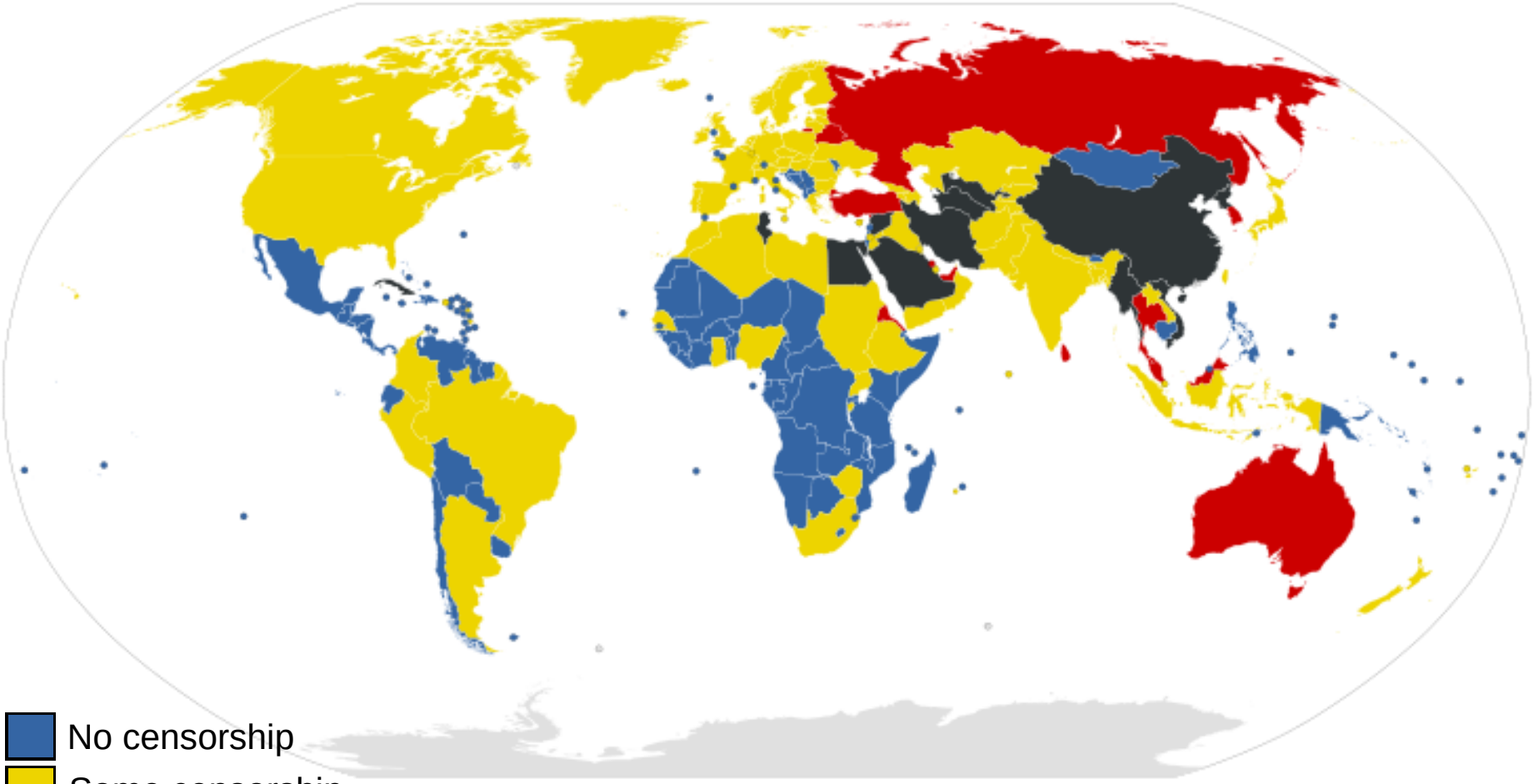
Telex | Anticensorship in the Network Infrastructure

Eric Wustrow
Scott Wolchok
Ian Goldberg*
J. Alex Halderman

University of Michigan
*University of Waterloo

In *Proceedings of the 20th USENIX Security Symposium* (2011)

Background | **Internet Censorship**



- No censorship
- Some censorship
- Country under surveillance from Reporters Without Borders
- Most heavily censored nations

Background | **Network-based Censorship**

Government censors

Block websites containing “offensive” content

Commonly employ blacklist approach

Observed techniques

IP blocking, DNS spoofing, forged RST packets

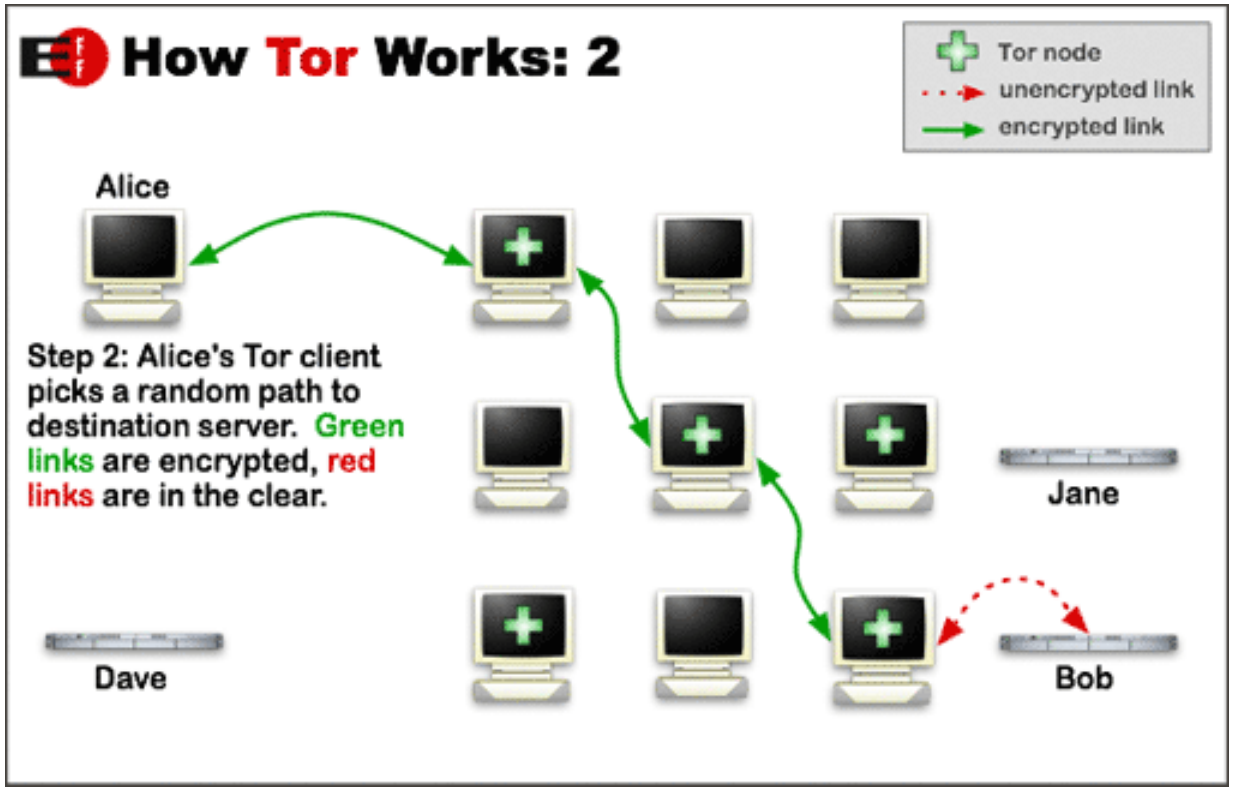
Popular countermeasures

Mostly proxy based — Tor, Freenet, Ultrasurf, …

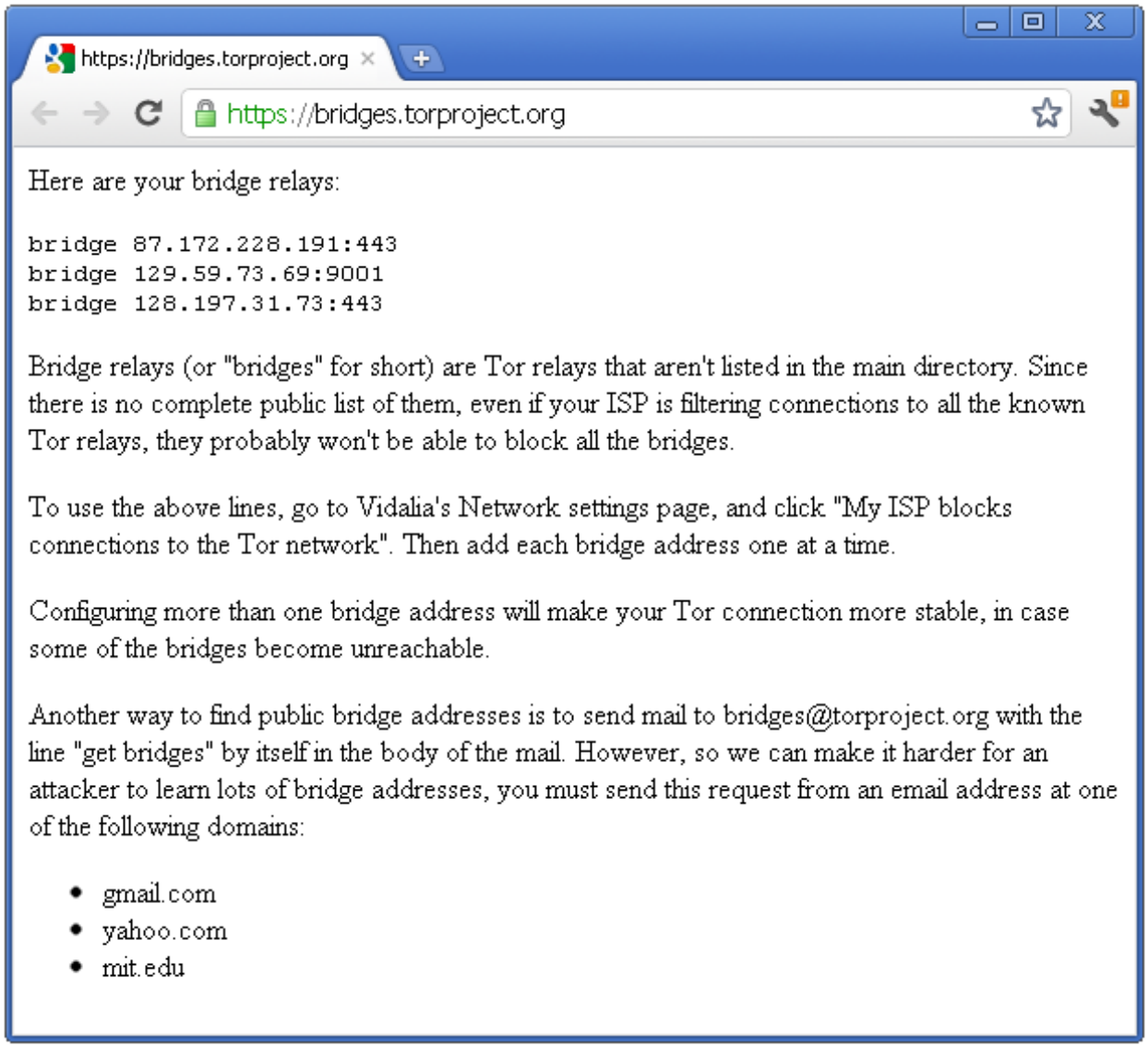
Problem: Cat-and-mouse game

Need to communicate proxy addresses to users ***but not to censors***, or else they’ll be blocked too!

Previous Work | **Tor**



Previous Work | **Tor Bridges**



Our Approach | **Telex**

Operates in the **network infrastructure**

- Components placed at ISP between the censor's network and non-blocked portions of the Internet.
- We call this end-to-middle proxying

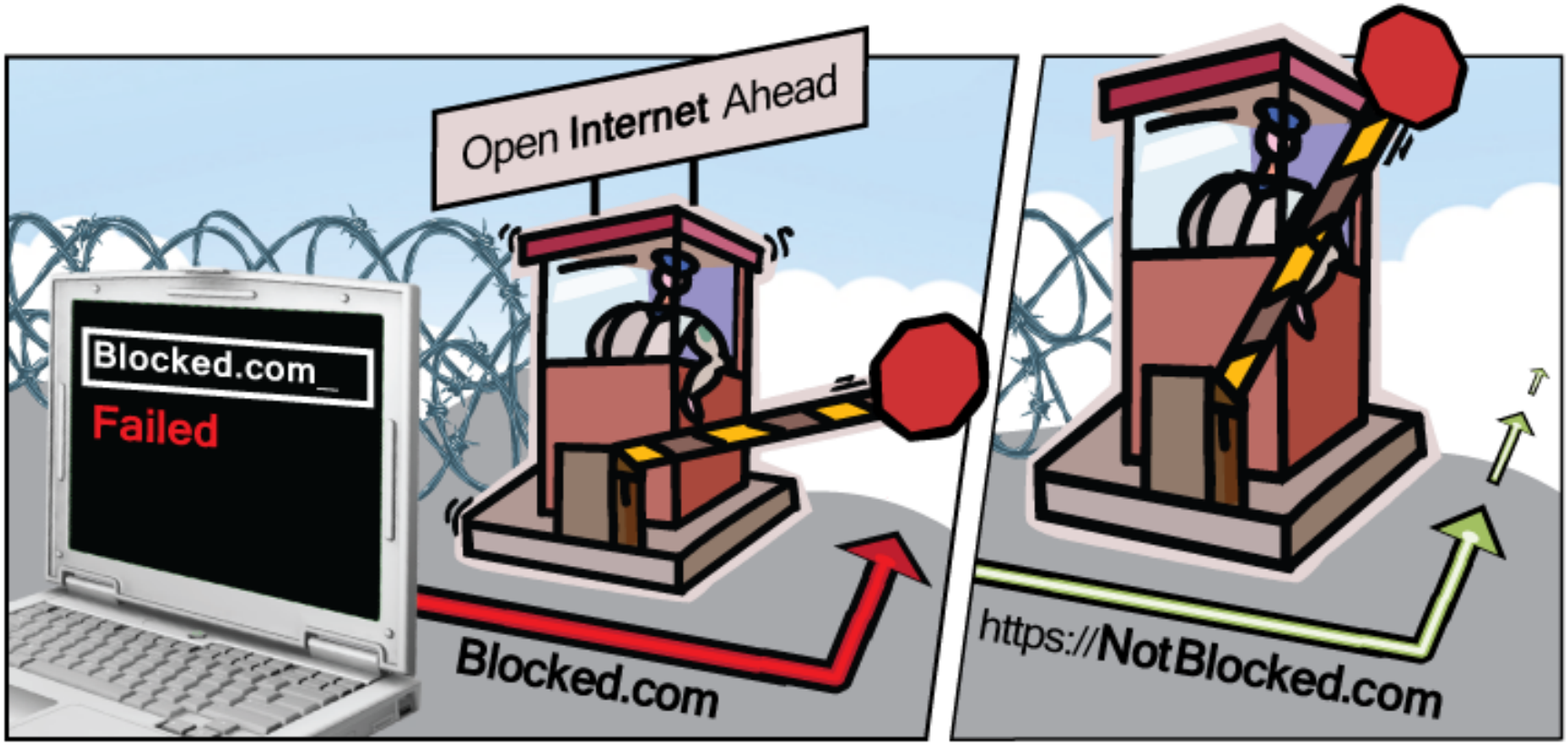
Focuses on **avoiding detection** by the censor

- Complements anonymity systems such as Tor

Employs a form of **deep-packet inspection**

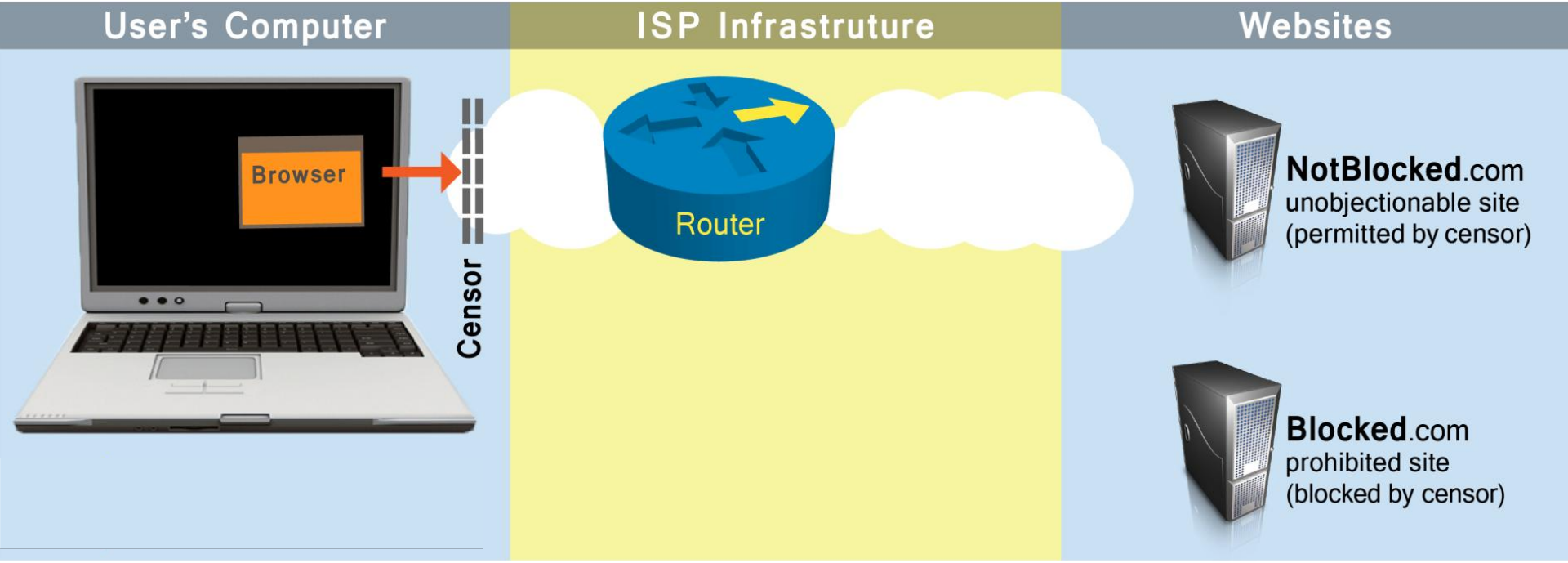
Has **no secrets** to communicate to users in advance

Telex | **Threat Model**



Censor … controls client's network, but not external network
… blocks according to a blacklist
… allows HTTPS connections to non-blocked sites

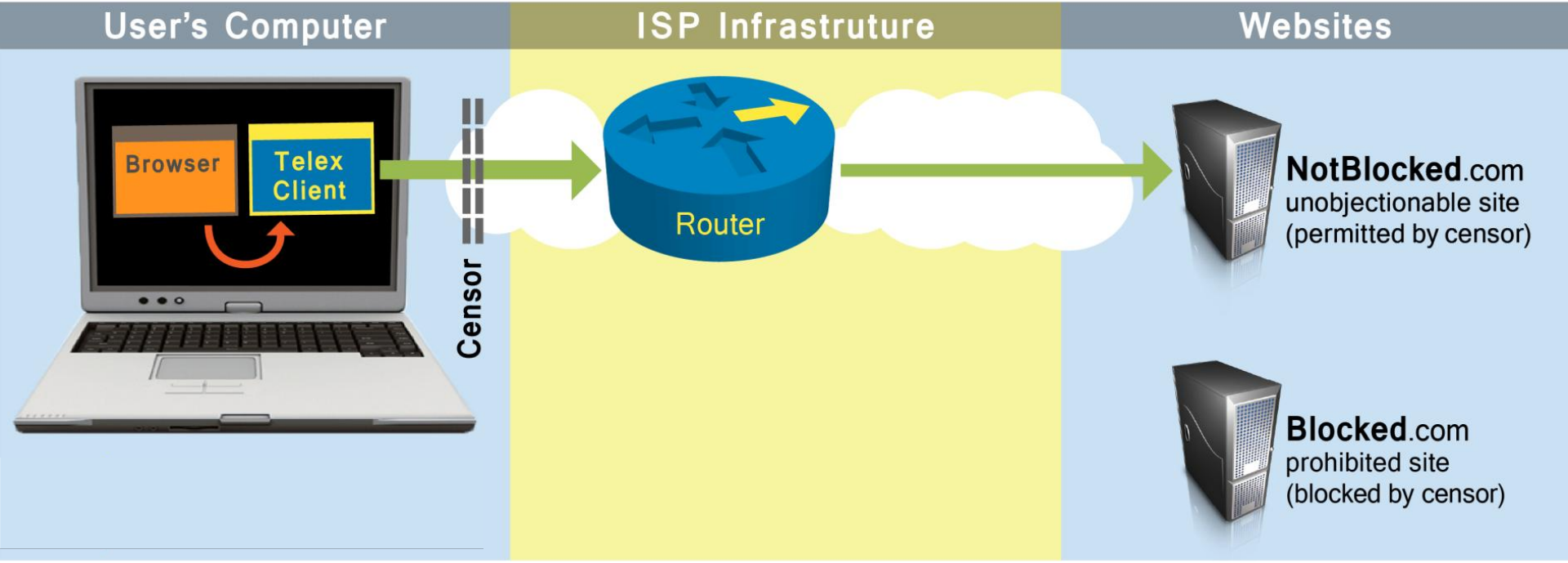
Telex | **Overview**



Request for **permitted** site

Request for **prohibited** site

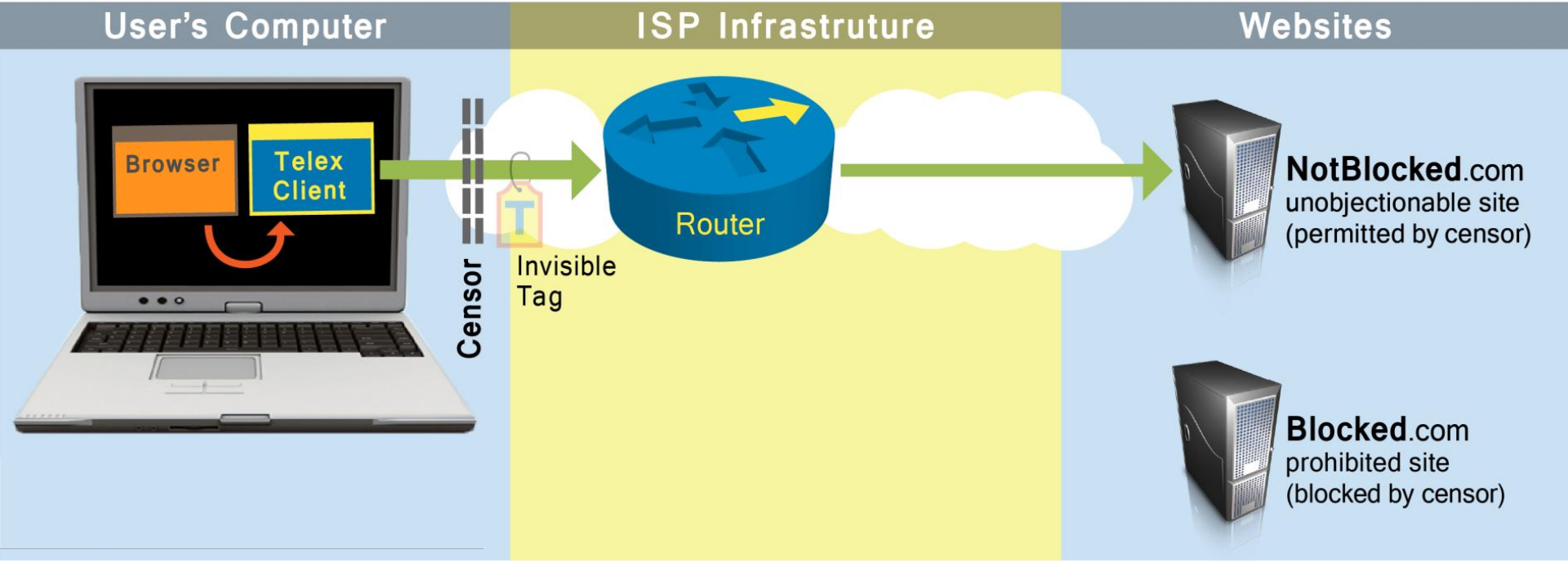
Telex | **Overview**



Request for **permitted** site

Request for **prohibited** site

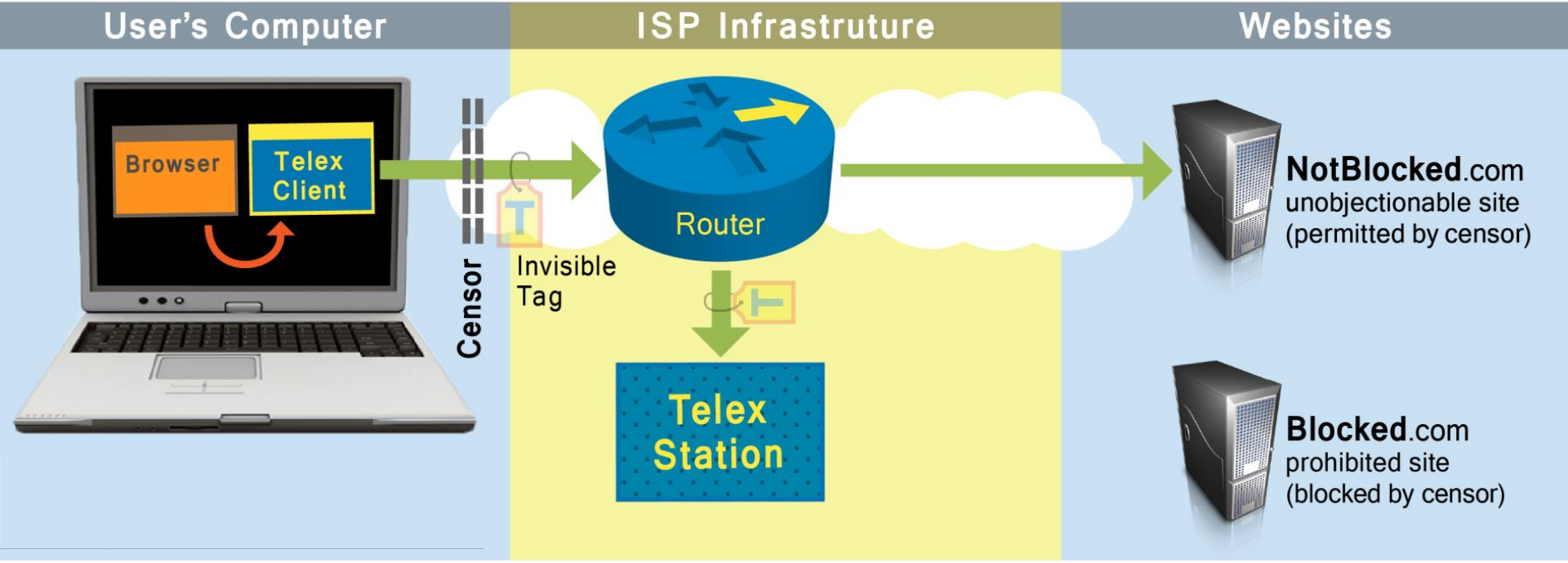
Telex | **Overview**



Request for **permitted** site

Request for **prohibited** site

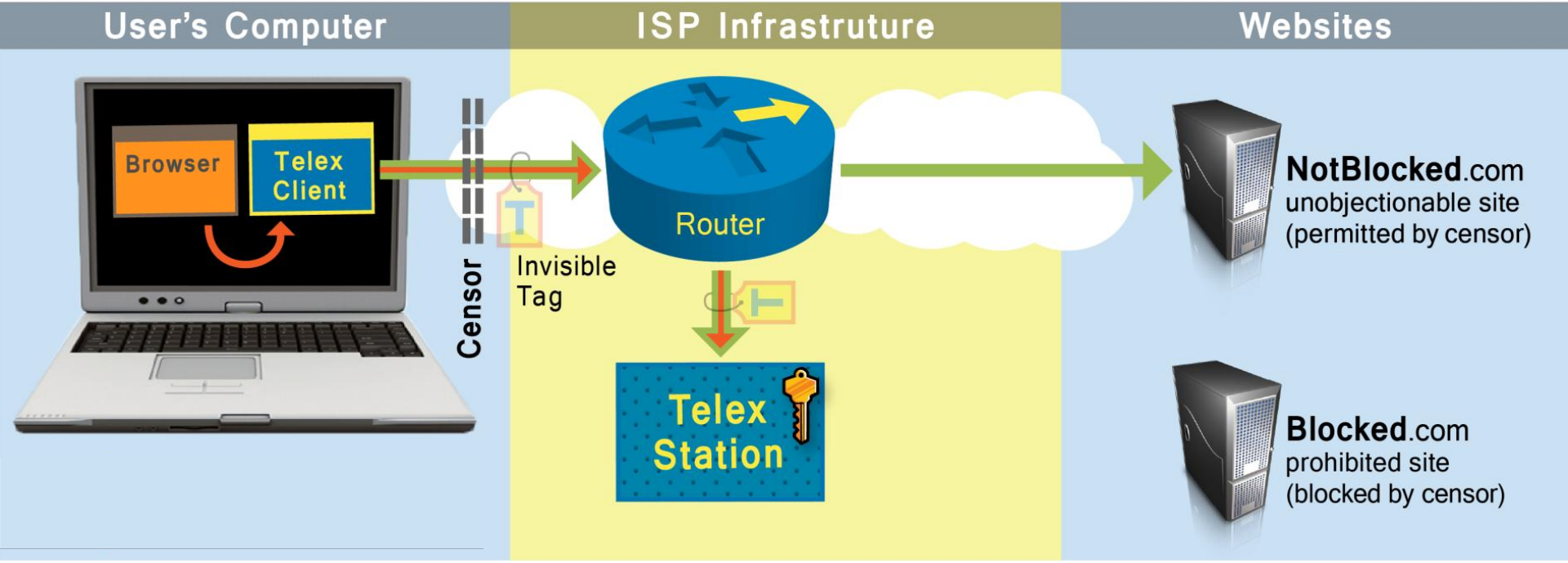
Telex | **Overview**



Request for **permitted** site

Request for **prohibited** site

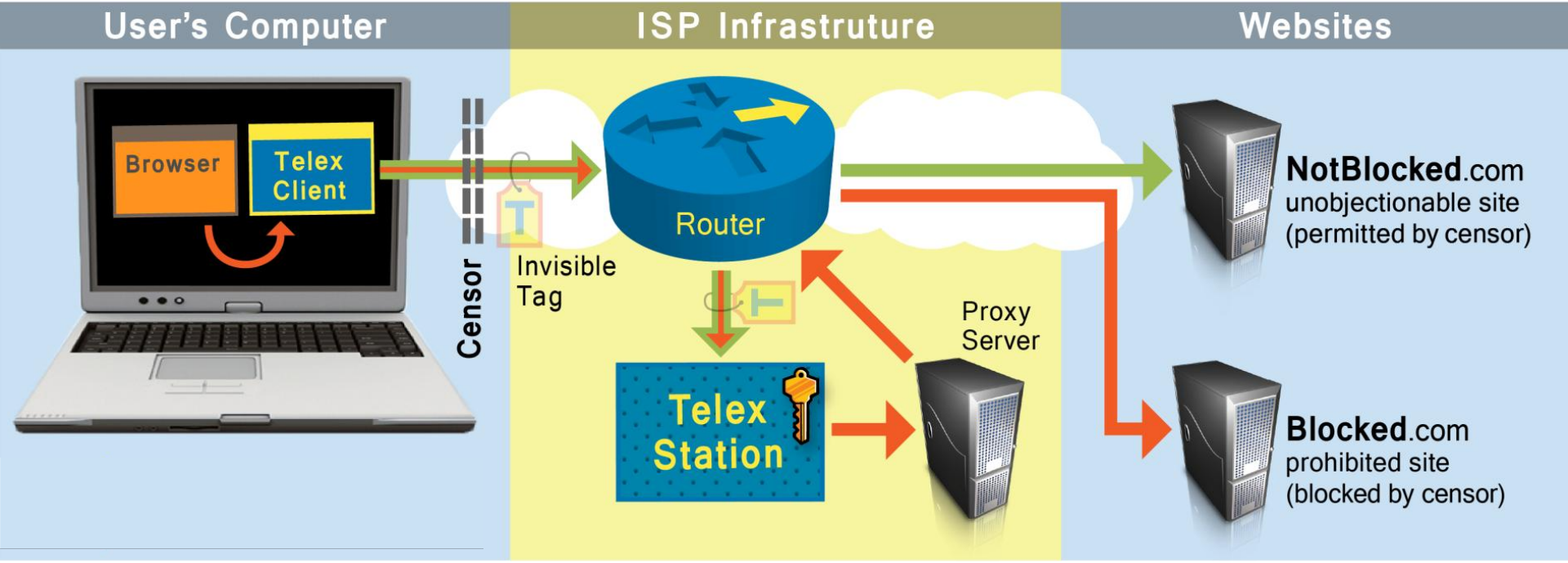
Telex | **Overview**



Request for **permitted** site

Request for **prohibited** site

Telex | **Overview**



Request for **permitted** site

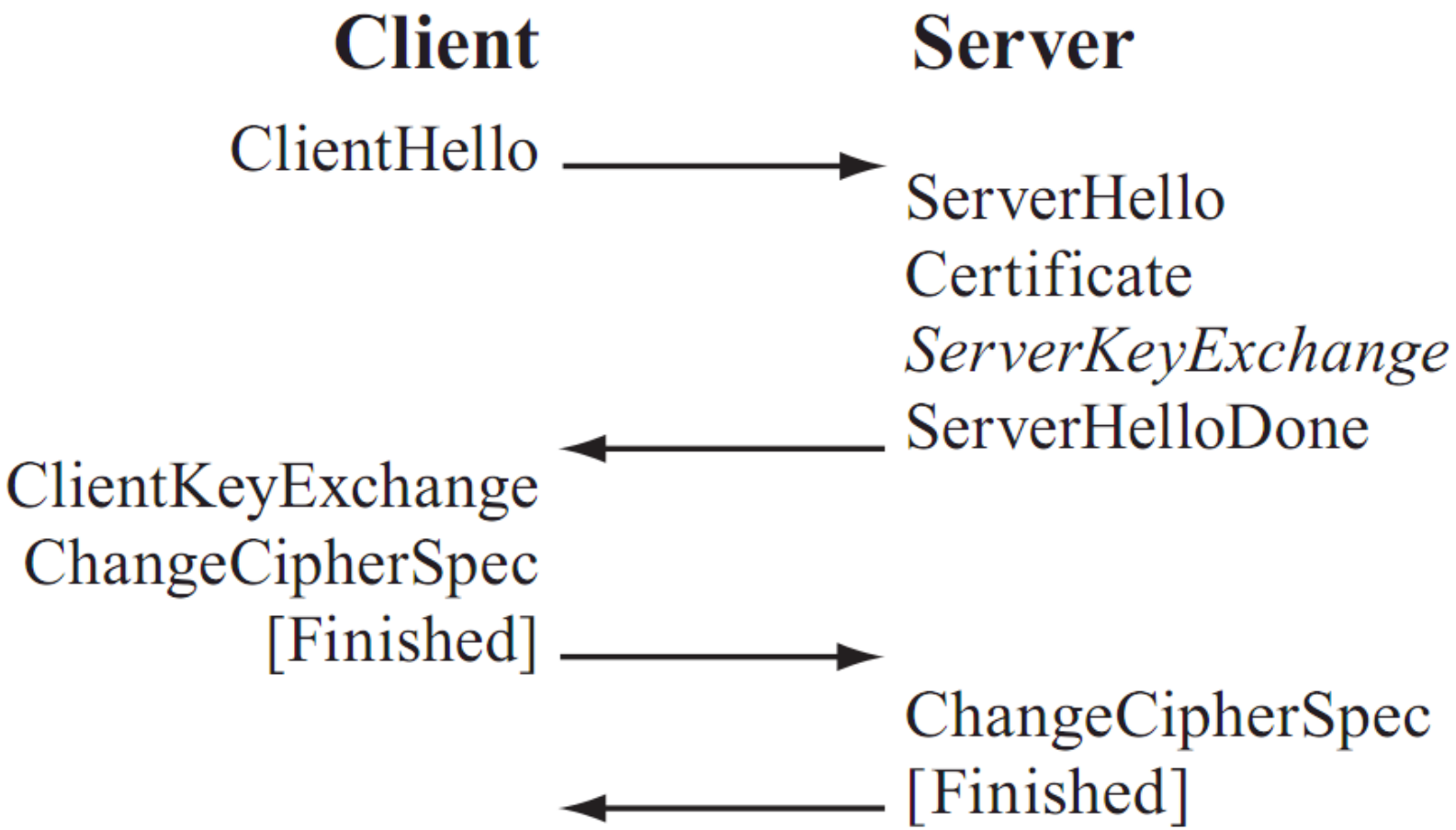
Request for **prohibited** site

Discussion | **ISP Cooperation**

Why would an ISP deploy Telex?

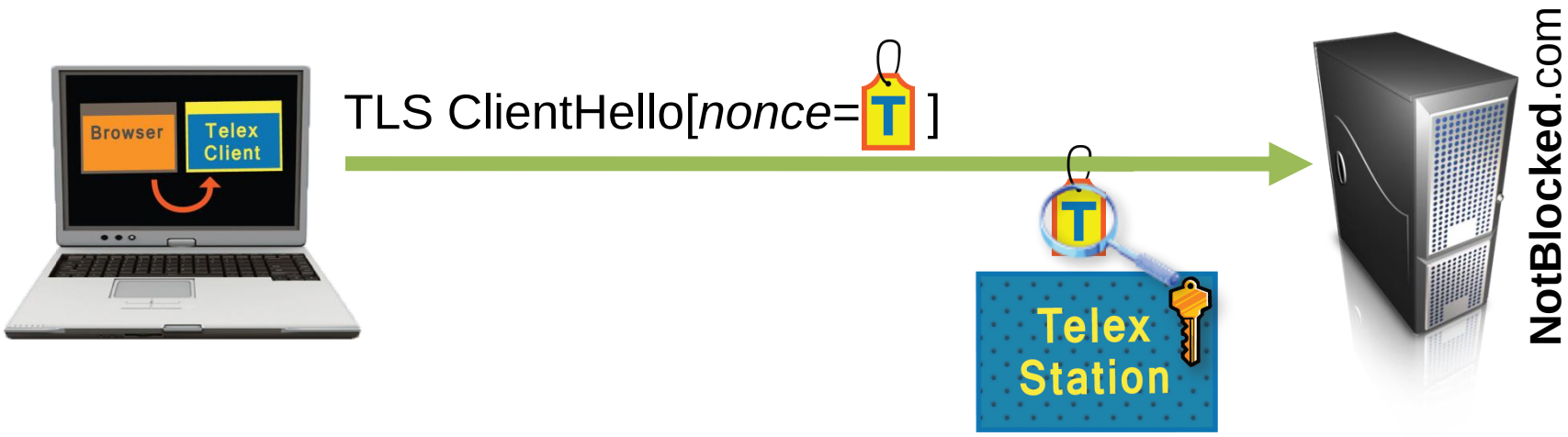
- Advance research
- Government Incentive$
- ISPs in a unique position to help

Details | **TLS handshake overview**



Details | **Telex-TLS Handshake**

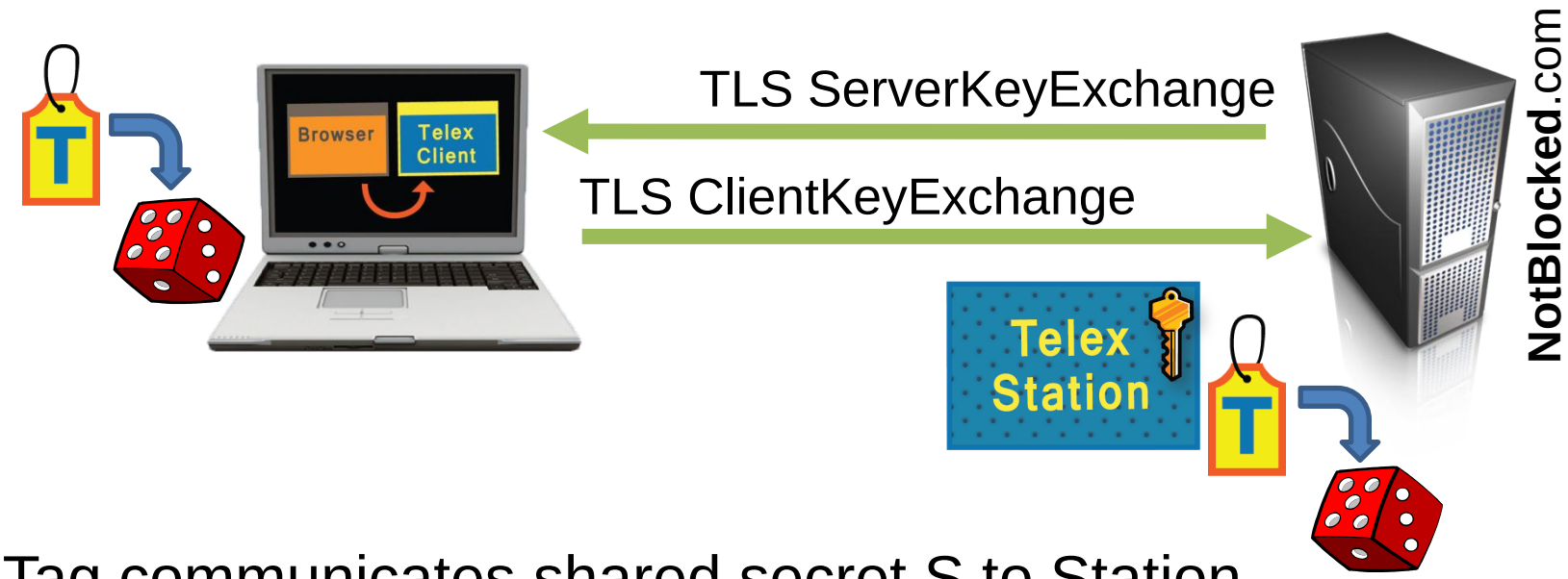
1. **Client** starts TLS connection to **NotBlocked**.com



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**

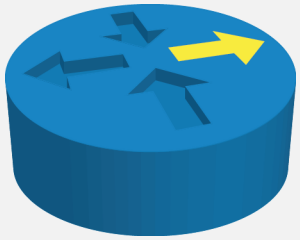
- Client (anyone) generates tags
- Station (and only the station) detects tags

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!*

Telex | **Prototype Implementation**

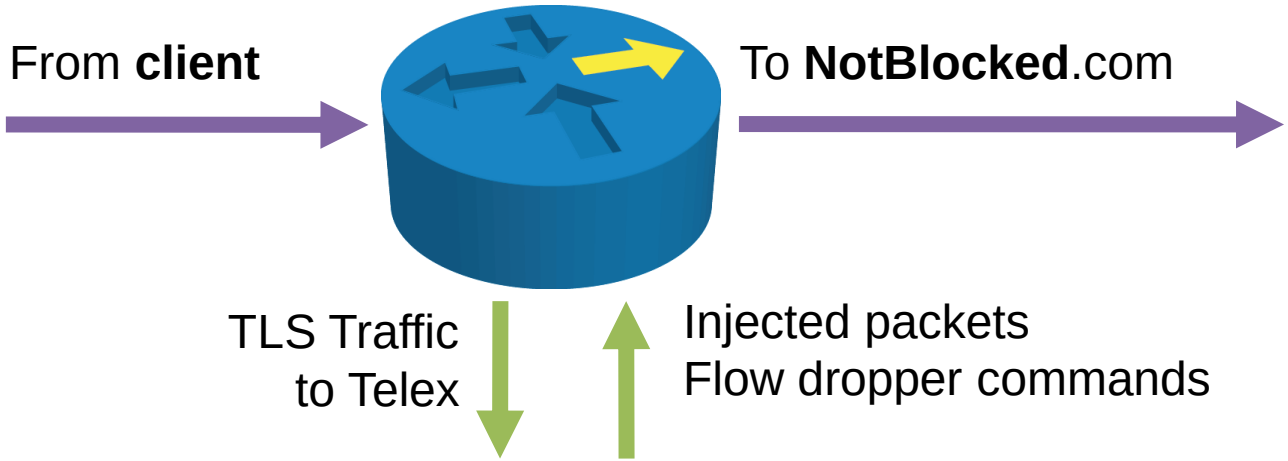


Flow Diversion

Tag Recognition

Proxy Services

Prototype | **Flow Diversion**



Inline router capable of dropping flows on command
(e.g. “stop automatically forwarding for client ↔NotBlocked.com”)

If DPI gets overloaded, router still forwards normal traffic

Prototype | **Tag Recognition**

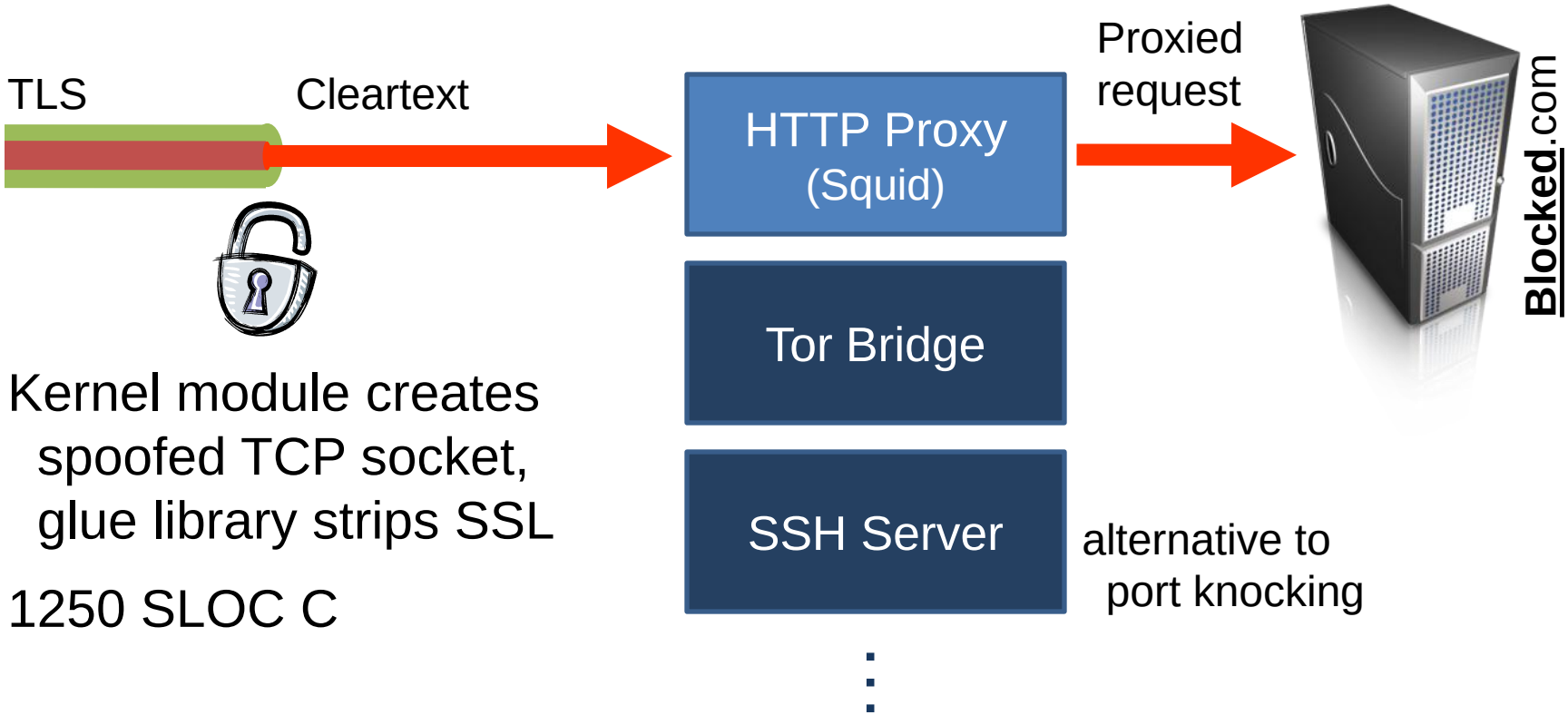
Reconstructs TCP flows, extracts TLS nonces, etc.

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

Prototype | **Proxy Service**

Shunts data between client's TLS connection and configurable services



Kernel module creates spoofed TCP socket, glue library strips SSL

1250 SLOC C

Prototype | **Telex Client**

Forwards arbitrary TCP port via tagged TLS connections

Based on libevent and (modified) OpenSSL

Currently for Windows and Linux

Prototype | **Test Deployment**

Single Telex Station on lab-scale “ISP” at Michigan

Hosted sites

NotBlocked.telex.cc
Unobjectionable content *

Blocked.telex.cc
Simulated censored site
only reachable via Telex

Early experiences

Three authors used Telex for daily browsing for a few months

Streamed HD YouTube via PlanetLab node in Beijing

Ultimately NotBlocked.telex.cc was blocked in China, but a real deployment would have more than one NotBlocked

NotBlocked.telex.cc

https://notblocked.telex.cc

NotBlocked.telex.cc Try **Blocked**.telex.cc, a "censored" site. You can only access it via Telex.

Eric says: Kittens are cute!

Prototype | **Users**

First 3 weeks | July 18—August 11, 2011

Next Steps | **We need your help**

Seeking ISP partners for next research phase: prototype deployment

Place operational station on (big) pipes at (real) partner ISPs

Planned architecture: Give us a :443 tap and an inline OpenFlow switch

Help us engineer performance and reliability at scale

Pioneer operational details and policies

Telex | **Conclusion**

End-to-middle proxying—
New approach to resisting Internet censorship

Focus on hiding use of the service

Based on public-key steganography,
repurposes DPI and MITM for ***anti***censorship

Proof-of-concept operating today, but wide-scale deployment needs ISP cooperation

Telex | Anticensorship in the Network Infrastructure

https://**telex.cc**

Eric Wustrow
Scott Wolchok
Ian Goldberg*
J. Alex Halderman

University of Michigan
*University of Waterloo