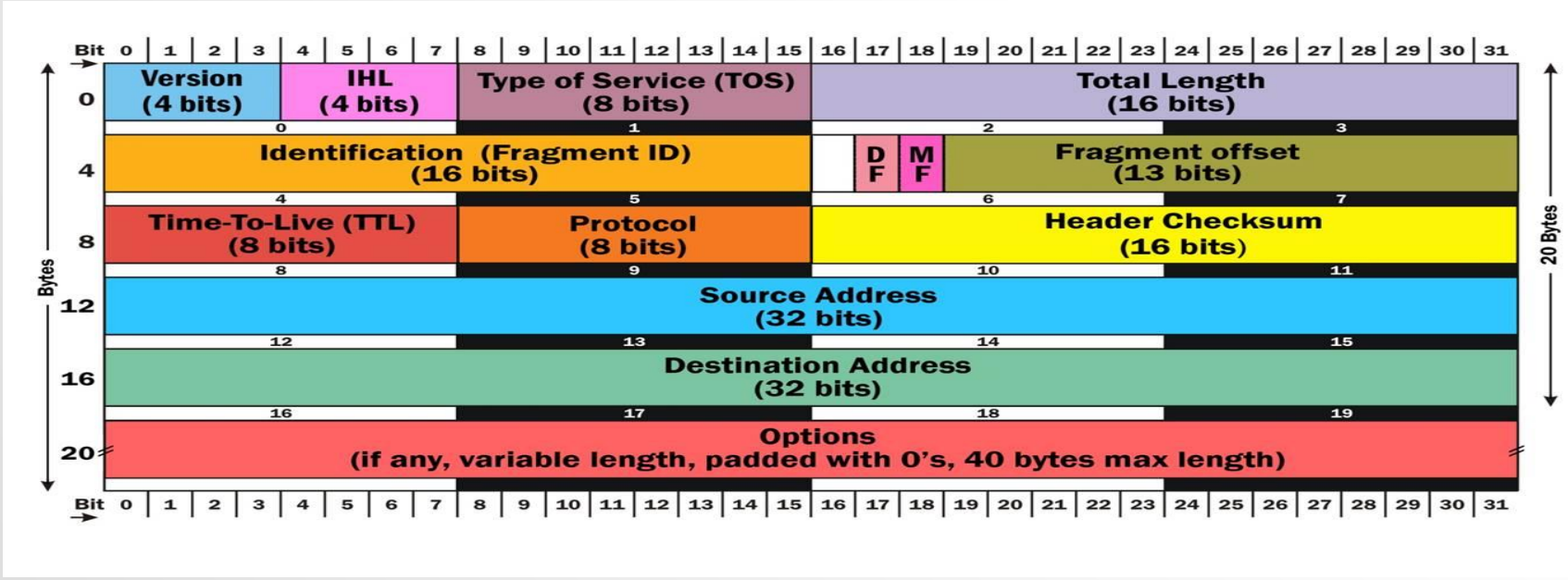


ETHERNET

- Is a Layer 2 Protocol – Data Link Layer
- Ethernet Frames need a destination MAC address
- If the destination MAC is in your ARP table, you can send the Frame
- If it isn't, you send a broadcast ARP request to find it
- If the destination host is on your subnet, you can send the frame directly
- If destination is on another subnet, you send the Frame to the Gateway's MAC address
- The Default Gateway is a router
- Routers make forwarding decisions based on layer 3 IP addresses
- Routers de-encapsulate the frame to see where they have to send it next
- Routers will send the frame to the next MAC address (it could be another router or the actual host

INTERNET PROTOCOL

- Is a Layer 3 Protocol – Network Layer
- Job is too send and route packets to other computers (hosts)
- IP routing is performed by all hosts, as well as routers
- Routers transport the packets across network boundaries
- Routers communicate with one another via routing protocols



TRANSMISSION CONTROL PROTOCOL (TCP)

- A layer 4 protocol – Transport Layer
- IT compliments the Internet Protocol (IP) – TCP/IP
- Reliable – ensures messages are delivered error-free and in order
- TCP is connection-oriented – must establish connection (3-way handshake)
- If segments are lost, it will retransmit them

TCP Header

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size														
16	128	Checksum																Urgent pointer (if URG set)															
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) ...																															

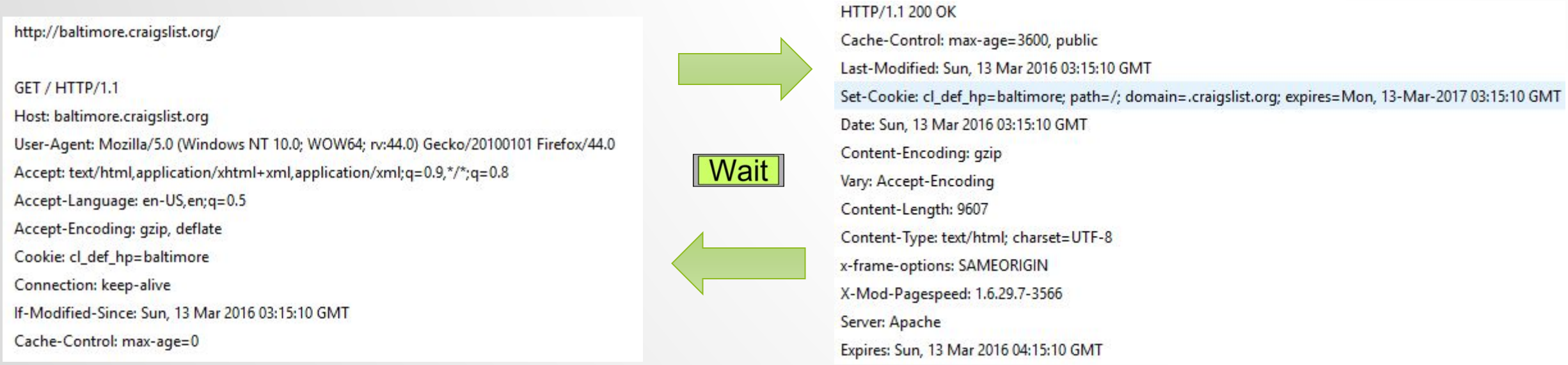
Well Known Ports	
HTTP	80
HTTPS	443
FTP	20/21
SMTP	25
QUAKE 3 Arena	27960

TCP CONNECTION

- Every connection includes 3-way handshake before sending data
  - SYN – Client sends SYN metadata
  - SYN, ACK – Server acknowledges the SYN with its metadata
  - ACK – The client acknowledges the server, then begins to make request
- Sample HTTP request – Open connection => TCP Socket

HYPERTEXT TRANSFER PROTOCOL

- A layer 1 protocol – Application Protocol
- HTTP 1.x is a text-based protocol
- It functions as a request/response protocol in client-server model
- Sample Request / Response



HTTP REQUEST METHODS

Method	Description
GET	Used to fetch data. The query parameters are provided inline via a query string. Each parameter will be a name value pair and will be separated with an &. The query string will start with a "?". ?name1=value1&name2=value2&name2=value3&name3=value4 (certain characters must be URLEncoded) http://www.walmart.com/search/?query=computer%20laptop https://www.google.com/search?q=traffic&ie=utf-8&oe=utf-8
HEAD	Used to fetch the response header only.
POST	Used to update data on the server or to POST form values. The data is sent after the headers and it always follows a blank line.
PUT	Used to insert a document on the server. This is useful for restful services.
DELETE	Used to remove a document on the server.
TRACE	Used to echo the received request so that a client can see what (if any) changes have been made by intermediate servers.
OPTIONS	Returns the HTTP methods that the server supports for the specified URL.
CONNECT	Converts the request connection to a TCP/IP tunnel, usually to facilitate HTTPs.
PATCH	I've never seen this one used before.

HTTP RESPONSE STATUS CODES

Code	Description
200	OK (success)
301	Permanently Moved
304	Not modified
400	Bad request
401	Unauthorized
403	Forbidden
405	Method not supported
408	Request timed out
500	Internal Server Error

CHROME DEVELOPER TOOLS