

Incident Response Report – Elastic SIEM-Based Architecture

Student: Saad Laksabi

Operating Systems Used: Parrot OS (VirtualBox), macOS (Host)

Elastic Stack Version: 7.17.28

Tools Utilized: Filebeat, Wireshark, Volatility, Elastic Search Cloud, VirtualBox, Bash, Parrot OS Forensics Utilities

1. IR Environment Setup

Elastic Search Cloud Setup:

Elastic Search Cloud was configured to receive log data from the Parrot OS VM using Filebeat. The installation process included adding the Elastic APT repository, resolving missing GPG keys, and manually installing Filebeat version 7.17.28. Configuration included:

- Editing `/etc/filebeat/filebeat.yml` to include the Elastic Cloud ID and credentials.
- Enabling system module with `sudo filebeat modules enable system`.
- Testing configuration with `sudo filebeat test output`.
- Starting the Filebeat service using `sudo systemctl start filebeat` and enabling it to persist with `sudo systemctl enable filebeat`.
- Log ingestion was verified in Elastic Search Cloud via the Discover and Logs UI.

Volatility Setup:

Volatility3 was installed from source in Parrot OS using the official GitHub repository. It was tested using a mock memory dump captured via `dd`:

```
sudo dd if=/dev/mem of=/home/user/memdump.raw bs=1M count=1024
```

The following analysis modules were run:

- `vol.py -f memdump.raw windows.pslist`
- `vol.py -f memdump.raw linux.netstat`

Wireshark Setup:

Wireshark was configured with capture filters to isolate suspicious TCP/UDP traffic:

- `tcp.port == 22` (SSH brute-force simulation)
- `udp.port == 53` (DNS tunneling check)

System Logging:

Syslog was configured on Parrot OS using rsyslog to generate and forward logs to

`/var/log/syslog`. Log generation was simulated by SSH login attempts and a fake malware execution (`/tmp/fake_malware.sh`).

Host Logs (macOS):

Host logs were exported using:

```
log show --predicate 'eventMessage contains "sshd"' --last 1d > mac_logs.txt
```

These were ingested into Elastic via drag-and-drop into the Discover panel.

2. Digital Evidence Management

Live System Data:

From Parrot OS:

- `ps aux` and `top -b -n1` outputs captured running processes.
- `ss -tulpn` captured open network ports.
- `ifconfig` documented system IP info.

Memory Acquisition & Analysis:

- Captured memory using `dd`.
- Used Volatility3 to examine `pslist`, `malfind`, `netscan`, and `cmdline` artifacts.

Disk Acquisition:

Disk imaging was performed using:

```
sudo dd if=/dev/sda of=~/.disk_image.dd bs=4M
```

Hash verified using:

```
sha256sum disk_image.dd
```

Chain of Custody:

A chain of custody form was created for each piece of evidence with:

- Evidence ID (e.g., MEM01)
- Description
- SHA256 hash
- Collector and timestamps

Timeline Creation:

Combined macOS and Parrot OS log events into a CSV timeline. Used Elastic's Kibana timeline

feature to visualize SSH logins, suspicious process activity, and simulated malware execution over a 2-hour period.

3. Incident Detection and Analysis

Log Analysis in Elastic:

Used KQL (Kibana Query Language) for targeted searches:

- `process.name : "fake_malware.sh"`
- `source.ip : 192.168.56.1` and `event.dataset : system.auth`
- `event.action : "user_login"` and `event.outcome : "failure"`

Suspicious Login Investigation:

- Timeline built around repeated failed SSH login attempts from 192.168.56.1 to 10.0.2.15 (Parrot).
- Correlated failed logins in `/var/log/auth.log` and Elasticsearch log streams.
- Alert rule created for 5+ failed logins in 2 minutes.

Security Incident Classification:

- **Incident 1: SSH brute force** – Medium severity, persistent failures from same IP.
- **Incident 2: Suspicious script execution** – High severity, detection of `fake_malware.sh`.
- **Incident 3: DNS tunneling simulation** – Low severity, excessive DNS queries from `dig` script.

Severity matrix used based on CIA impact.

4. Incident Response and Containment

Network Isolation Procedures:

- Network interfaces disabled using `ifconfig eth0 down`.
- VirtualBox adapter set to internal network.
- Firewall rule added:

```
sudo iptables -A INPUT -s 192.168.56.1 -j DROP
```

Containment Playbook:

- Shutdown affected services: `sudo systemctl stop ssh`
- Isolate system from network.
- Preserve disk/memory.
- Export logs immediately.

Evidence Preservation:

- Network traffic captured with `tcpdump -w evidence.pcap`.
 - Memory preserved using `dd`.
 - Script artifacts moved to `/evidence/fake_malware.sh` and hashed.
-

5. Post-Incident Procedures





System Recovery:

- VirtualBox snapshot restored.
- Parrot OS reinstalled from ISO with secure baseline.
- MAC IP reset and DNS cache flushed.

Root Cause Analysis:

- Unpatched SSH config allowed password-based auth.
- User error simulated by downloading and executing malicious file.
- Lack of outbound filtering allowed DNS tunneling simulation.

Recovery Validation Checklist:

- Reimaged Parrot and reinstalled tools – 
- SSH config set to key-based login only – 
- Logs verified to show no further suspicious activity – 
- Elastic integration tested – 

IR Process Improvement:

- Deploy a network monitoring tool like Suricata.
 - Create scripts for evidence hashing automation.
 - Harden services post-deployment.
 - Rotate credentials monthly.
-

6. Final Reflections

This project solidified practical IR techniques, improved my Linux CLI comfort, and helped me understand centralized logging with Elastic. By shifting to Elastic Search Cloud from Wazuh and completing all detection and containment steps using mock data, the incident lifecycle was fully demonstrated. All logs, reports, memory dumps, PCAPs, and disk images are archived in the evidence folder structure [/IR_Project/Evidence/](#).

Total Incidents Simulated: 3

Logs Analyzed: 5,500+ entries

Memory Size: 1GB raw image

Elastic SIEM Rules Created: 3

Submitted by:

Saad Laksabi

Date: June 4, 2025

