Basic Vulnerability Assessment and Penetration Testing

Steps for Various Types of Vulnerabilities

1. SQL Injection:

Manual:

* Add an apostrophe at the end of the URL and run the page. If it shows an error then the page may be vulnerable to SQL injection.
* The next step is to use the ORDER BY command in order to get a count of the number of records in the table running behind the webpage.

E.g: <url>id=1’ ORDER BY 9

* It will not show an error unless the number exceeds the number of records.
* The next step is to use the ‘UNION’ command. The union command is extensively used to extract the table names, column names etc.

E.g: <url>id=1’ UNION SELECT username,password FROM users --+

(--+ is used to remove commenting and is often used in manual SQLi)

* Usually, if the website is made in PHP the common database name is information\_schema. Hence we will use this database name in our examples.
* The entire command is as follows:

<url> UNION SELECT 1,2,column\_name,4,5 from information\_schema.columns where table\_schema=database() and table\_name='tablenamehere'--+

* In text boxes a true condition is used to perform SQLi.

E.g: ‘ OR ‘1’=’1

Automatic:

* The tool that we use is a python based tool that can be executed with Windows PowerShell and Python 2 installation.
* These commands work best for URL’s that have a GET parameter in them.

>>python sqlmap.py –u “url” –cookie=”key=<key>”

The cookie can be found with the help of chrome extensions. The cookie switch is used to bypass authentication. The command can work without cookie switch as well in the case where authentication bypass is not there.

* The above command will reveal any vulnerable parameters and also the type of SQLi Vulnerability.

>>python sqlmap.py –u “url” –cookie=”key=<key>” --dbs

This is used to find the name of the databases.

* Similarly, --tables, -- columns is used to find the names of the table and columns respectively. These are used in conjunction with the preceding information found like the name of the database, table and column name.
* Final command:

>>python sqlmap.py –u “url” –cookie=”key=<key>” –D <name of the database> -T <name of the table> -C <name of the column>

--dump

Dump switch makes all the records visible.

Hence SQLi is achieved using both manual and automatic methods. This is relevant for GET parameter URLs. For POST based SQLi BurpSuite requests are intercepted and then run in SQLMap.

1. Shell Upload/ Getting access to a Admin Panel

Often, one of the core mistakes that developers do is, they allow any type of file to be uploaded rather than the only the relevant one (like .jpg,.pdf). This the makes the website extremely vulnerable and a shell can easily be uploaded in the form of a PHP file. In this text we will refer to the ‘b374k shell’. Once uploaded, it can be run by clicking on the file. The CLI password is root. Remember this file should only be uploaded as a PHP file using notepad.

I have attached the notepad file in the resources.

Once the shell is active, the attacker can virtually get access to any function running on the website and also to sensitive information present in the databases.

1. Brute Force OTP’s

One of the most interesting attacks; this uses the software BurpSuite. BurpSuite is available for download FOC on the PortSwigger website. There is also a pro version available.

Steps:

1. Make sure your BurpSuite software is active and the intercept is on.
2. Navigate to the page which is requesting for the OTP.
3. After that enter any incorrect OTP and submit.
4. Go to the intercepted request in BurpSuite. Send the request to the intruder.
5. In the intruder section, clear all other attack points other than the OTP.
6. Choose a sniper attack and move to payloads tab.
7. There you can upload and create your list and start the attack.
8. Once the attack starts, the entry that differs in the content length is the OTP.

Repeater and Intruder are two very common tabs used in BurpSuite for many attacks.

1. Indirect Object Reference

This is a common error when it comes to GET parameters. If we simply change the value of the GET parameter then if this vulnerability is present we are redirected to the page holding that value, which should in the ideal scenario not be visible to us.

1. Checking for XSS

The thumb rule to check for XSS is to generate an alert box. The script can be added in the URL and then it can be seen whether the alert bpx is executed.

The Script: <script> alert(1)</script>

This is added immediately after the URL. One point that has to be taken care of is that, sometimes the developer blocks the usual way script tag is written, in that case you can change the case and add extra characters in order to execute the script tag.

E.g: <script>alert(1);</script>