SugarCoat:
Programmatically Generating Privacy-Preserving, Web-Compatible Resource Replacements for Content Blocking (CCS 2021)

Michael Smith†, Pete Snyder*, Benjamin Livshits*, Deian Stefan†

†UC San Diego   *Brave Software
About Brave: The Company

- Privacy focused browser
- Novel, private, performant, user first web funding model
- Opt-in, user-rewarding ad system
- Search engine: https://search.brave.com
- Video conferencing: https://talk.brave.com
About Brave: Research Department

- 5 researchers, 8 research-engineers
- Publication + product focused
- Topics like: privacy, security, cryptography, ML, etc.
- Standards bodies: IETF, W3C, WHATWG, etc.
- Active university <-> industry collaborations
About Brave: Deployed Privacy Research

- Supporting filter-lists with behavioral fingerprinting of trackers (S&P 2021)
- Filter list maintenance and optimization (SIGMETRICS 2020)
- Compatibility of browser storage policies (WWW 2022)
- Automated webcompat detection, browser covert channels, navigational tracking, and more…
SugarCoat: Programmatically Generating Privacy-Preserving, Web-Compatible Resource Replacements for Content Blocking (CCS 2021)

Michael Smith†, Pete Snyder*, Benjamin Livshits*, Deian Stefan†

†UC San Diego  *Brave Software
37% (and growing) of Web users rely on content blockers

CISA tells agencies to consider ad blockers to fend off 'malvertising'
built-in blocking

Firefox  Brave

browser add-ons

uBlock Origin  AdGuard
EasyList

/EasyPrivacy

—the largest filter list project

filter rules

||a2z.com/sping?
||aan.amazon.com^$third-party
||activengage.com/overwatch/
||ad-shield.io^$third-party
||ad.aloodle.com^*
||addthis.com/at/
||addthis.com/live/
||addthiscdn.com/live/
||addthisedge.com/live/
||adtoany.com/menu/transparent.gif
||adfox.yandex.ru^*
||adlog.com.com^*
||ads-trk.vidible.tv^*
||affiliates.mgmmirage.com^*
||affiliates.minglematch.com^*
||affirm.com/api/v2/cookie_sent
...


EasyList
EasyPrivacy

—the largest filter list project

Brave Filter List Engineer
@fanboynz

filter rules
In 2000, a Disneyland ticket cost $41. It's just increased to $164.

Reddit, this is my child. I apologize for nothing.
all-posts.json
host: api.reddit.com
origin: reddit.com
type: AJAX
decision: allow
Reddit, this is my child. I apologize for nothing.
clicktrack.js
host: doubleclick.com
origin: reddit.com
type: script

decision:

||doubleclick.com/clicktrack.js
...
clicktrack.js
host: doubleclick.com
origin: reddit.com
type: script

decision: block

||doubleclick.com/clicktrack.js
...
Is blocking a complete solution?
Is blocking a complete solution?

→ Where can blocking go wrong?
analytics.js
host: google-analytics.com
origin: reddit.com
type: script
decision:

|||google-analytics.com/analytics.js
"filter rule match"
analytics.js
host: google-analytics.com
origin: reddit.com
type: script

decision: ✗ block

||google-analytics.com/analytics.js
...
analytics.js
host: google-analytics.com
origin: reddit.com
type: script

decision: block

filter rule match
google-analytics.com/analytics.js

```javascript
window.ga = {};
ga.track = function(kind, url, tag) {
  ...
};
```
...```javascript
window.ga = {};
ga.track = function(kind, url, tag) {
  ...
};
...```
google-analytics.com.analytics.js

```javascript
window.ga = {};
ga.track = function(kind, url, tag) {
  ...
}...

reddit-cdn.com/FrontPage.js

// Track page view:
ga.track('pageview', location.href, 'default-frontpage');
// Load all the posts:
...
google-analytics.com/analytics.js

```javascript
window.ga = {};
ga.track = function(kind, url, tag) {
  ...
};
```
window.ga = {?
  ga.track = function(red, url, tag) {
    ...
  }?
};

// Track page view:
ga.track(‘pageview’, location.href, ‘default-frontpage’);

// Load all the posts:
...
ReferenceError
google-analytics.com/analytics.js

window.ga = function() { ga.track = function(red, url, tag) { ... }; ...

reddit-cdn.com/FrontPage.js

// Track page view: ga.track('pageview', location.href, 'default-frontpage');
// Load all the posts: ...

ReferenceError
So you broke Web compatibility.

Is that okay?
So you broke Web compatibility. Is that okay? → Not really, no.
analytics.js
host: google-analytics.com
origin: reddit.com
type: script

Option #1:
Exception Rules
Option #1: Exception Rules

analytics.js
host: google-analytics.com
origin: reddit.com
type: script
Option #1: Exception Rules

decision:

analytics.js
host: google-analytics.com
origin: reddit.com
type: script

goalieanalytics.com/analytics.js, domain=reddit.com

exception rule match
Option #1: Exception Rules

analytics.js
host: google-analytics.com
origin: reddit.com
type: script
decision:

@@ || google-analytics.com/analytics.js, domain=reddit.com

exception rule match
Exception Rules
Exception Rules

Good: Repair Web compatibility
Exception Rules

Good: Repair Web compatibility
Good: Easy to write
Exception Rules

Good: Repair Web compatibility
Good: Easy to write

6,405 exception rules
(Brave, EasyList, EasyPrivacy, uBlock Origin)
Exception Rules

Bad: Allow privacy harm

Good: Repair Web compatibility
Good: Easy to write

6,405 exception rules
(Brave, EasyList, EasyPrivacy, uBlock Origin)
Option #2:
Resource Replacements
Option #2: Resource Replacements

google-analytics.com/analytics.js

```javascript
... window.ga = {}; ga.track = function(kind, url, tag) { ... }; ... 
```
fake_ga.js

```
window.ga = {
    ...
    track (_kind, _url, _tag) {
        // Do nothing.
    },
    ...
};
```

"resource replacement"

Option #2: Resource Replacements

google-analytics.com/analytics.js

```
window.ga = {};
ga.track = function(kind, url, tag) {
    ...
};
```
fake ga.js

window.ga = {
    track (_kind, _url, _tag) {
        // Do nothing.
    },
    ...
};

"resource replacement"

Option #2:
Resource Replacements

google-analytics.com/analytics.js

window.ga = {};
ga.track = function(kind, url, tag) {
    ...
};

reddit-cdn.com/FrontPage.js

// Track page view:
ga.track('pageview', location.href, 'default-frontpage');

// Load all the posts:
...
analytics.js

**host:** google-analytics.com

**origin:** reddit.com

**type:** script

```javascript
||google-analytics.com/analytics.js,redirect=fake Ga.js
```
analytics.js

host: google-analytics.com
origin: reddit.com
type: script

decision:

||google-analytics.com/analytics.js,redirect=fake_ga.js
...

resource replacement rule match
analytics.js

host: google-analytics.com

origin: reddit.com

type: script

decision: replace

||google-analytics.com/analytics.js,redirect=fake Ga.js


resource replacement rule match
Resource Replacements
| Good: | Repair Web compatibility |
Resource Replacements

Good: Repair Web compatibility
Good: Prevent privacy harm
# Resource Replacements

<table>
<thead>
<tr>
<th>Good:</th>
<th>Repair Web compatibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Good:</td>
<td>Prevent privacy harm</td>
</tr>
<tr>
<td>Bad:</td>
<td>Need significant time and expertise</td>
</tr>
</tbody>
</table>
Resource Replacements

Good: Repair Web compatibility
Good: Prevent privacy harm
Bad: Need significant time and expertise
Bad: High maintenance costs
Resource Replacements

Good: Repair Web compatibility
Good: Prevent privacy harm
Bad: Need significant time and expertise
Bad: High maintenance costs

27 resource replacements (Brave, uBlock Origin)
Resource Replacements

Good: Repair Web compatibility
Good: Prevent privacy harm
Bad: Need significant time and expertise
Bad: High maintenance costs

SugarCoat automates these!
The rest of this talk...

SugarCoat

privacy-harming scripts → privacy-respecting resource replacements
The rest of this talk...

SugarCoat

privacy-harming scripts

- window.localStorage
- document.cookies
- fetch
The rest of this talk...

SugarCoat

privacy-harming scripts

window.localStorage

document.cookies

fetch

$mockLocalStorage = ...;

$mockCookies = ...;

$mockFetch = ...;
The rest of this talk...

Q1: What behaviors of the scripts do we rewrite?
Q2: What do we replace those script parts with?
Q3: How do we locate those parts in script code?
Q4: How do we actually transform script code?
Q5: How do we know if it works?
Q1: What behaviors of the scripts do we rewrite?
Q1: What behaviors of the scripts do we rewrite?

→ We rewrite accesses to privacy-sensitive Web APIs.
Q1: What behaviors of the scripts do we rewrite?
→ We rewrite accesses to privacy-sensitive Web APIs.

```javascript
var x = 1 + 2;
```
Q1: What behaviors of the scripts do we rewrite?
→ We rewrite accesses to privacy-sensitive Web APIs.

```javascript
var x = 1 + 2;
var y = 'Hello ' + x;
```
Q1: What behaviors of the scripts do we rewrite?

→ We rewrite accesses to privacy-sensitive Web APIs.

```javascript
var x = 1 + 2;
var y = 'Hello ' + x;
Math.min(x, 4);
```
Q1: What behaviors of the scripts do we rewrite?
→ We rewrite accesses to privacy-sensitive Web APIs.

**Storage APIs**
document.cookie
localStorage
sessionStorage
Q1: What behaviors of the scripts do we rewrite?

→ We rewrite accesses to privacy-sensitive Web APIs.

**Storage APIs**
document.cookie
localStorage
sessionStorage

used to **persist** tracking identifiers
Q1: What behaviors of the scripts do we rewrite?
→ We rewrite accesses to privacy-sensitive Web APIs.

**Storage APIs**
document.cookie
localStorage
sessionStorage

**Network APIs**
fetch
XMLHttpRequest

<table>
<thead>
<tr>
<th>Storage APIs</th>
<th>Network APIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>document.cookie</td>
<td>fetch</td>
</tr>
<tr>
<td>localStorage</td>
<td>XMLHttpRequest</td>
</tr>
<tr>
<td>sessionStorage</td>
<td></td>
</tr>
</tbody>
</table>

These APIs are used to persist tracking identifiers.
Q1: What behaviors of the scripts do we rewrite?

→ We rewrite accesses to privacy-sensitive Web APIs.

**Storage APIs**
document.cookie
localStorage
sessionStorage

used to persist tracking identifiers

**Network APIs**
fetch
XMLHttpRequest

used to exfiltrate personal data
Q1: What behaviors of the scripts do we rewrite?

→ We rewrite accesses to privacy-sensitive Web APIs.

Storage APIs
- localStorage
- sessionStorage

used to persist
tracking identifiers

Fingerprinting APIs
- CanvasRenderingContext2D.prototype.getImageData
- navigator.userAgent
- navigator.hardwareConcurrency
- etc...

used to exfiltrate personal data
Q2: What do we replace those script parts with?
Q2: What do we replace those script parts with?

→ We redirect privacy-sensitive Web API accesses to mock implementations.
Q2: What do we replace those script parts with?

→ We redirect privacy-sensitive Web API accesses to mock implementations.

```javascript
const fetch = async (url, init = null) => {
  throw new TypeError('Failed to fetch');
};
exports.fetch = {
  value: fetch, writable: true,
  configurable: true, enumerable: true
};
```
Q2: What do we replace those script parts with?

→ We redirect privacy-sensitive Web API accesses to mock implementations.

```javascript
const fetch = async (url, init = null) => {
  throw new TypeError('Failed to fetch');
};
exports.fetch = {
  value: fetch, writable: true,
  configurable: true, enumerable: true
};
```

Mocks are write-once, and reusable between resource replacement scripts.
The rest of this talk...

Q1: What parts of the scripts do we rewrite?
Q2: What do we replace those script parts with?
Q3: How do we locate those parts in script code?
Q4: How do we actually transform script code?
Q5: How do we know if it works?
The rest of this talk...

Q1: What parts of the scripts do we rewrite?

Q2: What do we replace those script parts with?

Q3: How do we locate those parts in script code?

Q4: How do we actually transform script code?

Q5: How do we know if it works?
The rest of this talk...

Q1: What parts of the scripts do we rewrite?
Q2: What do we replace those script parts with?
Q3: How do we locate those parts in script code?
Q4: How do we actually transform script code?
Q5: How do we know if it works?
Q3: How do we locate those parts in script code?
→ Dynamic runtime instrumentation of page behavior.
Q3: How do we locate those parts in script code?
→ Dynamic runtime instrumentation of page behavior.

PageGraph-enabled build of the Brave Browser
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.

PageGraph-enabled build of the Brave Browser

https://github.com/brave/brave-browser/wiki/PageGraph
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.

---

PageGraph-enabled build of the Brave Browser

- Script actor
  - id=789,
  - type="classic",
  - url="jQuery.js",
  - src="..."

Network resource
url="https://foo.com/bar.jpg"
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.

PageGraph-enabled build of the Brave Browser

Script actor
id=789,
type="classic",
url="jQuery.js",
src="..."

Network resource
url="https://foo.com/bar.jpg"

DOM node
tag="img"
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.

PageGraph-enabled build of the Brave Browser

Script actor
- id=789,
- type="classic",
- url="jQuery.js",
- src="..."

Network resource
- url="https://foo.com/bar.jpg"

HTTP request

DOM insert

DOM node
- tag="img"
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.

Web API
name="Window#localStorage", type="attribute"
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.

**Script actor**
- id=123,
- type="classic",
- url="analytics.js",
- src="..."

**Script actor**
- id=456,
- type="classic",
- url="FrontPage.js",
- src="..."

**Web API access**
- type="attributeGet"

**Web API**
- name="Window#localStorage",
- type="attribute"
Q3: How do we locate those parts in script code?
→ Dynamic runtime instrumentation of page behavior.

Script actor
id=123,
type="classic",
url="analytics.js",
src="..."
stack frame
srcLocation=58,
frameIndex=1

Script actor
id=456,
type="classic",
url="FrontPage.js",
src="..."
stack frame
srcLocation=275,
frameIndex=0

Web API access
type="attributeGet"
Web API
name="Window#localStorage",
type="attribute"
Q3: How do we locate those parts in script code?

→ Dynamic runtime instrumentation of page behavior.

**Script actor**
- id=123,
- type="classic",
- url="analytics.js",
- src="..."

**Stack frame**
- srcLocation=58,
- frameIndex=1

**Script actor**
- id=456,
- type="classic",
- url="FrontPage.js",
- src="..."

**Stack frame**
- srcLocation=275,
- frameIndex=0

**Web API access**
- type="attributeGet"

**Web API**
- name="Window#localStorage",
- type="attribute"
Q3: How do we locate those parts in script code?
→ Ideally, we would do this statically!
The rest of this talk...

Q1: What parts of the scripts do we rewrite?

Q2: What do we replace those script parts with?

Q3: How do we locate those parts in script code?

Q4: How do we actually transform script code?

Q5: How do we know if it works?
The rest of this talk...

Q1: What parts of the scripts do we rewrite?
Q2: What do we replace those script parts with?
Q3: How do we locate those parts in script code?
Q4: How do we actually transform script code?
Q5: How do we know if it works?
Q4: How do we actually transform script code?
Q4: How do we actually transform script code?

APIs used:
- Document#cookies
- Window#fetch
- Window#localStorage

privacy-harming script
Q4: How do we actually transform script code?

APIs used:
- Document#cookies
- Window#fetch
- Window#localStorage

$mockLocalStorage = ...;
Q4: How do we actually transform script code?

APIs used:
- Document#cookies
- Window#fetch
- Window#localStorage

```
$mockFetch = ...;
$mockLocalStorage = ...;
```
Q4: How do we actually transform script code?

APIs used:
- Document#cookies
- Window#fetch
- Window#localStorage

privacy-harming script
Q4: How do we actually transform script code?

```php
$mockCookies = ...;
+$mockFetch = ...;
+$mockLocalStorage = ...;
```

APIs used:
- `Document#cookies`
- `Window#fetch`
- `Window#localStorage`
Q4: How do we actually transform script code?

```javascript
function getTrackingId() {

    let trackingId = window.localStorage.getItem("trackingId");
    if (!trackingId) {
        trackingId = Math.random();
        window.localStorage.setItem("trackingId", trackingId);
    }
    return trackingId;
}
```

JavaScript Global Scope:

```javascript
window.localStorage = /* browser-provided implementation */
```
Q4: How do we actually transform script code?

```javascript
... function getTrackingId() {
    let trackingId = window.localStorage.getItem("trackingId");
    if (!trackingId) {
        trackingId = Math.random();
        window.localStorage.setItem("trackingId", trackingId);
    }
    return trackingId;
}
...
```

JavaScript Global Scope:

```javascript
window.localStorage = /* browser-provided implementation */
...
Q4: How do we actually transform script code?

```javascript
function getTrackingId() {
    let trackingId = window.localStorage.getItem("trackingId");
    if (!trackingId) {
        trackingId = Math.random();
        window.localStorage.setItem("trackingId", trackingId);
    }
    return trackingId;
}
```

JavaScript Global Scope:

```javascript
window.localStorage = /* browser-provided implementation */
```
Q4: How do we actually transform script code?

```javascript
function getTrackingId() {
  try {
    $replace(window, "localStorage", $mockLocalStorage);
    let trackingId = window.localStorage.getItem("trackingId");
    if (!trackingId) {
      trackingId = Math.random();
      window.localStorage.setItem("trackingId", trackingId);
    }
    return trackingId;
  }
}

// JavaScript Global Scope:

window.localStorage =
/* browser-provided implementation */
```
Q4: How do we actually transform script code?

```javascript
function getTrackingId() {
  try {
    $replace(window, "localStorage", $mockLocalStorage);
    let trackingId = window.localStorage.getItem("trackingId");
    if (!trackingId) {
      trackingId = Math.random();
      window.localStorage.setItem("trackingId", trackingId);
    }
    return trackingId;
  }
}
```

JavaScript Global Scope:

```javascript
window.localStorage = $mockLocalStorage
```
Q4: How do we actually transform script code?

privacy-harming script

... function getTrackingId() {
    try {
        $replace(window, "localStorage", $mockLocalStorage);
        let trackingId = window.localStorage.getItem("trackingId");
        if (!trackingId) {
            trackingId = Math.random();
            window.localStorage.setItem("trackingId", trackingId);
        }
        return trackingId;
    }
}
...

JavaScript Global Scope:

... window.localStorage = $mockLocalStorage
...
Q4: How do we actually transform script code?

JavaScript Global Scope:

```javascript
window.localStorage = $mockLocalStorage
```

privacy-harming script

```javascript
function getTrackingId() {
    try {
        $replace(window, "localStorage", $mockLocalStorage);
        let trackingId = window.localStorage.getItem("trackingId");
        if (!trackingId) {
            trackingId = Math.random();
            window.localStorage.setItem("trackingId", trackingId);
        }
        return trackingId;
    }
}
```
Q4: How do we actually transform script code?

```javascript
function getTrackingId() {
    try {
        $replace(window, "localStorage", $mockLocalStorage);
        let trackingId = window.localStorage.getItem("trackingId");
        if (!trackingId) {
            trackingId = Math.random();
            window.localStorage.setItem("trackingId", trackingId);
        }
        return trackingId;
    } finally {
        $restore(window, "localStorage");
    }
}
```

JavaScript Global Scope:

```javascript
window.localStorage = $mockLocalStorage
```
Q4: How do we actually transform script code?

```javascript
function getTrackingId() {
  try {
    $replace(window, "localStorage", $mockLocalStorage);
    let trackingId = window.localStorage.getItem("trackingId");
    if (!trackingId) {
      trackingId = Math.random();
      window.localStorage.setItem("trackingId", trackingId);
    }
    return trackingId;
  } finally {
    $restore(window, "localStorage");
  }
}
```

JavaScript Global Scope:
```
window.localStorage = /* browser-provided implementation */
```
Q4: How do we actually transform script code?

```javascript
function getTrackingId() {
    try {
        $replace(window, "localStorage", $mockLocalStorage);
        let trackingId = window.localStorage.getItem("trackingId");
        if (!trackingId) {
            trackingId = Math.random();
            window.localStorage.setItem("trackingId", trackingId);
        }
        return trackingId;
    } finally {
        $restore(window, "localStorage");
    }
}
...
The rest of this talk...

Q1: What parts of the scripts do we rewrite?
Q2: What do we replace those script parts with?
Q3: How do we locate those parts in script code?
Q4: How do we actually transform script code?
Q5: How do we know if it works?
# Privacy Evaluation: Script-Level

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Rewritten</th>
</tr>
</thead>
<tbody>
<tr>
<td># Storage Calls (Mean)</td>
<td>77</td>
<td>0</td>
</tr>
<tr>
<td># Storage Calls (Median)</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td># Storage Calls (Total)</td>
<td>130,494</td>
<td>0</td>
</tr>
<tr>
<td># Network Calls (Mean)</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td># Network Calls (Median)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td># Network Calls (Total)</td>
<td>9,095</td>
<td>0</td>
</tr>
</tbody>
</table>
Privacy Evaluation: Page-Level

Legend
- rewritten (network)
- default (network)
- rewritten (storage)
- default (storage)
Compatibility Evaluation

6 human evaluators

50 Web sites evaluated
(drawn from https://github.com/easylist/easylist/issues)

2 evaluators per site

3 conditions
(“default”, “blocked”, “rewritten”)

1 minute per site, per condition
Compatibility Evaluation

Rating Scale:

1 - no perceptible difference between the default and current conditions

2 - browsing experience altered, but evaluator still able to perform the same tasks as in the default condition

3 - evaluator not able to perform the same tasks as in the default condition
# Compatibility Evaluation

<table>
<thead>
<tr>
<th>Measure</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td># evaluators</td>
<td>6</td>
</tr>
<tr>
<td># Web sites evaluated</td>
<td>50</td>
</tr>
<tr>
<td>% agreement</td>
<td>90%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Measure</th>
<th>Mean</th>
<th>Median</th>
</tr>
</thead>
<tbody>
<tr>
<td>When blocking</td>
<td>2.86</td>
<td>3</td>
</tr>
<tr>
<td>With SugarCoat</td>
<td>1.03</td>
<td>1</td>
</tr>
</tbody>
</table>
SugarCoat

privacy-harming scripts → privacy-respecting resource replacements

already deployed in production!
Summary

- SugarCoat resources are shared and open
- Deployed in Brave today
- Used by 50m people daily
- UCSD <-> Brave collaboration

Peter Snyder
Senior Director of Privacy
pes@brave.com
www.peteresnyder.com