Web Application Pen-Testing

AY 2022/2023

Week 2.2 Practical

OWASP Top 10 - 2021

A07:2021-Identification and Authentication Failures Part 2

Enumeration & Exploitation of WordPress Authentication Using Metasploit

Session Hijacking Using Burp Suite

Document Version: 2 | Last Updated: 28th Oct 2022

Tools covered:

* Enumeration Tools: Metasploit
* Exploitation Tool: Metasploit, Burp Suite

Major changes to this version v3 compared to the previous version v2 of this document

* Cover page
* Overall changes to heading titles
* Session Hijacking Using Burp Suite

Table of Contents

[Enumeration & Exploitation of WordPress Authentication using Metasploit 3](#_Toc117835708)

[Launch Metasploit 3](#_Toc117835709)

[Search and Use Auxiliary / Exploits Modules related to WordPress in Metasploit 4](#_Toc117835710)

[Enumeration & Exploitation using Metasploit 5](#_Toc117835711)

[Post-Exploitation 9](#_Toc117835712)

[Session Hijacking Using Burp Suite 11](#_Toc117835713)

[Enabling Kali Linux Web Browser Proxy Settings 11](#_Toc117835714)

[Launch Burp Suite. 12](#_Toc117835715)

[Prepping Up Burp Suite 14](#_Toc117835716)

[Discovery of Session Vulnerability in WordPress Website using Burp Sequencer 16](#_Toc117835717)

[Exploitation of Session Vulnerability in WordPress Website using Burp Intruder 20](#_Toc117835718)

[Fuzzing using Burp Intruder 24](#_Toc117835719)

Note: This practical is a continuation of the Week 2.1 Practical. Though this Week 2.2 Practical can be done independently, it would be helpful to understand concepts that were already covered in Week 2.1 practical.

# Enumeration & Exploitation of WordPress Authentication using Metasploit

## Launch Metasploit

The Metasploit Framework is an open-source platform that supports vulnerability research, exploit development, and the creation of custom security tools.

Metasploit uses postgresql, therefore make sure it is started. Type the following command into the Kali Linux’s Terminal Emulator and press Enter:

sudo service postgresql start

Text

Description automatically generated

To initiate the Metasploit database, type the following command into the Kali Linux’s Terminal Emulator and press Enter:

sudo msfdb init

Text

Description automatically generated

Launch Metasploit, type the following command into the Kali Linux’s Terminal Emulator and press Enter:

sudo msfconsole

Text

Description automatically generated

Check if Metasploit has connected to the database successfully. Type the following command into the msf command line and press Enter:

db\_status

Text

Description automatically generated

## Search and Use Auxiliary / Exploits Modules related to WordPress in Metasploit

There are over 100 such modules already made available in Metasploit for you to use.

* Exploit Module - An exploit module executes a sequence of commands to target a specific vulnerability found in a system or application. An exploit module takes advantage of a vulnerability to provide access to the target system. Exploit modules include buffer overflow, code injection, and web application exploits.
* Auxiliary Module - An auxiliary module does not execute a payload. It can be used to perform arbitrary actions that may not be directly related to exploitation. Examples of auxiliary modules include scanners, fuzzers, and denial of service attacks.

[Sources: <https://docs.rapid7.com/metasploit/msf-overview/>

<https://www.offensive-security.com/metasploit-unleashed/msfconsole-commands/>]

Type the following command into the msf command line and press Enter:

search wordpress

Graphical user interface, text

Description automatically generated

We are looking for a particular auxiliary module that is well known as a WordPress Brute Force and User Enumeration Utility: auxiliary/scanner/http/wordpress\_login\_enum. In order to do this, let us further refine our search to filter out only this particular auxiliary module. Note that this is an auxiliary module because it does both enumeration and brute forcing (exploitation).

Type the following command into the msf command line and press Enter:

grep login\_enum search wordpress



## Enumeration & Exploitation using Metasploit

Let us now use this auxiliary module. Type the following command into the msf command line and press Enter:

use auxiliary/scanner/http/wordpress\_login\_enum

A screenshot of a computer

Description automatically generated with medium confidence

Know all options available to configure this Metasploit module. Type the following command into the msf command line and press Enter:

show options

Text

Description automatically generated

This provides a list of options that can be set for this module. As we must obtain the password for the target user account, we will set the below options:

* PASS\_FILE: Sets the http\_default\_pass.txt file, using which; this module will perform the dictionary attack
  + Type the following command into the msf command line and press Enter:
  + set PASS\_FILE /usr/share/wordlists/metasploit/http\_default\_pass.txt
* RHOST: Sets the target machine (here, the OWASPBWA\_IP address)
  + Type the following command into the msf command line (replace OWASPBWA\_IP with your OWASPBWA VM’s IP address) and press Enter:
  + set RHOSTS OWASPBWA\_IP
* RPORT: Sets the target machine port (here, the OWASPBWA\_IP port)
  + Type the following command into the msf command line and press Enter:
  + set RPORT 80
* TARGETURI: Sets the base path to the WordPress website (here, http:// OWASPBWA\_IP/wordpress]
  + Type the following command into the msf command line (replace OWASPBWA\_IP with your OWASPBWA VM’s IP address) and press Enter:
  + set TARGETURI http://OWASPBWA\_IP/wordpress
* USERNAME: Sets the username that was obtained via wpscan tool used in Week 2.1 Practical. Refer to OWASP Top 10 - Identification and Authentication Failures - Part 1 - v2.docx. (here, admin)
  + Type the following command into the msf command line and press Enter:
  + set USERNAME admin
* VERBOSE: Sets whether to print output for all attempts. In this case we want to see only the successful attempts, else it is too noisy. Therefore, we would set this to false.
  + Type the following command into the msf command line and press Enter:
  + set VERBOSE false

A picture containing text

Description automatically generated

All the options have successfully been set. Type run and press Enter to execute the auxiliary module.

run

Observe that the auxiliary module initially enumerates details such as the ID number and the stored location of the usernames: admin and user, and then begins to brute-force the login credentials by trying various passwords for the given username. Notice that in Week 2.1 Practical, wpscan was only able to enumerate 1 username: admin, but this Metasploit module has enumerated 2 usernames: admin, and user. The auxiliary module tests various passwords against the given usernames (admin, and user) and the cracked passwords are displayed, as shown in the screenshot.

Text

Description automatically generated

Now, open your Kali Linux Web Browser and type the following in the address bar (replace OWASPBWA\_IP with your OWASPBWA VM’s IP address):

http://OWASPBWA\_IP/wordpress/wp-login.php?

Graphical user interface, website

Description automatically generated

Enter the Username: user and Password: user and click “Login”.

Graphical user interface, text

Description automatically generated

You will observe: “Howdy, user” displayed at the top right-hand corner.

Graphical user interface

Description automatically generated

You can click “Sign Out” at the top right-hand corner. Now enter the Username: admin and Password: admin and click “Login”. NOTE: Wait for roughly 1 minute for the page to load, this happens due to the extended time taken by both the Kali Web Browser and Server to load the admin dashboard.

Graphical user interface, website

Description automatically generated

You will observe: “Howdy, admin” displayed at the top right-hand corner.

Graphical user interface, text, application, email

Description automatically generated

## Post-Exploitation

Post-Exploitation involves actions taken by the pen tester after the exploitation has been accomplished. Post-Exploitation is carried out to show the client the possible consequences of the exploitation. This would encourage the client to take immediate action to secure this website.

Now that we know the admin password for this WordPress website, there could be several post-exploitations we can carry out. But for now, let us proceed to deface this website as a post-exploitation action.

Click “Options” 🡪 “General” tab on the top menu bar. Change the “Weblog title:” to any text you want and click “Update Options”.

Graphical user interface, text, application, email

Description automatically generatedweb

Once done, the “Options saved” would be displayed on the top.

Graphical user interface, text, application, email

Description automatically generated

Now, open another tab in your Kali Linux Web Browser and type the following in the address bar (replace OWASPBWA\_IP with your OWASPBWA VM’s IP address):

http://OWASPBWA\_IP/wordpress/

You will now see a defaced website.

Graphical user interface, website

Description automatically generated

|  |  |
| --- | --- |
| Before | After: Post-Exploitation – Web Defacement |
| A screenshot of a computer  Description automatically generated | Graphical user interface, website  Description automatically generated |

# Session Hijacking Using Burp Suite

Visit http://OWASPBWA\_IP on your Kali Linux Web Browser

Graphical user interface, text

Description automatically generated

## Enabling Kali Linux Web Browser Proxy Settings

In the Kali Linux Web Browser, click on the right-hand top corner hamburger icon and click on “Settings“

Graphical user interface, text, application

Description automatically generated

Type “proxy” in the search field and click on “Settings…”

A screenshot of a computer

Description automatically generated with medium confidence

Select “Manual proxy configuration” and under “HTTP Proxy” type “127.0.0.1” and under “Port” type 8080, and click “OK”.

A screenshot of a computer

Description automatically generated

## Launch Burp Suite.

Graphical user interface, text, application, chat or text message

Description automatically generated

For any warnings click “OK”.

Graphical user interface, text, website

Description automatically generated

Click “Next”

Graphical user interface, text, application, email

Description automatically generated

Click “Start Burp”

Graphical user interface, text, application, email

Description automatically generated

## Prepping Up Burp Suite

Click on “Proxy” tab 🡪 “Intercept” tab

Graphical user interface, text, application, email

Description automatically generated

Click on the button “Intercept is on” to toggle it to “Intercept is off”. With this setting Burp Suite will not be intercepting requests and responses between the Kali Linux Web Browser and a webserver. With “Intercept is off” Burp Suite will be in listener mode, where it will just listen and record all the requests and responses made between the Kali Linux Web Browser and a webserver

Graphical user interface, text, application, email

Description automatically generated

Click on “HTTP history” tab, if you notice any records listed in the top pane, right click on the top row record, and click “Clear history”. This would make “HTTP history” tab clear with no previously recorded requests and responses between the Kali Linux Web Browser and a webserver.

Graphical user interface, text, application

Description automatically generated

Click “Yes”

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

## Discovery of Session Vulnerability in WordPress Website using Burp Sequencer

While the Burp Suite is still running, click on “OWASP WebGoat” inside the http://OWASPBWA\_IP/ (replace OWASPBWA\_IP with your OWASPBWA VM’s IP address)

Graphical user interface, text, email, website

Description automatically generated

Type in Username: webgoat, and Password: webgoat

Graphical user interface, text, website

Description automatically generated

Click on “Start WebGoat”

Graphical user interface, website

Description automatically generated

On the left menu, click on “Session Management Flaws” 🡪 “Hijack a Session”

Graphical user interface, website

Description automatically generated

Now look at the Burp Suite, you will notice that all the above Requests and their subsequent Responses have been recorded by Burp Suite inside the “HTTP History” tab. You will also notice in here, something important that had happened, the moment you clicked on the link “Hijack a Session” in the previous step, the http://OWASPBWA\_IP/WebGoat webserver Responded with a “Set-Cookie: WEAKID=XXXXX-XXXXXXXXXXXXX” (Note: Your WEAKID will be different from the one seen in the screenshot below). This WEAKID is a session token associated with this web page: “Hijack a Session”. We will now try to discover whether there is any vulnerability associated with the generation of this WEAKID session token.

Graphical user interface

Description automatically generated

Right click inside the “Response” pane where you see the WEAKID and click “Send to Burp Sequencer”. Immediately the “Sequencer” tab on the top row menu will light up for a while.

Graphical user interface, table

Description automatically generated

Click on the “Sequencer” tab. Under the “Cookie” radio button, use the drop-down menu to select the WEAKID=XXXXX-XXXXXXXXXXXXX (Note: Your WEAKID will be different from the one seen in the screenshot below), this will inform the Burp Sequencer that the token location is in the WEAKID. Now, click on “Start Live Capture”.

Graphical user interface, application

Description automatically generated

Burp Sequencer will issue over 20,000 unique GET requests to the http://OWASPBWA\_IP/WebGoat webserver, and in the process has collected 20,000 WEAKID Session Tokens which are extracted from the webserver responses to each of these GET requests. This is equivalent to you opening 20,000 Private Windows in Firefox and clicking on the link: “Hijack a Session” inside each of these 20,000 Private Windows and manually recording the WEAKIDs sent by the webserver. That would have been a huge undertaking to execute, luckily, we have Burp Sequencer to the rescue.

Click “Analyze now”. This would allow Burp Sequencer to Analyze the randomness among these 20,000 Session Tokens. By right, Session Tokens need to be random and unpredictable. Let us see how these Session Tokens do.

Graphical user interface, text, application, email

Description automatically generated

Give the Burp Sequencer time to complete the analysis on the Session Tokens. Once the analysis is done, we are informed that the randomness among these 20,000 Session Tokens is extremely poor. You can take time to explore the details under each tab below,

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application

Description automatically generated

Click on “Character-level analysis” tab. If you look closely, WEAKID Session Token is made up of 18 numbers. You will notice in the bar chart below, that only the 3rd, 4th, 17th, and 18th position numbers in the WEAKID indicate some level of randomness, with 18th position number being the most random number among all the other position numbers in the WEAKID. This is the reason why the Burp Sequencer has determined that the randomness among the collected 20,000 Session Tokens is extremely poor. Hooray! we have “Discovered” a Session Vulnerability in the “Hijack a Session” page of the http://OWASPBWA\_IP/WebGoat website. Now, it’s time to exploit this Session Vulnerability.

Chart

Description automatically generated

## Exploitation of Session Vulnerability in WordPress Website using Burp Intruder

Click “Save tokens” and save the collected 20,000 Session Tokens on the “Desktop” as webgoat\_tokens.txt. (Click the tiny house icon on the right to quickly navigate to the “Desktop”)

Graphical user interface, text

Description automatically generated

Right click on the Desktop and select “Open Terminal Here”

A screenshot of a computer

Description automatically generated with medium confidence

Sort the 20,000 Tokens in ascending order by typing in the Terminal the following command and press Enter:

sort webgoat\_tokens.txt

Graphical user interface

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with low confidence

You will notice predictable pattern to these Session Tokens, and if you study them more closely you will also observe that there are some gaps in the sequence (Note: Your Session Tokens will be different from those seen in the screenshot above).

For example, in the above screenshot of Session Tokens, 2 Session Tokens are highlighted in yellow colour, if you look closely, between these 2 Session Tokens, one Session Token is missing. This indicates that our Burp Sequencer did not receive this missing Session Token from the http://OWASPBWA\_IP/WebGoat webserver. This means that the webserver could have already given this missing Session Token to a legitimate user of the “Hijack a Session” webpage. If we can correctly resolve this missing Session Token, we can become this legitimate user, thus exploiting this session vulnerability.

Let us further deconstruct this understanding. [NOTE: For ease of doing this exploitation, pick a range in your Session Tokens, where the missing Session Token has ending 2-digits that need to be resolved.]

|  |  |
| --- | --- |
| 56230-16668251871**06** | Collected by Burp Sequencer |
| 5623**1**-16668251871**XY** | Missing Session Token – By following the sequence, we can accurately predict the 5th position number to be a “1”, however we are not able to predict the last 2 digits: the 17th position (**X**) number could be either 0 or 1, whereas the 18th position (**Y**) number could be any number between 0 ~ 9. |
| 56232-16668251871**17** | Collected by Burp Sequencer |

We will now make use of Burp Intruder to Fuzz the last 2 digits. When http://OWASPBWA\_IP/WebGoat webserver receives a WEAKID that had been previously issued to a legitimate user, then the webserver, responds with a “Congratulations!” message, and this “Hijack a Session” challenge is deemed completed with a green tick mark.

To do this, follow the steps below:

Now let us come back to the http://OWASPBWA\_IP/WebGoat, click on “Session Management Flaws” 🡪 “Hijack a Session” links on the left menu. Enter a random User Name: “user” and Password “test”, and click “Login”

Graphical user interface, website

Description automatically generated

This login POST request will be recorded by the Burp Suite inside the “HTTP history” tab. You can identify this record by scrolling all the way down and keeping a lookout for the POST method. Select this POST method, and you will notice, inside the “Request” pane below, that your Kali Linux Web Browser had sent a POST Request (to the WebGoat webserver), containing a WEAKID that it had received previously when you clicked on the “Hijack a Session” link.

Graphical user interface

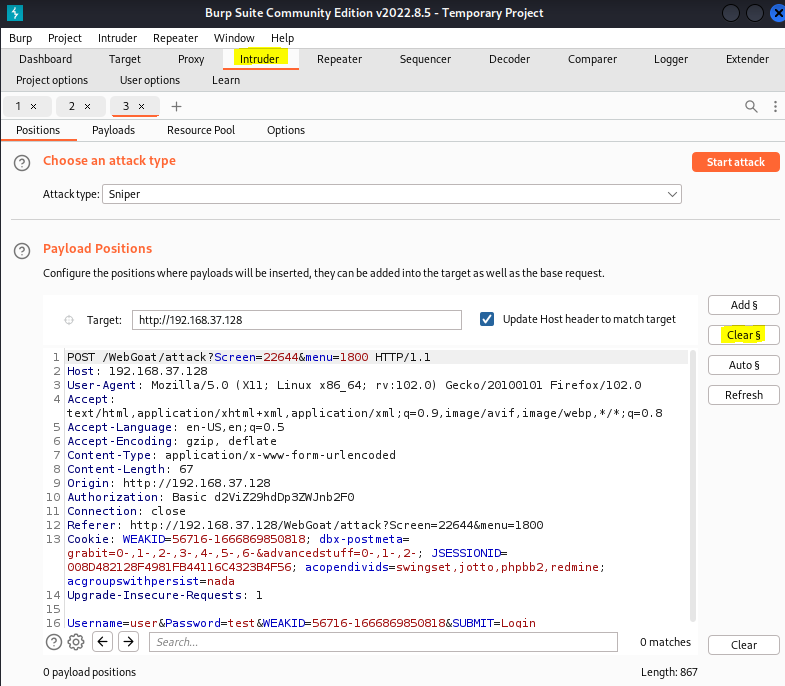
Description automatically generated with medium confidence

Right click inside this “Request” pane and select “Send to Intruder”. Immediately the “Intruder” tab on the top row menu will light up for a while.

Graphical user interface

Description automatically generated with medium confidence

Click on the “Intruder” tab. Click “Clear” on the right, to clear out any of the default selected positions.



### Fuzzing using Burp Intruder

Delete any existing WEAKID and replace it with our estimated missing Session Token (in this case: 5623**1**-16668251871**XY**). Under the “Attack type” select “Cluster bomb”. Next, highlight the **X** and click on “Add$” on the right, similarly, highlight the **Y** and click on “Add$” on the right. By doing this we are telling Burp Intruder to Fuzz these 2 positions.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Click on the “Payloads” tab, and configure as shown below. Basically, we are telling Burp Intruder that for the Payload set: 1 (which is our **X** position), the Payload type: would be a “Simple list” of 0 and 1. And for Payload set: 2 (which is our **Y** position), the Payload type: would be a “Numbers” between 0 ~ 9.

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, email

Description automatically generated

Graphical user interface, text, email

Description automatically generated

After making the above configurations, click “Start attack”. NOTE: Click OK on a pop-up window warning that appears after clicking the “Start attack”

By studying the “Length” column we can identify and select the odd one out (30091). Look at the “Request” tab in the lower pane and you will see the missing Session Token that we wanted to exploit, which is WEAKID=5623**1**-16668251871**11**.

A picture containing graphical user interface

Description automatically generated

Click on the “Response” tab in the lower pane and search for “congratulations”. You will notice that the WebGoat webserver has responded to Burp Intruder’s POST Request with a “Congratulations” message. This completes this challenge. If you go back to the “Hijack a Session” page and refresh it, you will see a green tick mark, indicating that you have successfully completed this challenge.

Graphical user interface, application

Description automatically generated