## Incident Response and Handling

In this project, I developed and executed an **Incident Response Plan (IRP)** to handle a simulated security breach within a cloud-hosted inventory management system. This system stored internal product data, supplier information, and some customer purchase history. The incident involved suspicious activity on the server hosting the system, which triggered an in-depth investigation and remediation process.

## Incident Response Plan (IRP): 5-Step Framework

To ensure effective incident handling, I followed the **5-step Incident Response Plan (IRP)** framework, ensuring a structured and systematic approach to manage the security breach.

**1. Preparation**

Preparation is the foundation of any effective incident response strategy. Before the incident, I ensured that all cloud servers were equipped with centralized logging, endpoint detection systems (CrowdStrike Falcon), and automatic backup policies. Additionally, I conducted tabletop exercises to familiarize the response team with the protocols in case of a real breach. This proactive approach ensured that we were ready for quick and efficient action.

**2. Identification**

The identification phase began when the monitoring system flagged unusual **outbound traffic** from the inventory backend server at 2:12 AM. I cross-referenced this activity with the system logs and confirmed that unauthorized access had occurred, as evidenced by elevated user privileges being leveraged without proper authorization. This raised the alarm that a security breach was in progress.

**3. Containment**

Immediately following identification, the next step was containment. I isolated the compromised virtual machine (VM) from the rest of the network by implementing **Azure Network Security Group (NSG) rules**, effectively blocking all inbound and outbound connections. This prevented the attacker from moving laterally within the environment and stopped any potential data exfiltration in its tracks.

**4. Eradication**

Once containment was confirmed, the eradication process began. I ran a full system scan with **CrowdStrike Falcon**, identifying a **malicious PowerShell script** running via a scheduled task. The attacker had used this script to maintain persistent access. I removed the script from the system, disabled the task, and reset all credentials associated with the compromised VM to prevent further access.

**5. Recovery**

With the breach contained and eradicated, I proceeded with system recovery. I restored the affected VM from the most recent clean backup, verified the integrity of all configurations, and reintroduced the system into the network under heightened monitoring. As an added precaution, I enforced **Multi-Factor**

**Authentication (MFA)** for all admin accounts to ensure that the risk of similar incidents occurring in the future was minimized.

## Digital Forensics: Tool Used

For the digital forensics portion of the incident, I employed **Autopsy**, a GUI-based digital forensic platform. Autopsy provided the necessary tools to conduct a thorough forensic investigation on the compromised VM.
Key actions included:

- **Extracting and analyzing Windows Event Logs:** I examined logs to identify unauthorized login attempts and unusual script executions.
- **Identifying unusual process activity:** I identified rogue processes and script activity that had been initiated by the attacker.
- **Collecting timestamps:** I pinpointed when the malicious PowerShell script was deployed and the duration of the attacker's login sessions.

**Evidence Collection & Chain of Custody** To ensure proper documentation and maintain the integrity of evidence, I followed established best practices for evidence handling:

1. **Log Files:** I retrieved system event logs that clearly showed unauthorized login activity and the execution of PowerShell commands related to the attack.
2. **Screenshot:** I captured a screenshot of the Task Scheduler, showing the rogue scheduled task that was executing the malicious PowerShell script.
3. **Chain of Custody Documentation:** A detailed log was maintained throughout the process to document:
    - Who accessed the evidence.
    - When and where it was collected.
    - How it was transferred and stored.

Every step of the evidence collection was carefully recorded and updated to ensure full compliance with digital forensics best practices.

## Incident Triage and Prioritization

To efficiently respond to multiple incidents, I developed a **triage model** to classify incidents by severity and their potential impact on the business. This classification helped prioritize incidents and determine the appropriate response actions.

| Incident Type | Severity | Business Impact | Response Priority |
|---|---|---|---|
| **Unauthorized Admin Access** | High | Direct threat to system integrity and data exfiltration | Urgent |

| | | | |
|---|---|---|---|
| **Phishing Attempt on Staff Email** | Medium | Possible credential compromise but no system breach | High |
| **Internal User Policy Violation** | Low | Minor infraction, no sensitive data involved | Low |

In this incident, **unauthorized admin access** was classified as **High Severity**, with a direct impact on system integrity and the potential for data exfiltration. As such, the response to this incident was given **Urgent priority**, and I acted immediately to contain the breach and prevent further damage.

## Post-Incident Analysis

After resolving the incident, I conducted a **post-mortem** to review the effectiveness of our response and identify areas for improvement. The incident was successfully contained, remediated, and no data loss was confirmed.

**Key Lessons Learned:**

1. **Privileged Access Protection:**
   The breach highlighted the need for stronger controls around privileged access. I implemented **just-in-time admin access** following the incident, ensuring that administrators could only access critical systems when necessary and for a limited duration.
2. **Proactive Monitoring and Alert Tuning:**
   While the monitoring system flagged the breach early, structured alert triaging could have prevented potential delays in identifying the breach. I updated the alert configurations to ensure critical alerts were prioritized and acted upon more swiftly.
3. **Backup Integrity:**
   The recovery process was smooth due to the fact that the most recent backup was clean. However, I also implemented **monthly backup restore tests** to ensure that future restores could be conducted smoothly without issues.