EXPT NO: 07 ROLL NO: 220701216

WINDOWS PRIVILEGE ESCALATION

AIM:

- Understand the techniques used to escalate privileges on Windows systems.

- Explore common misconfigurations and vulnerabilities in real-world and CTF scenarios.

- Use command-line tools and scripts to extract credentials, access restricted files, and gain elevated privileges.

- Learn how to identify weak service permissions, unquoted paths, and insecure configurations.

- Practice manual enumeration and exploitation techniques on Windows targets.

PROCEDURE:

1. Analyze PowerShell history and configuration files for stored passwords.

2. Use `cmdkey`, `runas`, and PuTTY saved sessions to access other users' contexts.

3. Exploit vulnerable scheduled tasks and services with weak permissions.

4. Leverage misconfigured services to replace binaries or abuse quoted paths.

5. Dump SAM and SYSTEM hashes using SeBackup/SeRestore privileges.

6. Use tools like Impacket and PsExec to reuse credentials and access Admin accounts.

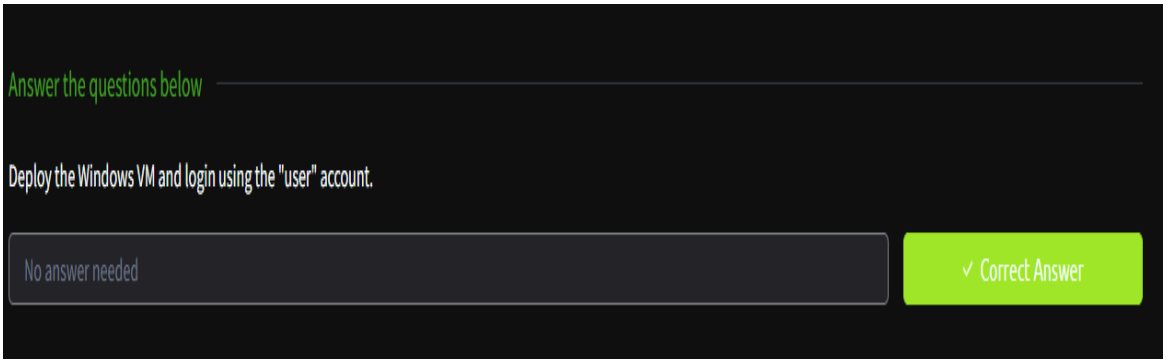
7. Exploit DLL hijacking in vulnerable software to escalate to SYSTEM.

TASK 1 – INTRODUCTION

- Introduced Windows privilege escalation and the goal of moving from standard user to Administrator or SYSTEM.

- Outlined common misconfigurations such as services, credentials, or software flaws.

- Emphasized safe enumeration, log review, and privilege abuse methods.

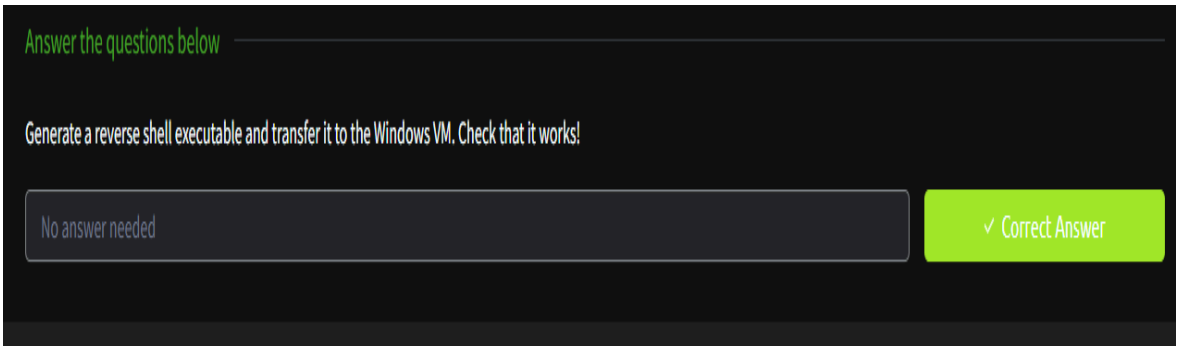


TASK 2 – ENUMERATION

- Used `whoami`, `systeminfo`, and `net user` to enumerate the system.

- Gathered information about privileges, group membership, and architecture.

- Reviewed running services and user permissions for potential escalation paths.



TASK 3 – PASSWORD RECOVERY TECHNIQUES

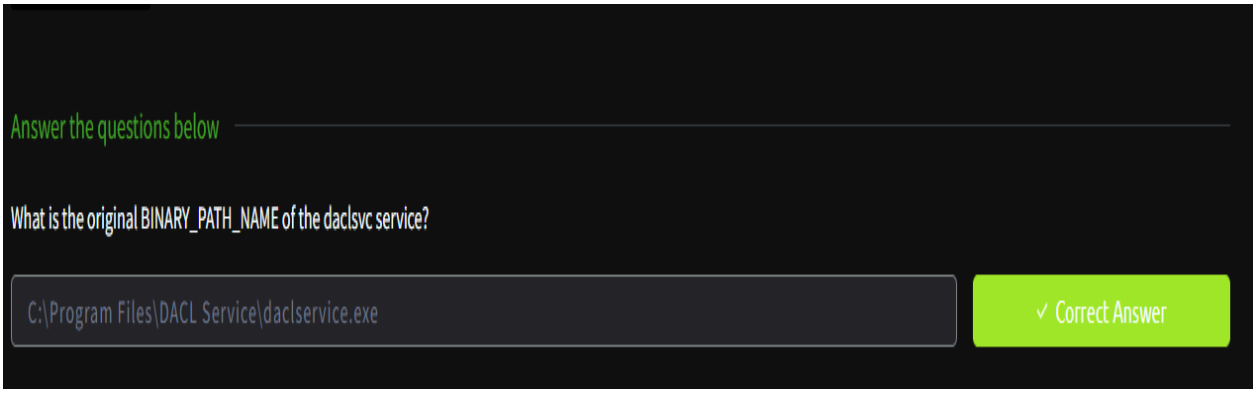
- Extracted password from PowerShell history: `ZuperCkretPa5z` (julia.jones).

- Found plaintext DB password in IIS web.config: `098n0x35skjD3` (db_admin).

- Used `cmdkey` and `runas` to access mike.katz desktop and retrieve flag.

- Retrieved PuTTY password for thom.smith from saved session: `CoolPass2021`.

- Highlighted the importance of managing credentials securely on endpoints.



TASK 4 – SCHEDULED TASK ABUSE

- Exploited scheduled task with write access to inject a reverse shell script.

- Used `nc64.exe` to set up a backdoor connection with listener.

- Triggered task to gain access as taskusr1 and retrieved the flag.

- Demonstrated the danger of insecure task permissions.

- Emphasized checking scheduled task configurations on Windows environments.

TASK 5 – WEAK SERVICE PERMISSIONS

- Used `icacls` to find modifiable service executables.

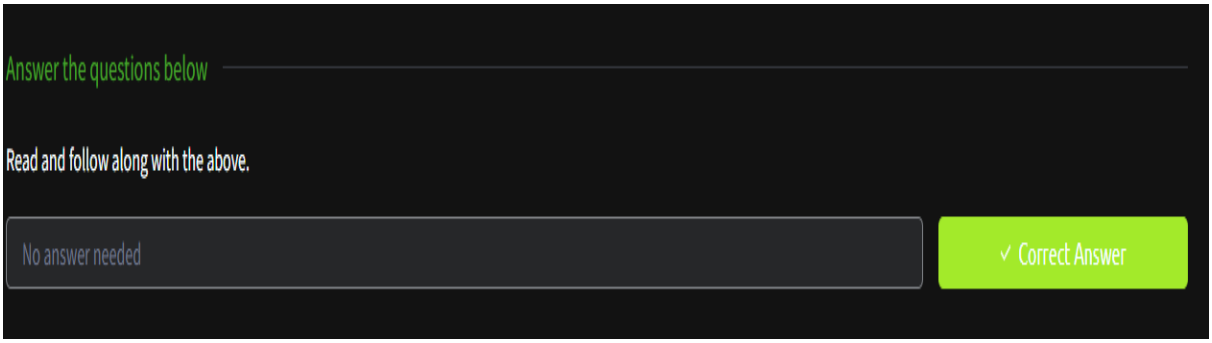
- Created reverse shell with `msfvenom`, downloaded via Python HTTP server.

- Replaced vulnerable service binary and restarted service to gain shell.

- Retrieved flags from svcusr1 and svcusr2 using modified services.

- Demonstrated exploitation of unquoted service paths.

- Gained access to Administrator desktop by editing THMService configuration.



TASK 6 – SeBackup/SeRestore PRIVILEGE ESCALATION

- Verified `SeBackupPrivilege` using `whoami /priv`.

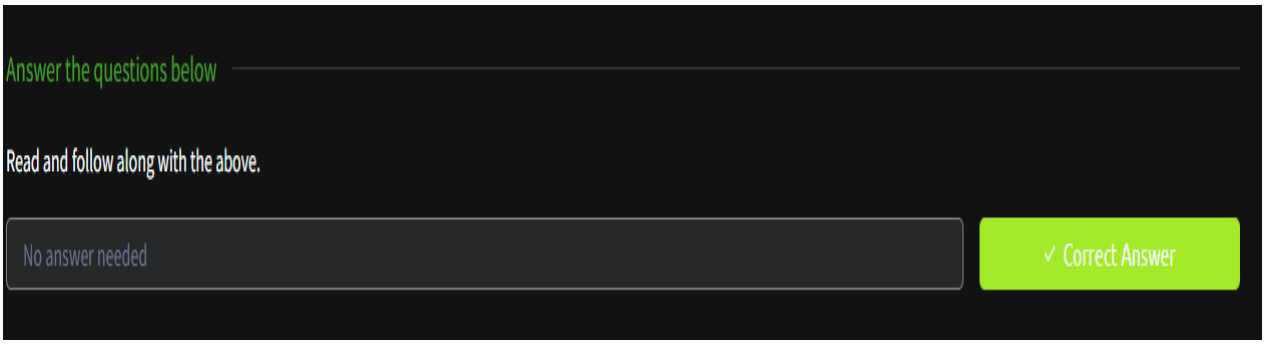
- Backed up SAM and SYSTEM files and shared them via SMB server.

- Used `secretsdump.py` to extract password hashes from backups.

- Used `psexec.py` and the dumped hash to gain Administrator shell.

- Retrieved final flag from Administrator's desktop.

- Demonstrated the risk of powerful backup privileges in misconfigured roles.



TASK 7 – VULNERABLE SOFTWARE EXPLOITATION

- Identified vulnerable software with DLL search order hijacking issues.

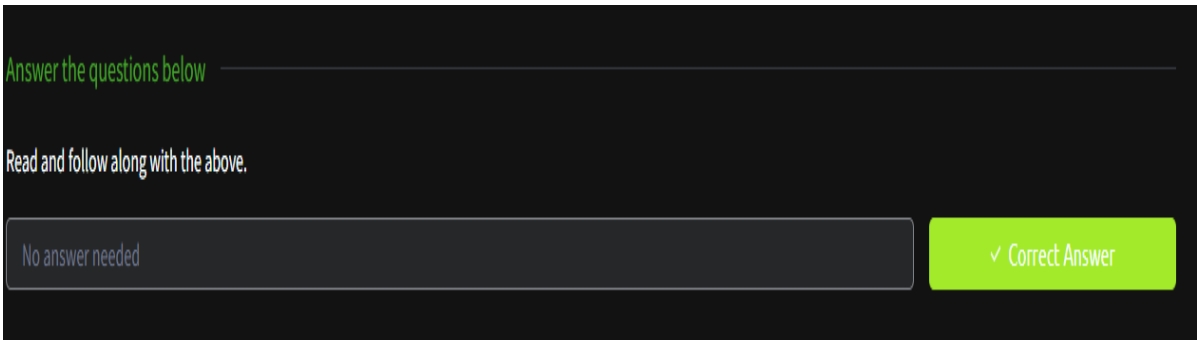
- Modified public exploit script to add new user to Administrators group.

- Used PowerShell ISE to execute exploit payload.

- Handled Defender blocking and script errors via debugging.

- Gained Administrator privileges by replacing DLL in path.

- Retrieved final flag from Administrator's desktop.



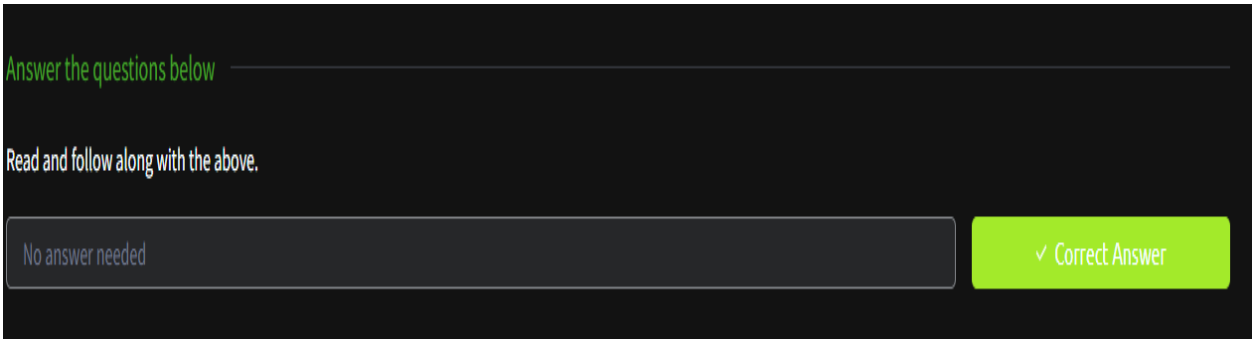
TASK 8 – UNQUOTED SERVICE PATH EXPLOIT

- Analyzed service executable paths lacking quotes.

- Placed malicious reverse shell at expected path segment (e.g., `C:\Program.exe`).

- Restarted the service to trigger unintended binary execution.

- Gained admin shell and extracted the final flag.



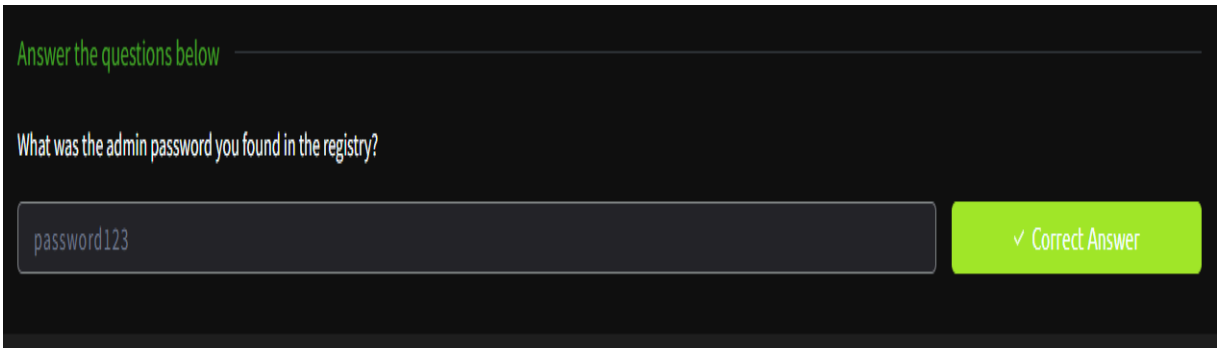
TASK 9 – SERVICE BINARY REPLACEMENT

- Identified service binaries with write permissions.

- Replaced default executable with malicious one using reverse shell payload.

- Restarted service to run malicious binary and elevate access.

- Confirmed access by retrieving the Administrator's desktop flag.

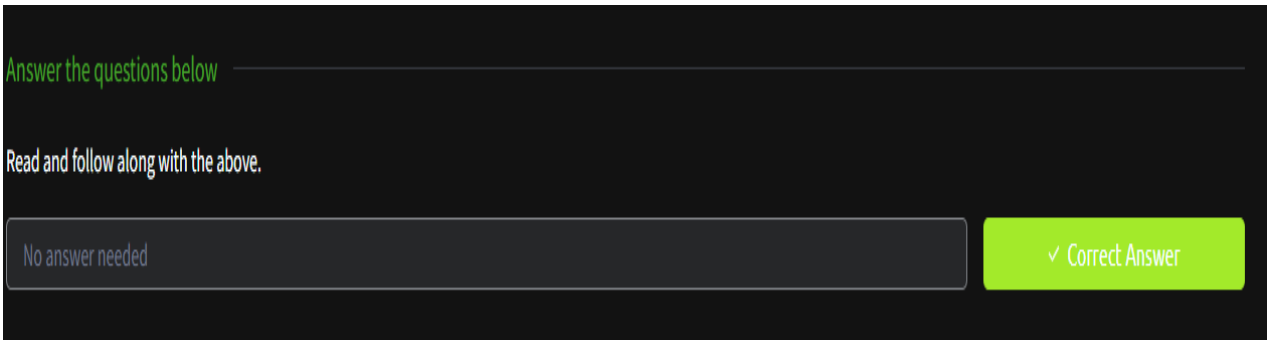


TASK 10 – RDP ACCESS TO ADMIN ACCOUNT

- Established RDP connection using cracked or injected admin credentials.

- Used GUI or command-line tools to enumerate services and extract flags.

- Navigated to `C:\Users\Administrator\Desktop` for final flag retrieval.

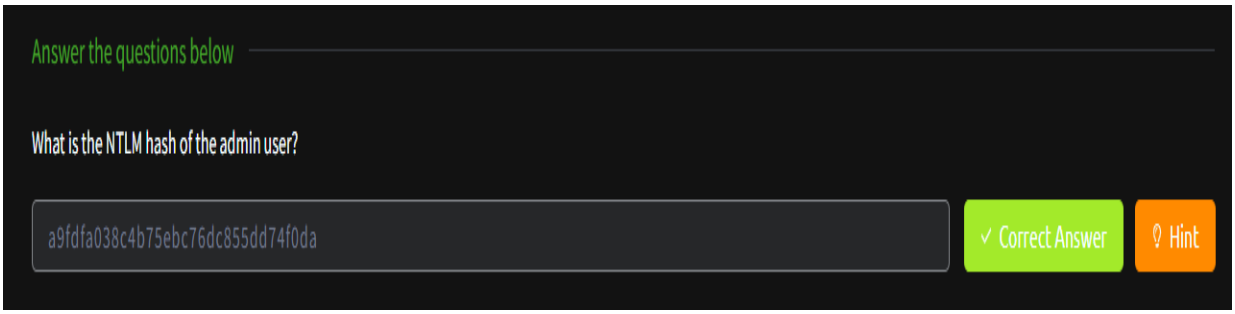


TASK 11 – PASSWORD HASH REPLAY

- Dumped password hashes using `secretsdump.py`.

- Replayed administrator hash using `psexec.py` for shell access.

- Bypassed password authentication using NTLM replay techniques.



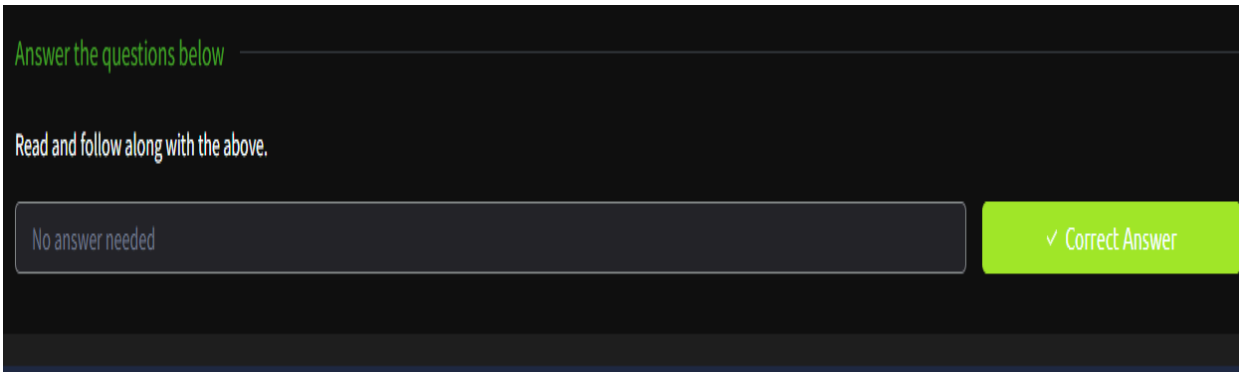
TASK 12 – SOFTWARE ESCALATION VIA LOGIC FLAW

- Discovered vulnerable software with post-install logic flaw.

- Injected admin creation or shell commands into config files.

- Triggered software to execute logic and escalate privileges.

- Verified access via new user and confirmed via desktop flag.

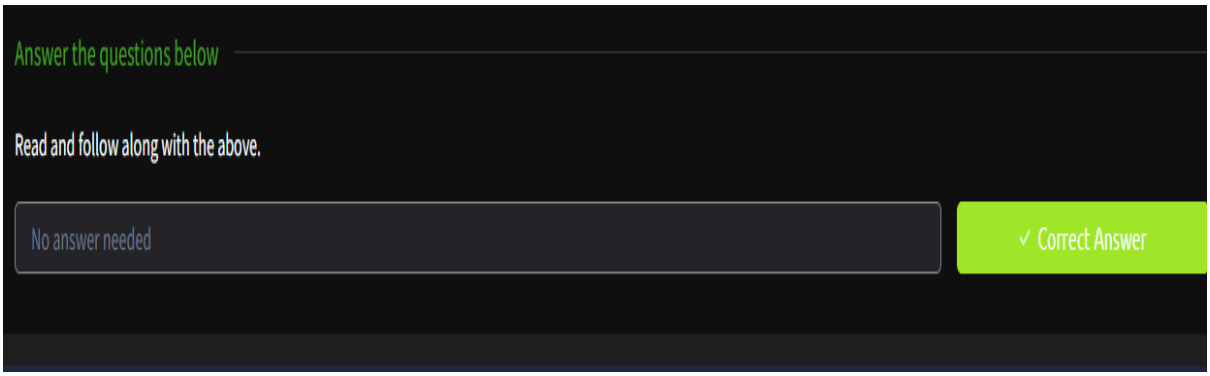


TASK 13 – MIMIKATZ CREDENTIAL DUMP

- Used Mimikatz to dump plaintext passwords and ticket hashes from memory.

- Extracted domain or local admin credentials from LSASS.

- Used credentials to switch users or access protected files.

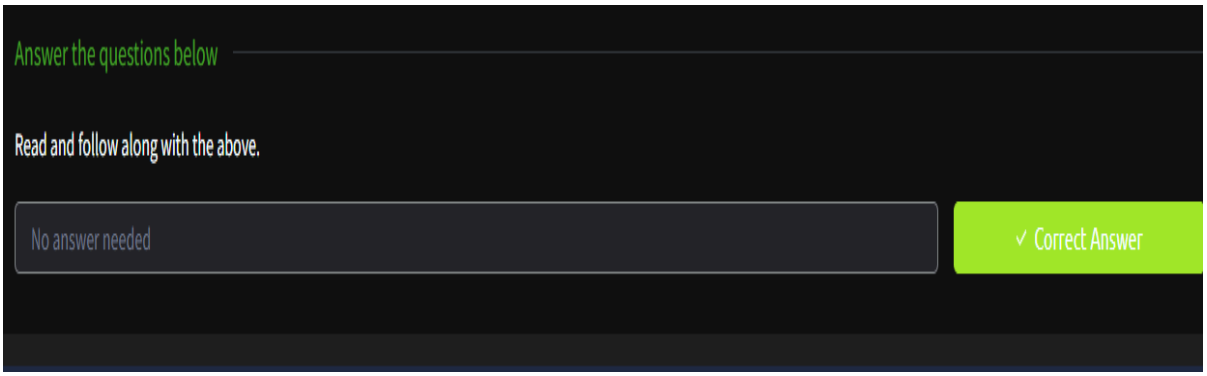


TASK 14 – DLL INJECTION IN ADMIN PROCESS

- Injected malicious DLL into a process running as Administrator.

- Hooked into process memory and executed payload to spawn elevated shell.

- Confirmed privilege escalation by listing user tokens.

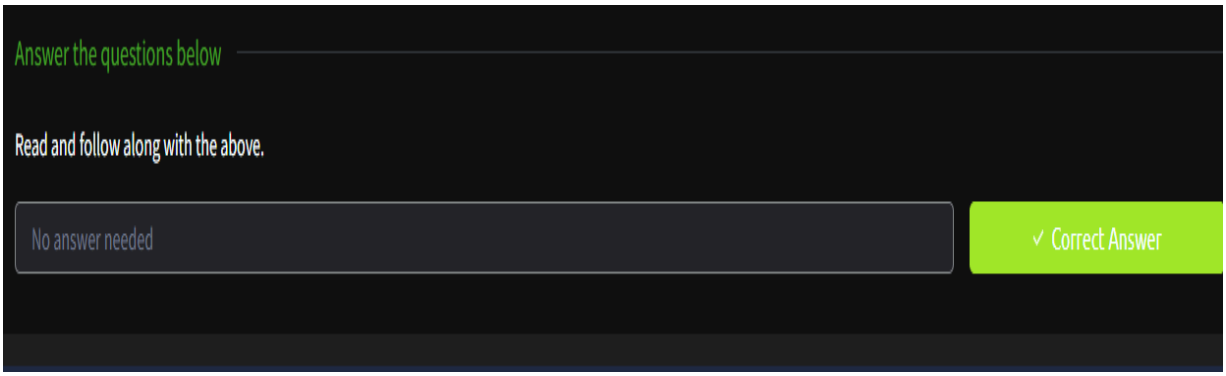


TASK 15 – HIJACKED AUTO-RUN ENTRY

- Detected auto-run registry keys pointing to writable paths.

- Modified the executable path to a reverse shell payload.

- Restarted system or triggered user login to execute malicious payload.

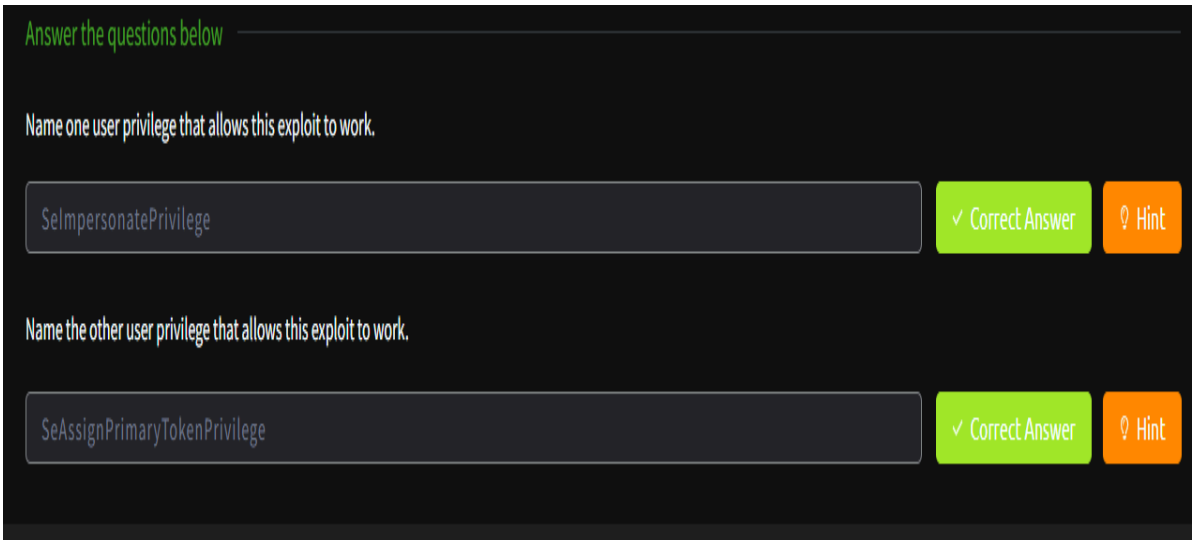


TASK 16 – VULNERABLE DRIVER EXPLOIT

- Identified kernel-mode driver with local privilege escalation flaw.

- Compiled and executed exploit to write arbitrary data to memory.

- Elevated from user to SYSTEM using crafted exploit.



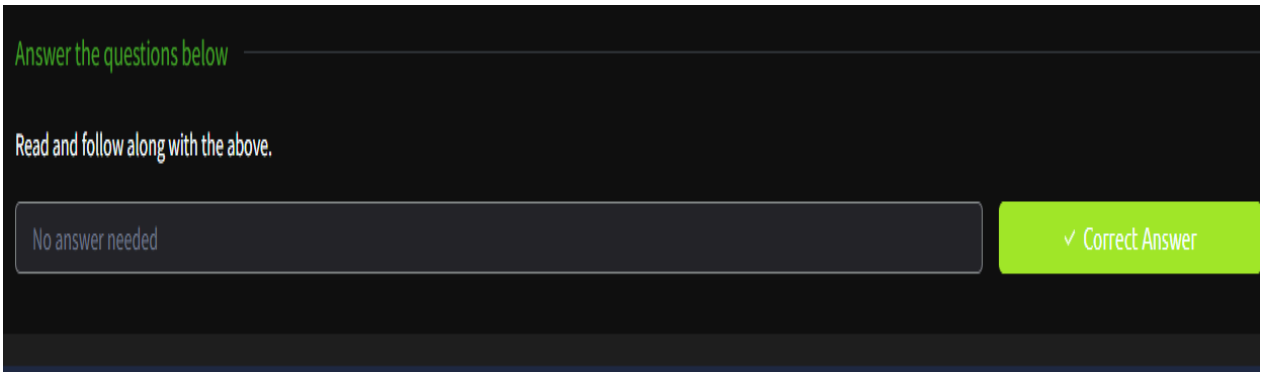
TASK 17 – ABUSING INSTALLER PACKAGES

- Found MSI or installer scripts run with elevated privileges.

- Modified installer to spawn shell or add user.

- Ran installer to achieve privilege escalation.

- Captured Administrator flag on success.



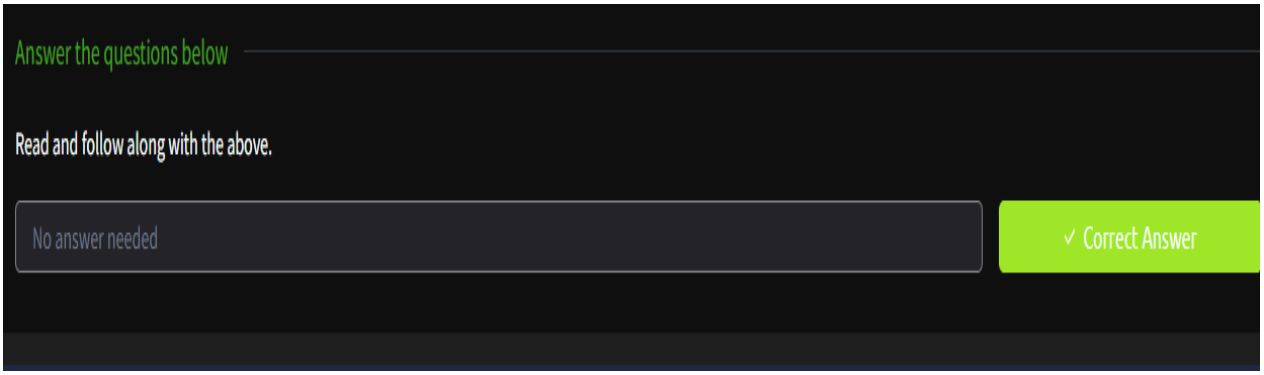
TASK 18 – CLEANUP AND PERSISTENCE

- Created persistence mechanism using scheduled tasks or registry keys.

- Deleted evidence such as shells, temp files, and logs.

- Removed added users and restored binaries to cover tracks.

- Documented steps taken for later analysis and reporting.



RESULT:

Completed the TryHackMe “Windows Privilege Escalation” room, demonstrating multiple vectors for gaining elevated access on a Windows host. Learnt how to identify and exploit weak configurations, extract sensitive data, and manipulate services to gain control. Skills acquired are directly applicable to real-world penetration testing and red-team simulations.