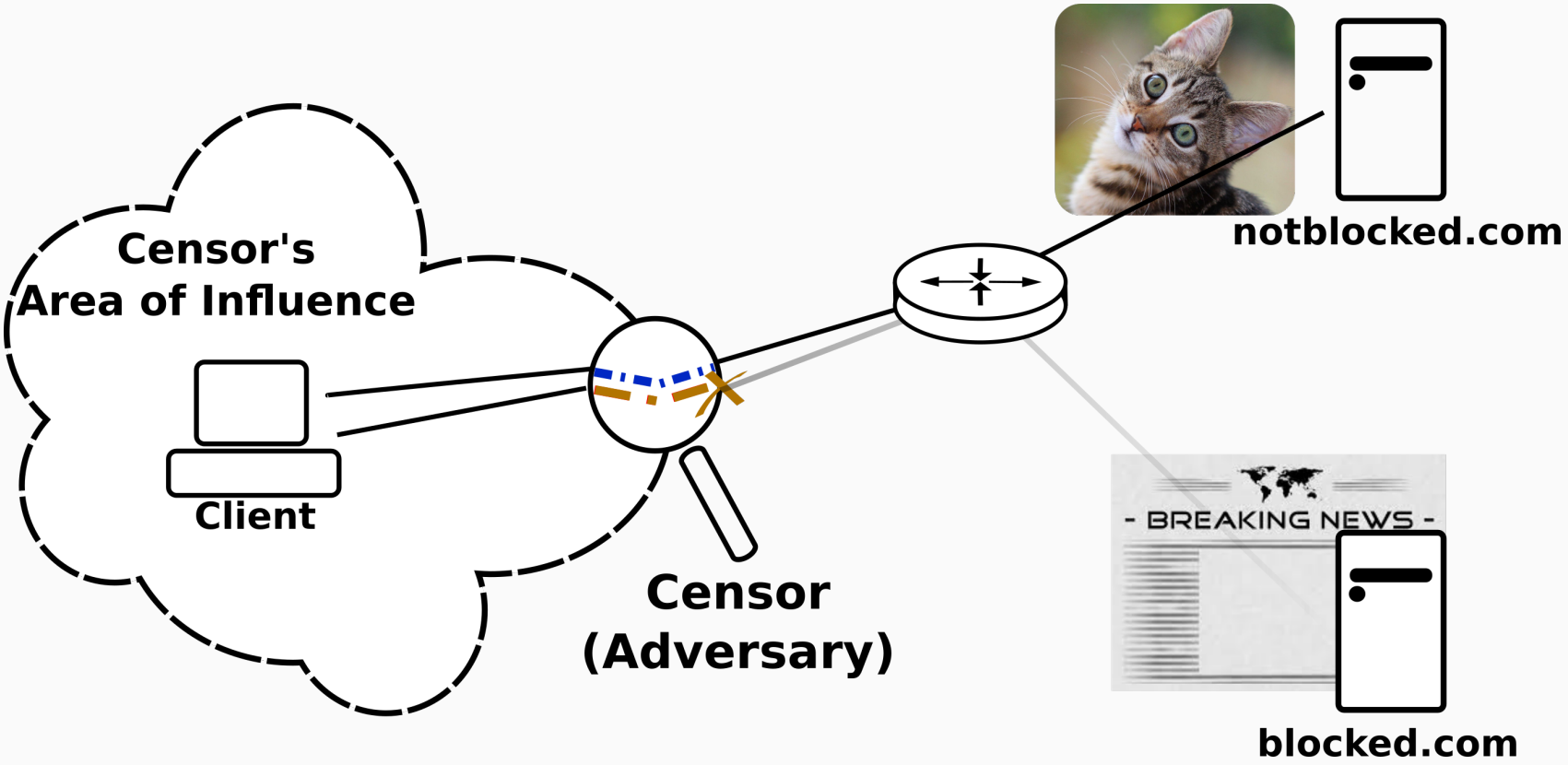
Perfect Imitation and Secure Asymmetry for Decoy Routing Systems with Slitheen

Cecylia Bocovich **Ian Goldberg**

20 June 2017
EPFL Summer Research Institute

Censorship



Censors may monitor, alter or block traffic that enters or leaves their area of influence.

Censorship Strategies

Censorship measurement studies in Iran [Aryan et al.], Pakistan [Nabi et al.], and China [Winter and Lindskog] show the following techniques:

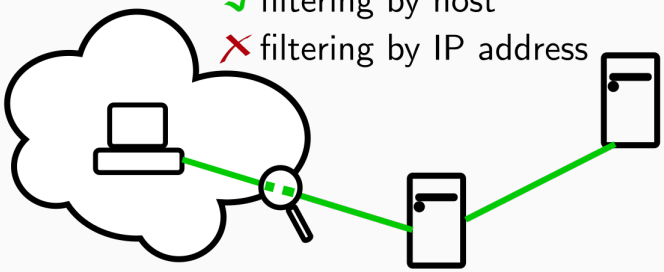
- Filtering by IP address
- Filtering by hostname
- Protocol-specific throttling
- URL keyword filtering
- Active probing
- Application-layer DPI

Censorship Circumvention

Simple Proxies

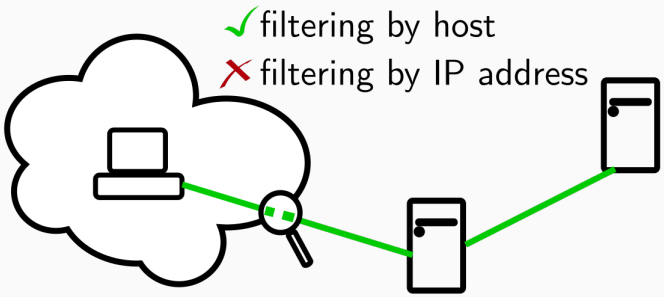
✓ filtering by host

✗ filtering by IP address

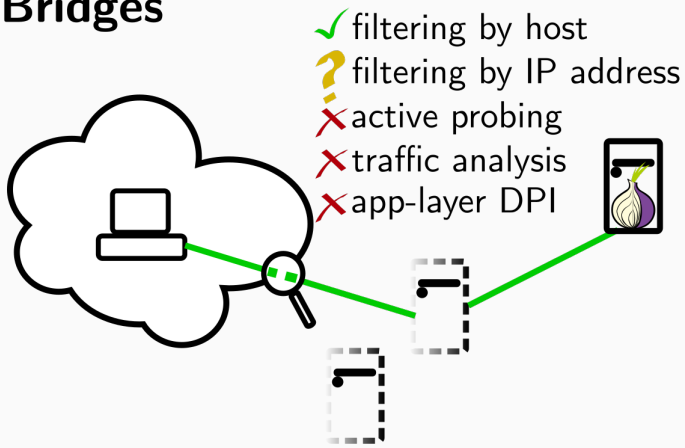


Censorship Circumvention

Simple Proxies



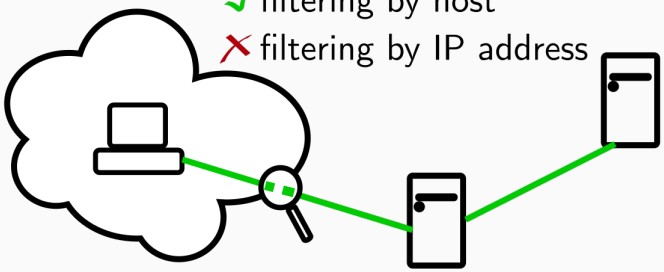
Tor Bridges



Censorship Circumvention

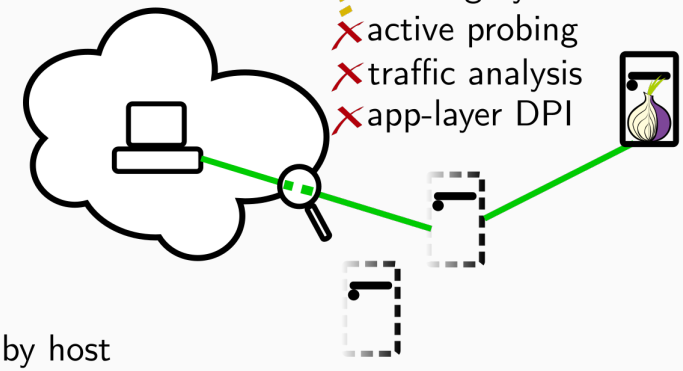
Simple Proxies

✓ filtering by host
✗ filtering by IP address

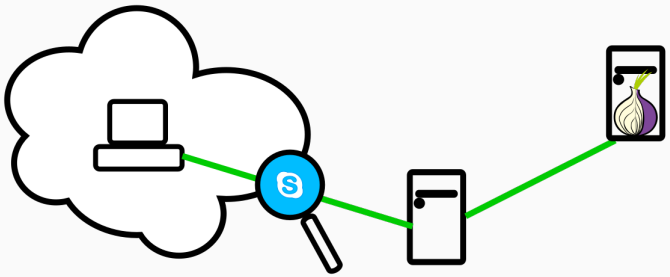


Tor Bridges

✓ filtering by host
? filtering by IP address
✗ active probing
✗ traffic analysis
✗ app-layer DPI



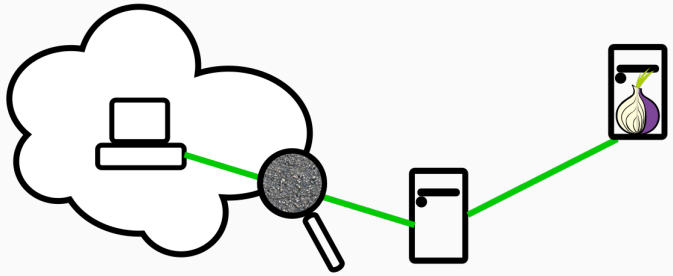
Pluggable Transports



SkypeMorph (Mohajeri Moghaddam et al., 2012)
StegoTorus (Weinberg et al., 2012)
Marionette (Dyer et al., 2015)

✓ filtering by host
✓ filtering by IP address
? active probing
? traffic analysis
? app-layer DPI

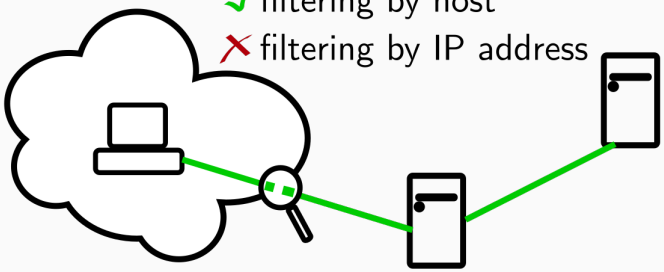
ScrambleSuit (Winter et al., 2012)
Obfsproxy (Dingledine, 2012)



Censorship Circumvention

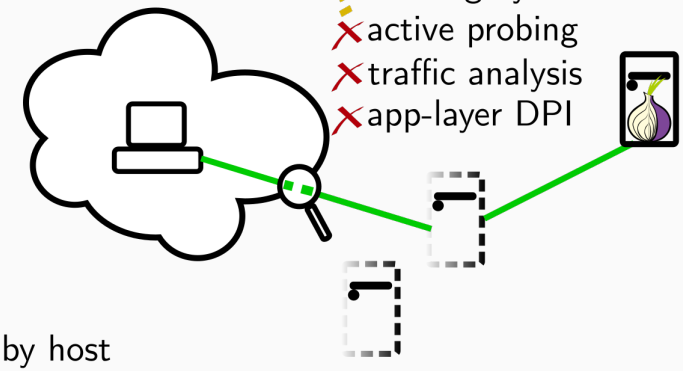
Simple Proxies

✓ filtering by host
✗ filtering by IP address

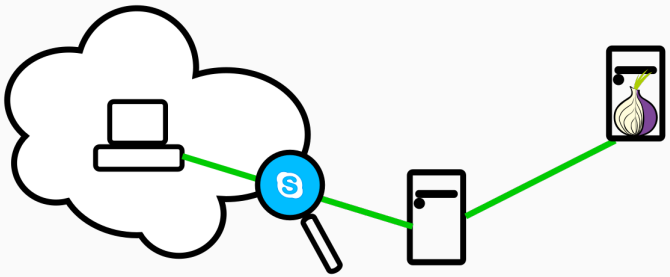


Tor Bridges

✓ filtering by host
? filtering by IP address
✗ active probing
✗ traffic analysis
✗ app-layer DPI



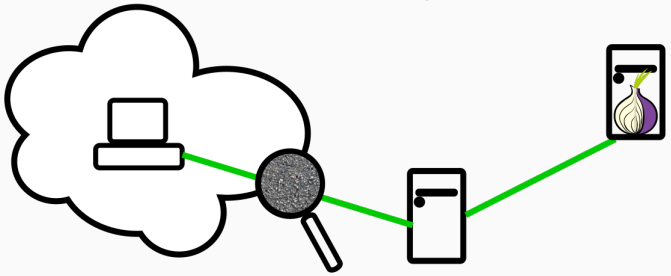
Pluggable Transports



SkypeMorph (Mohajeri Moghaddam et al., 2012)
StegoTorus (Weinberg et al., 2012)
Marionette (Dyer et al., 2015)

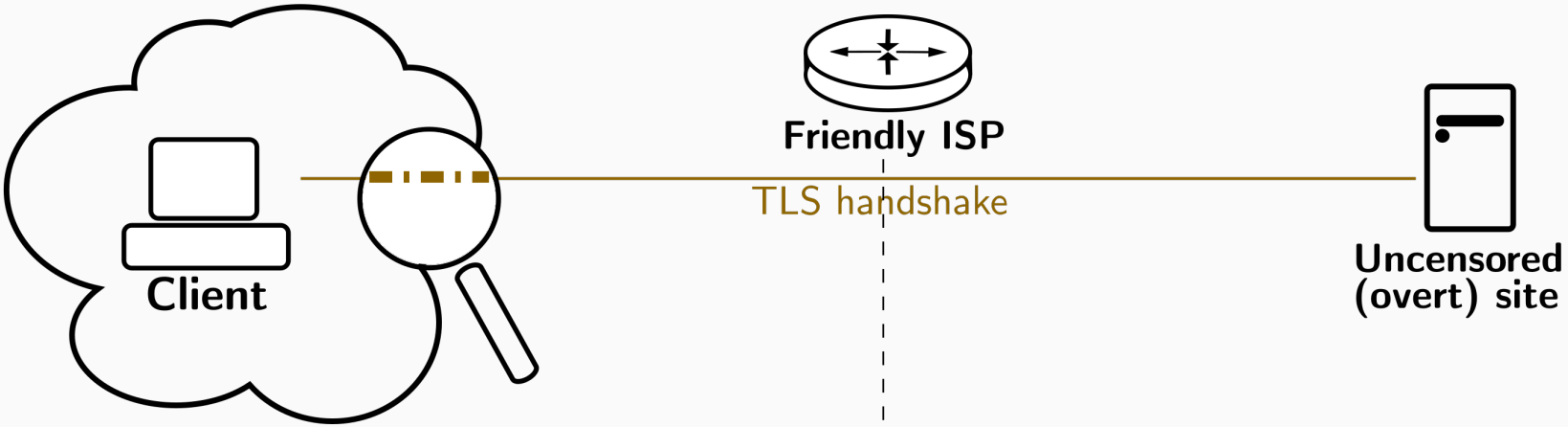
✓ filtering by host
✓ filtering by IP address
? active probing
? traffic analysis
? app-layer DPI

ScrambleSuit (Winter et al., 2012)
Obfsproxy (Dingledine, 2012)



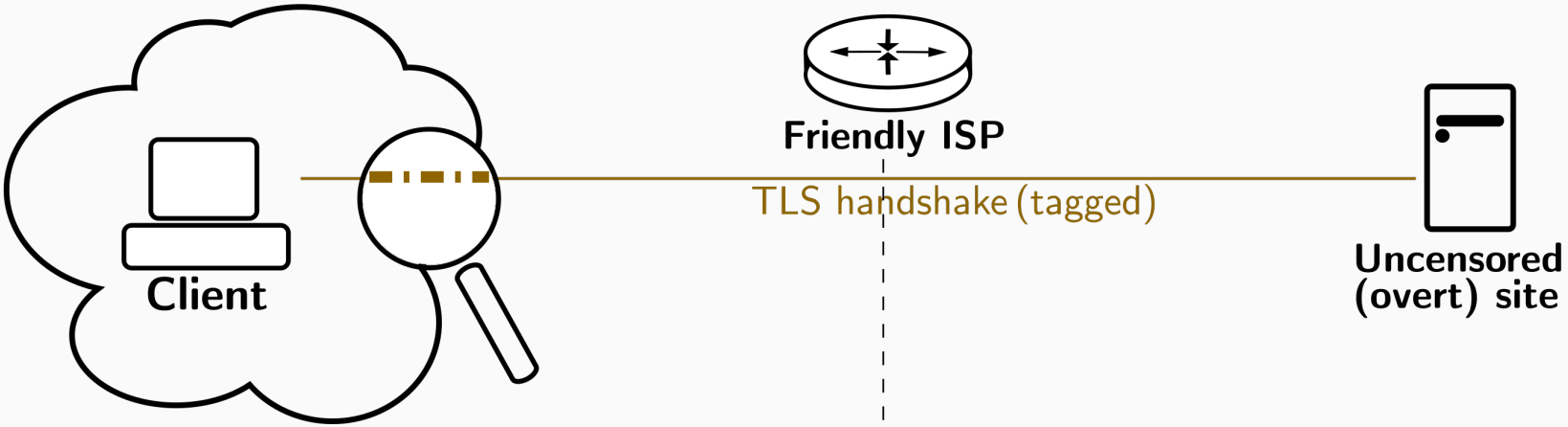
(Houmansadr et al., 2013)

Decoy Routing



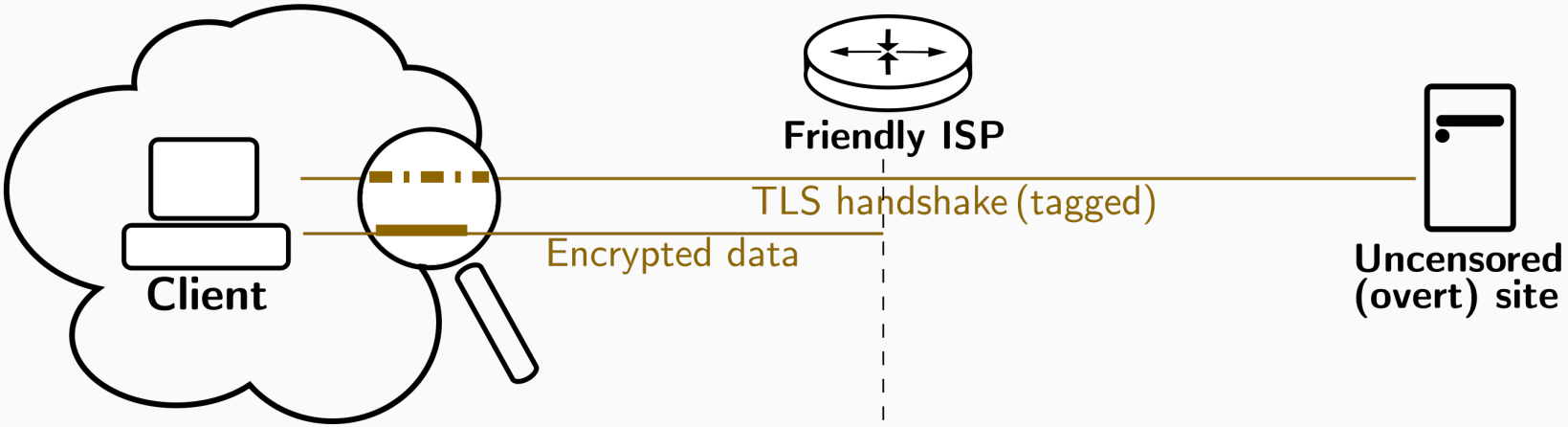
1. Establish TLS connection with overt site

Decoy Routing



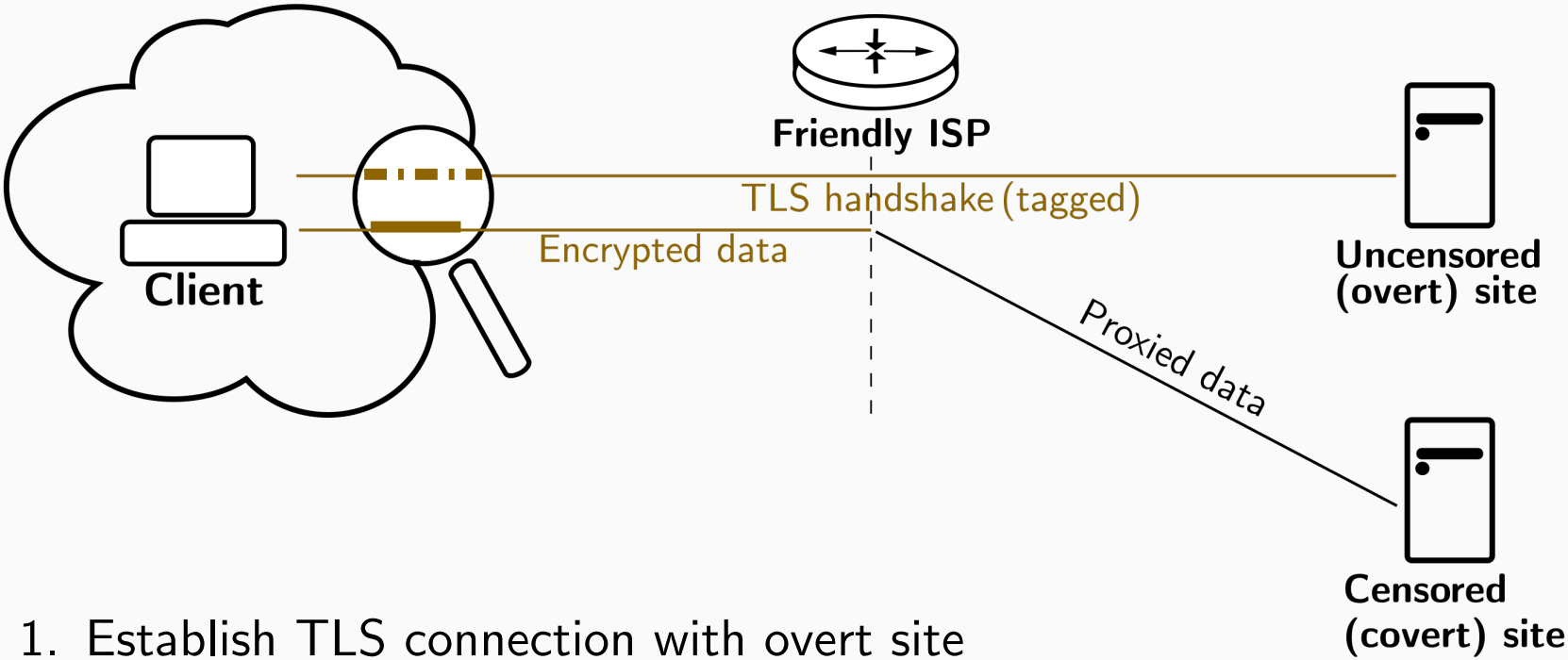
1. Establish TLS connection with overt site
2. Steganographically share TLS master secret with friendly ISP
   (Wustrow et al., 2011) (Houmansadr et al., 2011) (Karlin et al., 2011) (Wustrow et al., 2014) (Ellard et al., 2015)

Decoy Routing



1. Establish TLS connection with overt site
2. Steganographically share TLS master secret with friendly ISP
   (Wustrow et al., 2011) (Houmansadr et al., 2011) (Karlin et al., 2011) (Wustrow et al., 2014) (Ellard et al., 2015)
3. Sever or abandon connection to the overt site

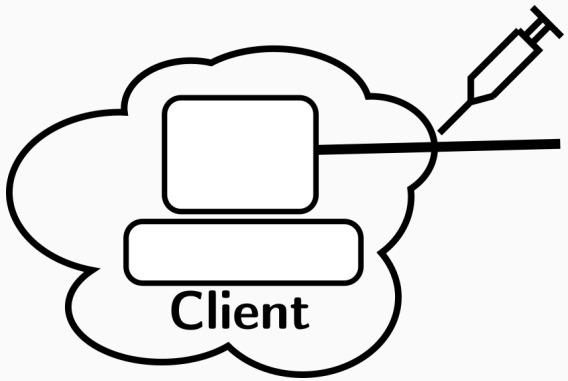
Decoy Routing



1. Establish TLS connection with overt site
2. Steganographically share TLS master secret with friendly ISP
   (Wustrow et al., 2011) (Houmansadr et al., 2011) (Karlin et al., 2011) (Wustrow et al., 2014) (Ellard et al., 2015)
3. Sever or abandon connection to the overt site
4. Proxy information between client and covert site

Attacks on Decoy Routing

(Wustrow et al., 2011)
(Schuchard et al., 2012)

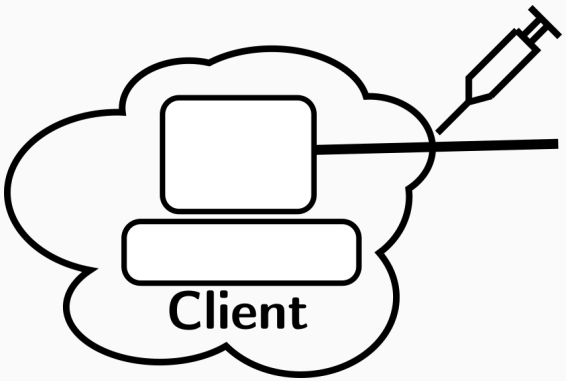


Active Attacks

- Replay attacks
- Man in the middle

Attacks on Decoy Routing

(Wustrow et al., 2011)
(Schuchard et al., 2012)

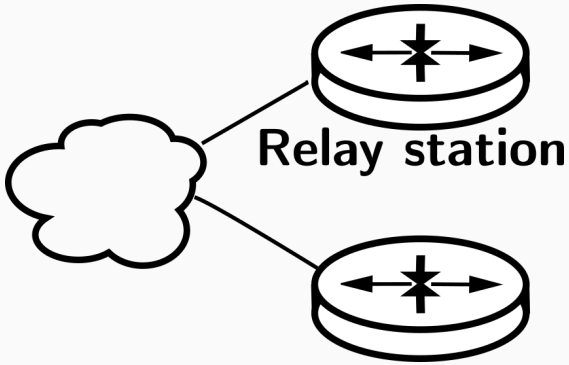


Active Attacks

- Replay attacks
- Man in the middle

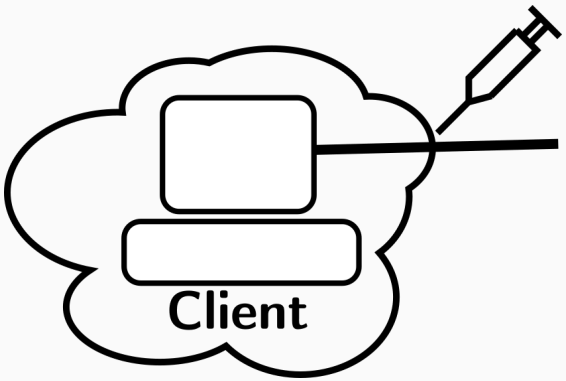
Routing-Based (RAD) Attacks

- TCP replay
- Crazy Ivan



Attacks on Decoy Routing

(Wustrow et al., 2011)
(Schuchard et al., 2012)

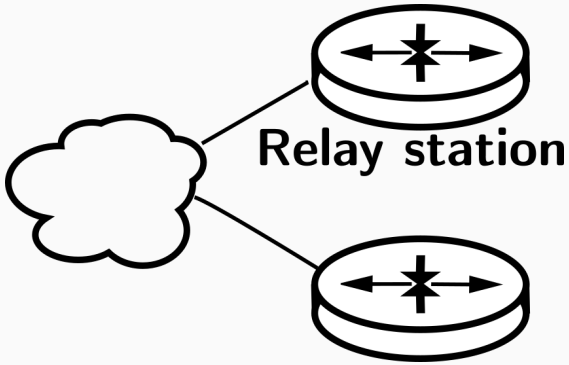


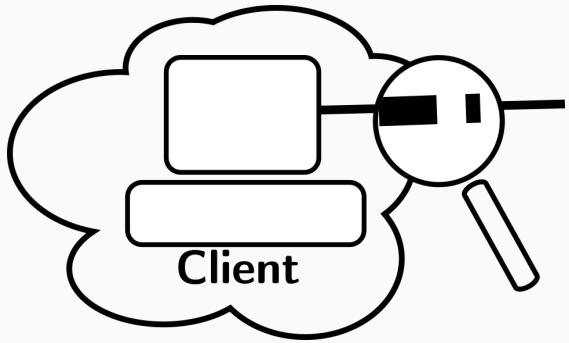
Active Attacks

- Replay attacks
- Man in the middle

Routing-Based (RAD) Attacks

- TCP replay
- Crazy Ivan



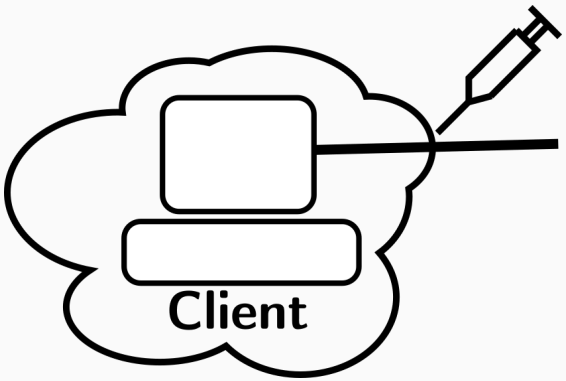


Passive Attacks

- Traffic analysis
- Latency analysis

Attacks on Decoy Routing

(Wustrow et al., 2011)
(Schuchard et al., 2012)

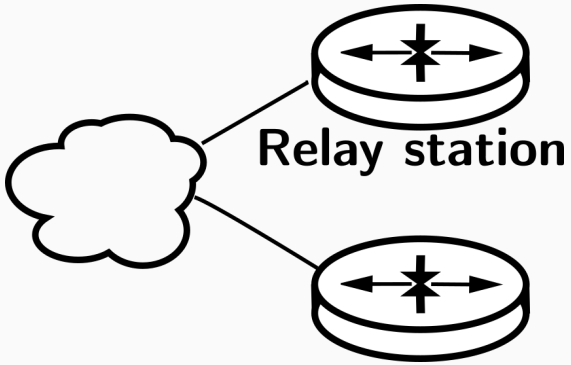


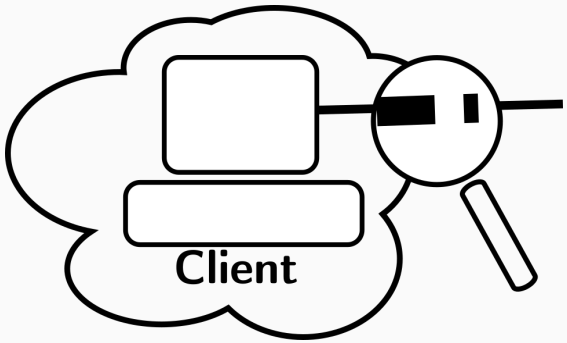
Active Attacks

- Replay attacks
- Man in the middle*

Routing-Based (RAD) Attacks

- TCP replay*
- Crazy Ivan

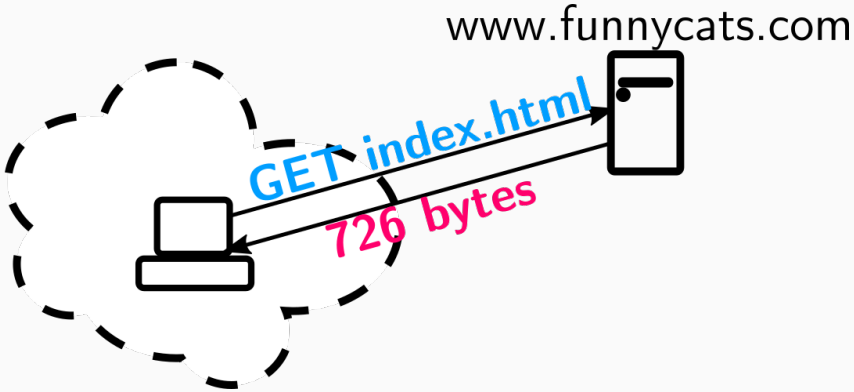




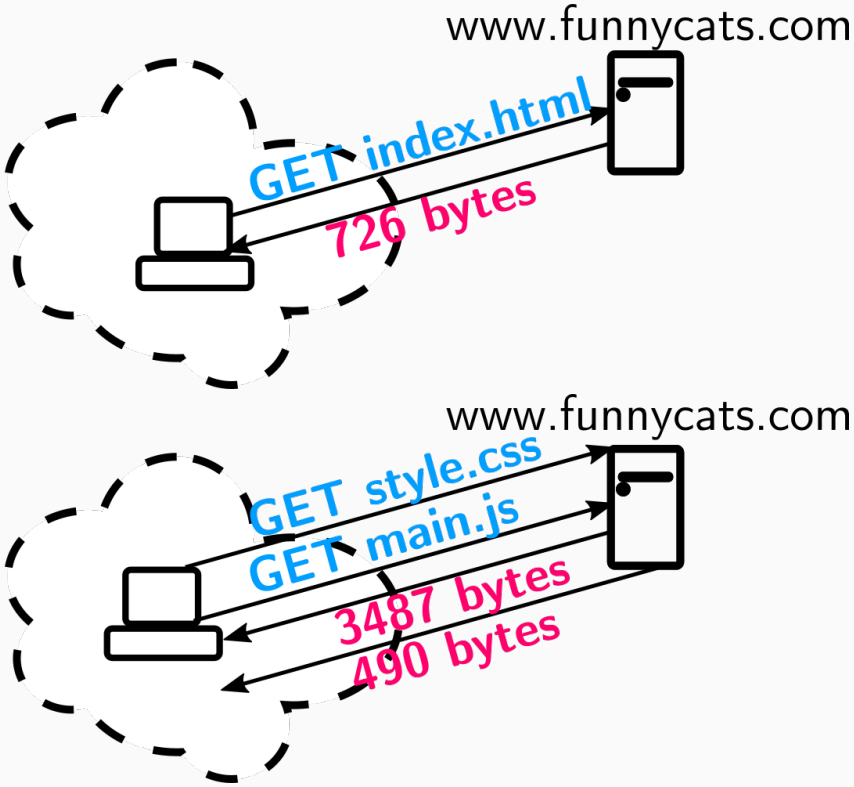
Passive Attacks

- Traffic analysis
- Latency analysis

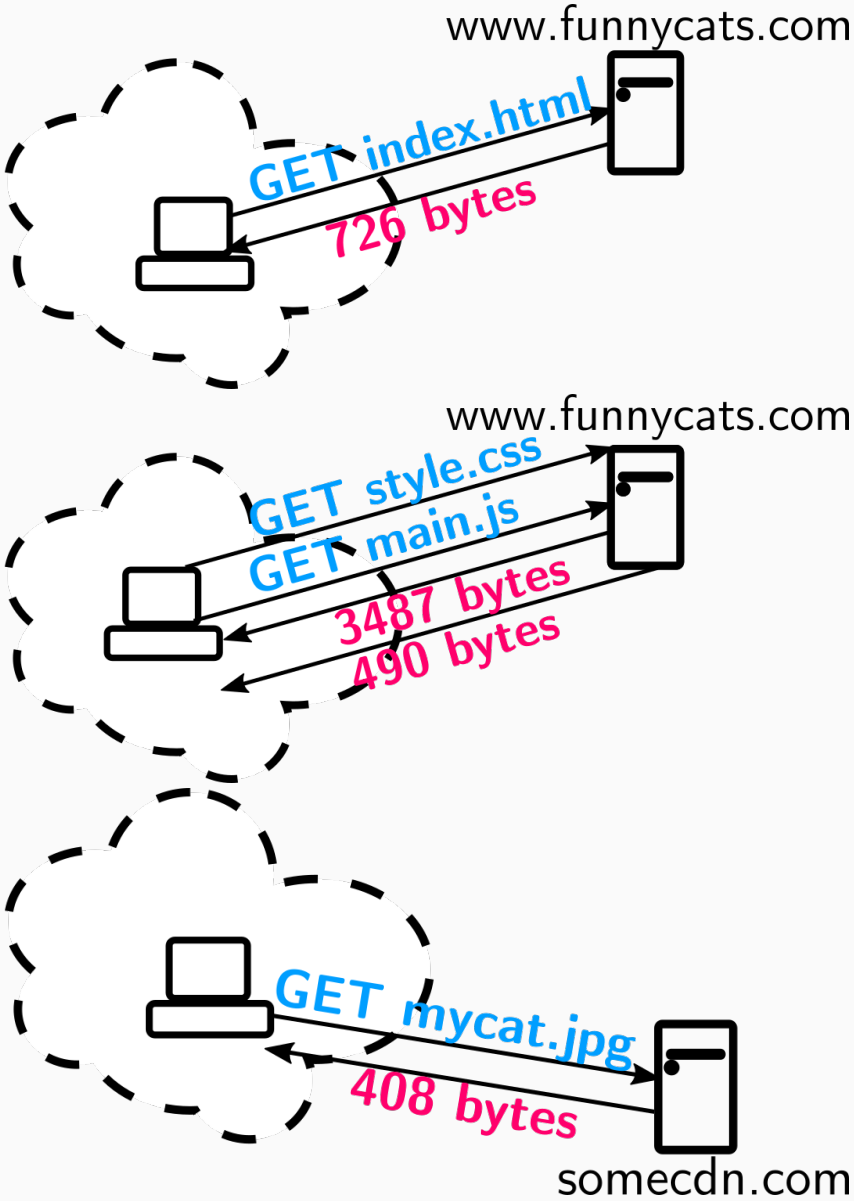
Traffic Analysis



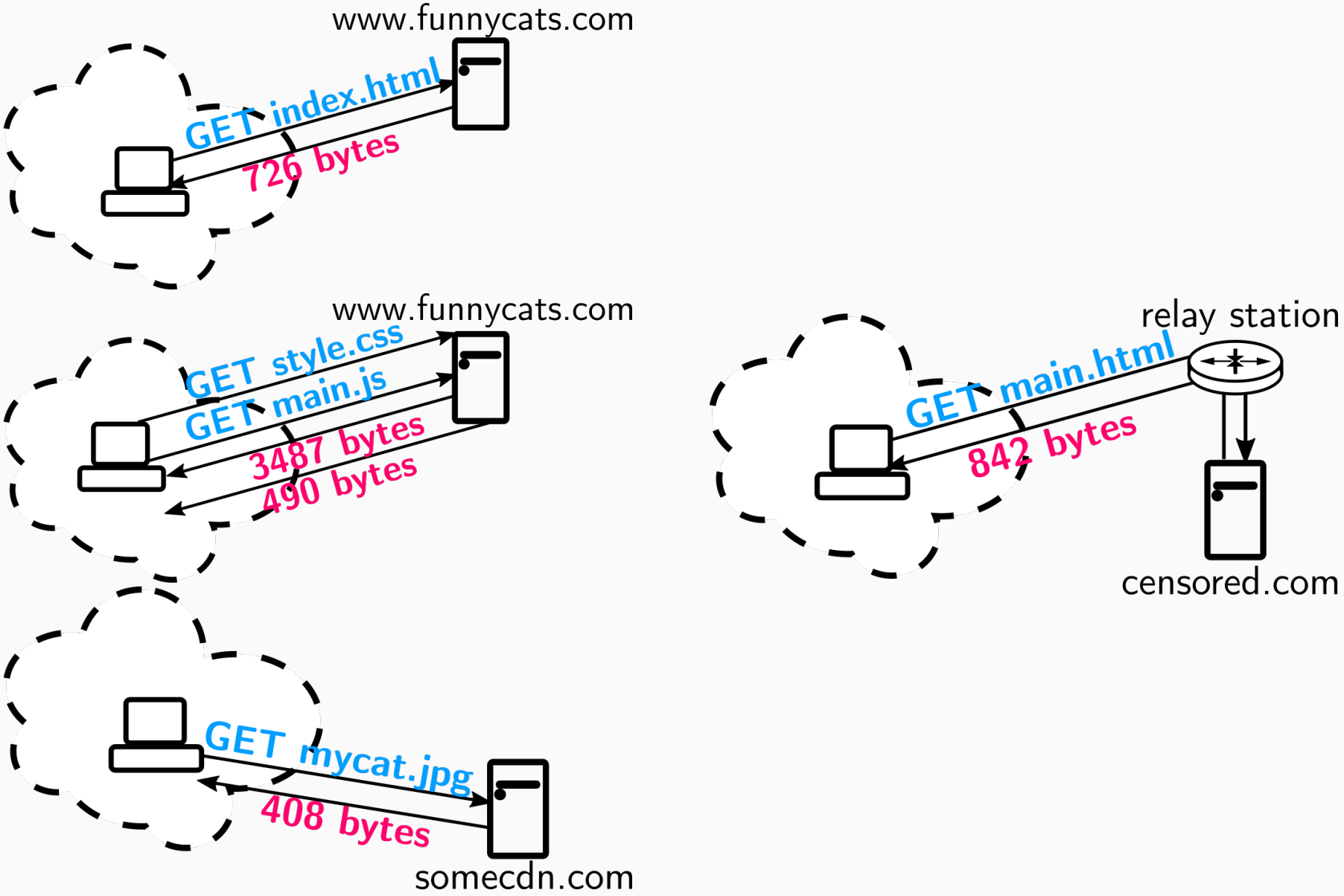
Traffic Analysis



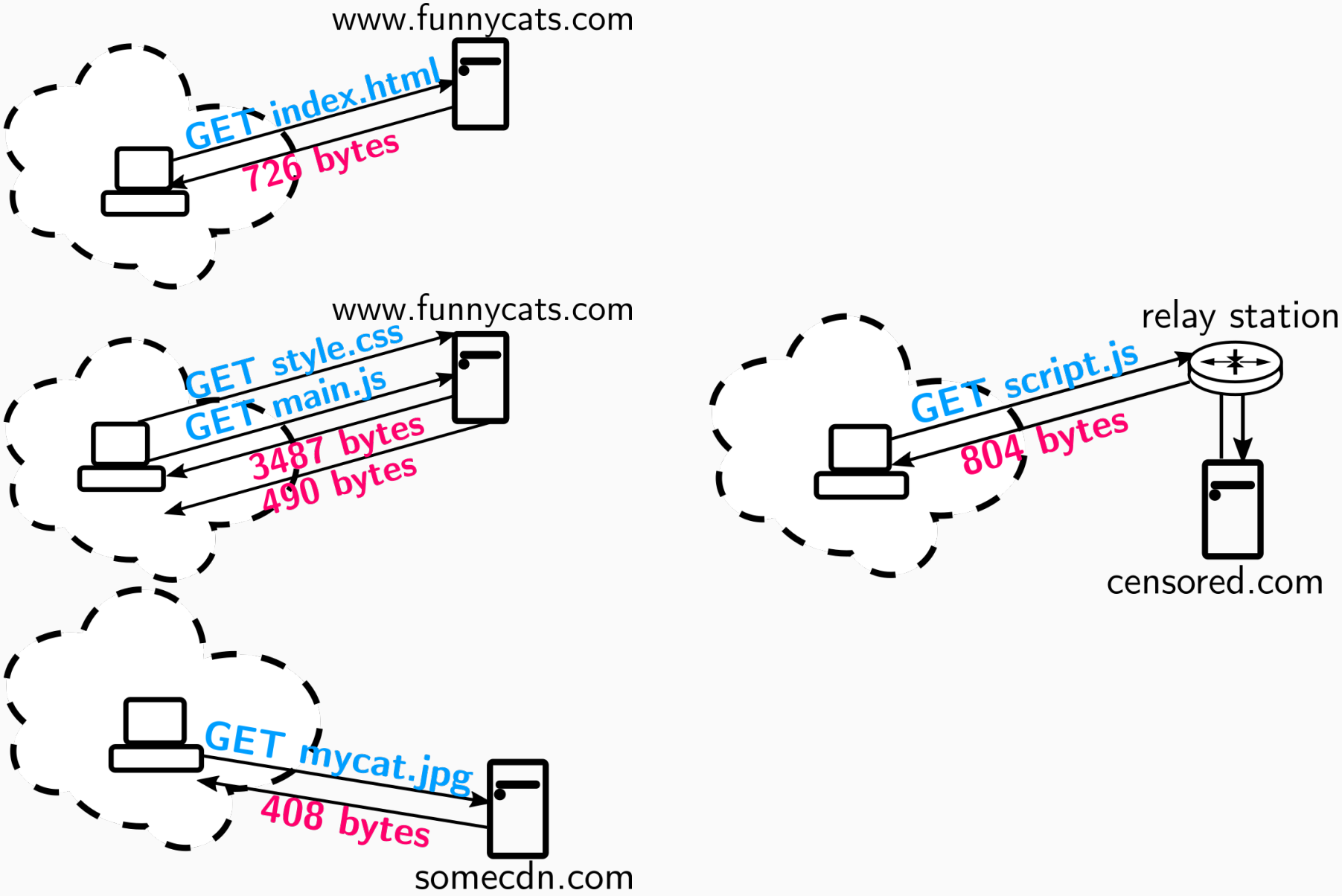
Traffic Analysis



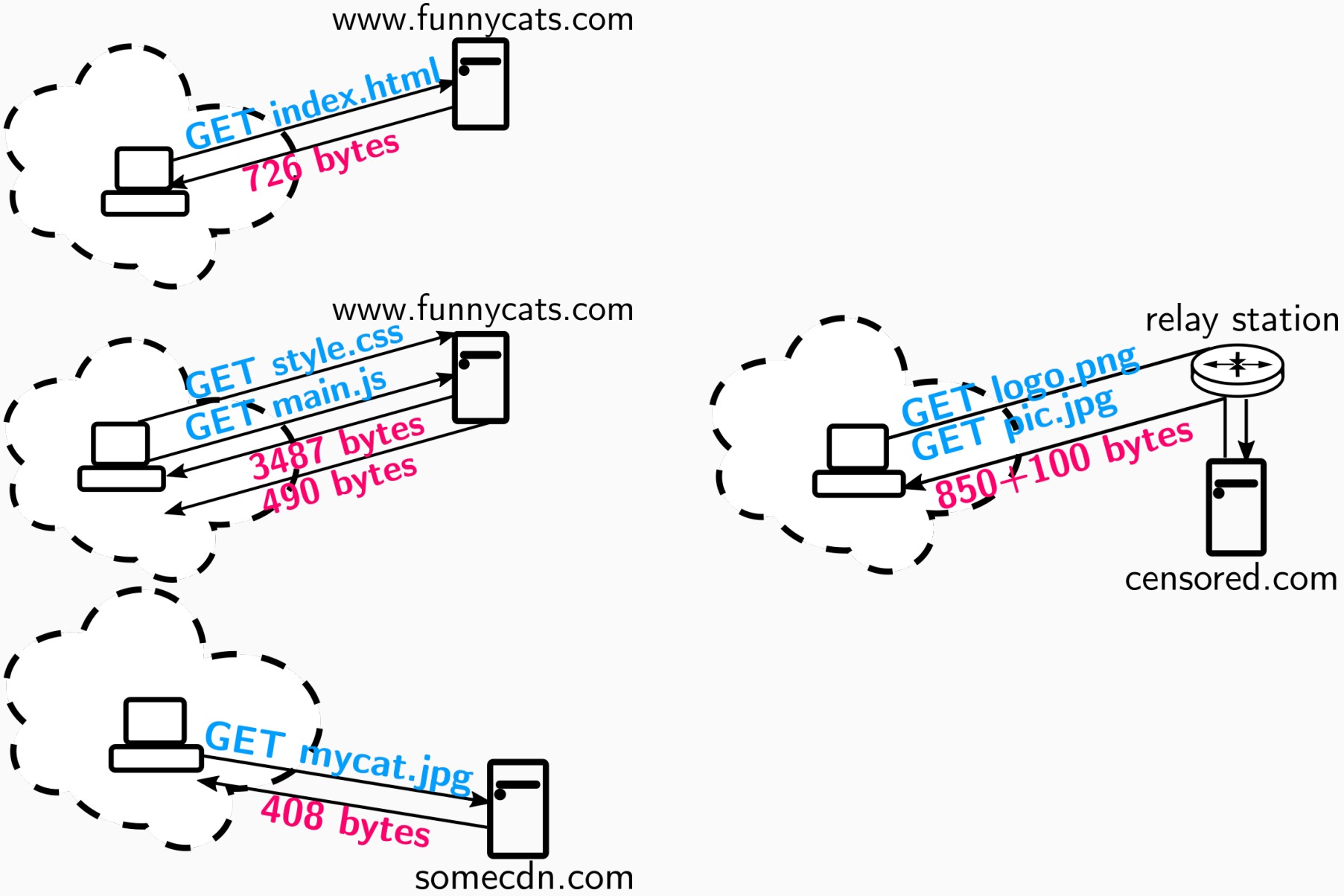
Traffic Analysis



Traffic Analysis

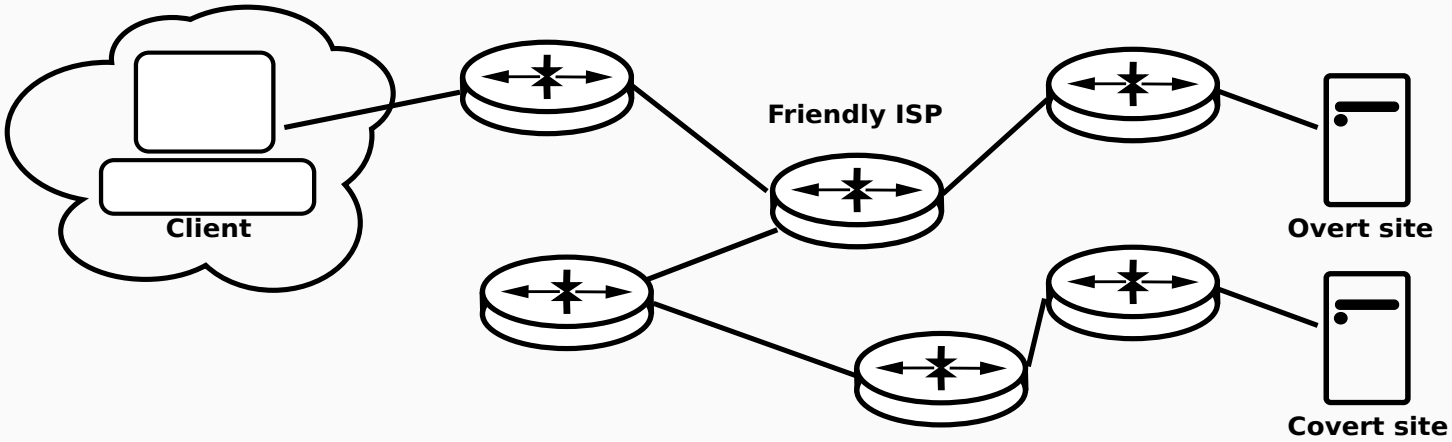


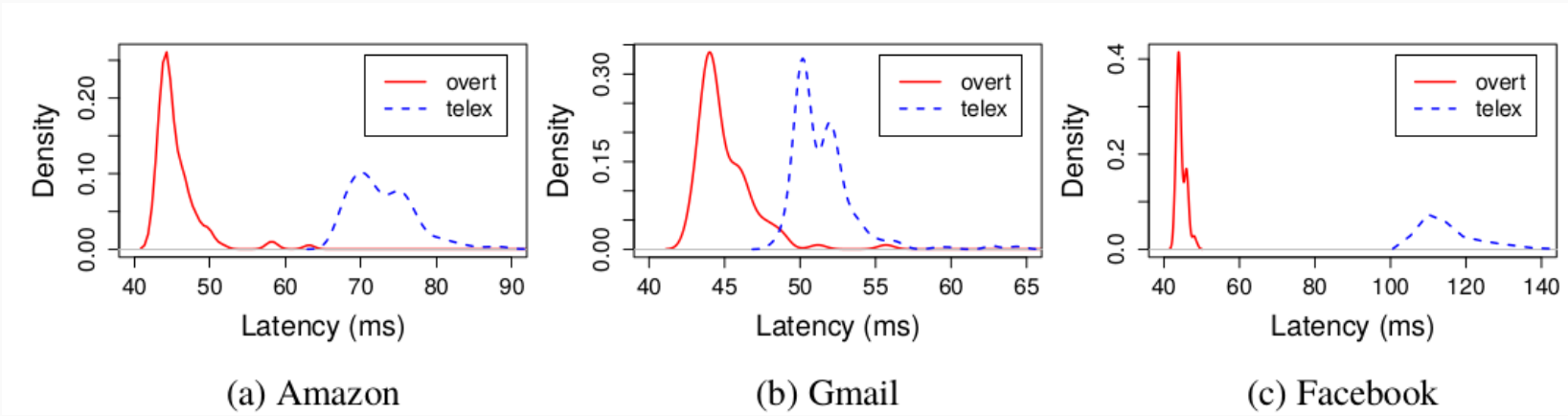
Traffic Analysis



Latency Analysis

(Schuchard et al., 2012)



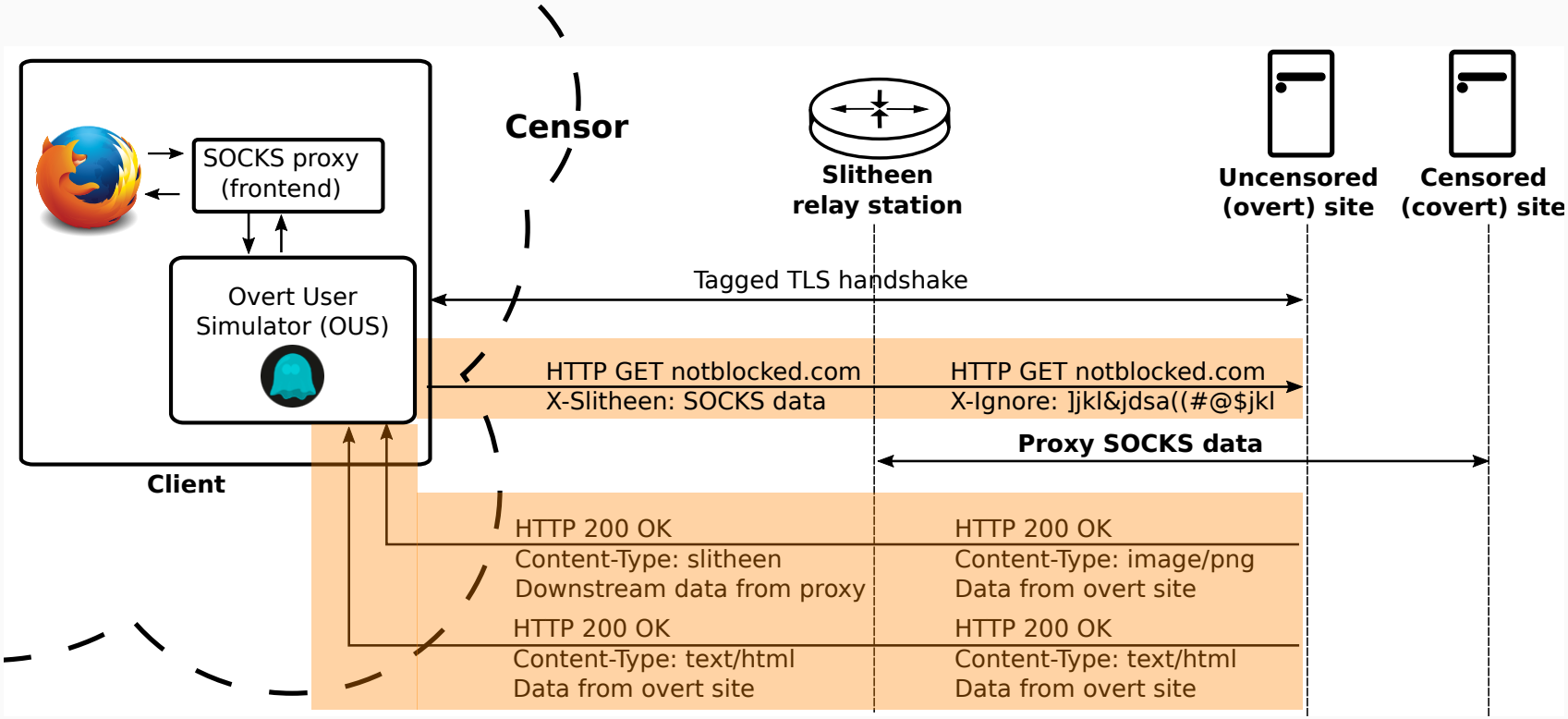


Slitheen

Slitheen traffic patterns to overt destinations **are identical to** a regular access to the overt site.

Covert content is squeezed into “leaf” resources (images, videos, etc.) that do not affect future connections for additional overt resources.

Architecture Overview



Tagging Procedure

- Relay station has keypair (r, g^r)

Tagging Procedure

- Relay station has keypair (r, g^r)
- Client picks s, uses $g^s \| H_1(g^{rs} \| \chi)$ as ClientHello random
  - Relay station (and *only* the relay station) can recognize the tag

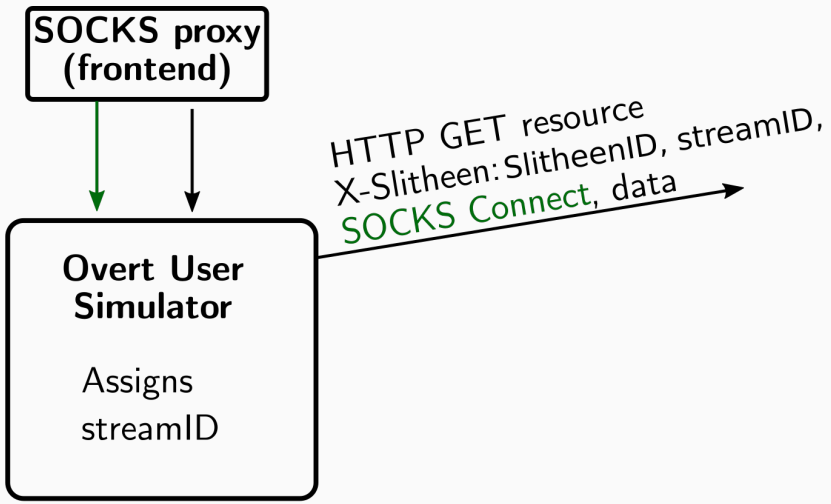
Tagging Procedure

- Relay station has keypair (r, g^r)
- Client picks s, uses $g^s \| H_1(g^{rs} \| \chi)$ as ClientHello random
  - Relay station (and *only* the relay station) can recognize the tag
- Client uses $H_2(g^{rs} \| \chi)$ as (EC)DHE private key
  - Relay station can compute the TLS master secret and MITM the connection

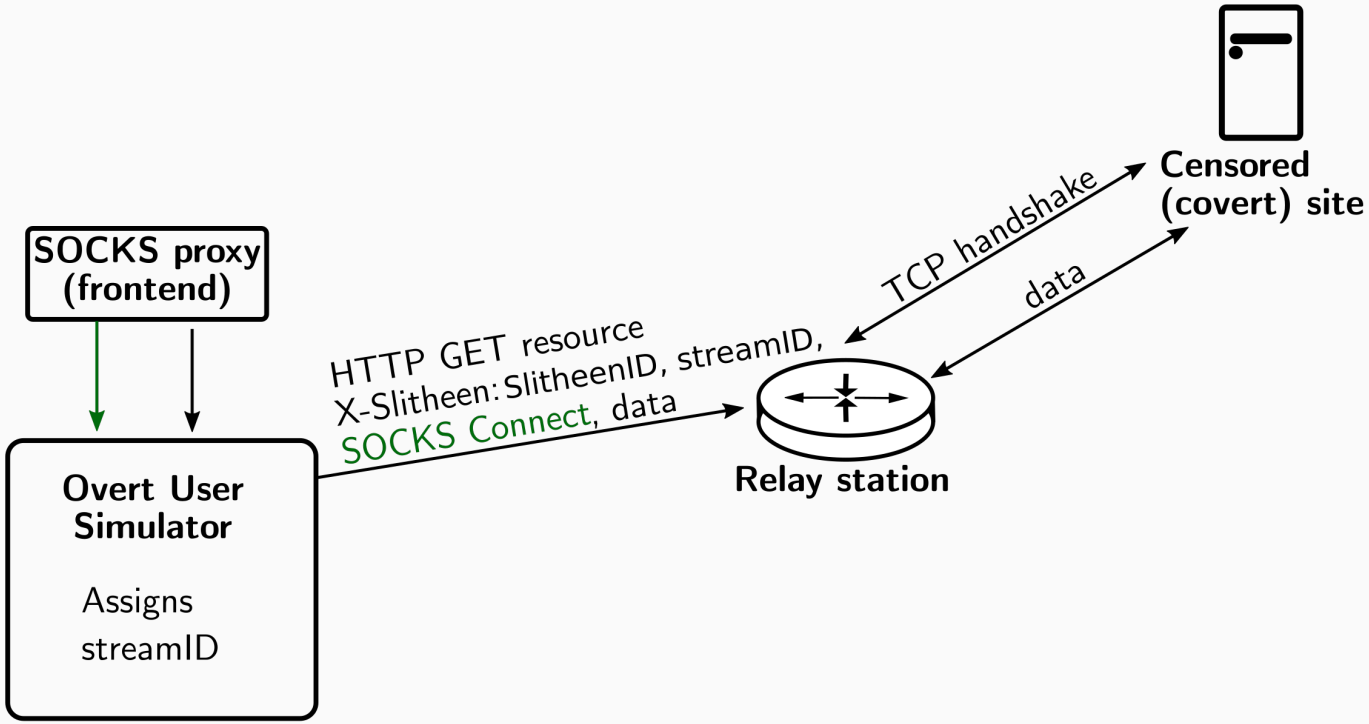
Tagging Procedure

- Relay station has keypair (r, g^r)
- Client picks s, uses $g^s \| H_1(g^{rs} \| \chi)$ as ClientHello random
  - Relay station (and *only* the relay station) can recognize the tag
- Client uses $H_2(g^{rs} \| \chi)$ as (EC)DHE private key
  - Relay station can compute the TLS master secret and MITM the connection
- Relay station modifies the server's Finished message to alert the client that Slitheen is active

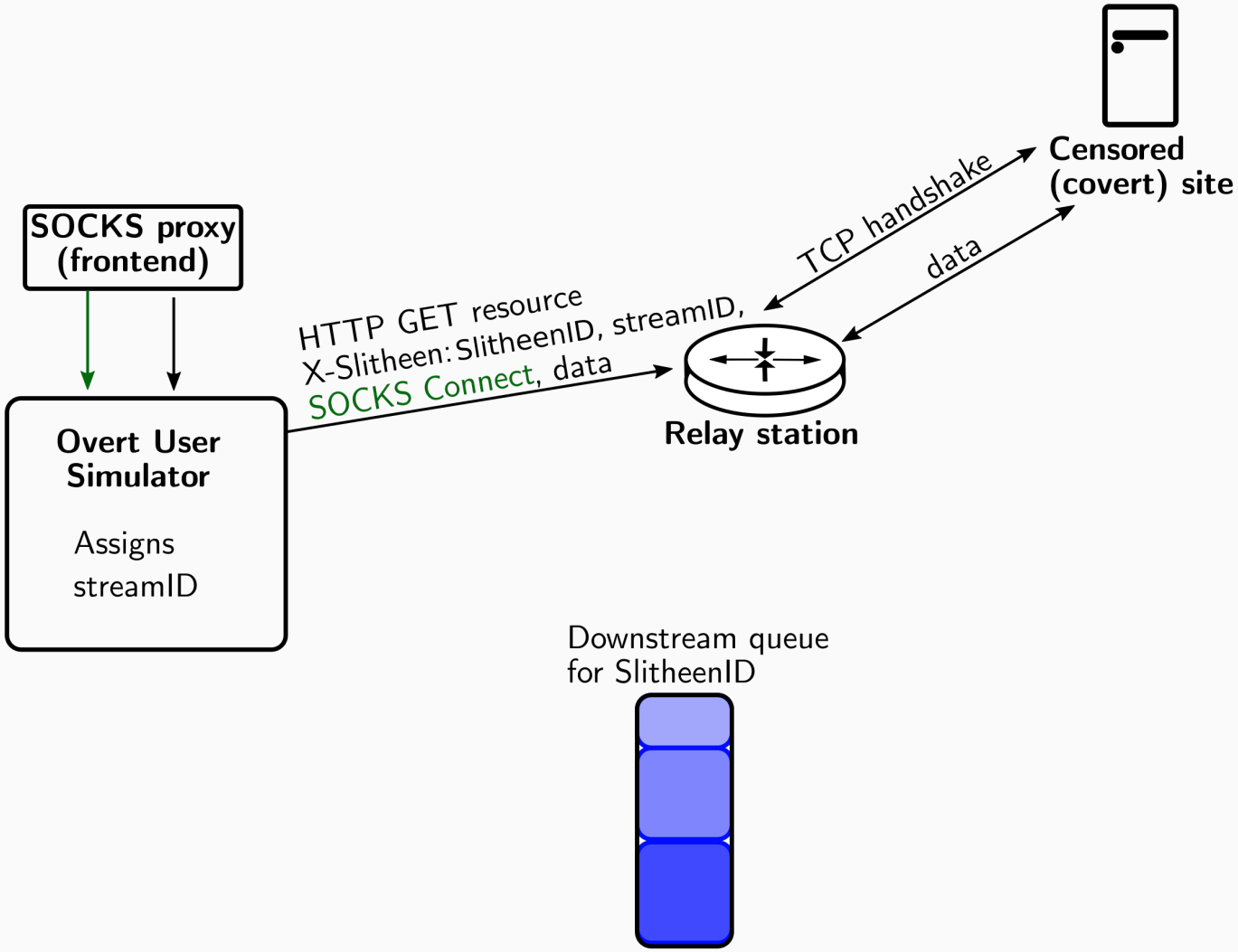
Data Replacement



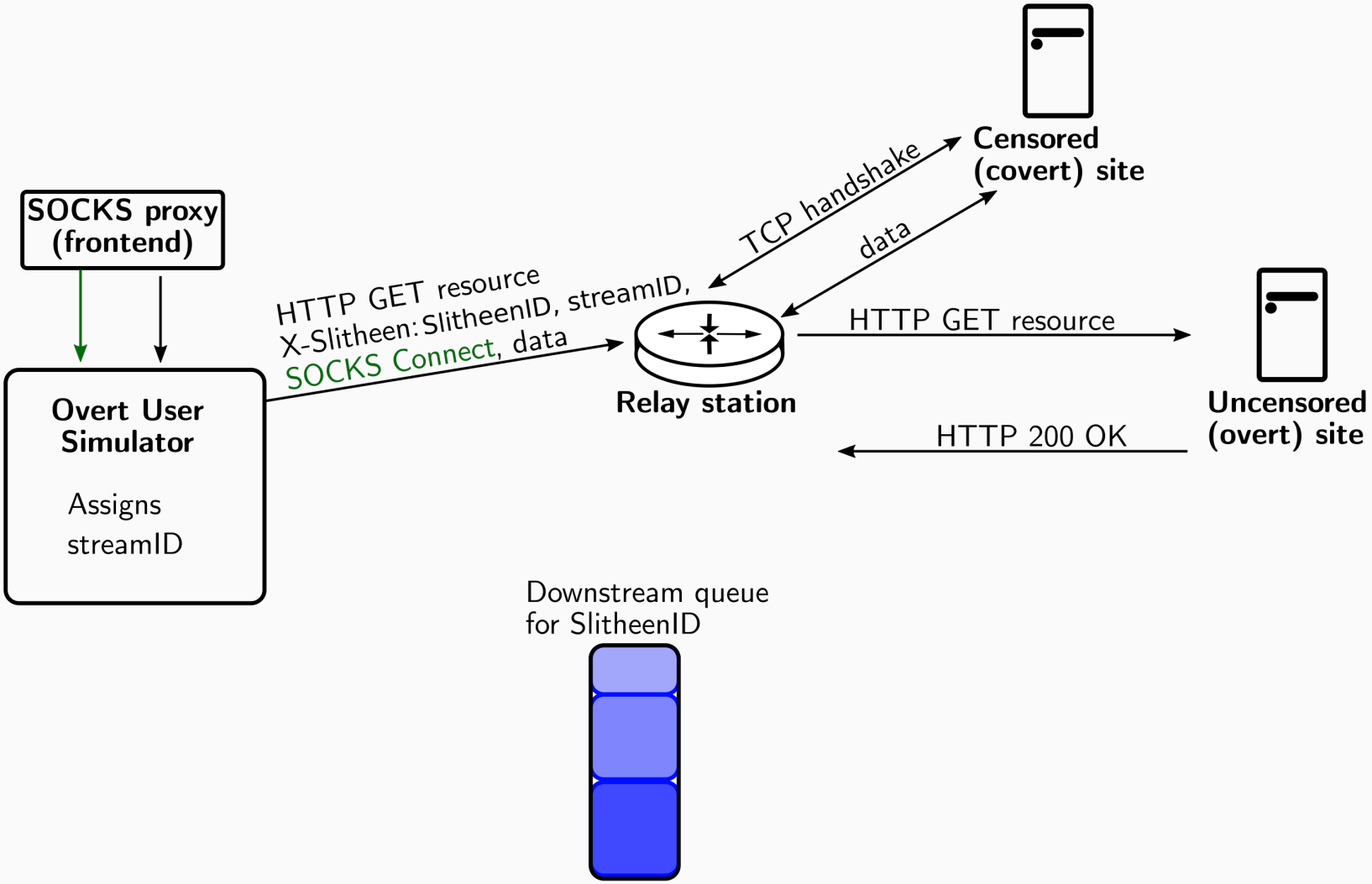
Data Replacement



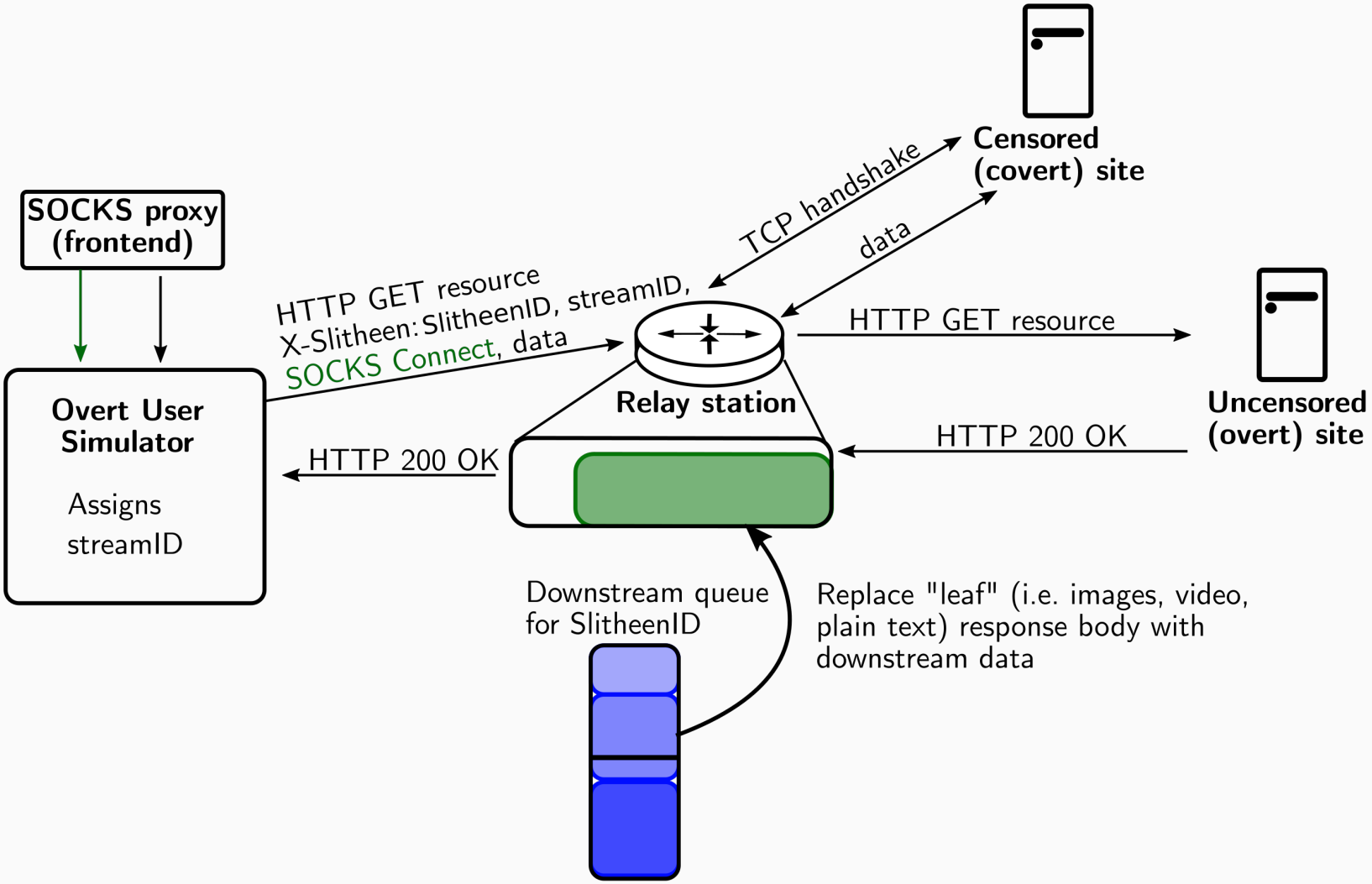
Data Replacement



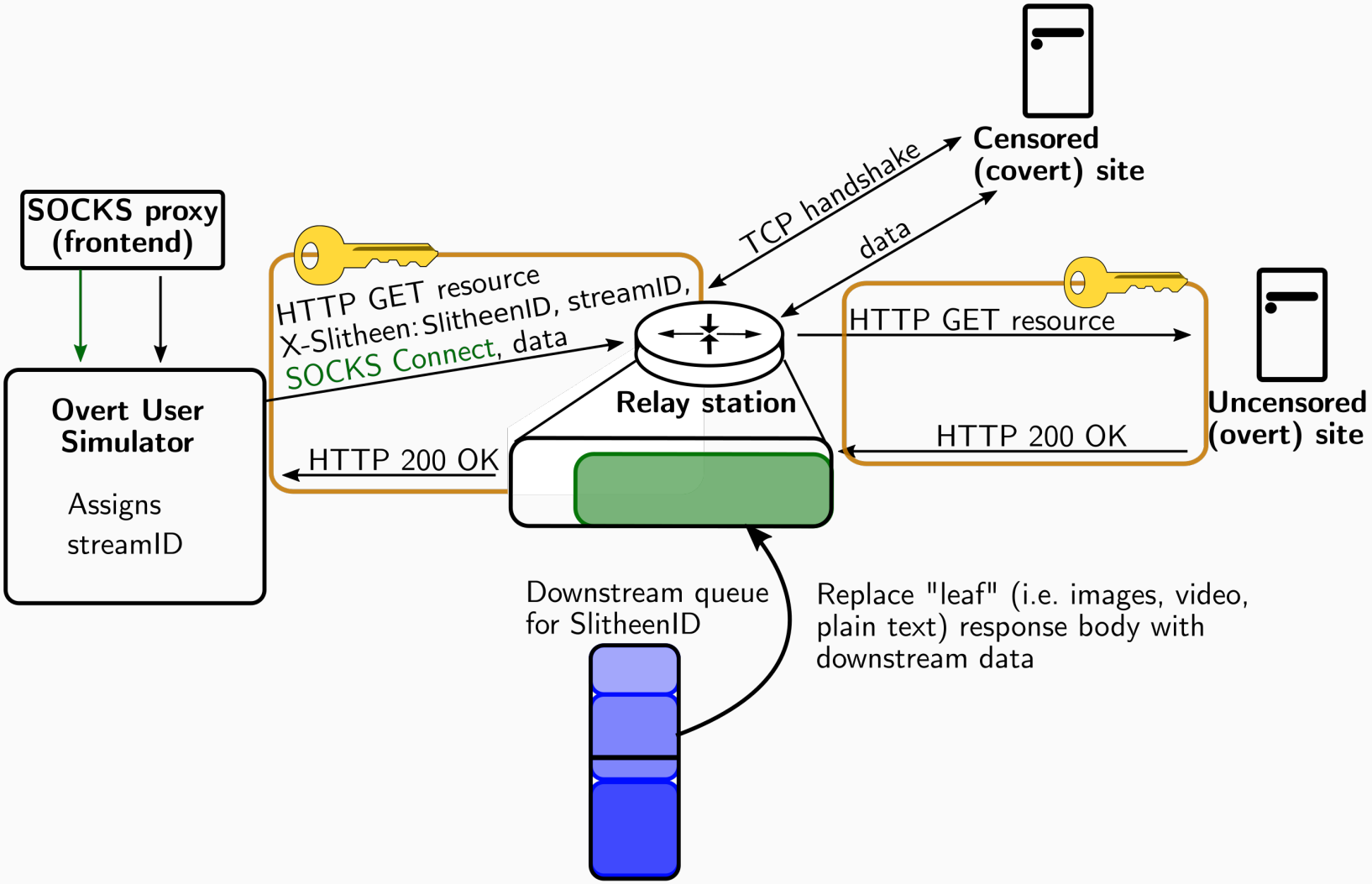
Data Replacement



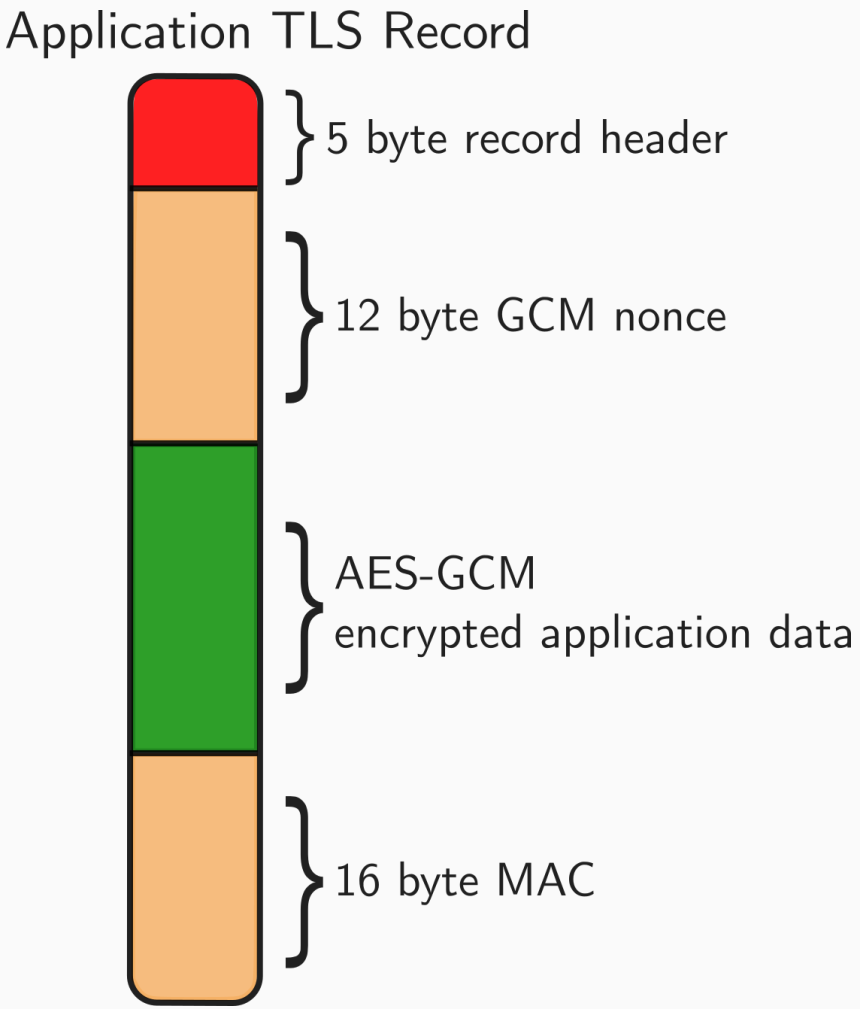
Data Replacement



Data Replacement

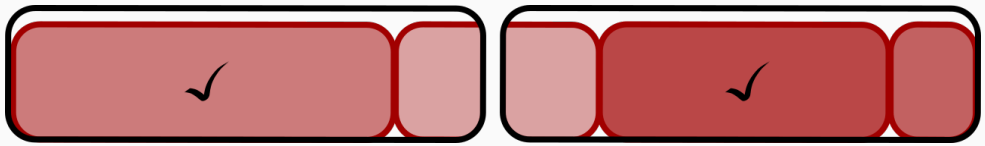


TLS Record Format

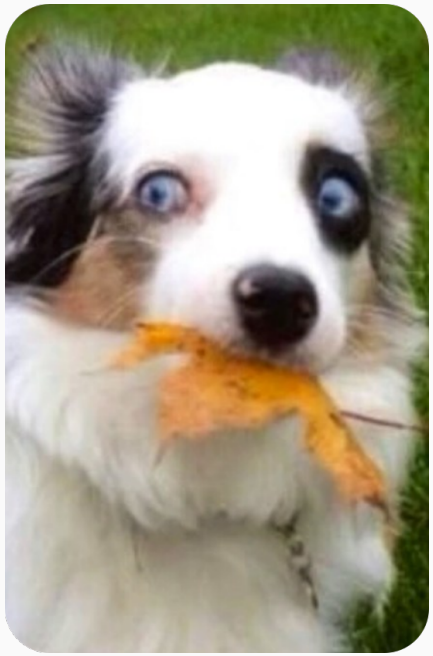


- Encrypted HTTP responses are sent from the overt site in a series of TLS records
- TLS records can be (and often are) fragmented across packets
- We do not delay packets at the relay station to reconstruct records

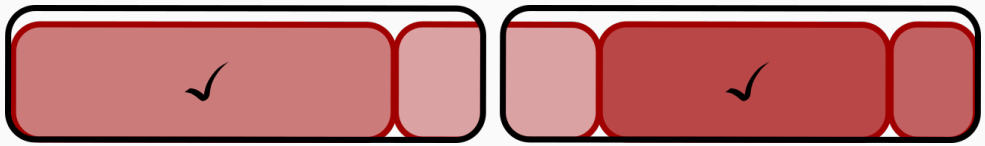
Finding Leaves



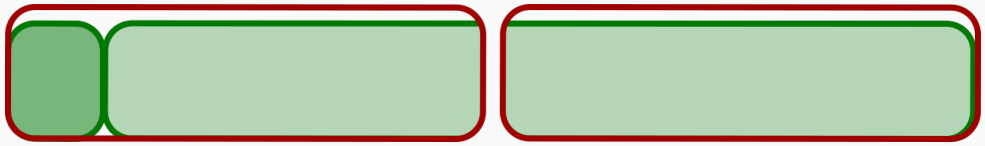
We can only decrypt a record after receiving **all** of it.



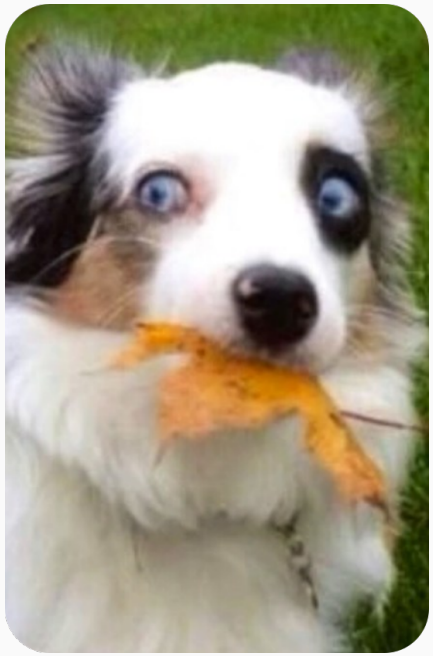
Finding Leaves



We can only decrypt a record after receiving **all** of it.

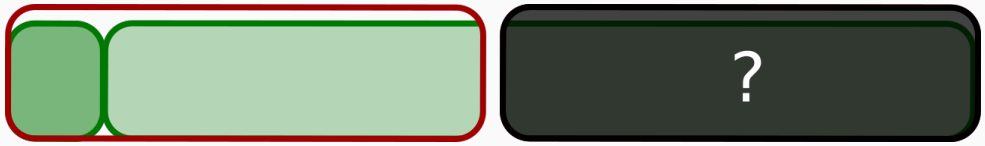


We only need to **decrypt the HTTP response header** to find leaves.



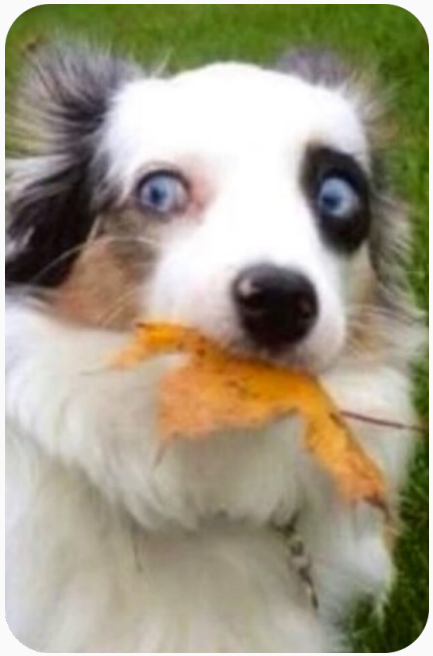
Finding Leaves

We can only decrypt a record after receiving **all** of it.

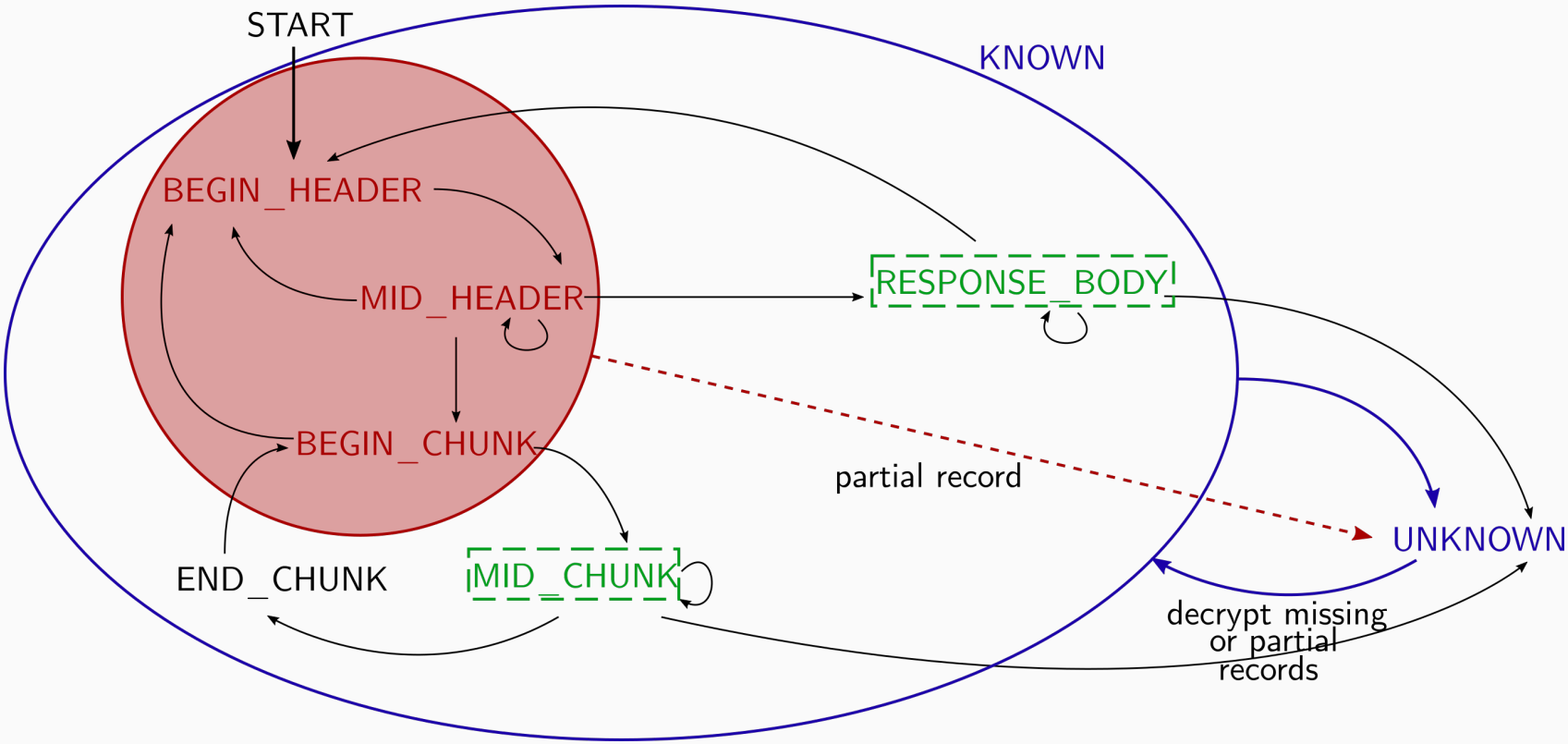


We only need to **decrypt the HTTP response header** to find leaves.

Misordered packets further complicate our decisions.

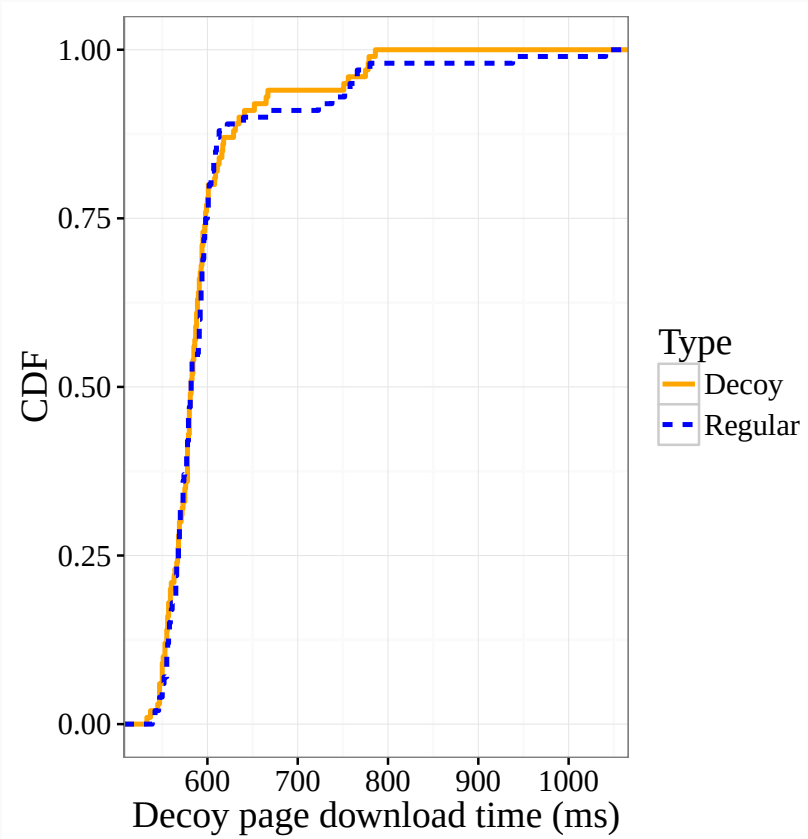


HTTP States

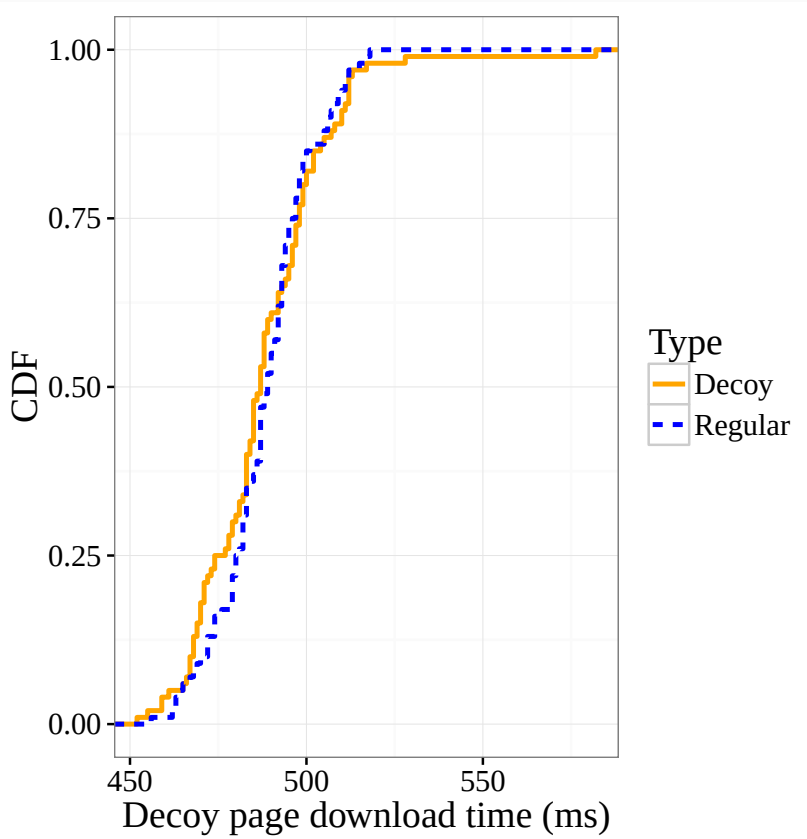

START
KNOWN
BEGIN_HEADER
MID_HEADER
RESPONSE_BODY
BEGIN_CHUNK
partial record
UNKNOWN
END_CHUNK
MID_CHUNK
decrypt missing
or partial
records

Latency Results

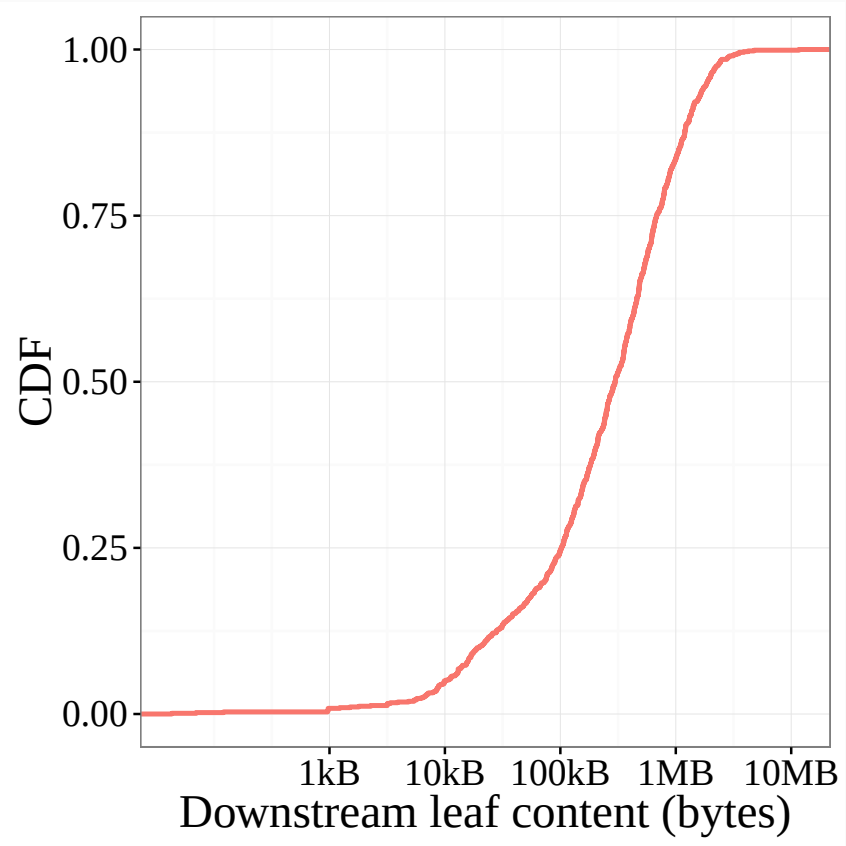
Gmail



Wikipedia



Bandwidth

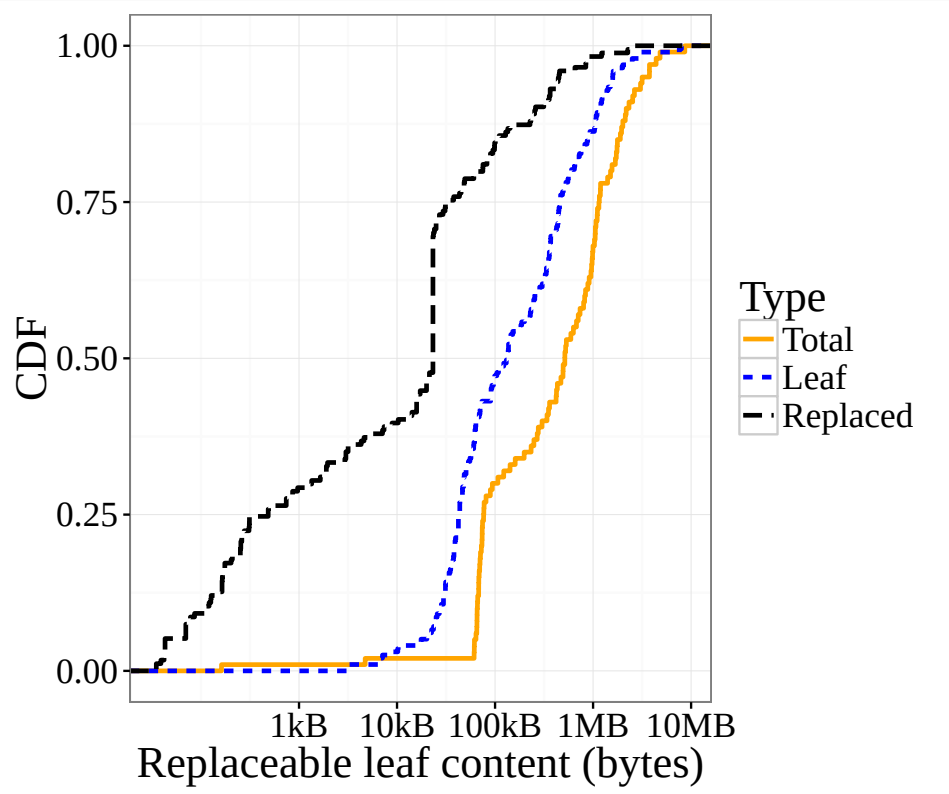


Downstream leaf content from the Alexa top 10,000 TLS sites

- Roughly 25% of all sites offer 500 kB or more of potentially replaceable content
- About 40% of traffic across all sites was leaf content

Realistic Bandwidth

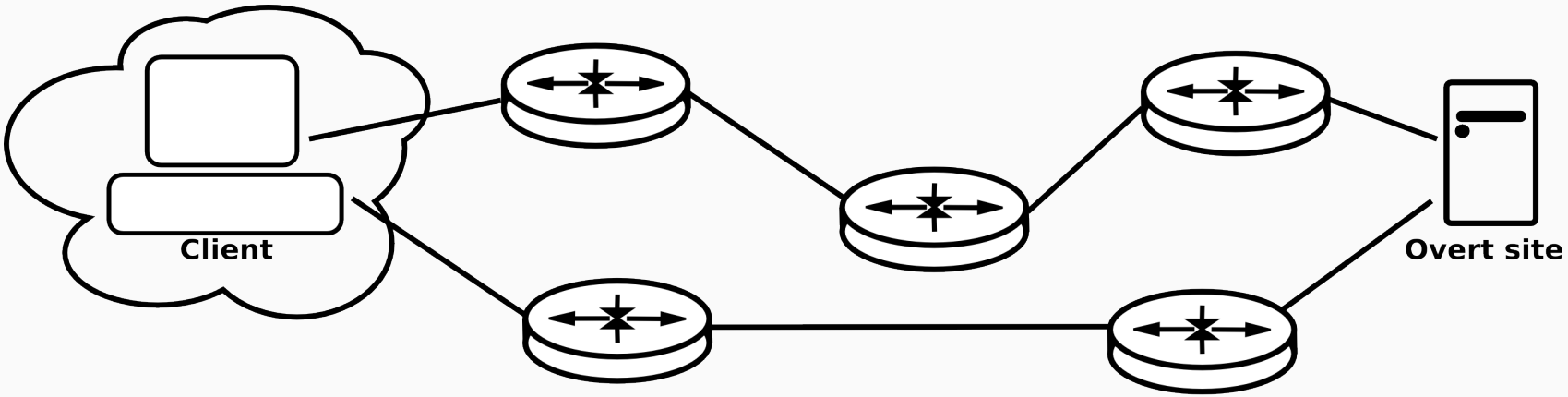
Site name	Leaf content (bytes)	% leaf content replaced	% total replaced
Gmail	8800 ± 100	87.7 ± 0.2	23 ± 9
Wikipedia	24000 ± 2000	100 ± 0	33 ± 4
Yahoo	400000 ± 100000	100.0 ± 0.2	40 ± 20
Facebook	40000 ± 10000	0 ± 0	0 ± 0



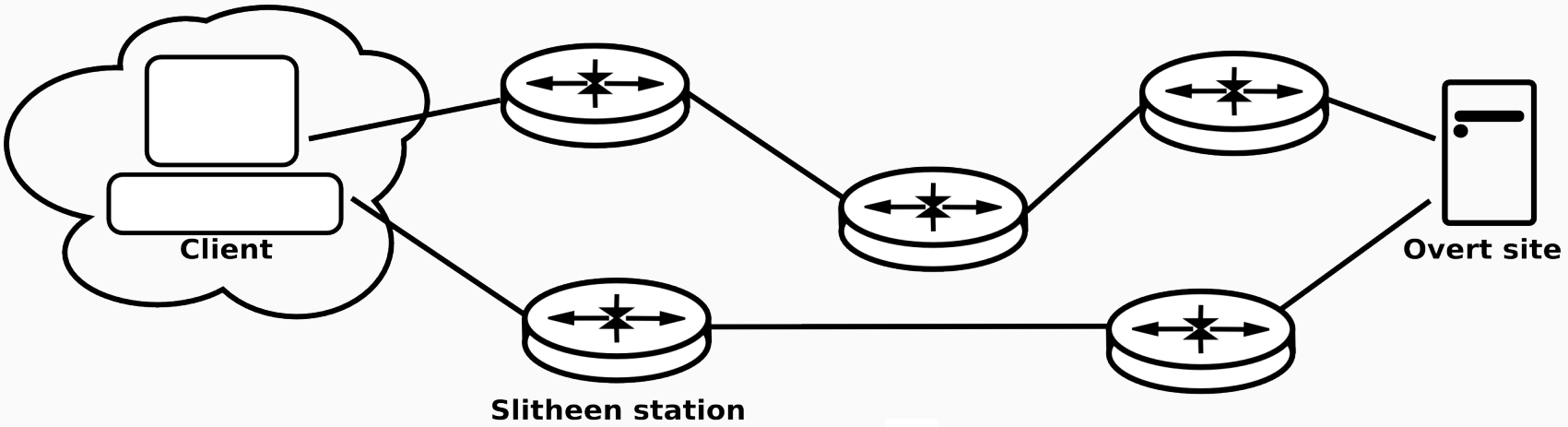
Comparison

	Telex	Cirripede	Curveball	TapDance	Rebound	Slitheen
No in-line blocking	○	○	○	●	○	○
Supports asymmetric routes	○	●	○	●	●	○
Defends against TCP replay attacks	●	●	●	○	●	●
Defends against latency analysis	○	○	○	○	●	●
Defends against website fingerprinting	○	○	○	○	○	●
RAD-resistant	○	○	○	○	○	○

Supporting Asymmetry and RAD-Resistance

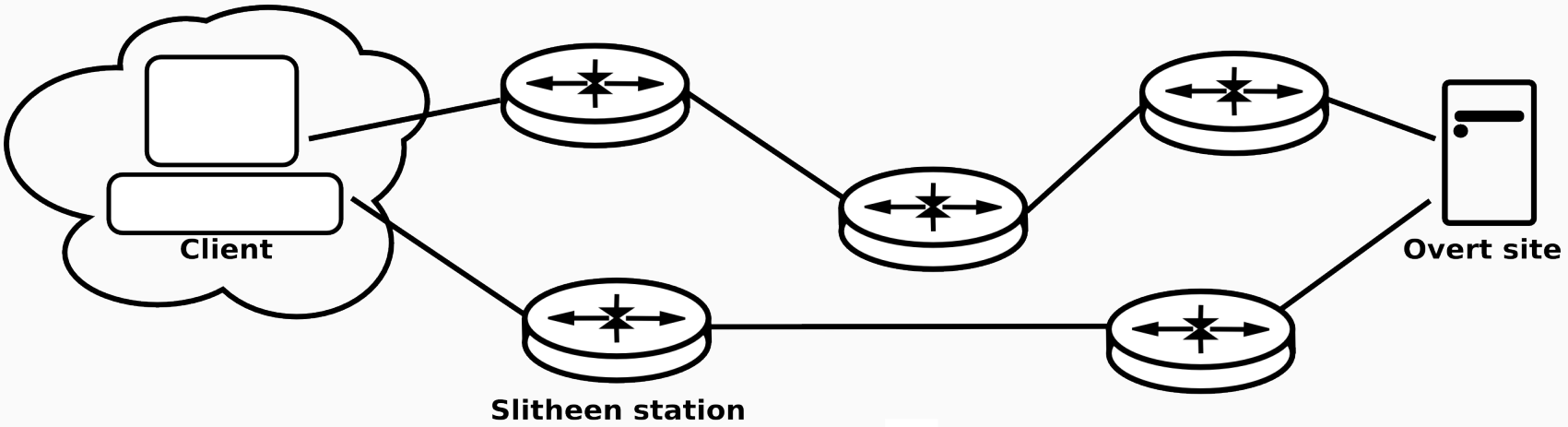


Supporting Asymmetry and RAD-Resistance



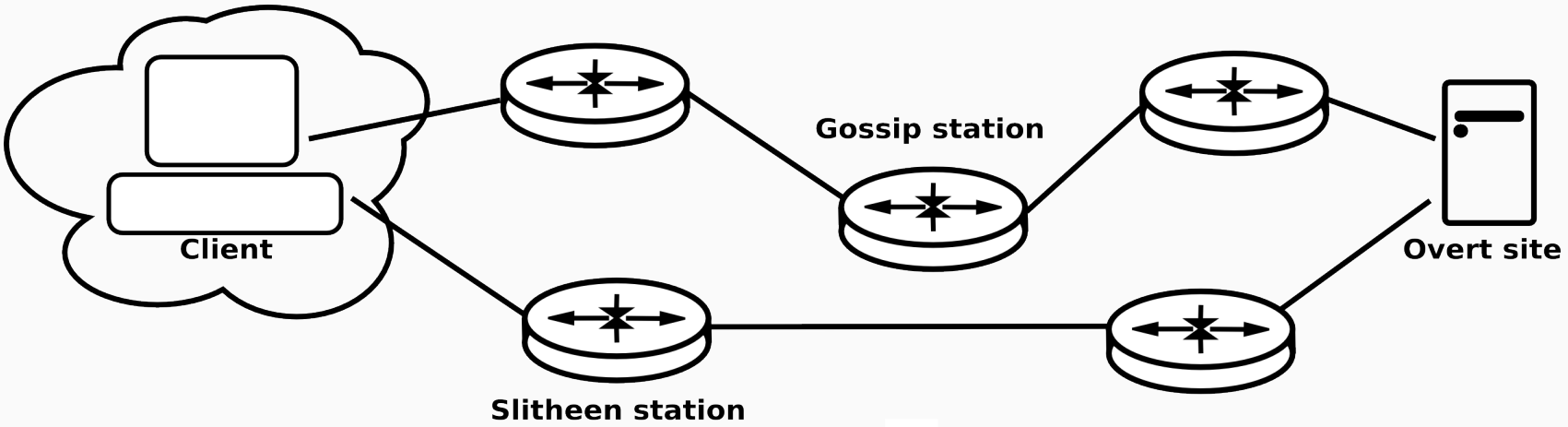
- Slitheen station is on downstream path
  - Opposite to TapDance, Rebound

Supporting Asymmetry and RAD-Resistance



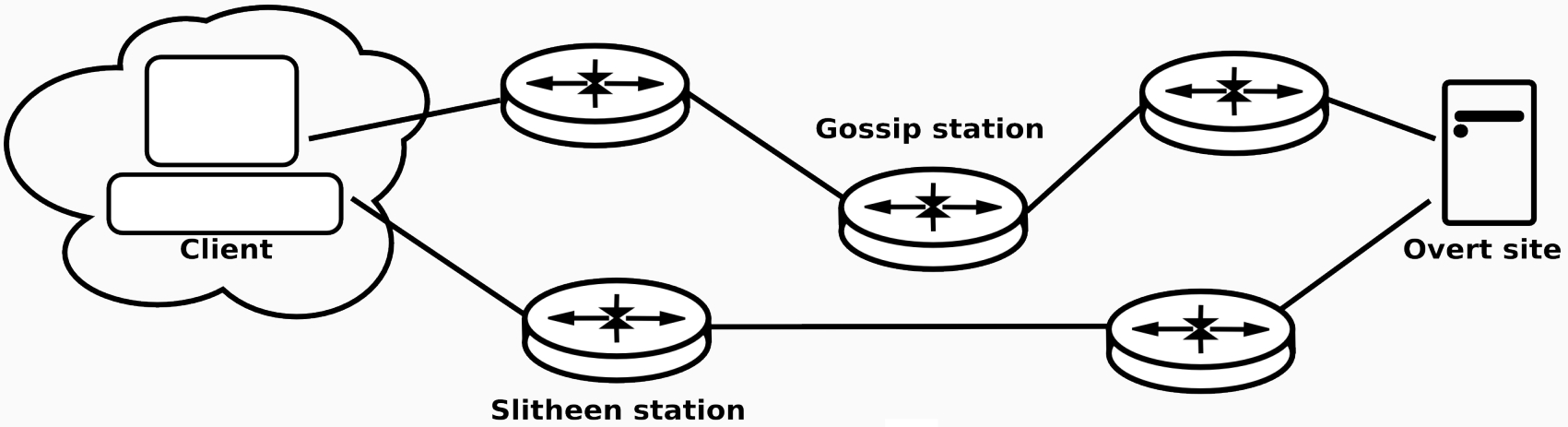
- Slitheen station is on downstream path
  - Opposite to TapDance, Rebound
- How does it identify tagged flows and learn the TLS master secret?

Supporting Asymmetry and RAD-Resistance



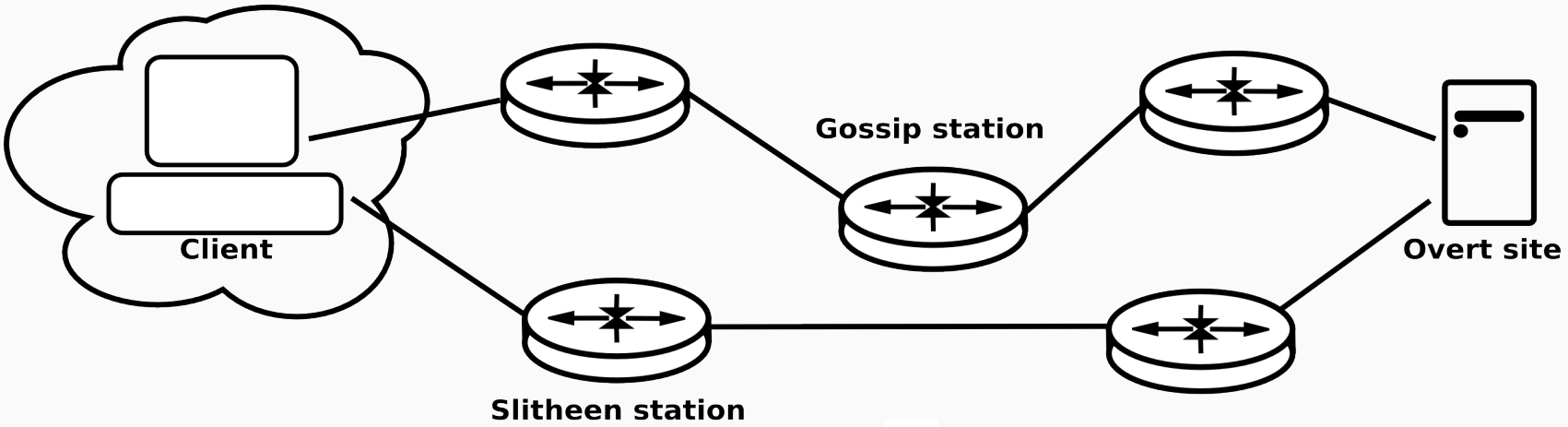
- Lightweight gossip station on upstream path
  - No flow blocking; just gets a copy of TLS flows
  - When it sees a TLS ClientHello (without having seen a TCP SYN ACK), broadcast it to Slitheen stations
  - If a Slitheen station claims the tag, send upstream TLS data to it

Supporting Asymmetry and RAD-Resistance



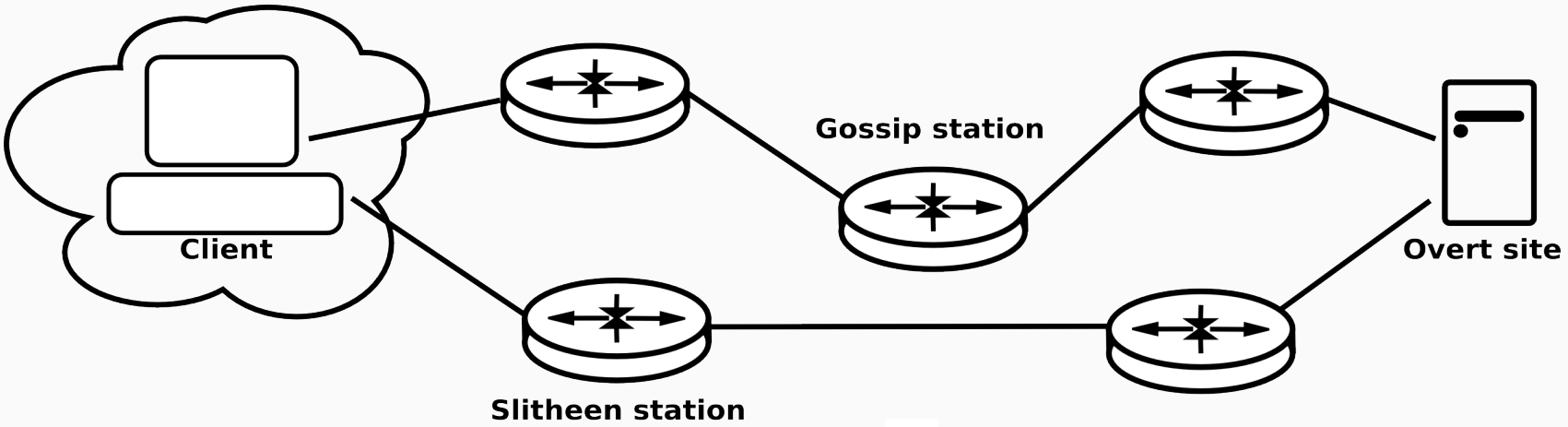
- But surely that upstream ClientHello won't get from the gossip station to the Slitheen station in time?
  - The Slitheen station needs it before the TLS handshake completes so that it can read and modify the Finished message

Supporting Asymmetry and RAD-Resistance



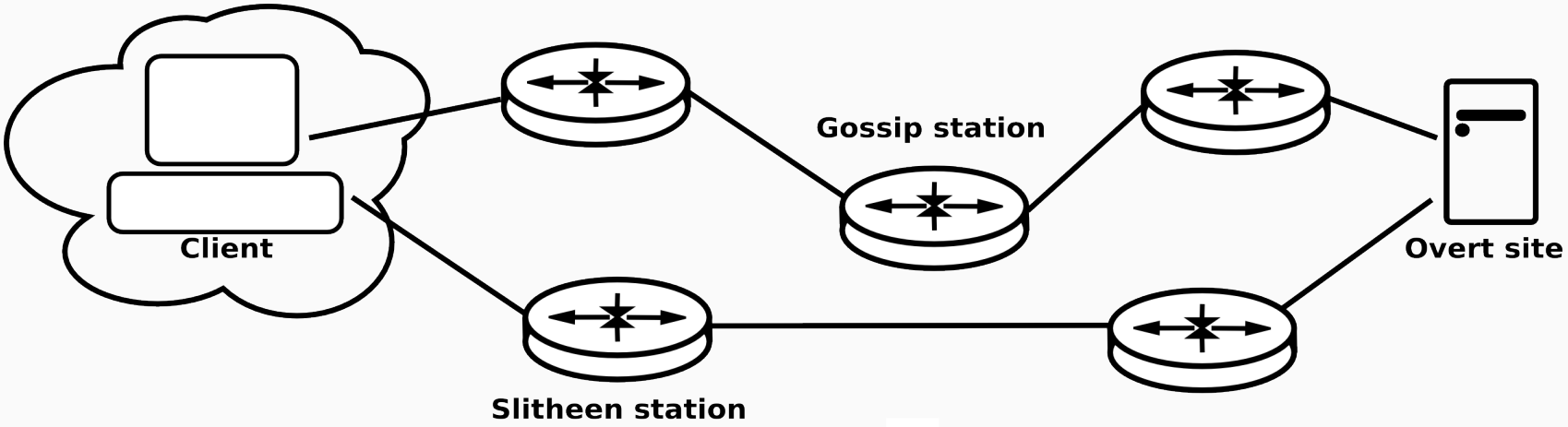
- Key idea: the client's Slitheen secret *s* on its *next* connection to that overt site will be selected as a function of the *previous* client-relay shared secret
  - The first connection acts as a Cirripede-esque registration
  - The Slitheen station can then predict that client's future ClientHello messages!

Supporting Asymmetry and RAD-Resistance



- Gossip stations offer a *two-tiered deployment* strategy
- No need for flow-blocking or traffic replacement routers
  - So easier to deploy

Supporting Asymmetry and RAD-Resistance



- Easier for censor to perform RAD attack on upstream data (change routing for *that one flow*) than downstream (advertise new BGP route to *everyone*)
  - Put lots of cheap gossip stations on possible upstream paths
  - More heavyweight Slitheen stations on more stable downstream paths

Comparison

	Telex	Telex+gossip	Cirripede	Curveball	Curveball+gossip	TapDance	Rebound	Slitheen	Slitheen+gossip
No in-line blocking	○	○	○	○	○	●	○	○	○
Supports asymmetric routes	○	●	●	○	●	●	●	○	●
Defends against TCP replay attacks	●	●	●	●	●	○	●	●	●
Defends against latency analysis	○	○	○	○	○	○	●	●	●
Defends against website fingerprinting	○	○	○	○	○	○	○	●	●
RAD-resistant	○	●	○	○	●	○	○	○	●

Summary

- Slitheen is a new proposal for a decoy routing system
- Slitheen addresses previously undefended passive attacks
- Our results show no discernible difference in latency between a "decoy access" to an overt destination and a regular access
- By design, Slitheen defends against website fingerprinting attacks by maintaining packet sizes, timings, and directionality
- The gossip protocol addresses the major challenges to deployability: RAD attacks, asymmetric flows, and concerns over inline blocking
- Implementation and source code of Slitheen (but not yet the gossip protocol) available:
  `https://crysp.uwaterloo.ca/software/slitheen/`