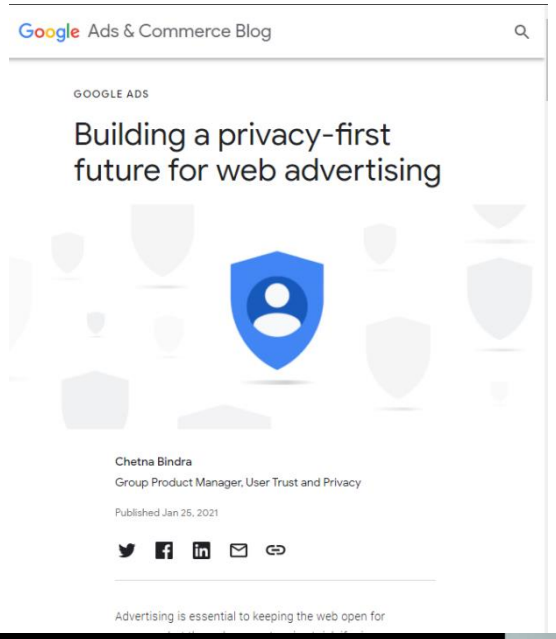
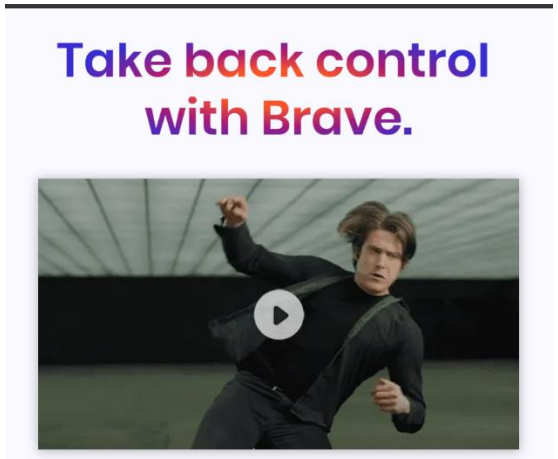
Over-the-top or built-in approaches to improve privacy at the network layer

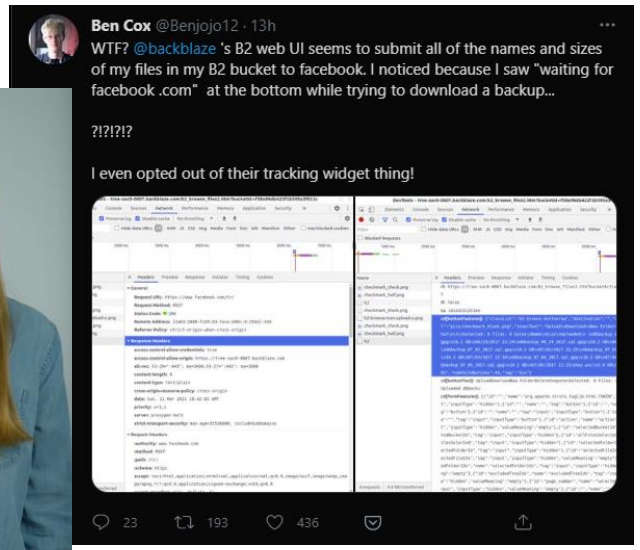
A tale of 2 complementary approaches to anonymize network traffic

Privacy protection

A global consumer demand





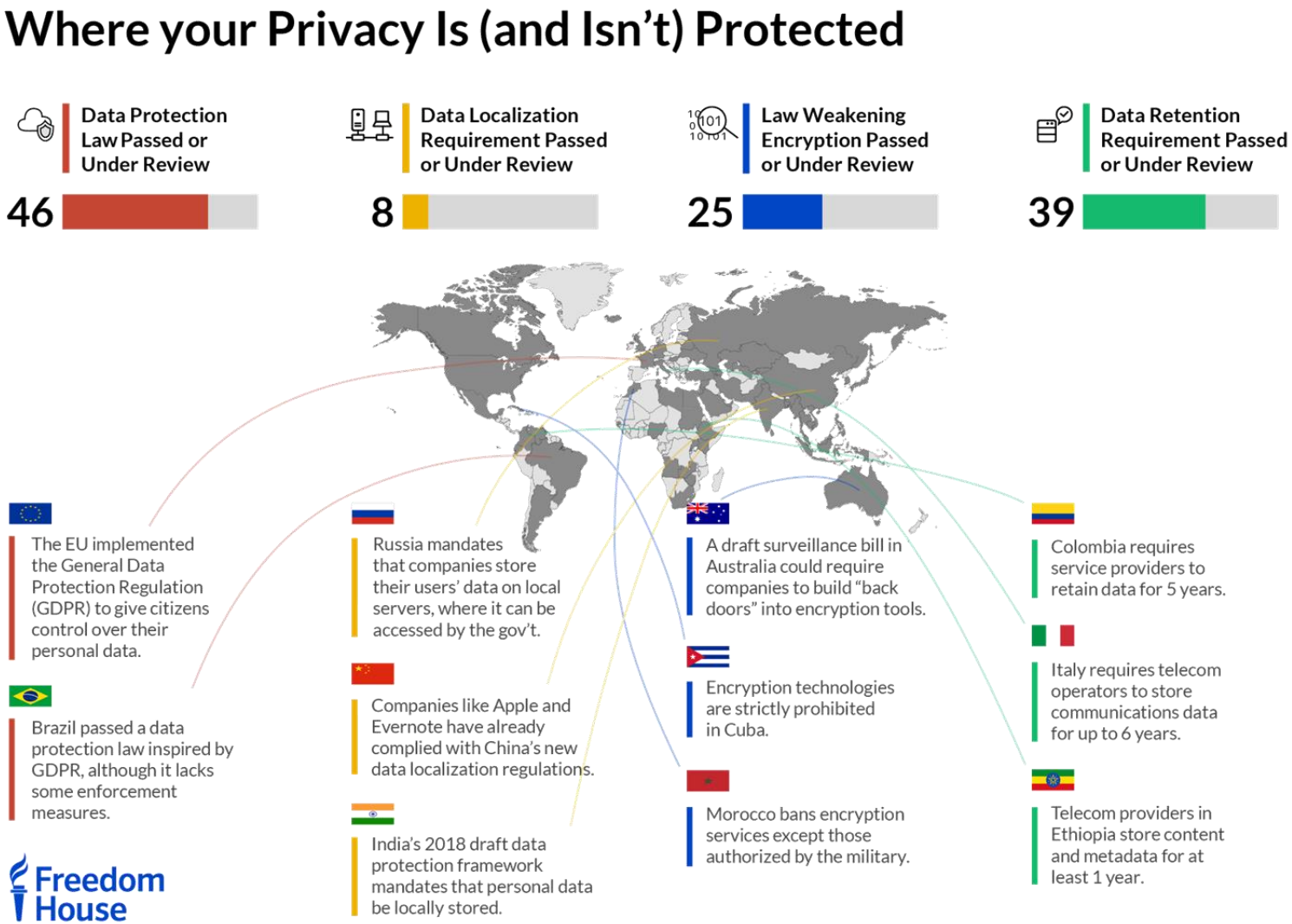


- Recent announcement by Google that it will propose alternatives to third party cookies together with privacy sandboxes in Chrome.
- Strong statements by Apple on privacy pillars during WWDC21 and in recent iOS 15 release
- Brave browser and wish to help user control their data
- New « Pixel tracking » technique to bypass cookies limitations

➔ To go further, need to eliminate other identity linking identifiers, and IP is one of them.

Privacy protection

A global regulatory concern



- Laws passed all over the world to protect citizen's privacy
  - GDPR in Europe
  - Cybersecurity Law in China with privacy protection guidelines
- Difficult balance to find
  - Data protection is sometimes considered as a potential danger for national security
  - Pressure in some standard bodies to insert backdoors allowing states to perform legal interception easily

Privacy protection from a network perspective

- Defense against someone eavesdropping a communication between a source and destination from one or several vantage points (not global)
  - Depending on the attacker and the level of privacy protection we want to have, multiple mechanisms can be used:
    - Trust in a third party (the ISP?) to protect user privacy ⇔ direct business relationship Vs. Indirect data reselling.
    - Effort required to determine the traffic source and destination: Address lookup / cryptographic attack / Timing and topology analysis
    - Need for destination to be hidden ⇔ Use of indirect routing or anonymous source-based routing
- Hiding the source from the destination in specific contexts
  - Requests to privacy-hungry services (Recent discussions in the Web community on 3rd party cookies and pixel-based tracking)
- Protecting against a global eavesdropper
  - Eavesdropping of ***all the links*** should be considered part of the threat model
  - If an actor controls ***all the nodes*** in the network, it is impossible to provide privacy in the network

Two approaches to implement privacy at the network level

Over-the-top approach

- Evolutionnary approach, similar to IPSec for privacy
- Main objective: hiding the source address of a packet or network flow, and increase privacy to face increasingly powerful adversaries
- Mostly based on trusted third parties ⇔ Dependent on the third party's willingness to protect the user's privacy
- Can be deployed easily with an appropriate business case.

Built-in approach

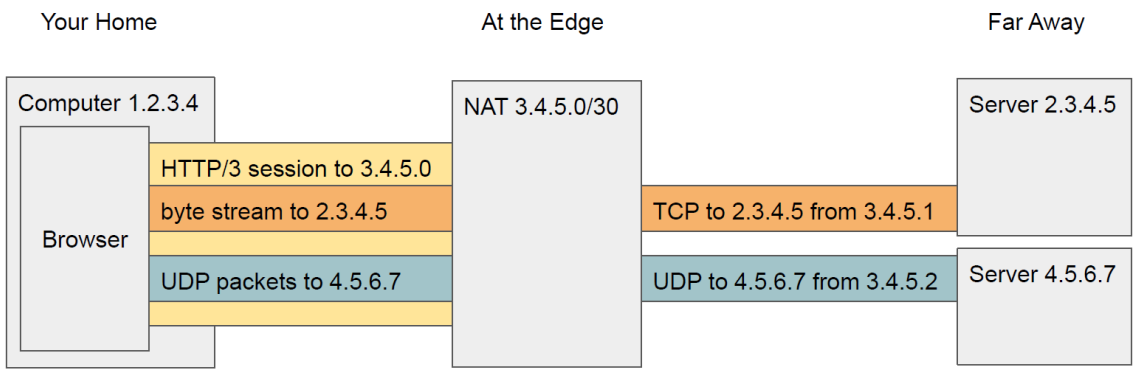
- Requires strong changes in the behavior of network protocols (Nearly clean slate approaches)
- Main objective: Protecting privacy against a state of the art adversary (post-Snowden)
- Possibility to avoid third party involvement provided we question the use of destination-based routing ⇔ Less dependencies
- Academic / Future internet projects

Gnatcatcher

A lightweight IP privacy approach from Google

What is Gnatcatcher?

- Initiative from Google presented during IETF 110 meeting in the PEARG working group
- Global Network Address Translation Combined with Audited and Trusted CDN or HTTP-proxy Eliminating Reidentification
- Combination of:
  - **Near-Path NAT** that allows groups of users to send their traffic through the same privatizing server, effectively hiding their IP addresses from the site host.
  - **Willful IP Blindness** which ensures that sites requiring access to IP addresses for legitimate purposes such as abuse prevention can do so, subject to certification and auditing.



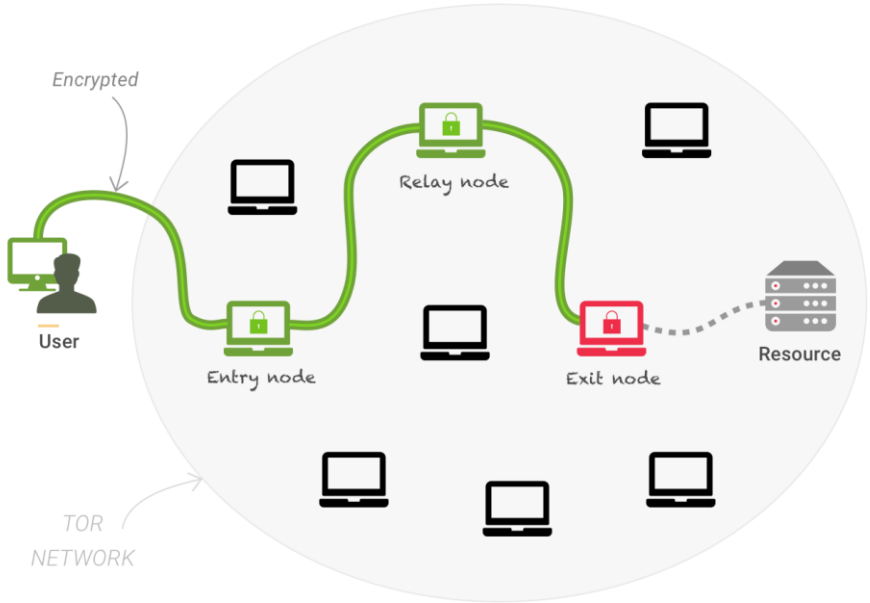
TOR

The second-generation onion router

TOR onion routing

Principle

- Idea taken from mix networks
- Use of proxies and relays to anonymize TCP traffic
- Data sent among a set of relay nodes in the form of recursively encrypted cells. Each node on the path decrypts the cell and relays it to the next node.
- Lightweight system:
  - Use of symetric key after a circuit construction procedure while mix networks use public key cryptography extensively
  - Weak against traffic analysis attacks as there is no packet shuffling mechanism in TOR



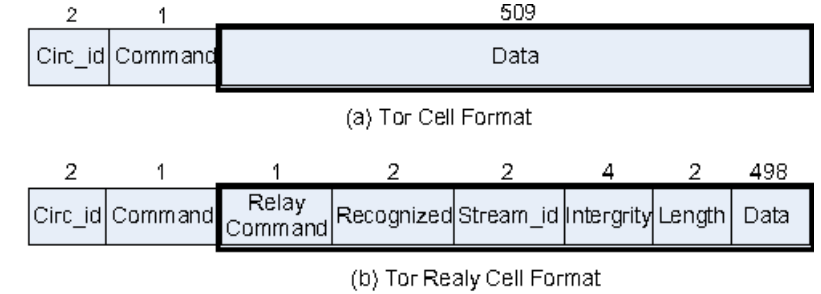
TOR onion routing

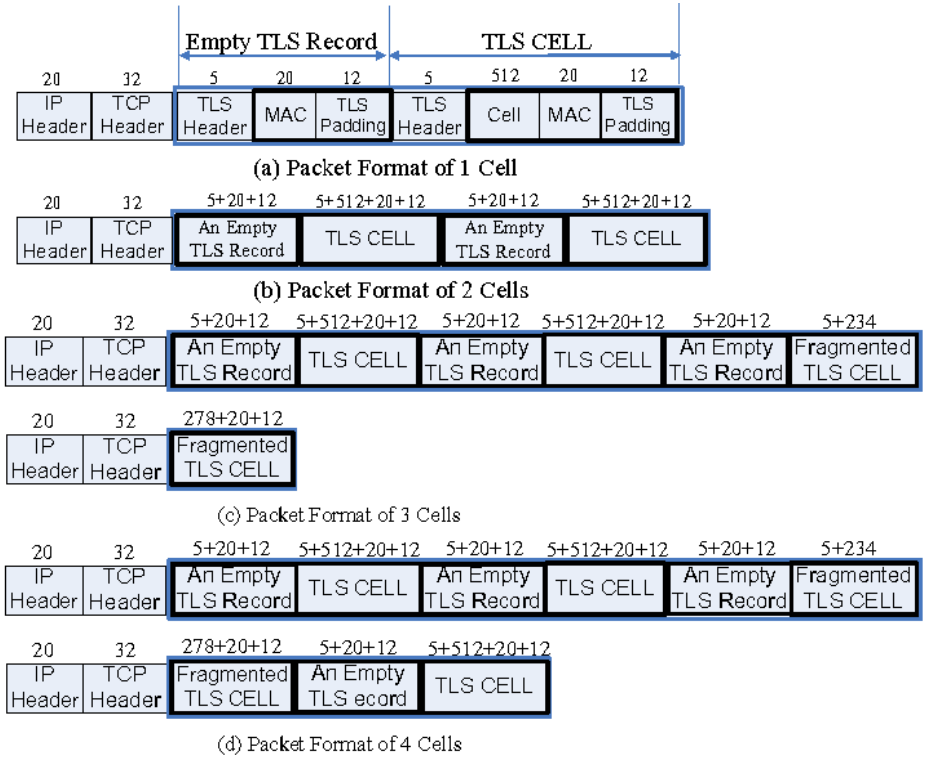
Cells and circuits

- Cells in TOR:
  - Traffic is sent in cells that are either Relay cells or Data cells.
  - Cells are 512 bytes long, and are multiplexed in TLS traffic between the TOR nodes
- Circuits in TOR:
  - TOR relays TCP traffic from an entry point to an exit node using circuits identified by an ID
  - Circuits are built step by step from the entry point to convey traffic anonymously. Nodes in the circuit exchange a symmetric key with the source that is then used to relay traffic.
  - TCP flows can be multiplexed in TOR using a Stream ID.

➔ *Symetric keys in TOR are not changed at each cell*

➔ *A rogue node is able to link cells using the Circuit ID / Stream ID*



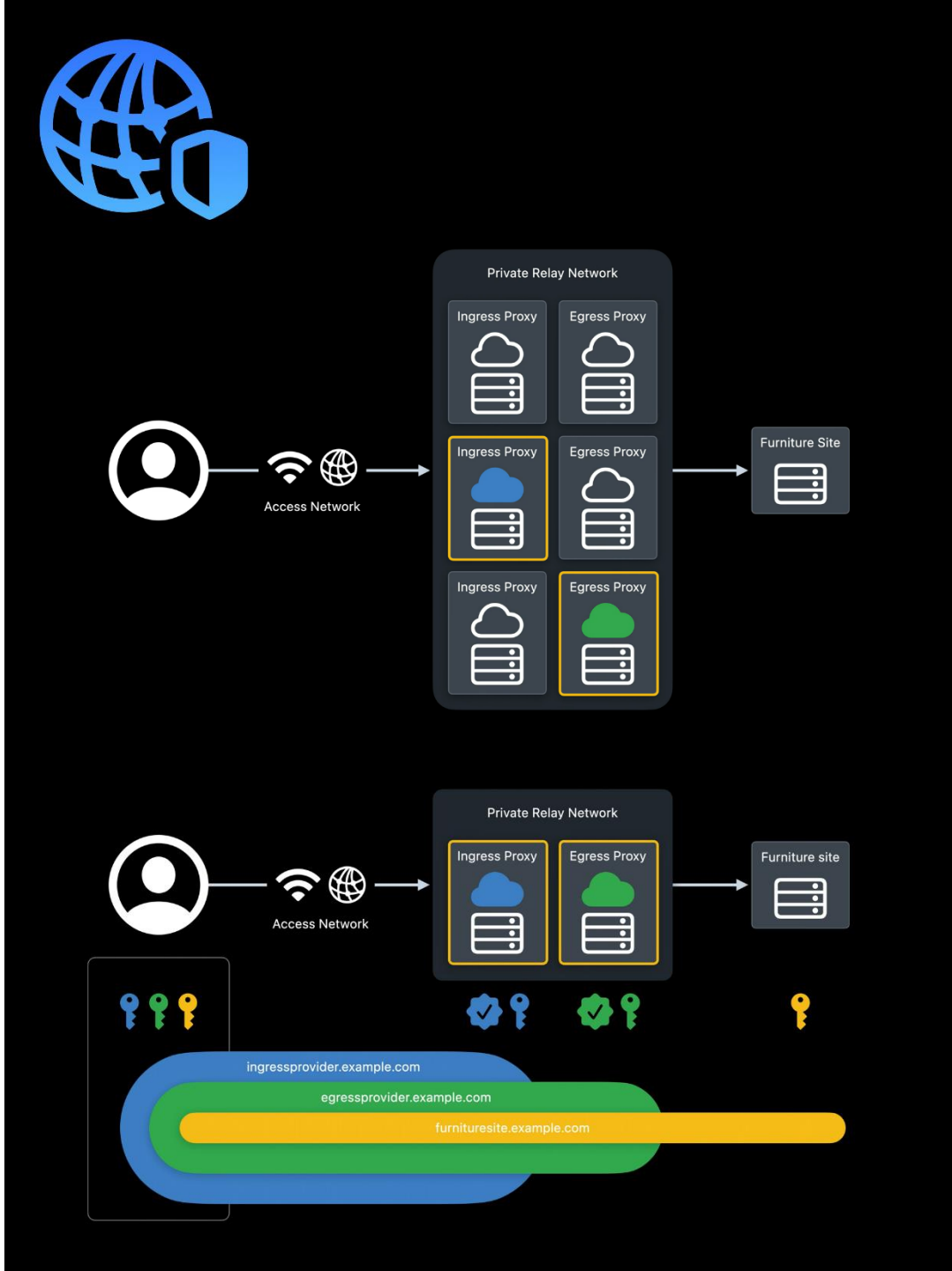


iCloud+ Private relay

An intermediate approach

What is iCloud+ Private Relay

- Product presented by Apple during WWDC21
- Use of a chain of 2 proxies to ensure source-destination unlinkability
  - Ingress proxy encrypts all traffic from a source and shields its address from remote servers
  - Egress proxy protects the destination from the ingress proxy
- Traffic tunneled in QUIC - HTTP/3 tunnels
- Traffic protected using temporary public / private key pairs given by a Private Relay Access Token Server
  - Access token are made unlinkable by use of cryptographic blinding
  - *Quite heavy from a cryptographic standpoint*



PHI

Path-Hidden Lightweight Anonymity Protocol at Network Layer

PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer

- Last of a series of lightweight anonymity protocols
  - Hiding the various AS positions to protect against topological attacks
  - Making sure that no AS knows both the source AND the destination addresses (*wrt.* LAP)
  - Can work on top of the typical Internet (*wrt.* Dovetail)
- PHI's contributions:
  1. PHI places nodes' states in a pseudo-random order in packet headers to prevent ASes to determine their place on a path

     ➔ *Topological attacks avoidance*
  2. Use of a back-off path construction method to eliminate the need for the source to fully control the path to destination

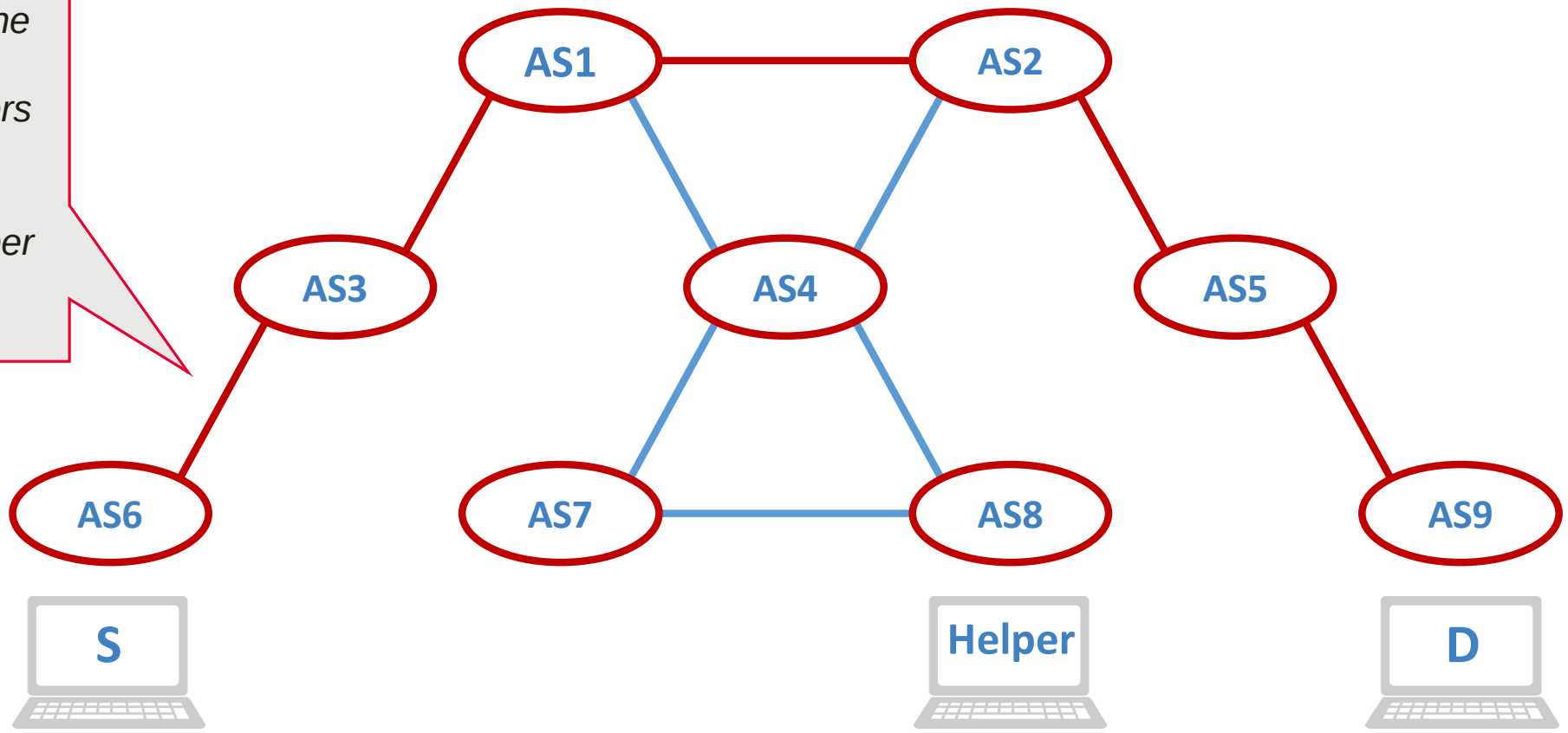
     ➔ *No need for strict source routing primitive*
  3. The payload's encryption is bound to the paths

     ➔ *Session hijacking protection*

Principle

PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer

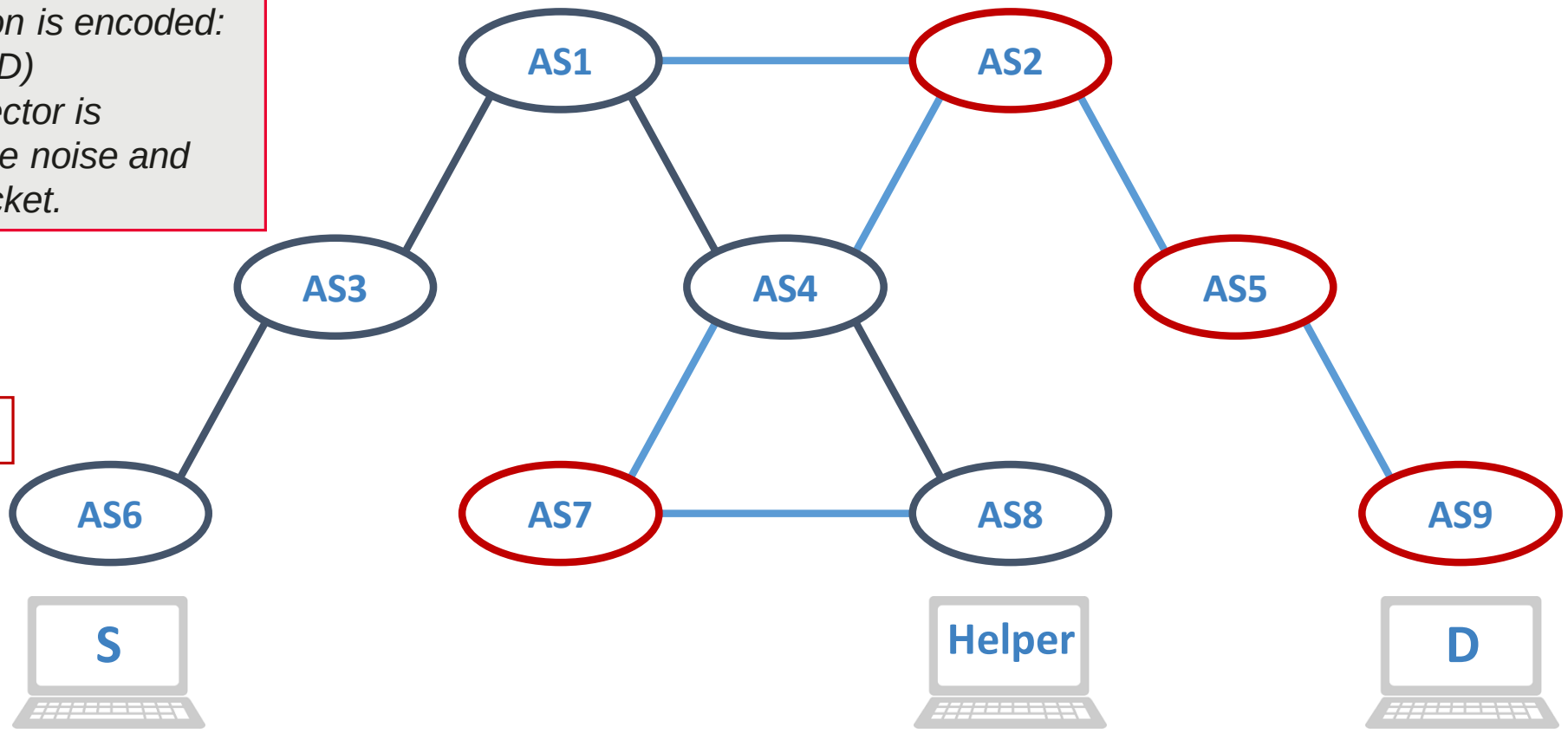
We want to hide the path from S to D from eavesdroppers in AS3, AS1, AS2, AS5 and AS9.
We will use a helper node to build the path from S to D.



Principle

PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer

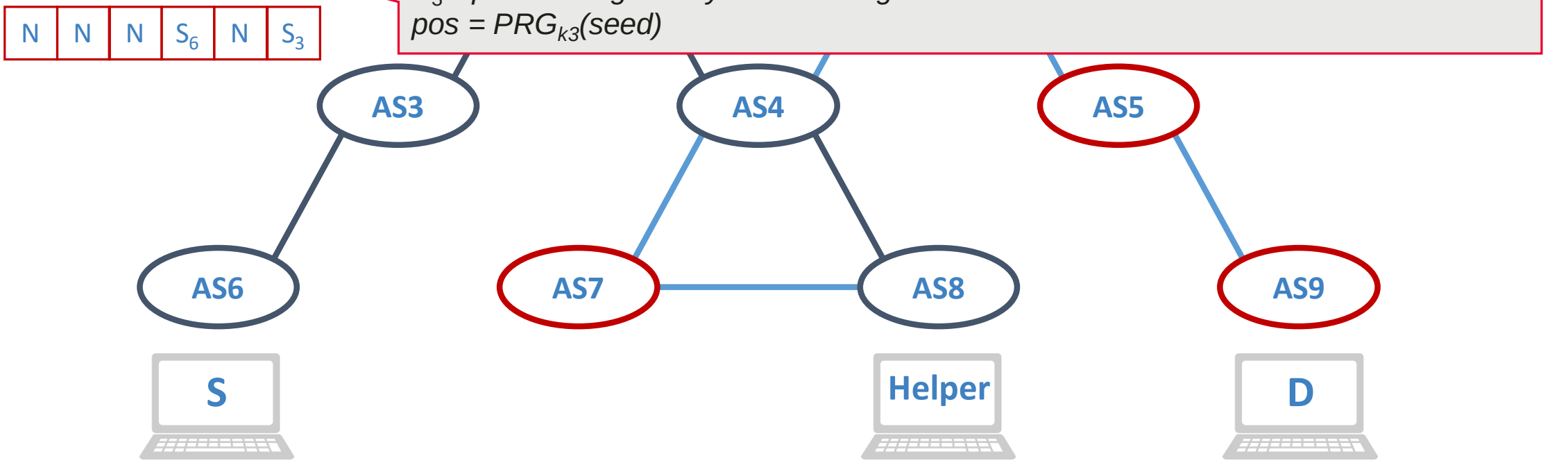
Path creation packet is sent to the Helper.
The final destination is encoded: $m = Enc(pub_K(H); D)$
A path segment vector is initialized with white noise and included in the packet.



Principle

PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer

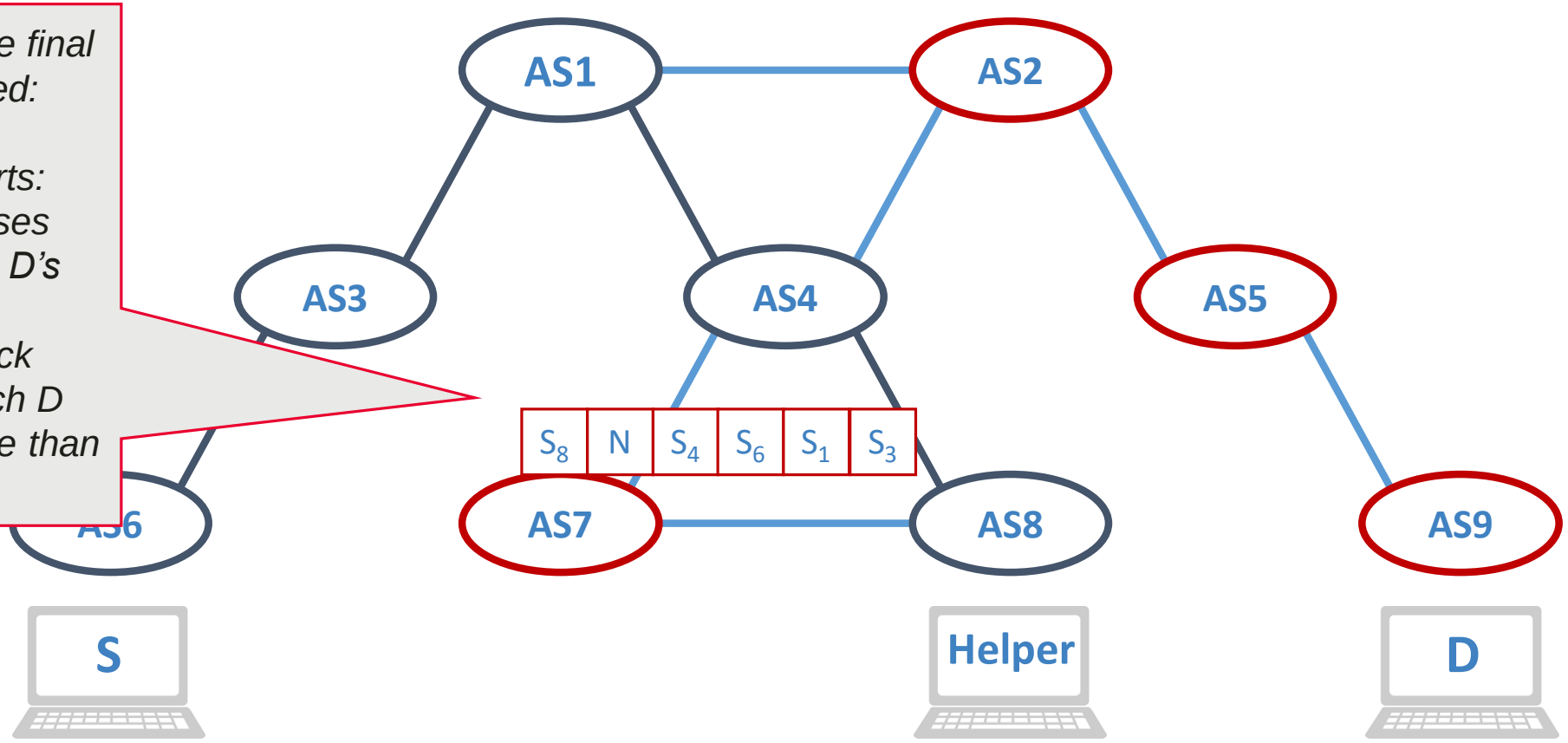
At each node a segment is encoded, and placed in a random position in the path vector.
S3 is computed this way:
$X_3 = Enc_{k3}(Ingress \rightarrow Egress \,||\, posprev \,||\, flags)$
$M_3 = MAC_{k3}(X3||M6)$
$S_3 = E_3 \,||\, M_3$
S_3's position is given by the following formula:
$pos = PRG_{k3}(seed)$



Principle

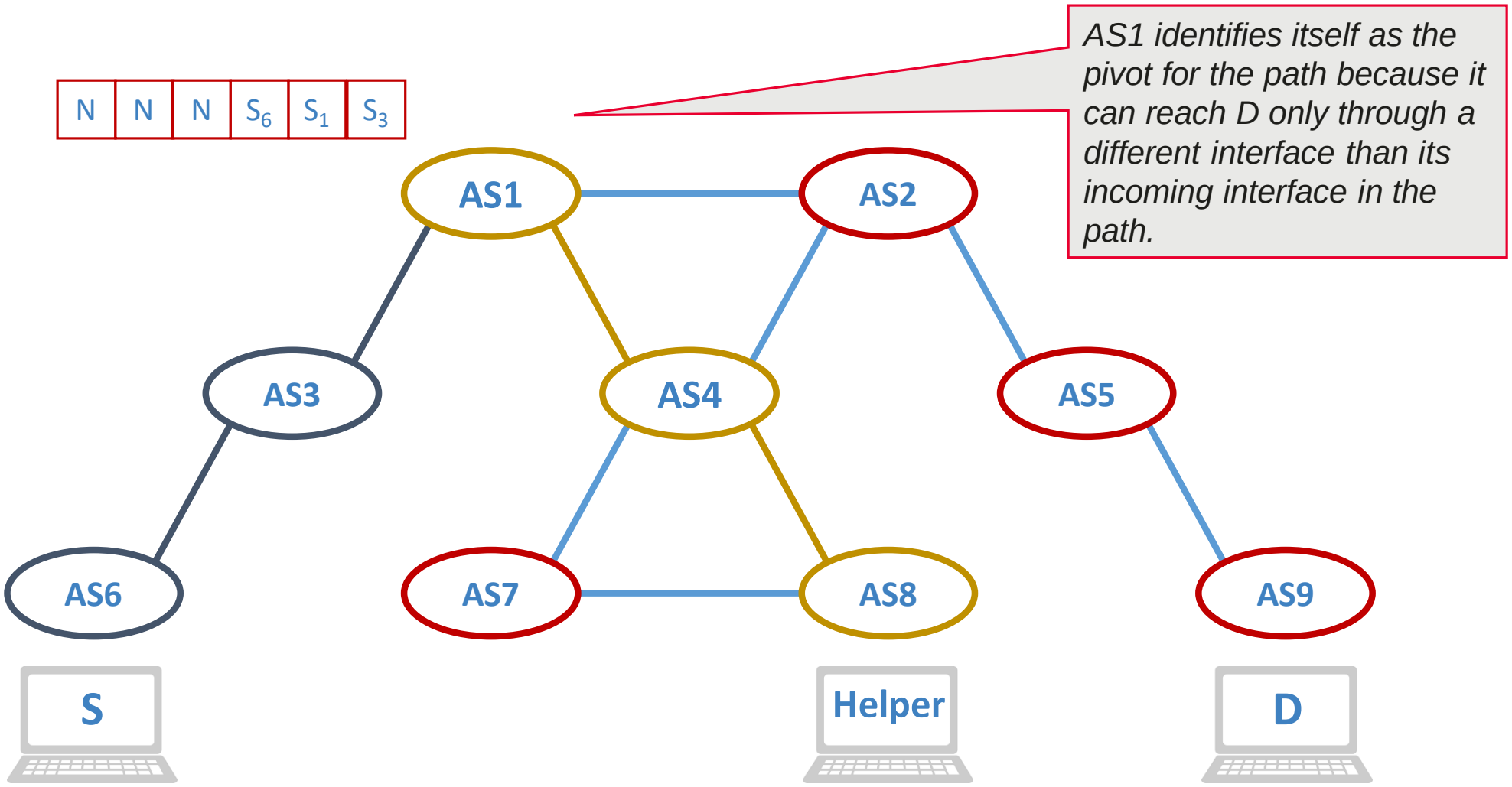
PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer

At the helper node, the final destination is decrypted:
D = Dec(prvKH; m)
A backoff process starts: the helper node reverses the path and provides D's address in clear text.
ASes on the path check whether they can reach D using another interface than the ingress interface.



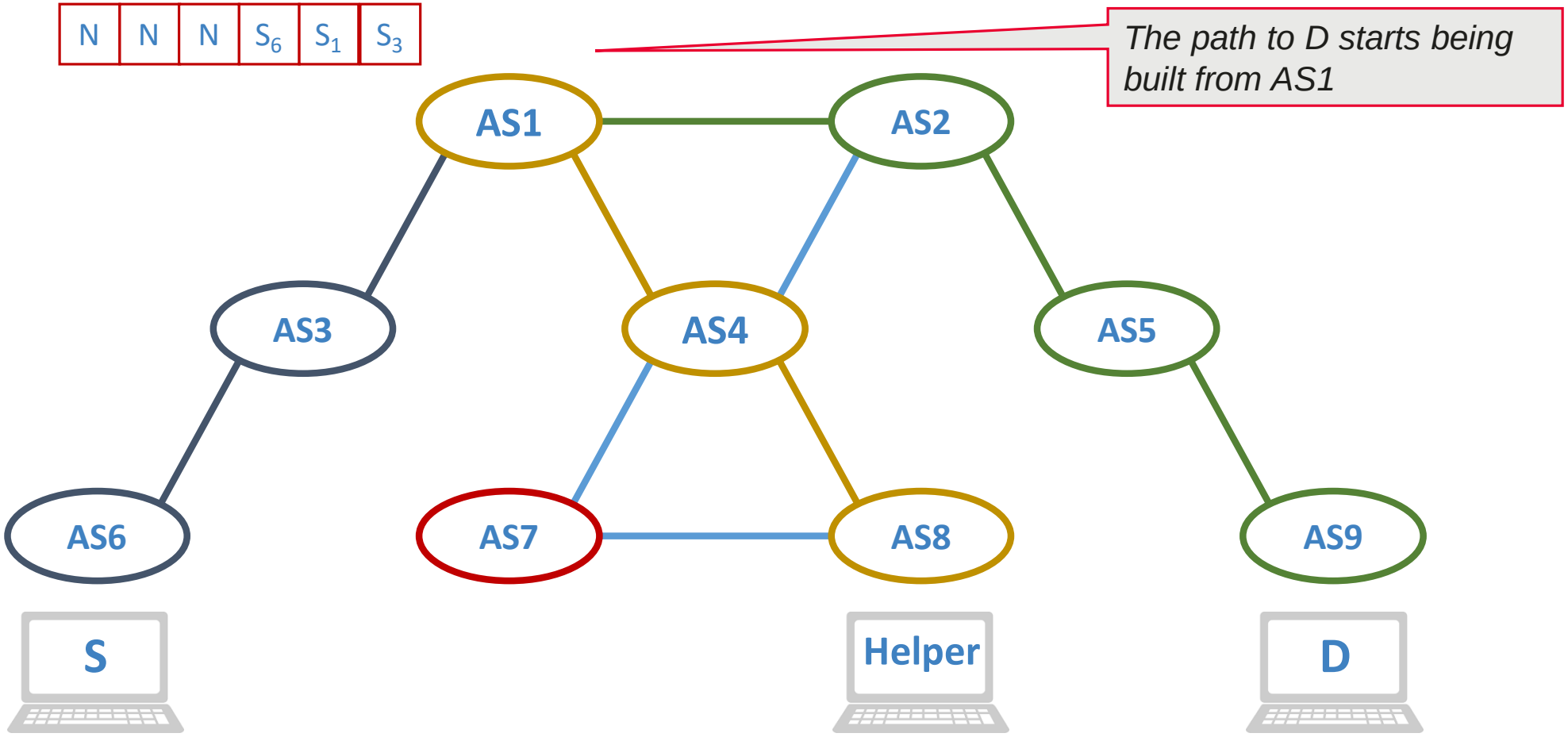
Principle

PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer



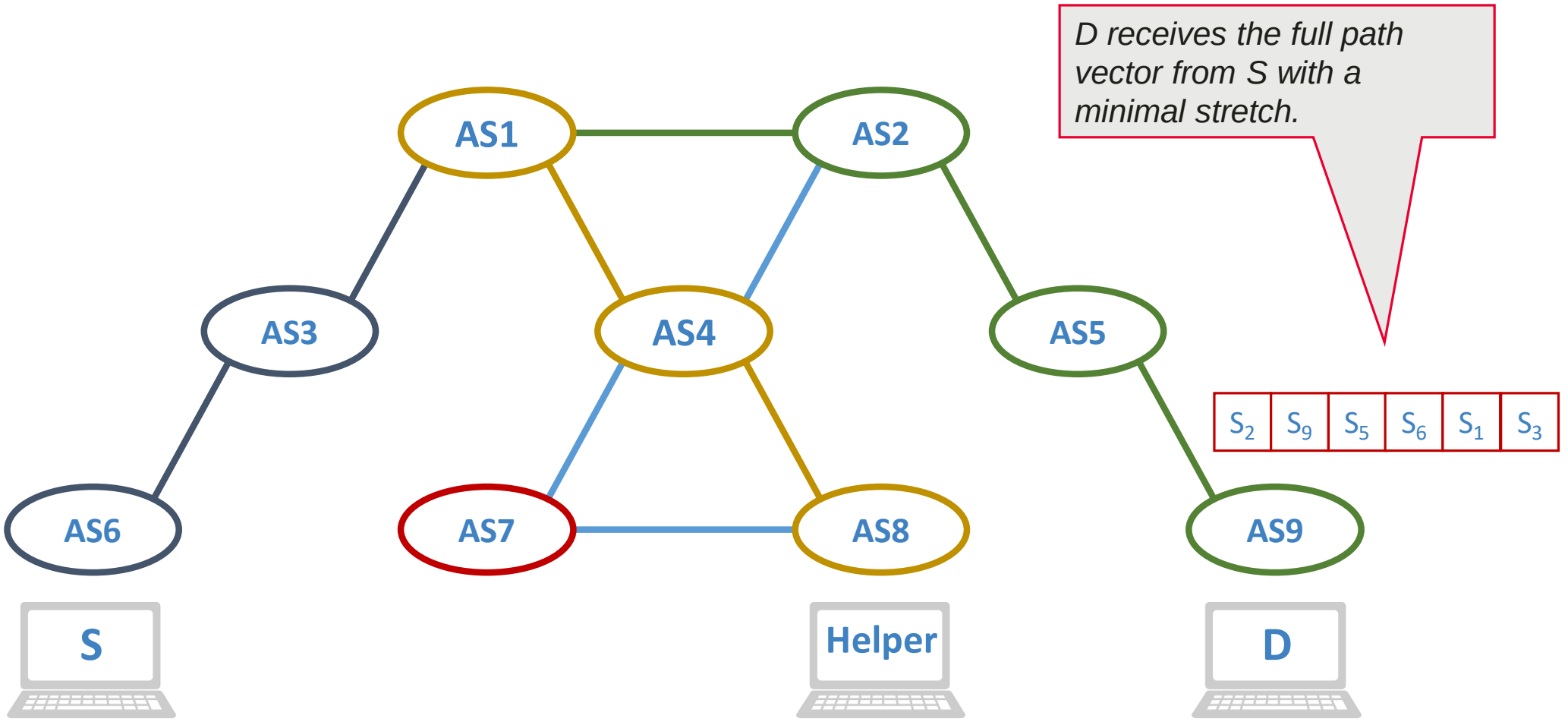
Principle

PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer



Principle

PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer



Pros and Cons

PHI: Path-Hidden Lightweight Anonymity Protocol at Network Layer

- Advantages
  - Protection against naïve topological attacks from randomization procedure
  - Source and destination identities are not revealed at the same time to a given AS.
  - No need to enforce path from the source
- Limitations:
  - Randomization procedure is imperfect and subject to collisions
    - ➔ The path vector's size needs to be 3 times larger than the largest path
    - ➔ Even with this countermeasure, several packets need to be sent to be sure the path is well established (The authors advise 3 concurrent tentatives)
  - The scheme is still vulnerable against elaborated topological attacks
    - Information about the distance to D can be learnt by an on-path AS by comparing the Path setup header with the data header
    - An on-path AS can modify the path segments to learn about its position in the path

Sphinx

A Compact and Provably Secure Mix Format

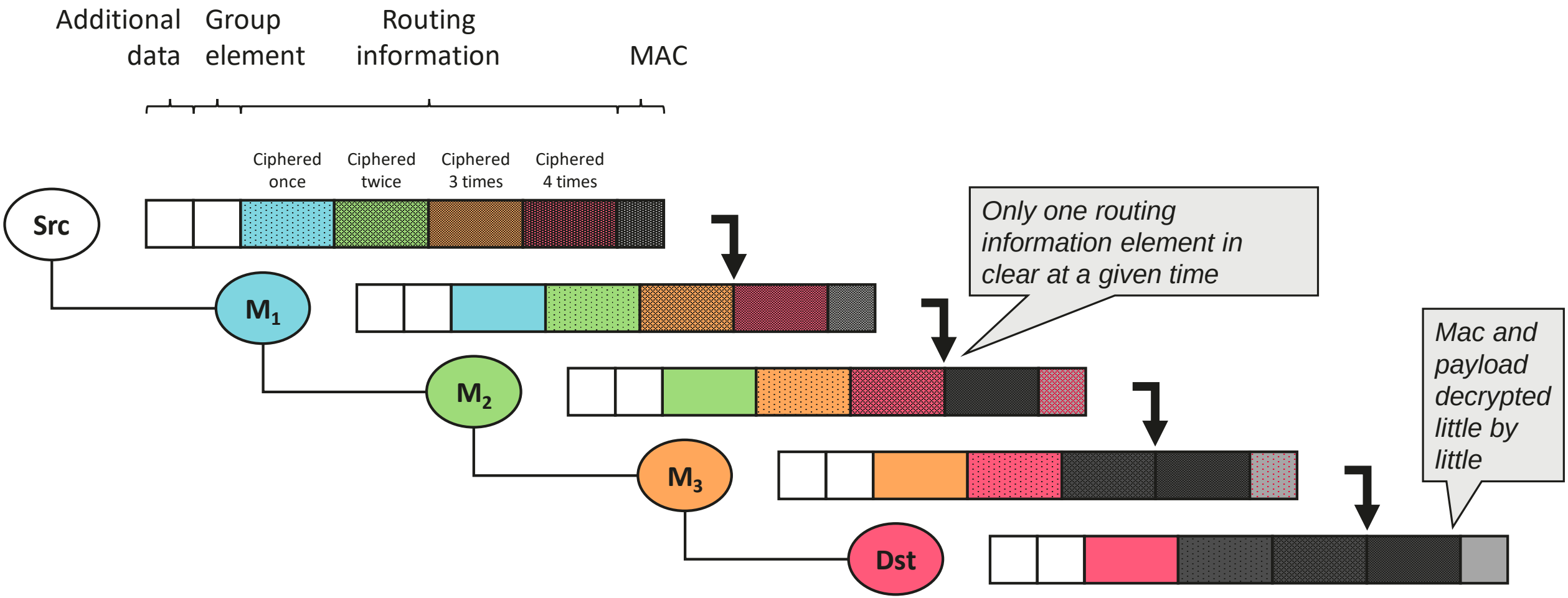
Sphinx: A Compact and Provably Secure Mix Format
Overview

- Sphinx [1] is a major Mix network project
  - Hard-to-trace communications
  - Use of a chain of proxy servers known as mixes which take in messages from multiple senders, shuffle them, and send them back out in random order to the next destination
  - ➔ *Break the link between the source of the request and the destination*
  - ➔ *Hard for eavesdroppers to trace end-to-end communications.*
  - ➔ *No trust in a single relay point needed*
- Interesting Sphinx properties
  - Provably secure format: Sphinx's anonymity properties are ensured as soon as the cryptographic primitives used by Sphinx are secure.
  - Quite strong attack resistance despite 10 years of efforts (1 attack published in 2020 [2], hard to put in place).
  - Projects such as HORNET or TARANET have shown that the untraceability granted by Sphinx is necessary to protect against a state-level passive observer using several vantage points in the network.

[1] Danezis, George, and Ian Goldberg. "*Sphinx: A compact and provably secure mix format*." 2009 30th IEEE Symposium on Security and Privacy. IEEE, 2009.
[2] Kuhn, Christiane, Martin Beck, and Thorsten Strufe. "*Breaking and (partially) fixing provably secure onion routing*." 2020 IEEE Symposium on Security and Privacy (SP). IEEE, 2020.

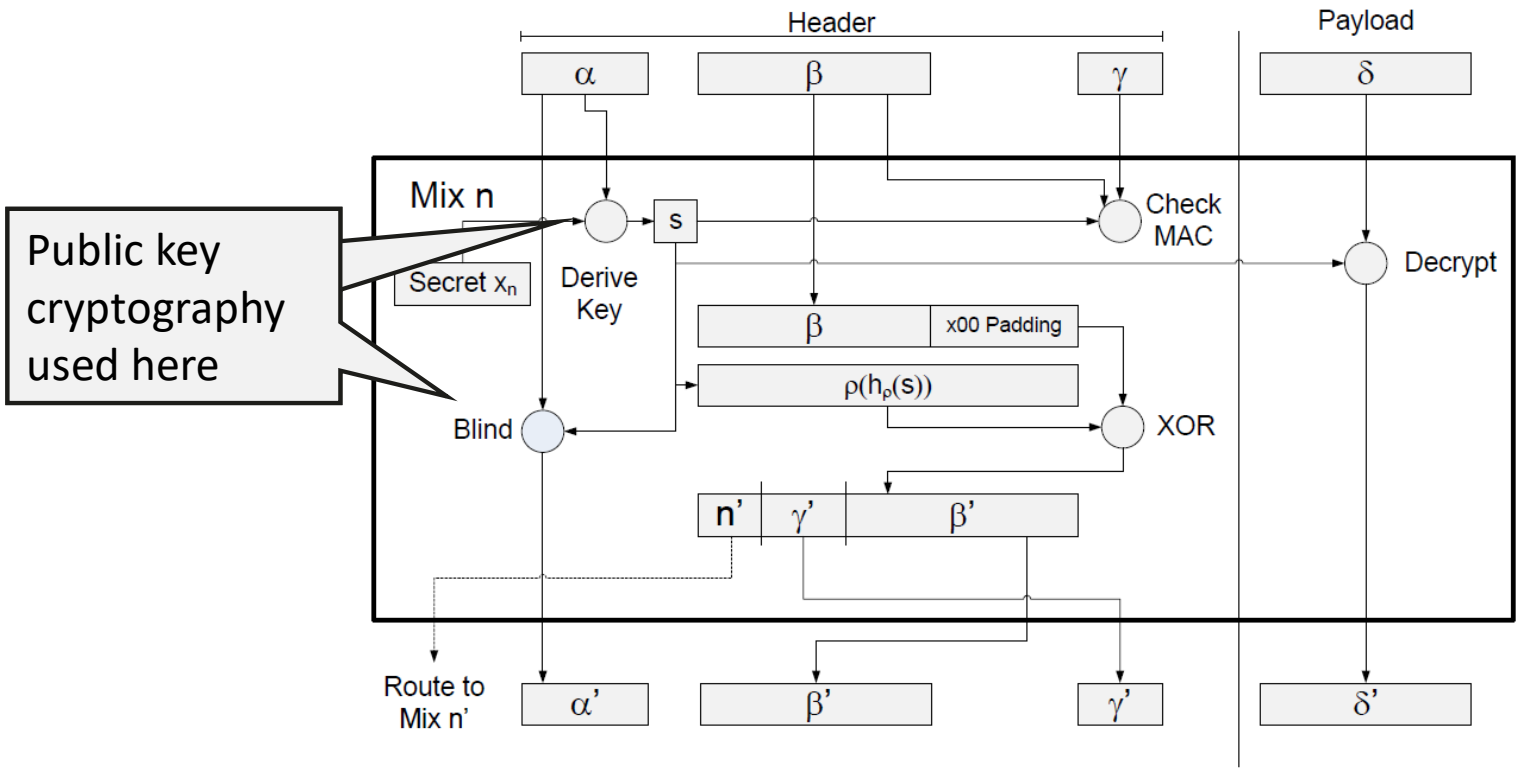
The Sphinx packet header processing

Structure



Cryptographic overhead in Sphinx

- Long setup at the source node to compute key material ➔ Heavy public key cryptography usage
- At intermediate nodes, 2 public key cryptography operations are delaying packet processing a lot
- Several symmetric key cryptography operations are involved in packet relaying

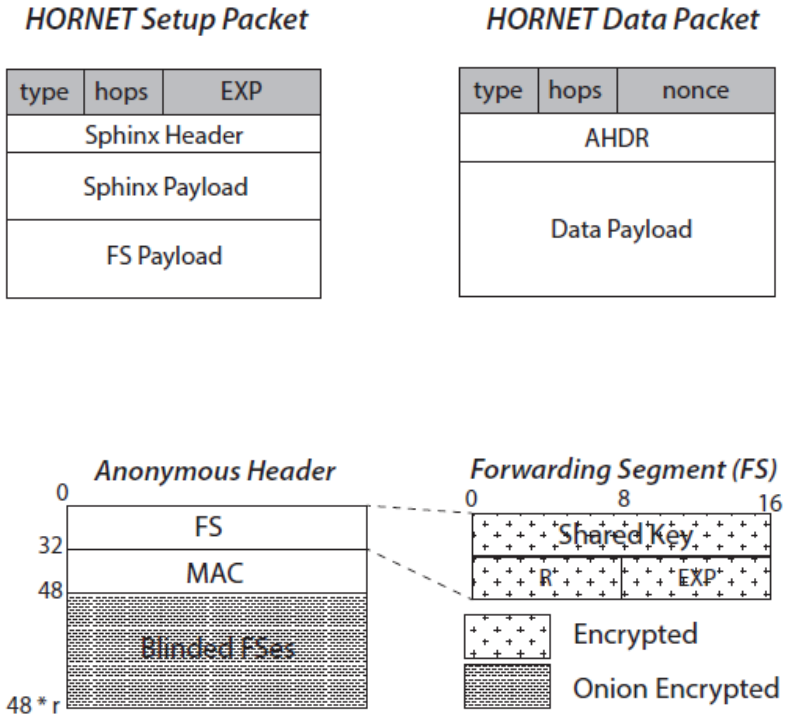


HORNET

High-speed Onion Routing at the Network Layer

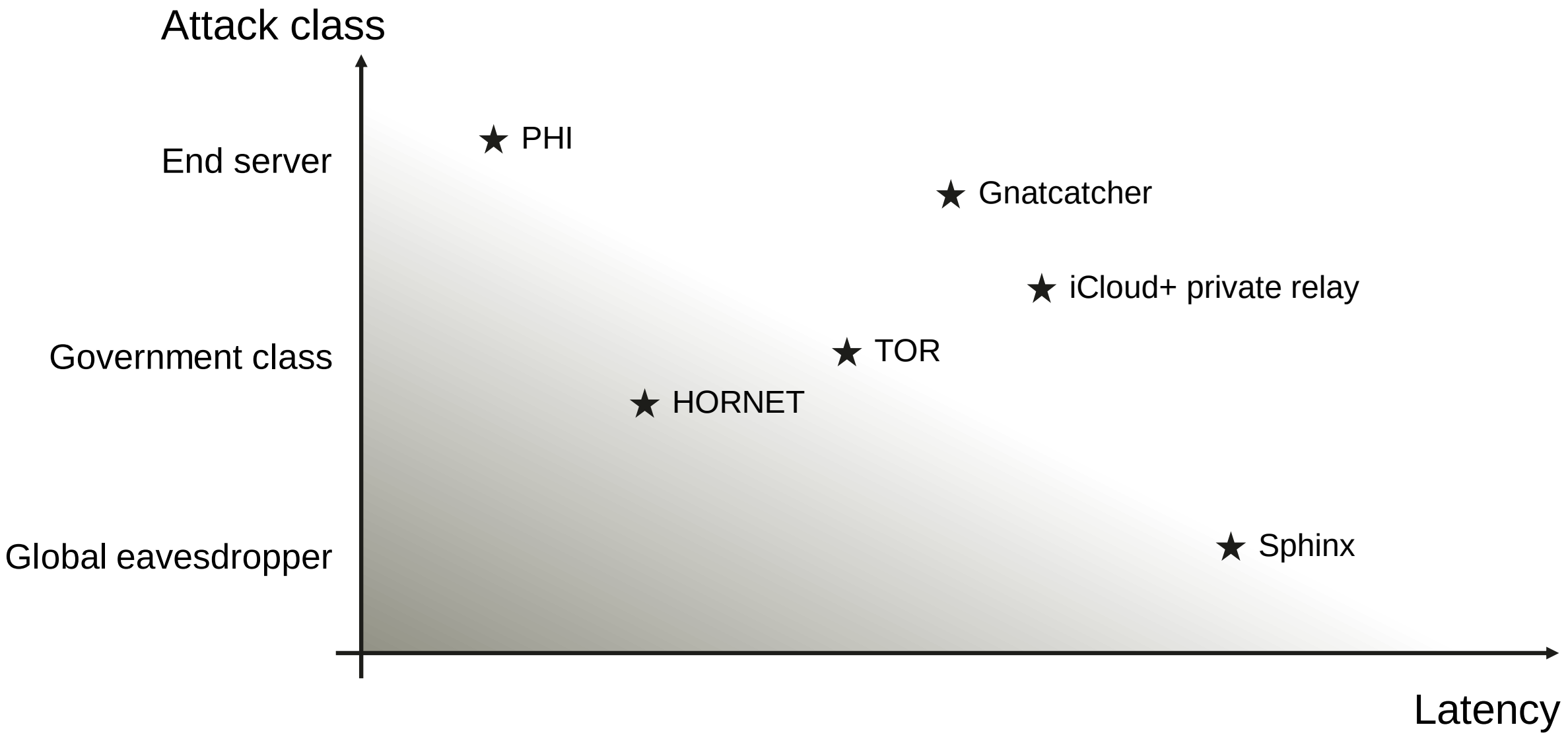
HORNET: High-speed Onion Routing at the Network Layer

- Project aiming at addressing the high computational load of the Sphinx approach to use it at the network layer
- Source routing approach
- 2 steps process:
  1. Path setup phase:
     - The source is using two Sphinx-like packets to collect Forwarding Segments (FS) from intermediate nodes on the path to a destination
     - A Forwarding Segment contains a routing segment, a shared secret key and an expiration time encrypted with a key known only by each intermediate node
  2. Data transmission phase:
     - The source uses the Forwarding Segments to build a source routed packet
     - Only symmetric key encryption is used ➔ ***Performance***



Findings from the state of the art

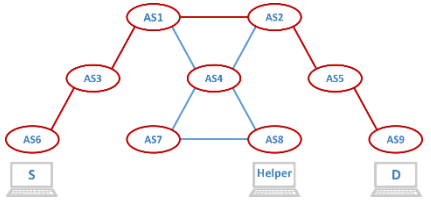
Positionning SoA projects on a map



Unlinking source and destination

From using a relay node to a source-routed approach

- One of the most used method to provide privacy for a network path is to use third party nodes and encryption of source / destination addresses
  - Pros: Simple approach
  - Cons: Require a certain level of trust in the relay node



➔ Trying to avoid using this approach to adopt an approach in which the trust required from potential relay nodes is ***limited***

➔ Use of a ***path built at the source***:

- Source addresses can be safely removed, replaced by:
  - The use of a return block, *i.e.* a ciphered pointer to a mix circuit
  - Making the path a loop including a return path to the source
- To improve the anonymity subset, we should prevent a node on the path to be able to determine the destination, the path length and its position in the path
- To prevent attacks based on an observation of the inter-AS topology, we can introduce routing policy violations by using relay nodes to avoid attacks based on AS ranking and relationship determination

Next steps ?

- Edit a draft from the presentation to compile a state of the art on privacy at the network layer / IP address privacy?
- Most deployed approaches to provide IP address privacy are using trusted or semi-trusted third parties
  ➔ Would it be interesting to explore the source routing based approach to IP address privacy?

Thank you!