#LocoMocoSec 2024

How blocking third-party cookies can fix the web's security model

Artur Janc, David Dworken
Google Information Security Engineering

Background: A simplified model of web security

Three broad classes of security problems in web applications:

1. *(lack of)* **Encryption**: Easy to build an application without encryption-in-transit
   - Vulnerabilities: Use of HTTP; mixed content/scripting; non-Secure cookies; PKI concerns
2. **Injections**: Core building blocks (HTML, URLs, JS) allow mixing code & data
   - Vulnerabilities: Various flavors of XSS; prototype pollution; DOM clobbering
3. *(lack of)* **Isolation**: Authenticated interactions with any cross-origin endpoint
   - Vulnerabilities: Cross-site request forgery (CSRF); clickjacking; XS-Search; XS-Leaks; XSSI

Most client-side web application vulnerabilities can be traced back to one of these root causes.

Background: A simplified model of web security

Three broad classes of security problems in web applications:

1. *(lack of)* **Encryption**: Easy to build an application without encryption-in-transit
   - Vulnerabilities: Use of HTTP; mixed content/scripting; non-Secure cookies; PKI concerns
2. **Injections**: Core building blocks (HTML, URLs, JS) allow mixing code & data
   - Vulnerabilities: Various flavors of XSS; prototype pollution; DOM clobbering

(lack of) **Isolation**: Authenticated interactions with any cross-origin endpoint
   - Vulnerabilities: Cross-site request forgery (CSRF); clickjacking; XS-Search; XS-Leaks; XSSI

Most client-side web application vulnerabilities can be traced back to one of these root causes.

The root cause of many of the web's isolation problems lies in its cookie model.

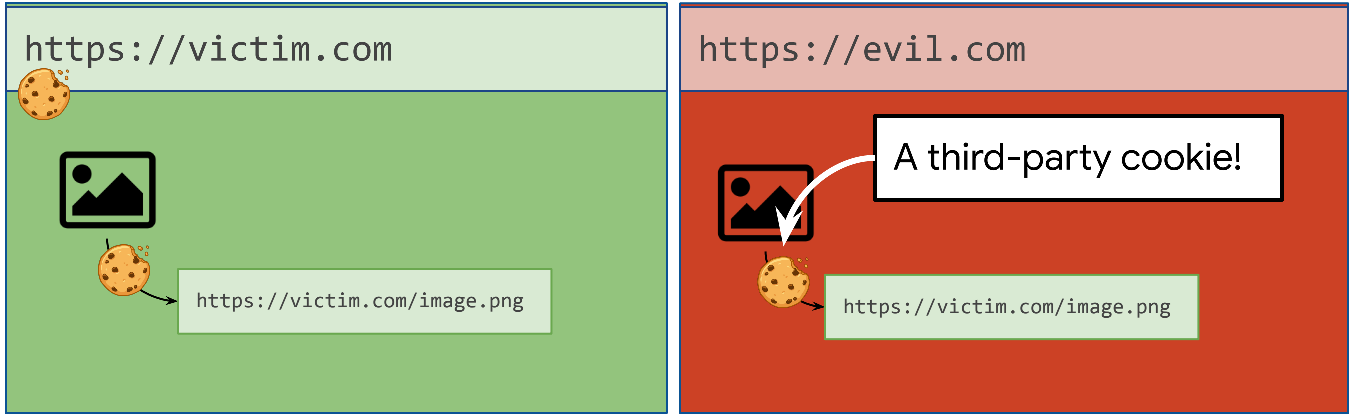
Cookies, in one slide

```
Set-Cookie: NAME=value; domain=.example.org; path=/; Secure;
```

A simple client-side store of information (commonly, authentication tokens) for a host or domain.

- Cookie attributes: `path`, `domain`, `expires`, `max-age`, `Secure`, `HttpOnly`
- `SameSite` attribute
  - `None`
  - `Lax`
  - `Strict`
- Cookie prefixes
  - `__Secure`
  - `__Host`

Ambient authority: In the original cookie model, once set, the cookie is always attached on requests to matching destinations, regardless of which site initiates the request.



victim.com server
handling a request for `/image.png`

Corporate needs you to find the differences between this picture and this picture.
They're the same p[illegible]e.

A few *completely safe* code examples

OUR WEBSITE:

```
<form action="/transfer">
  <input name="target" value="mkwst" />
  <input name="amount" value="10" />
```

form submission

```
<button onclick="deleteAccount()">
  Delete account</button>
```

clickable button

```
w("Content-Type: text/javascript")
w("var data = {'user':'${name}'}")
```

API endpoint

```
if search_result:
  log_to_db(search_query)
  return search_result
```

search functionality

A few ~~completely safe~~ code examples

OUR WEBSITE:

```
<form action="/transfer">
  <input name="target" value="m
  <input name="amount" value="10" />
```

CSRF

```
<button onclick="del
  Delete account</bu
```

clickjacking

```
w("Content-Type: text/javascri
w("var data = {'user':'${name
```

XSSI

```
if search_result:
  log_to_db(s
  return search_result
```

XS-Search / XS-Leak

EVIL.COM:

```
<form action="//victim/transfer">
<input name="target" value="bozo" />
<input name="amount" value="1000" />
```

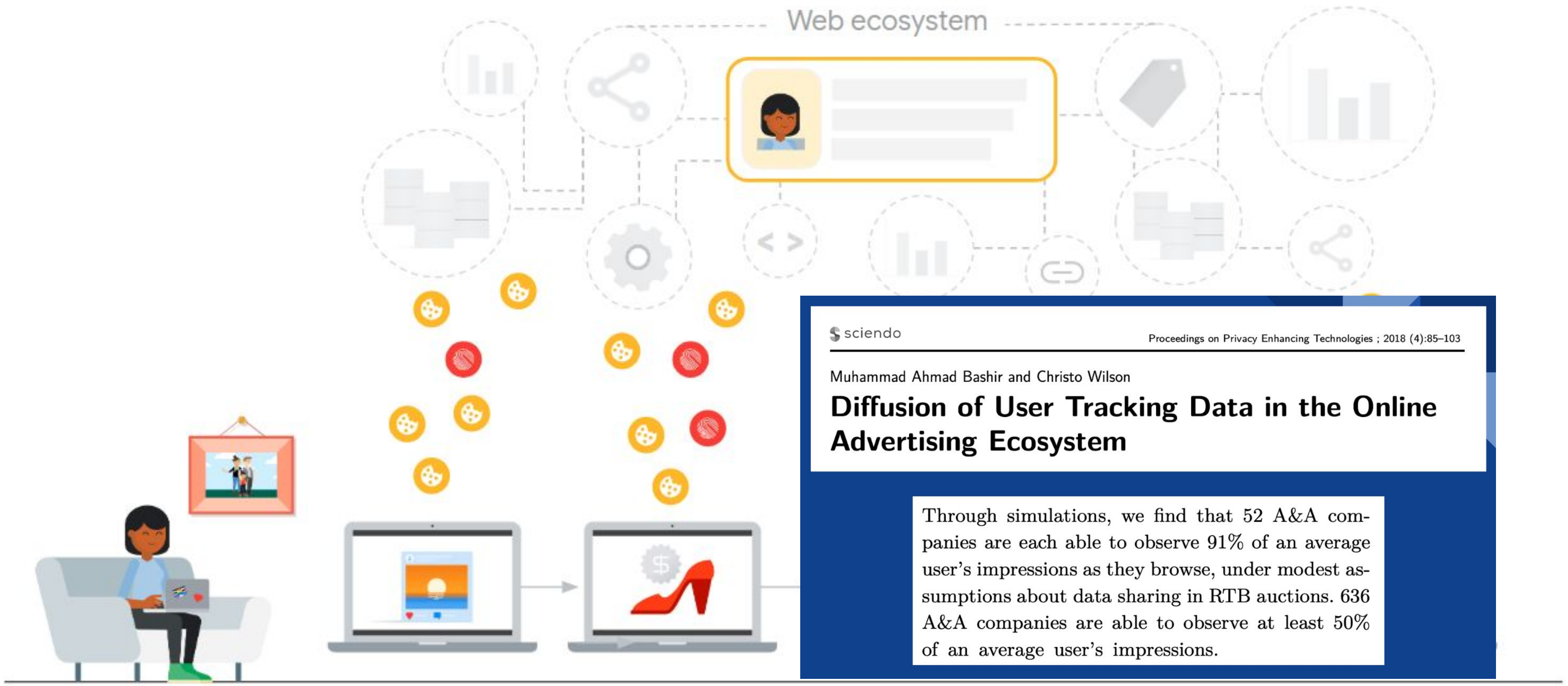
```
<iframe src="//victim/settings"
    style="opacity: 0"></iframe>
```

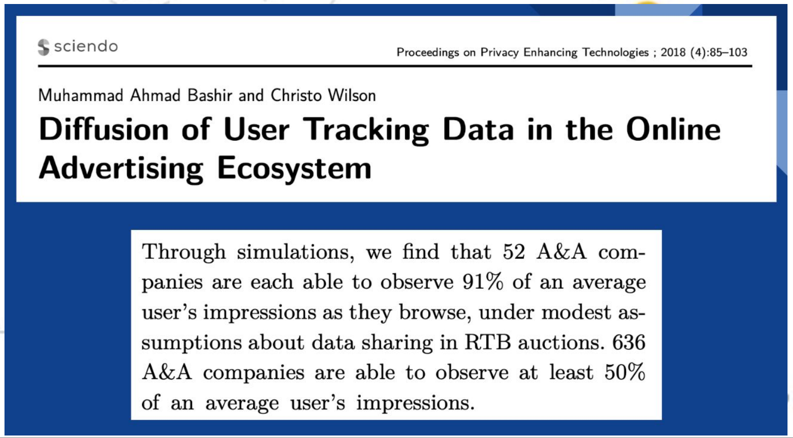
```
<script src="//victim/json" />
<script>alert(data)</script>
```

```
<script>t=performance.now()</script>
<img src="//victim/search?q=secret"
  onerror="t2=performance.now()" />
```

Addressing this in the web platform would fundamentally improve security.

Browser efforts to limit third-party cookies




sciendo

Proceedings on Privacy Enhancing Technologies ; 2018 (4):85–103

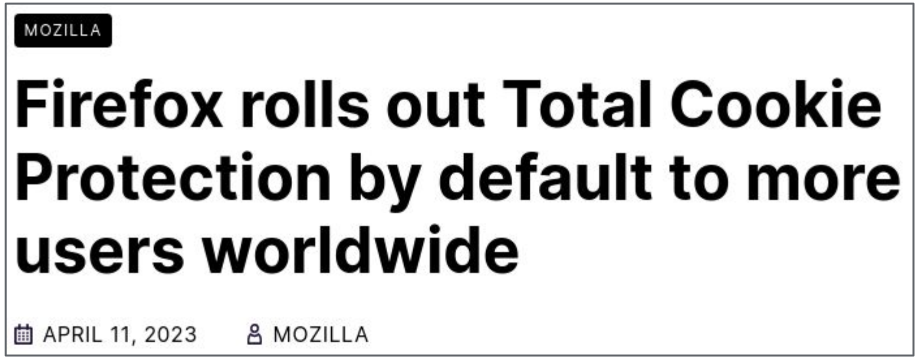
Muhammad Ahmad Bashir and Christo Wilson

Diffusion of User Tracking Data in the Online Advertising Ecosystem

Through simulations, we find that 52 A&A companies are each able to observe 91% of an average user's impressions as they browse, under modest assumptions about data sharing in RTB auctions. 636 A&A companies are able to observe at least 50% of an average user's impressions.

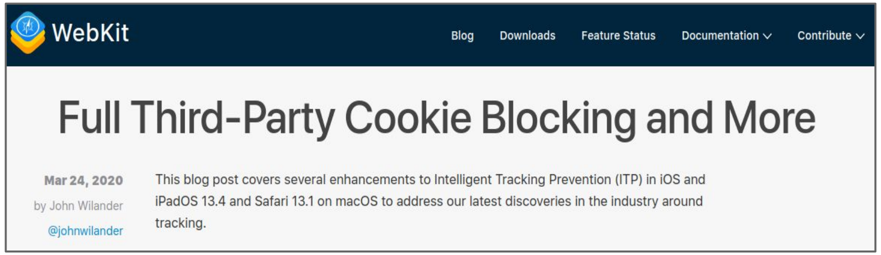
Web ecosystem

All browsers committed to restricting third-party cookies


MOZILLA

Firefox rolls out Total Cookie Protection by default to more users worldwide

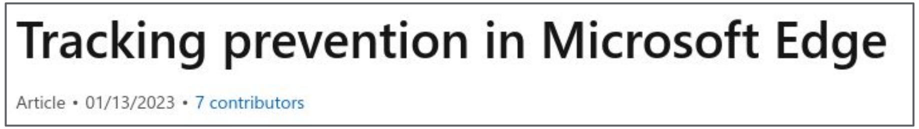
APRIL 11, 2023 MOZILLA


WebKit Blog Downloads Feature Status Documentation Contribute

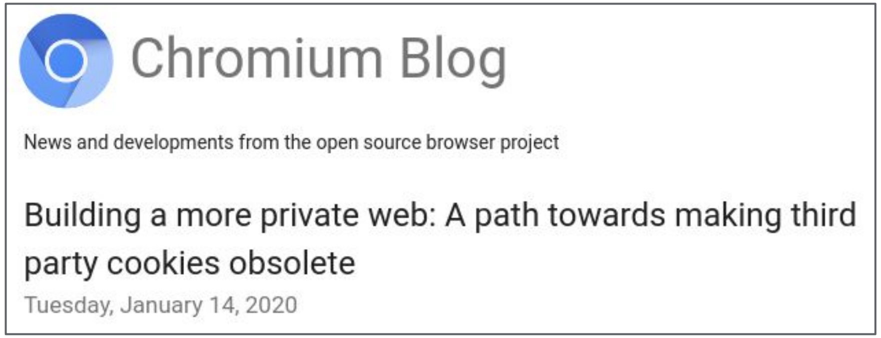
Full Third-Party Cookie Blocking and More

Mar 24, 2020
by John Wilander
@johnwilander

This blog post covers several enhancements to Intelligent Tracking Prevention (ITP) in iOS and iPadOS 13.4 and Safari 13.1 on macOS to address our latest discoveries in the industry around tracking.


Tracking prevention in Microsoft Edge

Article • 01/13/2023 • 7 contributors


Chromium Blog

News and developments from the open source browser project

Building a more private web: A path towards making third party cookies obsolete

Tuesday, January 14, 2020

What's the problem with just completely disabling third-party cookies?

Literature review: Value of a cookie estimates

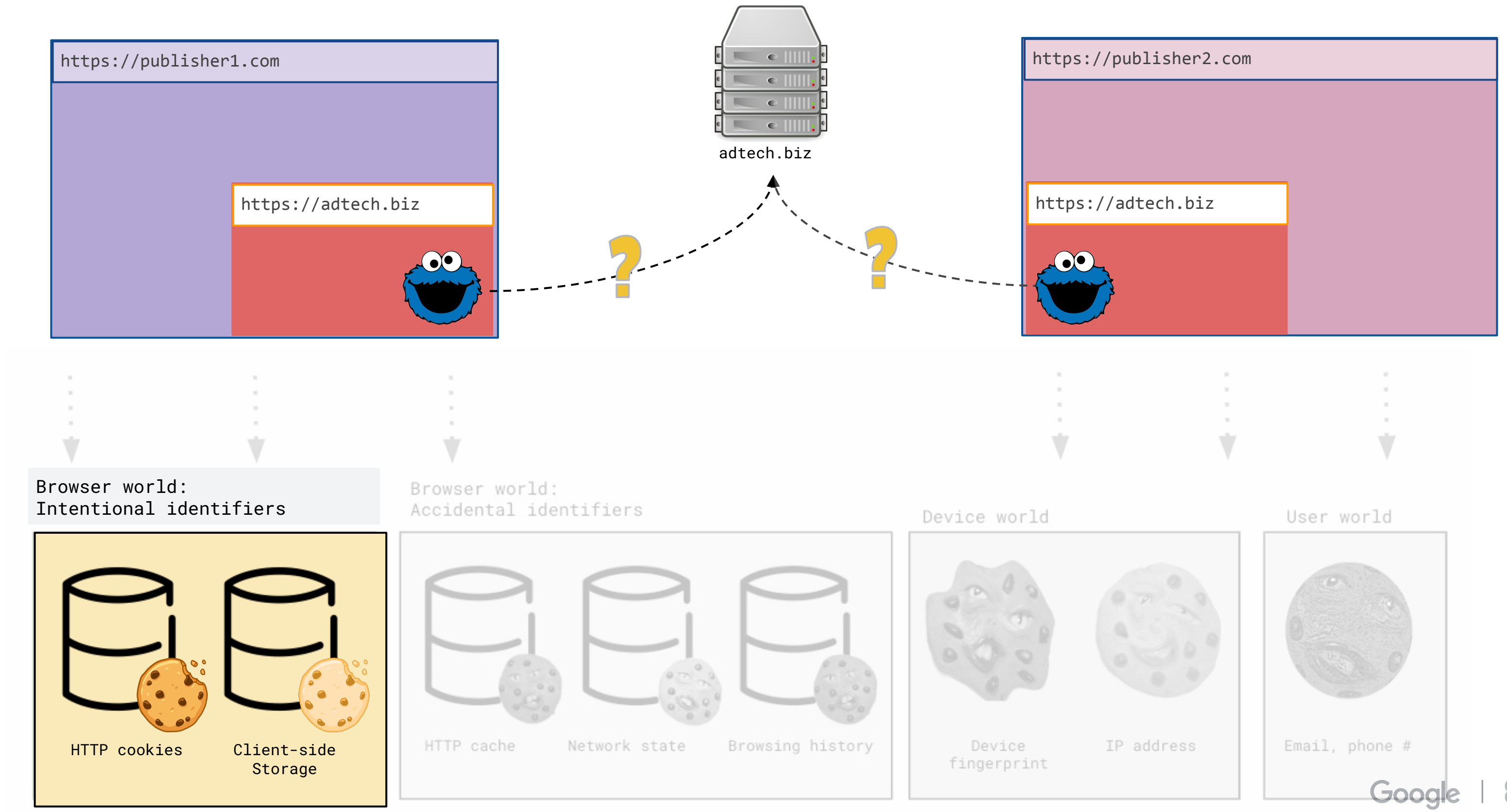
Study	Data	Method	Outcome	Estimate
Goldfarb & Tucker (2011)	9,596 ad campaigns	Natural experiment (e-Privacy Directive)	User purchase intent (surveyed)	65%
Beales & Eisenach (2014)	2 ad exchanges + "significantly diversified [company] operating multiple Internet-based enterprises"	Regression adjustment	Exchange/ publisher price	>66%†
Johnson, Shriver, & Du (2020)	Ad exchange (10K+ advertisers, publishers)	Regression adjustment	Exchange price+ Publisher, SSP, DSP, Advertiser	52%
Marotta, Abhishek, & Acquisti (2019)	large, multi-site publisher	Augmented inverse probability weighting	Publisher revenue	4%
Google (2019) (Ravichandran & Korula)	Google top 500 publishers	Experiment	Publisher revenue	52%
UK CMA Report (2020)	Google study's UK users	Experiment +subsampling + imputation	Publisher revenue	70% (Upper bound)

Notes: Value estimates measure loss in e.g. price without a cookie. Industry studies in grey. †Marginal effect estimates for new cookie (Figure A-1).

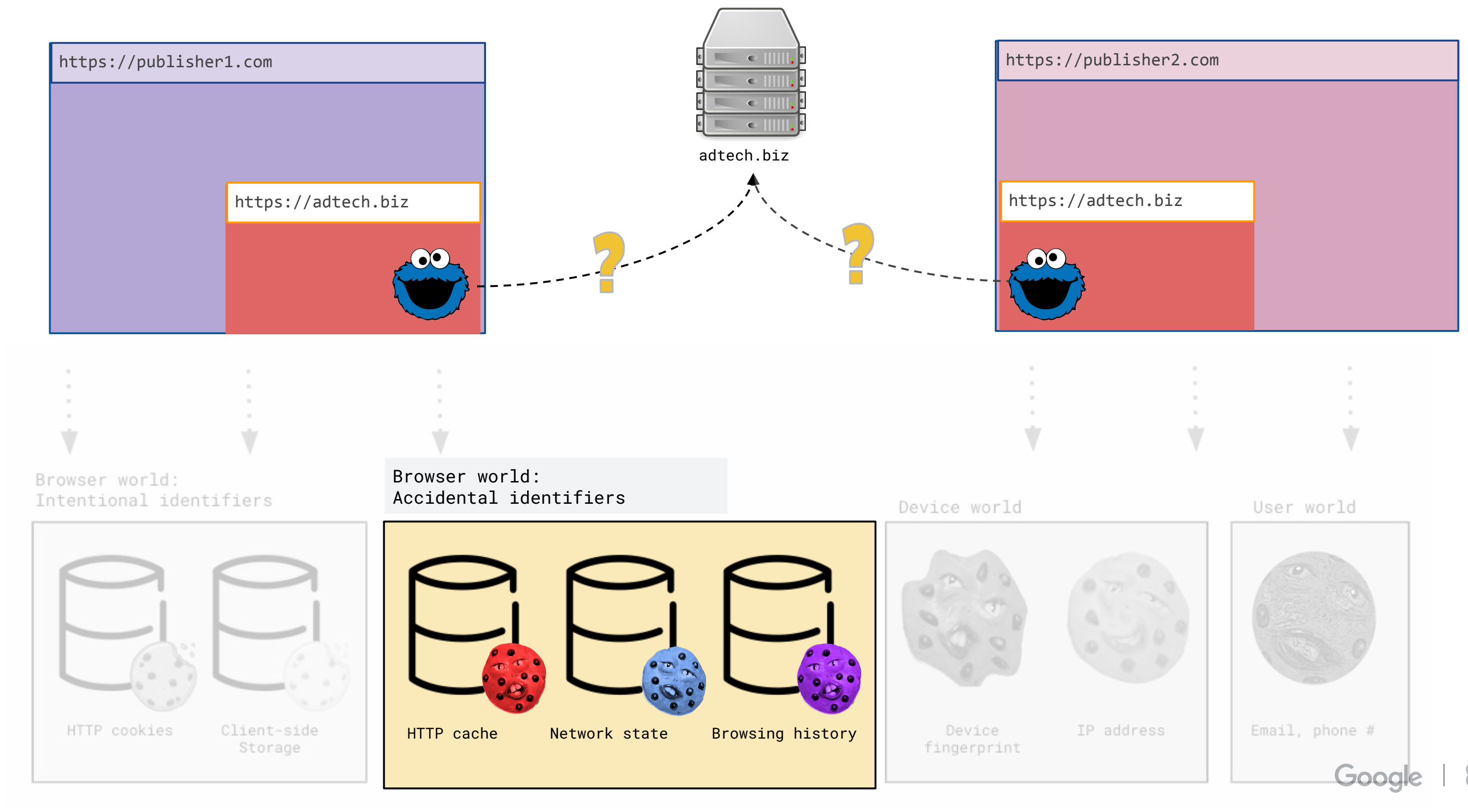
Studies:

Goldfarb, A. & Tucker, C. (2011). Privacy regulation and online advertising. *Management Science*.
Beales, J. H. & Eisenach, J. A. (2014). An empirical analysis of the value of information sharing in the market for online content. Technical report, Navigant Economics.
Johnson, G., Shriver, S., & Du, S. (2020) Consumer privacy choice in online advertising: Who opts out and at what cost to industry? *Marketing Science*.
Marotta, V., Abhishek, V., & Acquisti, A. (2019). Online tracking and publishers' revenues: An empirical analysis. Working paper.
Ravichandran, D., & Korula, N. (Google 2019) "Effect of disabling third-party cookies on publisher revenue" (Original blog post here)

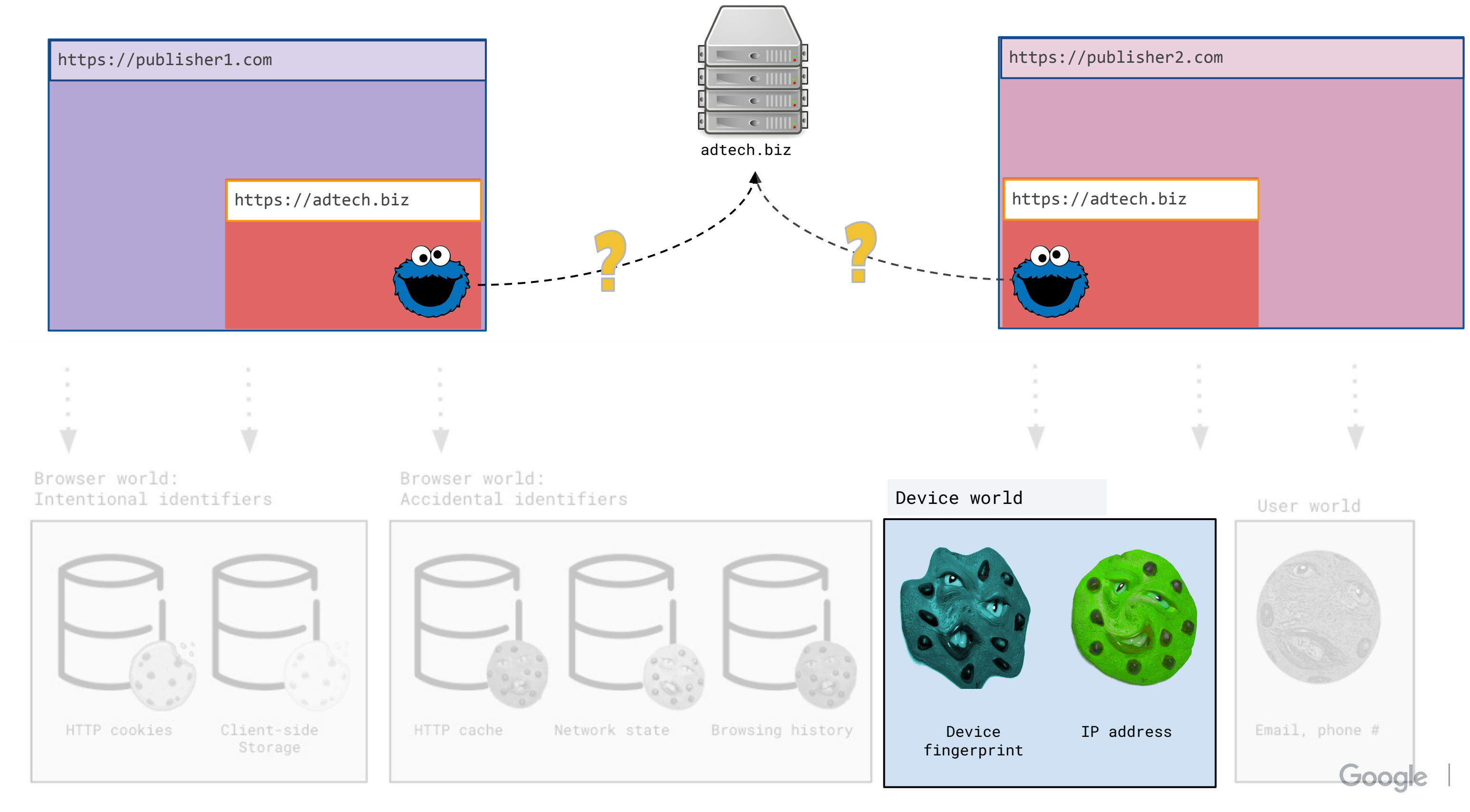
Fantastic identifiers and where to find them



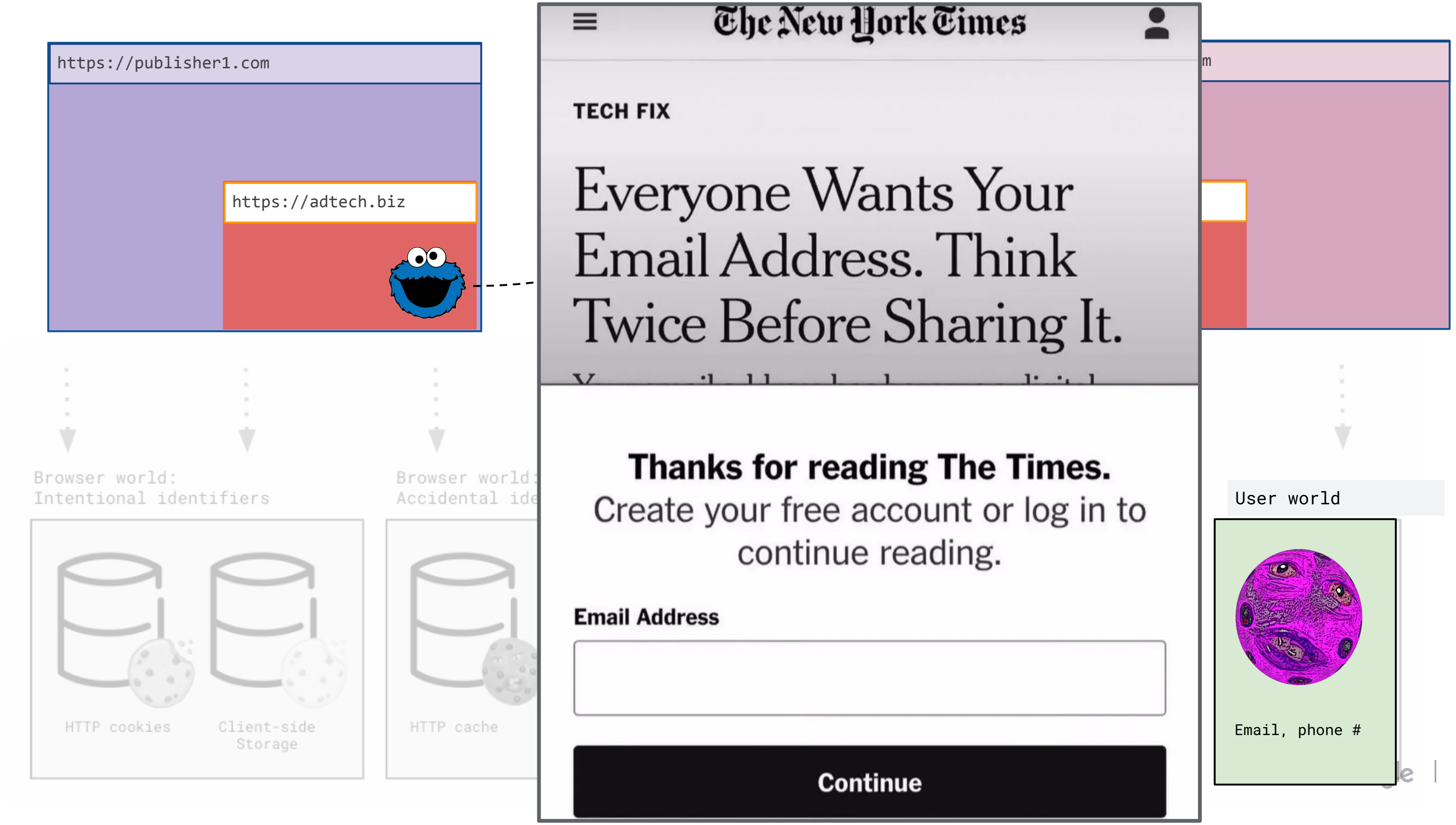
Fantastic identifiers and where to find them



Fantastic identifiers and where to find them



Fantastic identifiers and where to find them



The New York Times

TECH FIX

Everyone Wants Your Email Address. Think Twice Before Sharing It.

Thanks for reading The Times.
Create your free account or log in to continue reading.

Email Address

Continue

Removing third-party cookies from the web in 3 ~~easy~~ steps

1. **Limit the availability of alternative tracking mechanisms**
    - If trackers move to non-cookie-based alternatives, the result would be net negative for privacy. We need to prevent this from happening.

2. **Build new APIs** to replace legitimate use cases of third-party cookies
    - Ads functionality with protections against cross-site tracking
    - Account for every reasonable use of third-party cookies

3. **Actually restrict third-party cookies**
    - ... but provide escape hatches in case things break for users

This requires fundamental changes to the web platform, which can be a security win if we pay attention to the details.

Non-advertising major uses of third-party cookies

Identity Federation	● Many websites use identity federation (e.g. "Login with [Provider]") in a way that requires third-party cookies
Anti-fraud	● Combating fraud online can often benefit from using third-party cookies to better analyze behavior across sites (e.g. CAPTCHAs)
User-Content Serving	● Several classical solutions* for securely serving untrusted content rely on sandbox domains (e.g. googleusercontent.com) which can require third-party cookies for authentication
Many More	● Many more usages of third-party cookies including payments flows (3-D Secure), cross-site CORS requests, website analytics, and more

*goo.gle/modern-user-content-serving

Privacy goal: Robustly protect users from cross-site tracking using cookies or alternative web-based tracking mechanisms

Security goal: Build fundamental isolation boundaries that protect web services from common vulnerabilities

How these changes help web security

Third-party cookies are the original sin of the internet

OUR WEBSITE:

```
<form action="/transfer">
  <input name="target" value="mk
  <input name="amount" value="10" />
```

CSRF

```
<button onclick="del
  Delete account</bu
```

clickjacking

```
w("Content-Type: text/javascri
w("var data = {'user':'${name
```

XSSI

```
if search_result:
  log_to_db(s
  return search_result
```

XS-Search / XS-Leak

EVIL.COM:

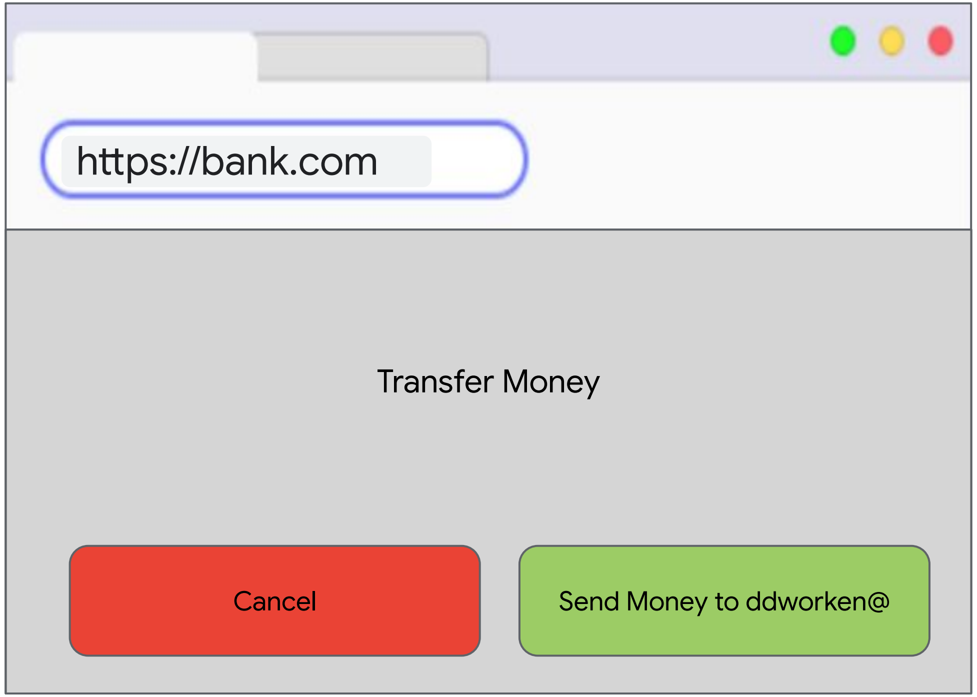
```
<form action="//victim/transfer">
<input name="target" value="urs" />
<input name="amount" value="1000" />
```

```
<iframe src="//victim/settings"
    style="opacity: 0"></iframe>
```

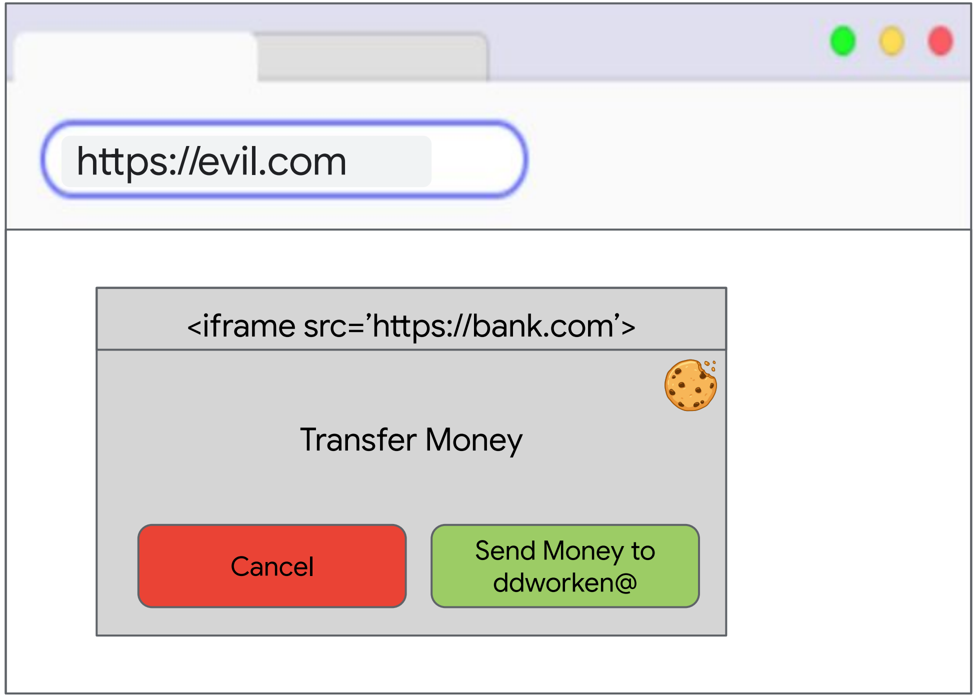
```
<script src="//victim/json" />
<script>alert(data)</script>
```

```
<script>t=performance.now()</script>
<img src="//victim/search?q=secret"
  onerror="t2=performance.now()" />
```

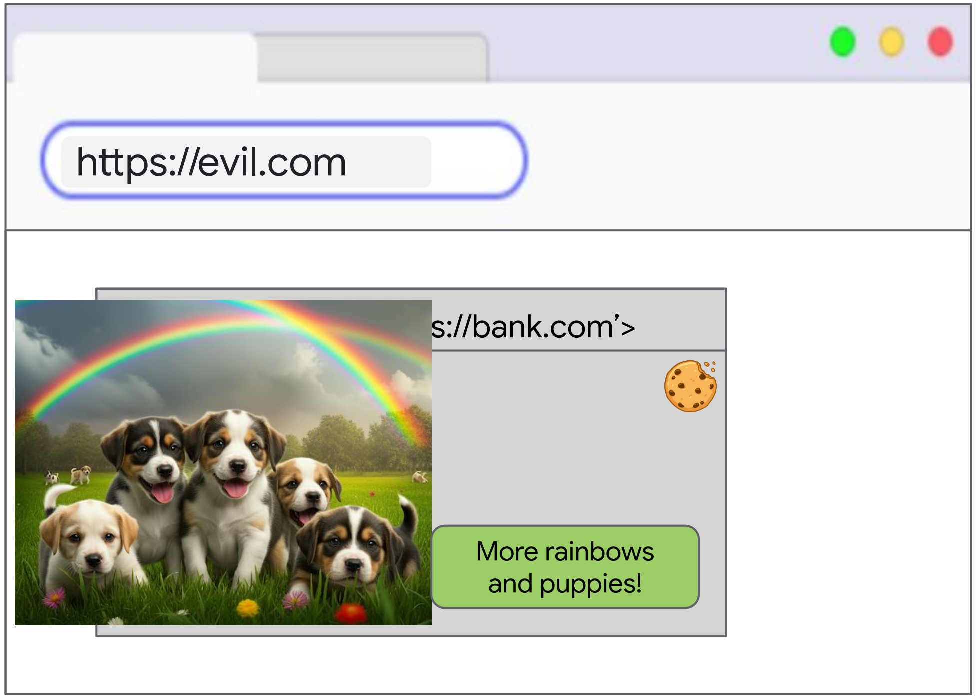
How 3PCD fixes clickjacking



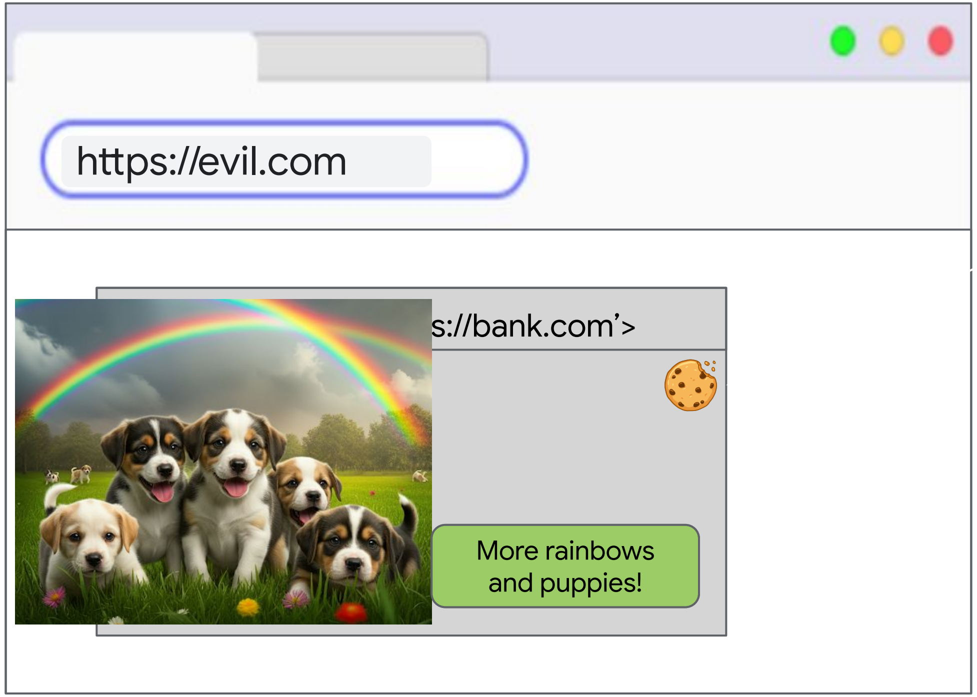
How 3PCD fixes clickjacking



How 3PCD fixes clickjacking

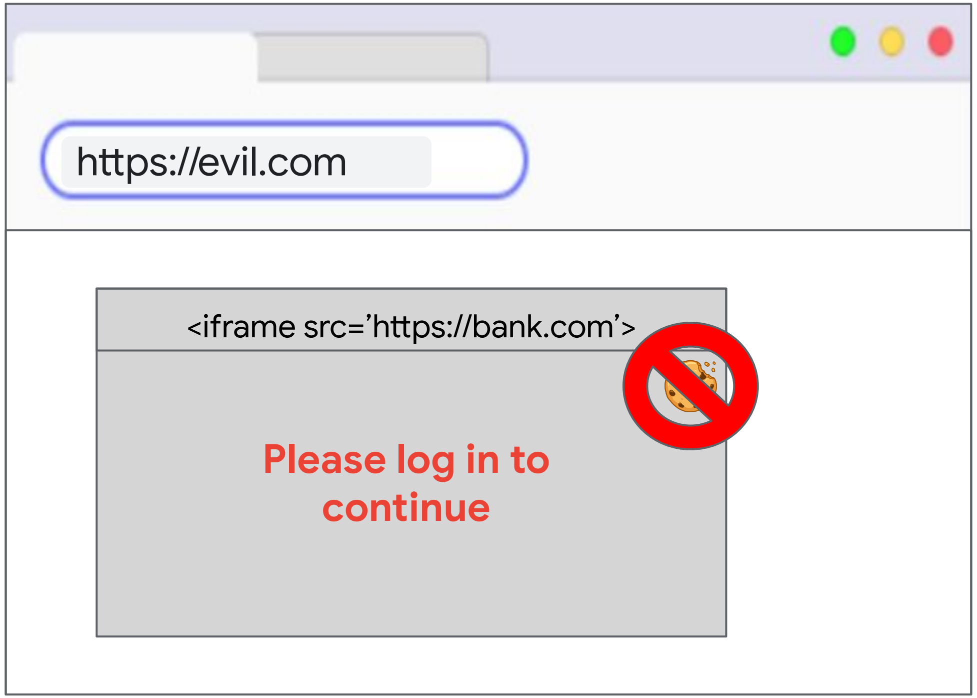


How 3PCD fixes clickjacking



Since bank.com and evil.com are cross-site, this is a third-party cookie

How 3PCD fixes clickjacking



Without third-party cookies, the iframe is unauthenticated, and thus clickjacking is fixed!

Fixing individual vulnerabilities with tools like X-Frame-Options and Cross-Origin-Resource-Policy

Fixing clickjacking, XSRF, and XS-Leaks by deprecating third-party cookies

Interlude: "Accidental" security benefits of cross-site tracking protections

3PCD is about more than just third-party cookies

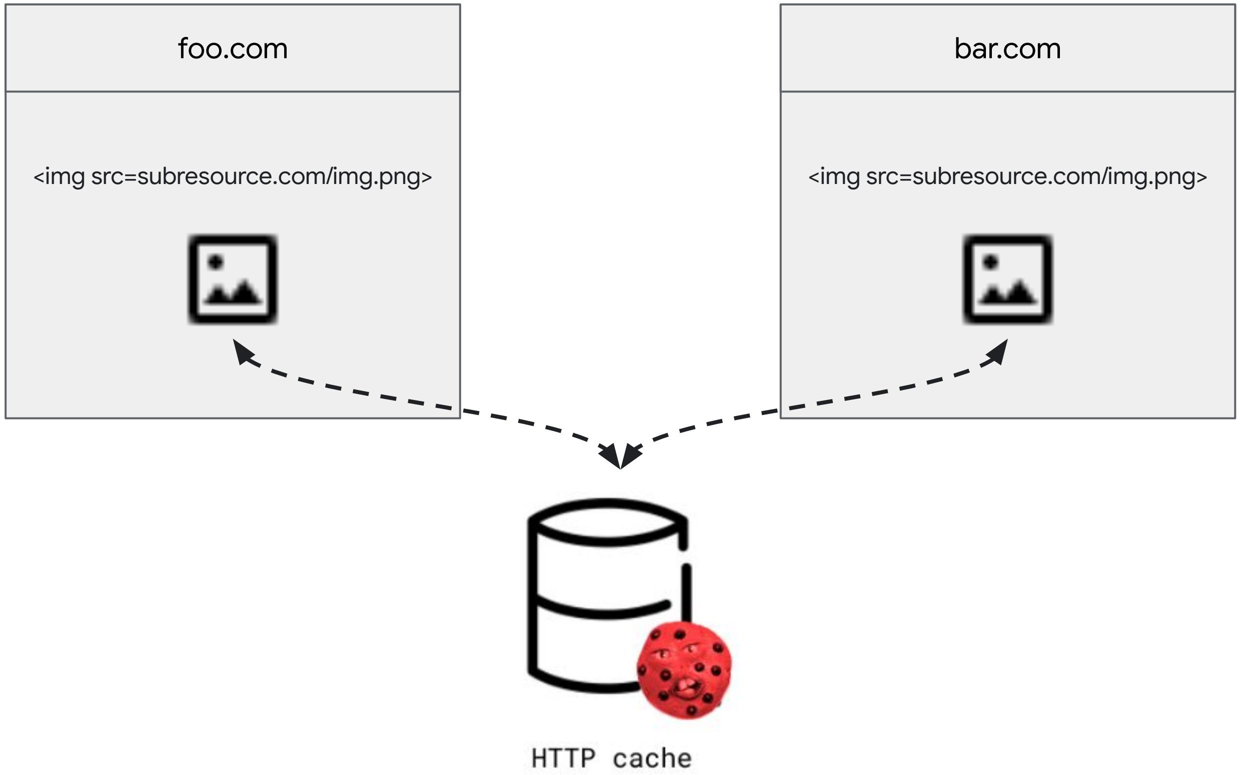
Deprecating third-party cookies

Eliminating other types of cross-site storage (`localStorage`, workers, `IndexedDB`)

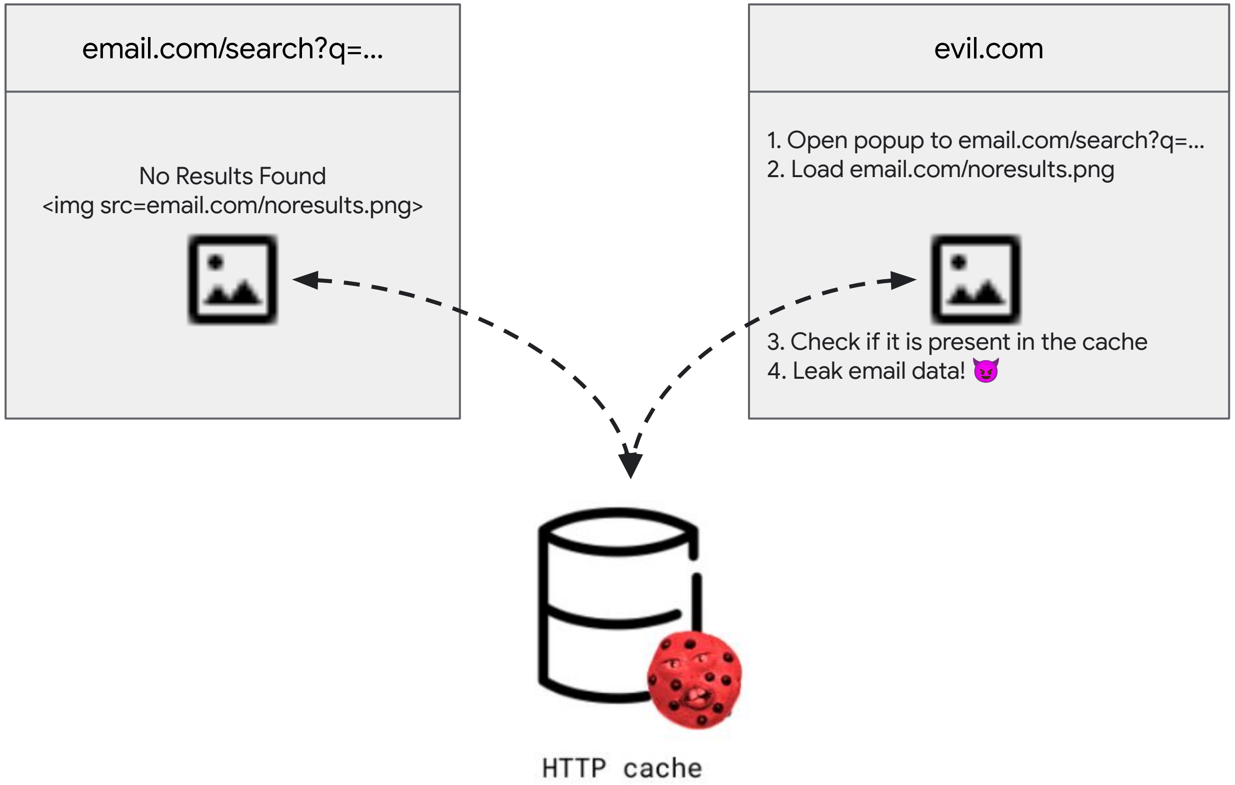
Partitioning global state (HTTP cache, network state, etc.)

Robust security and privacy isolation

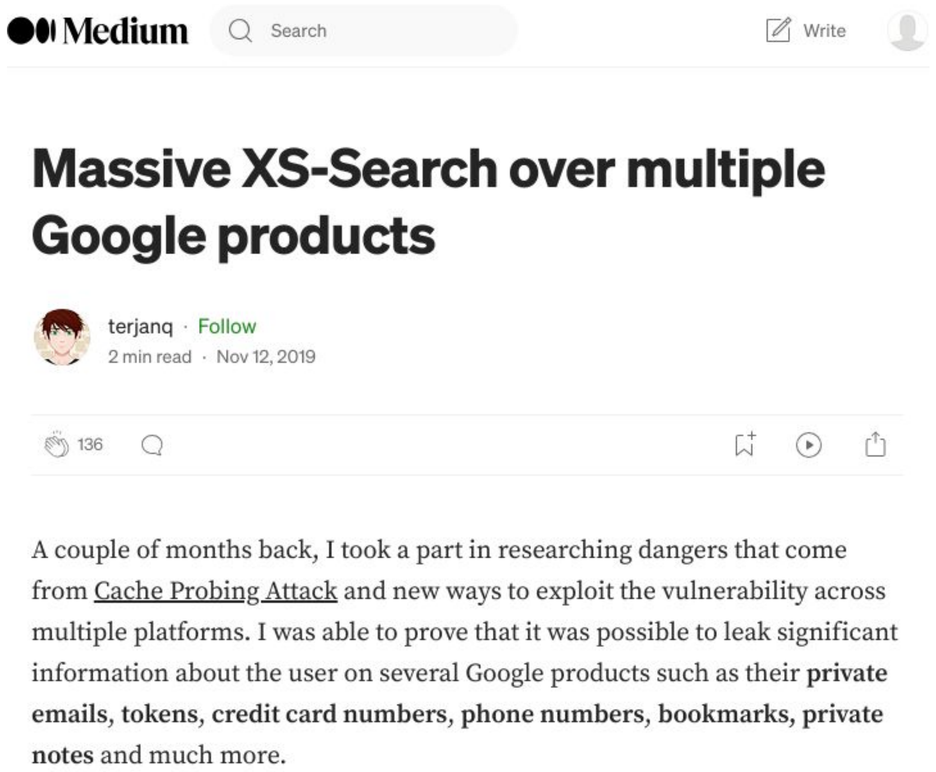
HTTP Cache Partitioning


foo.com
<img src=subresource.com/img.png>
bar.com
<img src=subresource.com/img.png>
HTTP cache

HTTP Cache and XS-Leaks


email.com/search?q=...
No Results Found
<img src=email.com/noresults.png>
evil.com
1. Open popup to email.com/search?q=...
2. Load email.com/noresults.png
3. Check if it is present in the cache
4. Leak email data! 😈
HTTP cache

HTTP Cache and XS-Leaks



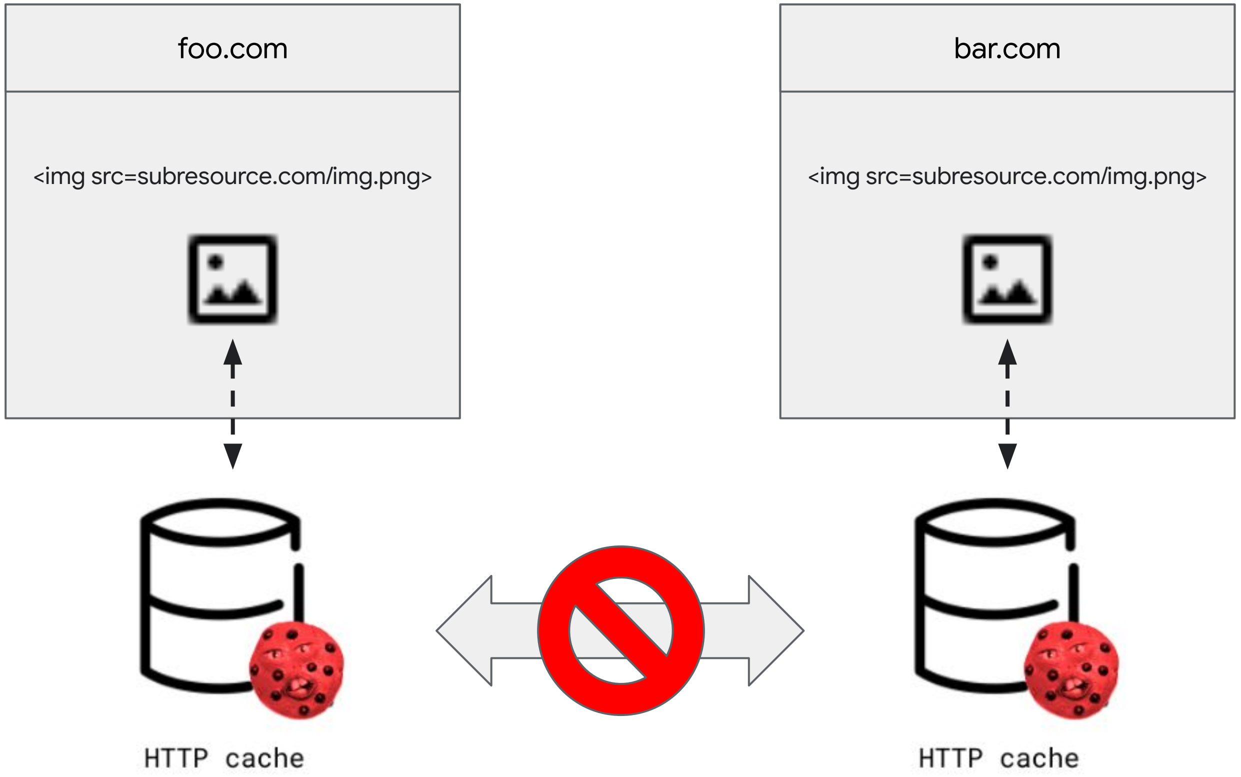
Massive XS-Search over multiple Google products

terjanq · Follow
2 min read · Nov 12, 2019

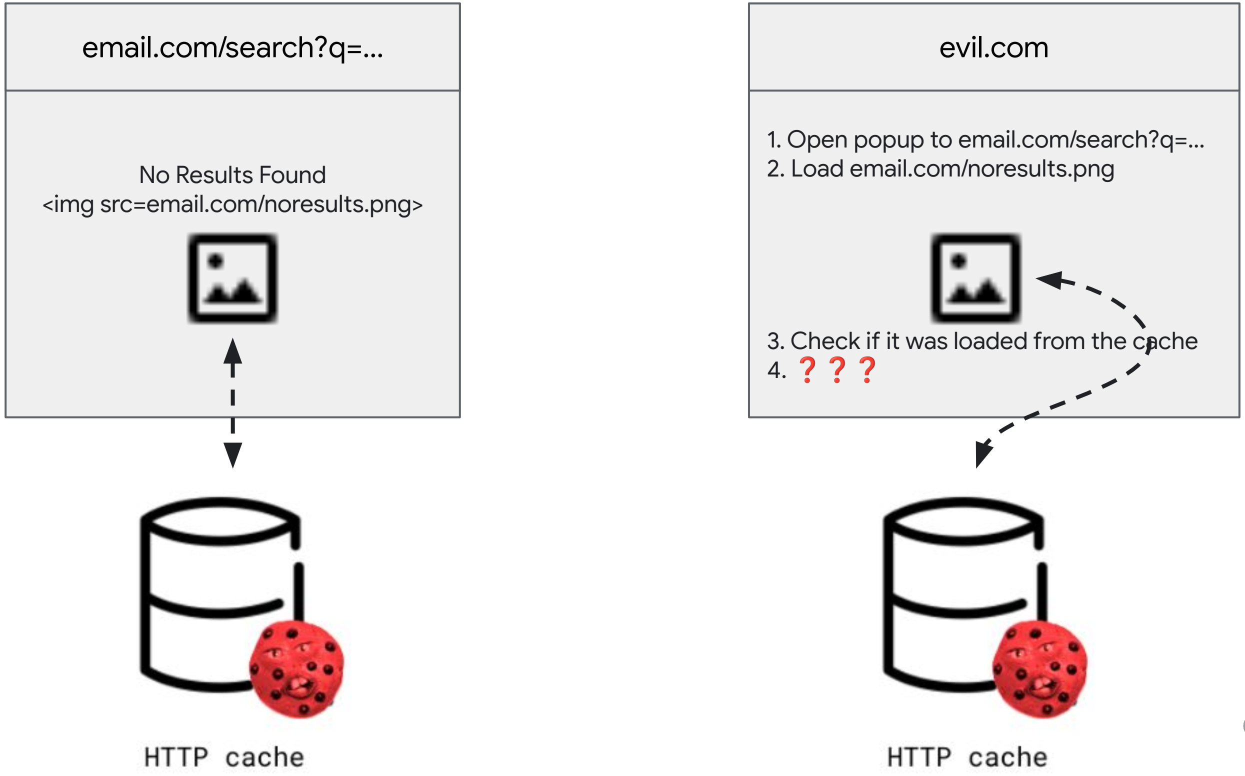
136

A couple of months back, I took a part in researching dangers that come from Cache Probing Attack and new ways to exploit the vulnerability across multiple platforms. I was able to prove that it was possible to leak significant information about the user on several Google products such as their **private emails, tokens, credit card numbers, phone numbers, bookmarks, private notes** and much more.

HTTP Cache Partitioning


foo.com
<img src=subresource.com/img.png>
HTTP cache
bar.com
<img src=subresource.com/img.png>
HTTP cache

HTTP Cache Partitioning

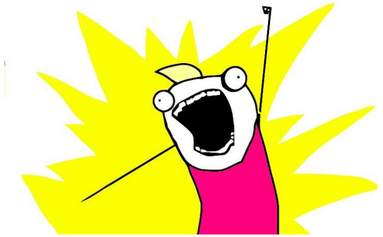

email.com/search?q=...
No Results Found
<img src=email.com/noresults.png>
HTTP cache
evil.com
1. Open popup to email.com/search?q=...
2. Load email.com/noresults.png
3. Check if it was loaded from the cache
4. ? ? ?
HTTP cache

Fixing individual vulnerabilities with tools like X-Frame-Options and Cross-Origin-Resource-Policy

Fixing clickjacking, XSRF, and XS-Leaks by deprecating third-party cookies

Fixing even more vulnerabilities by partitioning all global state

Partition all the things!



Network-state partitioning

Browsers contain all kinds of shared state in the network stack:

- Socket pools, DNS cache, TLS resumption, HSTS, etc

Partition it so that it can't be used for covert tracking

Fixes XS-Leaks that rely on this shared state

Client-side state partitioning

Sites can store state in the client-side via **localStorage** (and other mechanisms)

Partition it so that it can't be used as a cross-site cookie replacement

Fixes vulnerabilities that are enabled by client-side auth

:visited partitioning

Links are colored based on browser history

- Non-visited link
- Visited link

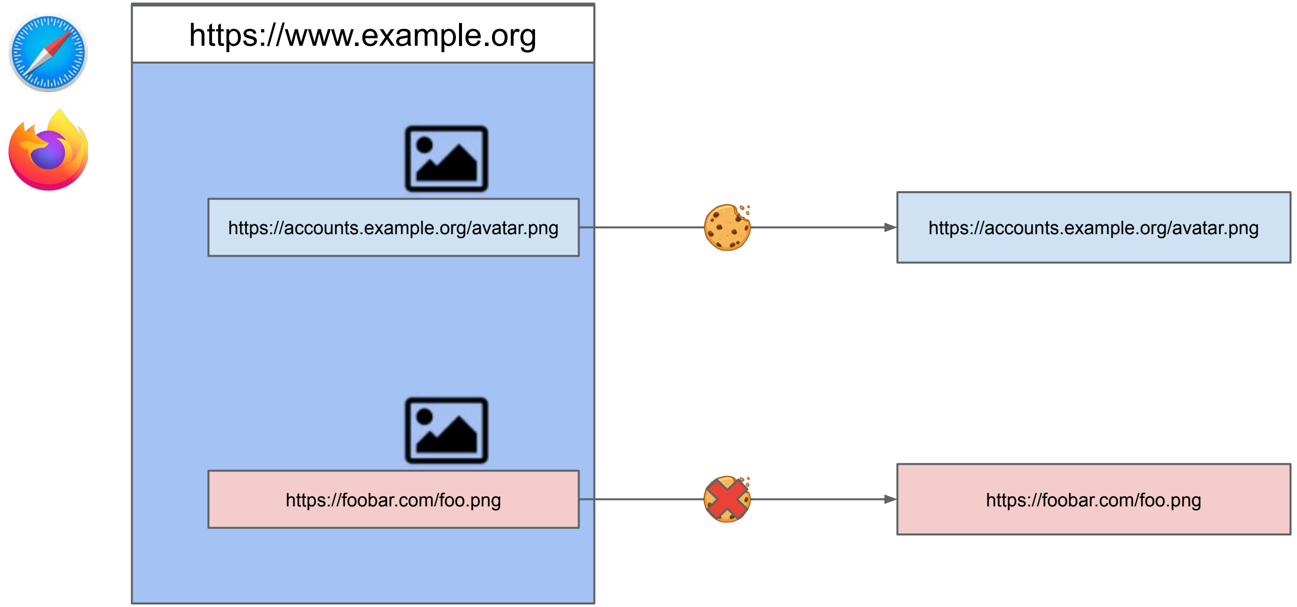
Partition browsing history on source-site

Fixes browsing history leaks

Back to cookies!

How should we block third-party cookies?

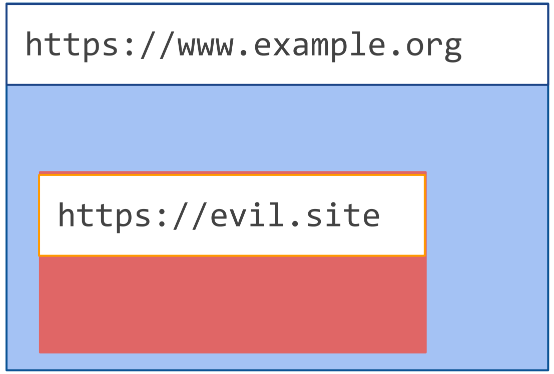
Allowing cookies for requests to top-level site


https://www.example.org
https://accounts.example.org/avatar.png
https://accounts.example.org/avatar.png
https://foobar.com/foo.png
https://foobar.com/foo.png

All requests for subresources that match the top-level site will carry that site's cookies

Problem: Embedding cross-site iframes is common

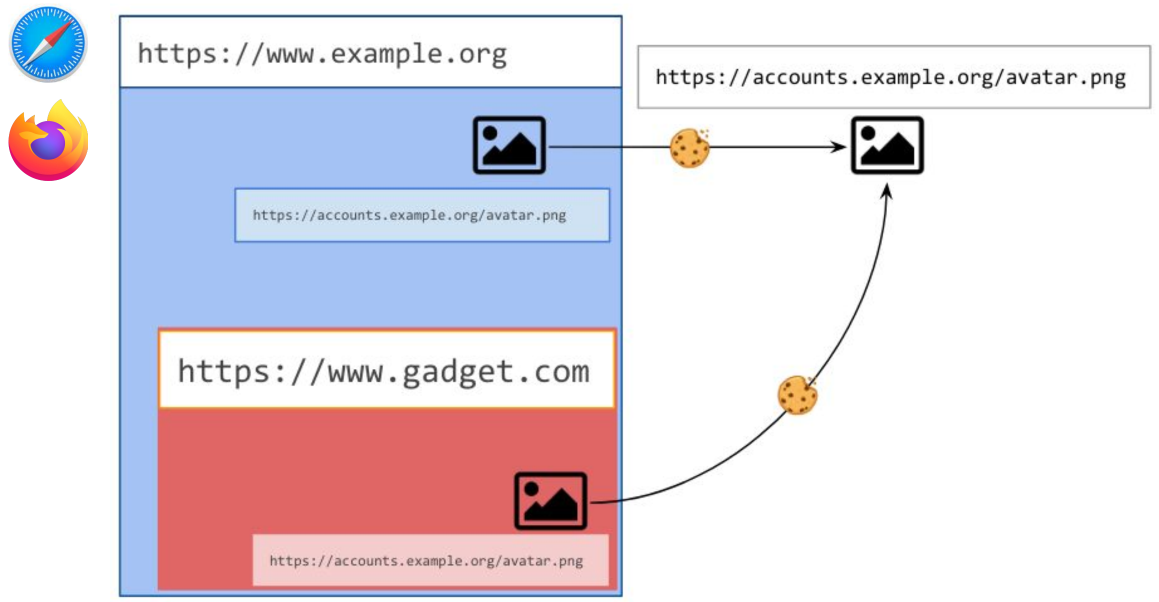
- Ads
- Conversion tracking frames
- Sanitized HTML allowing `<iframe>`s
- Embedded widgets from XSS-able domains
- ...



In the "Allowing cookies for requests to top-level site" model, any document with such an iframe would remove its entire site's web isolation protections.

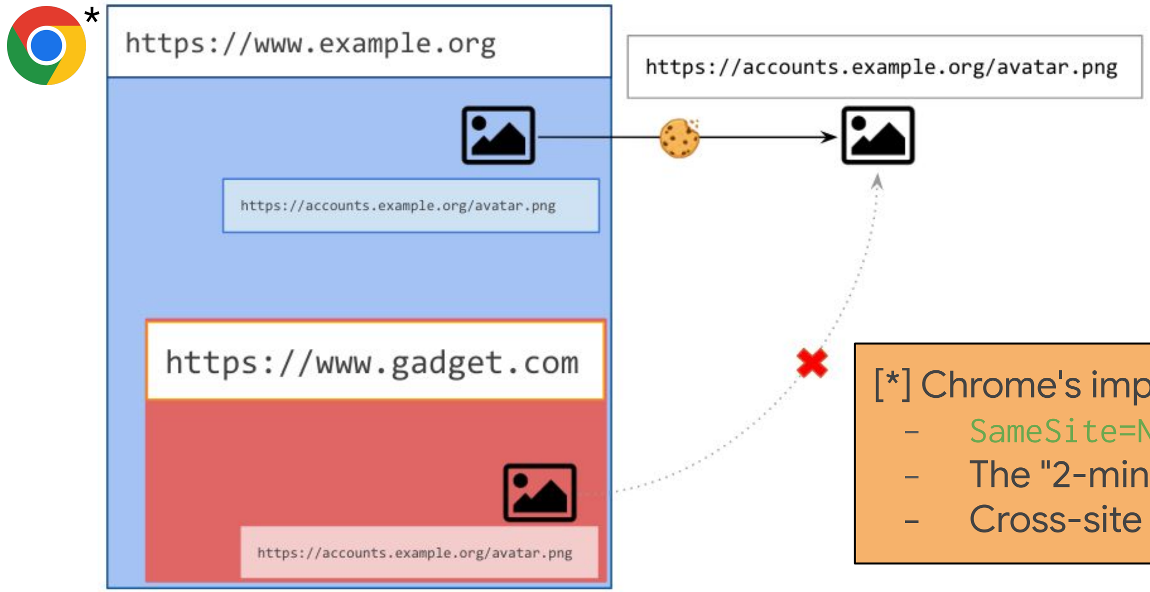
We don't want this.

Allowing cookies for requests to top-level site


https://www.example.org
https://accounts.example.org/avatar.png
https://www.gadget.com
https://accounts.example.org/avatar.png
https://accounts.example.org/avatar.png

All requests for subresources that match the top-level site will carry that site's cookies

The SameSite=Lax-by-default model


*
https://www.example.org
https://accounts.example.org/avatar.png
https://www.gadget.com
https://accounts.example.org/avatar.png
https://accounts.example.org/avatar.png

[*] Chrome's implementation departs from this model:
- SameSite=None is still available (!)
- The "2-minute Lax+POST" mitigation
- Cross-site ⟶ same-site redirects (crbug/1215167)

Uses the "site for cookies" algorithm from RFC6265bis, omitting sending cookies if the initiating document is cross-site, or there are cross-site ancestors or redirects.

Answer: Bring the web closer to the SameSite=Lax* model

[*] `Lax-allowing-unsafe`: Also allow cookies with top-level POST requests

What this would give us: A platform-enforced guarantee against loading authenticated cross-site resources or iframes.

All browsers are fairly close to getting there.

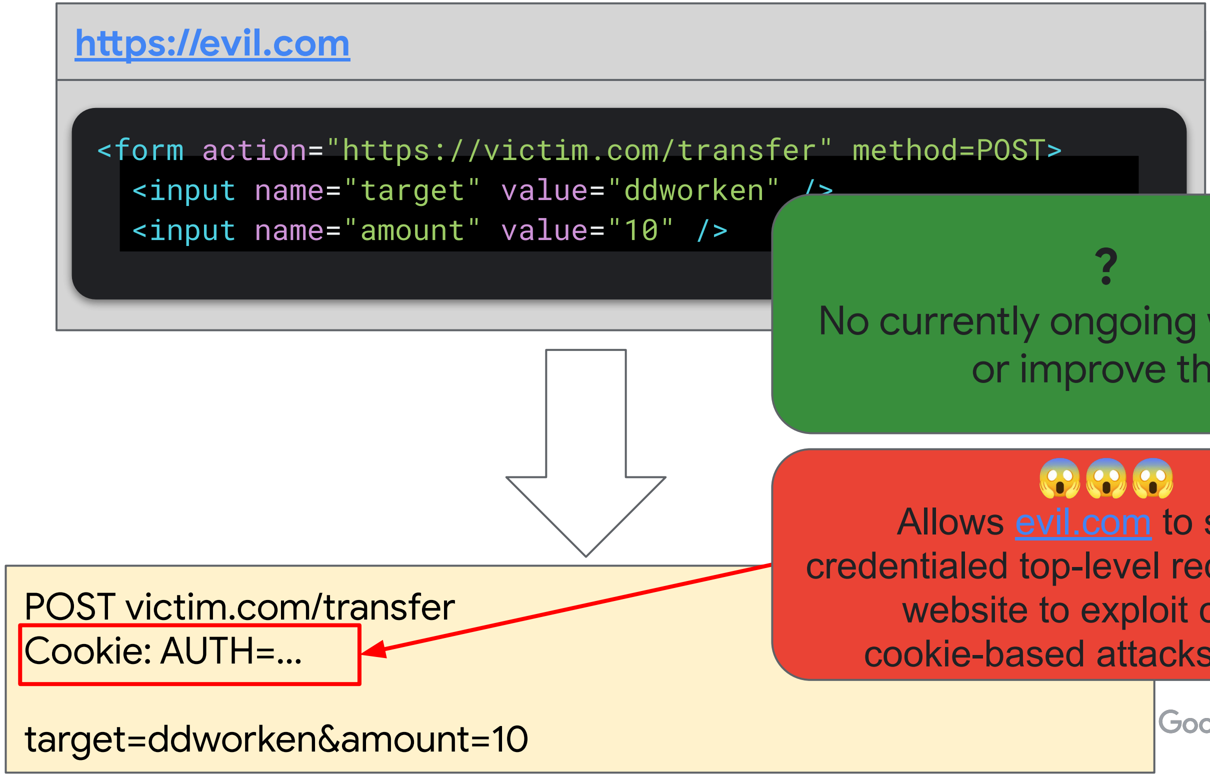
What browsers would need to do:

- Complete the third-party cookie deprecation process & fix known gaps
- Switch to the Lax-allowing-unsafe model
- Everyone: Agree on handling remaining under-defined behaviors...

Will 3PCD magically solve isolation for us?

It's Complicated

Navigational POST requests



https://evil.com

```
<form action="https://victim.com/transfer" method=POST>
  <input name="target" value="ddworken" />
  <input name="amount" value="10" />
```

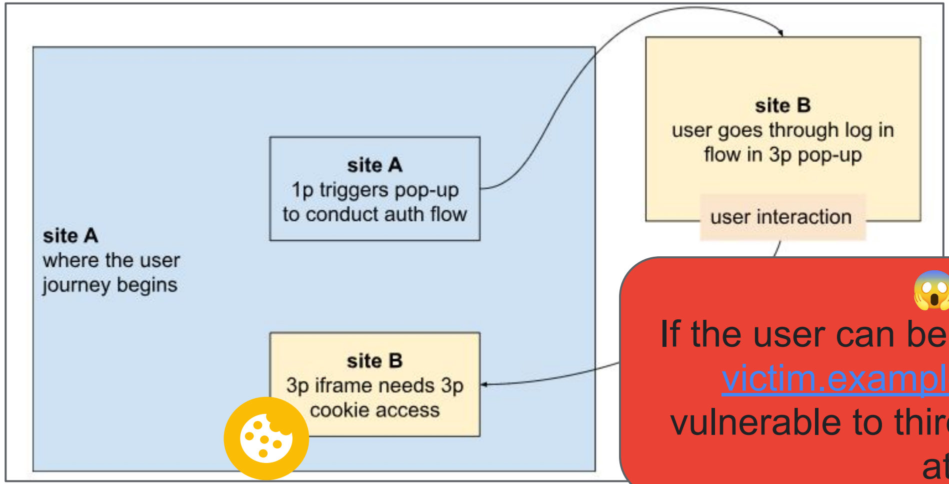
POST victim.com/transfer
Cookie: AUTH=...

target=ddworken&amount=10

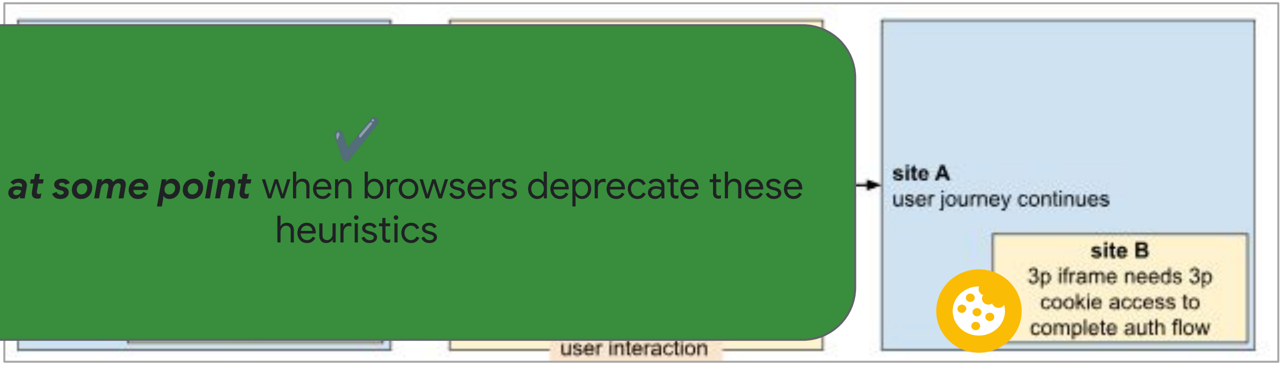
?
No currently ongoing work to fix or improve this

😱😱😱
Allows evil.com to send a credentialed top-level request to any website to exploit certain cookie-based attacks (XSRF)

Heuristics

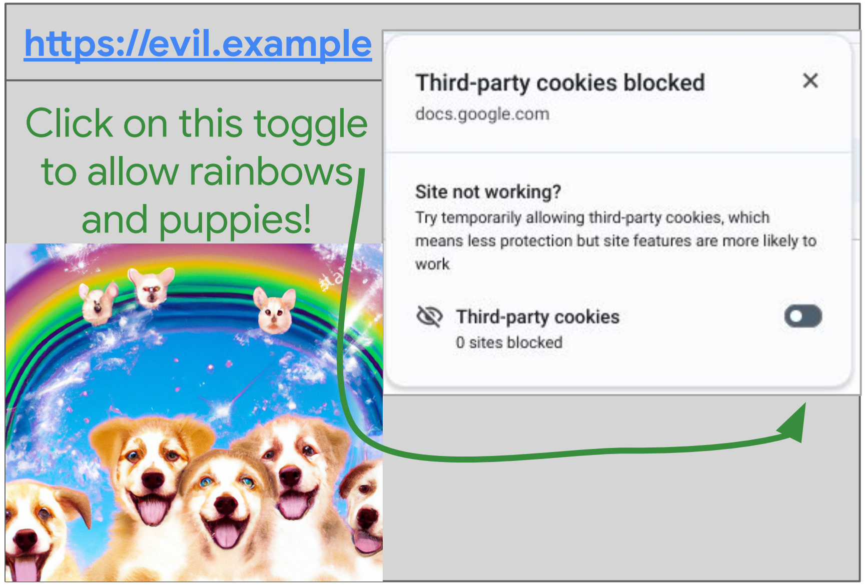


😱😱😱
If the user can be convinced to click on victim.example then it becomes vulnerable to third-party cookie based attacks



✔️
Fixed ***at some point*** when browsers deprecate these heuristics

User Bypass



https://evil.example

Click on this toggle to allow rainbows and puppies!

Third-party cookies blocked

docs.google.com

Site not working?

Try temporarily allowing third-party cookies, which means less protection but site features are more likely to work

Third-party cookies

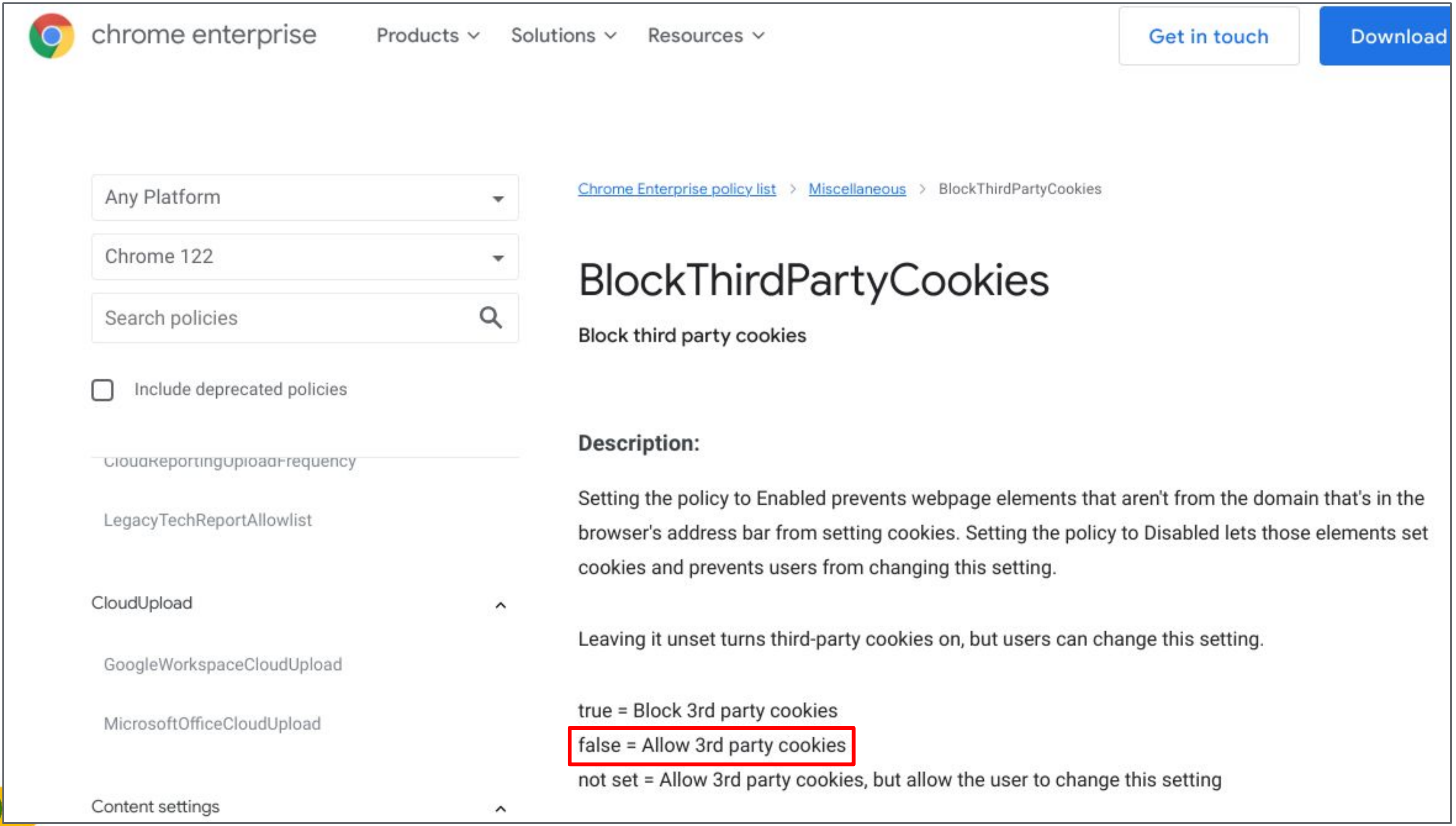
0 sites blocked

😱 If the user can be convinced to trigger User Bypass on evil.example then it can attack any website via third-party cookie based attacks

?

Maybe user bypass will go away or have increased friction ***at some point***

Enterprise policies


chrome enterprise
Products
Solutions
Resources
Get in touch
Download
Any Platform
Chrome 122
Search policies
Include deprecated policies
CloudReportingUploadFrequency
LegacyTechReportAllowlist
CloudUpload
GoogleWorkspaceCloudUpload
MicrosoftOfficeCloudUpload
Content settings

Chrome Enterprise policy list > Miscellaneous > BlockThirdPartyCookies

BlockThirdPartyCookies

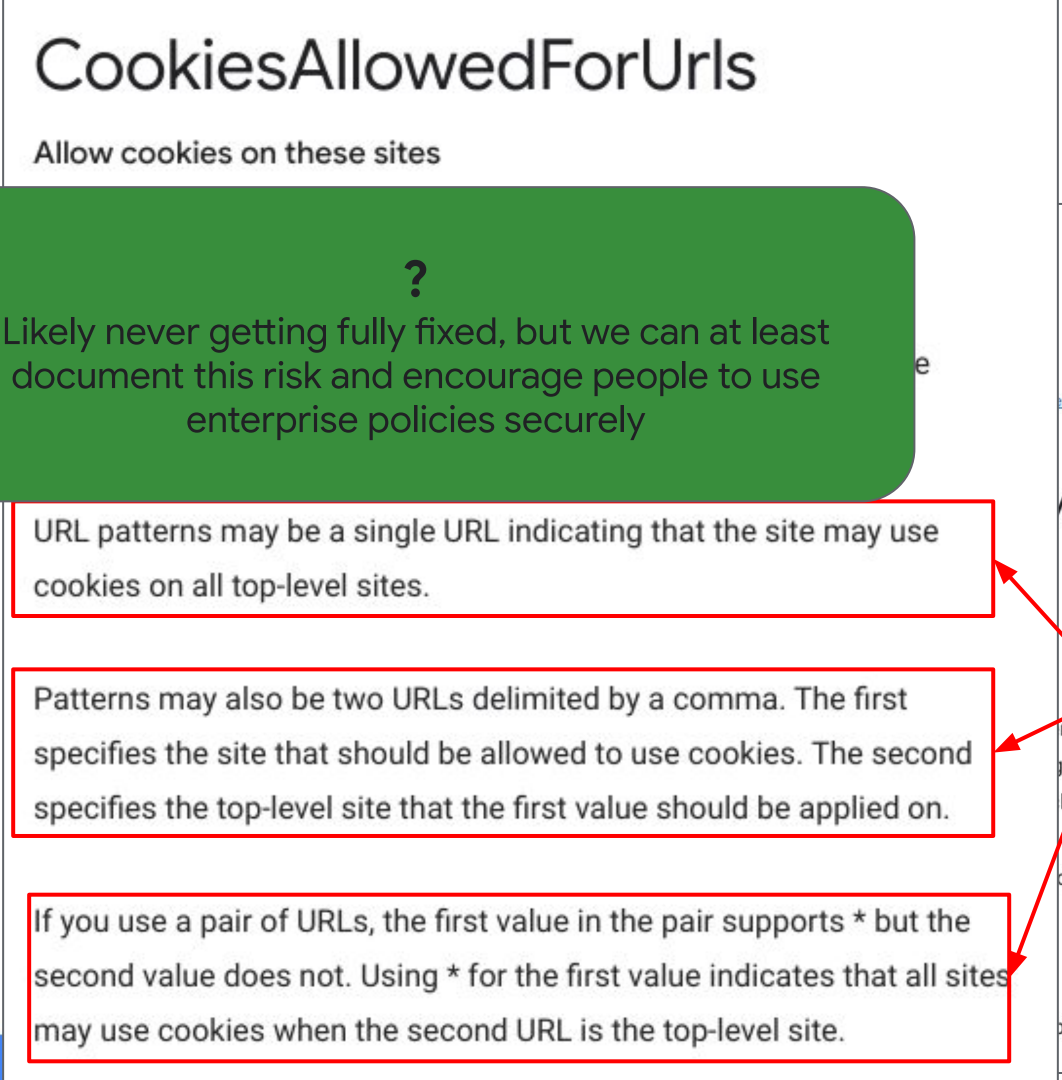
Block third party cookies

Description:

Setting the policy to Enabled prevents webpage elements that aren't from the domain that's in the browser's address bar from setting cookies. Setting the policy to Disabled lets those elements set cookies and prevents users from changing this setting.

Leaving it unset turns third-party cookies on, but users can change this setting.

true = Block 3rd party cookies
false = Allow 3rd party cookies
not set = Allow 3rd party cookies, but allow the user to change this setting



CookiesAllowedForUrls

Allow cookies on these sites

e

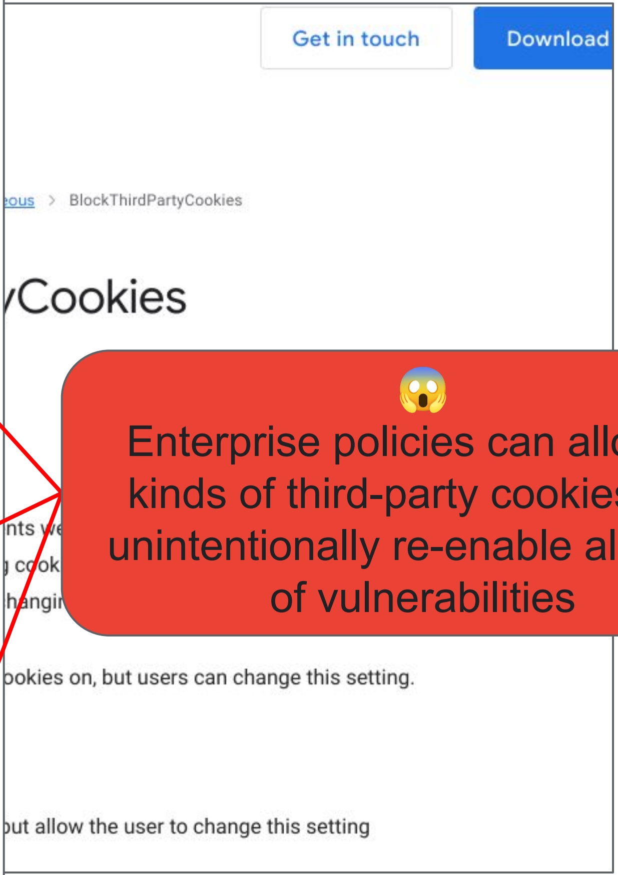
URL patterns may be a single URL indicating that the site may use cookies on all top-level sites.

Patterns may also be two URLs delimited by a comma. The first specifies the site that should be allowed to use cookies. The second specifies the top-level site that the first value should be applied on.

If you use a pair of URLs, the first value in the pair supports * but the second value does not. Using * for the first value indicates that all sites may use cookies when the second URL is the top-level site.

?

Likely never getting fully fixed, but we can at least document this risk and encourage people to use enterprise policies securely



/Cookies

nts we
g cook
hangi

ookies on, but users can change this setting.

ut allow the user to change this setting

😱

Enterprise policies can allow all kinds of third-party cookies and unintentionally re-enable all kinds of vulnerabilities

Isolation best practices for a modern web

Removing third-party cookies aims to provide default isolation for all webapps.

But until this is enforced in all browsers, there are some best practices to follow…

Creating cookies: Explicitly set them as SameSite=Lax

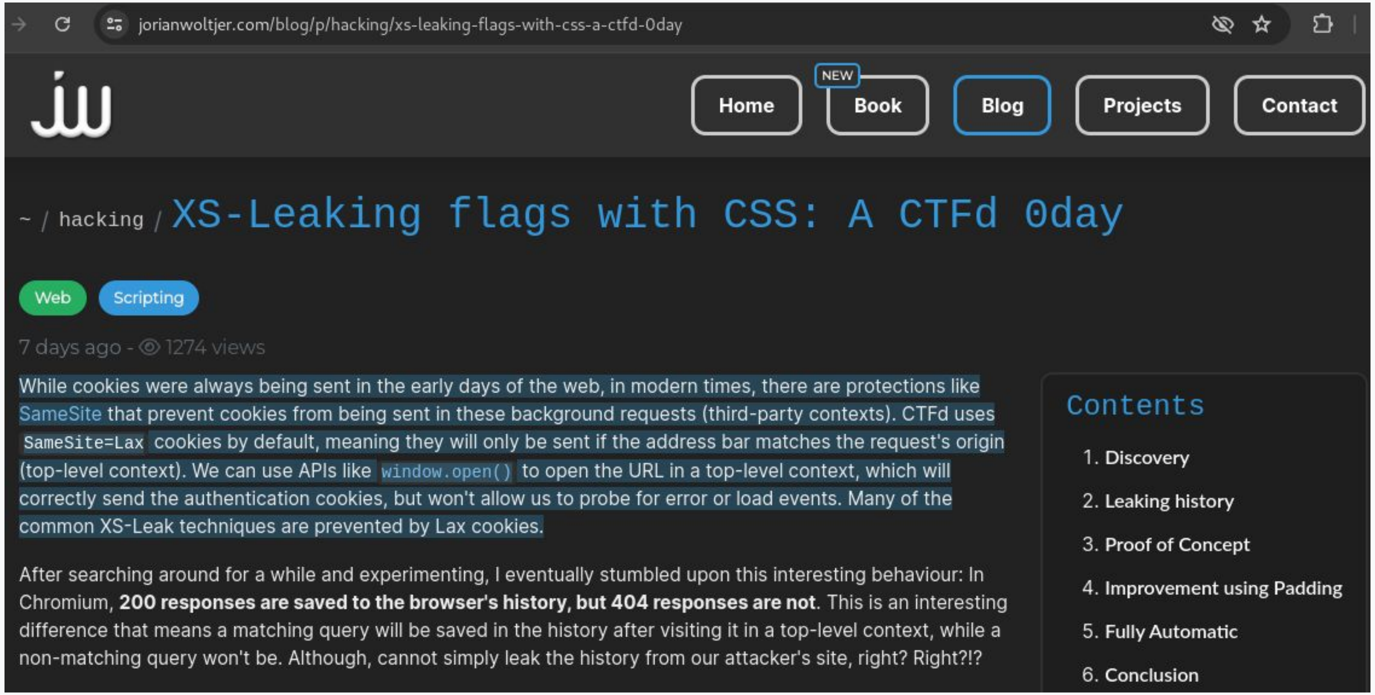
Today, web browsers' default cookie behaviors are less safe than `SameSite=Lax`:

- Safari and Firefox allow any iframe embedded on your site to make credentialed requests to any same-site endpoint.
- All browsers allow `POST` requests with the cookie via top-level navigations.

Setting an explicit `SameSite=Lax` attribute will enforce safer cookie behavior.

```
Set-Cookie: __Host-SESSION=[value]; path=/; SameSite=Lax; Secure;
```

Bonus: This will also make your application compatible with 3P cookie deprecation.


jorianwoltjer.com/blog/p/hacking/xs-leaking-flags-with-css-a-ctfd-0day

~ / hacking / XS-Leaking flags with CSS: A CTFd 0day

7 days ago - 1274 views

While cookies were always being sent in the early days of the web, in modern times, there are protections like SameSite that prevent cookies from being sent in these background requests (third-party contexts). CTFd uses `SameSite=Lax` cookies by default, meaning they will only be sent if the address bar matches the request's origin (top-level context). We can use APIs like `window.open()` to open the URL in a top-level context, which will correctly send the authentication cookies, but won't allow us to probe for error or load events. Many of the common XS-Leak techniques are prevented by Lax cookies.

After searching around for a while and experimenting, I eventually stumbled upon this interesting behaviour: In Chromium, **200 responses are saved to the browser's history, but 404 responses are not**. This is an interesting difference that means a matching query will be saved in the history after visiting it in a top-level context, while a non-matching query won't be. Although, cannot simply leak the history from our attacker's site, right? Right?!?

Contents

Use SameSite=None cookies only as a last resort

You might need to receive authenticated cross-site requests if you:

- Have multiple domains which interact with each other (e.g. use CORS APIs or embedded iframes that maintain logged-in functionality).
- Provide iframes that need to be embedded on any site and use the Storage Access API for authentication.

Tip: Create a second auth cookie that only works for cross-site endpoints.

```
Set-Cookie: SESSION=[value]; path=/; SameSite=Lax; Secure;
Set-Cookie: SESSION_3P=[value]; path=/; SameSite=None; Secure;
```

Opt-in protections: Fetch Metadata Request Headers & Cross-Origin Opener Policy

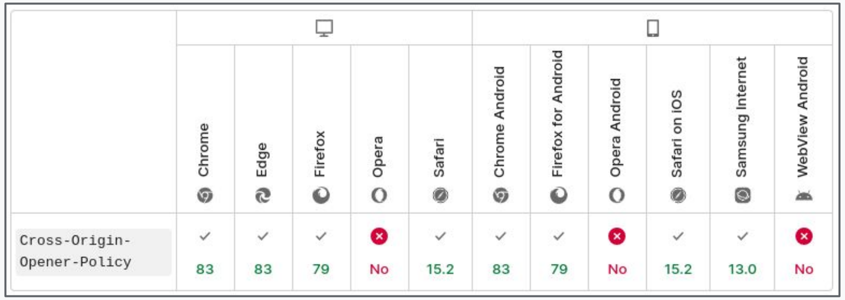
Fetch Metadata headers (`Sec-Fetch-Site` & co.) give servers reliable information about the source of all incoming HTTP requests and allow building general isolation policies.

- web.dev/fetch-metadata

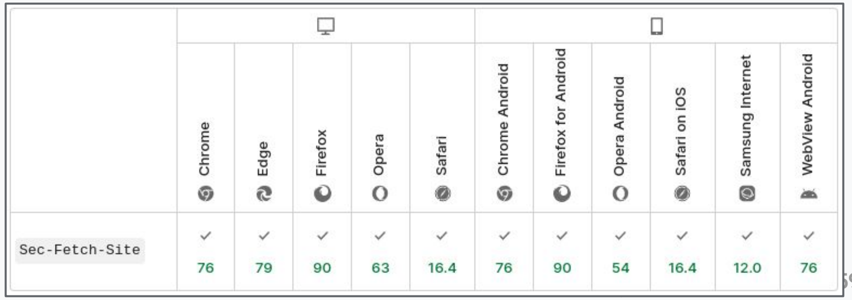
Cross-Origin Opener Policy (COOP) disables access to window properties.

- http.dev/cross-origin-opener-policy

Both are reliably supported by all major browsers:



	🖥					📱					
	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android	Opera Android	Safari on iOS	Samsung Internet	WebView Android
`Cross-Origin-Opener-Policy`	✓ 83	✓ 83	✓ 79	✗ No	✓ 15.2	✓ 83	✓ 79	✗ No	✓ 15.2	✓ 13.0	✗ No



	🖥					📱					
	Chrome	Edge	Firefox	Opera	Safari	Chrome Android	Firefox for Android	Opera Android	Safari on iOS	Samsung Internet	WebView Android
`Sec-Fetch-Site`	✓ 76	✓ 79	✓ 90	✓ 63	✓ 16.4	✓ 76	✓ 90	✓ 54	✓ 16.4	✓ 12.0	✓ 76

Safely Migrating to a Post-3P-Cookie World

Storage Access API (`document.requestStorageAccess()` & `Activate-Storage-Access`) allows an iframe to request its first-party cookies/storage if the user allows.

- **Tip**: Only use it on endpoints that legitimately need to be loaded in 3P contexts.

Related Website Sets allow several domains owned by one organization to declare their relationship and relax cookie restrictions on interactions between them.

- **Tip**: Only add domains that are fully trusted to your RWS. For domains you own, but don't completely control, use Service domains.

Beware of alternative "fixes" such as adding DNS CNAME mappings to third-party sites!

Wrapping up

The web is moving towards more isolation by default through removing third-party cookies and partitioning other browser state, fixing long-standing vulnerability classes.

Opt-in defense mechanisms (`SameSite` **cookies**, **Cross-Origin Opener Policy**, **Fetch Metadata** headers) fill in gaps in the short term, are universally supported in all browsers.

Interesting work happening in W3C working groups (WebAppSec, PrivacyCG) to hash out long-term behaviors for cookies and related APIs. **Join us and/or file bugs!**

MAHALO!