should test for “javascript:” instead of redirecting to your website as XSS will be possible here. Some common ways to bypass filters:

java%0d%0ascript%0d%0a:alert(0)
j%0d%0aava%0d%0aas%0d%0acrip%0d%0at%0d%0a:confirm`0`
java%07script:prompt`0`
java%09scrip%07t:prompt`0`
jjavascript**a**javascript**v**javascript**a**javascript**s**javascript**c**javascript**r**javascript**i**javascript **p**javascript**t:**confirm`0`

Server Side Request Forgery (SSRF)

Server Side Request Forgery is the in-scope domain issuing a request to an URL/endpoint you've defined. This can be for multiple reasons and sometimes it doesn't always signal the target is vulnerable. When hunting for SSRF I specifically look for features which already take an URL parameter. **Why?** Because as mentioned earlier I am looking for specific areas of a website where a developer may have created a filter to prevent malicious activity. For example, on large bug bounty programs I will instantly try to find their API console (*if one is available, usually found on their developer docs page*). This area usually contains features which already take a URL parameter and execute code. Think about webhooks. As well as hunting for features which handle a URL, just simply keep an eye out for common parameter names used for handling URLs. I found SSRF on Yahoo from doing simply this as a request was made which contained the parameter “url”. Another great example is this disclosed report from Jobert Abma, https://hackerone.com/reports/446593. The feature was right in front of him and required no special recon or brute forcing.

When testing for SSRF you should **always** test how they handle redirects. You can actually host a redirect locally via using XAMPP & NGrok. XAMPP allows you to run PHP code locally and ngrok gives you a public internet address (**don’t forget to turn it off when finished testing!** *refer to https://www.bugbountyhunter.com/ for a tutorial on using XAMPP to aid you in your security research*). Setup a simple