# Browser Fingerprinting

Tronje Krabbe
tronje@informatik.uni-hamburg.de

Uni Hamburg
Working Group on Security and Privacy

September 12, 2017

# Overview

Browser Fingerprinting

Thesis

Considerations

Schedule

# What is that?

The average browser exposes enough information to construct a 'fingerprint', which can uniquely identify a user. Information used can be:

- user agent
- installed fonts
- GPU model/vendor
- WebGL behavior
- ...

# Why is it 'bad'?

- users are usually not made aware
- more easily obfuscated than a cookie
- cannot be easily avoided

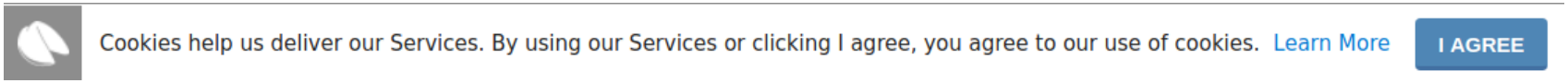


Figure: Reddit.com's cookie warning

# Obfuscation

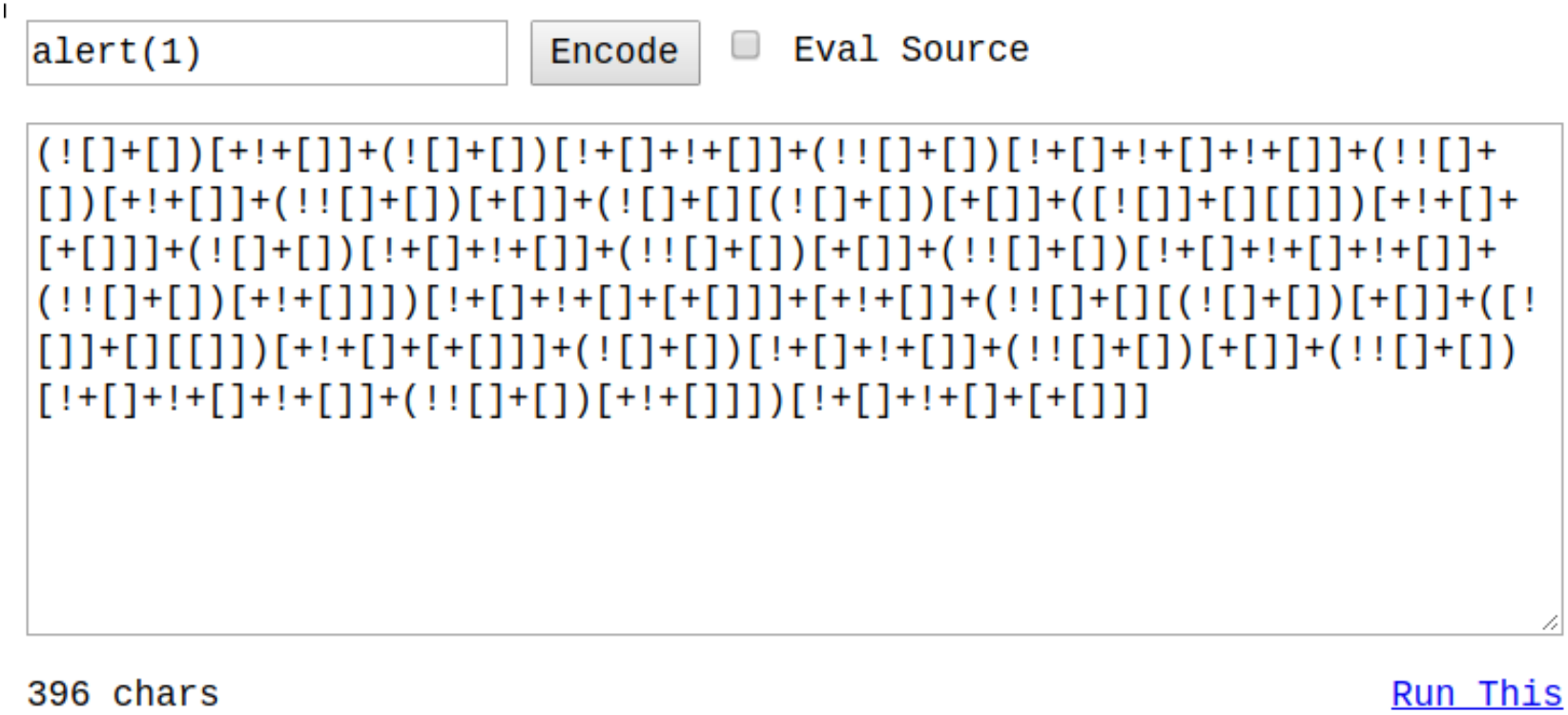


Figure: Obfuscated JavaScript code, thanks to JSFuck.com

# Thesis

Develop a software that can reliably detect whether a website employs fingerprinting, and integrate it into PrivacyScore.org

# How?

Simply record all JavaScript function calls, and see if it looks like a fingerprint is being constructed.

# Identify most commonly used fingerprinting methodologies

- Flash can be used, but unlikely to be widespread
- fonts and WebGL are popular, what about e.g. sound?
- how likely will these continue to be used?
- JavaScript and WebGL best candidates

# Identify most commonly used libraries

- site that uses a common library is easily 'convicted'
- could be mostly in-house libs $\rightarrow$ pretty much useless
- JavaScript is easily obfuscated

# False Positives

JavaScript calls that look like they're constructing a fingerprint may also be used for a completely different cause. Recall:

- user agent
- installed fonts
- GPU model/vendor
- WebGL behavior
- ...

# Scoring System

Instead of reporting **“yes, fingerprinting is going on!”** or **“no, fingerprinting is not going on!”**, give a score to represent a likelihood.

# Performance

- efficient algorithm
- language choice
  - Python
    - slow
    - OpenWPM & PrivacyScore are written in Python
  - C++, Rust
    - hard to integrate
    - more complicated than Python

# Schedule

*"The first 90 percent of the code accounts for the first 90 percent of the development time. The remaining 10 percent of the code accounts for the other 90 percent of the development time."* - Tom Cargill, Bell Labs

- 4 weeks: first implementation and tests in place
- 6 weeks: evaluation and improvements
- 4 weeks: writing the actual thesis paper
- about 1 month to spare

# Thank you!