AKSHATHA R

CYART INTERN

Task: 01 SOC ANALYST INTERN

1. SOC Fundamentals and Operations

Purpose of a SOC

A SOC is the **centralized unit** within an organization that manages and enhances the security posture. Its core purposes are:

- **Proactive Threat Detection:** Identifying suspicious activity or potential breaches **before** significant damage occurs.
- **Incident Response (IR):** Managing the entire lifecycle of a security incident, from initial containment and eradication to recovery and post-mortem analysis.[2]
- **Continuous Monitoring:** Maintaining 24/7 surveillance of the network, systems, and applications to ensure on-going security.

Key Roles

The SOC team operates as a hierarchy with specialized skills:

Role	Primary Responsibility	Focus
Tier 1 (L1) Analyst	**Triage and Initial Investigation.** Monitors incoming alerts, determines their validity (false positive vs. true positive), and escalates genuine incidents.	Speed, alert management, basic troubleshooting.
Tier 2 (L2) Analyst	**In-Depth Investigation and Response.** Handles escalated incidents from T1, performs forensics, determines root cause, and implements remediation steps.	Forensic analysis, advanced tools, scripting.
Tier 3 (L3) Analyst/Hunter	**Advanced Analysis, Malware Reversal, and Hunting.** Develops new detection methodologies, performs deep dives (threat hunting), and handles the most complex incidents.	Expertise, threat intelligence, reverse engineering.

SOC Manager	**Team Leadership and Strategy.** Oversees SOC operations, manages budgets, defines security policies, and reports to leadership.	Management, strategy, compliance.
Threat Hunter	**Proactive Search for Threats.** Operates under the assumption that a breach has already occurred, actively searching log data and systems for subtle, undetected malicious activity.	Creativity, hypothesis testing, advanced query languages.

Key Functions

These functions are the *how* the SOC achieves its purpose:

- **Log Analysis:** Collecting, parsing, and analysing logs from all relevant security devices (e.g., firewall, IDS/IPS), servers, and applications using a **SIEM** (Security Information and Event Management) platform.
- **Alert Triage:** The process of prioritizing, validating, and assigning a severity level to security alerts generated by monitoring systems. This is the **most crucial T1 function.**
- **Threat Intelligence Integration:** Incorporating external information about current and emerging threats (e.g., adversary TTPs, Indicators of Compromise - IOCs) into the SOC's tools and processes to improve detection accuracy and context.

Sample Security Event Report – Brute Force Attack

1. Event Summary

- **Event Type:** Brute-Force Login Attempt
- **Severity Level:** Medium → High (depending on account targeted)
- **Date/Time Detected:** 12 Nov 2025, 14:22 IST
- **Detected By:** Windows Event Viewer / SIEM
- **Affected System:** WIN-SERVER01 (Domain Controller)
- **Targeted Account:** `administrator`

2. Detection Details

The SOC detected a spike of failed login attempts through:

a. Event Viewer – Security Logs

Filtered for **Event ID 4625 – Failed Logon**

- 54 failed login attempts within **3 minutes**
- Same source IP: **192.168.10.45**
- Logon Type: **3 (Network Logon)**
- Failure Reason: *Unknown user name or bad password*

b. SIEM Alert Triggered

Rule Name: **Excessive Failed Logons – Possible Brute Force**
Threshold: 30 failed attempts within 5 minutes

3. Evidence Collected

Event ID 4625 Sample Log Snippet

```
Account For Which Logon Failed:
  Security ID: NULL SID
  Account Name: administrator

Network Information:
  Workstation Name: WS-PC32
  Source Network Address: 192.168.10.45
  Source Port: 51322

Failure Information:
  Status: 0xC000006D
  Sub Status: 0xC000006A (Bad password)
```

Additional Indicators

- No successful login attempts following failures
- Attempts show consistent 2–3 second intervals (automated tool behavior)
- No other systems targeted during this timeframe

4. Analysis

Key Findings

- The attack originated from **internal network IP (192.168.10.45)**

- High number of failed attempts → **Indicates automated brute-force tool**
- Attack targeted a **high-privilege admin account**, increasing severity
- No lateral movement detected after attempts
- No successful login → brute-force attempt was unsuccessful

Root Cause

Possible compromised workstation or insider testing:

- The source system **WS-PC32** was found logged in by a user with no administrative rights
- Unusual PowerShell activity detected (not malicious, but unapproved commands)

5. Containment Actions Taken

✔ Blocked source IP using firewall rule
✔ Disabled network connectivity for **WS-PC32**
✔ Forced password reset for "administrator" account
✔ Initiated malware scan on source system
✔ Enabled enhanced logging for privileged accounts
✔ Informed SOC Lead and Incident Response Team

6. Eradication & Recovery

- Malware scan on WS-PC32 completed → **No malware found**
- User questioned → Claimed testing passwords "accidentally"
- Device re-imaged as precaution
- Account lockout policies reviewed and updated

7. Recommendations (Lessons Learned)

Short-Term

- Enforce stricter **Account Lockout Policy**
  - Lockout threshold: 5 attempts
  - Lockout duration: 15 minutes
- Enable alerts for:
  - Privileged account login failures
  - High-frequency failed attempts

Long-Term

- Implement MFA for all admin accounts
- Segment internal network to limit lateral movement
- Provide user awareness training regarding unauthorized testing
- Add brute-force detection rule in SIEM with lower threshold

8. Final Status

Incident Closed – No successful compromise.
Root cause addressed and mitigations applied.

2. Security Monitoring Basics

Objectives of Security Monitoring

The core goal of security monitoring is to gain **visibility** into an organization's security posture and detect deviations from the norm.

- **Detect Anomalies:** Identifying activities that deviate significantly from established baselines of "normal" behavior (e.g., a user logging in from an unusual geographic location or downloading an unusually large amount of data).
- **Unauthorized Access:** Alerting on suspicious authentication attempts, successful or failed logins to critical systems, or the use of privileged accounts.
- **Policy Violations:** Detecting actions that violate organizational security policies or regulatory requirements (e.g., copying sensitive data to unapproved cloud storage or disabling antivirus software).

Key Monitoring Tools

Tool Category	Examples	Primary Function
SIEM (Security Information and Event	**Splunk**, **Microsoft Sentinel**, **Elastic**	**Centralized logging, correlation, and alerting.** Aggregates logs from all sources, normalizes the data, and uses

Management)	**Stack (ELK)**	correlation rules to detect multi-stage attacks.
Network Traffic Analyzers	**Wireshark**, **Tcpdump**, **Brim**	**Deep packet inspection.** Captures and analyzes network data packets to understand the flow of traffic, identify unusual protocols, and inspect malicious payloads.
IDS/IPS (Intrusion Detection/Prevention System)	**Snort**, **Suricata**	**Real-time traffic analysis.** Identifies known threats via signatures (**IDS**) and can actively block them (**IPS**).
Endpoint Detection and Response (EDR)	CrowdStrike, SentinelOne (Commercial)	Monitors and collects data from endpoint devices (laptops, servers) for suspicious processes and file activity.

These metrics are essential for measuring the efficiency and effectiveness of the SOC team's monitoring efforts:

- **False Positive Rate (FPR):** The percentage of alerts that are benign (not actual threats). A high FPR leads to **alert fatigue** and distracts analysts from real incidents.
- **False Negative Rate (FNR):** The percentage of actual threats that are **missed** by the detection systems. This is the most dangerous metric, as it indicates a silent breach.
- **Mean Time to Detect (MTTD):** The **average time** it takes from the start of a security event until the SOC team is aware of it. Reducing MTTD is a primary objective of security monitoring.
- **Mean Time to Respond (MTTR):** The average time required to fully **contain and remediate** a detected security incident.

How to Learn: Practical Steps

1. **Set up a Lab Environment:** Install the **Elastic Stack (Elasticsearch, Logstash, Kibana)**. It's free, open-source, and widely used. Use **Logstash** to ingest synthetic logs (e.g., system logs, web server logs).
2. **Analyze Sample Logs:** Once you have a log source, use **Kibana** (the visualization component) to practice writing queries. Look for simple suspicious patterns like:
    - Multiple failed login attempts followed by a success (Brute Force).
    - Unusual HTTP status codes (e.g., many 404s or 500s).
    - System command executions that shouldn't normally occur.
3. **Use Attack Scenarios:** Download **"Boss of the SOC (BOTS)" datasets** (often provided by Splunk) or similar public datasets. These provide pre-recorded, labeled log data from a simulated attack, allowing you to trace the attacker's steps using the SIEM.

Security Event Documentation Template

Field	Details
Date/Time	
Event Source	(Host/Endpoint/Server/Firewall/AD/DC/SIEM Alert Name)
Source IP	
Destination IP	*(If applicable)*
Event ID / Alert ID	
Severity	(Low / Medium / High / Critical)
Description	
Logs Observed	(Key log lines or summary)
Impact Analysis	(What it could affect: accounts, systems, data)
Action Taken	
Status	(Closed / Monitoring / Escalated)
Analyst Name	

✔ Sample Mock Event (Filled Example)

Scenario: Multiple failed logins from 192.168.1.10 (Possible brute-force attempt)

Field	Details
Date/Time	2025-11-13 18:40:12 IST
Event Source	Windows Server 2019 (Domain Controller)
Source IP	192.168.1.10
Destination IP	192.168.1.5 (DC)
Event ID / Alert ID	4625 — Failed Logon
Severity	Medium
Description	SIEM detected 25 failed login attempts within 2 minutes for user **admin_test** from IP 192.168.1.10. Behavior resembles brute-force attempt.
Logs Observed	Event ID 4625; Logon Type 3 (Network); Failure Reason: Unknown username or bad password.
Impact Analysis	Potential account compromise attempt. If successful, could lead to unauthorized access to internal network resources.
Action Taken	- Account locked by AD security policy

- Blocked source IP temporarily at firewall
- Notified system owner
- Enabled enhanced logging on the affected server |

  | **Status** | Monitoring |

  | **Analyst Name** | Akshatha R |

3. Log Management Fundamentals

Concept	Explanation
Log Lifecycle	The full journey of a log event: **Collection** (gathering from source), **Normalization** (converting to a common format), **Storage** (writing to the SIEM/storage layer), **Retention** (keeping logs for a defined period for forensics/compliance), and **Analysis** (querying and alerting).
Common Log Types	**Windows Event Logs** (Security, System, Application logs), **Syslog** (standard log format for Linux/Unix and network devices), and **HTTP Server Logs** (records of web requests, including status codes and user agents).
Normalization	The process of mapping diverse log fields (e.g., `src_ip`, `sourceAddress`, `client_ip`) into a **single, consistent scheme** (e.g., `source.ip`). Standard formats include **CEF** (Common Event Format) and the **Elastic Common Schema (ECS)**.

How to Perform: Practice Tasks

The following steps outline the processes for your practice tasks. Since the specific setup of your lab (e.g., Elastic SIEM version) can vary, these instructions focus on the **core commands and configurations**.

Practice Task 1: Log Collection Pipeline (Fluentd & Syslog)

This task assumes you have an **Ubuntu** virtual machine (VM) and a destination Elastic SIEM (or local file) to send logs to.

1. **Install Fluentd (td-agent) on Ubuntu:**

   Bash

```
# 1. Install prerequisites

sudo apt-get update

sudo apt-get install -y lsb-release gnupg apt-transport-
https



# 2. Add the Fluentd repository

curl -fsSL https://toolbelt.treasuredata.com/add-agent-
key.asc | sudo gpg --dearmor -o /usr/share/keyrings/td-
agent-keyring.gpg

echo "deb [arch=amd64 signed-by=/usr/share/keyrings/td-
agent-keyring.gpg]
https://packages.treasuredata.com/5/ubuntu/$(lsb_release
-cs)/ $(lsb_release -cs) contrib" | sudo tee
/etc/apt/sources.list.d/td-agent.list



# 3. Install td-agent (Fluentd)

sudo apt-get update

sudo apt-get install -y td-agent



# 4. Start the service

sudo systemctl start td-agent
```

2. Configure Fluentd to Collect Syslog:

   Edit the main configuration file, /etc/td-agent/td-agent.conf, and add an input and output section.

- **Input (Syslog):** Fluentd listens on port 5140 (default).
- **Output (Standard Output/File):** To verify logs easily.

Code snippet

```
# Example /etc/td-agent/td-agent.conf additions:



# 1. Input: Listen for Syslog messages on UDP port 5140

<source>

  @type syslog

  port 5140

  bind 0.0.0.0

  tag syslog

</source>



# 2. Output: Print matching logs to standard output for
verification

<match syslog.**>

  @type stdout

</match>
```

Restart Fluentd to load the new config:

Bash

```
sudo systemctl restart td-agent
```

3. Test Log Generation and Collection:

   Use the logger command to send a test message to the Syslog collector (Fluentd).

   Bash

   ```
   # Send a message to the local Syslog daemon, which should then forward it

   logger "Test message for Fluentd pipeline verification: Log Management Fundamentals"
   ```

   - **Verification:** Check the Fluentd logs or the SIEM interface. If using the `stdout` output, you should see the message in the Fluentd logs:

     Bash

     ```
     sudo tail -f /var/log/td-agent/td-agent.
     ```

While you specified **Elastic SIEM**, your query syntax is closer to **Kusto Query Language (KQL)** used by Microsoft Sentinel/Defender. For practice, we'll use the conceptual structure of the query as it applies to log analysis:

- **Goal:** Find how many failed login attempts occurred and from which IP address, in a security log.

KQL Syntax (Microsoft Sentinel)	**Elastic Search Query Language (EQL) (Conceptual)**	**Explanation**
`SecurityEvent`	`event.module: "windows" and winlog.event_id: 4625`	Filter the log table (`SecurityEvent` or `event.module: "windows"`) for the failed logon Event ID (**4625**).
`	where EventID == 4625`	**(Integrated into the filter above)**

`	summarize count() by IpAddress`	`aggregate by source.ip`

Setting Up Dashboards in Kibana (Elastic Stack)

Step 1: Collect & Index Logs

Make sure your logs are coming into Elasticsearch through:

- Winlogbeat / Filebeat (Windows logs)
- Auditbeat (Linux logs)
- Packetbeat (network logs)
- Elastic Agent (EDR + Sysmon)

Confirm indexes like:

- `winlogbeat-*`
- `sysmon-*`
- `logs-*`

Step 2: Create Visualization – Top 10 Source IPs

1. Go to **Kibana → Visualize Library**.
2. Click **Create Visualization → Lens**.
3. Drag **Terms** onto the workspace.
4. Set the field to **source.ip**.
5. Set **Size = 10**.
6. For values, use **Count** (number of events).
7. Optional: Filter for security events only

```
8. event.category: "authentication" OR event.code: 4625
```

Step 3: Visualization – Frequency of Critical Event IDs

1. **Create new Lens visualization**.
2. Drag **Date Histogram** to X-axis (Time).
3. Drag **Count** to Y-axis.
4. Add a filter:

```
5. winlog.event_id: (4625 OR 4768 OR 4740 OR 7045)
6. AND event.severity: "critical"
```

7. Display as **Line chart** or **Bar chart**.

Step 4: Build a Dashboard

1. Go to **Kibana → Dashboard**.
2. Click **Create Dashboard**.
3. Add your visualizations:
    - Top 10 Source IPs
    - Critical Event ID Frequency
    - Authentication Failures
    - Sysmon Process Creation spikes
    - Network connections by port
4. Save as **Security Monitoring Dashboard**.

2. Setting Up Dashboards in Grafana

Prerequisites:

- Data source configured (Elastic, Loki, Prometheus, or Windows exporter)
- Logs available with fields:
    - `source_ip`
    - `event_id`
    - `severity`

Step 1: Create Panel – Top 10 Source IPs

1. **Dashboard → New panel**
2. Query example (Loki):
3. `{source="winlog"} | json | stats count() by source_ip | sort desc | limit 10`
4. Choose **Bar Chart** or **Pie Chart**.

Step 2: Panel – Critical Event ID Frequency

Example query (Elastic datasource):

```
event.severity: "critical" AND event.code:*
```

Set:

- **X-axis:** `@timestamp` (time series)
- **Y-axis:** Count of logs
- **Break down by:** `event.code` (optional)

Step 3: Add Panels to Dashboard

1. Click **Add to dashboard**
2. Arrange panels visually:
    - Top attacking IPs (left)
    - Event trends over time (center)
    - Critical Event IDs (right)
    - Geo-Map of IPs (optional)
    - Sysmon Alerts (optional)

Use Pre-Built Dashboards

Kibana Pre-Built Dashboards

- Elastic Agent security dashboards
- Winlogbeat dashboards
- Sysmon dashboards
- Network dashboards
- Sigma rule dashboards (via Elastic SIEM)

Grafana Pre-Built Dashboards

- Community dashboards for:
    - Windows Event Logs
    - Suricata IDS
    - Zeek
    - Wazuh SIEM

4. Security Tools Overview and Practical Tasks

Key Tools Summary

- **SIEM (Security Information and Event Management):** Central platform for **collecting, correlating, and analyzing** security data from disparate sources (e.g., Splunk, QRadar).
- **EDR (Endpoint Detection and Response):** Tool deployed on endpoints (laptops, servers) that provides continuous **monitoring and active response** capabilities (e.g., CrowdStrike).

- **IDS/IPS (Intrusion Detection/Prevention System):** Monitors network traffic for suspicious activity (Detection) and can actively **block** it (Prevention) based on signatures or anomalies (e.g., Snort).
- **Vulnerability Scanners:** Tools that **identify security weaknesses** in systems and applications (e.g., Nessus, OpenVAS).

Practical Task 1: Snort Rule Testing

The process involves installing Snort, configuring it to read your rule, and generating traffic to trigger the alert.

1. **Install Snort on Ubuntu:** Snort installation is complex as it often requires compiling dependencies like DAQ. The simplest method for testing is often to use the package manager for Snort 2, or follow a detailed guide for Snort 3 compilation.
    - **Core Installation Command (Snort 2):**

      Bash

      ```
      sudo apt update

      sudo apt install snort
      ```

    - **Configuration:** You must define your `HOME_NET` variable in the `/etc/snort/snort.conf` file to be your lab network IP range.
    - **Rule File:** Your custom rule will go into the **`/etc/snort/rules/local.rules`** file.
2. **Write and Save the Snort Rule:** Use a text editor to add the following rule to `/etc/snort/rules/local.rules`:

   Code snippet

   ```
   alert tcp any any -> any 80 (msg:"Malicious Domain
   Detected: malicious.com"; content:"malicious.com";
   http_uri; sid:1000001; rev:1;)
   ```

    - **Explanation:**

- `alert tcp any any -> any 80`: Alert on any TCP traffic from any source (`any any`) to any destination IP on port 80 (standard HTTP).
- `msg:"..."`: The message displayed when the rule is triggered.
- `content:"malicious.com";`: Looks for the literal string "malicious.com" in the packet payload.
- `http_uri;`: Specifies that the content search should only look in the HTTP URI field (the path/domain requested), which makes it more specific.
- `sid:1000001;`: Unique Signature ID (must be in the user-defined range, e.g., 1000000+).
- `rev:1;`: Revision number.

3. **Test with `curl`:** Run Snort in packet logging or console alert mode (using your network interface, e.g., `eth0`), and then generate the alert using `curl` from a test machine pointing to the Snort machine's IP, or on the Snort machine itself:

   Bash

   ```
   # Run Snort in console mode

   sudo snort -A console -q -c /etc/snort/snort.conf -i eth0



   # From a separate terminal/VM, test the rule (assuming a
   web server is running

   # and "malicious.com" resolves or is configured in your
   hosts file):

   curl http://<Snort_VM_IP>/malicious.com
   ```

   Verification: Snort's console output should immediately display the alert message: `Malicious Domain Detected: malicious.com`.

Practical Task 2: Nessus Scan

1. **Run Nessus Essentials against Metasploitable2:**
    - **Setup:** Download and install Nessus Essentials (free for 16 IPs). Install **Metasploitable2** (an intentionally vulnerable VM) in your virtual environment (e.g., VirtualBox).
    - **Network:** Ensure the Nessus host (your Kali/other VM) and Metasploitable2 are on the **same isolated host-only network**.
    - **Scan:** In the Nessus web interface, create a new **Basic Network Scan**. Enter the IP address of the Metasploitable2 VM as the target.
    - **Recommended:** Use **Authenticated Credentials** (`msfadmin:msfadmin`) in the scan policy for a more thorough, local check (not just network-based checks).
2. **Export the Report and List Top 3 Vulnerabilities:**
    - Once the scan completes, export the results as a **CSV** or **Nessus file**.
    - **Vulnerability Results (Expected Top 3):** Metasploitable2 is severely outdated. The top vulnerabilities will almost always include services like **UnrealIRCd Daemon Backdoor**, **vsftpd 2.3.4 Backdoor**, and an insecure **Samba configuration**.
    - **CVSS Score:** These severe vulnerabilities often result in high base CVSS scores, likely **10.0 (Critical)** for remote code execution.

.**Practical Task 3: Osquery Monitoring**

1. **Install Osquery on Windows VM:** Download the **MSI installer** from the Osquery downloads page and run it on your Windows VM. It installs the interactive shell (`osqueryi.exe`) and the daemon service (`osqueryd.exe`).
2. **Query Running Processes (Interactive Shell):** Launch the interactive shell (you may need to navigate to `C:\Program Files\osquery`):

    PowerShell

    ```
    & 'C:\Program Files\osquery\osqueryi.exe'

    osquery> SELECT pid, name, path, cmdline FROM processes;
    ```

- **Analysis:** This query returns Process ID, Name, Executable Path, and the Command Line arguments for all running processes. This is a foundational step in host monitoring.

3. **Simulate and Query a Malicious Process:**
   - **Simulation:** Create a harmless batch file named **`evil.bat`** (or use a simple executable like `calc.exe`).

     Code snippet

     ```
     # C:\Users\YourUser\Desktop\evil.bat

     @echo off

     echo Starting malicious simulation...

     ping 127.0.0.1 -n 5 > nul
     ```

   - **Execution:** Double-click or run the batch file to start the process.
   - **Query:** Quickly run the query again, or refine it to search specifically for the process name

ELASTIC SIEM — Create Rule: "5+ Failed Logins in 5 Minutes"

➡ Step 1: Go to Elastic Security → Detections → Create Rule

- Rule Type: **Custom Query**

➡ Step 2: Set Rule Query

Use a query that detects failed login attempts:

event.dataset: "system.auth" AND event.action: "user_login" AND event.outcome: "failure"

Or for Windows logs via Winlogbeat:

winlog.event_id: 4625

➡ Step 3: Select Index Pattern

Enter:

```
security-login-*
```

or whatever your SSH/Winlogbeat login index is.

➡ Step 4: Set Threshold Condition

- Rule type: **Threshold**
- Field: `source.ip` (optional)
- Threshold:
  - **Count ≥ 5**
  - **Time window: 5 minutes**

This detects brute-force attempts.

➡ Step 5: Configure Rule Name & Actions

- Rule name: **5+ Failed Login Attempts (5 min)**
- Severity: **Medium**
- Risk score: 50
- Add actions (email, Slack, webhook) if desired.

➡ Step 6: Save & Enable Rule

➡ Step 7: Test the Rule

From attacker machine or Kali Linux:

```
for i in {1..7}; do ssh testuser@192.168.1.10; done
```

Enter wrong password repeatedly.

➡ Step 8: Validate Alert

Go to:

Elastic Security → Alerts → Status: Active

We should see an alert with:

- Source IP
- Username
- Count of failed logins

2. WAZUH — Custom Rule: "3+ Failed SSH Logins in 2 Minutes"

Wazuh stores events in `/var/ossec/logs/alerts/alerts.json` and shows them in the Wazuh dashboard.

Step 1: Enable SSH Auth Log Collection

Make sure Wazuh agent is collecting auth logs:

File:
`/var/ossec/etc/ossec.conf`

Ensure this block exists:

```
<localfile>
  <log_format>syslog</log_format>
  <location>/var/log/auth.log</location>
</localfile>
```

Restart Wazuh agent:

```
systemctl restart wazuh-agent
```

Step 2: Create a Custom Rule

File:
`/var/ossec/etc/rules/local_rules.xml`

Add the following:

```
<group name="authentication, sshd,">
  <rule id="100200" level="3">
    <match>Failed password</match>
    <description>SSH failed login attempt</description>
  </rule>

  <rule id="100201" level="7" frequency="3" timeframe="120" ignore="15">
    <if_matched_sid>100200</if_matched_sid>
```

```
    <description>3+ SSH failed logins in 2 minutes (Possible brute-force
attack)</description>
  </rule>
</group>
```

Meaning:

- Rule 100200 = individual failed login
- Rule 100201 = fires when **≥ 3** events occur within **120 seconds**

Restart Wazuh manager:

```
systemctl restart wazuh-manager
```

Step 3: Simulate Failed SSH Logins

From attacker machine:

```
for i in {1..5}; do ssh wronguser@192.168.1.x; done
```

Enter wrong password each time.

This generates `Failed password` logs in `/var/log/auth.log`.

Step 4: Validate Alert in Wazuh Dashboard

Go to:

Wazuh → Security Events → Rule ID 100201

Look for:

- Count of failed attempts
- Source IP
- Time window information
- Usernames involved

Step 5: Document the Rule's Effectiveness

Example documentation:

Alert Rule Validation Report

Rule Tested:
3+ SSH failed logins in 2 minutes
Rule ID: 100201
Platform: Wazuh Manager
Test Performed:
Five failed SSH login attempts from IP `192.168.1.55` within 1 minute.
Result:
Alert triggered successfully.
Alert Details:

- Matched Failed Password events: 5
- Threshold requirement: ≥ 3 events
- Source IP: 192.168.1.55
  Status: Rule functioning as expected.

5. Basic Security Concepts

1. CIA Triad

The **CIA Triad** is the foundational model for evaluating and implementing security policies. Every security control aims to protect one or more of these three components:

- **C - Confidentiality:** Ensuring that information is **accessible only to authorized parties**. This is achieved through encryption, access controls (like usernames/passwords), and need-to-know restrictions.
  - *Example:* Encrypting a hard drive so only the user with the key can read the data.
- **I - Integrity:** Ensuring that data is **accurate, complete, and untampered** throughout its lifecycle. This is achieved through hashing, digital signatures, and strict version control.
  - *Example:* Using a checksum to verify a downloaded file hasn't been altered during transmission.[5]
- **A - Availability:** Ensuring that **systems and data are accessible and usable** by authorized users when required.[6] This is achieved through redundancy, failover clusters, and disaster recovery plans.[7]

- *Example:* Using redundant servers (load balancing) to prevent service disruption if one server fails.[8]

2. Threat vs. Vulnerability vs. Risk

These three terms are often confused, but they have distinct meanings in risk management:

Concept	Definition	Example
Threat	A **potential danger** that could exploit a vulnerability. This is the **"who"** or **"what"** (e.g., an attacker, a malware strain, a flood).	A **hacker** attempting a SQL injection attack.
Vulnerability	A **weakness** in a system, design, or implementation that can be exploited by a threat. This is the **"flaw."**	An **outdated web server** lacking a security patch.
Risk	The **potential for loss or damage** resulting from a threat exploiting a vulnerability. This is calculated as **(Threat * Vulnerability) - Countermeasure**.	The **financial and reputational damage** if the hacker exploits the unpatched server.

---

3. Defense-in-Depth and Zero Trust

These are two critical architectural philosophies for designing a secure environment:

- **Defense-in-Depth (DiD):** This strategy involves layering multiple, independent security controls to protect resources.[9] If one layer fails (e.g., the firewall is breached), the attacker encounters the next layer (e.g., endpoint security, user access controls).[10] It assumes a single security measure will eventually fail.

- **Zero Trust (ZT):** A security model based on the principle: **"Never trust, always verify."** It assumes no user, device, or application, inside or outside the network perimeter, shoculd be trusted by default.[11] Every access request is strongly authenticated, authorized, and continuously validated before access is granted.[12]
    - *Key shift:* Moves security focus from the **network perimeter** to the **user, resource, and session**.

How to Learn

- **Flashcards (Anki):** This is the perfect approach for memorizing the precise definitions and relationships between these concepts. Use **Mnemonics** (e.g., CIA = Confidentiality, Integrity, Availability) to help recall.[13]
- **Case Studies:** Analyzing real-world incidents, like the **Equifax breach**, directly illustrates these concepts:
    - **Vulnerability:** Failure to patch a known flaw in the Apache Struts framework.[14]
    - **Threat:** A remote attacker exploiting that flaw.
    - **Risk:** Massive data loss and financial damage.
    - **CIA failure:** Confidentiality was lost (data stolen), and Integrity was potentially compromised.

6. Security Operations Workflow – Stages and Steps

1. Detection

- **Objective:** Identify potential security incidents through automated tools and monitoring.
- **Sources:**
    - SIEM alerts (e.g., from Splunk, QRadar, Azure Sentinel)
    - EDR/XDR alerts (e.g., CrowdStrike, SentinelOne)
    - IDS/IPS logs
    - Firewall and proxy logs
    - User reports (like suspicious emails or attachments)
- **Example:** SIEM generates an alert — “Suspicious login attempt from unknown IP.”

2. Triage

- **Objective:** Prioritize alerts based on impact and urgency.
- **Actions:**
    - Validate whether it's a true positive, false positive, or benign event.
    - Categorize the severity: *High*, *Medium*, *Low*.
    - Assign incident to a Tier 1 or Tier 2 analyst for further investigation.
- **Example:** Confirm the phishing email was received by multiple users — severity escalated to *High*.

3. Investigation

- **Objective:** Analyze the scope and root cause of the incident.
- **Actions:**
    - Correlate logs across systems (email gateway, endpoints, Active Directory, etc.).
    - Perform **IOC (Indicators of Compromise)** hunting: domains, URLs, hashes.
    - Check threat intelligence feeds for reputation of IPs/domains.
    - Identify affected systems or users.
- **Example:** Investigate sender domain reputation → found on threat feed as malicious.

4. Response

- **Objective:** Contain and remediate the threat, and prevent recurrence.
- **Actions:**
    - **Containment:** Block sender domain/IP, isolate affected endpoints.
    - **Eradication:** Remove malicious files, reset credentials.
    - **Recovery:** Restore from backups, monitor for reoccurrence.
    - **Post-Incident:** Document findings, lessons learned, update playbooks.
- **Example:** Block domain via email gateway, reset user passwords, and submit sample to sandbox.

How to Perform Practically

You can simulate the SOC workflow using these tools:

Tool	Purpose
TheHive	Incident response management and case tracking.
Cortex	Automate response actions (e.g., block IP, query IOC).
MISP	Threat intelligence sharing and IOC management.
Wazuh/Splunk	SIEM simulation for log collection and alert generation.

7. Incident Response Basics

IR Lifecycle (6 Core Stages)

1. **Preparation**
    - Create IR policies, communication plans, runbooks, and access lists.
    - Ensure tools like SIEM, EDR, backups, and monitoring are configured.
2. **Identification**
    - Detect and confirm the incident using alerts, logs, and threat intelligence.
    - Verify: *Is this a real incident or a false positive?*
    - Classify severity and impacted assets.
3. **Containment**
    - Short-term: Isolate affected systems (e.g., network segmentation, block malicious IPs).
    - Long-term: Apply patches, limit privileges, prevent lateral movement.
4. **Eradication**
    - Remove malware, delete malicious accounts, fix vulnerabilities.
    - Perform forensic analysis to ensure root cause is eliminated.

5. **Recovery**
    - Restore systems from clean backups.
    - Monitor closely for re-infection or repeated attacks.
    - Validate that systems are back to normal.
6. **Lessons Learned**
    - Document what happened, how it was handled, what worked, what didn't.
    - Update SOPs/runbooks and improve detection rules.

How to Learn Incident Response Effectively

1. Study the NIST SP 800-61 Framework

This is the most widely used IR guide. Focus on:

- IR phases
- Documentation standards
- Communication workflows
- Technical vs managerial response actions

2. Perform Tabletop Exercises (Highly Important for SOC interviews)

Simulated scenarios help build real-world IR thinking. Examples:

- **Ransomware attack** (files encrypted, ransom note appears)
- **Phishing compromise** (stolen credentials, lateral movement)
- **DDoS attack** (server overload, traffic spikes)
- **Data exfiltration** (large outbound traffic to unknown IP)

Ransomware Attack Simulation (Tabletop Exercise)

Scenario Name: *ShadowCrypt Ransomware Incident*

Environment: Mid-sized company, 300 employees, hybrid network (cloud + on-prem)

Stage 1 — Initial Detection (Identification Phase)

Alert from SIEM (10:15 AM):

- Multiple endpoints reporting:
  - **“Suspicious file encryption activity”**
  - **“High volume of file modifications in short time”**
  - **“Process: powershell.exe invoking unknown script”**
- EDR flags:
  - `shadowcrypt.exe` quarantined on 2 machines
- User ticket:
  - “My files suddenly have .shc extension and a ransom note popped up.”

Files Found:

- Extensions: **.shc**
- Ransom note: **README-RECOVER-FILES.txt**

Stage 2 — Investigation Data

Evidence Collected

1. Event Logs:

```
Cmd.exe /c powershell -enc SQBFAFcA...
Script attempts downloading payload from: hxxp://198.51.100.66/update.ps1
```

2. Firewall Logs:
Outbound traffic to unknown IPs at:

- `198.51.100.66`
- `203.0.113.45`

3. Lateral Movement Patterns:

- Mimikatz-like behavior detected
- Failed login attempts on file server
- Suspicious admin privilege escalation

4. Infection Spread:

- 12 endpoints encrypted
- 1 file server partially encrypted

Stage 3 — Containment Actions

Short-Term Containment

You must decide:

✔ Disconnect affected PCs from network
✔ Block malicious IPs
✔ Disable SMBv1
✔ Disable shared drives temporarily
✔ Reset credentials for high-privilege users

Stage 4 — Eradication Plan

Actions Required

- Use EDR to terminate ransomware process
- Delete scheduled tasks created by attacker
- Remove persistence mechanisms:
  - Run key in registry
  - Startup folder suspicious files
- Patch exploited vulnerability (SMB, VPN, or phishing entry point)
- Perform full malware scan

Root Cause Identified:

Employee opened a phishing email with attachment:
invoice_details.pdf.exe

Stage 5 — Recovery Steps

Restore Systems

- Restore encrypted systems from **last known clean backup**
- Rebuild 3 endpoints fully
- Validate:
  - No malicious traffic
  - No unusual PowerShell activity
  - Clean baseline images reloaded

Business Recovery

- File server restored
- 80% of files recovered
- Remaining lost files were not backed up

Stage 6 — Ransom Note

```
Your files are encrypted by ShadowCrypt.
```

```
Pay 2 BTC within 72 hours or lose all your data.
Contact: shadowhelp@protonmail.com
Wallet: 1FfmbHfnpaZjKFvyi1okTjJJusN455paPH
```

Stage 7 — Lessons Learned

- Improve email filtering → phishing was the entry point
- Mandatory user awareness training
- Enable strict PowerShell logging
- Deploy EDR to all endpoints
- Improve backup frequency
- Implement network segmentation
- Add ransomware detection rules to SIEM

8. Documentation Standards

Good documentation ensures consistency, clarity, and repeatability in SOC and Incident Response operations. It helps teams maintain accurate records and follow standard processes.

Examples of Documentation

1. Incident Reports

- Detailed explanation of a security incident
- Includes timeline, root cause, impact, actions taken, and lessons learned

2. Runbooks

- Step-by-step technical procedures for SOC analysts
- Example: “How to investigate a phishing alert” or “How to handle brute-force alerts”

3. SOPs (Standard Operating Procedures)

- Organization-wide instructions for routine or critical tasks
- Ensures all analysts follow the same steps

4. Post-Mortems / Lessons Learned

- After-action review after an incident
- Highlights what worked, what failed, and improvements needed

How to Perform / Improve Documentation Skills

1. Use Standard Templates

Using templates ensures your documentation is complete, clear, and consistent.

Recommended sources:

- **SANS Incident Handler's Handbook**
- **NIST SP 800-61 incident report structure**
- **Company-provided runbook/SOP templates**

Templates usually include:

- Scope
- Summary
- Timeline
- Technical details
- Root cause
- Recommendations
- Approvals

2. Practice Documenting Realistic Scenarios

A great learning method is creating mock reports.

Example Practice Scenario: DDoS Attack

Create a mock incident report using this scenario:

Scenario:

- At 14:30, monitoring tools detect a huge spike in inbound traffic.
- Website becomes unavailable for 25 minutes.
- Firewall logs show a large SYN flood from thousands of IPs.
- CDN provider confirms abnormal traffic patterns.

What to include in the report:

- **Executive Summary** (simple explanation for management)
- **Detection** (how the alert was triggered)
- **Analysis** (traffic logs, IOC identification, source IP patterns)
- **Containment** (rate limiting, firewall blocks, ISP notification)
- **Eradication/Recovery** (server stabilization, traffic normalization)
- **Lessons Learned** (improve rate limit rules, add anti-DDoS protection)