Mitigation of JavaScript-Based Data Generating Fingerprint Attacks

Nathan Joslin and Phu H. Phung
Intelligent Systems Security Lab (ISSec-Lab)

University *of* Dayton
Department of Computer Science

What? Why? Who?

What is Digital Fingerprinting?

- **Digital Fingerprinting:** The collection of attributes associated with a browser or device to form a unique ‘fingerprint’.
- **Stateless**: Unlike cookies, no information is saved client-side.
- **Silent**: User is completely unaware.

Source: https://deltafingerprinting.com/

Applications and Motivations

- **Benign Applications:**
  - Ad Fraud detection
  - Bot detection
  - Multi-Factor Authentication

- **Malicious Applications:**
  - Cross-site Web Tracking
  - Social Media Linking
  - Malware Targeting
  - Revealing Private Information

- **Concerns:**
  - Involuntary tracking and revealing of sensitive information
  - Voluntary MFA and fraud prevention

Who Uses Fingerprinting?

Fingerprinting the Fingerprinters - Iqbal et al. (2021)

- **Estimated Usage**:
    - 30.60% of Alexa top 1K
    - 10.18% of Alexa top 100K
- **Other Measurements:**
    - 14% of News sites
    - 6% of Shopping
    - 2,349 domains serve fingerprinting scripts
        - 3.78% considered tracking by Disconnect

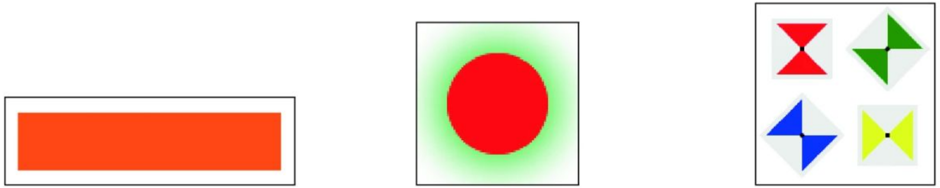
The Double Edged Sword - Senol, Ukani et al. (2024)

- **Estimated Usage**:
    - 25.75% of CrUX top 1K
    - 8.9% of CrUX top 100K
- **Other Measurements:**
    - 9.2% of Login Pages
    - 12.5% of Sign-up Pages
    - 60% of fingerprinting scripts on Login or Sign-up Pages use the Canvas API

Requirements of a Good Fingerprint

Fingerprint Entropy

- **Uniqueness Requirement:**
  - Must be able to distinguish between fingerprints
  - Increasingly difficult as the fingerprint dataset grows.
- **Consider:**
  - Feature dependencies
  - Domains of feature values



(a) *One rectangle (Entropy: 2.32 bits)*

(b) *Two ellipses (Entropy: 4.15 bits)*

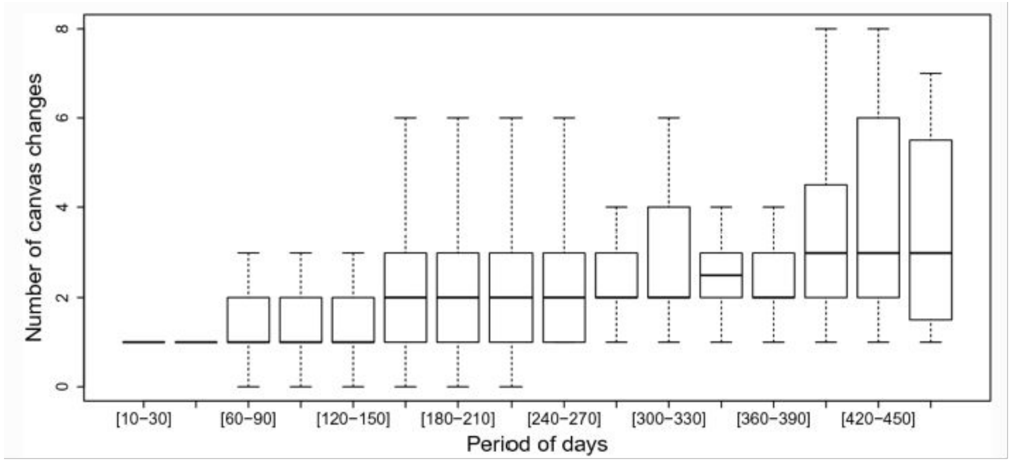
(c) *Four squares (Entropy: 4.66 bits)*

Cwm fjordbank glyphs vext quiz, 😃

Source: Laperdrix et al. (2019)

Fingerprint Stability

- **Stability Requirement:**
  - A fingerprint is only useful if it may be used for identification in the future.
- **Consider**:
  - Fingerprint type
  - Application needs
- **Improving Stability**:
  - Various algorithms have been developed to “reconstruct” a fingerprint.
- About 89% of desktop fingerprints are trackable overtime.
  - Source: Pugliese et al. (2020).

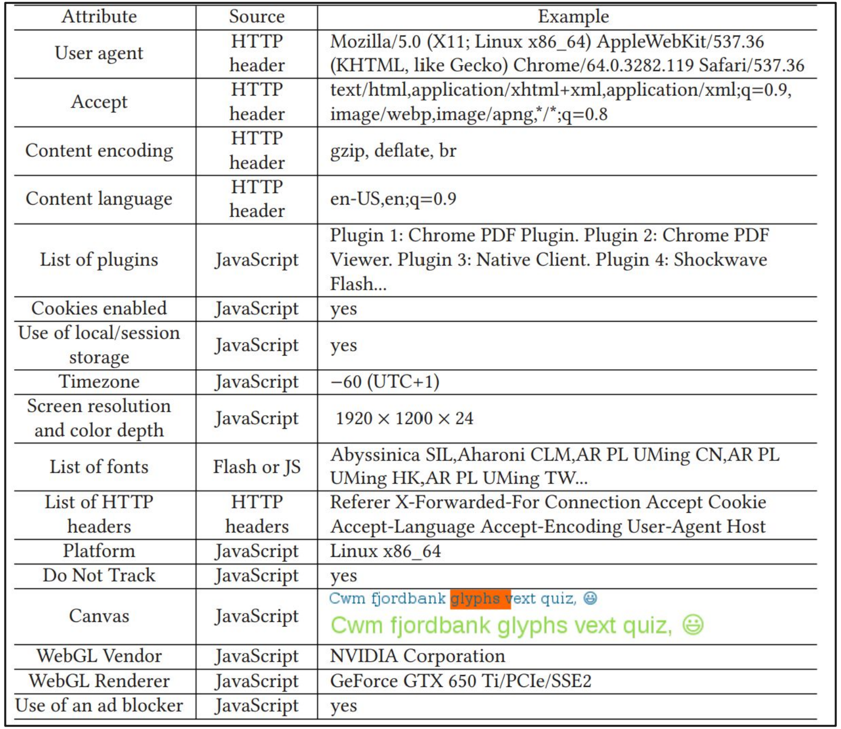


Source: Laperdrix et al. (2019)

Getting into JavaScript Based Fingerprinting

JavaScript Object-Based

- **Attack Vector:** JavaScript Objects
- **Goal:** Gather a variety of unique browser attributes by accessing object properties.
- **Ex.** Navigator objects for browser detection: Browser Type, User-Agent, etc.
- **Ex.** Screen objects for display configuration.

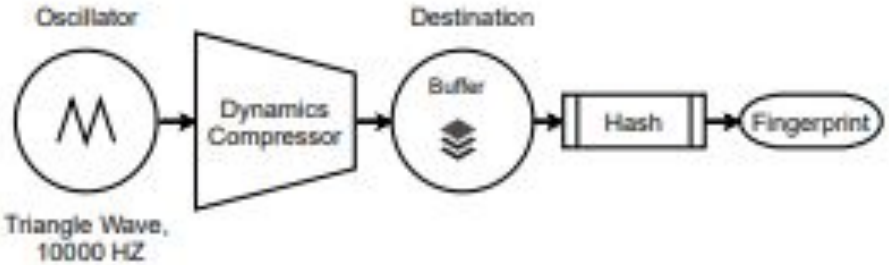


Attribute	Source	Example
User agent	HTTP header	Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.119 Safari/537.36
Accept	HTTP header	text/html,application/xhtml+xml,application/xml;q=0.9, image/webp,image/apng,*/*;q=0.8
Content encoding	HTTP header	gzip, deflate, br
Content language	HTTP header	en-US,en;q=0.9
List of plugins	JavaScript	Plugin 1: Chrome PDF Plugin. Plugin 2: Chrome PDF Viewer. Plugin 3: Native Client. Plugin 4: Shockwave Flash...
Cookies enabled	JavaScript	yes
Use of local/session storage	JavaScript	yes
Timezone	JavaScript	−60 (UTC+1)
Screen resolution and color depth	JavaScript	1920 × 1200 × 24
List of fonts	Flash or JS	Abyssinica SIL,Aharoni CLM,AR PL UMing CN,AR PL UMing HK,AR PL UMing TW...
List of HTTP headers	HTTP headers	Referer X-Forwarded-For Connection Accept Cookie Accept-Language Accept-Encoding User-Agent Host
Platform	JavaScript	Linux x86_64
Do Not Track	JavaScript	yes
Canvas	JavaScript	Cwm fjordbank glyphs vext quiz, 😃 Cwm fjordbank glyphs vext quiz, 😃
WebGL Vendor	JavaScript	NVIDIA Corporation
WebGL Renderer	JavaScript	GeForce GTX 650 Ti/PCIe/SSE2
Use of an ad blocker	JavaScript	yes

Source: Laperdrix et al. (2020)

Audio Context

- **Attack Vector:** Web Audio API
- **Goal:** Generate unique data by processing audio signals
- Less stable than other methods
  - Improved with clustering algorithms
- Lower entropy than other methods
  - 2-2.5 bits of entropy



Canvas

- **Attack Vector:** Canvas API
- **Goal:** Generate unique data by drawing a Canvas graphic, sometimes called a *challenge.*
- Maximum entropy is extracted by using pangrams, complex curves, and gradients.
  - **Max:** 13.87 bits

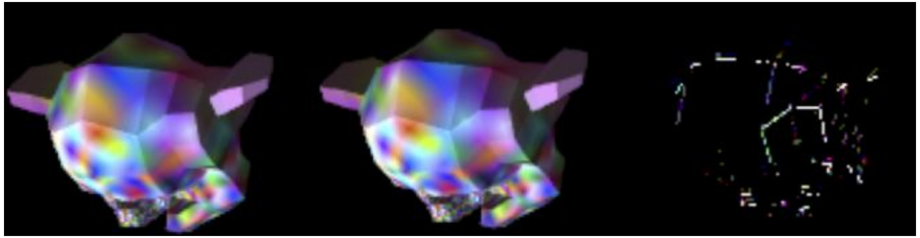
Cwm fjordbank glyphs vext quiz,
Cwm fjordbank glyphs vext quiz,

Cwm fjordbank glyphs vext quiz,

Sources: amiunique.org, Laperdrix et al. (2019)

WebGL

- **Attack Vector:** HTML Canvas Element
- **Goal:** Generate unique data by using the WebGL 3D graphics API
- A specific type of Canvas fingerprinting.



iMac Windows Difference

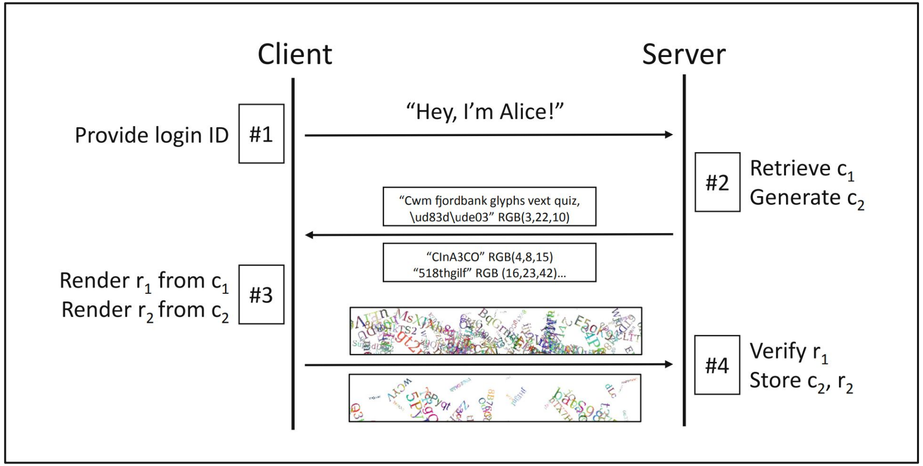
Wu, Shujiang et al. (2019).

Specific Applications: Benign vs. Malicious

Benign Application

Challenge/Response-Based Authentication:

- Proposed as another layer in a multi-factor authentication scheme
- This protocol authenticates a *device* using Canvas elements as a vector to generate a unique fingerprint.
- **Limitations:**
  - Works after the first visit
  - Requires a fallback mechanism such as SMS



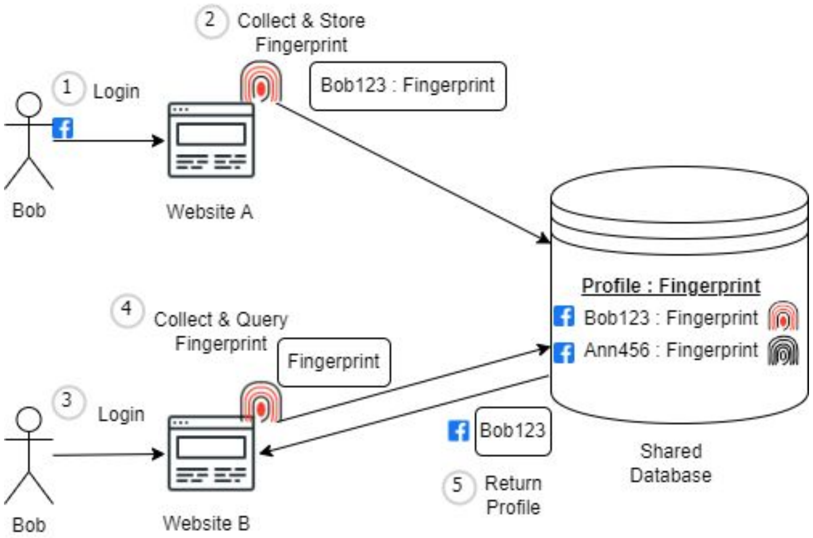
A challenge/response-based authentication mechanism proposed by Laperdrix et al. (2019).

Malicious Application

<u>Social Media Linking:</u>

- A theoretical scheme
- Significantly strengthens cross-site tracking by linking fingerprints to social media accounts
- **Limitations:**
  - Requires publicly available profiles
  - Requires collaboration between web applications

Source: [Khademi et al. (2015)]

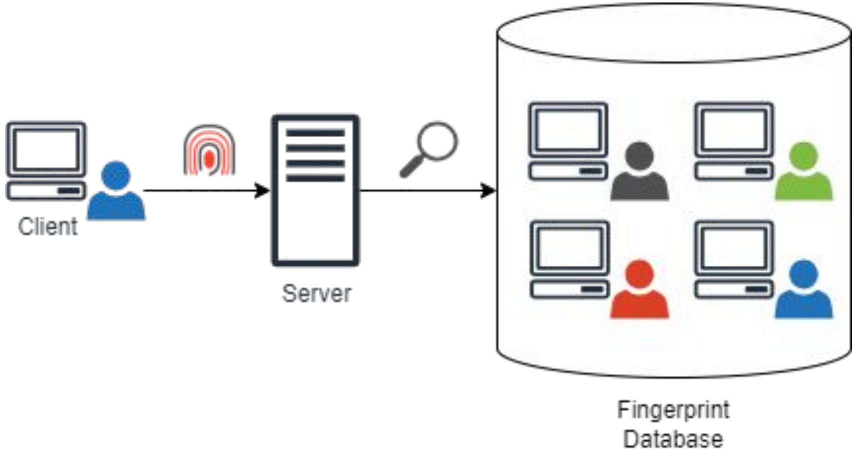


Overview

- What? Why? Who?
- Requirements
- Types
- Duality of Applications
- Mitigation Approaches ⬅
- Mitigation Examples
- MyWebGuard
- Mitigation Experiments

Fingerprinting Mitigation

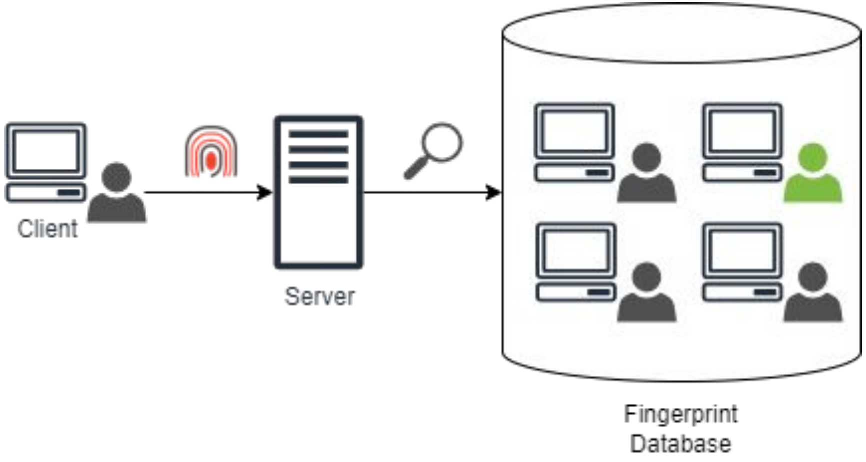
- **Policy Decision Making**
  - Machine Learning Based
  - Developer Defined
- **Enforcement Methods**
  - Normalization
  - Randomization
  - Interaction Blocking



Mitigation Approaches: Normalization

Normalization:

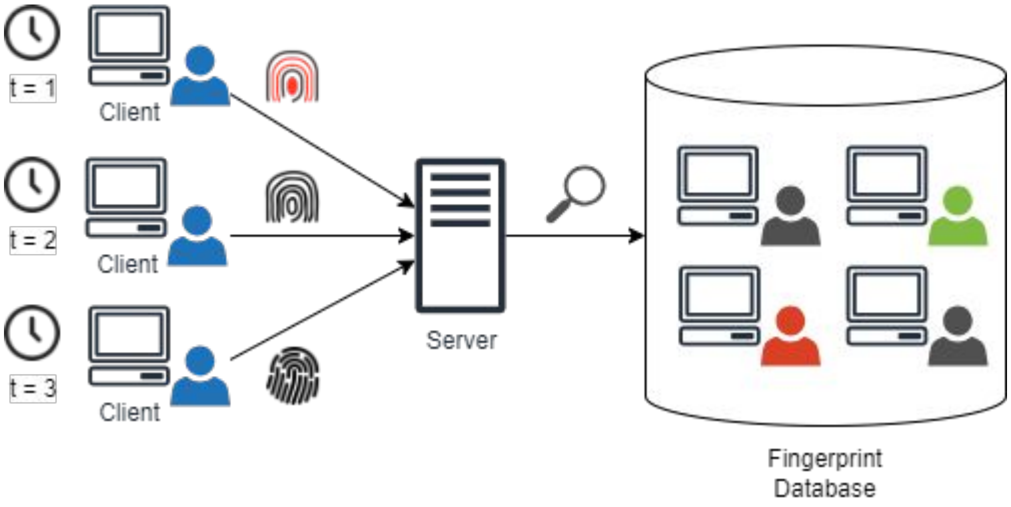
- **Goal:** "Hide in the crowd"
- Also known as *Attribute Standardizing*
- Reduces the uniqueness of fingerprints by setting attributes to a shared value.
- **Usage:** Actively used by Tor Browser, setting default values for many attributes.



Mitigation Approaches: Randomization

Randomization:

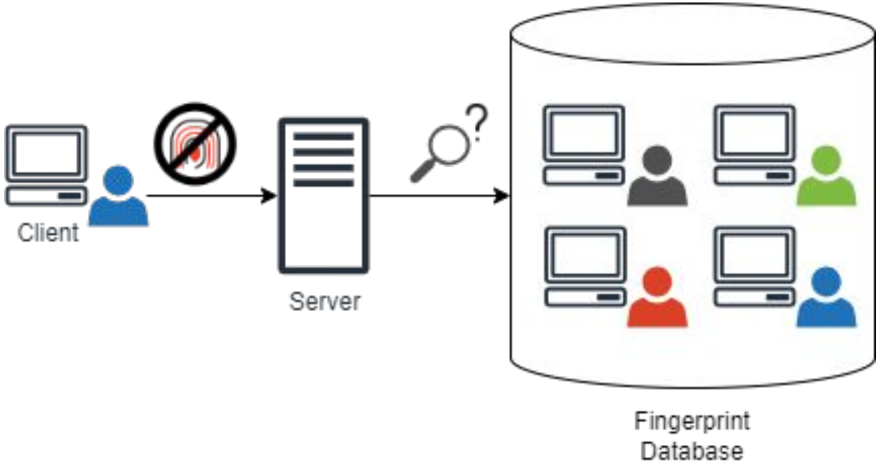
- **Goal:** “Moving target”
- Also known as *Attribute Varying*
- Increases the uniqueness of a particular fingerprint over time
- **Usage:** Used by Canvas poisoners to introduce noise to the collected data.
    - Brave Browser



Mitigation Approaches: Blocking

Blocking:

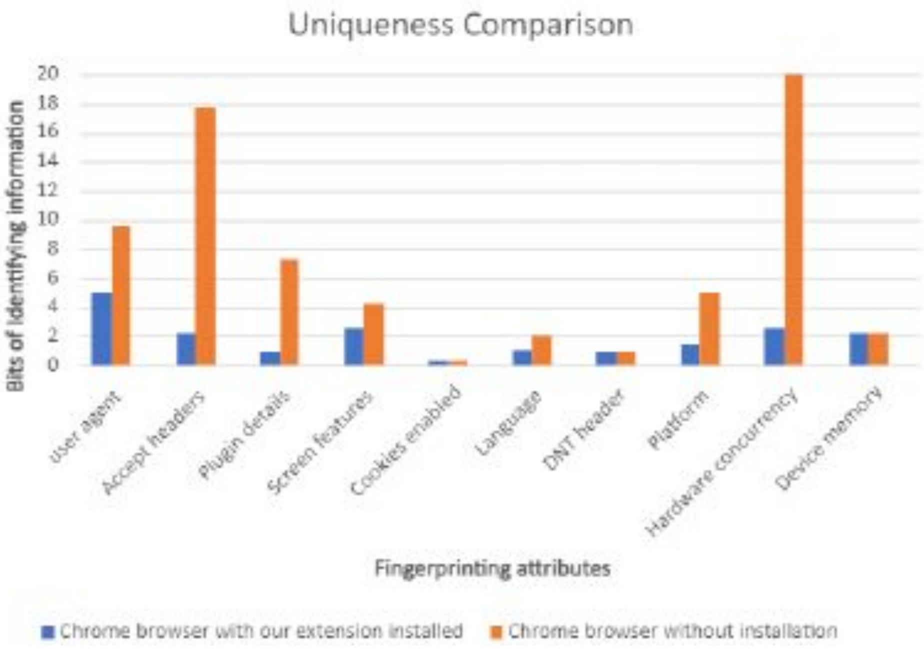
- **Goal:** Block by API or by domain
- Also known as *Interaction Blocking*
- **Usage:** API blocking is actively used by Tor Browser to prevent Canvas fingerprinting.



Normalization Example

***A Defense Against JavaScript Object-Based Fingerprinting and Passive Fingerprinting*: Ajay and Guptha(2022)**

- Spoofed HTTP headers to counter passive fingerprinters.
- Spoofed Navigator and Screen objects to counter active fingerprinters.
- **Limitations:**
  - Dynamic attributes are not protected



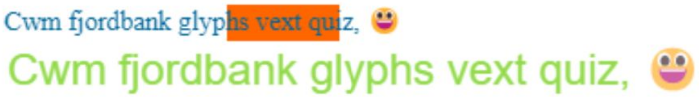
Source: Ajay and Guptha (2022)

Randomization Example

FP-Random: Laperdrix et al. (2017)

- Mostly protects against dynamic attributes
  - Ex. AudioContext, Canvas
- **Limitations:**
  - Static attributes are not protected
  - A "smart" fingerprinter may be able to detect the randomization

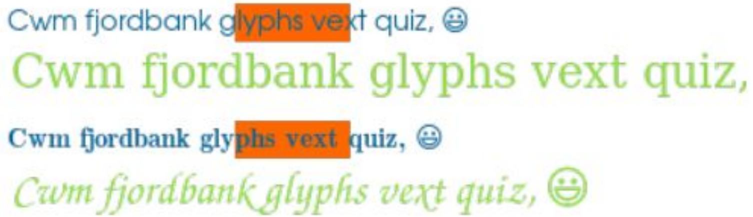
Original



Cwm fjordbank glyphs vext quiz, 😃

Cwm fjordbank glyphs vext quiz, 😃

Modified



Cwm fjordbank glyphs vext quiz, 😃

Cwm fjordbank glyphs vext quiz,

Cwm fjordbank glyphs vext quiz, 😃

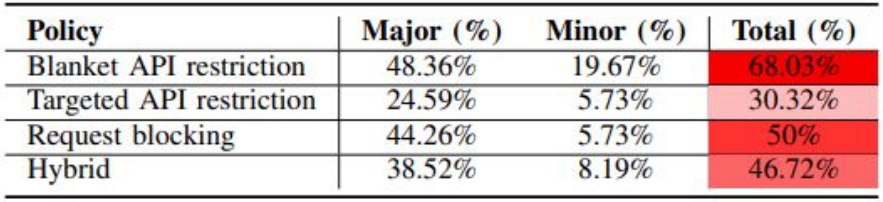
Cwm fjordbank glyphs vext quiz, 😃

Source: Laperdrix et al. (2017)

Blocking Example

FP-Inspector: Iqbal et al. (2021)

- An ML approach mostly designed for detection
- Static and dynamic analysis
- Blocking mitigation strategy shows high likelihood for site breakage



Policy	Major (%)	Minor (%)	Total (%)
Blanket API restriction	48.36%	19.67%	68.03%
Targeted API restriction	24.59%	5.73%	30.32%
Request blocking	44.26%	5.73%	50%
Hybrid	38.52%	8.19%	46.72%

Source: Iqbal et al. (2021)

Fine-Grained Policy Enforcement with IRMs

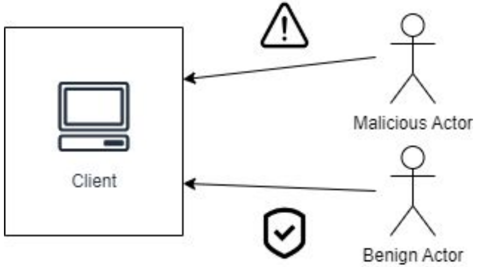
Problem Description

General Questions:

- Can we enforce policies on untrusted third-parties while allowing it from trusted organizations
- Can we enforce specific policies based on fingerprinter characteristics?

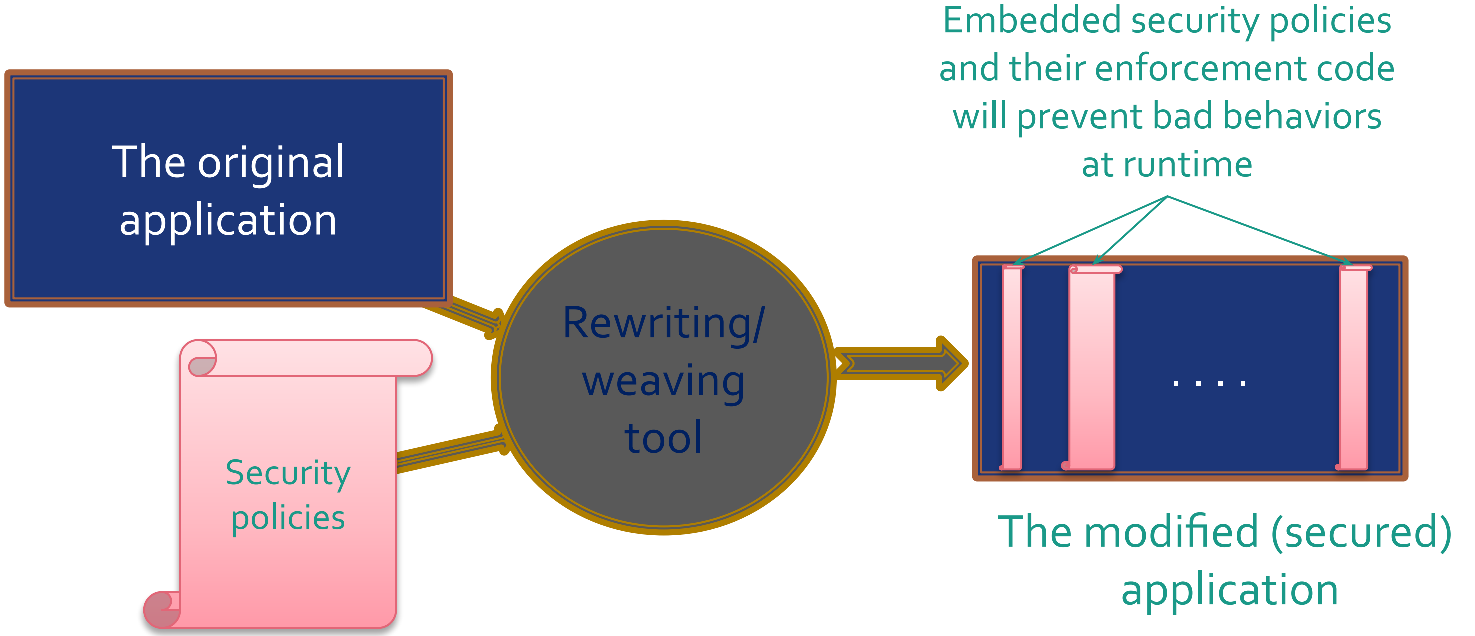
Specific Questions:

- Can we control fingerprinting by its origin?
- Can we apply known mitigation methods to prevent fingerprinting by type?



The Inlined Reference Monitor (IRM) Approach

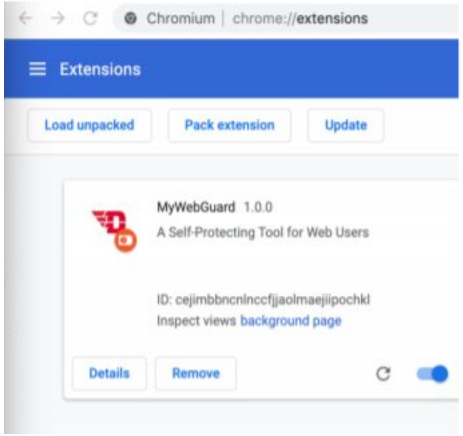
- IRM is a language-based security approach
  - Embed security enforcement code into applications



The original application
Security policies
Rewriting/ weaving tool
Embedded security policies and their enforcement code will prevent bad behaviors at runtime
. . . .
The modified (secured) application

An IRM Implementation in the browser: MyWebGuard

- Enforce code origin-based policy for any websites
    - Allow or disallow a JavaScript action based on
        - code origin
        - code behaviors
        - or user choice



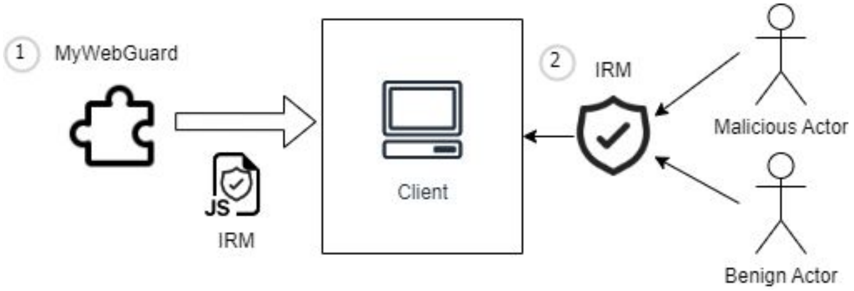
MyWebGuard 1.0.0
A Self-Protecting Tool for Web Users

ID: cejimbbncnlnccfjjaolmaejiipochkl
Inspect views background page

Proposed Mitigation Method

<u>MyWebGuard:</u>

- Inline Reference Monitor (IRM): Intercepts JS function calls or property accesses at runtime.
- Allows for *Code-Origin-Policy* enforcement.
- Fine-grained policies capable of handling both benign and malicious applications.

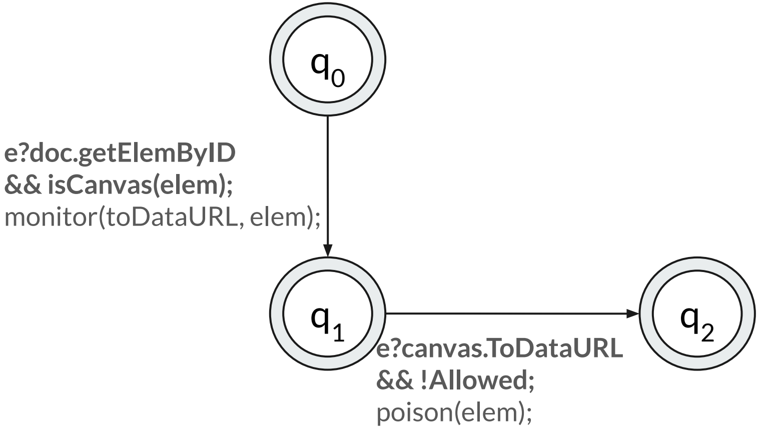


Experiments with Canvas Poisoning

Experiments with Canvas Fingerprinting

Canvas Fingerprinting Policy:

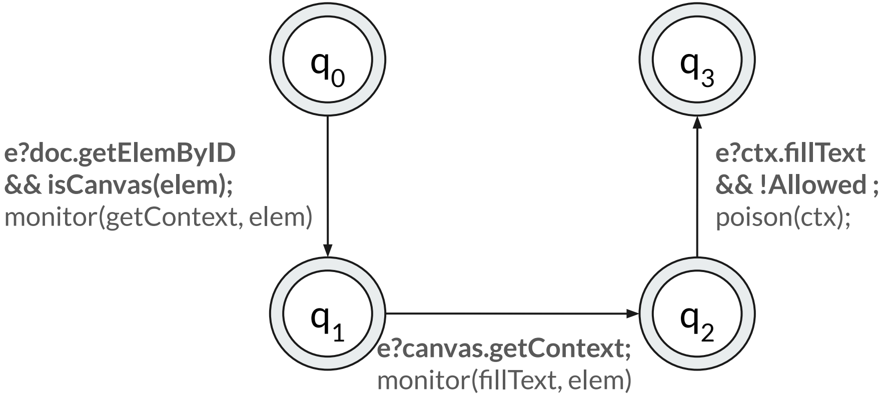
- **Goal**: Randomize the fingerprint.
- **Method**: Introduce noise *at the end* of the data collection process.
- **Event Set:** e = {...}
  - document.getElementByID
  - document.createElement
  - canvas.toDataURL



Experiments with Canvas Fingerprinting

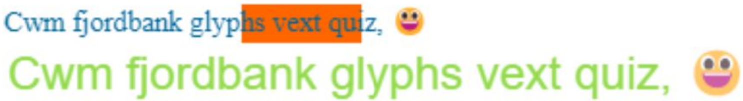
Canvas Fingerprinting Policy:

- **Goal**: Randomize the fingerprint.
- **Method**: Introduce noise *throughout* the data collection process.
- **Event Set:** e = {...}
  - document.getElementByID
  - document.createElement
  - canvas.getContext
  - context.fillText

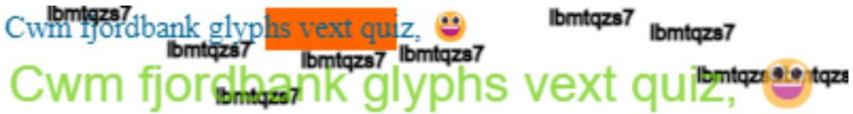


Experiments with Canvas Fingerprinting

Base Canvas Image


Cwm fjordbank glyphs vext quiz,
Cwm fjordbank glyphs vext quiz,

Poisoned Canvas Image


Cwm fjordbank glyphs vext quiz,
Cwm fjordbank glyphs vext quiz,
lbmtqzs7
lbmtqzs7
lbmtqzs7
lbmtqzs7
lbmtqzs7
lbmtqzs7
lbmtqzs7

Testing Tool Used: https://amiunique.org/

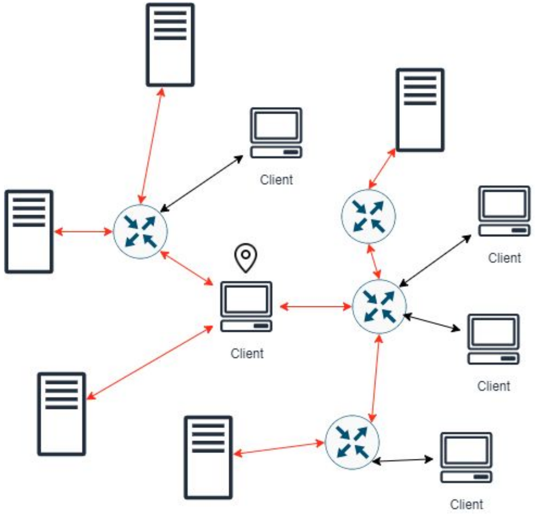
Creating a Link-Based Fingerprinter

PingLoc: The Link-Based Fingerprinting Prototype

Multilateration Cross-Site Image Resource Request Scheme

1. **Network Sampling:** Collect time-delay information using cross-site image requests, a *ping*.
2. **Data Processing:** Select appropriate data window, Remove lost packets.
3. **Feature Extraction:** Min, Max, Mean, Variance, Root-Mean-Square, Skew, Kurtosis.
4. **Model Training/Classification:** Feature vectors are used to train a machine-learning model, and are later used to classify/localize user browsers.

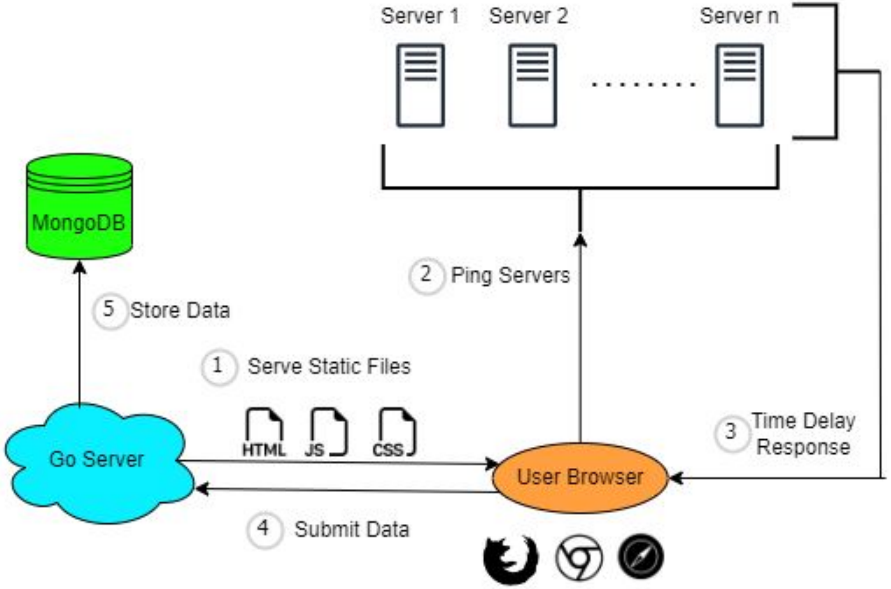
Source: Wu et al. (2021)



Red links indicate the paths taken by *pings*.

Implementing PingLoc

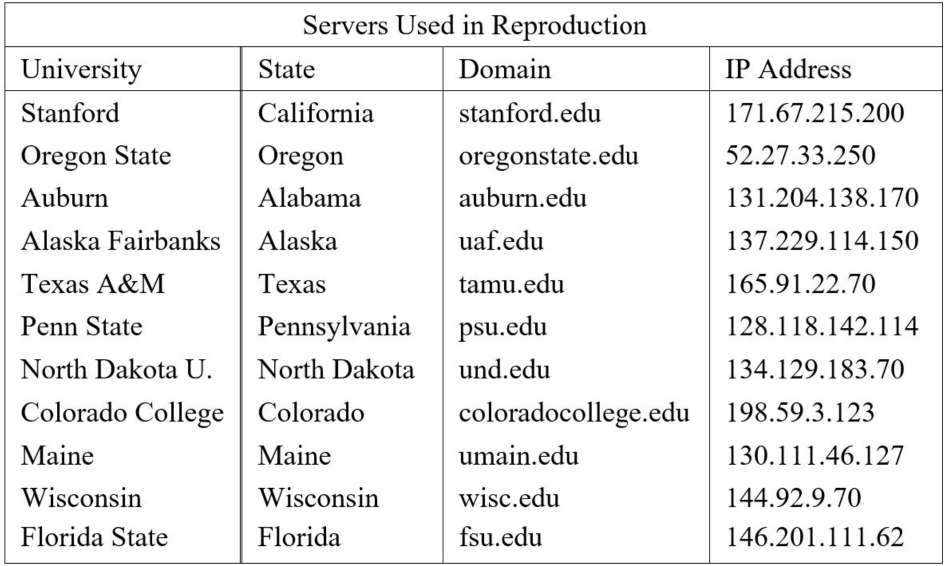
- **Modifications to Original Method**:
  - Different Set of Servers
  - Different Type of Servers: University, rather than industry
  - Request via IP Address, rather than domain
- **Goals**:
  - Increase accuracy
  - Increase robustness



Web App: https://mywebguard-antifingerprinting2-ea23e7d63788.herokuapp.com/

Concerns with CDNs

- **Assumption:** Large companies such as Google or Twitter are likely to use Content Delivery Networks (CDNs).
- **Inference:** CDNs may interfere with machine-learning classification, resulting in decreased accuracy.
- **Proposed Solution:**
  - Send requests with IP address, rather than domain name.
  - Send to Universities, as opposed to Industry



Servers Used in Reproduction

University	State	Domain	IP Address
Stanford	California	stanford.edu	171.67.215.200
Oregon State	Oregon	oregonstate.edu	52.27.33.250
Auburn	Alabama	auburn.edu	131.204.138.170
Alaska Fairbanks	Alaska	uaf.edu	137.229.114.150
Texas A&M	Texas	tamu.edu	165.91.22.70
Penn State	Pennsylvania	psu.edu	128.118.142.114
North Dakota U.	North Dakota	und.edu	134.129.183.70
Colorado College	Colorado	coloradocollege.edu	198.59.3.123
Maine	Maine	umain.edu	130.111.46.127
Wisconsin	Wisconsin	wisc.edu	144.92.9.70
Florida State	Florida	fsu.edu	146.201.111.62

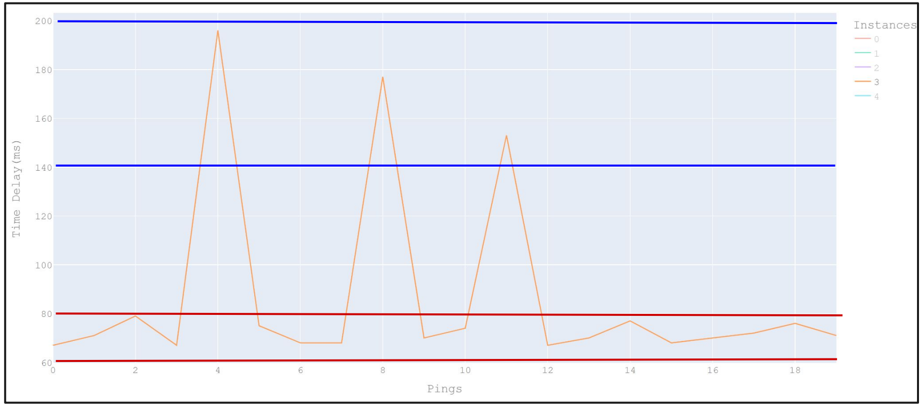
A table of geographically diverse universities. IP addresses were obtained using *nslookup*.

Proposed Features

- **Inspiration:** Turky N Alotaiby et al. [(2019)]
- **Goal:** Capture the typical behavior of a network when congested.

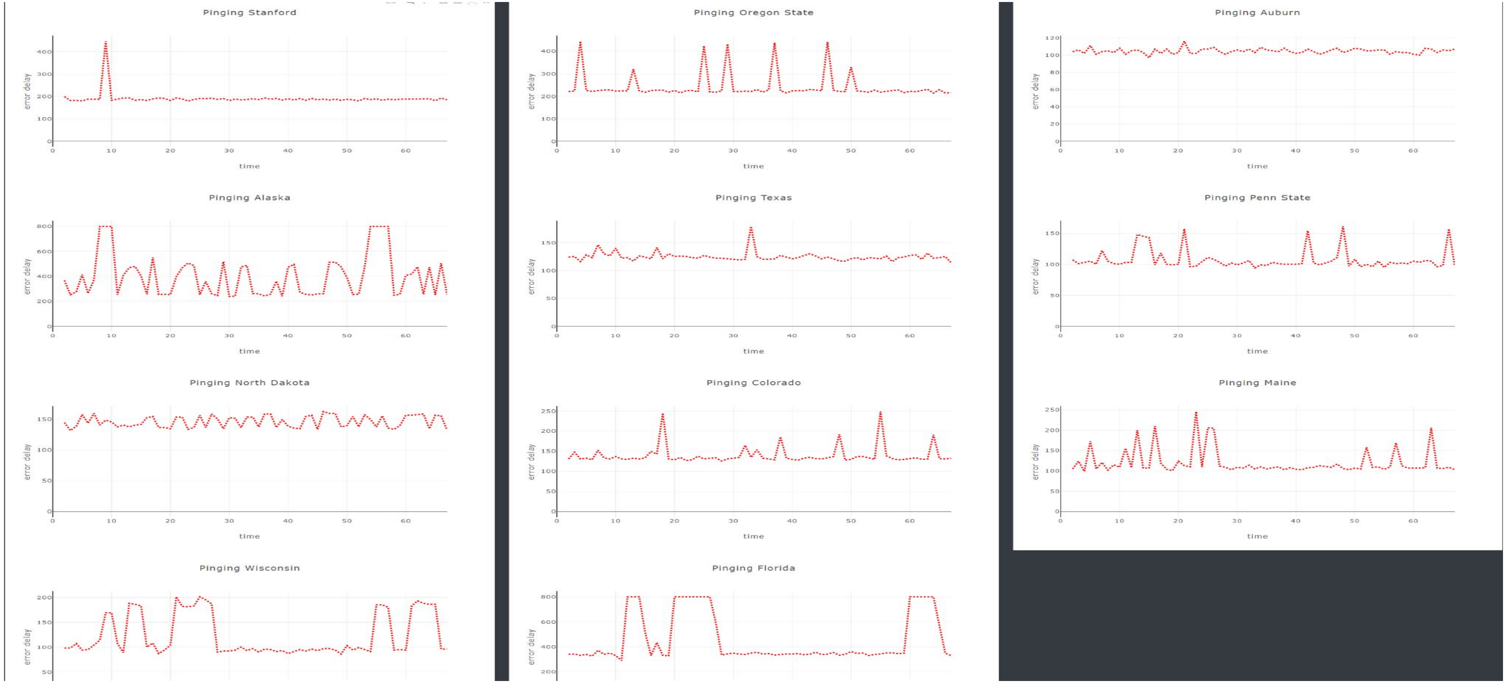
Additional Features:

1. Interquartile Range
2. Interquartile First Quarter (Q1)
3. Interquartile Third quarter (Q3)
4. Number of Lost Timed-Out Requests

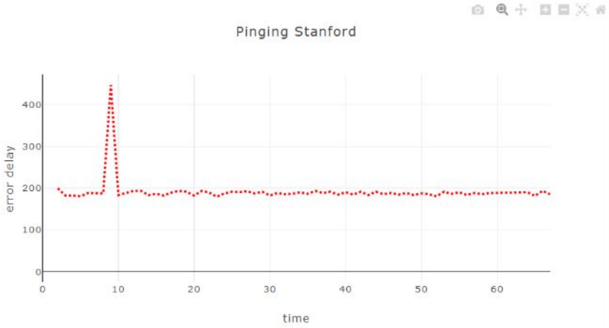


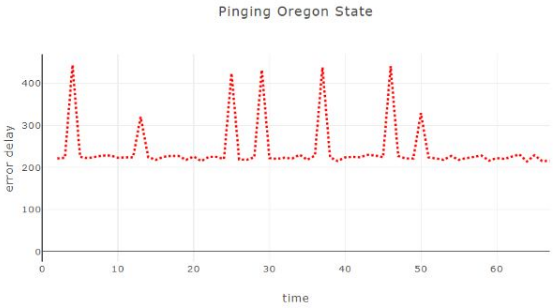
A single trace of time-delay values of one user to a university server. *Congested* network behavior is between the blue lines. *Non-Congested* network behavior is between the red lines

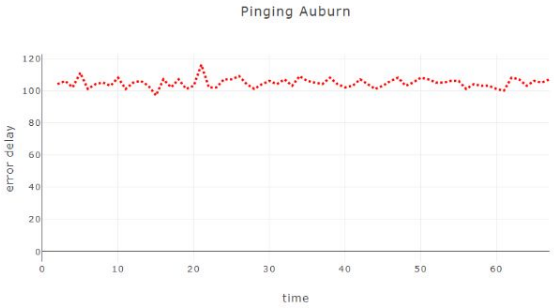
Data Collection: User’s Perspective



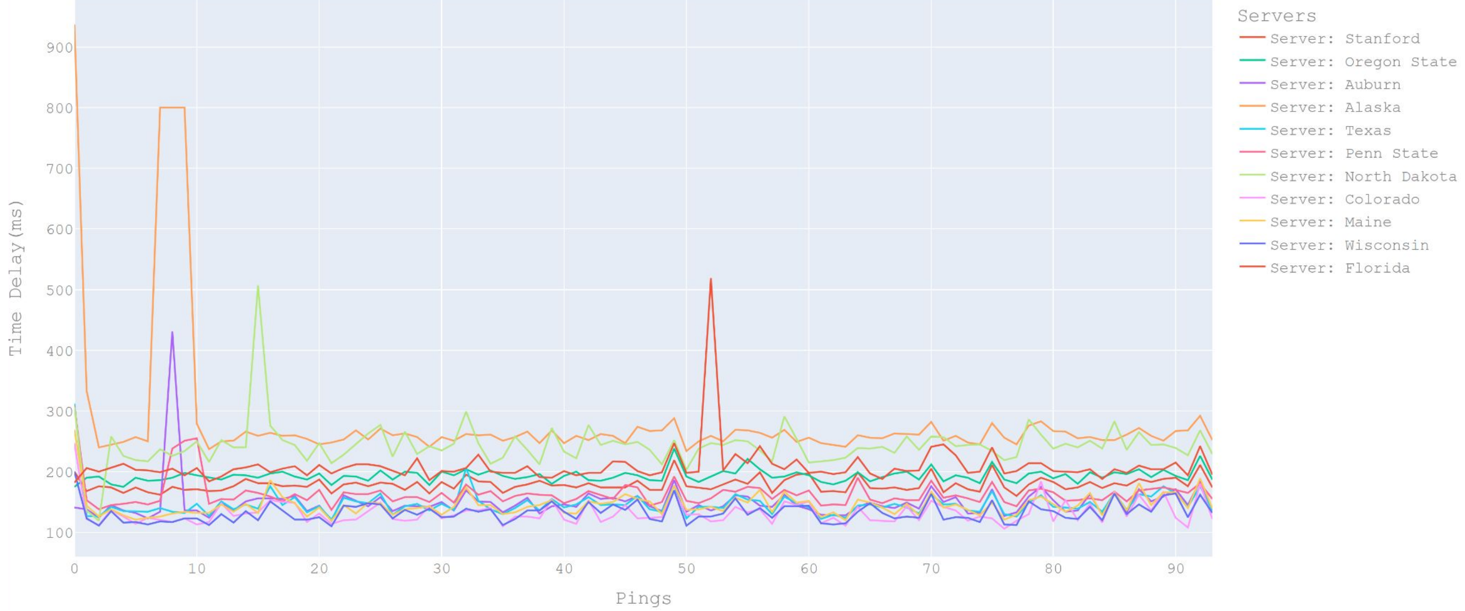
Data Collection: User’s Perspective


Pinging Stanford

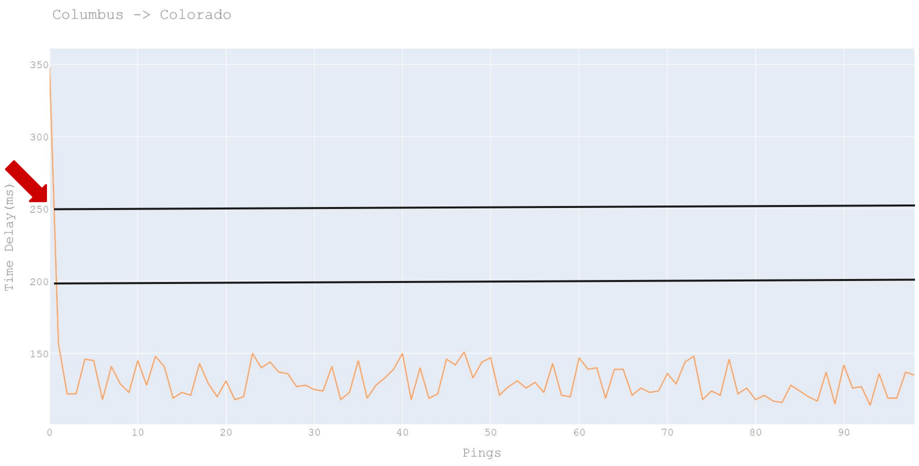

Pinging Oregon State


Pinging Auburn

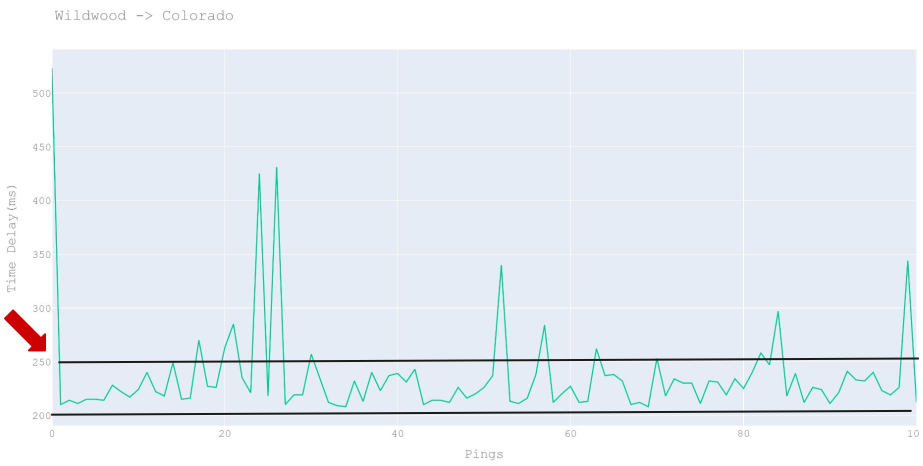
Example: Complete Data Collection Instance



Location Differences



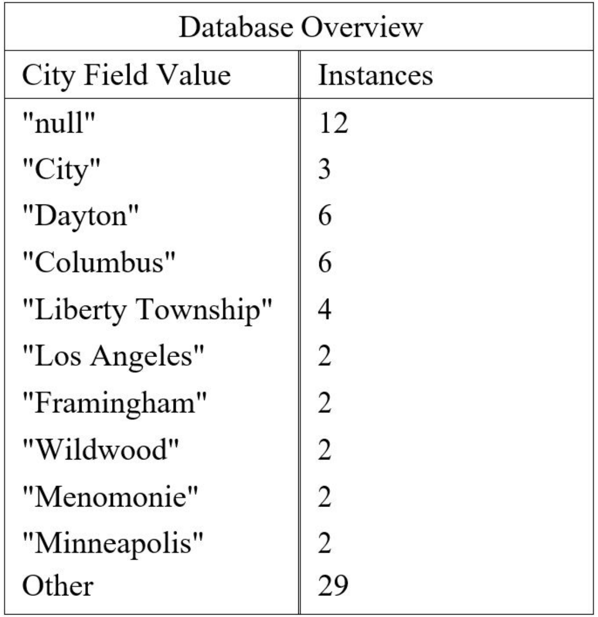
Columbus, OH



Wildwood, NJ

Data Collection

- **Collection Process:**
  - Chose to reach out to *real users*
  - Emails to friends and family
  - Posts on LinkedIn
- **Data Collected:**
  - 21% invalid user input
  - 41% single user, single city



Database Overview	
City Field Value	Instances
"null"	12
"City"	3
"Dayton"	6
"Columbus"	6
"Liberty Township"	4
"Los Angeles"	2
"Framingham"	2
"Wildwood"	2
"Menomonie"	2
"Minneapolis"	2
Other	29

Overview of the data obtained during the collection process. *Other* includes all cities with one participant.

Synthetic Data Generation

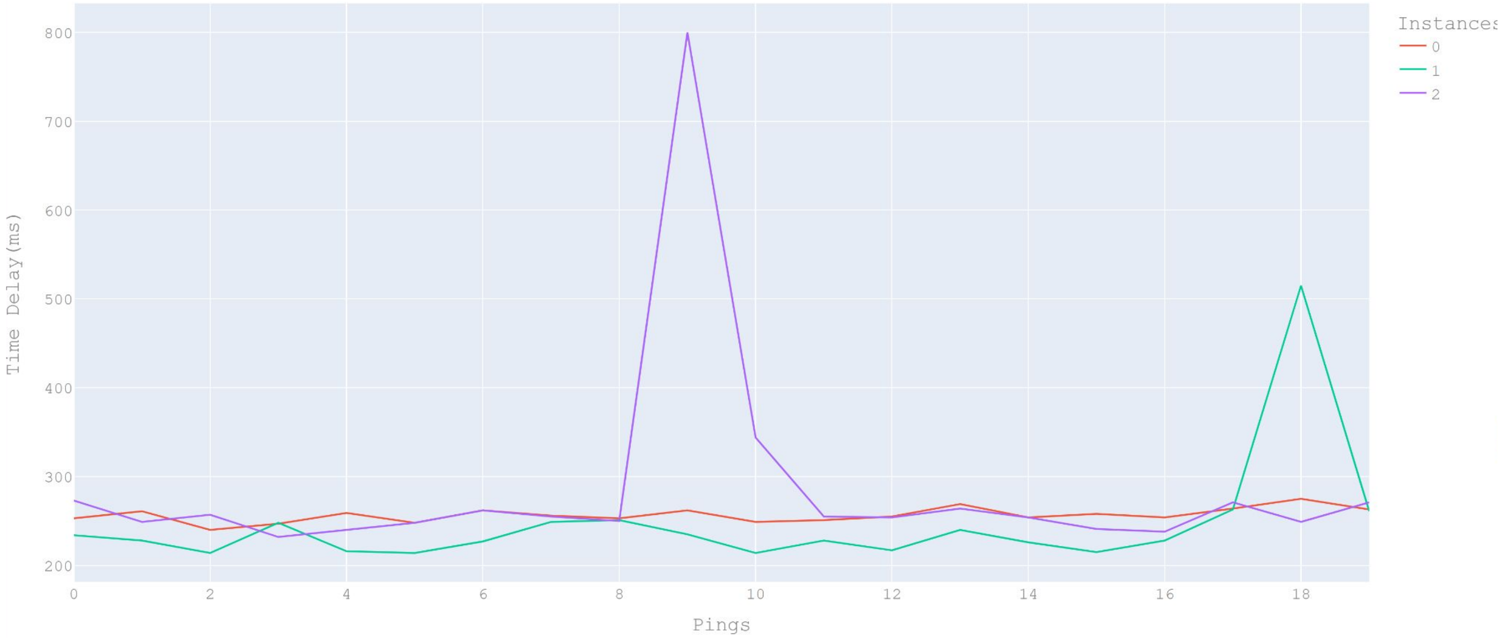
- **Data Generation:** Real data was duplicated with randomized noise added to each time-delay value.
- **Limitation:** Randomized noise may not accurately reflect actual network link-state behavior

Parameters:

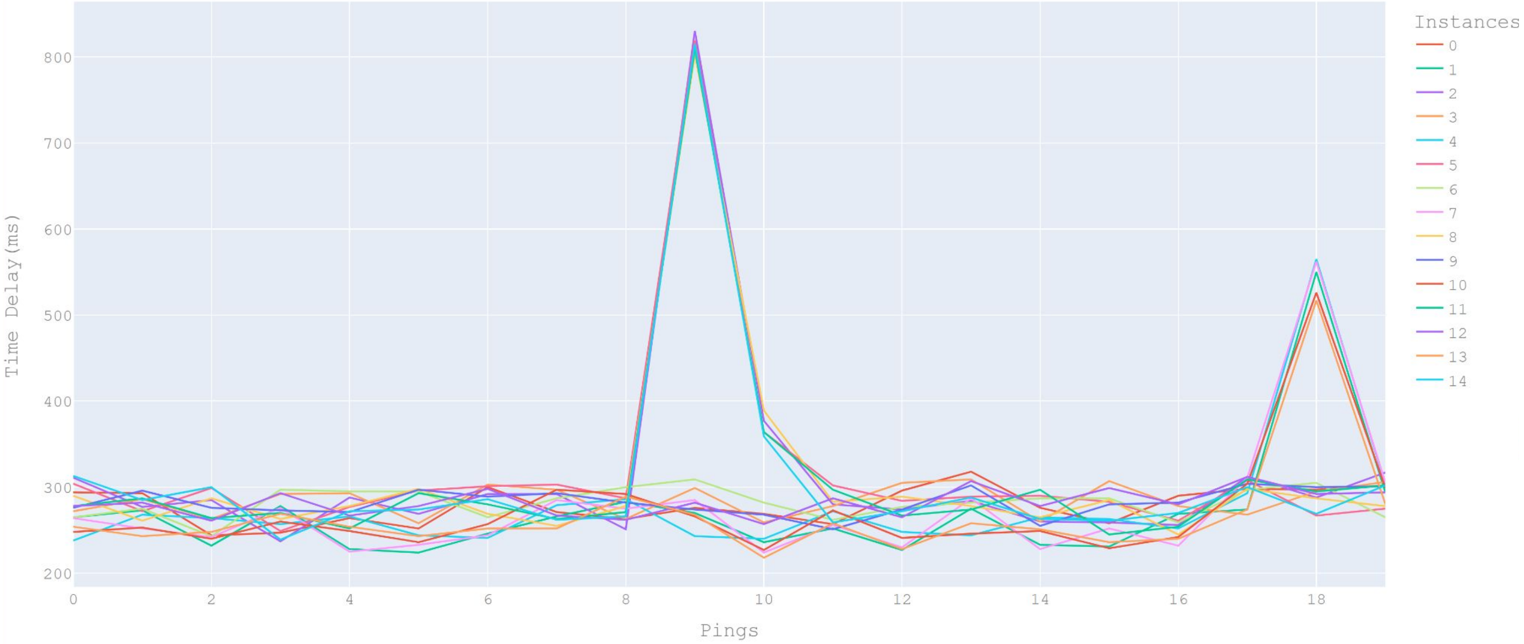
- **Window_of_Pings:** An array of time-delay values of length N.
  - $[Ping_1, Ping_2, \ldots, Ping_N]$
- **Randomized_Vector:** An array of length N, populated with random values between [0,1]
- **Max_Noise:** The maximum value of noise (ms) to introduce.

Synthetic Data = **Window_of_Pings** + (**Randomized_Vector** × **Max_Noise**)

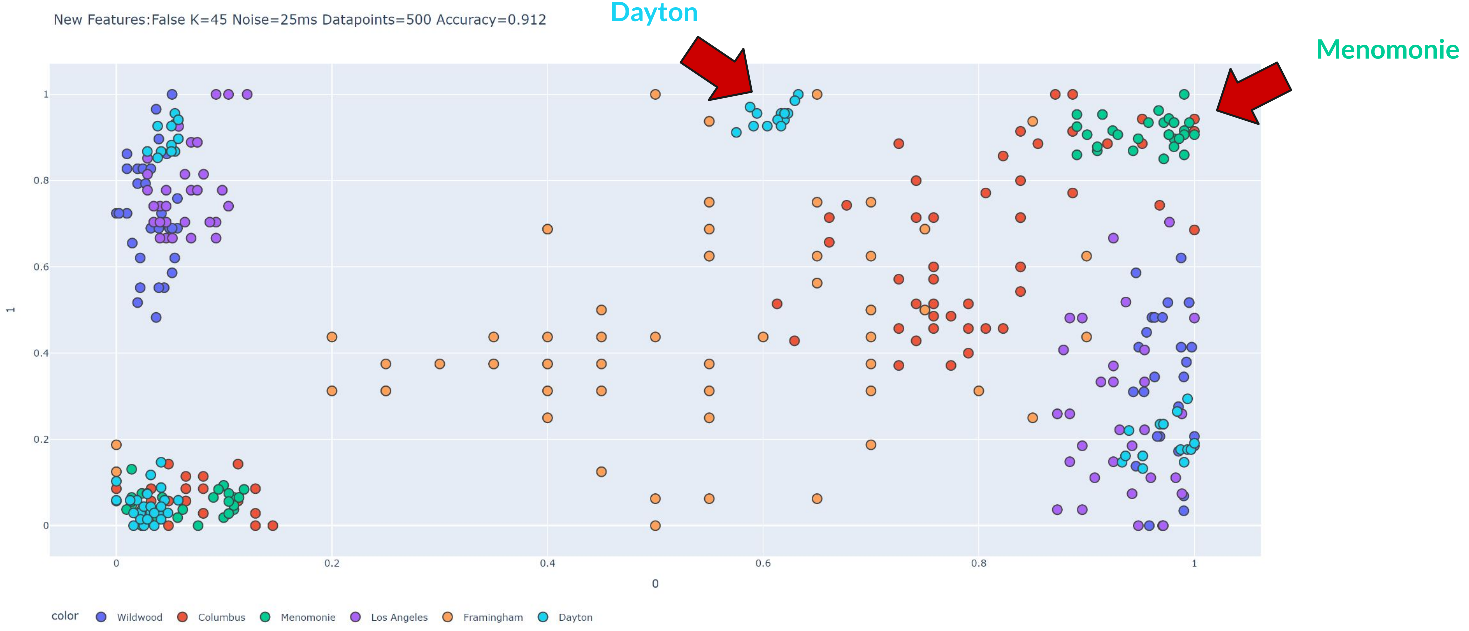
Original Data



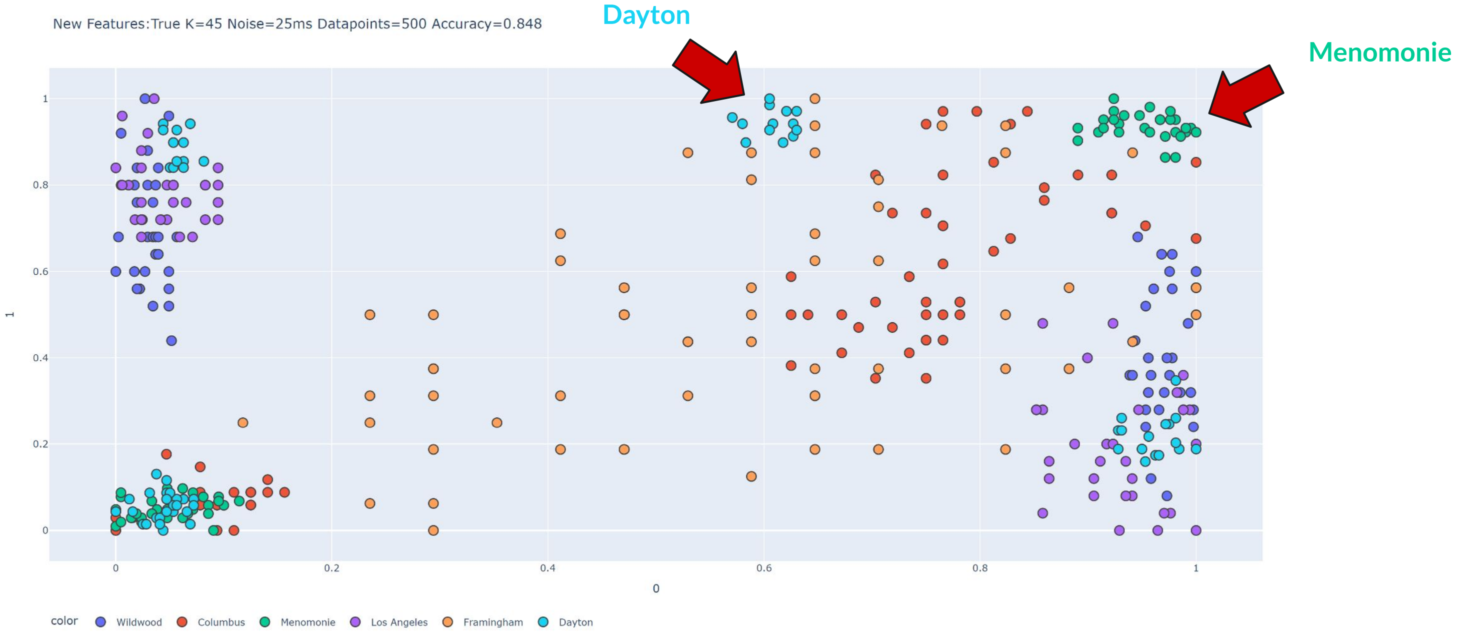
Synthetic Data: 0-50ms of Noise, Replicated x5



Results: KNN ML Model with Original Feature Extraction Method



Results: KNN ML Model with Custom Feature Extraction Method

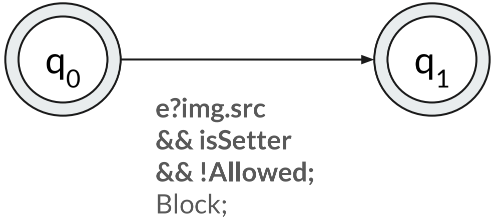


Experiments with Link-Based Fingerprinting

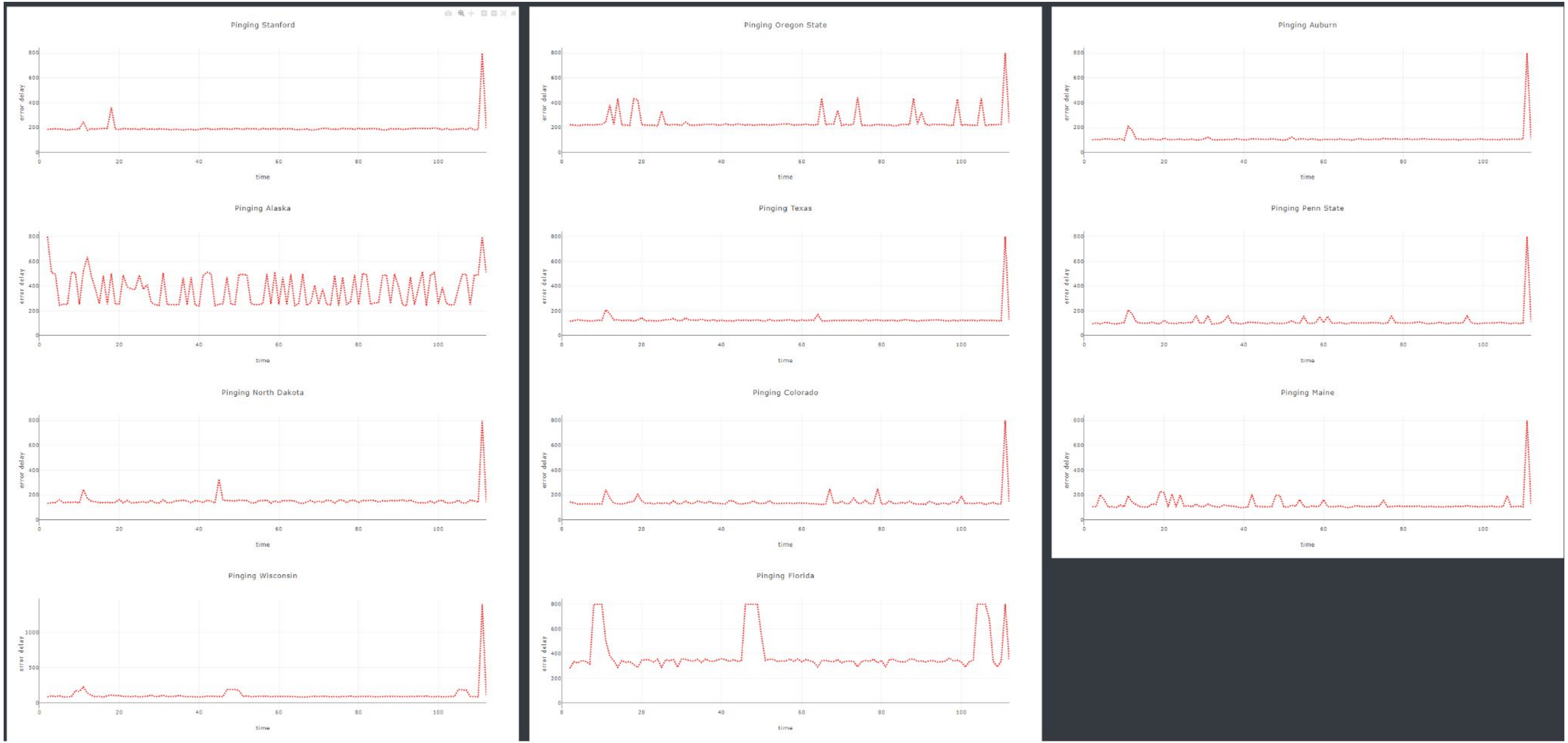
Experiments with Linked-Based Fingerprinting

<u>Link-Based Fingerprinting Policy:</u>

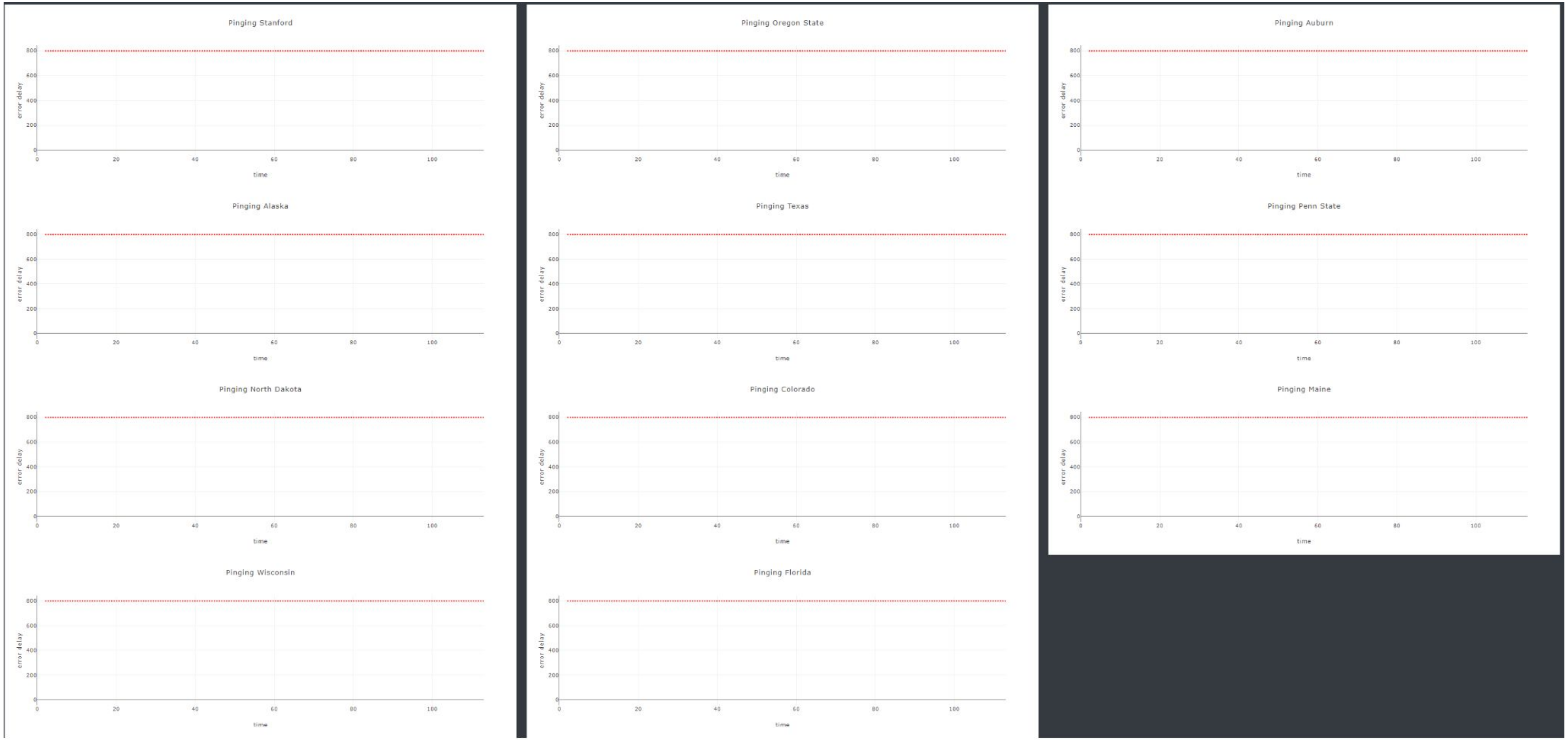
- **Goal**: Interaction Blocking.
- **Method**: Use allow/block lists to control the loading of images by code origin.
- Event Set: e = {...}
    - img.src



Results: Mitigation Policy (Before)



Results: Mitigation Policy (After)



Limitations of Link-Based Fingerprinting

- **Server Set:** University infrastructure may be less stable than industry.
  - Although we use university servers to avoid CDNs, industry leaders use CDNs to provide faster and more reliable content. As a result relying on this infrastructure may be more effective.
- **Scalability:** Due to the reliance on bursts of HTTP requests, link-based fingerprinting may cause network congestion at scale.
- **Uniqueness:** Link-based fingerprinters cannot uniquely identify users
  - Does not yield enough information to differentiate between users at the came location
  - But, remains a novel technique as it provides a new vector for obtaining user geolocation
- Cannot be used in challenge/response based authentication
  - **Other Applications:** Localizing Users, Session Hijacking Prevention

Recap

- What? Why? Who?
- Requirements
- Types
- Duality of Applications

- Mitigation Approaches
- Mitigation Examples
- MyWebGuard
- Mitigation Experiments

Overview of our related work at ISSec-Lab

- Using the Inlined Reference Monitors (IRM), a language-based security approach, to enforce policies or detect potential malicious behaviors to ensure security at runtime

JavaScript/Web Application Security
(AsiaCCS'09, AppSec'10, ACSAC'12, TDSC'15, FDSE19, SNCS'20)

Securing untrusted in-vehicle applications
(CompSAC'08, RTIC'08, RTIC'10)

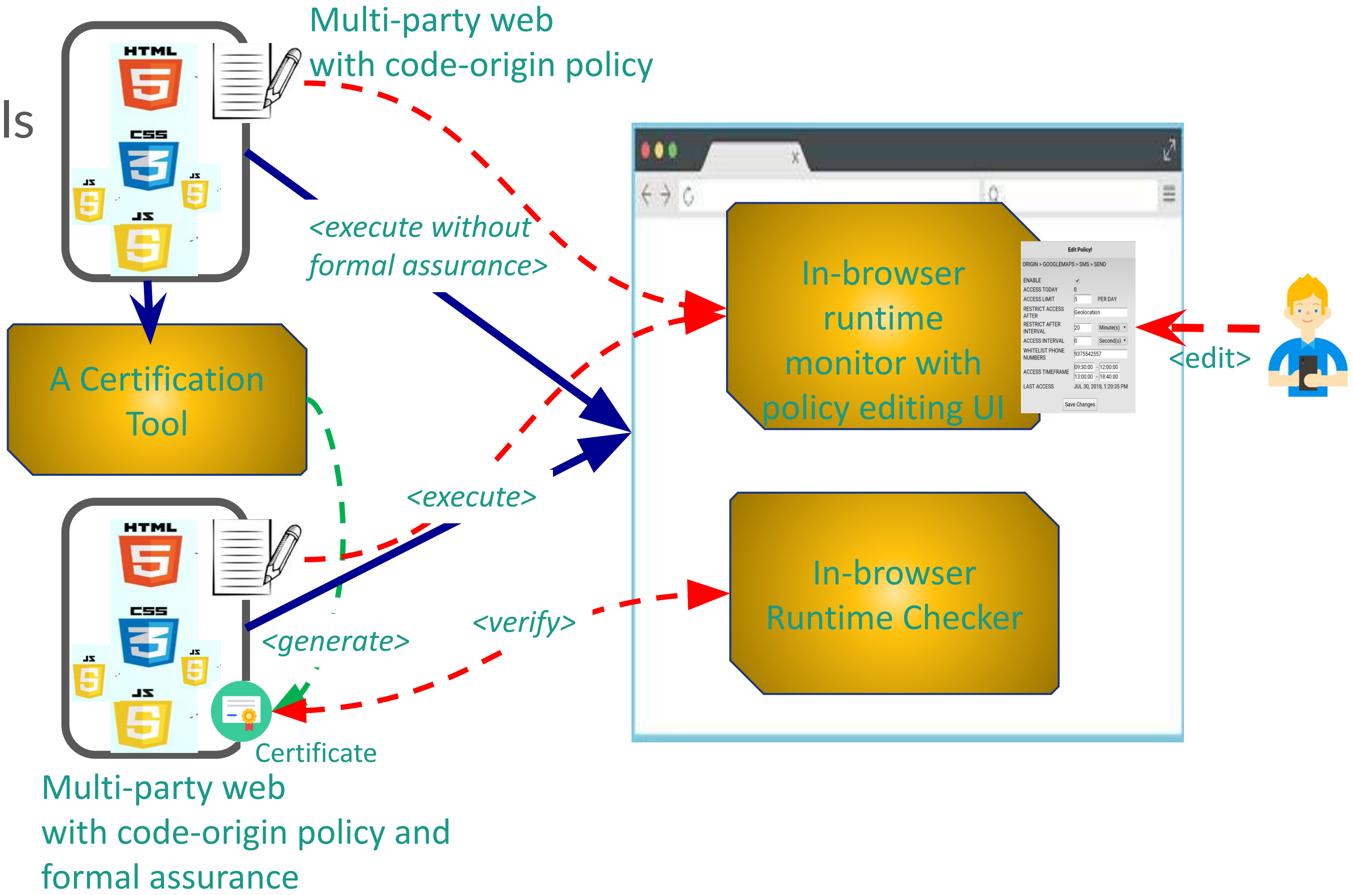
The IRM Approach

Hybrid/Web-based Mobile Apps
(C&S22, JCS'20, MOST'17)

The CPS/Internet of Things infrastructures
(SCC'22, IoT'18, ICIOT'17, MobWIS'12)

Code-Origin Policy with Formal Assurance Approach

- Explore existing formal tools
  - NuSMV
  - SPIN
  - JaVerT
  - Datalog



Multi-party web with code-origin policy

<execute without formal assurance>

A Certification Tool

In-browser runtime monitor with policy editing UI

Edit Policy!

ORIGIN > GOOGLEMAPS > SMS > SEND

ENABLE	☑	
ACCESS TODAY	0	
ACCESS LIMIT	5	PER DAY
RESTRICT ACCESS AFTER	Geolocation	
RESTRICT AFTER INTERVAL	20	Minute(s)
ACCESS INTERVAL	0	Second(s)
WHITELIST PHONE NUMBERS	9375542557	
ACCESS TIMEFRAME	09:30:00 - 12:00:00 13:00:00 - 18:40:00	
LAST ACCESS	JUL 30, 2018, 1:20:35 PM	

Save Changes

<edit>

<execute>

In-browser Runtime Checker

<generate>

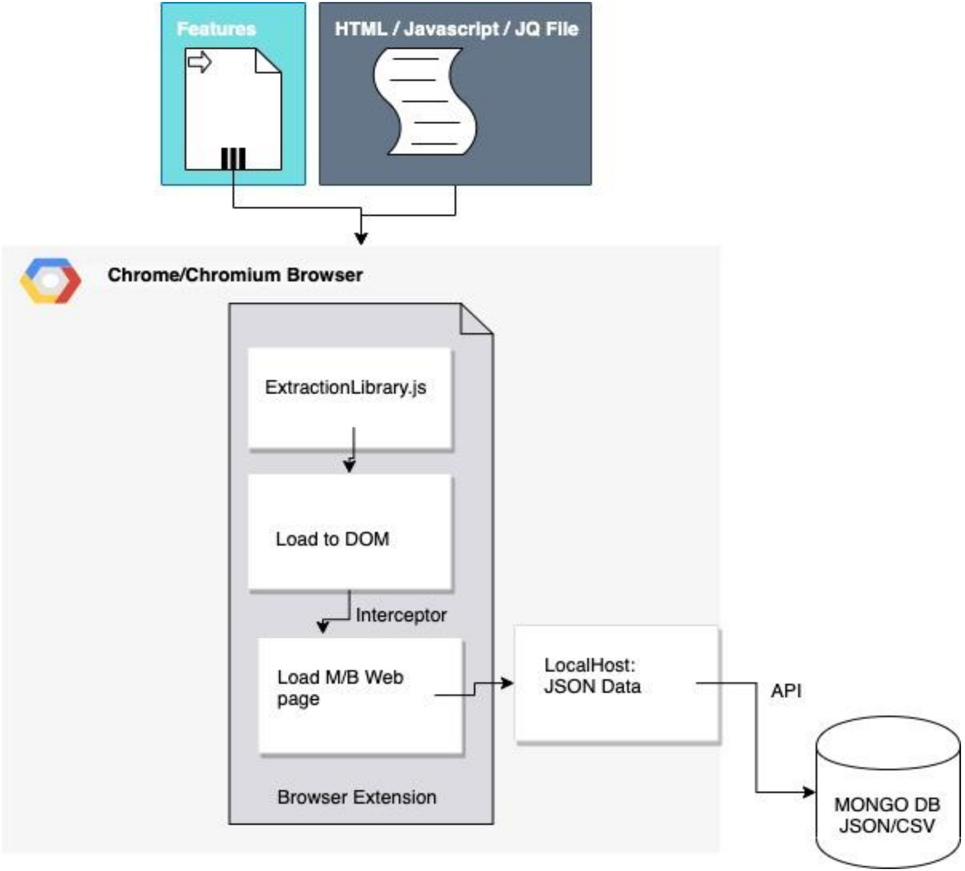
<verify>

Certificate

Multi-party web with code-origin policy and formal assurance

Dynamic analysis method for malicious JS

- Based on a runtime monitor, can be integrated within a browser
  - Currently implemented as a browser extension
    - ❑ *The browser extension is based on our previous work MyWebGuard*
  - Extracted runtime features will be used for machine learning models for maliciousness classification

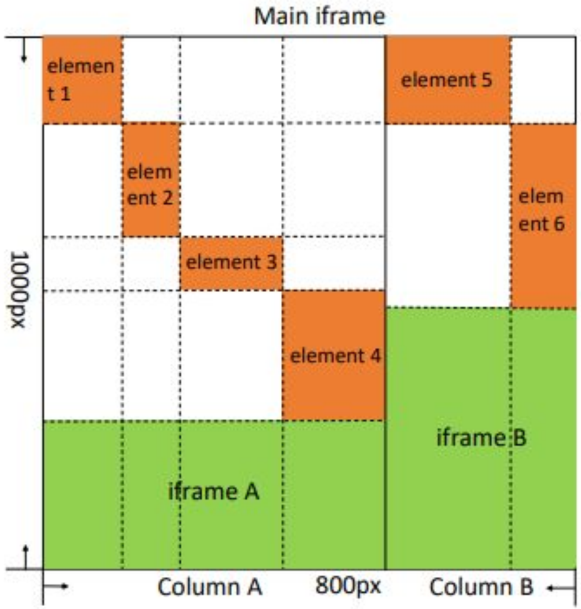


Questions?

Bonus Slides

CSS/HTML Based

- **Attack Vector:** CSS and HTML DOM Elements
- **Goal:** Identify the presence of unique plugins, extensions, or fonts.
- **Ex.** StylisticFP (2023) - IBM
  - Makes inferences about a browser's environment by creating iframes and HTML elements
- More likely to reveal sensitive information.



Source: Lin et al. (2023)

Sandboxing

- “Pay for leakage” policy
  - Also known as a *information budget* or *privacy budget*
- Each origin is isolated within its own environment
- Allows for unlimited client-side use of sensitive data, while limiting first-party or third-party external uses of same data

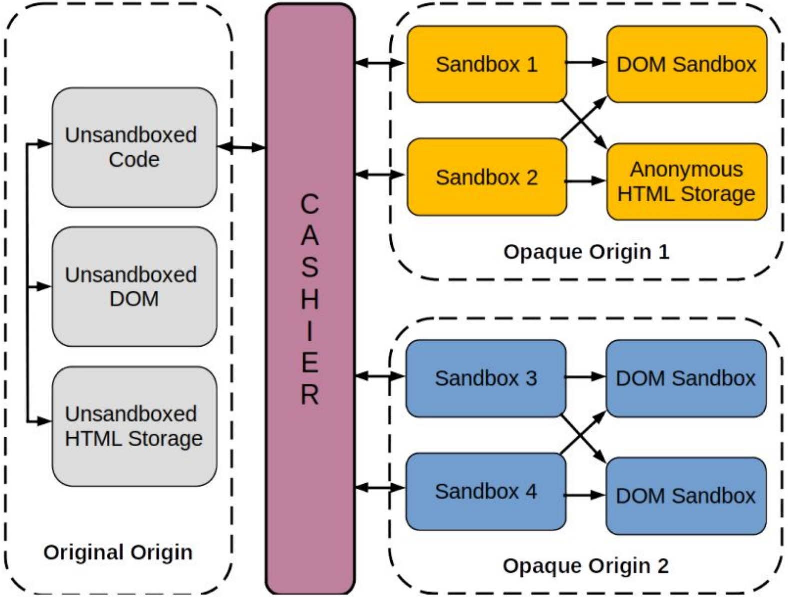


Figure 2: Partitioning the DOM using opaque origins and DOM sandboxes, which are based on anonymous iframes.

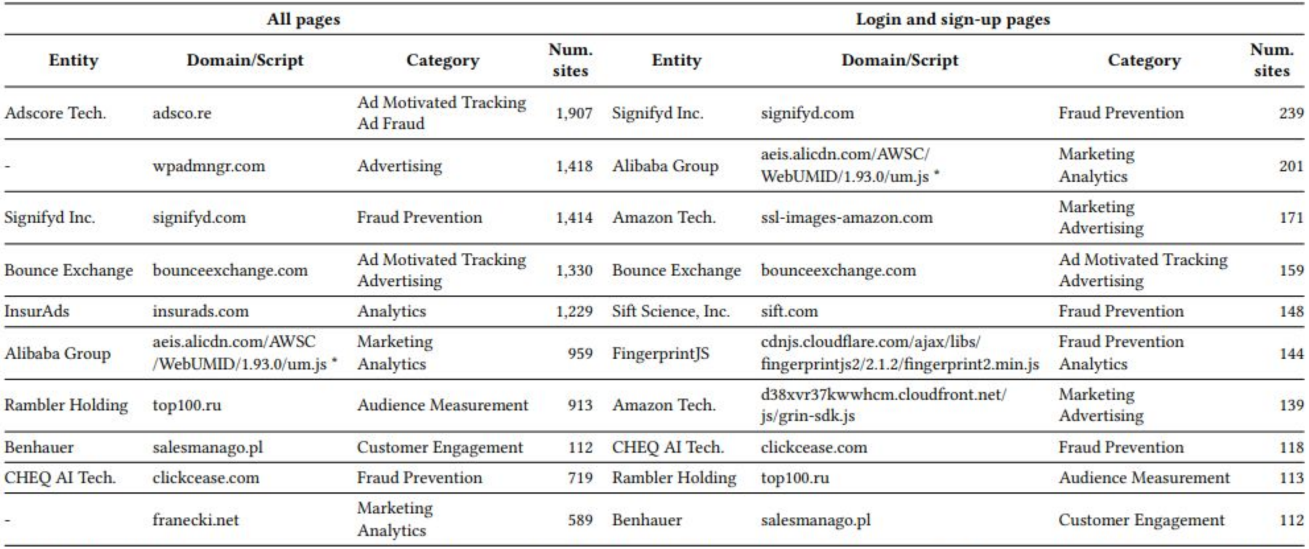
Source: Torok and Levy (2023)

Effectiveness of Fingerprinters

- ***Hiding in the Crowd*** (Gomex-Boix et al. 2018)
    - A large scale study for evaluating the effectiveness of different types of fingerprinters
    - 2 million fingerprints
    - Estimated *33.6% of fingerprints are unique*
    - Did not include all possible attributes, such as Timezone and Content Language

- ***DrawnApart*** (Laor, Mehanna et al. 2020)
    - Canvas specific study with a focus on exploiting manufacturing differences of GPUs
    - 2,500 devices, 370,000 fingerprints
    - Estimated *67% improvement* to trackability when adding canvas to other fingerprinting methods.

Top Fingerprinting Domains



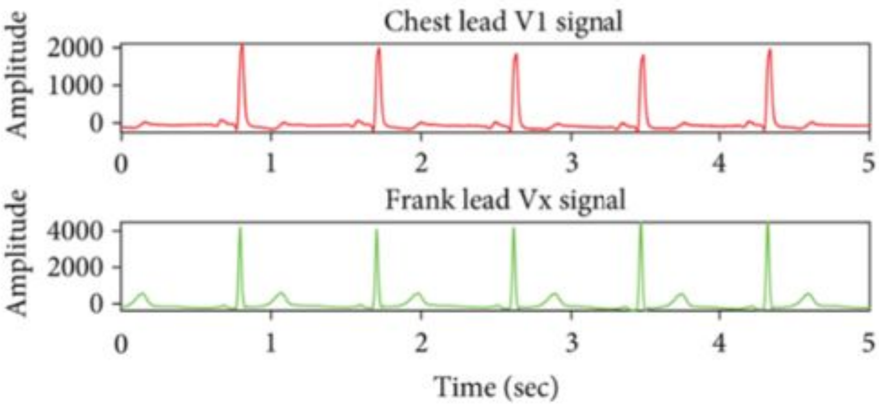
All pages				Login and sign-up pages			
Entity	**Domain/Script**	**Category**	**Num. sites**	**Entity**	**Domain/Script**	**Category**	**Num. sites**
Adscore Tech.	adsco.re	Ad Motivated Tracking Ad Fraud	1,907	Signifyd Inc.	signifyd.com	Fraud Prevention	239
-	wpadmngr.com	Advertising	1,418	Alibaba Group	aeis.alicdn.com/AWSC/ WebUMID/1.93.0/um.js *	Marketing Analytics	201
Signifyd Inc.	signifyd.com	Fraud Prevention	1,414	Amazon Tech.	ssl-images-amazon.com	Marketing Advertising	171
Bounce Exchange	bounceexchange.com	Ad Motivated Tracking Advertising	1,330	Bounce Exchange	bounceexchange.com	Ad Motivated Tracking Advertising	159
InsurAds	insurads.com	Analytics	1,229	Sift Science, Inc.	sift.com	Fraud Prevention	148
Alibaba Group	aeis.alicdn.com/AWSC /WebUMID/1.93.0/um.js *	Marketing Analytics	959	FingerprintJS	cdnjs.cloudflare.com/ajax/libs/ fingerprintjs2/2.1.2/fingerprint2.min.js	Fraud Prevention Analytics	144
Rambler Holding	top100.ru	Audience Measurement	913	Amazon Tech.	d38xvr37kwwhcm.cloudfront.net/ js/grin-sdk.js	Marketing Advertising	139
Benhauer	salesmanago.pl	Customer Engagement	112	CHEQ AI Tech.	clickcease.com	Fraud Prevention	118
CHEQ AI Tech.	clickcease.com	Fraud Prevention	719	Rambler Holding	top100.ru	Audience Measurement	113
-	franecki.net	Marketing Analytics	589	Benhauer	salesmanago.pl	Customer Engagement	112

Source: Senol, Ukani et al. (2024)

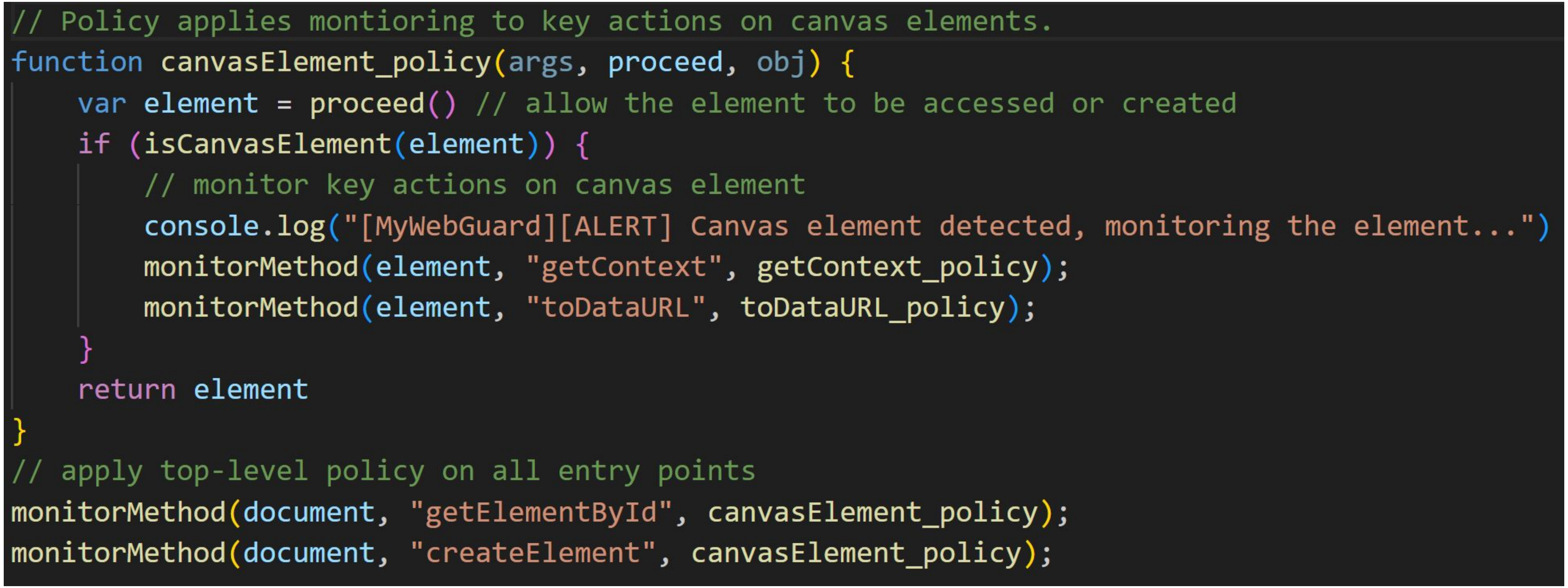
Inspiration for Proposed Features

ECG-Based Subject Identification**:** Turky N Alotaiby et al. (2019)

1. Interquartile Range
2. Interquartile First Quarter (Q1)
3. Interquartile Third quarter (Q3)

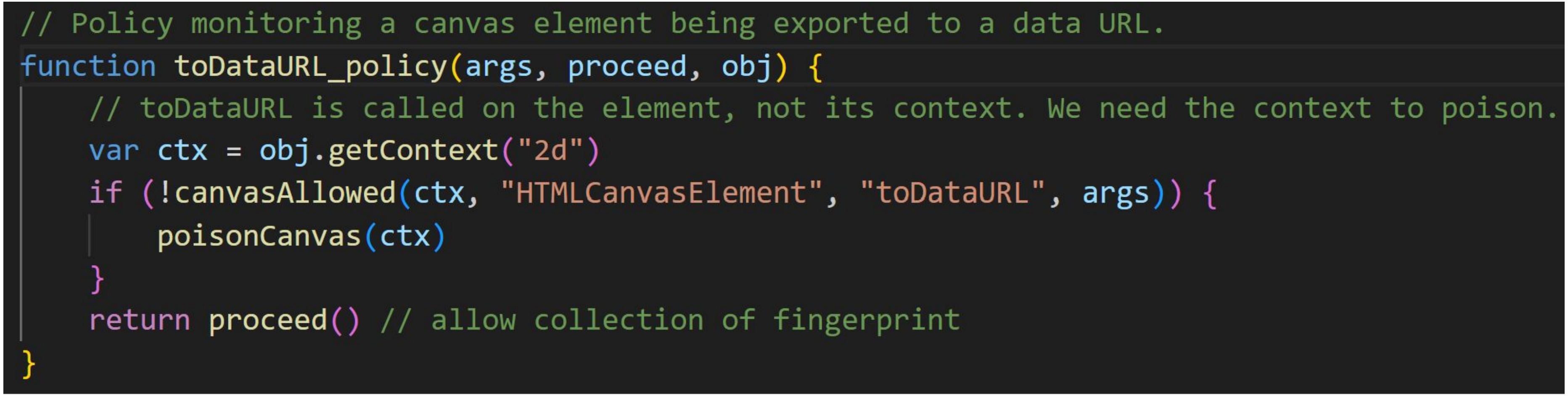


Implementation: Top-Level Canvas Monitoring



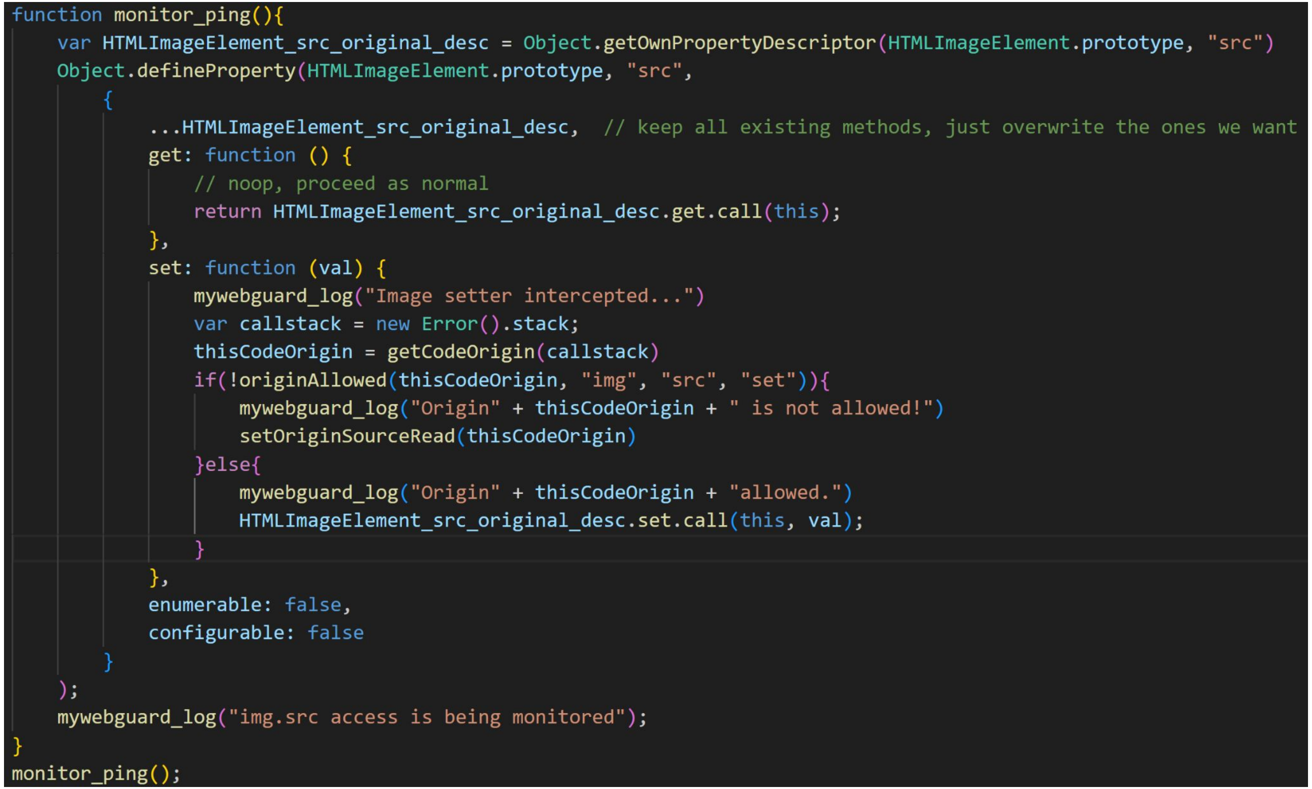
```
// Policy applies montioring to key actions on canvas elements.
function canvasElement_policy(args, proceed, obj) {
    var element = proceed() // allow the element to be accessed or created
    if (isCanvasElement(element)) {
        // monitor key actions on canvas element
        console.log("[MyWebGuard][ALERT] Canvas element detected, monitoring the element...")
        monitorMethod(element, "getContext", getContext_policy);
        monitorMethod(element, "toDataURL", toDataURL_policy);
    }
    return element
}
// apply top-level policy on all entry points
monitorMethod(document, "getElementById", canvasElement_policy);
monitorMethod(document, "createElement", canvasElement_policy);
```

Implementation: toDataURL Policy Enforcement



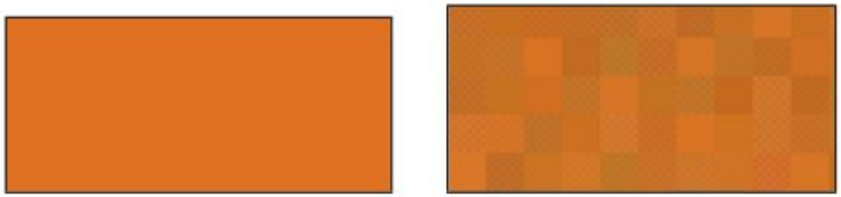
```javascript
// Policy monitoring a canvas element being exported to a data URL.
function toDataURL_policy(args, proceed, obj) {
    // toDataURL is called on the element, not its context. We need the context to poison.
    var ctx = obj.getContext("2d")
    if (!canvasAllowed(ctx, "HTMLCanvasElement", "toDataURL", args)) {
        poisonCanvas(ctx)
    }
    return proceed() // allow collection of fingerprint
}
```

Implementation: Ping Policy Enforcement



```
function monitor_ping(){
    var HTMLImageElement_src_original_desc = Object.getOwnPropertyDescriptor(HTMLImageElement.prototype, "src")
    Object.defineProperty(HTMLImageElement.prototype, "src",
        {
            ...HTMLImageElement_src_original_desc,  // keep all existing methods, just overwrite the ones we want
            get: function () {
                // noop, proceed as normal
                return HTMLImageElement_src_original_desc.get.call(this);
            },
            set: function (val) {
                mywebguard_log("Image setter intercepted...")
                var callstack = new Error().stack;
                thisCodeOrigin = getCodeOrigin(callstack)
                if(!originAllowed(thisCodeOrigin, "img", "src", "set")){
                    mywebguard_log("Origin" + thisCodeOrigin + " is not allowed!")
                    setOriginSourceRead(thisCodeOrigin)
                }else{
                    mywebguard_log("Origin" + thisCodeOrigin + "allowed.")
                    HTMLImageElement_src_original_desc.set.call(this, val);
                }
            },
            enumerable: false,
            configurable: false
        }
    );
    mywebguard_log("img.src access is being monitored");
}
monitor_ping();
```

Other Canvas Poisoner Example



(a) *Without a poisoner* (b) *With a poisoner*

Source: Laperdrix et al. (2019)

Other

- **Simulacrum** (2022): “Dom reality shifting”
  - A solid defense against HTML/CSS based fingerprinting
  - Protects against extension based fingerprinting

Links

- WebApp
- Github
- Wu et al. (2021)
- Phung et al. (2020)
- AmIUnique
- BroswerLeaks