CSCI-1680
How to (try) to be anonymous
Wrapup

Nick DeMarinis

Based partly on lecture notes by Rodrigo Fonseca, Scott Shenker and John Jannotti

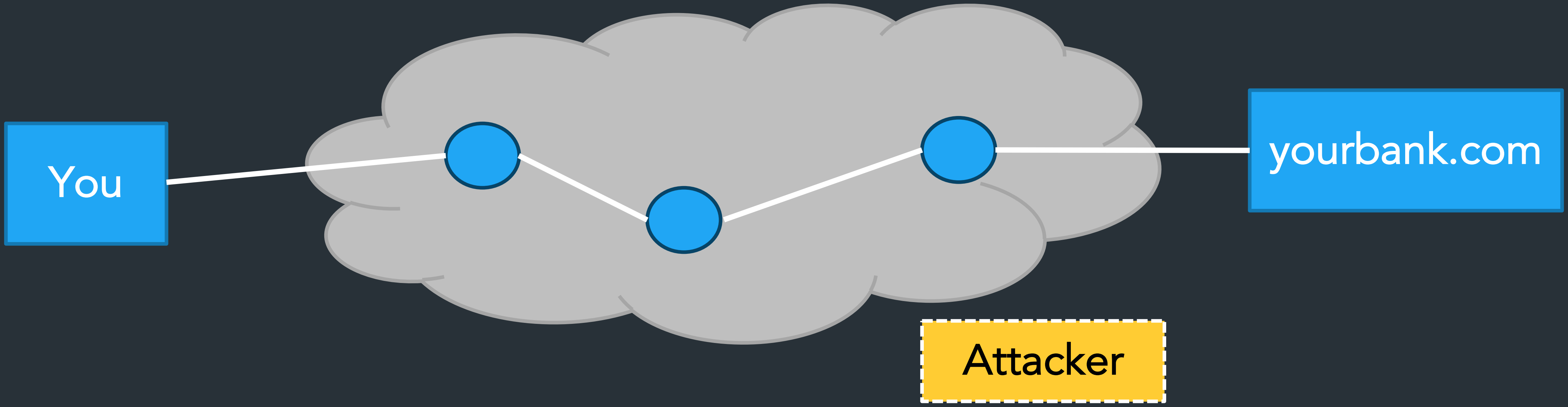
Administrivia

- HW5: Due Monday, 12/9
- Final project: Due 12/16
- SRC problem: Due 12/16 (will be some form of extra credit)
- Office hours: see the calendar

- Course feedback
  - University feedback
  - Critical Review
  - I will send you a form

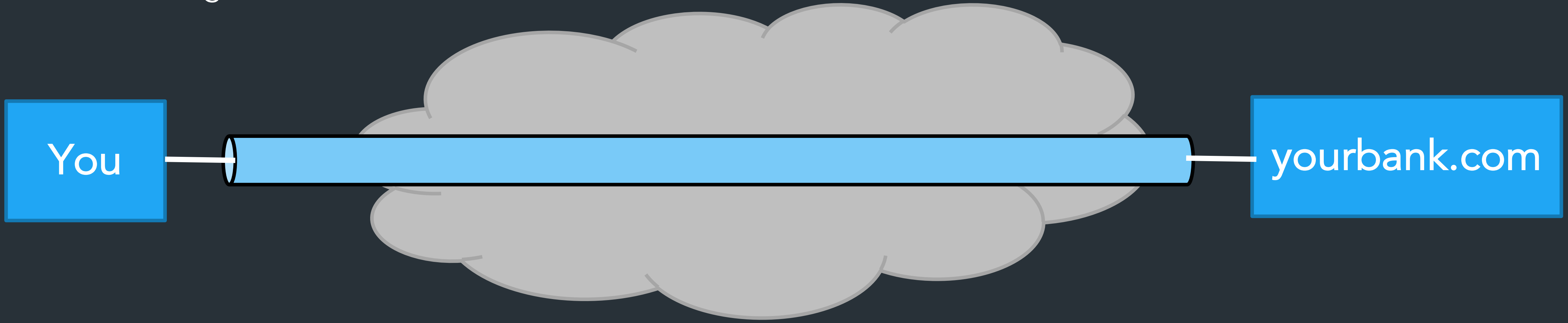
My (major) TODOs

1. I owe you grades on Snowcast, TCP
2. I owe you a bunch of lecture notes
3. I will be watching Ed for final project questions


You
yourbank.com
Attacker

Warmup

With TLS we get this:



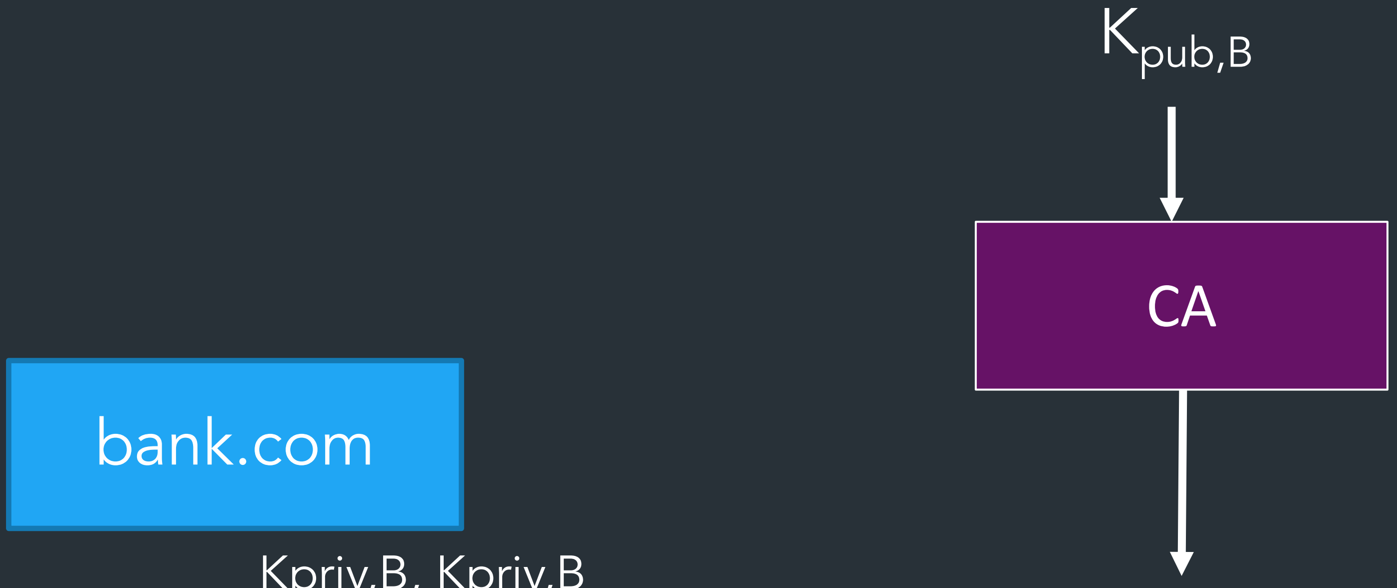
Are we good? Have we solved web security?

Warmup

bank.com

Kpriv,B, Kpriv,B

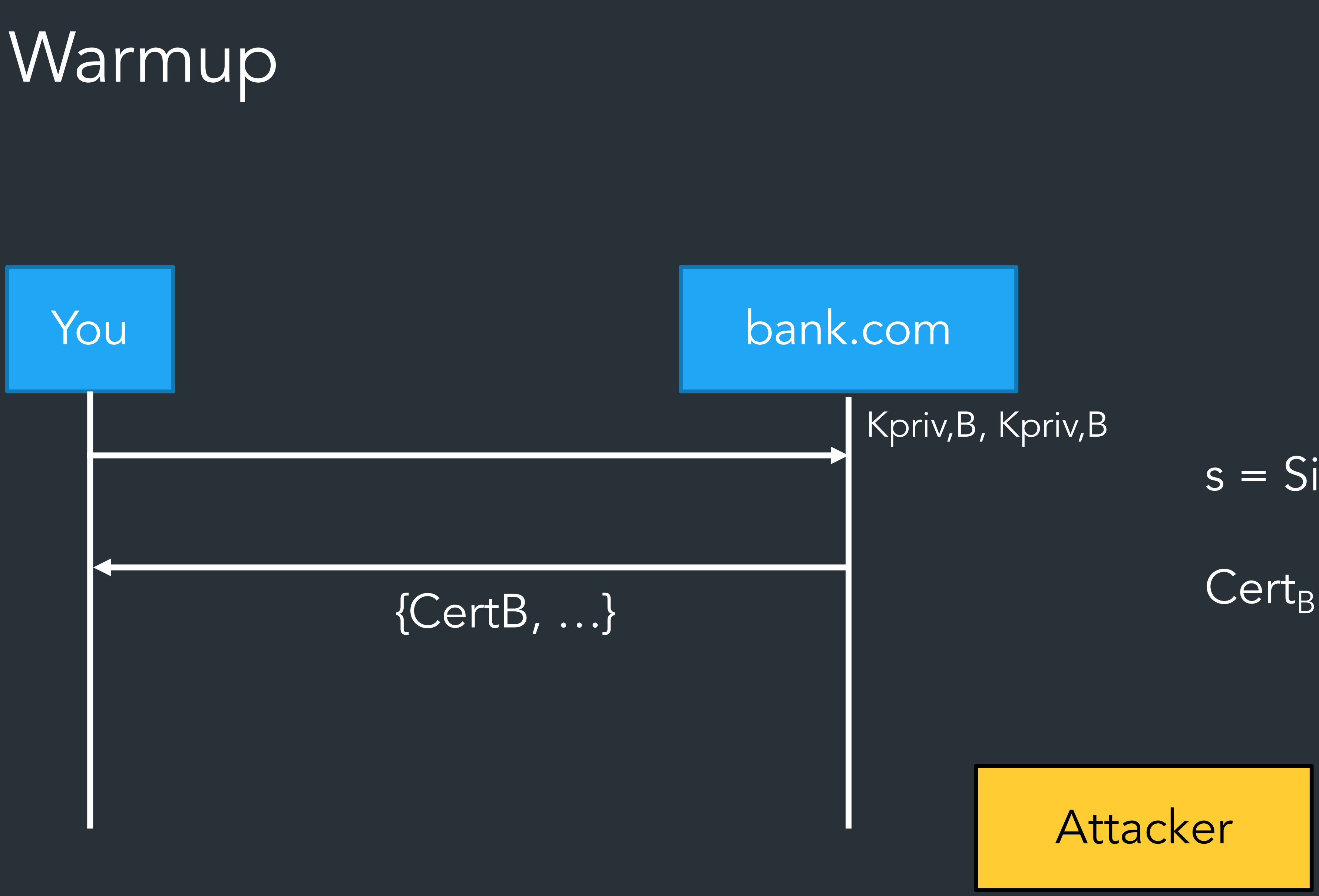
Warmup



$$s = \text{Sign}(K_{priv,CA}, \{K_{pub,B}, \dots\})$$

$$\text{Cert}_B = \{K_{pub,B}, \text{metadata}, s\}$$

Warmup

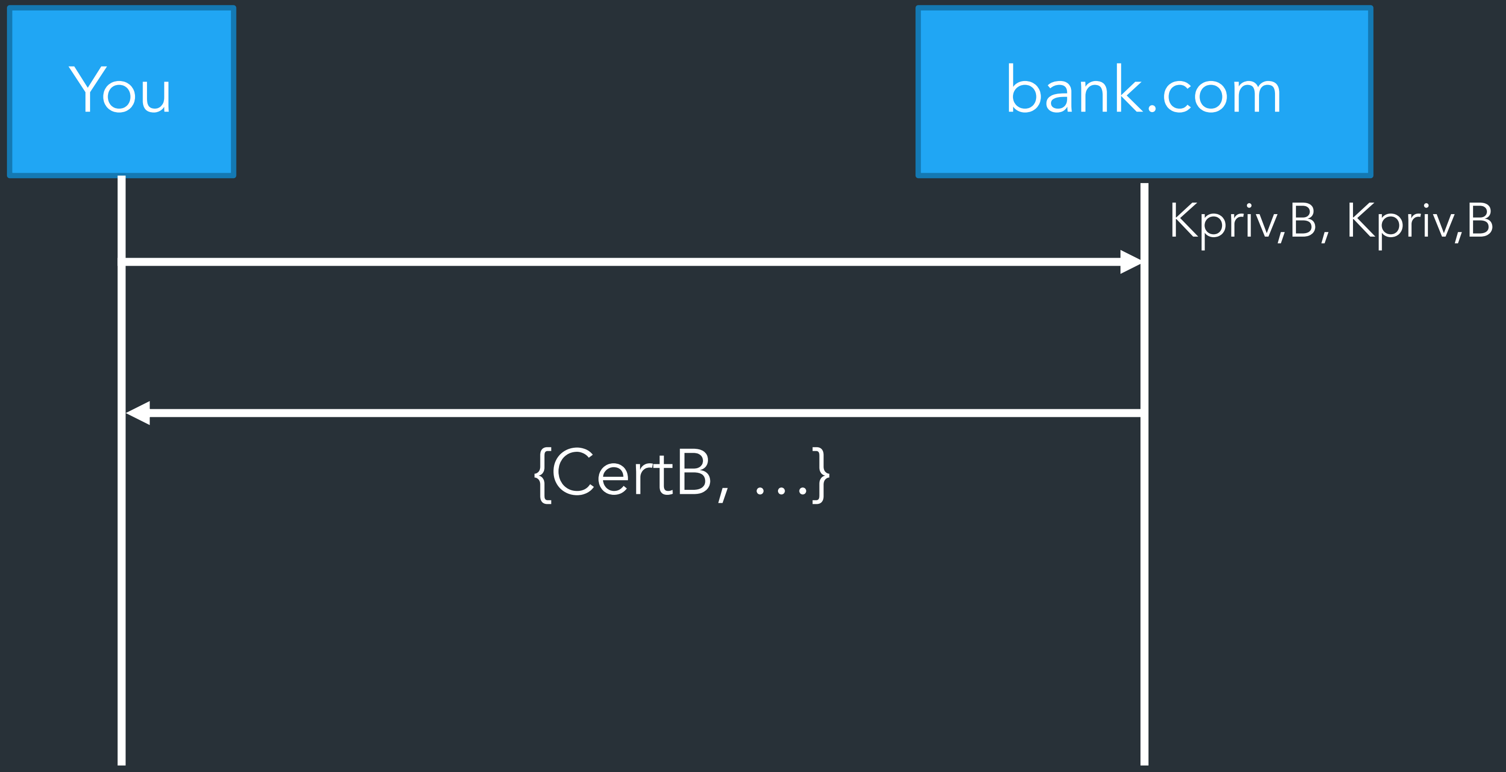


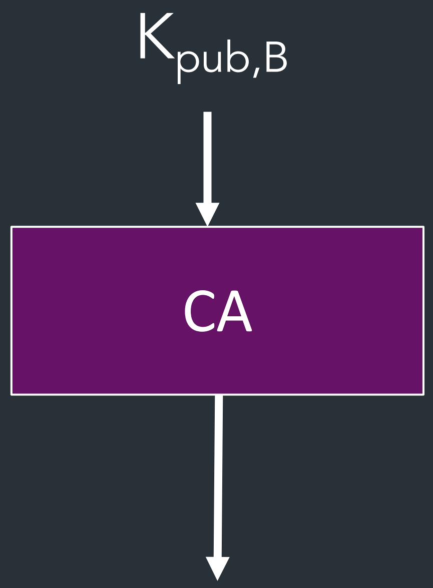
$$s = \text{Sign}(K_{priv,CA}, \{K_{pub,B}, \ldots\})$$

$$\text{Cert}_B = \{K_{pub,B}, \text{metadata}, s\}$$

Warmup

What happens if attacker obtains Kpriv,B?
What about Kpriv,CA?





$s = \text{Sign}(K_{priv,CA}, \{K_{pub,B}, \dots\})$

$\text{Cert}_B = \{K_{pub,B}, \text{metadata}, s\}$

Today's Lecture

- More about Tor
- Wrapup

Q: If private key is compromised, can attacker decrypt data?

Q: If private key is compromised, can attacker decrypt data?

Not if TLS connection uses forward secrecy

⇒ Cannot recover session key if server private key leaked

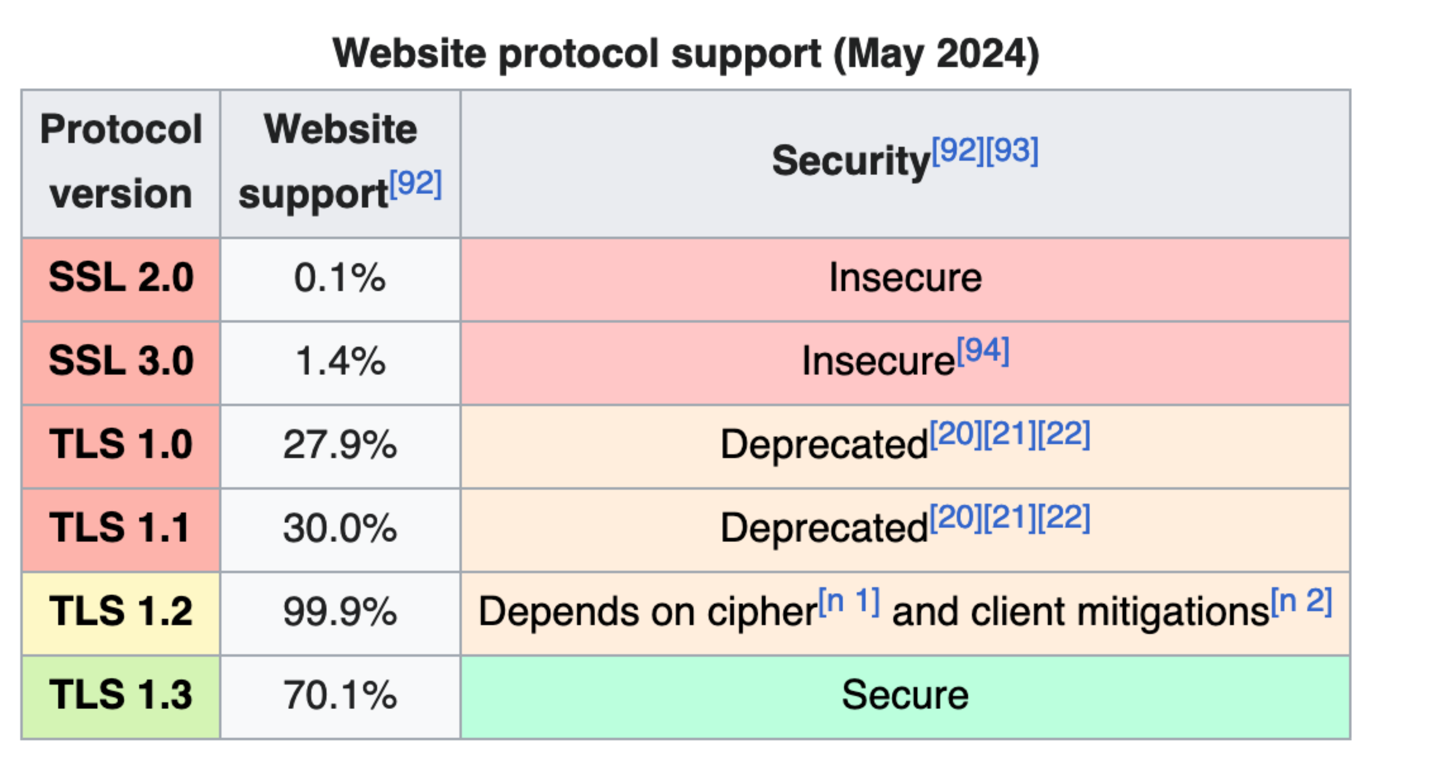
⇒ Once optional, now required by TLS 1.3 (2018)

Q: If private key is compromised, can attacker decrypt data?

Not if TLS connection uses forward secrecy

⇒ Cannot recover session key if server private key leaked

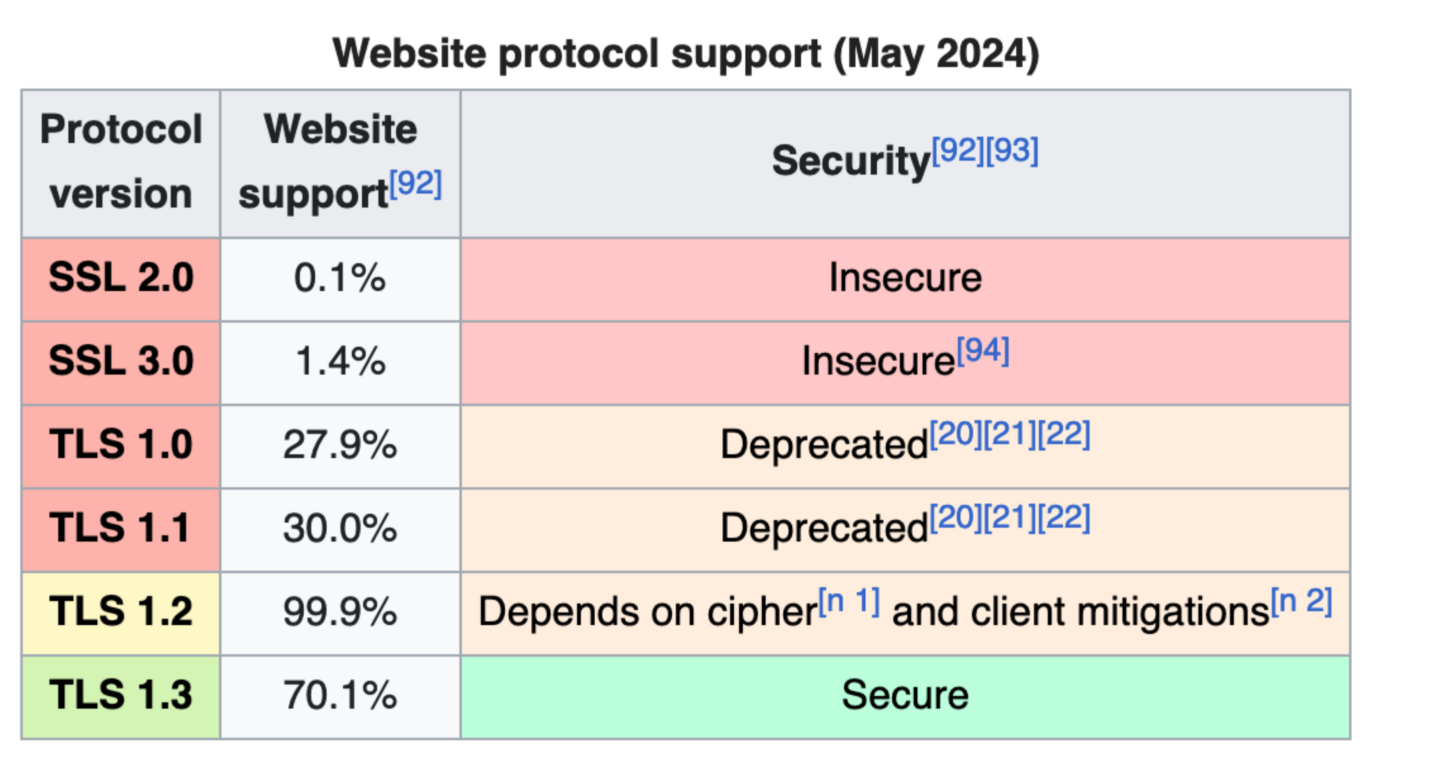
⇒ Once optional, now required by TLS 1.3 (2018)



Website protocol support (May 2024)

Protocol version	Website support[92]	Security[92][93]
SSL 2.0	0.1%	Insecure
SSL 3.0	1.4%	Insecure[94]
TLS 1.0	27.9%	Deprecated[20][21][22]
TLS 1.1	30.0%	Deprecated[20][21][22]
TLS 1.2	99.9%	Depends on cipher[n 1] and client mitigations[n 2]
TLS 1.3	70.1%	Secure

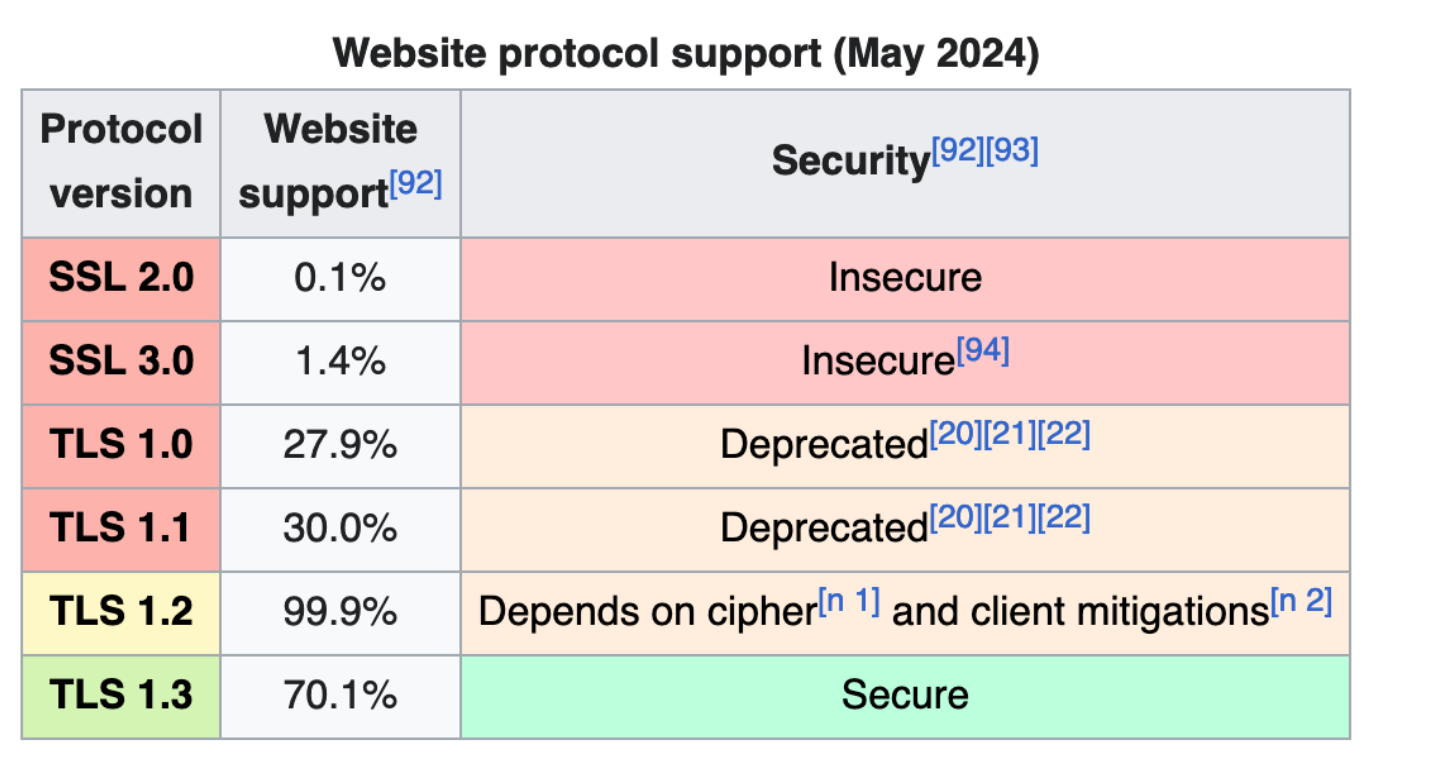
In practice, TLS 1.3 rollout delayed by many broken TLS implementations (eg. in-network middleboxes/proxies) …



Website protocol support (May 2024)

Protocol version	Website support[92]	Security[92][93]
SSL 2.0	0.1%	Insecure
SSL 3.0	1.4%	Insecure[94]
TLS 1.0	27.9%	Deprecated[20][21][22]
TLS 1.1	30.0%	Deprecated[20][21][22]
TLS 1.2	99.9%	Depends on cipher[n 1] and client mitigations[n 2]
TLS 1.3	70.1%	Secure

In practice, TLS 1.3 rollout delayed by many broken TLS implementations (eg. in-network middleboxes/proxies) …

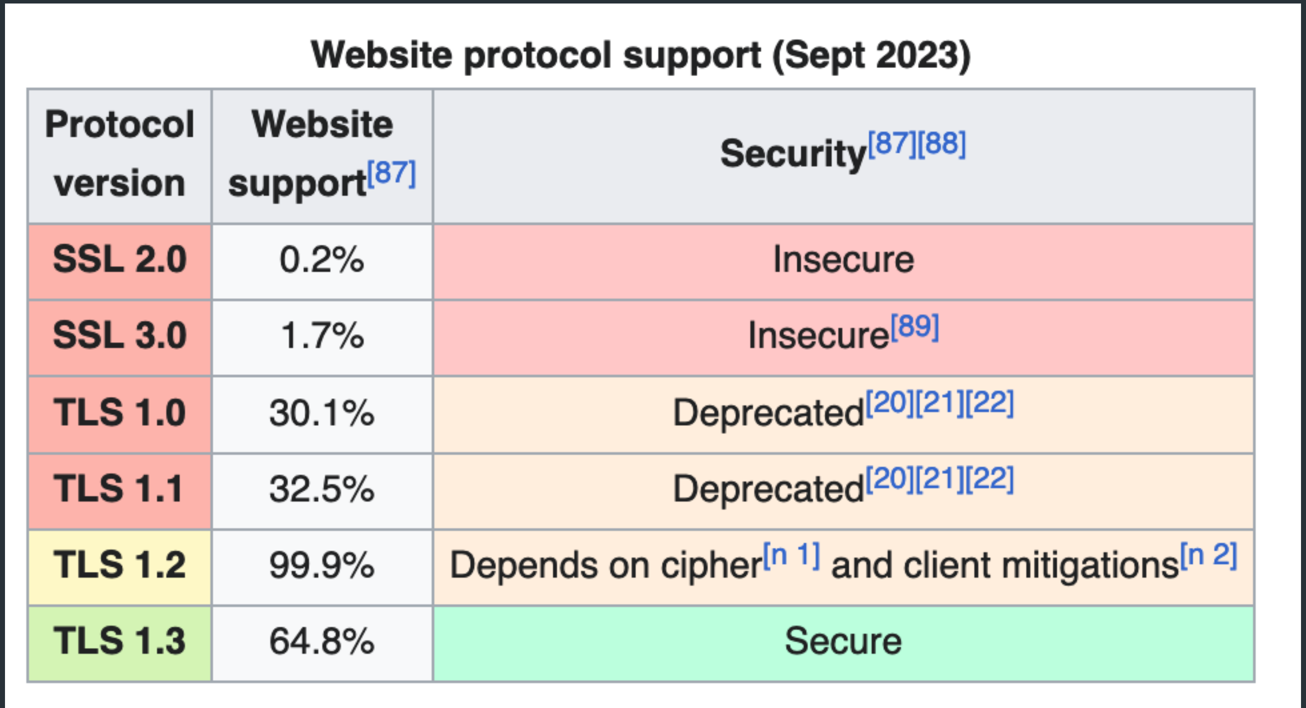


Website protocol support (May 2024)

Protocol version	Website support[92]	Security[92][93]
SSL 2.0	0.1%	Insecure
SSL 3.0	1.4%	Insecure[94]
TLS 1.0	27.9%	Deprecated[20][21][22]
TLS 1.1	30.0%	Deprecated[20][21][22]
TLS 1.2	99.9%	Depends on cipher[n 1] and client mitigations[n 2]
TLS 1.3	70.1%	Secure

In practice, TLS 1.3 rollout delayed by many broken TLS implementations (eg. in-network middleboxes/proxies) …

Remember how we said don’t propagate buggy behavior in TCP?



Website protocol support (Sept 2023)

Protocol version	Website support[87]	Security[87][88]
SSL 2.0	0.2%	Insecure
SSL 3.0	1.7%	Insecure[89]
TLS 1.0	30.1%	Deprecated[20][21][22]
TLS 1.1	32.5%	Deprecated[20][21][22]
TLS 1.2	99.9%	Depends on cipher[n 1] and client mitigations[n 2]
TLS 1.3	64.8%	Secure

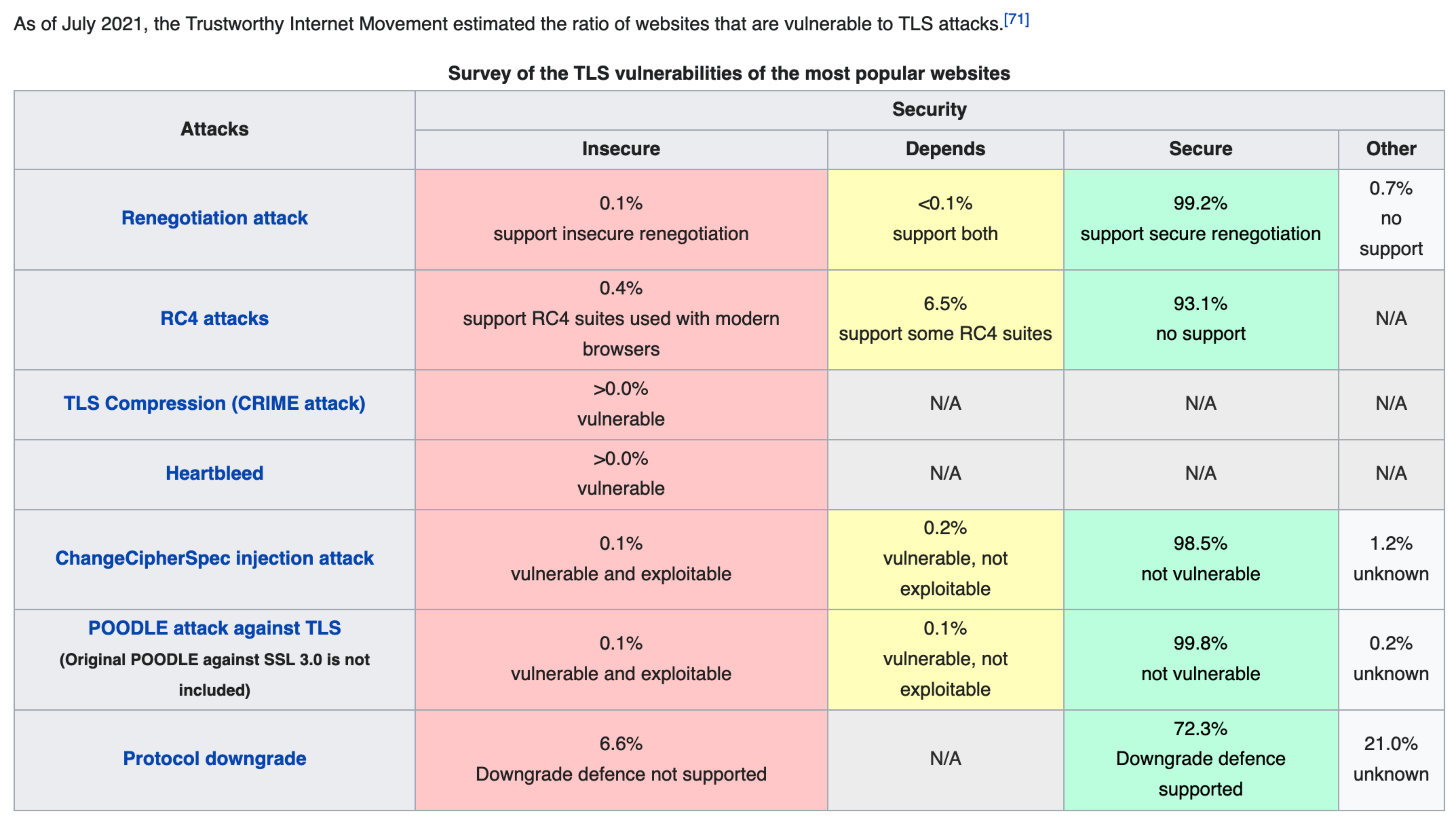
In general, implementing security protocols is hard to get right

In general, implementing security protocols is hard to get right

=> TLS libraries are very critical and need lots of oversight/auditing

=> Servers (and clients) need to be updated with latest standards/fixes

Se



As of July 2021, the Trustworthy Internet Movement estimated the ratio of websites that are vulnerable to TLS attacks.[71]

Survey of the TLS vulnerabilities of the most popular websites

Attacks	Security			
	Insecure	Depends	Secure	Other
Renegotiation attack	0.1% support insecure renegotiation	<0.1% support both	99.2% support secure renegotiation	0.7% no support
RC4 attacks	0.4% support RC4 suites used with modern browsers	6.5% support some RC4 suites	93.1% no support	N/A
TLS Compression (CRIME attack)	>0.0% vulnerable	N/A	N/A	N/A
Heartbleed	>0.0% vulnerable	N/A	N/A	N/A
ChangeCipherSpec injection attack	0.1% vulnerable and exploitable	0.2% vulnerable, not exploitable	98.5% not vulnerable	1.2% unknown
POODLE attack against TLS (Original POODLE against SSL 3.0 is not included)	0.1% vulnerable and exploitable	0.1% vulnerable, not exploitable	99.8% not vulnerable	0.2% unknown
Protocol downgrade	6.6% Downgrade defence not supported	N/A	72.3% Downgrade defence supported	21.0% unknown

Wikipedia table, source: https://www.ssllabs.com/ssl-pulse/

So, are we good?

If we use TLS, is it enough?

Overall, depends on your threat model…

- Server still knows who you are, even if connection is encrypted

- Even encrypted traffic leaks information!

Overall, depends on your threat model…

- Server still knows who you are, even if connection is encrypted

=> IPs can be traced to location (to varying levels of precision)

=> Your browser may leak info (cookies, mouse usage, etc.)

- Even encrypted traffic leaks information!

=> Name of server: DNS, Server Name Indicator (SNI)

=> Traffic patterns (timing of packets, protocols, …)

Securing the transport layer not enough => info leaks based on other layers

Why?

- Avoiding censorship
- Avoiding surveillance (by person, or an organization)
- Anonymous reporting (journalists, whistleblowers)

Why?

- Avoiding censorship
- Avoiding surveillance (by person, or an organization)
- Anonymous reporting (journalists, whistleblowers)

Room 641A: alleged wiretapping room in a datacenter for an Internet backbone…
https://en.wikipedia.org/wiki/Room_641A

How can we deal with this?

Mechanisms to provide more security at the network layer

How can we deal with this?

Mechanisms to provide more security at the network layer

⇒ Security for all your network traffic => not just one 5-tuple

⇒ Can (try to) provide more anonymity

VPN: secure tunnel for network traffic

=> Connect a host to a private network

Virtual Private Network (VPN)

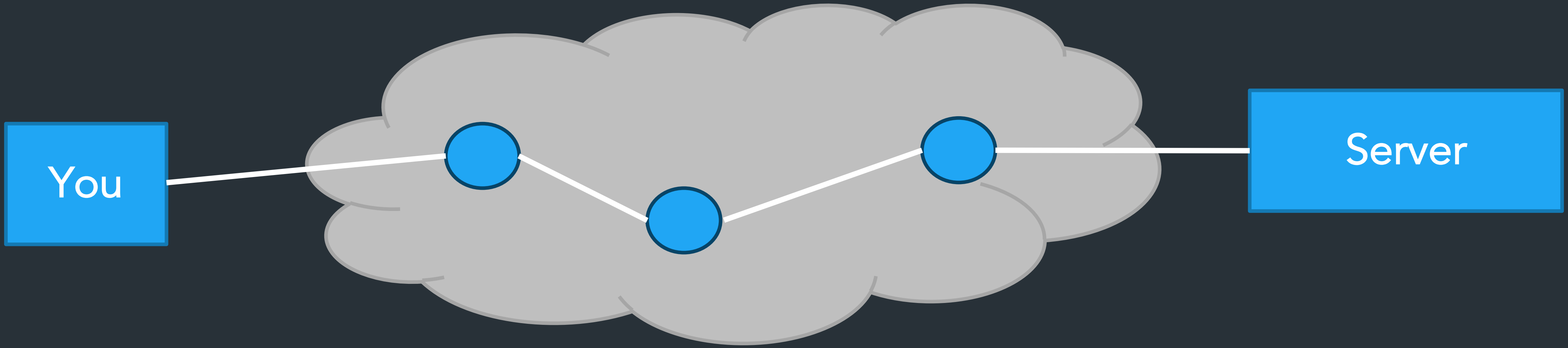
Secure tunnel for arbitrary network traffic (any IP packets)

Use for

=> Accessing a private network (remote access internal network)

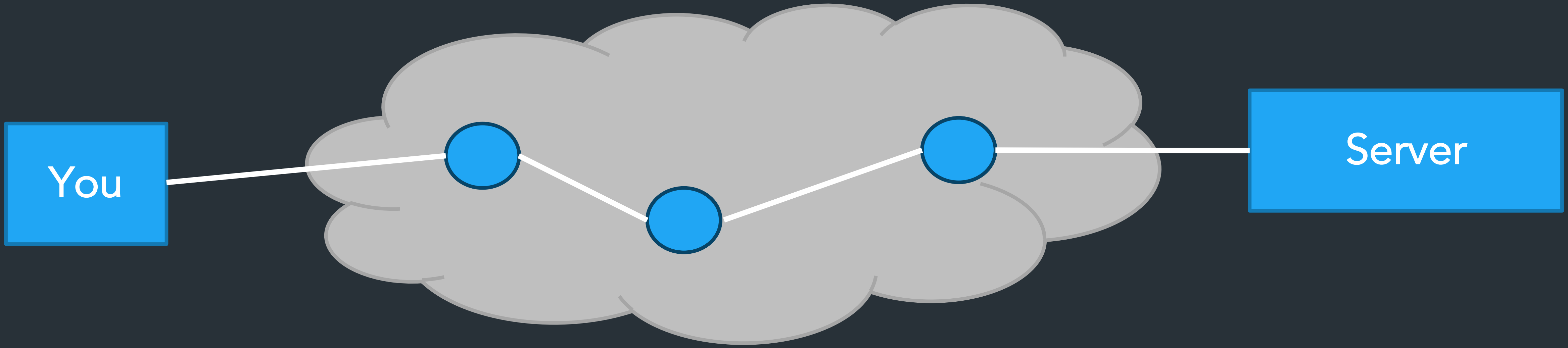
=> Secure proxy for your traffic: traffic appears to originate from VPN server

Problems?

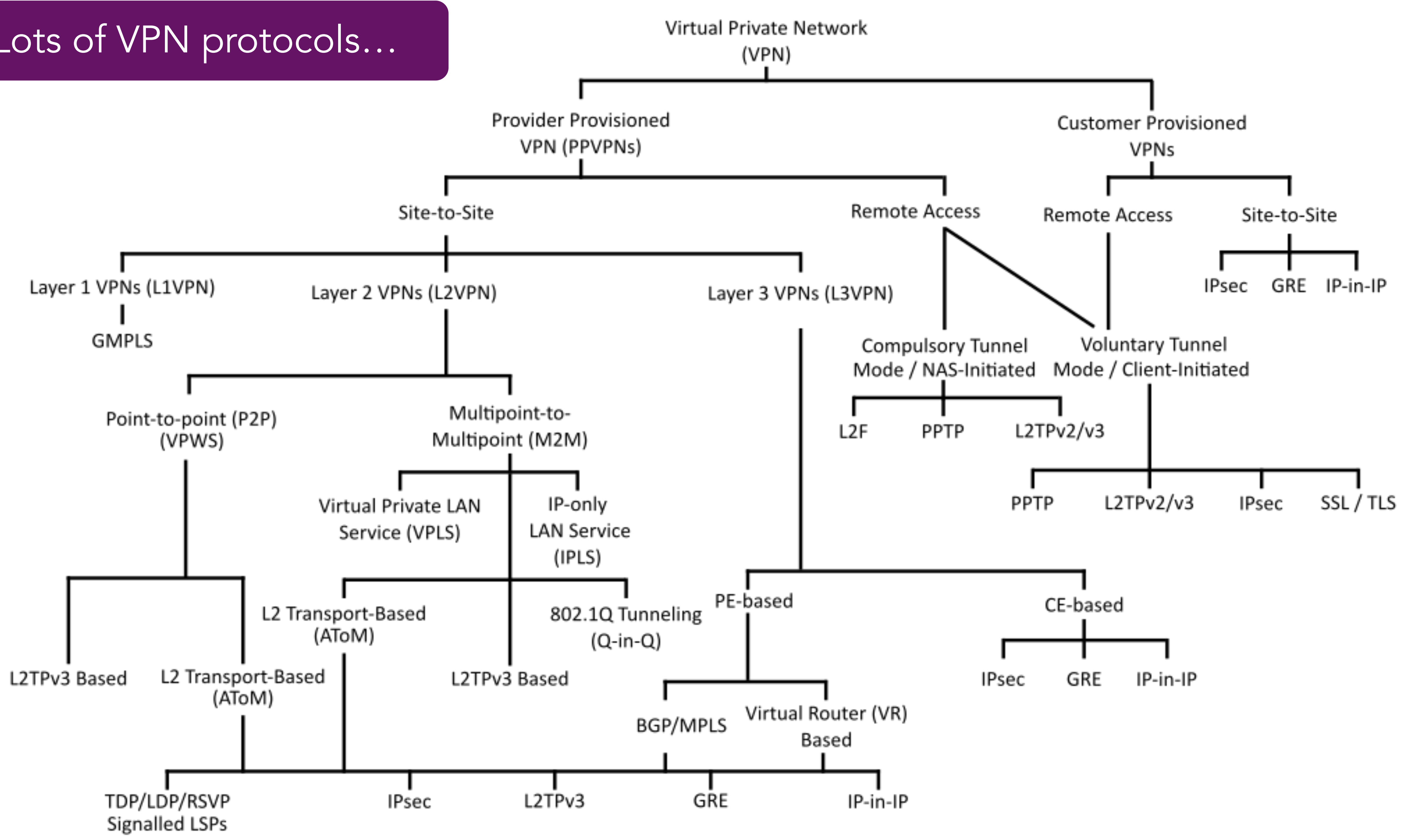


VPN: secure tunnel for network traffic

=> Connect a host to a private network



Lots of VPN protocols…


Virtual Private Network (VPN)
Provider Provisioned VPN (PPVPNs)
Customer Provisioned VPNs
Site-to-Site
Remote Access
Remote Access
Site-to-Site
IPsec
GRE
IP-in-IP
Layer 1 VPNs (L1VPN)
Layer 2 VPNs (L2VPN)
Layer 3 VPNs (L3VPN)
GMPLS
Compulsory Tunnel Mode / NAS-Initiated
Voluntary Tunnel Mode / Client-Initiated
L2F
PPTP
L2TPv2/v3
PPTP
L2TPv2/v3
IPsec
SSL / TLS
Point-to-point (P2P) (VPWS)
Multipoint-to-Multipoint (M2M)
Virtual Private LAN Service (VPLS)
IP-only LAN Service (IPLS)
L2 Transport-Based (AToM)
802.1Q Tunneling (Q-in-Q)
PE-based
CE-based
L2TPv3 Based
L2 Transport-Based (AToM)
L2TPv3 Based
IPsec
GRE
IP-in-IP
BGP/MPLS
Virtual Router (VR) Based
TDP/LDP/RSVP Signalled LSPs
IPsec
L2TPv3
GRE
IP-in-IP

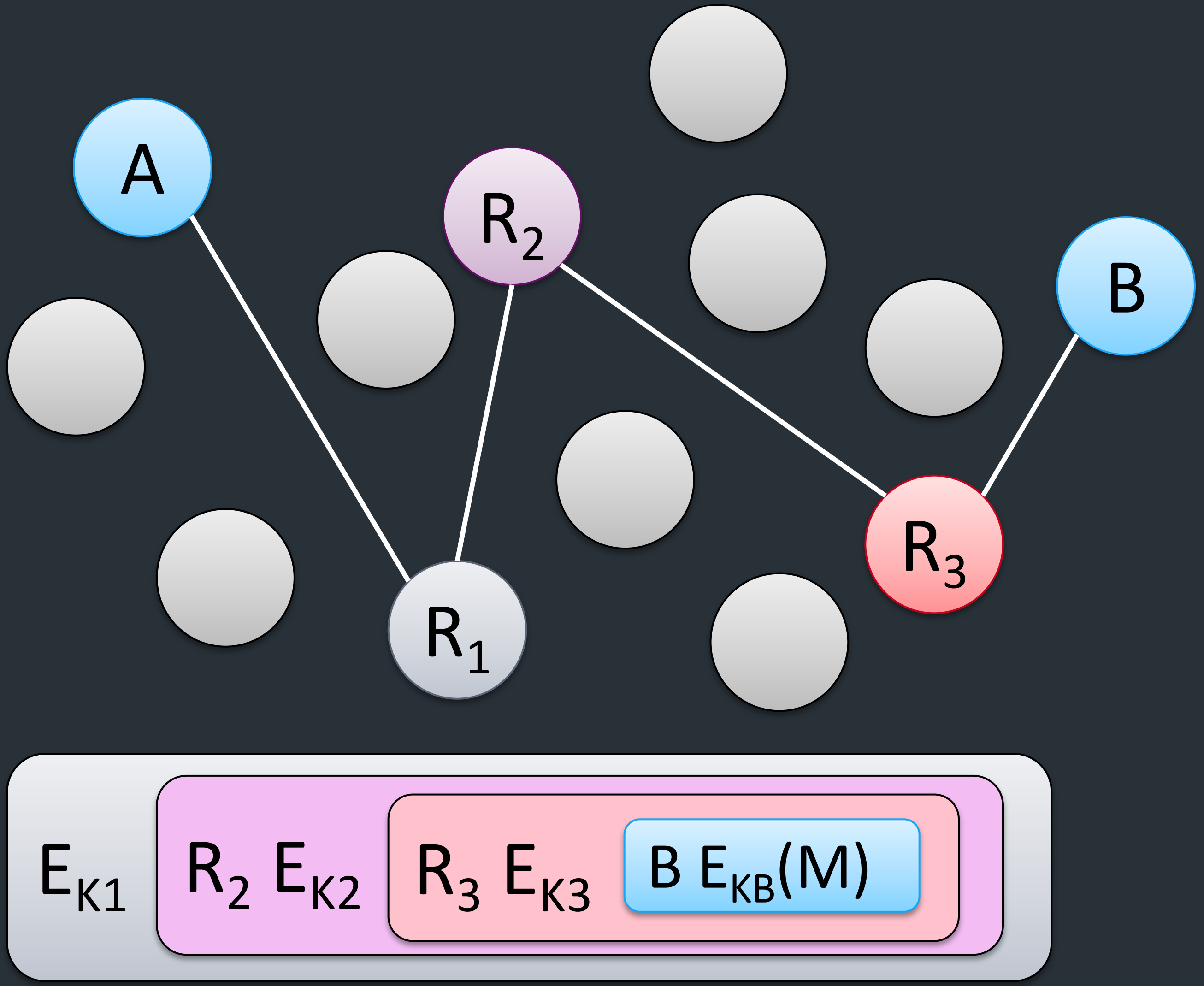
Can we do better?

Tor

- Onion routing service: build encrypted circuit on tor relay network
- Network of relays, mainly operated by volunteers
- Started in 1990s from Naval Research Lab, now maintained by The Tor Project (a non-profit)

Onion Routing

- Layered encryption
  - Build onion inside out
- Routing
  - Peel onion outside in
- Each router knows only previous and next



E_{K1} R_2 E_{K2} R_3 E_{K3} B $E_{KB}(M)$

What if the server wants to help?

Onion services: server connects to tor directly => no need for an exit node!

What if the server wants to help?

Onion services: server connects to tor directly => no need for an exit node!

- Accessible via .onion domain: special DNS TLD not in root zone
- Site addresses based on public key of server, client looks up using distributed hash table (DHT)

What if the server wants to help?

Onion services: server connects to tor directly => no need for an exit node!

- Accessible via .onion domain: special DNS TLD not in root zone
- Site addresses based on public key of server, client looks up using distributed hash table (DHT)

Examples

- *New York Times:*
  `https://www.nytimesn7cgmftshazwhfgzm37qxb44r64ytbb2dj3x62d2lljsciiyd.onion`
- Facebook
  `https://facebookwkhpilnemxj7asaniu7vnjjbiltxjqhye3mhbshg7kx5tfyd.onion`
- Cloudflare public DNS
  `dns4torpnlfs2ifuz2s2yf3fc7rdmsbhm6rw75euj35pac6ap25zgqad.onion`

Wrapping up

- This is our last formal lecture
- From here: work on final project

What I hope you have learned

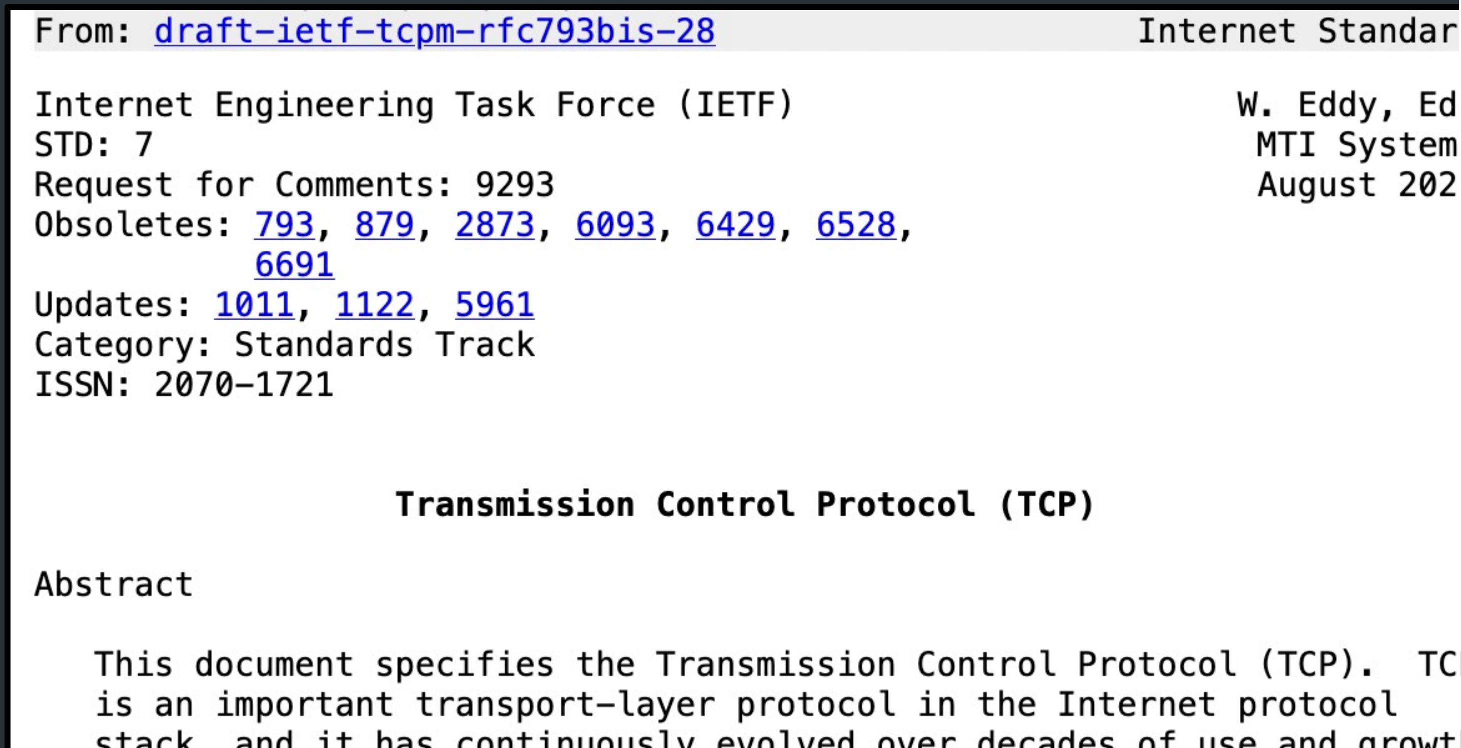
We can't cover (or remember) everything

Hope you learn important tools/principles to understand networking challenges you encounter

Protocols Ways to communicate between *heterogeneous* systems

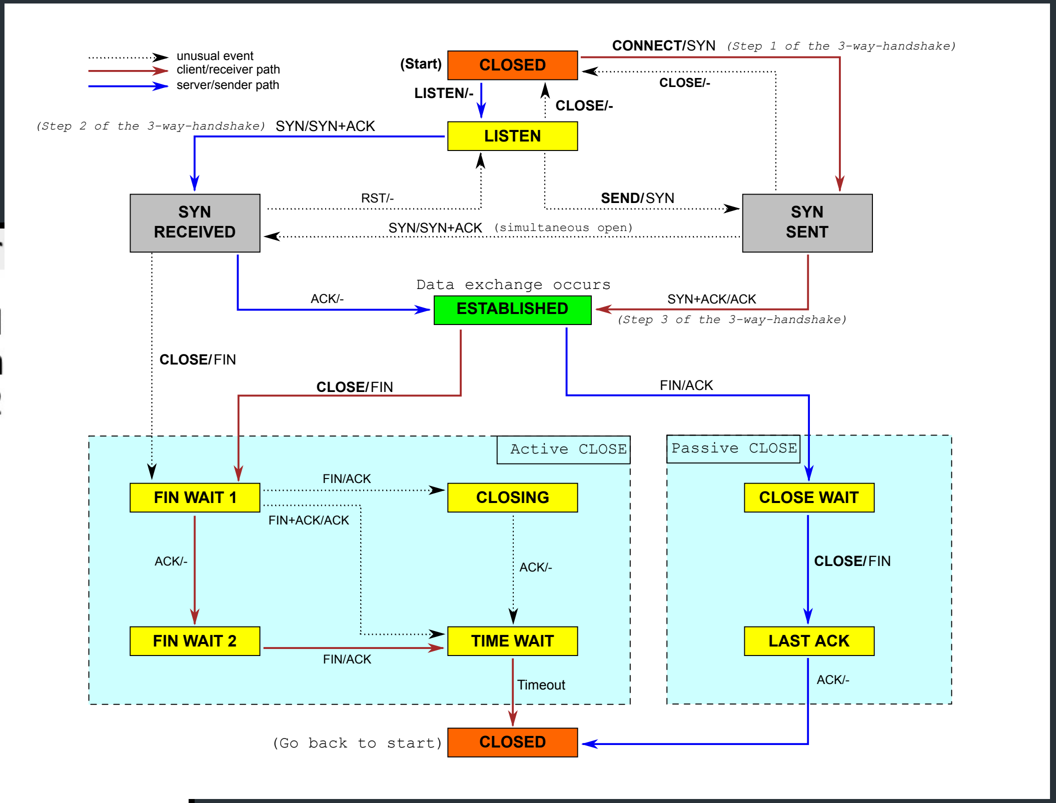
Network programming

```
conn, err :=  net.Dial(“tcp”, “10.0.0.1:80”)
. . .

someBuf := make([]byte, . . .)
conn.Write(someBuf)
```



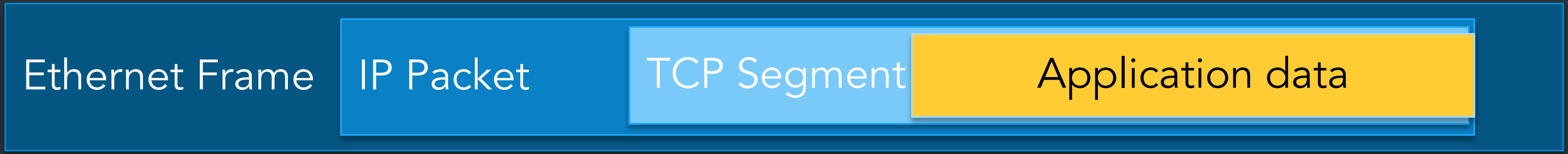
From: draft-ietf-tcpm-rfc793bis-28 Internet Standar

```
Internet Engineering Task Force (IETF)                    W. Eddy, Ed
STD: 7                                                    MTI System
Request for Comments: 9293                                August 202
Obsoletes: 793, 879, 2873, 6093, 6429, 6528,
           6691
Updates: 1011, 1122, 5961
Category: Standards Track
ISSN: 2070-1721


                Transmission Control Protocol (TCP)

Abstract

   This document specifies the Transmission Control Protocol (TCP).  TCP
   is an important transport-layer protocol in the Internet protocol
   stack, and it has continuously evolved over decades of use and growth
```



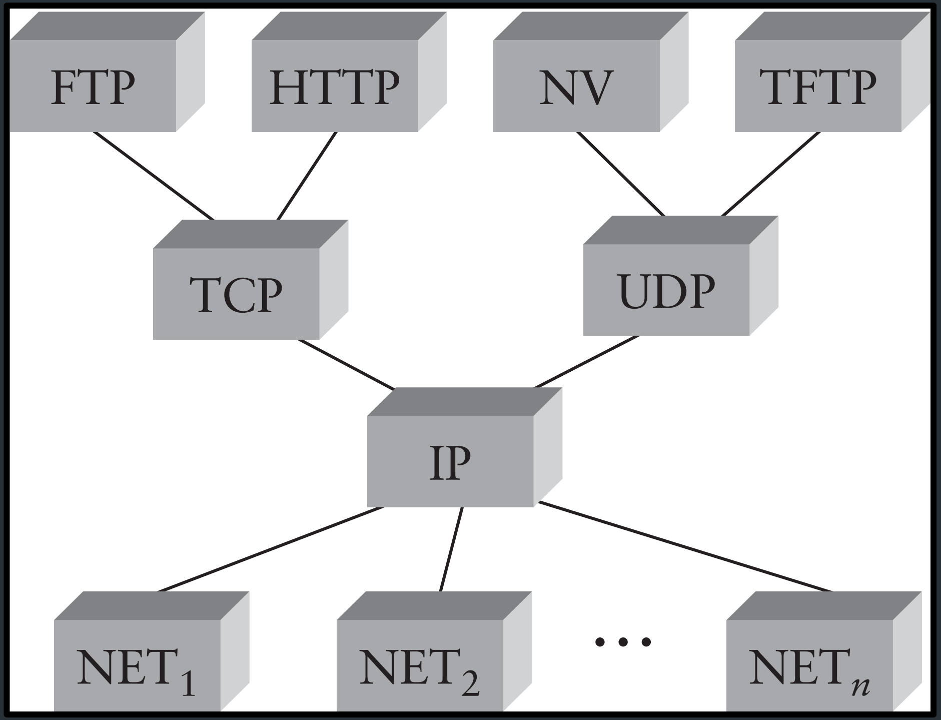
Layering / Encapsulation

Building abstractions and interfaces to hide lower-level details from “higher” layers



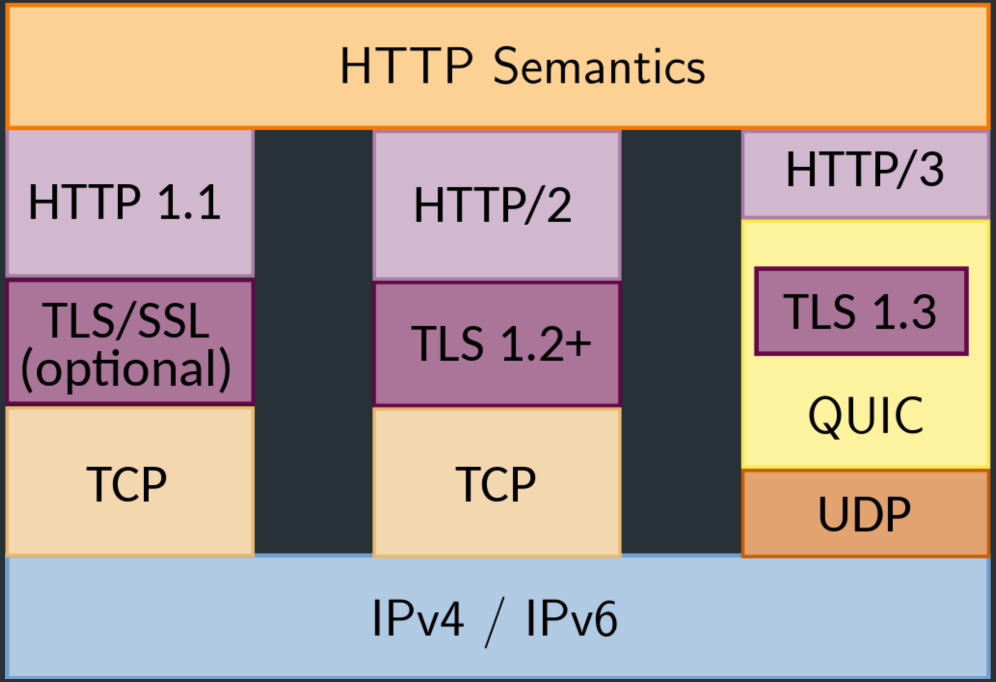
Abstractions are great!

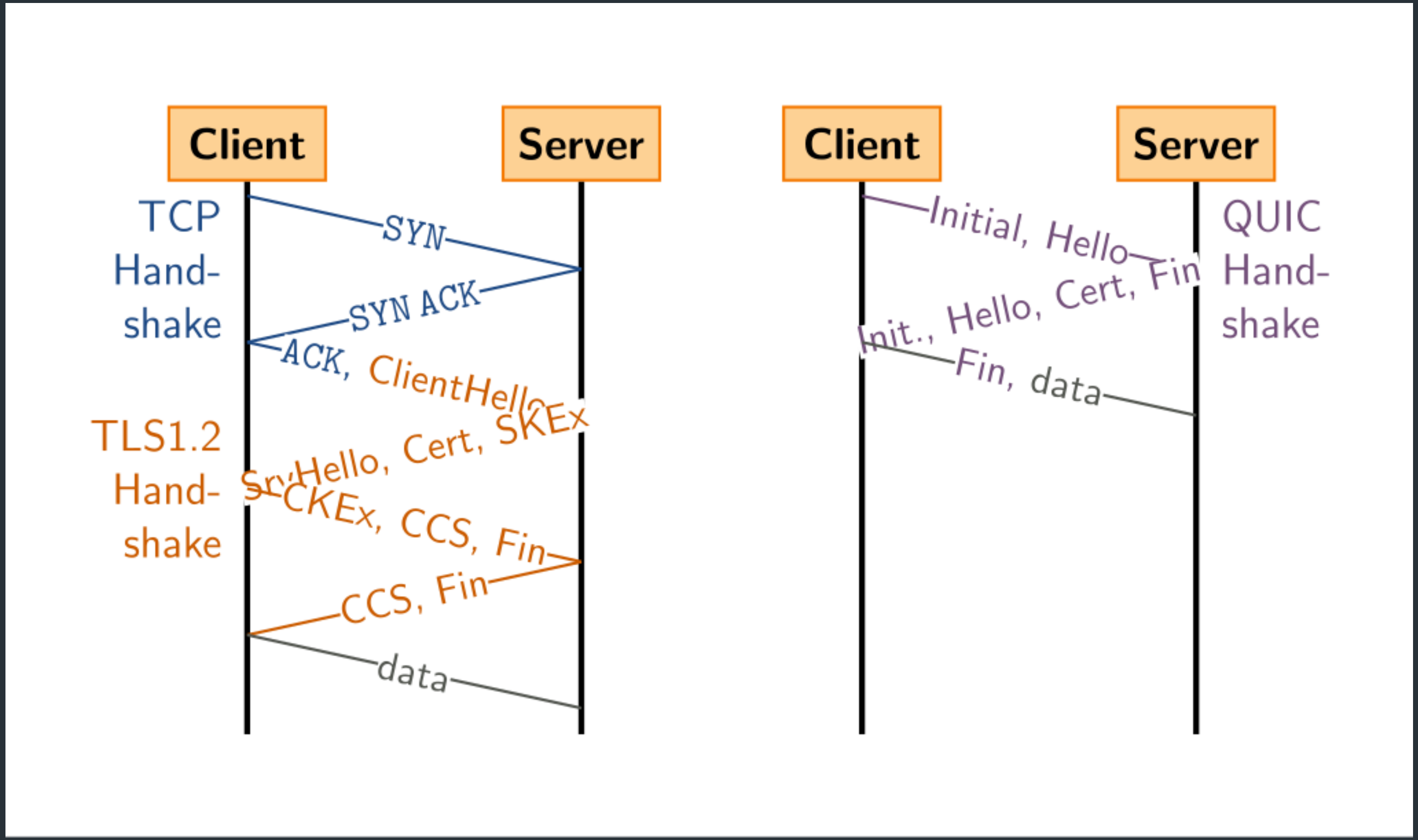
- Can support huge variety of devices, protocols
- Allows independent evolution => new protocols!



… until they aren’t

Sometimes, need to break them





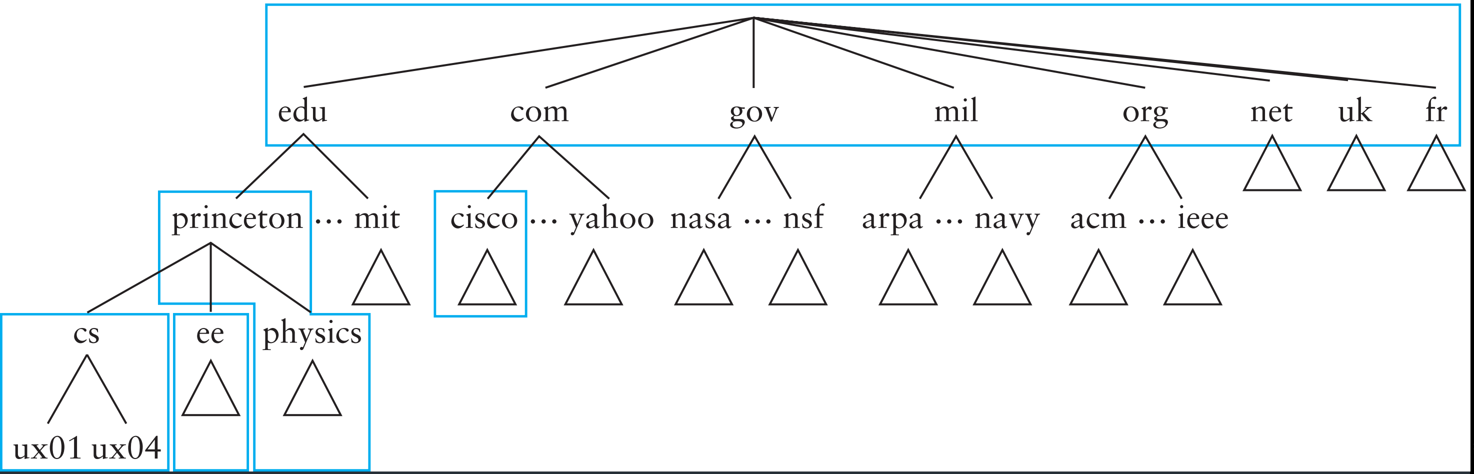
Naming

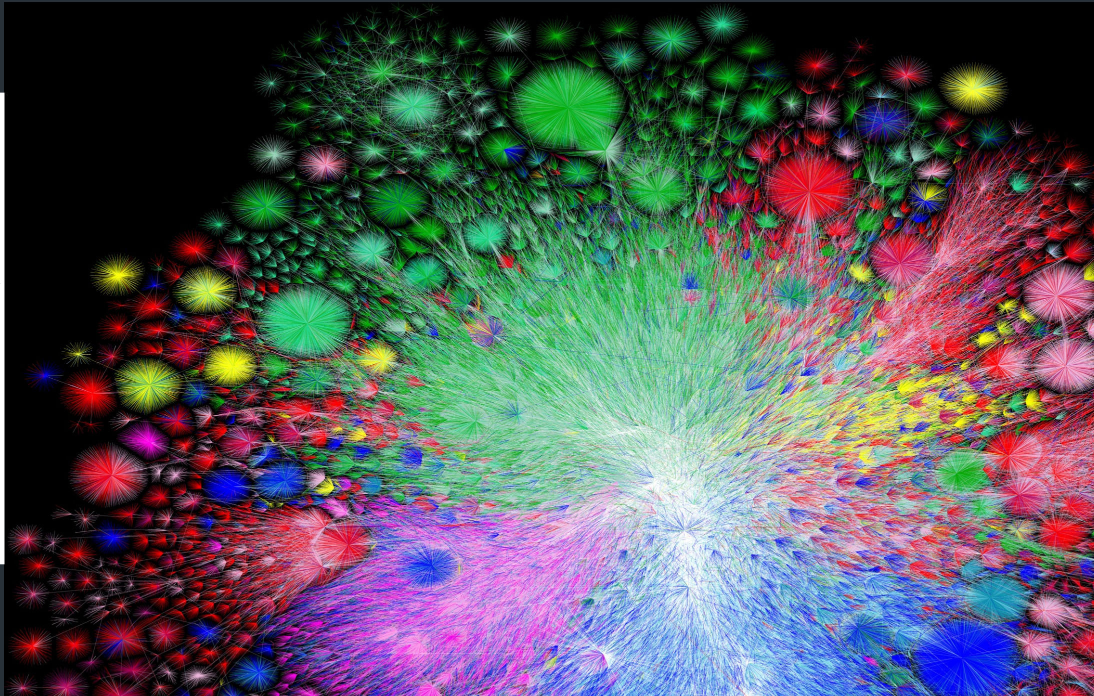
Indirection: abstract low-level info with a higher-level name

=> Human-readable DNS names

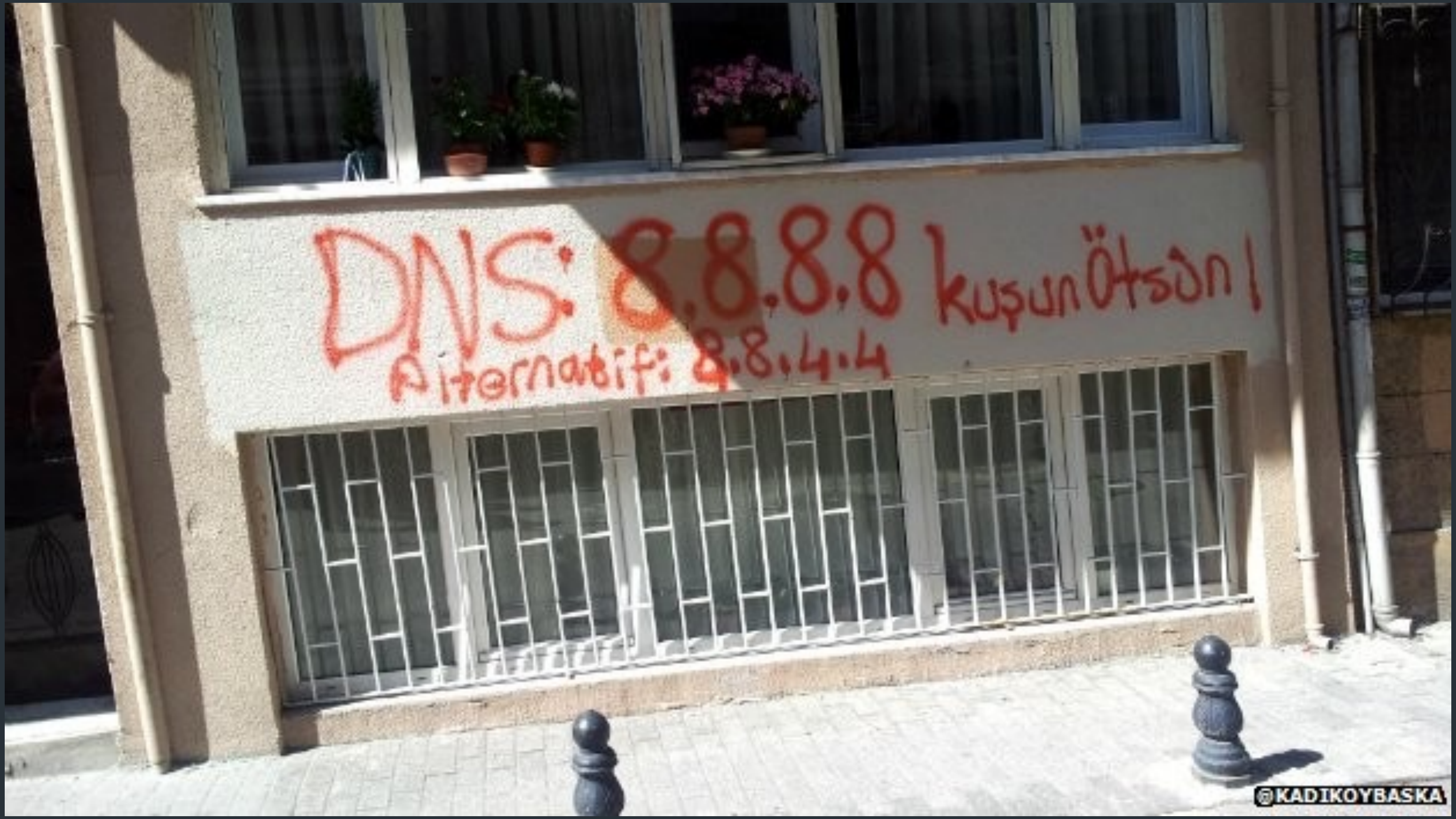
=> Scalability: redundancy, proxies, load balancing

Can leverage hierarchy of naming => scalability (IP, DNS, …)





How naming, etc. can be controlled…



Changing DNS servers in response to blocking of Twitter in Turkey (2014)

Writeup, with more links: https://www.thousandeyes.com/blog/internet-censorship-around-the-world

Lots of challenges out there

Our Internet architecture was designed in the 1980s, where modern scale and complexity was unimaginable

Lots of challenges out there

Our Internet architecture was designed in the 1980s, where modern scale and complexity was unimaginable

Now…

- No one knows how big the Internet is
- No one is in charge
- Anyone can add any application
- Packets traverse many paths, countries, regulatory domains

Thank you!
Please stay in touch!