Breaking Fraud & Bot Detection Solutions

Mayank Dhiman
Stealth Security
Agenda

• Architectural Overview
• Threat Model
• Issues & Attacks
• Takeaways
Fraud Detection

• Defend against fraudulent logins, payments etc.

• Look for anomalies in activity of a user, given past activity.
Bot Detection

• Defend against bots trying to test credential dumps, scraping etc.

• Bot detection solutions look for anomalies across entire populations and time periods.
Cloud Deployment

1. Web Request
2. Fingerprint.js
3. Service Provider
4. Form Submission
5. Risk Score
6. Block
6. Allow
Inline Deployments

Client Browser

1. Web Request
2. Fingerprint.js
3. Fingerprint

Inline Device

4. Allow
4. Block

Web Server
Threat Model

- Attacker has full control over the browser.
- Attacker can craft requests and modify responses according to the responses from the web server.
Fundamental Issue 1

- Attacker can reverse engineer the entire sensor
# Browser Fingerprinting

```markdown
## Browser Fingerprinting

<table>
<thead>
<tr>
<th>Browser Characteristic</th>
<th>bits of identifying information</th>
<th>one in x browsers have this value</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Limited supercookie test</td>
<td>0.39</td>
<td>1.31</td>
<td>DOM localStorage: Yes, DOM sessionStorage: Yes, IE userData: No</td>
</tr>
<tr>
<td>Hash of canvas fingerprint</td>
<td>14.2</td>
<td>18792.03</td>
<td>d68a8387d4f25241664deb501fe16955</td>
</tr>
<tr>
<td>Screen Size and Color Depth</td>
<td>4.29</td>
<td>19.6</td>
<td>1440x900x24</td>
</tr>
<tr>
<td>Browser Plugin Details</td>
<td>9.25</td>
<td>610.61</td>
<td>Plugin 0: Chromium PDF Viewer; ; mh/zbmdgcfjbbpaojofohoejgiehja; (; applicatio n/pdf; pdf). Plugin 1: Chromium PDF Viewer; Portable Document Format; internal-p df-viewer; (Portable Document Format; application/x-google-chrome-pdf; pdf). Plugin 2: Native Client; ; internal-nacl-plugin; (Native Client Executable; application/x-nac l; ) (Portale Native Client Executable; application/x-pnacl; ).</td>
</tr>
<tr>
<td>Time Zone</td>
<td>5.66</td>
<td>50.46</td>
<td>480</td>
</tr>
<tr>
<td>DNT Header Enabled?</td>
<td>0.8</td>
<td>1.74</td>
<td>True</td>
</tr>
<tr>
<td>HTTP_ACCEPT Headers</td>
<td>7.22</td>
<td>148.8</td>
<td>text/html, <em>/</em>; q=0.01 gzip, deflate en-US,en;q=0.8</td>
</tr>
<tr>
<td>Hash of WebGL fingerprint</td>
<td>18.06</td>
<td>272484.5</td>
<td>e3cea23259d3808d0672d33457a79450</td>
</tr>
<tr>
<td>Language</td>
<td>0.91</td>
<td>1.88</td>
<td>en-US</td>
</tr>
<tr>
<td>Platform</td>
<td>3.06</td>
<td>8.33</td>
<td>MacIntecl</td>
</tr>
<tr>
<td>User Agent</td>
<td>20.06</td>
<td>1089938.0</td>
<td>Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.98 Safari/537.36</td>
</tr>
<tr>
<td>Touch Support</td>
<td>0.58</td>
<td>1.49</td>
<td>Max touchpoints: 0; TouchEvent supported: false; onTouchStart supported: false</td>
</tr>
<tr>
<td>Are Cookies Enabled?</td>
<td>0.2</td>
<td>1.15</td>
<td>Yes</td>
</tr>
</tbody>
</table>
```

https://panopticlick.eff.org/

---

**OWASP**

Open Web Application Security Project

[OWASP Logo](https://www.owasp.org)
Browser Fingerprinting

- **Hardware**
  - CPU Architecture & Device Memory
  - GPU Canvas Fingerprinting
  - Audio Stack Fingerprinting

- **Software**
  - UserAgent
  - OS Version

- **Storage**
  -LocalStorage
  -SessionStorage

- **Display**
  - Color Depth
  - Screen Size

- **Browser Customizations**
  - Fonts
  - Plugins
  - Codecs
  - Mime Types
  - Time zone
  - User Language

- **Misc.**
  - Floating point calculations
  - Add behavior/callbacks/objects to DOM to check a real JS execution engine

---

OWASP
Open Web Application Security Project

www.owasp.org
Browser Fingerprinting (Fingerprintjs2)

Your browser fingerprint: ecf9e4942d57d4067112531d4f97e4fc

Detailed information:
user_agent = Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2
language = en-US
color_depth = 24
device_memory = -1
pixel_ratio = 1
hardware_concurrency = 8
resolution = 1920,1080
available_resolution = 1920,1076
timezone_offset = 480
session_storage = 1
local_storage = 1
indexed_db = 1
open_database = 1
cpu_class = unknown
navigator_platform = MacIntel
do_not_track = unknown
regular_plugins = Chromium PDF Viewer:::application/pdf~pdf,Native Client:::application/x-nacl~,application/x-pnacl~
canvas = canvas winding;yes~canvas fp:datimage/png;base64,iVBORw0KgoAAAAASUlEQAAgA9AAAAAICAYAAdCwGnoBAAAgA
webgl = data:image/png;base64,iVBORw0KgoAAAAASUlEQAAgA9AAAAAICAYAAdCwGnoBAAAgA
webgl_vendor = ATI Technologies Inc.-AMD Radeon R9 M370X OpenGL Engine
adblock = false
has_lited_languages = true
has_lited_resolution = false
has_lited_os = false
has_lited_browser = false
touch_support = 0,true,false
js_fonts = Andale Mono,Arial,Arial Black,Arial Hebrew,Arial Narrow,Arial Rounded MT Bold,Arial Unicode MS,Comic

https://github.com/Valve/fingerprintjs2
User Behavior

• Mouse
  – Coordinates of where the move moved
  – Coordinates of clicks

• Keyboard
  – Stream of key presses

• Touchpad
  – Coordinates of where the screen was touched
User Behavior

• Device Orientation
  – 3D angle of device whenever the orientation changes

• Device Position
  – Record speed of change of device’s position.

Timing information along with event type can be used to create a very accurate picture of what interactions took place on the webpage.
Anti-Tampering & Anti-Reversing

• JavaScript Obfuscation
• XOR based packed code
• Randomize name/location of the JavaScript file to load
• Dynamic Fields
Payload

- Payload Encoding (Base64)
- Symmetric Encryption (DES)
- Custom Encryption Schemes
Fundamental Issue II

• There are no guarantees of the correct execution of JavaScript
Headless Browsers

• Browser without a GUI, often used for automation and testing.
• Either render full JS or run JS in a virtual DOM.
Stripping Attack

1. Web Request
2. Fingerprint.js
3. Service Provider
4. Form Submission
5. Risk Score
6. Allow
7. Block

Client Browser

Mitigator

Web Server
Stripping Attack

1. Web Request
2. Fingerprint.js
3. Service Provider
4. Form Submission
5. Risk Score
6. Allow
7. Block

Client Browser -> Web Request -> Fingerprint.js -> Service Provider

Service Provider

Web Server

OWASP
Open Web Application Security Project
WWW.O Waksp.org
Stripping Attack

1. Web Request
2. Fingerprint.js
3. Fingerprints
4. Allow
4. Block

Client Browser

Inline Device

Web Server

OWASP
Open Web Application Security Project

www.owasp.org
Stripping Attack

1. Web Request
2. Fingerprint.js
3. Form POST
4. Block

Client Browser → MITM Proxy → Inline Device → Web Server

OWASP
Open Web Application Security Project
WWW.OVASP.ORG
Replay Attacks

- No check on freshness of payload.
Dynamic Tokens

• A dynamic token is generated, which is derived from the timestamp.
• Same logic can be replicated in a script.
Fundamental Issue III

• There are no guarantees of the legitimacy of the data collected by the JavaScript sensors.
Forging Browser Fingerprints

• FPRANDOM – Modified browser which introduces noise during browser fingerprint.
• OpenWPM – Web Privacy Measurement software.
• Database of Normal Fingerprints

https://github.com/plaperdr/fprandom
https://github.com/citp/OpenWPM
// original source: https://github.com/Valve/fingerprintsjs2

function getCanvasFp() {
  var result = ''
  var canvas = document.createElement('canvas')
  var ctx = canvas.getContext('2d')
  ctx.textBaseline = 'alphabetic'
  ctx.fillStyle = '#f60'
  ctx.fillRect(121, 1, 62, 20)
  ctx.fillStyle = '#00f9'
  ctx.fillRect(11pt Arial'
  ctx.fillStyle = 'rgba(102, 204, 0, 0.2)'
  ctx.font = '18pt Arial'
  ctx.fillText('Cwm fjordbank glyphs vext quiz, \ud83d\ude03', 4, 45)
  result = 'Canvas Fingerprint is: ' + canvas.toDataURL()
  return result
}

undefined

getCanvasFp()
Forging Browser Fingerprints

```javascript
// original source: https://github.com/Valve/fingerprintjs2

function getCanvasFp() {
    var result = ""
    var canvas = document.createElement('canvas')
    var ctx = canvas.getContext('2d')
    ctx.textBaseline = 'alphabetic'
    ctx.fillStyle = '#f60'
    ctx.fillRect(125, 1, 62, 20)
    ctx.fillStyle = '#069'
    ctx.font = '11pt Arial'
    ctx.fillText('Cwm fjordbank glyphs vext quiz, \ud83d\ude03', 2, 15)
    ctx.fillStyle = 'rgba(102, 264, 0, 0.2)'
    ctx.font = '18pt Arial'
    ctx.fillText('Cwm fjordbank glyphs vext quiz, \ud83d\ude03', 4, 45)

    result = 'Canvas Fingerprint is: ' + canvas.toDataURL()
    return result
}

undefined

getCanvasFp()

"Canvas Fingerprint is: FakeCanvasFingerprint!!!" = $1
```
Bad Guys Are Already Doing this

- Anti-Detect* $399 in the underground

https://krebsonsecurity.com/2015/03/antidetect-helps-thieves-hide-digital-fingerprints/
User Behavior

- Replay with changed timestamps
- Add ripples and disturbances
- Use MITM Proxy
Fundamental Issue IV

• JavaScript can’t protect all flows.
Fundamental Issue V

• The mitigative action acts as an oracle for the attacker.
Other Issues

• Fraud/Bot Detection Solutions are themselves Fingerprintable.

• Similar issues exist for mobile app SDK based solutions.
Takeaways

- Implementation and Architectural Issues in multiple deployments.
- JavaScript runs in an attacker controlled environment.
- Understand the limitations of such solutions.
- Protect all flows.
Questions?