**Doctor walkthrough**

# **Index**

[Index 1](#_Toc168590440)

[List of pictures 1](#_Toc168590441)

[Disclaimer 2](#_Toc168590442)

[Reconnaissance 2](#_Toc168590443)

[Initial foothold 2](#_Toc168590444)

[User flag 3](#_Toc168590445)

[Privilege escalation 5](#_Toc168590446)

[Appendix A – CVE explanation 6](#_Toc168590447)

[CVE-2021-3156 6](#_Toc168590448)

# **List of pictures**

[Figure 1 - nMap scan results 2](#_Toc168590451)

[Figure 2 - Email domain found 2](#_Toc168590452)

[Figure 3 - New resource found 3](#_Toc168590453)

[Figure 4 - Login page found 3](#_Toc168590454)

[Figure 5 - SSTI exploit 4](#_Toc168590455)

[Figure 6 - shell.sh code 4](#_Toc168590456)

[Figure 7 - User shell 4](#_Toc168590457)

[Figure 8 - Change password request 4](#_Toc168590458)

[Figure 9 - User flag 5](#_Toc168590459)

[Figure 10 - Sudo vulnerable version 5](#_Toc168590460)

[Figure 11 - Privesc and root flag 6](#_Toc168590461)

# **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’re willing and curious to know and learn about Ethical Hacking, Security and Penetration Testing.

# **Reconnaissance**

The results of an initial nMap scan are the following:

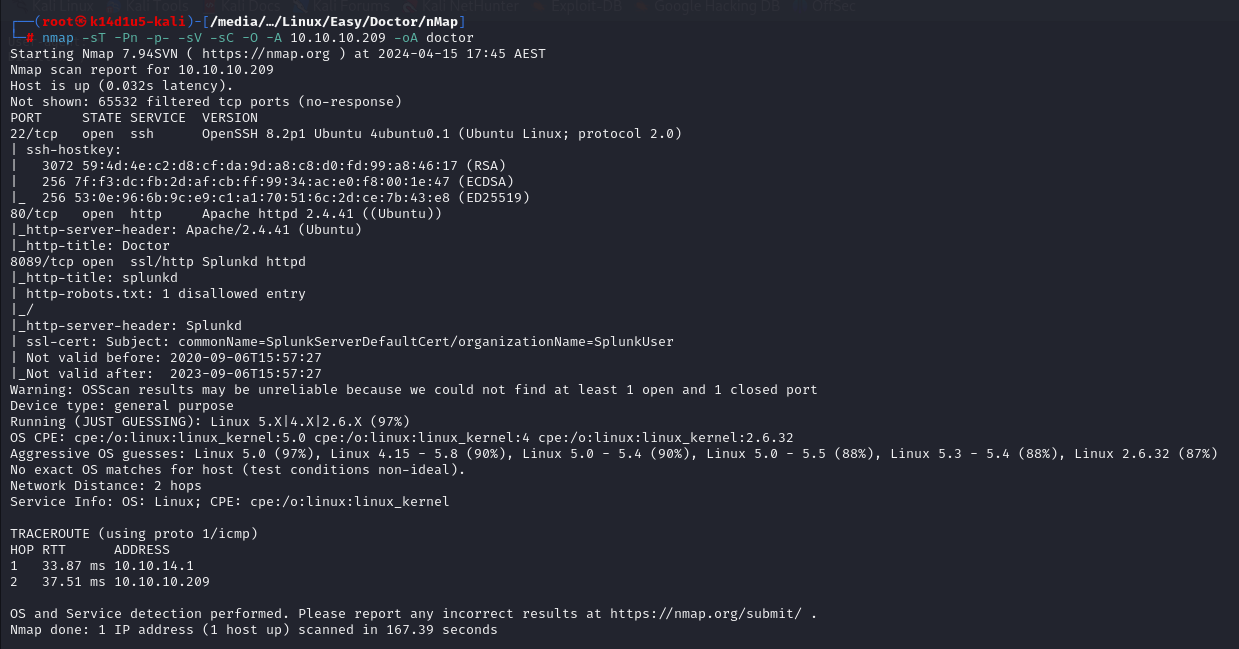


Figure 1 - nMap scan results

Open ports are 22, 80 and 8089. So, nMap told me that the box has SSH service enabled, an application running on port 80 and a Splunk service running on port 8089. Also, nMap detected a Linux OS, probably Linux 5.0.

# **Initial foothold**

Analyzing the web site, I found some interesting information. First of all, I found some email with as domain:

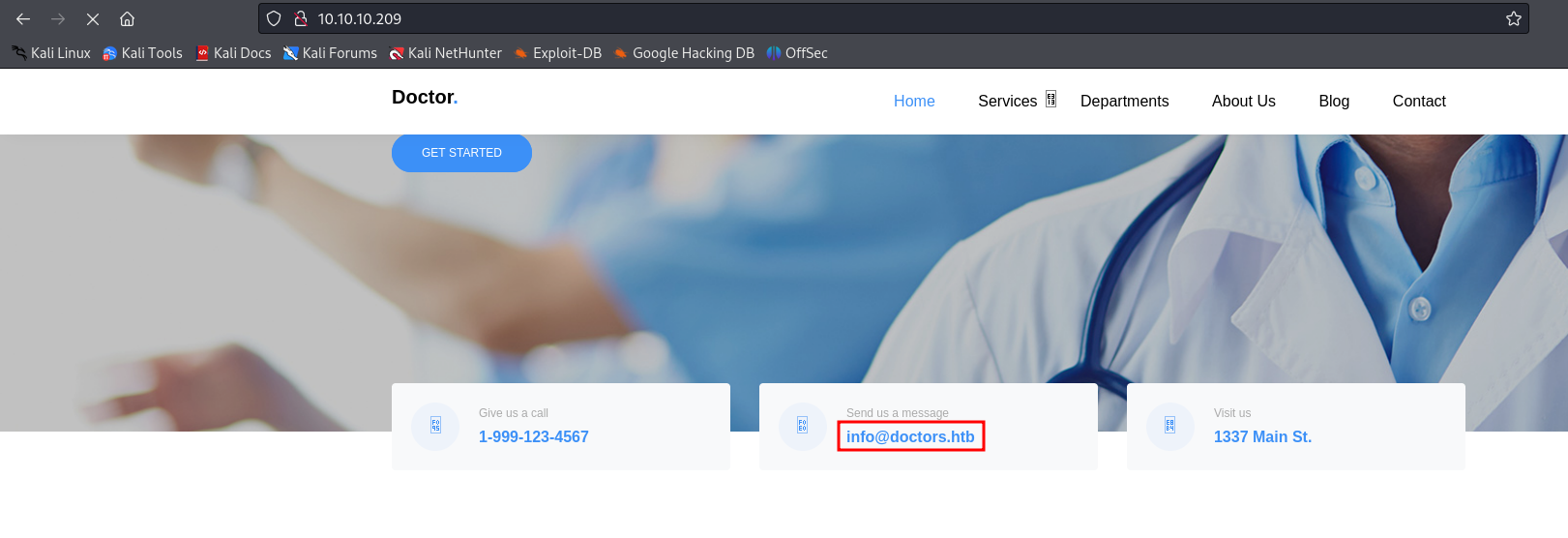


Figure 2 - Email domain found

Also, I found a hidden page analyzing the web application source code:

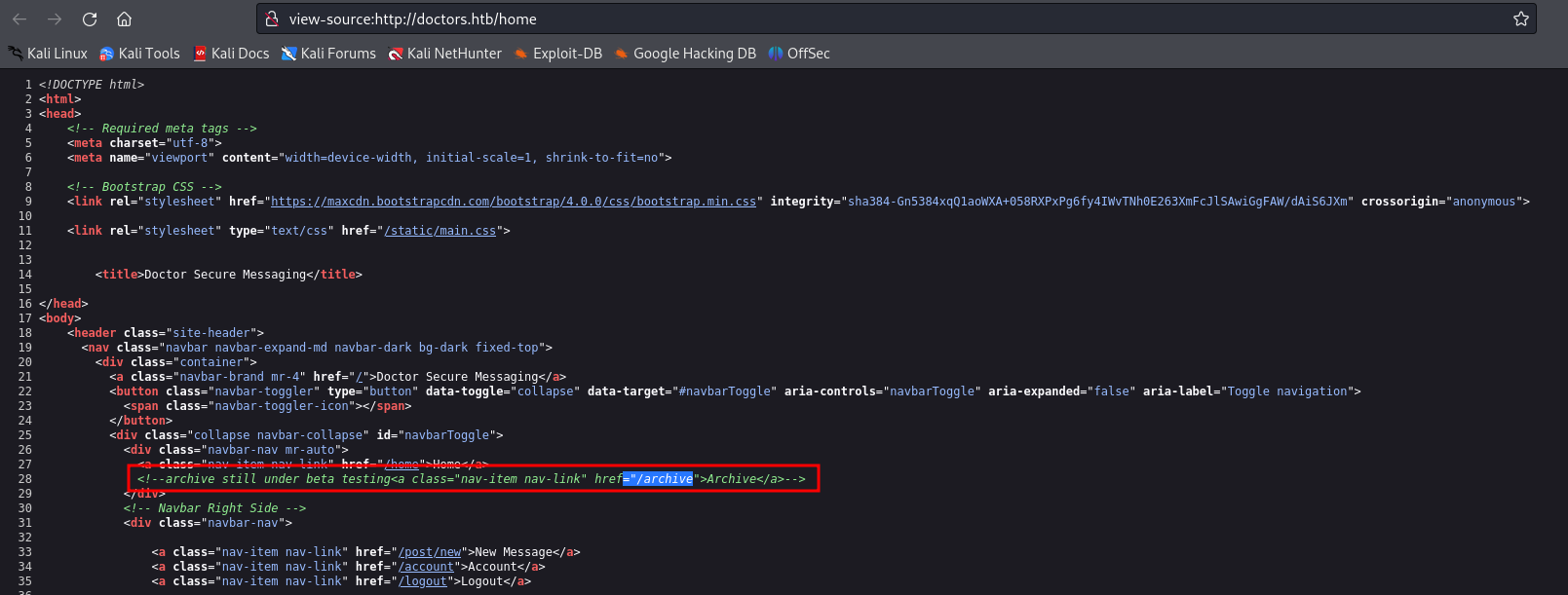


Figure 3 - New resource found

Since I found a domain, I use it to create a new entry in my file. Also, I use this domain to run Dirbuster. In this case, I found a login page:

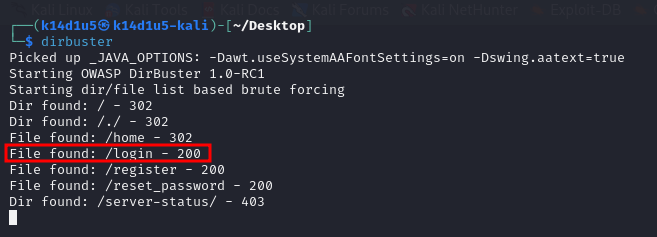


Figure 4 - Login page found

In this page, I can register an account. I do it and in the logged page I can insert a message. In particular, any message I send, it will be shown in the **/archive** page I found from the source code. I tried to exploit an XSS vulnerability and it worked, but it is not useful in this case.

# **User flag**

To get a shell, the right vulnerability to exploit in the logged page is the SSTI vulnerability. To do it, I used the payload in the following picture:

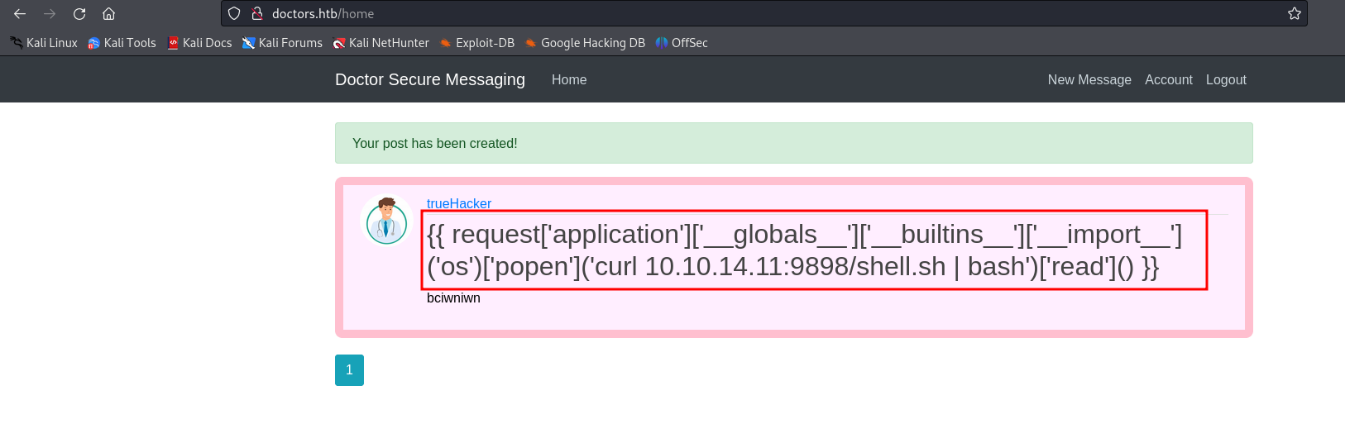


Figure 5 - SSTI exploit

The shell.sh file contains the following code:

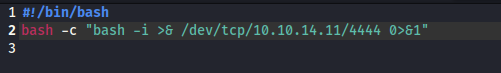


Figure 6 - shell.sh code

Using this payload, I received a shell, as shown in the following picture:

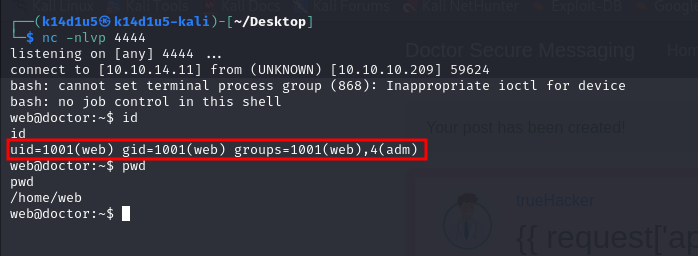


Figure 7 - User shell

However, this user is not the one that has the user flag. So, I uploaded and run Linpeas. In this way I found an interesting file called **/var/log/apache2/backup**. Inside it, I found the following change password request where the **email** parameter contains a password:

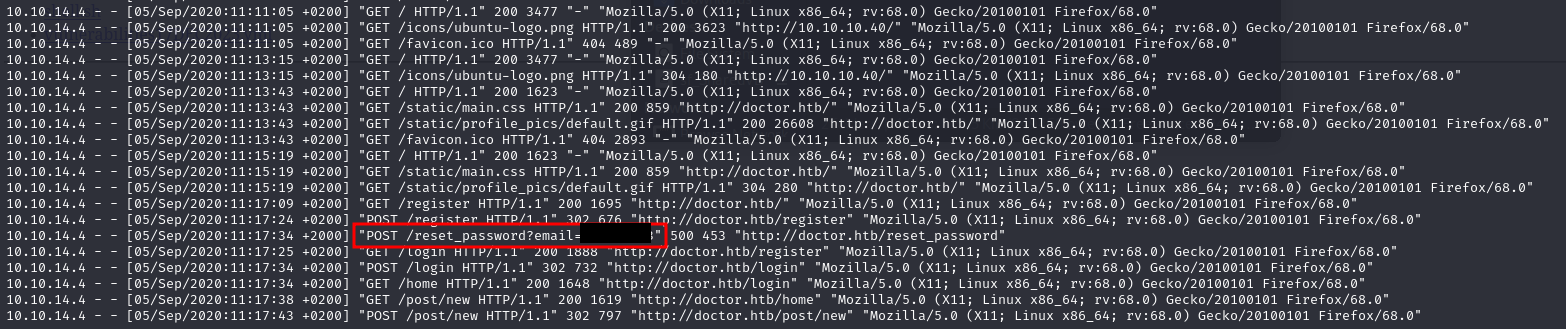


Figure 8 - Change password request

Analyzing the box, I found an account named **shaun**, so I used these credentials to try to log as **shaun** user:

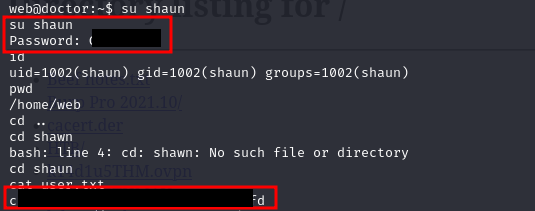


Figure 9 - User flag

As shown in the previous image, I succeed to log in as **shaun** an I retrieved the user flag.

# **Privilege escalation**

Now it is time to elevate privileges. To do it, I run again Linpeas and I found a sudo vulnerable version to the CVE-2021-3156:

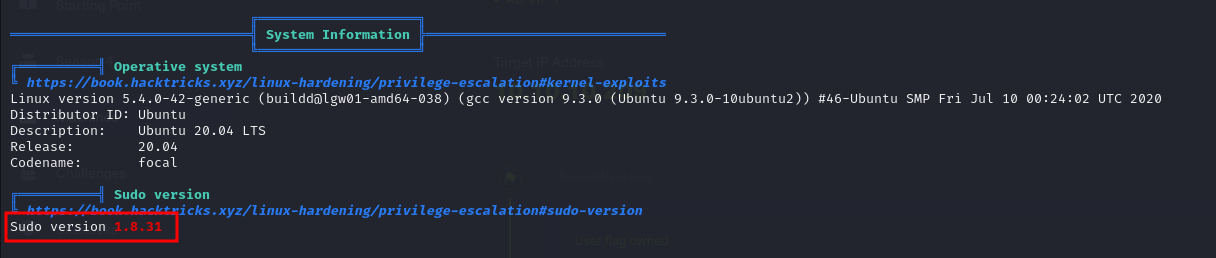


Figure 10 - Sudo vulnerable version

So, I downloaded the respective exploit and run. In this way, I became root and I retrieve the root flag:

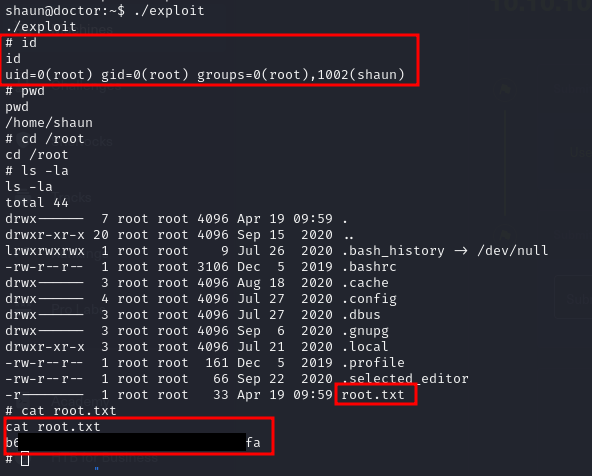


Figure 11 - Privesc and root flag

# **Appendix A – CVE explanation**

## **CVE-2021-3156**

The researcher Baron Samedit discovered that:

* Executing **sudo** in “shell” mode () and
* Using the sudoedit command with the options ( flag) or ( and flags);
* It’s possible to escape special characters in the command’s arguments using a backslash at the end.

A bug in the **sudo** code, related to the **sudoedit** command, permits to avoid the escape characters and overflow the heap-based buffer through a command-line argument that ends with a single backslash character.

From an attacker perspective, this buffer overflow vulnerability allows the attacker to control the size of the buffer, and control the contents (using null bytes if necessary) of the overflow itself. This is what enables the malicious user to execute custom code on the host with ***root*** privileges.