Anonymous-Communication Systems

Terminology, Mix-Nets, Circuit-Based Systems, Tor, Censorship Resilience

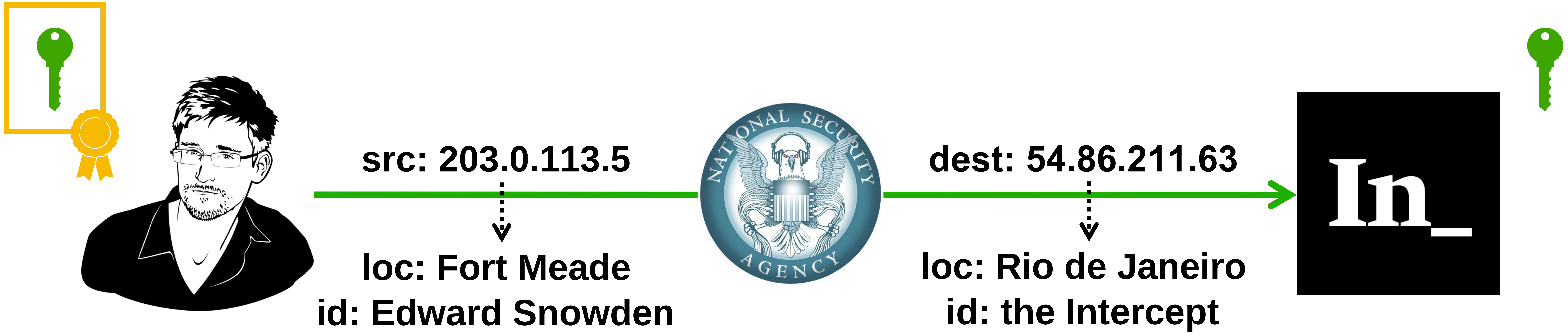
Network Security AS 2020

20 October 2020

Markus Legner
(based on slides by Daniele Asoni)

ETH zürich

Why could anonymity be desirable?



- IP addresses still leak *metadata* information:
  - Who talks to whom, at what time, for how long, how frequently…
  - NSA can log connection metadata, and later incriminate Snowden
- We may want to hide from the destination itself (to avoid retaliation)

Why could anonymity be desirable?

- It is not just for (alleged) criminals who want to act with impunity!
- ***Military applications***
  - Covert intelligence gathering
  - Covert attacks
  - Penetration testing on own infrastructure
  - Undercover agents communicating out of a monitored country
- ***Trade, industrial R&D***
  - Detect price discrimination
  - Hide revealing patent searches in an untrusted database

US government increases funding for Tor, giving $1.8m in 2013

Despite attempts by the National Security Agency to crack the anonymous browser, the US increased state funding through third parties

https://www.theguardian.com/technology/2014/jul/29/us-government-funding-tor-18m-onion-router

Half of the Tor Project's funding now comes from the private sector

Tor Project reports $4.2 million income in 2017, of which only 51 percent came from government funds.

By Catalin Cimpanu for Zero Day | December 10, 2018 -- 16:47 GMT (16:47 GMT) | Topic: Security

Why could anonymity be desirable?

- ***Anonymous reporting***
  - Tips regarding criminal activity, but also accidents
- ***Human rights***
  - Free speech, whistleblowing, censorship avoidance
- Building block for other technologies:
  - Crypto-currencies (Bitcoin, Ethereum)
  - ***Electronic Voting***

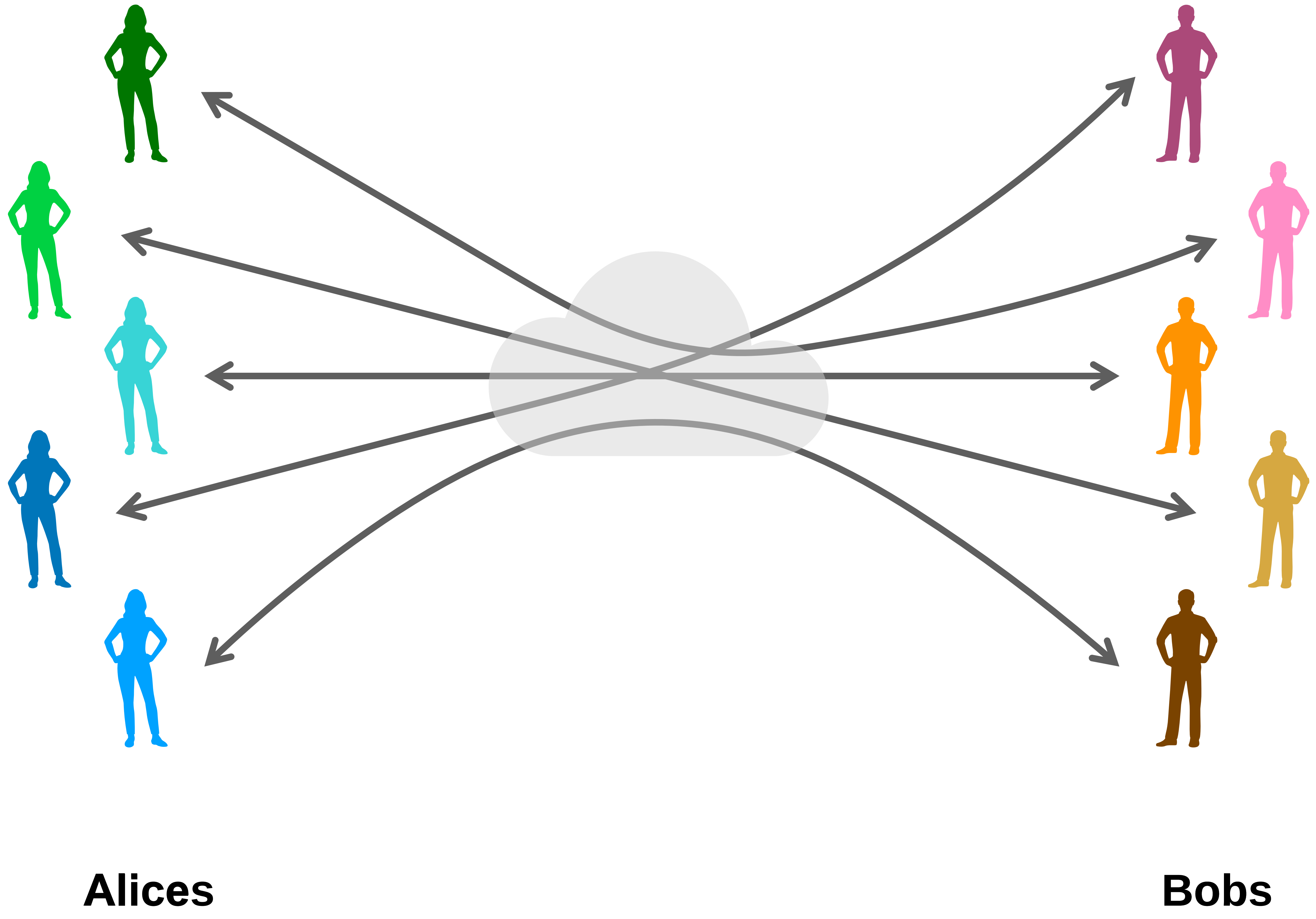
What is anonymity?

(Sender/receiver) anonymity, unlinkability, unobservability…

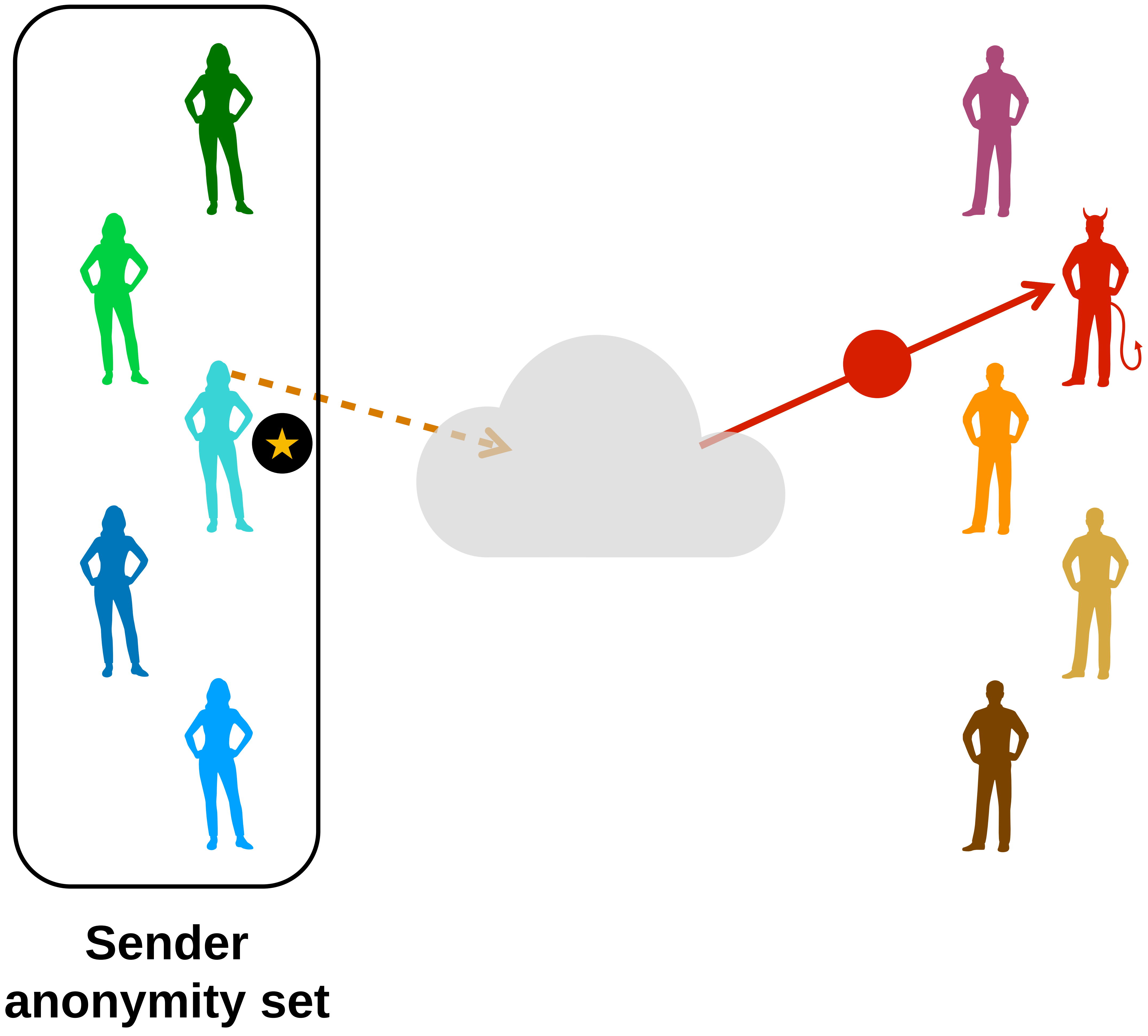
Definitions…

- Defining anonymity (and related concepts) is tricky
  - In the literature there are various definitions and approaches
  - Anonymity is *not* a property of individual messages or flows
    → **You cannot be anonymous on your own!**
- We adopt a high-level, intuitive (= not formal) set of definitions
  - Based on: A. Pfitzmann, M. Hansen. *A terminology for talking about privacy by data minimization* (https://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf)

Setting

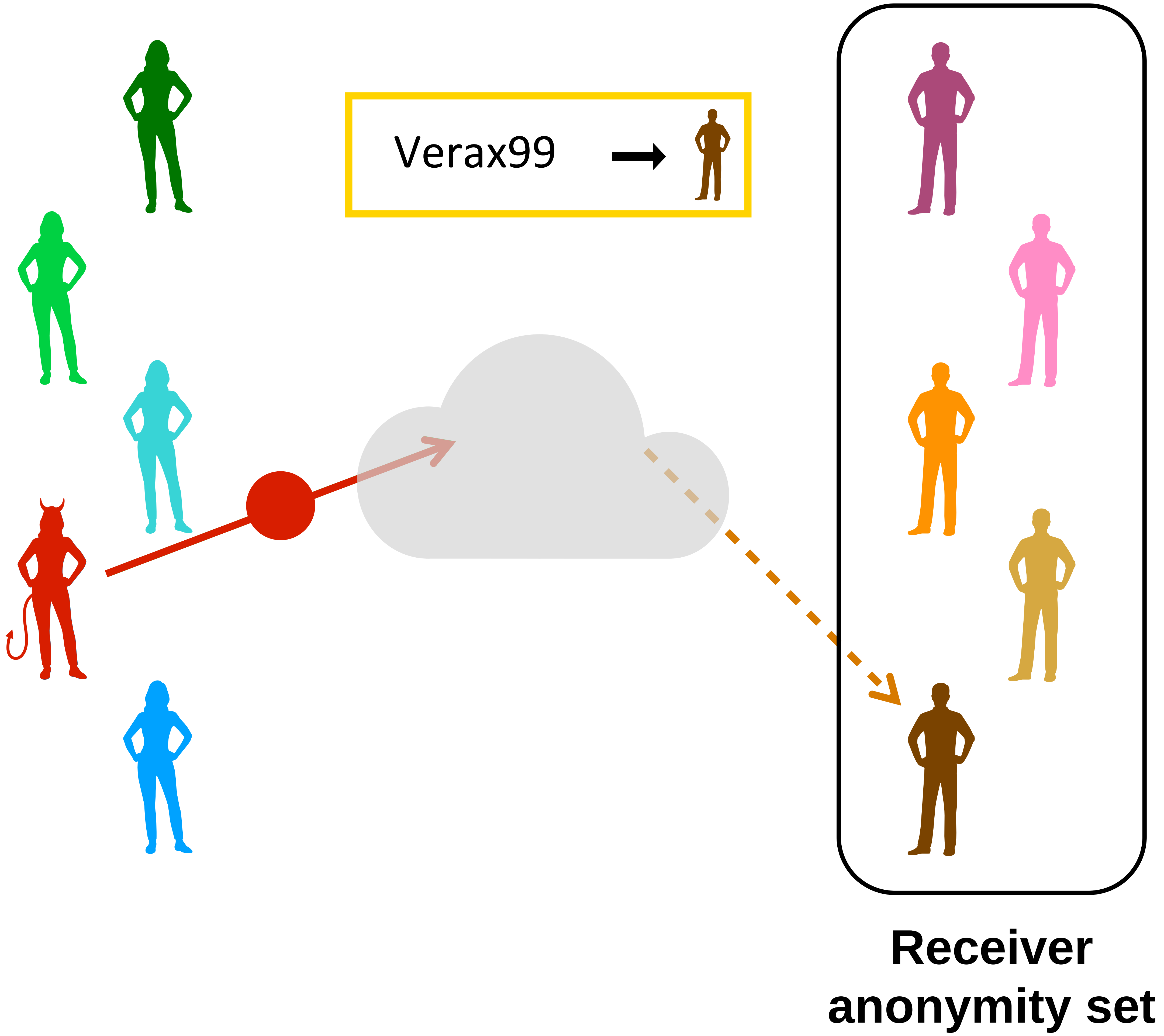


Terminology: “sender anonymity”



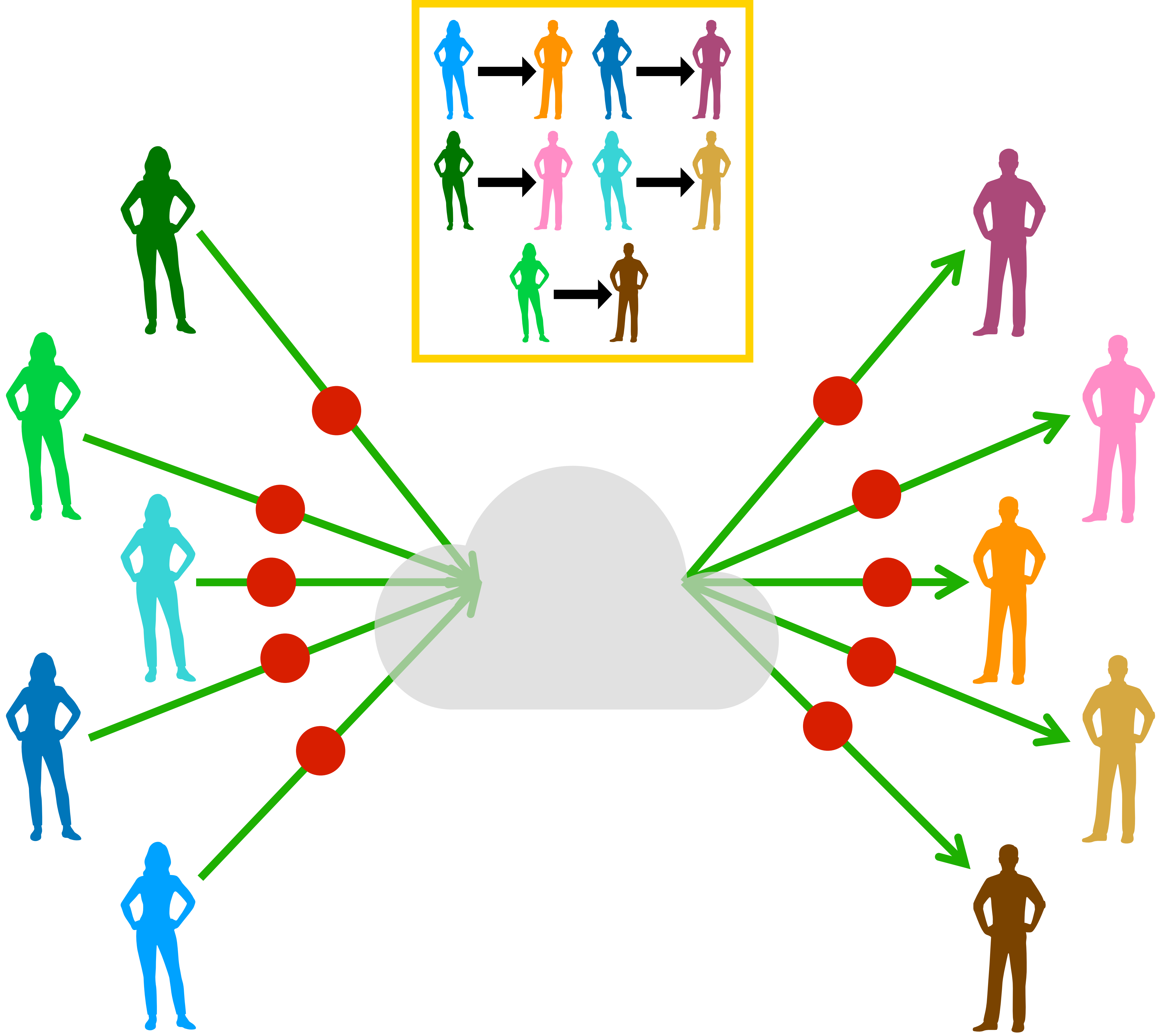
- *Sender anonymity setting*
  - Adversary knows/is receiver
  - Adversary may learn message
  - *Sender is unknown*
- *Sender anonymity set*
  - Set of all senders/individuals *indistinguishable from real sender*
  - Can be used as a (rough) metric
  - Small set ➞ little anonymity
- *Return address*
  - Token provided by original sender

Terminology: “receiver anonymity”



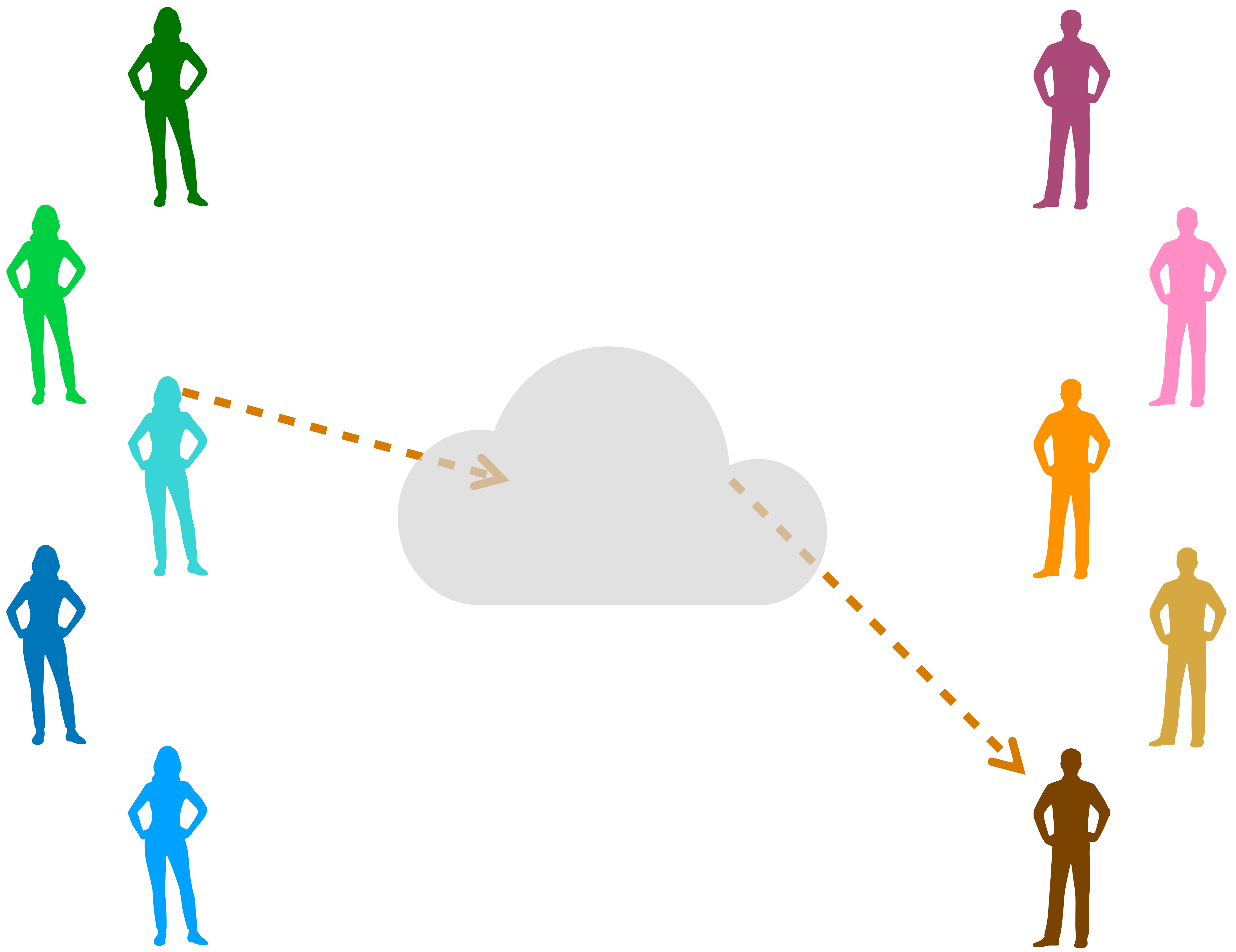
- *Receiver anonymity setting*
  - Adversary knows/is sender
  - Adversary may choose message
  - Receiver is unknown
- *How does destination receive traffic?*
  - Hidden service (pseudonym known)
- *Receiver anonymity set*
  - Set of all receivers/individuals *indistinguishable from real receiver*

Terminology: “unlinkability”



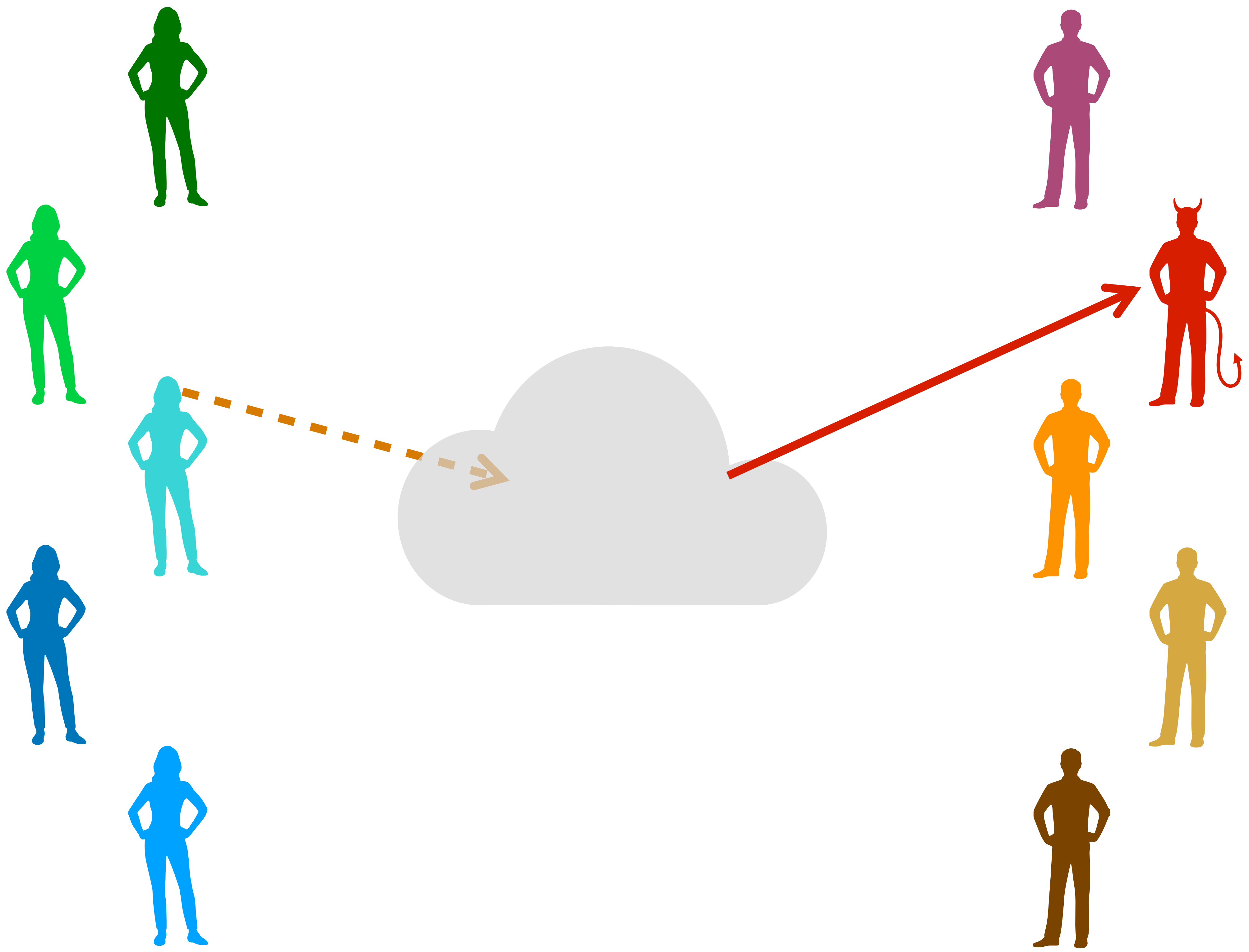
- *(Sender-receiver) unlinkability*
  - Adversary knows senders
  - Adversary knows receivers
  - Link between senders and receivers is unknown
- Multiple users need to communicate at the same time
- Anonymity ➞ Unlinkability

Terminology: “unobservability”



- *Unobservability*
  - Adversary cannot tell whether any communication is taking place
- How can this be achieved in practice?
  - Wireless communications: DSSS
  - Wired: *always send traffic!*
- Unobservability → Anonymity

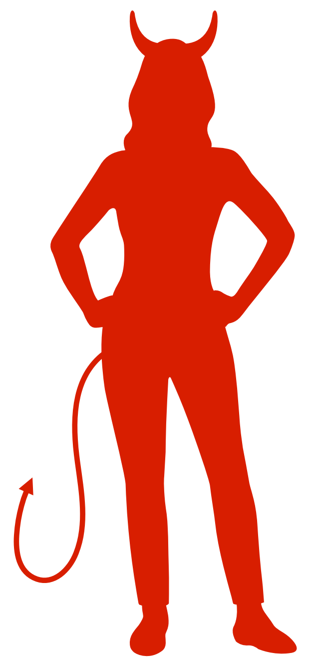
Terminology: “plausible deniability”

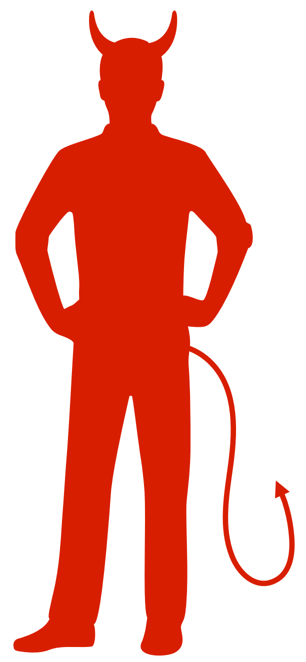


- Adversary cannot prove that any particular individual was responsible for a message (or other action)
- Anonymity ➞ Plausible deniability

Threat model(s)

- There are various types of adversary that can be considered
- Degree of control: *local* or *global*
- Type of control: *network* or *compromised infrastructure*
  - Various combinations are possible
  - The infrastructure is never fully compromised
- Type of behavior: *passive* or *active*
- Often not clearly specified ➞ unclear guarantees





The Harvard bomb threat: why the anonymity set matters…

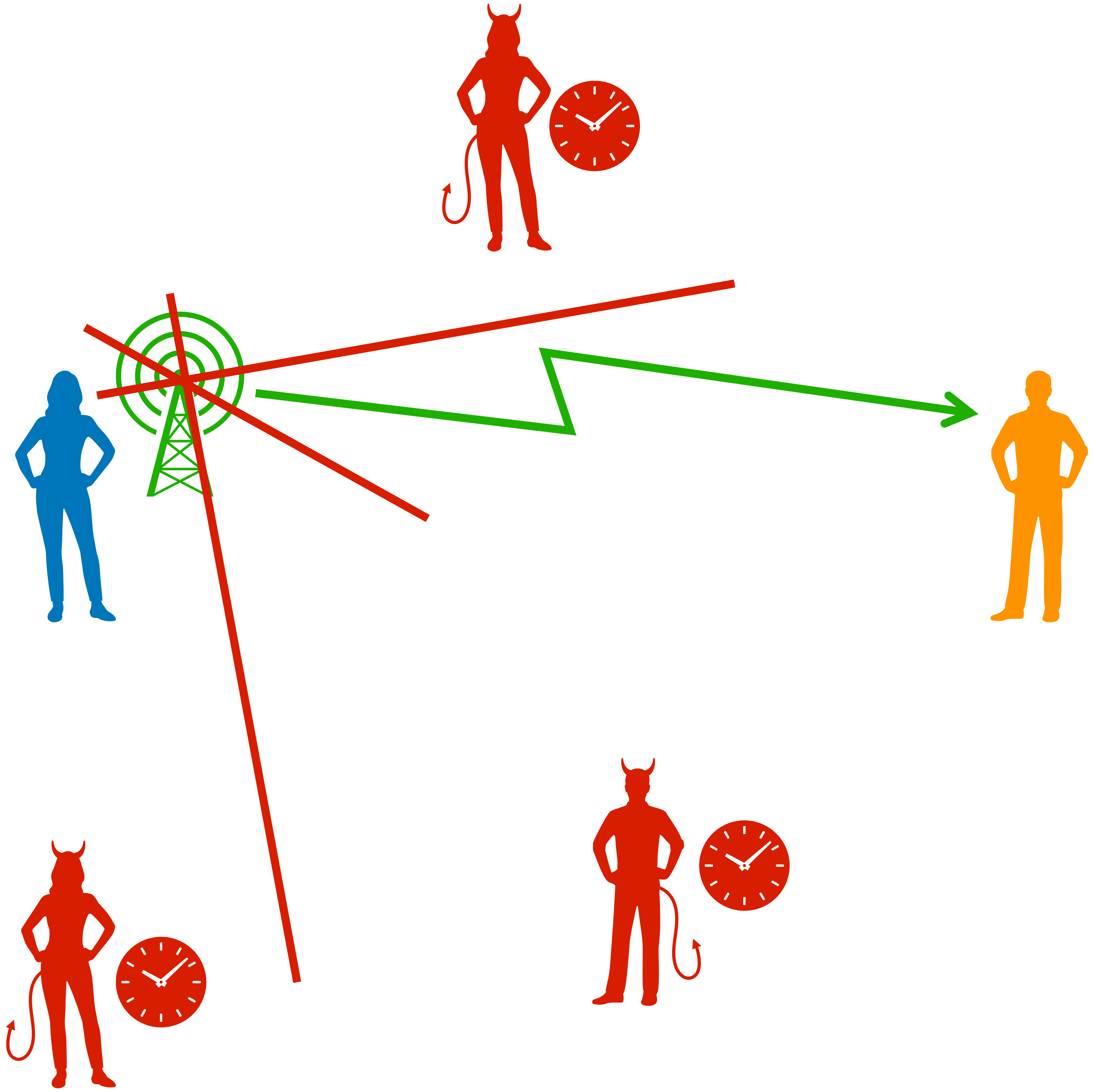
- During an exam session in 2013, a bomb threat was sent at Harvard university
- The sender was a student, who sent the email from an anonymous mail server, which he accessed via Tor
- The sender was arrested and confessed the same day
- Questions:
  - What went wrong?
  - How could he have increased the anonymity?

Basics of anonymous communication

Mix-nets and circuit-based (onion-routing) systems

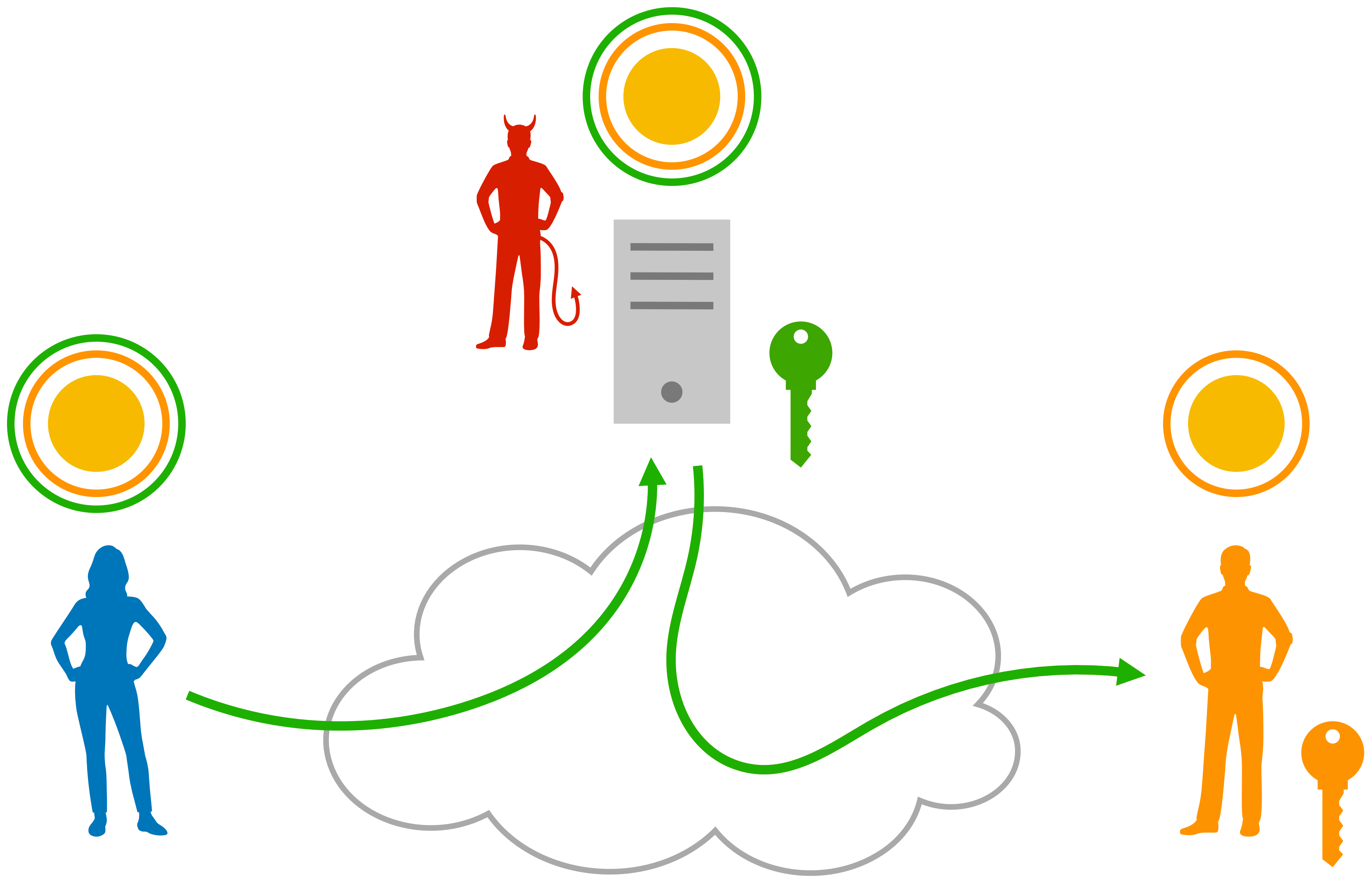
What mechanism can we use?

- Wireless communication: *broadcast*
  - Receiver anonymity is guaranteed
  - Sender can de-anonymized!
    - Localization through triangulation
    - Sender can move
    - Use DSSS if destination is trusted
- Alternative: hijacked connection
  - burner phone, hacked WiFi
  - network ID ≠ personal sender ID



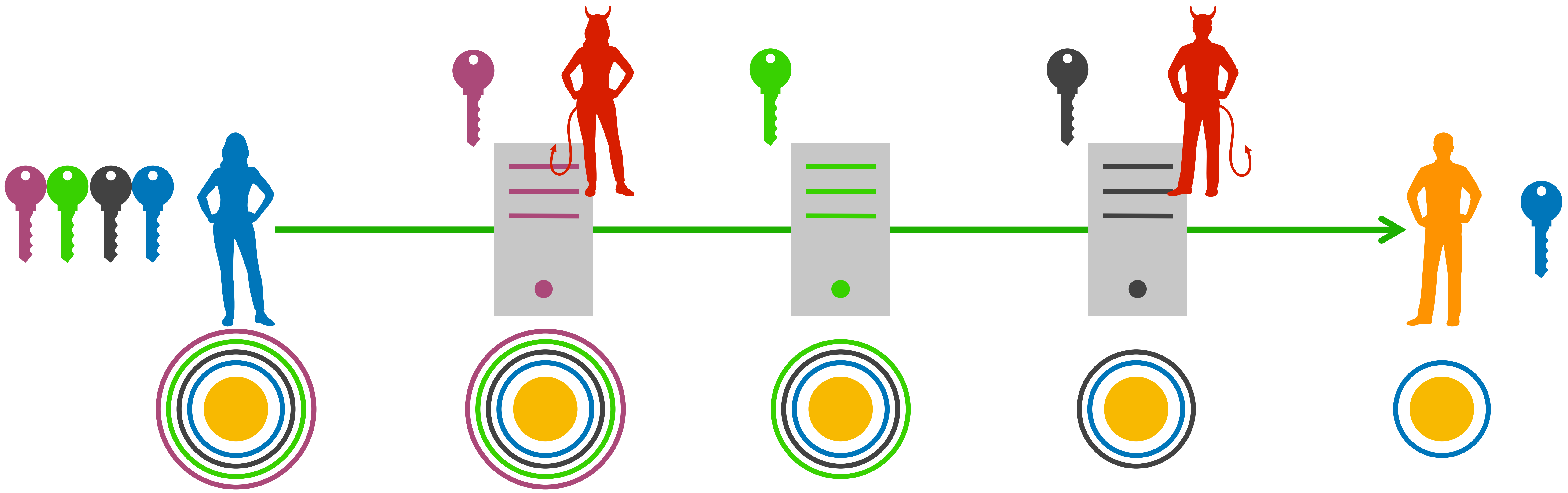
What mechanism can we use?

- Simple idea: use a *proxy* or *VPN*
- Use *layered encryption* to hide content from proxy
- **Problem**: proxy can see (and record) metadata
  - Addresses of communication partners
  - Amount of data
  - …
- How to avoid single point of failure?



Notation: ● := { Addr, {KeyID}, Enc_K(●) }

Solution: cascade of multiple proxies



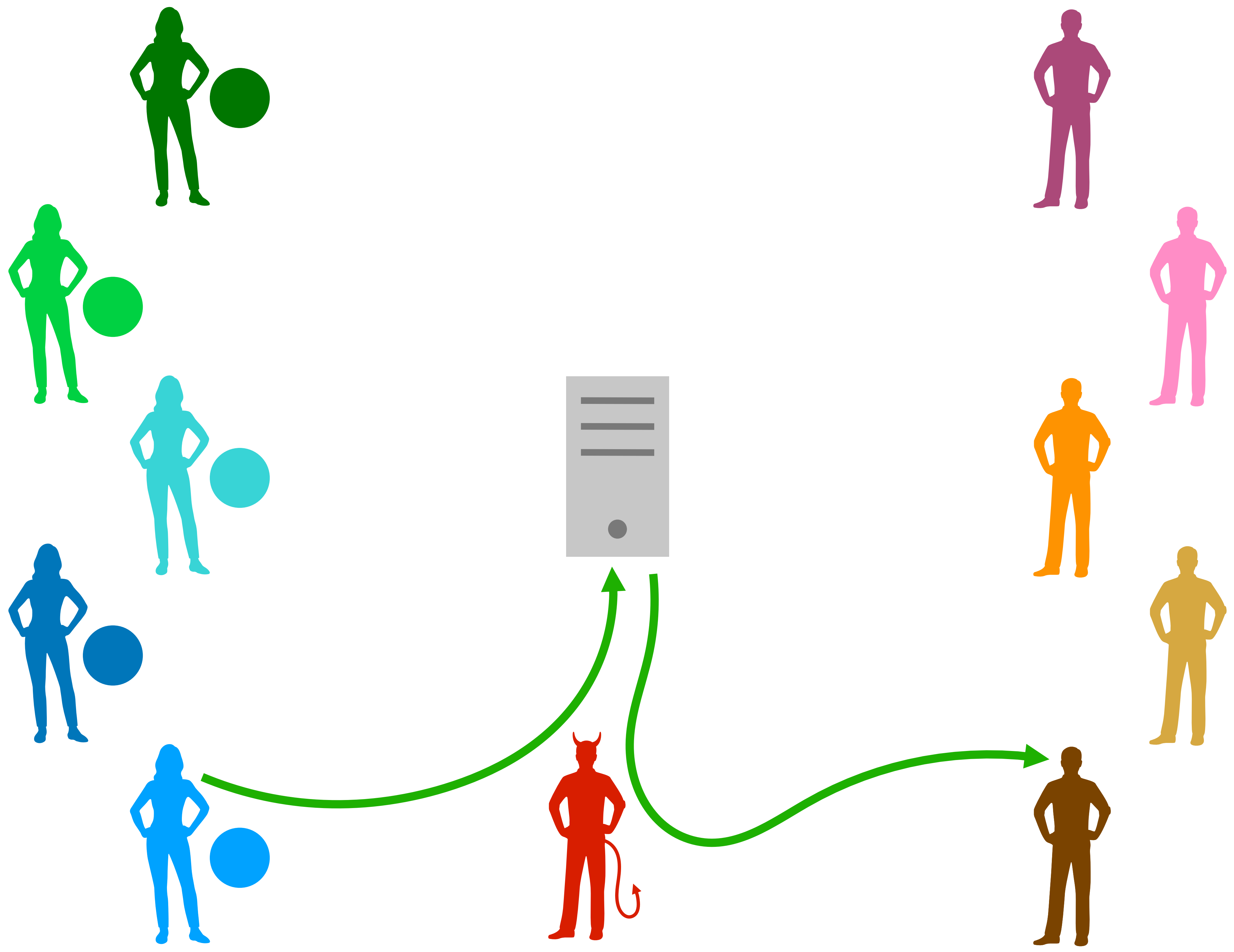
- Use multiple proxies to avoid single point of failure (*cascade*)
  - Each proxy only sees addresses of two neighbors
  - Should work as long as the message traverses at least one honest proxy
- Message and forwarding information is encrypted multiple times (onion)

Notation: **:= { Addr, {KeyID}, Enc$_K$( ) }**

Mix-nets

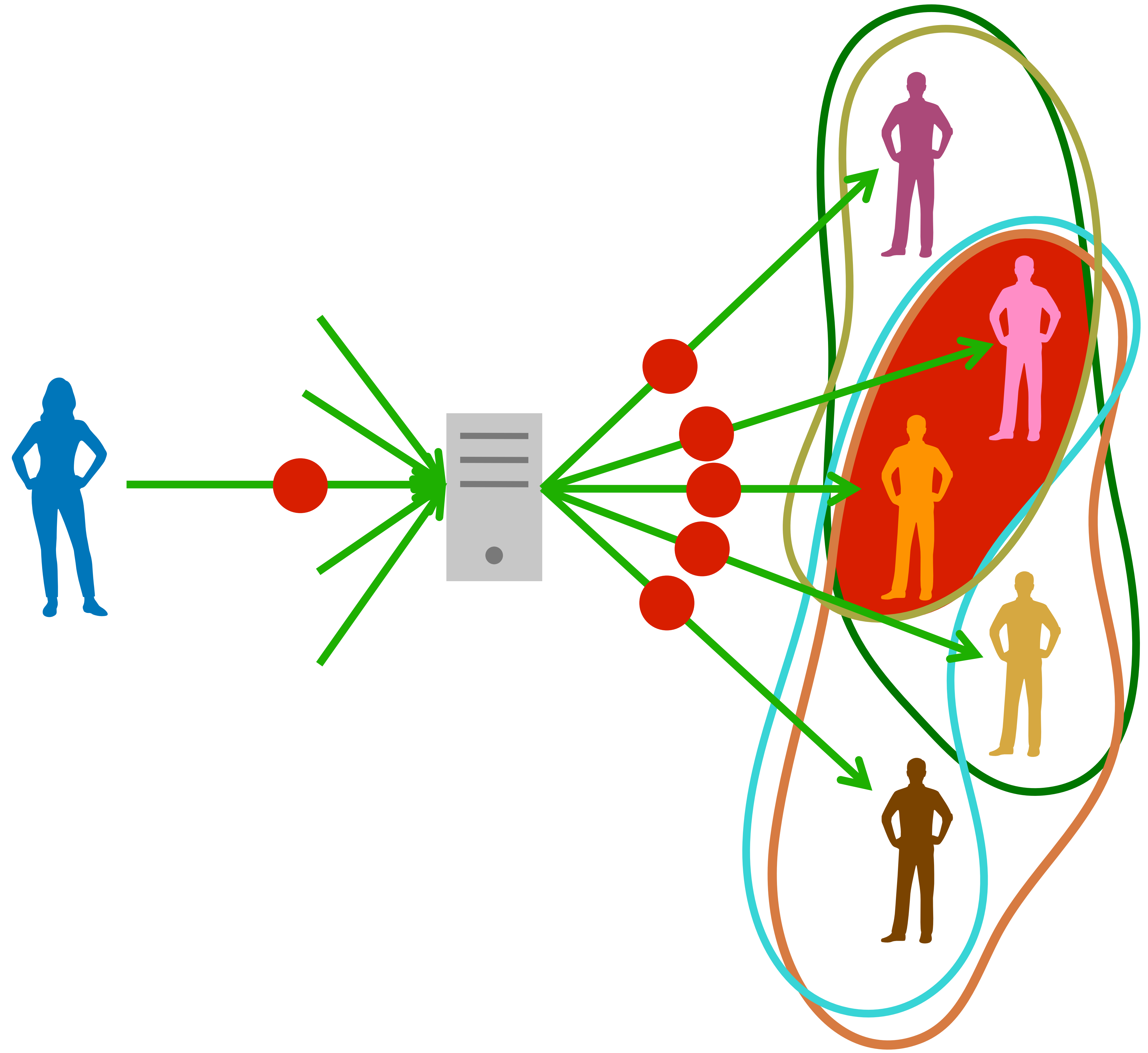
Batching and mixing

- Problem: network attacker can observe in- and outgoing messages
- Each proxy should perform *batching*
  - Collect several messages before forwarding (threshold)
- Additionally, the proxies should change the order of (*mix*) the messages
- This is called a *threshold mix*
- Important: messages need to be padded to a **fixed length** to make them indistinguishable!
- Are we fully anonymous now?



Intersection attack

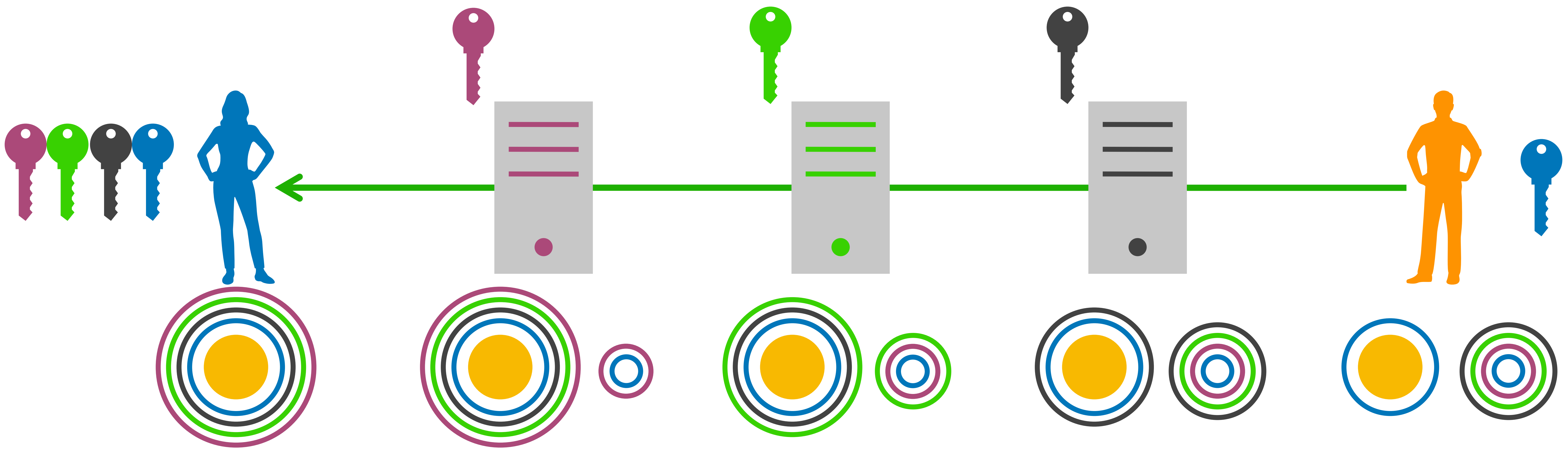
- Often, users only communicate with a small subset of other users
- Idea: every time a message is seen by the target, register the sets of destinations
- This is called *intersection attack*
  - Kesdogan et al., *Limits of anonymity in open environments, IH,* 2002
- More effective: *statistical disclosure*
  - Danezis and Serjantov, *Statistical disclosure or intersection attacks on anonymity systems, IH,* 2005



Cover traffic for unobservability

- To achieve full unobservability, use *cover traffic*
  - Also called dummy, chaff, or padding traffic
  - Prevents statistical disclosure
- Both for sending and for receiving
  - Often, the mix stores messages for receivers
  - Receivers regularly try to retrieve messages
  - If there is a message, it is downloaded, otherwise a dummy message is returned by the mix
- Now we are fully anonymous… as long as one mix is honest!

How to send replies in mix-nets?



- Sending replies back is non-trivial
- The sender knows what keys will be established with each mix
- The original sender can prepare a *return address*
- Danezis and Goldberg. *Sphinx: A compact and provably secure mix format*, S&P 2009

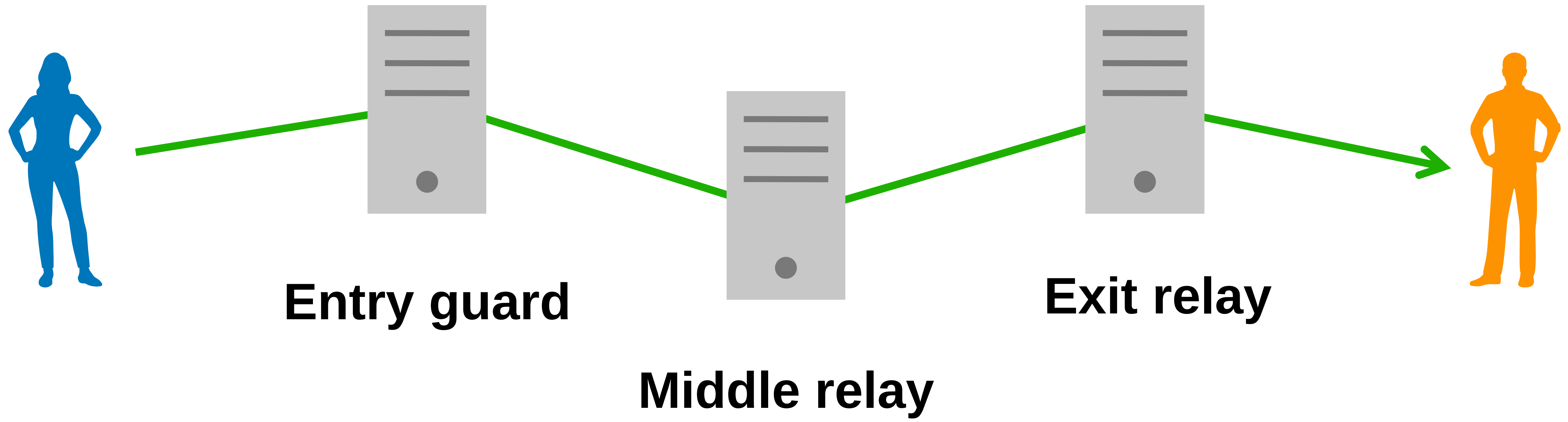
Circuit-based anonymity networks (onion routing)

Can we speed up mix-nets?

- Problem of mix-nets: high latency due to batching and mixing
- We would like to have a system that can *support web browsing*
- Short answer: “Yes, but only lowering the anonymity guarantees”
  - Long answer is more complicated: P. Syverson. *Sleeping dogs lie on a bed of onions but wake when mixed* (https://petsymposium.org/2011/papers/hotpets11-final10Syverson.pdf)
- Main ideas:
  - *Layered encryption*, *no batching and mixing*, *no cover traffic*
  - Flow-based: establish a *virtual circuit* (keys) once per flow, reuse it for all packets in the flow using only *symmetric key crypto*
- Constrained threat model
  - Only *local adversary*, which cannot launch confirmation attacks

Terminology

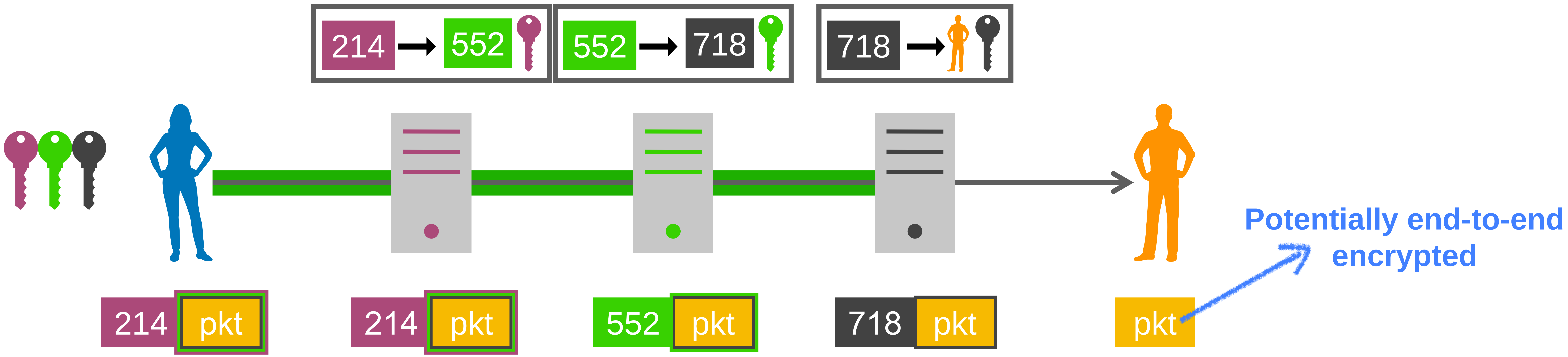
- ***Circuit-based anonymous communication systems***
  - Commonly known as ***Onion Routing Systems***
  - Confusing terminology (layered encryption is “onion routing”)
- The nodes are called *relays* (also *nodes* or *routers*)
- The virtual circuit is also called tunnel (especially if it is at layer 3)



Life-cycle of a circuit

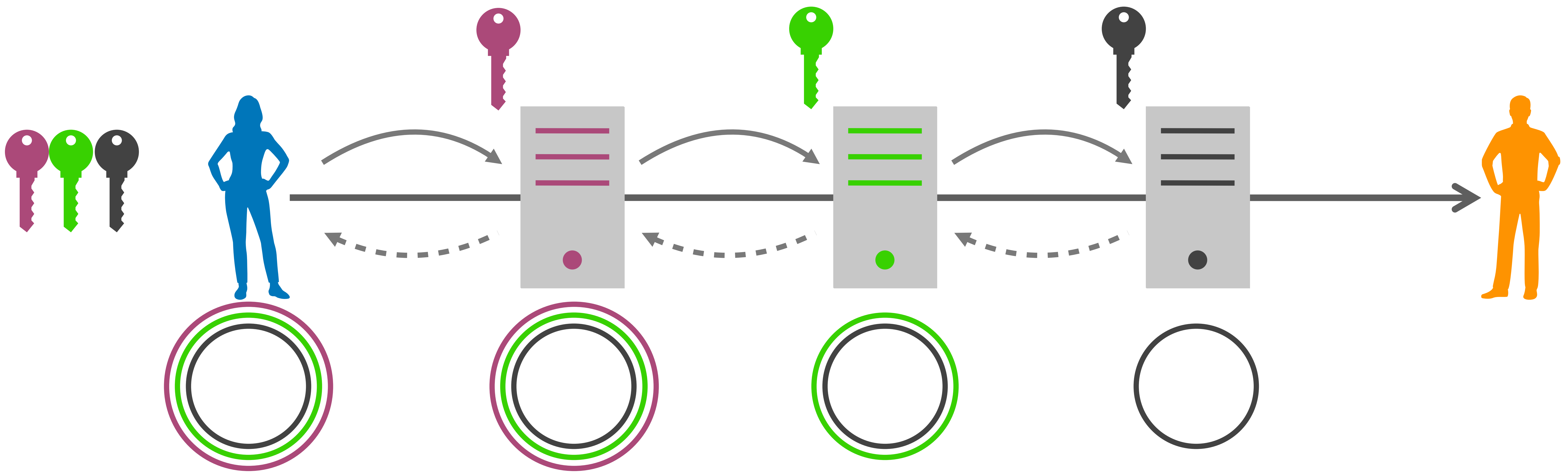
- ***Circuit setup***
  - Initially, sender knows long-term public keys of relays
  - The sender negotiates shared keys with all relays on the path (this requires expensive *asymmetric key cryptography*)
  - The relays store the necessary state
- ***Data forwarding***
  - Packets for one or more flows are forwarded along the circuit
  - Only *symmetric key cryptography* is used (AES)
- ***Circuit tear-down***
  - The circuit is destroyed to free state on relays or to prevent attacks

Data forwarding



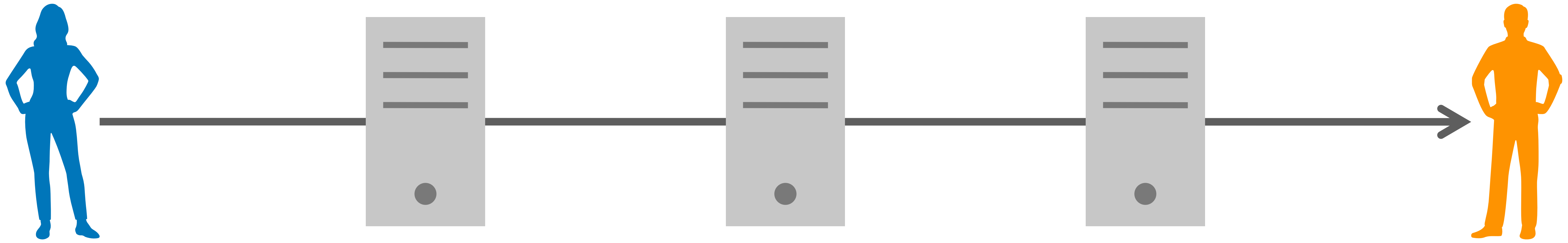
- The sender has established a circuit (keys and *per-link IDs*)
- A data packet is encrypted as usual (layered encryption)
- The ID of the next relay is added in clear text
  - To protect against network adversaries, links can be encrypted (TLS)

Direct circuit setup



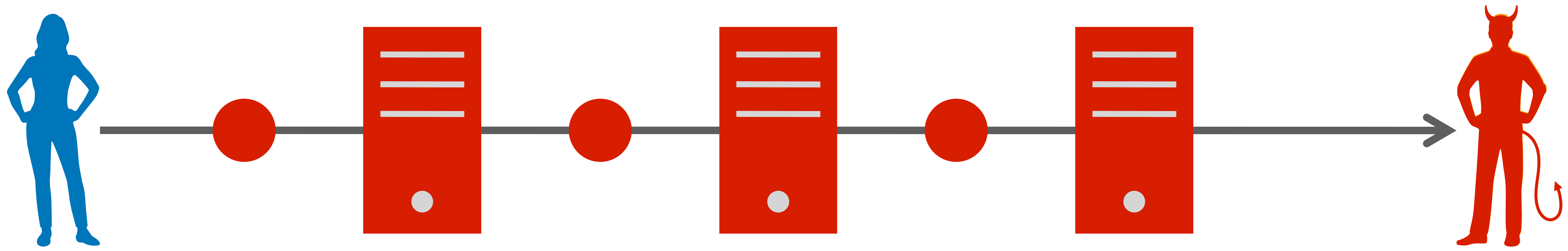
- Establish state on relays by using a normal packet as for mixes
  - Message for each node contains address of next node and ephemeral Diffie–Hellman share
  - Each node replies with its own ephemeral Diffie–Hellman share
- Relatively fast (though relays need to perform asymmetric crypto)

Forward security



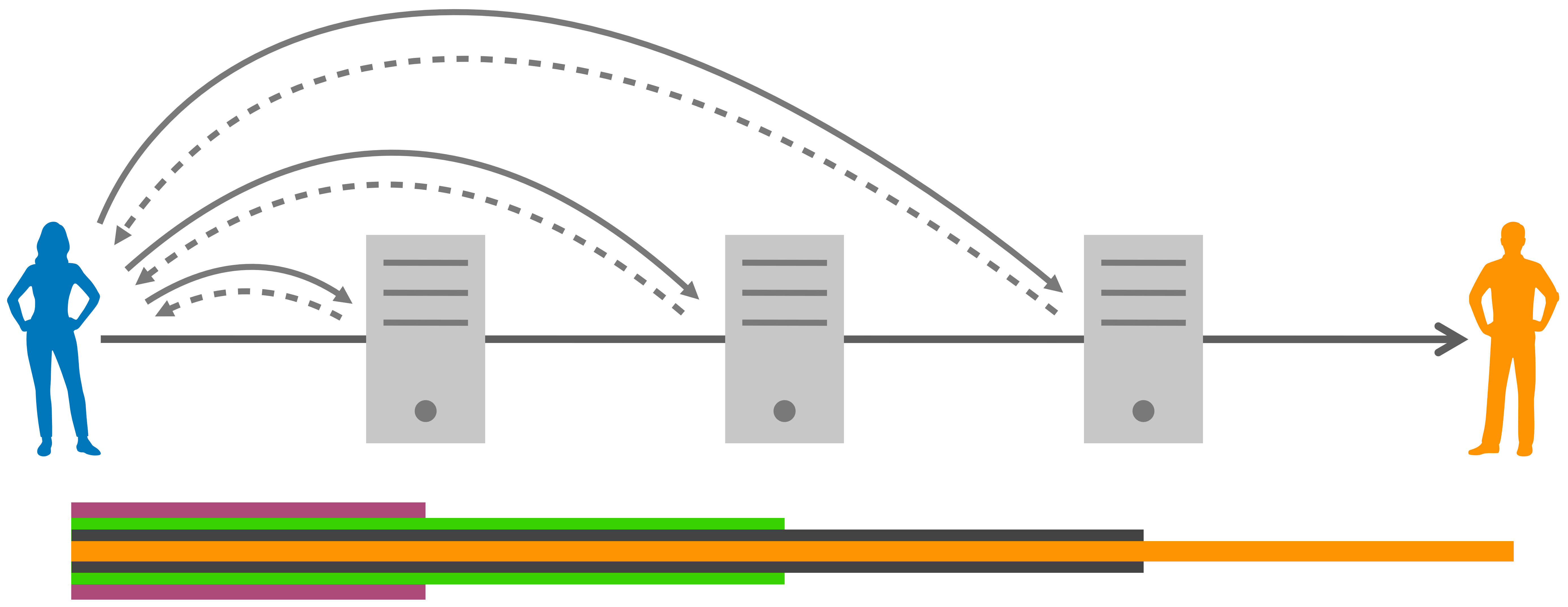
- Forward security: if long-term keys are compromised, anonymity of previously established circuits is preserved
- The direct setup does *not* provide (immediate) *forward security* for link between communication partners
  - No ephemeral information can be used to encrypt setup message
    - Need to use long-term public key of each node for encryption
    - Similar to 0-RTT data in TLS
- Forward security for later packets can be achieved through Diffie–Hellman exchanges

Forward security: deanonymization



- Assumption: the adversary can record all network traffic
- Once a relay is compromised, the adversary can replay all setup packets, and see which corresponds to the target circuit
- To prevent this (with direct setup): change public keys of the relays
  - This is called *eventual forward security* – significant overhead!

Telescopic circuit setup



- Keys are negotiated one relay at a time
- The circuit is “extended” by one hop at a time
  - Ephemeral session keys are negotiated before the circuit is extended
  - That’s why it is called telescopic
- This setup is slower… but it offers *immediate forward security*:
  - As soon as the circuit is closed, the session keys are deleted

Circuit tear-down

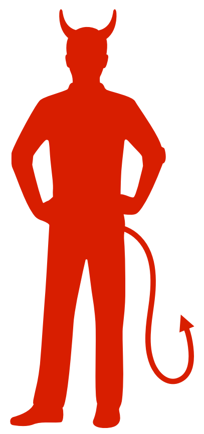
- Can be initiated both by sender and by intermediate relays
  - The sender communicates the tear-down to one relay at a time, starting from the furthest away
  - The exit relay may tear down the circuit if a corrupt packet is detected, or some other attack
- Circuits have a limited lifetime, so they will eventually be destroyed

Comparison of mix-nets and onion routing

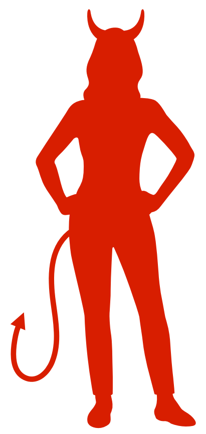
	Mix-net	Onion routing
Forwarding system	Message-based	Circuit-based
Layered encryption	✔ (asymmetric)	✔ (symmetric)
Mixing and batching	✔	✘
Cover traffic	✔ (optional)	✘
Forward security	✘	✔ (telescopic setup)
Latency	High	Low/medium

Attacks on circuit-based anonymous-communication systems

Traffic analysis, higher-layer attacks

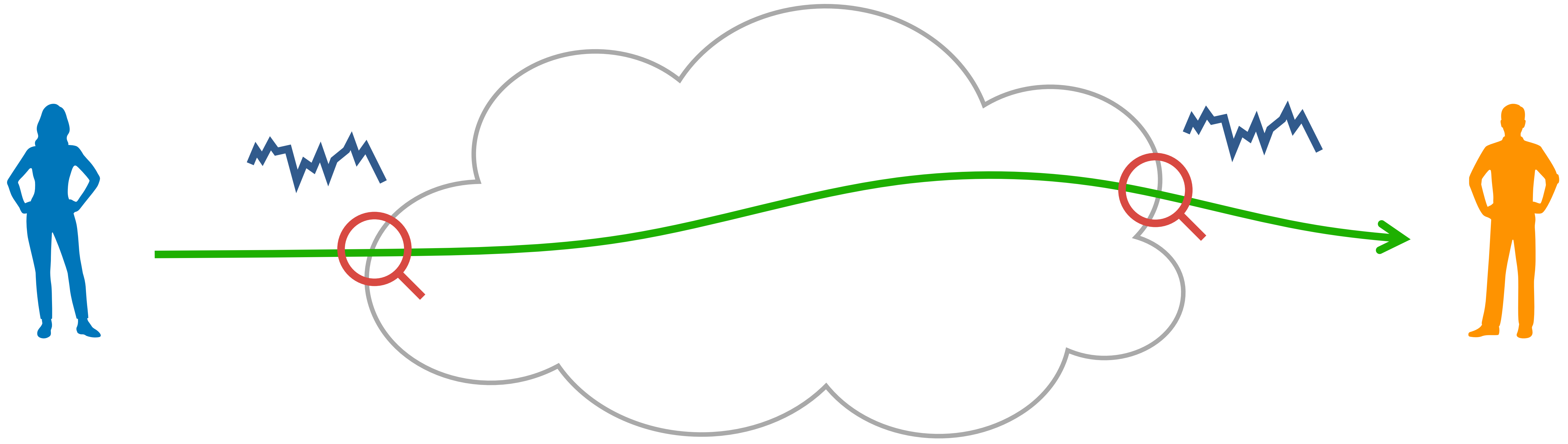


Attacks



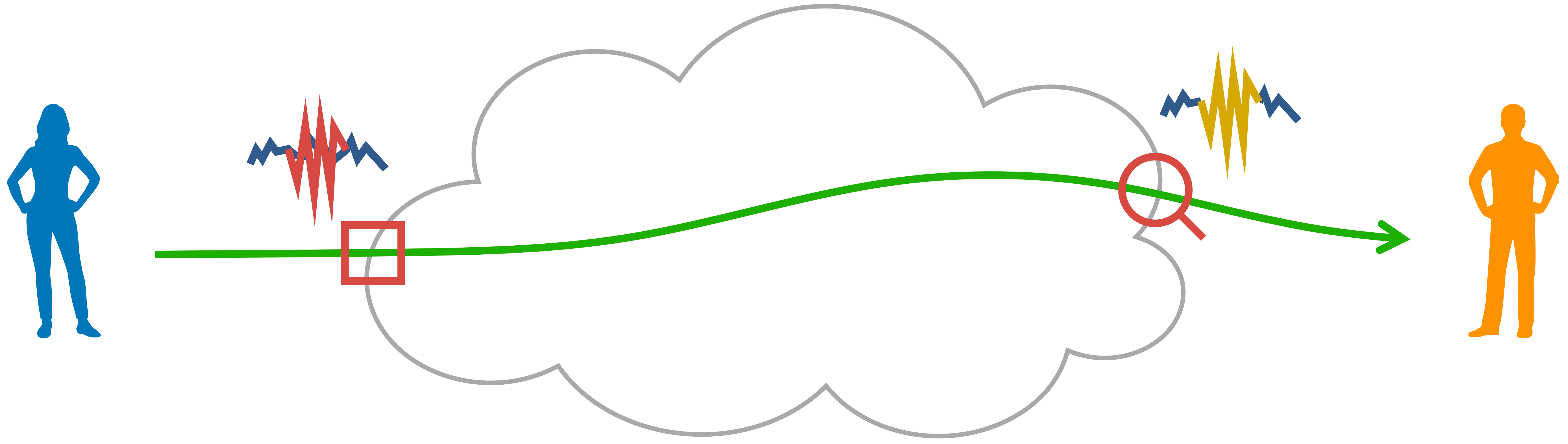
- A number of attacks has been proposed against these systems
- For many it is unclear whether they fit the stated threat models
  - Some of them are practical, requiring limited resources
  - Others are only achievable by state-level adversaries (Five Eyes)
- *Traffic-analysis attacks* (confirmation attacks):
  flow fingerprinting, website fingerprinting
- *Higher-layer attacks*: stack fingerprinting

Passive traffic analysis



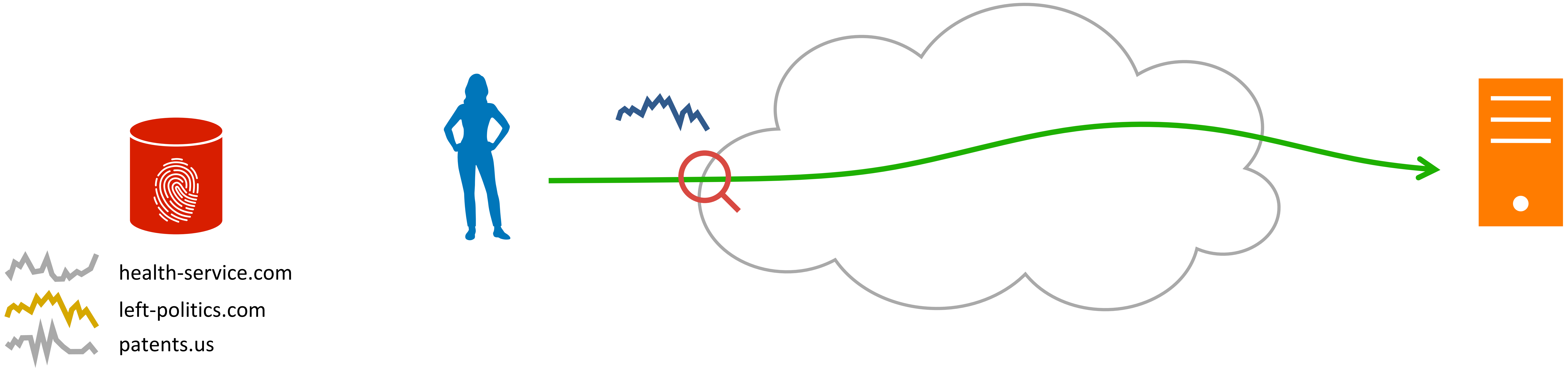
- The adversary observes the edges of the network, recording traffic patterns
  - Flow length, bandwidth pattern, inter-packet timings
- Real-time detection is challenging
  - Alternative is store and compare later ➞ large amount of storage!

Active traffic analysis



- The adversary actively modifies packet timings
  - Inter-packet timings (delaying/reordering packets)
  - Packet drops also possible but detectable
- *Flow watermarking*: inject one bit of information (marked or not)
- *Flow fingerprinting*: inject multiple bits (e.g., sender IP address!)

Website fingerprinting

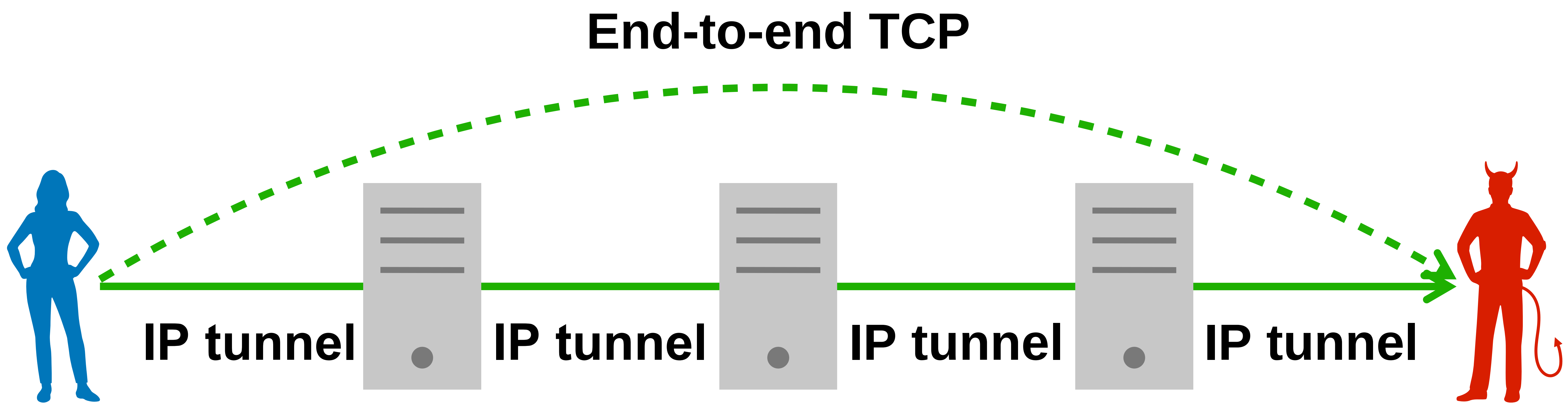


- Adversary needs only one observation point (ISP, other WiFi user…)
- Adversary has built a database of fingerprints of websites
- Particularly effective for interactive applications (health/tax forms)
  - Chen et al., *Side-channel leaks in web applications: a reality today, a challenge tomorrow*, S&P 2010

Traffic analysis resistance

- There have been proposals to incorporate cover traffic and mixing
  - Significant overhead
  - Scalability becomes an issue (large volumes of cover traffic!)
- Only suitable for few applications (VoIP) with low bandwidth
  - Le Blond et al., *Herd: a scalable, traffic analysis resistant anonymity network for VoIP systems*, ACM SIGCOMM, 2015
- To be really secure:
  - Restricted set of flow duration and bandwidth combination

Higher-layer attacks



- OS Network stack fingerprinting
  - Compromised adversary can probe TCP stack
  - Solution: per-hop TCP
  - Still, TLS or HTTP layer may be identifiable!
    - Have a look at https://amiunique.org/fp

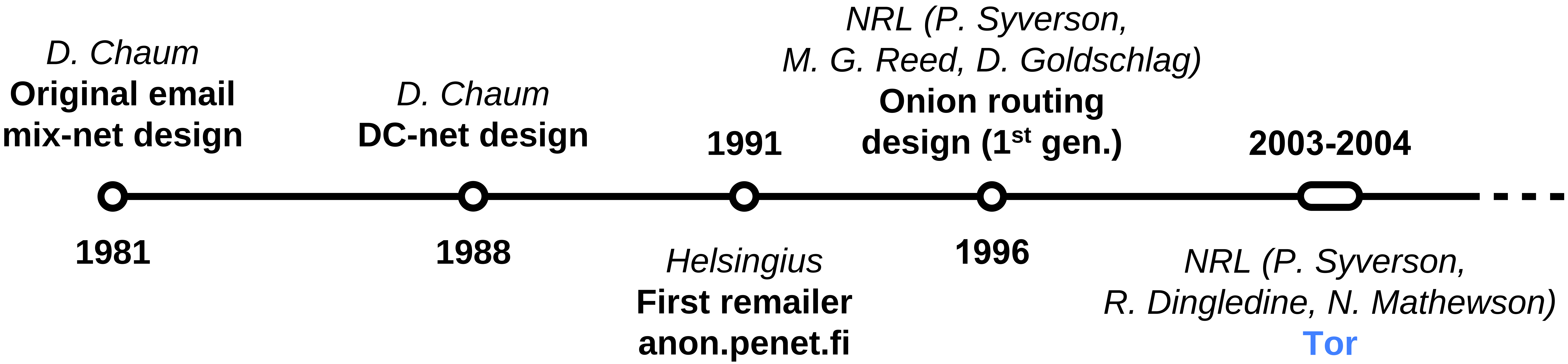
Higher-layer attacks

- Most de-anonymization is still done through other means:
  - Trick user into downloading malware
  - Trick user into downloading file that will access the Internet directly
  - Analyze user behavior like texts
- To achieve anonymity, all layers need to be anonymized:
  - Any gap will break anonymity
  - (This is unlike other security properties)

Tor: the second generation onion router

Cells, circuits and streams, hidden services, directory authorities, bridges

Historical overview

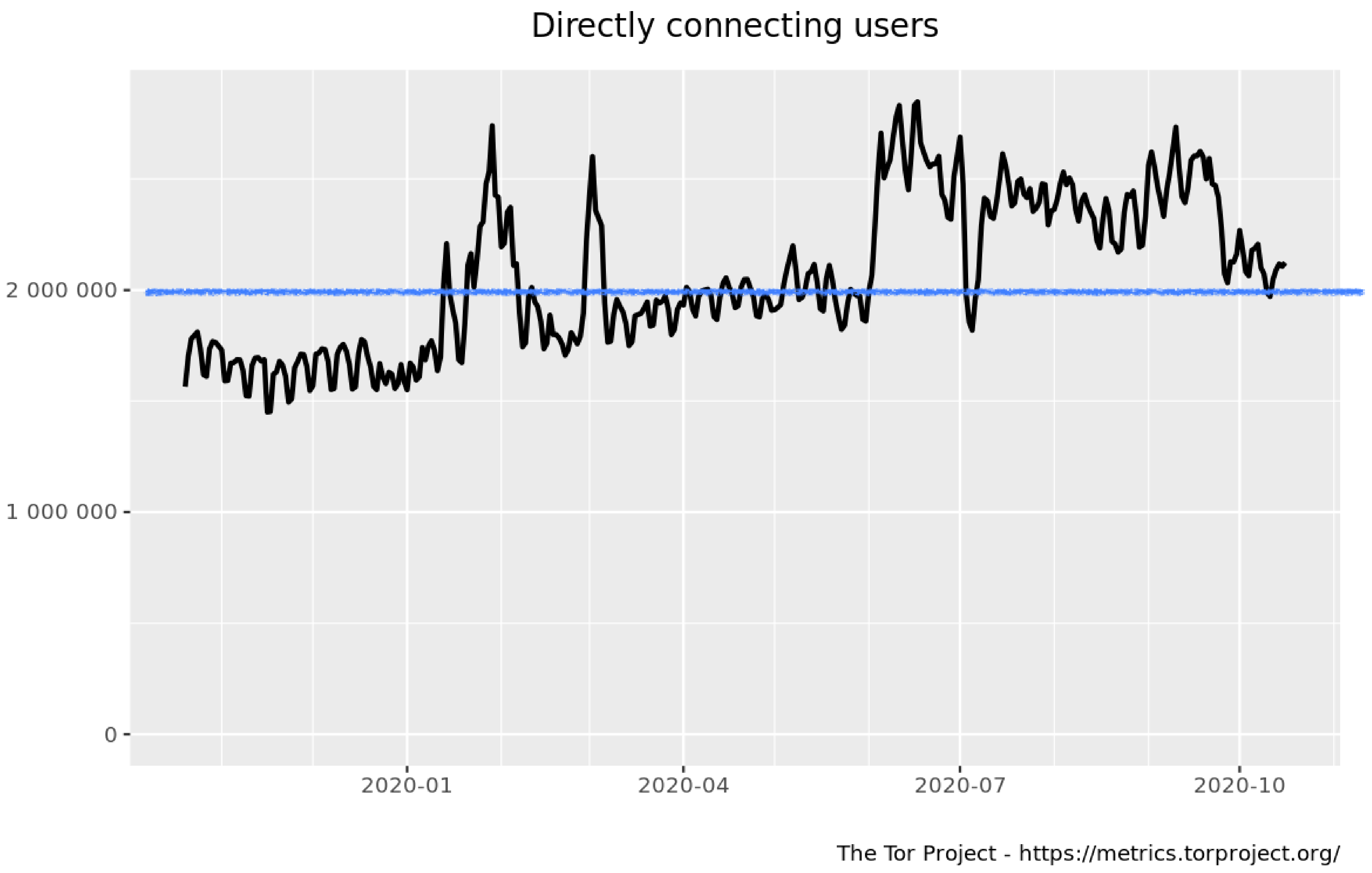


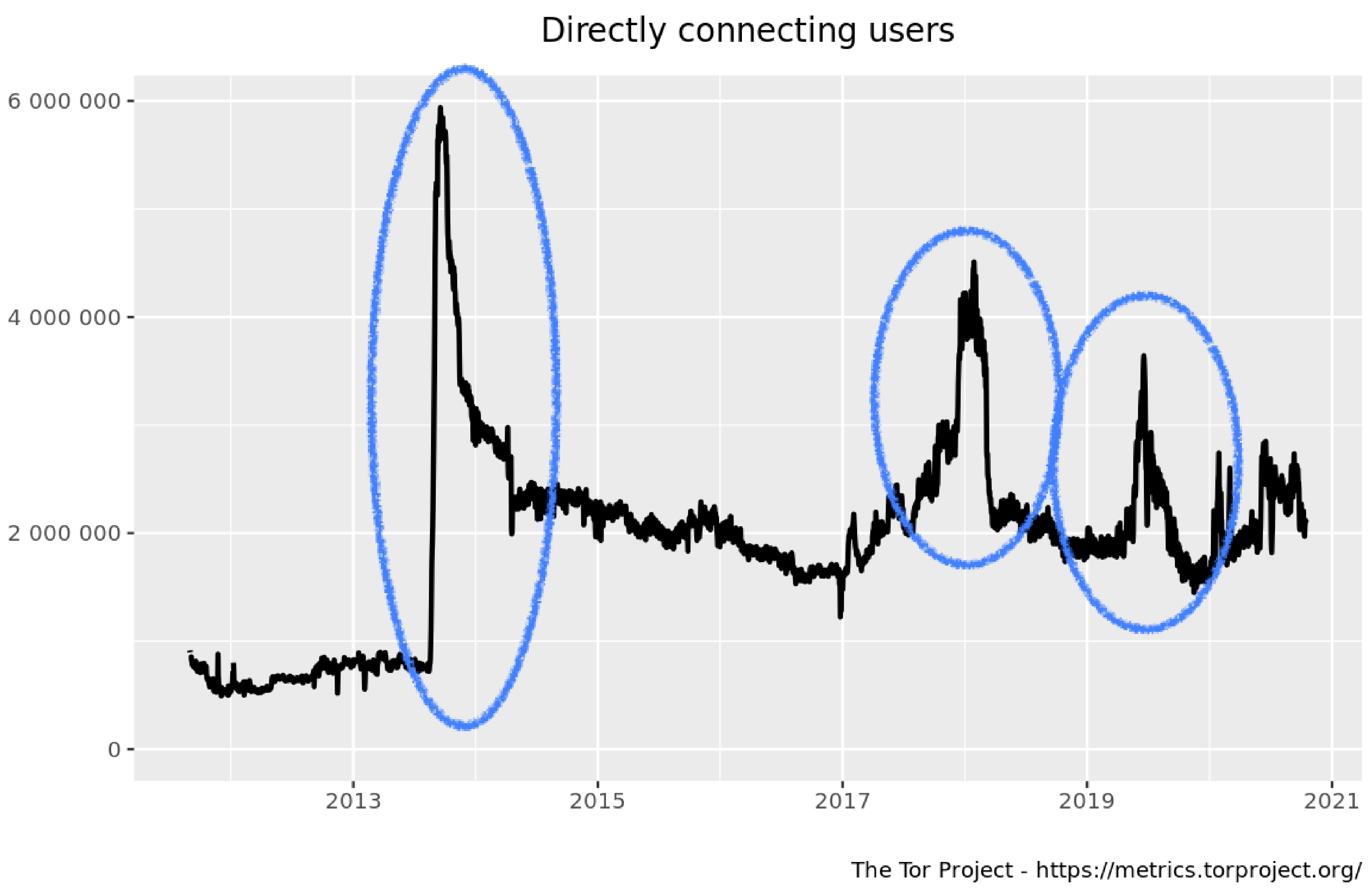
- Chaum is considered the “father” of anonymous communications
- In the 90s: more extensive research and experiments
- End of 90s: onion routing (NRL, ZKS’s Freedom Network)
- For more information see: Danezis, Diaz, and Syverson, *Systems for anonymous communications*, Handbook of Financial Cryptography and Security, 2009

Genesis and evolution

- NRL's Goldschlag, Reed, and Syverson worked on onion routing (1996)
- First prototype had 5 nodes, used cover traffic and mixing
- *Dingledine* and *Mathewson* worked with *Syverson* on **Tor** (2003-2004)
- In 2006 *The Tor Project* was founded with support from EFF and others
- Today Tor is the most widely used anonymous-communication system
- It has evolved significantly in the meantime (not very well documented)
  - Dingledine, Mathewson, Murdoch, and Syverson, *Tor: the second-generation onion router*, 2014 draft http://sec.cs.ucl.ac.uk/users/smurdoch/papers/tor14design.pdf
  - See also https://blog.torproject.org/top-changes-tor-2004-design-paper-part-1

Number of daily users

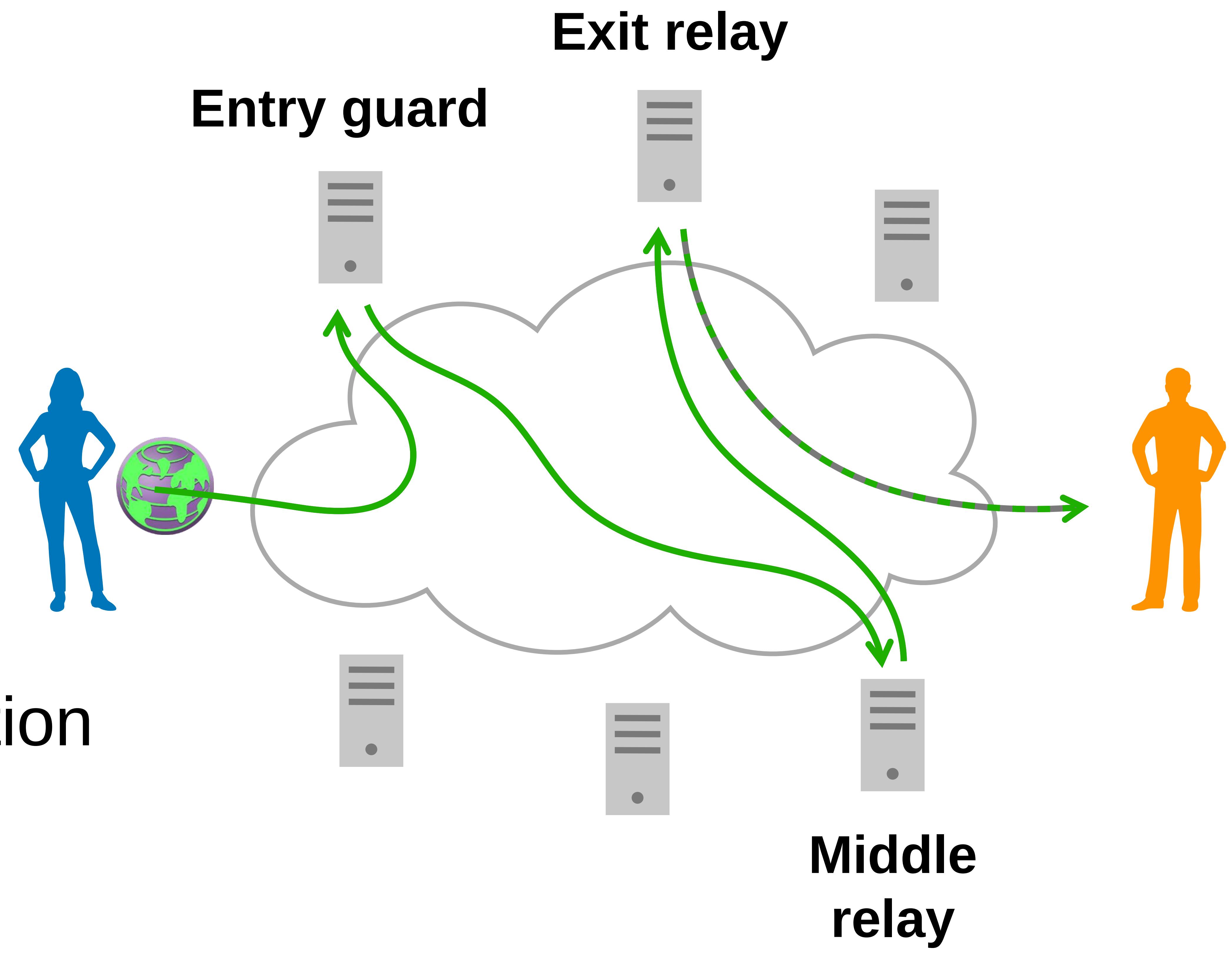




- Typically around 2M daily users (estimated based on directory lookup frequency)
- Spike starting Sep 2013 due to Mevade botnet
- Spikes between Dec 2017 and Mar 2018 related to DDoS attack in Germany
- May 2019: Tor becomes directly accessible in Iran; blocked again end of Jun 2019
- https://trac.torproject.org/projects/tor/wiki/doc/MetricsTimeline

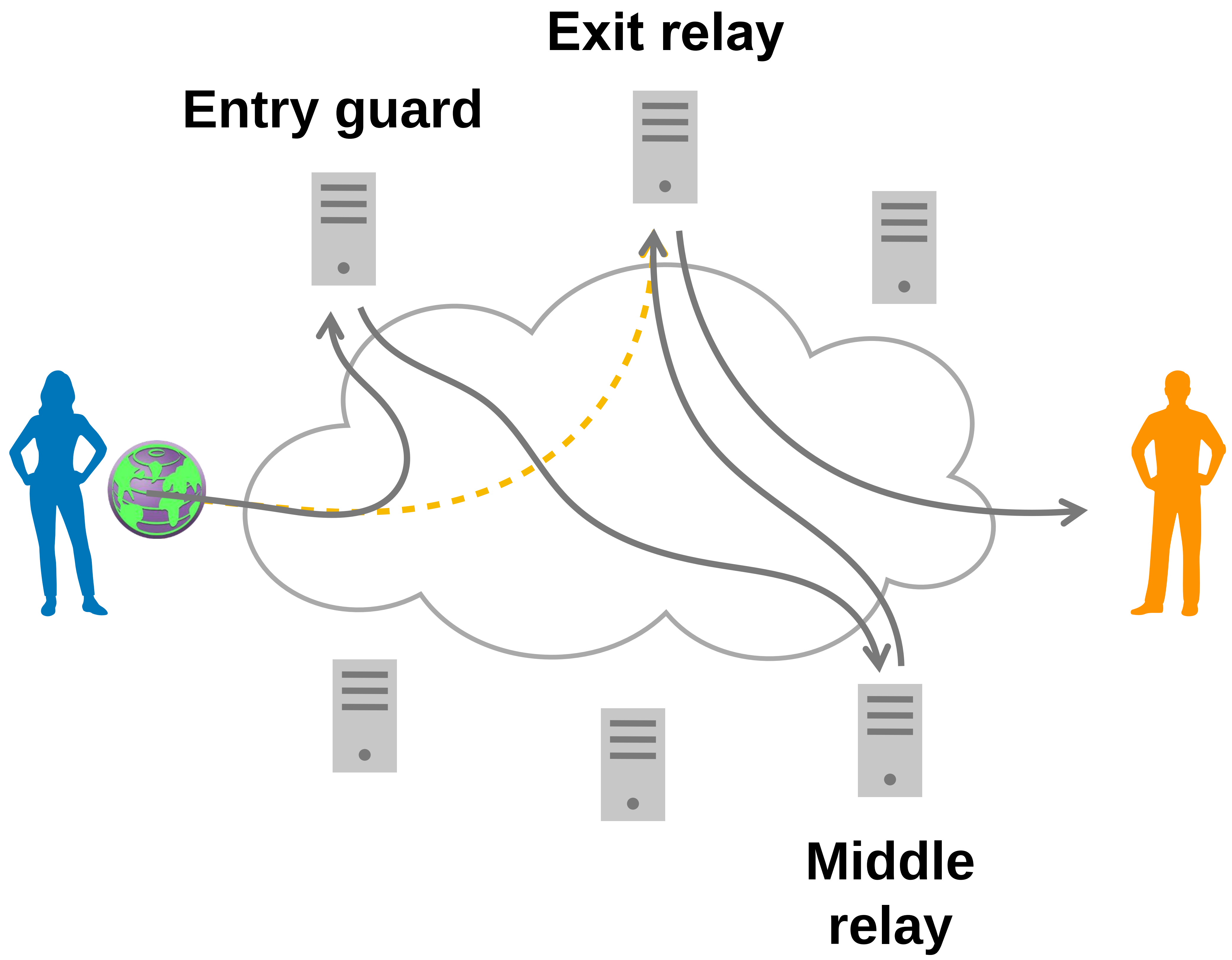
Tor: basics

- Circuits established over *3 relays*
- Telescopic setup (forward security!)
- *Per-hop TCP*, established on the fly
  - Avoid TCP stack fingerprinting
- *Per-hop TLS* (except on the last hop)
  - Multiple circuits over same TLS connection
  - End-to-end HTTPS is possible



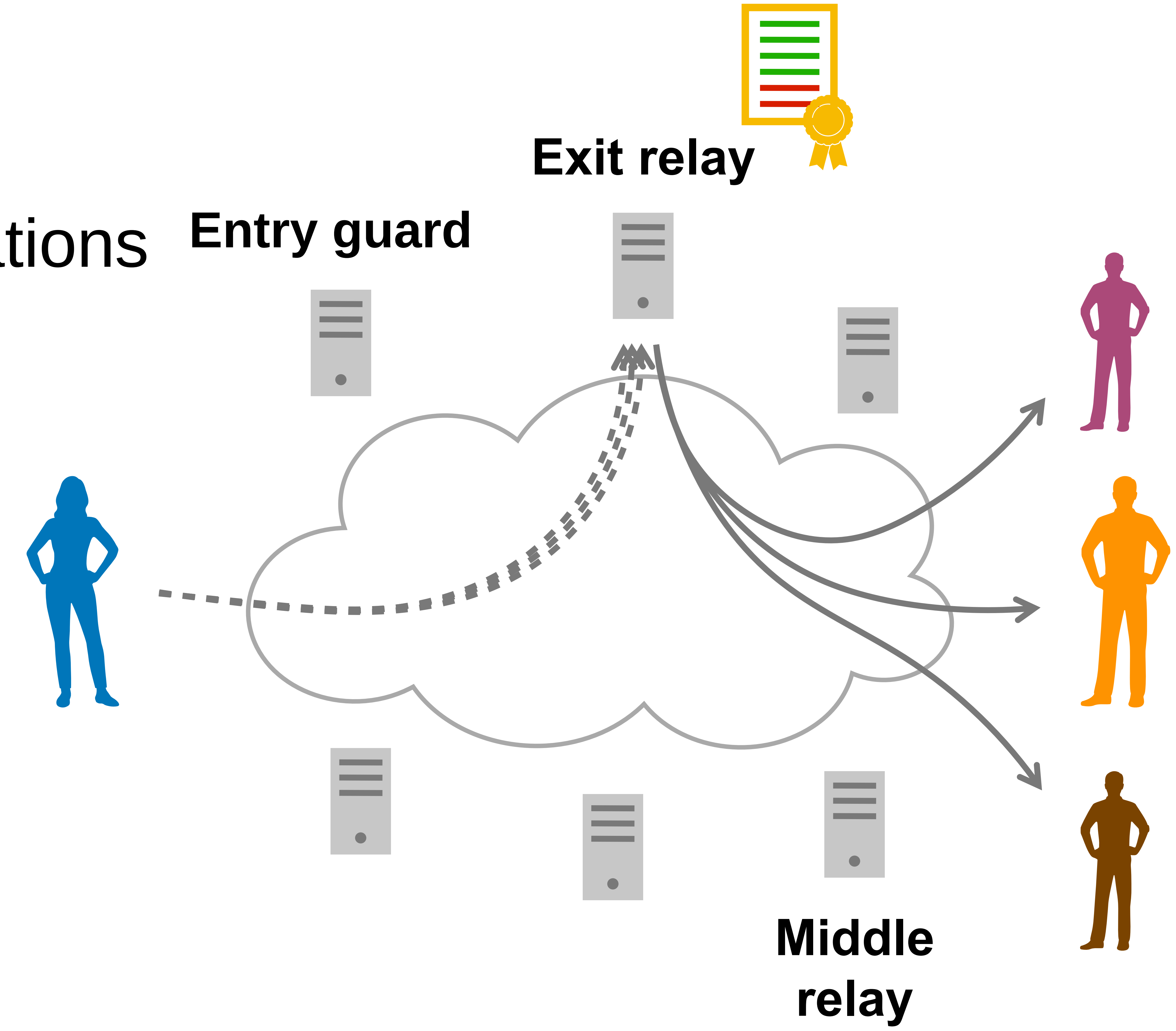
Tor: basics

- Main tool: Tor browser (Firefox)
  - Cleans HTTP/HTTPS traffic
- Supports SOCKS proxy
  - Any TCP application can make use of a Tor connection
- *End-to-"end" integrity* checking
  - Establishes a secure channel between client and exit relay

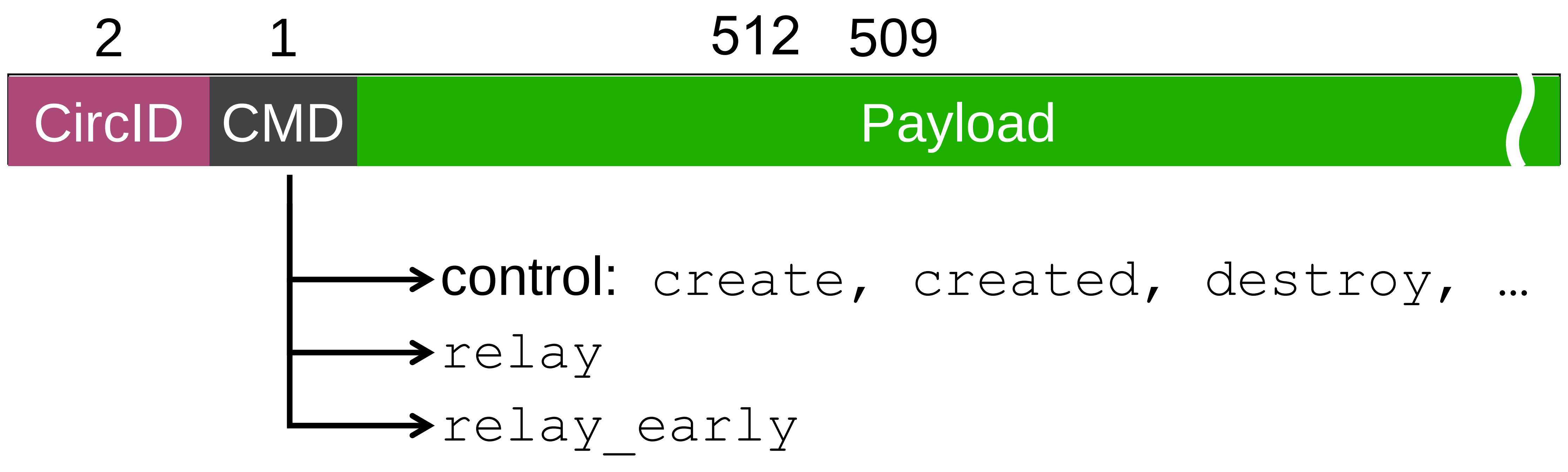


Tor: additional features

- *Exit policies* (exit can restrict the destinations they connect to)
- Multiple *streams* per circuit
- Censorship resistance (*bridges, pluggable transports*)
- *Hidden services*
  - Provide receiver anonymity
  - Use `.onion` URL *(not in DNS)*

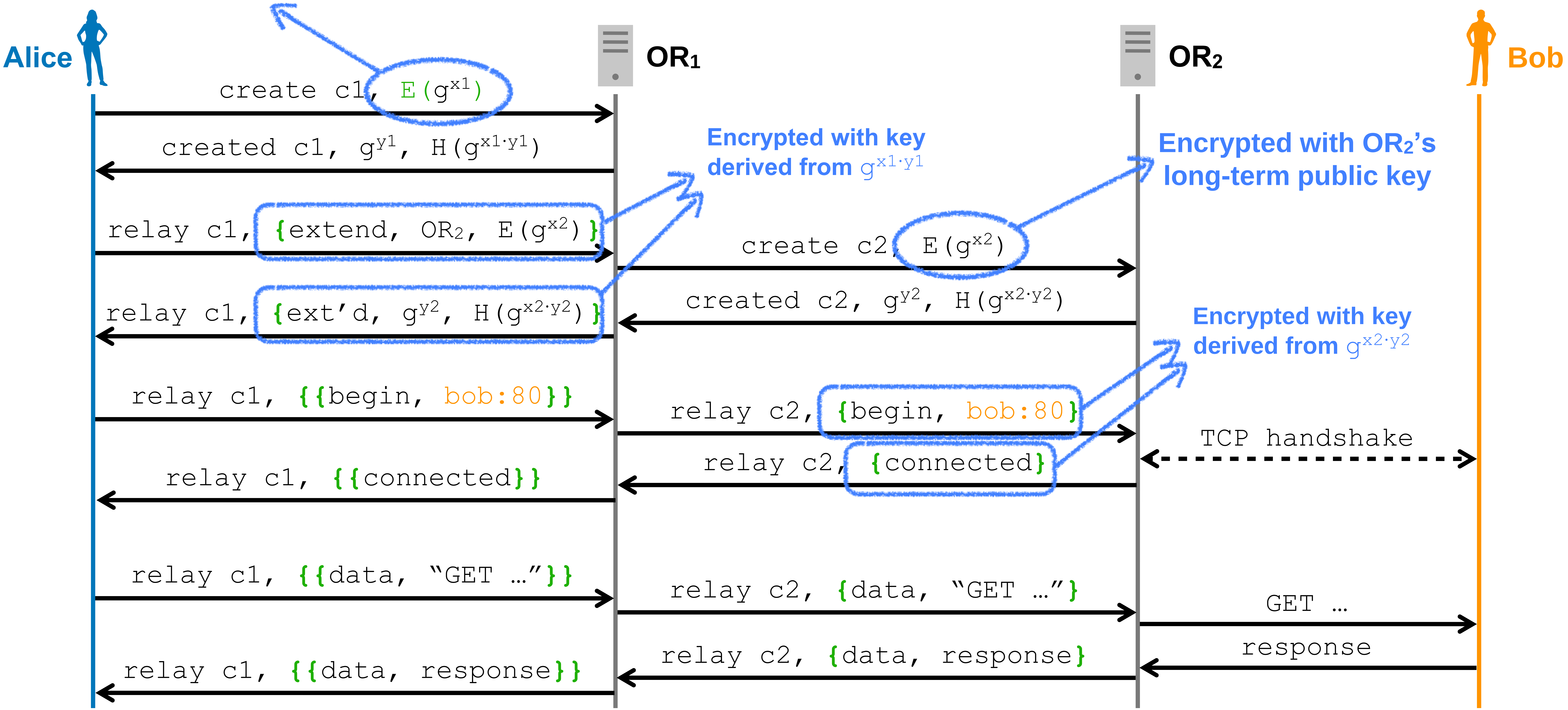


Tor cells



- Basic unit is the cell (512 bytes)
- It contains a circuit ID and a command field (in cleartext)
- Same for cells in both directions
- A relay cell's payload is decrypted, and *its digest is checked*:
  - If correct: check command
  - Otherwise replace circuit ID and forward cell along
  - → only exit relay sees payload

Example circuit communication



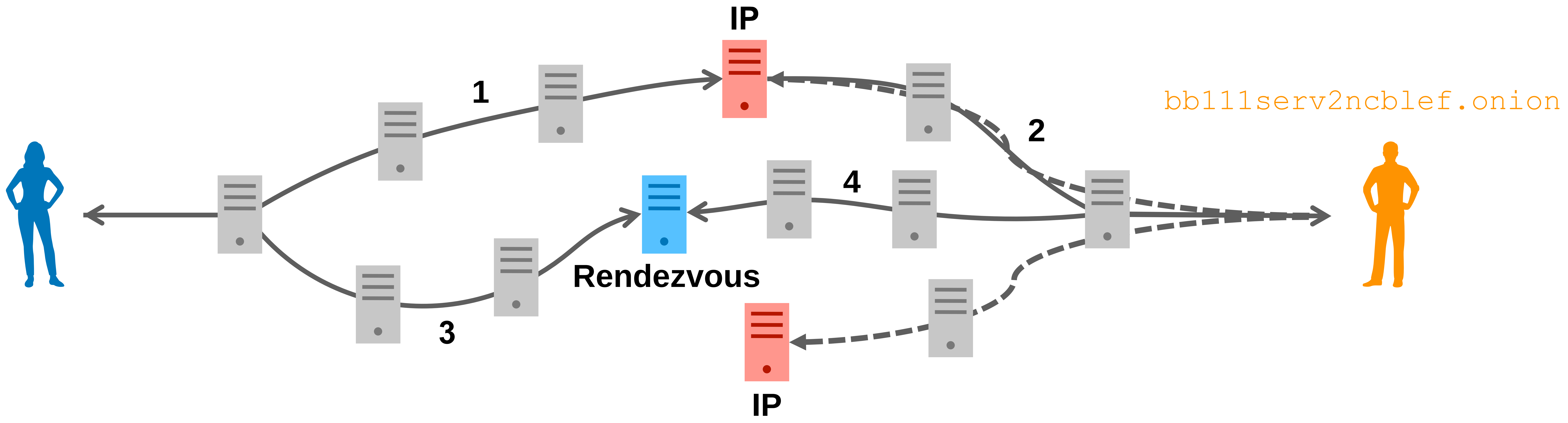
Circuit setup with three hops

Circuit extension with `relay_early`

- Given what we have seen so far, what is the maximum path length?
- There is no limit! Is there a problem with this?
- Path of arbitrary length can be used for *very cheap DoS*
  - Simply create a circuit that goes through all honest nodes, dozens of times: incredibly large amplification factor
- Solution: Tor `extend` cells can only be contained in `relay_early` cells
- *Each relay allows only 8* `relay_early` *cells per circuit*
  - Maximum path length capped at 9

Circuit extension with `relay_early`

Hidden services



- The hash of Bob's public key is the identifier of his hidden service
- Bob has connections to a set of *introduction points* (IP)
- To communicate, Alice connects to an IP and suggests a *rendezvous*
- Bob can connect to the rendezvous and start the communication

Hidden services for non-anonymous servers

- Facebook has a hidden service: *facebookcorewwwi.onion*. Why?
- Facebook is in favor of people connecting over Tor
  - It allows people in censored countries (Iran, China, …) to access FB
- Users connect to Facebook through Tor to reduce tracing, hide location information, etc.
- However, with normal connection, Facebook sees all communications over Tor coming from a few exit nodes:
  - The hidden service avoids interference with their filtering heuristics

Directory authorities

- How do the clients know what relays there are?
  - (If an adversary can supply the list, de-anonymization is trivial!)
- *10 directory authorities* running a consensus algorithm
- The authorities track the state of relays, store their public keys
- Client software (Tor browser) comes with a list of the authorities' keys
  - A client accepts a consensus document if signed by ≥ 50%
- The centralized authorities are an *important weakness* of Tor
  - An adversary compromising 5 authorities can compromise Tor

Tor Metrics

Directory authorities

Home » Services » Relay Search » Search for flag:authority

Relay Search

flag:authority

flag:authority

Show 10 entries

Nickname†	Advertised Bandwidth	Uptime	Country	IPv4	IPv6	Flags	Add. Flags	ORPort	DirPort	Type
dizum (2)	3.87 MiB/s	5d 2h		45.66.33.45	-			443	80	Relay
Serge (1)	1.53 MiB/s	1d 22h		66.111.2.131	2610:1c0:0:5::131			9001	9030	Relay
moria1 (1)	500 KiB/s	11d 10h		128.31.0.34	-			9101	9131	Relay
tor26 (1)	75 KiB/s	8d 2h		86.59.21.38	2001:858:2:2:aabb:0:563b:1526			443	80	Relay
bastet (1)	50 KiB/s	14d 21h		204.13.164.118	2620:13:4000:6000::1000:118			443	80	Relay
maatuska (8)	50 KiB/s	13d 18h		171.25.193.9	2001:67c:289c::9			80	443	Relay
dannenberg (1)	40 KiB/s	25d 16h		193.23.244.244	2001:678:558:1000::244			443	80	Relay
Faravahar (1)	40 KiB/s	112d 19h		154.35.175.225	2607:8500:154::3			443	80	Relay
gabelmoo (1)	40 KiB/s	20d 6h		131.188.40.189	2001:638:a000:4140::ffff:189			443	80	Relay
longclaw (1)	38 KiB/s	4d 2h		199.58.81.140	-			443	80	Relay
Total	**6.21 MiB/s**									

Showing 1 to 10 of 10 entries

Directory authorities

- Every relay periodically reports a signed statement (state, stats.)
- DAs also act as *bandwidth authorities*: verify bandwidth of nodes
  - This determines the stable and fast flags, and weight
  - See also https://blog.torproject.org/lifecycle-new-relay
- Every hour, the DAs compute and sign a new consensus document
- Sybil protection: DAs limit the number of relays per IP subnet
- Centralized architecture can be a problem for scalability:
  - Almost every relay acts as a *directory cache*

Censorship resistance in Tor

- Problem: relay nodes are publicly listed and can be blocked
- The Tor network contains several *bridge relays* (or *bridges*)
  - Not listed in main Tor directory, downloaded on demand
  - Used to circumvent censors which black-list IP addresses of Tor relays
  - Not (all) publicly available, some distributed through friends networks
- Problem: deep packet inspection allows detection of Tor traffic
- Solution: obfuscate the traffic (*pluggable transports*)

Pluggable Transports

- Obfuscation tries to hide Tor traffic features
  - Packet lengths
  - Timing
  - Additional encryption
  - …
- Censors improve their detection heuristics
  - → This gives rise to the *censorship arms race*

Currently deployed PTs

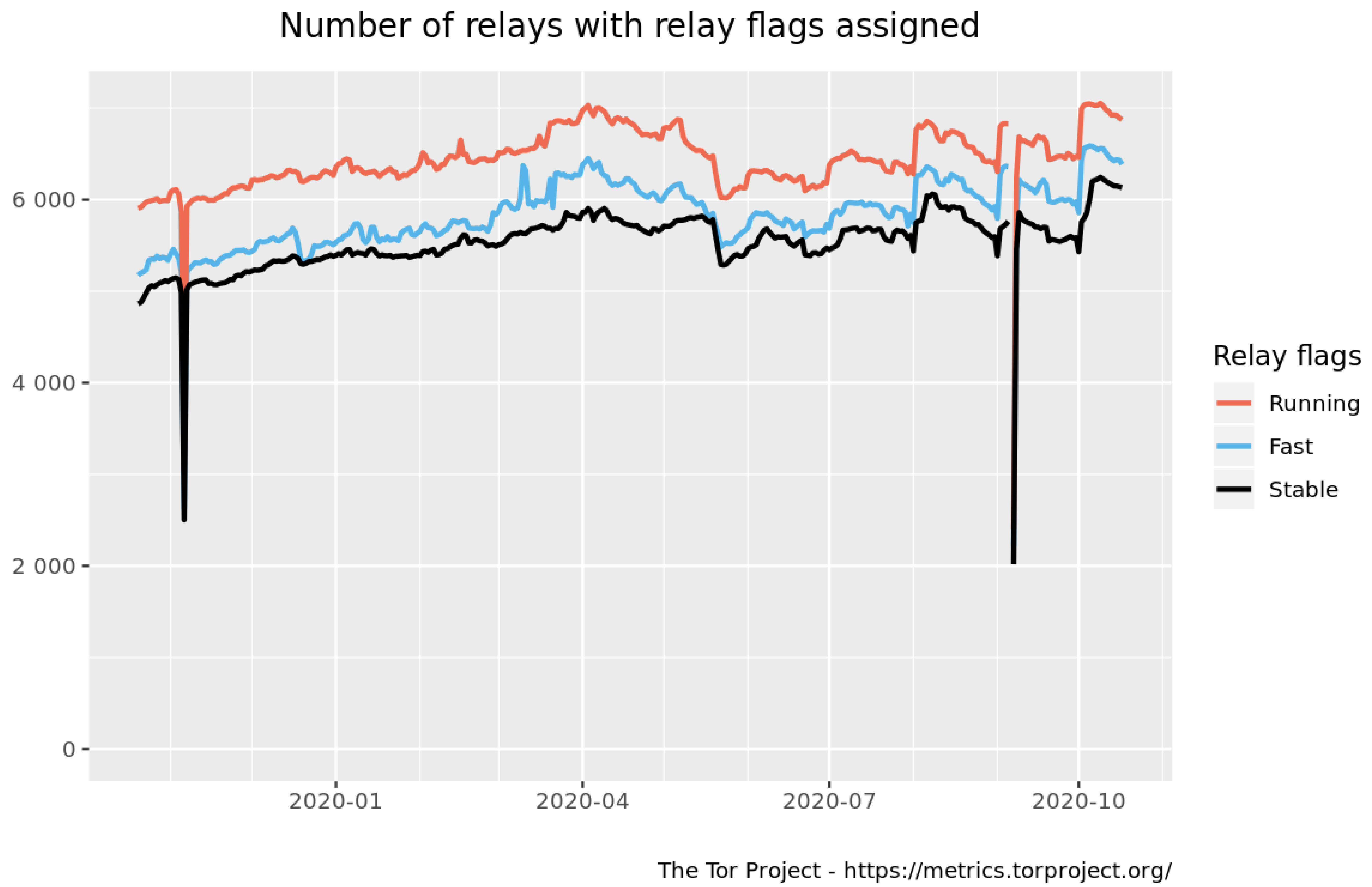
These Pluggable Transports are currently deployed in Tor Browser, and you can start using them by downloading and using Tor Browser.

- obfs4
  - **Description:** Is a transport with the same features as ScrambleSuit but utilizing Dan Bernstein's elligator2 technique for public key obfuscation, and the ntor protocol for one-way authentication. This results in a faster protocol.
  - **Language:** Go
  - **Maintainer:** Yawning Angel
  - **Evaluation:** obfs4 Evaluation
- meek
  - **Description:** Is a transport that uses HTTP for carrying bytes and TLS for obfuscation. Traffic is relayed through a third-party server (Google App Engine). It uses a trick to talk to the third party so that it looks like it is talking to an unblocked server.
  - **Language:** Go
  - **Maintainer:** David Fifield
  - **Evaluation:** meek Evaluation
- Format-Transforming Encryption (FTE)
  - **Description:** It transforms Tor traffic to arbitrary formats using their language descriptions. See the research paper.
  - **Language:** Python/C++
  - **Maintainer:** Kevin Dyer
  - **Evaluation:** FTE Evaluation
- ScrambleSuit
  - **Description:** Is a pluggable transport that protects against follow-up probing attacks and is also capable of changing its network fingerprint (packet length distribution, inter-arrival times, etc.).
  - **Language:** Python
  - **Maintainer:** Philipp Winter
  - **Evaluation:** ScrambleSuit Evaluation

https://2019.www.torproject.org/docs/pluggable-transports.html.en

Tor statistics

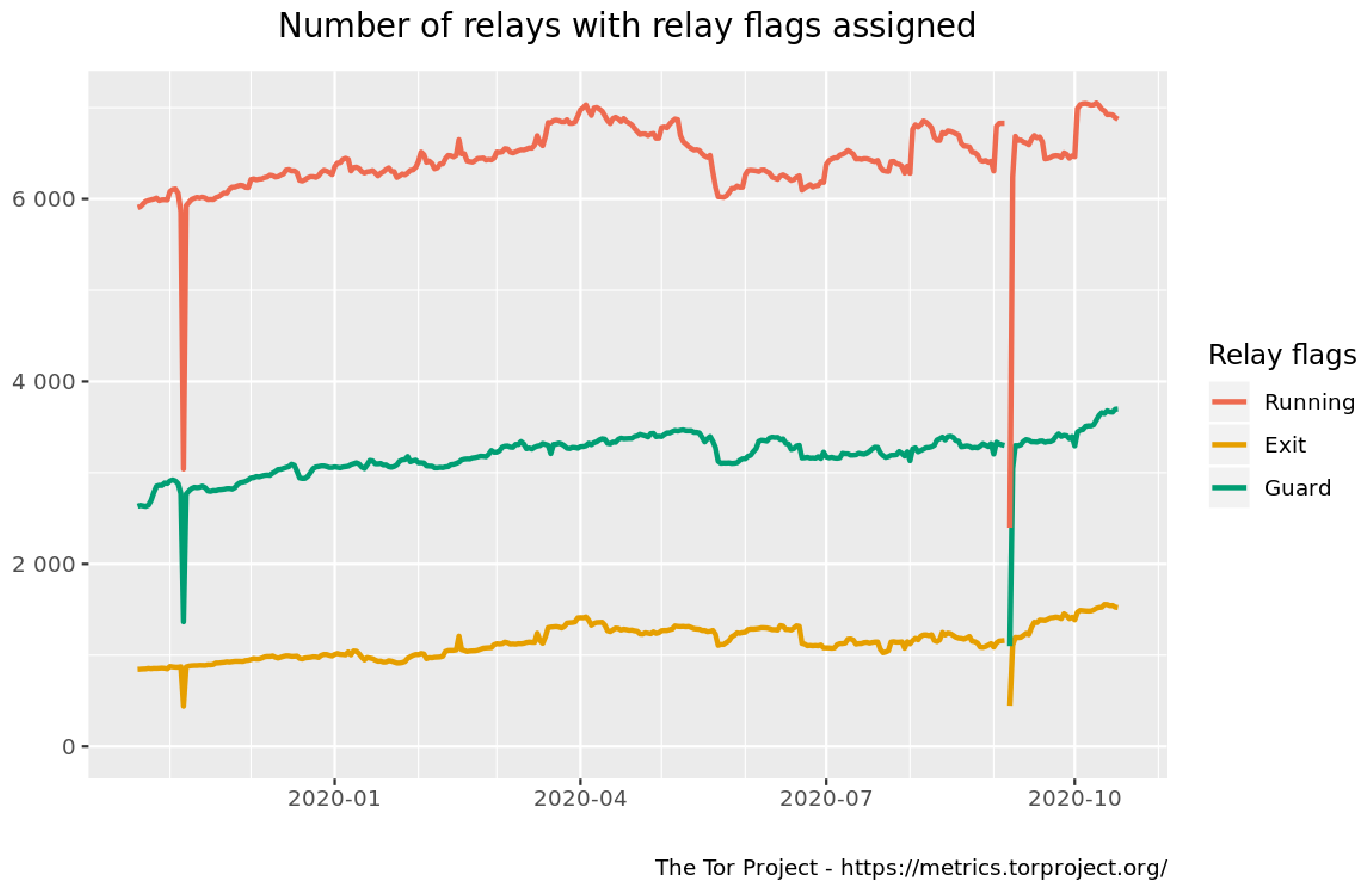
Number of relays



- *Running* = all relays, *Fast* = high bandwidth, *Stable* = up for a long time

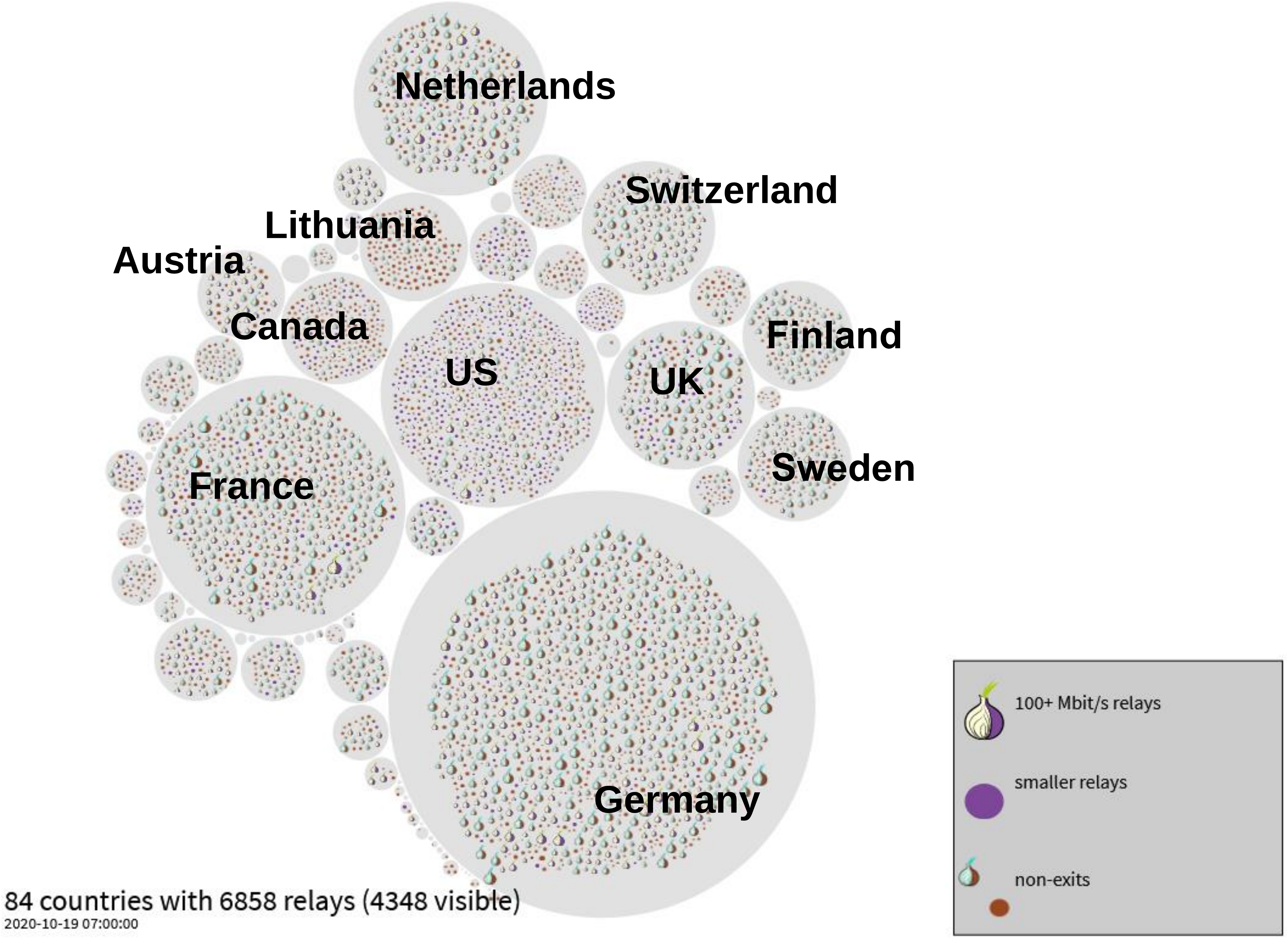
Tor Metrics

Number of relays



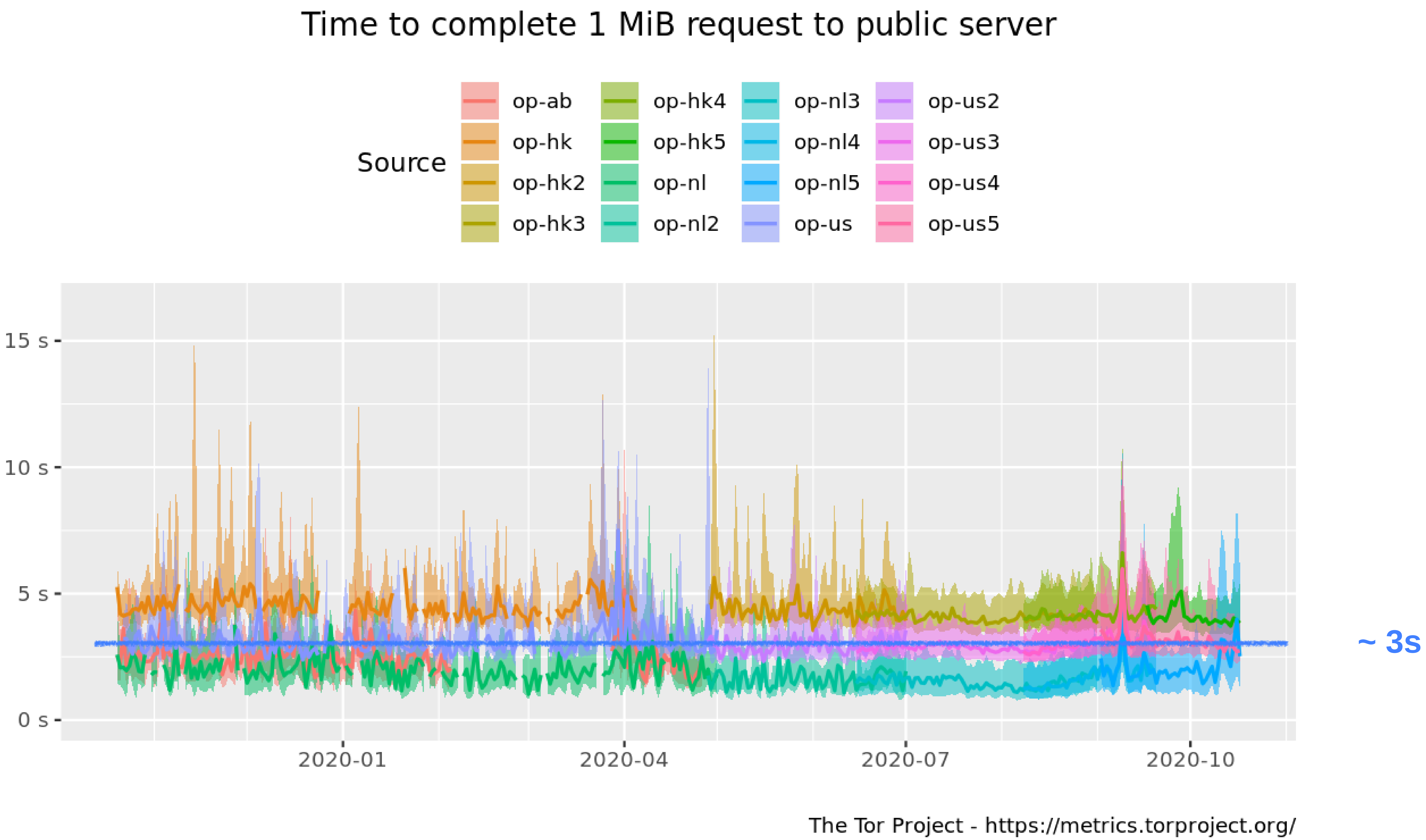
- The *exit relays* and *entry guards* are a fraction of all the relays

Relays per country

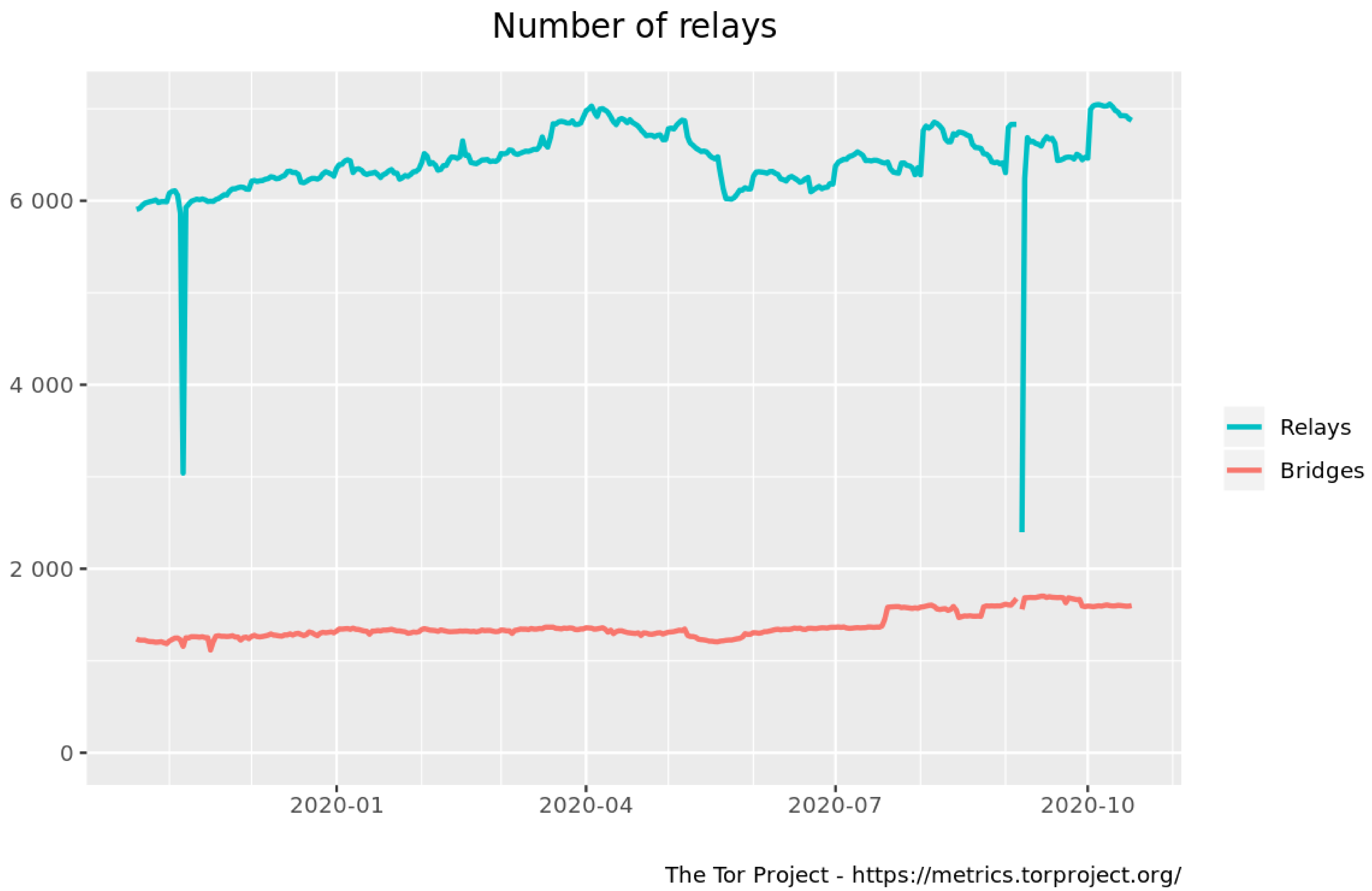


https://metrics.torproject.org/bubbles.html#country

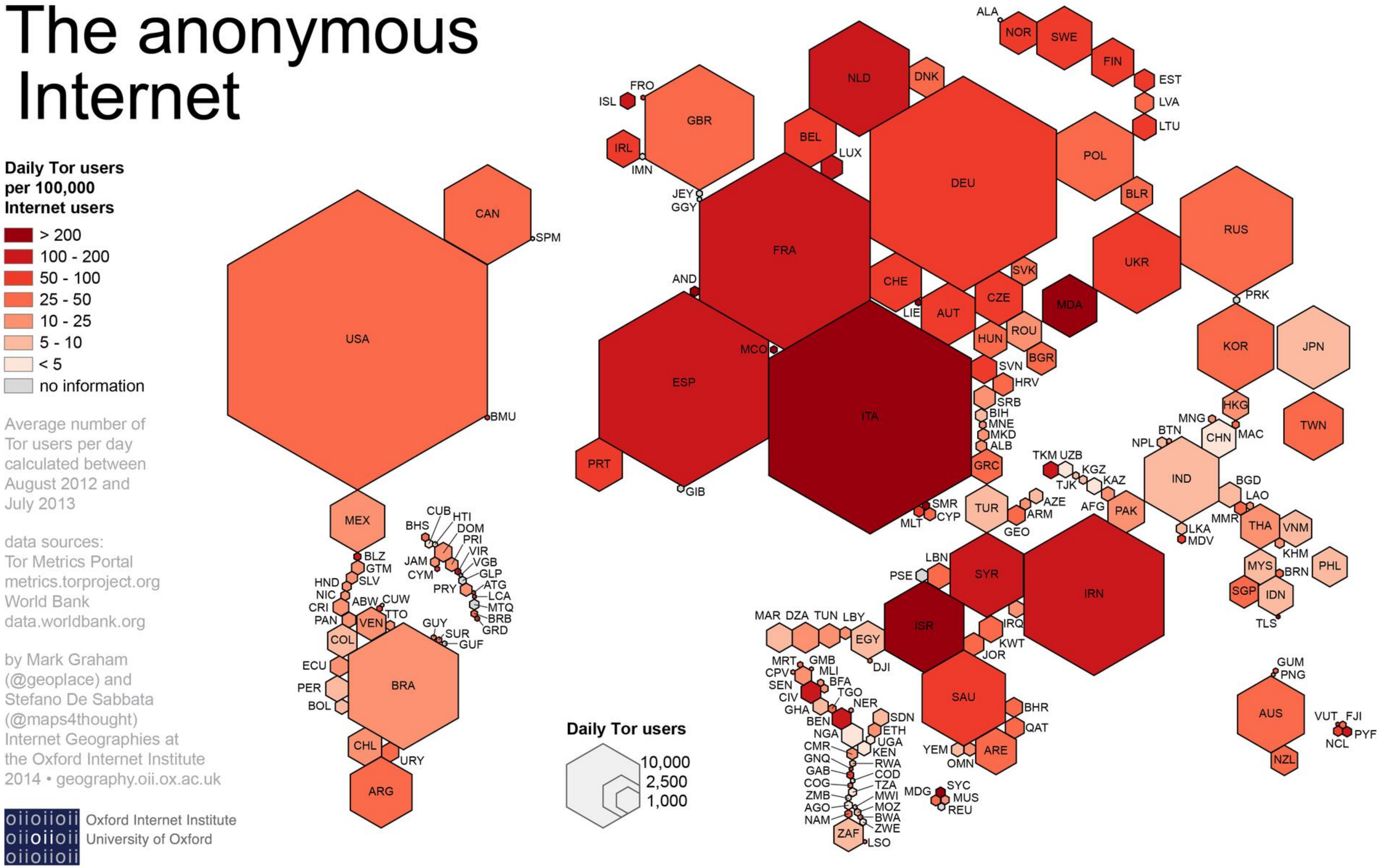
Performance (file download)



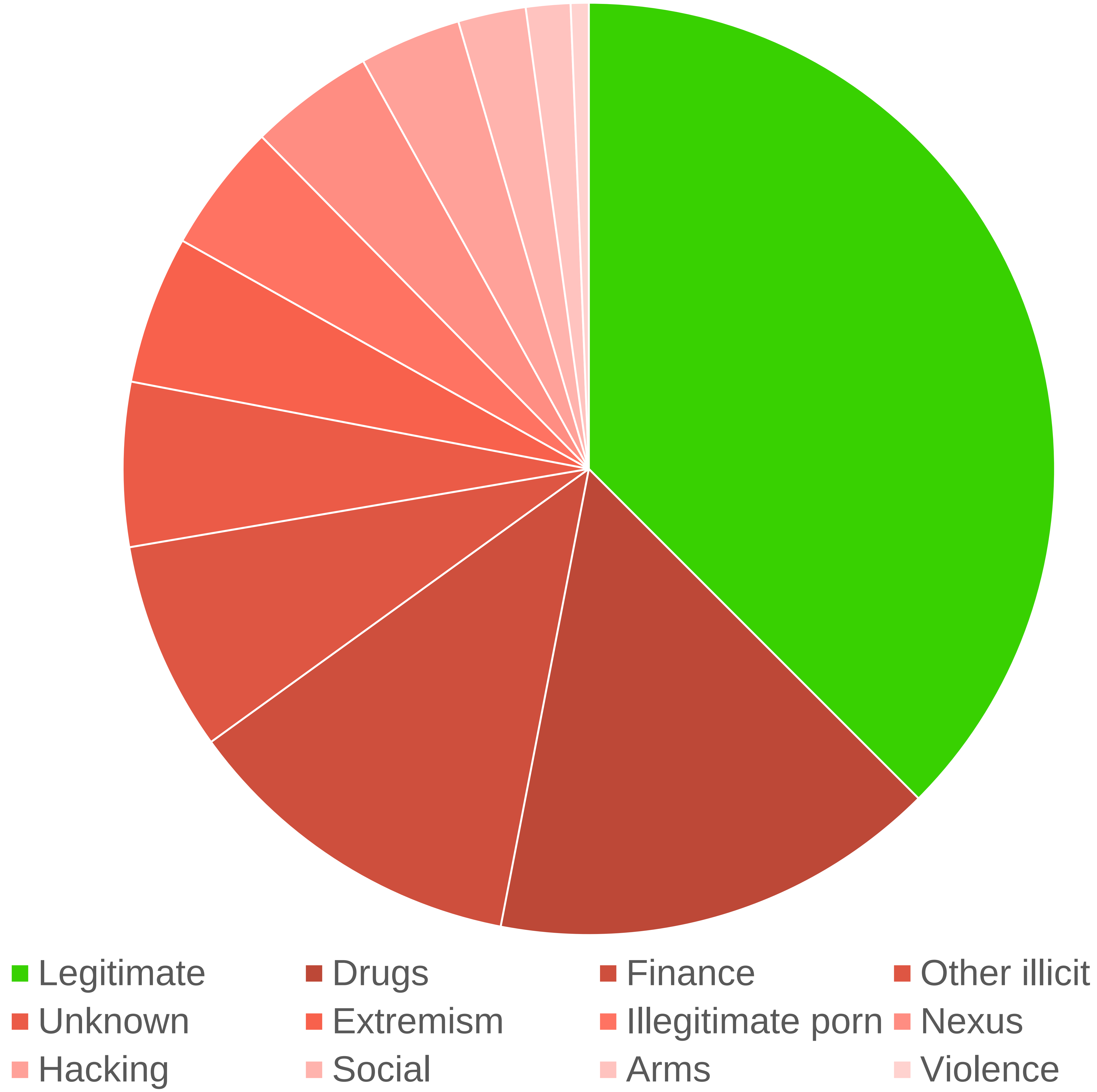
Bridges in the Tor network



Who uses Tor?


The anonymous Internet
Daily Tor users per 100,000 Internet users
> 200
100 - 200
50 - 100
25 - 50
10 - 25
5 - 10
< 5
no information
Average number of Tor users per day calculated between August 2012 and July 2013
data sources:
Tor Metrics Portal
metrics.torproject.org
World Bank
data.worldbank.org
by Mark Graham (@geoplace) and Stefano De Sabbata (@maps4thought)
Internet Geographies at the Oxford Internet Institute
2014 • geography.oii.ox.ac.uk
Oxford Internet Institute
University of Oxford
Daily Tor users
10,000
2,500
1,000
USA
CAN
MEX
BRA
ARG
GBR
FRA
DEU
ESP
ITA
NLD
RUS
IRN
IND
JPN
KOR
AUS

What is Tor used for?



Category	Websites
None	2,482
Other	1,021
Drugs	423
Finance	327
Other illicit	198
Unknown	155
Extremism	140
Illegitimate pornography	122
Nexus	118
Hacking	96
Social	64
Arms	42
Violence	17
Total	5,205
Total active	2,723
Total illicit	1,547

D. Moore & T. Rid. *Cryptopolitik and the Darknet.* 2016
https://doi.org/10.1080/00396338.2016.1142085

Summary

What you should remember about anonymous-communication systems

- You cannot be anonymous on your own → *anonymity set*
- Multiple relays and layered encryption enable anonymous communication
- Two main types of anonymous-communication systems:
  - *Mix-nets*: slow, strong guarantees
  - *Circuit-based (onion-routing) systems*: low latency, possible attacks for strong adversary
  - Tor is the most widely used onion-routing system
- Anonymous communication is a tool that can be used for both good and bad purposes