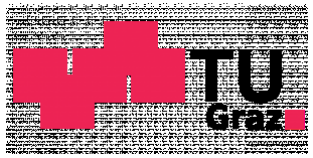
# Computer Organization and Networks

## Chapter 10: Networking III

Winter 2022/2023

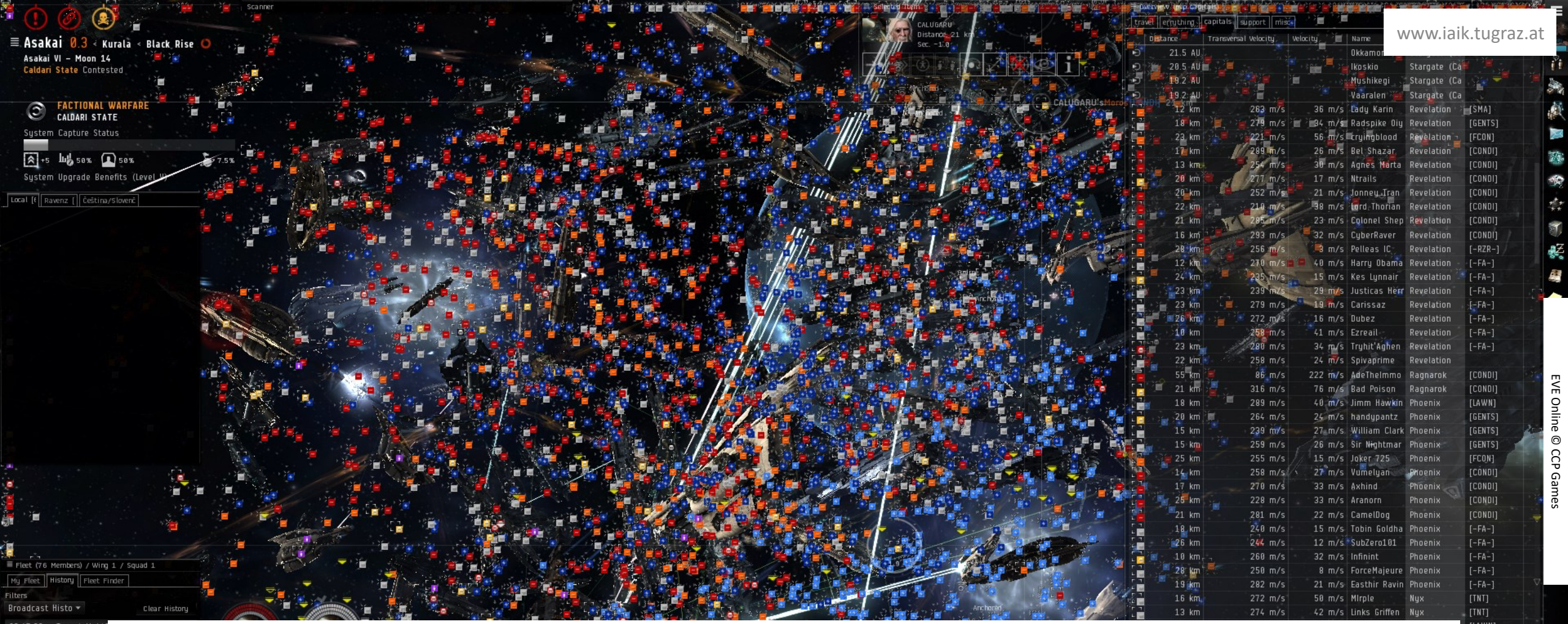



Jakob Heher, www.iaik.tugraz.at
he/his

# Recap

- **Data Link Layer**: send data to locally connected devices
  - Ethernet, Wi-Fi, Bluetooth, …
- **Network Layer:** send data to devices over the internet
  - IPv4, IPv6, …
- **Transport Layer:** structure the data into individual connections
  - TCP, UDP, …

- What's left?
  - Actually send useful data!




# The Application Layer

(some select applications)

# Domain Name System

- UDP port 53
- Transforms *host names* into IP addresses
  - **`online.tugraz.at`** ➠ **`129.27.2.210`**
- Hierarchical structure
  - **`.`** ➠ *root nameservers* (typically hardcoded)
  - **`at.`** ➠ ask **`127.30.48.1`** (**`dns.nic.at`**)
  - **`tugraz.at.`** ➠ ask **`129.27.2.3`** (**`ns1.tu-graz.ac.at`**)
  - **`online.tugraz.at.`** ➠ it's at **`129.27.2.210`**

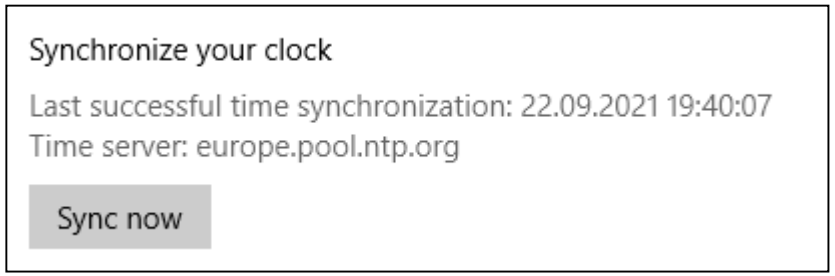
# Domain Name System

- Typically, the client queries a *DNS resolver* on port 53
  - Well-known public resolvers:
    `1.1.1.1` (Cloudflare), `8.8.8.8` (Google), `9.9.9.9` (Quad9)
- The DNS resolver performs the actual recursive lookup if needed
  - This allows centralized caching of responses!

- DNS resolver address can also be determined via DHCP
  - Recall: **D**ynamic **H**ost **C**onfiguration **P**rotocol
    - It does IP address auto-configuration, we talked about it 😉

# Network Time Protocol

- UDP port 123
- Time synchronization over the internet
- Synchronized clocks are required for many operations
  - Time-based 2FA tokens, expiry of SSL certificates, Kerberos tokens, …



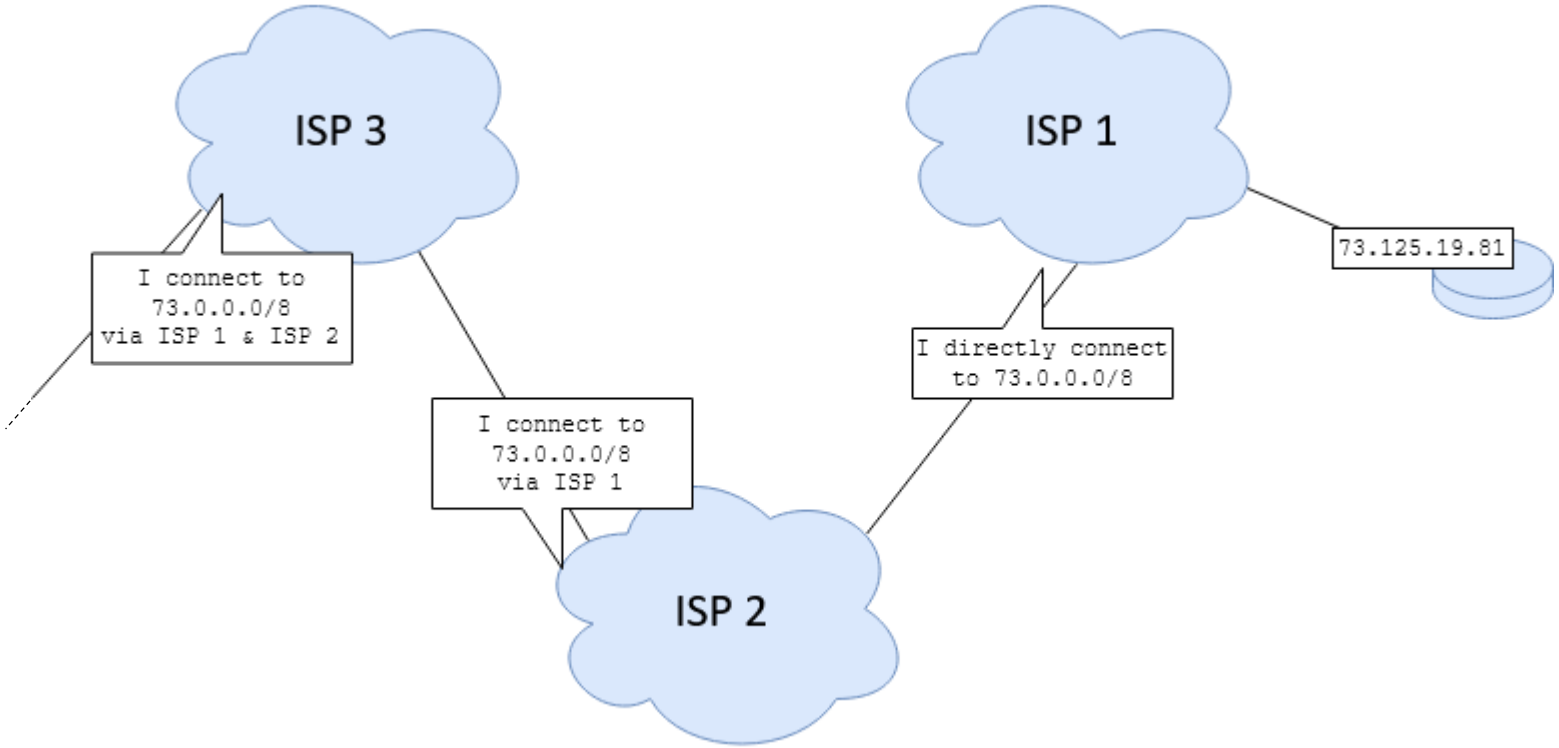

```
gallantron@ipn009:~$ ntpdate -d pool.ntp.org
23 Sep 14:11:04 ntpdate[159]: ntpdate 4.2.8p12@1.3728-o (1)
Looking for host pool.ntp.org and service ntp
91.206.8.34 reversed to svn.mediainvent.at
host found : svn.mediainvent.at

server 91.206.8.34, port 123
stratum 2, precision -21, leap 00, trust 000
refid [161.143.24.141], root delay 0.000610, root dispersion 0.029587
transmitted 4, in filter 4
reference time:    e4f6e9b1.246b51c0  Thu, Sep 23 2021 13:55:29.142
originate timestamp: e4f6ed5e.e6ff1685  Thu, Sep 23 2021 14:11:10.902
transmit timestamp:  e4f6ed5e.e602ffcb  Thu, Sep 23 2021 14:11:10.898
filter delay:  0.04388  0.03754  0.03914  0.03960
        0.00000  0.00000  0.00000  0.00000
filter offset: -0.00477 -0.00492 -0.00568 -0.00318
        0.000000 0.000000 0.000000 0.000000
delay 0.03754, dispersion 0.00082
offset -0.004928
```

# Secure SHell

- TCP port 22
- Secure remote administration
  - Unless you use a(n insecure) password…
- Using SSH as a building block in other applications is popular
  - SSH provides authentication + encryption
  - Example: **`git`**

# Border Gateway Protocol

- TCP port 179
- Responsible for maintaining the global IP routing table
  - Essentially a distributed shortest-path graph algorithm



# HyperText Transfer Protocol

- TCP port 80 (HTTP), TCP port 443 (**HTTP** over **S**SL)
  - SSL adds authentication & encryption – more on this next year!
- Every web page you view uses it

- Simple concept: ask the server for a document
  - The meaning of “document” has evolved greatly over the years
  - Originally: actual *document*, a static piece of content
  - Today: anything you can possibly imagine, often dynamically generated
    - Many applications communicate via HTTP due to its ubiquitous support

# HTTP request

Method

Requested resource

```
GET /document.html HTTP/1.1\r\n
Accept: text/html\r\n
Host: webserver.net\r\n
User-Agent: SimpleWebBrowser\r\n
Connection: keep-alive\r\n
\r\n
```

Request line

Headers

Blank line – end of headers

# HTTP response

Status code

```
HTTP/1.1 200 OK\r\n
Content-Type: text/html; charset=UTF-8\r\n
Content-Length: 47\r\n
Server: MyWebServer\r\n
Connection: close\r\n
\r\n
<html><head><title>Hello!</title></head></html>
```

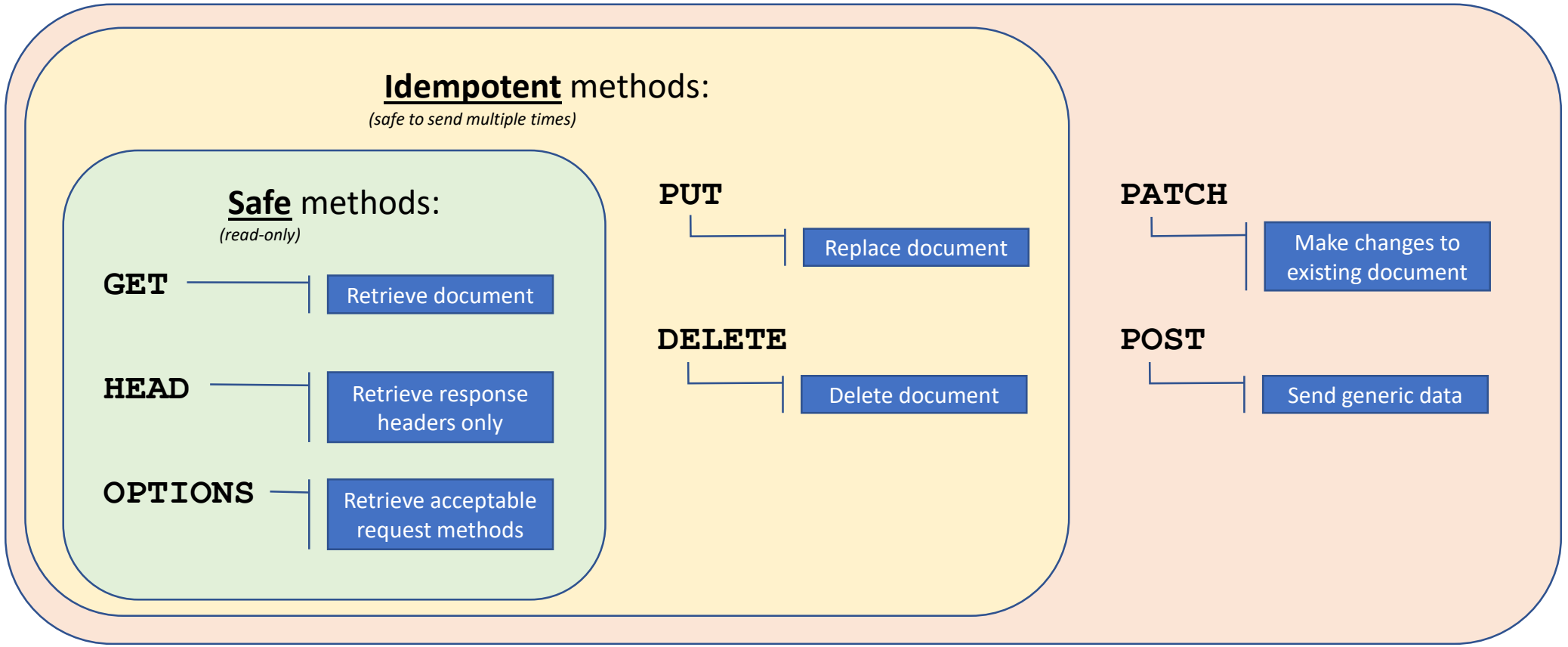
Status line

Headers

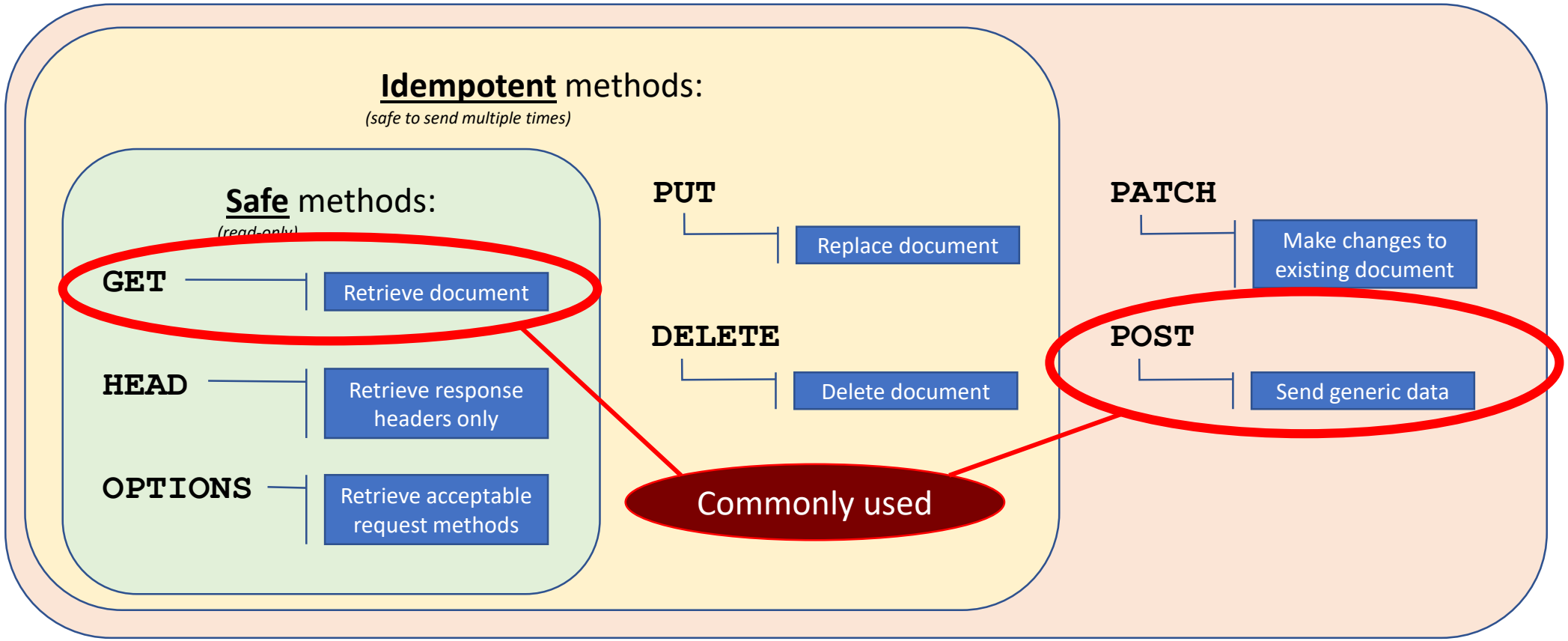
Blank line – end of headers

Requested document

# HTTP methods



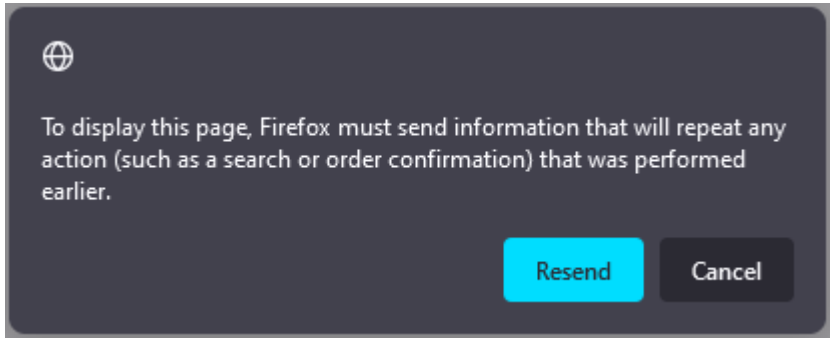
# HTTP methods



# HTTP methods

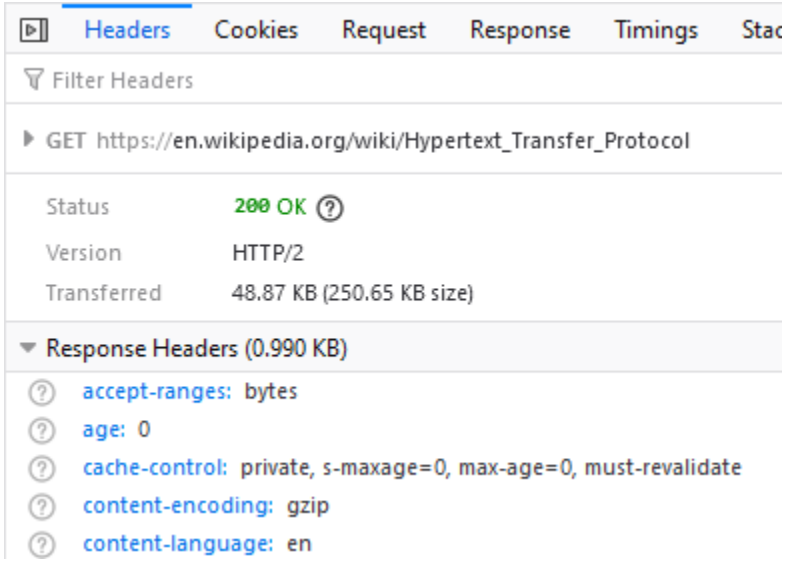
- Method functionality is purely by convention
  - There's nothing stopping you from deleting a file when a **`GET`** request is made
  - Just because you *can*, doesn't mean you *should*…

- Clients will offer different degrees of safeguards for different methods
  - Example: Reloading the result of a **`POST`** request triggers a dialog box



# HTTP – you can try this at home!

- Open a new browser tab
- Enable the developer tools (F12 in Firefox and Chrome)
- Switch to the “Network” tab
- Open your favorite website
- Each line is one HTTP request being made
  - Click on them to see what’s happening!



# HTTP/1.1: Head-of-Line blocking

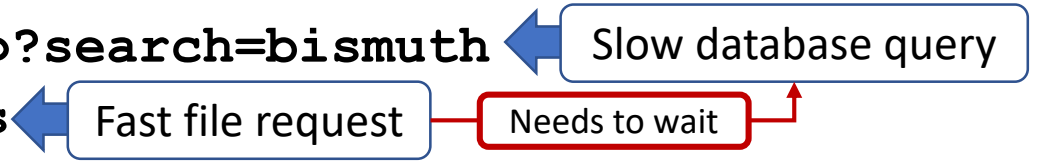
- Multiple requests can be sent over a single connection!
  - But: the responses still need to come in order…
    - **`search.php?search=bismuth`**

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8"><meta http-equiv="x-ua-compatible" content="ie-edge">
    <title>Searching for 'bismuth'…</title>
    <link rel="stylesheet" href="css/search.css">
    <script defer src="js/search.js"></script>
  </head>
  <body>
    <h1>Your results are:</h1>
    <ul>
      <li><a href="gem.php?id=83">Bismuth<a></li>
      <li><a href="alloy.php?id=LBE">Lead-Bismuth Eutectic</a></li>
    </ul>
  </body>
</html>
```

We'll need these two, too!

# HTTP/1.1: Head-of-Line blocking

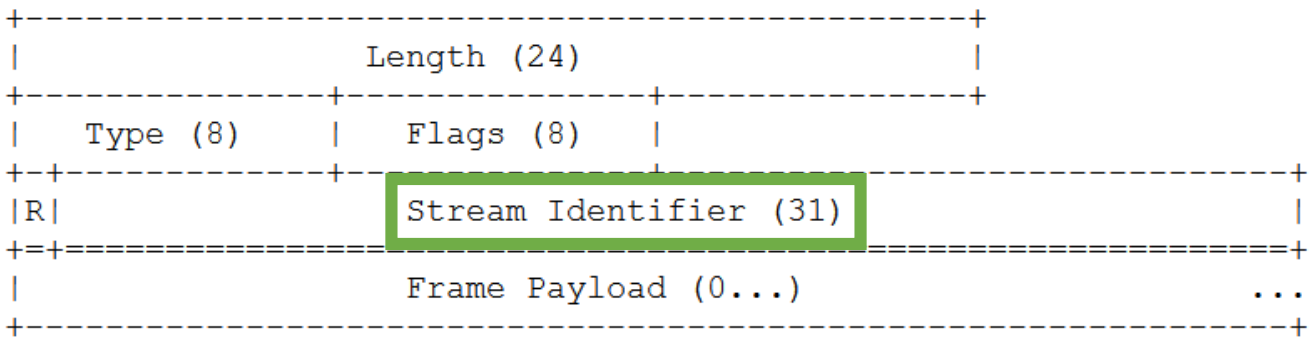
- Multiple requests can be sent over a single connection!
  - But: the responses still need to come in order…
    - **`search.php?search=bismuth`**
    - **`search.css`**




  - Even though we know we need the file, we can't retrieve it…

# HTTP/2

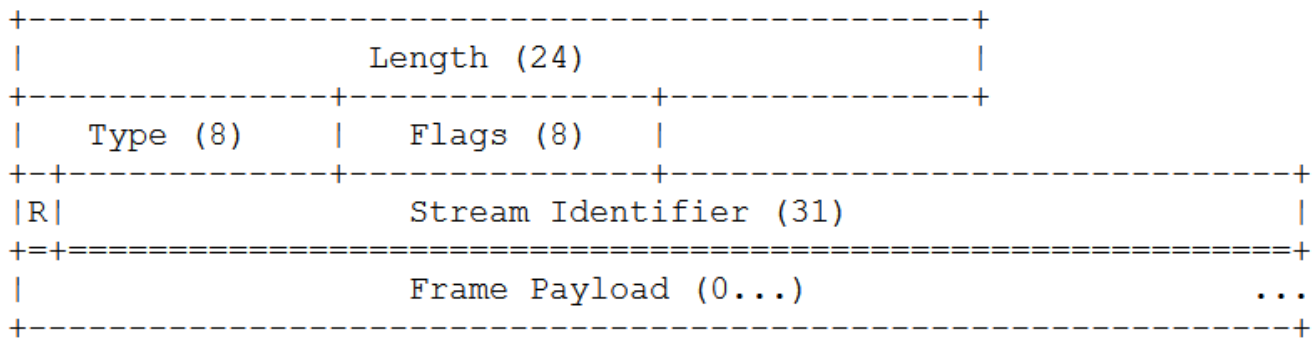
- No longer human readable
  - Binary representation is more compact




- Same request/response semantics, header fields, etc…
- **Stream Identifier** allows multiple responses to be sent in parallel

# HTTP/2

- No longer human readable
  - Binary representation is more compact




- Same request/response semantics, header fields, etc…
- Stream Identifier allows multiple responses to be sent in parallel
- Still runs over a TCP connection…

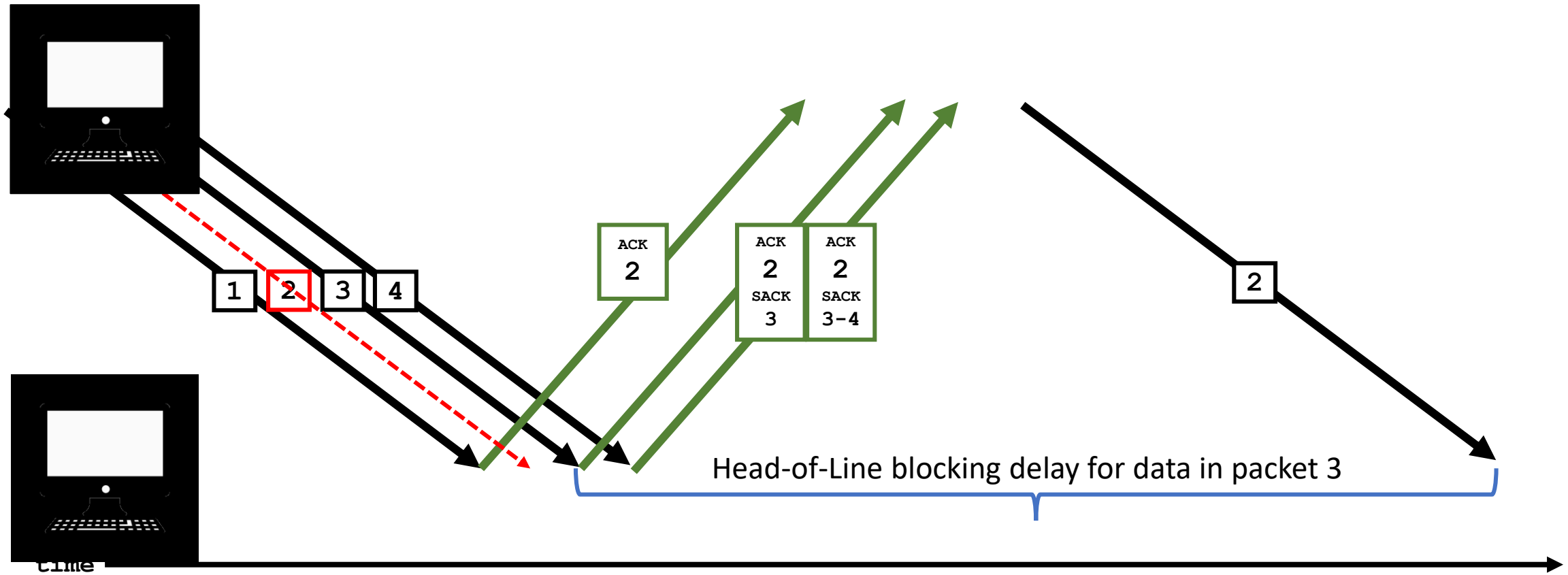



# The Transport Layer

(again, for a brief interlude)

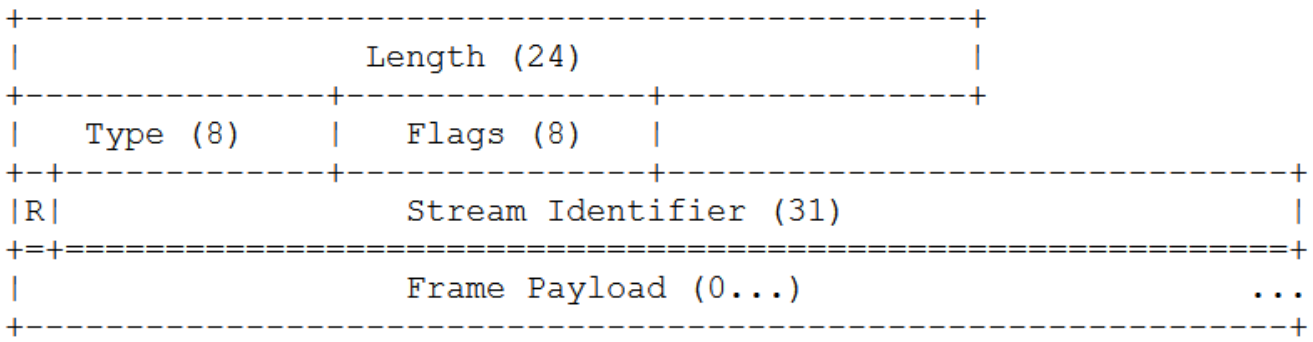
# TCP: Head-of-Line blocking

- The TCP connection is still a single byte stream
  - While we wait for packet 2, we cannot process packets 3 and 4…



# HTTP/2

- No longer human readable
  - Binary representation is more compact




- Same request/response semantics, header fields, etc…
- Stream Identifier allows multiple responses to be sent in parallel
- Still runs over a TCP connection…
  - Packet loss on any stream “pauses” all streams’ data

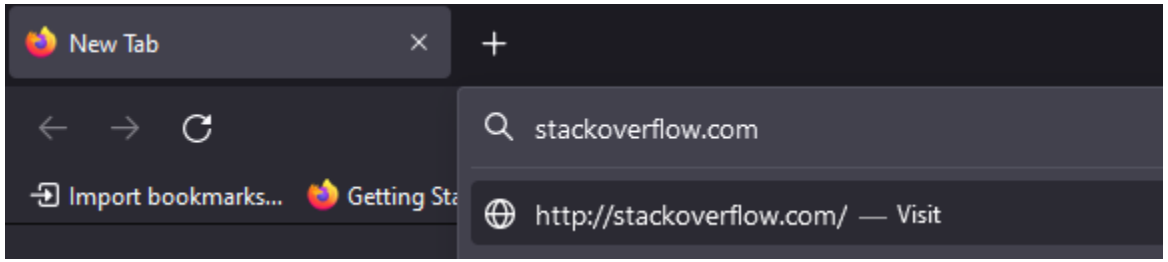
# HTTP/3

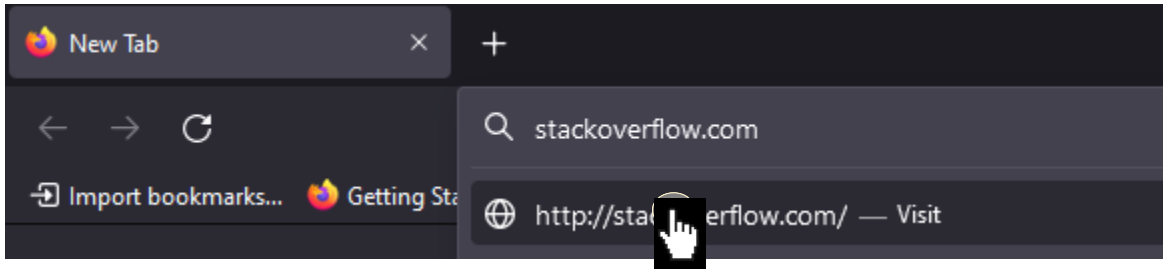
- QUIC to replace TCP
  - **Q**uick **U**DP **I**nternet **C**onnection
    - Originally developed by Google, since standardized by IETF
  - UDP at the transport layer for minimal overhead
  - Provides byte stream facilities similar to TCP
    - Aware of multiple data streams within a single connection
    - … re-invents the wheel, in a way
- Otherwise identical to HTTP/2

# … and many, many, many, many more …

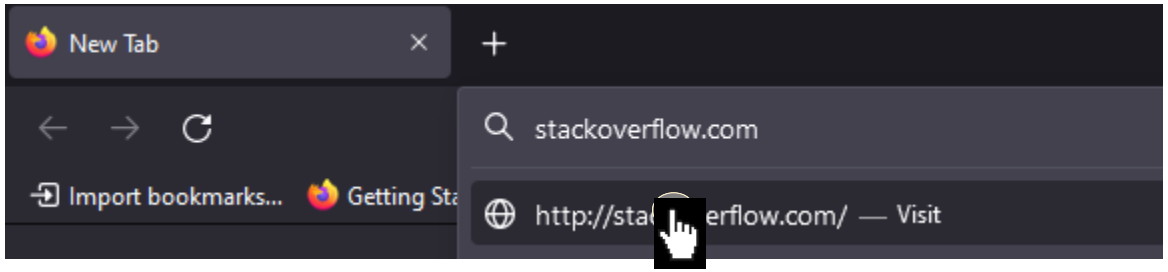
- Any program with network features is part of “the application layer”
  - Try running **`netstat`** while logged into your favorite game!

# Putting it all together

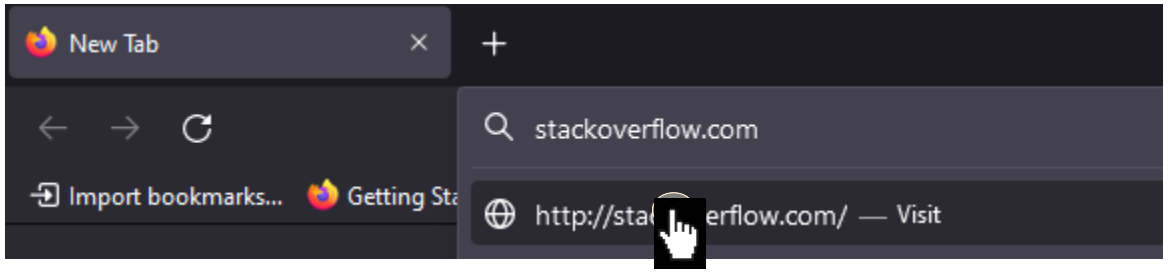

New Tab
stackoverflow.com
Import bookmarks...
Getting Sta
http://stackoverflow.com/ — Visit




- OK, what happens?




- Step 1: We need to figure out where **`stackoverflow.com`** is!
  - Let's ask our favorite DNS resolver!
  - DNS resolvers listen on UDP port 53
  - UDP is stateless, so we can just send our DNS request




- Let's ask our favorite DNS resolver where **`stackoverflow.com`** is!

Hi, transport layer. Please send the following data to 8.8.4.4 on UDP port 53:

```
b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f l o w 03 c o m 00 00 ff 00 01
```

- Let's ask our favorite DNS server where **`stackoverflow.com`** is!

Hi, transport layer. Please send the following data to 8.8.4.4 on UDP port 53:

`b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f l o w 03 c o m 00 00 ff 00 01`

Pick an unused UDP port… `49640`

Attach a UDP header… done.

Network layer, please send the following data to 8.8.4.4:

`c1 e8 00 35 00 23 00 00`

`b8 35 01 00 00 01 00 00 00 00 0 l o w 03 c o m 00 00 ff 00 01`

| 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 | Length | | | | | | | | | | | | | | | | Checksum | | | | | | | | | | | | | | |

Pick an unused UDP port… 49640
Attach an UDP header… done.
Network layer, please send the following data to 8.8.4.4:

```
c1 e8 00 35 00 23 00 00
b8 35 01 00 00 01 00 00 00 00 00 0d s t a c k o v e r f l o w 03 c o m 00 00 ff 00 01
```

Our IP address… it's 192.168.42.18.
What's the next hop towards 8.8.4.4? The default gateway, 192.168.42.254.
What's the MAC address for that… ah, it's f2:d3:09:9d:53:1a.
Link Layer, please send the following data to f2:d3:09:9d:53:1a via port eth0:

```
45 00 00 3f 1a 8a 00 00 ff 11 aa 5d c0 a8 2a 12 08 08 04 04 c1 e8 00 35 00 23 00 00
b8                                                          03 c o m 00 00 ff 00 01
```

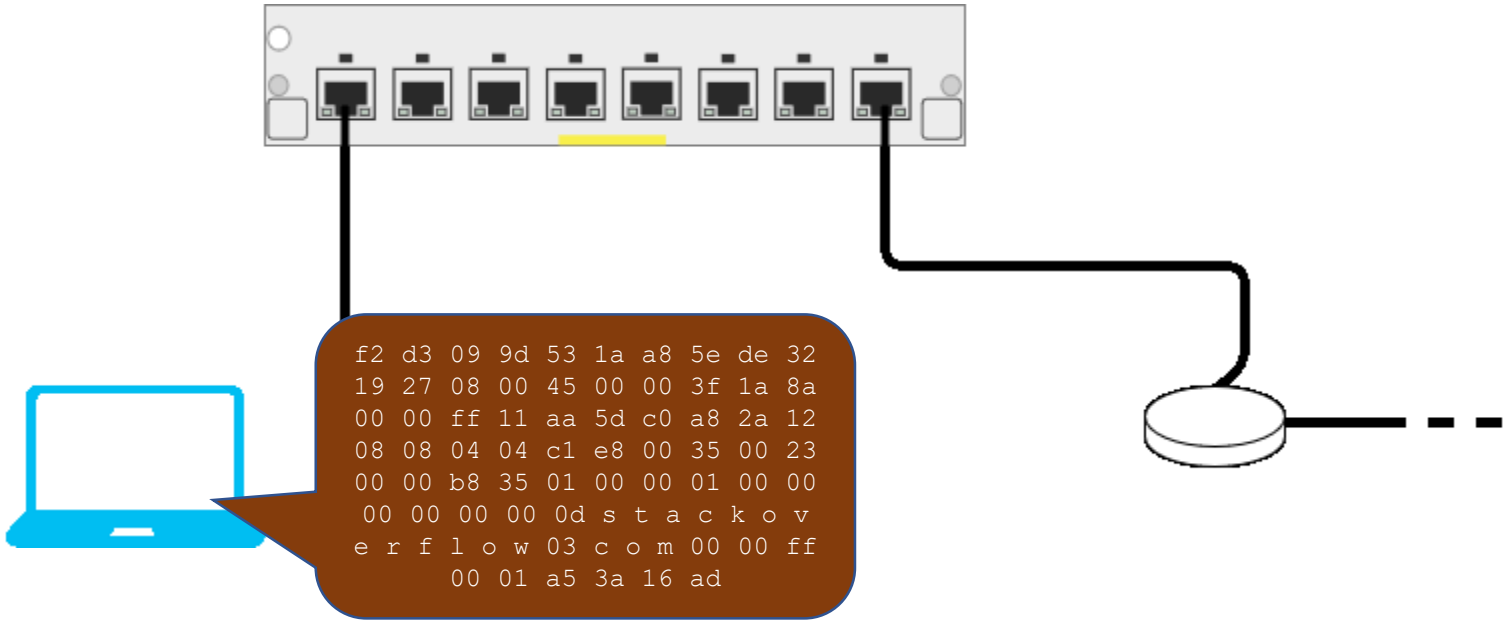
| *Offsets* | Octet | 0 | | | | | | | | 1 | | | | | | | | 2 | | | | | | | | 3 | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Octet | Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| 0 | 0 | Version | | | | Header Length | | | | DSCP | | | | | | ECN | | Total Length | | | | | | | | | | | | | | | |
| 4 | 32 | Identification | | | | | | | | | | | | | | | | Flags | | | Fragment Offset | | | | | | | | | | | |
| 8 | 64 | Time To Live | | | | | | | | Protocol | | | | | | | | Header Checksum | | | | | | | | | | | | | | |
| 12 | 96 | Source IP Address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 16 | 128 | Destination IP Address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

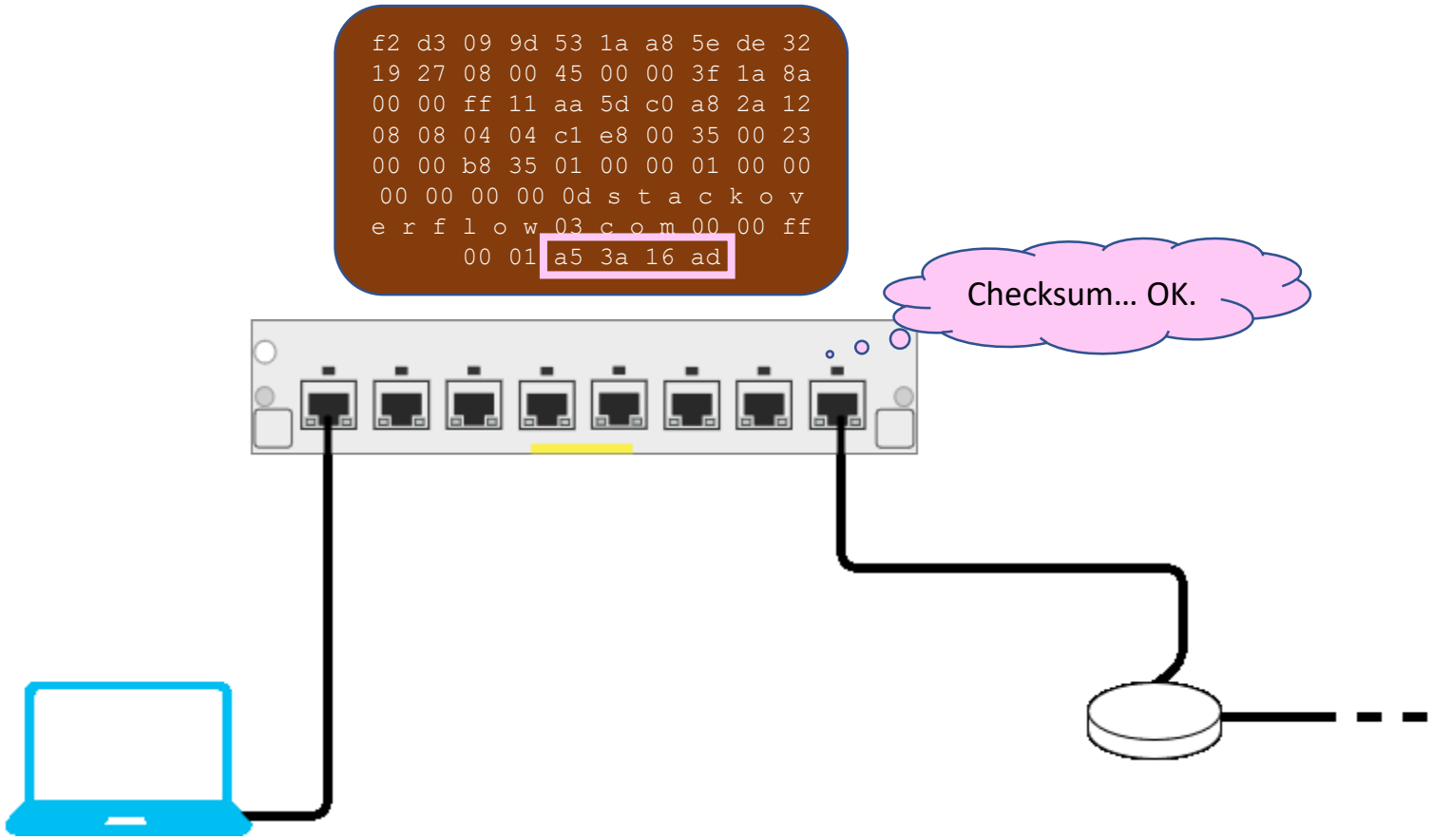
Our IP address… it's 192.168.42.18.
What's the next hop towards 8.8.4.4? The default gateway, 192.168.42.254.
What's the MAC address for that… ah, it's `f2:d3:09:9d:53:1a`.
Link Layer, please send the following data to `f2:d3:09:9d:53:1a` via port `eth0`:

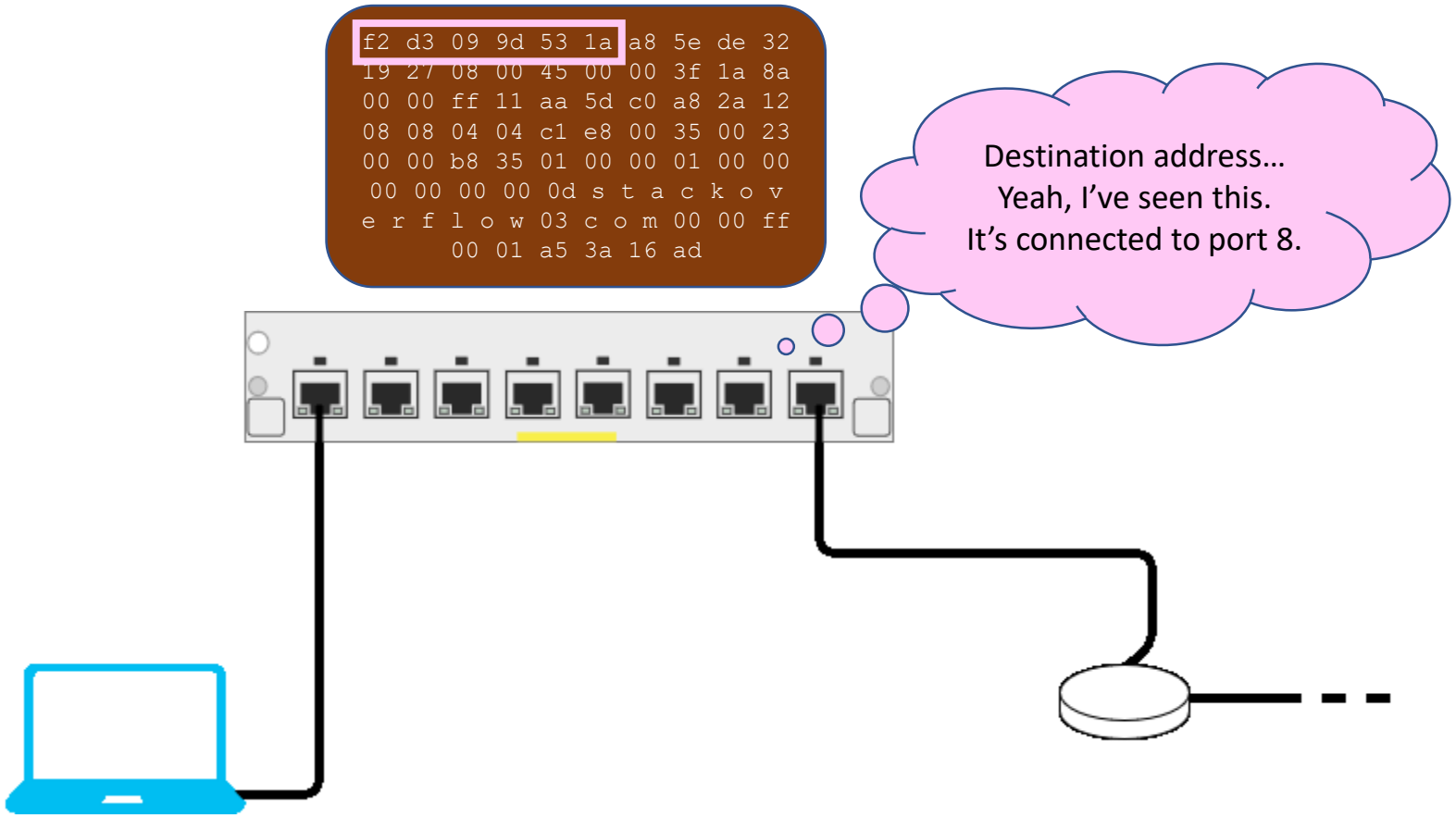
```
45 00 00 3f 1a 8a 00 00 ff 11 aa 5d c0 a8 2a 12 08 08 04 04 c1 e8 00 35 00 23 00 00
b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f l o w 03 c o m 00 00 ff 00 01
```

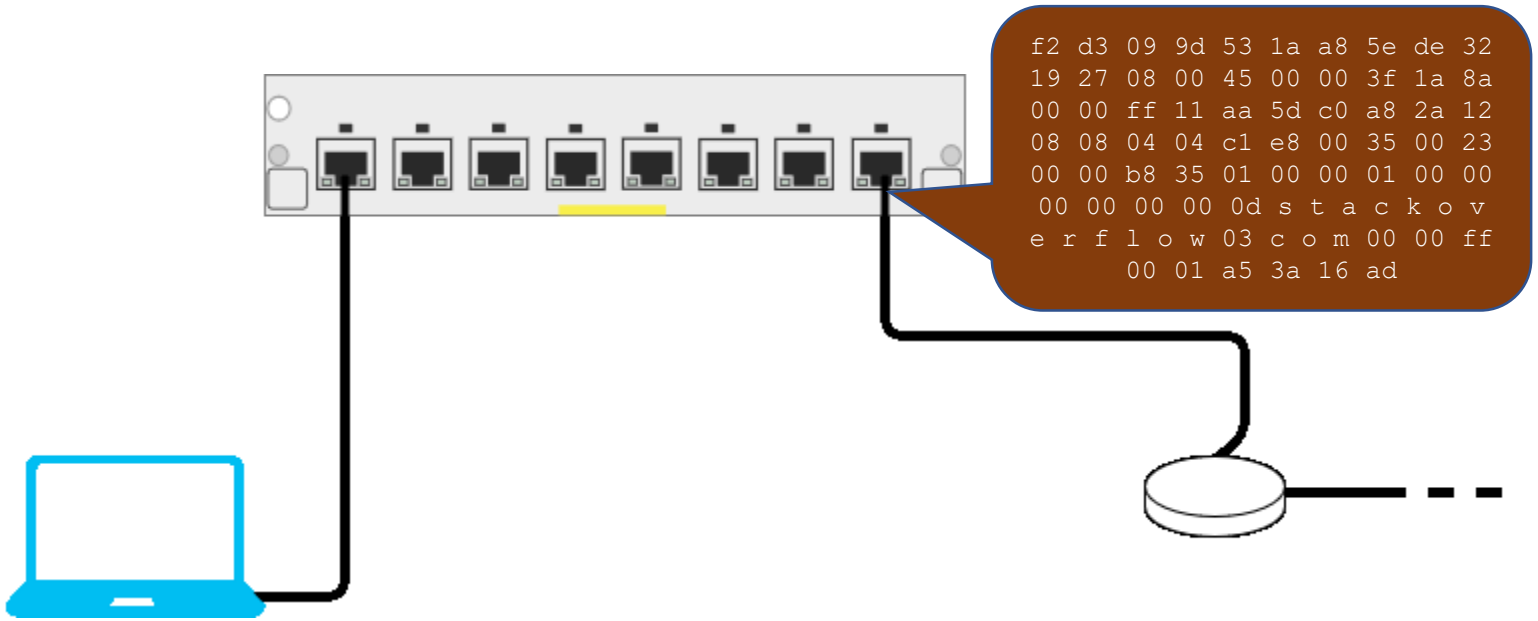
Our MAC address is `a8:5e:de:32:19:27`.
Hi, other side. Here's frame data modulated into the electrical current flowing over the wire:

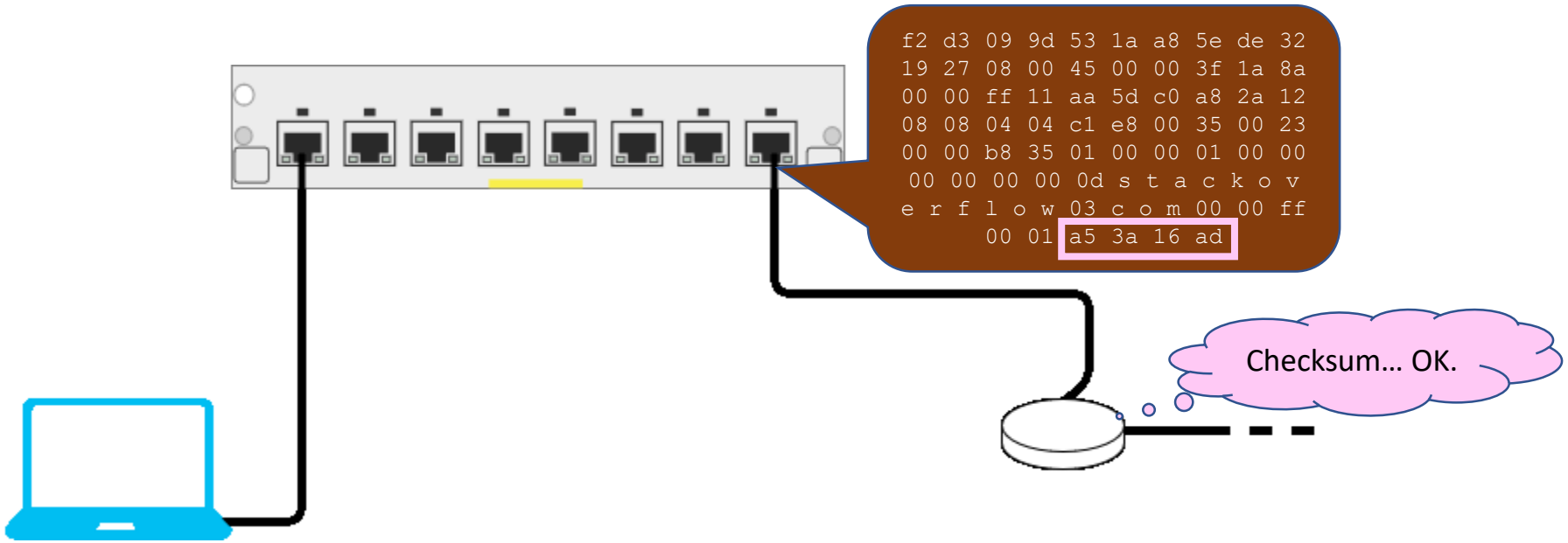
```
f2 d3 09 9d 53 1a a8 5e de 32 19 27 08 00 45 00 00 3f 1a 8a 00 00 ff 11 aa 5d c0 a8 2a 12 08
08 04 04 c1 e8 00 35 00 23 00 00 b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f
l o w 03 c o m 00 00 ff 00 01 a5 3a 16 ad
```

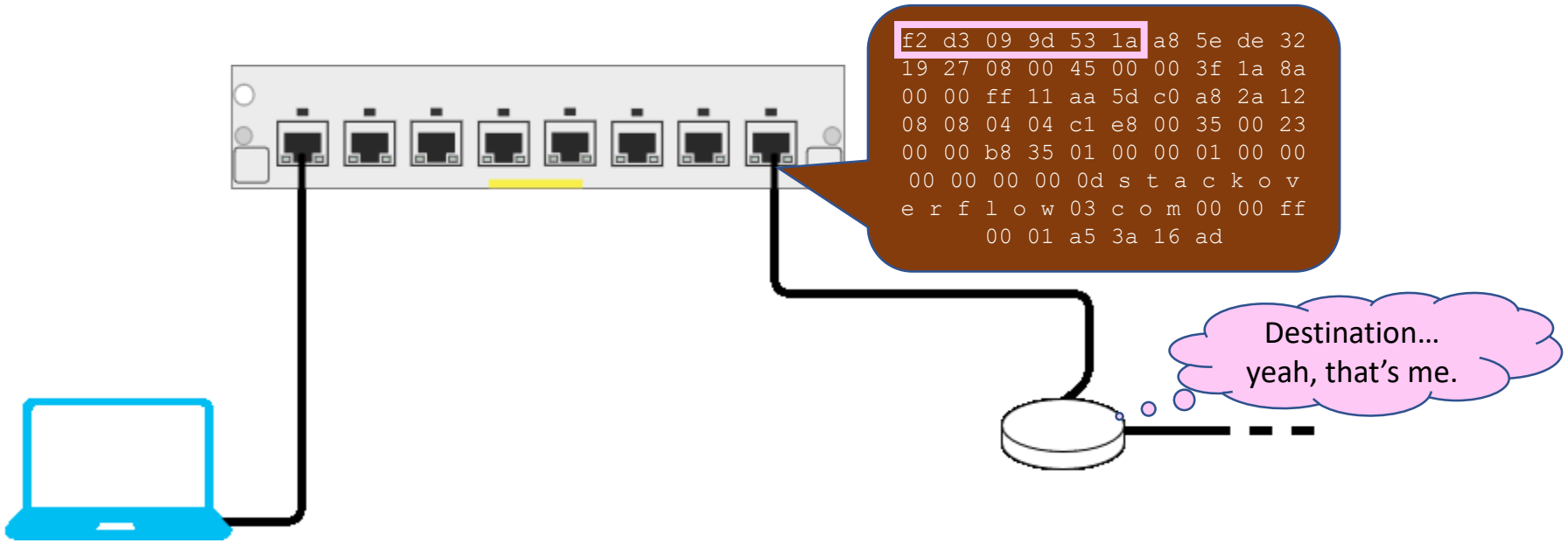

f2 d3 09 9d 53 1a a8 5e de 32
19 27 08 00 45 00 00 3f 1a 8a
00 00 ff 11 aa 5d c0 a8 2a 12
08 08 04 04 c1 e8 00 35 00 23
00 00 b8 35 01 00 00 01 00 00
00 00 00 00 0d s t a c k o v
e r f l o w 03 c o m 00 00 ff
00 01 a5 3a 16 ad


f2 d3 09 9d 53 1a a8 5e de 32
19 27 08 00 45 00 00 3f 1a 8a
00 00 ff 11 aa 5d c0 a8 2a 12
08 08 04 04 c1 e8 00 35 00 23
00 00 b8 35 01 00 00 01 00 00
00 00 00 00 0d s t a c k o v
e r f l o w 03 c o m 00 00 ff
00 01 a5 3a 16 ad
Checksum… OK.


f2 d3 09 9d 53 1a a8 5e de 32
19 27 08 00 45 00 00 3f 1a 8a
00 00 ff 11 aa 5d c0 a8 2a 12
08 08 04 04 c1 e8 00 35 00 23
00 00 b8 35 01 00 00 01 00 00
00 00 00 00 0d s t a c k o v
e r f l o w 03 c o m 00 00 ff
00 01 a5 3a 16 ad
Destination address…
Yeah, I've seen this.
It's connected to port 8.


f2 d3 09 9d 53 1a a8 5e de 32
19 27 08 00 45 00 00 3f 1a 8a
00 00 ff 11 aa 5d c0 a8 2a 12
08 08 04 04 c1 e8 00 35 00 23
00 00 b8 35 01 00 00 01 00 00
00 00 00 00 0d s t a c k o v
e r f l o w 03 c o m 00 00 ff
00 01 a5 3a 16 ad


f2 d3 09 9d 53 1a a8 5e de 32
19 27 08 00 45 00 00 3f 1a 8a
00 00 ff 11 aa 5d c0 a8 2a 12
08 08 04 04 c1 e8 00 35 00 23
00 00 b8 35 01 00 00 01 00 00
00 00 00 00 0d s t a c k o v
e r f l o w 03 c o m 00 00 ff
00 01 a5 3a 16 ad
Checksum… OK.


f2 d3 09 9d 53 1a a8 5e de 32
19 27 08 00 45 00 00 3f 1a 8a
00 00 ff 11 aa 5d c0 a8 2a 12
08 08 04 04 c1 e8 00 35 00 23
00 00 b8 35 01 00 00 01 00 00
00 00 00 00 0d s t a c k o v
e r f l o w 03 c o m 00 00 ff
00 01 a5 3a 16 ad
Destination…
yeah, that's me.

Hey, Network Layer, data arrived for you:

```
45 00 00 3f 1a 8a 00 00 ff 11 aa 5d c0 a8 2a 12 08 08 04 04 c1 e8 00 35 00 23 00 00
b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f l o w 03 c o m 00 00 ff 00 01
```

Hey, Network Layer, data arrived for you:

```
45 00 00 3f 1a 8a 00 00 ff 11 aa 5d c0 a8 2a 12 08 08 04 04 c1 e8 00 35 00 23 00 00
b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f l o w 03 c o m 00 00 ff 00 01
```

Let's see… destinat… add…4.4? That's not me, I need to pass this on…
First, decrea…e T… and recalculate the checksum… then…
Ne…op i… rection is… 192.168.255.254…
I have i… AC… cached… `a7:a2:23:95:d6:a6`.
Link Layer, plea…end… a to `a7:a2:23:95:d6:a6` via port `eth1`:

```
45 00 00 3f 1a 8a 00 00 fe 11 ab 5d c0 a8 2a 12 08 08 04 04 c1 e8 00 35 00 23 00 00
b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f l o w 03 c o m 00 00 ff 00 01
```

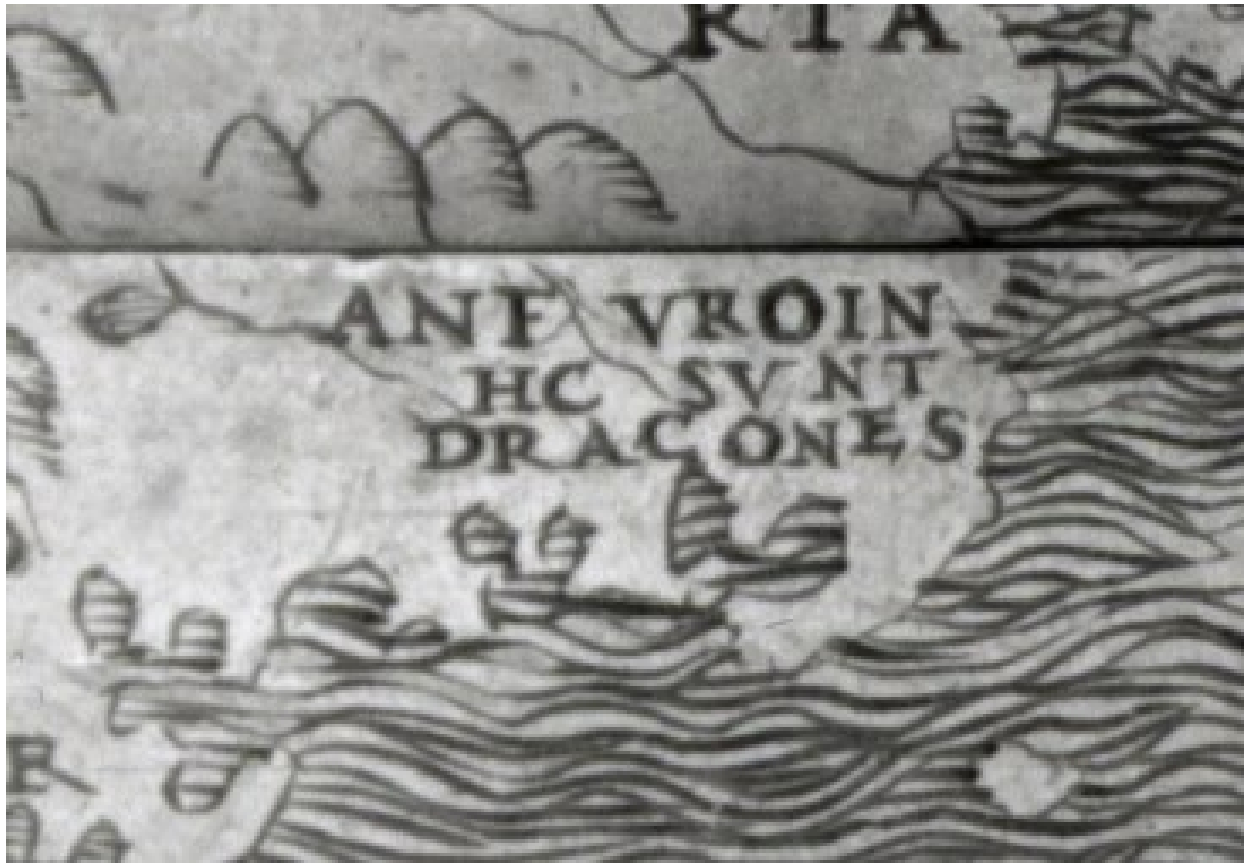
Let's see… destination address 8.8.4.4? That's not me, I need to pass this on…
First, decrease the TTL field, and recalculate the checksum… then…
Next hop in that direction is… 192.168.255.254…
I have its MAC address cached… `a7:a2:23:95:d6:a6`.
Link Layer, please send this data to `a7:a2:23:95:d6:a6` via port `eth1`:

```
45 00 00 3f 1a 8a 00 00 fe 11 ab 5d c0 a8 2a 12 08 08 04 04 c1 e8 00 35 00 23 00 00
b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f l o w 03 c o m 00 00 ff 00 01
```

Our MAC address on that interface… `58:42:30:35:8a:08`.
Hi, other side, here's data:

```
a7 a2 23 95 d6 a6 58 42 30 35 8a 08 08 00 45 00 00 3f 1a 8a 00 00 ff 11 aa 5d c0 a8 2a 12 08
08 04 04 c1 e8 00 35 00 23 00 00 b8 35 01 00 00 01 00 00 00 00 00 00 0d s t a c k o v e r f
l o w 03 c o m 00 00 ff 00 01 f6 11 22 44
```


…






# Looking Forward…

# Let’s Get Dangerous

- So far, everyone has played nice…
  - Real life is difficult, and people are people
- INP.33404UF Information Security
  - Suddenly, everyone is evil – and so are you
  - See you next year!