

Intro to Networking

TCP, HTTP, and APIs - oh my!

Andrew Kerr | October 20, 2015

me@andrewjkerr.com

whoami

whoami

- Fifth year Software Engineering @ UF

whoami

- Fifth year Software Engineering @ UF
- Secretary of UFSIT for > 2yrs

whoami

- Fifth year Software Engineering @ UF
- Secretary of UFSIT for > 2yrs
- Full stack web developer



whoami

- Fifth year Software Engineering @ UF
- Secretary of UFSIT for > 2yrs
- Full stack web developer
- Former security intern at Tumblr



whoami

- Fifth year Software Engineering @ UF
- Secretary of UFSIT for > 2yrs
- Full stack web developer
- Former security intern at Tumblr
- Former intern at BlockScore

Topics

Topics

- Networking layers

Topics

- Networking layers
- Wireshark

Topics

- Networking layers
- Wireshark
- TCP

Topics

- Networking layers
- Wireshark
- TCP
- HTTP

Topics

- Networking layers
- Wireshark
- TCP
- HTTP
- APIs

Tools

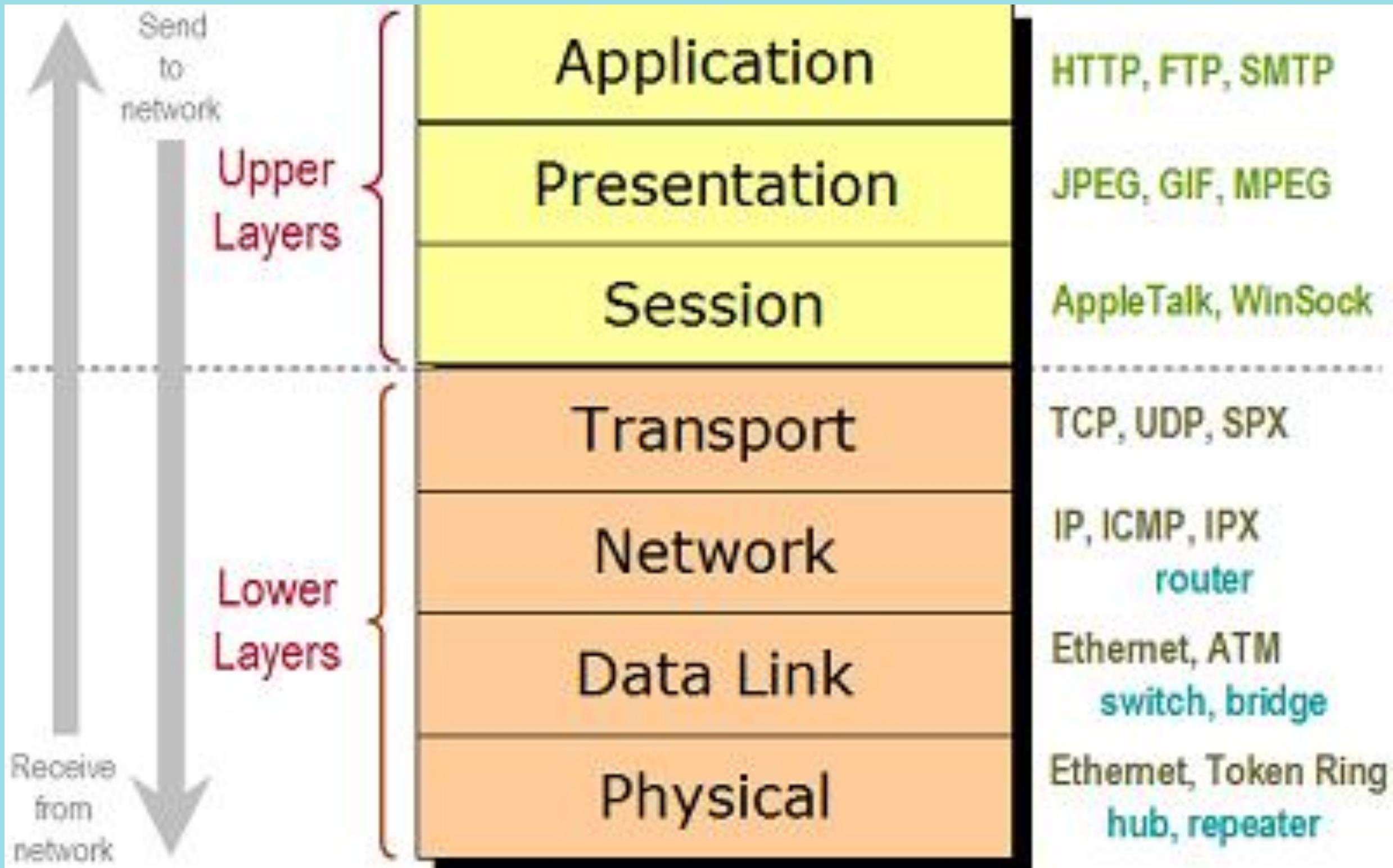
Tools

- Wireshark
- Google Chrome
- cURL
- Postman

Networking Layers

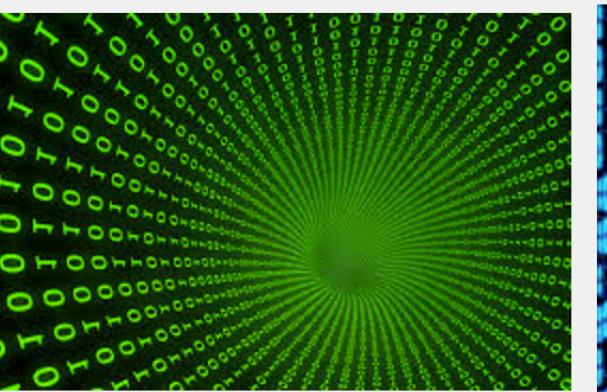
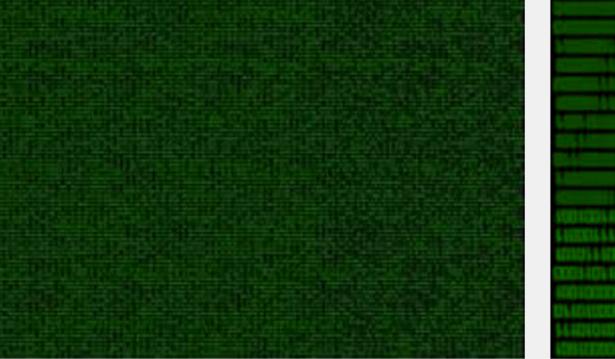
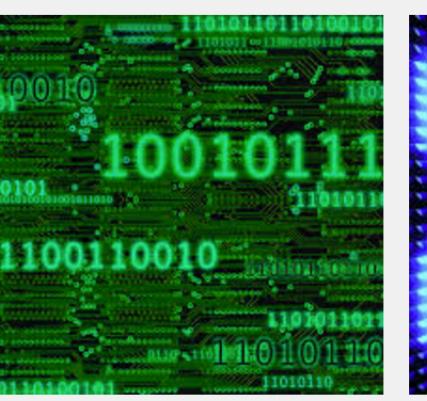
OSI Model

**The OSI model is a conceptual
model!**

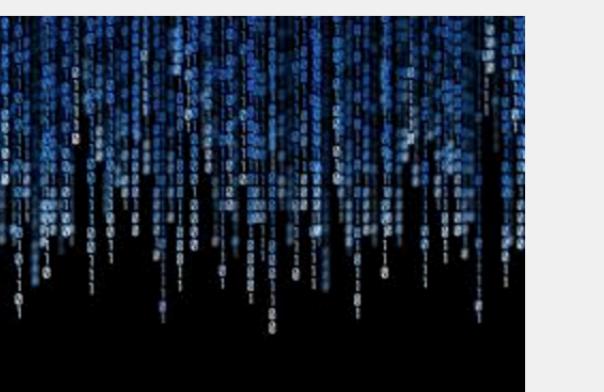
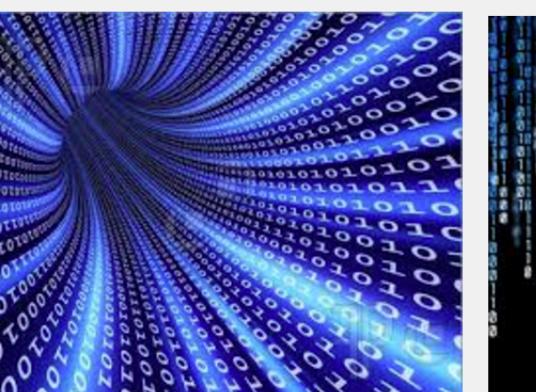


**The layers work together to
network your computer to the
world**

How do we send data back and forth?



001	0001	1010	0010	0001	0004	0128
016	0000	0028	0000	0016	0000	0020
001	0004	0000	0000	0000	0000	0000
000	0000	0010	0000	0000	0000	0204
384	0084	c7c8	00c8	4748	0048	e8e9
69	0059	8849	00a9	2828	0028	fdfc
819	0019	9898	0098	d9d8	00d8	5857
b7a	007a	bab9	00b9	3a3c	003c	8888
888	8888	8888	288e	be88	8888	8888
788	8888	8888	7667	778e	8828	8888
abd	8818	8888	467c	585f	8814	8188
817	88aa	88bb	8b83	88f3	88bd	e988
e841	c988	b328	6871	688e	958b	
862	5884	7e81	3788	lab4	5a34	3eec
c88	5ccb	8888	8888	8888	8888	8888
888	8888	8888	8888	8888	8888	0000
000	0000	0000	0000	0000	0000	0000
000	0000	0000	0000	0000	0000	0000



010100 01101000 01101001 01110011
100000 01101001 01110011 00100000
110100 01101000 01100101 00100000
110100 01110101 01110100 01101111
110010 01101001 01100001 01101100
100000 01110100 01101111 00100000
101100 01100101 01100001 01110010
101110 00100000 01100010 01101001
101110 01100001 01110010 01111001
101110 00100000 01001001 00100000
101000 01101111 01110000 01100101
100000 01110001 01101111 01110101
100000 01100101 01101110 01101010
101111 01111001 00100000 01101001
110100 00100001



The Past:

- Dial-up modem would translate 1s and 0s into tones

The Past:

- Dial-up modem would translate 1s and 0s into tones
- Receiving modem would decode 1s and 0s and change it back into data

Why doesn't this work now?

Too much data!

Too much data!

(Aka, you kids and your Netflix and chill)

What do we do now?

Fiber!!

Ok, but how do we send data?

packets and segments!

packets

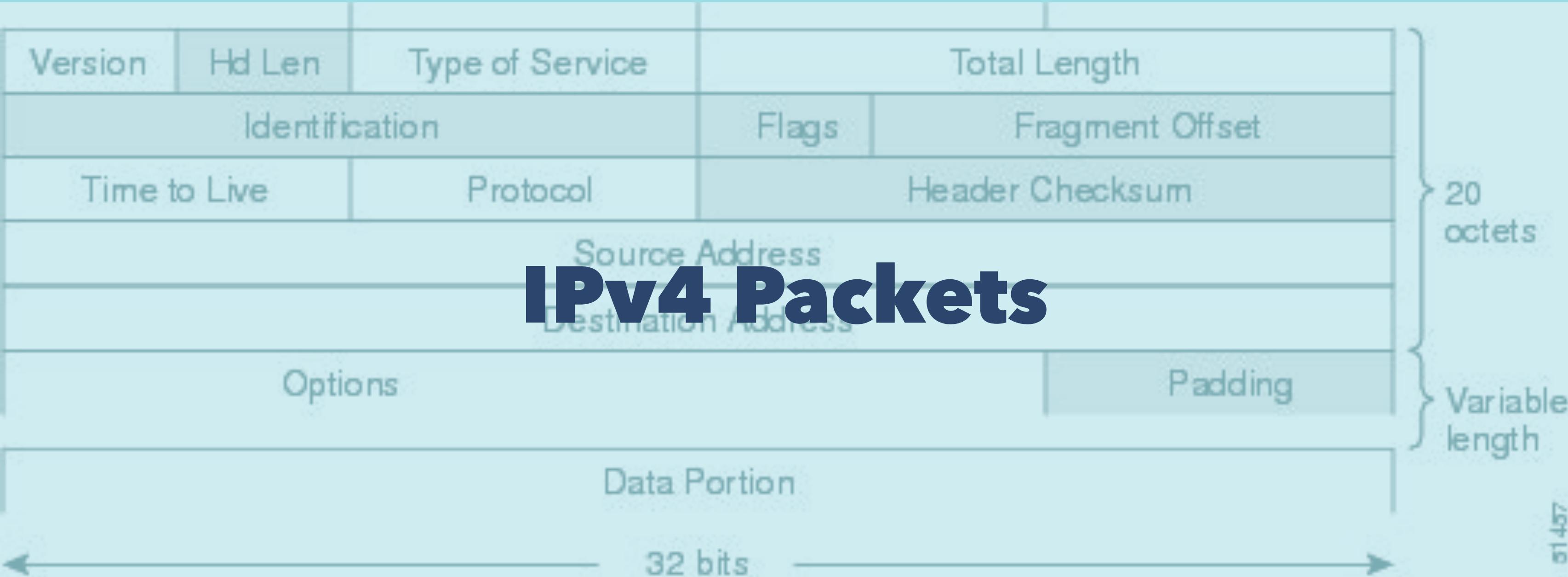
- Formatted unit of data

packets

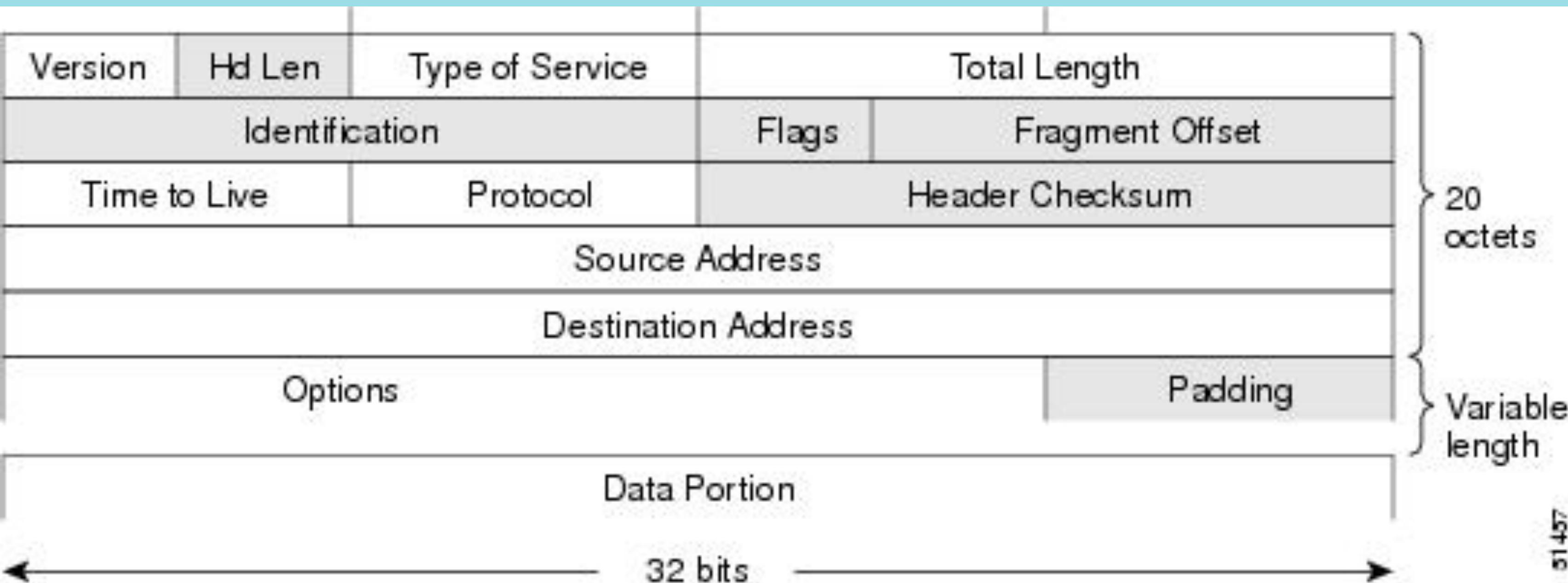
- Formatted unit of data
- Carried by a network

Packets

- Formatted unit of data
- Carried by a network
- Control information and payload



IPv4 Packets



Layer 4 Segments

TCP

(Segments)

TCP Segments

- "Transmission Control Protocol"

TCP Segments

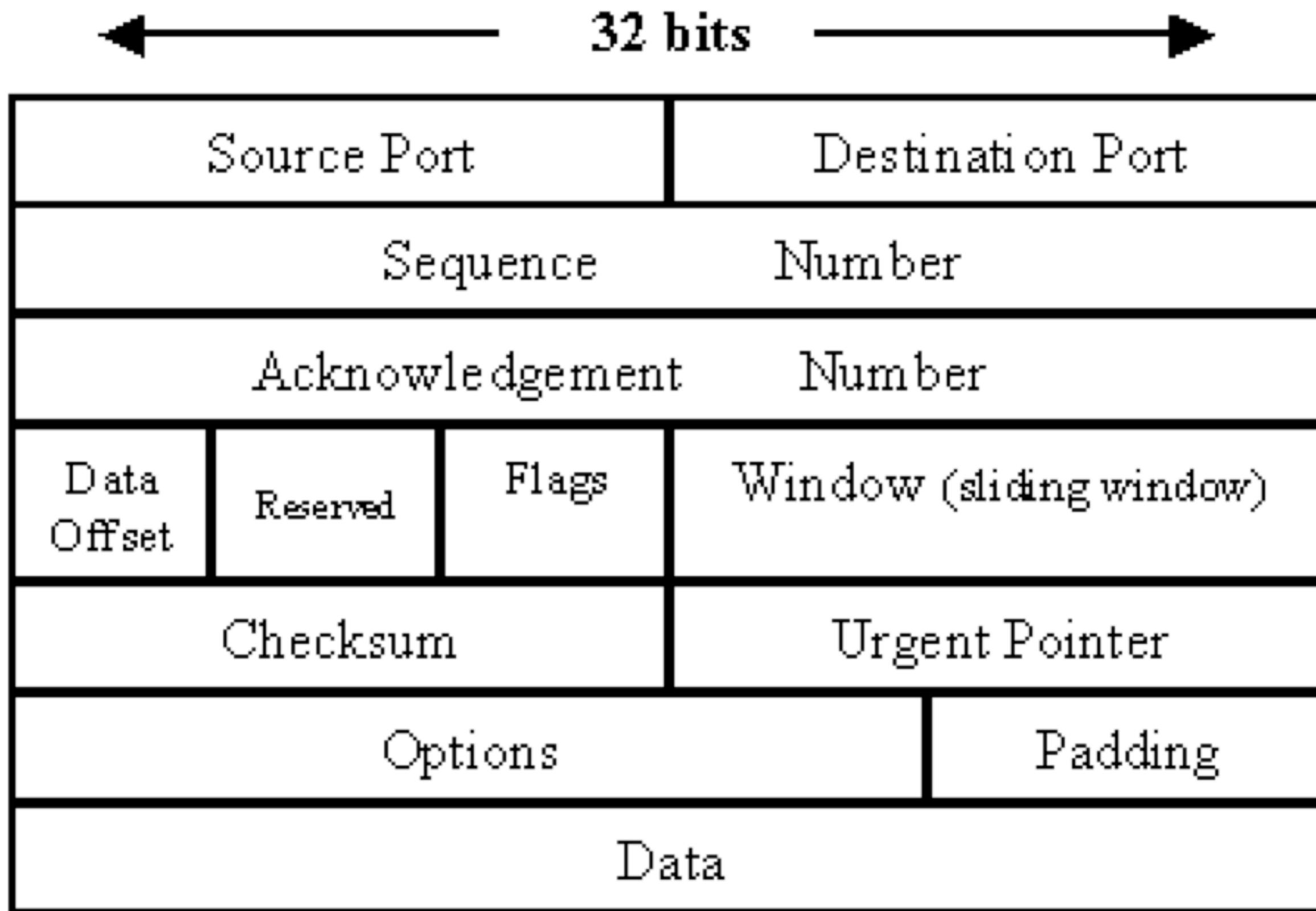
- "Transmission Control Protocol"
- Ordered

TCP Segments

- "Transmission Control Protocol"
- Ordered
- Error checked

TCP Segments

- "Transmission Control Protocol"
- Ordered
- Error checked
- Accepts a data stream and breaks it into chunks



Putting it all together...

TCP/IP

- Work together to transfer data!

TCP/IP

- Work together to transfer data!
- IP packet has source/destination IP, TCP segment has source/destination port

TCP/IP

- Work together to transfer data!
- IP packet has source/destination IP, TCP segment has source/destination port

TCP Handshake

TCP Handshake

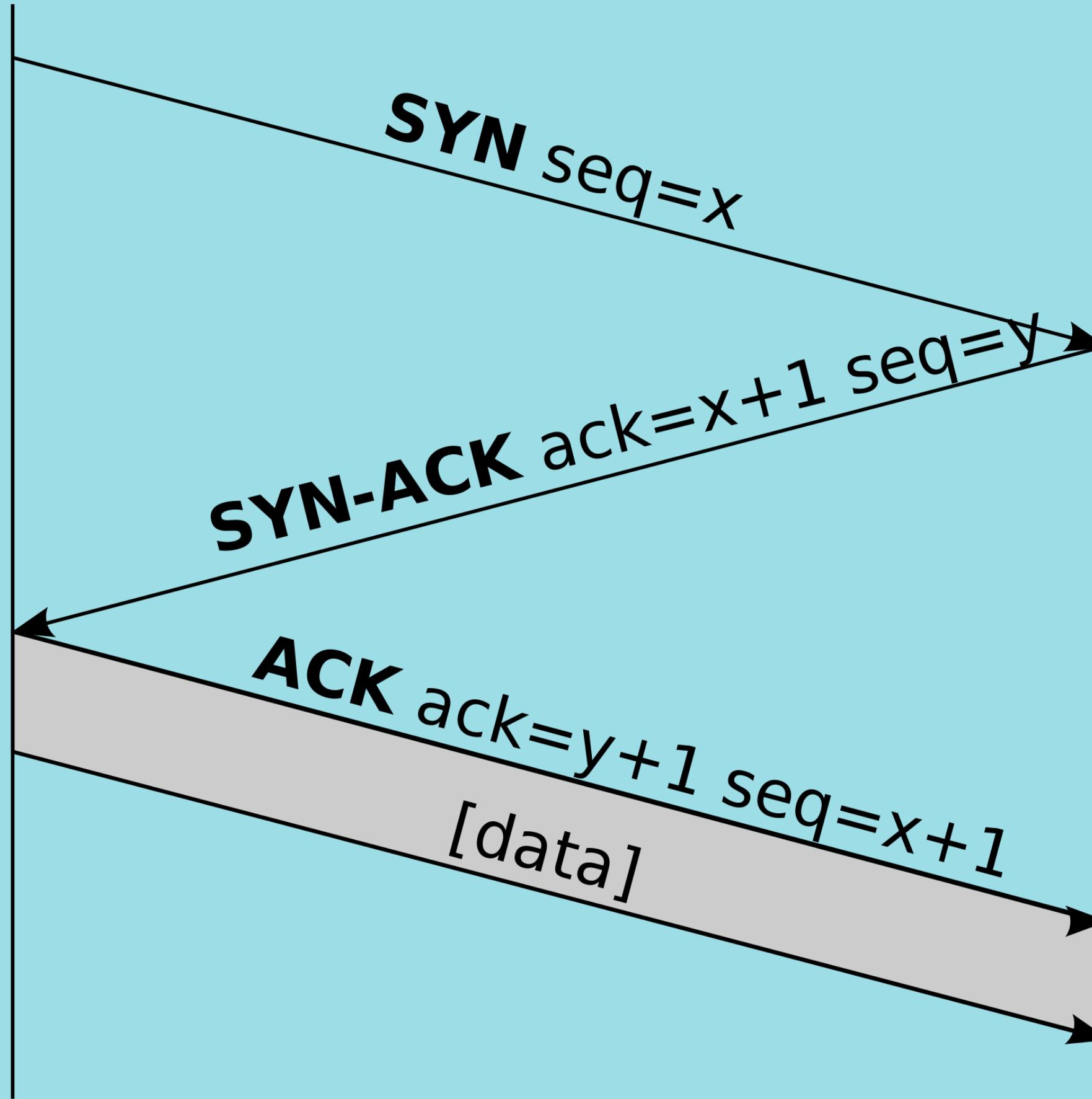
- A "three way handshake" before a client connects to a server

TCP Handshake

- A "three way handshake" before a client connects to a server
- Must be completed before data is exchanged

Client

Server



**Cool dude, but how do I load my
Tumblr dashboard?!**

**Cool dude, but how do I load my
Twitter timeline?!**

**Cool dude, but how do I load my
Facebook newsfeed?!**

**Cool dude, but how do I load my
pictures of funny cats?!**

**Cool dude, but how do I load my
pictures of cute cats?!**

Or, basically...

**How do we interact with things
on the web?**

Hypertext Transfer Protocol

Hypertext Transfer Protocol (HTTP)

Hypertext Transfer Protocol (HTTP)

- Provides methods (verbs) that we can perform

Hypertext Transfer Protocol (HTTP)

- Provides methods (verbs) that we can perform
- Defined GET, POST, and HEAD in HTTP/1.0

Hypertext Transfer Protocol (HTTP)

- Provides methods (verbs) that we can perform
- Defined GET, POST, and HEAD in HTTP/1.0
- Added OPTIONS, PUT, DELETE, TRACE, and CONNECT in HTTP/1.1

Hypertext Transfer Protocol (HTTP)

- Provides methods (verbs) that we can perform
- Defined **GET**, **POST**, and HEAD in HTTP/1.0
- Added OPTIONS, **PUT**, **DELETE**, TRACE, and CONNECT in HTTP/1.1

How does HTTP work?

How does HTTP work?

1. Client makes a request

How does HTTP work?

1. Client makes a request

```
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding:gzip, deflate, sdch
Accept-Language:en-US,en;q=0.8,es;q=0.6,sl;q=0.4
Cache-Control:no-cache
Connection:keep-alive
DNT:1
Host:ufsit.org
Pragma:no-cache
Upgrade-Insecure-Requests:1
User-Agent:(trimmed)
```

How does HTTP work?

1. Client makes a request
2. Server responds!

How does HTTP work?

1. Client makes a request
2. Server responds!

```
Access-Control-Allow-Origin:*
Cache-Control:max-age=600
Content-Encoding:gzip
Content-Type:text/html; charset=utf-8
Date:Tue, 20 Oct 2015 07:24:25 GMT
Expires:Tue, 20 Oct 2015 07:34:25 GMT
Last-Modified:Mon, 19 Oct 2015 22:56:06 GMT
Server:GitHub.com
Transfer-Encoding:chunked
```

Ok, Wireshark time!