

Apr  
il  
202  
1



# APRIL Hacking HTTP CORS



A theory to practice approach

Minh Tuấn - Cyber Security Research  
Team



# TABLE OF CONTENTS



Same Origin Policy

HTTP CORS

Simple Request

Preflight HTTP  
Request

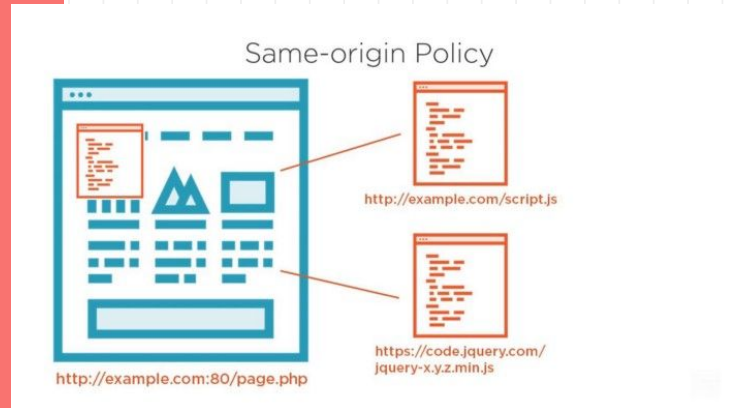
\*

SameSite Cookie

Hacking

Reward

# Same Origin Policy





## Same Origin Policy



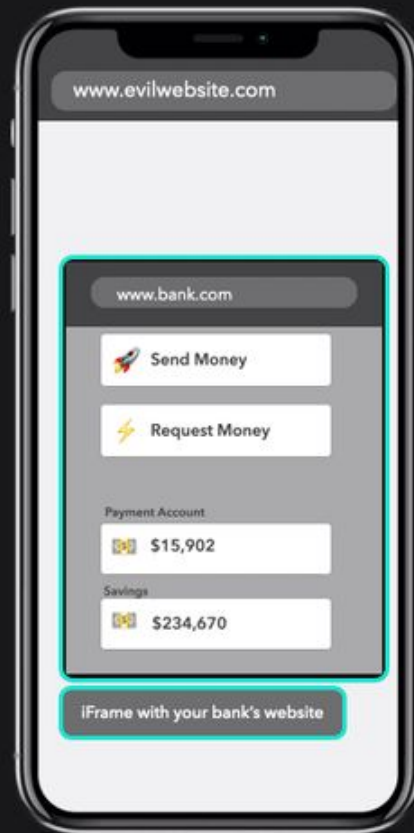
In computing, the same-origin policy (sometimes abbreviated as SOP) is an important concept in the web application security model. Under the policy, a web browser permits scripts contained in a first web page to access data in a second web page, but only if both web pages have the same origin. An origin is defined as a combination of URI scheme, host name, and port number. This policy prevents a malicious script on one page from obtaining access to sensitive data on another web page through that page's Document Object Model.



# Same Origin Policy



**Without**  
**Same-Origin Policy**



Evil Website's Devs

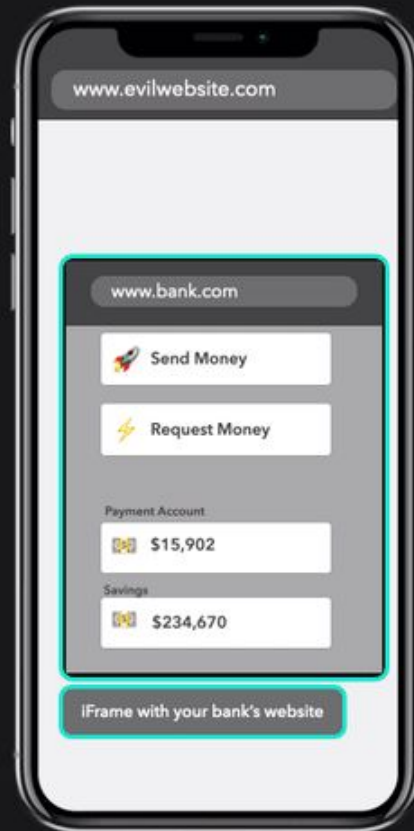




# Same Origin Policy



With  
Same-Origin Policy



Evil Website's Devs





# Same Origin Policy



STT	A	B
1	<a href="http://site.com">http://site.com</a>	<a href="https://site.com">https://site.com</a>
2	<a href="http://www.site.com">http://www.site.com</a>	<a href="http://site.com">http://site.com</a>
3	<a href="https://site.com">https://site.com</a>	<a href="https://api.site.com">https://api.site.com</a>
4	<a href="https://site.com">https://site.com</a>	<a href="https://site.com:8080">https://site.com:8080</a>
5	<a href="https://site.com">https://site.com</a>	<a href="https://site.com/page">https://site.com/page</a>



# Same Origin Policy



Original

`https://www.mywebsite.com`

Same  
Origin

Different **path**

`https://www.mywebsite.com/page`

Cross  
Origin

Different **domain**

`https://www.anotherwebsite.com`

Different **protocol**

`http://www.mywebsite.com`

Different **subdomain**

`https://api.mywebsite.com`

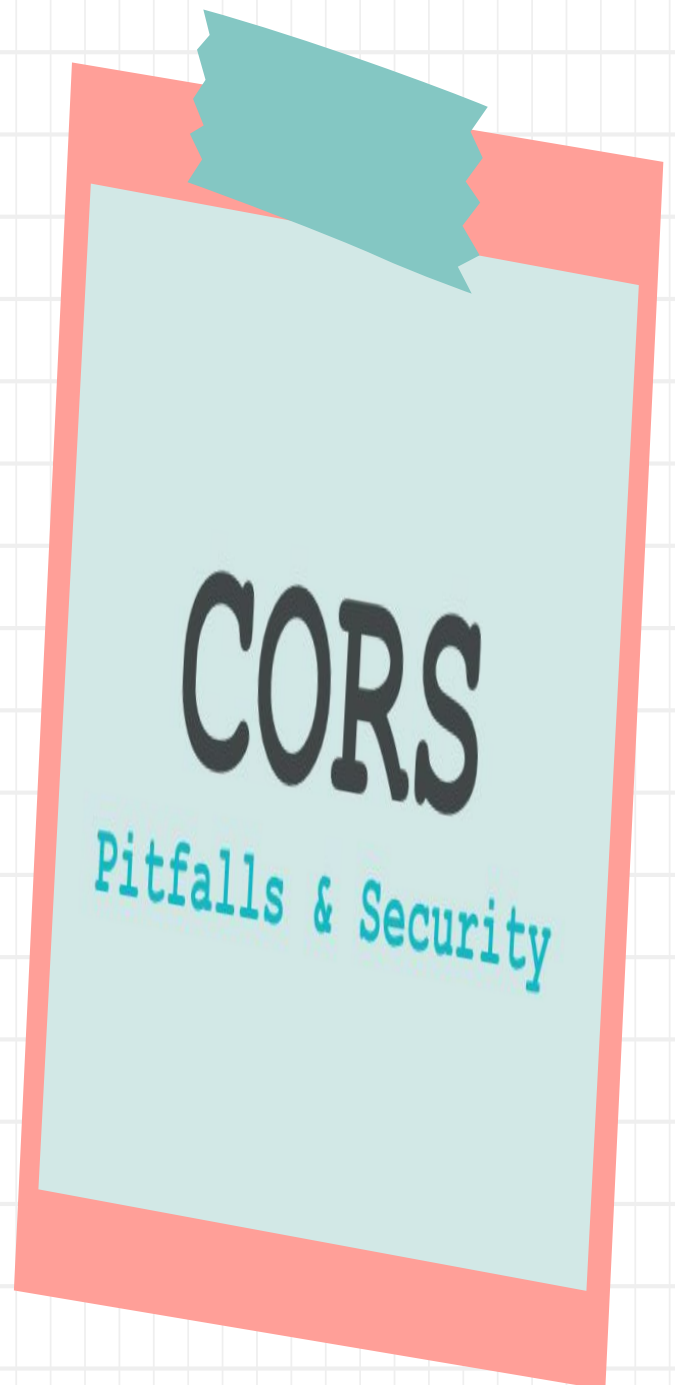
Different **port**

`https://www.mywebsite.com:8080`



HTTP CORS

Why must?



<https://www.mywebsite.com/>

<https://www.mywebsite.com/users.json>

<https://api.mywebsite.com/users.json>

<https://www.randomwebsite.com/users.json>



CORS



Allowed Origins

`https://www.mywebsite.com`



CORS



Allowed Origins

`https://www.mywebsite.com`





```
> function cors() {  
  var xhttp = new XMLHttpRequest();  
  xhttp.onreadystatechange = function () {  
    if (this.readyState == 4) {  
      document.getElementById("demo").innerHTML = console.log(this.responseText);  
    }  
  };  
  xhttp.open("GET", "https://about.viblo.asia", true);  
  xhttp.setRequestHeader('Content-Type', 'text/plain');  
  xhttp.send();  
}
```

< undefined

> cors();

< undefined

✖ Access to XMLHttpRequest at 'https://about.viblo.asia/' from origin 'https://viblo.asia' has been blocked by CORS policy: No 'Access-Control-Allow-Origin' header is present on the requested resource. [tong-quan-ve-phuong-\\_g-php-bJzKmR4wZ9N:1](#)

[609753c...js:2](#)

✖ ▶ Uncaught TypeError: Cannot set property 'innerHTML' of null  
at XMLHttpRequest.xhttp.onreadystatechange (<anonymous>:5:55)  
at XMLHttpRequest.r (609753c...js:2)

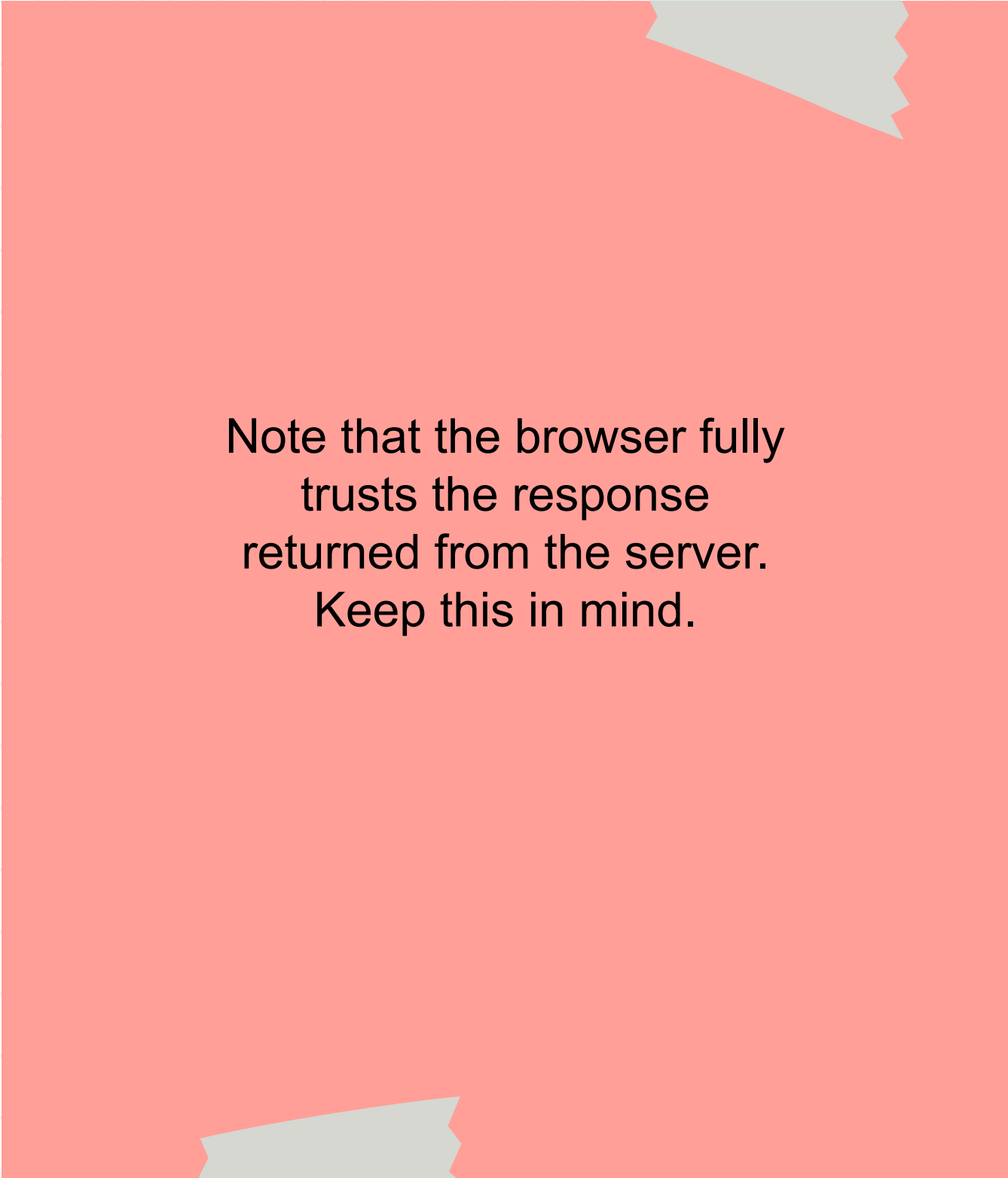
[609753c...js:2](#)

✖ ▶ GET https://about.viblo.asia/ net::ERR\_FAILED


[609753c...js:2](#)

>





Note that the browser fully trusts the response returned from the server. Keep this in mind.

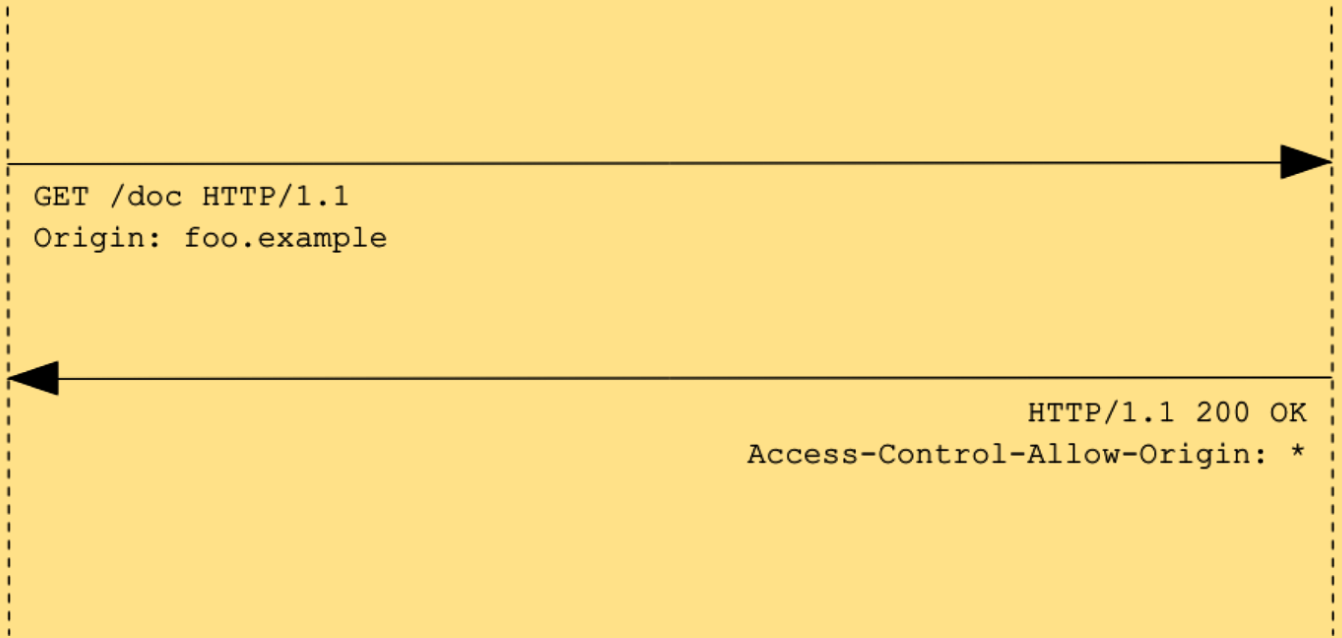


**SIMPLE  
REQUEST**



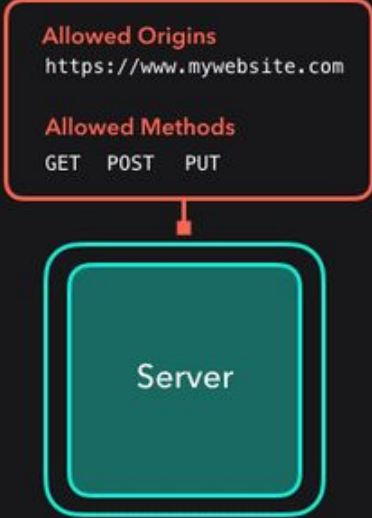
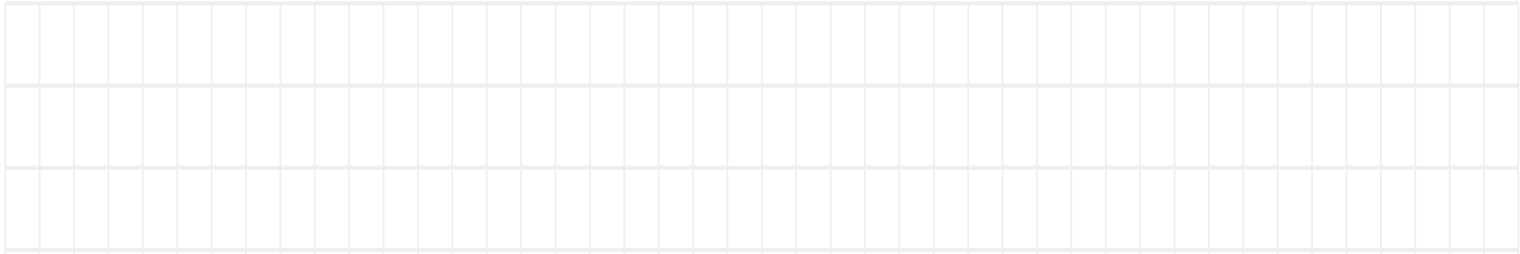
# Client

# Server



FEB  
202  
1

Preflight HTTP  
Request



CORS

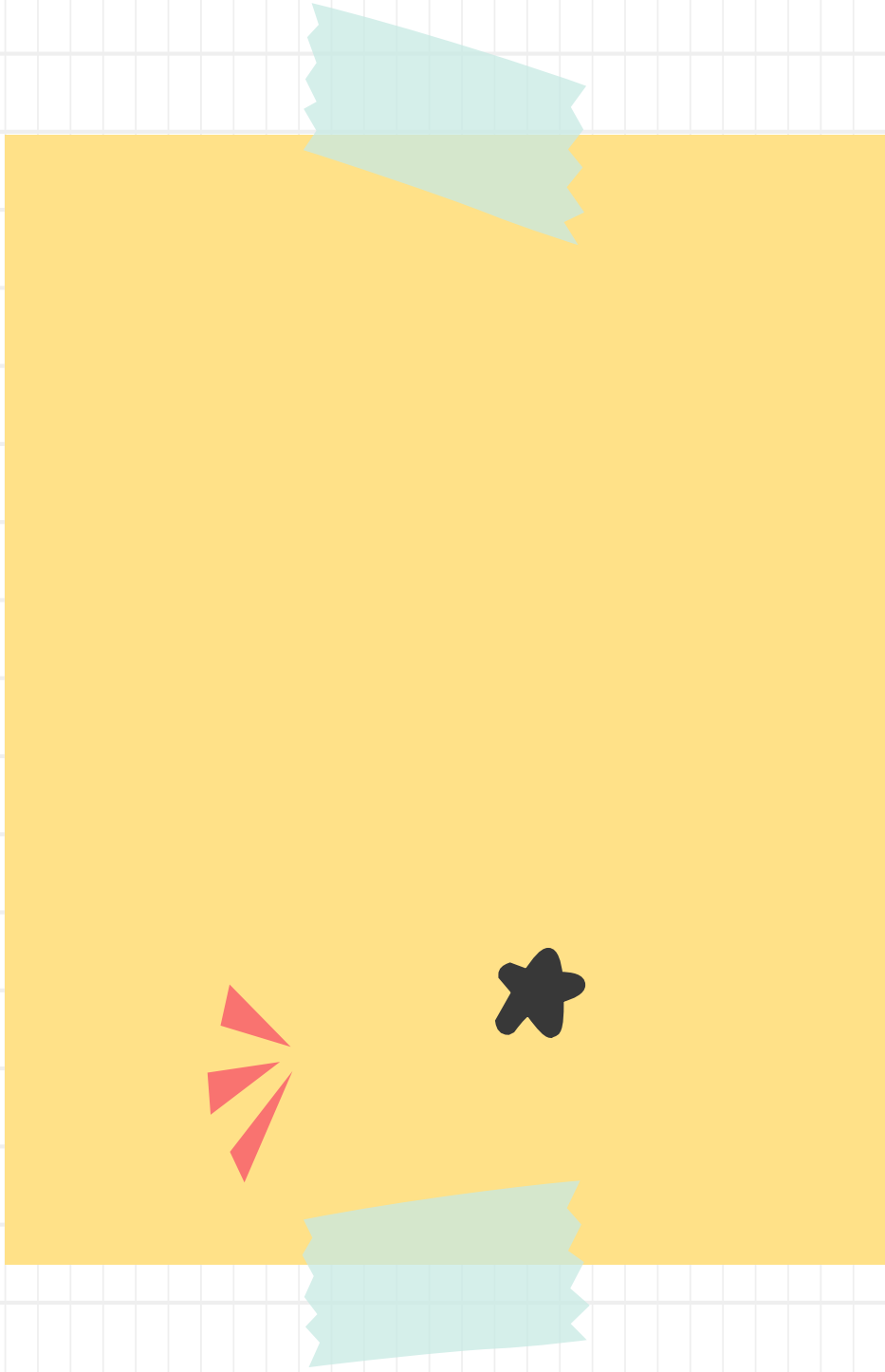
Browser

Allowed Origins  
https://www.mywebsite.com

Allowed Methods  
GET POST PUT

Server

Header	Used for <i>Preflight</i> HTTP?	HTTP Type
<b>Access-Control-Allow-Origin</b>	<b>NO</b>	Response
Access-Control-Allow-Credentials	<b>NO</b>	Response
Access-Control-Allow-Headers	<b>YES</b>	Response
Access-Control-Allow-Methods	<b>YES</b>	Response
Access-Control-Expose-Headers	<b>NO</b>	Response
Access-Control-Max-Age	<b>YES</b>	Response
Access-Control-Request-Headers	<b>YES</b>	Request
Access-Control-Request-Method	<b>YES</b>	Request
<b>Origin</b>	<b>NO</b>	Request
Timing-Allow-Origin	<b>NO</b>	Response





```
1 HTTP/1.0 200 OK
2 Server: BlueServer/5.3.3.13
3 Date: Tue, 03 Nov 2020 08:24:08 GMT
4 P3P: CP="CAO COR CURa ADMa DEVa OUR IND ONL COM DEM PRE"
5 Access-Control-Allow-Origin: *
6 Set-Cookie: session=49254647686d756039983e04084d4121; path=/
7 Connection: close
8 Content-Type: text/html; charset=UTF-8
9 Content-Length: 16036
10 Cache-Control: no-cache, no-store
11 X-Frame-Options: SAMEORIGIN
12
```



```
1 HTTP/1.1 200 OK
2 Server: nginx/1.16.1
3 Content-Type: application/json
4 Connection: close
5 Cache-Control: max-age=0, must-revalidate, no-cache, no-store, private
6 Date: Wed, 04 Nov 2020 01:06:12 GMT
7 Set-Cookie: news_laravel_session=eyJpdiiI6Im9ySEVEVkd0dnFBQWdpQ1JTTm1TM2c9
8 Access-Control-Allow-Origin: *
9 Access-Control-Allow-Credentials: true
0 Access-Control-Allow-Methods: GET, POST, OPTIONS, HEAD
1 Access-Control-Allow-Headers: DNT,X-Mx-ReqToken,Keep-Alive,User-Agent,X-Req
2 Content-Length: 254
3
4 {
  "count":5,
  "url":"https://54.244.193.184/tina-smith-jason-lewis-face-off-in-race
  "headline":"Sen. Tina Smith, Jason Lewis face off in race for Minnesota
```



DEMO

[https://careers.takeaway.c  
om](https://careers.takeaway.com)



# Same-Site Cookie



- SameSite = Strict
- SameSite = Lax
- SameSite = None



*Browser*

Personal, yet  
safe, experience  
offered by  
SameSite



# Same-Site Cookie



## SameSite is here

Cookies that do not specify a SameSite attribute will be treated as if they specified **SameSite=Lax** (supported in Chrome and Safari)

### SameSite=Lax

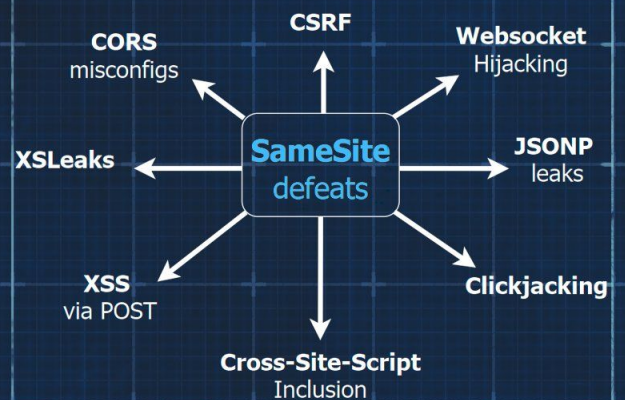
Allows the cookie to be sent on some cross-site requests.  
[top-level navigation+GET/HEAD)

### SameSite=Strict

Never allows the cookie to be sent on a cross-site request.  
Only when the user types the website in the URL bar and presses enter

### SameSite=None

Cookies will be sent in all contexts (like before)



@HackerScrolls



# Same-Site Cookie



## 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.

## Making the Web Safer

To protect users from CSRF attacks, browsers need to change the way cookies are handled. The two primary changes are:

- When not specified, cookies will be treated as `SameSite=Lax` by default
- Cookies that explicitly set `SameSite=None` in order to enable cross-site delivery must also set the `Secure` attribute. (In other words, they must require HTTPS.)

Web sites that depend on the old default behavior must now explicitly set the `SameSite` attribute to `None`. In addition, they are required to include the `Secure` attribute. Once this change is made inside of Firefox, if web sites fail to set `SameSite` correctly, it is possible those sites could break for users.



# Same-Site Cookie



## Samesite: cookies different behavior

	CHROME				FIREFOX/IE/EDGE				SAFARI			
	No Attr	Lax	Strict	None	No Attr	Lax	Strict	None	No Attr	Lax	Strict	None
<b>Just a link</b> <code>&lt;a href="//host/csrf"&gt;click_me&lt;/a&gt;</code>	+	+	-	+	+	+	-	+	+	+	-	+
<b>Classic POST CSRF</b> <code>&lt;form action="//host/csrf" method=POST&gt;</code>	-	-	-	+	+	-	-	+	+	-	-	+
<b>POST CSRF (fresh 120 sec cookie)</b> <code>&lt;form action="//host/csrf" method=POST&gt;</code>	+	-	-	+	+	-	-	+	+	-	-	+
<b>Image</b> <code>&lt;img src="//host/csrf"&gt;</code>	-	-	-	+	+	-	-	+	-	-	-	-
<b>Iframe</b> <code>&lt;iframe src="//host/csrf"&gt;&lt;/iframe&gt;</code>	-	-	-	+	+	-	-	+	-	-	-	-
<b>Open new window</b> <code>window.open('//host/csrf')</code>	+	+	-	+	+	+	-	+	+	+	-	+
<b>JS async cross-domain request</b> <code>fetch(), XMLHttpRequest(), WebSocket()</code>	-	-	-	+	+	-	-	+	-	-	-	-
<b>User types the URL in browser</b>	+	+	+	+	+	+	+	+	+	+	+	+
<b>The default browser opens the clicked link from desktop app</b>	+	+	+	+	+	+	+	+	+	+	+	+

cookies will not be sent

cookies will be sent

differs from Google Chrome

No attr: cookies do not have 'Samesite' attribute

@HackerScrolls

Apr  
202  
1



# Hacking

**Taking advantage of  
unsafe CORS  
configuration**

## Request

```
Pretty Raw \n Actions \n Select extension... \n1 POST /index.php?rest_route=/quiz-survey-master/v1/questions/1 HTTP/1.1
2 Host: wordpress.lc
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:84.0) Gecko/2010
4 Accept: application/json, text/javascript, */*; q=0.01
5 Accept-Language: vi-VN,vi;q=0.8,en-US;q=0.5,en;q=0.3
6 Accept-Encoding: gzip, deflate
7 Referer: http://wordpress.lc/wp-admin/admin.php?page=mlw_quiz_options&
8 Content-Type: application/json
9 X-WP-Nonce: 0376badf1b
10 X-Requested-With: XMLHttpRequest
11 Content-Length: 738
12 Origin: http://abc.com
13 DNT: 1
14 Connection: close
15 Cookie: wp-settings-time-2=1601275284; wp-settings-2=libraryContent%3Dl
16
17 {
  "id": "1",
  "quizID": "1",
  "type": "0",
  "name": "Add description here!",
  "question_title": "1234",
  "answerInfo": "",
  "comments": "1",
```

## Response

```
Pretty Raw Render \n Actions \n1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Tue, 26 Jan 2021 04:05:19 GMT
4 Content-Type: application/json; charset=UTF-8
5 Connection: close
6 X-Robots-Tag: noindex
7 Link: <http://wordpress.lc/index.php?rest_route=/>; rel="https://api.
8 X-Content-Type-Options: nosniff
9 Access-Control-Expose-Headers: X-WP-Total, X-WP-TotalPages, Link
10 Access-Control-Allow-Headers: Authorization, X-WP-Nonce, Content-Disp
11 Expires: Wed, 11 Jan 1984 05:00:00 GMT
12 Cache-Control: no-cache, must-revalidate, max-age=0
13 X-WP-Nonce: 0376badf1b
14 Allow: POST, PUT, PATCH, GET
15 Access-Control-Allow-Origin: http://abc.com
16 Access-Control-Allow-Methods: OPTIONS, GET, POST, PUT, PATCH, DELETE
17 Access-Control-Allow-Credentials: true
18 Vary: Origin
19 Content-Length: 20
20
21 {
  "status": "success"
}
```

## Request

### P Request

```
1 Pretty Raw \n Actions Select extension...
2
3 1 GET /accounts/100000428/metadata HTTP/1.1
4 2 Host: users-y.lucidstaging.app
5 3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:84.0) Gecko/20100101
6 4 Firefox/84.0
7 5 Accept: application/json
8 6 Accept-Language: vi-VN,vi;q=0.8,en-US;q=0.5,en;q=0.3
9 7 Accept-Encoding: gzip, deflate
10 8 Origin: null
11 9 DNT: 1
12 10 Referer: https://y.lucidstaging.app/teams/100000428
13 11 Connection: close
14 12 Cookie: lt-anonymous-id="0.3fb11d00726837d81775cb83063"; ab_id=
15 13 iUrzi7zXap69urqdAArODE59d/w=; lucidauth=
16 14 90f8fdf41d744531d66dc2cda1accbe0d869f3b5-ts%3D1612168337%26uid%3D1394; account_id=
17 15 100000428; showteam=1; userId=1394; username=
18 16 c2lsZW50LXdpbmQtMzcwOEBidWdjc93ZG5pbmphLmNvbQ==; isReg=1; km_ni=1394; apt.uid=
19 17 AP-YJF8ENEFCIIY-2-2-1612168358939-76856386.0.0; lt-session-data=
20 18 {"id":"0.79c9683b1776074d2b5","lastUpdatedDate":"2021-02-02T02:23:44.866Z"};
21 19 lt-pageview-id="0.d555bc9177608d8ae4"; apt.sid=
22 20 AP-YJF8ENEFCIIY-2-2-1612228943907-95854196; visit_source=Other Source;
23 21 lucidauthshort=9a08606ea8a0e3e4a50ca711e3df35c758ec97dd-ts=1612232594; km_ai=1394;
24 22 usertesting=dj1bJMhuB6mkNdrwDBsjnMaf1gZYYL+M
25 23
```

## Response


```
1 Pretty Raw Render \n Actions Select
2
3 1 HTTP/1.1 200 OK
4 2 vary: Authorization,Origin,Accept-Encoding
5 3 set-cookie: lucidauthshort=cfacaa0bc8a764cb98f9e8ddcf8b4066d2a7473a-ts=
6 4 x-lucid-flow-id: c6208e54079af54
7 5 x-lucid-principal: uid:1394
8 6 access-control-max-age: 0
9 7 access-control-allow-origin: null
10 8 access-control-expose-headers: Link
11 9 access-control-allow-credentials: true
12 10 Content-Length: 35214
13 11 content-type: application/json
14 12 date: Wed, 03 Feb 2021 02:07:04 GMT
15 13 keep-alive: timeout=60
16 14 access-control-allow-methods: OPTIONS,HEAD,GET,PUT,DELETE,POST,PATCH
17 15 x-frame-options: SAMEORIGIN
18 16 strict-transport-security: max-age=86400
19 17 connection: close
20 18
21 19 [
22 20 {
23 21 "account":"https://users-y.lucidstaging.app/accounts/100000428",
24 22 "product":"spark",
25 23 "name":"BrandAssetImages",
26 24 "value":[""]
27 25 }
28 26 ]
```



REWARD



## Unsafe Cross-Origin Resource Sharing in www.golfgalaxy.com

 DICK'S Sporting Goods (Private) · Updated a month ago

**P4**

**Unresolved**

\$0

5 points

Comments **0**

## (CORS) Cross-origin resource sharing misconfiguration lead to information leak

Takeaway.com · Updated 7 days ago

**P2**

**Unresolved**

\$300

20 points

Comments **8**





# THANKS!

**Does anyone have any questions?**

[nguy.minh.tuan@sun-asterisk.com](mailto:nguy.minh.tuan@sun-asterisk.com)

[research.sun-asterisk.com](http://research.sun-asterisk.com)

