**Ready walkthrough**

# **Index**

[Index 1](#_Toc199171209)

[List of pictures 1](#_Toc199171210)

[Disclaimer 2](#_Toc199171211)

[Reconnaissance 2](#_Toc199171212)

[Initial foothold 2](#_Toc199171213)

[User flag 3](#_Toc199171214)

[Privilege escalation 3](#_Toc199171215)

[Personal comments 7](#_Toc199171216)

[Appendix A – CVE-2018-19571 7](#_Toc199171217)

[Appendix B – CVE-2018-19585 7](#_Toc199171218)

[References 7](#_Toc199171219)

# **List of pictures**

[Figure 1 - nMap scan results 2](#_Toc199171119)

[Figure 2 - GitLab version 2](#_Toc199171120)

[Figure 3 - User shell 3](#_Toc199171121)

[Figure 4 - User flag 3](#_Toc199171122)

[Figure 5 - gitlab.rb file 3](#_Toc199171123)

[Figure 6 - Password found in gitlab.rb file 4](#_Toc199171124)

[Figure 7 – Container root user 4](#_Toc199171125)

[Figure 8 - PID 1 wasn't init process 5](#_Toc199171126)

[Figure 9 - Init process not found 5](#_Toc199171127)

[Figure 10 - Docker container recognized by Linpeas 5](#_Toc199171128)

[Figure 11 - Container exploitation clue 5](#_Toc199171129)

[Figure 12 - Docker vulnerabilities 6](#_Toc199171130)

[Figure 13 - Enabled capabilities 6](#_Toc199171131)

[Figure 14 - AppArmor running check 6](#_Toc199171132)

[Figure 15 - Privilege escalation and root flag 7](#_Toc199171133)

# **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:

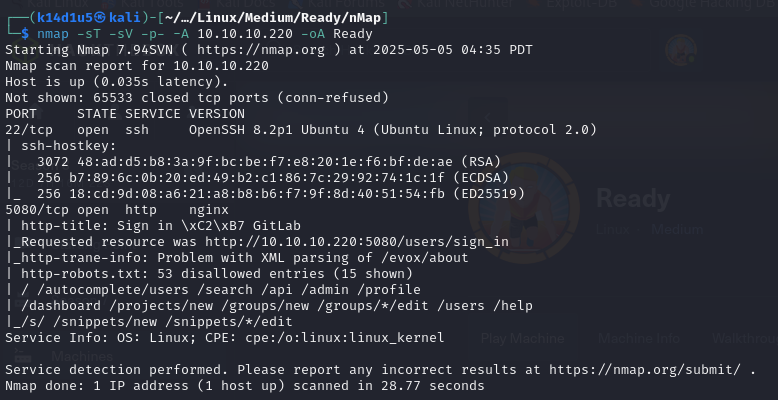


Figure 1 - nMap scan results

Open ports are 22 and 5080. Therefore, SSH service (22) was enabled. Also, a web application was deployed on port 5080. Lastly, nMap provided Linux as operative system, probably Ubuntu, but any other information about it.

# **Initial foothold**

The only port I was able to analyze on this box was 5080. Therefore, I analyzed the web application running FFUF tool. In this way, I found some paths. Some of them was relative to some user registered on the GitLab application. In addition, I was able to register a new user and I did it. After logged in with the user I just created, I was able to found the GitLab version, as shown in the following:

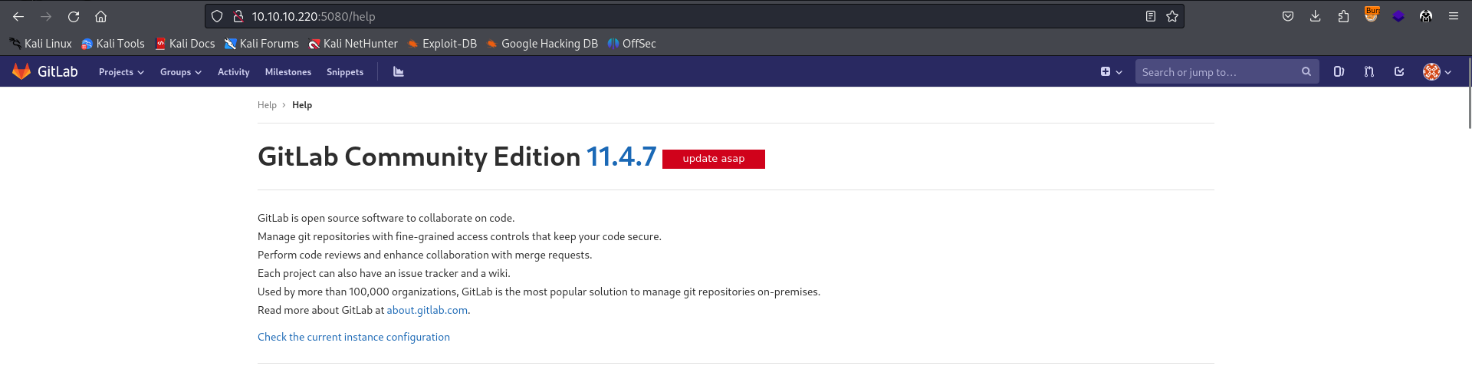


Figure 2 - GitLab version

# **User flag**

At this point, I looked on the Internet if some exploits were available against the GitLab version and I found an interesting one. Since I have all information I needed, I just run the exploit and, luckily, I obtained the user shell, as shown in the following picture:

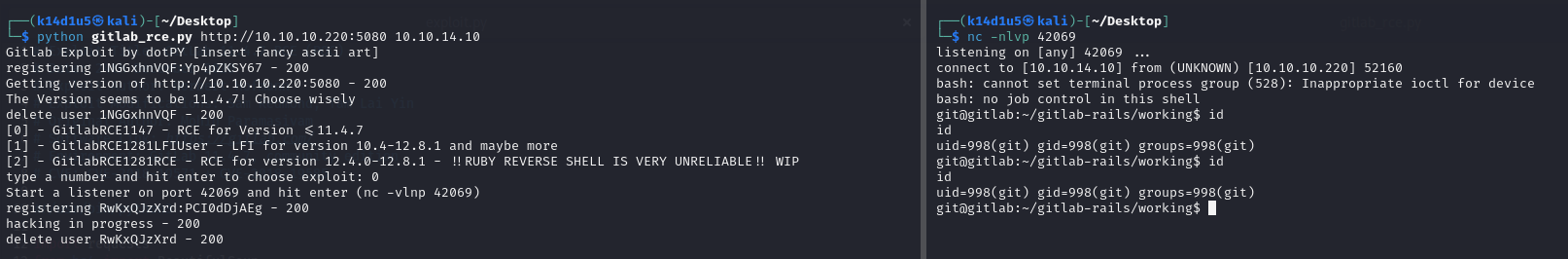


Figure 3 - User shell

Using this shell, I was already able to retrieve the user flag:

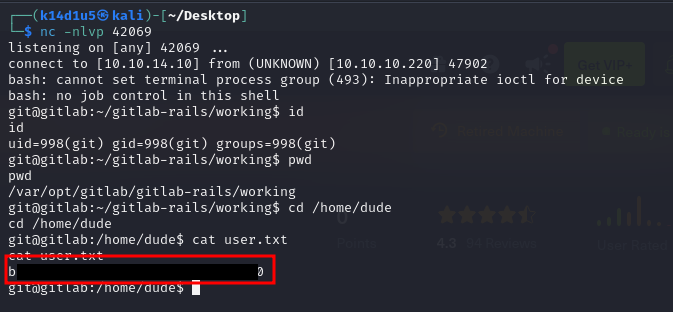


Figure 4 - User flag

# **Privilege escalation**

This was the moment to escalate my privileges. To achieve this goal, I looked for some interesting information on the file system. First of all, I found a file named . It contained something that looks like a password. I tried to use it running the command (I needed to upgrade my shell with a python one), but it didn’t work. Therefore, I kept to look for other information. This time, I found an interesting database file named . In this file I found another password:

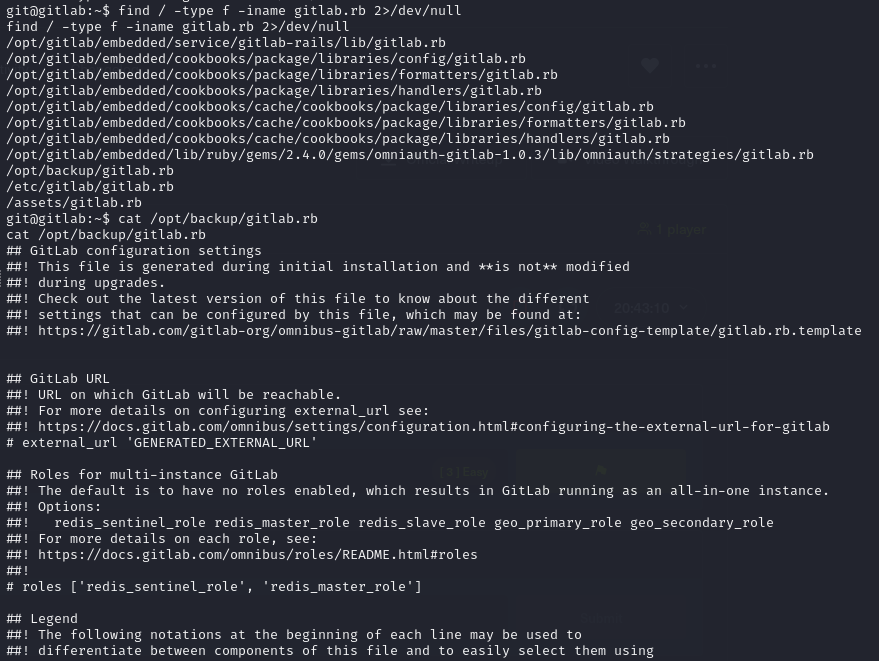


Figure - gitlab.rb file

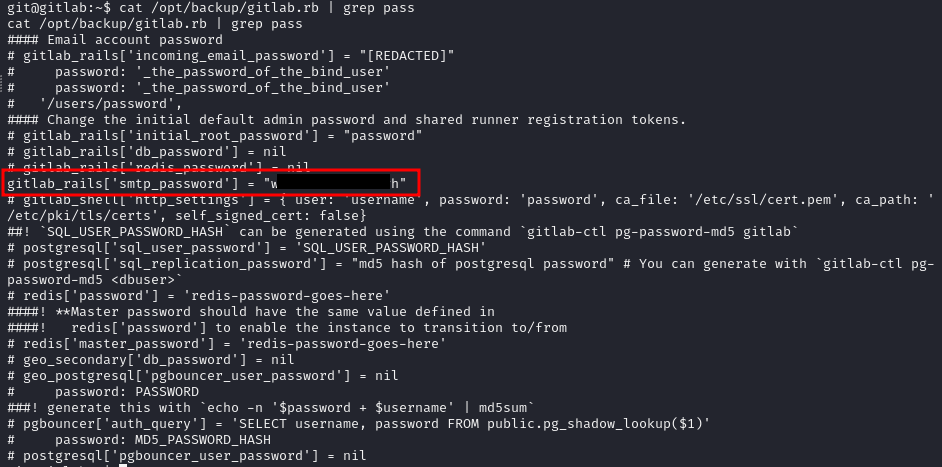


Figure 6 - Password found in gitlab.rb file

I was lucky this time because I became root using this password running the command:



Figure 7 – Container root user

At this point I looked for the root flag, but it wasn’t there and it was very strange. Honestly, I thought for a while that creator forgot to insert the root flag. Even if command told me I was root, there were something strange. For example, the root flag didn’t exist and command didn’t work. Therefore, I investigated more and I found other interesting signs. In particular, the process with PID 1 wasn’t the process and the process didn’t exist on the box, as shown in the following pictures:

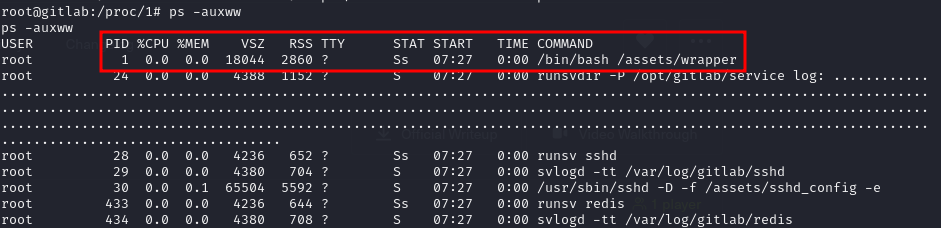


Figure 8 - PID 1 wasn't init process

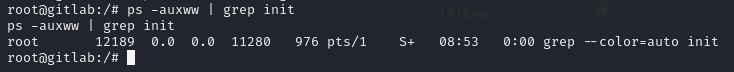


Figure 9 - Init process not found

In addition, even Linpeas provided some proofs that I was in a container:

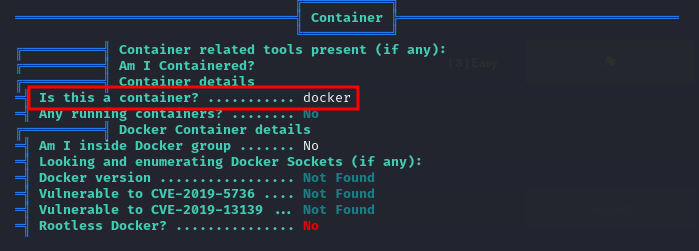


Figure 10 - Docker container recognized by Linpeas

Linpeas was useful to find a clue on how to exploit the container, as well:

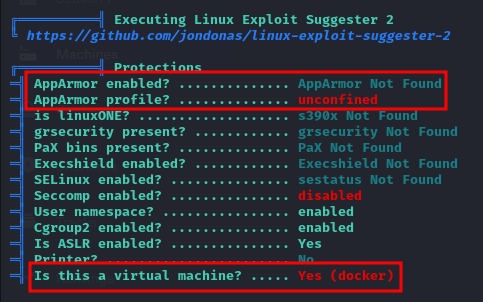


Figure 11 - Container exploitation clue

In particular, Linpeas suggested the following two methods:

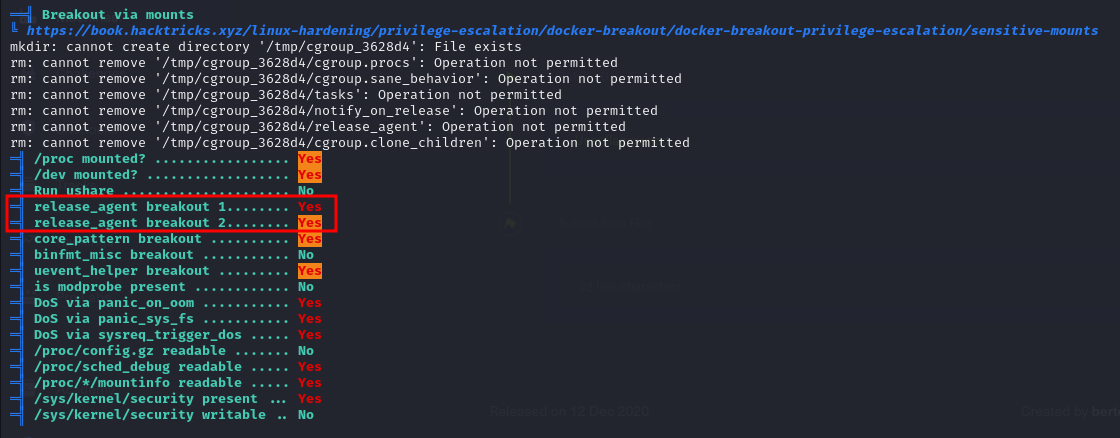


Figure 12 - Docker vulnerabilities

Looking on the Internet for some way to exploit the container, I found some interesting tutorial. First of all, I needed to check which enabled capabilities I had:

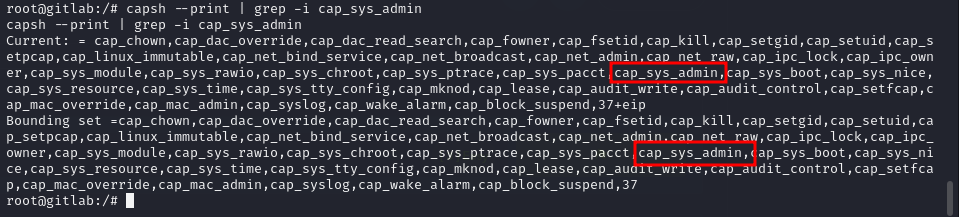


Figure 13 - Enabled capabilities

Luckily, the capability was enabled. Second, I needed to check if AppArmor was running:

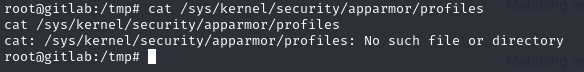


Figure 14 - AppArmor running check

Since the file was empty or didn’t exist, AppArmor was not running. This condition was perfect to run the exploit I found. Following the tutorial, I was able to became root on the machine and retrieve the root flag:

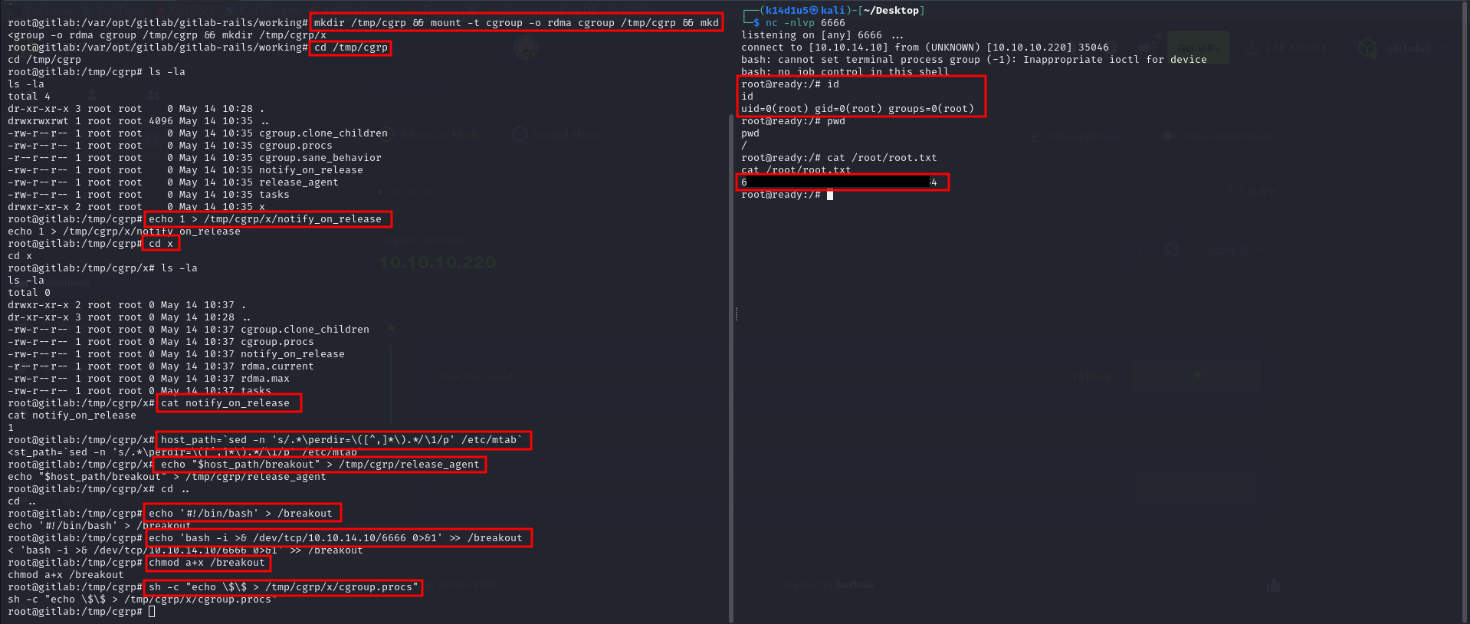


Figure 15 - Privilege escalation and root flag

# **Personal comments**

I was very surprised by this box. The most interesting part was the privilege escalation. This was the first time I needed to exploit a container to retrieve a flag. It was not easy because I had some clues that could make me think I was really root, but it was just in a container. So, I learned about I need to pay attention to some other details and I improved my skills for sure. However, I lost a lot of time because of the GitLab exploit. In fact, some of them I tried it was not correct. They missed a very important part to make them work. In conclusion, it was a very good and interesting box and I evaluate medium on platform (if I remember well).

# **Appendix A – CVE-2018-19571**

This vulnerability affects an unknown code of the *Webhooks* component. The manipulation with an unknown input leads to a server-side request forgery vulnerability. The web server receives a URL or similar request from an upstream component and retrieves the contents of this URL, but it does not sufficiently ensure that the request is being sent to the expected destination.

# **Appendix B – CVE-2018-19585**

This vulnerability affects an unknown code of the component *Project Mirroring*. The manipulation with an unknown input leads to a CRLF injection vulnerability. The product uses CRLF (carriage return line feeds) as a special element, e.g. to separate lines or records, but it does not neutralize or incorrectly neutralizes CRLF sequences from inputs.

# **References**

1. GitLab 11.4.7 Remote Code Execution - <https://liveoverflow.com/gitlab-11-4-7-remote-code-execution-real-world-ctf-2018/>;
2. GitLab 11.4.7 Remote Code Execution PoC - <https://github.com/dotPY-hax/gitlab_RCE>;
3. Understanding Docker container escaping - <https://blog.trailofbits.com/2019/07/19/understanding-docker-container-escapes/>;
4. Container escaper checker - <https://github.com/teamssix/container-escape-check>;
5. Digging into cgroups escaping - <https://0xdf.gitlab.io/2021/05/17/digging-into-cgroups.html>.