Anything to Hide? Studying Minified and Obfuscated Code in the Web

Philippe Skolka  Cristian-Alexandru Staicu  Michael Pradel

TU Darmstadt

www.software-lab.org

15th of May 2019
Code Distribution on the Web

```javascript
function sum(first, second) {
  return first + second;
}
```

- **write**
- **deploy**
- **serve**
Code Distribution on the Web

```javascript
function sum(first, second) {
    return first + second;
}
```

```javascript
function sum(n, u) {
    return n + u;
}
```
Common Transformations: Minification and Obfuscation

Hand-written

```javascript
function hi(name) {
    console.log("Hi" + " " + name);
}
hi();
```

**Objective:** ↑ maintainability
Common Transformations: Minification and Obfuscation

**Hand-written**

```javascript
function hi(name) {
    console.log("Hi " + " " + name);
}
hi();
```

**Objective:** ↑ maintainability

**Minified**

```javascript
function hi(i){console.log("Hi "+i)}hi();
```

**Objective:** ↓ code size
Common Transformations: Minification and Obfuscation

Hand-written

```javascript
function hi(name) {
    console.log("Hi " + " " + name);
}
hi();
```

Objective: ↑ maintainability

Minified

```javascript
function hi(i){console.log("Hi "+i)}hi();
```

Objective: ↓ code size

Obfuscated

```javascript
var a=[\x6c\x6f\x67\'] ; var b=function(c,d){
c=c-0x0 ; var e=a[c] ; return e ;} ; function
    c(d){console[b('0x0')]('\x48\x69'+'\x20'+d);}c();
```

Objective: ↓ understandability
**Research Questions**

**RQ1:** How prevalent is transformed code on the web?
Research Questions

**RQ1:** How prevalent is transformed code on the web?

**RQ2:** Which tools are used for obfuscation on the web?
Research Questions

RQ1: How prevalent is transformed code on the web?

RQ2: Which tools are used for obfuscation on the web?

RQ3: Does prevalence differ among website categories?

RQ4: What behavior is hidden using obfuscation?

RQ5: How do transformations impact performance?

RQ6: How do transformations impact correctness?
Research Questions

RQ1: How prevalent is transformed code on the web?

RQ2: Which tools are used for obfuscation on the web?

RQ3: Does prevalence differ among website categories?

RQ4: What behavior is hidden using obfuscation?
Research Questions

RQ1: How prevalent is transformed code on the web?

RQ2: Which tools are used for obfuscation on the web?

RQ3: Does prevalence differ among website categories?

RQ4: What behavior is hidden using obfuscation?

RQ5: How do transformations impact performance?
Research Questions

**RQ1:** How prevalent is transformed code on the web?

**RQ2:** Which tools are used for obfuscation on the web?

**RQ3:** Does prevalence differ among website categories?

**RQ4:** What behavior is hidden using obfuscation?

**RQ5:** How do transformations impact performance?

**RQ6:** How do transformations impact correctness?
Research Questions

**RQ1:** How prevalent is transformed code on the web?

**RQ2:** Which tools are used for obfuscation on the web?

**RQ3:** Does prevalence differ among website categories?

**RQ4:** What behavior is hidden using obfuscation?

**RQ5:** How do transformations impact performance?

**RQ6:** How do transformations impact correctness?
Why Machine Learning?

Large scale study, expensive to do manually
Heuristics hard to get right
Training data easy to acquire
Why Machine Learning?

- Large scale study, expensive to do manual
Why Machine Learning?

- Large scale study, expensive to do manual
- Heuristics hard to get right
Why Machine Learning?

- Large scale study, expensive to do manual
- Heuristics hard to get right
- Training data easy to acquire
Methodology

Different transformation tools

(Transformation)
Methodology

Different transformation tools → (Transformation)

Train binary classifiers → (Training)
Methodology

Different transformation tools

Train binary classifiers

Classifiers

Web code

Classifiers

Answer questions

Runtime analysis

(Transformed)

(Training)

(Study)
Methodology

- Different transformation tools
- Train binary classifiers
- Web code
- Runtime analysis
- Classifiers
- Answer questions

(Transformation)
(Training)
(Study)
# Transformation Tools

<table>
<thead>
<tr>
<th>Minifiers</th>
<th>Obfuscators</th>
</tr>
</thead>
<tbody>
<tr>
<td>UglifyJS</td>
<td>javascript-obfuscator</td>
</tr>
<tr>
<td>babel-minify</td>
<td>javascriptobfuscator.com</td>
</tr>
<tr>
<td>Google Closure Compiler</td>
<td>DaftLogic Obfuscator</td>
</tr>
<tr>
<td>javascript-minifier.com</td>
<td>jfogs</td>
</tr>
<tr>
<td>Matthias Mullie Minify</td>
<td>JSObfu</td>
</tr>
<tr>
<td>YUI Compressor</td>
<td></td>
</tr>
</tbody>
</table>

- **11 tools** with a total of 46 different configuration
- transform files from the “150k Javascript Dataset”\(^1\)

---

\(^1\)Raychev, V., Bielik, P., Vechev, M. and Krause, A., *Learning Programs from Noisy Data*, POPL ’16
Classification Tasks

**Seven** binary classifiers:

- **Transformation classifier**: Is the code transformed?
- **Obfuscation classifier**: Is the code obfuscated?
- **Tool-X classifier**: Is the code produced by a given obfuscation tool?
Seven binary classifiers:

- **TRANSFORMATION** classifier
  Is the code transformed?
Seven binary classifiers:

- **TRANSFORMATION** classifier
  Is the code transformed?

- **OBFUSCATION** classifier
  Is the code obfuscated?
Classification Tasks

**Seven** binary classifiers:

- **TRANSFORMATION** classifier
  Is the code transformed?

- **OBFUSCATION** classifier
  Is the code obfuscated?

- **TOOL-X** classifier * 5
  Is the code produced by a given obfuscation tool?
Binary Classifiers

Convolutional neural network
- reuse existing architecture [Mou et al., AAAI, 2016]
Binary Classifiers

Convolutional neural network
  - reuse existing architecture [Mou et al., AAAI, 2016]

- input to the network: simplified abstract syntax tree (AST) representation of code
## Binary Classifiers

**Convolutional neural network**
- reuse existing architecture [Mou et al., AAAI, 2016]

- input to the network: simplified abstract syntax tree (AST) representation of code
- 30 feature vector size, 50 epochs, batch size 1
function sum(first, second) {
    return first + second;
}
function sum(first, second) {
    return first + second;
}
function sum(first,second) {
    return first + second;
}
function sum(first, second) {
    return first + second;
}
**Accuracy of the Classifiers**

**Validation set =**
2,500 files from the corpus and their transformed versions

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRANSFORMATION (no spaces info)</td>
<td>85.58%</td>
</tr>
<tr>
<td><strong>TRANSFORMATION</strong></td>
<td>95.06%</td>
</tr>
<tr>
<td>OBFUSCATION (no spaces info, no identifiers length)</td>
<td>75.43%</td>
</tr>
<tr>
<td>OBFUSCATION (no spaces info)</td>
<td>99.83%</td>
</tr>
<tr>
<td><strong>OBFUSCATION</strong></td>
<td>99.95%</td>
</tr>
<tr>
<td>TOOL-JSObfu</td>
<td>100%</td>
</tr>
<tr>
<td>TOOL-jsobfcom</td>
<td>100%</td>
</tr>
<tr>
<td>TOOL-jfogs</td>
<td>99.56%</td>
</tr>
<tr>
<td>TOOL-daft-logic</td>
<td>100%</td>
</tr>
<tr>
<td>TOOL-jsobf</td>
<td>100%</td>
</tr>
</tbody>
</table>
Accuracy of the Classifiers: User Study

- **five users** and **200 scripts** from the web:
  - 50 positive and 50 negative classified by **TRANSFORMATION**
  - 50 positive and 50 negative classified by **OBFUSCATION**
Accuracy of the Classifiers: User Study

- **five users** and **200 scripts** from the web:
  - 50 positive and 50 negative classified by TRANSFORMATION
  - 50 positive and 50 negative classified by OBFUSCATION
- **0.81 inter-rater pairwise agreement**
Accuracy of the Classifiers: User Study

- **five users** and **200 scripts** from the web:
  - 50 positive and 50 negative classified by **TRANSFORMATION**
  - 50 positive and 50 negative classified by **OBFUSCATION**
- **0.81** inter-rater pairwise agreement
Methodology

1. Different transformation tools
2. Train binary classifiers
3. Web code
4. Runtime analysis

(Transformation)

(Training)

Classifiers

(Study)

Answer questions
Experimental Setup

- top 100,000 most popular websites
Experimental Setup

- top 100,000 most popular websites
- both inlined and included scripts
Experimental Setup

- top 100,000 most popular websites
- both inlined and included scripts
- more than 400,000 unique scripts
Experimental Setup

- top **100,000** most popular websites
- both inlined and included scripts
- more than **400,000 unique scripts**
- each script mapped to a category, e.g., ”news”
RQ1: How prevalent is transformed code on the web?

web scripts judged by TRANSFORMATION classifier

- regular: 61.50%
- transformed: 38.50%

2,500 scripts are obfuscated.
RQ1: How prevalent is transformed code on the web?

web scripts judged by TRANSFORMATION classifier

- regular: 61.50%
- transformed: 38.50%

One in three scripts are transformed

regular/minified
obfuscated

- regular/minified: 99.33%
- obfuscated: 0.67%

> 2,500 scripts are obfuscated
RQ1: How prevalent is transformed code on the web?

web scripts judged by **TRANSFORMATION** classifier

- Regular: 61.50%
- Transformed: 38.50%

One in three scripts are transformed

web scripts judged by **OBFUSCATION** classifier

- Regular/minified: 99.33%
- Obfuscated: 0.67%

2,500 scripts are obfuscated
RQ1: How prevalent is transformed code on the web?

web scripts judged by **TRANSFORMATION** classifier

- regular: 61.50%
- transformed: 38.50%

One in three scripts are transformed

web scripts judged by **OBFUSCATION** classifier

- regular/minified: 99.33%
- obfuscated: 0.67%

> 2,500 scripts are obfuscated
RQ2: Which tools are used for obfuscation on the web?
RQ2: Which tools are used for obfuscation on the web?

**Obfuscation**

**TOOL: daft-logic**

2,842 unique obfuscated scripts
RQ2: Which tools are used for obfuscation on the web?

2,842 unique obfuscated scripts
RQ3: Does prevalence differ among website categories?

Transformed scripts in different categories

<table>
<thead>
<tr>
<th>Category</th>
<th>Transformed</th>
<th>Regular</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adult/Sexually Explicit (591)</td>
<td>70%</td>
<td>30%</td>
</tr>
<tr>
<td>Advertisements (12128)</td>
<td>60%</td>
<td>40%</td>
</tr>
<tr>
<td>Arts &amp; Entertainment (10084)</td>
<td>80%</td>
<td>20%</td>
</tr>
<tr>
<td>Computing &amp; Internet (28399)</td>
<td>50%</td>
<td>50%</td>
</tr>
<tr>
<td>Education (13341)</td>
<td>30%</td>
<td>70%</td>
</tr>
<tr>
<td>Job Search &amp; Career Development (1250)</td>
<td>45%</td>
<td>55%</td>
</tr>
<tr>
<td>News (21107)</td>
<td>20%</td>
<td>80%</td>
</tr>
<tr>
<td>Search Engines (6400)</td>
<td>50%</td>
<td>50%</td>
</tr>
<tr>
<td>Shopping (11976)</td>
<td>40%</td>
<td>60%</td>
</tr>
<tr>
<td>Travel (7029)</td>
<td>30%</td>
<td>70%</td>
</tr>
<tr>
<td>Other (*)</td>
<td>40%</td>
<td>60%</td>
</tr>
</tbody>
</table>
RQ3: Does prevalence differ among website categories?

Obfuscated scripts in different categories

Percentage of scripts

Category

Adult/Sexually Explicit (591)
Arts & Entertainment (10084)
Chat (193)
Computing & Internet (28399)
Glamour & Intimate Apparel (418)
Job Search & Development (1250)
News (21107)
Search Engines (6400)
Shopping (11976)
Travel (7029)
Other (*)
RQ4: What behavior is hidden using obfuscation?

- perform lightweight **dynamic analysis** in Node.js
RQ4: What behavior is hidden using obfuscation?

- perform lightweight **dynamic analysis** in Node.js
- collect and analyze traces with accessed properties
RQ4: What behavior is hidden using obfuscation?

- perform lightweight **dynamic analysis** in Node.js
- collect and analyze traces with accessed properties
- multiple scripts access **privacy sensitive APIs**:
  - 11% read the **cookie**
  - 10% access the **userAgent**
  - 3% read the **referrer**
  - 10% inject additional JavaScript code

Several scripts seem to perform browser fingerprinting.
RQ4: What behavior is hidden using obfuscation?

- perform lightweight **dynamic analysis** in Node.js
- collect and analyze traces with accessed properties
- multiple scripts access **privacy sensitive APIs**:
  - 11% read the **cookie**
  - 10% access the **userAgent**
  - 3% read the **referrer**
  - 10% inject additional JavaScript code
- several scripts seem to perform browser fingerprinting
Methodology

Different transformation tools

Train binary classifiers

Classifiers

Web code

Classifiers

Answer questions

Runtime analysis

(Transformation)

(Training)

(Study)
Experimental Setup

- 10 libraries with more than 400 tests each
Experimental Setup

- 10 libraries with more than 400 tests each
- 46 transformed versions of the libraries
Experimental Setup

- **10 libraries** with more than 400 tests each
- 46 transformed versions of the libraries
- for each version, run the tests **20 times**
Experimental Setup

- **10 libraries** with more than 400 tests each
- 46 transformed versions of the libraries
- for each version, run the tests **20 times**
- run tests on a machine with 6 cores and 16GB RAM
Experimental Setup

- **10 libraries** with more than 400 tests each
- 46 transformed versions of the libraries
- for each version, run the tests **20 times**
- run tests on a machine with 6 cores and 16GB RAM
- compare number of failing tests and performance of transformed vs. original code
RQ5: How do transformations impact performance?
RQ6: How do transformations impact correctness?

Proportion of valid code for 10 libraries

- minified-babel
- minified-butternut
- minified-jsmminify
- minified-uglify
- minified-yui
- obfuscated-daft-logic
- obfuscated-jfogs
- obfuscated-jsobf
- obfuscated-jsobf-com
- regular-regular

30% are not semantics-preserving
RQ6: How do transformations impact correctness?

30% are not semantics-preserving
Conclusions

100,000
most popular websites
Conclusions

100,000 most popular websites

1 in 3 scripts are transformed
Conclusions

100,000
most popular websites

1 in 3
scripts are transformed

>2,500
obfuscated scripts
Conclusions

100,000 most popular websites

1 in 3 scripts are transformed

>2,500 obfuscated scripts

transformations are prevalent on the web
Conclusions

100,000 most popular websites

1 in 3 scripts are transformed

>2,500 obfuscated scripts

- transformations are prevalent on the web
- obfuscation is seldom
Conclusions

100,000 most popular websites

1 in 3 scripts are transformed

>2,500 obfuscated scripts

- transformations are prevalent on the web
- obfuscation is seldom
- ML models effective at analyzing web code
Conclusions

100,000
most popular websites

1 in 3
scripts are transformed

>2,500
obfuscated scripts

- transformations are prevalent on the web
- obfuscation is seldom
- ML models effective at analyzing web code
## Obfuscation Techniques

<table>
<thead>
<tr>
<th>Transformation techniques</th>
<th>Obfuscation tools</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>jsobf</td>
</tr>
<tr>
<td>String splitting</td>
<td>✓</td>
</tr>
<tr>
<td>Keyword substitution</td>
<td></td>
</tr>
<tr>
<td>String concatenation</td>
<td></td>
</tr>
<tr>
<td>Encoding the entire code</td>
<td></td>
</tr>
<tr>
<td>Encrypting the entire code</td>
<td></td>
</tr>
<tr>
<td>Identifier encoding</td>
<td>✓</td>
</tr>
<tr>
<td>String encoding</td>
<td>✓</td>
</tr>
<tr>
<td>Dead code injection</td>
<td>✓</td>
</tr>
<tr>
<td>Control flow flattening</td>
<td>✓</td>
</tr>
<tr>
<td>String array</td>
<td>✓</td>
</tr>
<tr>
<td>Code protecting techniques</td>
<td>✓</td>
</tr>
</tbody>
</table>
### Classification Tasks

**Seven** binary classifiers:

1. **TRANSFORMATION** classifier
   - no
   - regular
   - minified
   - obfuscated

2. **OBFUSCATION** classifier
   - no
   - regular
   - minified
   - obfuscated

3. **TOOL-X** classifier *5*
   - no
   - regular
   - minified
   - obf. w/o TOOL-X
   - obfuscated with TOOL-X
Examples of Obfuscated Code

```javascript
$axure.loadDocument(function() {
  var _ = function() { var r=0,a=arguments; for(var i=0; i<a.length; i+=2) r[a[i-1]] = a[i+1]; return r; }
  var _creator = function() { return _b, (c,d,e,f,g,f,h,f,i,d,j,k,l,m,d,n,f,o,
f,p,f,q,[]), r,d,s,t,u,d), v, _(w, [(_x,y,z,A,B,C,D, [(_x,E,z,A,B,F,(_x,G,z,A,B,H,(_x,J,(_x,Z,A,B,J),(_x,K,z,A,B,L),(_x,M,z,N,B,O,D,[(_x,P,z,A,B,Q),(_x,R,z,A,B,S),(_x,T,z,A,E,
U),(_x,V,z,A,B,W),(_x,X,z,N,B,O,D,[(_x,Y,z,A,B,Z)],(_x,xa,z,N,B,O,D,[(_x,bb,z,A,B,Bc,d,[(_x,bd,z,A,B,be)],(_x,bf,z,A,B,bg)],(_x,bh,z,N,B,O,D,[(_x,bi,z,A,B,bj,D,
([_x,bk,z,A,B,bl]),([_x,bn,z,A,B,bn]),([_x,bo,z,A,B,bp]),([_x,bq,z,A,B,bR,D,[(_x,bs,z,A
,B,bi)),([_x,bu,z,A,B,bv),([_x,bw,z,A,B,bx),([_x,by,z,A,B,bz),([_x,ba,z,A,B,bB),([_x,bc,z
A,B,bD)],([_x,bE,z,N,B,O,D,[(_x,bF,z,A,B,bG,D,[(_x,bH,z,N,B,O,D,[(_x,bI,z,A,B,bI)],([_x,bK,z,A,B,bL,D,[(_x,bm,z,A,B,blN)],([_x,bO,z,A,B,bp),([_x,bQ,z,A,B,bR),([_x,bS,z,A,B,bT),([_x,bU,z,A,B,bV),([_x,bW,z,N,B,O,D,[(_x,bX,z,A,B,bY),(_x,bZ,z,A,B,ca),(_x,bz,bC,cc),(_x,cd,z,A,B,ce)],(_x,cf,z,N,B,O,D,[(_x,cf,z,A,B,ca),(_x,ck,z,A,B,cl),([_x,cm,z,A,B,cm),(_x,co,z,A,B,cp)],(_x,cq,z,N,B,O,D,[(_x,cr,z,A,B,cs),(_x,ct,z,A,B,ct),(_x,cv,z,A,B,cv),(_x,cv,z,A,B,cv),(_x,cv,z,A,B,cv)])

(document.write(String.fromCharCode(115,117,112,112,111,114,116,64,104,111,115,116,46,98,103)))

(function(_){[function(_){return(_, [[fromCharCode].source],(0x62,7.5e+1,0x74,9,
9E+1,1022,0,153)+(0117,0112,1.12e+2,0106,1,116,0x36,5.6E+1,5.1e+1,0x36,0163,45,0x
57,8.3e+1)])(String)=(function(_){return(_, [[fromCharCode].source],(7.8E+1,109,
0137,0146,0x75,0163,71,0156,8.8e+1,0x47,104,119,0x45,9.7E+1,99,8.1e+1,6.6E+1,109)+
(5.7e+1,0131,5.3e+1,0142,1.07e+2,52,49)])(String);})(window);
```
SVM Classifier

- consider most popular 30,000 tokens in our dataset
- identifiers embedding

```javascript
function sum(first, second) {
  return first + second;
}
```

```
{sum → 1, first → 2, second → 2}
```

```
foo ... first ... sum ... second
  0 0 2 0 1 0 2
```

- we use tf-idf values to compute the vector entries