CCCCO CySA+ Lab Series

Lab 2: Web Application Scanning

Document Version: 2019-09-06

Contents

Introduction 3
Objectives 3
Lab Topology 4
Lab Settings 5
1 Scanning a Website for Vulnerabilities 6
2 Investigating Website Attack Techniques 11

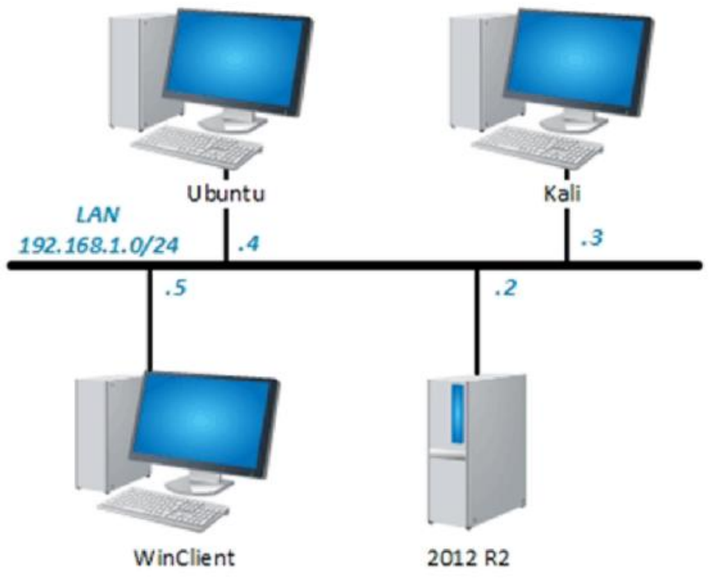
Introduction

Web application penetration testing is a complete field within the penetration testing discipline. All of the action takes place at the application level. Many of the same types of tactics that are used for a general penetration test also apply to web application testing. In this lab, you will be using the Kali machine to attack the Ubuntu machine.

Objectives

- Scan websites for vulnerabilities with Nikto
- Scan websites for vulnerabilities with ZAP

Lab Topology



Lab Settings

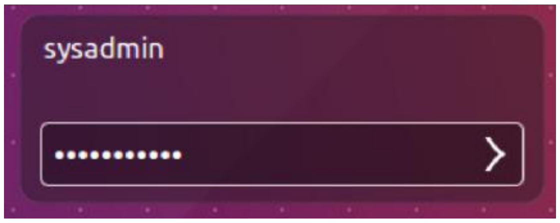
The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account	Password
2012 R2	192.168.1.2	Administrator	Password123
WinClient	192.168.1.5	student	Password123
Kali	192.168.1.3	root	toor
Ubuntu	192.168.1.4	sysadmin	Password123

1 Scanning a Website for Vulnerabilities

This lab will make use of two pieces of software in order to scan a website for potential vulnerabilities. The first piece of software is Nikto, which is useful for scanning web server misconfigurations and rogue files. The other piece of software is W3af, which is a comprehensive web application vulnerability scanner.

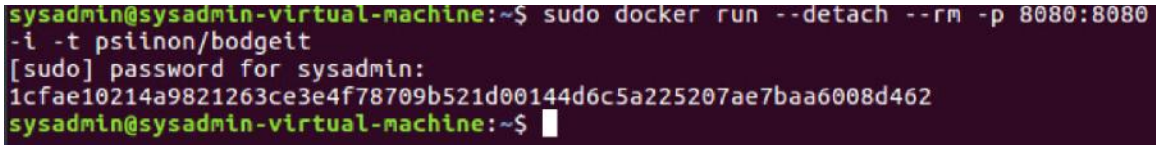
1. Launch the **Ubuntu** machine to access the graphical login screen.
2. Log in as `sysadmin` using the password `Password123`.



3. Once logged in, open a **Terminal** window.

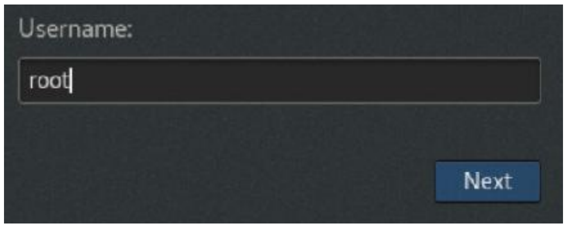
4. Begin by starting the *Bodgeit* website inside a docker container. A docker container is a form of virtualization that utilizes the OS in order to allow software to run inside of an isolated, virtual instance in any Linux environment. In order to start the *Bodgeit* docker container, enter the following command, using the password **Password123** when prompted:

```
sysadmin@sysadmin-virtual-machine:~$ sudo docker run --detach --rm -p
8080:8080 -i -t psiinon/bodgeit
```



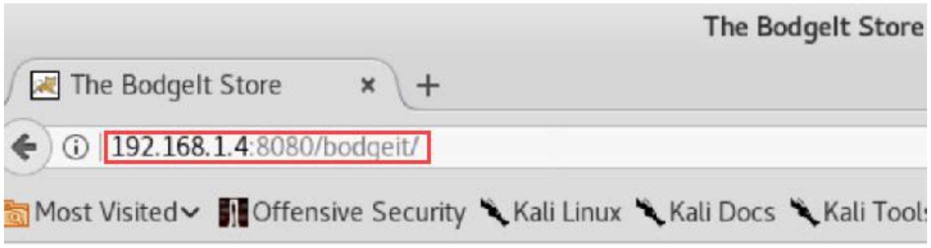
```
sysadmin@sysadmin-virtual-machine:~$ sudo docker run --detach --rm -p 8080:8080
-i -t psiinon/bodgeit
[sudo] password for sysadmin:
1cfae10214a9821263ce3e4f78709b521d00144d6c5a225207ae7baa6008d462
sysadmin@sysadmin-virtual-machine:~$ 
```

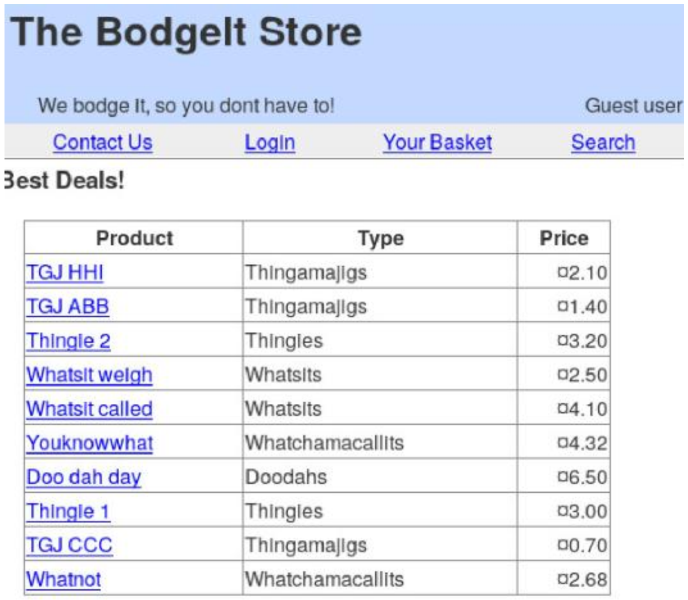
5. Launch the **Kali** machine to access the graphical login screen.
6. Press **ENTER** to bring up the log in screen. Log in as **root** using the password **toor**.



7. Open the **Firefox** application.

8. In order to ensure that the application is correctly running on the target, in the URL bar, navigate to **http://192.168.1.4:8080/bodgeit/**. Confirm that the website has successfully loaded. You may then close the **Firefox** window.




The BodgeIt Store

We bodge It, so you dont have to!

Guest user

Contact Us Login Your Basket Search

Best Deals!

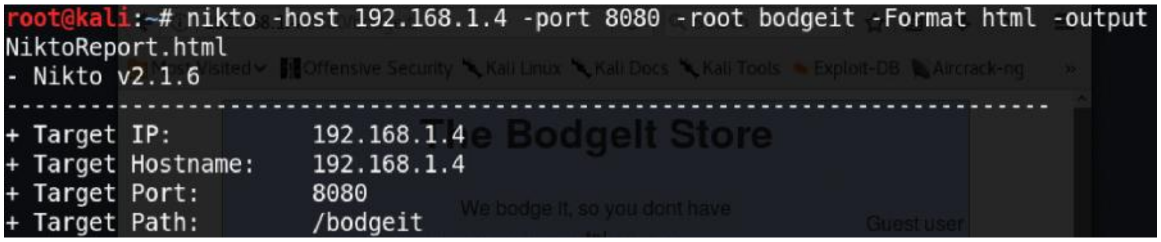
Product	Type	Price
TGJ HHI	Thingamajigs	¤2.10
TGJ ABB	Thingamajigs	¤1.40
Thingle 2	Thingles	¤3.20
WhatsIt weigh	WhatsIts	¤2.50
WhatsIt called	WhatsIts	¤4.10
Youknowwhat	Whatchamacallits	¤4.32
Doo dah day	Doodahs	¤6.50
Thingle 1	Thingles	¤3.00
TGJ CCC	Thingamajigs	¤0.70
Whatnot	Whatchamacallits	¤2.68

Begin by using *Nikto* to scan the *Bodgeit* site. *Nikto* is used to test for website misconfigurations that could allow an attacker to compromise the web server. Once you have finished, an .html file reporting any problems or vulnerabilities that were discovered will be generated.

9. Open a **Terminal** window.

10. You will now use *Nikto* to test the *Bodgeit* site for website misconfigurations. Do so by entering the following command:

```
root@kali:~# nikto -host 192.168.1.4 -port 8080 -root bodgeit -Format htm -output NiktoReport.html
```



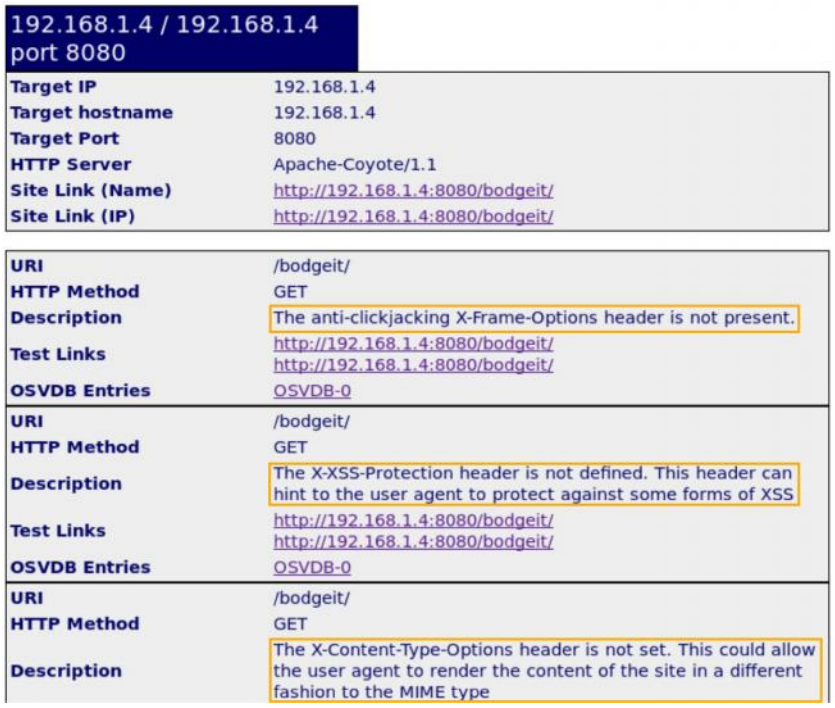
```
root@kali:~# nikto -host 192.168.1.4 -port 8080 -root bodgeit -Format html -output
NiktoReport.html
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.1.4
+ Target Hostname:    192.168.1.4
+ Target Port:        8080
+ Target Path:        /bodgeit
```

11. In order to view the saved report in *Firefox*, enter the following command:

```
root@kali:~# firefox NiktoReport.html
```

```
root@kali:~# firefox NiktoReport.html
```

12. You will see three issues found by *Nikto*. A *Google* search will allow you to elaborate on the vulnerabilities and how to fix them. Close the **Firefox** window.



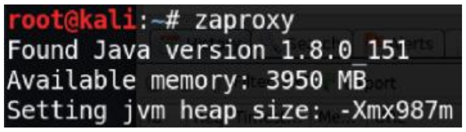
192.168.1.4 / 192.168.1.4
port 8080

Target IP	192.168.1.4
Target hostname	192.168.1.4
Target Port	8080
HTTP Server	Apache-Coyote/1.1
Site Link (Name)	http://192.168.1.4:8080/bodgeit/
Site Link (IP)	http://192.168.1.4:8080/bodgeit/

URI	/bodgeit/
HTTP Method	GET
Description	The anti-clickjacking X-Frame-Options header is not present.
Test Links	http://192.168.1.4:8080/bodgeit/ http://192.168.1.4:8080/bodgeit/
OSVDB Entries	OSVDB-0
URI	/bodgeit/
HTTP Method	GET
Description	The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
Test Links	http://192.168.1.4:8080/bodgeit/ http://192.168.1.4:8080/bodgeit/
OSVDB Entries	OSVDB-0
URI	/bodgeit/
HTTP Method	GET
Description	The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type

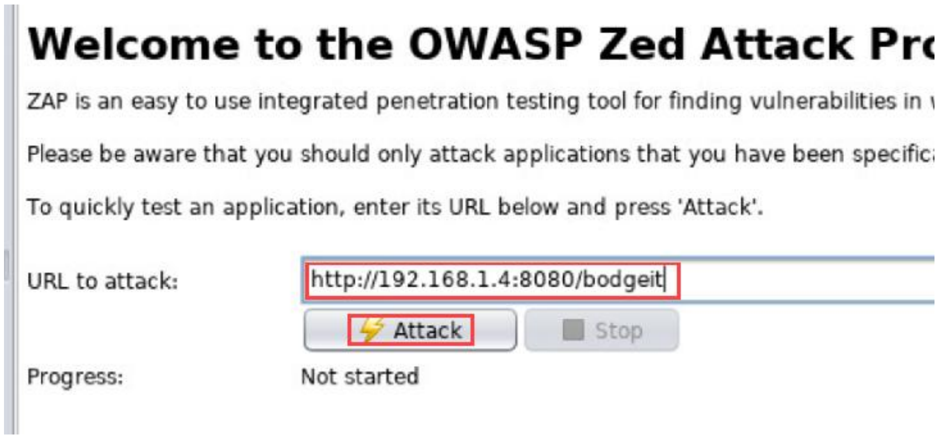
13. Now, you will use *OWASP ZAP (Zed Attack Proxy)* to scan the *Bodgeit* site. *ZAP* is one of the most popular free web security tools. Not only can it help to find security vulnerabilities in your web applications automatically, but experienced penetration testers can use the program for manual security testing, as well. In order to launch *ZAP*, navigate back to the **Terminal** window and enter the following command:

```
root@kali:~# zaproxy
```



```
root@kali:~# zaproxy
Found Java version 1.8.0_151
Available memory: 3950 MB
Setting jvm heap size: -Xmx987m
```

14. Once the GUI loads, enter the target URL of **http://192.168.1.4:8080/bodgeit/** and click **Attack**.



Welcome to the OWASP Zed Attack Pro

ZAP is an easy to use integrated penetration testing tool for finding vulnerabilities in

Please be aware that you should only attack applications that you have been specific

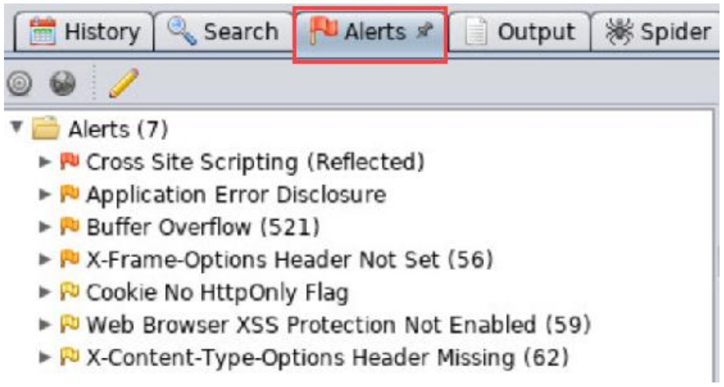
To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://192.168.1.4:8080/bodgeit

Progress: Not started

Please note that this scan will take approximately 7 minutes, as it is running around 80,000 tests.

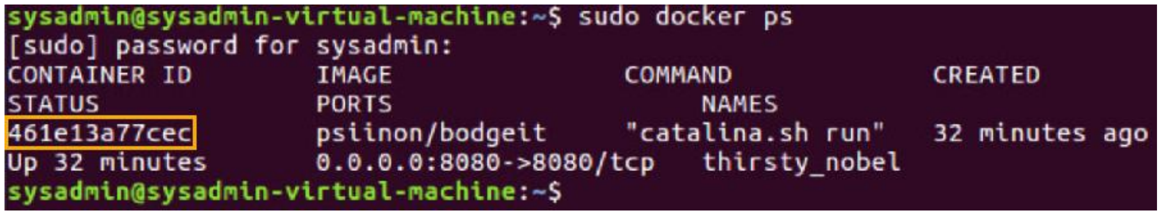
15. Once the scan is finished, review the scan results by navigating to the **Alerts** tab in the lower-left corner. After selecting a vulnerability, a description can be found in the *Description* section of the lower-right pane. Once you have reviewed the vulnerabilities, close the *OWASP* window.



Alerts (7)
- Cross Site Scripting (Reflected)
- Application Error Disclosure
- Buffer Overflow (521)
- X-Frame-Options Header Not Set (56)
- Cookie No HttpOnly Flag
- Web Browser XSS Protection Not Enabled (59)
- X-Content-Type-Options Header Missing (62)

16. In order to begin the next part of the lab, the *Bodgeit* website must be shut down. To do this, return to the **Ubuntu** machine.
17. In the **Terminal** window, enter the following command in order to obtain the *Container ID*. Use the password *Password123* if prompted.

```
sysadmin@sysadmin-virtual-machine:~$ sudo docker ps
```

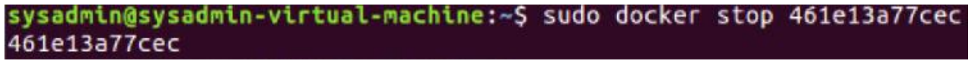


```
sysadmin@sysadmin-virtual-machine:~$ sudo docker ps
[sudo] password for sysadmin:
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS              PORTS                    NAMES
461e13a77cec        psiinon/bodgeit     "catalina.sh run"   32 minutes ago
Up 32 minutes       0.0.0.0:8080->8080/tcp   thirsty_nobel
sysadmin@sysadmin-virtual-machine:~$
```

Note the *Container ID* for *Bodgeit*. It will be different every time the program is run.

18. In order to shut down the *Bodgeit* container, enter the following command:

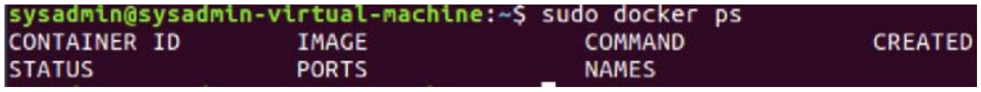
```
sysadmin@sysadmin-virtual-machine:~$ sudo docker stop <containerid>
```



```
sysadmin@sysadmin-virtual-machine:~$ sudo docker stop 461e13a77cec
461e13a77cec
```

19. To confirm that the docker was stopped, run the following command:

```
sysadmin@sysadmin-virtual-machine:~$ sudo docker ps
```



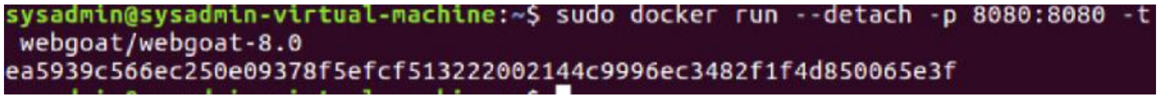
```
sysadmin@sysadmin-virtual-machine:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED
STATUS              PORTS               NAMES
```

2 Investigating Website Attack Techniques

Now that you have discovered several web vulnerabilities with *Nikto* and *OWASP ZAP*, you can explore how to exploit these vulnerabilities. This will give better insight into how common coding problems are exploited.

1. In order to start the *WebGoat* docker container, enter the following command:

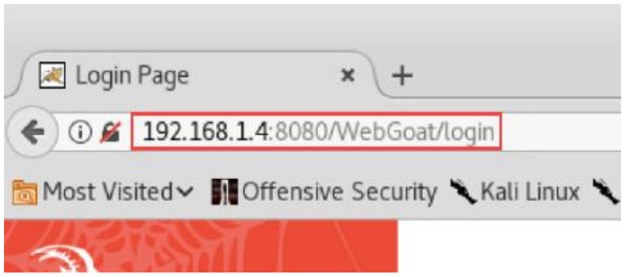
```
sysadmin@sysadmin-virtual-machine:~$ sudo docker run --detach -p 8080:8080 –
t webgoat/webgoat-8.0
```



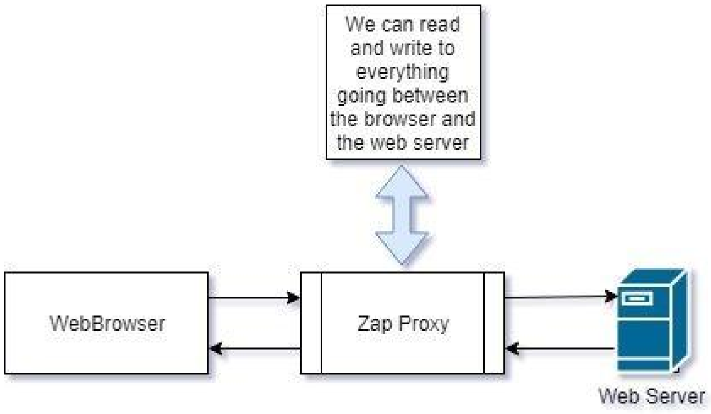
```
sysadmin@sysadmin-virtual-machine:~$ sudo docker run --detach -p 8080:8080 -t
 webgoat/webgoat-8.0
ea5939c566ec250e09378f5efcf513222002144c9996ec3482f1f4d850065e3f
```

2. Navigate to the **Kali** machine.
3. Open a **Firefox** window.

4. In the URL bar, navigate to **http://192.168.1.4:8080/WebGoat/**
   Please note that the web address is case-sensitive.

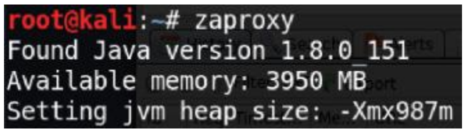


A web proxy sits between your browser and a web server/service and allows you to monitor and alter the requests and responses from that website. This allows you to see hidden fields in a request. For the first example, you will be using *ZAP* to serve as web proxy software.


We can read and write to everything going between the browser and the web server
WebBrowser
Zap Proxy
Web Server

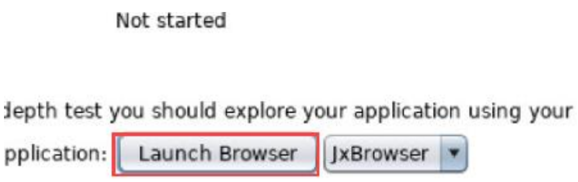
5. Return to the **Terminal** window. In order to launch **ZAP**, enter the following command:

```
root@kali:~# zaproxy
```



```
root@kali:~# zaproxy
Found Java version 1.8.0_151
Available memory: 3950 MB
Setting jvm heap size: -Xmx987m
```

6. Once the GUI appears, click on the **Launch Browser** button. This button can be found in the upper-right window pane.

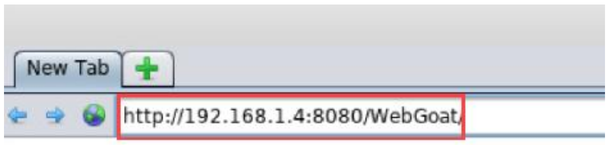


Not started

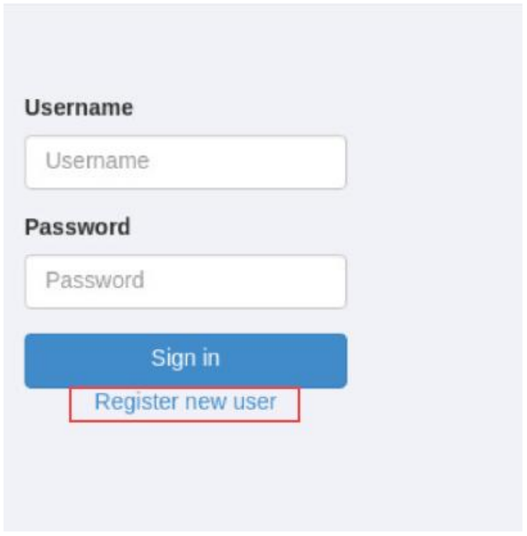
depth test you should explore your application using your

pplication:

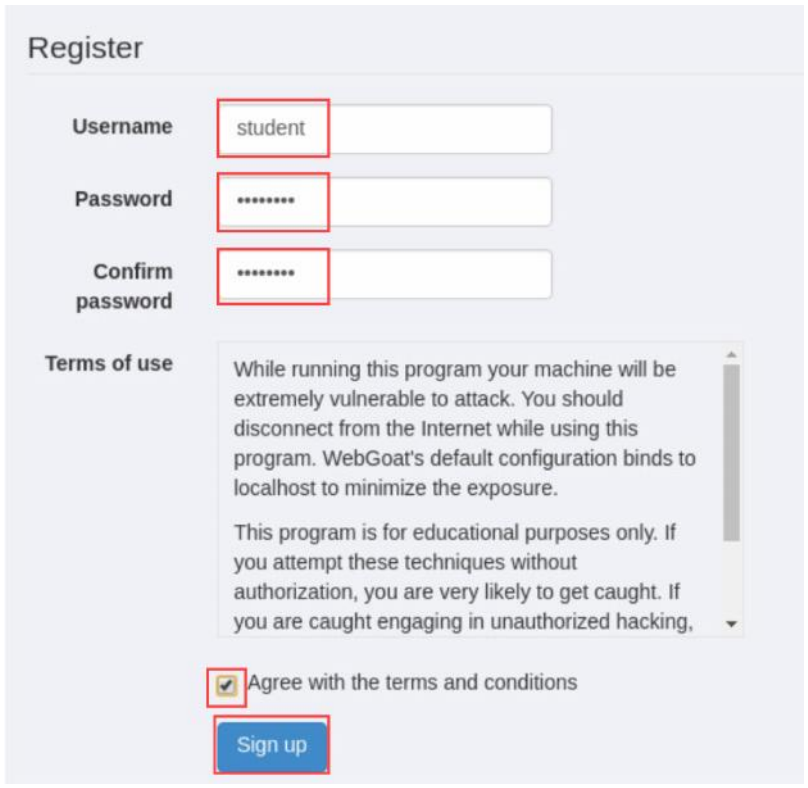
7. In the URL window of the launched browser, navigate to **http://192.168.1.4:8080/WebGoat**
Please note that the web address is case-sensitive.



8. Underneath **Sign in**, click on **Register new user**.



9. Set the username to **student** and the password to **password**, then click on **Agree with the terms and conditions**, and **Sign up**.



Register

Username: student

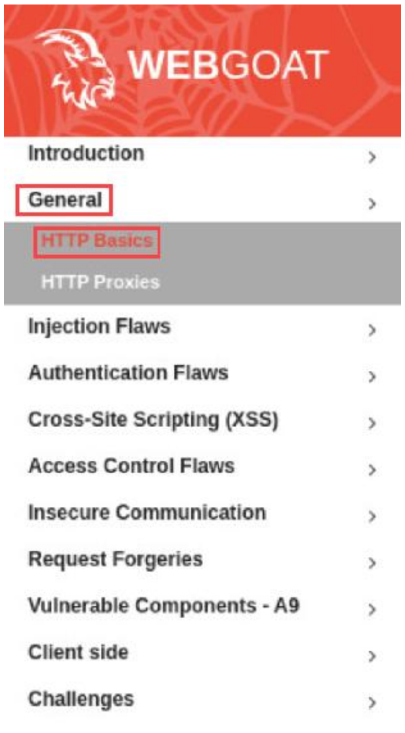
Password: ••••••••

Confirm password: ••••••••

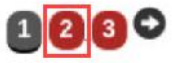
Terms of use: While running this program your machine will be extremely vulnerable to attack. You should disconnect from the Internet while using this program. WebGoat's default configuration binds to localhost to minimize the exposure.

This program is for educational purposes only. If you attempt these techniques without authorization, you are very likely to get caught. If you are caught engaging in unauthorized hacking,

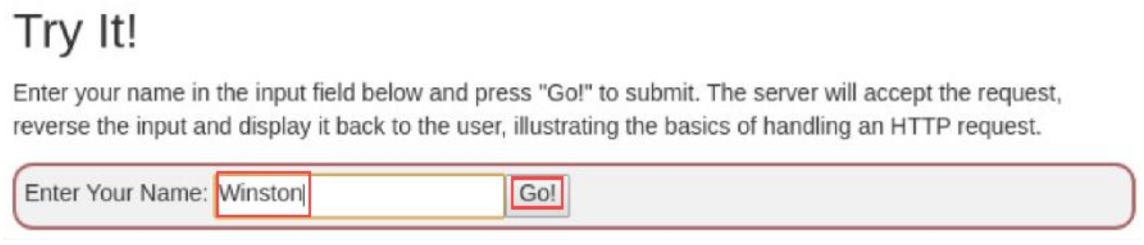
10. In order to begin a lesson on using a web proxy, click **General**-> **HTTP Basics** on the left-hand side of the screen.



11. Read through the first page. Navigate to the second page by clicking the number **2** button on the *HTTP Basics* screen.



12. On the second page, enter your name and click **Go**!

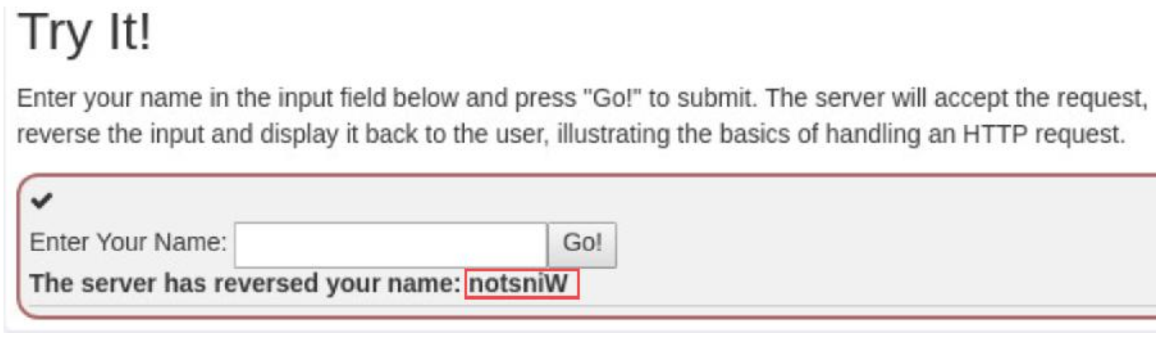


Try It!

Enter your name in the input field below and press "Go!" to submit. The server will accept the request, reverse the input and display it back to the user, illustrating the basics of handling an HTTP request.

Enter Your Name: Winston

13. This option reverses the inputted name and sends it back to you.



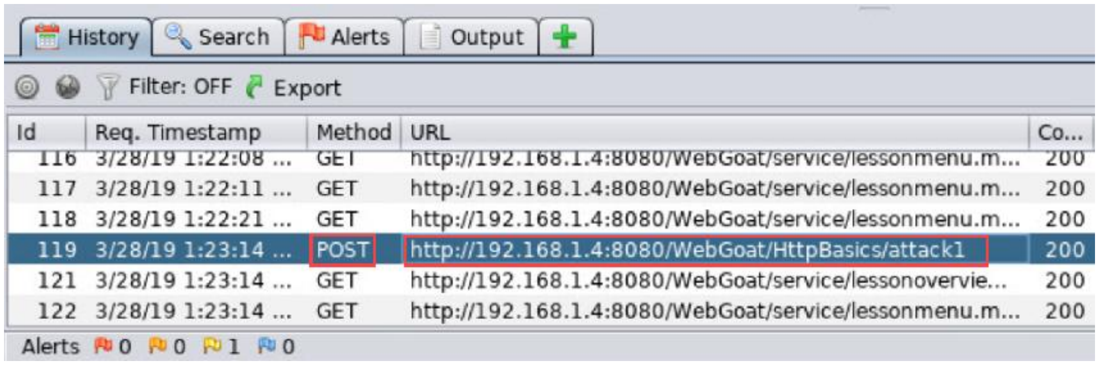
Try It!

Enter your name in the input field below and press "Go!" to submit. The server will accept the request, reverse the input and display it back to the user, illustrating the basics of handling an HTTP request.

Enter Your Name:

The server has reversed your name: notsniW

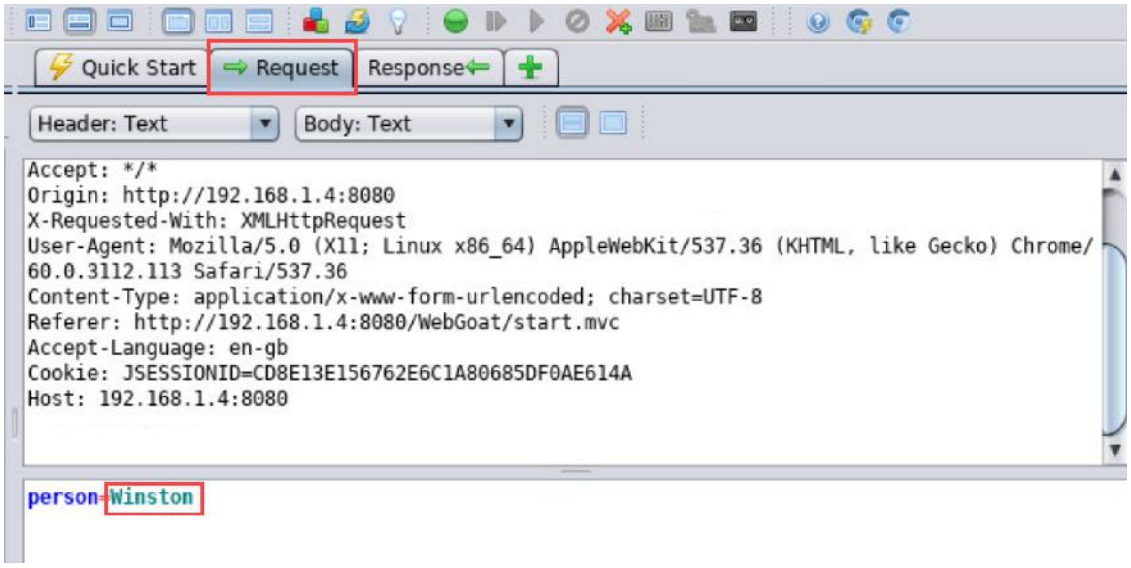
14. In order to see what this transaction looks like, return to the proxy window. In the bottom panel, select the entry in the *History* tab with the line that ends with "**attack1**". Notice under the *Method* column that this is an *HTML POST* request.



Id	Req. Timestamp	Method	URL	Co...
116	3/28/19 1:22:08 ...	GET	http://192.168.1.4:8080/WebGoat/service/lessonmenu.m...	200
117	3/28/19 1:22:11 ...	GET	http://192.168.1.4:8080/WebGoat/service/lessonmenu.m...	200
118	3/28/19 1:22:21 ...	GET	http://192.168.1.4:8080/WebGoat/service/lessonmenu.m...	200
119	3/28/19 1:23:14 ...	POST	http://192.168.1.4:8080/WebGoat/HttpBasics/attack1	200
121	3/28/19 1:23:14 ...	GET	http://192.168.1.4:8080/WebGoat/service/lessonovervie...	200
122	3/28/19 1:23:14 ...	GET	http://192.168.1.4:8080/WebGoat/service/lessonmenu.m...	200

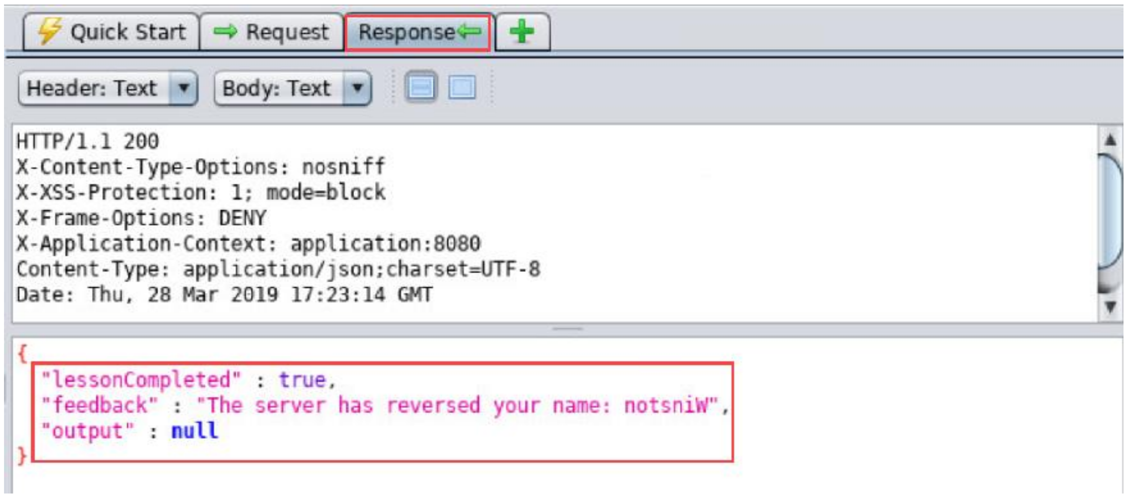
Alerts 0 0 1 0

15. Click on the **Request** tab to see what transaction took place when you clicked *Go!*



```
Accept: */*
Origin: http://192.168.1.4:8080
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
60.0.3112.113 Safari/537.36
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Referer: http://192.168.1.4:8080/WebGoat/start.mvc
Accept-Language: en-gb
Cookie: JSESSIONID=CD8E13E156762E6C1A80685DF0AE614A
Host: 192.168.1.4:8080

person=Winston
```

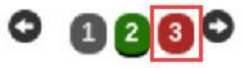
16. Click on the **Response** tab to see what occurred afterward.



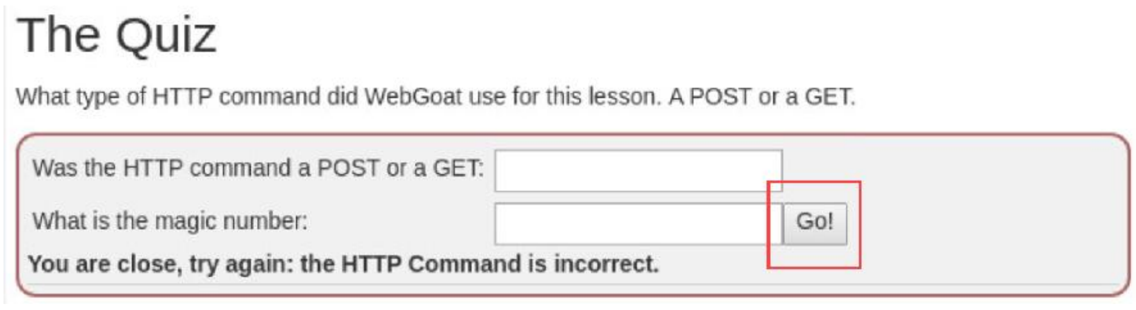
```
HTTP/1.1 200
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Frame-Options: DENY
X-Application-Context: application:8080
Content-Type: application/json;charset=UTF-8
Date: Thu, 28 Mar 2019 17:23:14 GMT

{
  "lessonCompleted" : true,
  "feedback" : "The server has reversed your name: notsniW",
  "output" : null
}
```

Web pages often pass hidden variables to the web server that users never come in contact with. For the next exercise, this variable is referred to as a *magic number*.

17. Navigate back to the *WebGoat* browser window and click on number **3** button in the *HTTP Basics* category.



18. In order to generate the magic number, click the **Go**! button.



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:

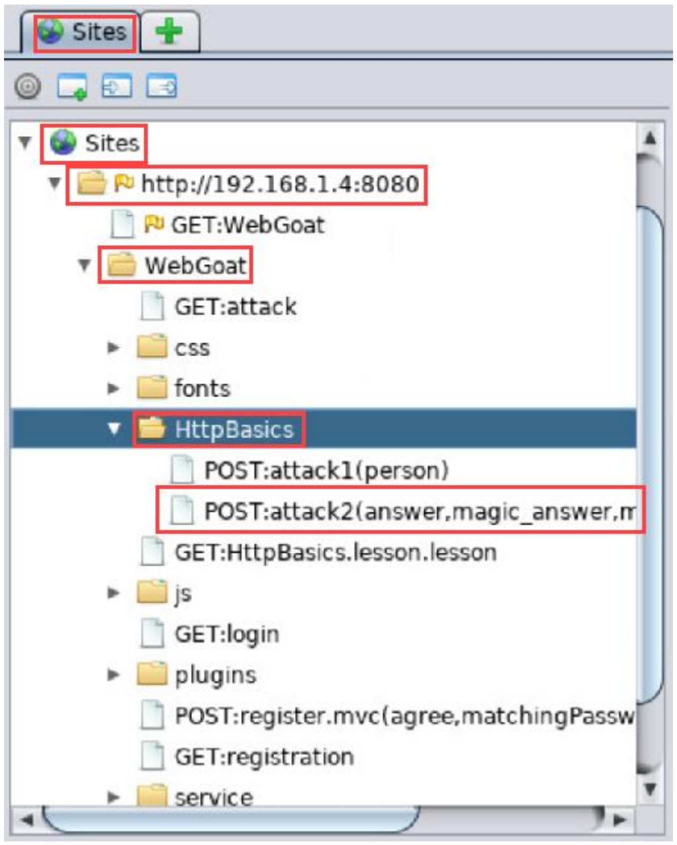
What is the magic number:

Go!

You are close, try again: the HTTP Command is incorrect.

The error given here is a necessary step. In clicking **Go!** you have generated a second *POST* attack which contains a hidden variable magic number. Without pressing **Go!** and generating the error message the first time, the traffic required to correctly fill in the fields will not exist.

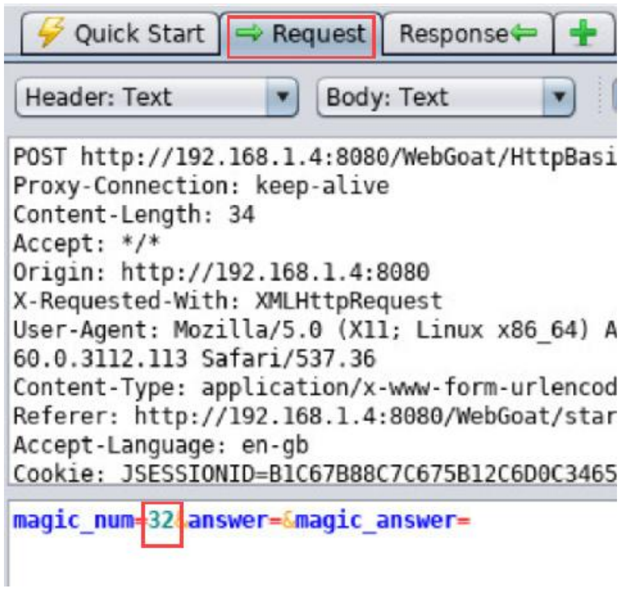
19. Return to the *web proxy* window. In the *Sites* pane, navigate to **Sites**-> **http://192.168.1.4:8080** -> **WebGoat** -> **HTTPBasics**. Click on **POST:attack2**.



```
Sites
  http://192.168.1.4:8080
    GET:WebGoat
    WebGoat
      GET:attack
      css
      fonts
      HttpBasics
        POST:attack1(person)
        POST:attack2(answer,magic_answer,m
      GET:HttpBasics.lesson.lesson
      js
      GET:login
      plugins
      POST:register.mvc(agree,matchingPassw
      GET:registration
      service
```

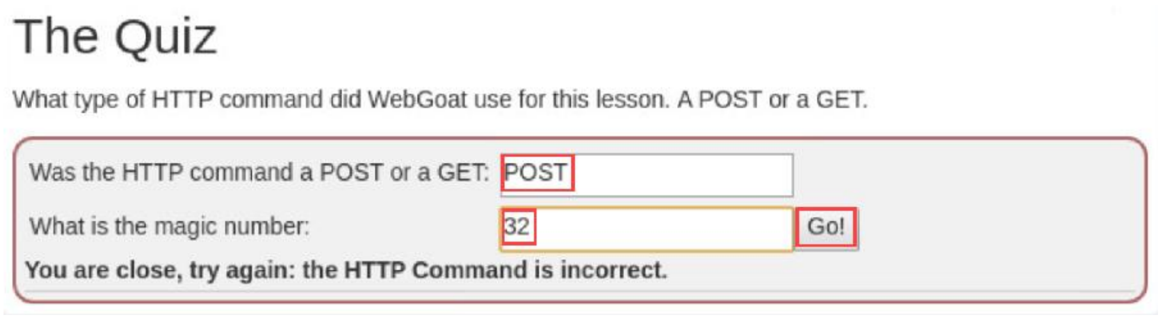
In the *Sites* pane, you will notice both *GET* and *POST* methods. The *GET* method requests data from a web server/service resource. The *POST* method sends data to create or update a resource.

20. Notice in the **Request** pane in the right side of the screen, a magic number is posted.



```
POST http://192.168.1.4:8080/WebGoat/HttpBasi
Proxy-Connection: keep-alive
Content-Length: 34
Accept: */*
Origin: http://192.168.1.4:8080
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (X11; Linux x86_64) A
60.0.3112.113 Safari/537.36
Content-Type: application/x-www-form-urlencod
Referer: http://192.168.1.4:8080/WebGoat/star
Accept-Language: en-gb
Cookie: JSESSIONID=B1C67B88C7C675B12C6D0C3465

magic_num=32&answer=&magic_answer=
```

21. Return to the *WebGoat* browser window. Enter **POST** and your **magic number** into the answers field. Click **Go!** to test that you have accurately found your magic number.

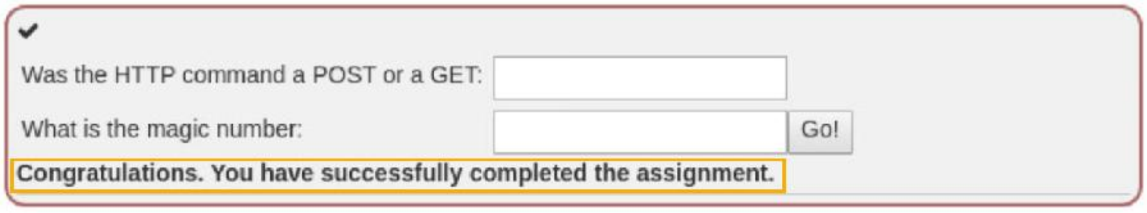


The Quiz

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:	32

You are close, try again: the HTTP Command is incorrect.



✓

Was the HTTP command a POST or a GET:	
What is the magic number:	

Congratulations. You have successfully completed the assignment.

22. This concludes the lab. You may now end the reservation.