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

Overview

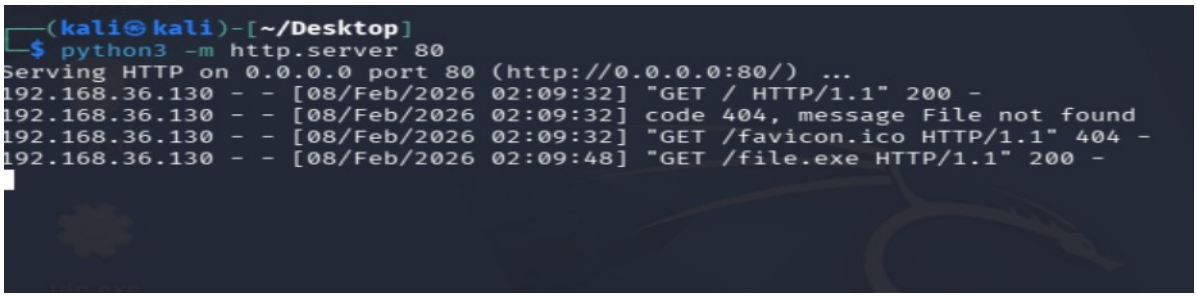
This section documents the conceptual process of persistence establishment on a Windows system within a controlled laboratory environment. The objective was to understand how attackers maintain access after an initial compromise and how such techniques can be detected and mitigated by defenders.

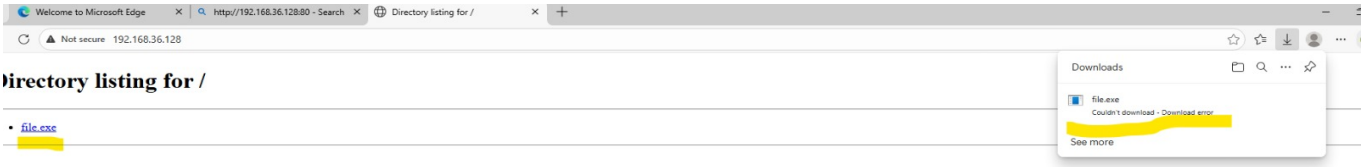
Step 1: Payload Creation

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.

Step 2: Payload Delivery

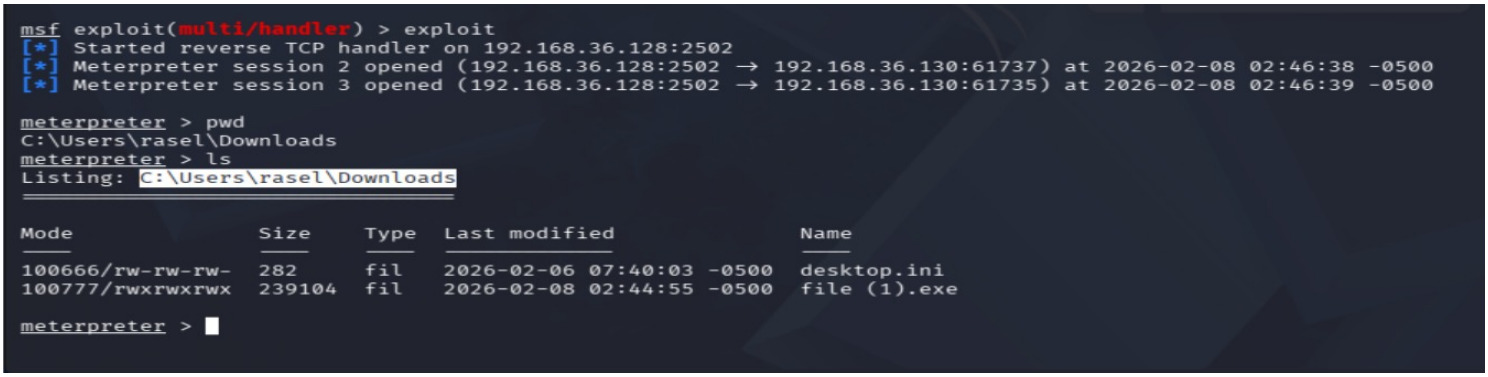
The payload file was transferred to the Windows machine using a temporary Python-based web server.





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.



Step 4: Persistence Script Preparation

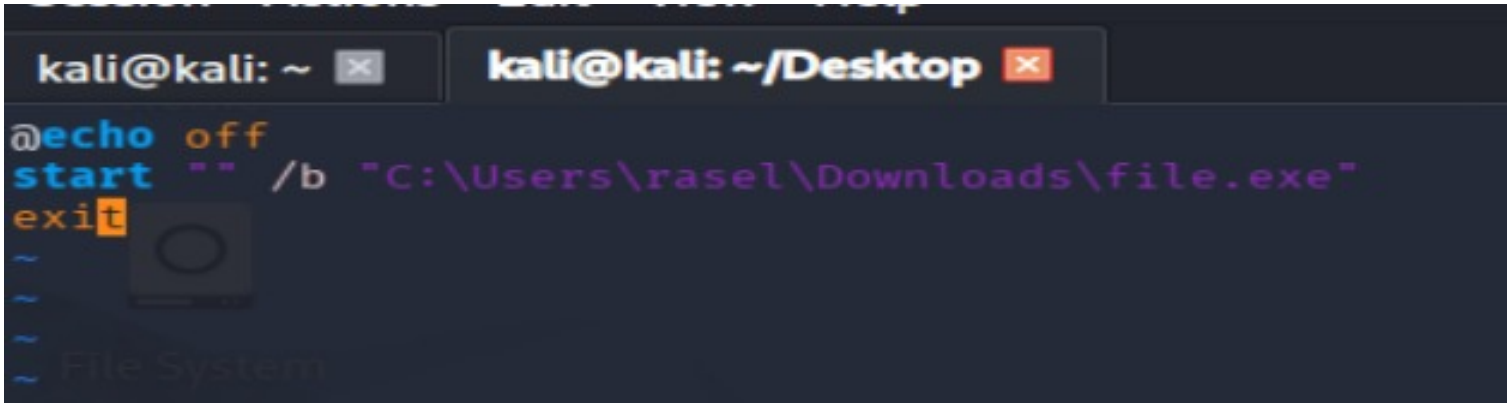
To simulate persistence techniques, automation scripts were created to execute the payload silently in the background during system startup.

Two scripting approaches were evaluated:

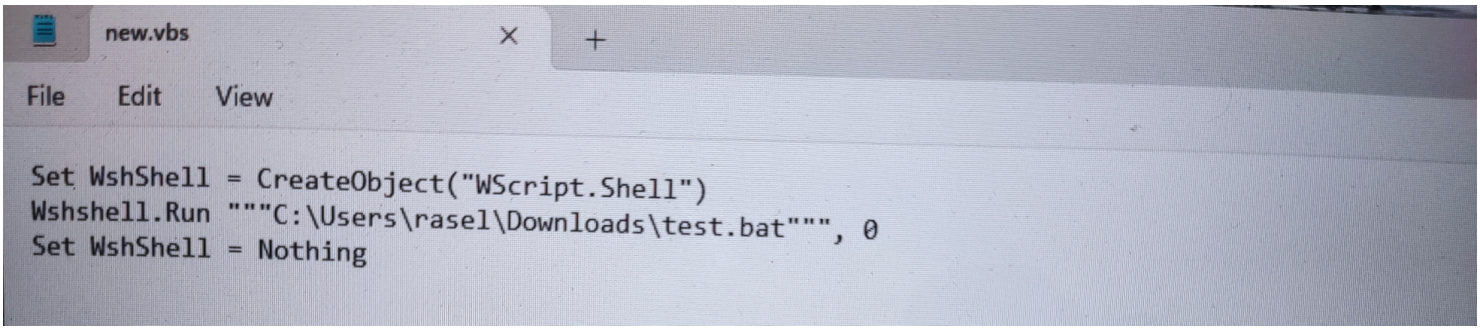
- Batch script (.bat)
- VBScript (.vbs)

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

Batch file (.bat):



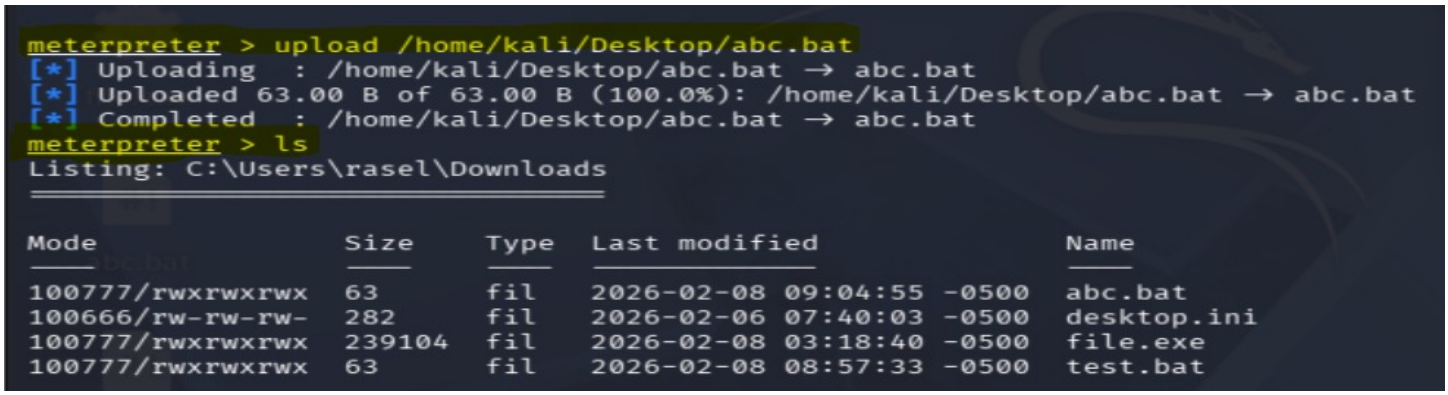
VBScript file (.vbs):



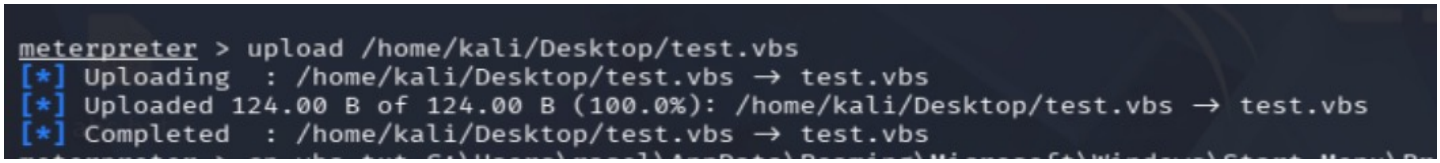
Step 5: Uploading the Script File

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:

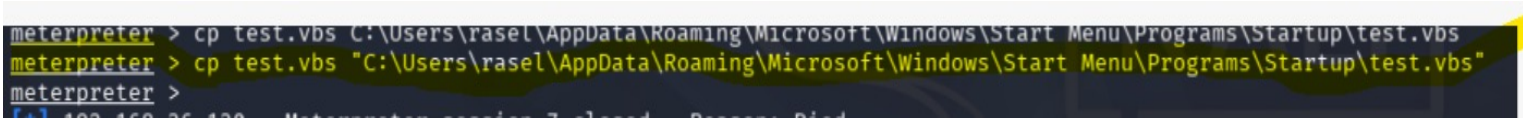


VBScript file uploading process:



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.



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.