Announcement

- Quiz 5
  - Opens at 9am on 28 Mar, 2023
  - Closes at 6:30pm on 3 April 2023
  - Covers Week 9 and 10 lecture slides

- Week 13 Lecture
  - Mini-Project presentation
  - Talk on digital twin for smart grid security
  - Q&A and open discussion

CS 5321 Network Security
Week11: Anti-censorship

Daisuke MASHIMA

http://www.mashima.us/daisuke/index.html

2022/23 Sem 2

CENSORSHIP

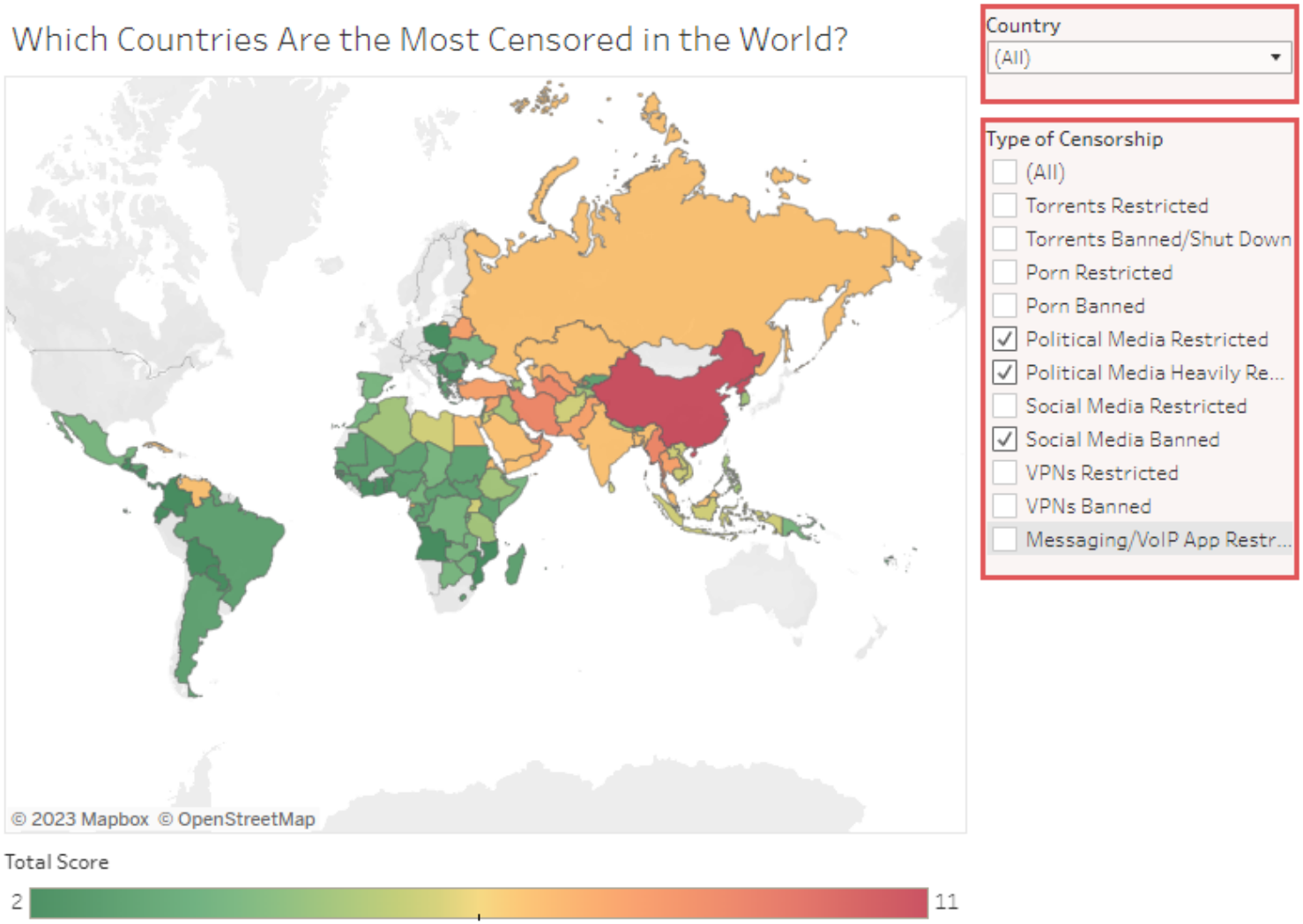
Internet Censorship

- The Internet is a big **threat** to repressive regimes!
- Repressive regimes **censor** the Internet:
  - IP filtering, DNS monitoring, Deep packet-inspection, etc.
- Circumvention systems

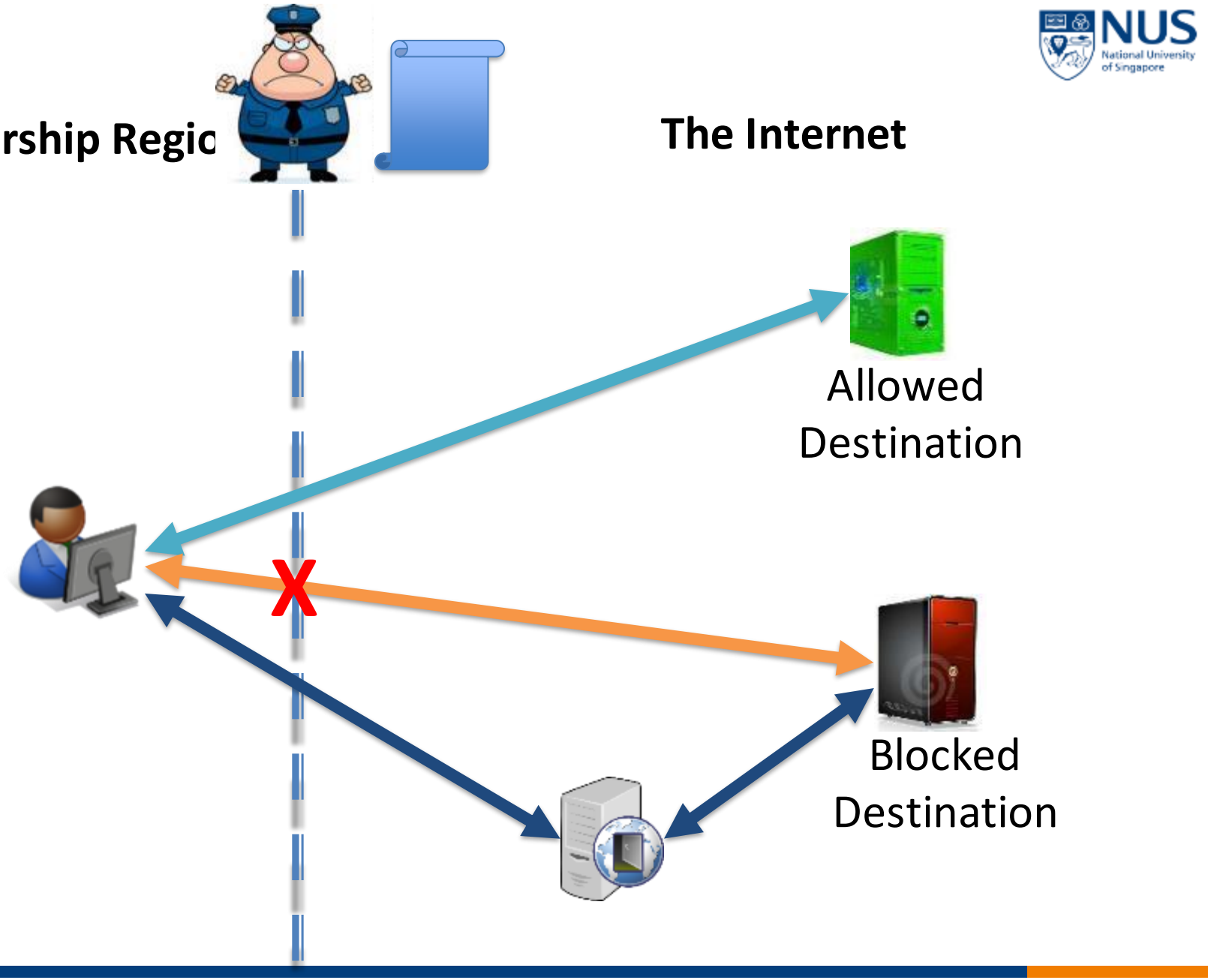
Commonly censored content

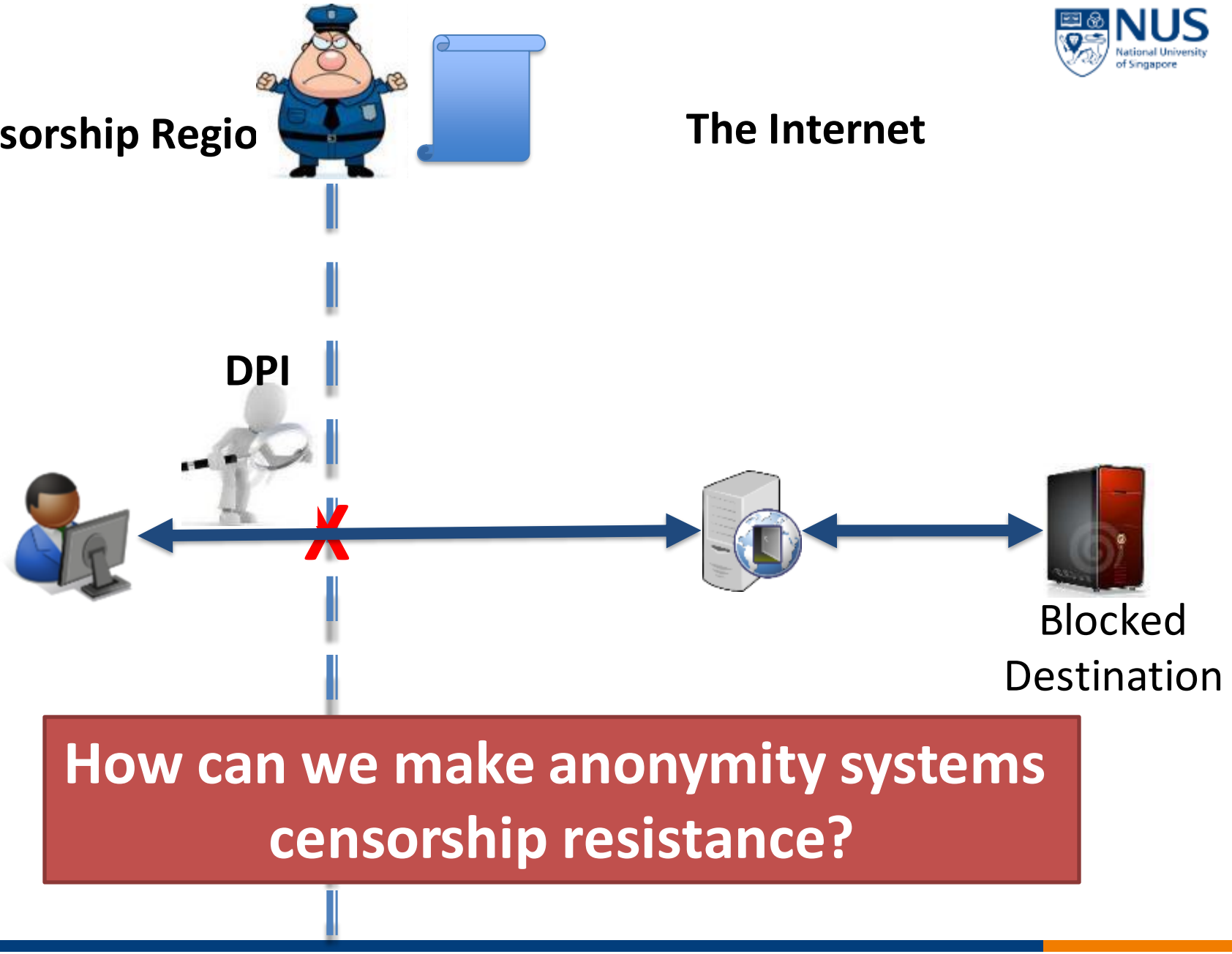
- Unapproved news
- Wikipedia (usually partially)
- Facebook
- Google (all services), YouTube
- Twitter
- Content prohibited by state religion

A Global Map of Internet Restrictions



https://www.comparitech.com/blog/vpn-privacy/internet-censorship-map/





Naïve Solutions

- VPNs
  - Censor can enumerate VPNs easily
- Tor Bridges
  - Bridges: relays that aren't listed in the main Tor directory
  - Censor can enumerate all bridges

Desired properties for anti-censorship systems?

- Unobservability
- Unblockability
- Plausible deniability
- Deployment feasibility
- Scalability

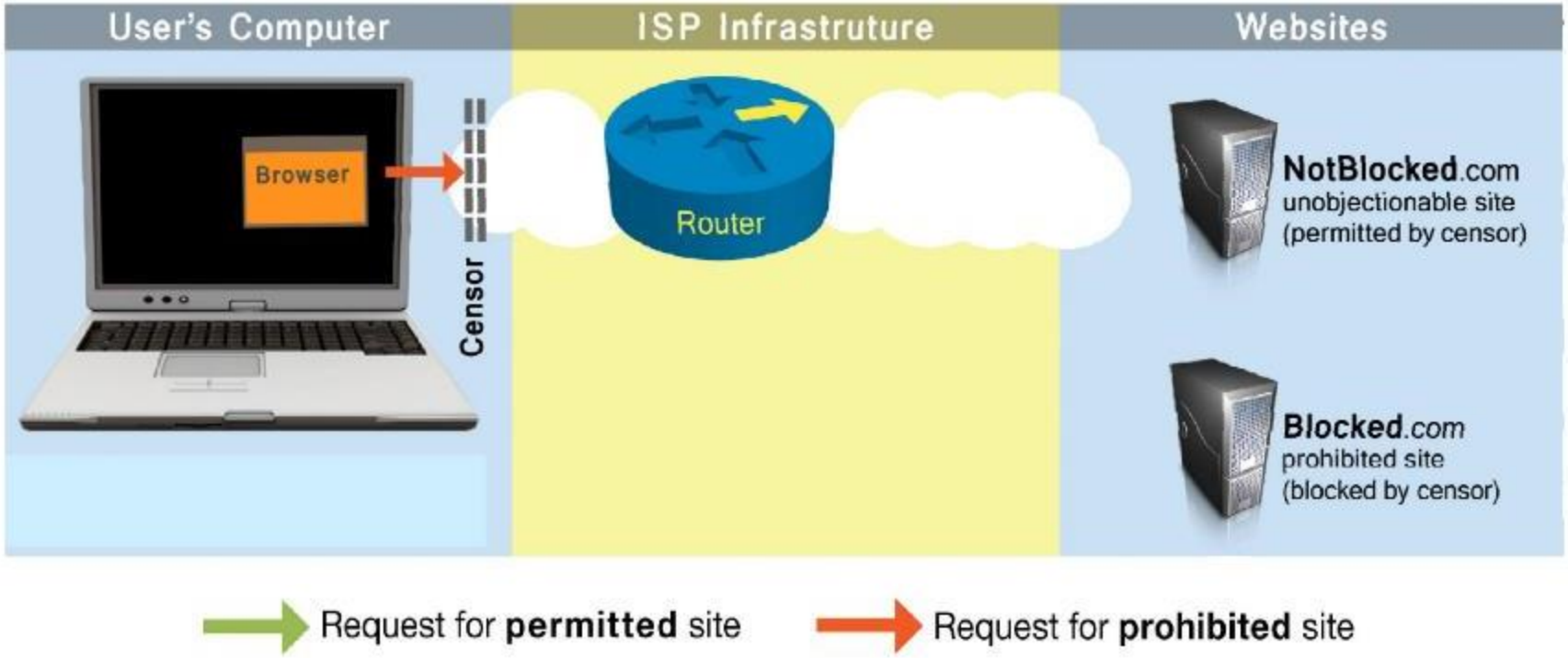
ANTI-CENSORSHIP VIA DECOY ROUTING

Network-aided anti-censorship

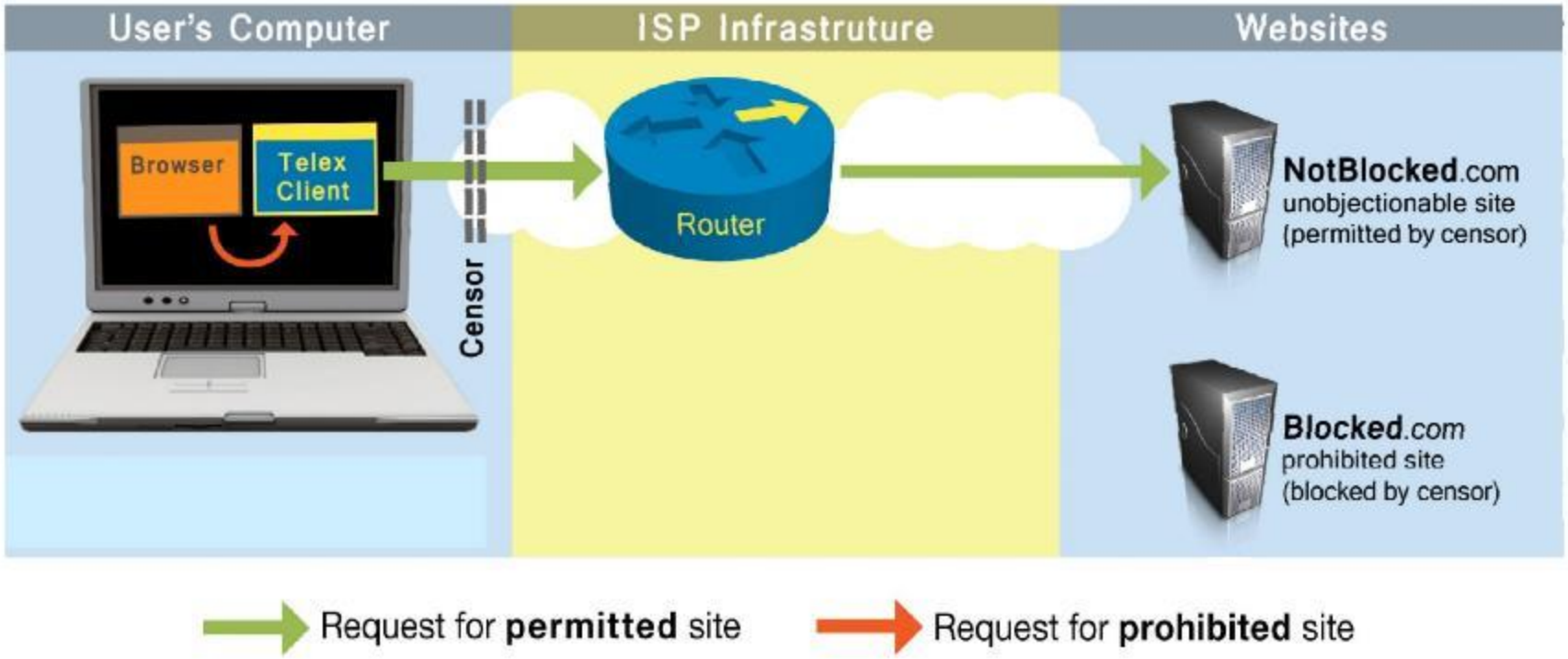
- **Friendly ISPs** can help people in repressed regimes access censored contents
  - *Is it realistic?*
  - *For idealism, goodwill, public relations, or financial incentives from government*

- Three proposals in the same year
  - Decoy Routing [FOCI'11]
  - Telex [USENIX Security'11]
  - Cirripede [CCS'11]

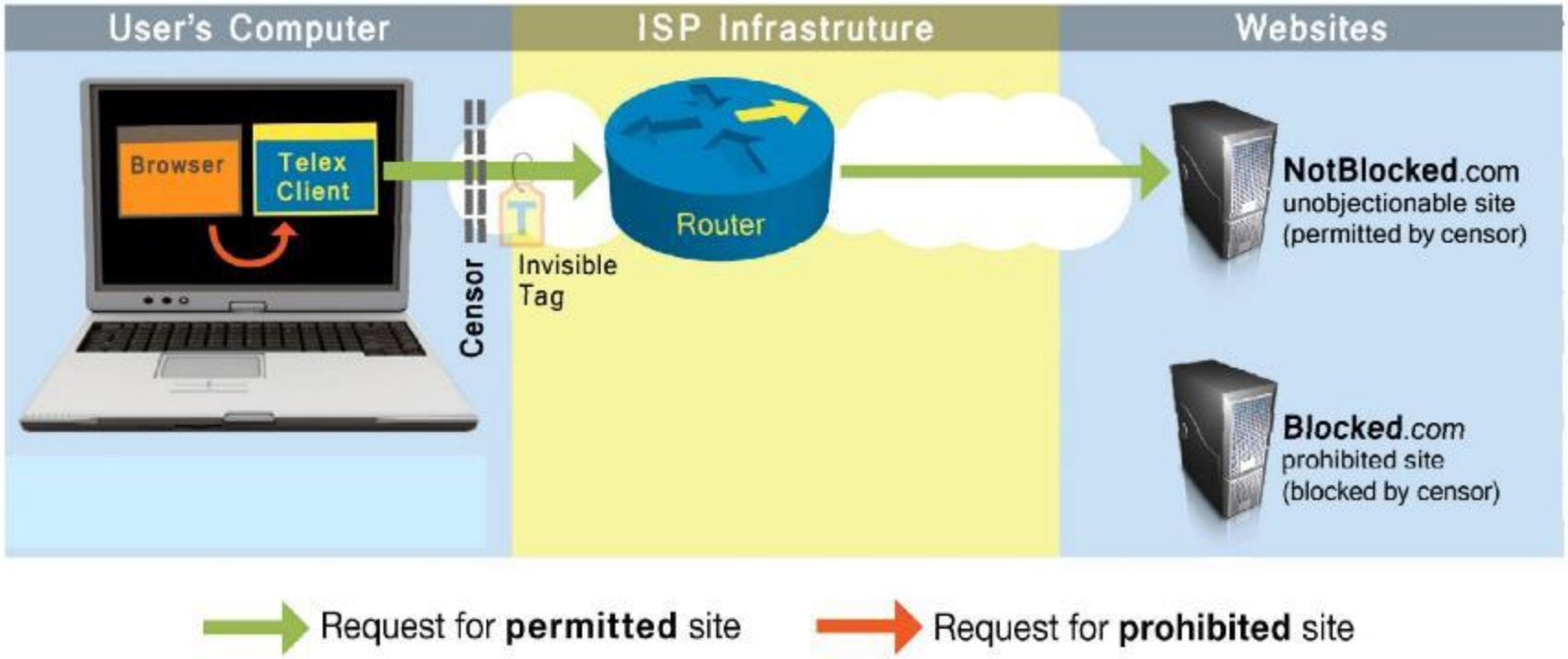
Telex: Overview



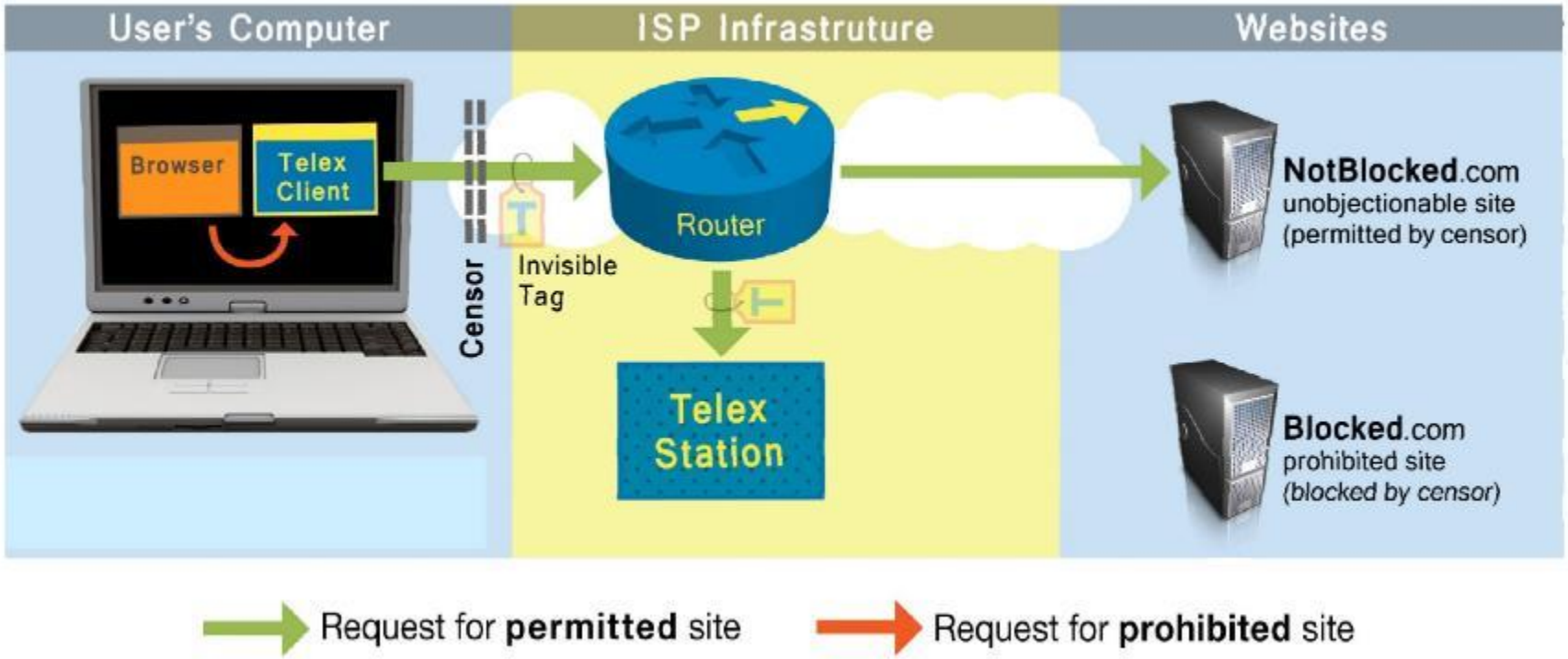
Telex: Overview



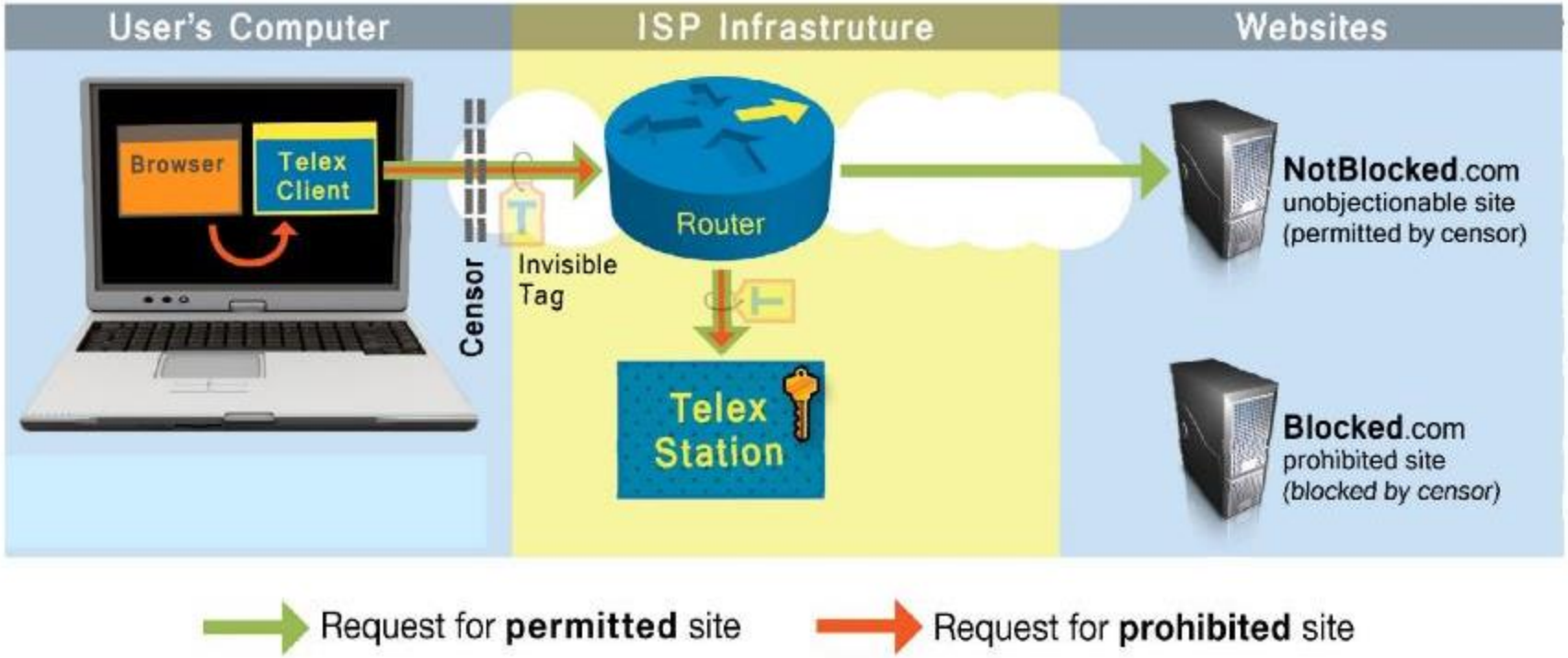
Telex: Overview



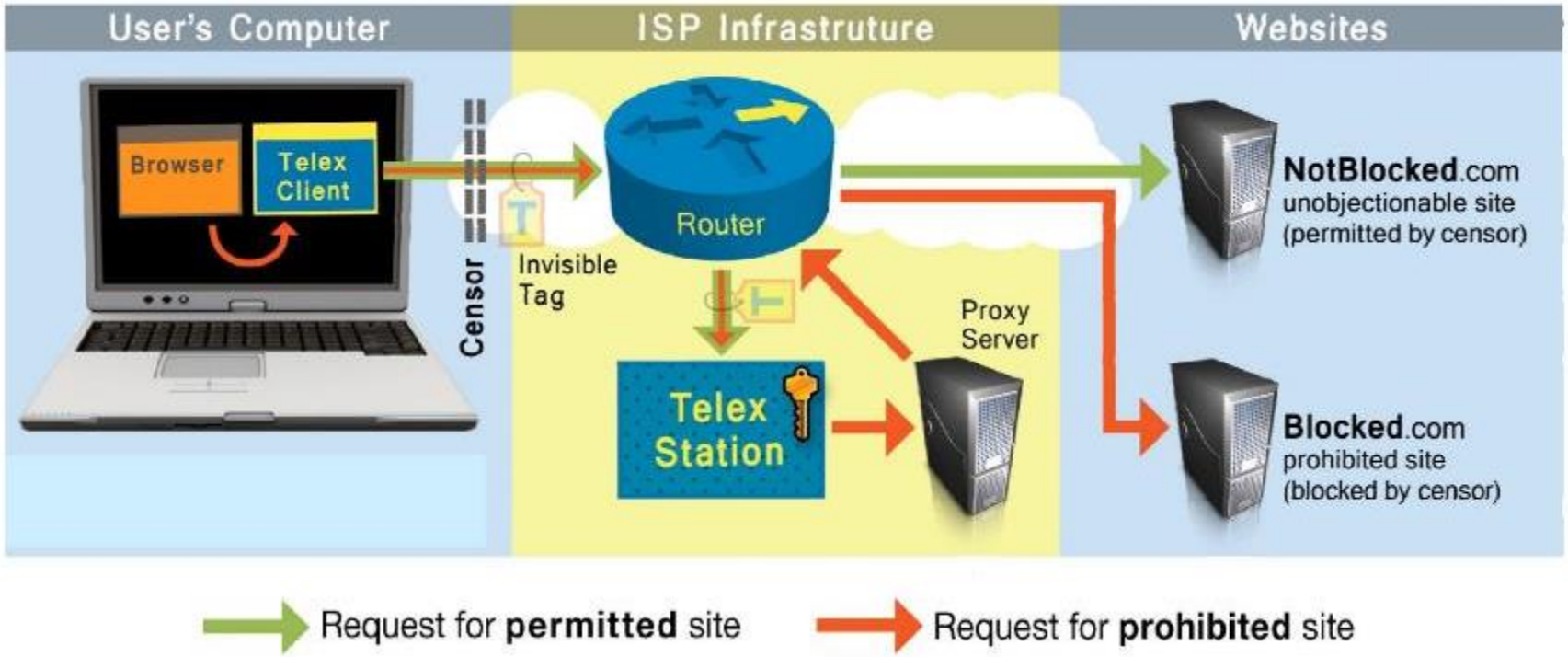
Telex: Overview



Telex: Overview

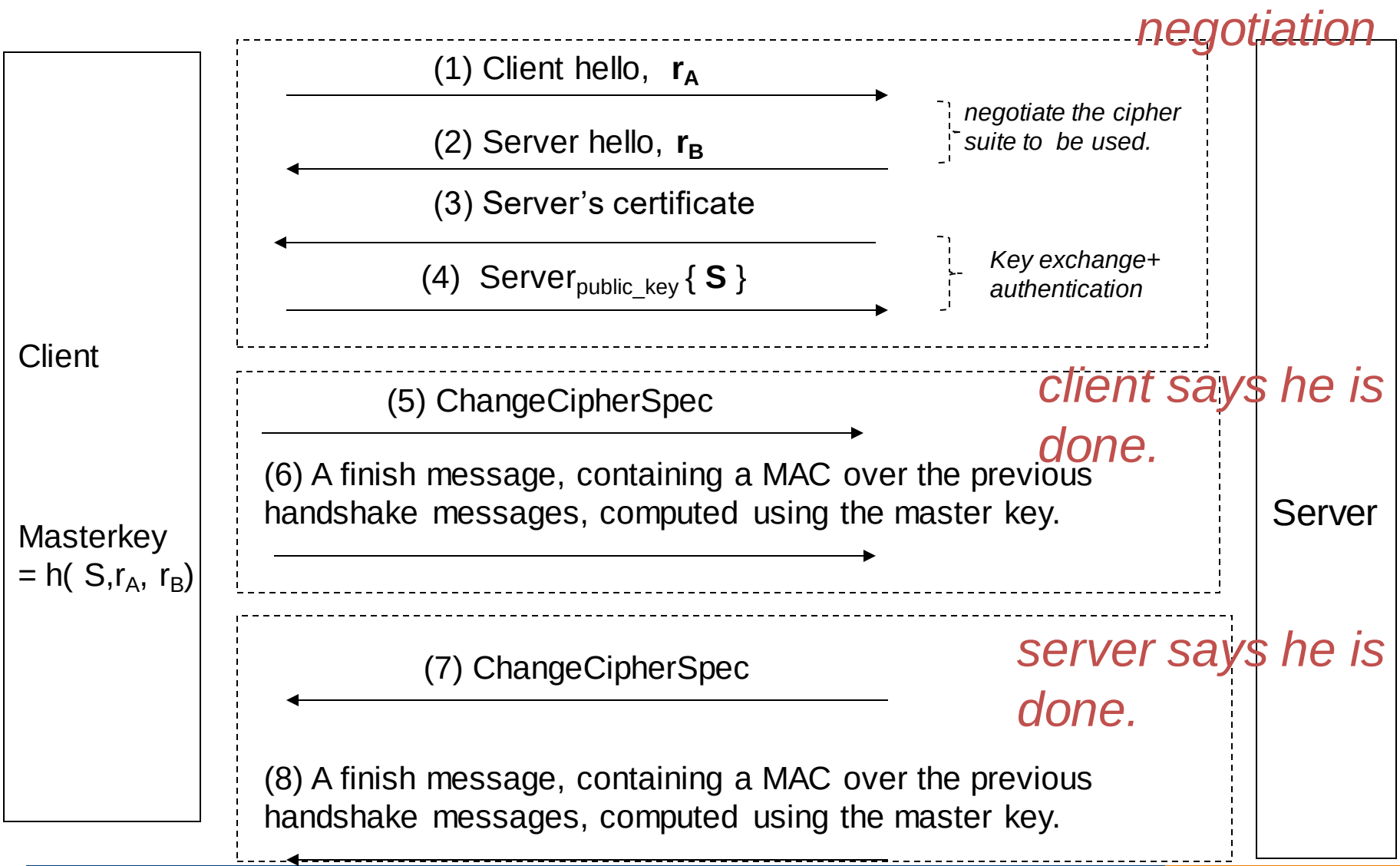


Telex: Overview



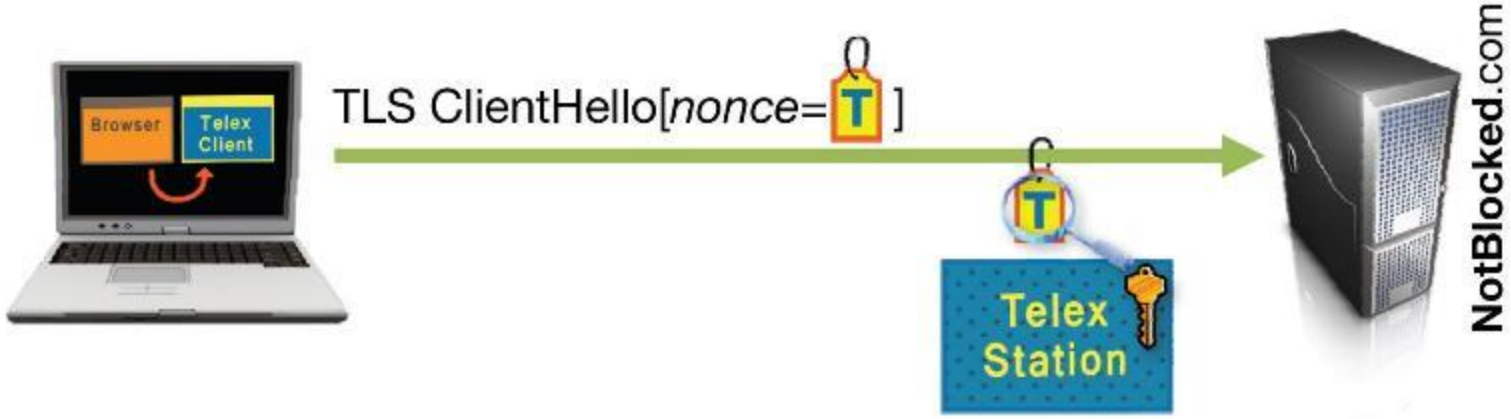
Recap: TLS handshake (up to TLS 1.2)

- The handshaking steps can be grouped into 3 phases



Details: Telex-TLS Handshake

1. **Client** starts TLS connection to **NotBlocked**.com



2. **Station** recognizes T using private key,
   but **Censor** can't tell from normal random nonce

Details: Telex-TLS Handshake

3. Client negotiates TLS session key with NotBlocked and leaks it to Station

- Tag communicates shared secret S to Station
- Client uses S in place of random coins for key generation
- Station simulates Client, derives same TLS key

Details: Telex-TLS Handshake

4. Station verifies Finished message from NotBlocked, switches from observer to MITM

5. Client sends encrypted request for blocked content
6. Station intercepts, decrypts, and proxies request

Details: Connection Tagging

Application of **public-key steganography**

- Client (anyone) generates tags
- Station (and only the station) detects tags

Our requirements:

- Short (28 bytes)
- Indistinguishable from random (for the censor)
- Conveys a shared secret
- Fast to recognize (for the station)
- Low false positives

Solution: Diffie-Hellman over elliptic curves … *with a twist!*

Tag generation

- Public: g, $\alpha = g^r$
- Context: X

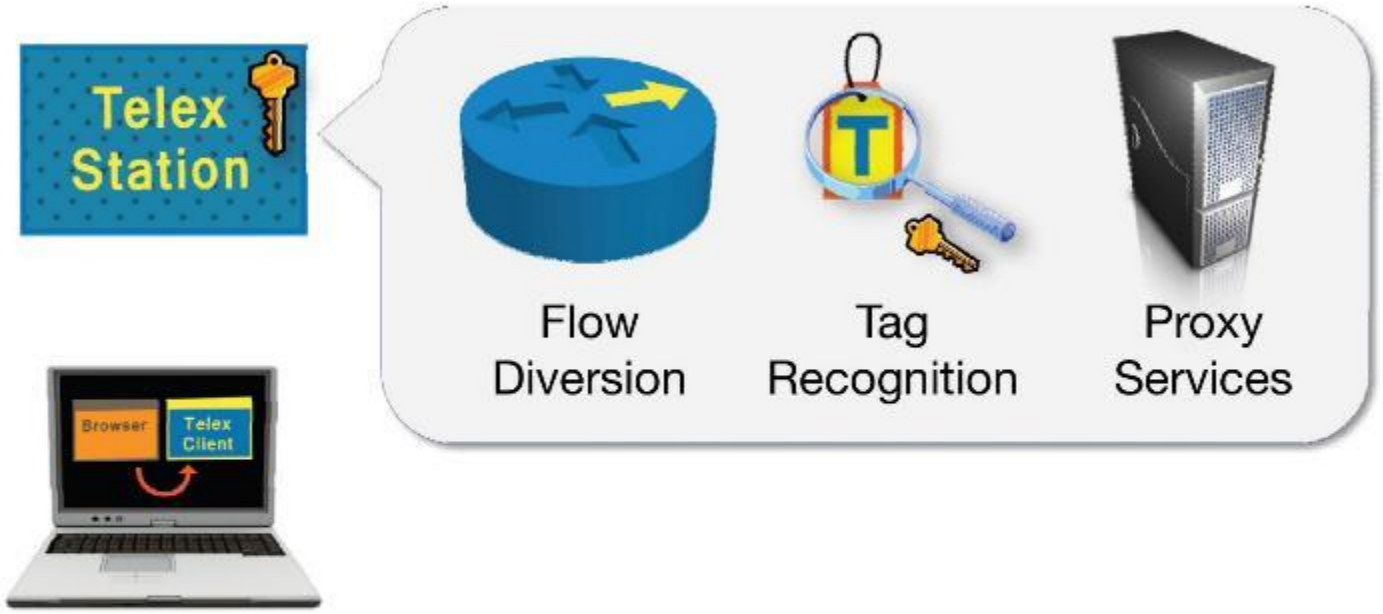
Nonce in ClientHello

- Telex client
  - Randomly pick s
  - Output $g^s \| H_1(\alpha^s \| X)$
  - Key ← $H_2(\alpha^s \| X)$

Seed for PRG to generate S

- Telex station
  - Private: r
  - Input $\beta \| h$
  - If h == $H_1(\beta^r \| X)$:
    - Key ← $H_2(\beta^r \| X)$
  - Else: do nothing

Details: Prototype Implementation



CAUTION Experimental proof-of-concept software.
Not safe for use under real-world censorship!

Prototype: Tag Recognition

Reconstructs TCP flows, extracts TLS nonces, etc.

Based on Bro for flow reconstruction, fast elliptic curve code
Checks 11,000 tags/second-core on 3GHz Intel Core 2 Duo

When tag found, commands router to drop flow,
then explicitly forwards packets until end of TLS handshake

300 SLOC Bro script; 450 SLOC C++

Prototype: Telex Client

Forwards arbitrary TCP port via tagged TLS connections

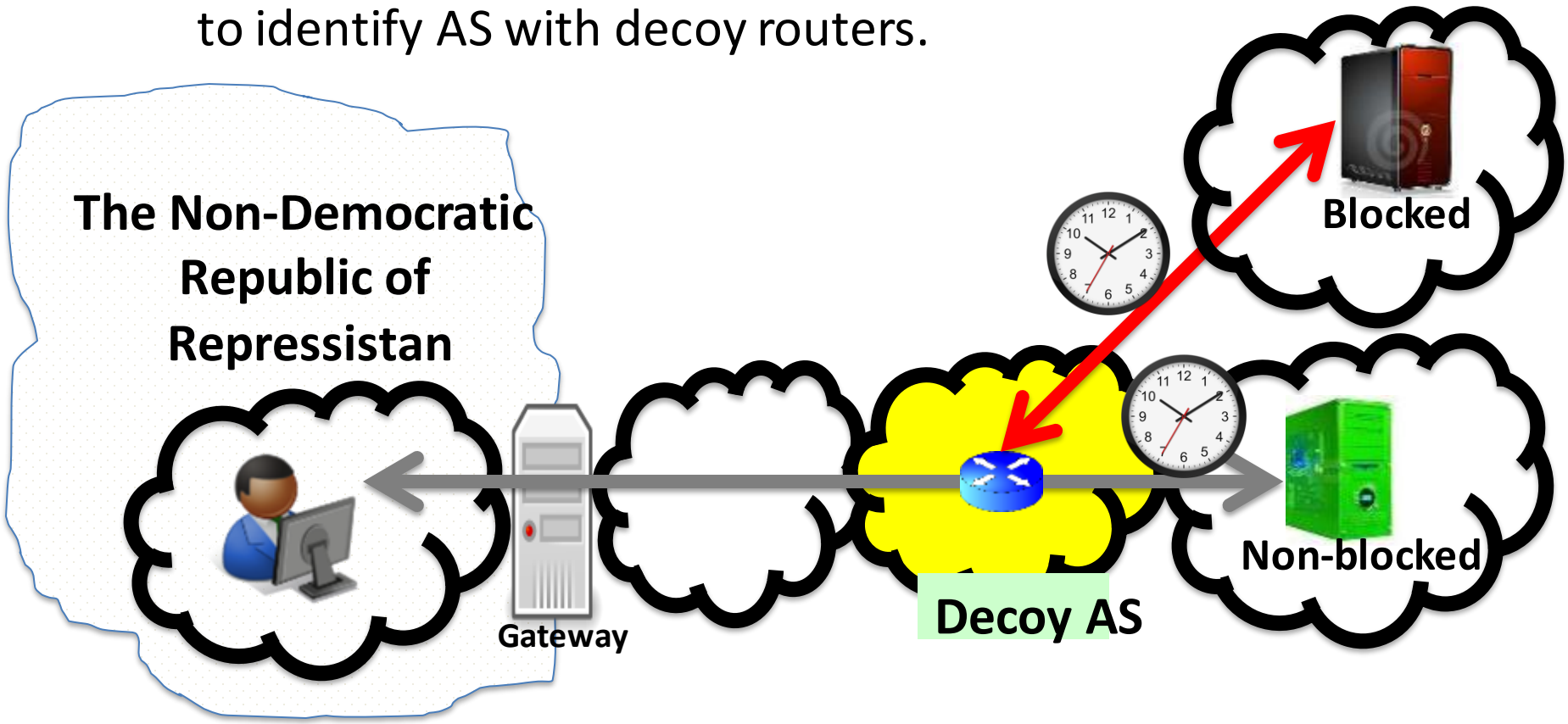
Based on libevent and (modified) OpenSSL

Currently Windows and Linux

1200 SLOC C++

Can censor counter Telex?

- Yes.
  - Telex stations are publicly known.
  - Even if there is no public list, censors could do many things to identify AS with decoy routers.

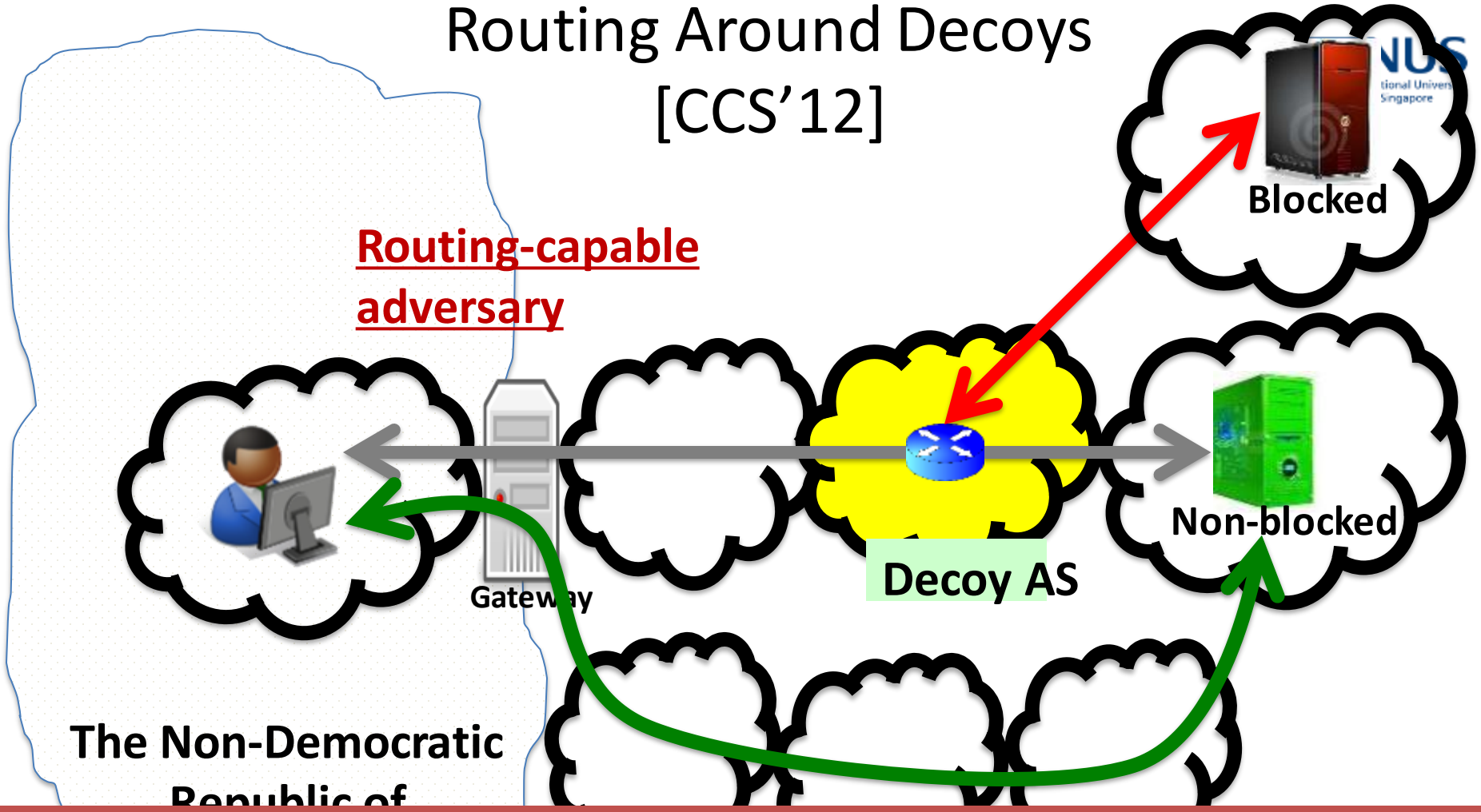


How to counter?

- Problem of decoy routing
  - Assume passive adversary only!
  - In practice, censors are active (and resourceful)

- How can the censor break decoy routing?
  - hint: Censorship authorities, in general, have control in many ISPs

Routing Around Decoys [CCS'12]

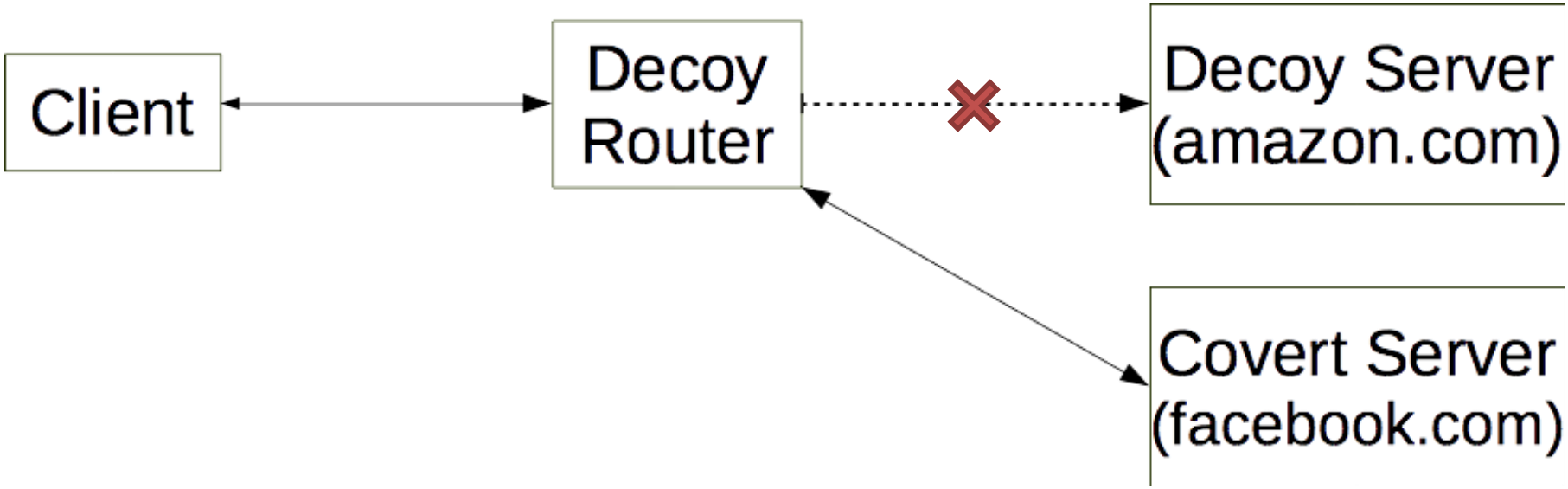


Decoy routing: a futile attempt?

- ✓ No, decoy routing still can be used because routing around decoys is *expensive* [NDSS14]

Practical deployment concerns to Friendly ISPs

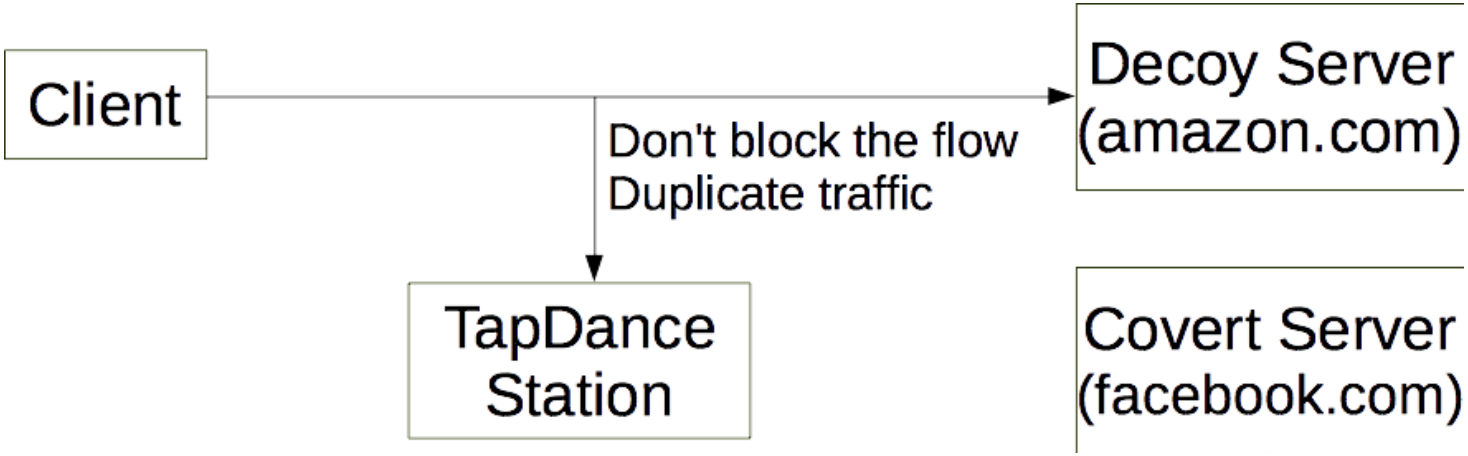
- Decoy routers (e.g., Telex station) *blocks* flows to decoy servers. Otherwise, (apparently) duplicated response from decoy server may trigger suspicion of the censor.
  - this requires ***in-line* blocking**, which is prohibited by ISPs



TapDance: Decoy routing with no blocking [UsenixSec14]

- Main difference of TapDance: it **doesn't block the flow.**
- But how?
  - Client voluntarily provide client-decoy key to station
  - Then, client sends incomplete HTTP request to decoy
  - Station proxies traffic to covert server; yet, still not blocking traffic to decoy
    - Decoy server will anyway ignore them due to TCP seq number mismatch

```
GET / HTTP/1.1\r\n
Host: www.site.com\r\n
X-Ignore: AAAAAAAAAAA…\r\n
```



ANTI-CENSORSHIP VIA USING OTHER PROTOCOLS

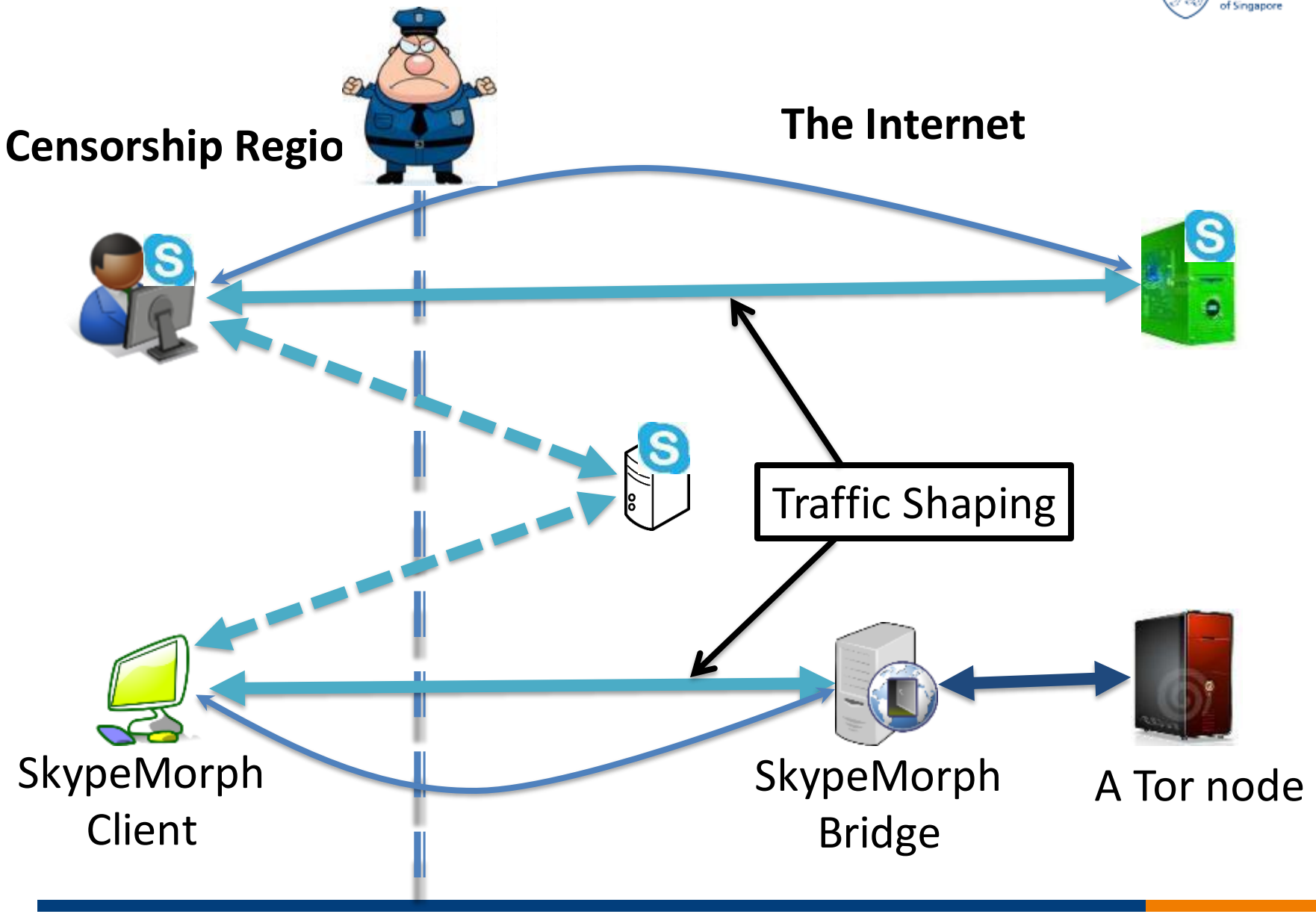
We need **unobservable circumvention**

Censors should not be able to identify circumvention traffic or end-hosts through passive, active, or proactive techniques

Parrot systems

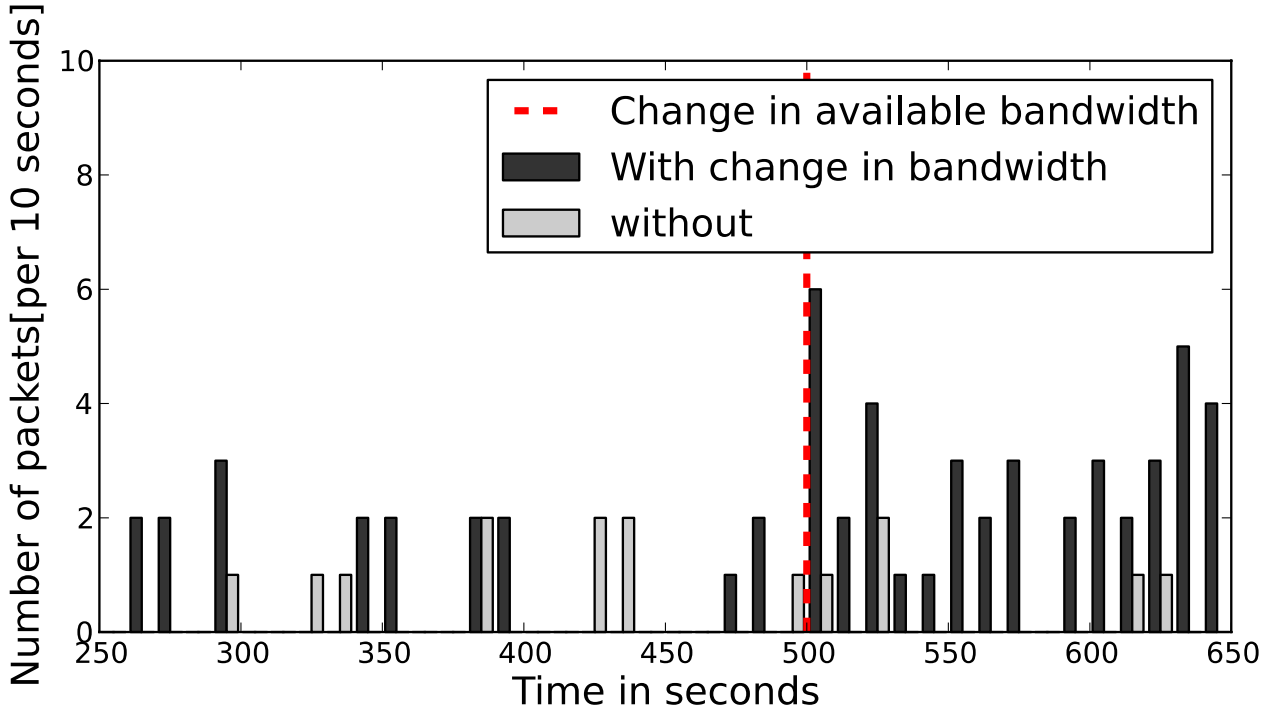
- ***Imitate*** a popular protocol
  - SkypeMorph (CCS'12)
  - StegoTorus (CCS'12)
  - CensorSpoofer (CCS'12)

SkypeMorph



Difficulty of imitation

- What if packet is dropped, reordered, delayed, etc.? Such events naturally occurs.

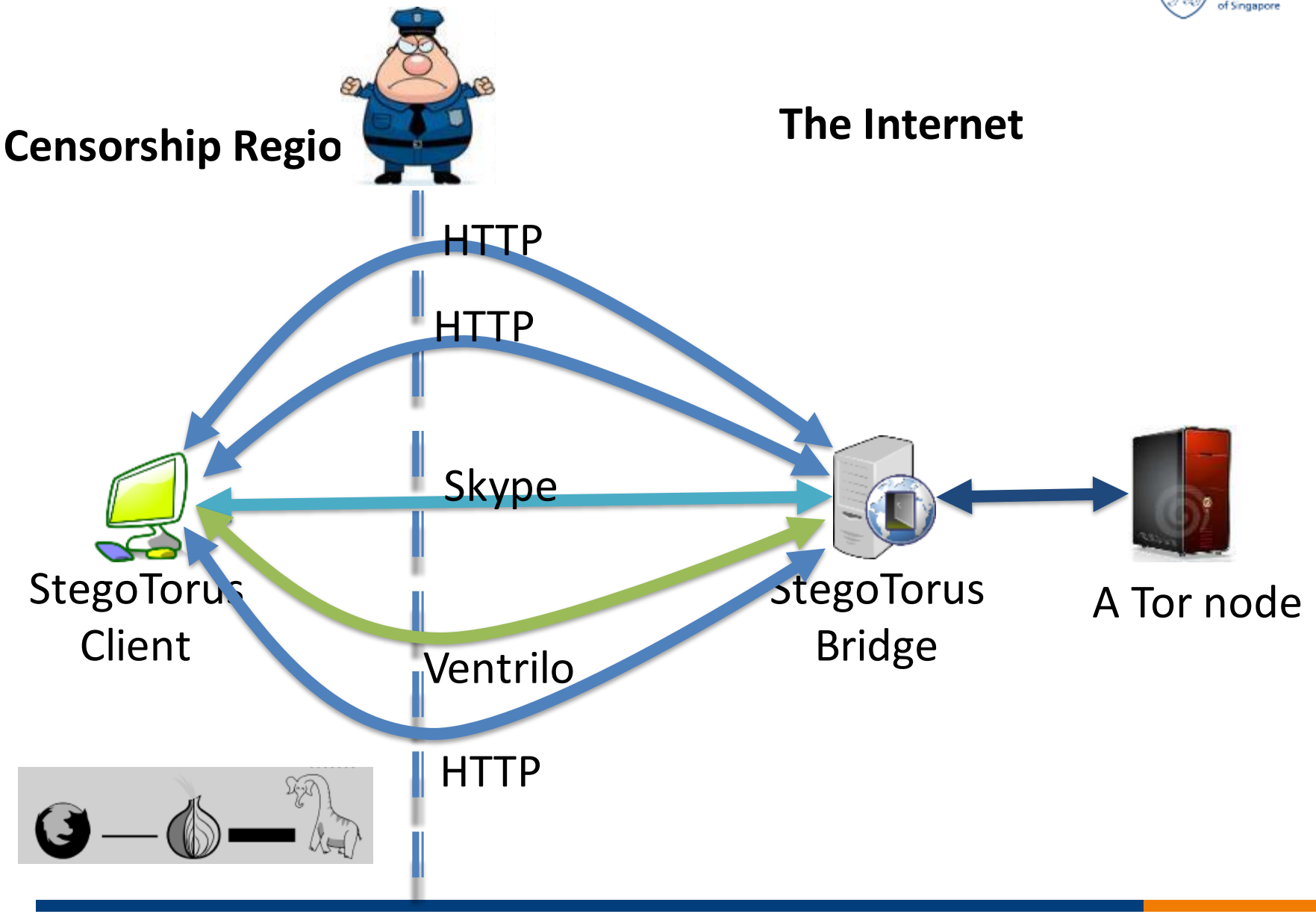


- Immediately after bandwidth is changed, the number of packets for true Skype client increase on its TCP control channel, but SkypeMorph does not implement the TCP control channel.

Other tests

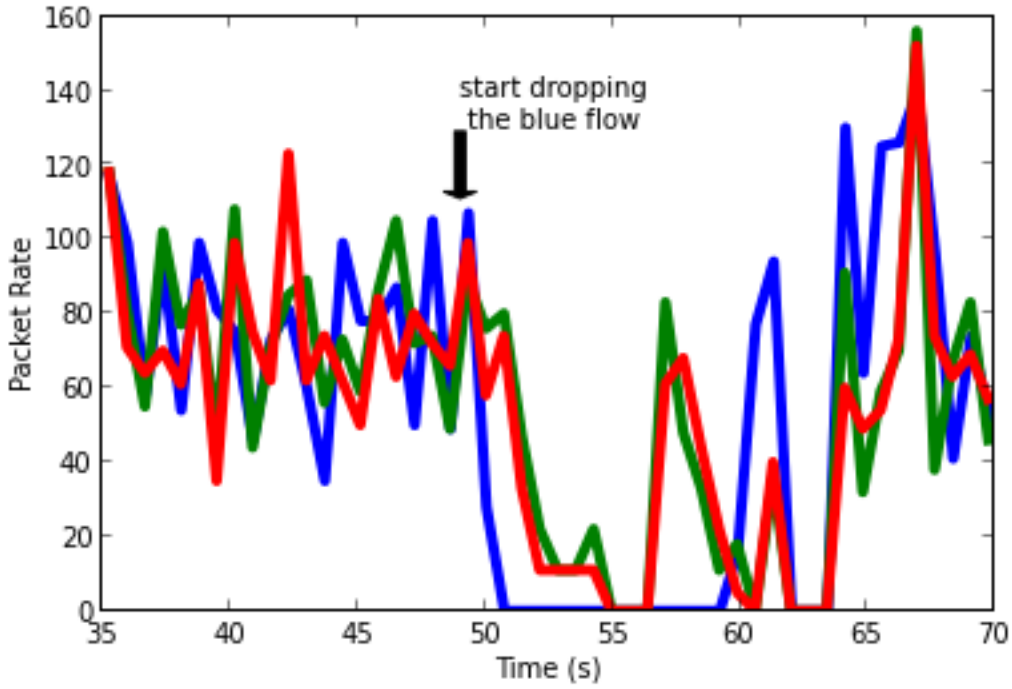
Test	Skype	SkypeMorph+
Flush Supernode cache	Serves as a SN	Rejects all Skype messages
Drop UDP packets	Burst of packets in TCP control	No reaction
Close TCP channel	Ends the UDP stream	No reaction
Delay TCP packets	Reacts depending on the type of message	No reaction
Close TCP connection to a SN	Initiates UDP probes	No reaction
Block the default TCP port	Connects to TCP ports 80 and 443	No reaction

StegoTorus



StegoTorus is unobservable?

- Dependencies between links
  - “once packets on one StegoTorus connection are dropped, the other two belonging to the same link immediately slow down.”



StegoTorus-HTTP

- Does not look like a typical HTTP server!
- Most HTTP methods not supported!

HTTP request	Real HTTP server	StegoTorus's HTTP module
GET existing	Returns "200 OK" and sets `Connection` to `keep-alive`	Arbitrarily sets `Connection` to either `keep-alive` or `Close`
GET long request	Returns "404 Not Found" since URI does not exist	No response
GET non-existing	Returns "404 Not Found"	Returns "200 OK"
GET wrong protocol	Most servers produce an error message, e.g., "400 Bad Request"	Returns "200 OK"
HEAD existing	Returns the common HTTP headers	No response
OPTIONS common	Returns the supported methods in the Allow line	No response
DELETE existing	Most servers have this method not activated and produce an error message	No response
TEST method	Returns an error message, e.g., "405 Method Not Allowed" and sets `Connection=Close`	No response
Attack request	Returns an error message, e.g., "404 Not Found"	No response

Lesson 1

Unobservability by imitation is fundamentally flawed!

Imitation Requirements

Correct	**SideProtocols**
IntraDepend	**InterDepend**
Err	**Network**
Content	**Patterns**
Users	**Geo**
Soft	**OS**

Lesson 2

Partial imitation is worse than no imitation!

(Often easier than mounting traffic analysis on TOR)

Alternatively, running original protocols? e.g., FreeWave [NDSS13]: IP over Voice-over-IP

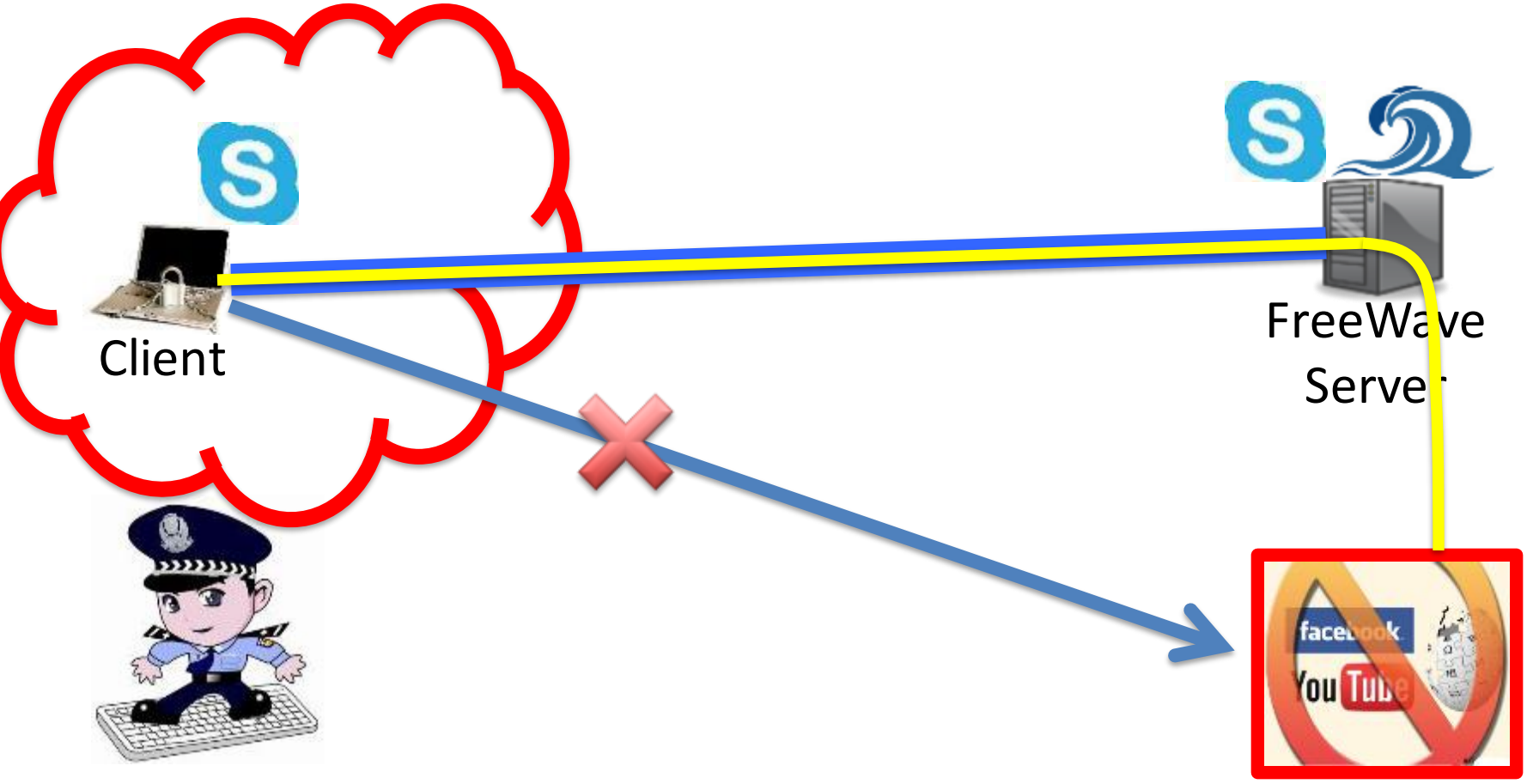
- Target protocol: Voice-over IP (VoIP)

- Why VoIP
  - Widely used protocol (only 663 Million Skype users)
    - Collateral damage to block
  - Encrypted

- How to hide?
  - The dial-up modems are back!

http://dedis.cs.yale.edu/dissent/papers/freewave-slides.pptx

FreeWave architecture



http://dedis.cs.yale.edu/dissent/papers/freewave-slides.pptx

Basic components

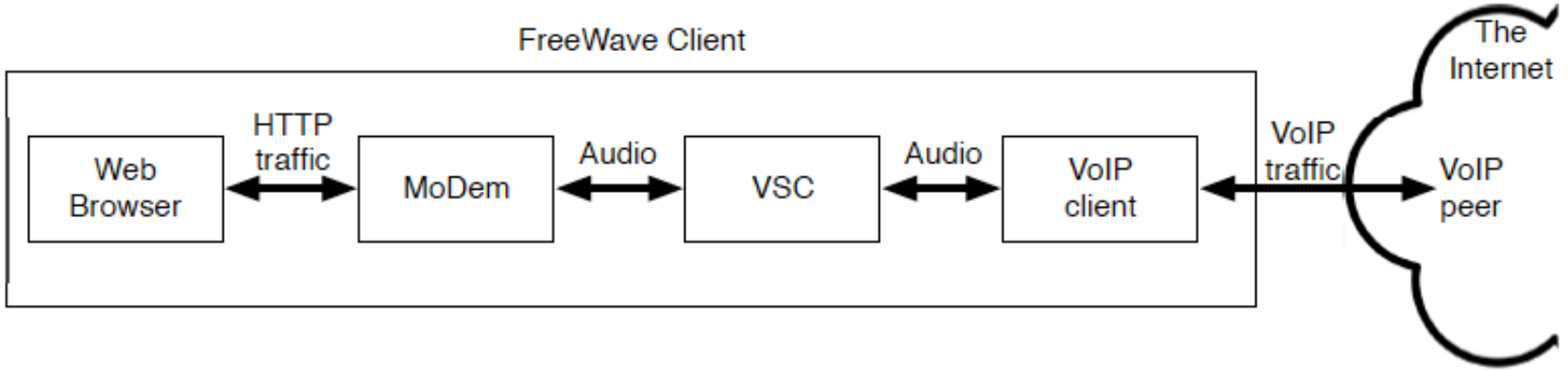


Figure 2. The main components of FreeWave client.

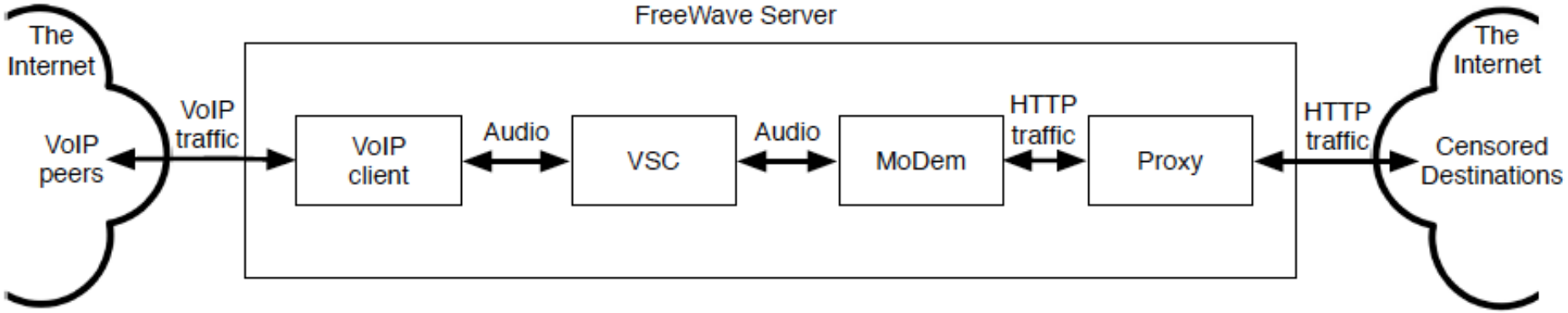


Figure 3. The main components of FreeWave server.

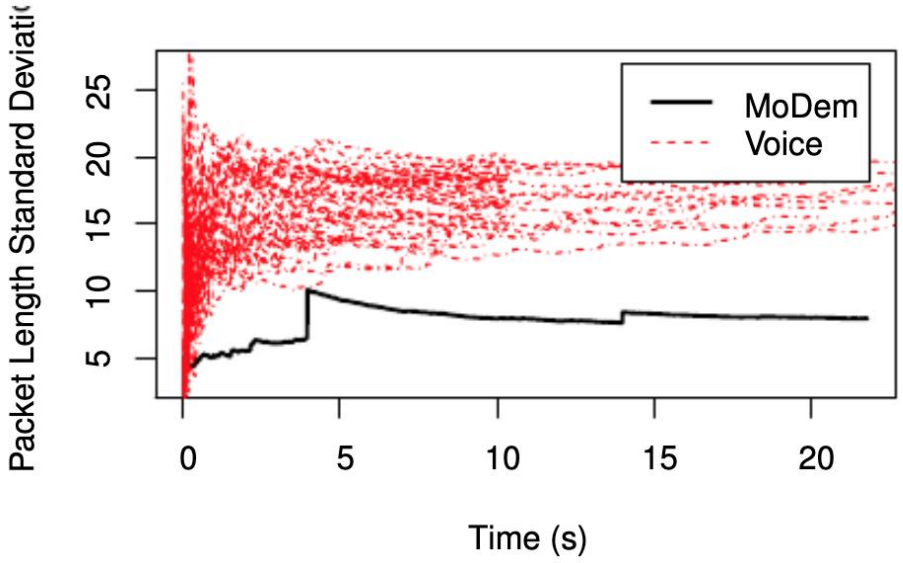
http://dedis.cs.yale.edu/dissent/papers/freewave-slides.pptx

Is FreeWave unobservable?

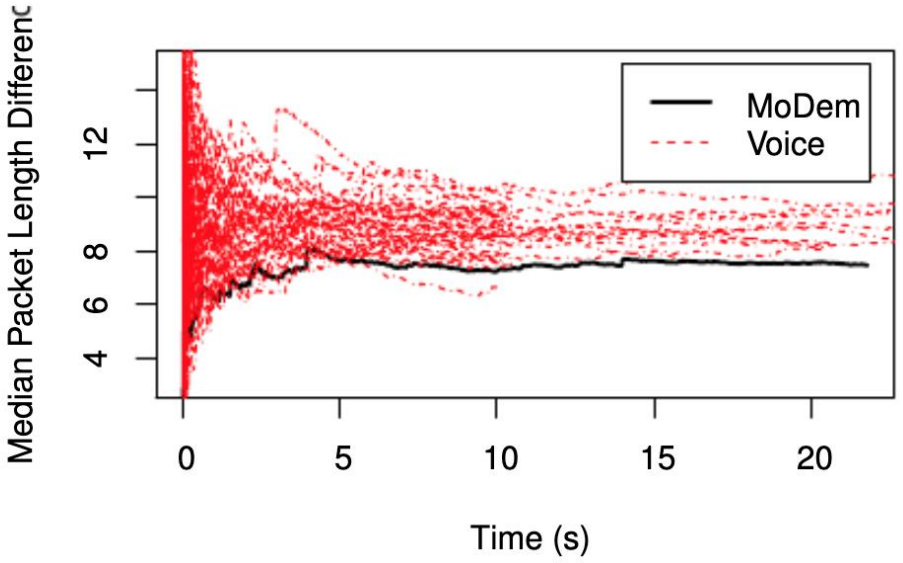
- Unfortunately, it has been shown to be observable!
- How?
  - The cover protocol (e.g., Skype) and the proxy protocol (e.g., web proxy) have mismatches!

Cover Your ACKs [CCS'13]

- **Differential error tolerance**
  - Skype UDP connections are error tolerant
  - Few packet losses (e.g., 5%) are unnoticeable in Skype
  - However, few losses can completely block data traffic
- **Different packet length distribution**



(a) Packet length standard deviation over time



(b) Average Packet Differences Over Time

Summary

- Anonymity is useful but ***insufficient*** against censorship
- Internet censorship is widely deployed in today's Internet
- Many anti-censorship techniques have been proposed:
  - Imitating protocols
  - Running cover protocols
  - Decoy routings
- In practice: must pay extra attention when using anti-censorship tools; particularly, when developing such tools!

Questions?

NEXT WEEK:
BLOCKCHAIN SECURITY

Two papers to read

- [Paper 1] "Hijacking Bitcoin: Routing Attacks on Cryptocurrencies", Apostolaki et al. (IEEE S&P'17)

- [Paper 2] "A Stealthier Partitioning Attack against Bitcoin Peer-to-Peer Network", Tran et al. (IEEE S&P'20)