Lab 2

Chetan Sahrudhai Kimidi - ckimidi

***Abstract*—Exploring HTTP, experimenting with Zap, and additionally working with inotify-tools.**

I. Introduction

HTTP, the Hypertext Transfer Protocol, its basics and proxies, the concepts of CIA and proper usage of browser supplied developer tools is what I worked on, in this lab. Additionally, I solved the bonus tasks of decrypting a message and using the inotify-tools library.

II. WebGoat - General and Crypto Sections

A. HTTP Basics

Here, I learnt how HTTP functions. The first challenge was the illustration of a basic HTTP request, which simply reversed the input I entered. The main challenge was to determine the type of request and a certain magic number. As seen in Fig. 1, I solved it using Inspect Element.

B. HTTP Proxies

I solved the challenge successfully, as seen in Fig. 2, which was to intercept and modify a request, as instructed. Initially, I had followed all the previous instructions such as filtering all POST requests, setting a particular break-point and enabling it. Then, I switched on the green circle (Intercept), pressed the submit button on the browser, found the request, modified 3 lines as instructed and then resent it.

C. Developer Tools

The Developer tools such as the Inspect, Sources, Console and Network tabs, were elaborated here. I solved both challenges, as seen in Fig. 3 and Fig. 4.

D. CIA Triad

This subsection elaborated about the three security fundamentals of confidentiality, integrity and availability. I solved the quiz correctly, as referenced in [2].

E. Crypto Basics - Steps 1,2,3,4 and 8

The first challenge was about Base64 Encoding, which can be seen in Fig. 5. Next, as seen in Fig. 6, this challenge focused on finding a password from a XOR encoded string. The last challenge was Fig. 7. Step-8 of Crypto Basics was solved as seen in Fig. 10.

III. ZAP and FoxyProxy

I have already installed and used ZAP in the first section. As seen in Fig. 8, I have also used ZAP to find out the magic number for the quiz in step 3 of HTTP Basics. I clicked on Go in the browser, found the request in ZAP, scanned the body of the request to get to know the magic number.

I also added the FoxyProxy Standard extension to FireFox, and learnt to modify the browser's proxy settingss, as seen in Fig. 9.

IV. inotify-tools

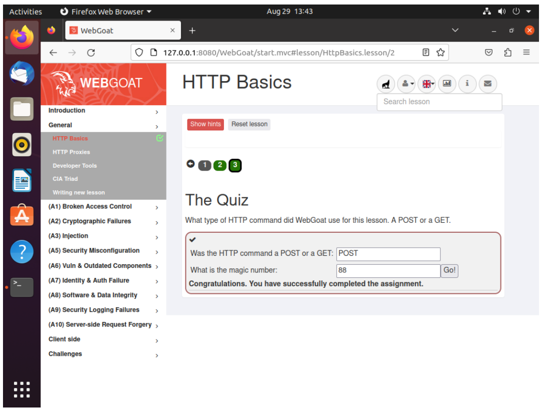
This library of tools is useful for monitoring events. To demonstrate the usage of inotifywait and inotifywatch, I have monitored the /root folder. Both commands basically establish watches and monitor the specified file/directory, but the key difference is inotifywait will terminate once any event takes place, whereas inotifywatch will continuously monitor and collect statistics, until terminated by the user or by a specified timeout. This can be seen in Fig. 11.

V. Conclusion

In summary, I successfully completed all the General section tasks, setup and used ZAP to intercept requests, and monitored /root using inotify-tools.

References

[1] A reference video link for inotify-tools setup
[2] A folder of screenshots taken in Lab-2


Activities
Firefox Web Browser
Aug 29 13:43
WebGoat
127.0.0.1:8080/WebGoat/start.mvc#lesson/HttpBasics.lesson/2
WEBGOAT
Introduction
General
HTTP Basics
HTTP Proxies
Developer Tools
CIA Triad
Writing new lesson
(A1) Broken Access Control
(A2) Cryptographic Failures
(A3) Injection
(A5) Security Misconfiguration
(A6) Vuln & Outdated Components
(A7) Identity & Auth Failure
(A8) Software & Data Integrity
(A9) Security Logging Failures
(A10) Server-side Request Forgery
Client side
Challenges
Search lesson
Show hints
Reset lesson
1 2 3
Go!

HTTP Basics

The Quiz

What type of HTTP command did WebGoat use for this lesson. A POST or a GET.

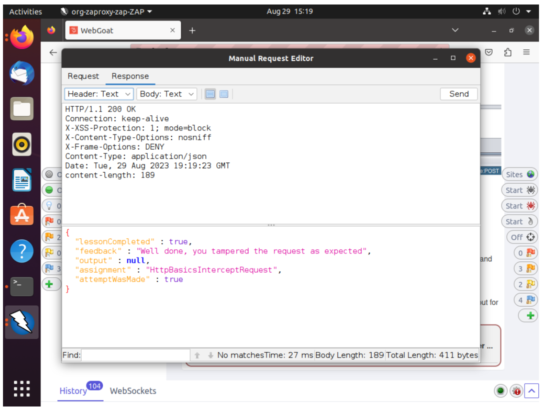
✓

Was the HTTP command a POST or a GET: POST

What is the magic number: 88

Congratulations. You have successfully completed the assignment.

Fig. 1. Magic number found!

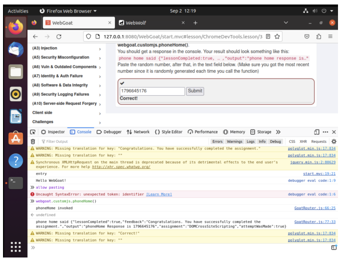


```
HTTP/1.1 200 OK
Connection: keep-alive
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Content-Type: application/json
Date: Tue, 29 Aug 2023 19:19:23 GMT
content-length: 189
```

```
{
  "lessonCompleted" : true,
  "feedback" : "Well done, you tampered the request as expected",
  "output" : null,
  "assignment" : "HttpBasicsInterceptRequest",
  "attemptWasMade" : true
}
```

Find: No matchesTime: 27 ms Body Length: 189 Total Length: 411 bytes

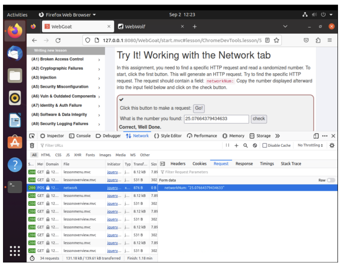
Fig. 2. HTTP Proxies Challenge done!



webgoat.customjs.phoneHome().
You should get a response in the console. Your result should look something like this:
`phone home said {"lessonCompleted:true, … ,"output":"phone home response is.."`
Paste the random number, after that, in the text field below. (Make sure you got the most recent number since it is randomly generated each time you call the function)

✓
1796645176 Submit
Correct!

Fig. 3. Random number found!



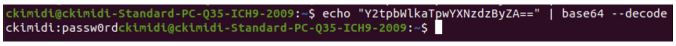
Try It! Working with the Network tab

In this assignment, you need to find a specific HTTP request and read a randomized number. To start, click the first button. This will generate an HTTP request. Try to find the specific HTTP request. The request should contain a field: `networkNum:` Copy the number displayed afterward into the input field below and click on the check button.

✓
Click this button to make a request: Go!
What is the number you found: 25.07664379434633 check
Correct, Well Done.

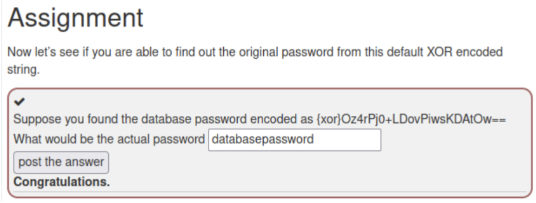
Form data
networkNum: "25.07664379434633"

Fig. 4. Network number found!



```
ckimidi@ckimidi-Standard-PC-Q35-ICH9-2009:~$ echo "Y2tpbWlkaTpwYXNzdzByZA==" | base64 --decode
ckimidi:passw0rdckimidi@ckimidi-Standard-PC-Q35-ICH9-2009:~$
```

Fig. 5. Base64 Encoding

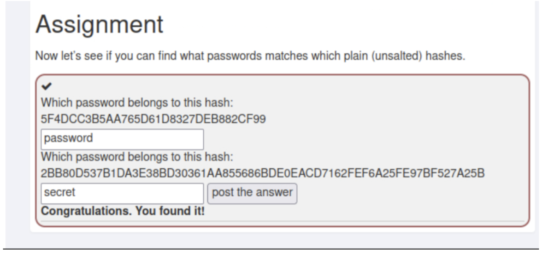


Assignment

Now let's see if you are able to find out the original password from this default XOR encoded string.

✓
Suppose you found the database password encoded as {xor}Oz4rPj0+LDovPiwsKDAtOw==
What would be the actual password databasepassword
post the answer
Congratulations.

Fig. 6. XOR decode

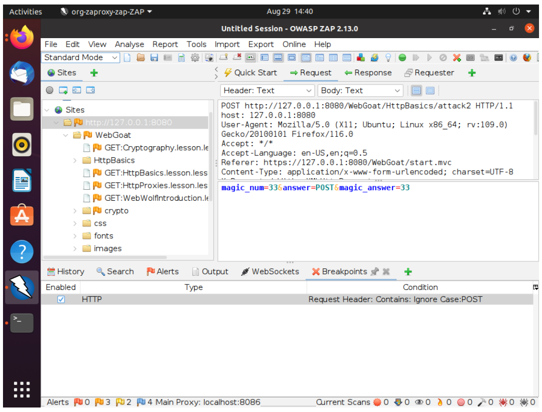


Assignment

Now let's see if you can find what passwords matches which plain (unsalted) hashes.

✓
Which password belongs to this hash:
5F4DCC3B5AA765D61D8327DEB882CF99
password
Which password belongs to this hash:
2BB80D537B1DA3E38BD30361AA855686BDE0EACD7162FEF6A25FE97BF527A25B
secret post the answer
Congratulations. You found it!

Fig. 7. Passwords to hashes

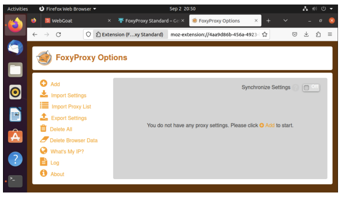


```
POST http://127.0.0.1:8080/WebGoat/HttpBasics/attack2 HTTP/1.1
host: 127.0.0.1:8080
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0)
Gecko/20100101 Firefox/116.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Referer: https://127.0.0.1:8080/WebGoat/start.mvc
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
```

```
magic_num=33&answer=POST&magic_answer=33
```

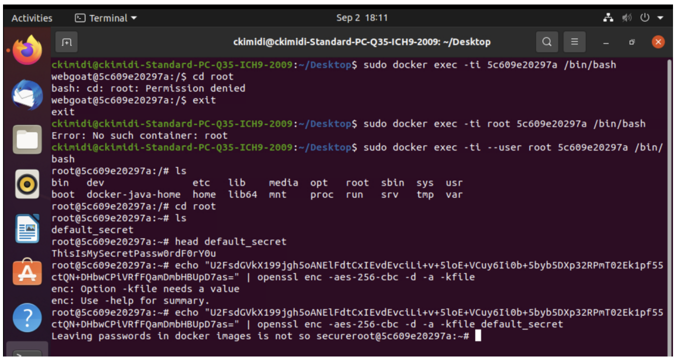
Enabled	Type	Condition
☑	HTTP	Request Header: Contains: Ignore Case:POST

Fig. 8. Using ZAP to solve HTTP Basics step 3 quiz



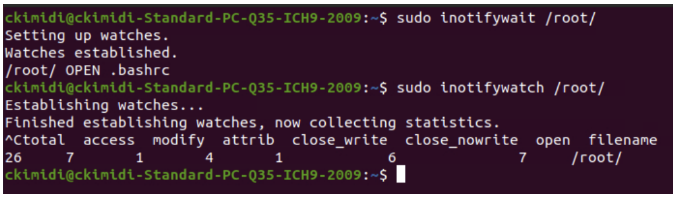
You do not have any proxy settings. Please click Add to start.

Fig. 9. FoxyProxy



```
ckimidi@ckimidi-Standard-PC-Q35-ICH9-2009:~/Desktop$ sudo docker exec -ti 5c609e20297a /bin/bash
webgoat@5c609e20297a:/$ cd root
bash: cd: root: Permission denied
webgoat@5c609e20297a:/$ exit
exit
ckimidi@ckimidi-Standard-PC-Q35-ICH9-2009:~/Desktop$ sudo docker exec -ti root 5c609e20297a /bin/bash
Error: No such container: root
ckimidi@ckimidi-Standard-PC-Q35-ICH9-2009:~/Desktop$ sudo docker exec -ti --user root 5c609e20297a /bin/
bash
root@5c609e20297a:/# ls
bin   dev                etc   lib    media  opt   root  sbin  sys  usr
boot  docker-java-home   home  lib64  mnt    proc  run   srv   tmp  var
root@5c609e20297a:/# cd root
root@5c609e20297a:~# ls
default_secret
root@5c609e20297a:~# head default_secret
ThisIsMySecretPassw0rdF0rY0u
root@5c609e20297a:~# echo "U2FsdGVkX199jgh5oANElFdtCxIEvdEvciLi+v+5loE+VCuy6Ii0b+5byb5DXp32RPmT02Ek1pf55
ctQN+DHbwCPiVRfFQamDmbHBUpD7as=" | openssl enc -aes-256-cbc -d -a -kfile
enc: Option -kfile needs a value
enc: Use -help for summary.
root@5c609e20297a:~# echo "U2FsdGVkX199jgh5oANElFdtCxIEvdEvciLi+v+5loE+VCuy6Ii0b+5byb5DXp32RPmT02Ek1pf55
ctQN+DHbwCPiVRfFQamDmbHBUpD7as=" | openssl enc -aes-256-cbc -d -a -kfile default_secret
Leaving passwords in docker images is not so secureroot@5c609e20297a:~#
```

Fig. 10. Crypto Basics step 8



```
ckimidi@ckimidi-Standard-PC-Q35-ICH9-2009:~$ sudo inotifywait /root/
Setting up watches.
Watches established.
/root/ OPEN .bashrc
ckimidi@ckimidi-Standard-PC-Q35-ICH9-2009:~$ sudo inotifywatch /root/
Establishing watches...
Finished establishing watches, now collecting statistics.
^Ctotal  access  modify  attrib  close_write  close_nowrite  open  filename
26       7       1       4       1            6              7     /root/
ckimidi@ckimidi-Standard-PC-Q35-ICH9-2009:~$
```

Fig. 11. inotifywait and inotifywatch