CSRF is dead

(Or is it?)

Stephen Rees-Carter
twitter.com/valorin
Senior Developer
at Defiant / Wordfence

Laracon EU 2020

CSRF

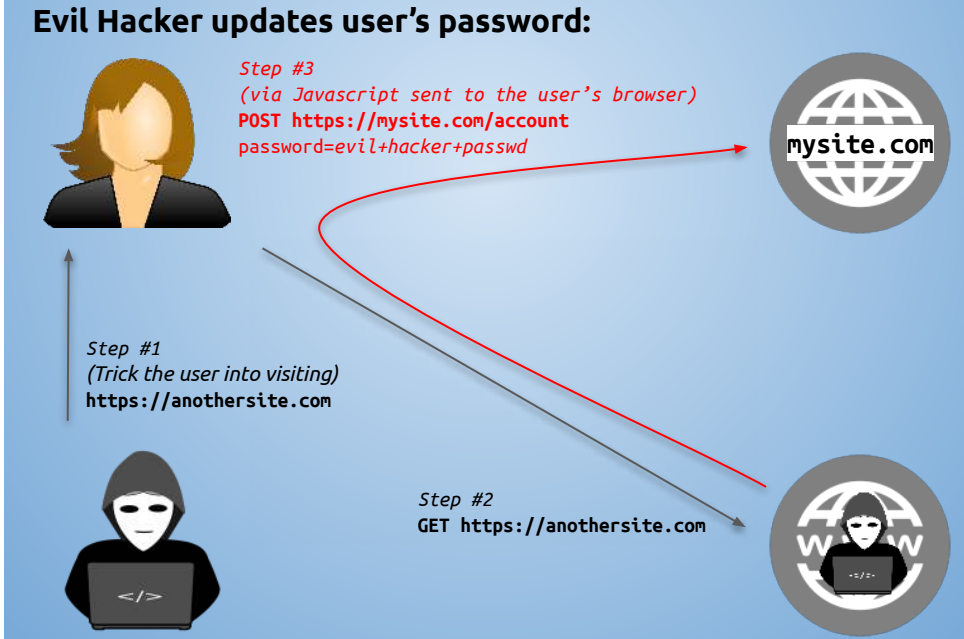
Cross Site Request Forgery

User updates their own password:

Step #1
POST https://mysite.com/account
password=*correct+horse+battery+staple*

CSRF

Cross Site Request Forgery



CSRF

Defending Against Attack

1. CSRF tokens or Nonces
    a. Required in all requests
    b. Known-secret based protection
2. Verify `Origin` or `Referer` header
    a. Cannot be modified by client
3. Client-side cryptographic magic
    a. Some apps work in specific ways

+ SameSite Cookies!

SameSite cookie attribute

```
Set-Cookie: app_session=eyJpdiI6ImNQWTBCU3VERW...;

Set-Cookie: app_session=eyJpdiI6ImNQWTBCU3VERW...; SameSite=Strict

Set-Cookie: app_session=eyJpdiI6ImNQWTBCU3VERW...; SameSite=None; Secure

Set-Cookie: app_session=eyJpdiI6ImNQWTBCU3VERW...; SameSite=Lax
```

Can I use samesite ? Settings

6 results found

'SameSite' cookie attribute - OTHER

Same-site cookies ("First-Party-Only" or "First-Party") allow servers to mitigate the risk of CSRF and information leakage attacks by asserting that a particular cookie should only be sent with requests initiated from the same registrable domain.

Usage	% of all users	?
Global	85.23% + 7.08% =	92.31%
Australia	83.15% + 9.95% =	93.1%

Current aligned | Usage relative | Date relative | Apply filters | Show all | ?

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
		74			5 12.4				
	18	75	3 80	4 5 13	13.3				
1 2 11	81	76	3 81	5 13.1	13.4	all	3 81	12.12	11.1
		77	3 83	5 TP					
		78	3 84						
			3 85						

SameSite=Strict

Cross-Site/Third-Party Requests

Blocks all CSRF attacks (when cookies are required)

Embedded Content

`<iframe>` `<img>`

Unsafe Requests

`POST/PUT/DELETE/...`

Safe Requests

`GET/HEAD`

SameSite=None; Secure

Cross-Site/Third-Party Requests (HTTPS Only)

Blocks no CSRF attacks

Embedded Content

<iframe> <img>

Unsafe Requests

POST/PUT/DELETE/…

Safe Requests

GET/HEAD

SameSite=None

(Without “Secure”)

Will not be sent on any request, HTTPS or HTTP.

SameSite=Lax

Cross-Site/Third-Party Requests

Blocks CSRF attacks on “unsafe” requests.

Embedded Content

<iframe> <img>

Unsafe Requests

POST/PUT/DELETE/…

Safe Requests

GET/HEAD

Story Time...

⚠ A cookie associated with a cross-site resource at https://hackwp.valorin.dev/ was set without the (index):1
`SameSite` attribute. It has been blocked, as Chrome now only delivers cookies with cross-site requests if they are set with `SameSite=None` and `Secure`. You can review cookies in developer tools under Application>Storage>Cookies and see more details at https://www.chromestatus.com/feature/5088147346030592 and https://www.chromestatus.com/feature/5633521622188032.

Improving privacy and security on the web

Tuesday, May 7, 2019

Improving cookie controls in Chrome

We announced at I/O that we will be updating Chrome to provide users with more transparency about how sites are using cookies, as well as simpler controls for cross-site cookies. We will preview these new features later this year.

We are making a number of upcoming changes to Chrome to enable these features, starting with modifying how cookies work so that developers need to explicitly specify which cookies are allowed to work across websites — and could be used to track users. The mechanism we use builds on the web's SameSite cookie attribute, and you can find the technical details on web.dev.

https://blog.chromium.org/2019/05/improving-privacy-and-security-on-web.html

Developers: Get Ready for New SameSite=None; Secure Cookie Settings

Wednesday, October 23, 2019

Chrome Enforcement Starting in February 2020

With Chrome 80 in February, Chrome will treat cookies that have no declared SameSite value as `SameSite=Lax` cookies. Only cookies with the `SameSite=None; Secure` setting will be available for external access, provided they are being accessed from secure connections. The Chrome Platform Status trackers for SameSite=None and Secure will continue to be updated with the latest launch information.

Mozilla has affirmed their support of the new cookie classification model with their intent to implement the `SameSite=None; Secure` requirements for cross-site cookies in Firefox. Microsoft recently announced plans to begin implementing the model starting as an experiment in Microsoft Edge 80.

https://blog.chromium.org/2019/10/developers-get-ready-for-new.html

SameSite by default cookies

Treat cookies that don't specify a SameSite attribute as if they were SameSite=Lax. Sites must specify SameSite=None in order to enable third-party usage. – Mac, Windows, Linux, Chrome OS, Android

#same-site-by-default-cookies

Enabled

Cookies without SameSite must be secure

If enabled, cookies without SameSite restrictions must also be Secure. If a cookie without SameSite restrictions is set without the Secure attribute, it will be rejected. This flag only has an effect if "SameSite by default cookies" is also enabled. – Mac, Windows, Linux, Chrome OS, Android

#cookies-without-same-site-must-be-secure

Enabled

SameSite Cookie Changes in February 2020: What You Need to Know

Monday, February 3, 2020

With the stable release of Chrome 80 this month, Chrome will begin enforcing a new secure-by-default cookie classification system, treating cookies that have no declared SameSite value as `SameSite=Lax` cookies. Only cookies set as `SameSite=None; Secure` will be available in third-party contexts, provided they are being accessed from secure connections.

https://blog.chromium.org/2020/02/samesite-cookie-changes-in-february.html

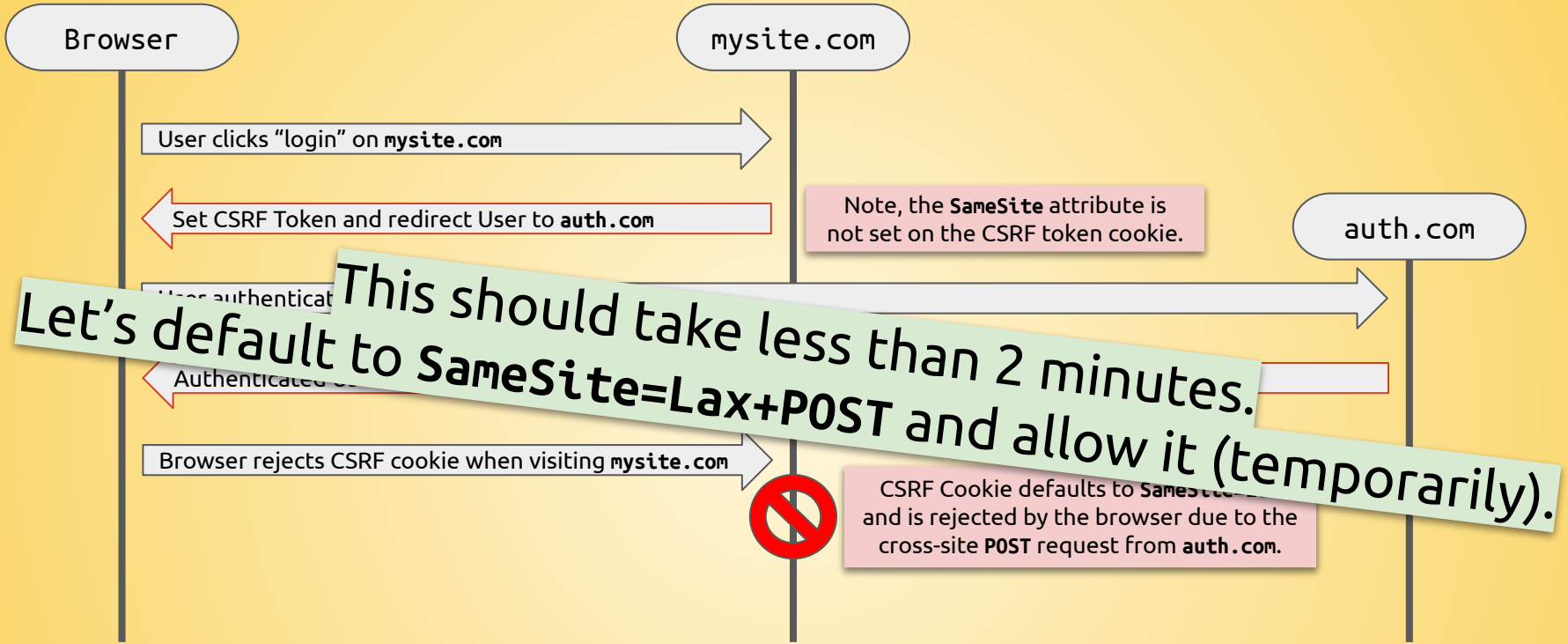
Temporarily rolling back SameSite Cookie Changes

Friday, April 3, 2020

With the stable release of Chrome 80 in February, Chrome began enforcing secure-by-default handling of third-party cookies as part of our ongoing effort to improve privacy and security across the web. We've been gradually rolling out this change since February and have been closely monitoring and evaluating ecosystem impact, including proactively reaching out to individual websites and services to ensure their cookies are labeled correctly.

However in light of the extraordinary global circumstances due to COVID-19, we are temporarily rolling back the enforcement of SameSite cookie labeling, starting today. While most of the web ecosystem was prepared for this change, we want to ensure stability for websites providing essential services including banking, online groceries, government services and healthcare that facilitate our daily life during this time. As we roll back enforcement, organizations, users and sites should see no disruption.

https://blog.chromium.org/2020/04/temporarily-rolling-back-samesite.html



Wait a sec… this will break my Auth flow!

(Example based from the widely used OpenID Connect authentication flow used by Azure Active Directory and Microsoft Account authentication.)

"Same-Site" domains

mysite.com

static.mysite.com

account.mysite.com

"Cross-Site" domains

github.io

valorin.github.io

laravel.github.io

*Subdomains of domains on the **Public Suffix List** (https://publicsuffix.org/) are considered "cross-site".*

Demo time...

Recent Requests

	Cookie	iframe	image	GET	POST
Same-Site Requests	http_shared_none	NO	NO	NO	NO
	http_shared_none_secure	NO	NO	NO	NO
	http_shared_lax	YES	YES	YES	YES
	http_shared_strict	YES	YES	YES	YES
	http_shared_default	YES	YES	YES	YES
	http_shared_invalid	YES	YES	YES	YES
	https_shared_none	NO	NO	NO	NO
	https_shared_none_secure	YES	YES	YES	YES
	https_shared_lax	YES	YES	YES	YES
	https_shared_strict	YES	YES	YES	YES
	https_shared_default	YES	YES	YES	YES
	https_shared_invalid	YES	YES	YES	YES
Cross-Site Requests	http_external_none	NO	NO	NO	NO
	http_external_none_secure	NO	NO	NO	NO
	http_external_lax	NO	NO	YES	NO
	http_external_strict	NO	NO	NO	NO
	http_external_default	NO	NO	YES	YES
	http_external_invalid	NO	NO	YES	YES
	https_external_none	NO	NO	NO	NO
	https_external_none_secure	YES	YES	YES	YES
	https_external_lax	NO	NO	YES	NO
	https_external_strict	NO	NO	NO	NO
	https_external_default	NO	NO	YES	YES
	https_external_invalid	NO	NO	YES	YES

Delayed Requests

Cookie	iframe	image	GET	POST
http_shared_none	NO	NO	NO	NO
http_shared_none_secure	NO	NO	NO	NO
http_shared_lax	YES	YES	YES	YES
http_shared_strict	YES	YES	YES	YES
http_shared_default	YES	YES	YES	YES
http_shared_invalid	YES	YES	YES	YES
https_shared_none	NO	NO	NO	NO
https_shared_none_secure	YES	YES	YES	YES
https_shared_lax	YES	YES	YES	YES
https_shared_strict	YES	YES	YES	YES
https_shared_default	YES	YES	YES	YES
https_shared_invalid	YES	YES	YES	YES
http_external_none	NO	NO	NO	NO
http_external_none_secure	NO	NO	NO	NO
http_external_lax	NO	NO	YES	NO
http_external_strict	NO	NO	NO	NO
http_external_default	NO	NO	YES	NO
http_external_invalid	NO	NO	YES	NO
https_external_none	NO	NO	NO	NO
https_external_none_secure	YES	YES	YES	YES
https_external_lax	NO	NO	YES	NO
https_external_strict	NO	NO	NO	NO
https_external_default	NO	NO	YES	NO
https_external_invalid	NO	NO	YES	NO

Recent Requests

Cookie	iframe	image	GET	POST
http_shared_none	NO	NO	NO	NO
http_shared_none_secure	NO	NO	NO	NO
http_shared_lax	YES	YES	YES	YES
http_shared_strict	YES	YES	YES	YES
http_shared_default	YES	YES	YES	YES
http_shared_invalid	YES	YES	YES	YES
https_shared_none	NO	NO	NO	NO
https_shared_none_secure	YES	YES	YES	YES
https_shared_lax	YES	YES	YES	YES
https_shared_strict	YES	YES	YES	YES
https_shared_default	YES	YES	YES	YES
https_shared_invalid	YES	YES	YES	YES
http_external_none	NO	NO	NO	NO
http_external_none_secure	NO	NO	NO	NO
http_external_lax	NO	NO	YES	NO
http_external_strict	NO	NO	NO	NO
http_external_default	NO	NO	YES	YES
http_external_invalid	NO	NO	YES	YES
https_external_none	NO	NO	NO	NO
https_external_none_secure	YES	YES	YES	YES
https_external_lax	NO	NO	YES	NO
https_external_strict	NO	NO	NO	NO
https_external_default	NO	NO	YES	YES
https_external_invalid	NO	NO	YES	YES

SameSite=Lax+POST
(Allowed <2 mins)

Delayed Requests

Cookie	iframe	image	GET	POST
http_shared_none	NO	NO	NO	NO
http_shared_none_secure	NO	NO	NO	NO
http_shared_lax	YES	YES	YES	YES
http_shared_strict	YES	YES	YES	YES
http_shared_default	YES	YES	YES	YES
http_shared_invalid	YES	YES	YES	YES
https_shared_none	NO	NO	NO	NO
https_shared_none_secure	YES	YES	YES	YES
https_shared_lax	YES	YES	YES	YES
https_shared_strict	YES	YES	YES	YES
https_shared_default	YES	YES	YES	YES
https_shared_invalid	YES	YES	YES	YES
http_external_none	NO	NO	NO	NO
http_external_none_secure	NO	NO	NO	NO
http_external_lax	NO	NO	YES	NO
http_external_strict	NO	NO	NO	NO
http_external_default	NO	NO	YES	NO
http_external_invalid	NO	NO	YES	NO
https_external_none	NO	NO	NO	NO
https_external_none_secure	YES	YES	YES	YES
https_external_lax	NO	NO	YES	NO
https_external_strict	NO	NO	NO	NO
https_external_default	NO	NO	YES	NO
https_external_invalid	NO	NO	YES	NO

SameSite=Lax+POST
(Blocked >2 mins)

Laravel & SameSite

What you need to know.

config/session.php

```
/*
|--------------------------------------------------------------------------
| Same-Site Cookies
|--------------------------------------------------------------------------
|
| This option determines how your cookies behave when cross-site requests
| take place, and can be used to mitigate CSRF attacks. By default, we
| will set this value to "lax" since this is a secure default value.
|
| Supported: "lax", "strict", "none", null
|
*/

'same_site' => 'lax',
```

Laravel defaulted to **`SameSite=Lax`** in 7.0.0.
Older versions specified **`null`** (not set).

```php
use Illuminate\Support\Facades\Cookie;

Cookie::queue(
    $name,
    $value,
    $minutes  = 0,
    $path     = null,
    $domain   = null,
    $secure   = null,
    $httpOnly = true,
    $raw      = false,
    $sameSite = null
);
```

What Option Do I Use?

- Use **`SameSite=Strict`** if…
  - User shouldn't be automatically logged in
  - Actions must be performed over **`GET`** requests

- Use **`SameSite=None`** if…
  - **`POST`** requests or embedded content (**`<iframe>`**/**`<img>`**) needed between third-party domains

- Use ***`<nothing>`*** if…
  - You like unexpected behaviour to confuse your users

- **Otherwise, just use `SameSite=Lax`.**

Is CSRF dead?

Stephen Rees-Carter @valorin · Jan 23

Ok, web security people... Is CSRF dead now that the major browsers are going to default cookies to SameSite=Lax?

4 1 1

n00py
@n00py1

Replying to @valorin and @mmaunder

Unlikely. Even with samesite=lax 90% of my CSRF PoCs will still work.

1:46 PM · Jan 23, 2020 · Twitter for iPhone

n00py
@n00py1

Replying to @valorin and @mmaunder

Most of it is related to client work, but basically I've found that a lot of the time I can perform state changing actions using GET requests, many actions that typically send POST can also be sent as GET interchangeably.

1:56 PM · Jan 23, 2020 · Twitter Web App

Is CSRF dead?

No :-(

Thank you!

Slides & Links

src.id.au/csrf

Contact Me

stephen@rees-carter.net

twitter.com/valorin