**Pandora walkthrough**

# **Index**

[Index 1](#_Toc194247951)

[List of pictures 1](#_Toc194247952)

[Disclaimer 2](#_Toc194247953)

[Reconnaissance 2](#_Toc194247954)

[Initial foothold 3](#_Toc194247955)

[User flag 5](#_Toc194247956)

[Privilege escalation 6](#_Toc194247957)

[Personal comments 7](#_Toc194247958)

[Appendix A – CVE-2021-32099 7](#_Toc194247959)

[Appendix B – CVE-2020-13851 7](#_Toc194247960)

[References 8](#_Toc194247961)

# **List of pictures**

[Figure 1 - nMap scan results 2](#_Toc194247938)

[Figure 2 - nMap UDP scan results 3](#_Toc194247939)

[Figure 3 - nMap scan to analyze SNMP 3](#_Toc194247940)

[Figure 4 - Credentials found 4](#_Toc194247941)

[Figure 5 - SSH login 4](#_Toc194247942)

[Figure 6 - API call found 4](#_Toc194247943)

[Figure 7 - Local web application 5](#_Toc194247944)

[Figure 8 - SQL Injection successful 5](#_Toc194247945)

[Figure 9 - Session ID retrieved 6](#_Toc194247946)

[Figure 10 - Exploit and user flag 6](#_Toc194247947)

[Figure 11 - Second interesting file 7](#_Toc194247948)

[Figure 12 - Privilege escalation preparation 7](#_Toc194247949)

[Figure 13 - Privilege escalation and root flag 7](#_Toc194247950)

# **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:

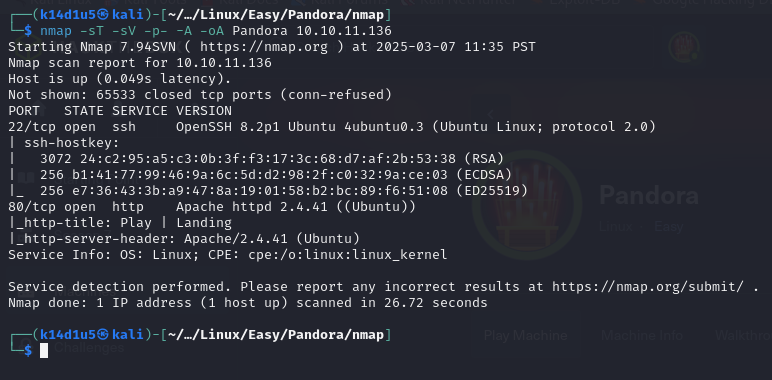


Figure 1 - nMap scan results

Open TCP ports ae 22 and 80. Service enabled are SSH (22) and there is a web application running on port 80. Also, nMap recognized Linux as operative system.

However, this box needed to run an UDP scan too. UDP scan results are the following:

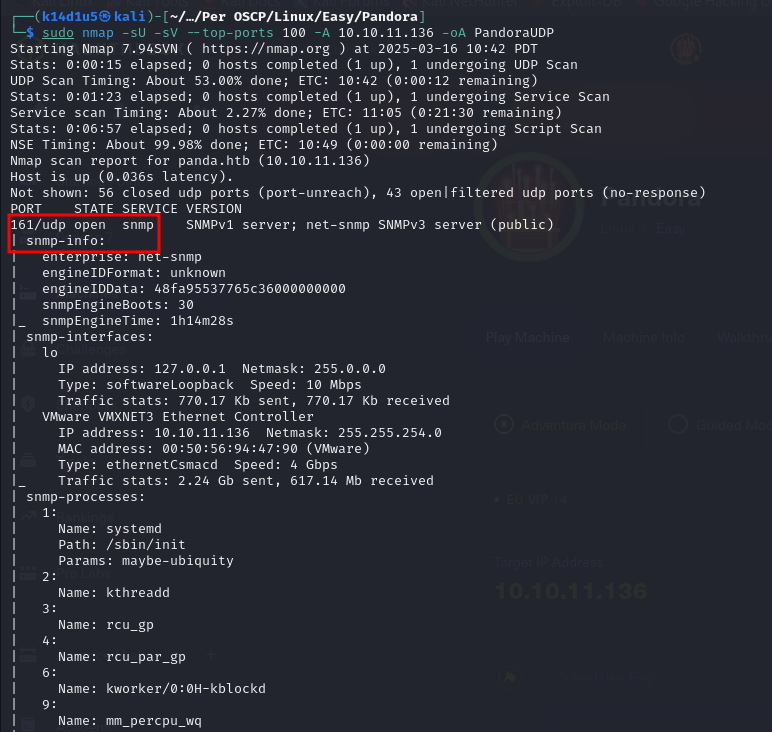


Figure 2 - nMap UDP scan results

On UDP, nMap found the SNMP service enabled on port 161. Since the UDP scan is time consuming, I run it just on the top 100 ports.

# **Initial foothold**

The first tries I did was browsing and analyzing the web application. However, all analysis on the web application didn’t provide nothing of interesting. Next task was analyzing the SNMP service found on UDP. The first information I found was a community string:

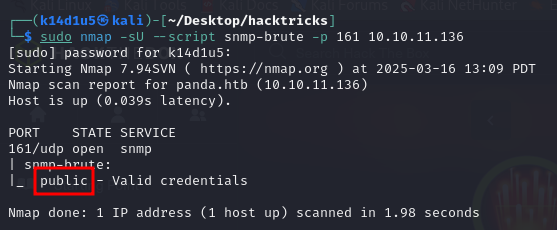


Figure 3 - nMap scan to analyze SNMP

However, I didn’t find any information about SNMP version. So, since I hadn’t any credentials, I supposed that version could be 1 or 2c (version 3 require credentials). At this point I searched more information using SNMP. Since I’d like a more readable output, I commented row 4 in the file. At this point I run the commands and to analyze the service using both 1 and 2c versions. I was very lucky. In fact, analyzing the output files I found credentials:



Figure 4 - Credentials found

Since I had some credentials, I tried to log in the target machine via SSH. Again, I was lucky and it worked!



Figure 5 - SSH login

Even I had an SSH login, the user was not the one who had the user flag. So, I started to explore the file system and I found an interesting custom script named . Even it was an executable, I tried to print its content. In this way, I found a request to a local service API:

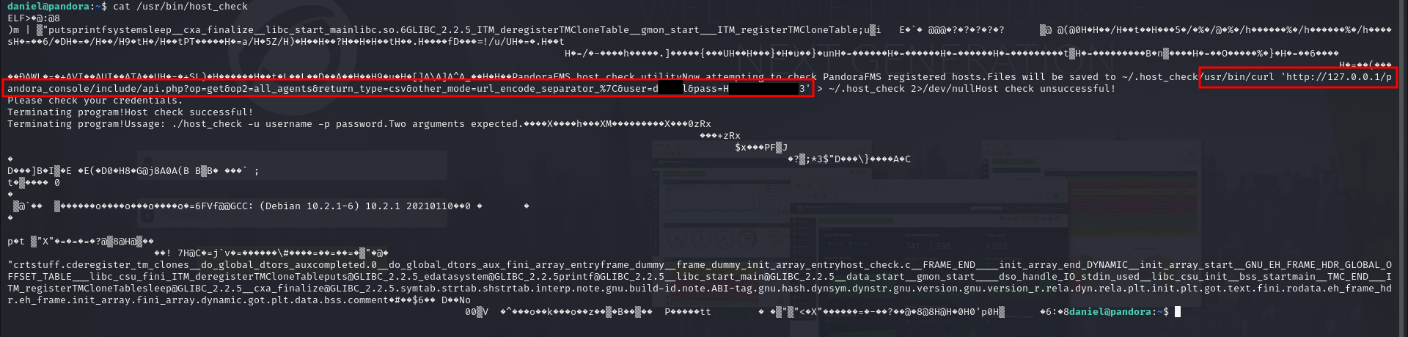


Figure 6 - API call found

# **User flag**

Since I found an interesting local endpoint, I tried to access to it using **Chisel** tool and browsed to it:

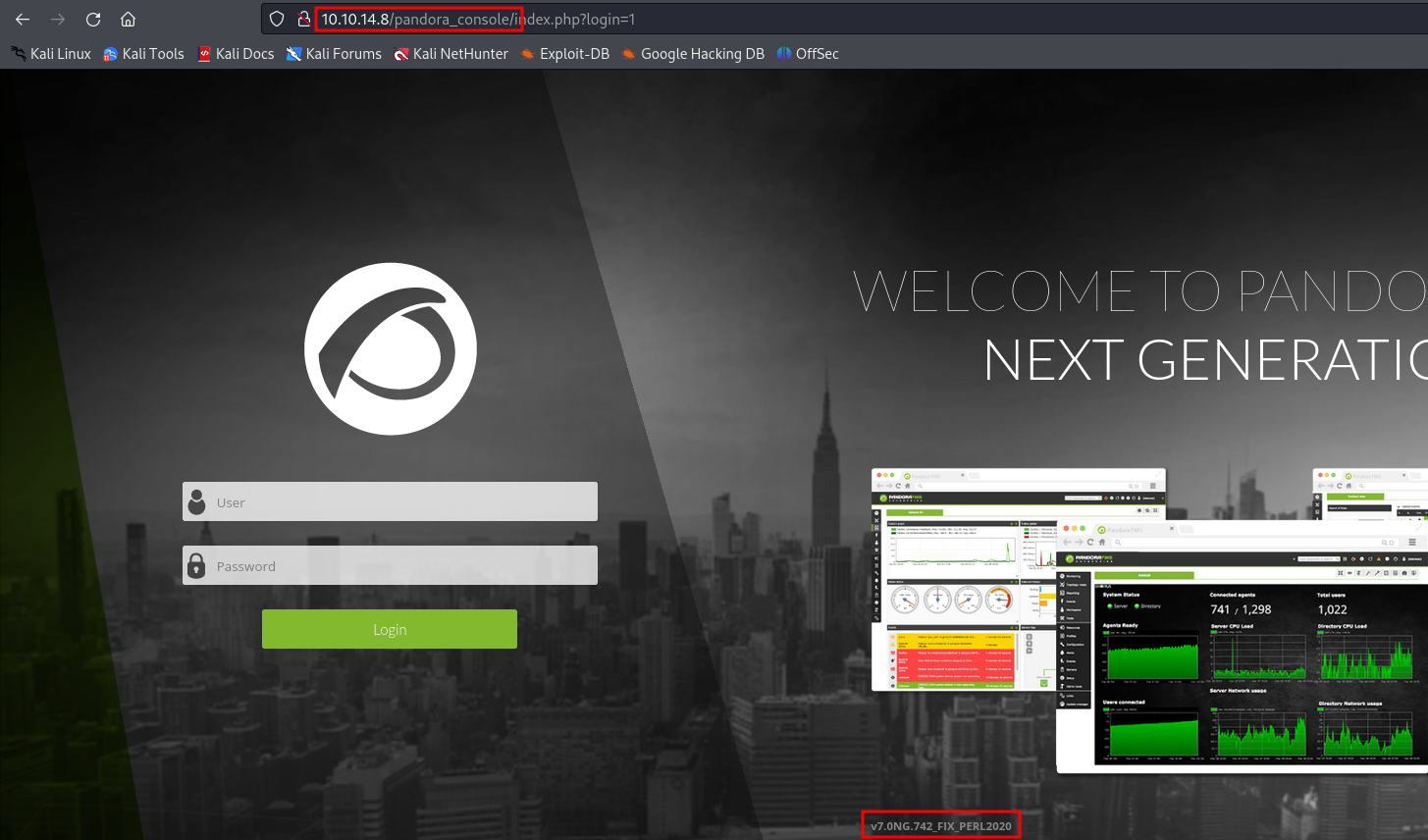


Figure 7 - Local web application

At this point I looked for possible exploits against this application on the Internet and there were several. In particular, based on the CVE-2021-32099, I found that the application was vulnerable to SQL Injection on a specific path:



Figure 8 - SQL Injection successful

In particular, I was able to retrieve session ID values, as shown in the following:

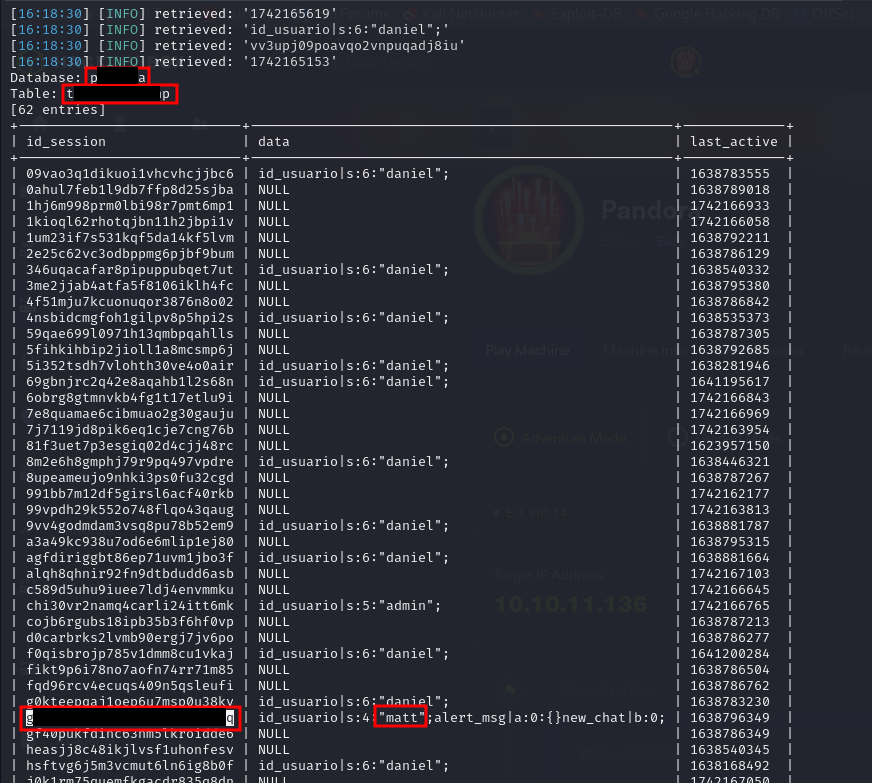


Figure 9 - Session ID retrieved

At this point, I explored the other CVE exploits I found and finally I was able to obtain a user shell with the user and retrieve the user flag:

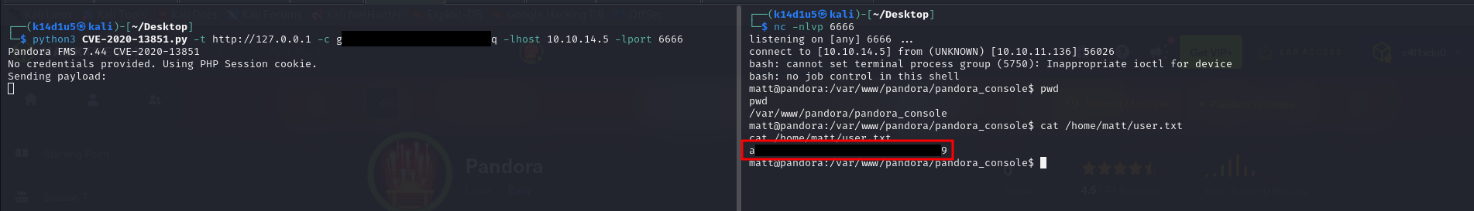


Figure 10 - Exploit and user flag

# **Privilege escalation**

Again, I analyzed the file system and, again, I found an interesting file named :

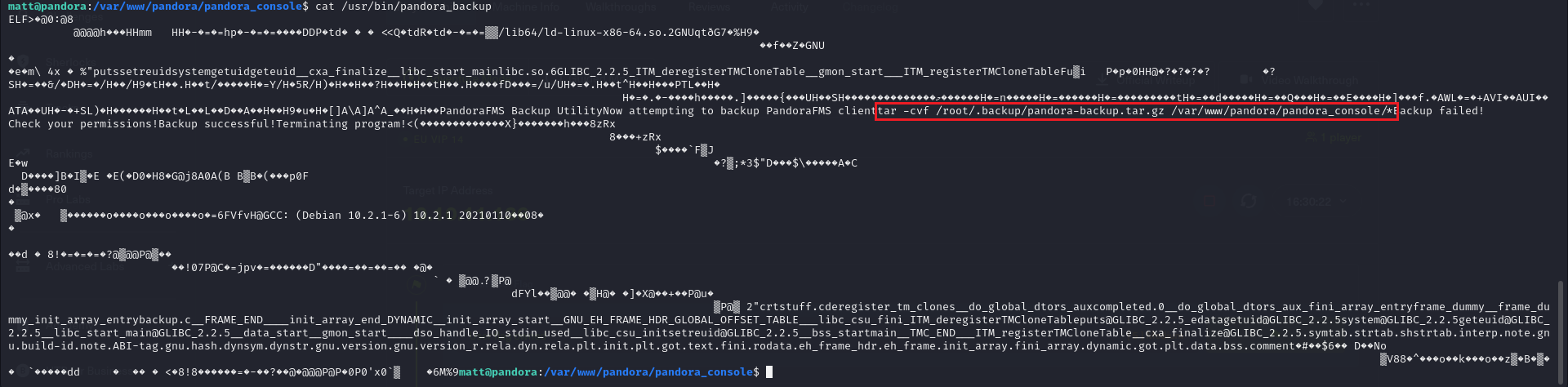


Figure 11 - Second interesting file

At this point, I needed to force the system to execute an executable named to obtain a root shell. To do so, I modified the PATH environment variable putting the path at the beginning:



Figure 12 - Privilege escalation preparation

So, I created in the path a file named which contains the code and made it executable. Lastly, I just needed to execute the script to obtain the root shell:

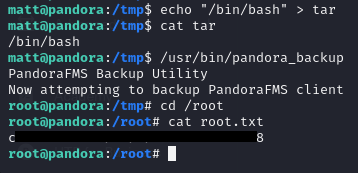


Figure 13 - Privilege escalation and root flag

I didn’t take a proof, but this exploit worked because the script has the SUID flag set.

# **Personal comments**

This box is very nice, but has just a problem: you have to identify the right CVE to exploit and all of them were plausible. In my opinion, this effort is too much for an Easy box. So, I evaluate this flag as Medium. The root flag was funny, but quite easy.

# **Appendix A – CVE-2021-32099**

CVE-2021-32099 affects some unknown processing of the file of the component . The manipulation of the argument **session\_id** with an unknown input leads to a SQLInjection vulnerability. The product constructs all or part of an SQL command using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended SQL command when it is sent to a downstream component.

# **Appendix B – CVE-2020-13851**

CVE-2020-13851 affects some unknown processing of the component . The manipulation with an unknown input leads to an injection vulnerability. The product constructs all or part of a command, data structure, or record using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify how it is parsed or interpreted when it is sent to a downstream component. The attack may be initiated remotely. Required for exploitation is a simple authentication. The technical details are unknown and an exploit is not publicly available.

# **References**

1. SNMP Pentest cheatsheet: <https://www.hackingdream.net/2023/08/snmp-pentest-cheatsheet-port-161.html>;
2. CVE-2021-32099: <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-32099>;
3. CVE-2020-13851: <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-13851>.