RUB

XSinator.com: From a Formal Model to the Automatic Evaluation of Cross-Site Leaks in Web Browsers

RUHR-UNIVERSITÄT BOCHUM

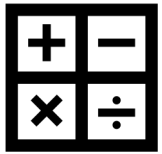
Lukas Knittel, Christian Mainka, Marcus Niemietz, Dominik Noß, Jörg Schwenk

HORST GÖRTZ INSTITUTE

Overview

XS-Leak Ingredients

1. *Detectable Difference*
2. *Inclusion Method*
3. *Leak Technique*



Formal Model

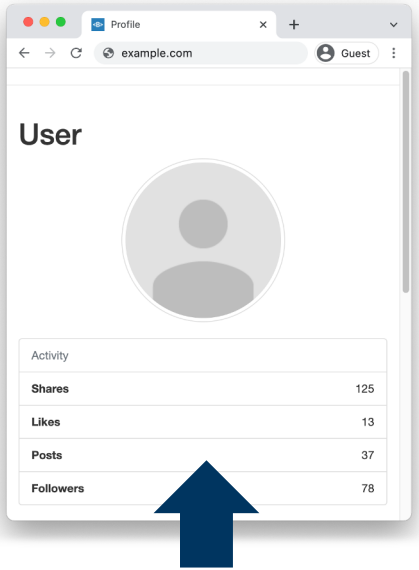
- gain in-depth insights
- systematically produce new attacks

XSinator

- browser test suite
- implements 34 XS-Leaks

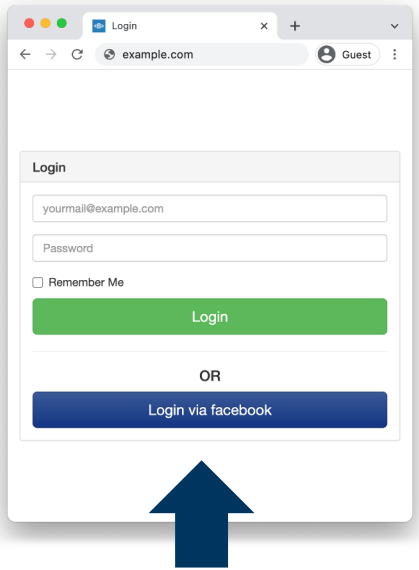
XS-Leak Ingredient 1: Detectable Difference

Case 1:
Logged in



0 Iframes

Case 2:
Not logged in



1 hidden Iframe

Status Code

Detect server errors (5XX), client errors (4XX), authentication errors (401)

API Usage

Detect Web APIs' usage (e.g., WebSocket, Service Worker, Payment API)

Redirects

Detect HTTP, HTML, or JavaScript redirects

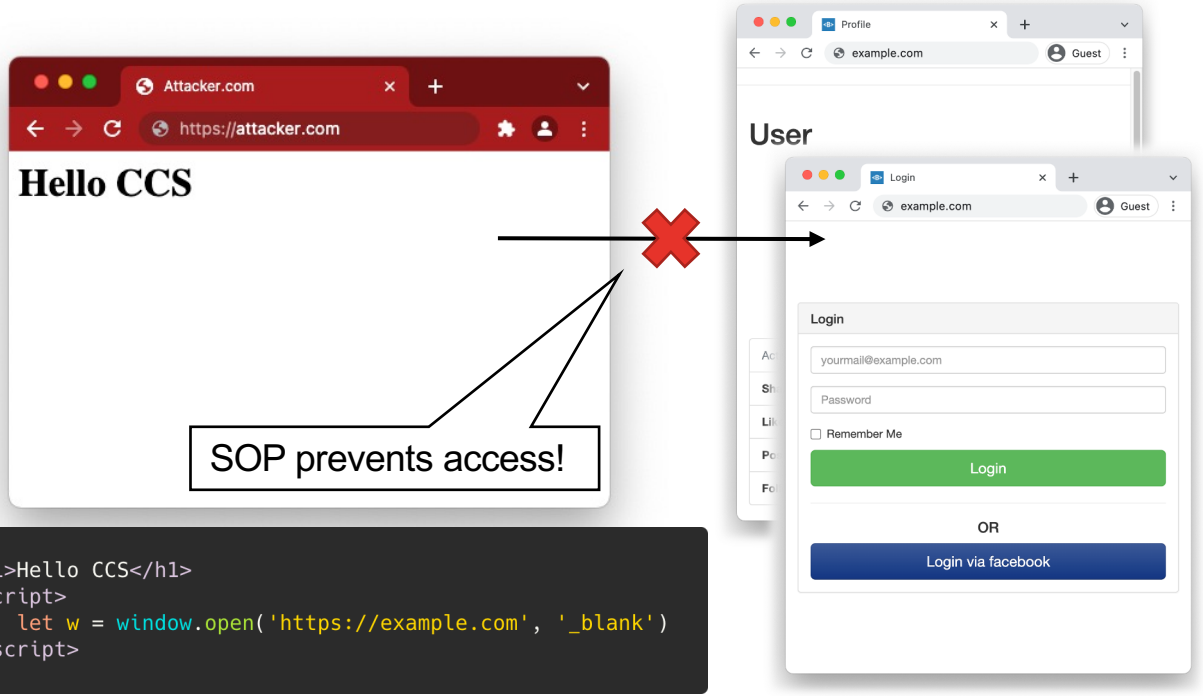
Page Content

Detect differences in the HTTP response body

HTTP Header

Detect differences in the HTTP response header

XS-Leak Ingredient 2: Inclusion Method




```
<h1>Hello CCS</h1>
<script>
    let w = window.open('https://example.com', '_blank')
</script>
```

HTML Elements

`<script>, <img>, <link>`

Frames

`<iframe>, <object>, <embed>`

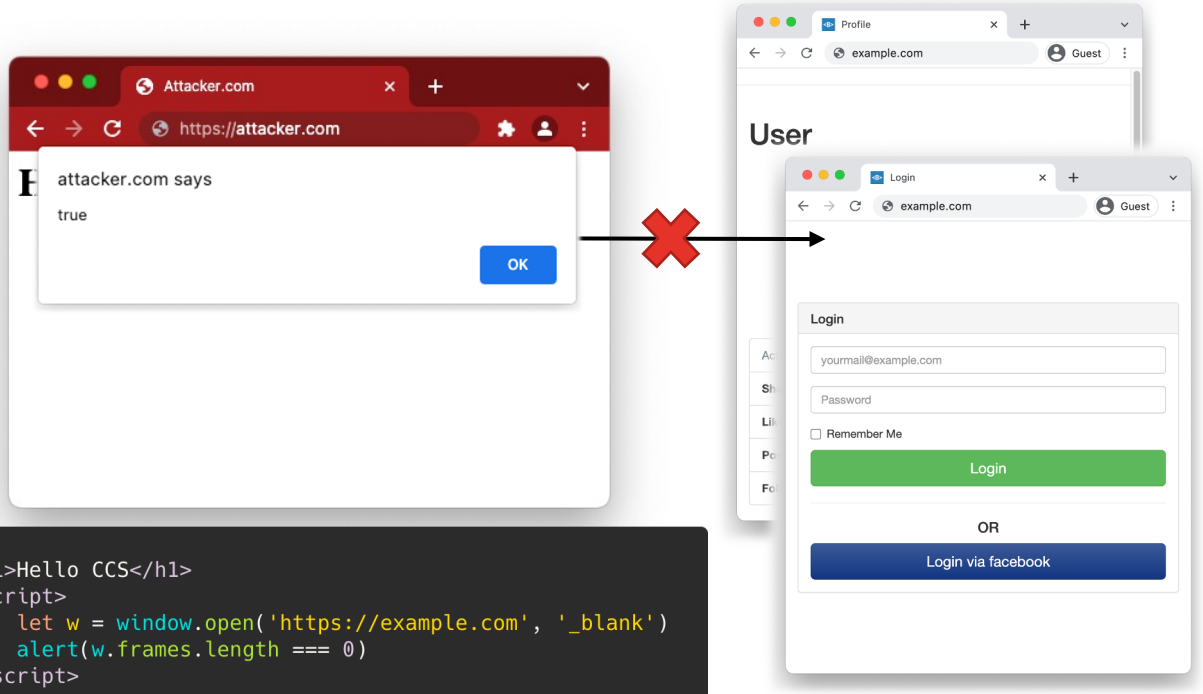
Pop-ups

`window.open()`

JavaScript Requests

Fetch API

XS-Leak Ingredient 3: Leak Technique




```
<h1>Hello CCS</h1>
<script>
    let w = window.open('https://example.com', '_blank')
    alert(w.frames.length === 0)
</script>
```

Event Handler

Error Messages

Global Limits

Global State

Performance API

Readable Attributes

Formal XS-Leak Description

Definition 1 – State-dependent resource

A state-dependent resource sdr is a 2-tuple $(url, (s, d))$, where $(s, d) \in \{(s_0, d_0), (s_1, d_1)\}$.

- url is a URL resource on the target web application.
- $S = \{s_0, s_1\}$ is a set of two different states of the target web application.
- $D = \{d_0, d_1\}$ is a set that represents the difference of the web application's behavior that depends on s_0 and s_1.

Formal XS-Leak Description

Definition 2 – Cross-Site Leak

A Cross-Site Leak is a function $xsl()$ that outputs a bit b', that is $b' = xsl(sdr, i, t)$

- $sdr \in SDR$ is a state-dependent resource.
- $i \in I$ is an inclusion method to request a cross-origin resource.
- $t \in T$ is a leak technique which can be used to observe state-dependent resources cross-origin.

If there exists an inclusion method i and a leak technique t such that $xsl((url, (s_b, d_b)), i, t) = b$ then the difference d is **detectable**.

New Attacks – Detect WebSockets

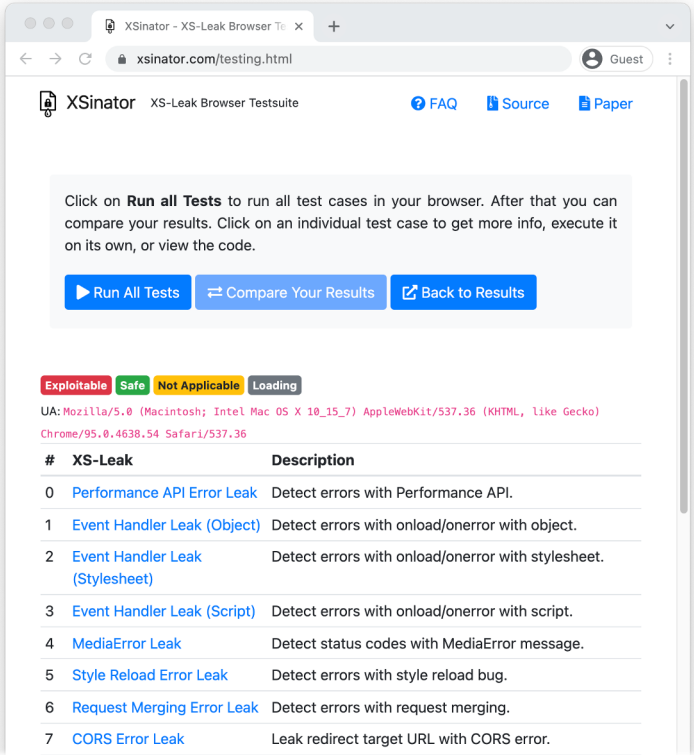
Global browser connection limit

Max 200 WebSockets

Max 256 WebSockets

Detect number of WebSockets on a page

XSinator - XS-Leak Browser Test Suite



Automatically tests 34 XS-Leaks in the browser

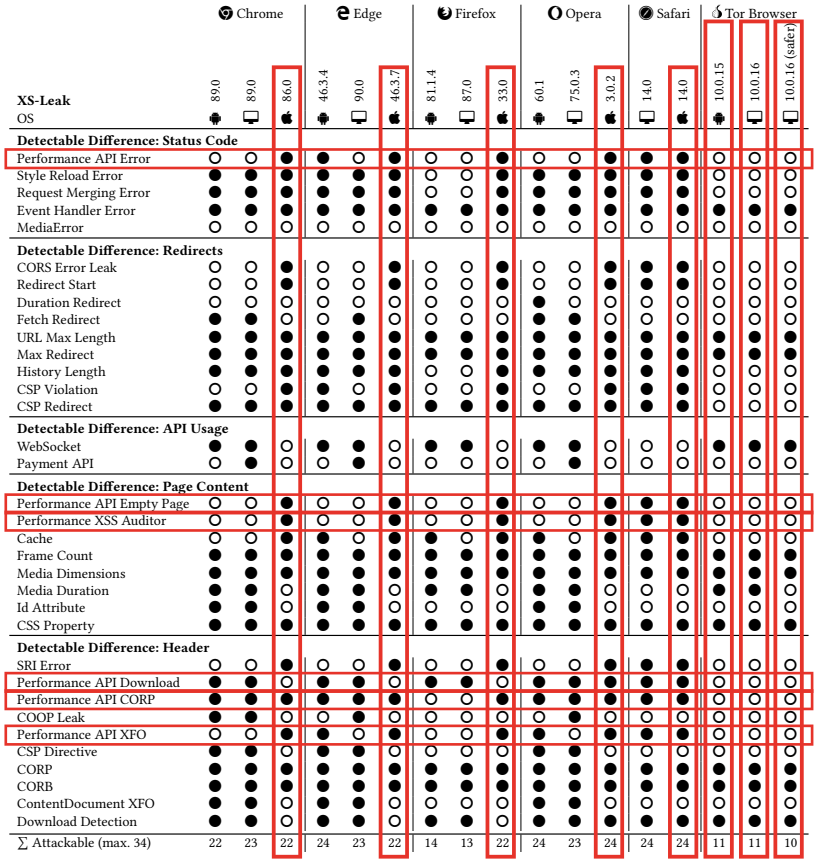
- **Testing site** acts as the attacker site
- **Vulnerable web application** simulates the state-dependent resource
- **Database** stores previous test results

Results

Browsers in iOS behave similarly due to the same underlying browser engine.

Tor has more restrictive browser settings; some APIs are explicitly deactivated.

The Performance API allows an attacker to leak various characteristics.



	Chrome			Edge			Firefox			Opera			Safari		Tor Browser		
XS-Leak	89.0	89.0	86.0	46.3.4	90.0	46.3.7	81.1.4	87.0	33.0	60.1	75.0.3	3.0.2	14.0	14.0	10.0.15	10.0.16	10.0.16 (safer)
OS																	
Detectable Difference: Status Code																	
Performance API Error	○	○	●	●	○	●	○	○	●	○	○	●	●	●	○	○	○
Style Reload Error	●	●	●	●	●	●	○	○	●	●	●	●	●	●	○	○	○
Request Merging Error	●	●	●	●	●	●	○	○	●	●	●	●	●	●	○	○	○
Event Handler Error	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
MediaError	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
Detectable Difference: Redirects																	
CORS Error Leak	○	○	●	○	○	●	○	○	●	○	○	●	●	●	○	○	○
Redirect Start	○	○	●	○	○	●	○	○	●	○	○	●	●	●	○	○	○
Duration Redirect	○	○	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○
Fetch Redirect	●	●	○	○	●	○	○	○	○	●	●	○	○	○	○	○	○
URL Max Length	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
Max Redirect	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
History Length	●	●	●	●	●	●	○	○	●	●	●	●	●	●	○	○	○
CSP Violation	○	○	●	●	○	●	○	○	●	○	○	●	●	●	○	○	○
CSP Redirect	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	○	○
Detectable Difference: API Usage																	
WebSocket	●	●	○	●	●	○	●	●	○	●	●	○	○	○	●	●	●
Payment API	○	●	○	○	●	○	○	○	○	○	●	○	○	○	○	○	○
Detectable Difference: Page Content																	
Performance API Empty Page	○	○	●	○	○	●	○	○	●	○	○	●	●	●	○	○	○
Performance XSS Auditor	○	○	●	○	○	●	○	○	●	○	○	●	●	●	○	○	○
Cache	○	○	●	●	○	●	●	○	●	●	○	●	●	●	○	○	○
Frame Count	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
Media Dimensions	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
Media Duration	●	●	○	●	●	○	●	●	○	●	●	○	○	○	●	●	○
Id Attribute	●	●	○	●	●	○	○	○	○	●	●	○	○	○	○	○	○
CSS Property	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
Detectable Difference: Header																	
SRI Error	○	○	●	○	○	●	○	○	●	○	○	●	●	●	○	○	○
Performance API Download	●	●	○	●	●	○	●	●	○	●	●	●	●	●	○	○	○
Performance API CORP	●	●	●	●	●	●	○	○	●	●	●	●	●	●	○	○	○
COOP Leak	●	●	○	○	●	○	○	○	○	○	●	○	○	○	○	○	○
Performance API XFO	○	○	●	●	○	●	○	○	●	●	○	●	●	●	○	○	○
CSP Directive	●	●	○	●	●	○	○	○	○	●	●	○	○	○	○	○	○
CORP	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
CORB	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●
ContentDocument XFO	●	●	○	●	●	○	○	○	○	○	○	○	○	○	○	○	○
Download Detection	●	●	○	●	●	○	●	●	○	●	●	●	●	●	●	●	●
∑ Attackable (max. 34)	22	23	22	24	23	22	14	13	22	24	23	24	24	24	11	11	10

Table 5: Evaluation results overview categorized by its detectable differences. Successful attacks are depicted with full circuits (●), safe browser are indicated with empty circuits (○).

Conclusion

- Introduce a formal model for XS-Leaks
- Analyze known XS-Leaks and show how they fit in our model
- Contribute 14 new XS-Leak attacks
- Implemented XSinator, an easy-to-use website to evaluate browsers
- Evaluated 56 browser/OS combinations against a set of 34 XS-Leaks

Thank you for listening!
Any Questions?

xsinator.com
lukas.knittel@rub.de
kunte_ctf