**Butler walkthrough**

# **Index**

[Index 1](#_Toc204351739)

[List of pictures 1](#_Toc204351740)

[Disclaimer 2](#_Toc204351741)

[Reconnaissance 2](#_Toc204351742)

[Initial foothold 2](#_Toc204351743)

[User flag 4](#_Toc204351744)

[Privilege escalation 5](#_Toc204351745)

[Personal comments 7](#_Toc204351746)

# **List of pictures**

[Figure 1 - nMap scan results 2](#_Toc204351750)

[Figure 2 - ffuf scan results 3](#_Toc204351751)

[Figure 3 - Jenkins version 3](#_Toc204351752)

[Figure 4 - Groovy reverse shell script 4](#_Toc204351753)

[Figure 5 - User shell 4](#_Toc204351754)

[Figure 6 - User information 5](#_Toc204351755)

[Figure 7 - Searching unquoted paths 5](#_Toc204351756)

[Figure 8 - Writing permission on partial path 5](#_Toc204351757)

[Figure 9 - Running process 6](#_Toc204351758)

[Figure 10 - Privilege escalation payload generated 6](#_Toc204351759)

[Figure 11 - Reverse shell uploaded and service restarted 6](#_Toc204351760)

[Figure 12 - Shell as NT AUTHORITY SYSTEM 7](#_Toc204351761)

# **Disclaimer**

I do this box to learn things and challenge myself. I’m not a kind of penetration tester guru who always knows where to look for the right answer. Use it as a guide or support. Remember that it is always better to try it by yourself. All data and information provided on my walkthrough are for informational and educational purpose only. The tutorial and demo provided here is only for those who are willing and curious to know and learn about Ethical Hacking, Security and Penetration Testing.

Just to say: I am not an English native person, so sorry if I did some grammatical and syntax mistakes.

# **Reconnaissance**

The results of an initial nMap scan are the following:

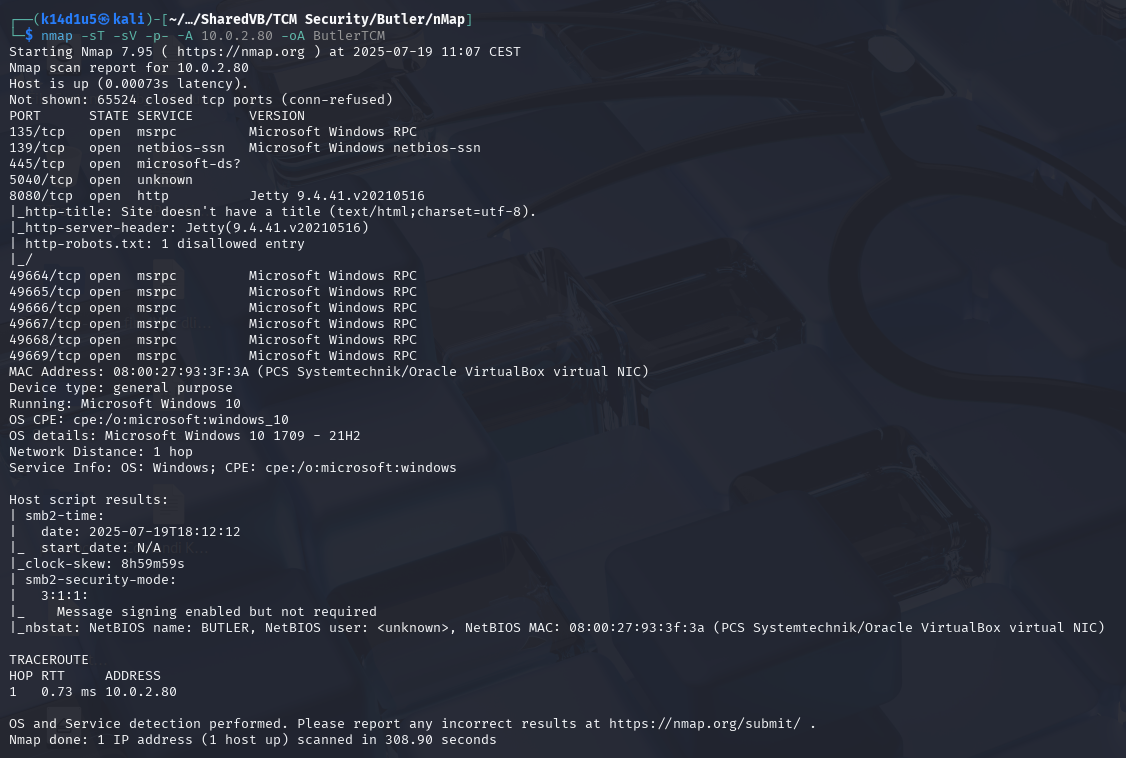


Figure 1 - nMap scan results

Open ports are 135, 139, 445, 5040, 8080, 49664, 49665, 49666, 49667, 49668, 49669. Therefore, enabled services are RPC (135, 49664, 49665, 49666, 49667, 49668, 49669), NetBIOS (139) and SMB (445). Also, an unknown service is running on port 5040 and a web application is running on port 8080. Lastly, nMap recognized Windows 10 as operative system.

# **Initial foothold**

First of all, I tried to connect to the SMB service. I was able to do it in an anonymous session, but I was not able to read any files. At this point, I run ffuf tool to find some interesting path on the web application. Even if it was not able to properly complete its task, I found the following path:

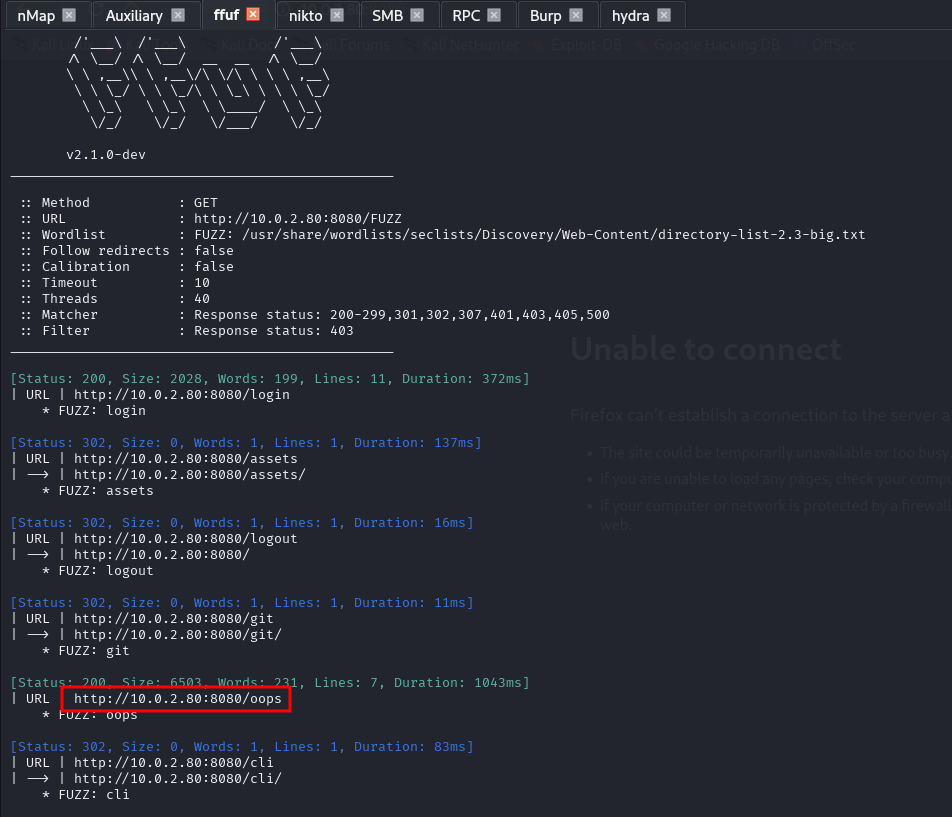


Figure 2 - ffuf scan results

I browsed to this page and I found the Jenkins version, as shown in the following picture:

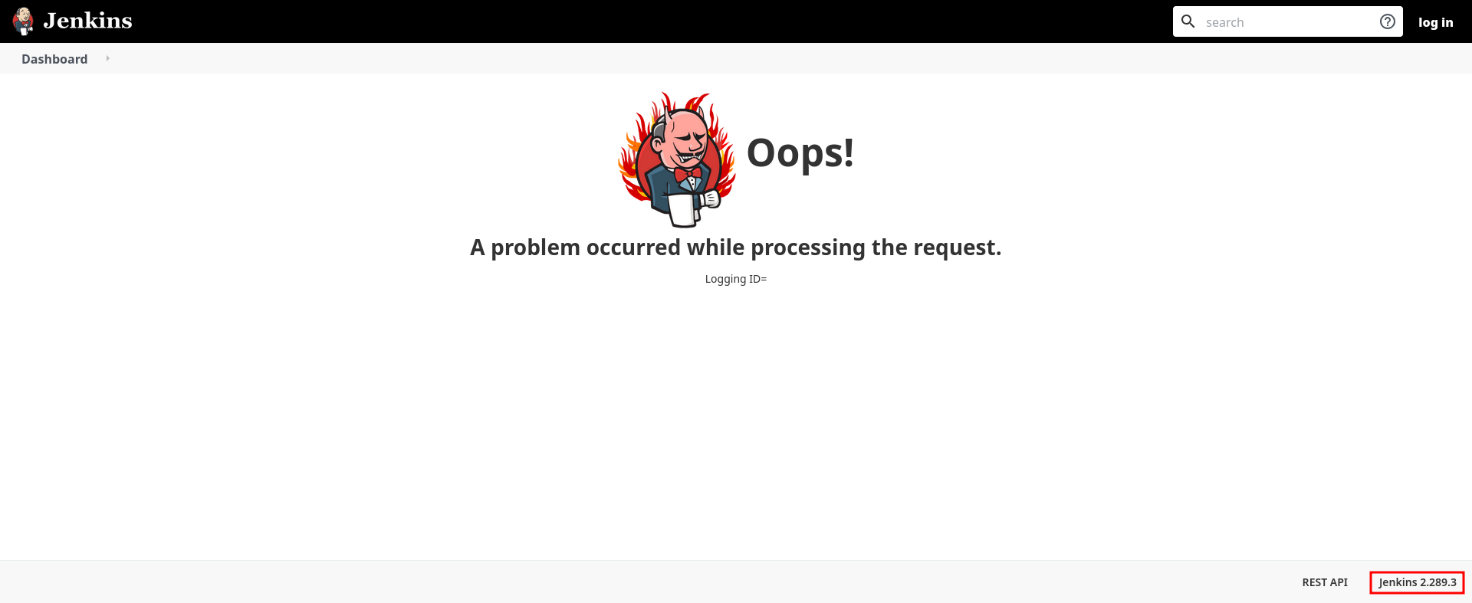


Figure 3 - Jenkins version

Also, I browsed to the web application, where I found a login form.

# **User flag**

I tried to run a brute force attack against the login form, but even in this case the tool I used (hydra) after a certain number of requests broke the web application. I thought that I wasn’t able to use any automatic tools and I started to look for something else. However, I didn’t find anything else. At this point, I thought to build a simple and short wordlist to use as username and password and tried to brute force again the login form. Luckily, I found the login credentials in this way. The credentials were . Since I found a valid login, I looked on the Internet for a useful exploit. I found one about Groovy script. Therefore, I tried to obtain a reverse shell running the following script from the path :

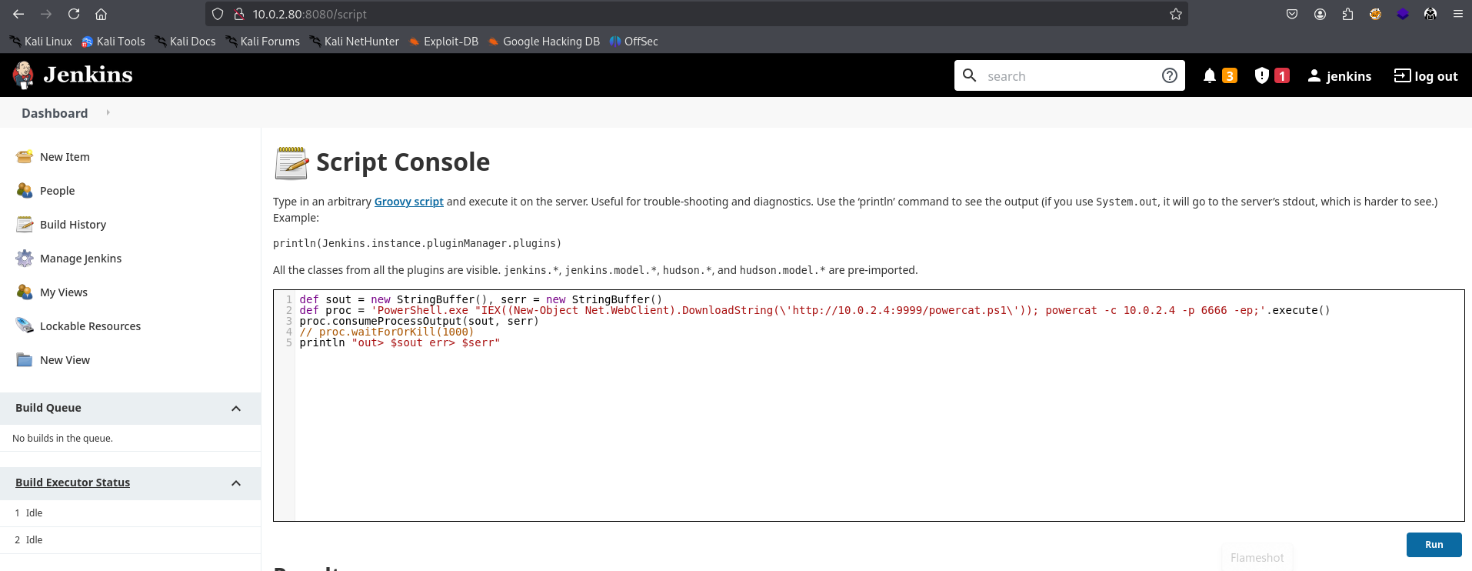


Figure 4 - Groovy reverse shell script

It worked and I obtained a shell as user:

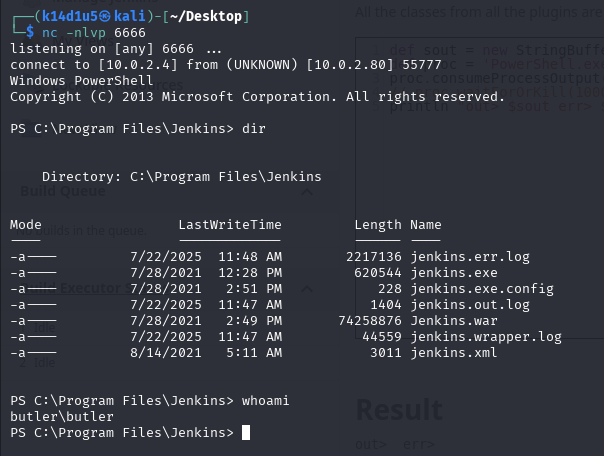


Figure 5 - User shell

# **Privilege escalation**

At this point, I needed to escalate my privileges. The first check I performed was about my user information:

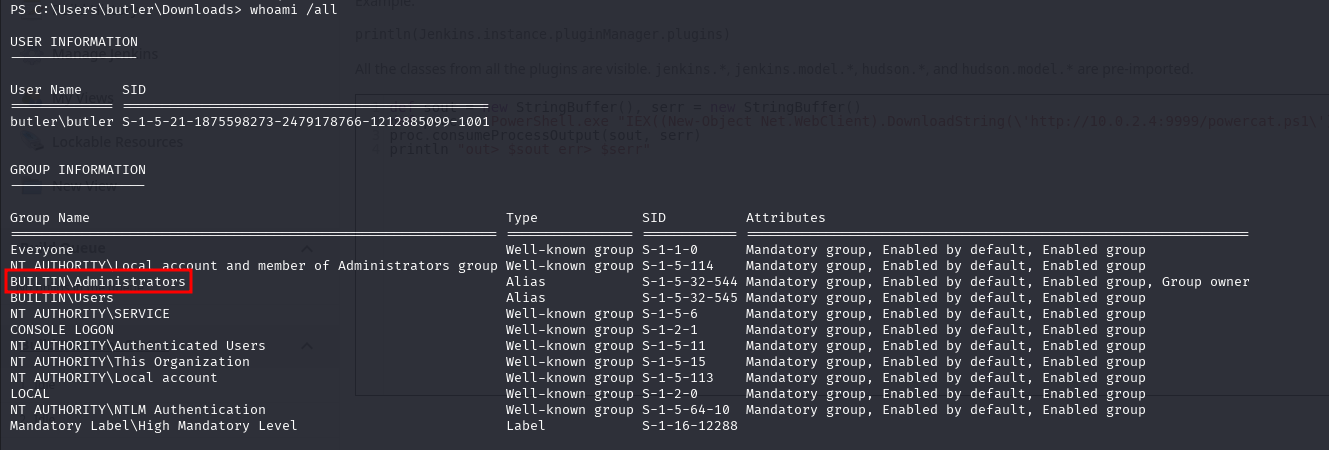


Figure 6 - User information

In this way, I found out user was in the group. Also, I checked if some programs were possibly vulnerable to “Unquoted paths” injection:

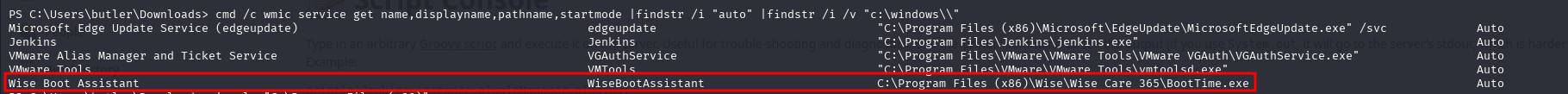


Figure 7 - Searching unquoted paths

Luckily, I found that could be vulnerable. At this point, I checked if I was able to write in some point of its path, as shown in the following picture:

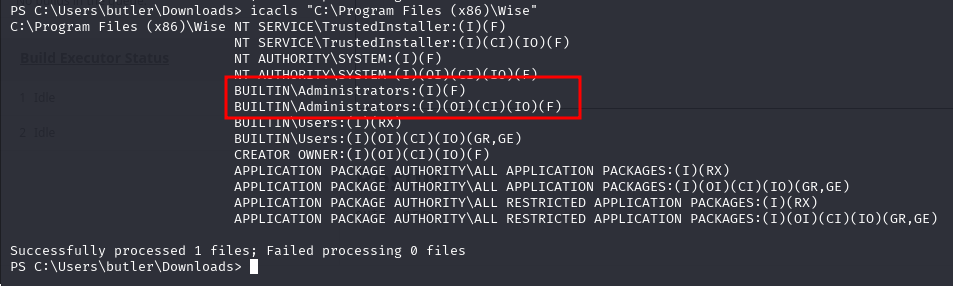


Figure 8 - Writing permission on partial path

I found out that the group where my user is in was able to write in that part of the path. Therefore, I checked if a service regarding this program was running:

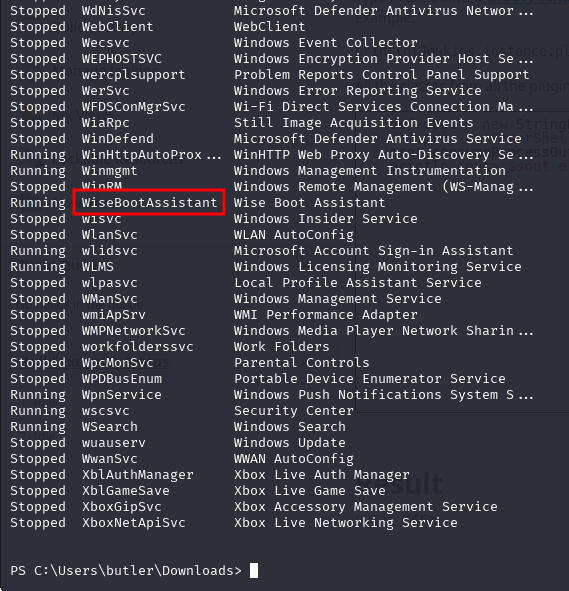


Figure 9 - Running process

Luckily, the program had a service running. At this point, I created a payload to open a new reverse shell:

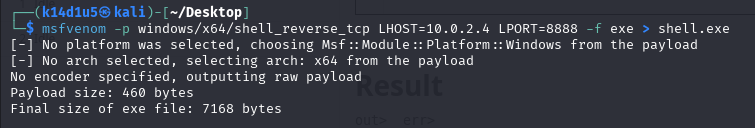


Figure 10 - Privilege escalation payload generated

Therefore, I uploaded the payload in the correct path with the correct name and I restarted the service:

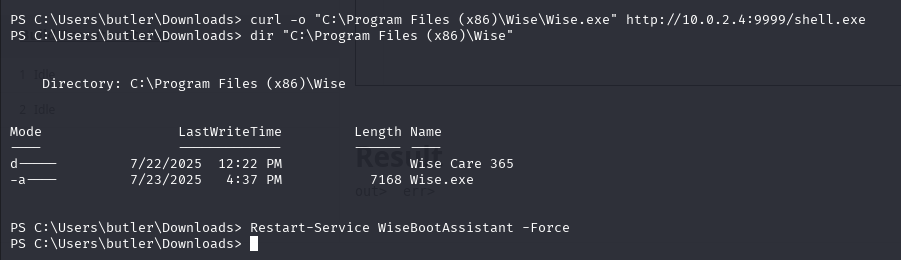


Figure 11 - Reverse shell uploaded and service restarted

Finally, I obtained the shell as , as shown in the following picture:

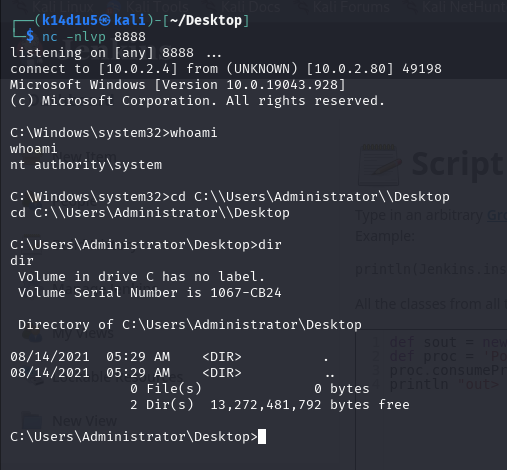


Figure 12 - Shell as NT AUTHORITY SYSTEM

# **Personal comments**

I consider this box interesting and useful to approach to the Windows penetration testing. In contrast to the official solution, I had a little bit different result by nMap, but it was not an issue (I found more open ports). A real problem was that I was not able to use tool like hydra to automized brute force task. For this reason, I lost a lot of time to find web application credentials. Also, a very important point to note was that the user had not permission to write in a path to exploit unquoted paths, but the group where it is in had this permission. This means that even the user can write in that specific path and unquoted paths could be exploited. Lastly, it was important to generate a MSFVenom non staged payload. In fact, I tried a staged one, but it didn’t work. In conclusion, I consider easy this box, but it was a very nice one.