AKSHATHA R

CYART INTERNSHIP

SOC ANALYST INTERN

Threat Hunting Methodologies: Core Concepts

Threat Hunting is a **proactive** security discipline that involves systematically and iteratively searching through networks, endpoints, and logs to detect and isolate advanced threats that evade existing security controls.

1. Proactive Hunting vs. Reactive Response

The core distinction of threat hunting lies in its initiation point:

- **Proactive Threat Hunting (Hypothesis-Driven):**
    - **Starts With:** A **hypothesis** or an assumption (e.g., "Attackers might be using PowerShell to encode commands and evade detection").
    - **Goal:** To find **unknown** or **undetected** malicious activity that has bypassed current defenses.
    - **Example:** Searching all authentication logs for **T1078 (Valid Accounts)** misuse, specifically looking for logins from accounts outside of typical business hours or geographic locations.
- **Reactive Incident Response (IR):**
    - **Starts With:** A **known alert** (e.g., a malware signature match from an EDR) or identified incident.
    - **Goal:** To contain, eradicate, and recover from a **known** breach.

2. Hunting Frameworks

These frameworks provide structured processes for conducting hunts:

- **SqRR (Search, Query, Retrieve, Respond):**
    - A simple, cyclical model focusing on the data workflow:
        1. **Search:** Define the hunt and form the hypothesis.
        2. **Query:** Translate the hypothesis into a specific search query (e.g., KQL, SPL, VQL) against log data.

3. **Retrieve:** Extract and analyze the relevant data that matches the query.
4. **Respond:** Take action on the findings (e.g., create an alert, update a detection rule, initiate IR).

- **TaHiTI (Targeted Hunting integrating Threat Intelligence):**
  - This framework emphasizes using **Threat Intelligence (TI)** to focus the hunting effort, making it highly efficient.
  - The TI (e.g., an APT group's known TTPs or IOCs) is used to prioritize which systems or data sources are likely compromised, rather than blindly searching everything.
  - TI informs the hypothesis, making the search much more **targeted**.

3. Data Sources for Effective Hunting

Effective threat hunting requires access to high-fidelity, comprehensive data across the entire IT environment. Key sources include:

- **Endpoint Logs (EDR/Agent Logs):** Process creation, file modifications, service installations, DLL loading. Crucial for detecting $TTPs$ on individual machines.
- **Network Traffic Logs (NetFlow/PCAPs):** Connections, volume, protocol usage, and DNS requests. Essential for identifying Command and Control (C2) communications.
- **Authentication Logs (Active Directory/Domain Controller):** Failed and successful login attempts (Event ID 4624/4625), privilege assignment (Event ID 4672). Critical for $T1078$ and lateral movement hunts.
- **Threat Intelligence Feeds:** IOCs (IPs, domains, hashes) and TTPs (MITRE ATT&CK) used to inform and validate hunting hypotheses.
- **Proxy/Web Logs:** User-agent strings, destination URLs, and data transfer sizes. Useful for detecting suspicious outbound connections.

SOAR (Security Orchestration, Automation, and Response) is a technology that helps Security Operations Centers (SOCs) manage security incidents and operations by standardizing and automating the workflow. The key objective is to **increase efficiency** and **reduce Mean Time to Respond (MTTR)** by handling repetitive, low-level tasks automatically.

1. SOAR Components

SOAR is defined by the interaction of its three main components:

- **Orchestration (Workflow Integration):**
    - The ability to connect and coordinate multiple disparate security tools (e.g., firewall, EDR, threat intelligence platform) to work together in a standardized process.
    - **Goal:** Create seamless, end-to-end workflows across different technologies.
- **Automation (Task Execution):**
    - The capability to execute tasks or sequences of actions without human intervention.
    - **Examples:** Automatically **auto-ticketing** (creating an incident in a system like Jira or TheHive), enriching an alert with geolocation data, or checking a file hash against VirusTotal.
- **Response (Actionable Outcome):**
    - The final, automated action taken against a threat based on analysis.
    - **Examples: Auto-containment** (isolating a host via EDR), blocking a malicious domain on a firewall, or deleting a phishing email from all user inboxes.

2. Playbook Development

A **Playbook** is the structured, automated workflow that defines the steps a SOAR platform takes when a specific type of incident occurs.

- **Design Principles:** Playbooks should be designed to handle the 80% of common, repeatable incidents, freeing analysts for complex threats.
- **Example Scenario (C2 Traffic):**
    1. **Trigger:** SIEM/Wazuh alert on outbound connection to a suspicious IP (potential C2).
    2. **Enrichment:** Automatically query threat intelligence feeds (e.g., AlienVault OTX) for the IP's reputation.

3. **Decision:** If the IP is confirmed malicious.
4. **Action: Automate IP blocking** by sending a command to the firewall (e.g., CrowdSec) and initiating host containment via the EDR.

3. Integration with SIEM/EDR

SOAR's effectiveness depends entirely on its ability to integrate with the tools that generate alerts and those that enforce security policies:

- **Integration with SIEM (Security Information and Event Management, e.g., Elastic, Splunk, Wazuh):**
  - **Input:** The SIEM feeds **validated alerts** into the SOAR platform, initiating the playbook execution.
  - **Output:** SOAR can update the SIEM case with enrichment data or close the ticket upon successful resolution.
- **Integration with EDR (Endpoint Detection and Response):**
  - **Actionable Response:** EDR tools allow the SOAR platform to perform critical **response actions** directly on the endpoint, such as:
    - Host isolation.
    - Retrieving forensic artifacts (logs, files).
    - Terminating malicious processes

Root Cause Analysis (RCA) □

RCA is a structured investigation focused on finding the **fundamental, underlying cause** of a security incident, rather than simply fixing the immediate symptom. Identifying this root cause is the key to preventing similar incidents in the future.

5 Whys Technique

The 5 Whys is an iterative technique that explores cause-and-effect relationships by repeatedly asking "Why?". This process helps drill down from the surface symptom to the **systemic failure** (the ultimate root cause).

Example Progression:

- **Symptom:** Server was compromised.
- **Why 1:** Because it ran unpatched software.

- **Why 2:** Because the patching schedule was missed.
- **Why 3:** Because the patch management team is understaffed.
- **Root Cause: Staffing/Process** Failure in resource allocation.

Fishbone Diagram (Ishikawa Diagram)

The Fishbone Diagram is a visualization tool that helps organize potential causes into major categories before drilling down to the root cause. This ensures a comprehensive, multi-angle review.

- **Categories (The "Bones"):** Common categories used in security RCA:
  - **People:** Lack of training, failure to follow procedures.
  - **Process:** Missing procedures, poor change management, no patching schedule.
  - **Technology:** Outdated software, misconfigured firewalls/security tools, lack of logging.
  - **Environment:** External threats, regulatory constraints, unexpected dependencies.

2. Lessons Learned Process (Post-Mortem)

The **Lessons Learned** or **Post-Mortem** is a formal, **no-blame review** conducted after an incident has been contained and eradicated. Its purpose is to objectively assess the effectiveness of the response and identify actionable improvements.

Objectives

- **Review the Incident Timeline:** Identify any delays, points of confusion, or missed opportunities for early containment.
- **Evaluate Tool Effectiveness:** Assess if security **tools** (EDR, SIEM, Firewall) functioned as expected or if configuration/licensing gaps exist.
- **Assess Process Effectiveness:** Determine if procedures (e.g., escalation matrix, communication plan) were clear and sufficient.
- **Identify Training Gaps:** Pinpoint where personnel lacked necessary skills or experience to handle the situation efficiently.

Outcome

The process culminates in a formal report detailing the findings, confirmed root cause, and a list of **prioritized action items** to improve the organization's defensive posture and response procedures.

3. Metrics and KPIs for Performance Measurement □

Key Performance Indicators (KPIs) are vital for quantifying the SOC's performance, measuring the success of implemented improvements, and communicating risk to executive leadership.

Metric	Calculation / Definition	Objective
Mean Time to Detect (MTTD)	Average time from incident *start* to its *detection* by security controls or analysts.	**Lower** is better. Indicates effective monitoring and alerting.
Mean Time to Respond (MTTR)	Average time from incident *detection* to full *containment, eradication, and recovery*.	**Lower** is better. Indicates efficient and automated response procedures.
Dwell Time	The total time an attacker is present in the environment (from compromise to detection).	**Lower** is the most critical objective.
False Positive Rate	The ratio of incorrect alerts to total alerts.	Should be **low**. A high rate leads to analyst fatigue and missed real threats.

Adversary Emulation Techniques: Core Concepts

Adversary Emulation is a structured approach to security testing that involves simulating the **Tactics, Technhiques, and Procedures (TTPs)** of known threat groups or malware families to validate an organization's detection and response capabilities.

1. Adversary Emulation (Simulation)

Emulation differs from traditional penetration testing because it focuses on testing the **defenders (the SOC/Blue Team)** against a specific, real-world attack model, rather than just finding unpatched vulnerabilities.

- **Goal:** To determine if the SOC can **detect, triage, and respond** to a specific, realistic attack chain.
- **TTPs (Tactics, Techniques, and Procedures):** Emulation involves executing sequences of actions based on observed threat actor behavior, often mapped to the **MITRE ATT&CK** knowledge base (e.g., $T1566$ - Phishing, $T1210$ - Exploitation of Remote Services).

2. Emulation Frameworks

Frameworks and tools automate and standardize the process of simulating complex TTPs, allowing for repeatable and measurable tests.

- **MITRE Caldera:** A leading open-source framework designed to automate adversary emulation. It uses a **planner** to dynamically execute a sequence of $TTPs$ (called **abilities** in Caldera) on a target host based on an initial access point.
- **Simulation Example:** To test email security and endpoint detection, an emulation might
    1. Simulate a user clicking a link (initial access).
    2. Execute a fileless command ($T1059$ - Command and Scripting Interpreter).
    3. Attempt to establish persistence ($T1547$ - Boot or Logon Autostart Execution).

3. Red-Blue Team Collaboration

Adversary Emulation thrives on collaboration between offensive and defensive teams to improve security effectiveness.

- **Red Team Role (Offense):** Executes the emulation plan to mimic the threat actor.
- **Blue Team Role (Defense/SOC):** Monitors their tools (SIEM, EDR) to detect and respond to the emulation activities.
- **Outcome:** The results directly inform the defense strategy:
    - **Validate Controls:** Confirms whether existing security tools generate alerts for the specific TTPs.
    - **Improve Rules:** Identifies detection gaps and leads to the creation of new, highly-targeted **detection rules** (e.g., new YARA rules or SIEM correlation searches) to stop that specific threat in the future.

Key Objective

The ultimate objective is to transform the SOC from passively monitoring alerts to proactively anticipating and preparing for the behaviors of specific, relevant threat groups.

Security Metrics and Executive Reporting

This area of study focuses on translating the technical performance of the Security Operations Center (SOC) into quantifiable business risk and strategic recommendations for leadership.

1. Advanced SOC Metrics

These metrics move beyond basic counts to provide deeper insight into the effectiveness and efficiency of the SOC's operations.

- **Dwell Time:**
    - **Definition:** The amount of time an attacker is present within the network **from the moment of initial compromise until the point of detection**.
    - **Significance:** A critically important metric. **Lower Dwell Time** indicates superior security controls and hunting capabilities. A high dwell time suggests a blind spot that allows threats to persist unnoticed for long periods.
- **False Positive Rate (FPR):**

    - **Definition:** The ratio of incorrect alerts (alerts that are not actual threats) to the total number of alerts generated.
    - **Significance:** A high FPR leads to **alert fatigue** among analysts, reducing the efficiency and potentially causing genuine threats to be missed. Optimizing detection rules to lower this rate is a continuous goal.
- **Incident Resolution Rate:**
    - **Definition:** The percentage of security incidents that are fully contained, eradicated, and closed within a specified timeframe (e.g., within 24 hours).
    - **Significance:** Measures the **efficiency and capacity** of the SOC team to handle their workload effectively.

2. Executive Reporting

Effective reporting transforms technical data into business language that non-technical stakeholders can use for decision-making and resource allocation.

- **Presentation:** Use **clear visualizations** (charts, graphs) and a compelling **narrative** that focuses on risk and mitigation, not just technical details.
- **Focus Areas:**
    - **Risk Reduction:** Instead of reporting only on "blocked IPs," report on "how risk was reduced" (e.g., "The reduction in Dwell Time from 60 days to 7 days means we reduced the potential business impact of a breach by 88%").
    - **Strategic Needs:** Tie metrics to requests for resources, such as a high MTTD metric being used to justify purchasing a new EDR system or hiring more analysts.
- **Incident Summaries:** When reporting on a specific incident, the summary must be concise, covering **what happened, the business impact, and the steps taken to prevent recurrence.**

3. Continuohus Improvement

Metrics serve as the quantifiable foundation for identifying weaknesses and driving continuous process and tool enhancement.

- **Gap Identification:** Metrics act as signals for process failures. For example:
    - A consistently **high MTTD** suggests a failure in the **Detection Phase** (e.g., poor log visibility, missing correlation rules, or outdated monitoring tools).
    - A high **MTTR** suggests a failure in the **Response Phase** (e.g., manual processes, lack of SOAR automation, or slow escalation).
- **Proposing Solutions:** Metrics provide the evidence needed to propose solutions (e.g., "To reduce our MTTD, we recommend implementing the new endpoint data source").

The ultimate objective is to establish a data-driven loop: **Measure $\rightarrow$ Analyze $\rightarrow$ Improve $\rightarrow$ Measure Again.**

PRATICAL

1. Hypothesis Development & Elastic Security Query

Hypothesis: An adversary is using compromised credentials to assign administrative privileges to a standard user account to maintain persistence.

Tool: Elastic Security (Kibana) **Query (KQL):** `event.code: "4672" and user.name: "testuser"`

Evidence Log

Timestamp	User	Event ID	Notes
2025-08-18 15:00:00	testuser	4672	Unexpected admin role assigned; user is not in IT.
2025-08-18 15:05:12	testuser	4674	Attempted access to sensitive registry keys.

Export to Sheets

2. Threat Intelligence & Velociraptor Validation

Task: Identify if the source of the login is a known malicious actor and check for post-escalation artifacts.

- **AlienVault OTX (Threat Intel):** Search for the source IP found in the Elastic logs. If the IP is flagged under **T1078 (Valid Accounts)**, it confirms the credentials were leaked.
- **Velociraptor (Endpoint Analysis):** Run a VQL query to check for suspicious processes spawned by `testuser` immediately after the privilege escalation.

Velociraptor VQL Query:

SQL

```
SELECT Name, CommandLine, ParentName, Exe FROM pslist()
WHERE User =~ "testuser" AND Name =~ "powershell.exe|cmd.exe"
```

3. Threat Hunting Report

Executive Summary: Incident ID-2025-08-18

Mapping: MITRE ATT&CK **T1078** (Valid Accounts) & **T1078.002** (Domain Accounts).

Findings: A hunt initiated by an unauthorized **Event ID 4672** (Special Logon) revealed that user account `testuser` was granted administrative privileges at 15:00:00 UTC. Cross-referencing with **AlienVault OTX** confirmed the source IP is associated with known credential harvesting campaigns. **Velociraptor** telemetry showed `testuser` executing an obfuscated PowerShell script to modify domain persistence settings.

Recommendation: Immediately disable `testuser` credentials, revoke domain admin permissions, and initiate a full password reset for the domain.

Remediation Query (Velociraptor)

Once the hunt validates that `testuser` is running malicious processes under escalated privileges, use this VQL query to terminate the activity across the domain.

VQL Remediation Task:

SQL

```
-- Identify and terminate suspicious processes for 'testuser'
SELECT Name, ProcessID, CommandLine,
       { SELECT kill(pid=ProcessID) FROM info() } AS Action
FROM pslist()
WHERE User =~ "testuser"
  AND Name =~ "powershell|cmd|psexec"
```

2. Hunting Evidence Dashboard (Elastic Security)

In your practice, you would visualize the **Event ID 4672** spike to identify if this is an isolated incident or a widespread credential compromise.

- **Filter:** `winlog.event_id: 4672`
- **Aggregation:** Top Users by Count

3. Final Hunting Report (MITRE Mapping)

Category	Detail
Hunting Goal	Detect credential misuse and privilege escalation.
MITRE ATT&CK	**T1078.002** (Valid Accounts: Domain Accounts)
Indicators (IOCs)	IP: `192.168.1.105` (Flagged in OTX); Event: `Logon ID 0x3E7`
Summary	Hunt confirmed an unauthorized Special Logon (4672) for `testuser`. AlienVault OTX validated the source IP as suspicious. Post-logon activity included `powershell.exe` execution identified via Velociraptor.
Status	**Remediated** (Process killed; Account disabled).

2. SOAR Playbook Workflow Diagram

Playbook Execution Summary

The objective of this playbook is to reduce the "Time to Respond" (TTR) by orchestrating actions between Splunk Phantom, CrowdSec, and TheHive. By automating the reputation check and the subsequent blocking of malicious entities, security analysts can focus on higher-level investigation while routine threats are mitigated in real-time.

Playbook Test Results

Based on the simulation conducted in Wazuh, here is the verified status of each automation node:

Playbook Step	Status	Notes
Check IP	Success	IP flagged as malicious via reputation service.
Block IP	Success	CrowdSec successfully blocked **192.168.1.102**.
TheHive Ticket	Success	Incident logged and synchronized for tracking.

Playbook Summary (Google Docs Draft)

Title: Phishing Automated Response Playbook (PARP-01)

Summary: This playbook automates response to phishing alerts by integrating Wazuh with Phantom. It enriches IP data via threat intelligence, automatically triggers a block in CrowdSec for malicious indicators, and logs the incident in TheHive. This reduces Mean Time to Respond (MTTR) by eliminating manual blocking and documentation steps.

4. Technical Integration

To practice the "Block via CrowdSec" task within Splunk Phantom, would configure a **REST API Call** node with the following logic:

Action: POST **URL:** `https://<CrowdSec_LAPI>/v1/decisions` **Body:**

JSON

```
{
  "value": "192.168.1.102",
  "type": "ip",
  "origin": "Splunk-Phantom-SOAR",
  "reason": "Phishing detection via SOAR Practice",
  "duration": "24h"
}
```

This post-incident analysis phase focuses on transforming security data into actionable intelligence. By identifying the origin of the phishing breach and quantifying response effectiveness, the SOC can implement targeted improvements to defense mechanisms.

Root Cause Analysis (The 5 Whys)

The 5 Whys method is utilized here to move beyond the surface-level error (user clicking a link) to identify the systemic failure in the email filtering infrastructure.

Question	**Answer**
1. Why was the email opened?	The user clicked a malicious link within a fake "Invoice" email.
2. Why was the link clicked?	The email appeared legitimate due to a spoofed sender address.
3. Why was the email delivered?	Weak email filtering rules failed to flag the spoofed header.
4. Why were rules weak?	SPF/DKIM/DMARC verification was not strictly enforced for external mail.
5. Why was it not	Configuration was skipped during the last mail server

Question	Answer
enforced?	migration. (**Root Cause**)

Causal Mapping: Fishbone Diagram

A Fishbone (Ishikawa) diagram helps categorize the contributors to the incident across people, processes, and technology.

SOC Metrics Calculation

Measuring Mean Time to Detect (MTTD) and Mean Time to Respond (MTTR) provides a benchmark for organizational agility during a threat.

- **MTTD (Detection Time):** 2 Hours
- **MTTR (Response Time):** 4 Hours

Metric Summary:

The mock incident yielded an MTTD of 2 hours and an MTTR of 4 hours. These metrics highlight a moderate detection delay attributed to filtering gaps. To improve posture, the SOC must prioritize automating alert triage to reduce detection latency and refine automated blocking playbooks to lower response times. (49 words)

3. Detection: The Wazuh Alert

When a suspicious file is downloaded, Wazuh's File Integrity Monitoring (FIM) or Sysmon integration triggers an alert. Wazuh captures the file path and metadata.

Alert Simulation Data

Alert ID	Description	Source IP	Priority	Status
005	Suspicious File Download	192.168.1.102	**High**	Open

Practical Tip: In the Wazuh Dashboard, you would see this under **Security Events**. You can extract the `syscheck.sha1` or `syscheck.md5` field directly from the JSON log to send to your automation pipeline.

2. Configuration: Integrating Tools

To automate the validation, you must link these tools.

Step A: Wazuh to TheHive Integration

Wazuh does not have a native "button" for TheHive. You must use a **Custom Integration Script**.

- **Path:** `/var/ossec/integrations/custom-thehive.py`
- **Action:** This script takes the JSON output from Wazuh and uses an API POST request to create a "Case" in TheHive automatically.

Step B: Enabling VirusTotal in Cortex

TheHive uses **Cortex** to run "Analyzers."

1. Log into **Cortex**.
2. Go to **Organization** > **Analyzers**.
3. Find **VirusTotal_GetReport** and click **Configure**.
4. Enter your **VirusTotal API Key**.

3. Automated Validation Workflow

Once configured, the manual triage work is replaced by a digital "handshake."

1. **Ingestion:** Wazuh sends the file hash to TheHive.
2. **Creation:** TheHive creates a case and adds the hash as an **Observable**.
3. **Analysis:** Cortex sees the new observable and automatically queries VirusTotal.
4. **Enrichment:** The results (e.g., "35/70 engines detect this as Emotet") appear as a report attached to the case.

Summary of Results (50 Words)

TheHive receives the Wazuh alert and extracts the file hash. Cortex automatically triggers a VirusTotal lookup, confirming a malicious status (e.g., 42/70 detections). TheHive then tags the alert as "Malicious," allowing the analyst to skip manual searching and proceed immediately to host isolation or file deletion.

4. Visual Evidence (Mock Process)

Visualizing the Triage UI

When you log in, your dashboard should transition from "Alert" to "Analyzed Intelligence" like this:

TheHive Dashboard View:

Case #005: Suspicious File Download

- **Observable:** `e3b0c44298fc1c149afbf4c8996fb...`
- **Cortex Status:** [**COMPLETED**]
- **VirusTotal Result:** [**MALICIOUS (45 Detections)**]
-

5. Analysis with Velociraptor (Live Forensic)

Velociraptor allows you to query live systems without taking them offline. To identify suspicious connections, you use the **VQL (Velociraptor Query Language)** interface.

The Practical Step:

1. Access the Velociraptor GUI and select your **Target Client**.
2. Go to **Collected Artifacts** and click the **Plus icon** to add a collection.
3. Search for `Windows.Network.Netstat`.
4. Launch the artifact.
5. **Identify Anomalies:** Look for processes like `cmd.exe`, `powershell.exe`, or unknown binaries (`temp.exe`) that have an **ESTABLISHED** connection to an external IP address.

2. Evidence Acquisition with FTK Imager

Once the connection is identified, you must "freeze" the state of the machine to analyze the malicious code running in memory.

The Practical Step:

1. Run **FTK Imager Lite** (portable version) from a secure USB drive on the suspect VM.
2. Select **File > Capture Memory**.
3. Choose your destination path and ensure **"Create AD1 file"** or **"Include pagefile"** is checked if you need deep analysis.
4. **Crucial Step:** Once finished, FTK Imager will present a "Verification Results" box. **Copy the SHA256 hash immediately.** This is your primary proof that the evidence hasn't been altered.

3. Maintaining the Chain-of-Custody

The Chain-of-Custody (CoC) ensures that the evidence you found in Velociraptor and captured with FTK Imager is admissible in a legal or corporate disciplinary proceeding.

Documenting the Evidence Collection

Every time the file is moved (e.g., from the infected VM to a Forensic Workstation), it must be logged.

Item	Description	Collected By	Date	Hash Value (SHA256)
Volatile RAM	Memory Dump of Win-VM-01	SOC Lead	2025-08-18	`7f83b1657ff1...`
Network Log	Velociraptor Netstat CSV Export	SOC Lead	2025-08-18	`a1b2c3d4e5f6...`

4. Practical Explanation: Bridging the Tools

- **Velociraptor** tells you **who** is talking to **where** (The "Live" lead).
- **FTK Imager** captures the **what** (The "Physical" evidence).
- **Chain-of-Custody** ensures the **integrity** (The "Legal" requirement).

Why use both? If you only use Velociraptor, an attacker can delete the malware, and your "proof" is gone. By using FTK Imager to capture the RAM right after seeing the suspicious connection in Velociraptor, you capture the attacker's tools while they are still loaded in memory.

. Emulation Simulation: Spearphishing (T1566)

In this scenario, a Caldera agent installed on a target Windows machine executes an adversary profile designed to simulate a successful spearphishing attachment execution.

The Practical Workflow

1. **Caldera Setup:** You create an "Operation" using the **Spearphishing** ability. This might involve downloading a payload via a PowerShell script hidden in a link.
2. **Wazuh Observation:** Wazuh monitors for suspicious process spawning (e.g., `Outlook.exe` spawning `powershell.exe`) or unusual file creation in temporary directories.
3. **Correlation:** Match the Caldera activity logs with the Wazuh dashboard alerts.

Emulation Log Table

Timestamp	TTP	Detection Status	Notes
2025-08-18 17:00:00	**T1566.001**	**Detected**	Phishing attachment blocked by real-time FIM.
2025-08-18 17:05:00	**T1059.001**	**Partially Detected**	PowerShell execution flagged, but obfuscation bypassed one rule.

2. Practical Configuration: Linking the Tools

Step A: Configuring Caldera

Install the **Sandcat** (C2 agent) on your target VM. Create an operation that focuses on the **Initial Access** tactic.

- **Target:** Win-VM-SOC-Lab
- **Adversary Profile:** "Phish_and_Execute"

Step B: Enhancing Wazuh Detection

To detect a spearphishing link execution, standard logging might not be enough. You need to enable **Sysmon** on the target VM and ingest those logs into Wazuh.

1. Install Sysmon with a configuration like `SwiftOnSecurity`.
2. Update the Wazuh agent `ossec.conf` to monitor the **Microsoft-Windows-Sysmon/Operational** event log.
3. Create a custom rule in the Wazuh Manager to trigger when `powershell.exe` has an `Outlook.exe` parent.

3. Emulation Report (Summary)

Report Title: Spearphishing Emulation Test (T1566)

Summary: The emulation successfully triggered 75% of expected alerts. Wazuh’s File Integrity Monitoring (FIM) immediately flagged the creation of the malicious attachment. However, a significant detection gap was identified: the PowerShell execution bypassed standard "Command Line" rules due to Base64 obfuscation. While the activity was logged in Sysmon, no high-priority alert was raised in real-time.

Recommendation: Implement specific Wazuh rules to monitor for EncodedCommand flags in PowerShell and correlate them with parent processes associated with productivity software (e.g., Outlook, Teams) to close the visibility gap.

4. Bridging the Gap (Visualizing the Feedback Loop)

The goal is to move from "Blind" to "Informed."

1. **Simulate:** Caldera executes a malicious command.
2. **Detect:** Wazuh misses it (Silent failure).
3. **Analyze:** You look at the raw Sysmon logs.
4. **Improve:** You write a new rule to detect that specific behavior.
5. **Retest:** Run Caldera again; this time, the SOC receives a "High" alert.

Metrics Dashboard in Elastic Security

Elastic Security (Kibana) allows you to build visualizations using ES|QL or classic lens panels to track the health of your detection pipeline.

The Dashboard Setup

- **Mean Time to Detect (MTTD):** Measures the average time from the start of activity to alert generation.
    - *Calculation:* `(Alert Time - Event Time) / Total Incidents`
- **Mean Time to Respond (MTTR):** Measures efficiency from detection to mitigation.
    - *Calculation:* `(Mitigation Time - Alert Time) / Total Incidents`
- **False Positive Rate:** Tracks rule accuracy.
    - *Goal:* Decrease this to prevent "Alert Fatigue."

Metric	Current Value	Target (SLA)
MTTD	2.5 Hours	< 1 Hour
MTTR	4.2 Hours	< 2 Hours
False Positive Rate	18%	< 10%

[Image: Kibana dashboard with three gauges showing red/yellow/green zones for MTTD, MTTR, and False Positives]

2. Mock Incident: Dwell Time Analysis (Google Sheets)

"Dwell Time" is the total duration an attacker has access before being expelled.

Dwell Time Table (Example Incident)

Phase	Timestamp	Duration
Compromise (Entry)	2025-08-18 09:00:00	-
Detection (Alert)	2025-08-18 11:30:00	2.5h (MTTD)
Resolution (Exit)	2025-08-18 15:45:00	4.25h (MTTR)
Total Dwell Time	-	**6.75 Hours**

Findings (Analysis Summary):

Analysis of the mock incident reveals a total dwell time of 6.75 hours. The detection phase (MTTD) accounted for 37% of the risk window. A 2.5-hour delay in identification allowed the attacker to begin lateral movement, confirming a gap in real-time lateral move detection for non-privileged accounts.

---

3. Executive Summary (Google Docs Draft)

Executive Report: SOC Operations – Q3 Performance Review

Summary:

This quarter, the SOC successfully processed 15 critical incidents with a Mean Time to Detect (MTTD) of 2.5 hours and a Mean Time to Respond (MTTR) of 4.2 hours. While containment speed is within industry benchmarks, detection latency remains a critical risk.

The average Dwell Time for simulated spearphishing attacks peaked at nearly 7 hours, providing ample opportunity for data exfiltration.

Key Gap:

A high False Positive Rate (18%) is currently diverting 20% of analyst capacity toward benign events, directly delaying response times for genuine threats.

Recommendations:

1. **Detection Tuning:** Dedicate 10 engineering hours weekly to "Noise Cancellation" in Elastic Security to lower the FP rate below 10%.
2. **Automation (SOAR):** Implement automated endpoint isolation for high-fidelity credential dumping alerts to reduce MTTR from 4 hours to <15 minutes.
3. **Visibility:** Increase log ingestion for server-side lateral movement logs (WMI/SMB) to close detection gaps identified in dwell analysis.

8. Attack Simulation & Adversary Emulation

The initial breach is executed via Metasploit, followed by Caldera to simulate post-exploitation lateral movement.

- **Initial Breach:** Targeting the Metasploitable2 Samba service using `usermap_script`. This grants immediate root access.
- **Emulation (T1210):** Caldera's Sandcat agent triggers remote service exploitation to test Wazuh's behavioral detection.

Wazuh Detection Log: | Timestamp | Source IP | Alert Description | MITRE Technique | | :--- | :--- | :--- | :--- | | 2025-08-18 16:00:00 | 192.168.1.102 | Samba `usermap_script` Exploit | **T1210** |

2. Automated Triage & Containment (SOAR)

This phase demonstrates "Security Orchestration, Automation, and Response."

1. **Alert:** Wazuh captures the exploit and forwards the JSON event to **TheHive**.
2. **Case:** TheHive creates a case and tags the indicator (Attacker IP: `192.168.1.102`).
3. **Containment:** A SOAR playbook triggers **CrowdSec**, which updates firewall rules on the VM host to drop all traffic from the attacker IP.
4. **Verification:** A ping test confirms the attacker is blocked while the service remains online for other users.

3. Post-Incident Analysis: Fishbone Diagram (Draw.io)

For the Root Cause Analysis (RCA), use the "5 Whys" to populate a Fishbone (Ishikawa) diagram. This visualizes why the incident occurred across various vectors.

RCA Data Points:

- **People:** Lack of security awareness regarding legacy dev environments.
- **Process:** Shadow IT migration process failed to trigger a vulnerability scan.
- **Technology:** Legacy Samba version (3.0.20) in production-linked VLAN.
- **Management:** Segment classified incorrectly as "Air-gapped."

Getty Images

4. Metrics & Stakeholder Reporting

The data from **Elastic Security** is summarized for executive review.

Key Performance Indicators (KPIs):

- **MTTD:** 45 Seconds (Detection via Wazuh).
- **MTTR:** 3.5 Minutes (Automated blocking via CrowdSec).
- **Dwell Time:** 4 Minutes total.

Executive Stakeholder Briefing (Draft)

Incident Summary: On Aug 18, our SOC identified and neutralized a root-level exploit on a legacy development server within 4 minutes of the breach. **Business Impact:** Neutralized before data exfiltration occurred. System downtime was limited only to the attacker's connection. **Metrics:** Automated response (MTTR) performed 90% faster than manual isolation. **Strategic Improvements:** We are decommissioning the vulnerable Samba instance and implementing automated "Asset Discovery" scans to prevent unmanaged legacy servers from entering our network in the future.