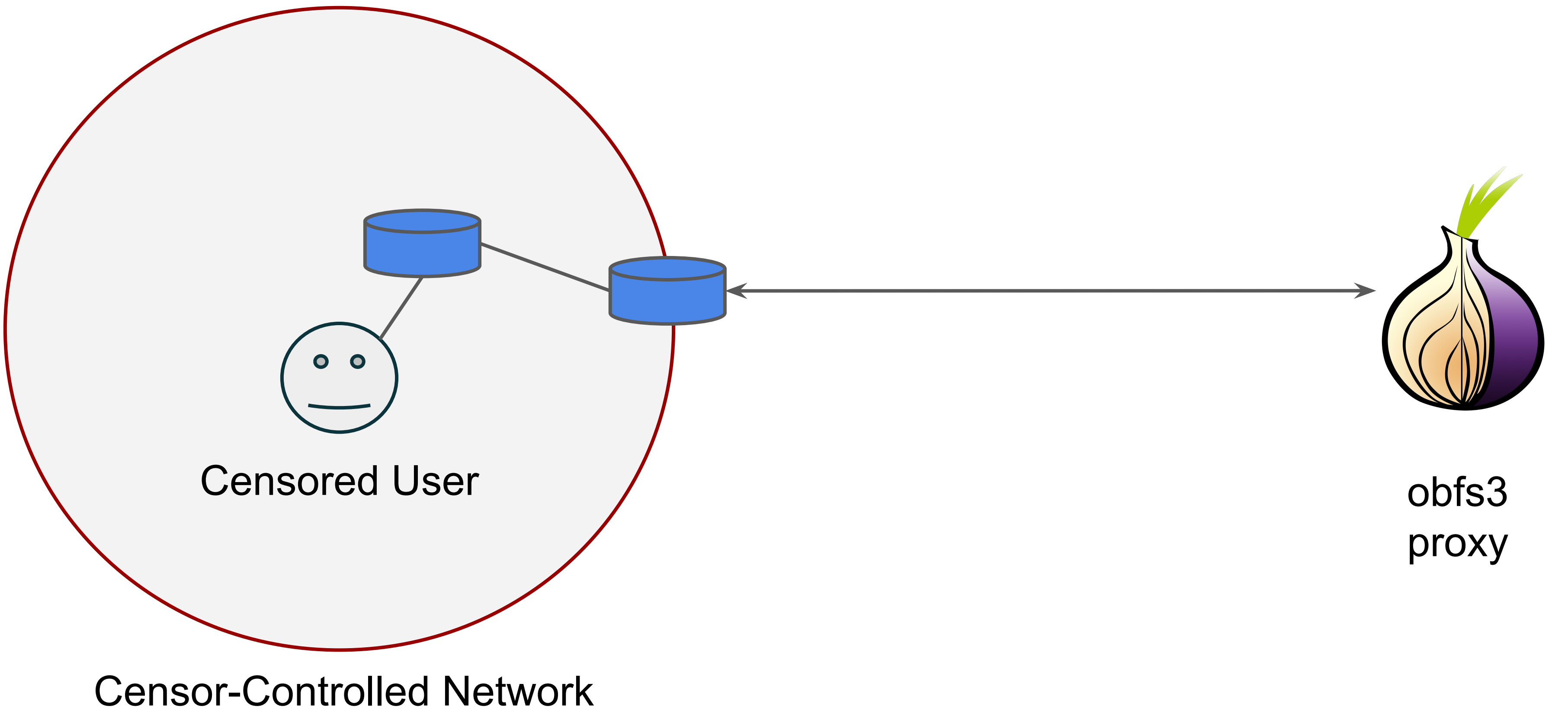
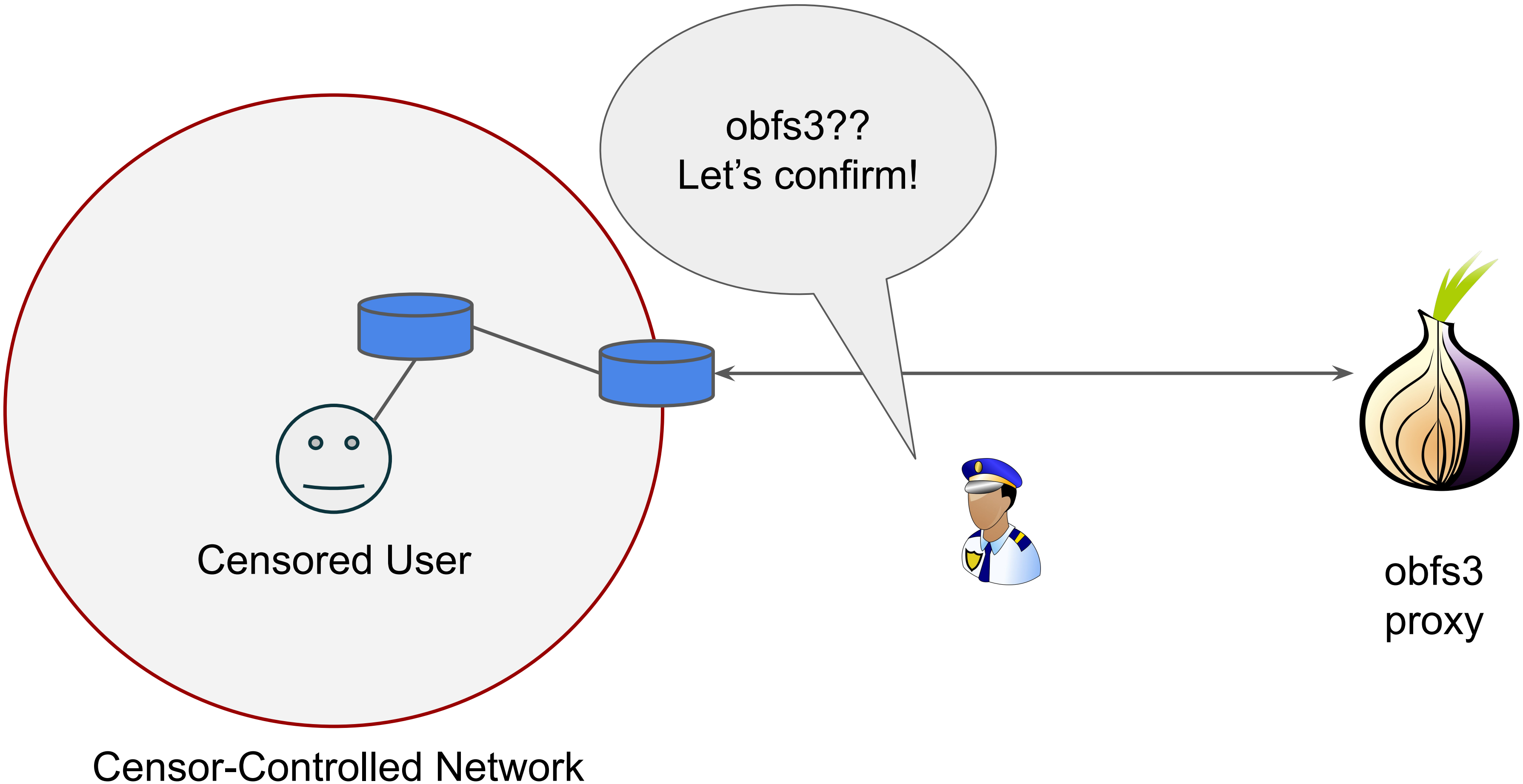
# HTTPT: A Probe-Resistant Proxy

Sergey Frolov, Eric Wustrow
University of Colorado Boulder

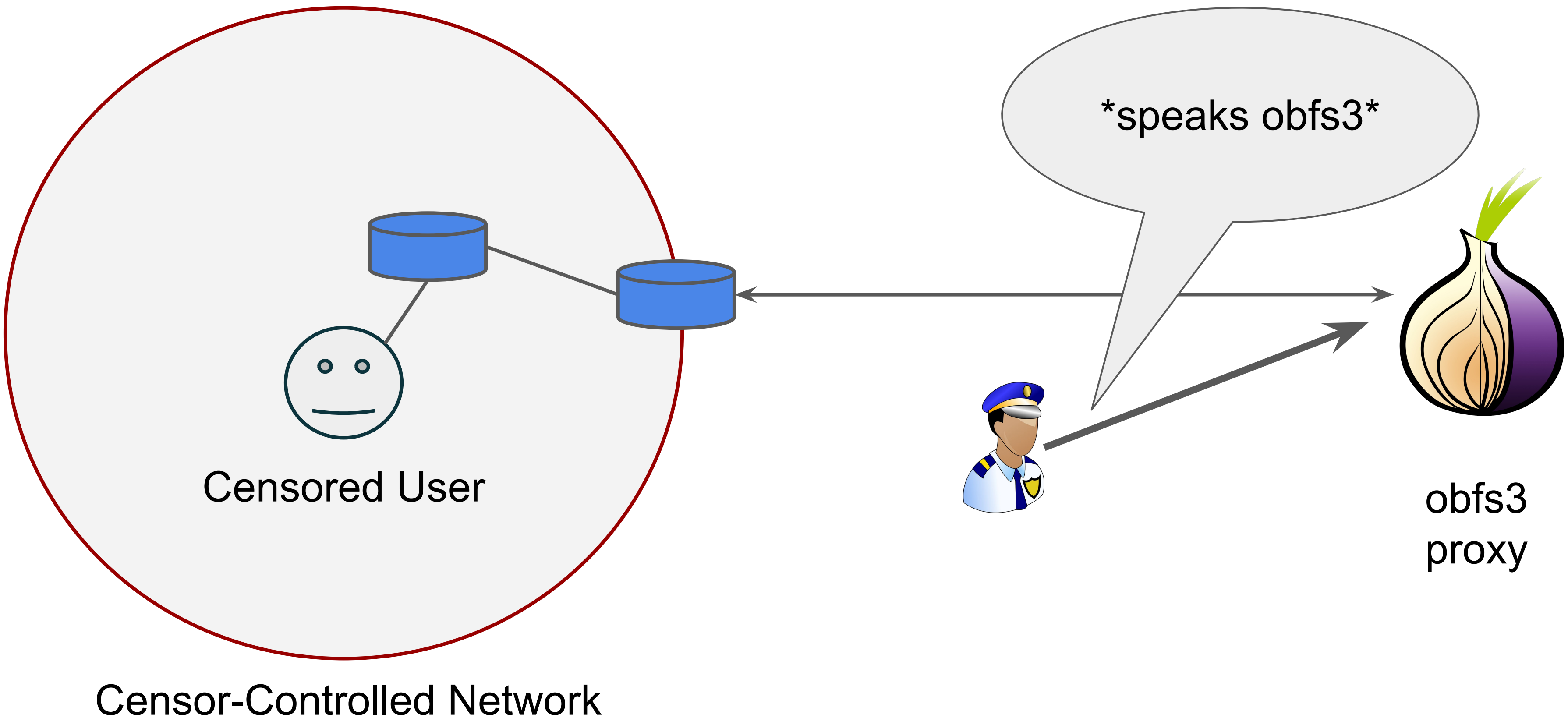
# Proxies



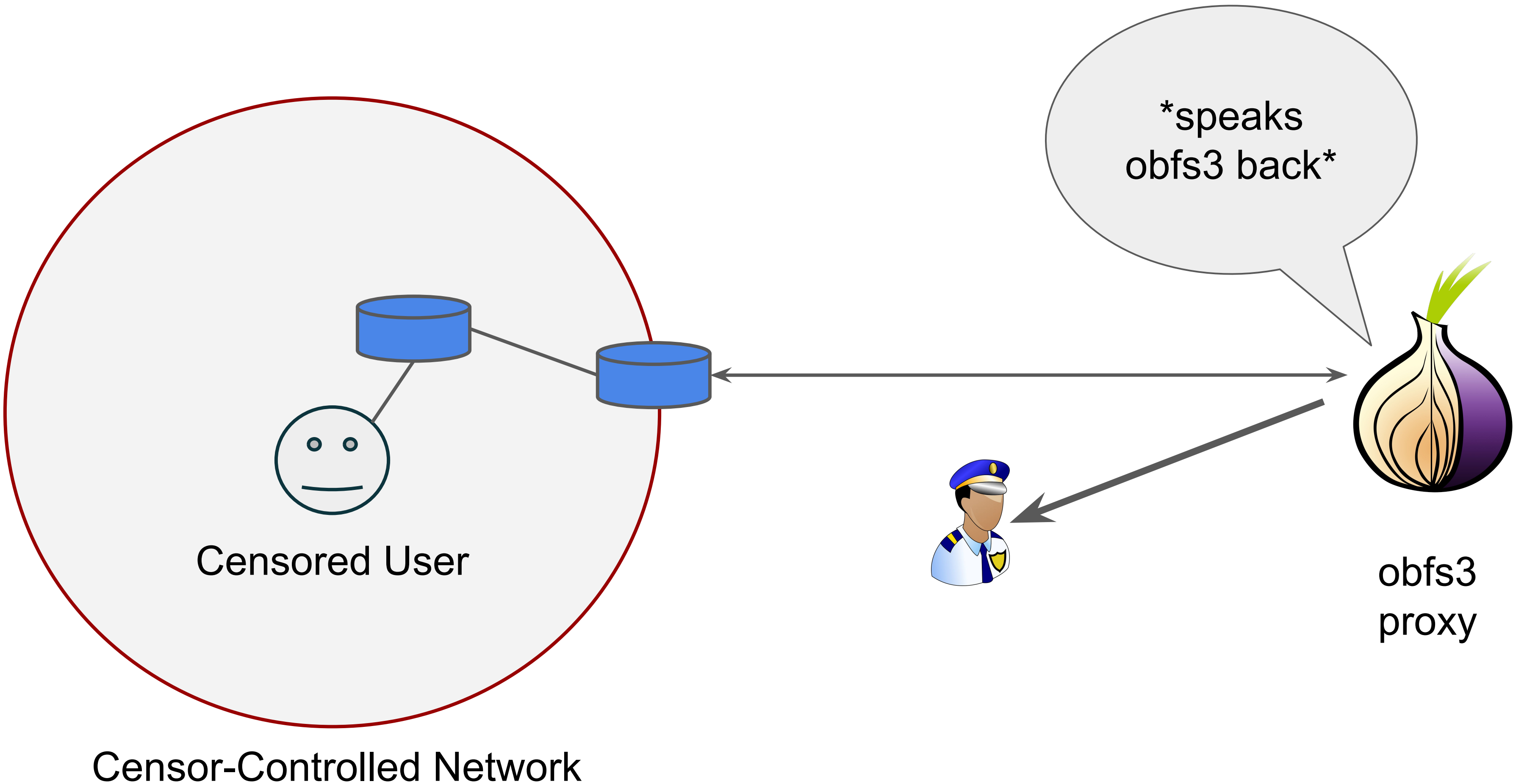
# Active Probing



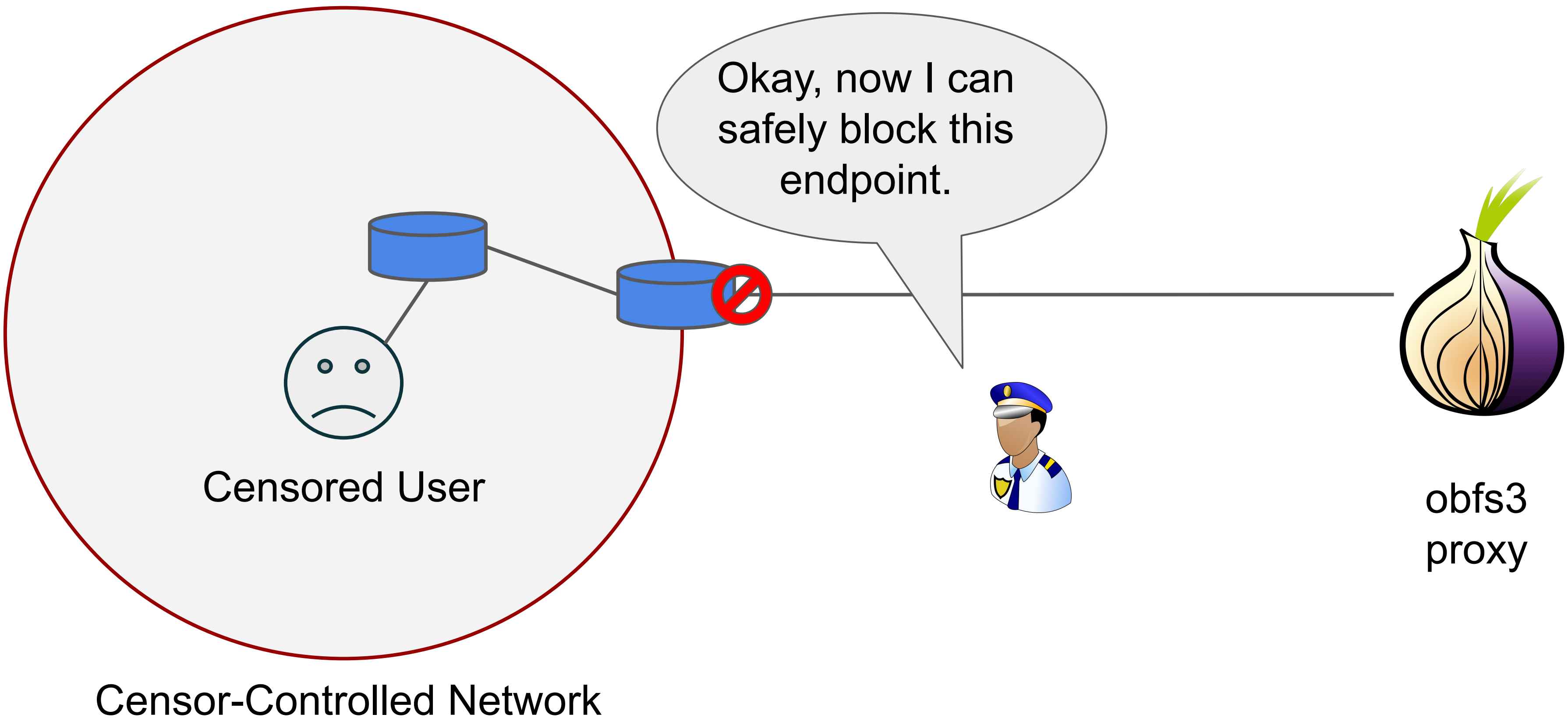
# Active Probing



# Active Probing



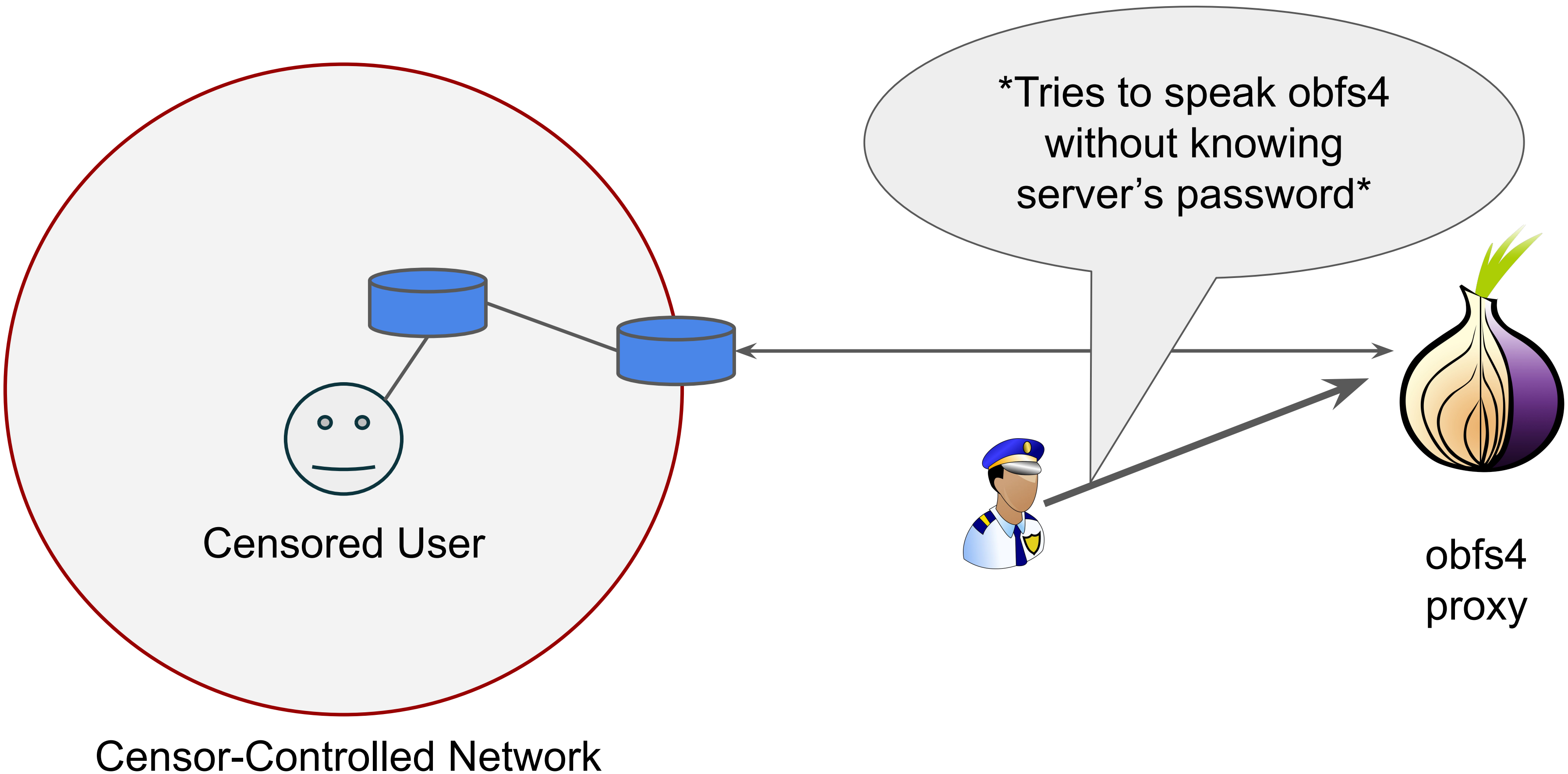
# Active Probing



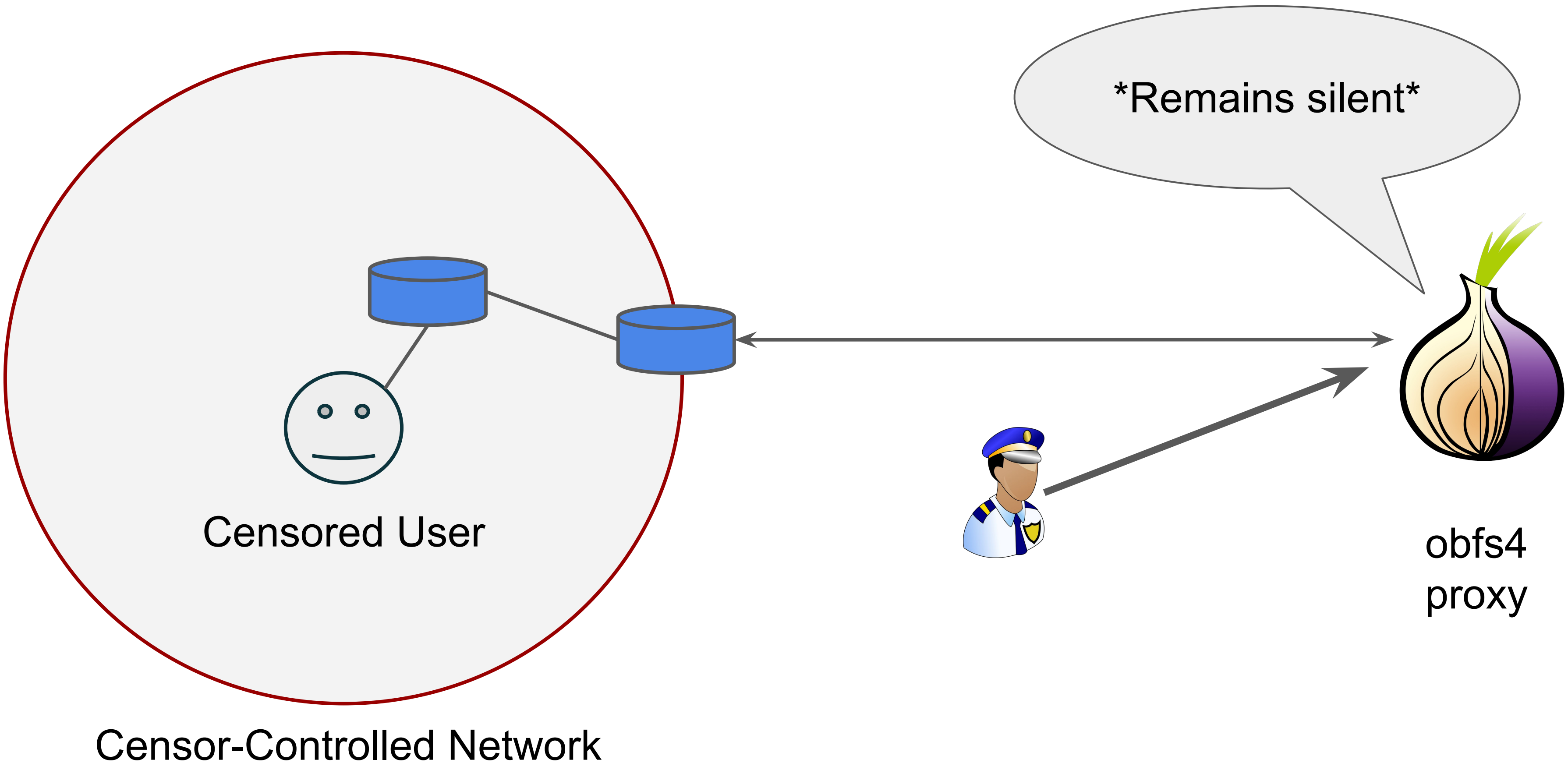
# Thwarting Active Probing

- Probe-Resistant proxies
  - Require knowledge of **shared secret** to use
  - Don’t know secret? Server remains **silent**

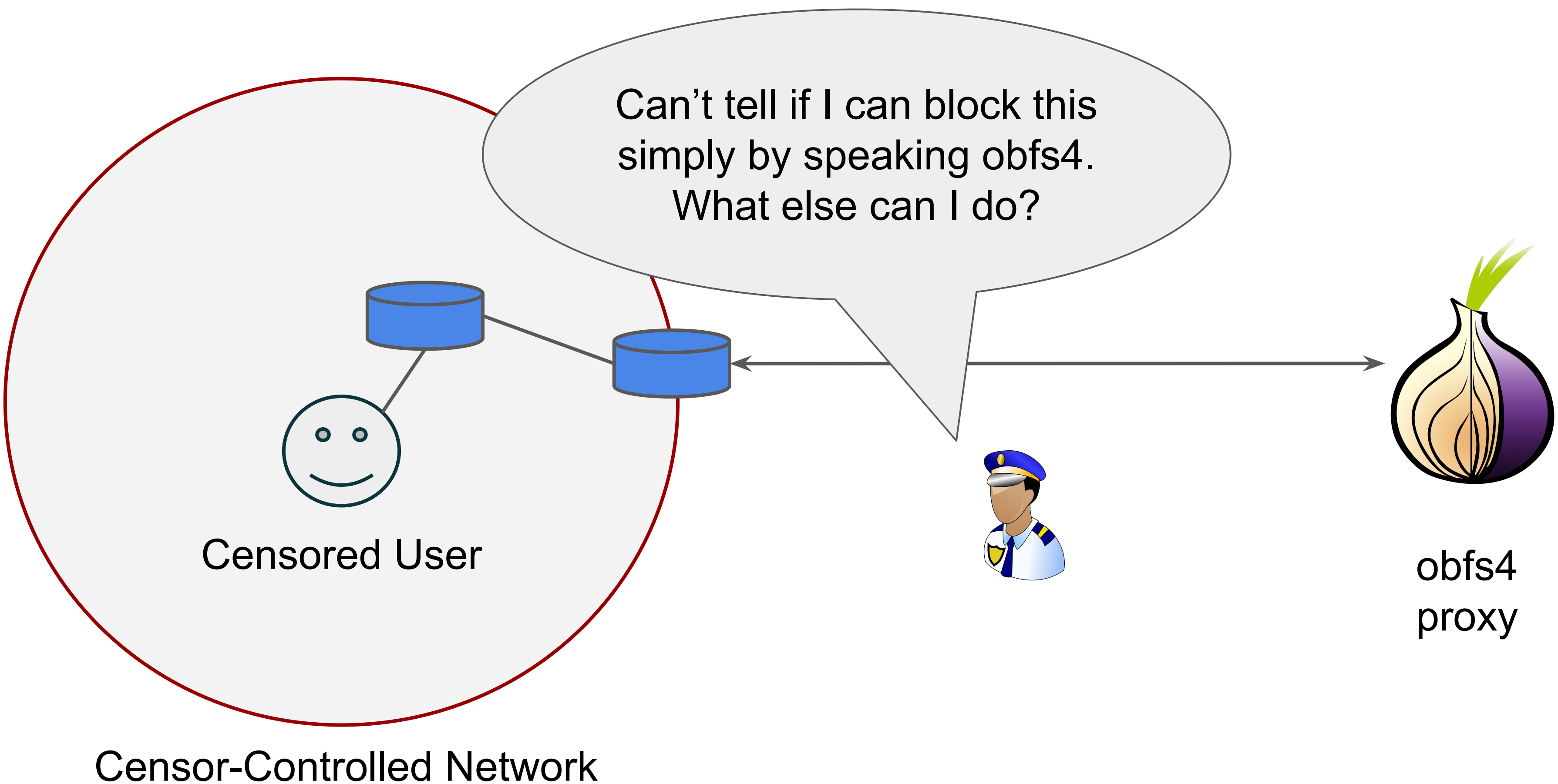
# Thwarting Active Probing



# Thwarting Active Probing



# Thwarting Active Probing



# Advanced Active Probing

Replay Attack - censor resends observed client messages

Some probe-resistant proxies implement a nonce cache to prevent replays of previous connections. However,

- GFW thwarts this defense by permuting replays [*]
- The behavior of not responding at all to replays may be unusual

[*] https://gfw.report/blog/gfw_shadowsocks

# Advanced Active Probing

Fingerprint the server:

1. Send probes using a few popular protocols: ~94% of servers[*] respond with data to at least one popular protocol
2. Remaining “non-responsive” applications are fingerprintable further using Close Threshold and Close Timeout[*]

[*] Frolov, S., Wampler, J., and Wustrow, E. *“Detecting Probe-resistant Proxies”*. NDSS 2020

# Thwarting Advanced Active Probing

Instead of trying to achieve probing resistance by not responding, we hide our proxy server behind another **popular** server application, which would

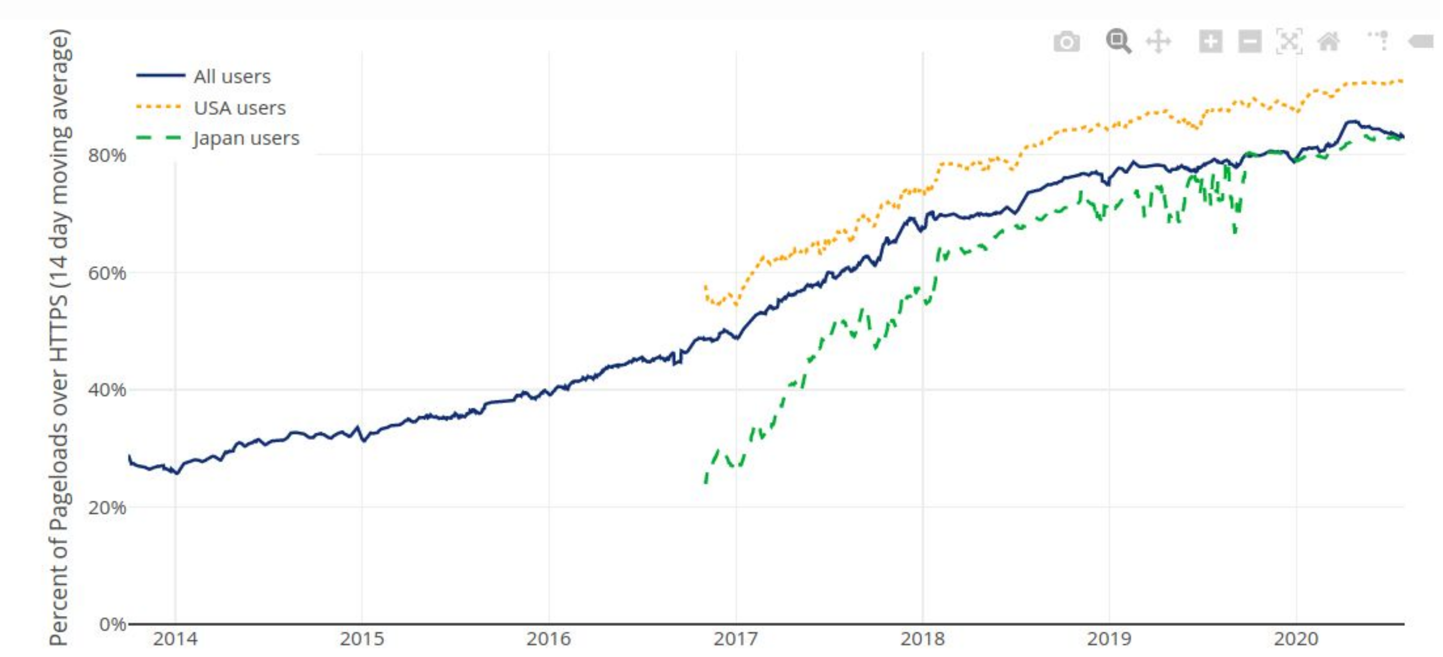
- Tunnel circumvention traffic
- Provide natural responses to the censors' probes.

We evaluate use of HTTPS with Web Servers

# Why HTTPS

HTTPS is common and crucial for the Internet

- Unlikely to be blocked outright




[*] https://letsencrypt.org/stats/

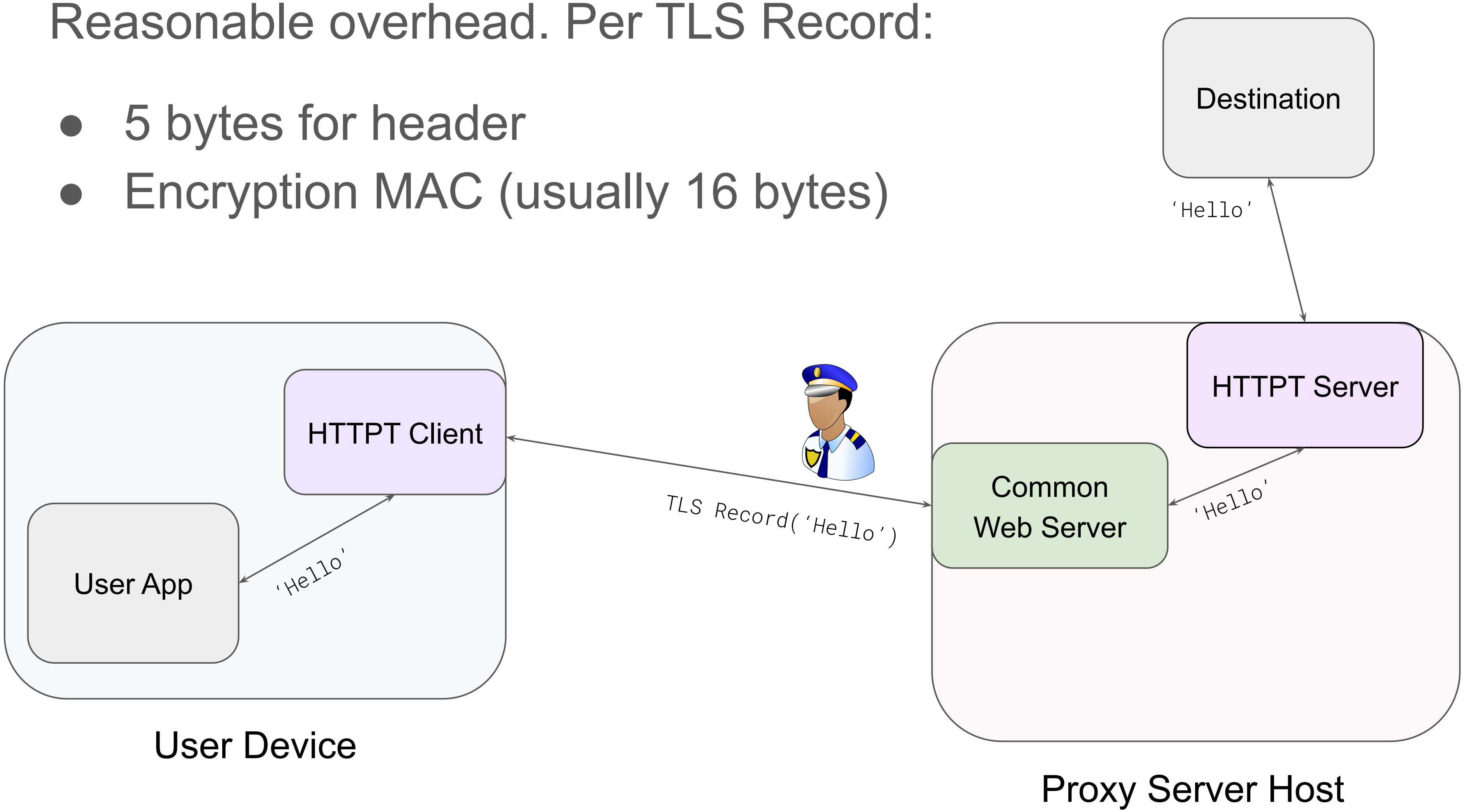
# Why HTTPS

- HTTPS is heterogenous
- HTTPS is used for non-circumvention traffic proxying
- TLS handshake includes bidirectional nonces
- May use existing web servers with actual users

# Why HTTPS

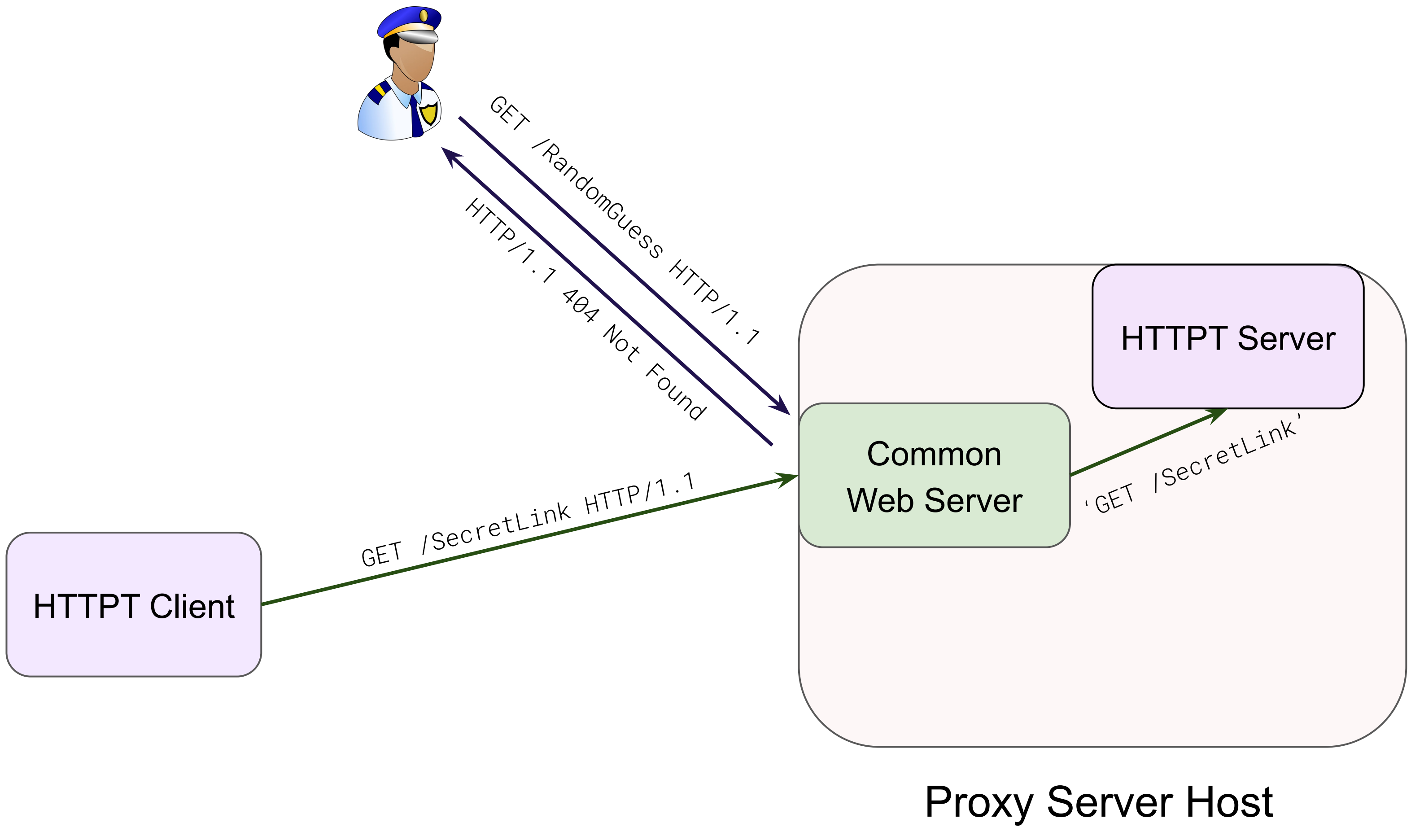
Reasonable overhead. Per TLS Record:

- 5 bytes for header
- Encryption MAC (usually 16 bytes)



# Proving knowledge of the secret

Include the secret in the URL of the initial HTTP request



# What content to serve on the index page?

- Existing website
- An error page
    - Only 48.78% of websites scanned by censys respond with 200 OK[*]
- Copying Content
    - wget random website
- Restricted Access
- Proxy traffic to another website

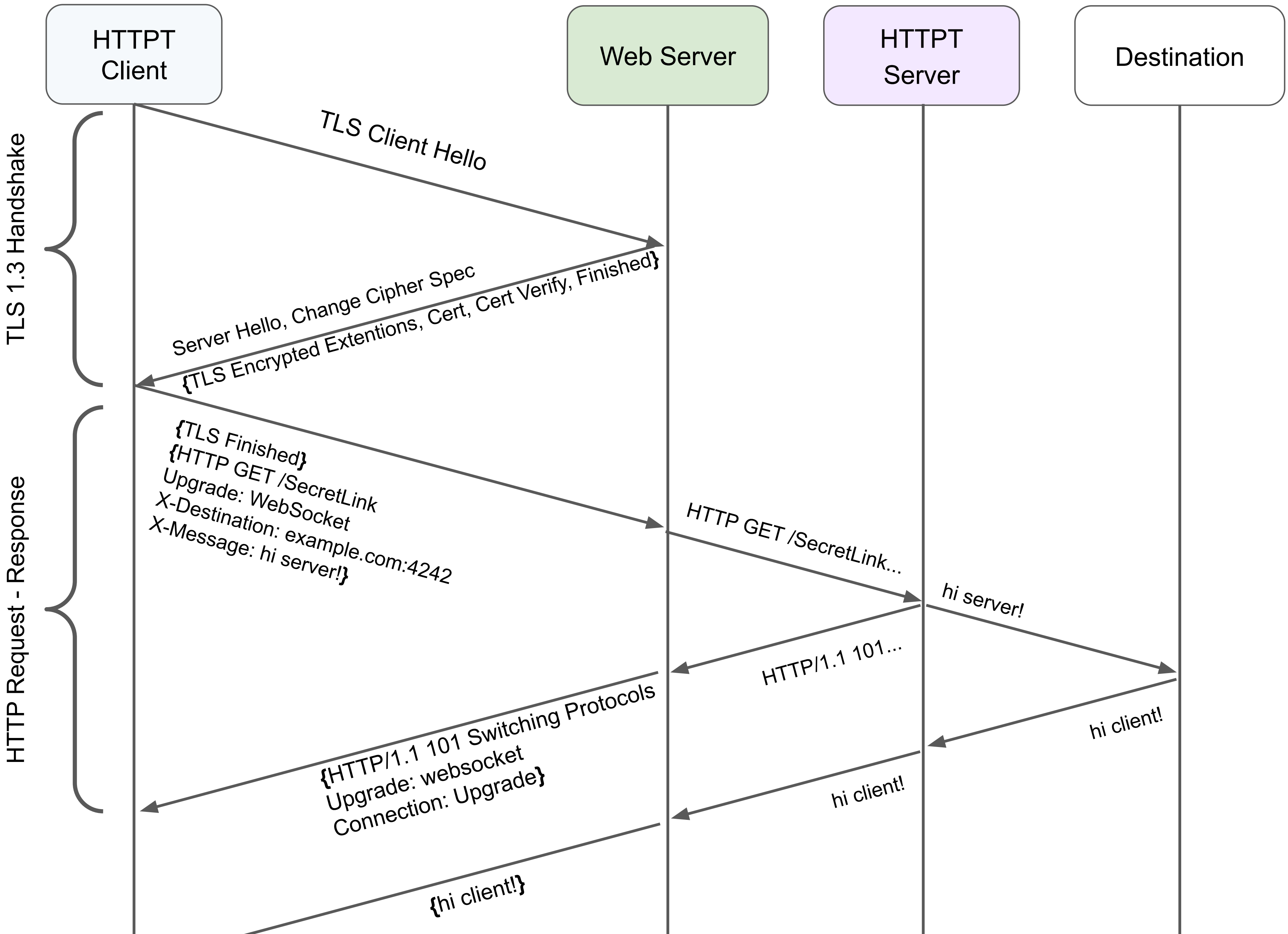
[*] https://censys.io/ipv4/report?field=443.https.get.status_line.raw, June 2020

# Minimizing overhead

One way to get a tunnel with no overhead beyond TLS is to use a WebSockets reverse proxy — feature universally supported by the web servers

HTTPT client starts by sending the HTTP Request with '`Upgrade: WebSocket`' header

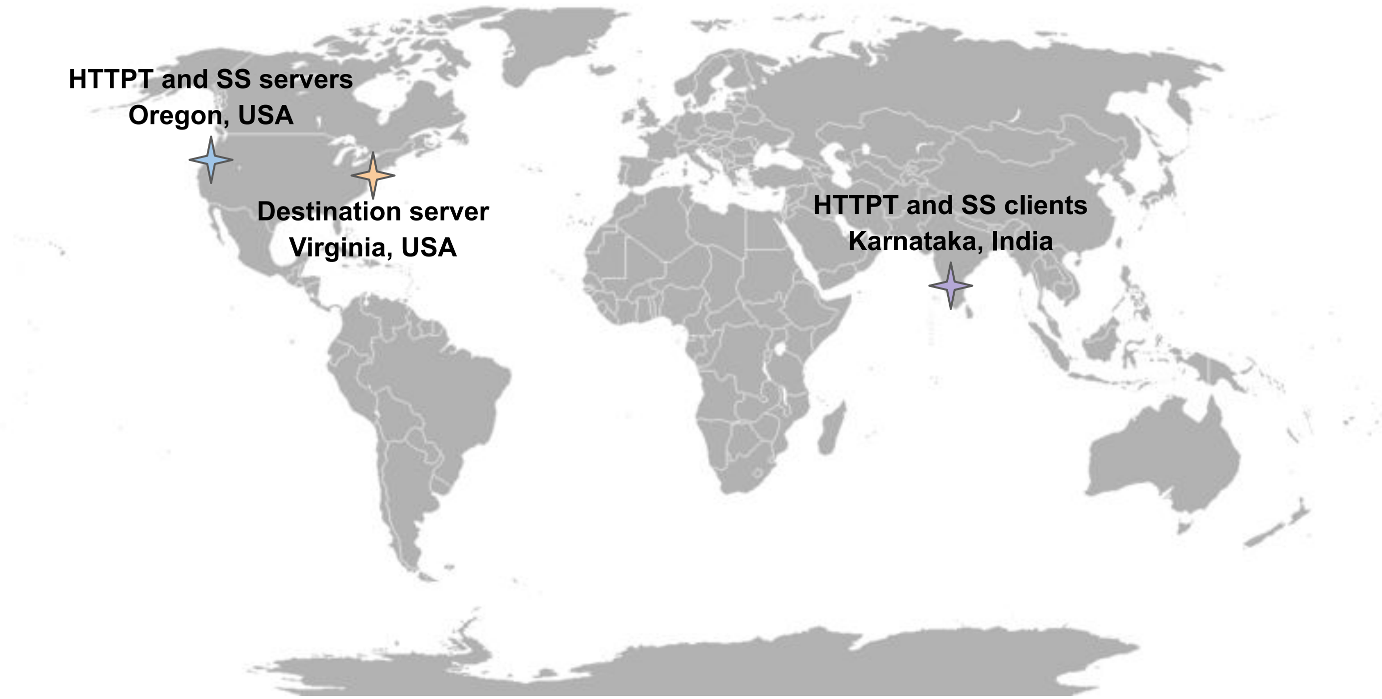
HTTPT server responds with `101 Switching Protocol` status code


HTTPT Client
Web Server
HTTPT Server
Destination
TLS 1.3 Handshake
TLS Client Hello
Server Hello, Change Cipher Spec
{TLS Encrypted Extentions, Cert, Cert Verify, Finished}
HTTP Request - Response
{TLS Finished}
{HTTP GET /SecretLink
Upgrade: WebSocket
X-Destination: example.com:4242
X-Message: hi server!}
HTTP GET /SecretLink...
hi server!
HTTP/1.1 101...
hi client!
{HTTP/1.1 101 Switching Protocols
Upgrade: websocket
Connection: Upgrade}
hi client!
{hi client!}

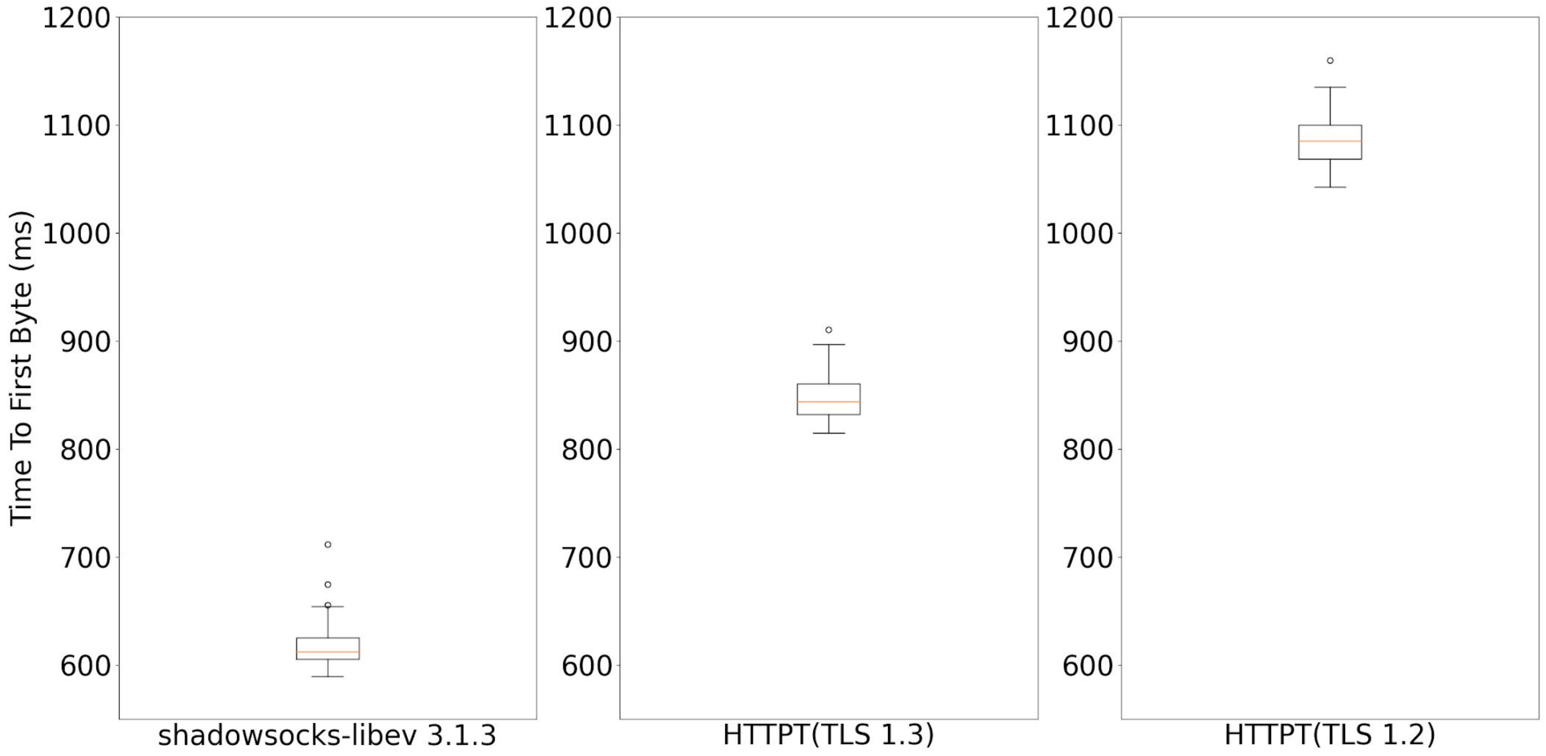
# Performance Evaluation

Compare HTTPT to Shadowsocks proxy

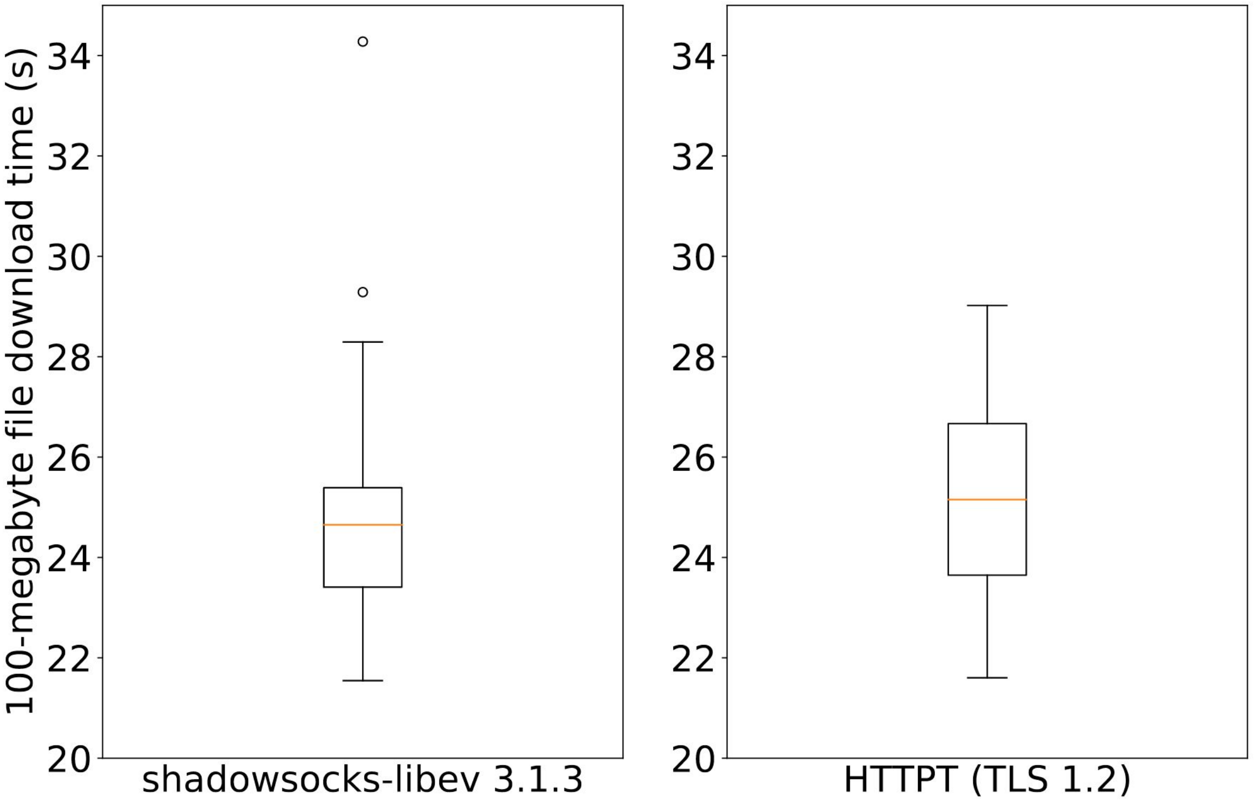
- Does not use padding
- 0-RTT connection establishment (-> no forward secrecy)



# Time To First Byte



# 100-Mb file download time



# Next Steps

- HTTP/2 support: take advantage of multiplexing
- TurboTunnel
- Optional Padding

# Conclusion

HTTPS-based proxy

- Defends against active probing
- Does not need original website content to provide plausible responses to probes
- Performs comparable to lightweight proxies

# FIN

Thanks for watching!