TEL AVIV UNIVERSITY אוניברסיטת תל אביב

THE HEBREW UNIVERSITY OF JERUSALEM

DNS FLaRE: A Flush-Reload Attack on DNS Forwarders


Gilad Moav


Yehuda Afek


Anat Bremler-Barr


Amit Klein

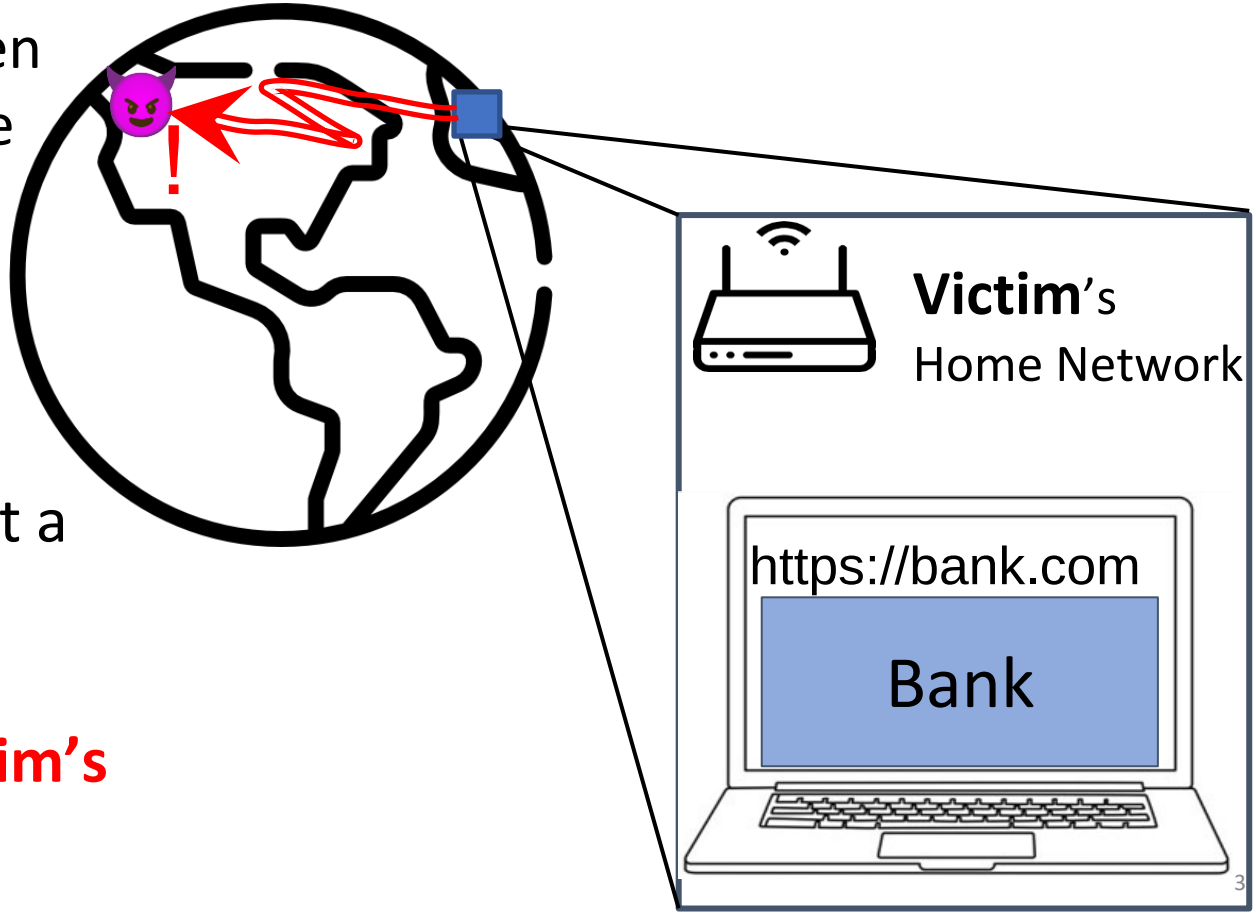
USENIX Security, 2025 Seattle USA

DNS FLaRE: A Flush-Reload Attack on DNS Forwarders

- Cache-based **side channel attack**.
- Infer **browsing** & **IoT** activity within the victim's home network.
- Threat model – nothing malicious installed on victim's computer.

Use-case 1 - Browsing Activity

😈Attacker : knows when victim visits a website like **a bank website**.

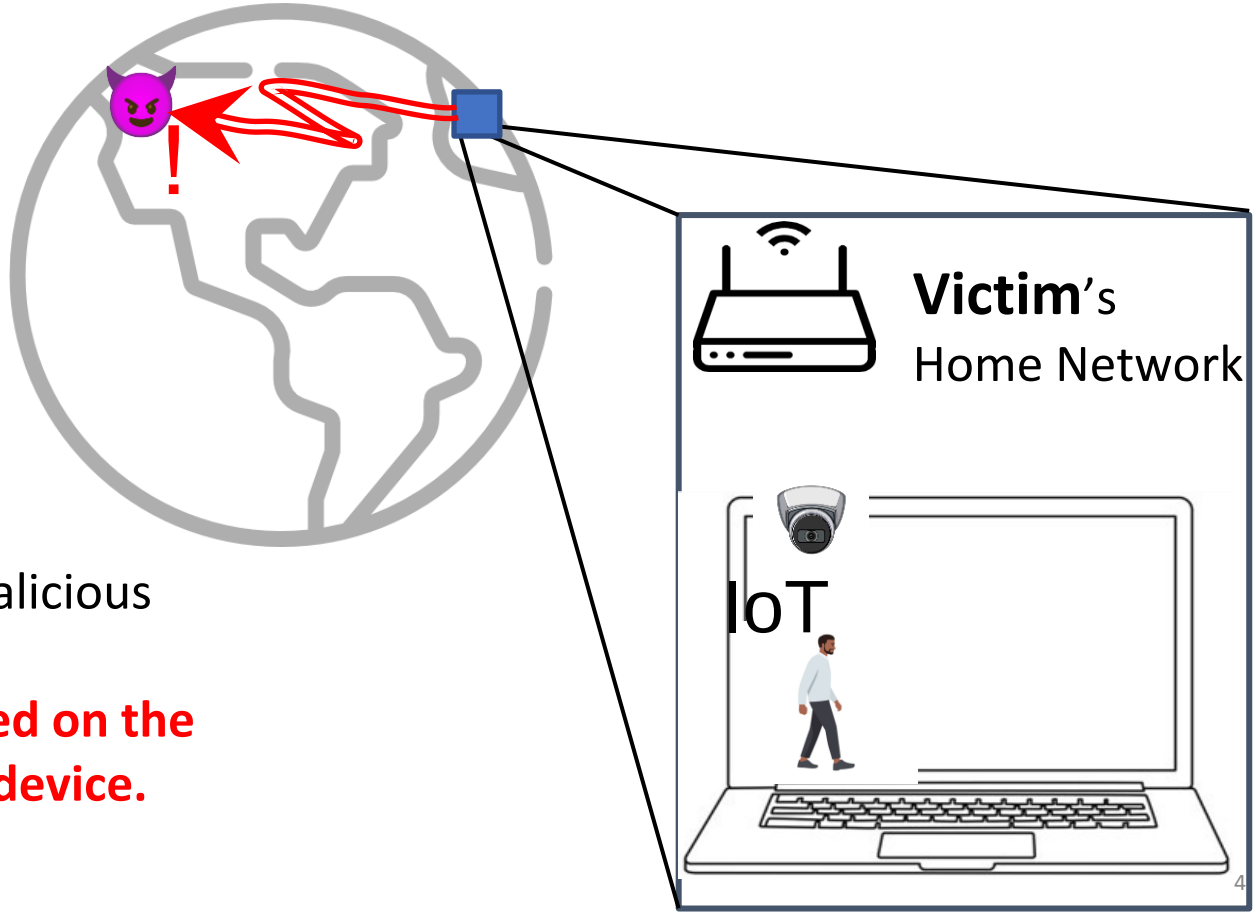


Threat model:
Victim tricked to visit a malicious website.

Nothing malicious installed on the victim's computer.

Use-case 2 - IoT Activity

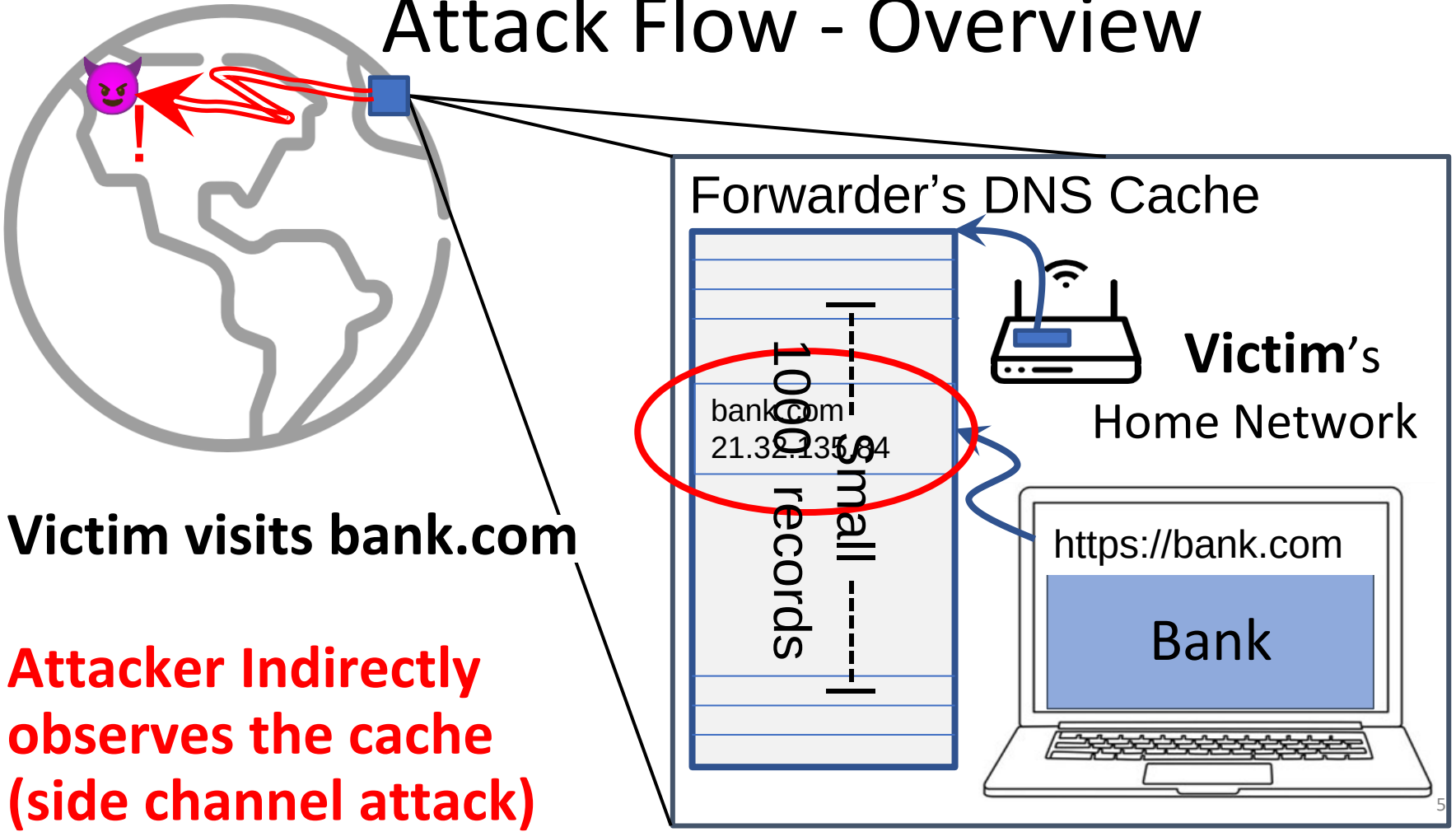
😈 Attacker : knows when an IoT event occurred like movement detection



Threat model:
Victim tricked to visit a malicious website.

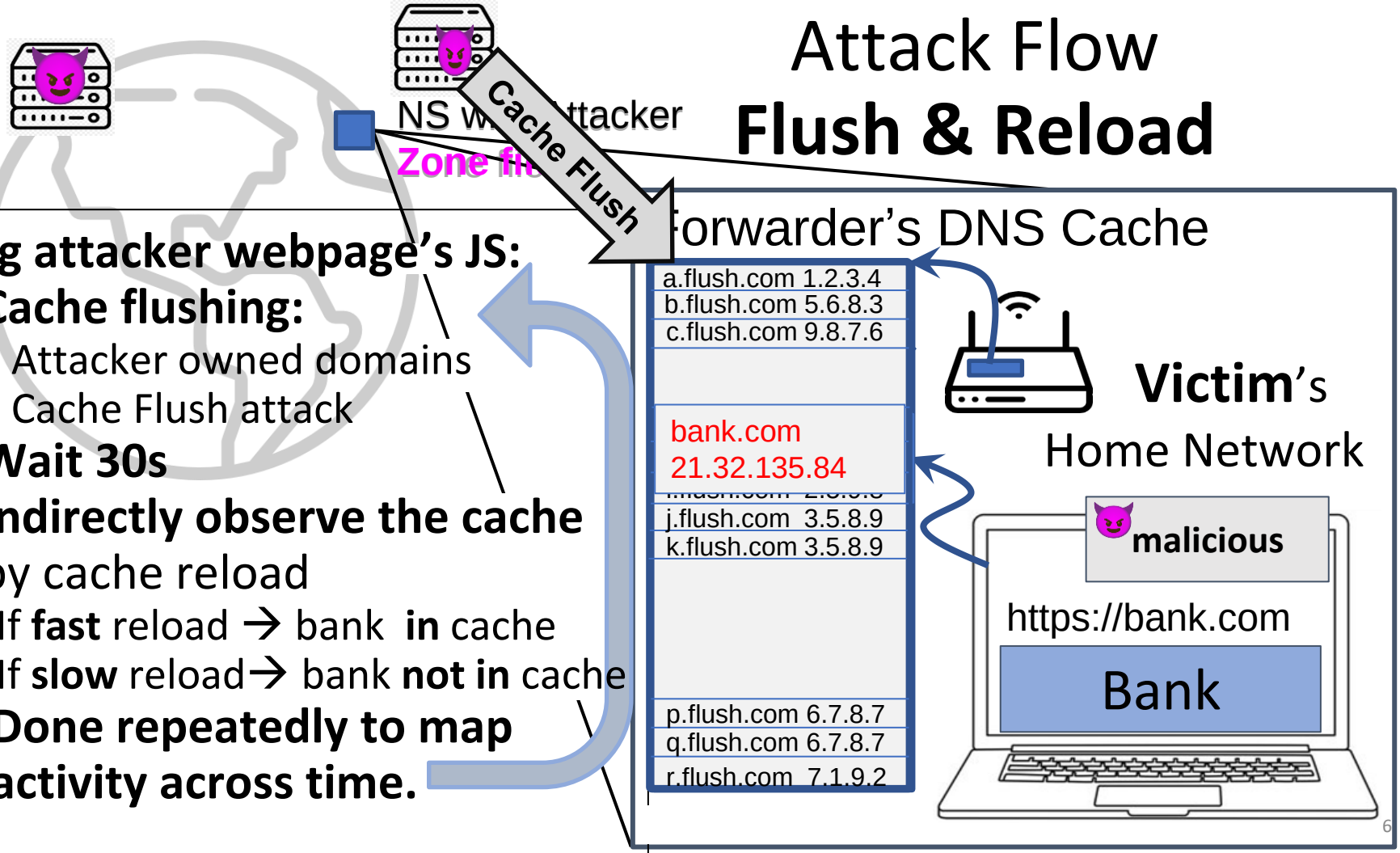
Nothing malicious installed on the victim's computer or IoT device.

Attack Flow - Overview



1. **Victim visits bank.com**
2. **Attacker Indirectly observes the cache (side channel attack)**

Attack Flow
Flush & Reload



Using attacker webpage's JS:

1. **Cache flushing:**
   - Attacker owned domains
   - Cache Flush attack
2. **Wait 30s**
3. **Indirectly observe the cache** by cache reload
   - If **fast** reload → bank **in** cache
   - If **slow** reload→ bank **not in** cache
4. **Done repeatedly to map activity across time.**

4 major challenges

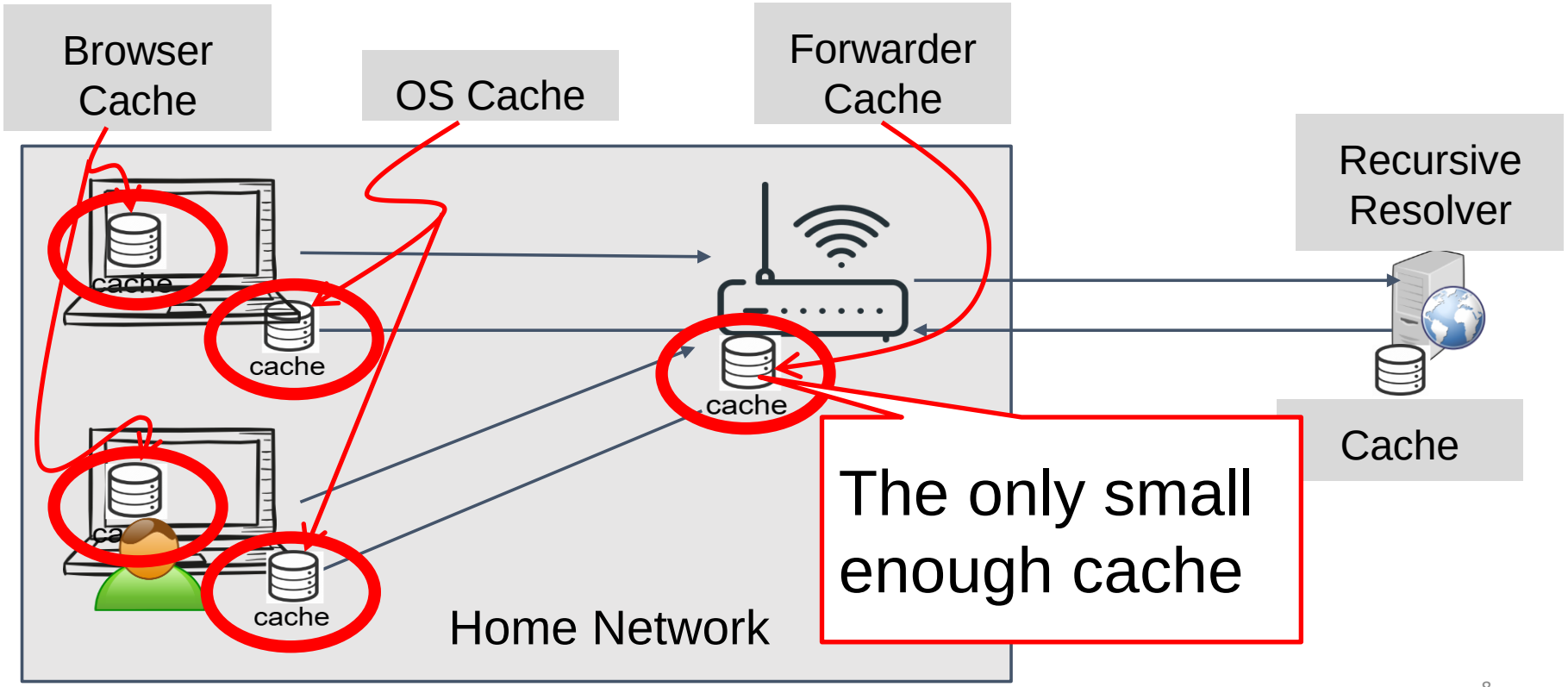
Challenges

- How to flush?
- Which cache?
- How to time a re-load?
- Circumvent other caches?

Solutions idea

- DNS Cache Flush [UsenixSec24]
- Forwarder cache small enough
- (1) Port 0 technique (2) HTTP reload
- Different domain origins

DNS cache levels, at home



✗ 2 major challenges

Challenges		Solutions idea
• How to flush?	✓	• DNS Cache Flush [UsenixSec24]
• Which cache?	✓	• Forwarder cache small enough
• How to time a re-load?	?	• (1) Port 0 technique (2) HTTP reload
• Circumvent other caches?	?	• Different domain origins

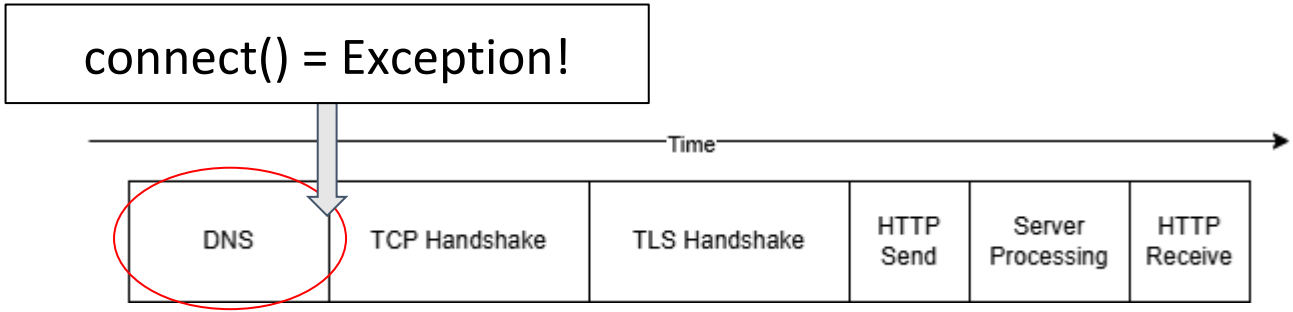
Cache Timing:

Challenge: JS cross-origin/site DNS time measurement impossible [PerformanceResourceTiming].

We designed two measurements techniques:

1) Direct DNS technique - request a **HTTP** resource from the **domain on port 0**.
   On most OSs: connect() + port 0 = Exception!

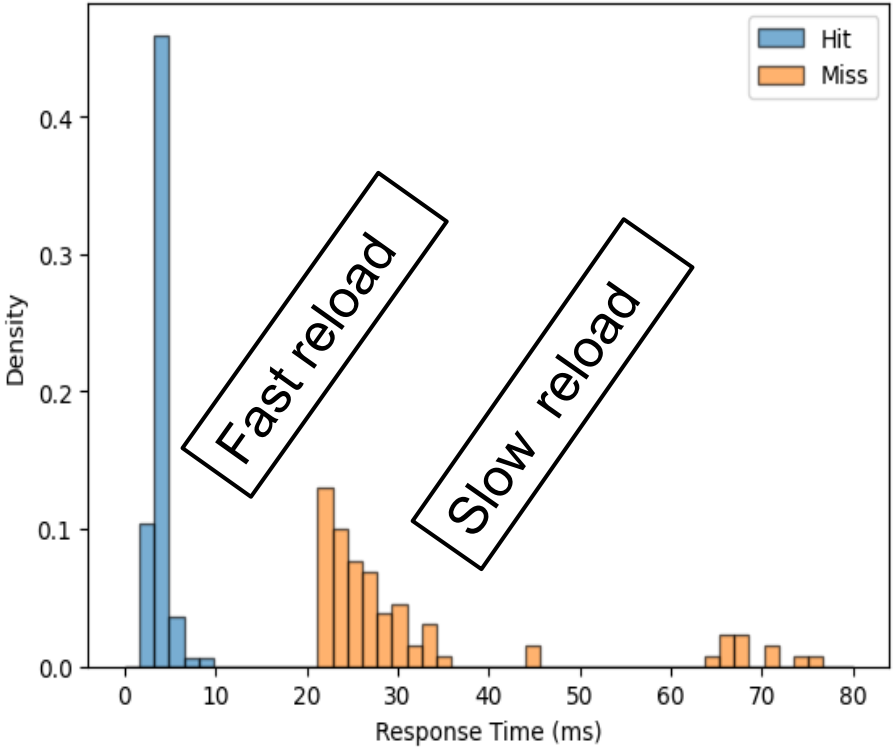
Still several technical difficulties which are discussed in the paper.



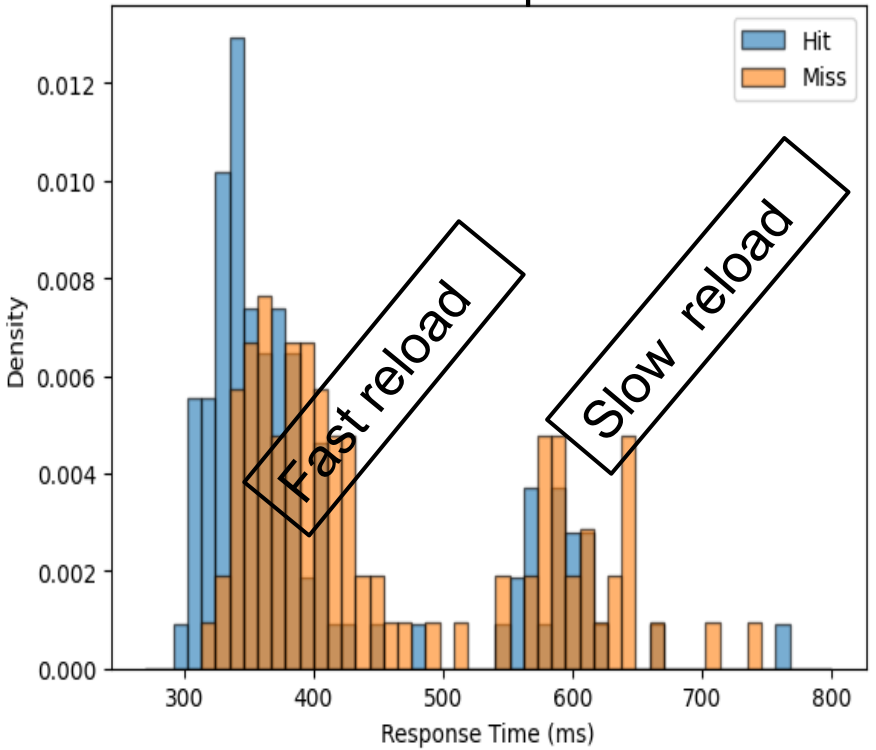
2) HTTP - measuring total time to receive an HTTP response.

Cache Timing

Method 1:
DNS Resolution Time



Method 2:
HTTP Resource Response Time



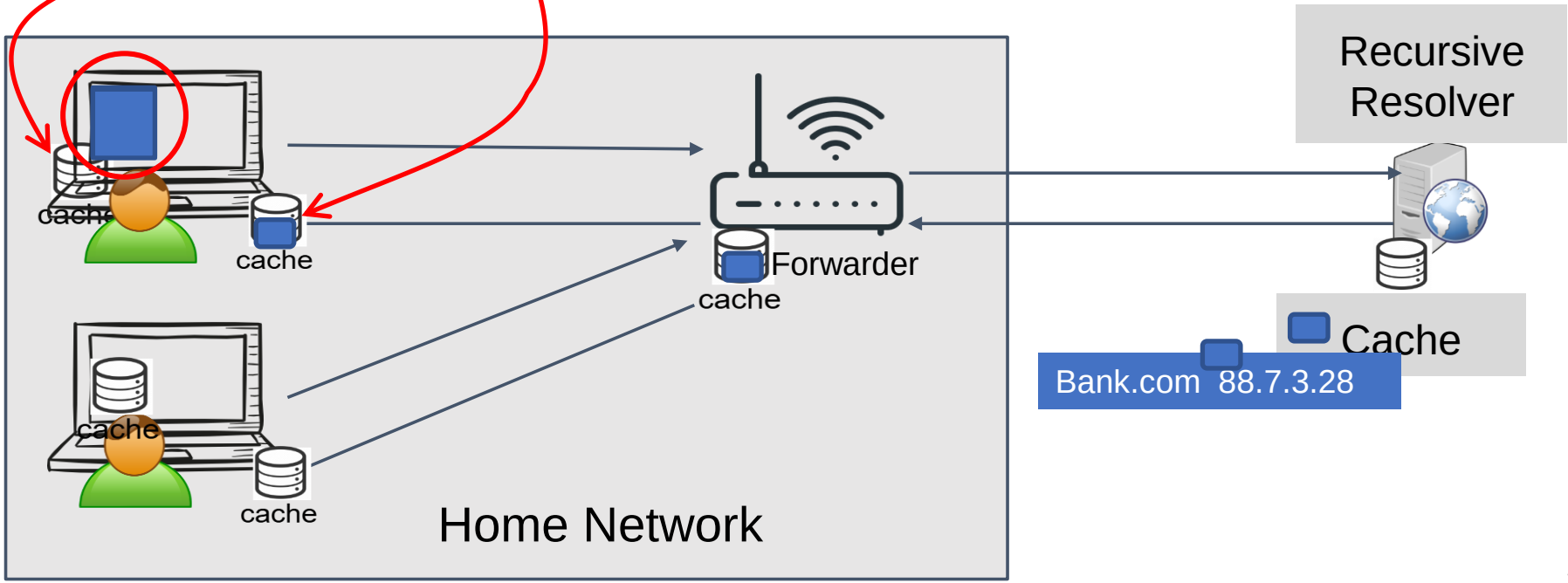
We use Gaussian Naive Bayes classifier on a per-website basis.

✗ 2 major challenges

Challenges		Solutions idea
• How to flush?	✓	• DNS Cache Flush [UsenixSec24]
• Which cache?	✓	• Forwarder cache small enough
• How to time a re-load?	✓	• (1) Port 0 technique (2) HTTP reload
• Circumvent other caches?	?	• Different domain origins

Ongoing Timing Measurements

1. Problem: Browser & OS caches !! before the forwarder

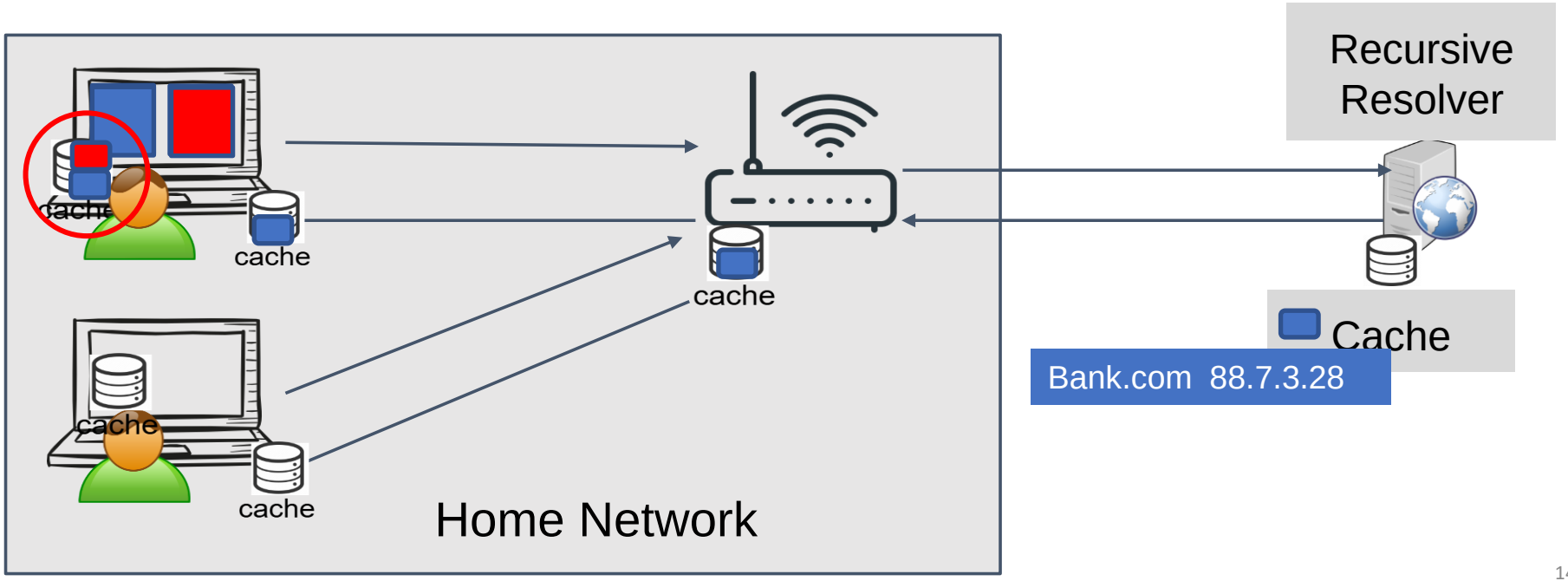


2. Mechanism: Network State Partitioning

[Chromium team mitigating Felten et. al. attack]

Partitions the browser's DNS cach, by origin domain ( )

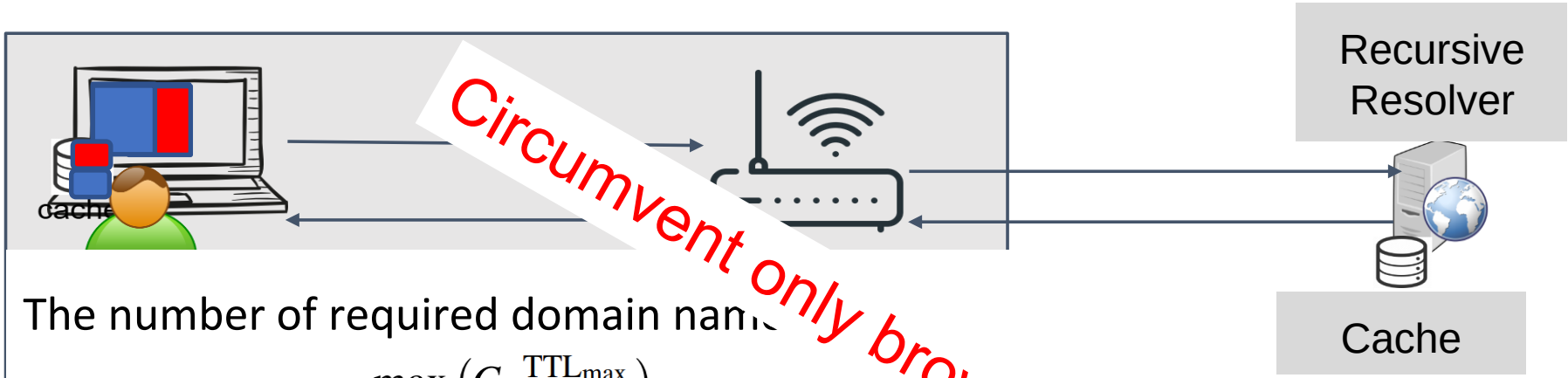
Different origins different cache!



3. Solution:

Reload measurement with different origin !

Use Network State Partitioning! Buy extra domains: attack1.com, attack2.com…



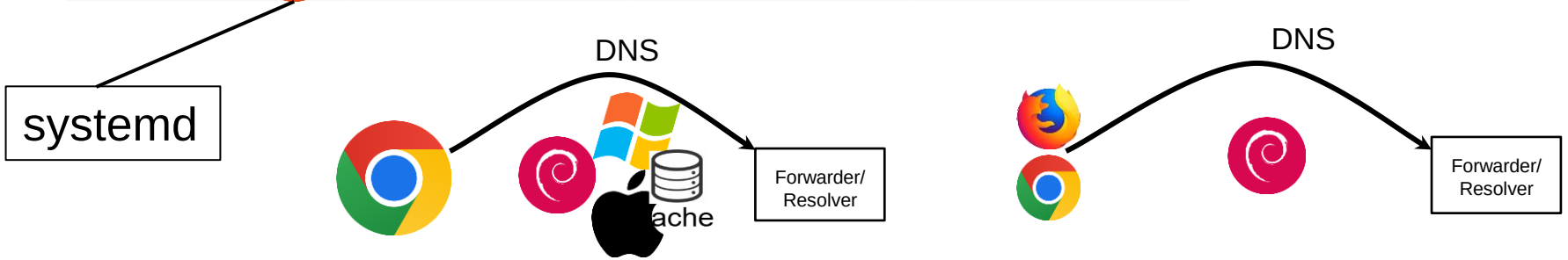
The number of required domain nam

$$\max\left(C, \frac{\mathrm{TTL}_{\max}}{I}\right)$$

- C - Number of iterations in the calibration
- TTL_{max} - Maximum authoritative-TTL
- I - Time interval between measurements

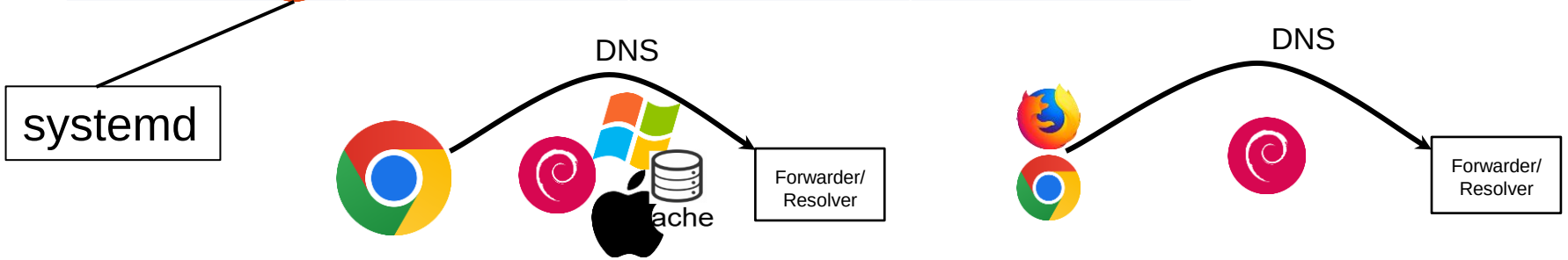
Vulnerable {OS x Browser} configurations

OS \ Browser	Chromium	Safari	Firefox
Windows			
Mac OS			
Linux			
Linux			



Vulnerable {OS x Browser} configurations

OS \ Browser	Chromium	Safari	Firefox
Windows	HTTP, DNS ✓	N/A	✗
Mac OS	HTTP, DNS ✓	✗	✗
Linux	HTTP ✓	N/A	HTTP ✓
Linux	HTTP ✓	N/A	HTTP ✓



✓ 4 major challenges

Challenges		Solutions idea
• How to flush?	✓	• DNS Cache Flush [UsenixSec24]
• Which cache?	✓	• Forwarder cache small enough
• How to time a re-load?	✓	• (1) Port 0 technique (2) HTTP reload
• Circumvent other caches?	✓	• Different domain origins

DNS FlaRE using Multiple FQDNs per Website

Why use one FQDN for a website when you can use multiple!?

Visiting x.com goes to:

1. x.com:
2. https://abs.twimg.com/responsive-web/client-web/vendor-91c40cd8.e33ef86a.js
3. https://video.twimg.com/
4. abs-0.twimg.com,
5. api.x.com,
6. twitter.com

Classification

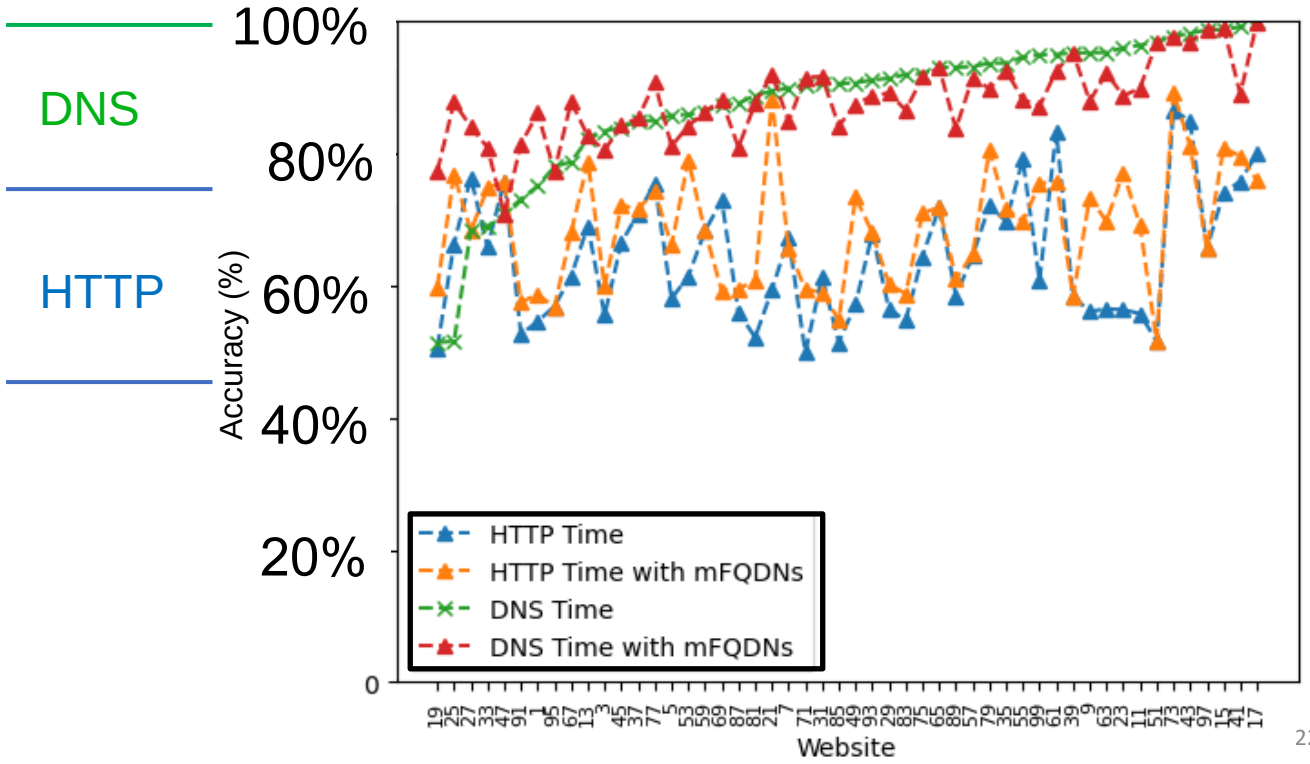
We use Gaussian Naive Bayes classifier on a per-website basis.

In the multiple-FQDNs approach, the response times of each domain serve as individual features .

Experimental Results

Results

86% avg. accuracy in hit/miss classification.



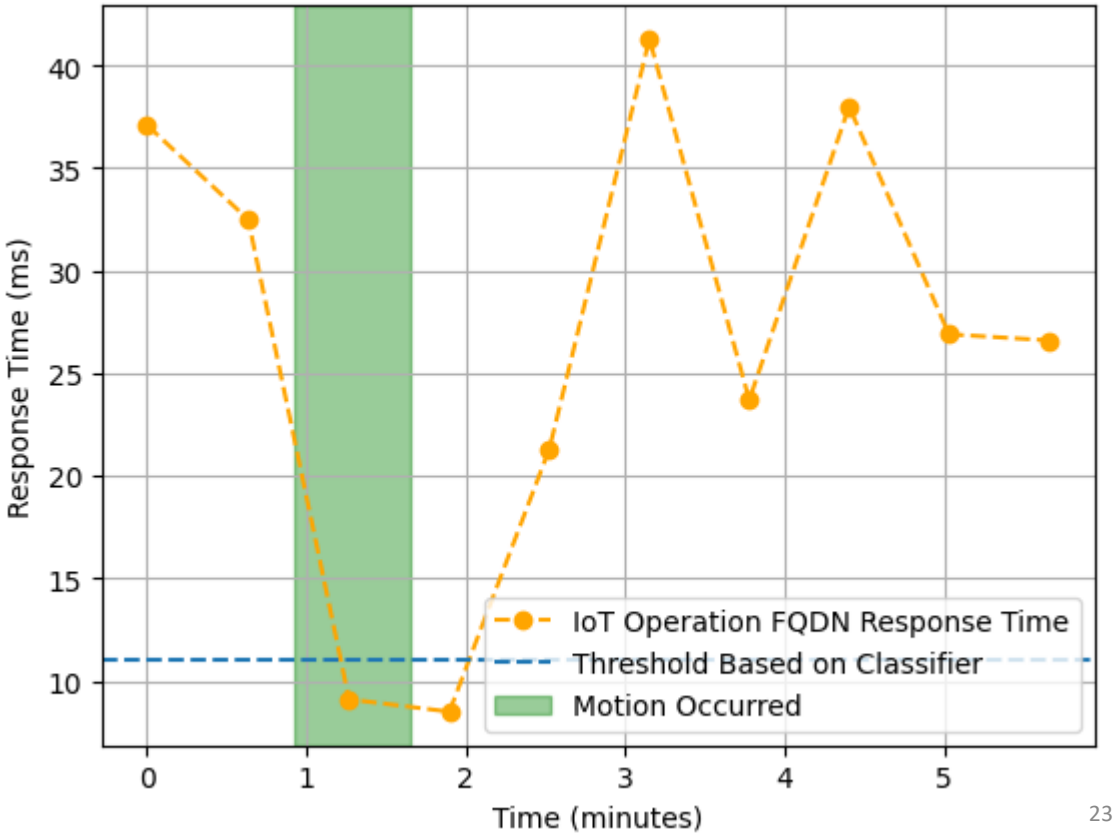
IoT profiling

Movement at victim's home

Get alert-3.eye4.com

alert-3.eye4.com is now in DNS cache

Alert movement detected



Responsible Disclosure

- Chromium: Blocked Port 0 in recent releases
- Opera: Updated base to Chromium 135 (which includes the fix)
- Safari :Patched and issued CVE-2025-43227.
- systemd: Acknowledged the issue; will open a public ticket post-embargo
- dnsmasq: Declined changes; noted cache size is configurable

Thank you.

For more information
https://deepness-lab.org