CSCI-1680
How to (try) to be anonymous

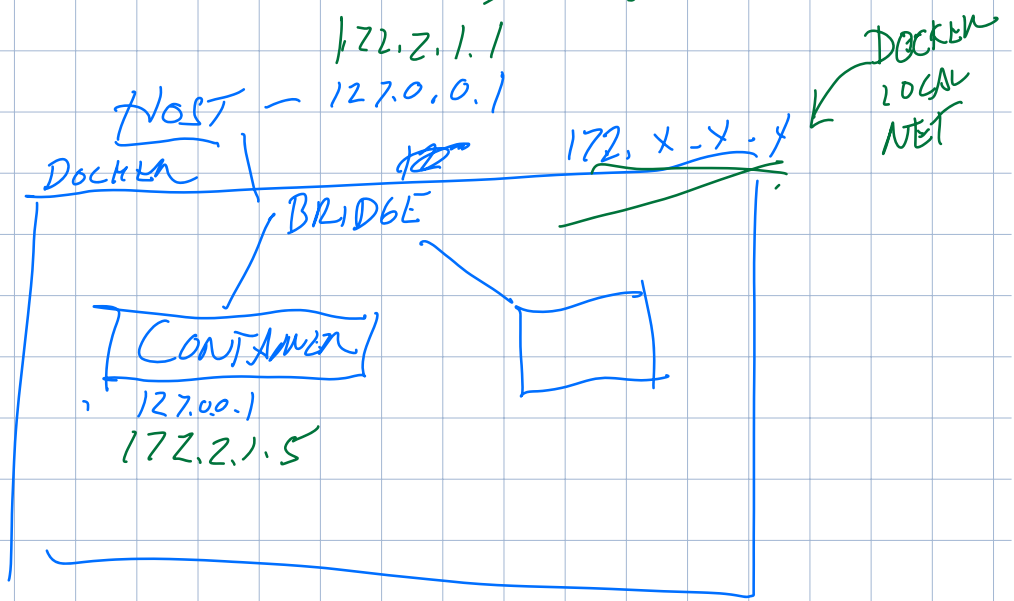
Nick DeMarinis

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

Administrivia

- Final project: proposal feedback on Gradescope
- HW4 (short): due Friday, 12/8
- Most office hours end Friday, some updates this week
  - After 12/8: I will still have hours, but schedule my differ => see calendar

CONTAINER NETWORKING (ASIDE)

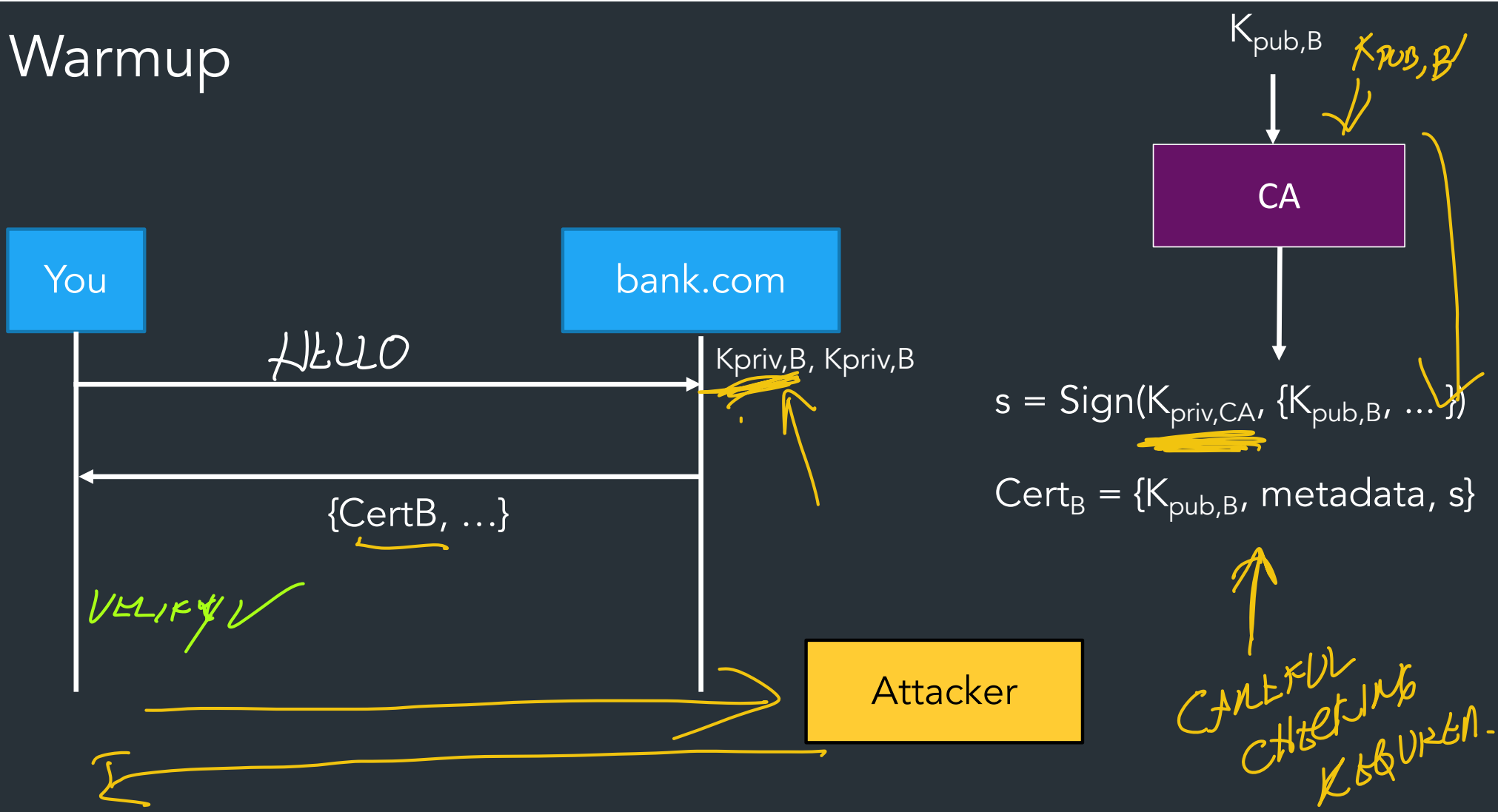


IDEA: CONTAINERS IN ISOLATED NETWORK

- ONLY EXPOSE CONTAINER PORTS ON REQUEST.

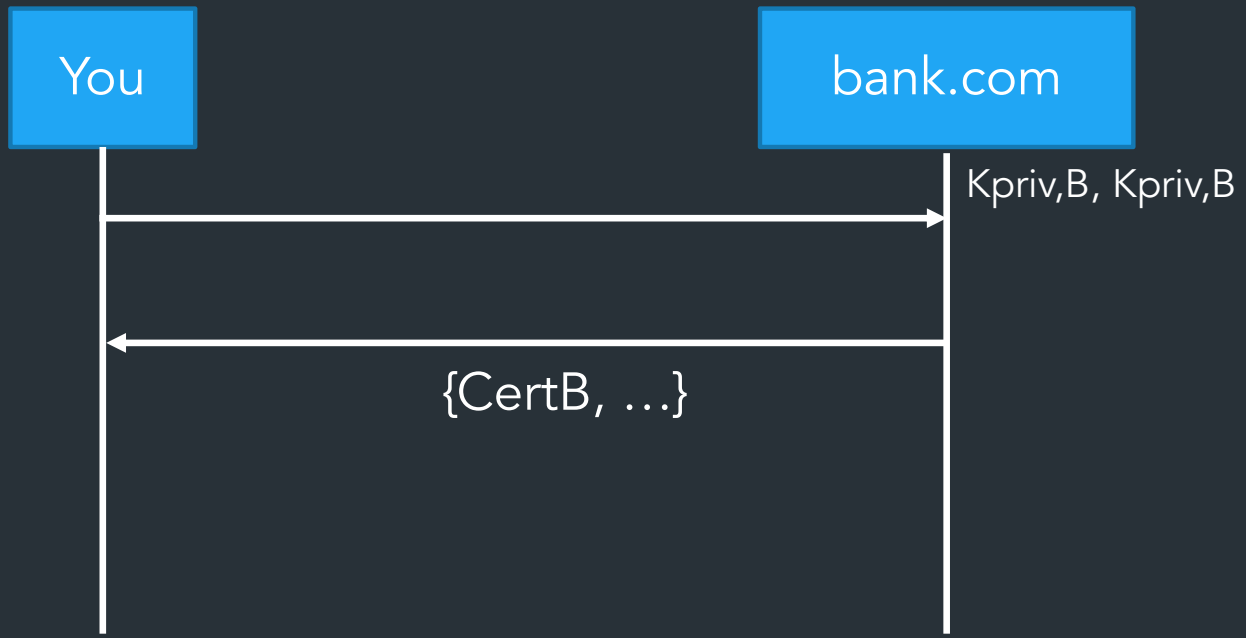
⇒ VSCODE DOES THIS FOR YOU!

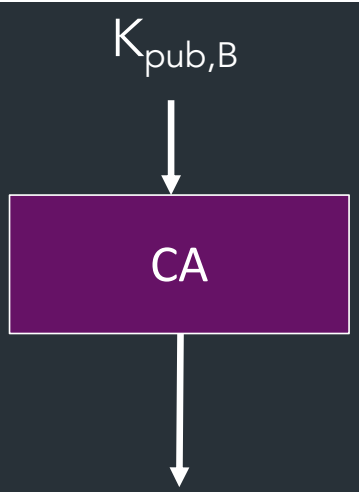
Warmup


You
bank.com
HELLO
Kpriv,B, Kpriv,B
{CertB, …}
VERIFY
Attacker
$K_{pub,B}$
CA
$s = Sign(K_{priv,CA}, \{K_{pub,B}, \ldots\})$
$Cert_B = \{K_{pub,B}, metadata, s\}$
CAREFUL CHECKING KEYS.

Warmup

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

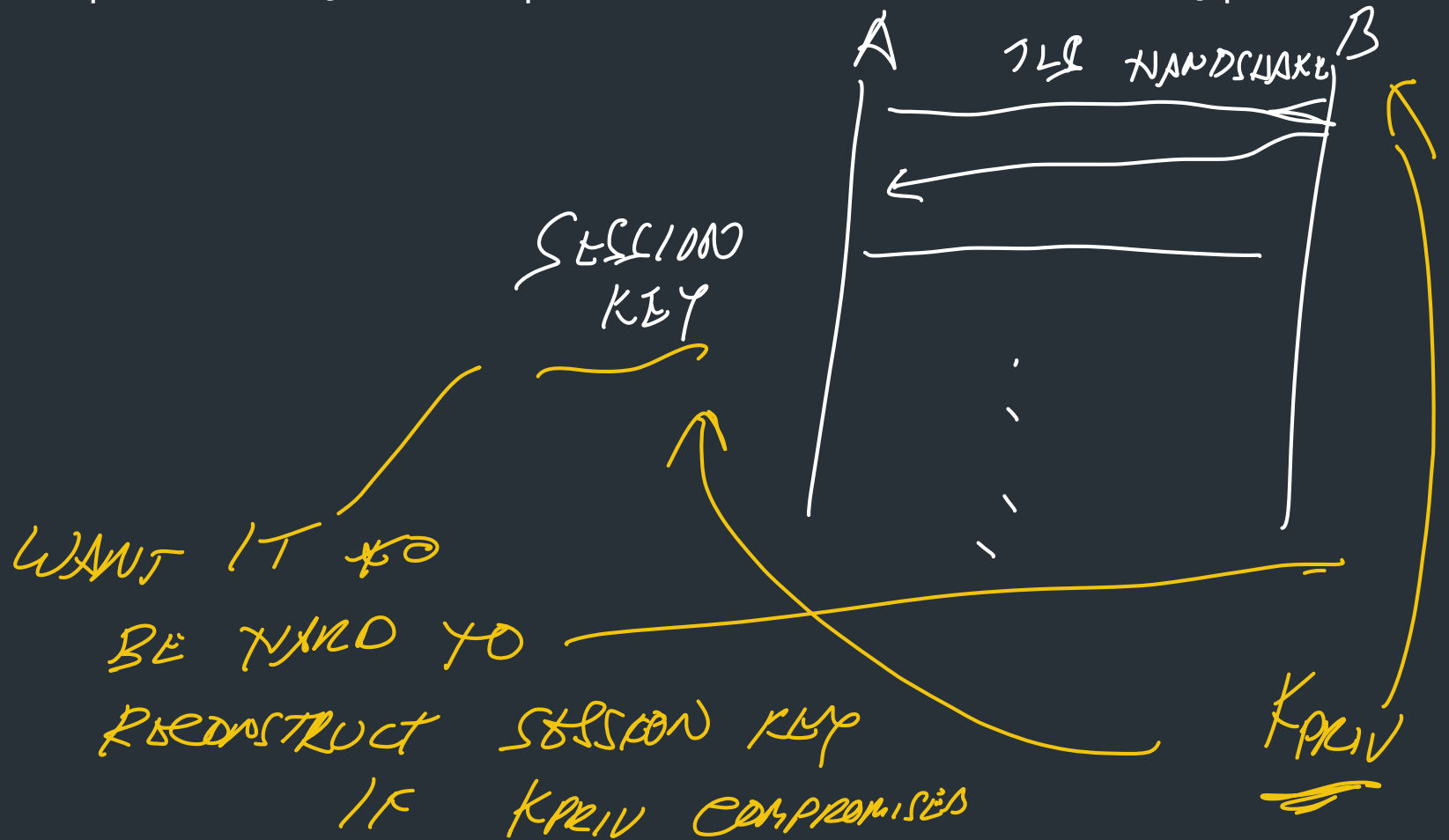




$s = Sign(K_{priv,CA}, \{K_{pub,B}, \ldots\})$

$Cert_B = \{K_{pub,B}, metadata, s\}$

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)

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 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?

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

=> NEED TO STAY WITHIN BEST PRACTICES.

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/

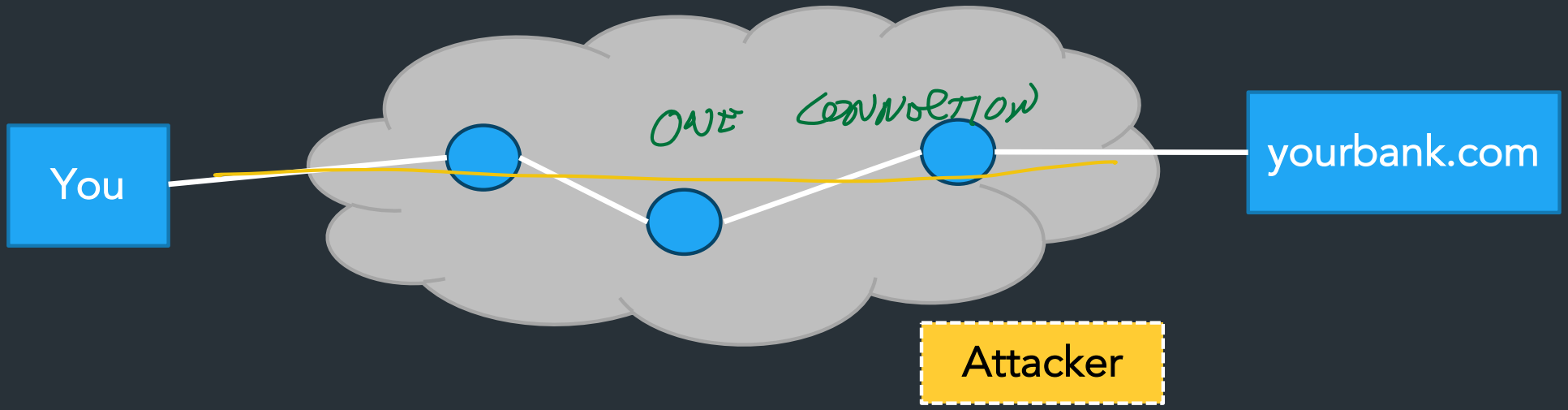
Rogue Certificates?

- In 2011, DigiNotar, a Dutch root certificate authority, was compromised
- The attacker created rogue certificates for popular domains like google.com and yahoo.com
- DigiNotar was distrusted by browsers and filed for bankruptcy
- See the incident investigation report by Fox-IT

- In 2017, Google questioned the certificate issuance policies and practices of Symantec
- Google's Chrome would start distrusting Symantec's certificates unless certain remediation steps were taken
- See back and forth between Ryan Sleevi (Chromium team) and Symantec
- The matter was settled with DigiCert acquiring Symantec's certificate business

So are we good?

If we use TLS, is it enough?


You
One connection
yourbank.com
Attacker

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)

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

How can we deal with this?

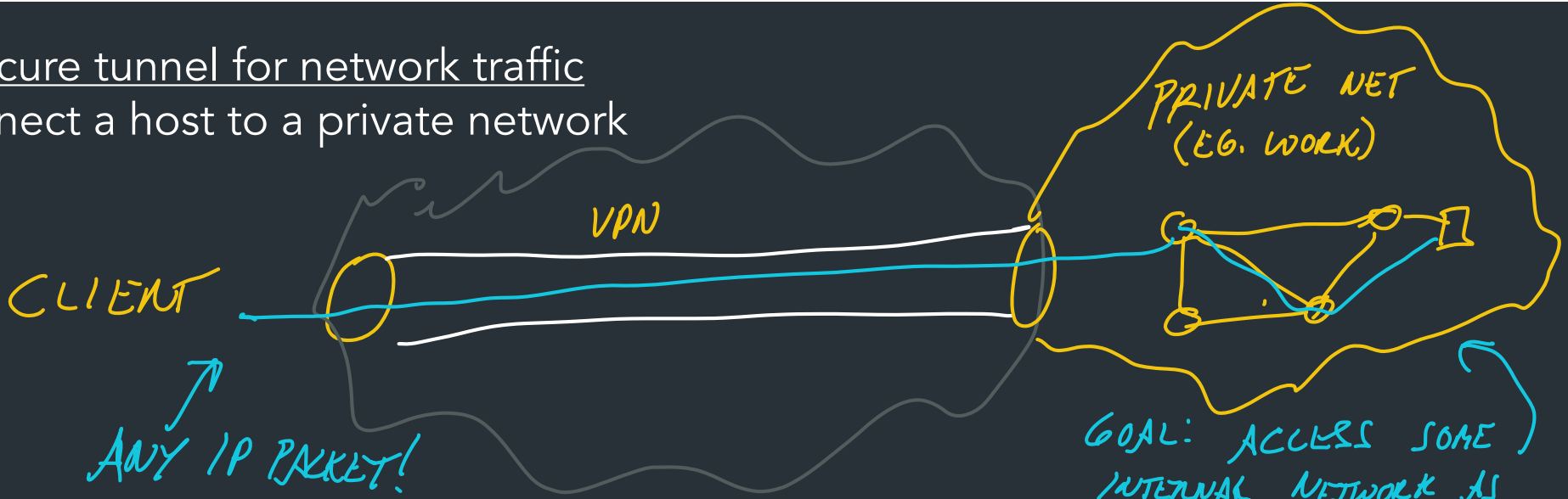
Mechanisms to provide more security at the network layer

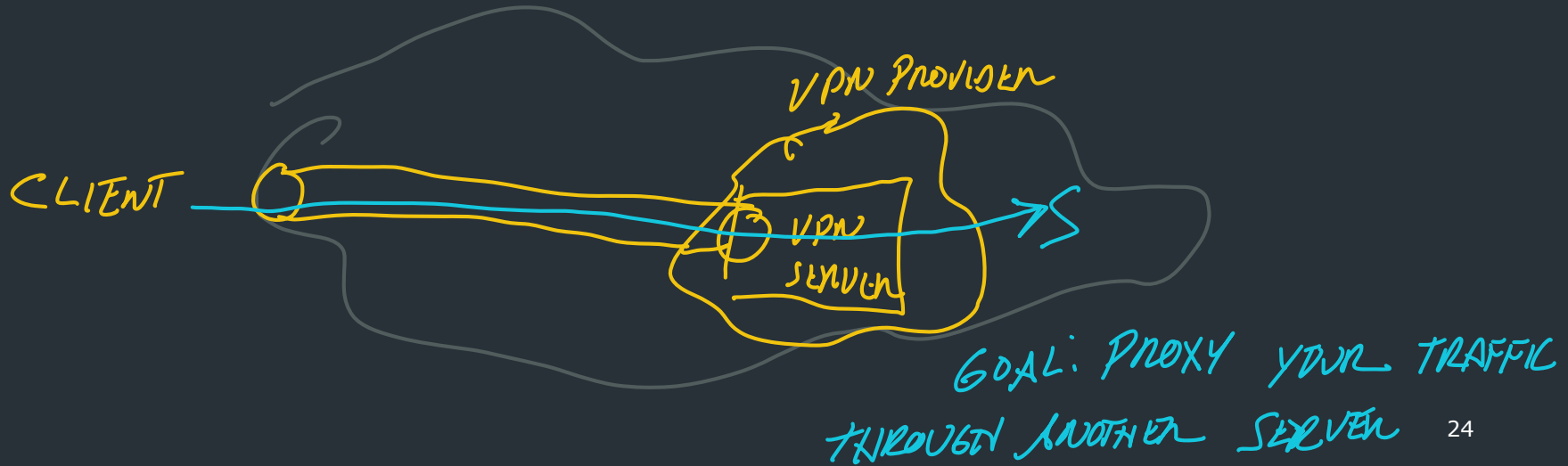
⇒ Security for all your network traffic => not just one 5-tuple

⇒ 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)

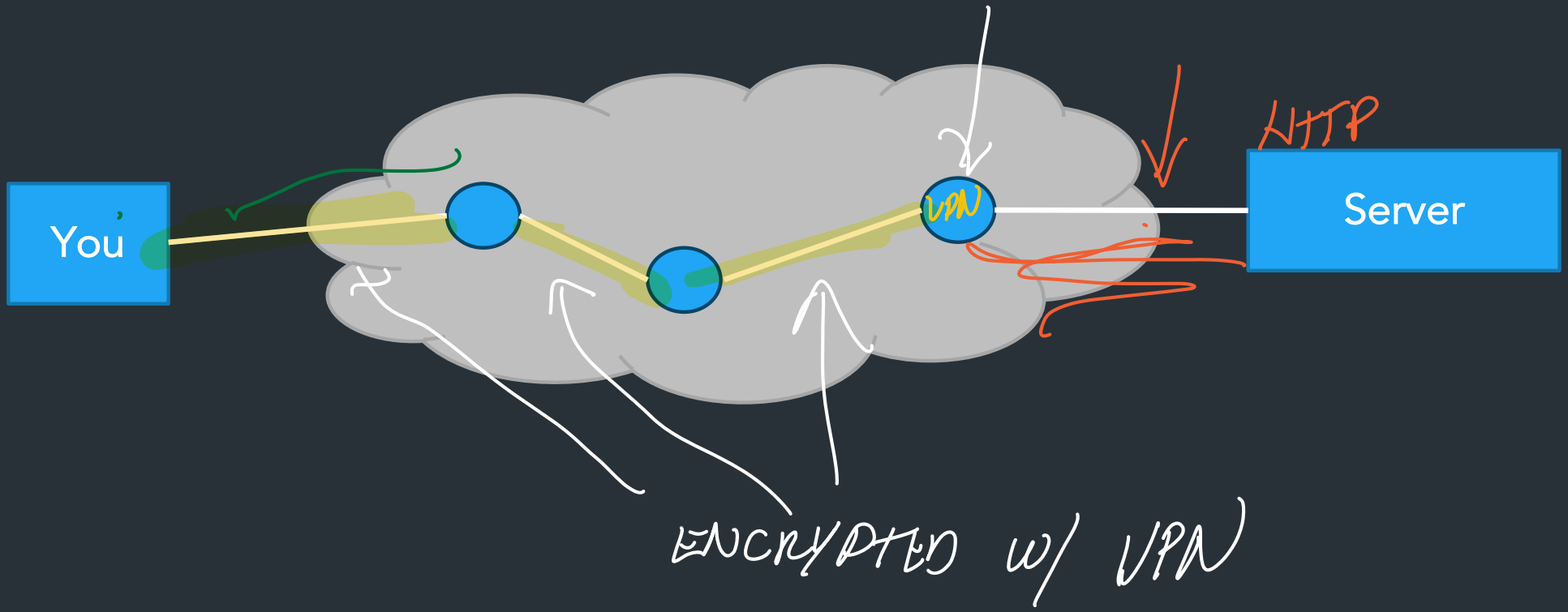
Use for

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

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

VPN: secure tunnel for network traffic

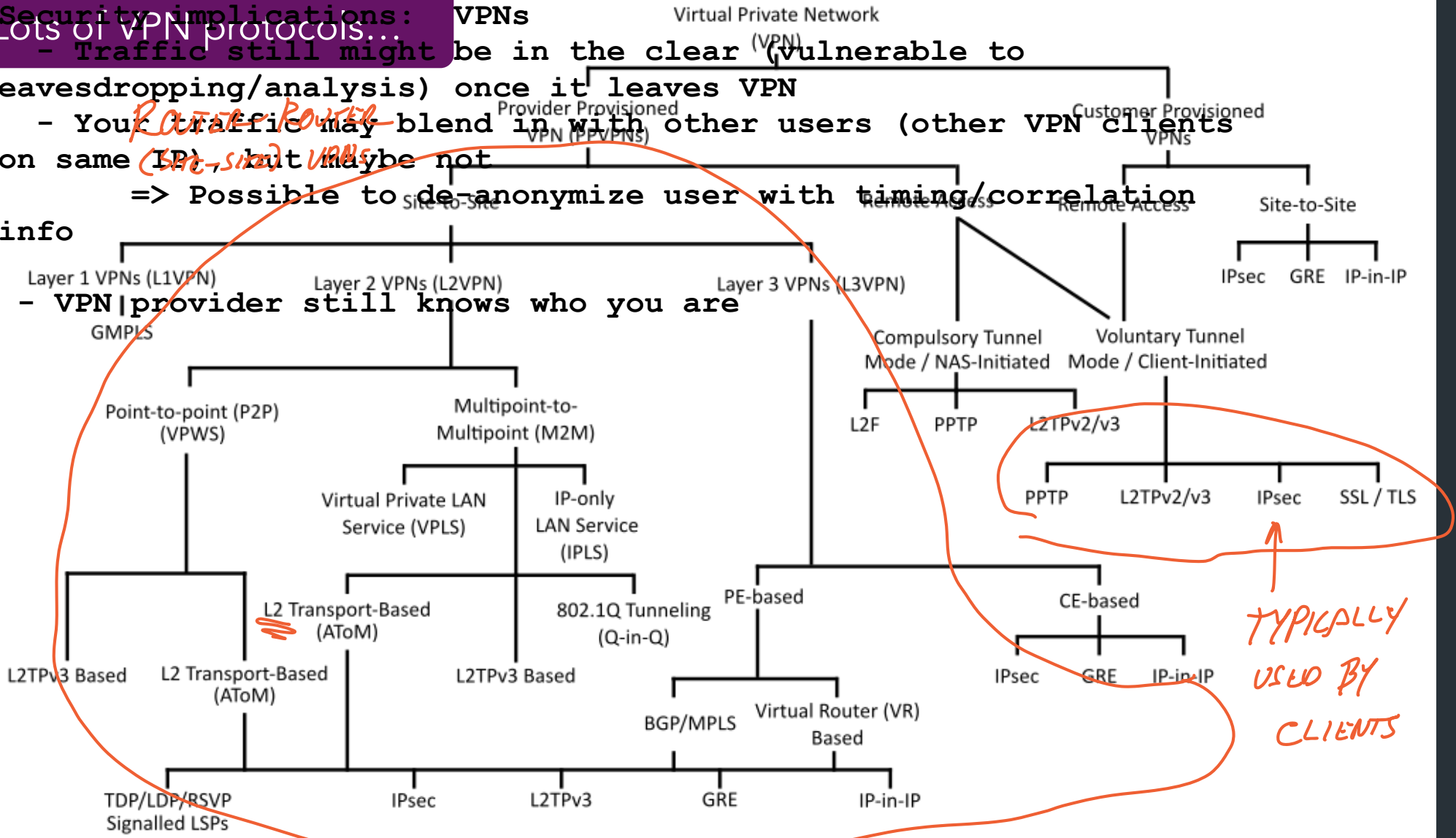
=> Connect a host to a private network

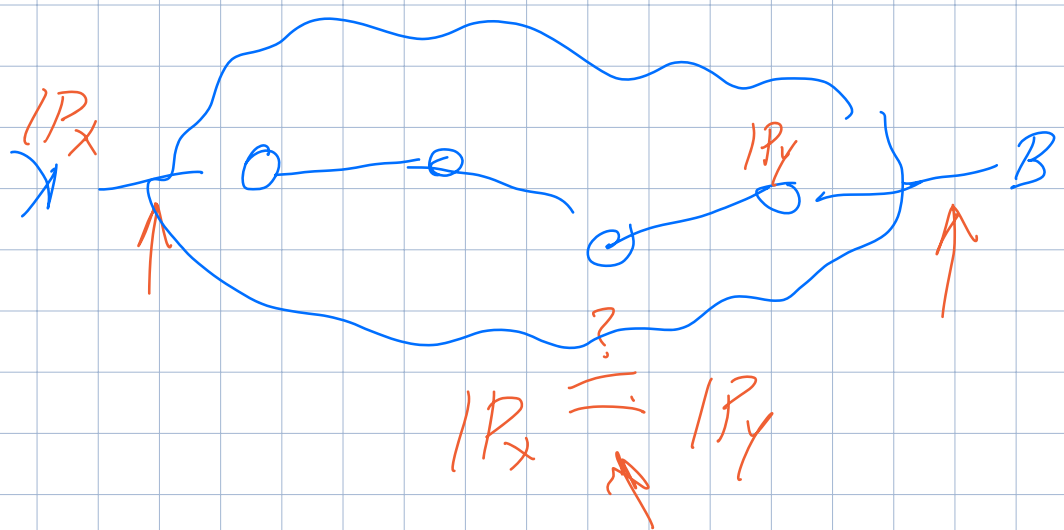


Lots of VPN protocols...

Security implications: VPNs

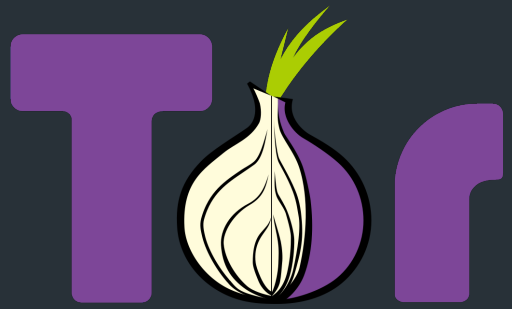
- **Traffic still might be in the clear (vulnerable to eavesdropping/analysis) once it leaves VPN**
- **Your traffic may blend in with other users (other VPN clients on same IP), but maybe not**
  - **=> Possible to de-anonymize user with timing/correlation info**
- **VPN provider still knows who you are**




IPx
IPy
B
?
IPx 二 IPy

Can we do better?

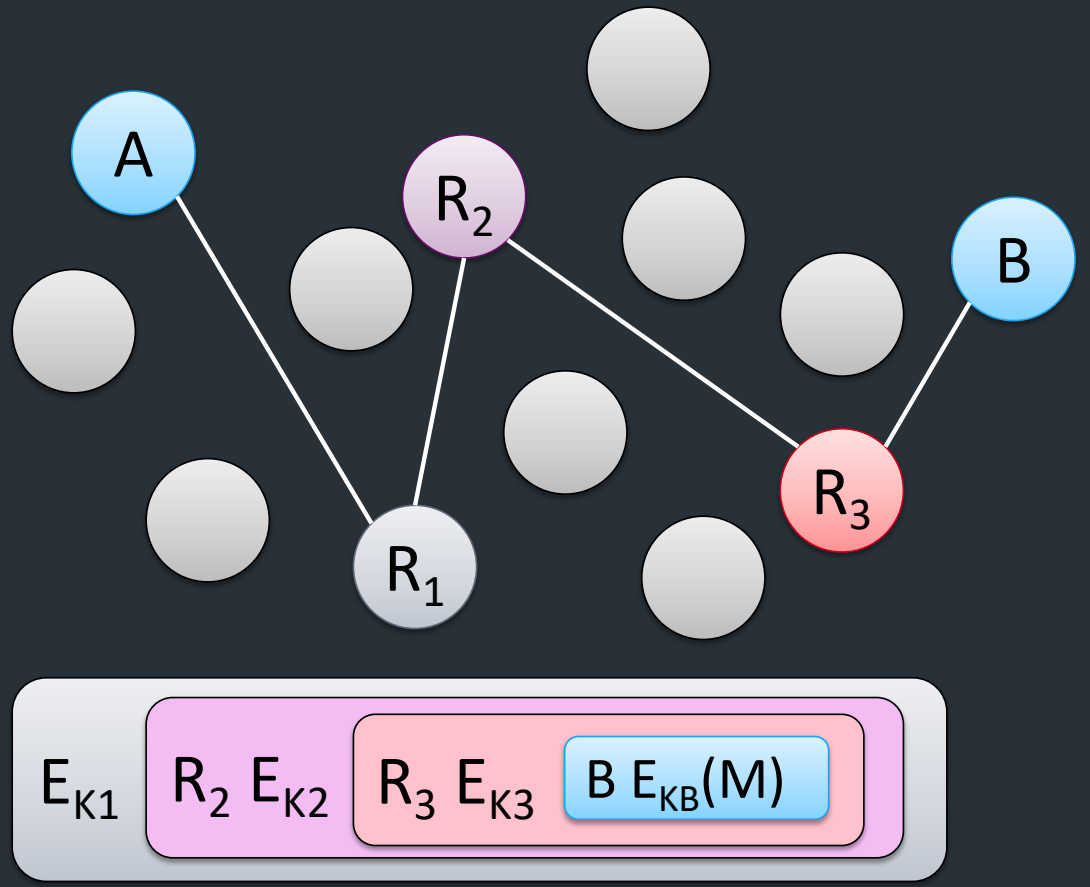
Tor



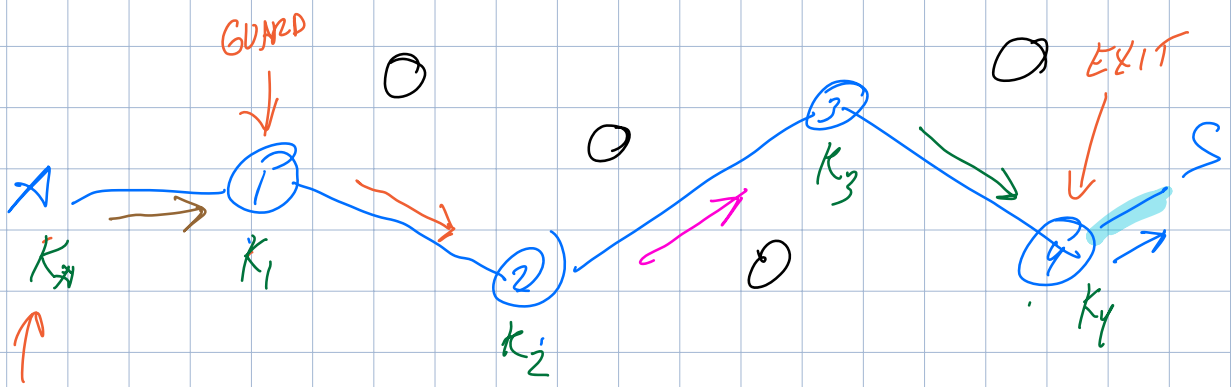
- Onion routing service: build encrypted circuit on tor relay network
- Network of relays, mainly operated by volunteers
- Started in 1990s from Naval Research Lab, now maintained by The Tor Project (a non-profit)

Onion Routing

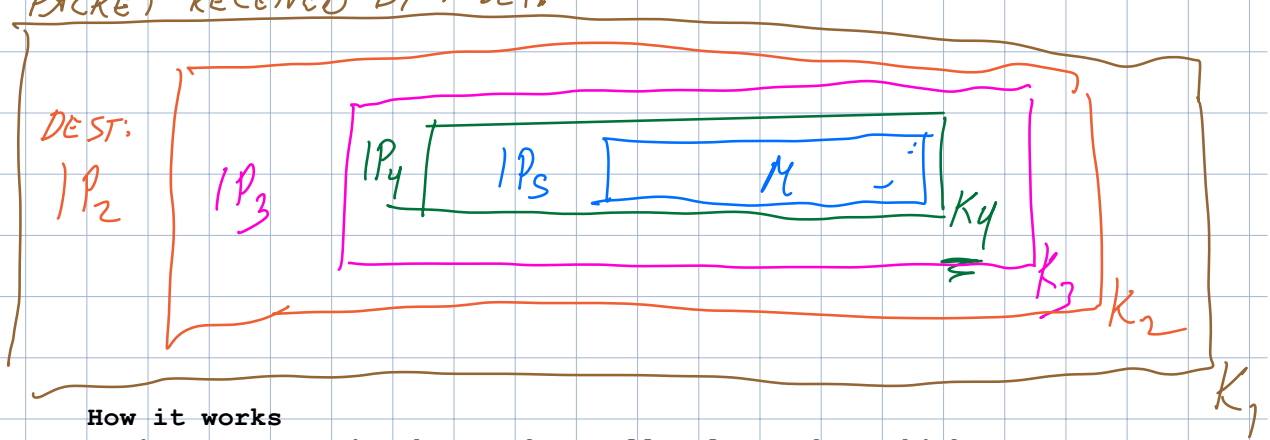
- Layered encryption
  - Build onion inside out
- Routing
  - Peel onion outside in
- Each router knows only previous and next



ONION ROUTING



PACKET RECEIVED BY NODE 1:



How it works
- Directory service knows about all relay nodes, which are run by many volunteers
- At connection start, establish a "circuit" or path from a subset of relays => know a symmetric key for each relay
  - First node is "guard node"
  - Last relay node is "exit node"

Send packets with layers of encryption => at each hop, relay can decrypt its layer of the encryption with its key
  - Unless it's the last node, it can only see the destination of the next node and the encrypted packet
    (At exit node, the packet is in cleartext => goes to destination)

Thus:
  - Only the guard node knows about the host
  - Only exit node knows the packet's true destination
  - Relays in middle only know about next hop