Day 8: Detecting and Removing Malicious Cron Jobs

Objective

This lab simulates a scenario where an attacker establishes persistence on a Linux system using a cron job. The following narrative documents each stage of the investigation and response, step by step.

1. Initial Suspicion of Malicious Activity

Unusual system behavior raised the possibility that an attacker had created an unauthorized scheduled task. Since cron is a common persistence mechanism, the investigation focused on checking for suspicious scheduled jobs.

2. Creation of a Simulated Malicious Script

A harmless script was placed in `/tmp/` to represent a malicious payload:

```
ubuntu@ubuntu:~/script-lab$ cat << 'EOF' > /tmp/fake_malicio
us.sh
#!/bin/bash
echo "malicious cron executed at $(date)" >> /tmp/fake_cron.
log
EOF
```

The script was given execution permissions.
Attackers frequently use `/tmp/` because it is writable by all users, making it a realistic place for a malicious script.

3. Simulation of Attacker Persistence (Malicious Cron Job Added)

A cron job was added that executes the script every minute:

```
ubuntu@ubuntu:~/script-lab$ crontab -l 2>/dev/null > /tmp/cu
rrent_cron
echo "* * * * * /tmp/fake_malicious.sh" >> /tmp/current_cron

crontab /tmp/current_cron
rm /tmp/current_cron
```

This simulates a typical persistence method where a script is repeatedly triggered by the system's scheduler.

4. Detection of the Malicious Cron Job

The investigator checked the scheduled tasks using:

```
ubuntu@ubuntu:~/script-lab$ crontab -l
* * * * * /tmp/fake_malicious.sh
```

The result clearly showed a cron entry pointing to `/tmp/`, a strong indicator of malicious behavior.

```
ubuntu@ubuntu:~/script-lab$ cat /tmp/fake_cron.log
malicious cron executed at Fri Nov 21 08:39:01 PM CET 2025
```

Additional deeper inspection was performed using: This ensured that no hidden or system-level cron jobs referenced the script.

6. Removal of Malicious Persistence

The malicious cron entry was removed using:

```
ubuntu@ubuntu:~/script-lab$ crontab -l | grep -v "fake_malic
ious.sh" | crontab -
```

This command loads the crontab, filters out the malicious line, and writes back a clean version.

A follow-up check ensured that the crontab contained no unauthorized entries.

7. Cleanup of Malicious Artifacts

Leftover files were removed to ensure full remediation:

```
ubuntu@ubuntu:~/script-lab$ crontab -l
ubuntu@ubuntu:~/script-lab$ rm /tmp/fake_malicious.sh /tmp/f
ake_cron.log
ubuntu@ubuntu:~/script-lab$
```

This step prevents future use of the attacker's script, even if another persistence mechanism is later discovered.

8. Final System State

After removal of the cron job and cleanup of the associated files, the system was restored to a known safe state. No remaining evidence of persistence was found, and no further malicious execution occurred.

The incident response process successfully completed the full cycle:

- Detection
- Investigation
- Verification
- Remediation
- Cleanup
- Validation