Statically Detecting JavaScript Obfuscation and Minification Techniques in the Wild

Marvin Moog†*, Markus Demmelt†, Michael Backes*, and Aurore Fass*

† Saarland University
* CISPA Helmholtz Center for Information Security

DSN 2021
Motivation - JavaScript

Popular client-side programming language

- JavaScript usage ≥ 97% websites [1]

Attack vector

- Aims at harming victims, e.g., exploiting vulnerabilities, stealing sensitive user data
- Code transformations:
  - hide the maliciousness of the code
  - impede its detection

[1] https://w3techs.com
Malicious JavaScript - Code Transformations

```
var m = "0000171Tj697udGgPLUNvh7xQD4TnGri4eurZGa7Rase6Lv5syE";
var usahzucesg = 'p';
var ZHUL = 'h';
var z bajbygel p = 'S';
for (var i=0; i<x.length; i++){
    var e = WScript.CreateObject("M"+zbajbygel p+"XML2.XMLHTTP");
    try {
        var yd gyqegoj v = "G"+"E"+"T";
        var m ter = ":"+"";
        var tj kh = x[i];
        var zn = '/';
        var kt = 't';
        e.open(yd gyqegoj v, ZHUL :+:kt :+:kt :+:kt :+:usahzucesg+m ter+zn+zn+tj kh+
        zn :+:counter :+:zn :+:zn :+:zn :+:?"+m, false);
        e.send(); [...]
```
Motivation - JavaScript

Popular client-side programming language

- JavaScript usage $\geq 97\%$ websites [1]

- Code transformations:
  - optimize website performance (e.g., save bandwidth / reduce loading times)
  - protect code privacy and intellectual property

Attack vector

- Aims at harming victims, e.g., exploiting vulnerabilities, stealing sensitive user data

- Code transformations:
  - hide the maliciousness of the code
  - impede its detection

[1] https://w3techs.com
Aim: hindering code analysis

- identifier obfuscation
- string obfuscation
- global array
- no alphanumeric
- dead-code injection
- control-flow flattening
- self-defending
- debug protection

Aim: reducing code size

- minification simple (e.g., shortening variable names, deleting whitespaces)
- minification advanced (e.g., function inlining, conditional operator)

[8] https://github.com/anseki/gnirts
Contribution

- Empirical study of JavaScript code transformations
  - benign vs. malicious code transformation techniques
  - evolution over time
Approach Overview

Detector 1 → Obfuscated → Detector 2

Detector 1 → Minified → Detector 2

Detector 1 → Regular → Detector 2

Detectors 1 and 2 → Transformation techniques
### Approach Overview

1. **Abstract Code Representation**
2. **Feature Extraction**
3. **Random Forest Classifier**

**Predictions**

- **Detector 1**
  - Obfuscated Minified
  - Regular
  - Accuracy ≥ 99.5%

- **Detector 2**
  - Accuracy ≥ 87%

**Data Flow**
- Data
  - True
  - False

**Features**
- Automatically selected features
- Hand-picked features

Code Transformations in the Wild

Alexa Top 10k websites

- 89.40% of the websites contain $\geq 1$ transformed script
- 68.60% of the scripts are transformed

Manual analysis:
- 83/100 regular
- 96/100 minified
- 99/100 obfuscated
- 100/100 transformed
Code Transformations in the Wild

Alexa Top 10k websites

- 89.40% of the websites contain ≥ 1 transformed script
- 68.60% of the scripts are transformed

- Most prevalent transformation techniques:
  - minification simple
  - minification advanced

- Minification used to reduce loading times, i.e., improve website performance

npm Top 10k packages

- 15.14% of the packages contain ≥ 1 transformed script
- 8.70% of the scripts are transformed

- Most prevalent transformation techniques:
  - minification simple
  - minification advanced

- Transformation/minification not popular
Evolution of Code Transformations over Time

Alexa Top 2k

npm Top 2k
## Code Transformations in Malicious JavaScript

<table>
<thead>
<tr>
<th>Source</th>
<th>Collected</th>
<th>#JS</th>
<th>Transformed?</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNC</td>
<td>2015-2017</td>
<td>4,514</td>
<td>65.94%</td>
</tr>
<tr>
<td>Hynek</td>
<td>2015-2017</td>
<td>29,484</td>
<td>73.07%</td>
</tr>
<tr>
<td>BSI</td>
<td>2017</td>
<td>36,475</td>
<td>28.93%</td>
</tr>
</tbody>
</table>
Malicious vs. Benign Code Transformations

**Malicious JS (BSI)***

---

**Benign JS (Alexa Top 2k)***

---

- **Minification Simple**
- **Minification Advanced**
- **Identifier Obfuscation**
- **String Obfuscation**

**Technique Usage Probability (%)**

**Tested month**

- 2015-05
- 2016-01
- 2017-01
- 2018-01
- 2019-01
- 2020-01
Conclusion

- Studied the prevalence of JavaScript code transformations

Detector 1 ➔ Obfuscated ➔ Detector 2 ➔ Transformation techniques ➔ Minified ➔ Regular

Benign JS: Alexa Top 2k
Benign JS: npm Top 2k
Malicious JS: BSI

Thank you