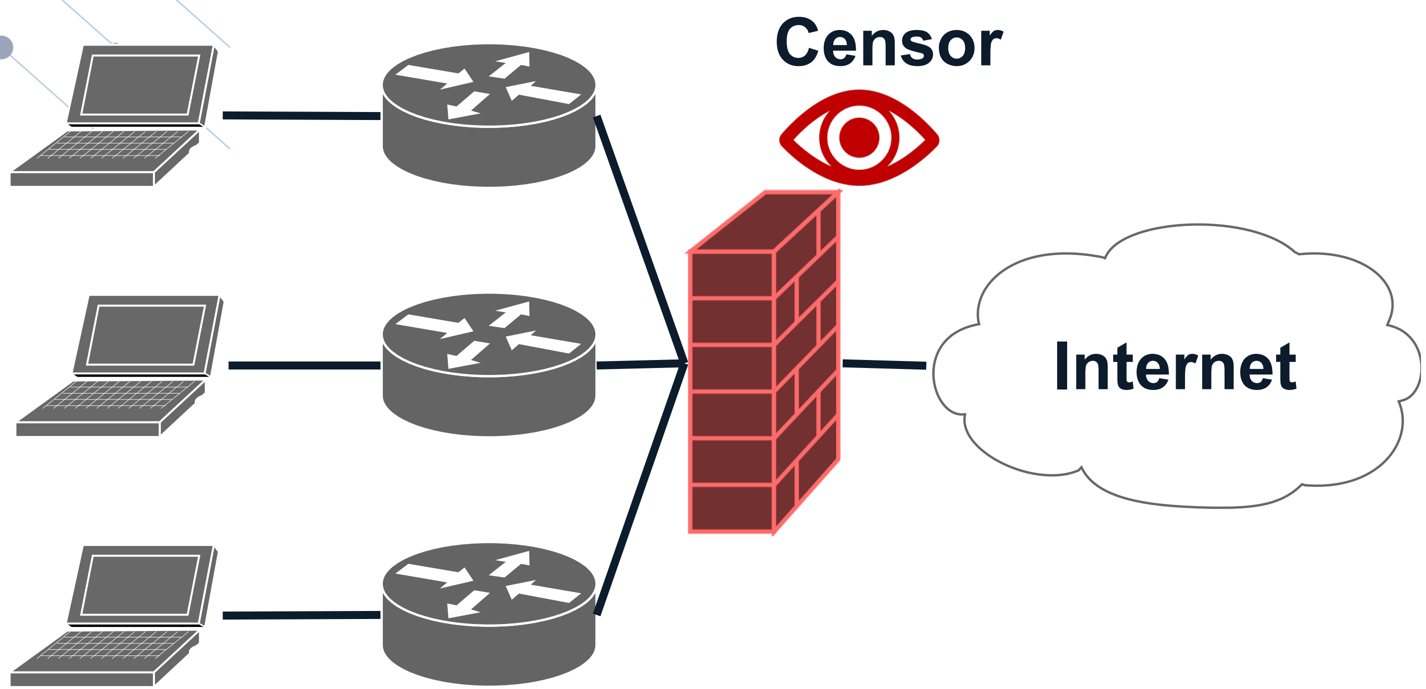
On Precisely Detecting Censorship Circumvention in Real-World Networks

Ryan Wails (Georgetown University; US Naval Research Laboratory)
George Arnold Sullivan (University of California, San Diego)
Micah Sherr (Georgetown University)
Rob Jansen (US Naval Research Laboratory)

NDSS SYMPOSIUM/2024

Presented by Internet Society

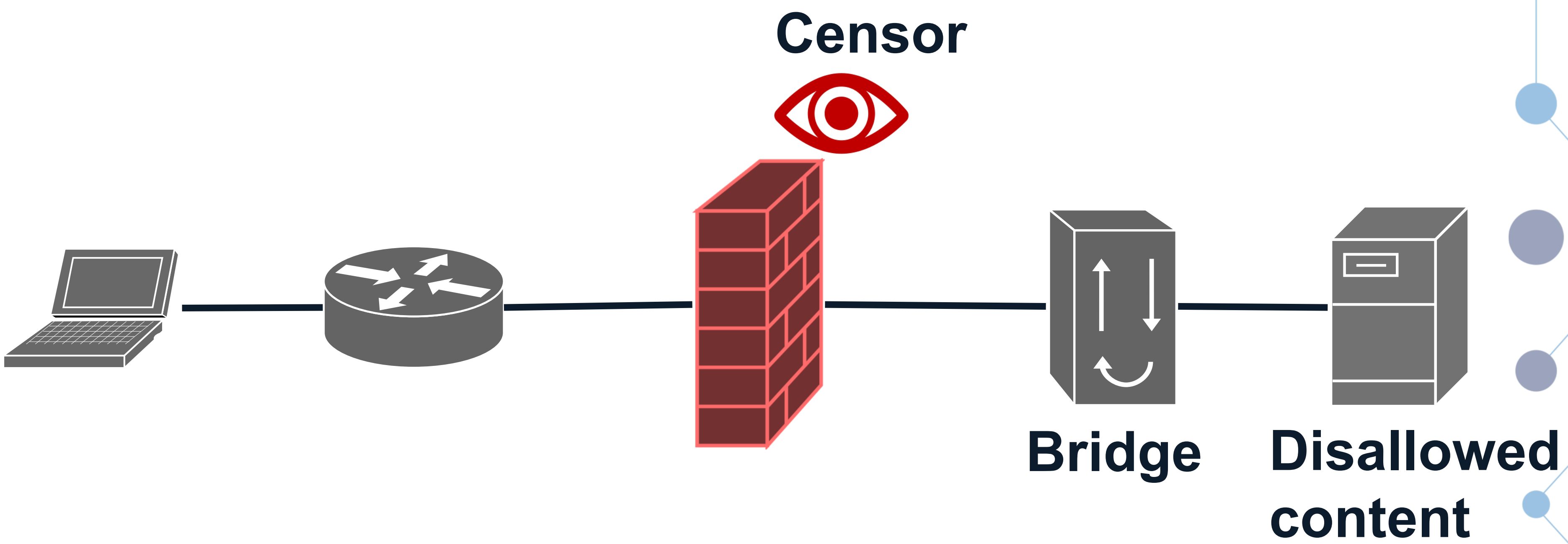


Packet **analysis** may include

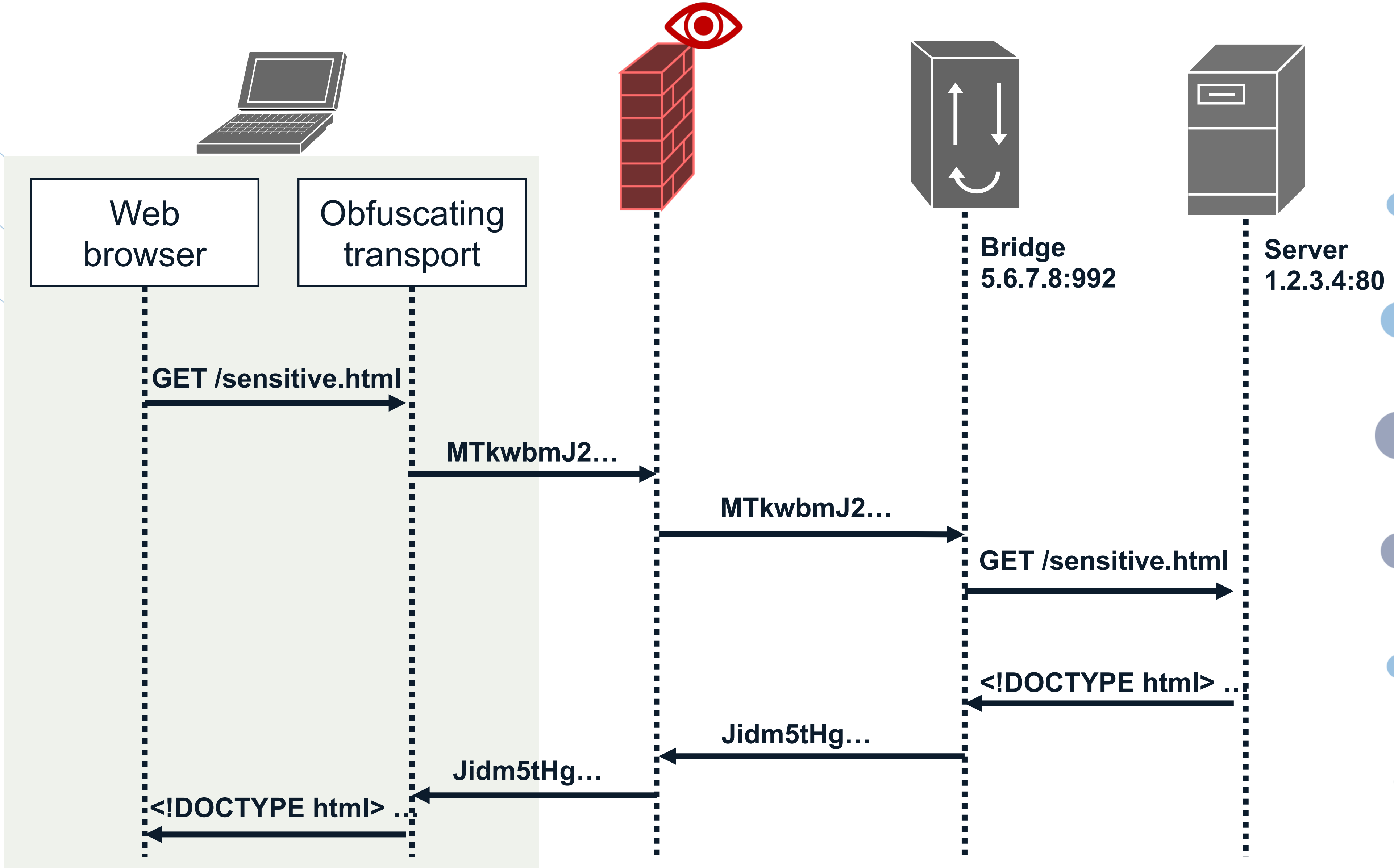
- TCP/IP features
- Payload features
- State

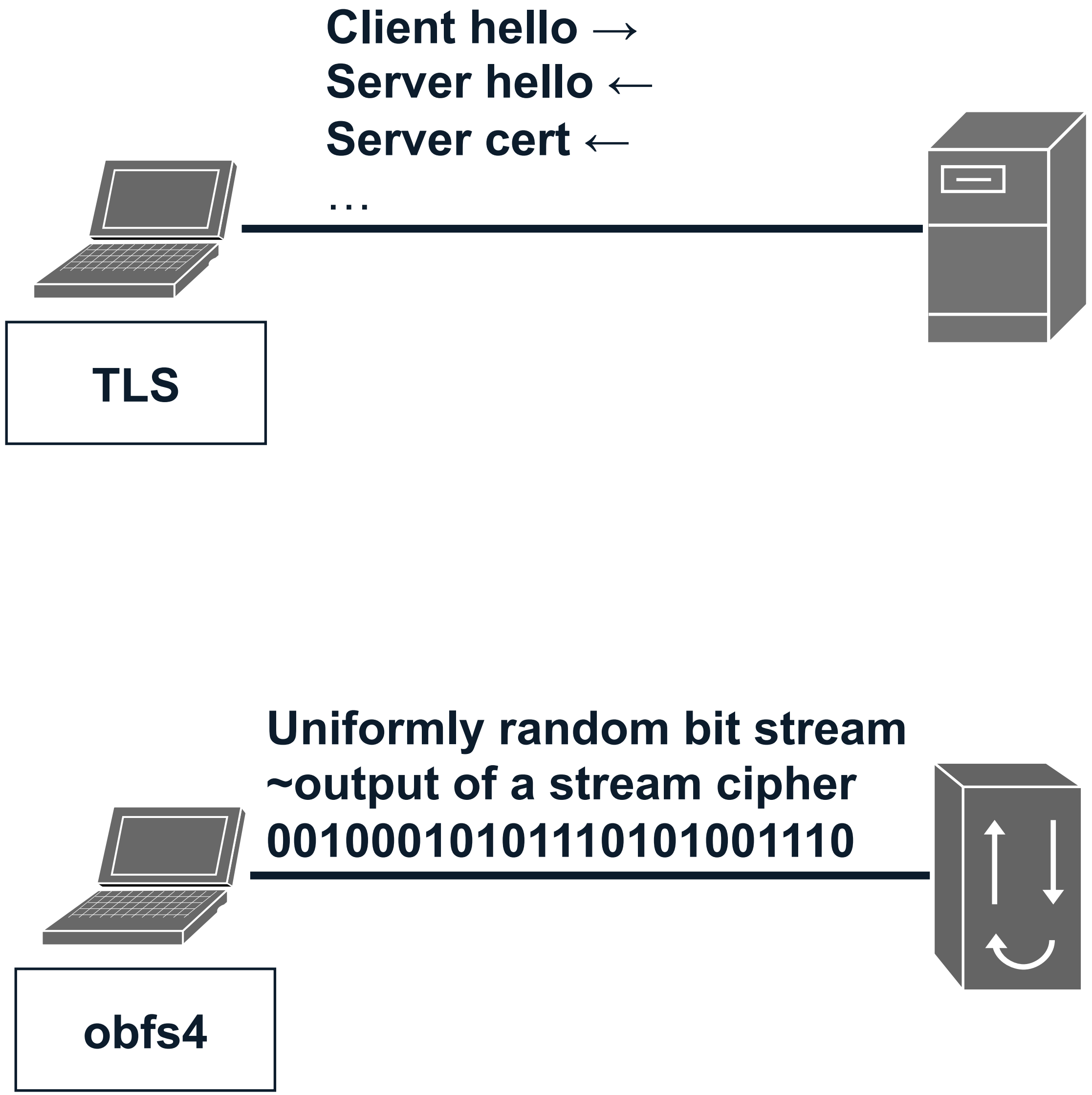
Blocking actions may involve

- Dropping
- Injecting
- Modifying
- Throttling



Introducing a **bridge (proxy)** can modify the TCP/IP features of packets and packet payloads


Web browser
Obfuscating transport
GET /sensitive.html
MTkwbmJ2…
MTkwbmJ2…
Bridge
5.6.7.8:992
Server
1.2.3.4:80
GET /sensitive.html
<!DOCTYPE html> …
Jidm5tHg…
Jidm5tHg…
<!DOCTYPE html> …



Randomization is a common approach for obfuscation

- obfs4 / lyrebird
- Shadowsocks
- VMess / V2Ray
- OpenVPN with XOR patch

Real-world censors are trying to block fully randomized traffic

How the Great Firewall of China Detects and Blocks Fully Encrypted Traffic

Mingshi Wu
GFW Report

Jackson Sippe
University of Colorado Boulder

Danesh Sivakumar
University of Maryland

Jack Burg
University of Maryland

Peter Anderson
Independent researcher

Xiaokang Wang
V2Ray Project

Kevin Bock
University of Maryland

Amir Houmansadr
University of Massachusetts Amherst

Dave Levin
University of Maryland

Eric Wustrow
University of Colorado Boulder

USENIX Sec ’23

Academics approaches exist too…

Seeing through Network-Protocol Obfuscation

Liang Wang
University of Wisconsin
liangw@cs.wisc.edu

Kevin P. Dyer
Portland State University
kdyer@cs.pdx.edu

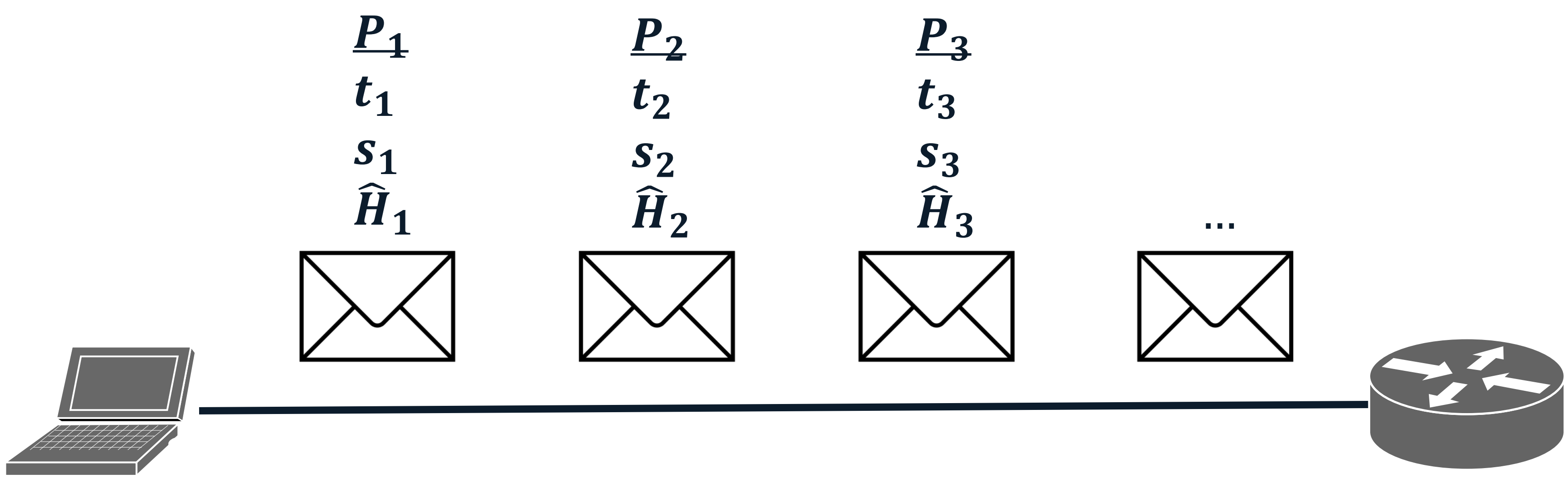
Aditya Akella
University of Wisconsin
akella@cs.wisc.edu

Thomas Ristenpart
Cornell Tech
ristenpart@cornell.edu

Thomas Shrimpton
University of Florida
teshrim@cise.ufl.edu

ACM CCS '15

Wang et al.'s method



- t_i := ith packet's timestamp
- s_i := ith packet's size
- $\hat{H}_i$:= ith packet's "entropy"

$$\hat{H}(p) = -\sum_{j=0..255} f_j \log_2 f_j$$

Decision tree flow classifier with **summary statistic** features:

- $\mathrm{top}_5\ \boldsymbol{s_i}$
- $\min \boldsymbol{\hat{H}_i}, \max \boldsymbol{\hat{H}_i}, \mathrm{mean}\ \widehat{\boldsymbol{H}_i}$
- Histogram of $t_{i+1} - t_i$ for ACKs

In our work, we

- Re-evaluate Wang's classifier with modern data set of **real-world** network traffic statistics

  Spoiler ⚠️ too noisy to work in practice

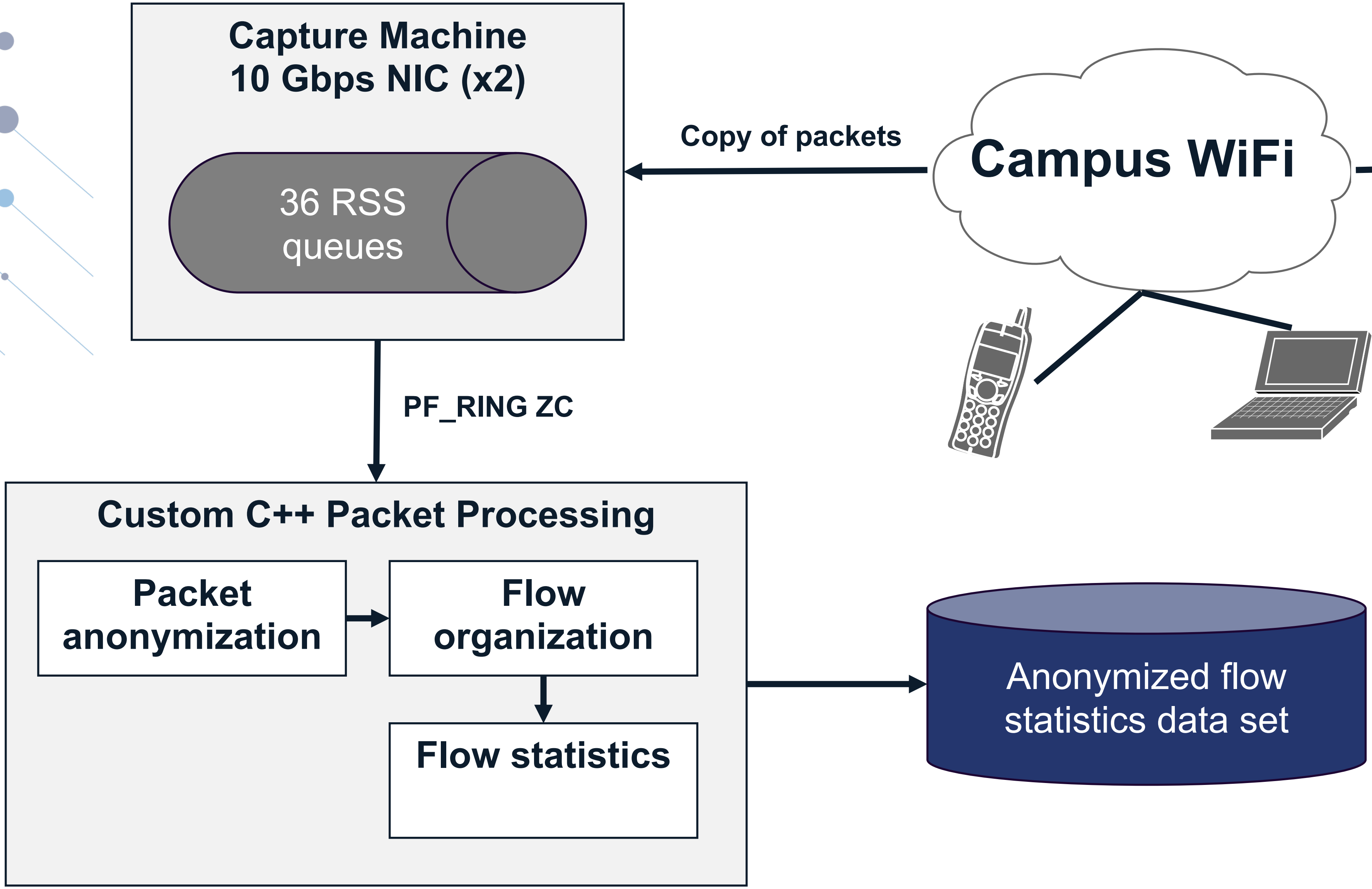
- Apply modern **deep learning** classifiers to the problem

  Spoiler ⚠️ also too noisy to work in practice

- Rephrase the problem in terms of **host-centric classification**

  Spoiler ⚠️ classifying hosts is much easier

1. **Network data collection**
2. Classic flow-based classification results
3. Neural net flow-based classification results
4. Host-based classification technique

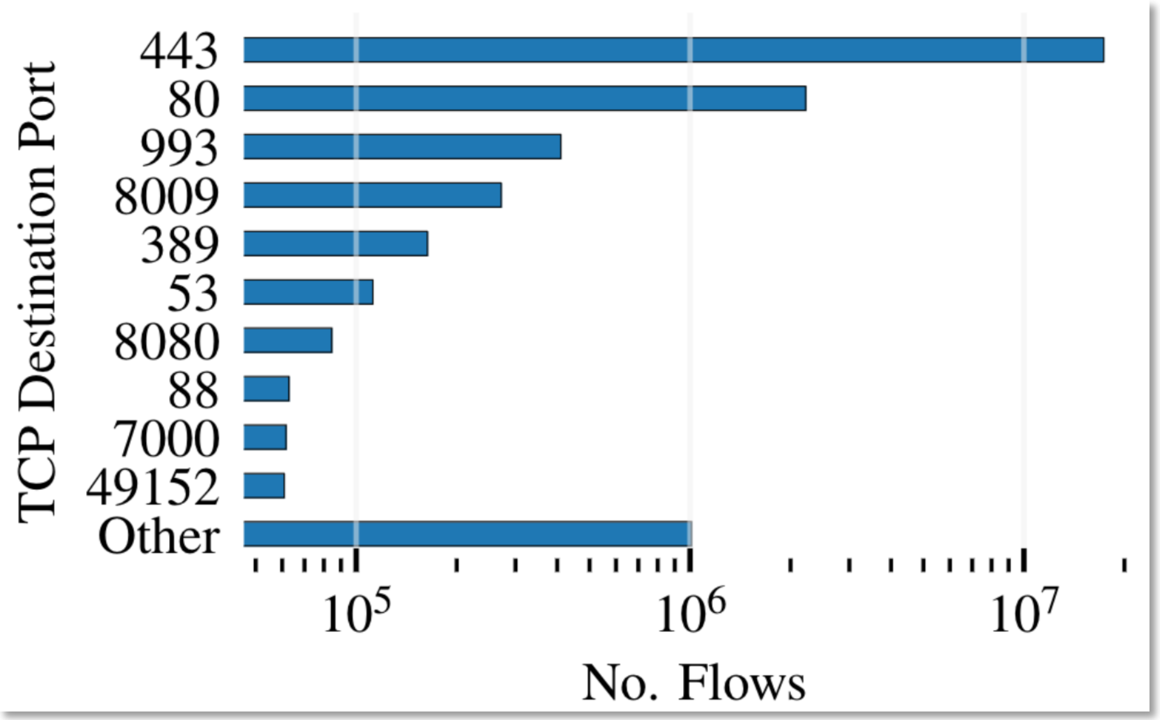

Capture Machine
10 Gbps NIC (x2)
36 RSS
queues
Copy of packets
Campus WiFi
obfs4
PF_RING ZC
Custom C++ Packet Processing
Packet
anonymization
Flow
organization
Flow statistics
Anonymized flow
statistics data set

Safety measures:

- Existing network tap and data protection scheme with IRB and staff approval
- Capture machine was physically secured and on isolated network; multi-FA required
- Never stored packet payloads
- Anonymized IP addresses w/ HMAC
- Only one approved team member had access to capture machine and hashes

Basic collection statistics:

- 60 million flows
- 600,000 hosts
- Injected 80,000 obfs4 flows from 8 bridges



1. Network data collection
2. **Classic flow-based classification results**
3. Neural net flow-based classification results
4. Host-based classification technique

Decision-tree performance classifying **obfs4** flows

TPR	98%
FPR	06%
FPR on non-training protocols	11%
FPR on rare protocols: rank > 10	08%
FPR on rare protocols: rank > 100	15%
FPR on rare protocols: rank > 1000	19%

The base rate reality
Assuming a 1000:1 benign:circumventing ratio,
precision is 2% !!

1. Network data collection
2. Classic flow-based classification results
3. **Neural net flow-based classification results**
4. Host-based classification technique

We tried:

- A sparse denoising autoencoder [1]
- A convolutional neural network (CNN) [1]
- "Deep Fingerprinting" CNN [2] ← **Best perf**

[1] V. Rimmer et al. "Automated website fingerprinting through deep learning." In: NDSS '18.

[2] P. Sirinam et al. "Deep Fingerprinting: Undermining website fingerprinting defenses with deep learning." In: ACM CCS '18.

Input: $\langle d_i \cdot s_i \rangle_{i=1}^{5000}$ where

$d_i \in [-1, 1]$ is the ith packets direction
and
$s_i \in [0, 1]$ is the ith packet's normalized size

Why should this work?

obfs4 exhibits unique packet size distributions:

⟨ 1410, 1410, 1410, 307 | -1410, -1410, -805 | 1410, … ⟩

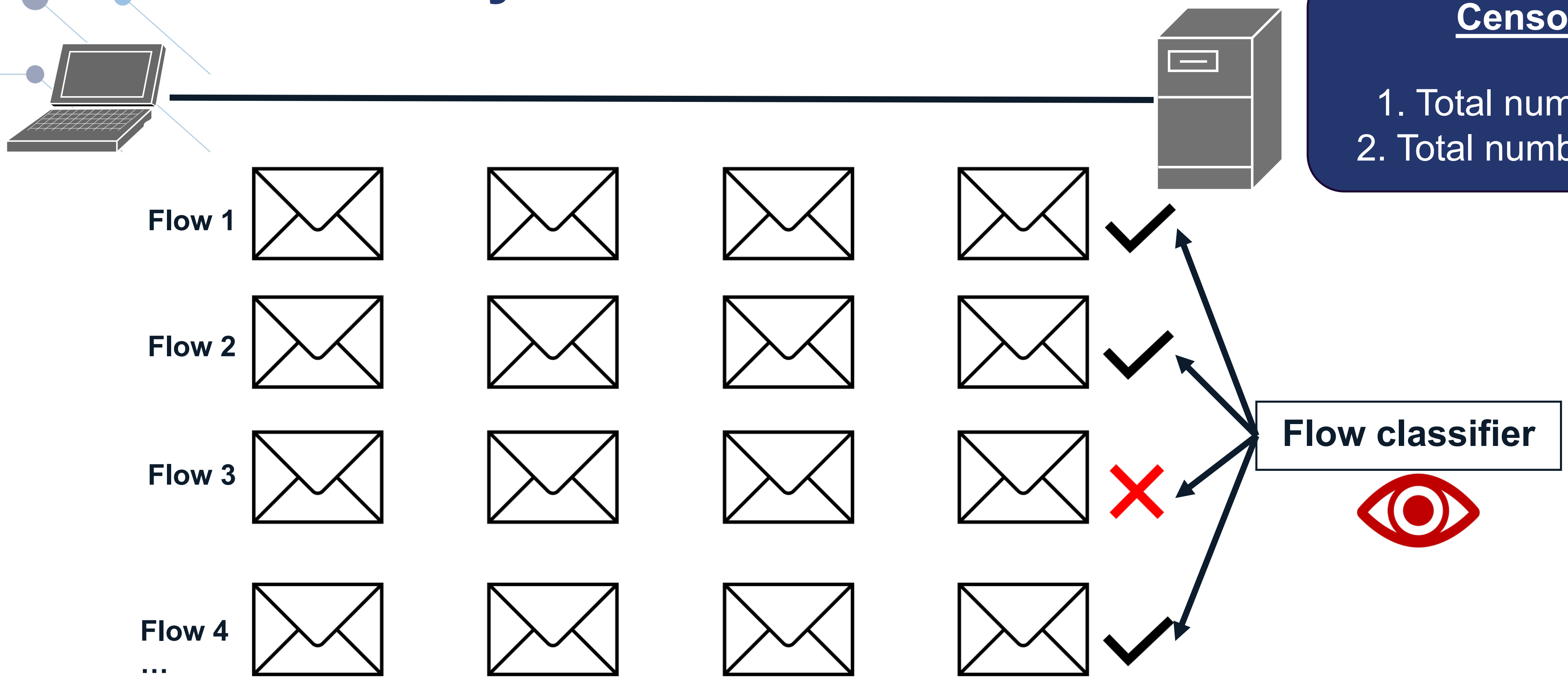
DF (CNN) performance classifying **obfs4** flows

TPR	100%
FPR	0.3%
FPR on non-training protocols	0.4%
FPR on rare protocols: rank > 10	0.2%
FPR on rare protocols: rank > 100	0.5%
FPR on rare protocols: rank > 1000	0.6%

The base rate reality
Assuming a 1000:1 benign:circumventing ratio,
precision is 26% !!

1. Network data collection
2. Classic flow-based classification results
3. Neural net flow-based classification results
4. **Host-based classification technique**

Key idea: censors are free to maintain state for each host and may choose to **classify hosts** instead of flows



Censor's state

1. Total number of ✗s (p)
2. Total number of flows (m)

(under simplifying assumptions)

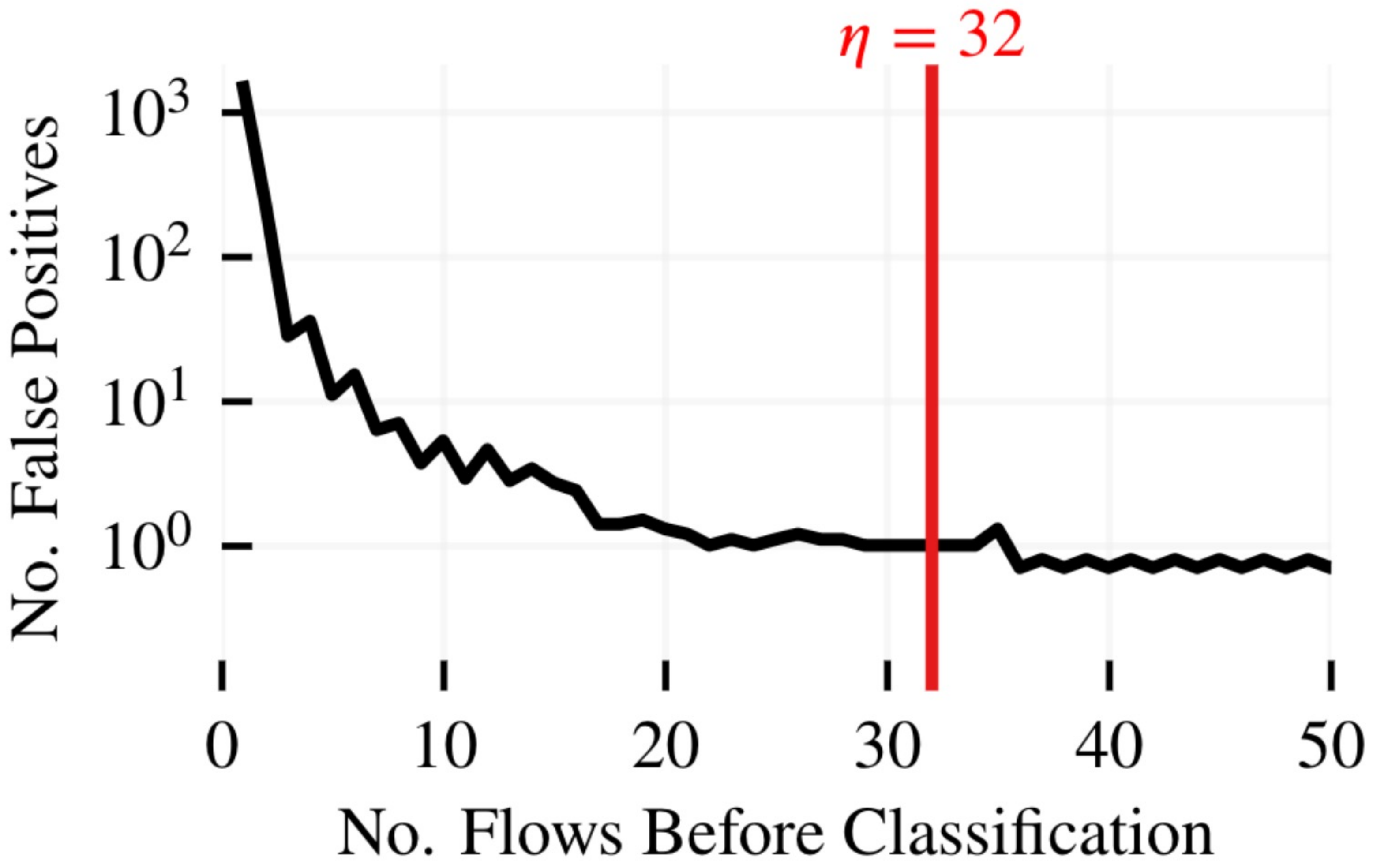
- $\mathbb{E}[{}^{p}/_{m}]$ = TPR for a circumventing host
- $\mathbb{E}[{}^{p}/_{m}]$ = FPR for a benign host

For big enough m, classify host as circumventing if

${}^{p}/_{m} > \tau$ for $\tau = {}^{(TPR+FPR)}/_{2}$ $\tau \approx 0.5$

Wait for $\eta = \left\lceil \frac{\ln 4/\alpha^2}{(TPR - FPR)^2} \right\rceil$ flows for desired error rate α

$\eta \approx 30$
for α = 1e-6



False positives detecting obfs4

See our paper for:

- Classification performance against a hypothetical tweak of obfs4 that reduces apparent randomness
- Classification performance against the Snowflake circumvention system
- Deep learning classification throughput
- Further exploration of the effect of the base rate on classification

Takeaways and future directions:

- Flow-based classification is probably too noisy for censors to employ effectively
- Host-based analysis requires few additional resources but disproportionally increases classification performance
- **Flash proxying** (Snowflake) is a promising countermeasure to host-based attacks
- **Protocol polymorphism** is another promising countermeasure (FTE, Marionette, Proteus, and WATER)