Experiment No. 3

Aim: Manual request manipulation using Burp Repeater and Intruder.

Learning Objective: To learn how to manually perform requests manipulation using Burp Repeater and Intruder.

Theory:

What Is a Burp Suite?

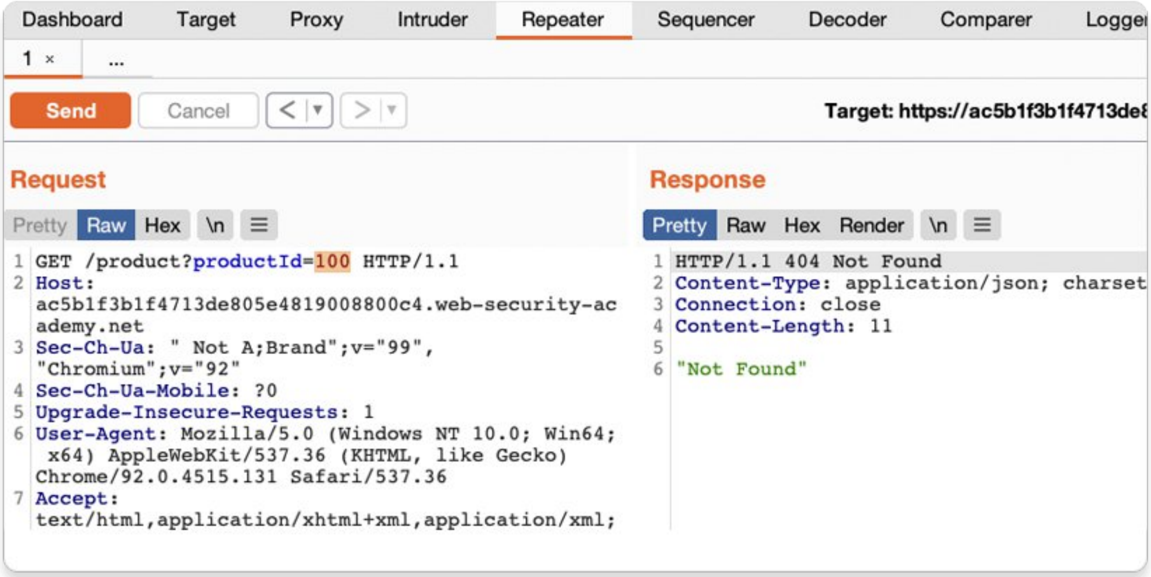
Burp Suite is a web application security testing platform. It provides manual and automated tools to help cybersecurity professionals and developers identify vulnerabilities in web applications.

Developed by PortSwigger, Burp Suite integrates into the testing process, offering a suite of modular tools for tasks such as scanning, crawling, and analysis. This platform is useful for both manual and automated testing, offering flexibility and integration capabilities. It supports numerous extensions, which allow users to tailor the suite to meet project needs.

By resending the same request with different input each time, you can identify and confirm a variety of input-based vulnerabilities. This is one of the most common tasks you will perform during manual testing with Burp Suite.

Step 1: Resend the request with different input

Change the number in the productId parameter and resend the request. Try this with a few arbitrary numbers, including a couple of larger ones.



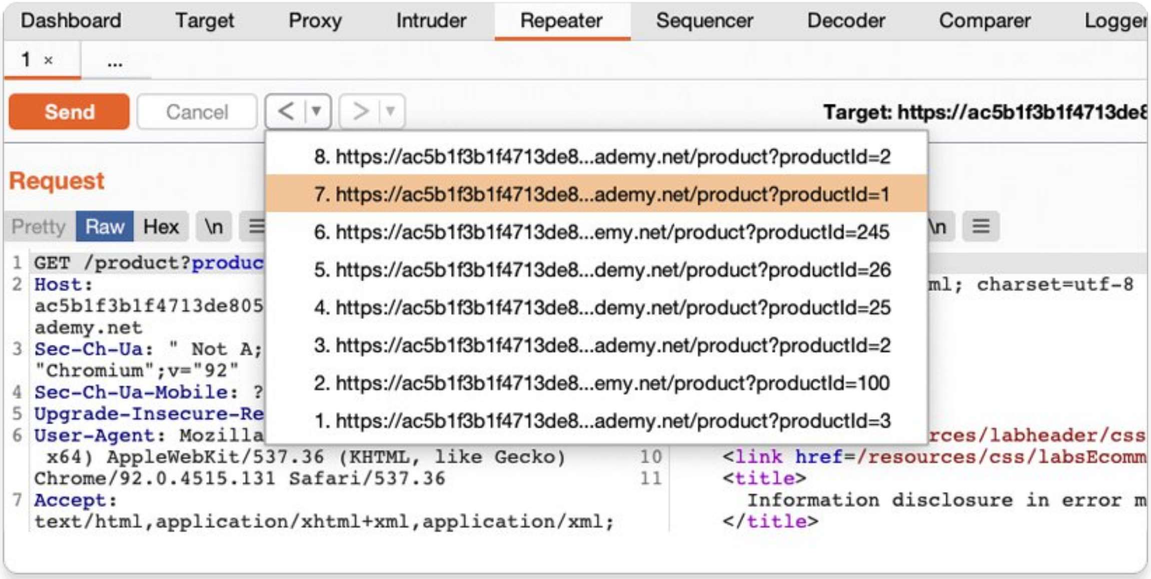
```
GET /product?productId=100 HTTP/1.1
Host: ac5b1f3b1f4713de805e4819008800c4.web-security-academy.net
Sec-Ch-Ua: " Not A;Brand";v="99", "Chromium";v="92"
Sec-Ch-Ua-Mobile: ?0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/92.0.4515.131 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;
```

```
HTTP/1.1 404 Not Found
Content-Type: application/json; charset
Connection: close
Content-Length: 11

"Not Found"
```

Step 2: View the request history

Use the arrows to step back and forth through the history of requests that you've sent, along with their matching responses. The drop-down menu next to each arrow also lets you jump to a specific request in the history.



Target: https://ac5b1f3b1f4713de8

8. https://ac5b1f3b1f4713de8...ademy.net/product?productId=2
7. https://ac5b1f3b1f4713de8...ademy.net/product?productId=1
6. https://ac5b1f3b1f4713de8...emy.net/product?productId=245
5. https://ac5b1f3b1f4713de8...demy.net/product?productId=26
4. https://ac5b1f3b1f4713de8...demy.net/product?productId=25
3. https://ac5b1f3b1f4713de8...ademy.net/product?productId=2
2. https://ac5b1f3b1f4713de8...emy.net/product?productId=100
1. https://ac5b1f3b1f4713de8...ademy.net/product?productId=3

```
GET /product?produc
Host:
ac5b1f3b1f4713de805
ademy.net
Sec-Ch-Ua: " Not A;
"Chromium";v="92"
Sec-Ch-Ua-Mobile: ?
Upgrade-Insecure-Re
User-Agent: Mozilla
 x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4515.131 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;
```

```
ml; charset=utf-8
rces/labheader/css
<link href=/resources/css/labsEcomm
<title>
  Information disclosure in error m
</title>
```

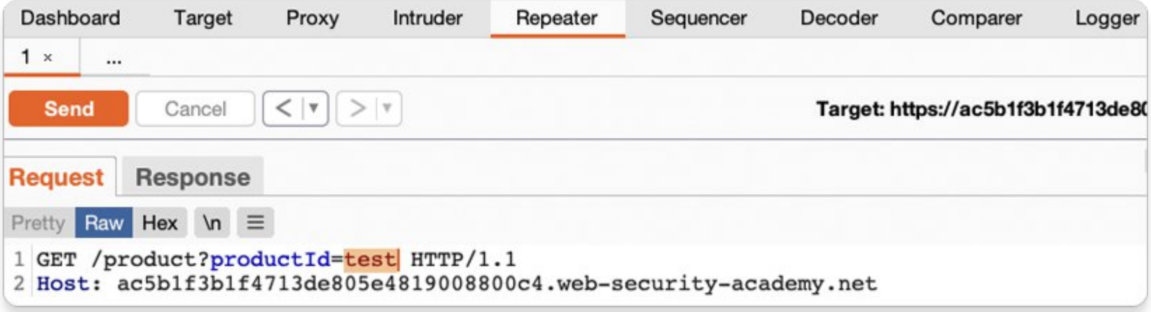
This is useful for returning to previous requests that you've sent in order to investigate a particular input further.

Compare the content of the responses, notice that you can successfully request different product pages by entering their ID, but receive a Not Found response if the server was unable to find a product with the given ID. Now we know how this page is supposed to work, we can use Burp Repeater to see how it responds to unexpected input.

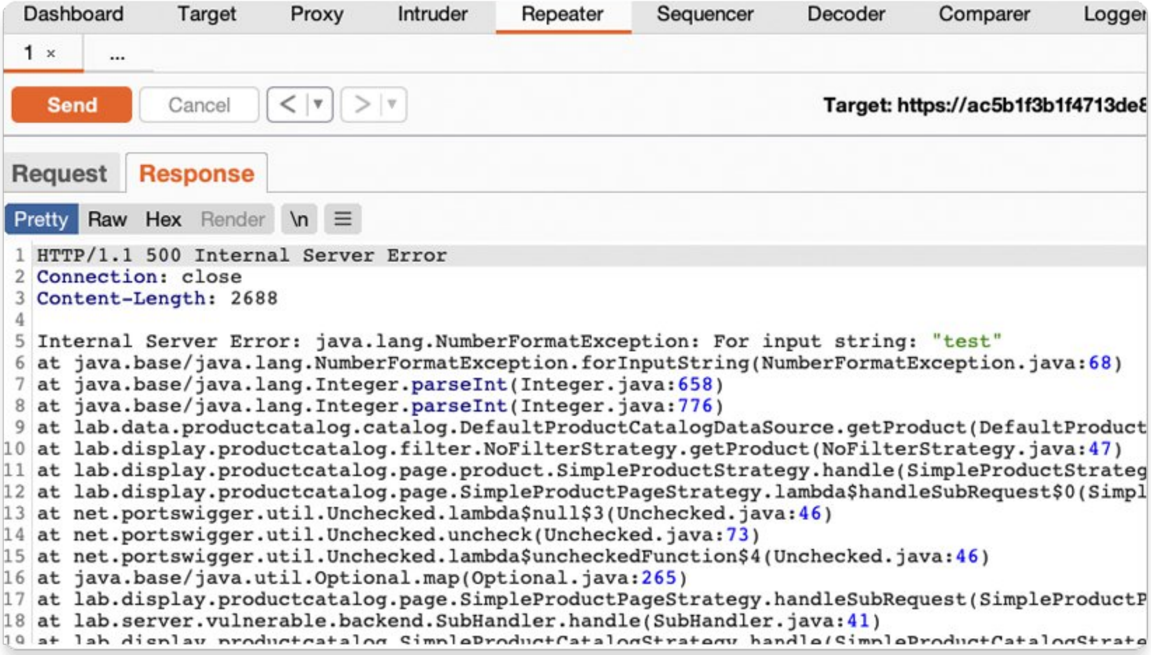
Step 3: Try sending unexpected input

The server seemingly expects to receive an integer value via this productId parameter. Let's see what happens if we send a different data type.

Send another request where the productId is a string of characters.



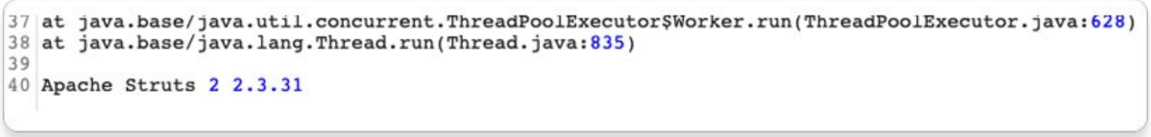
```
GET /product?productId=test HTTP/1.1
Host: ac5b1f3b1f4713de805e4819008800c4.web-security-academy.net
```



```
HTTP/1.1 500 Internal Server Error
Connection: close
Content-Length: 2688

Internal Server Error: java.lang.NumberFormatException: For input string: "test"
at java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:68)
at java.base/java.lang.Integer.parseInt(Integer.java:658)
at java.base/java.lang.Integer.parseInt(Integer.java:776)
at lab.data.productcatalog.catalog.DefaultProductCatalogDataSource.getProduct(DefaultProduct
at lab.display.productcatalog.filter.NoFilterStrategy.getProduct(NoFilterStrategy.java:47)
at lab.display.productcatalog.page.product.SimpleProductStrategy.handle(SimpleProductStrateg
at lab.display.productcatalog.page.SimpleProductPageStrategy.lambda$handleSubRequest$0(Simpl
at net.portswigger.util.Unchecked.lambda$null$3(Unchecked.java:46)
at net.portswigger.util.Unchecked.uncheck(Unchecked.java:73)
at net.portswigger.util.Unchecked.lambda$uncheckedFunction$4(Unchecked.java:46)
at java.base/java.util.Optional.map(Optional.java:265)
at lab.display.productcatalog.page.SimpleProductPageStrategy.handleSubRequest(SimpleProductP
at lab.server.vulnerable.backend.SubHandler.handle(SubHandler.java:41)
at lab.display.productcatalog.SimpleProductCatalogStrategy.handle(SimpleProductCatalogStrate
```

Notice that the response tells you that the website is using the Apache Struts framework - it even reveals which version.



```
at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:628)
at java.base/java.lang.Thread.run(Thread.java:835)

Apache Struts 2 2.3.31
```

In a real scenario, this kind of information could be useful to an attacker, especially if the named version is known to contain additional vulnerabilities.

Learning Outcome: Learned to manually perform requests manipulation using Burp Repeater and Intruder.

Conclusion: ……………………………………………………………………………
……………………………………………………………………………………………………
……………………………………………………………………………………………………
……

Name:

Class: BE-CSE

Roll No.:

For Faculty Use

Correction Parameters	**Formative Assessment [40%]**	**Timely completion of Practical [40%]**	**Attendance / Learning Attitude [20%]**	
Marks Obtained				