CS 88: Security and Privacy

21: Network Layer, Anonymity

11-29-2022

slides adapted from UC Berkeley and NEU

SWARTHMORE COLLEGE

TLS in Practice

TLS: Efficiency

- Public-key cryptography: Minor costs
  - Client and server must perform Diffie-Hellman key exchange or RSA encryption/decryption
- Symmetric-key cryptography: Effectively free
  - Modern hardware has dedicated support for symmetric-key cryptography
  - Performance impact is negligible
- Latency: Extra waiting time before the first message
  - Must perform the entire TLS handshake before sending the first message

TLS Provides End-to-End Security

- TLS provides end-to-end security: Secure communication between the two endpoints, with no need to trust intermediaries
  - Even if everybody between the client and the server is malicious, TLS provides a secure communication channel
  - End-to-end security does not help if one of the endpoints is malicious (e.g. communicating with a malicious server)
  - Example: An local network attacker (on-path) tries to read our Wi-Fi session, but can't read TLS messages
  - Example: A man-in-the-middle tries to inject TCP packets, but packets will be rejected because the MAC won't be correct
- Using TLS defends against most lower-level network attacks

TLS Does Not Provide Anonymity

- Anonymity: Hiding the client's and server's identities from attackers
- An attacker can figure out who is communicating with TLS
  - The certificate is sent during the TLS handshake, containing the server's name
  - The client may also indicate the name of the server in the ClientHello (called Server Name Indication, or SNI)
  - An attacker can see IP addresses and ports of the underlying IP and TCP protocols

TLS Does Not Provide Availability

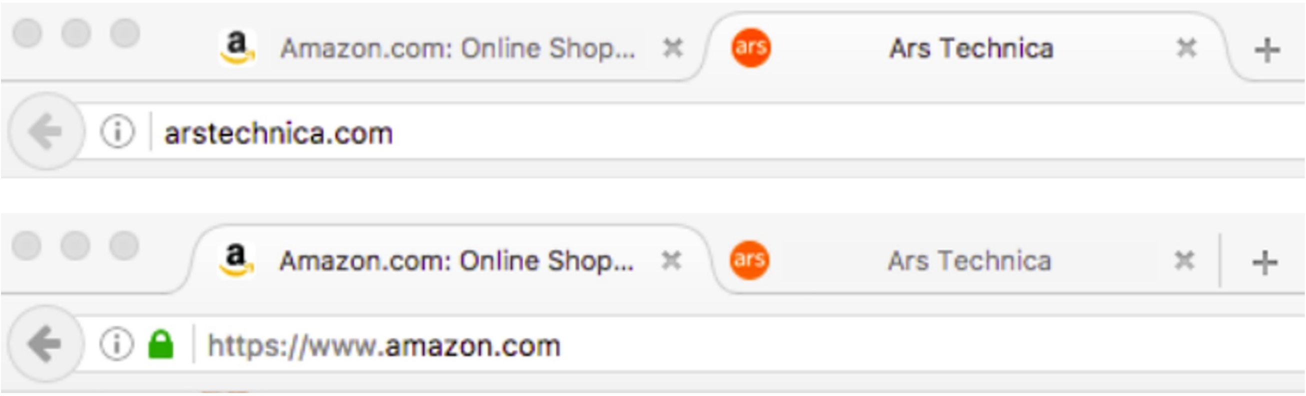
- Availability: Keeping the connection open in the face of attackers
- An attacker can stop a TLS connection
  - MITM can drop encrypted TLS packets
  - On-path attacker can still do RST injection to abort the underlying TCP connection
- Result: A TLS connection can still be censored
  - The censor can block TLS connections

TLS for Applications

- Internet layering: TLS provides services to higher layers (the application layer)
- HTTPS: The HTTP protocol run over TLS
  - In contrast, HTTP runs over plain TCP, with no TLS added
- Other secure application-layer protocols besides HTTPS exist
  - Pretty much anything that runs over TCP can also run over TLS, since the bytestream abstraction is maintained
  - Example: Email protocol can use the STARTTLS command to uses TLS to secure communications
- TLS does not defend against application-layer vulnerabilities
  - Example: SQL injection, XSS, CSRF, and buffer overflow vulnerabilities in the application are still exploitable over TLS

TLS in Browsers

- Original design:
  - When your browser communicates with a server over TLS, your browser displays a lock icon
  - If TLS is not used, there is no lock icon
- What the lock icon means
  - Communication is encrypted (TLS guarantee)
  - You are talking to the legitimate server (TLS guarantee)
  - Any external images or scripts are also fetched over TLS



This website uses HTTP: no lock icon

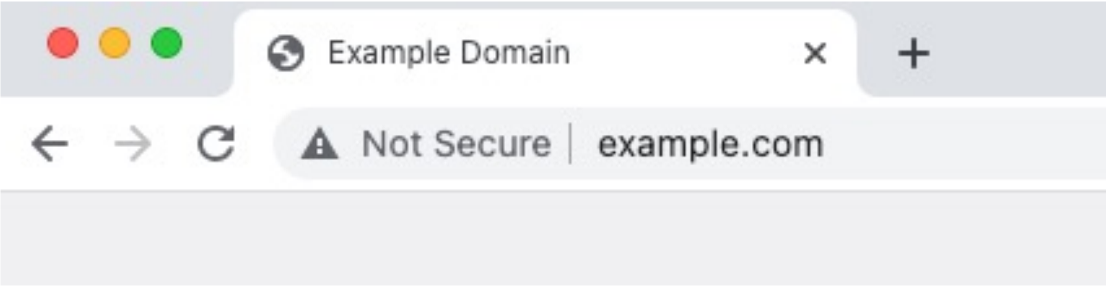
This website uses HTTPS: lock icon

TLS in Browsers

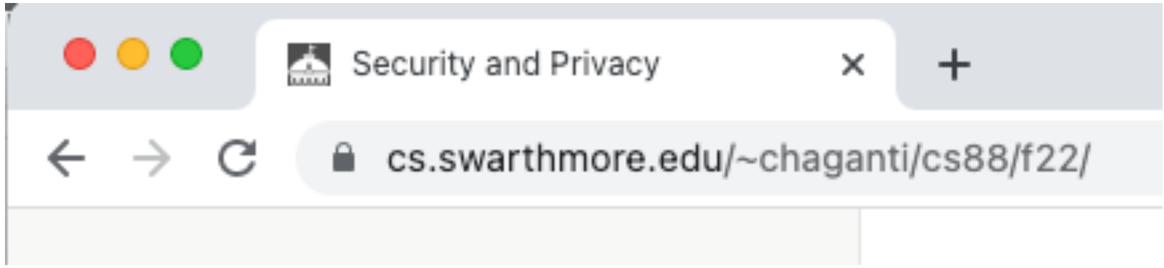
- What users think the lock icon means
  - This website is trustworthy, no matter where the lock icon actually appears
- Attack: The attacker adds their own lock icon somewhere on the page
  - The user thinks they're using TLS, but actually is not using TLS
- Attack: The user might be communicating with an attacker's website over TLS
  - The lock icon appears, but the user is actually vulnerable!

TLS in Browsers

- Modern design: Add a “not secure” icon to connections that don’t use TLS
  - Adds a signal on unencrypted sites
  - Encourages websites to stop supporting all unencrypted, HTTP traffic and redirect to HTTPS



This website uses HTTP: insecure icon



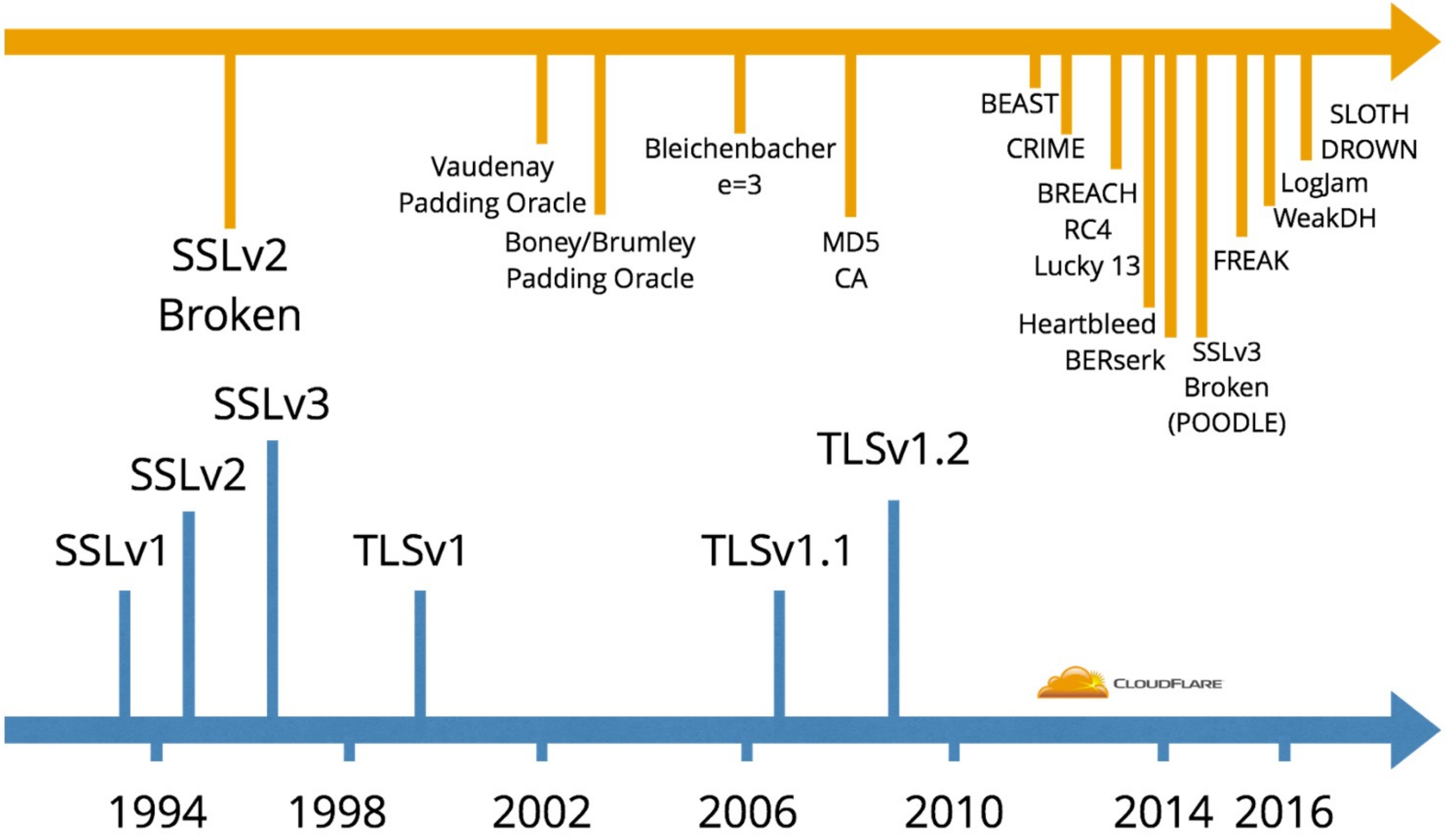
This website uses HTTPS: lock icon

TLS 1.3: the new standard

- Several years of collaboration between industry and academia
  - Standardized by IETF in
- Major differences:
  - RSA key exchange removed: no passive decryption attacks
  - Only secure DFH parameters allowed: no bad choices in parameters
  - Handshake encrypted immediately after key exchange: limits metadata available to eavesdropper
  - Protocol downgrade protection: protects against being downgraded to prior insecure versions

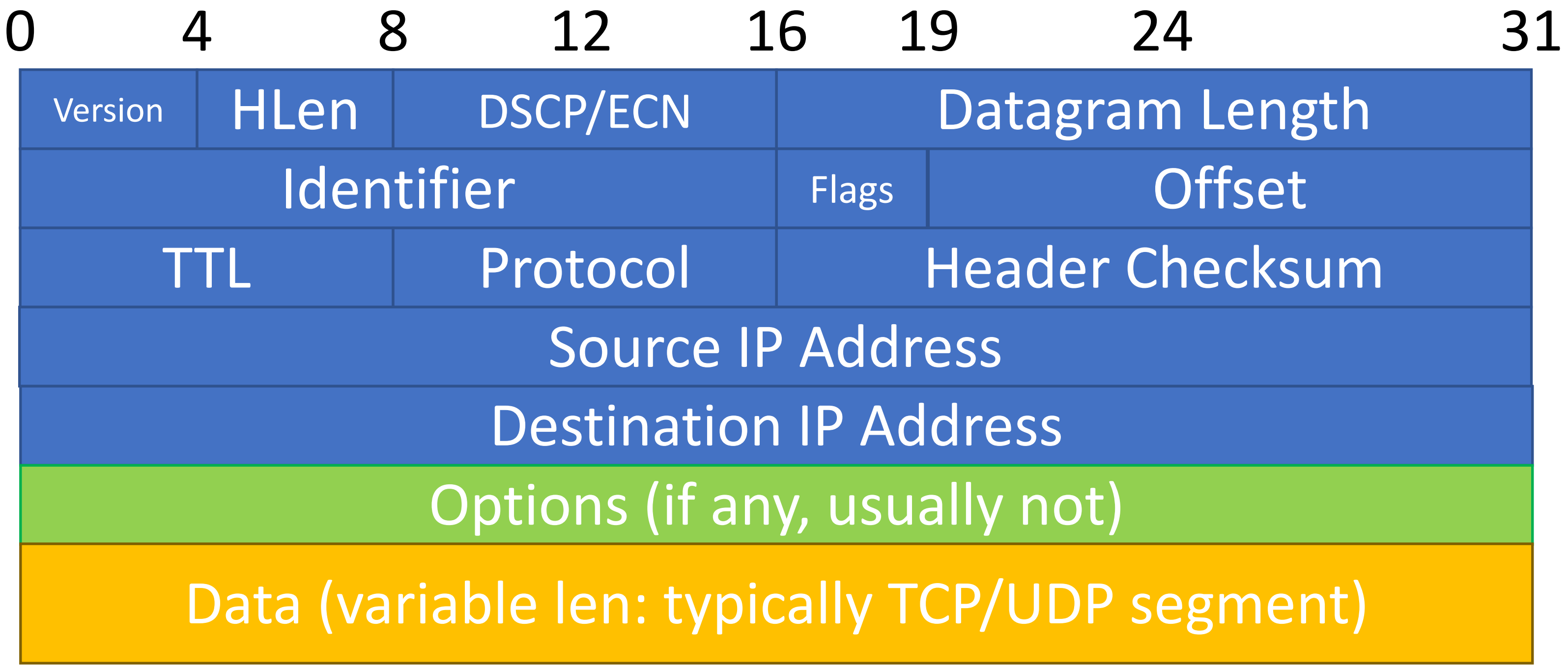
TLS v. 1.2 and below have had a lot of vulnerabilities

- Early versions of SSL developed before cryptographic protocol design was fully understood.
- Later protocol versions retained insecure options for backwards compatibility.



IP Datagrams

- IP Datagrams are like a letter
  - Totally self-contained
  - Include all necessary addressing information
  - No advanced setup of connections or circuits



How does an end host get an IP address?

- Static IP: hard-coded
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - "plug-and-play"

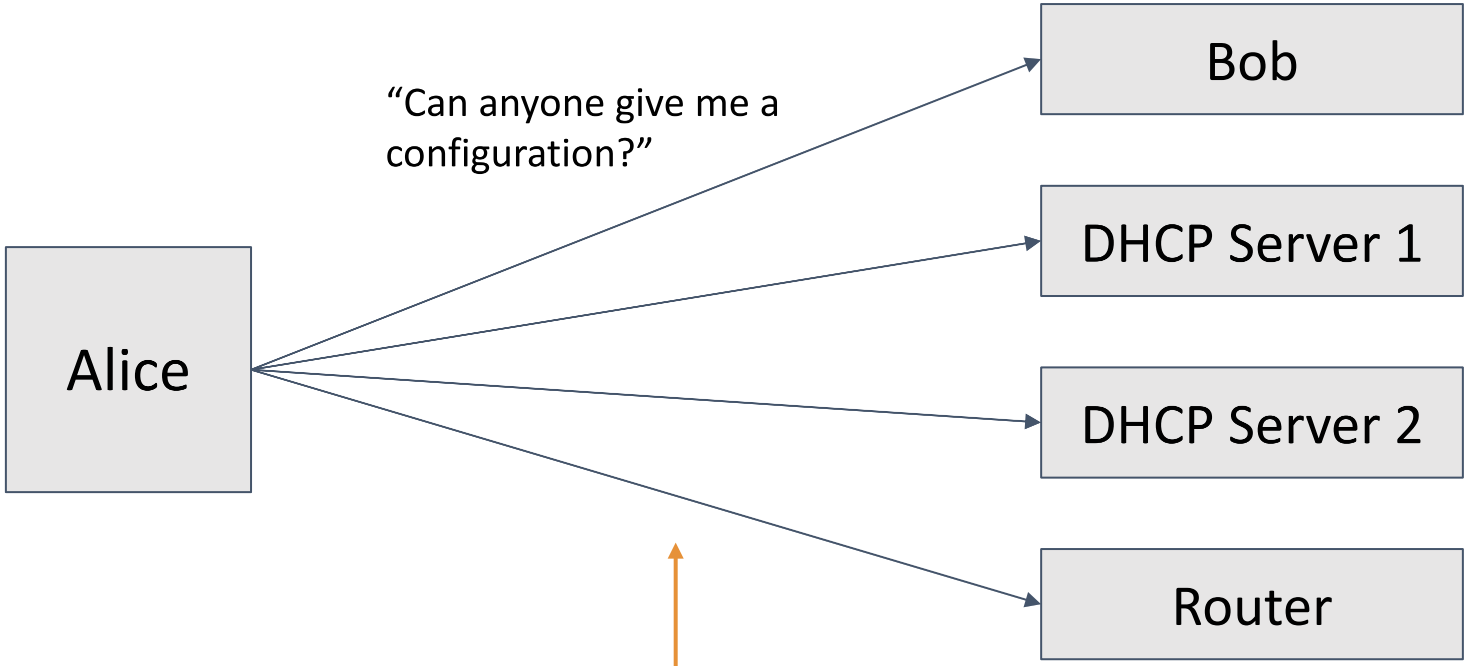
Dynamic Host Configuration Protocol (DHCP)

Alice's configuration	
My IP	???
DNS Server	???
Gateway	???

Alice wants to connect to the network, but she's missing a configuration.

Dynamic Host Configuration Protocol (DHCP)

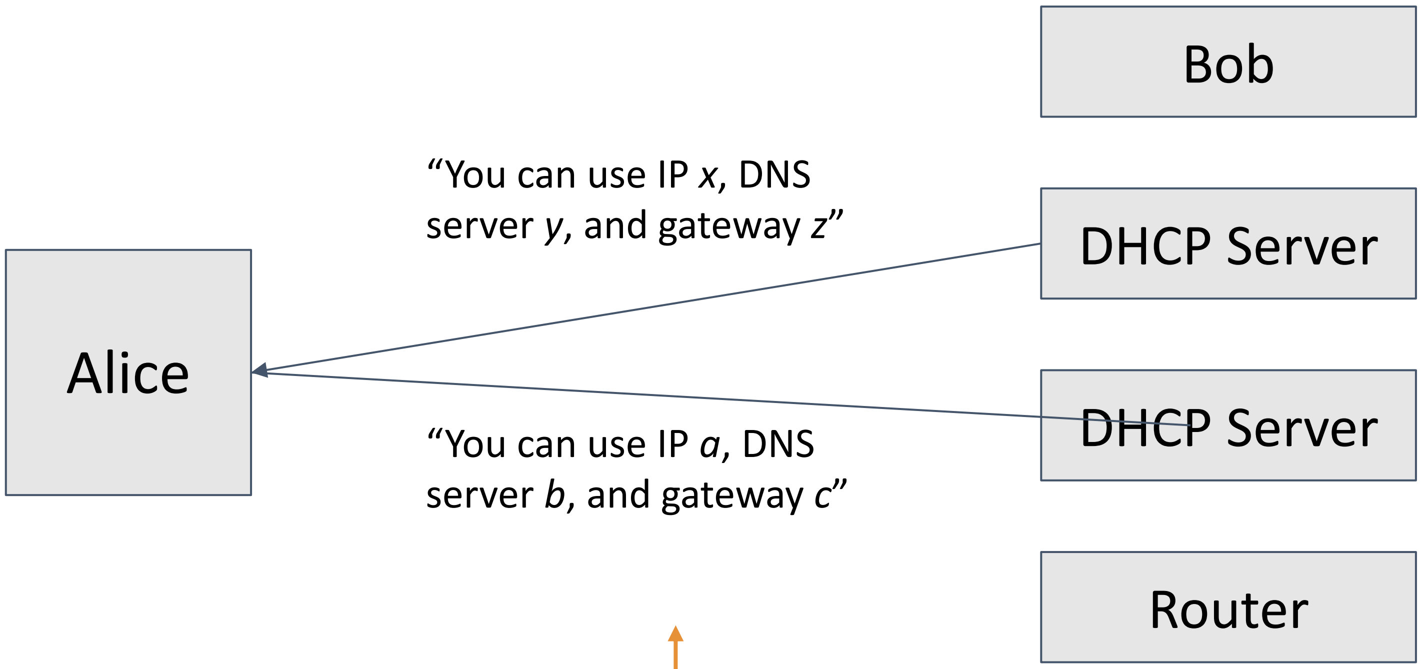
Alice's configuration	
My IP	???
DNS Server	???
Gateway	???



1. **Client Discover**: Alice broadcasts a request for a configuration.

Dynamic Host Configuration Protocol (DHCP)

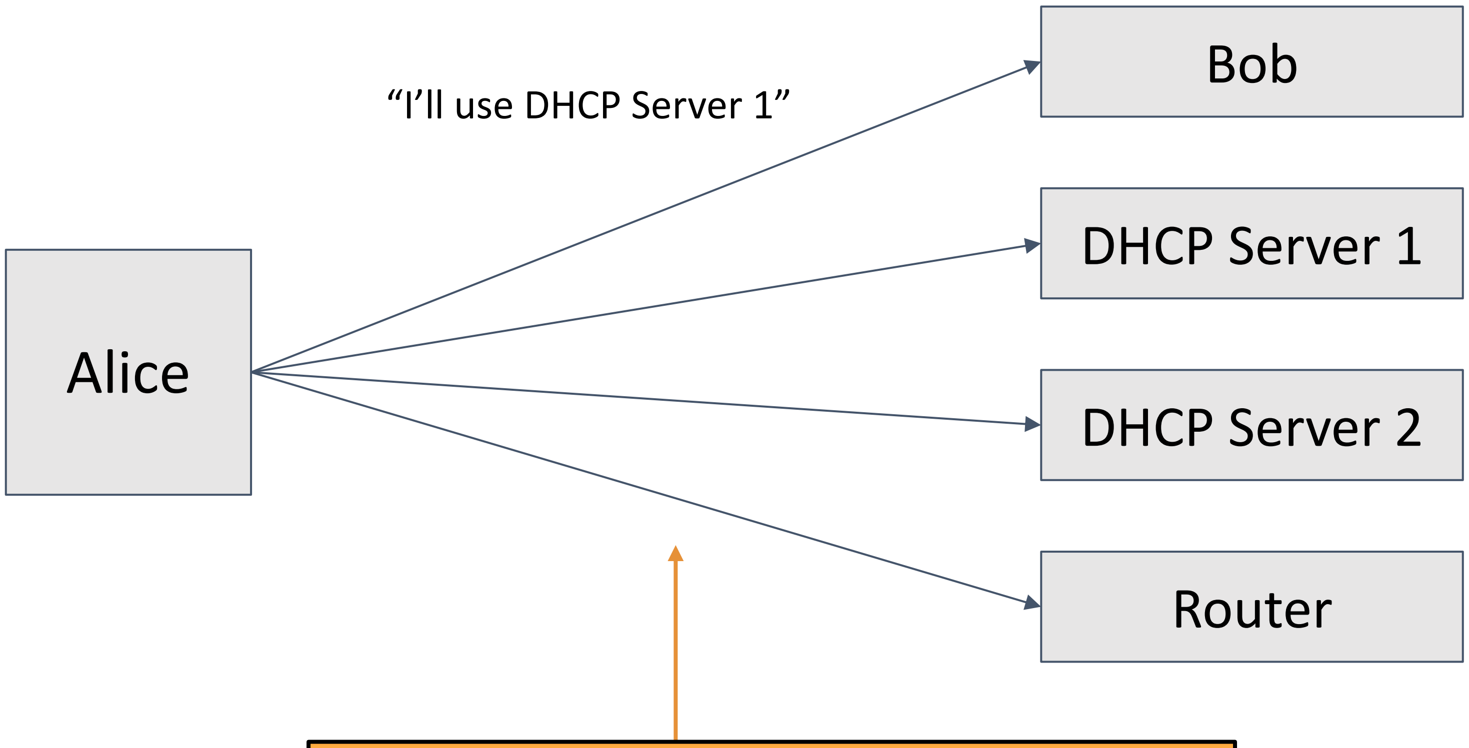
Alice's configuration	
My IP	???
DNS Server	???
Gateway	???



2. **DHCP Offer**: Any DHCP server can reply with an offer for Alice.

Dynamic Host Configuration Protocol (DHCP)

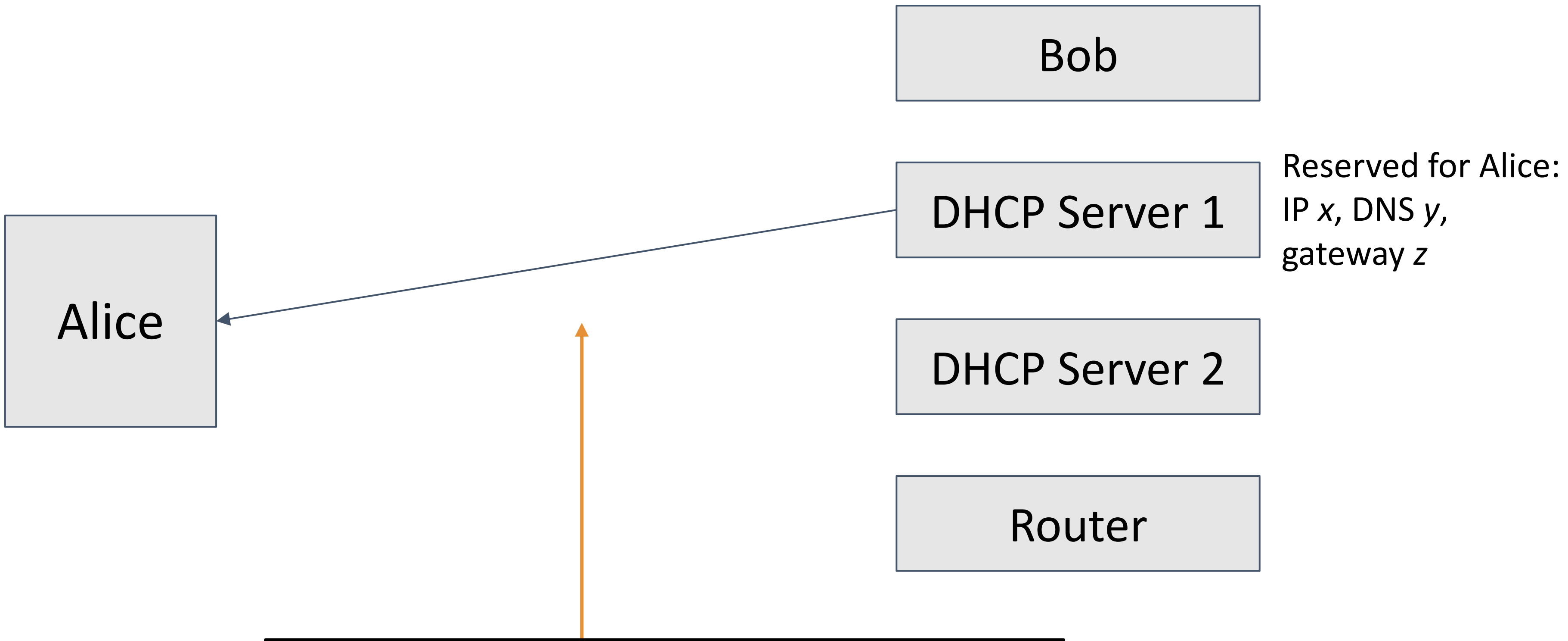
Alice's configuration	
My IP	???
DNS Server	???
Gateway	???



3. **Client Request**: Alice broadcasts which configuration she has chosen.

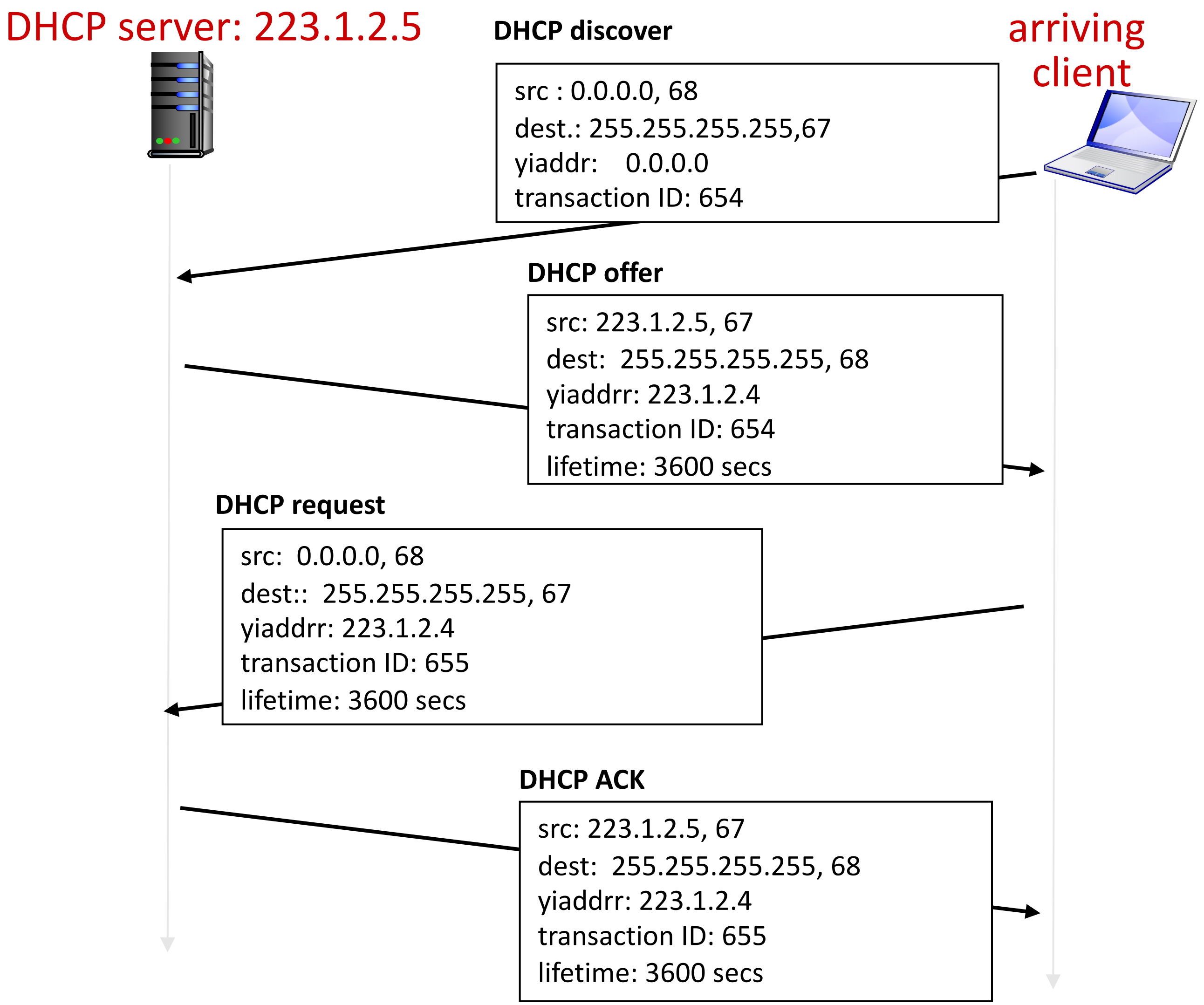
Dynamic Host Configuration Protocol (DHCP)

Alice's configuration	
My IP	x
DNS Server	y
Gateway	z



4. **DHCP Acknowledgement**: The chosen DHCP server confirms that the configuration has been set for Alice.

DHCP client-server scenario



DHCP: More than IP Addresses

DHCP can return more than just allocated IP address on subnet:

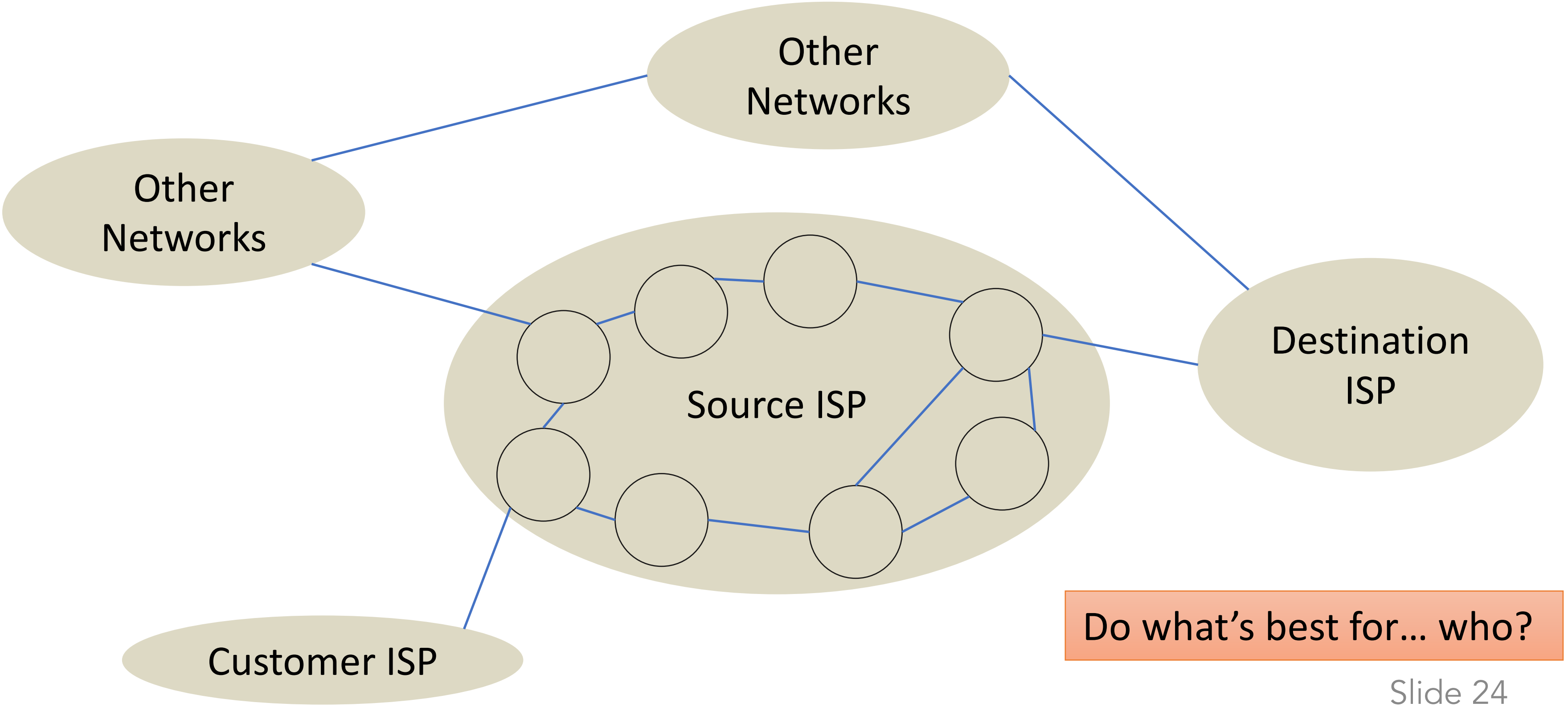
- address of first-hop router for client (default GW)
- name and IP address of DNS server(s)
- subnet mask

Internet/inter-AS Routing

Goal:
Get traffic from one AS to another.

Routing Policy

- How should the ISP route the customer's traffic to the destination?



Do what's best for… who?

Which routes a BGP router advertises will depend on…

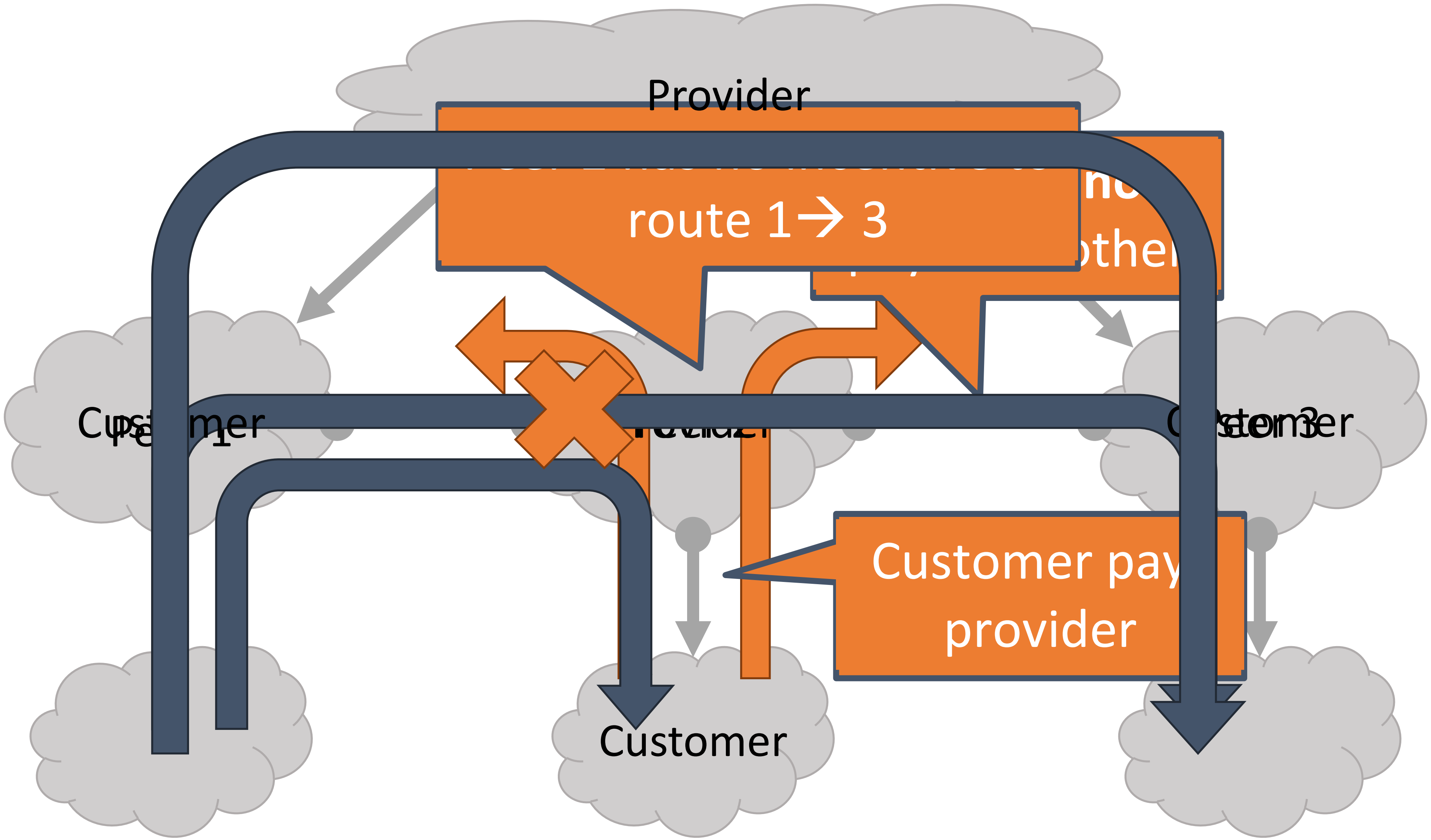
A. which ISPs have contractual agreements.

B. the shortest path to a subnet/prefix.

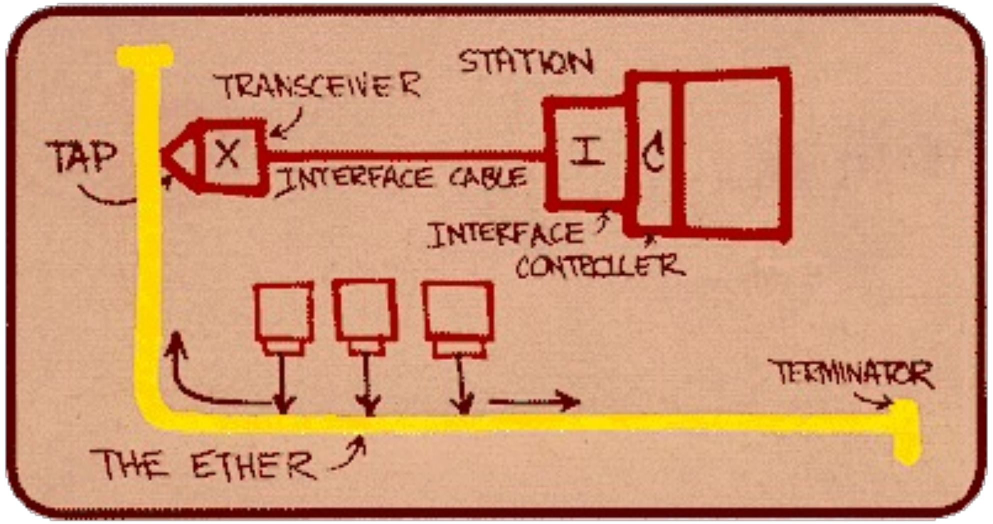
C. which subnets are customers of an ISP.

D. More than one of the above. (which?)

BGP Relationships



Ethernet



Metcalfe's Ethernet sketch

"Dominant" wired LAN technology:

- cheap $20 for NIC
- first widely used LAN technology
- simpler, cheaper than token LANs and ATM
- kept up with speed race: 10 Mbps – 10 Gbps

WiFi (802.11)

Senders do carrier sensing like Ethernet.

Assumes hosts can hear each other!

MAC Addresses

- MAC (or LAN or physical or Ethernet) address:
  - 48 bit MAC address
  - e.g.: 1A-2F-BB-76-09-AD

hexadecimal (base 16) notation
(each digit represents 4 bits)

MAC vs. IP Addresses

- 32-bit IP address
- IP hierarchical address not portable!
  - address depends on IP subnet to which node is attached
- used by network layer for end-to-end routing

- 48 bit MAC address burned in NIC ROM.
- MAC flat address: portability
  - can move LAN card from one LAN to another
- used locally to get from one interface to another physically-connected interface

Analogy:

MAC address: like Social Security Number

IP address: like postal address

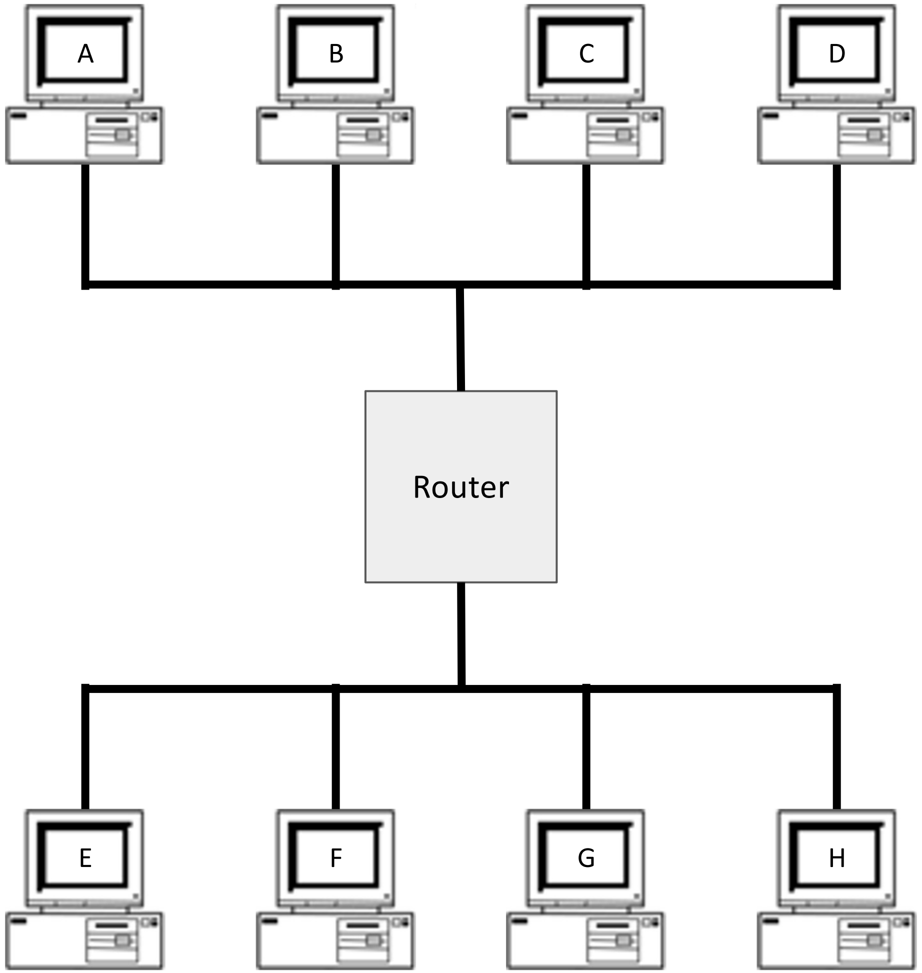
Ethernet: unreliable, connectionless

- Connectionless: no handshaking between sending and receiving NICs
- Unreliable: receiving NIC doesn't send acks or nacks to sending NIC
  - data in dropped frames recovered only if initial sender uses higher layer reliable delivery (e.g., TCP), otherwise dropped data lost
- Ethernet's MAC protocol:
  CSMA/CD with binary exponential backoff

Address Resolution Protocol (ARP)

Review: Layer 2 and Layer 3

- Local area network (LAN): A set of machines connected in a local network
  - The MAC identifies devices on layer 2
- Internet protocol (IP): Many LANs connected together with routers
  - The IP identifies devices on layer 3



Address Resolution Protocol (ARP)

- **ARP**: Translates layer 3 IP addresses to layer 2 MAC addresses
  - Example: Alice wants to send a message to Bob on the local network, but Alice only knows Bob's IP address (**1.2.3.4**). To use layer 2 protocols, she must learn Bob's MAC address.
- Steps of the protocol
  a. Alice checks her cache to see if she already knows Bob's MAC address.
  b. If Bob's MAC address is not in the cache, Alice **broadcasts** to everyone on the LAN:
     "What is the MAC address of **1.2.3.4**?"
  c. Bob responds by sending a message only to Alice: "My IP is **1.2.3.4** and my MAC address is **ca:fe:f0:0d:be:ef**." Everyone else does nothing.
  d. Alice caches Bob's MAC address.

Address Resolution Protocol (ARP)

Alice knows Bob's IP address (**`1.2.3.4`**) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC

Alice

Bob

Charlie

Dave

Router

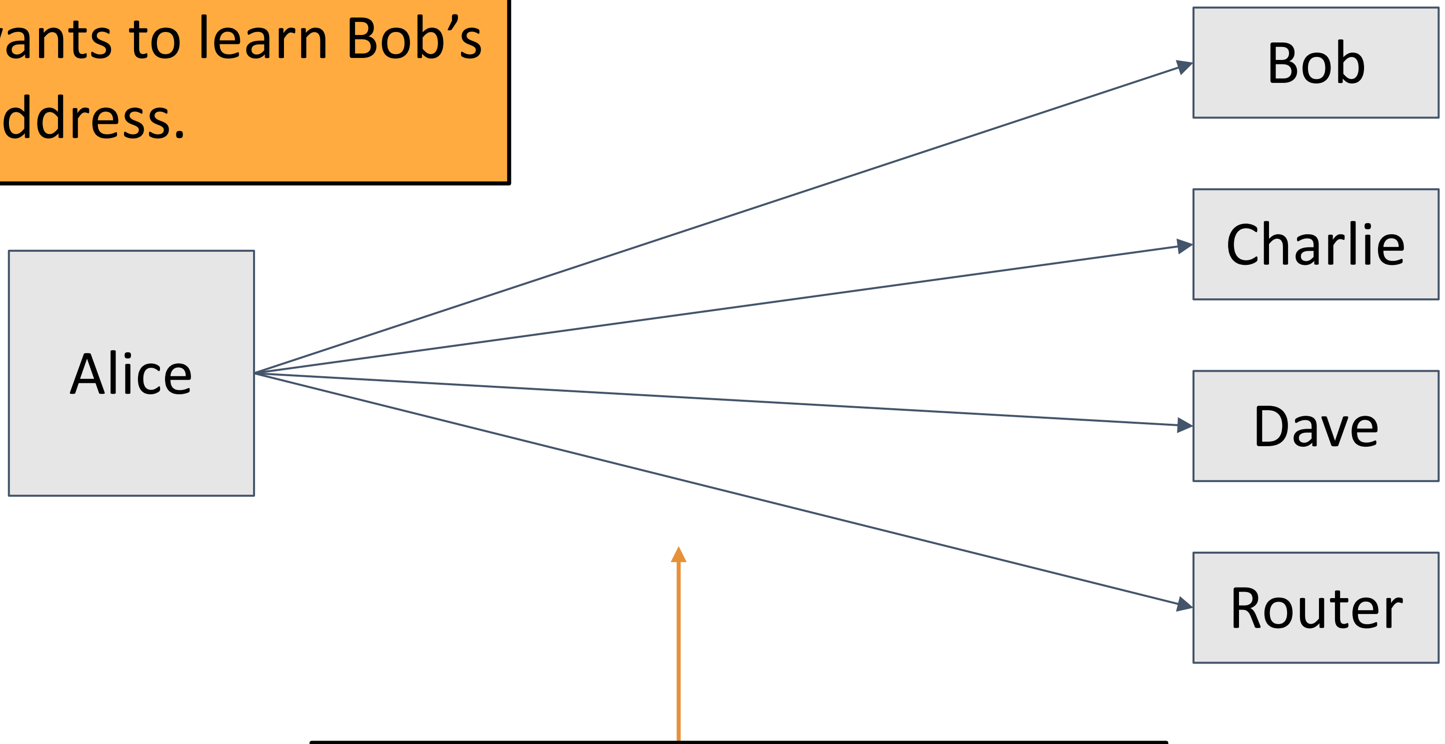
1. Alice checks her cache to see if she already knows the MAC address corresponding to **`1.2.3.4`**.

Since her cache is empty, she must make a request to find out.

Address Resolution Protocol (ARP)

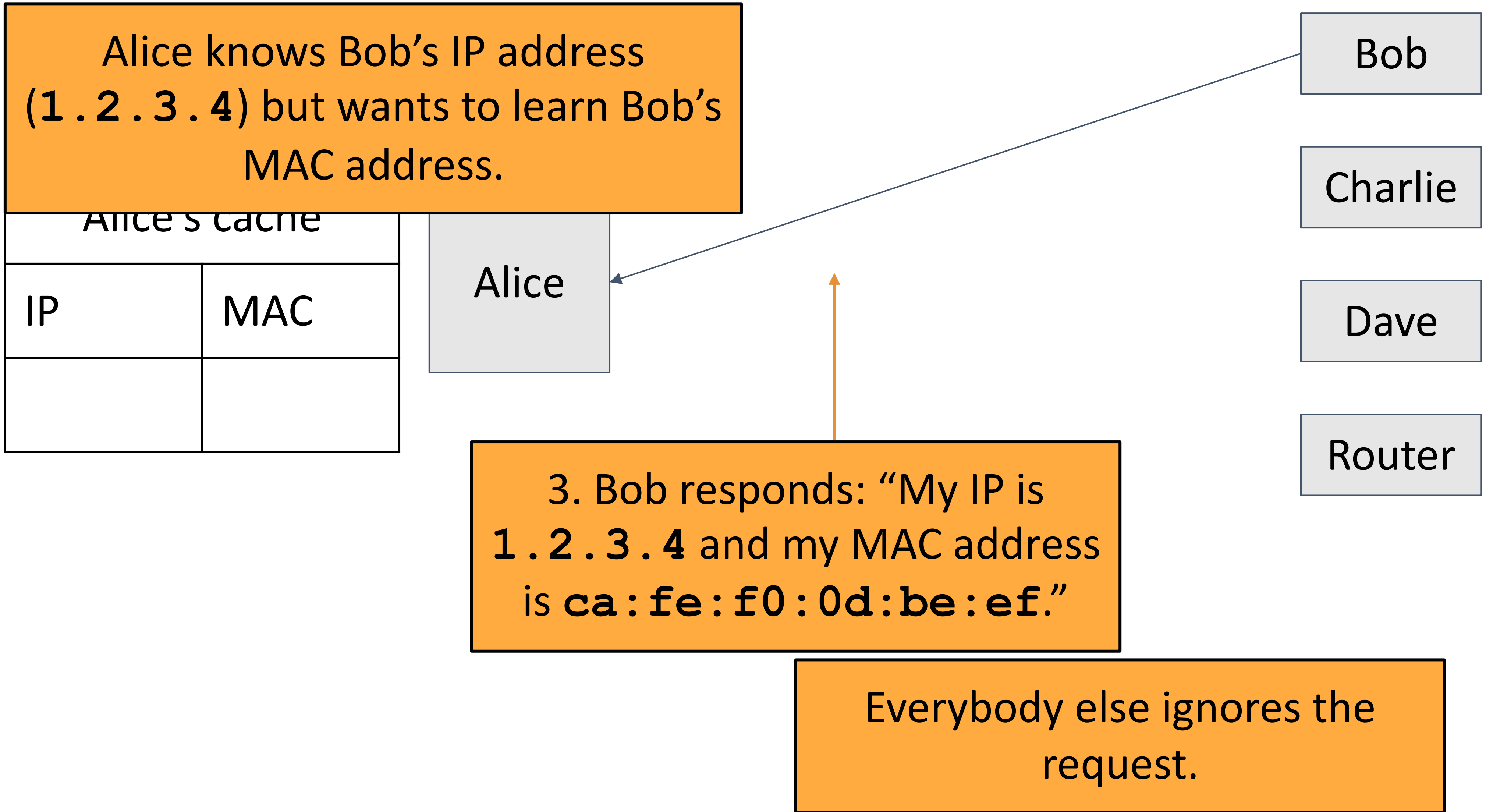
Alice knows Bob's IP address (**`1.2.3.4`**) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC



2. Alice asks everyone else on the local network: "What is the MAC address of **`1.2.3.4`**?"

Address Resolution Protocol (ARP)



Address Resolution Protocol (ARP)

Alice knows Bob's IP address (**`1.2.3.4`**) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC
`1.2.3.4`	**`ca:fe:f0:0d:be:ef`**

4. Alice adds Bob's MAC address to her cache.

Address Resolution Protocol (ARP)

- If Bob is outside of the LAN, Alice knows this
  - Bob’s IP is not on the same “subnet” as Alice
- But Alice knows the IP address of the “Gateway router”
  - Recall: The router’s job is to make sure that the packet will be forwarded towards Bob (Layer 3)
- So instead Alice generates an ARP request for the gateway router
  - Layer 2 MAC address of the frame is set to the router
  - Layer 3 IP address of the packet remains set as Bob's
  - The router will forward the packet to some other LAN to get it closer to Bob

Attacks on ARP

Alice knows Bob's IP address (**1.2.3.4**) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC

Alice

Bob

Charlie

Mallory

Router

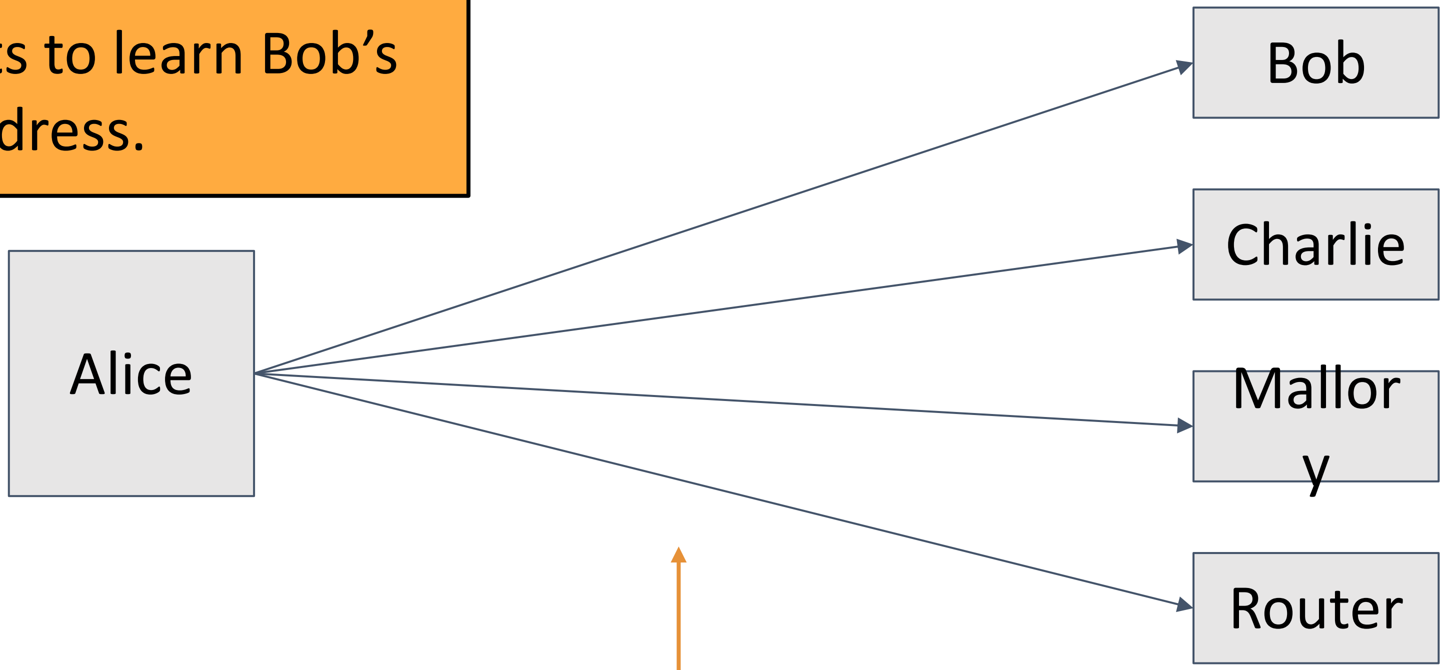
1. Alice checks her cache to see if she already knows the MAC address corresponding to **1.2.3.4**.

Since her cache is empty, she must make a request to find out.

Attacks on ARP

Alice knows Bob's IP address (**1.2.3.4**) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC

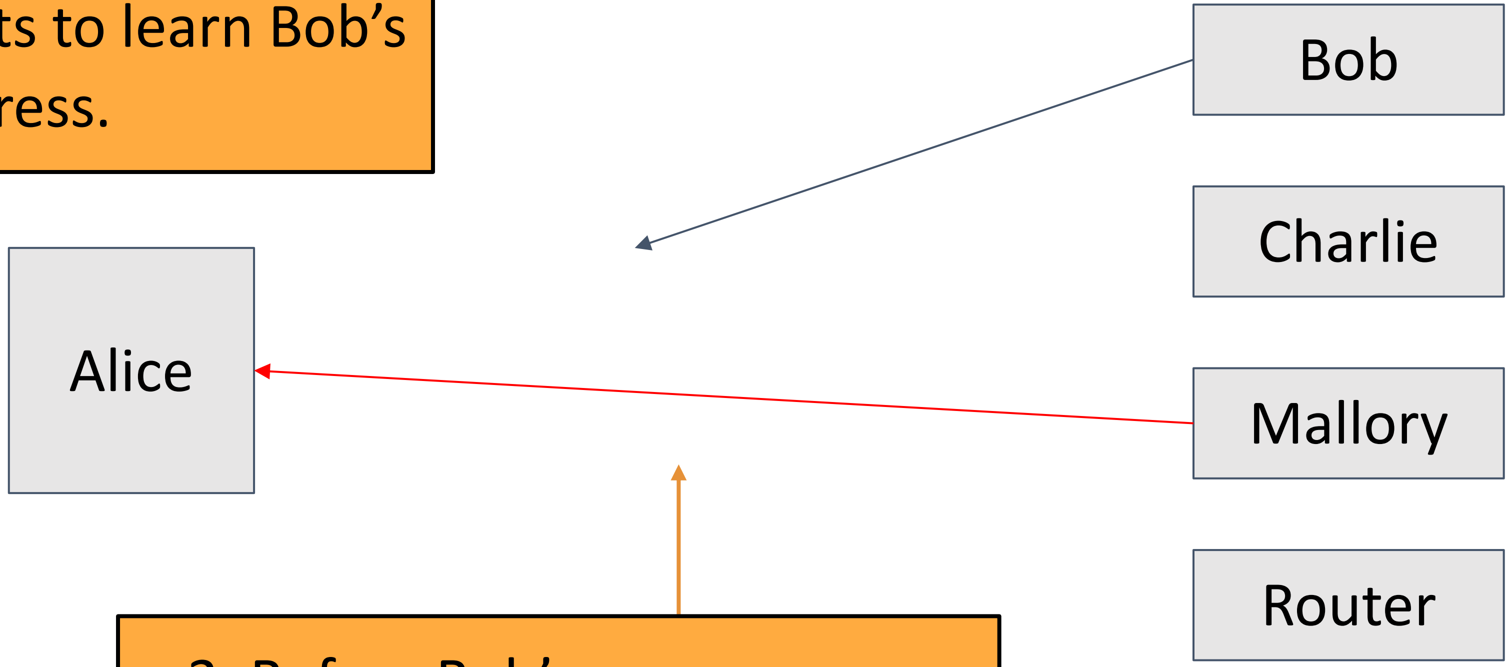


2. Alice asks everyone else on the local network: "What is the MAC address of **1.2.3.4**?"

Attacks on ARP

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC



3. Before Bob's response can arrive, Mallory sends a malicious response: "My IP is **1.2.3.4** and my MAC address is **66:66:66:66:66:66**."

Attacks on ARP

Alice knows Bob's IP address (`1.2.3.4`) but wants to learn Bob's MAC address.

Alice's cache	
IP	MAC
1.2.3.4	**66:66:66:66:66:66**

Alice

4. Alice adds Mallory's malicious address to her cache.

Bob

Charlie

Mallory

Router

“On the Internet, nobody knows you’re a dog.”

You Are Not Anonymous

- Your IP address can be linked directly to you
  - ISPs store communications records
  - Usually for several years (Data Retention Laws)
  - Law enforcement can subpoena these records
- Your browser is being tracked
  - Cookies, Flash cookies, E-Tags, HTML5 Storage
  - Browser fingerprinting
- Your activities can be used to identify you
  - Unique websites and apps that you use
  - Types of links that you click

Wiretapping is Ubiquitous

- Wireless traffic can be trivially intercepted
  - Airsnort, Firesheep, etc.
  - Wifi and Cellular traffic!
  - Encryption helps, if it's strong
    - WEP and WPA are both vulnerable!
- Tier 1 ASs and IXPs are compromised
  - NSA, GCHQ, "5 Eyes"

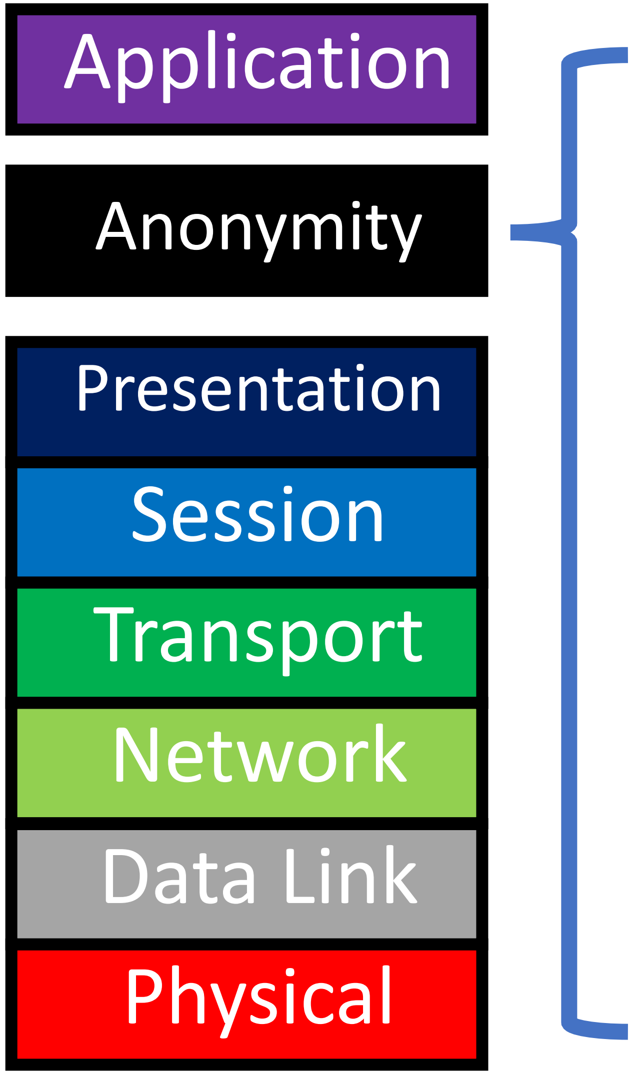
Who Uses Anonymity Systems?

- "If you're not doing anything wrong, you shouldn't have anything to hide."
  - Implies that anonymous communication is for criminals
- The truth: who uses Tor?
  - Journalists
  - Law enforcement
  - Human rights activists
  - Normal people
  - Business executives
  - Military/intelligence personnel
  - Abuse victims
- Fact: Tor was/is developed by the Navy

Why Do We Want Anonymity?

- To protect privacy
  - Avoid tracking by advertising companies
  - Viewing sensitive content
    - Information on medical conditions
    - Advice on bankruptcy
- Protection from prosecution
  - Not every country guarantees free speech
  - Downloading copyrighted material
- To prevent chilling-effects
  - It's easier to voice unpopular or controversial opinions if you are anonymous

Anonymity Layer



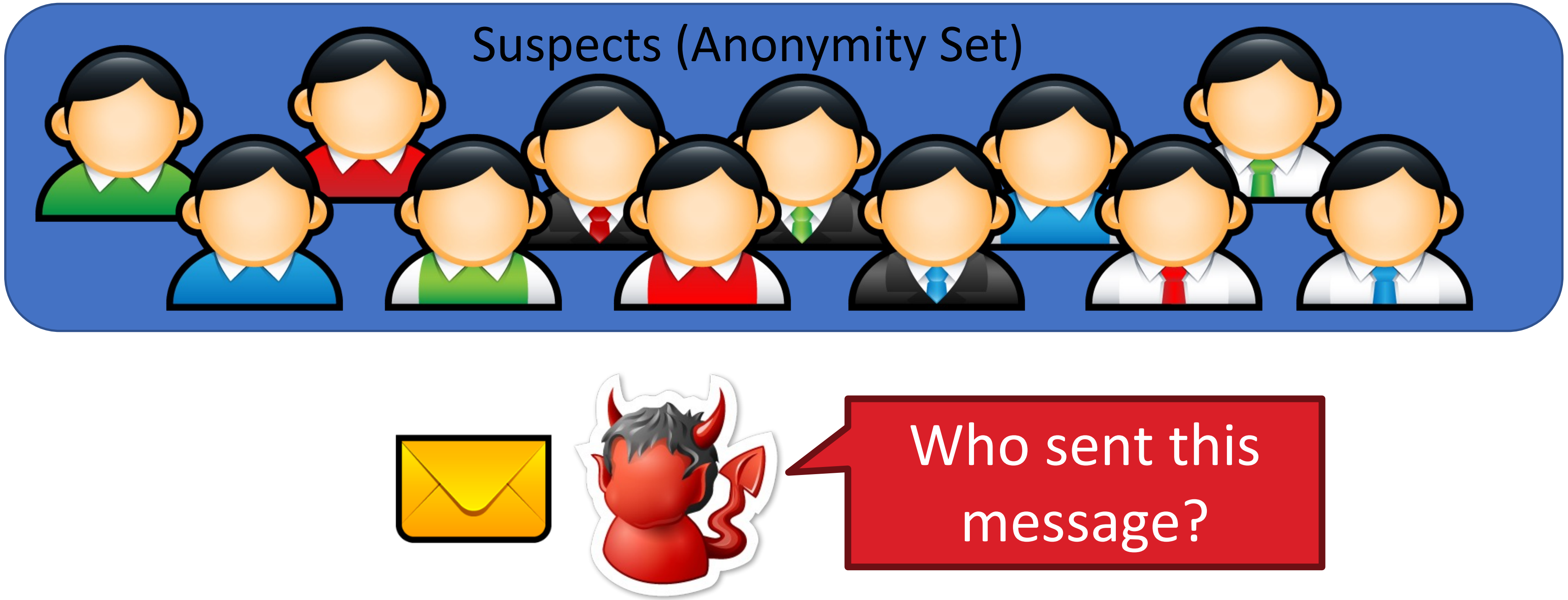
- Function:
  - Hide the source, destination, and content of Internet flows from eavesdroppers
- Key challenge:
  - Defining and quantifying anonymity
  - Building systems that are resilient to deanonymization
  - Maintaining performance

Outline

- ❑ Definitions and Examples
- ❑ Crowds
- ❑ Chaum Mix / Mix Networks
- ❑ Tor

Quantifying Anonymity

- How can we calculate how anonymous we are?
  - Anonymity Sets

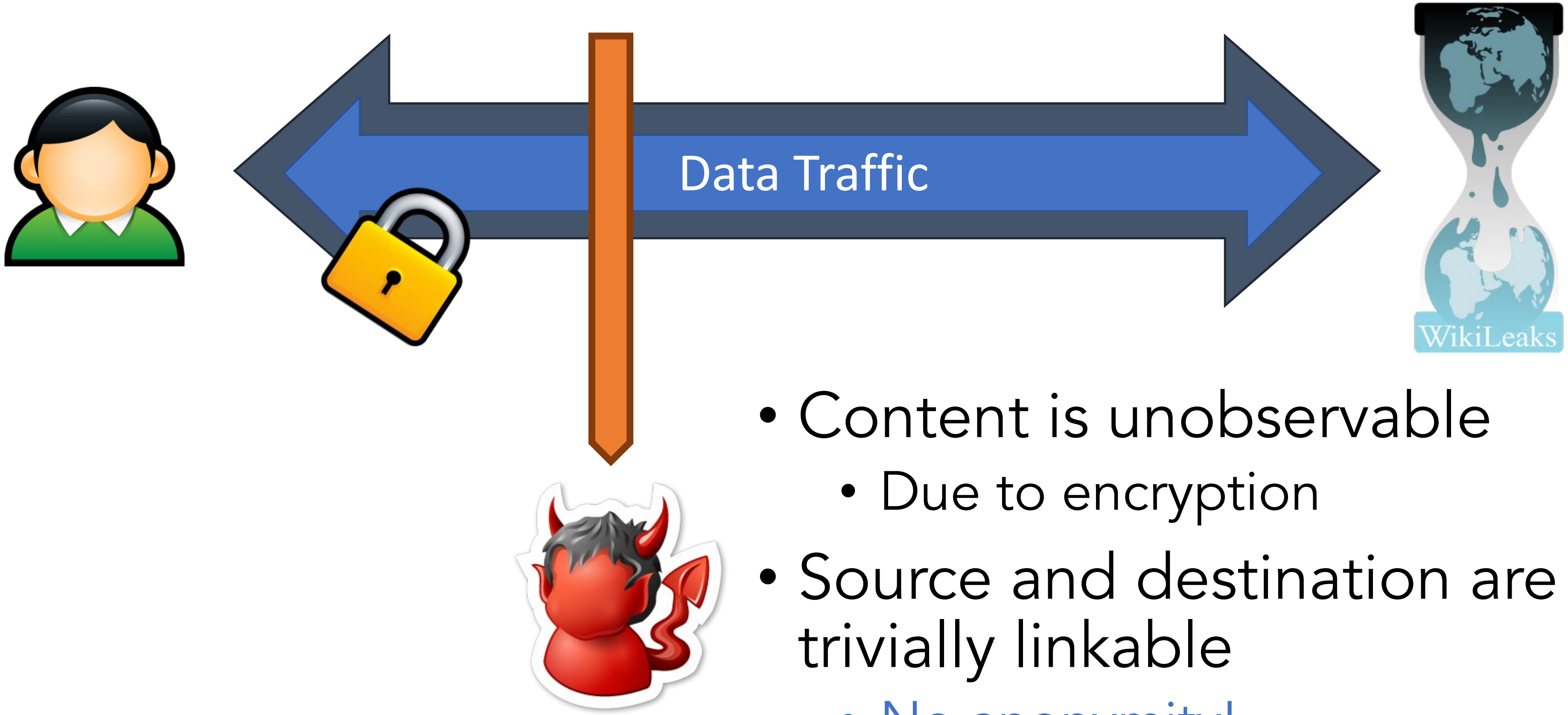


□ Larger anonymity set = stronger anonymity

Other Definitions

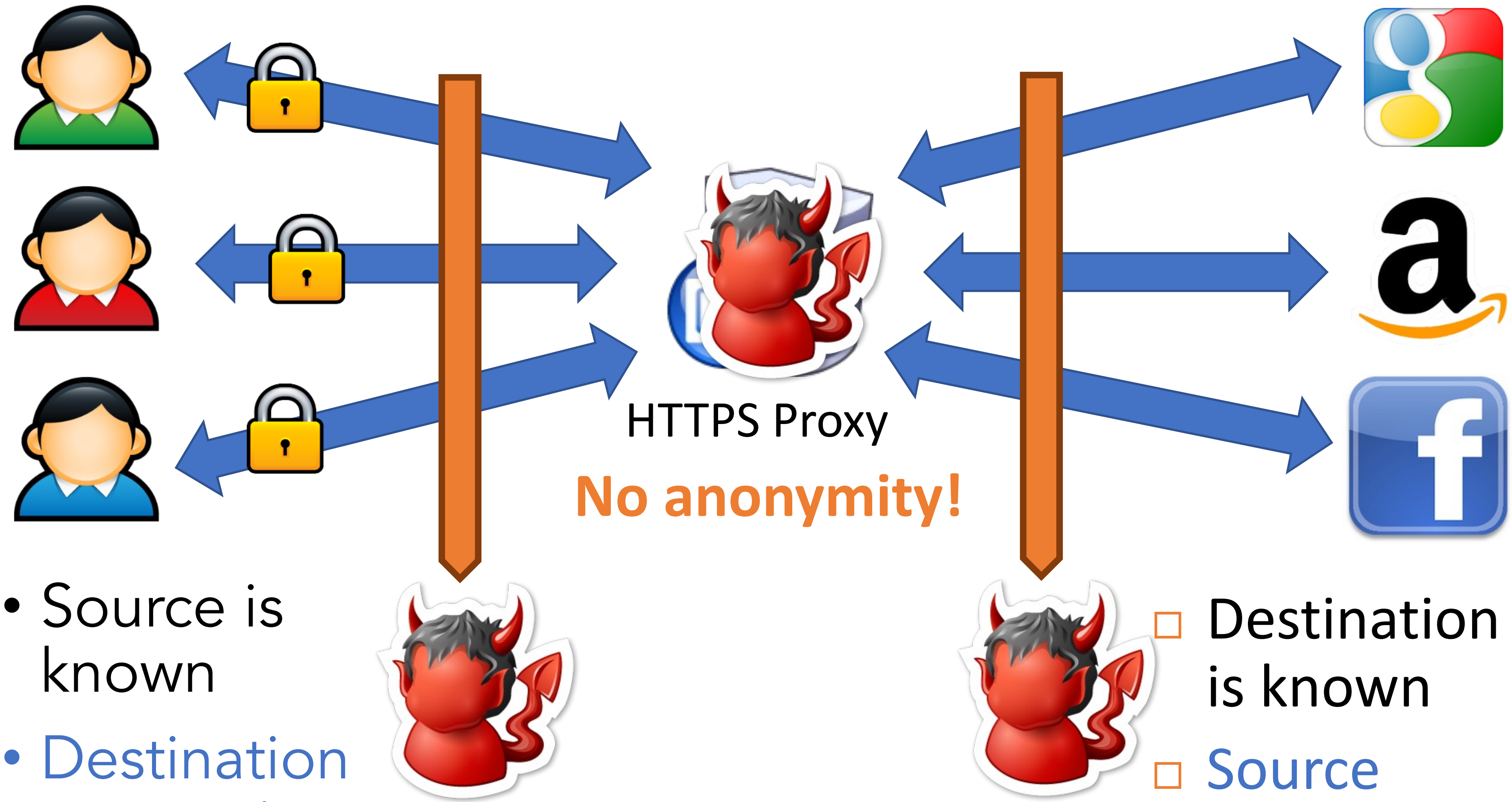
- Unlinkability
  - From the adversaries perspective, the inability the link two or more items of interest
    - E.g. packets, events, people, actions, etc.
  - Three parts:
    - Sender anonymity (who sent this?)
    - Receiver anonymity (who is the destination?)
    - Relationship anonymity (are sender A and receiver B linked?)
- Unobservability
  - From the adversaries perspective, items of interest are indistinguishable from all other items

Crypto (SSL)



- Content is unobservable
  - Due to encryption
- Source and destination are trivially linkable
  - No anonymity!

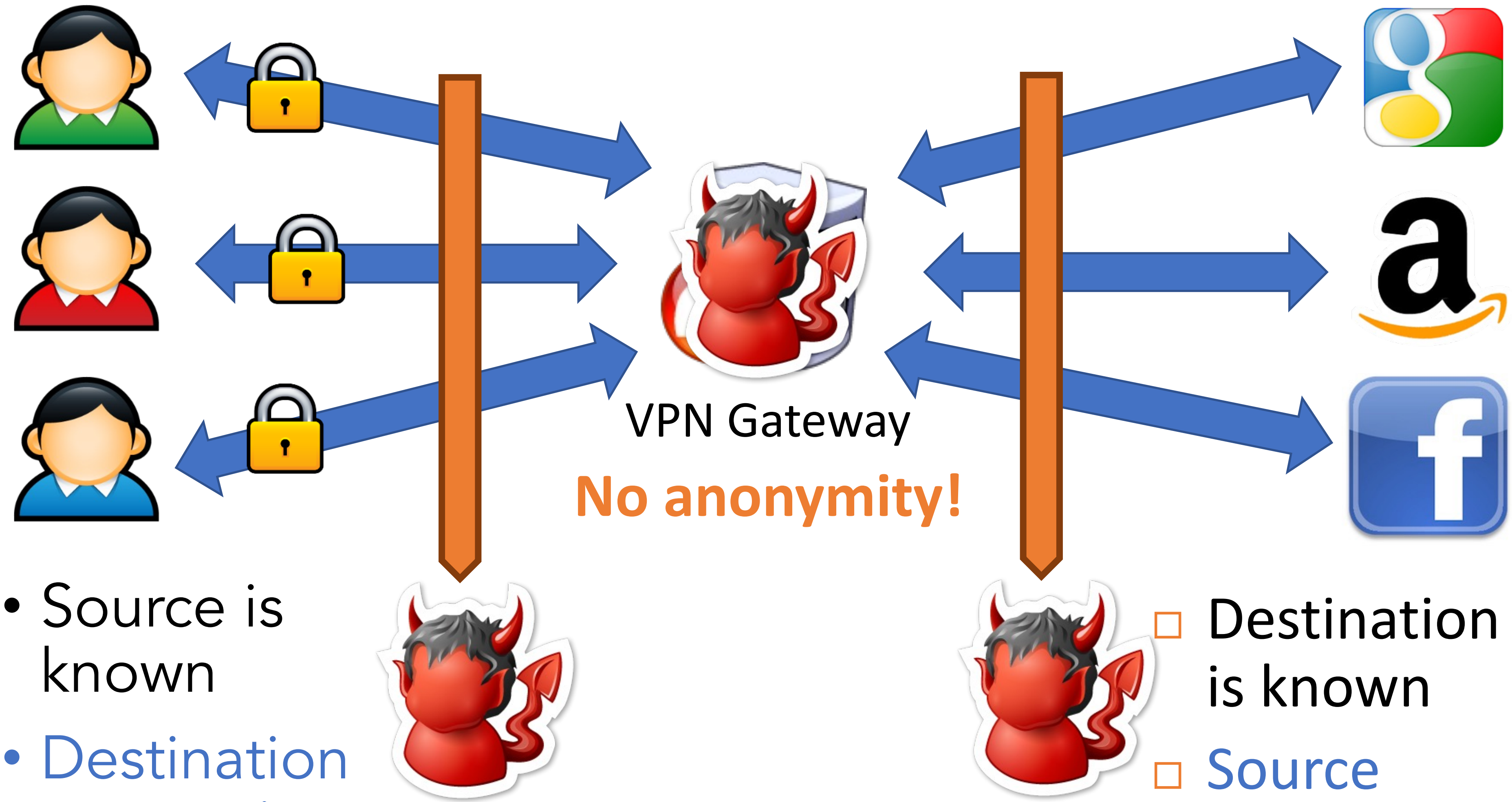
Anonymizing Proxies



- Source is known
- Destination anonymity

- ☐ Destination is known
- ☐ Source anonymity

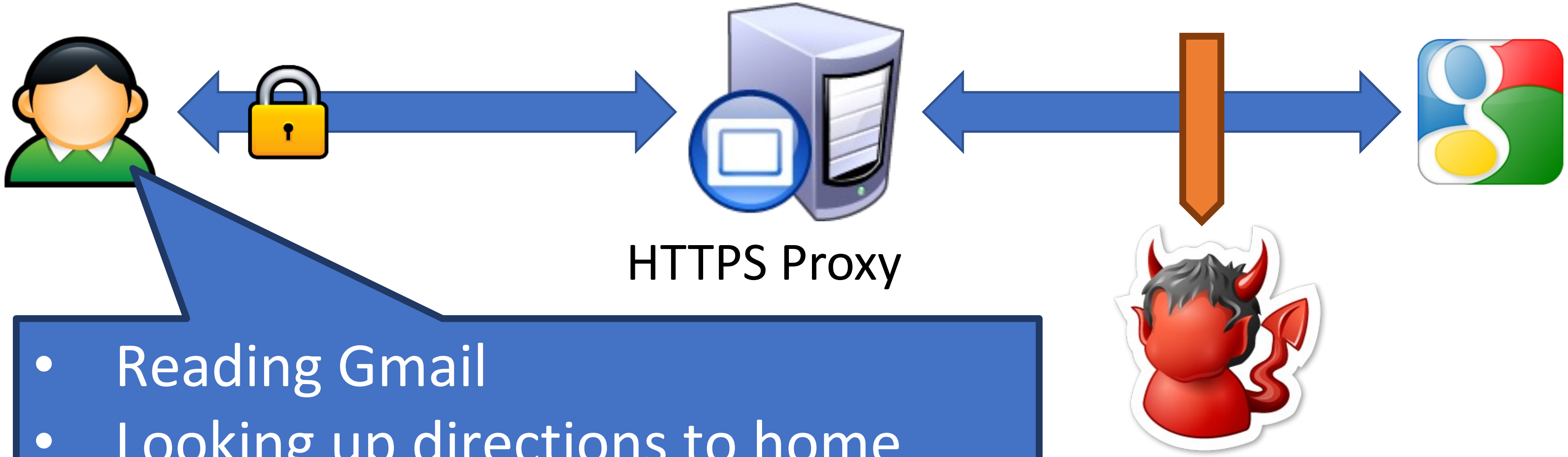
Anonymizing VPNs



- Source is known
- Destination anonymity

- Destination is known
- Source anonymity

Using Content to Deanonymize

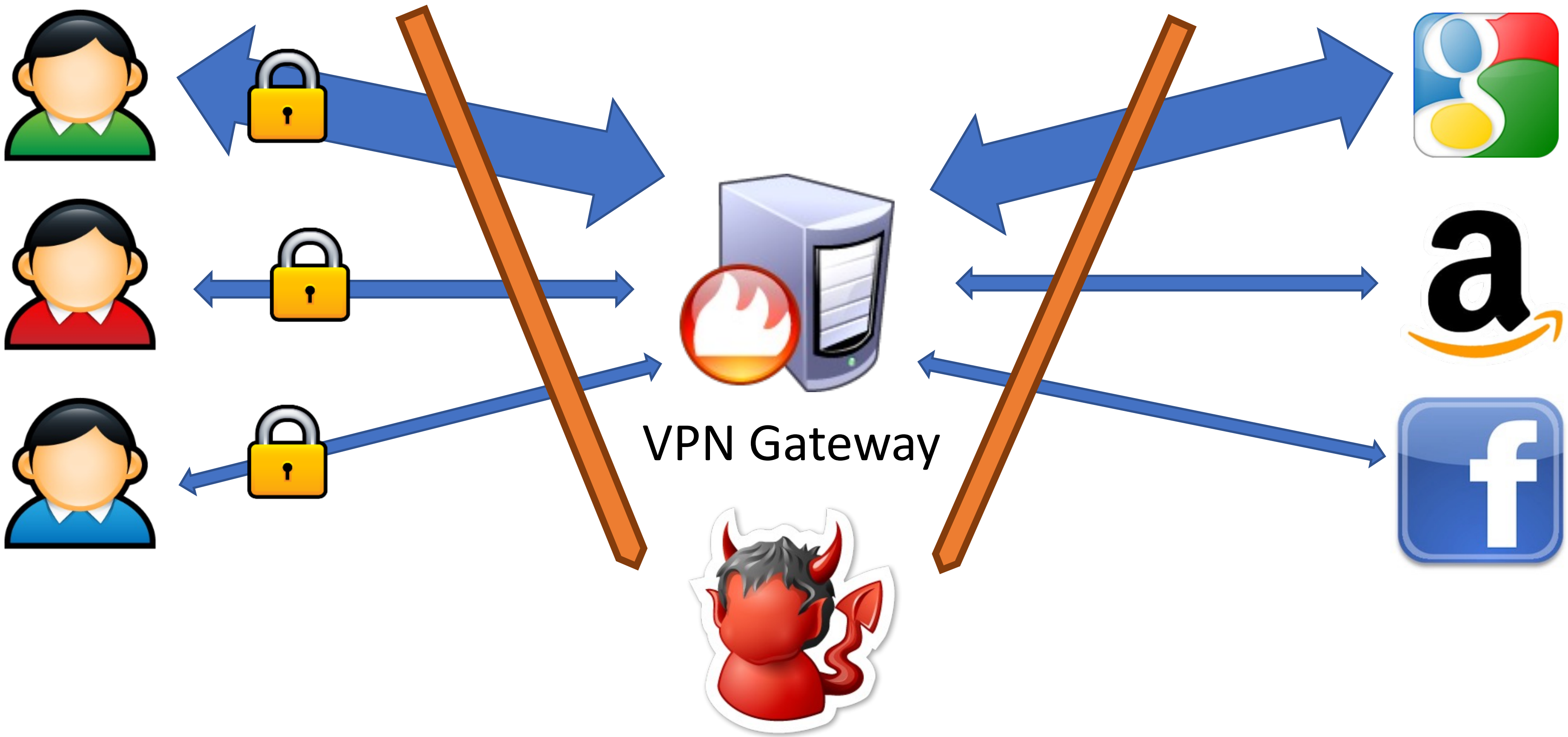


- Reading Gmail
- Looking up directions to home
- Updating your Facebook profile
- Etc…

No anonymity!

- Fact: the NSA leverages common cookies from ad networks, social networks, etc. to track users

Statistical Inference Attacks



- Statistical analysis of traffic patterns can compromise anonymity, i.e. the **timing** and/or **volume** of packets

Data To Protect

- Personally Identifiable Information (PII)
  - Name, address, phone number, etc.
- OS and browser information
  - Cookies, etc.
- Language information
- IP address
- Amount of data sent and received
- Traffic timing

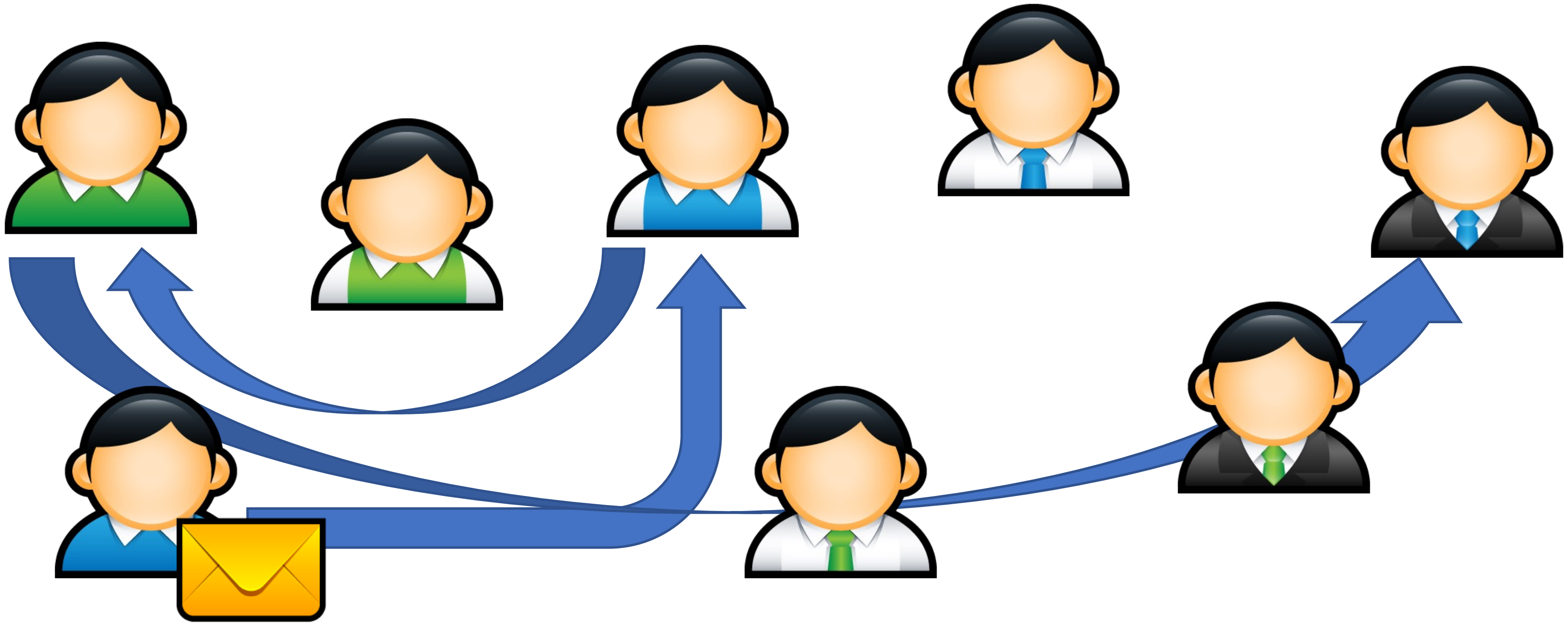
Outline

- ❑ Definitions and Examples
- ❑ Crowds
- ❑ Chaum Mix / Mix Networks
- ❑ Tor

Crowds

- Key idea
  - Users' traffic blends into a crowd of users
  - Eavesdroppers and end-hosts don't know which user originated what traffic
- High-level implementation
  - Every user runs a proxy on their system
  - Proxy is called a jondo
    - From "John Doe," i.e. an unknown person
  - When a message is received, select $x \in [0, 1]$
    - If $x > p_f$: forward the message to a random jondo
    - Else: deliver the message to the actual receiver

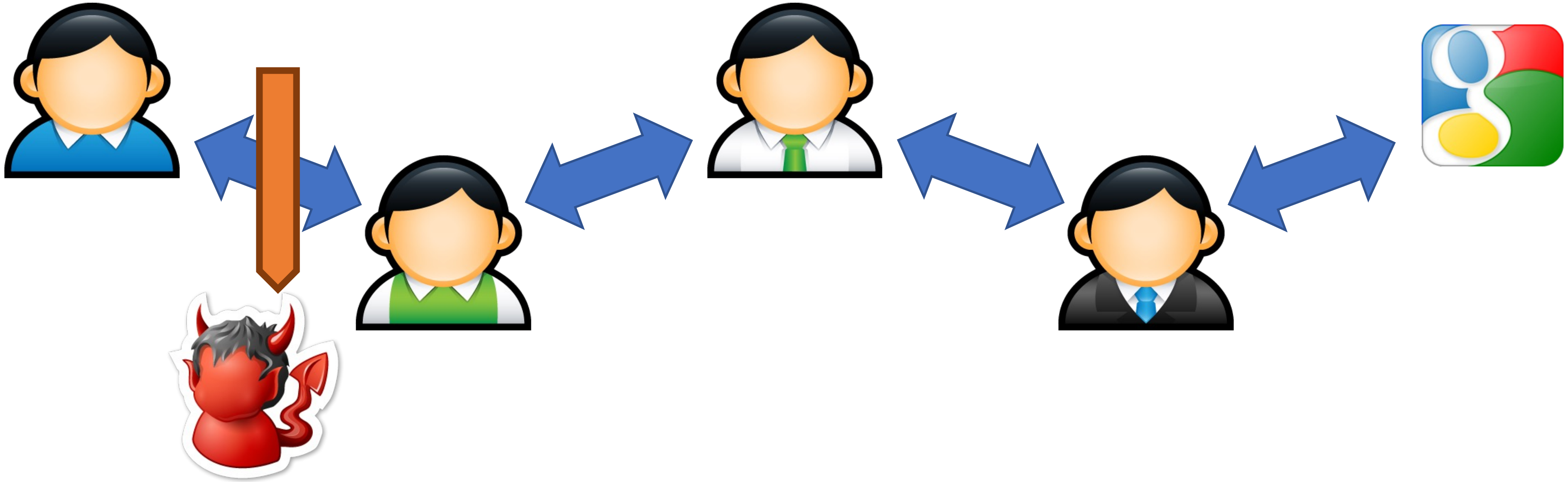
Crowds Example



- Links between users use public key crypto
- Users may appear on the path multiple times

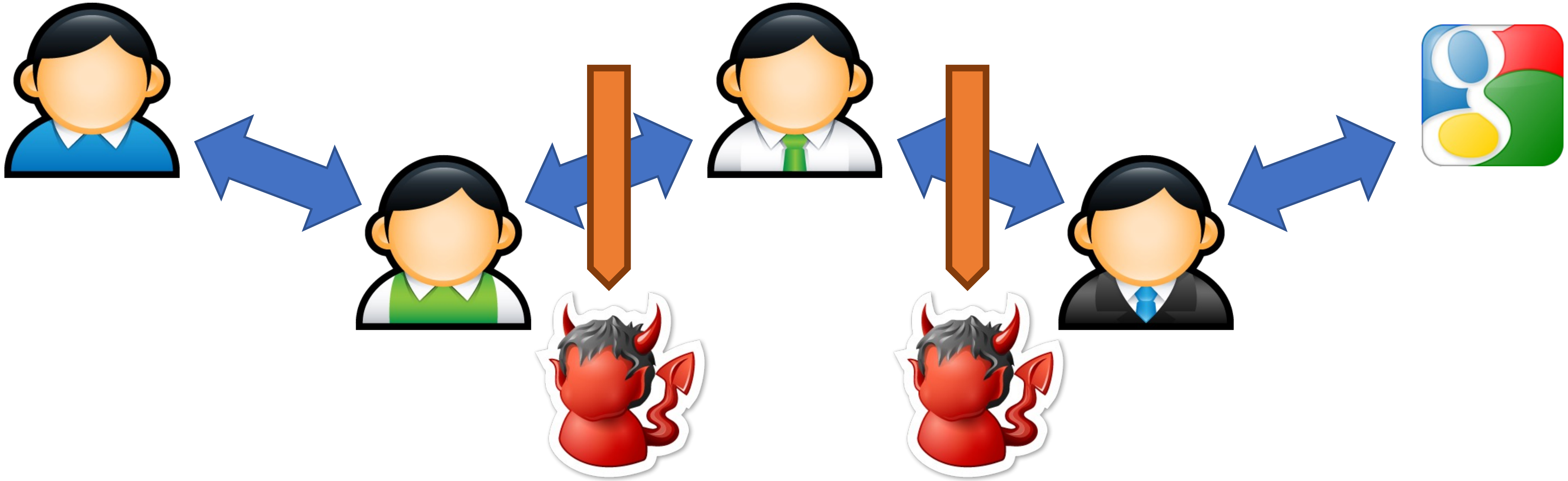
Final Destination

Anonymity in Crowds



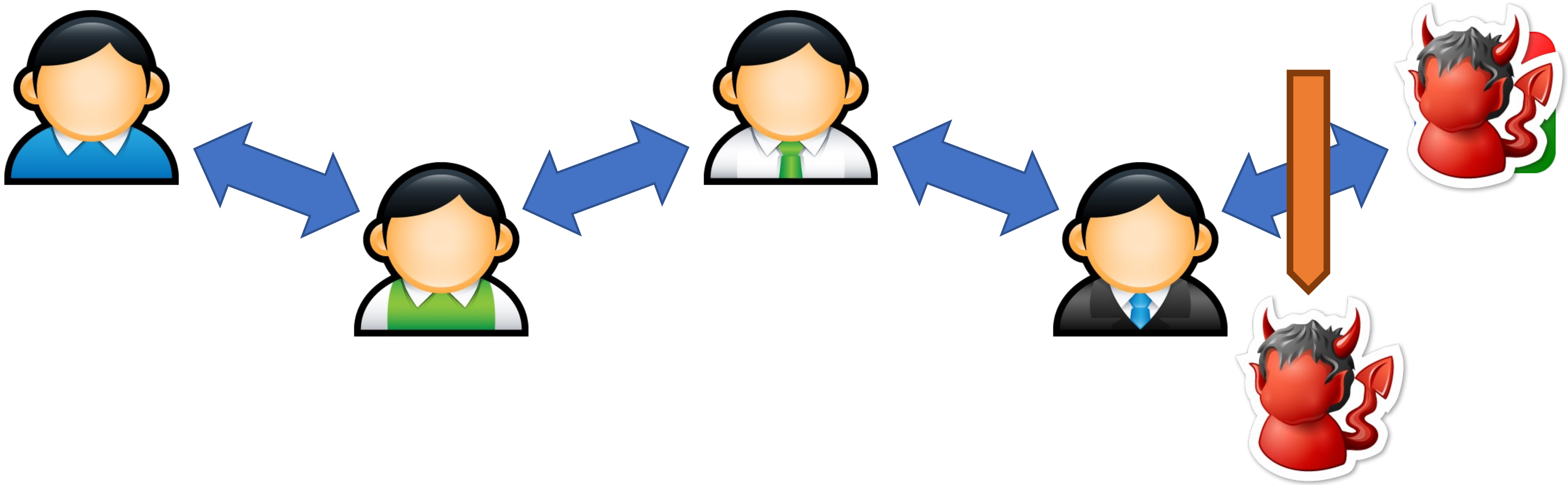
- No source anonymity
  - Target receives *m* incoming messages (*m* may = 0)
  - Target sends *m + 1* outgoing messages
  - Thus, the target is sending something
- Destination anonymity is maintained
  - If the source isn't sending directly to the receiver

Anonymity in Crowds



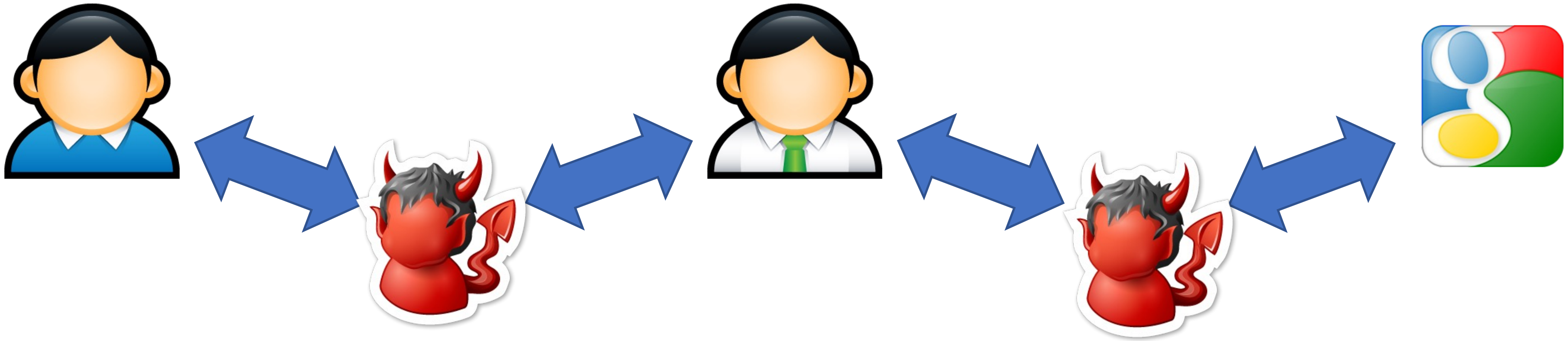
- Source and destination are anonymous
  - Source and destination are jondo proxies
  - Destination is hidden by encryption

Anonymity in Crowds



- Destination is known
  - Obviously
- Source is anonymous
  - O(n) possible sources, where n is the number of jondos

Anonymity in Crowds



- Destination is known
  - Evil jondo is able to decrypt the message
- Source is somewhat anonymous
  - Suppose there are c evil jondos and n total jondos
  - If $p_f > 0.5$, and $n > 3(c + 1)$, then the source cannot be inferred with probability > 0.5

Other Implementation Details

- Crowds requires a central server called a Blender
  - Keep track of who is running jondos
    - Kind of like a BitTorrent tracker
  - Broadcasts new jondos to existing jondos
  - Facilitates exchanges of public keys

Summary of Crowds

- The good:
  - Crowds has excellent scalability
    - Each user helps forward messages and handle load
    - More users = better anonymity for everyone
  - Strong source anonymity guarantees
- The bad:
  - Very weak destination anonymity
    - Evil jondos can always see the destination
  - Weak unlinkability guarantees

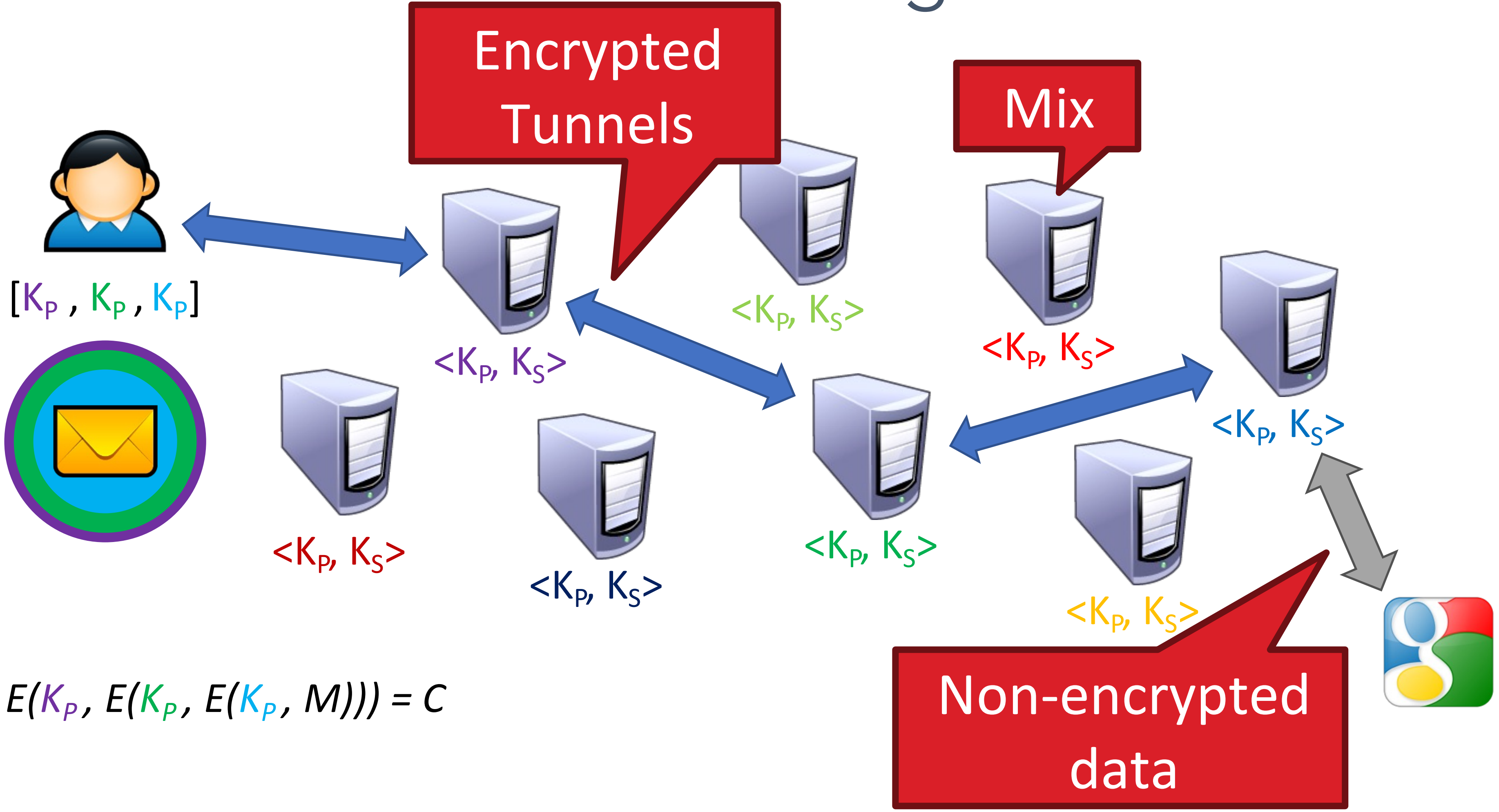
Outline

- ❑ Definitions and Examples
- ❑ Crowds
- ❑ Chaum Mix / Mix Networks
- ❑ Tor

Mix Networks

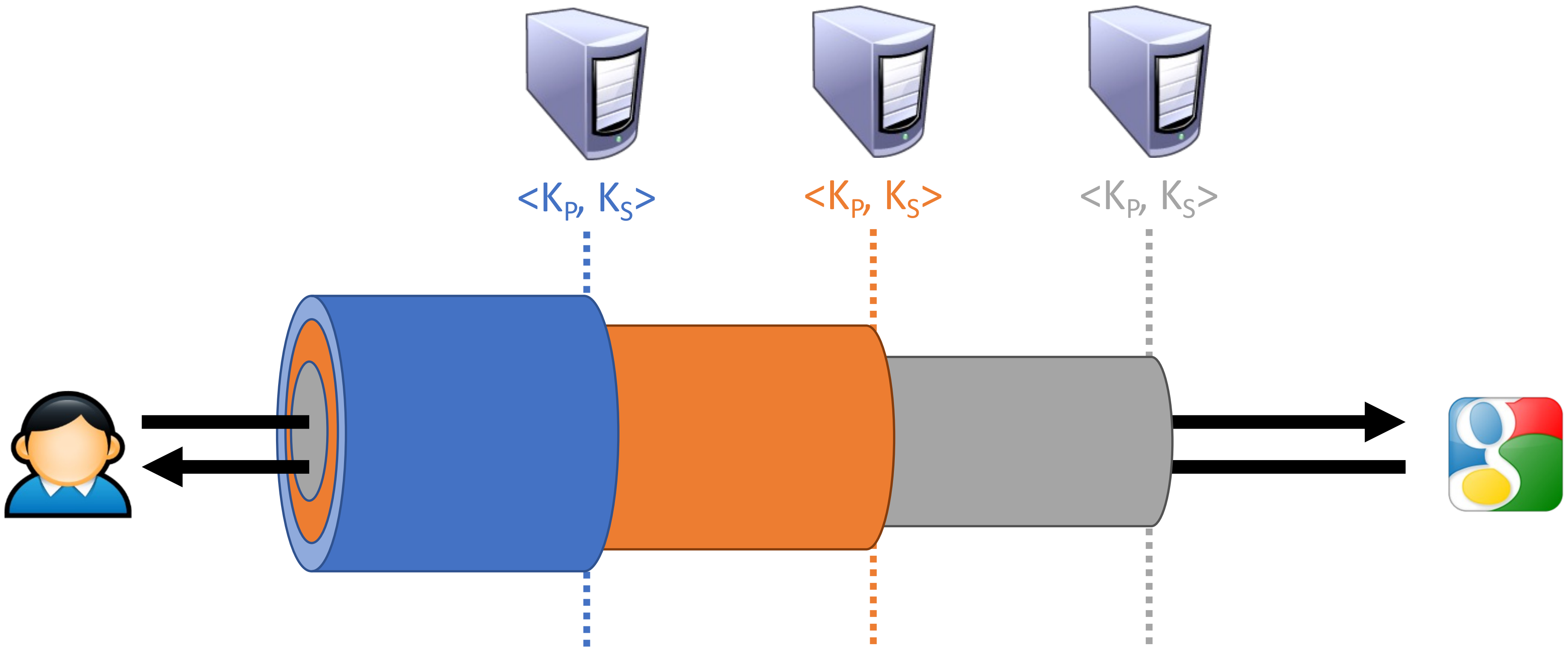
- A different approach to anonymity than Crowds
- Originally designed for anonymous email
  - David Chaum, 1981
  - Concept has since been generalized for TCP traffic
- Hugely influential ideas
  - Onion routing
  - Traffic mixing
  - Dummy traffic (a.k.a. cover traffic)

Mix Proxies and Onion Routing



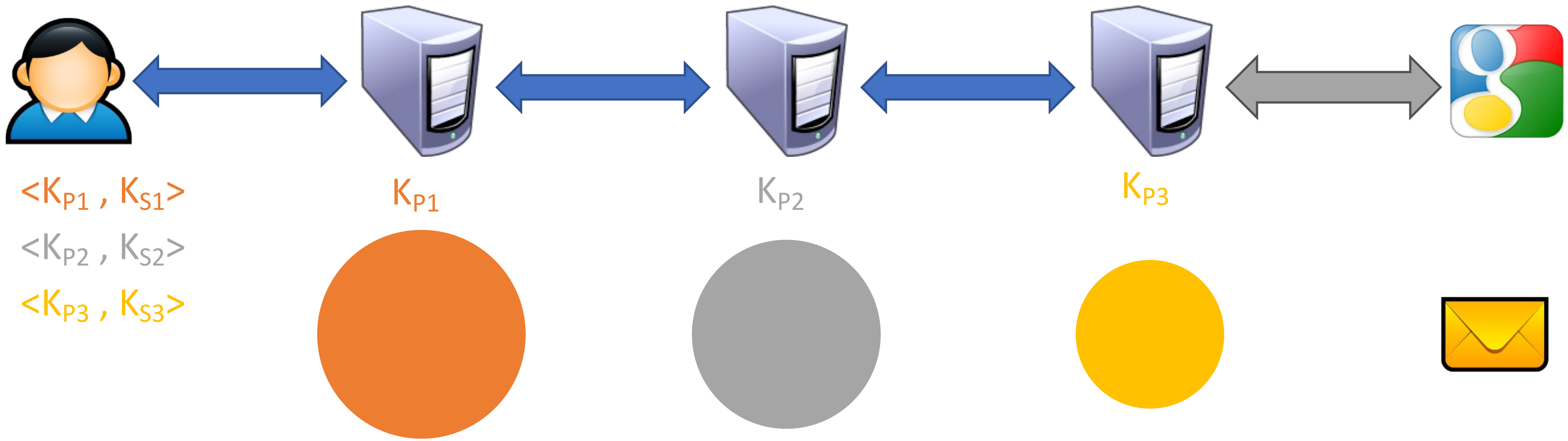
- Mixes form a cascade of anonymous proxies
- All traffic is protected with layers of encryption

Another View of Encrypted Paths



Return Traffic

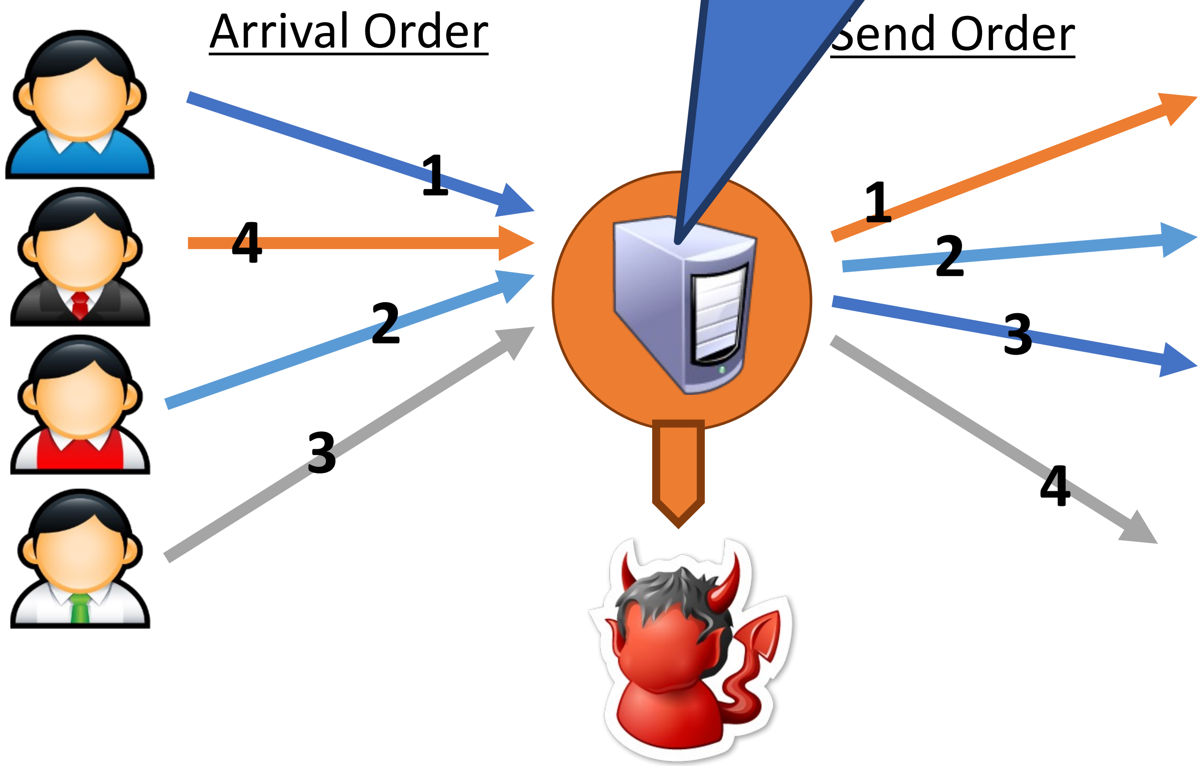
- In a mix network, how can the destination respond to the sender?
- During path establishment, the sender places keys at each mix along the path
  - Data is re-encrypted as it travels the reverse path



Traffic Mixing

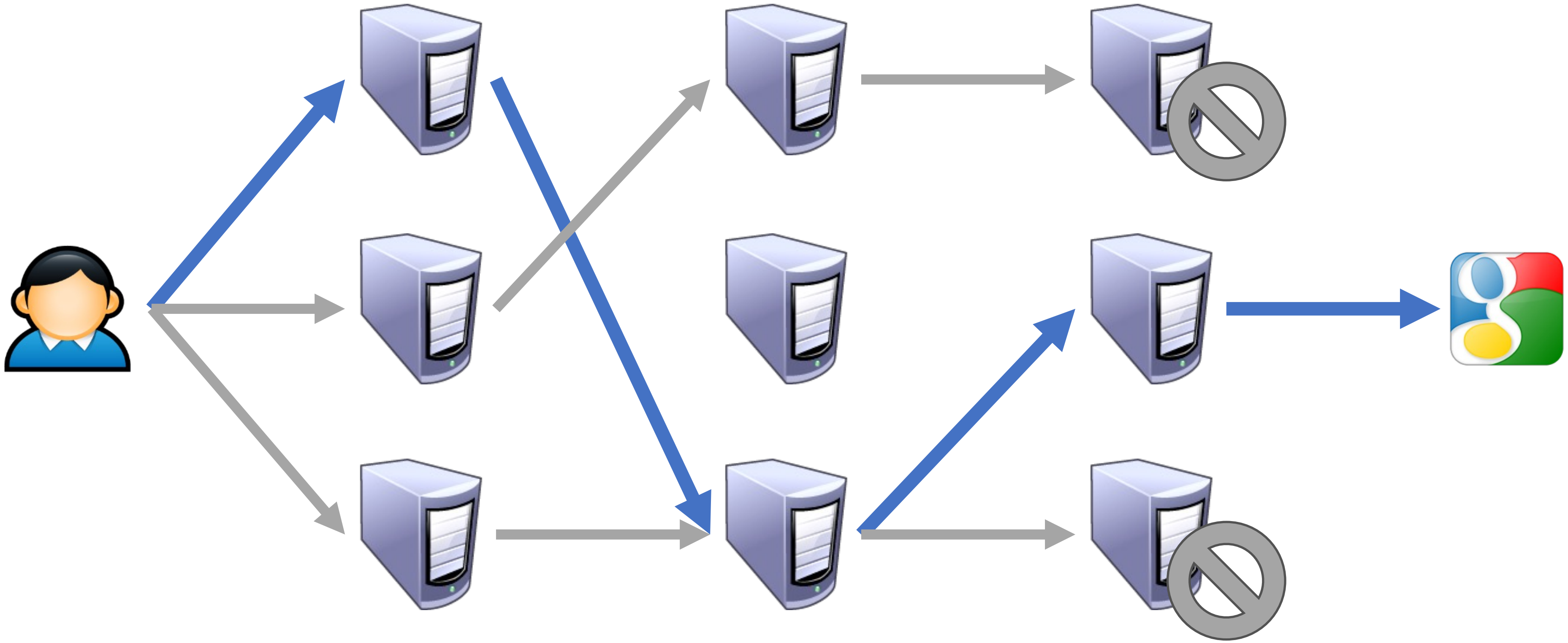
- Hinders timing attacks
  - Messages may be artificially delayed
  - Temporal correlation is warped
- Problems:
  - Requires lots of traffic
  - Adds latency to network flows

- Mix collects messages for *t* seconds
- Messages are randomly shuffled and sent in a different order



Dummy / Cover Traffic

- Simple idea:
  - Send useless traffic to help obfuscate real traffic



Legacy of Mix Networks

- Hugely influential ideas
  - Onion routing
  - Traffic mixing
  - Dummy traffic (a.k.a. cover traffic)

Outline

- ❑ Definitions and Examples
- ❑ Crowds
- ❑ Chaum Mix / Mix Networks
- ❑ Tor

Tor: The 2nd Generation Onion Router

- Basic design: a mix network with improvements
  - Perfect forward secrecy
  - Introduces guards to improve source anonymity
  - Takes bandwidth into account when selecting relays
    - Mixes in Tor are called relays
  - Introduces hidden services
    - Servers that are only accessible via the Tor overlay

Deployment and Statistics

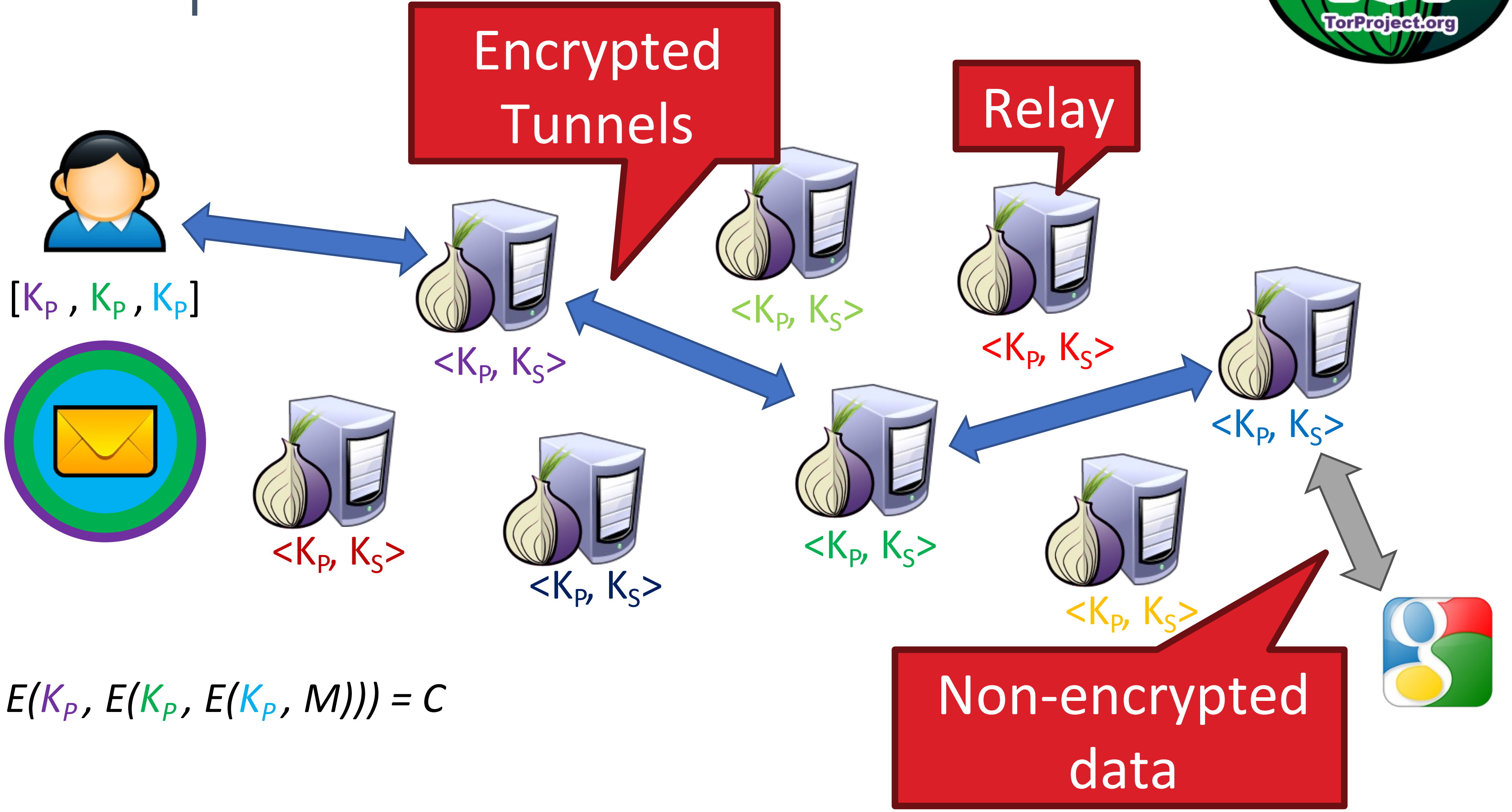
- Largest, most well deployed anonymity preserving service on the Internet
  - Publicly available since 2002
  - Continues to be developed and improved
- Currently, ~5000 Tor relays around the world
  - All relays are run by volunteers
  - It is suspected that some are controlled by intelligence agencies
- 500K – 900K daily users
  - Numbers are likely larger now, thanks to Snowden

Celebrities Use Tor

How Do You Use Tor?

1. Download, install, and execute the Tor client
   - The client acts as a SOCKS proxy
   - The client builds and maintains circuits of relays
2. Configure your browser to use the Tor client as a proxy
   - Any app that supports SOCKS proxies will work with Tor
3. All traffic from the browser will now be routed through the Tor overlay

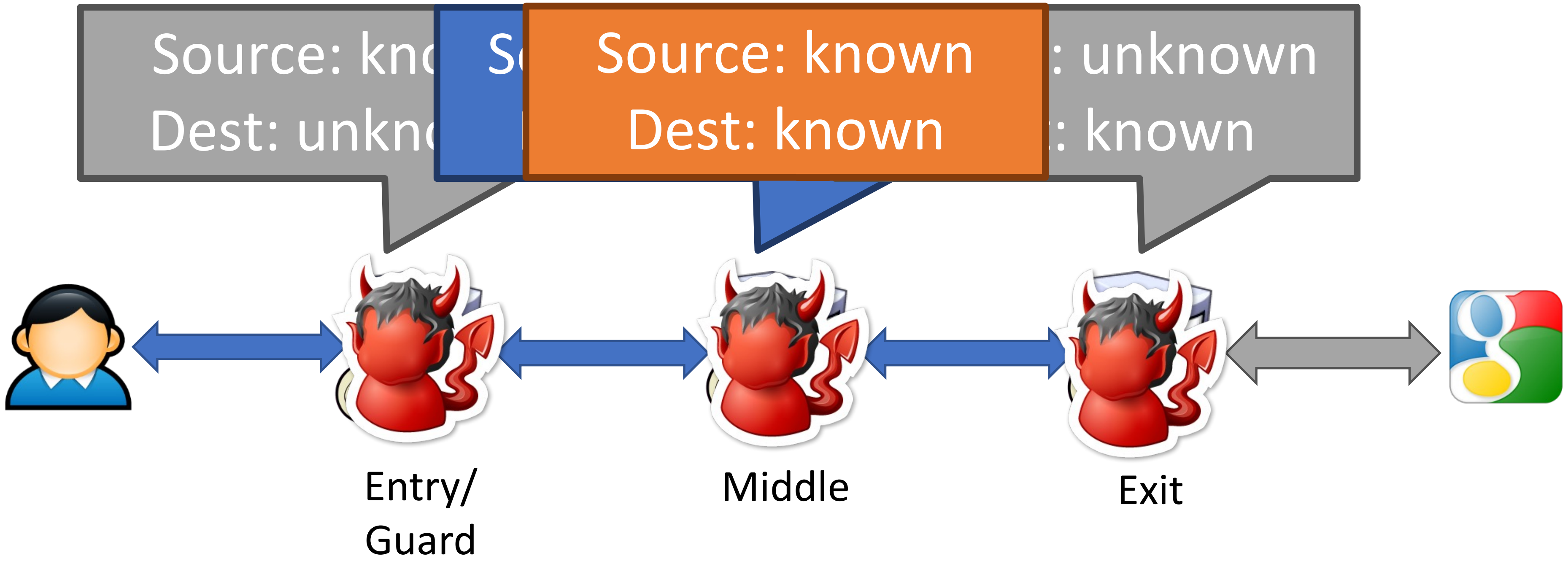
Tor Example



$E(K_P, E(K_P, E(K_P, M))) = C$

- Relays form an anonymous circuit
- All traffic is protected with layers of encryption

Attacks Against Tor Circuits



- Tor users can choose any number of relays
  - Default configuration is 3
  - Why would higher or lower number be better or worse?

Predecessor Attack

- Assumptions:
  - *N* total relays
  - *M* of which are controlled by an attacker
- Attacke
  - M/N
  - (M-1)/
  - Roug
- Howeve
  - Over time, the chances for the attacker to be in the correct positions improves!

- This is the predecessor attack
- Attacker controls the first and last relay
- Probability of being in the right positions increases over time

Circuit Lifetime

- One possible mitigation against the predecessor attack is to increase the circuit lifetime
  - E.g. suppose your circuit was persistent for 30 days
  - Attacker has 1 chance of being selected as guard and exit
- Problems?
  - If you happen to choose the attacker as guard and exit, you are screwed
  - A single attacker in the circuit (as guard or exit) can still perform statistical inference attacks
  - Tor relays are not 100% stable, long lived circuits will die
- Bottom line: long lived circuits are not a solution
  - Tor's default circuit lifetime is 10 minutes

Selecting Relays

- How do clients locate the Tor relays?
- Tor Consensus File
  - Hosted by trusted directory servers
  - Lists all known relays
    - IP address, uptime, measured bandwidth, etc.
- Not all relays are created equal
  - Entry/guard and exit relays are specially labelled
  - Why?
- Tor does not select relays randomly
  - Chance of selection is proportional to bandwidth
  - Why? Is this a good idea?

Discussion question

- Consider the scenario where you are in a censored country and the censor choses not to block Tor, the censor is the adversary, and no Tor relays exist within this country. How many Tor relays must your traffic pass through, including the exit node, to prevent the censor from blocking your traffic?

A. 1
B. 2
C. 3
D. Tor doesn’t stop this adversary

Guard Relays

- Guard relays help prevent attackers from becoming the first relay
  - Tor selects 3 guard relays and uses them for 3 months
  - After 3 months, 3 new guards are selected
- Only certain relays may become guards:
  - Have long and consistent uptimes…
  - Have high bandwidth…
  - Are manually vetted by the Tor community
- Problem: what happens if you choose an evil guard?
  - M/N chance of full compromise (i.e. source and destination)

Exit Relays

- Relays must self-elect to be exit nodes
- Why?
  - Legal problems.
  - If someone does something malicious or illegal using Tor and the police trace the traffic, the trace leads to the exit node
- Running a Tor exit is not for the faint of heart

Discussion question

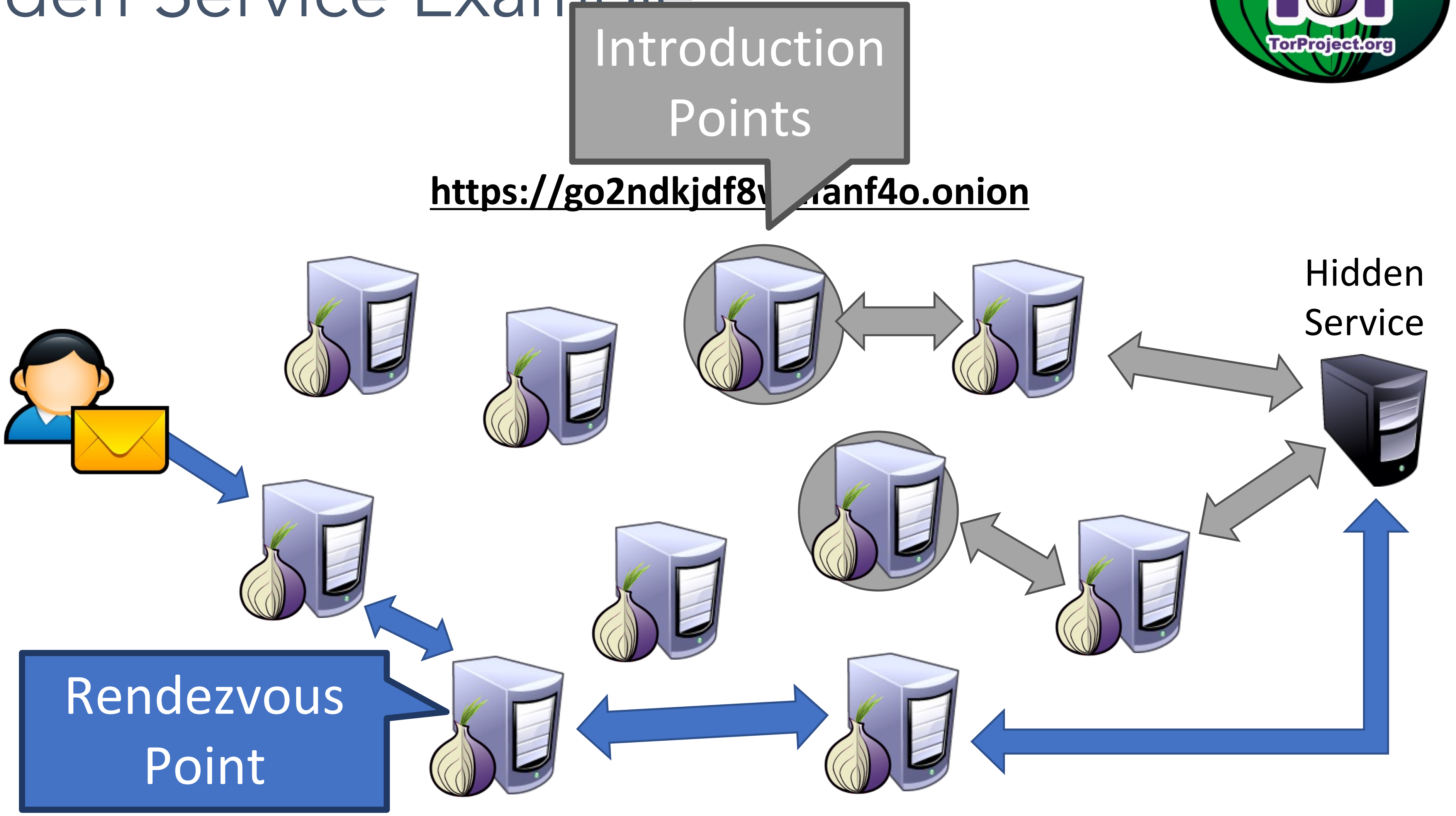
Consider the scenario where you are the only user of Tor on a network that keeps detailed logs of all IPs contacted. You use Tor to email a threat. The network operator is made aware of this threat and that it was sent through Tor and probably originated on the operator's network. How many Tor relays must your traffic pass through, including the exit node, to guarantee the network operator can't identify you as the one who sent the threat?

A. 1
B. 2
C. 3
D. Tor doesn't stop this adversary

Hidden Services

- Tor is very good at hiding the source of traffic
  - But the destination is often an exposed website
- What if we want to run an anonymous service?
  - i.e. a website, where nobody knows the IP address?
- Tor supports Hidden Services
  - Allows you to run a server and have people connect
  - … without disclosing the IP or DNS name
- Many hidden services
  - Tor Mail, Tor Char
  - DuckDuckGo
  - Wikileaks
  - The Pirate Bay
  - Silk Road (2.0? 3.0?)

Hidden Service Example



- Onion URL is a hash, allows any Tor user to find the introduction points

Perfect Forward Secrecy

- In traditional mix networks, all traffic is encrypted using public/p
- Problem
  - All futu
  - If past
- Tor imple
  - The client negotiates a new public key pair with each relay
  - Original keypairs are only used for signatures
    - i.e. to verify the authenticity of messages

- An attacker who compromises a private key can still eavesdrop on future traffic
- … but past traffic is encrypted with ephemeral keypairs that are not stored

Tor Bridges

- Anyone can look up the IP addresses of Tor relays
  - Public information in the consensus file
- Many countries block traffic to these IPs
  - Essentially a denial-of-service against Tor
- Solution: Tor Bridges
  - Essentially, Tor proxies that are not publicly known
  - Used to connect clients in censored areas to the rest of the Tor network
- Tor maintains bridges in many countries

Obfuscating Tor Traffic

- Bridges alone may be insufficient to get around all types of censorship
  - DPI can be used to locate and drop Tor frames
  - Iran blocked all encrypted packets for some time
- Tor adopts a pluggable transport design
  - Tor traffic is forwarded to an obfuscation program
  - Obfuscator transforms the Tor traffic to look like some other protocol
    - BitTorrent, HTTP, streaming audio, etc.
  - Deobfuscator on the receiver side extracts the Tor data from the encoding

Discussion question

Consider the scenario where there is a single hostile Tor node but you don't know that node's identitity, and that node can be an exit node. You want to keep confidential from this node what HTTP sites you are visiting through Tor. How many Tor relays must your traffic pass through, including the exit node, to guarantee this adversary can't know what sites you visit?

A. 1
B. 2
C. 3
D. Tor doesn't stop this adversary

Conclusions

- Presented a brief overview of popular anonymity systems
  - How do they work?
  - What are the anonymity guarantees?
- Introduced Tor
- Lots more work in anonymous communications
  - Dozens of other proposed systems
    - Tarzan, Bluemoon, etc.
  - Many offer much stronger anonymity than Tor
  - … however, performance is often a problem

Anonymous P2P Networks

- Goal: enable censorship resistant, anonymous communication and file storage
  - Content is generated anonymously
  - Content is stored anonymously
  - Content is highly distributed and replicated, making it difficult to destroy
- Examples
  - FreeNet
  - GNUnet

Sources

1. Crowds: http://avirubin.com/crowds.pdf
2. Chaum mix: http://www.ovmj.org/GNUnet/papers/p84-chaum.pdf
3. Tor: https://svn.torproject.org/svn/projects/design-paper/tor-design.pdf
4. Predecessors attack: http://prisms.cs.umass.edu/brian/pubs/wright-tissec.pdf