PROFESSIONAL

Getting started with the Burp Collaborator client

Last updated: January 27, 2023 **Read time:** 2 Minutes

In this tutorial, you will learn how to use the Burp Collaborator client. You will test whether you can induce a target site to make a request to an arbitrary server that could potentially be controlled by an attacker.

Step 1: Access the lab

Open Burp's browser, and use it to access the following URL:

```
https://portswigger.net/web-security/ssrf/blind
/lab-out-of-band-detection
```

Click **Access the lab** and log in to your PortSwigger account if prompted. This opens your own instance of a deliberately vulnerable shopping website.

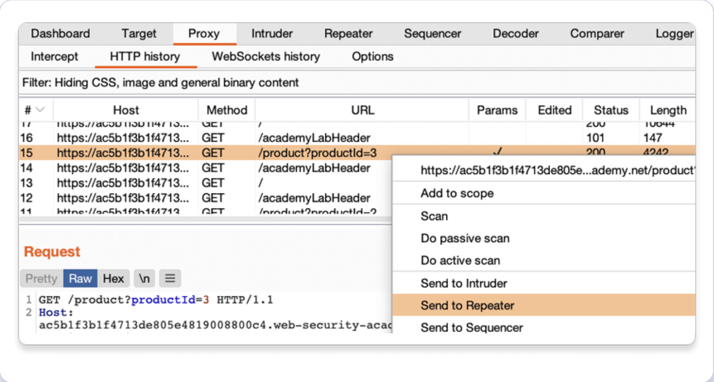
Step 2: Browse the target site

In the browser, explore the site by clicking on a couple of the product pages.

Step 3: Send an interesting request to Repeater

In Burp, go to the **Proxy > HTTP history** tab.

Right-click a `GET /product?productId=[...]` request and select **Send to Repeater**.



Filter: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status	Length
17	https://ac5b1f3b1f4713...	GET	/			200	10644
16	https://ac5b1f3b1f4713...	GET	/academyLabHeader			101	147
15	https://ac5b1f3b1f4713...	GET	/product?productId=3	✓		200	4242
14	https://ac5b1f3b1f4713...	GET	/academyLabHeader				
13	https://ac5b1f3b1f4713...	GET	/				
12	https://ac5b1f3b1f4713...	GET	/academyLabHeader				
11	https://ac5b1f3b1f4713...	GET	/product?productId=2				

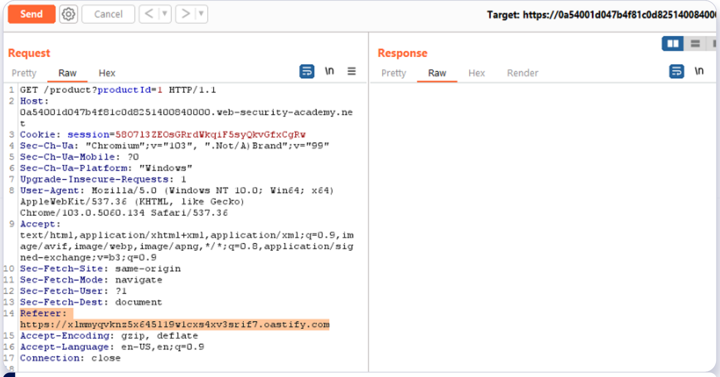
```
GET /product?productId=3 HTTP/1.1
Host:
ac5b1f3b1f4713de805e4819008800c4.web-security-acac
```

Step 4: Inject a Collaborator payload into the request

Go to the **Repeater** tab. Highlight the URL in the `Referer` header, right-click, and select **Insert Collaborator payload**. This replaces the `Referer` URL with a URL that points to the Collaborator server, for example:

```
204119i326shak9tnk6k36z8jlahj74r.oastify.com
```

Send the request.



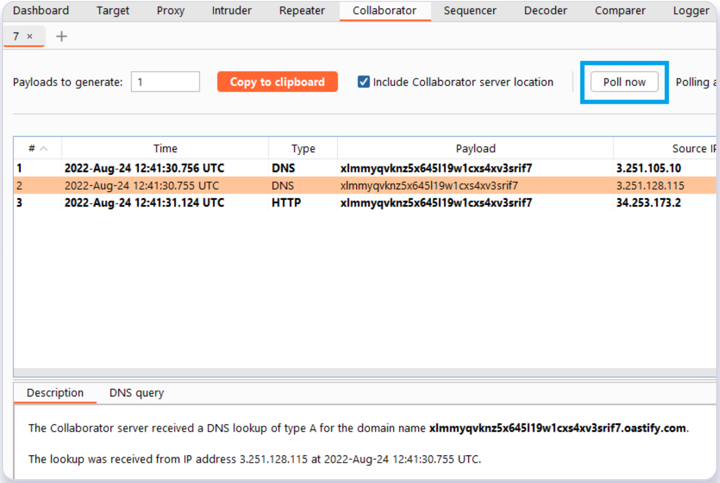
Target: https://0a54001d047b4f81c0d82514008400

```
GET /product?productId=1 HTTP/1.1
Host:
0a54001d047b4f81c0d8251400840000.web-security-academy.ne
t
Cookie: session=580713ZEOsGRrdWkqiF5syQkvGfxCgRw
Sec-Ch-Ua: "Chromium";v="103", ".Not/A)Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/103.0.5060.134 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,im
age/avif,image/webp,image/apng,*/*;q=0.8,application/sig
ned-exchange;v=b3;q=0.9
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer:
https://xlmmyqvknz5x645119wlcxs4xv3srif7.oastify.com
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

> ### Note
>
> The Collaborator server domain name may change, as we periodically add new domain names. For more information, see Burp Collaborator Client.

Step 5: Poll for interactions

Go to the **Collaborator** tab. Collaborator client polls for interactions every 60 seconds, so you may see some interactions listed already. If not, click **Poll now**. Interactions received as a result of your Collaborator payloads are displayed. This confirms that the target site made a request to the arbitrary server.



Payloads to generate: 1

☑ Include Collaborator server location

#	Time	Type	Payload	Source IP
1	2022-Aug-24 12:41:30.756 UTC	DNS	xlmmyqvknz5x645l19w1cxs4xv3srif7	3.251.105.10
2	2022-Aug-24 12:41:30.755 UTC	DNS	xlmmyqvknz5x645l19w1cxs4xv3srif7	3.251.128.115
3	2022-Aug-24 12:41:31.124 UTC	HTTP	xlmmyqvknz5x645l19w1cxs4xv3srif7	34.253.173.2

The Collaborator server received a DNS lookup of type A for the domain name **xlmmyqvknz5x645l19w1cxs4xv3srif7.oastify.com**.

The lookup was received from IP address 3.251.128.115 at 2022-Aug-24 12:41:30.755 UTC.

In this case, you see both HTTP and DNS interactions. Click on an interaction to view more details.

Summary

Congratulations, you have now successfully:

- Generated a Collaborator payload.
- Inserted a Collaborator payload in a request.
- Induced the application to send a request to your Collaborator subdomain, and identified this by polling the server for interactions.

You now know how to use the Burp Collaborator client to generate a proof of concept for invisible vulnerabilities, in this case, blind SSRF.

What next?

This tutorial is just an initial proof of concept. To learn how you can exploit this kind of behavior in the wild, check out the Web Security Academy, in particular:

- Blind SSRF.
- Blind SQL injection.

In this tutorial, we manually tested a single input using Burp Repeater. In practice, you may want to test multiple inputs at once. For more information, see Testing multiple inputs with the Burp Collaborator Client.

Was this article helpful?

YES, THANKS!   NOT REALLY