JavaScript Template Attacks

Michael Schwarz, Florian Lackner, Daniel Gruss
February 25, 2019

IAIK – Graz University of Technology
• Many (undocumented) properties in JavaScript sandboxes
• Many (undocumented) properties in JavaScript sandboxes
• Properties should not leak environment info
Motivation

- Many (undocumented) properties in JavaScript sandboxes
- Properties should not leak environment info
- Information useful for exploits and side-channel attacks
Motivation

- Many (undocumented) properties in JavaScript sandboxes
- Properties should not leak environment info
- Information useful for exploits and side-channel attacks
- Also usable for fingerprinting
• Theory: JavaScript sandbox is environment agnostic
• Theory: JavaScript sandbox is environment agnostic
• Code gives same results independent of platforms or hardware
Identifying Properties

- Theory: JavaScript sandbox is environment agnostic
- Code gives same results **independent of platforms or hardware**
- `window.Array.name` is always “Array”
• Theory: JavaScript sandbox is environment agnostic
• Code gives same results independent of platforms or hardware
• `window.Array.name` is always “Array”
• Some defined exceptions, e.g., user agent
• Theory: JavaScript sandbox is environment agnostic
• Code gives same results independent of platforms or hardware
• `window.Array.name` is always “Array”
• Some defined exceptions, e.g., user agent
• Tor browser → identifying properties anonymized
Properties leaking info about hardware or software...
• Properties leaking info about hardware or software...
  • ...can be used to track users (→ fingerprinting)
• Properties leaking info about hardware or software...
  • ...can be used to track users (→ fingerprinting)
  • ...make phishing more plausible
• Properties leaking info about hardware or software...
  • ...can be used to track users (→ fingerprinting)
  • ...make phishing more plausible
  • ...allow selecting fitting exploits
• Properties leaking info about hardware or software...
  • ...can be used to track users (→ fingerprinting)
  • ...make phishing more plausible
  • ...allow selecting fitting exploits
  • ...provide necessary information for side-channel attacks
Information Leakage

- Properties leaking info about hardware or software...
  - ...can be used to track users (→ fingerprinting)
  - ...make phishing more plausible
  - ...allow selecting fitting exploits
  - ...provide necessary information for side-channel attacks

→ indirect security risk
Automated Leakage Identification

- Manually finding leakage $\rightarrow$ time consuming
Automated Leakage Identification

- Manually finding leakage $\rightarrow$ time consuming
- Automate the task
Automated Leakage Identification

- Manually finding leakage → time consuming
- Automate the task
- Idea of template attacks: change a factor, compare results
JavaScript Template Attacks - Overview

Profile Phase

Explore

Profiling Phase
JavaScript Template Attacks - Overview

Profiling Phase

Collect #1

Explore

Template

Michael Schwarz (@misc0110) et al. — IAIK – Graz University of Technology
JavaScript Template Attacks - Overview

Collect #1

Collect #2

Template

Explore

Profiling Phase

Michael Schwarz (@misc0110) et al. — IAIK – Graz University of Technology
JavaScript Template Attacks - Overview

Profiling Phase

Explore

Collect #1

Collect #2

Collect #n

Template

Michael Schwarz (@misc0110) et al. — IAIK – Graz University of Technology
JavaScript Template Attacks - Overview

Profiling Phase
- Explore
- Collect #1
- Collect #2
- Collect #n

Analysis Phase
- Template
- Cleanup

Michael Schwarz (@misc0110) et al. — IAIK – Graz University of Technology
JavaScript Template Attacks - Overview

- **Profiling Phase**
  - Explore
  - Collect #1
  - Collect #2
  - Collect #n

- **Analysis Phase**
  - Template
  - Cleanup
  - Extraction
JavaScript Template Attacks - Overview

Profiling Phase
- Explore
- Collect #1
- Collect #2
- Collect #n

Analysis Phase
- Template
- Cleanup
- Extraction
- Properties
JavaScript Template Attacks - Overview

Profiling Phase:
- Collect #1
- Collect #2
- Collect #n

Analysis Phase:
- Cleanup
- Extraction
- Properties

Michael Schwarz (@misc0110) et al. — IAIK – Graz University of Technology
Profiling Phase

- Exploration of properties
Profiling Phase

- Exploration of properties
  - Reflections to iterate over all objects
Profiling Phase

- Exploration of properties
  - Reflections to iterate over all objects
  - Recursively, until all objects are discovered
Profiling Phase

- Exploration of properties
  - Reflections to iterate over all objects
  - Recursively, until all objects are discovered
- Collection of property values
Profiling Phase

- Exploration of properties
  - Reflections to iterate over all objects
  - Recursively, until all objects are discovered

- Collection of property values
  - For every discovered property, acquire value
Profiling Phase

- Exploration of properties
  - Reflections to iterate over all objects
  - Recursively, until all objects are discovered

- Collection of property values
  - For every discovered property, acquire value
  - Repeat with changing environments
Template

- Template is a table, rows are properties, columns are environments
- Template is a table, rows are properties, columns are environments

<table>
<thead>
<tr>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>window.Array.name</code></td>
</tr>
<tr>
<td><code>window.window.Array.name</code></td>
</tr>
<tr>
<td><code>navigator.platform</code></td>
</tr>
<tr>
<td><code>performance.timeOrigin</code></td>
</tr>
<tr>
<td><code>window.SharedWorker</code></td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
- Template is a table, rows are properties, columns are environments

<table>
<thead>
<tr>
<th>Property</th>
<th>Environment 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>window.Array.name</td>
<td>Array</td>
</tr>
<tr>
<td>window.window.Array.name</td>
<td>Array</td>
</tr>
<tr>
<td>navigator.platform</td>
<td>Linux x86_64</td>
</tr>
<tr>
<td>performance.timeOrigin</td>
<td>1551003902225</td>
</tr>
<tr>
<td>window.SharedWorker</td>
<td>function SharedWorker()</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
- Template is a **table**, rows are **properties**, columns are **environments**

<table>
<thead>
<tr>
<th>Property</th>
<th>Environment 1</th>
<th>Environment 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>window.Array.name</code></td>
<td>Array</td>
<td>Array</td>
</tr>
<tr>
<td><code>window.window.Array.name</code></td>
<td>Array</td>
<td>Array</td>
</tr>
<tr>
<td><code>navigator.platform</code></td>
<td>Linux x86_64</td>
<td>Linux armv7l</td>
</tr>
<tr>
<td><code>performance.timeOrigin</code></td>
<td>1551003902225</td>
<td>1551003815955</td>
</tr>
<tr>
<td><code>window.SharedWorker</code></td>
<td>function SharedWorker()</td>
<td>null</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
- Template is a **table**, rows are **properties**, columns are **environments**

<table>
<thead>
<tr>
<th>Property</th>
<th>Environment 1</th>
<th>Environment 2</th>
<th>Environment 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>window.Array.name</td>
<td>Array</td>
<td>Array</td>
<td>Array</td>
</tr>
<tr>
<td>window.window.Array.name</td>
<td>Array</td>
<td>Array</td>
<td>Array</td>
</tr>
<tr>
<td>navigator.platform</td>
<td>Linux x86_64</td>
<td>Linux armv7l</td>
<td>Win32</td>
</tr>
<tr>
<td>performance.timeOrigin</td>
<td>1551003902225</td>
<td>1551003815955</td>
<td>1551003721632</td>
</tr>
<tr>
<td>window.SharedWorker</td>
<td>function SharedWorker()</td>
<td>null</td>
<td>function SharedWorker()</td>
</tr>
</tbody>
</table>
- Cleanup template (remove duplicates and non-static values)
• Cleanup template (remove duplicates and non-static values)

<table>
<thead>
<tr>
<th>Property</th>
<th>Environment 1</th>
<th>Environment 2</th>
<th>Environment 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>window.Array.name</td>
<td>Array</td>
<td>Array</td>
<td>Array</td>
</tr>
<tr>
<td>window.window.Array.name</td>
<td>Array</td>
<td>Array</td>
<td>Array</td>
</tr>
<tr>
<td>navigator.platform</td>
<td>Linux x86_64</td>
<td>Linux armv71</td>
<td>Win32</td>
</tr>
<tr>
<td>performance.timeOrigin</td>
<td>1551003902225</td>
<td>1551003815955</td>
<td>1551003721632</td>
</tr>
<tr>
<td>window.SharedWorker</td>
<td>function SharedWorker()</td>
<td>null</td>
<td>function SharedWorker()</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
### Properties

<table>
<thead>
<tr>
<th>Browser</th>
<th>MDN</th>
<th>JavaScript Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firefox</td>
<td>2247</td>
<td>15 709</td>
</tr>
<tr>
<td>Chrome</td>
<td>2698</td>
<td>13 570</td>
</tr>
<tr>
<td>Edge</td>
<td>1806</td>
<td>9 666</td>
</tr>
<tr>
<td>Firefox Android</td>
<td>2104</td>
<td>15 612</td>
</tr>
<tr>
<td>Chrome Android</td>
<td>2676</td>
<td>13 119</td>
</tr>
<tr>
<td>Tor browser</td>
<td>2247†</td>
<td>15 639</td>
</tr>
</tbody>
</table>
• Analyse template (**remove values which are the same for all environments**)
- Analyse template (remove values which are the same for all environments)

<table>
<thead>
<tr>
<th>Property</th>
<th>Environment 1</th>
<th>Environment 2</th>
<th>Environment 3</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>window.Array.name</code></td>
<td>Array</td>
<td>Array</td>
<td>Array</td>
</tr>
<tr>
<td><code>window.window.Array.name</code></td>
<td>Array</td>
<td>Array</td>
<td>Array</td>
</tr>
<tr>
<td><code>navigator.platform</code></td>
<td>Linux x86_64</td>
<td>Linux armv7l</td>
<td>Win32</td>
</tr>
<tr>
<td><code>performance.timeOrigin</code></td>
<td>1551003902225</td>
<td>1551003815955</td>
<td>1551003721632</td>
</tr>
<tr>
<td><code>window.SharedWorker</code></td>
<td>function SharedWorker()</td>
<td>null</td>
<td>function SharedWorker()</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
• JavaScript allows **defining properties at runtime**
Artifical Properties

- JavaScript allows defining properties at runtime
- Add “artifical” properties before the profiling phase
Artificial Properties

- JavaScript allows defining properties at runtime
- Add "artifical" properties before the profiling phase
- Artificial properties are properties containing results of functions
Artificial Properties

- JavaScript allows defining properties at runtime
- Add “artificial” properties before the profiling phase
- Artificial properties are properties containing results of functions
- → Gather even more information about the environment
We show 2 new side channels against the JIT compiler.
JIT Artificial Properties

- We show 2 new side channels against the JIT compiler
- Detect internal memory allocator block size
• We show 2 new side channels against the JIT compiler
• Detect internal memory allocator block size
\[ \to \] Timing differences when re-allocating memory
• We show 2 new side channels against the JIT compiler
• Detect internal memory allocator block size
→ Timing differences when re-allocating memory
• Distinguish 32 bit from 64 bit systems
We show 2 new side channels against the JIT compiler
Detect internal memory allocator block size
→ Timing differences when re-allocating memory
Distinguish 32 bit from 64 bit systems
→ JIT can use more registers on 64-bit systems
var a = 0.9, b = c = d = e = f = g = 0;

for (var i = 0; i < 10000000; i++) {
    b = 1.0 / a;
    c = 2.2 / b;
    d = 3.4 / c;
    e = 4.1 / d;
    f = 5.8 / e;
    g = 6.6 / f;
    // no operation
    a = a + b + c + d + e + f + g + g;
}

var a = 0.9, b = c = d = e = f = g = h = 0;

for (var i = 0; i < 10000000; i++) {
    b = 1.0 / a;
    c = 2.2 / b;
    d = 3.4 / c;
    e = 4.1 / d;
    f = 5.8 / e;
    g = 6.6 / f;
    h = 7.1 / g;
    a = a + b + c + d + e + f + g + h;
}
vaddss %xmm0,%xmm1,%xmm1
vdivsd %xmm7,%xmm6,%xmm6
vmovsd %xmm7,0x8(%esp)
vxorpd %xmm2,%xmm2,%xmm2
vxorpd %xmm7,%xmm7,%xmm7

vaddsd %xmm0,%xmm1,%xmm0
vdivsd %xmm2,%xmm11,%xmm3
vaddsd %xmm2,%xmm0,%xmm0
vdivsd %xmm3,%xmm10,%xmm4

x86-32

x86-64
ISA Side Channel

- x86-32:
  - 32-bit: 100%
  - 64-bit: 0%

- x86-64:
  - 32-bit: 1.35%
  - 64-bit: 98.65%

Michael Schwarz (@misc0110) et al. — IAIK – Graz University of Technology
Results
• Distinguish browser including exact version
• Distinguish browser including exact version
• Both number and value of properties differ significantly
• Distinguish browser including exact version
• Both number and value of properties differ significantly
• toString as simple artificial property
• Distinguish browser including exact version
• Both number and value of properties differ significantly
• toString as simple artificial property
→ different string representations
• Distinguish browser including exact version
• Both number and value of properties differ significantly
• toString as simple artificial property
  ➔ different string representations
• 5796 different properties between Firefox and Chrome
• Distinguish browser including exact version
• Both number and value of properties differ significantly
• toString as simple artificial property
  → different string representations
• 5796 different properties between Firefox and Chrome
• Distinguished all 40 tested browsers
• Most extensions modify or add properties
Privacy Extensions

- Most extensions modify or add properties
- Installed privacy extensions (e.g., Canvas Defender, Ghostery)
• Most extensions modify or add properties
• Installed privacy extensions (e.g., Canvas Defenser, Ghostery)
• Not only presence, but also settings (e.g., protection level)
• Most extensions modify or add properties
• Installed privacy extensions (e.g., Canvas Defenser, Ghostery)
• Not only presence, but also settings (e.g., protection level)
• Canvas Defender only renamed original functions → automatically detected
Privacy Extensions

- Most extensions modify or add properties
- Installed privacy extensions (e.g., Canvas Defenser, Ghostery)
- Not only presence, but also settings (e.g., protection level)
- Canvas Defender only renamed original functions → automatically detected
  → Circumvents extension
• **Private** mode, e.g.,
- **Private** mode, e.g.,
  - Shared workers unavailable (Firefox)
• **Private** mode, e.g.,
  • Shared workers unavailable (Firefox)
  • Local databases unavailable (Edge)
• **Private mode**, e.g.,
  • Shared workers unavailable (Firefox)
  • Local databases unavailable (Edge)

• **Operating system**, e.g.,
- **Private mode**, e.g.,
  - Shared workers unavailable (Firefox)
  - Local databases unavailable (Edge)

- **Operating system**, e.g.,
  - Virtual-reality displays (Windows, partly on macOS)
• **Private mode**, e.g.,
  - Shared workers unavailable (Firefox)
  - Local databases unavailable (Edge)

• **Operating system**, e.g.,
  - Virtual-reality displays (Windows, partly on macOS)
  - Different font dimensions
• **Private mode**, e.g.,
  • Shared workers unavailable (Firefox)
  • Local databases unavailable (Edge)

• **Operating system**, e.g.,
  • Virtual-reality displays (Windows, partly on macOS)
  • Different font dimensions

• **CPU vendor** (WebGL and ISA side channel)
More Results

- **Private mode**, e.g.,
  - Shared workers unavailable (Firefox)
  - Local databases unavailable (Edge)

- **Operating system**, e.g.,
  - Virtual-reality displays (Windows, partly on macOS)
  - Different font dimensions

- **CPU vendor** (WebGL and ISA side channel)

- **Virtual machine**, e.g.,
More Results

- **Private mode**, e.g.,
  - Shared workers unavailable (Firefox)
  - Local databases unavailable (Edge)
- **Operating system**, e.g.,
  - Virtual-reality displays (Windows, partly on macOS)
  - Different font dimensions
- **CPU vendor** (WebGL and ISA side channel)
- **Virtual machine**, e.g.,
  - WebGL vendor (Firefox)
• **Private mode**, e.g.,
  • Shared workers unavailable (Firefox)
  • Local databases unavailable (Edge)

• **Operating system**, e.g.,
  • Virtual-reality displays (Windows, partly on macOS)
  • Different font dimensions

• **CPU vendor** (WebGL and ISA side channel)

• **Virtual machine**, e.g.,
  • WebGL vendor (Firefox)
  • Strange screen resolution
You can find our proof-of-concept implementation on:

- https://github.com/IAIK/jstemplate
Future Work

- Properties returned by function calls
• Properties returned by function calls
• Requires understanding function semantics
Future Work

- Properties returned by function calls
- Requires understanding function semantics
  → Number and type of arguments, side effects (e.g., close())
Future Work

- Properties returned by function calls
- Requires understanding function semantics
  → Number and type of arguments, side effects (e.g., close())
- New web standards (e.g., Web USB, Web NFC)
Non-static Properties

- Non-static properties can be used as distribution
Non-static Properties

- Non-static properties can be used as distribution

![Graph showing DOM Parsing timings for two users](chart.png)
Non-static Properties

- Non-static properties can be used as distribution

$\rightarrow$ timings depends e.g., on CPU speed
• JavaScript Template attacks detects various environment properties
• JavaScript Template attacks detects various environment properties
• Enables exploits, side-channel attacks and plausible phishing
Conclusion

- JavaScript Template attacks detects various environment properties
- Enables exploits, side-channel attacks and plausible phishing
- Tool for browser vendors to find leakage
• JavaScript Template attacks detects various environment properties
• Enables exploits, side-channel attacks and plausible phishing
• Tool for browser vendors to find leakage
• Advances field of fingerprinting
JavaScript Template Attacks

Michael Schwarz (@misc0110), Florian Lackner, Daniel Gruss (@lavados)
February 25, 2019

IAIK – Graz University of Technology