CS232

Lecture 21: Anonymous Communications

November 21, 2018

"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"
  - ~1% of all Internet traffic
  - Focus on encrypted traffic

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

Outline

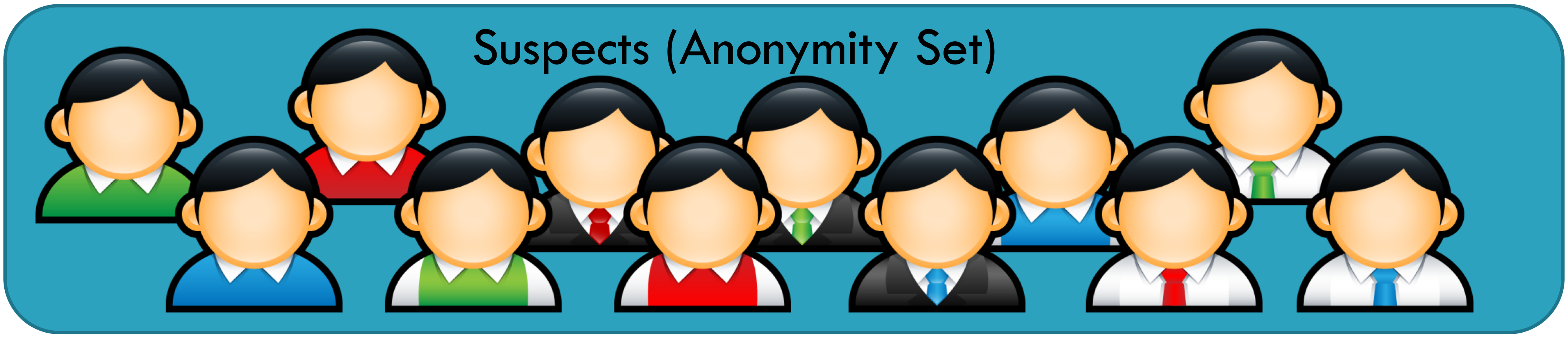
- Definitions and Examples
- Crowds
- Chaum Mix / Mix Networks
- Tor

What is Anonymity?

- Informally: can’t tell who did what…
  - Who wrote this blog post?
  - Who’s been reading my webpages
  - Who’s been emailing patent attorneys?

More Formally: Quantifying Anonymity

- Indistinguishability within an 'anonymous set'
  - Basic anonymity set size; probability distribution within set

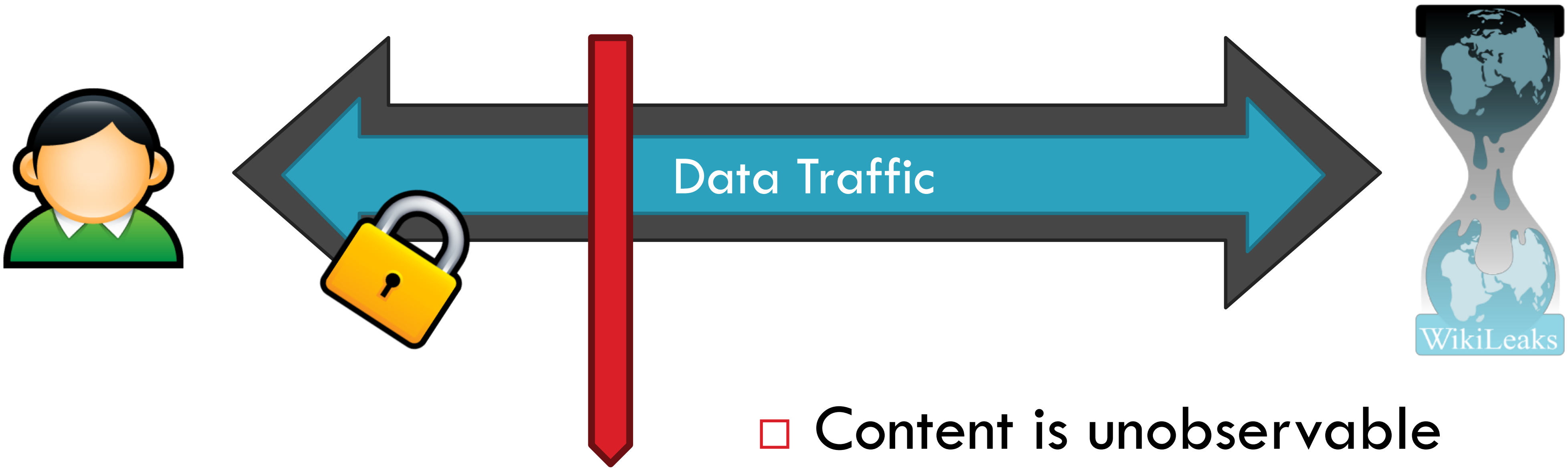


- Larger anonymity set = stronger anonymity

Other Definitions

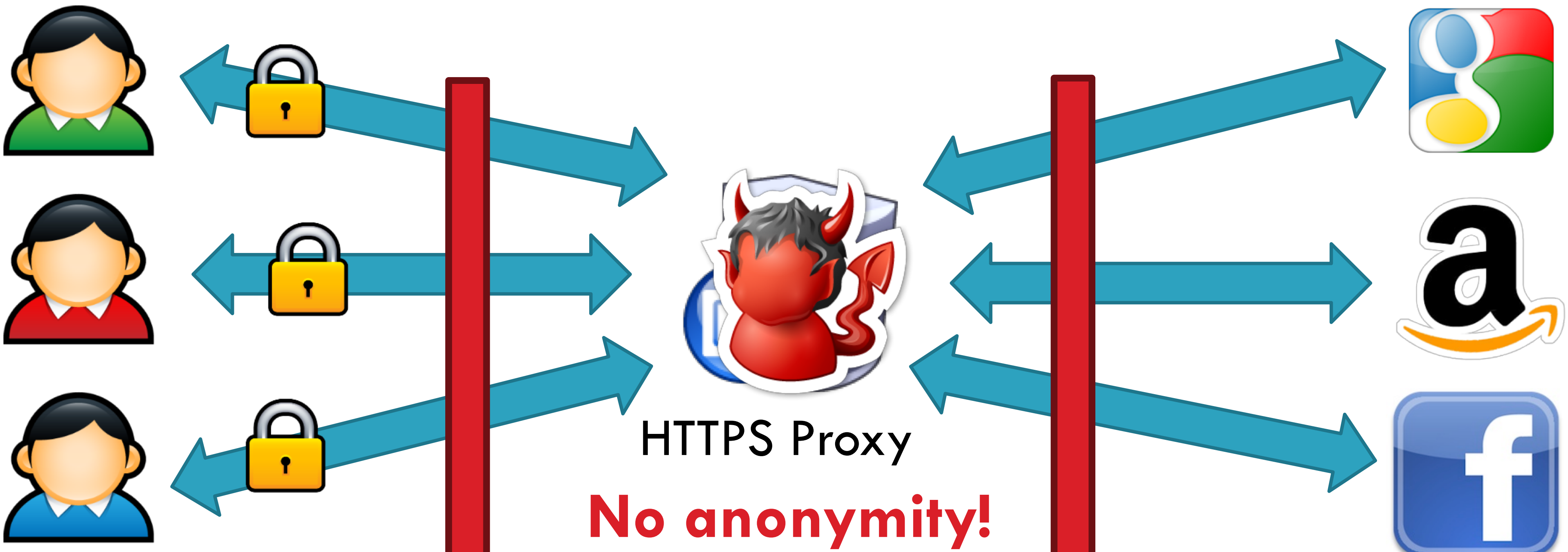
- Unlinkability
  - From the adversaries perspective, the inability the link two or more items of interest
    - E.g. packets, events, people, actions, etc.
  - Three parts:
    - Sender anonymity (who sent this?)
    - Receiver anonymity (who is the destination?)
    - Relationship anonymity (are sender 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!

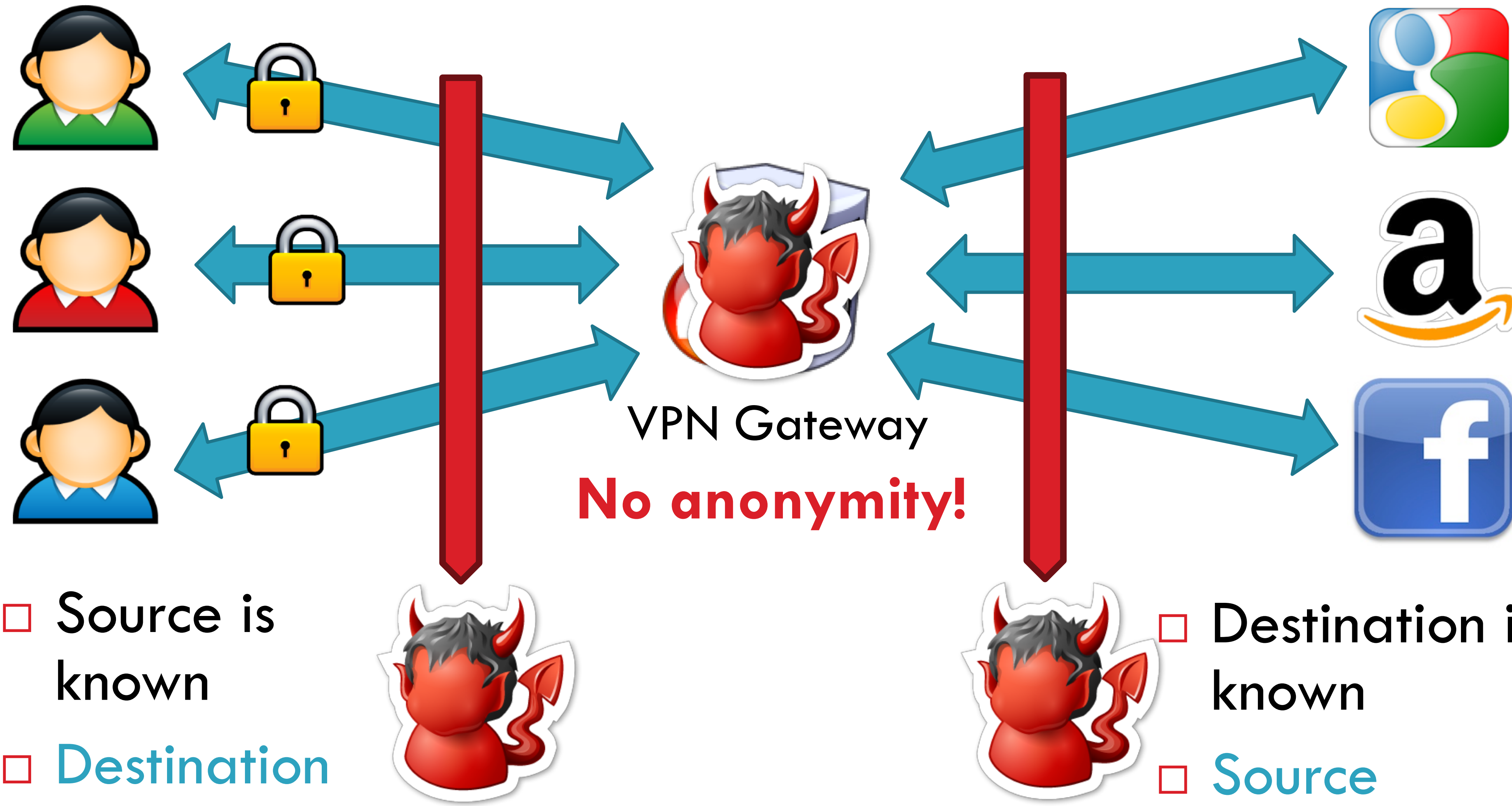
Anonymizing Proxies



- Source is known
- Destination anonymity

- Destination is known
- Source anonymity

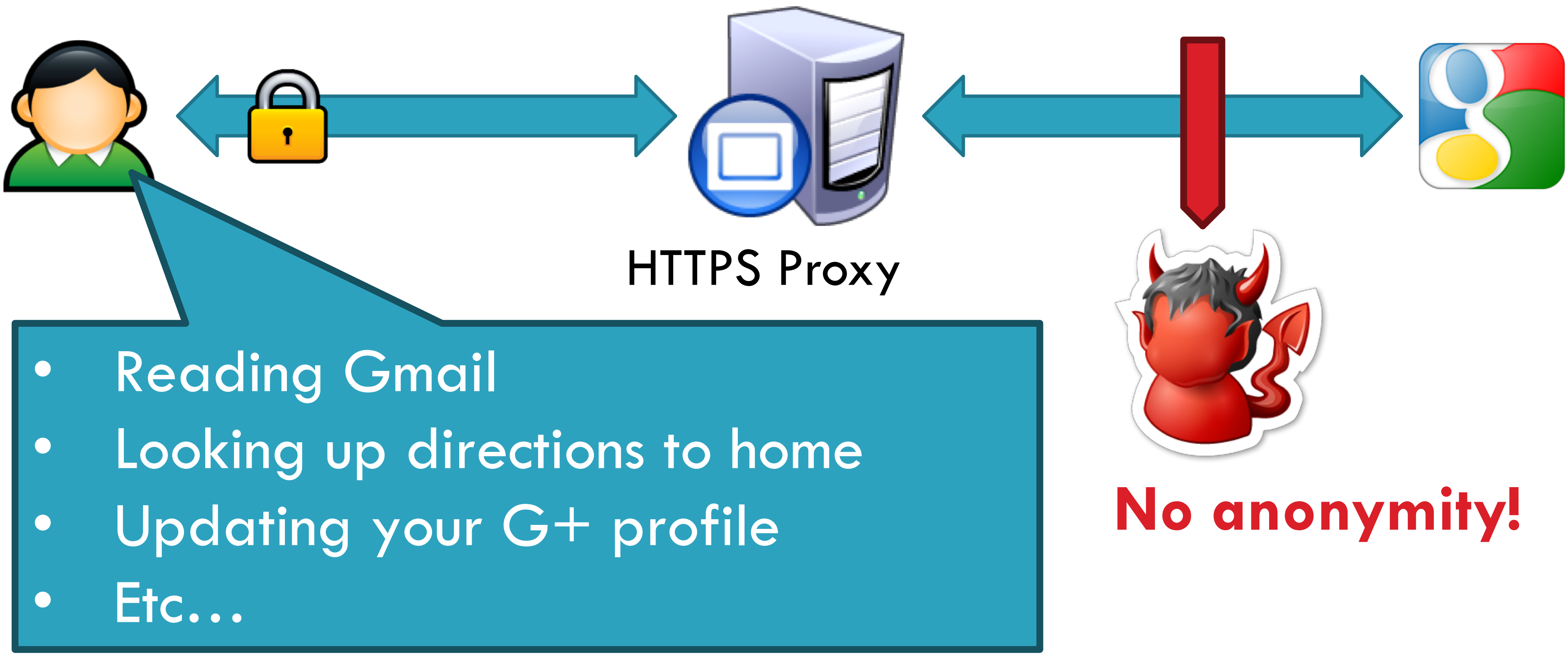
Anonymizing VPNs



- Source is known
- Destination anonymity

- Destination is known
- Source anonymity

Using Content to Deanonymize



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

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
- DCs and Crowds
- Chaum Mix / Mix Networks
- Tor

Dining Cryptographers

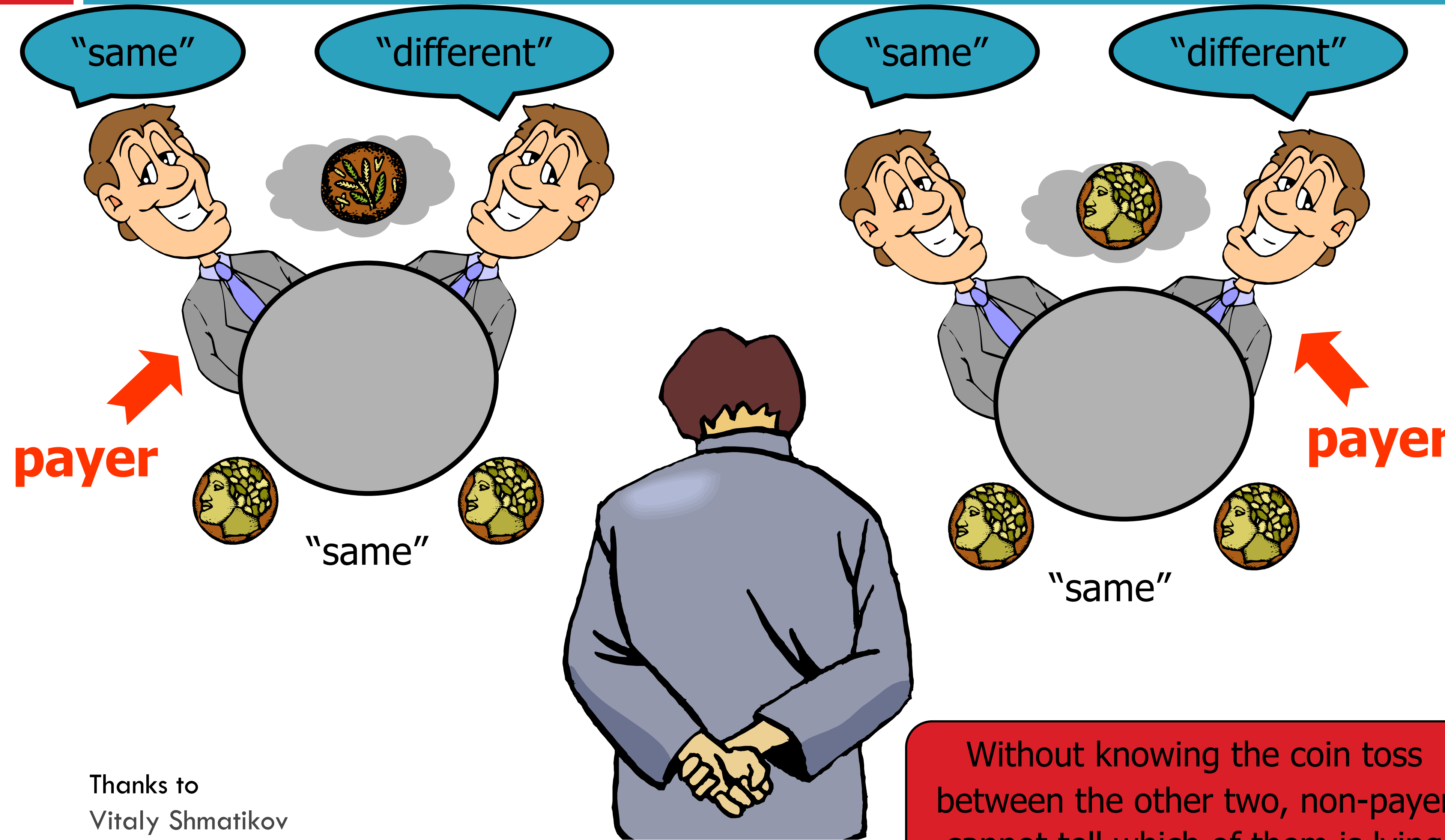
- Clever idea how to make a message public in a perfectly untraceable manner
  - David Chaum. "The dining cryptographers problem: unconditional sender and recipient untraceability." Journal of Cryptology, 1988.
- Guarantees information-theoretic anonymity for message senders
  - Unusually strong form of security: defeats adversary who has unlimited computational power
- Impractical, requires huge amount of randomness
  - In group of size *N*, need *N* random bits to send 1 bit

Three-Person DC Protocol

- Three cryptographers are having dinner. Either NSA is paying for the dinner or one of them is paying, but wishes to remain anonymous.

1. Each diner flips a coin and shows it to his left neighbor
   - Every diner will see 2 coins: her own and her right neighbor's
2. Each diner announces whether the two coins are the same. If she is the payer, she lies (says opposite)
3. Odd number of "same" $\Rightarrow$ NSA is paying;
   - Even number of "same" $\Rightarrow$ one of them is paying
   - But a non-payer cannot tell which of the other two is paying!

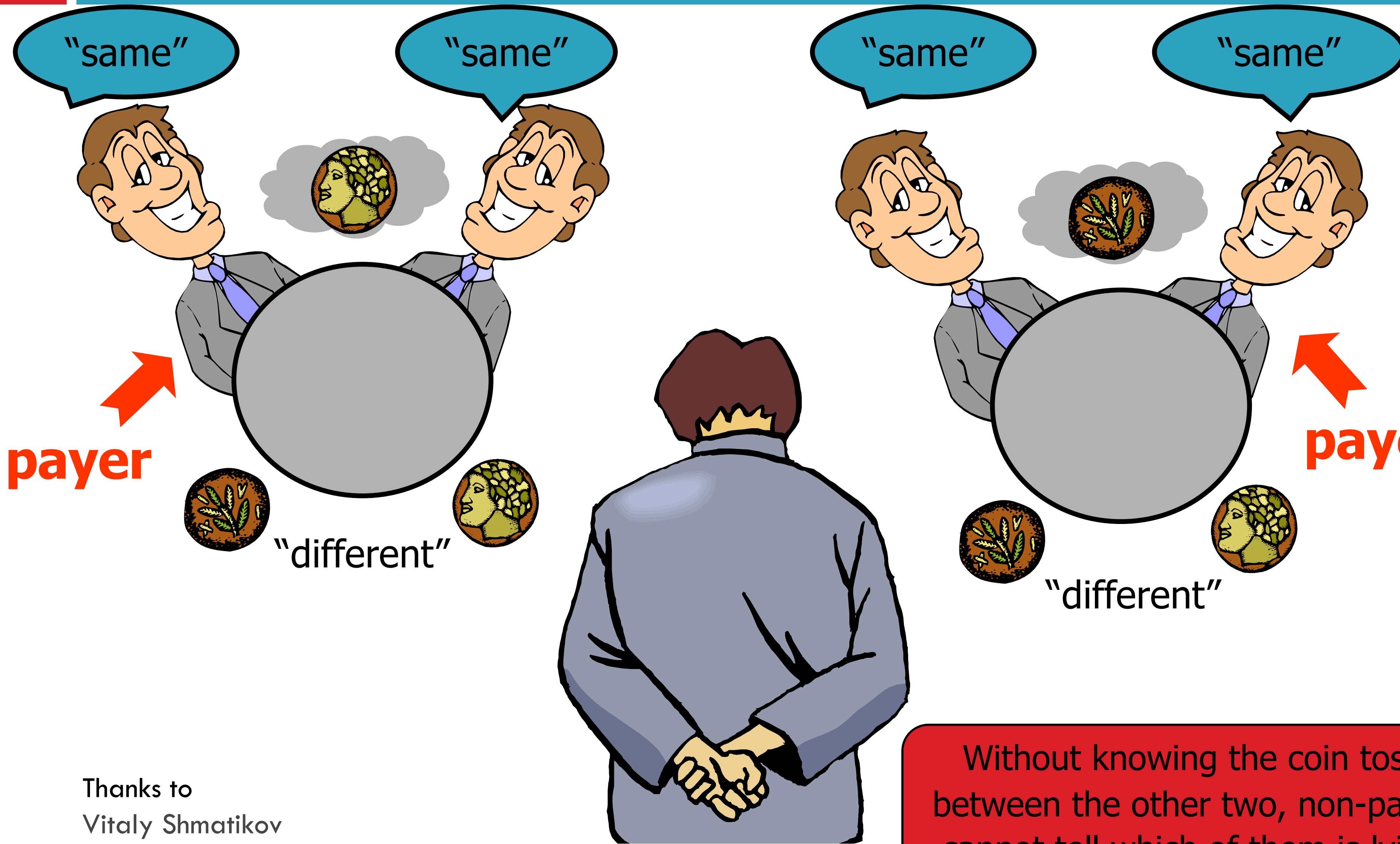
Non-Payer's View: Same Coins



Without knowing the coin toss between the other two, non-payer cannot tell which of them is lying

Thanks to
Vitaly Shmatikov

Non-Payer's View: Different Coins



Without knowing the coin toss between the other two, non-payer cannot tell which of them is lying

Thanks to
Vitaly Shmatikov

Sending Data via DC-Nets

- Generalize network to any group of size N
- For each bit of message, every user generates 1 random bit and sends it to 1 neighbor
  - Every user learns 2 bits (his own and his neighbor's)
- Encode message bit by bit
  - Each user announces (own bit XOR neighbor's bit)
  - Sender announces (own bit XOR neighbor's bit XOR message bit)
- XOR of all announcements = message bit
  - Every randomly generated bit occurs in this sum twice (and is canceled by XOR), message bit occurs once

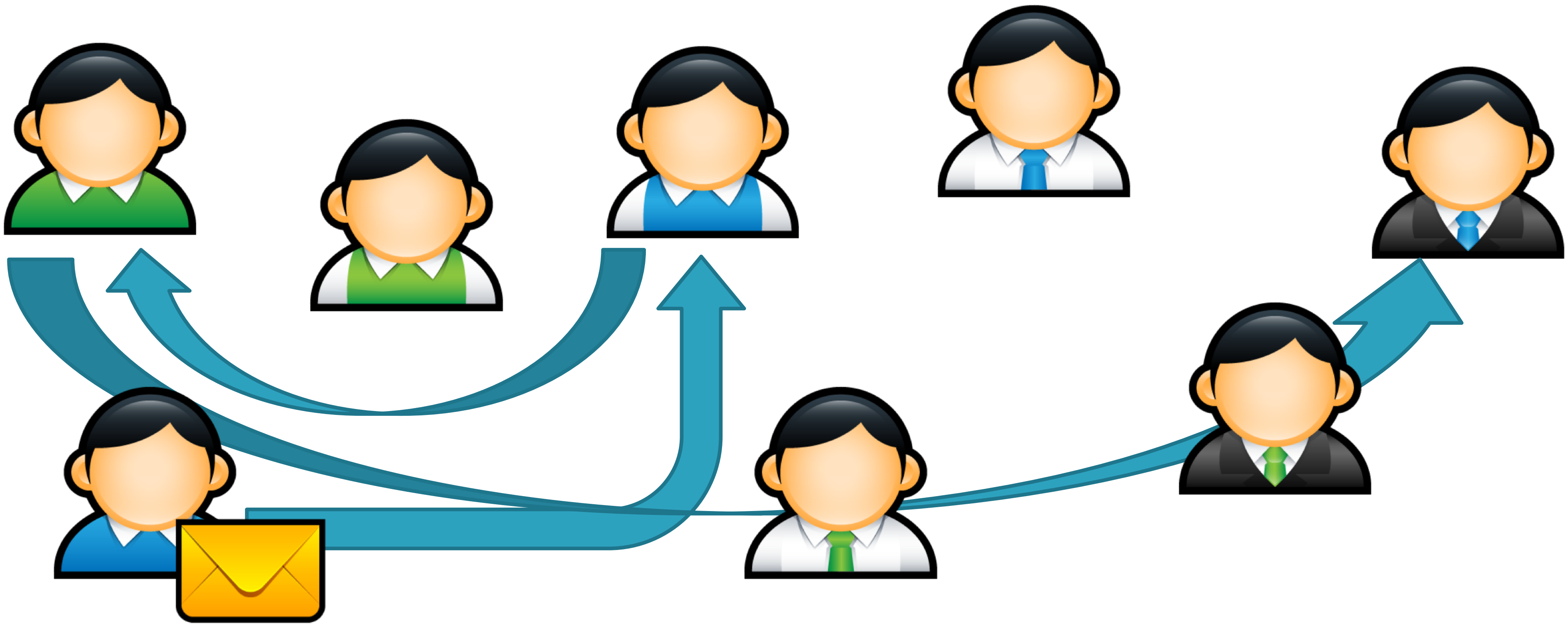
DC-Based Anonymity is Impractical

- Requires secure pairwise channels between group members
  - Otherwise, random bits cannot be shared
- Requires massive communication overhead and large amounts of randomness
- DC-net (a group of dining cryptographers) is robust even if some members cooperate (collude)
  - Guarantees perfect anonymity for the other members
- A great protocol to analyze
  - Difficult to reason about each member's knowledge

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

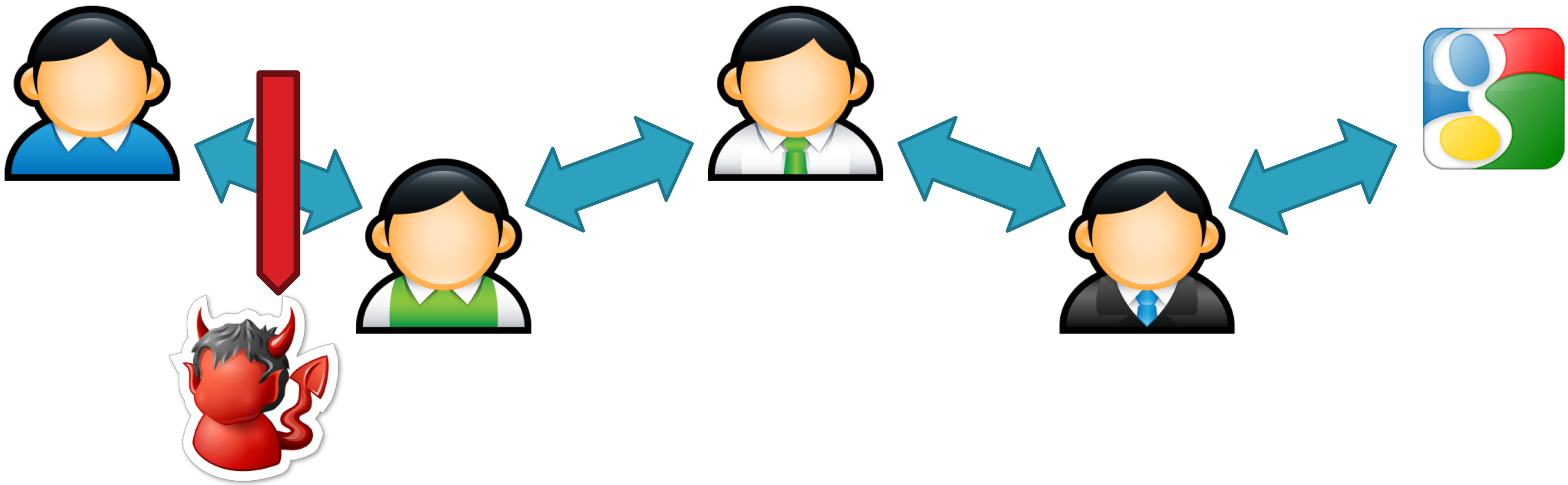
Crowds Example



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

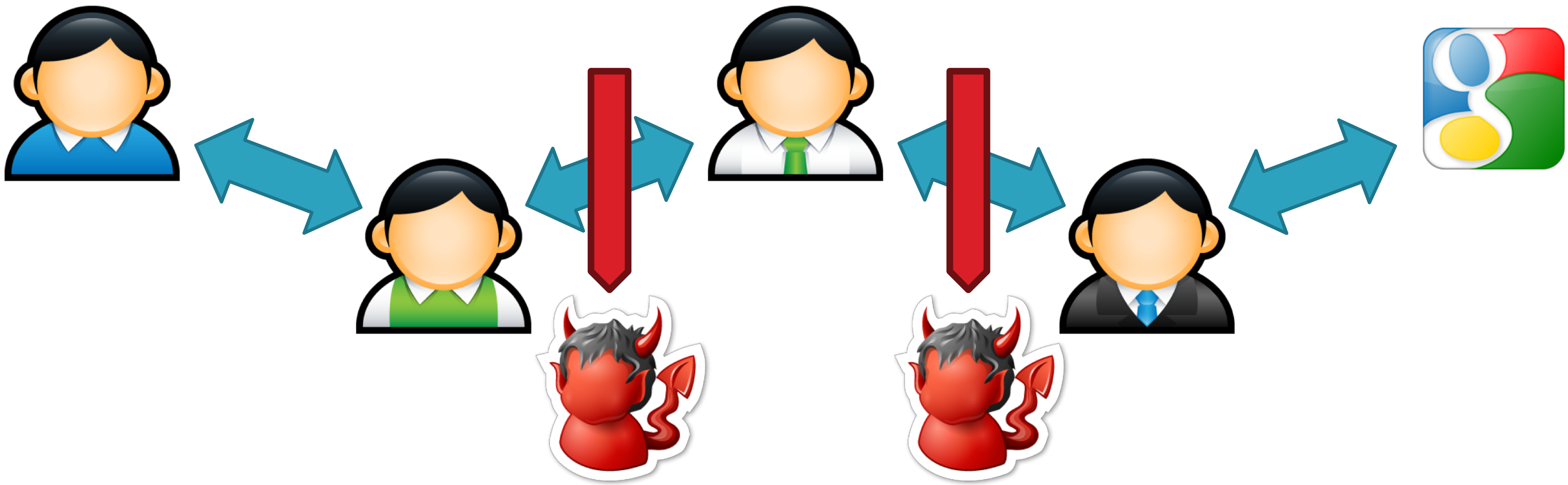
Final Destination

Anonymity in Crowds



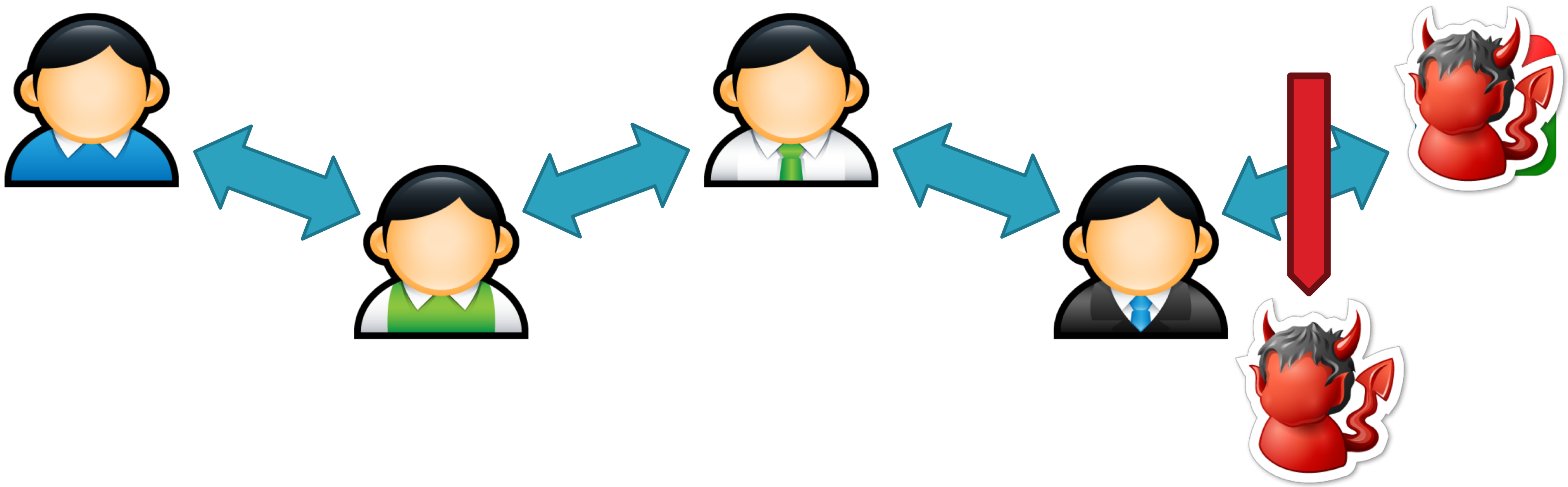
- No source anonymity
  - Target receives *m* incoming messages (*m* may = 0)
  - Target sends *m + 1* outgoing messages
  - Thus, the target is sending something
- Destination anonymity is maintained
  - If the source isn't sending directly to the receiver

Anonymity in Crowds



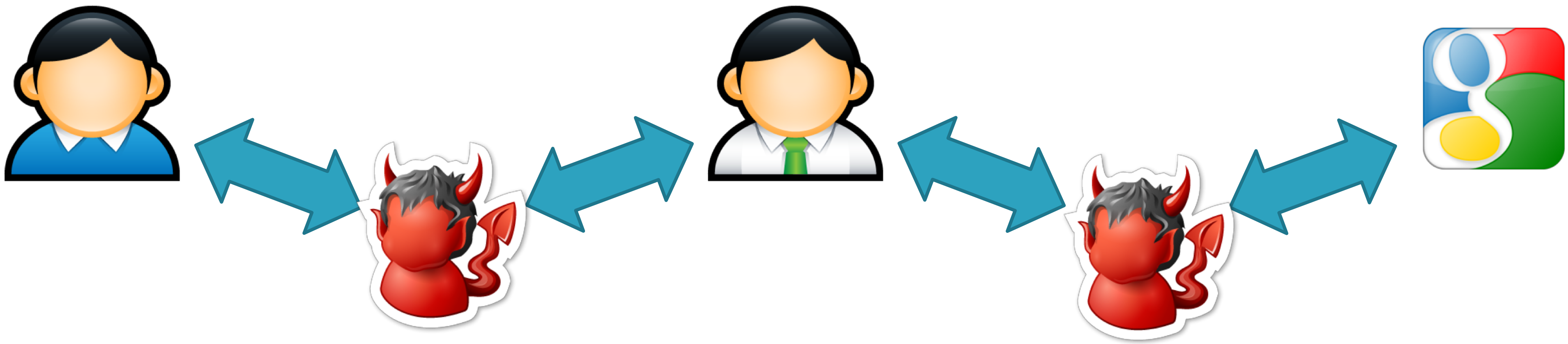
- Source and destination are anonymous
  - Source and destination are jondo proxies
  - Destination is hidden by encryption

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 in the system
  - 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

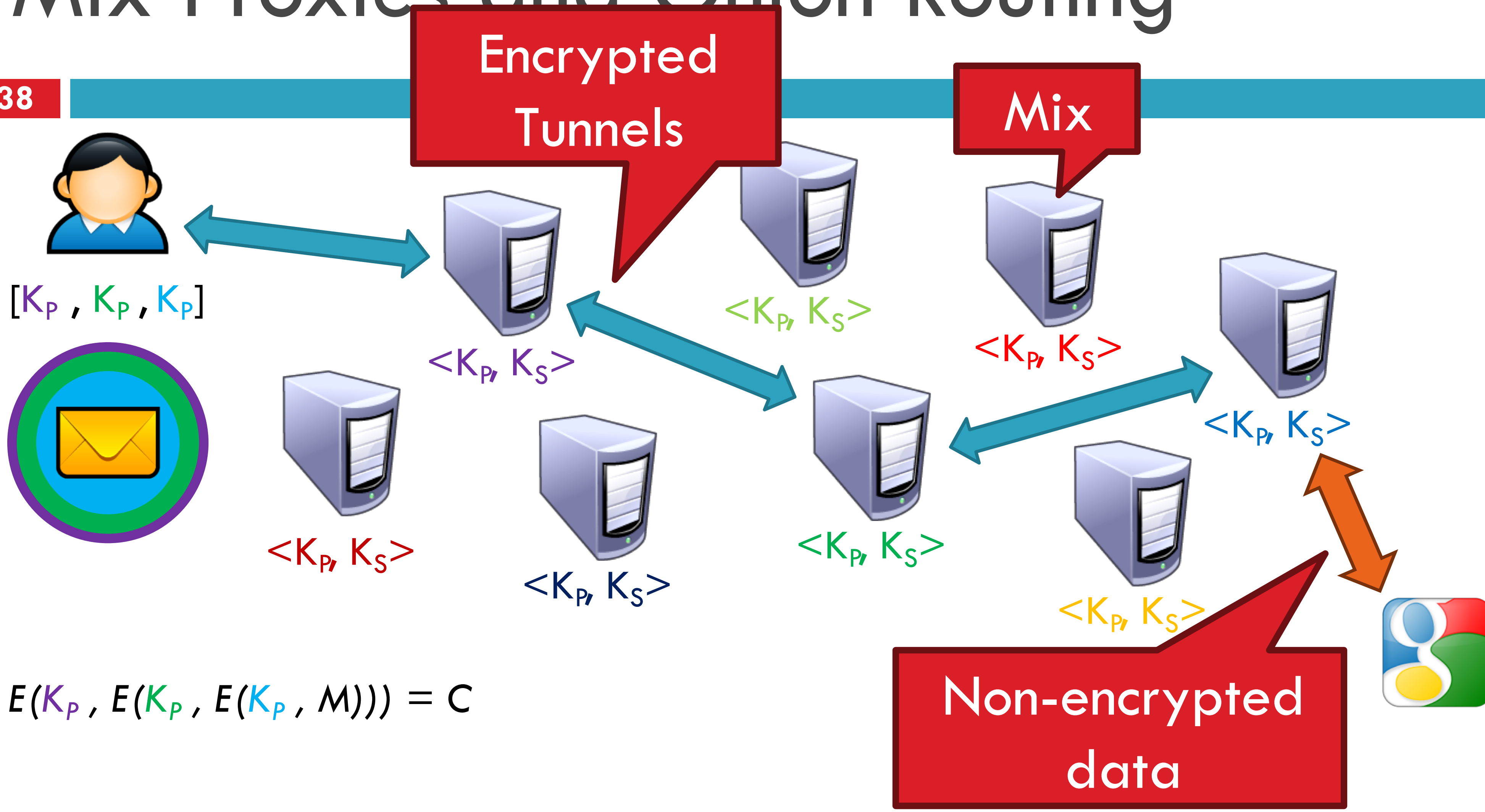
Outline

- 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)

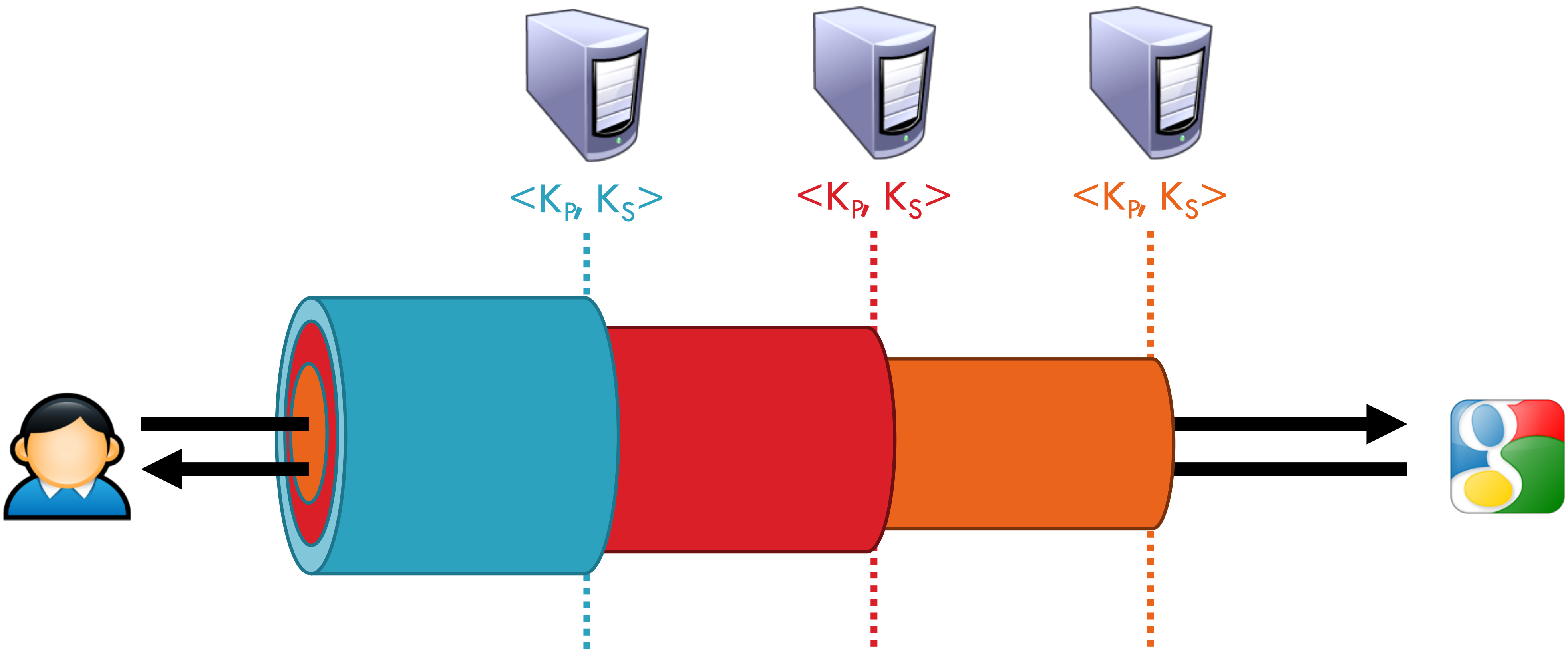
Mix Proxies and Onion Routing



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

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

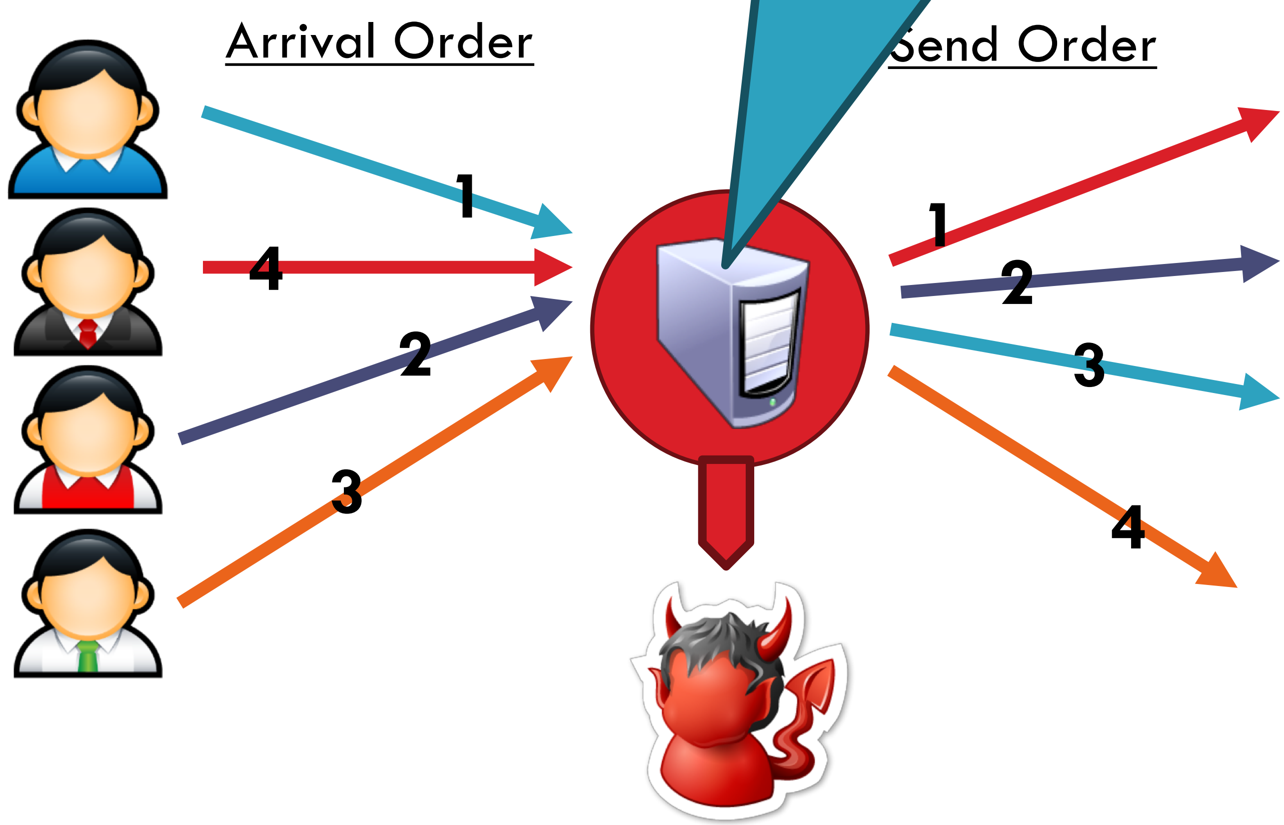
Another View of Encrypted Paths



Traffic Mixing

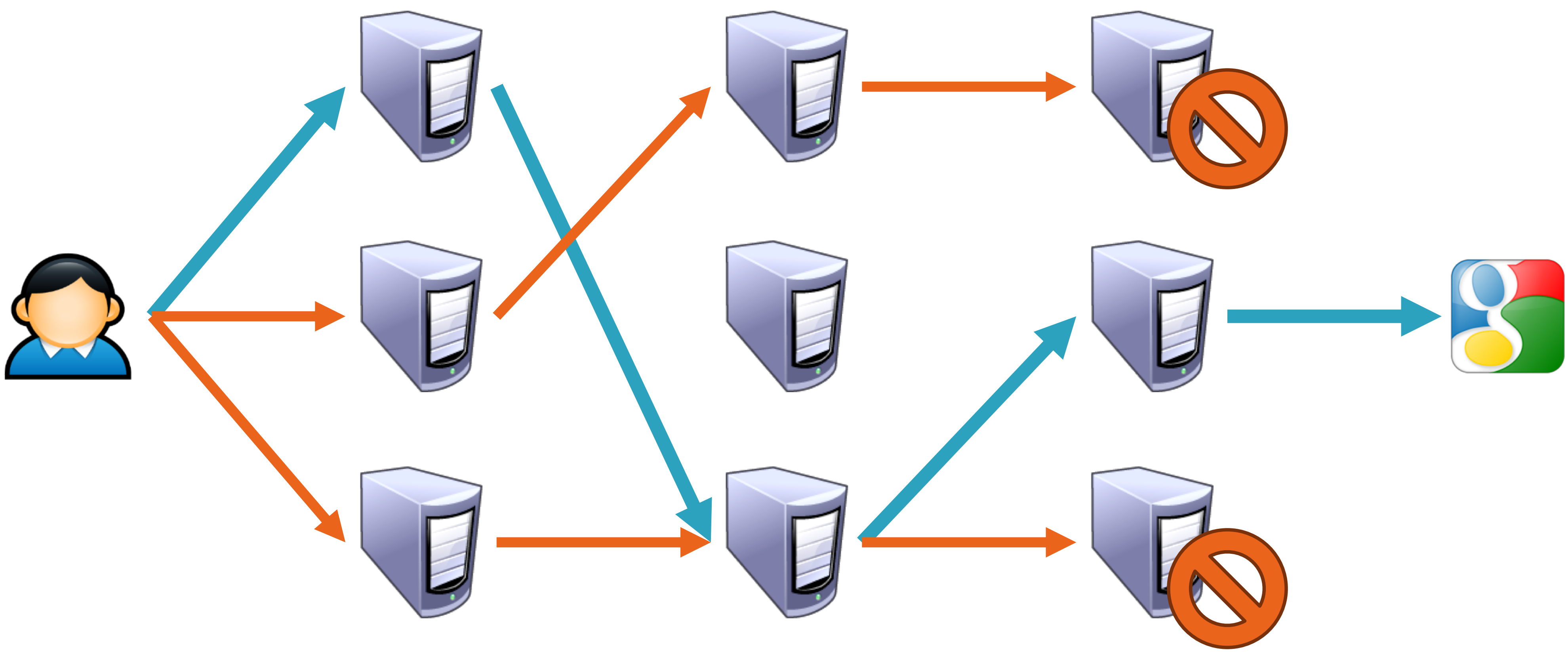
- Hinders timing attacks
  - Messages may be artificially delayed
  - Temporal correlation is warped
- Problems:
  - Requires lots of traffic
  - Adds latency to network flows

- 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



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

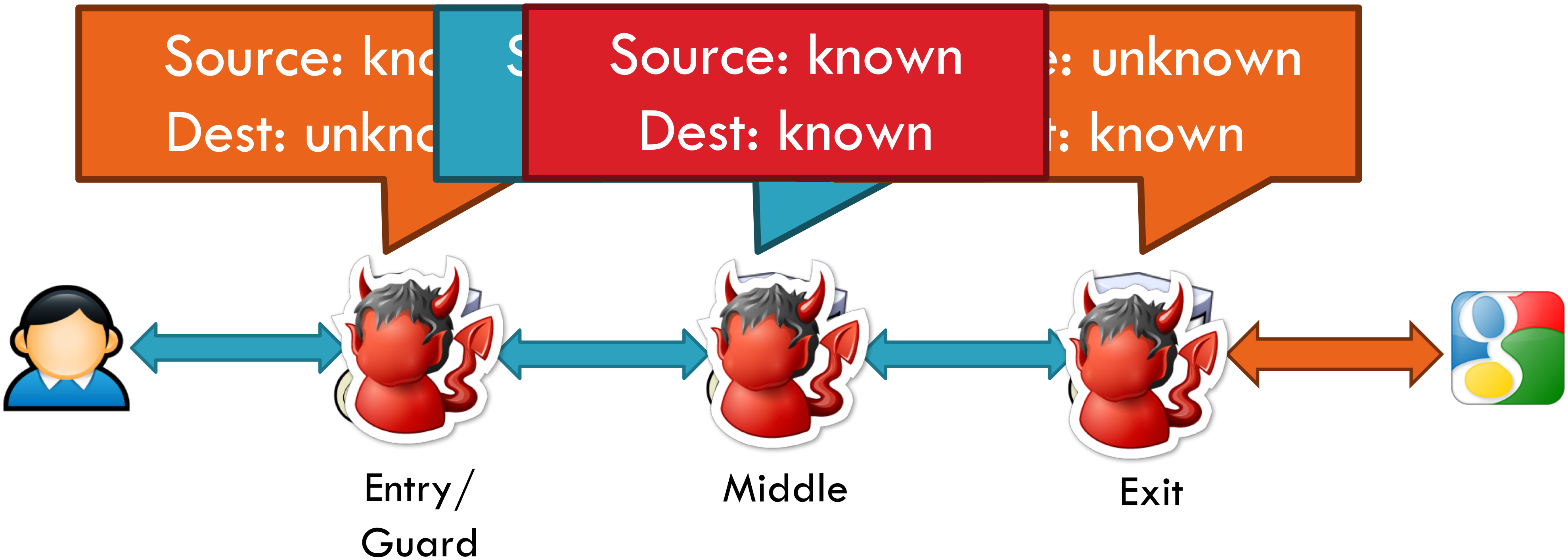
How Do You Use Tor?

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

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 roughly proportional to bandwidth
  - Why? Is this a good idea?

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
  - M
- Att
  - N
  - (

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

  - Roughly $(M/N)^2$ chance overall, for a single circuit
- However, client periodically builds new circuits
  - Over time, the chances for the attacker to be in the correct positions improves!

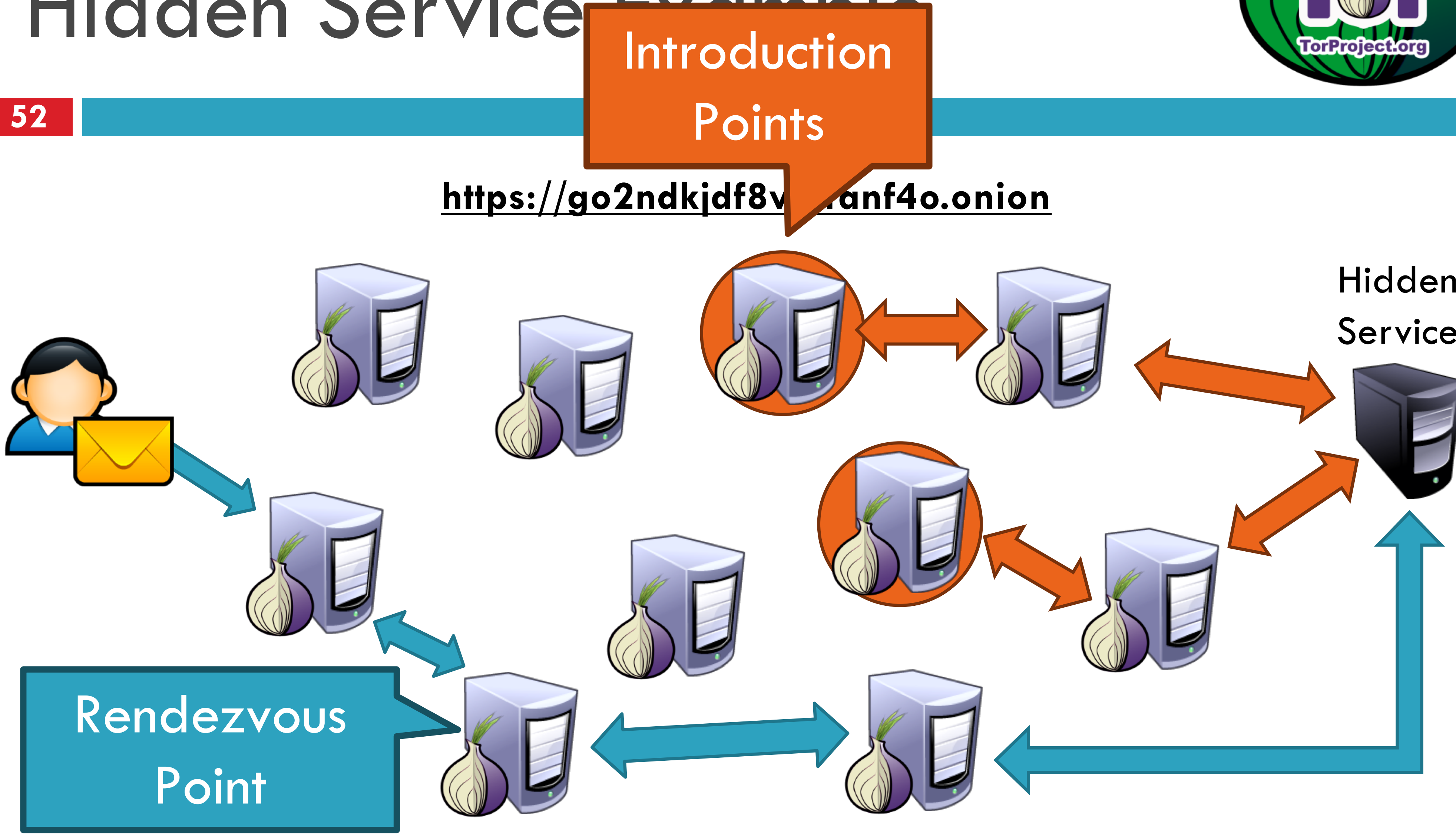
Guard Relays

- 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 relays that:
  - Have long and consistent uptimes…
  - Have high bandwidth…
  - And are manually vetted may become guards
- Problem: what happens if you choose an evil guard?
  - M/N chance of full compromise

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)

Hidden Service Example



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

Perfect Forward Secrecy

- In t ... g
  pub
- Pro
  - A
  - I

- 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 implements Perfect Forward Secrecy (PFC)
  - 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

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

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