

Incident Response: Detection & Analysis

Alexis Ahmed

Red & Blue Team Instructor @INE

Red Team Lead @HackerSploit

Key Concepts

- + Threat Detection Foundations
- + SIEM Operations
- + Triage & Escalation
- + Endpoint/Host Analysis
- + Network Analysis

MAJOR TOPICS

- + Detection Foundations
- + Log Sources & Telemetry
- + Log Collection & Aggregation
- + SIEM Operations & Detection Engineering
- + Incident Triage Process
- + IOC Identification, Analysis & Enrichment
- + First Response
- + Endpoint Analysis
- + SIEM Based Log Analysis
- + Windows Log Analysis
- + Linux Log Analysis
- + Network Traffic Analysis



LEARNING OUTCOMES

- + Explain the role and importance of the detection and analysis phase in the incident response lifecycle.
- + Differentiate between events, alerts, and incidents, and describe the process of alert triage.
- + Identify key log sources across endpoint, network, and application layers, and explain their relevance to incident response.
- + Collect and analyze logs from Windows and Linux systems using native and third-party tools.
- + Configure and utilize centralized logging solutions such as Splunk and the ELK stack for effective log analysis.
- + Develop and evaluate detection rules, manage alert noise, and prioritize incident response efforts.
- + Identify, enrich, and contextualize indicators of compromise (IOCs) using internal data and threat intelligence.
- + Perform initial response actions and deep analysis to assess the scope and impact of security incidents.
- + Conduct endpoint and log-based investigations using tools like Sysmon, wevtutil, EvtxECmd, and Chainsaw.
- + Analyze network traffic using PCAPs, flow data, and logs to detect scanning, attacks, and exfiltration.

PREREQUISITES

- + Understanding of the OSI Model and TCP/IP Networking
- + Familiarity with Common Network Protocols
- + Basic Knowledge of Operating Systems (Windows and Linux):
- + Basic experience with command-line tools (e.g., PowerShell, Bash)
- + Familiarity with security concepts and terminology



LET'S GO!





The Role of Detection & Analysis in IR

Detection & Analysis

The ***Detection and Analysis*** phase of Incident Response (IR) is a critical stage where security teams ***identify, validate, and assess potential security incidents***.

It bridges the gap between preparation and active response, ensuring that threats are detected early, analyzed thoroughly, and escalated appropriately for containment and remediation.

Purpose

- + To detect suspicious activity through automated systems or manual observation.
- + To analyze and triage alerts and events to determine if they constitute a real security incident.
- + To scope and classify incidents, establishing their severity, impact, and priority.
- + To enable timely response by ensuring only confirmed, prioritized incidents are escalated.

Key Activities in the Detection & Analysis Phase

1. Event Collection

- + Security tools (e.g., firewalls, EDRs, SIEMs, IDS/IPS) collect raw telemetry such as logs, events, flows, and system alerts.
- + Common data sources include:
 - + Windows Event Logs, Sysmon, Linux syslog
 - + Network traffic (NetFlow, PCAPs)
 - + Application logs (web servers, VPNs)

Key Activities in the Detection & Analysis Phase

2. Alert Generation

- + Events are analyzed by detection tools (e.g., SIEMs) to generate alerts based on:
 - + Signature-based rules
 - + Behavioral analytics
 - + Correlation engines or machine learning

Key Activities in the Detection & Analysis Phase

3. Alert Triage

- + Analysts review incoming alerts to:
 - + Validate their legitimacy
 - + Eliminate false positives
 - + Prioritize based on severity and impact
- + Triage uses:
 - + Context enrichment (e.g., asset value, user identity)
 - + Threat intelligence (e.g., known IOCs, malware hashes)

Key Activities in the Detection & Analysis Phase

4. Incident Identification

- + Confirmed alerts are escalated into incidents when they meet certain criteria (e.g., policy violation, confirmed compromise).

- + This step involves:
 - + Scoping the incident (what systems/users are affected?)
 - + Determining the entry point, attacker actions, and intent
 - + Assessing potential data loss or business impact

Key Activities in the Detection & Analysis Phase

5. Host and Network Analysis

- + Deep-dive investigation into endpoints and network traffic:
 - + On endpoints: process trees, file activity, registry changes, login behavior
 - + On networks: unusual communications, lateral movement, C2 traffic

Key Activities in the Detection & Analysis Phase

6. Documentation and Handoff

- + All findings are documented for:
 - + Handoff to containment and eradication teams
 - + Building timelines and attack paths
 - + Improving detection content (rules, playbooks)

Skills Required in This Phase

- + Proficiency in log analysis and SIEM usage
- + Knowledge of Windows and Linux internals
- + Network traffic analysis (PCAP, protocol decoding)
- + Use of forensic and investigative tools (e.g., Wireshark, Velociraptor, Splunk, Zeek)
- + Familiarity with MITRE ATT&CK to map behaviors to known techniques

Outcomes

- + Confirmed and scoped security incidents
- + Timeline of attacker activity
- + Prioritized incident classification
- + Handoff to Containment, Eradication & Recovery phase



Understanding Events, Alerts & Incidents

Understanding Events, Alerts & Incidents

In the context of security monitoring and incident response, it is essential to distinguish between events, alerts, and incidents.

Misunderstanding these terms can lead to inefficient triage, wasted analyst time, or even missed breaches/intrusions.

Events

- + An **event** is any ***observable occurrence*** in a system, network, or device.
- + Examples:
 - + A user logs into a workstation
 - + A file is opened or modified
 - + A process is created or terminated
 - + A packet is transmitted across a network

Events

- + Key Characteristics:
 - + Generated in large volumes (millions per day in large networks)
 - + Most are benign and part of normal operations
 - + Logged for auditing, troubleshooting, and monitoring

Think of events as raw data. They're the atoms of your security telemetry.

Alerts

- + An **alert** is a notification generated by a security monitoring system (e.g., SIEM, IDS, antivirus) ***when a predefined rule or condition is met***, often indicating potentially suspicious or malicious activity.
- + Examples:
 - + A login attempt from an unauthorized country
 - + A malware signature detected in memory
 - + A user added to the Domain Admins group unexpectedly
 - + An unusually large outbound data transfer

Alerts

- + Key Characteristics:
 - + Based on detection rules, heuristics, or behavioral baselines
 - + May include false positives
 - + Require triage and validation by analysts

An alert is a signal: “Hey, something might be wrong.”

Incidents

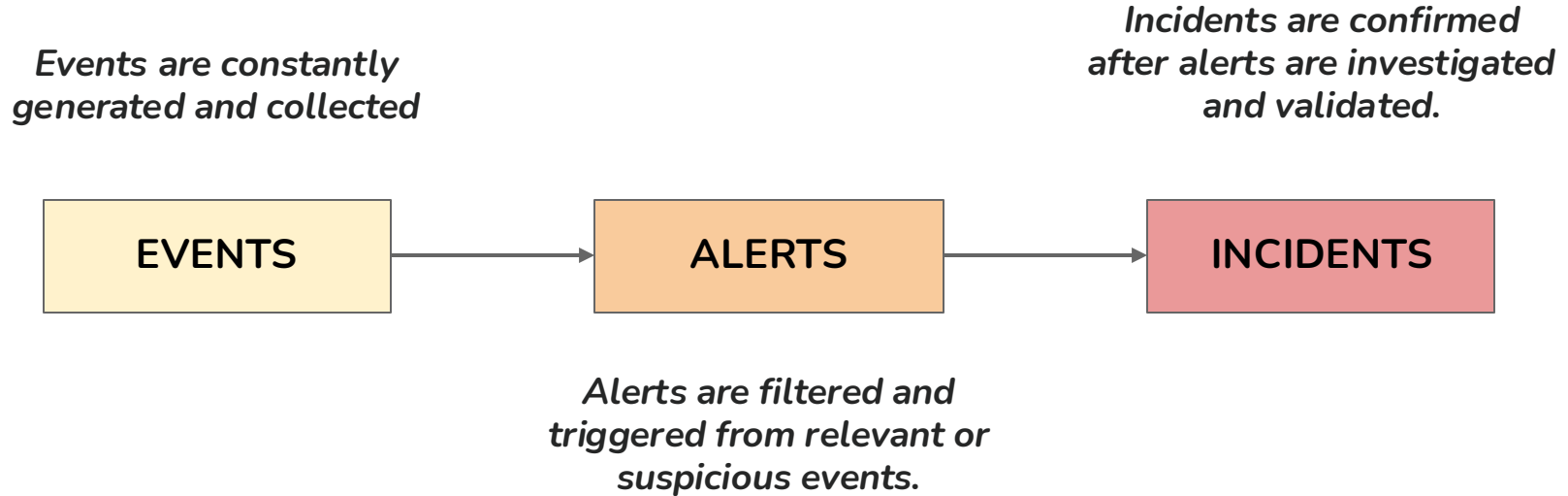
- + An **incident** is a **confirmed** or **suspected** breach of security policy or acceptable use, as determined by the analysis of alerts and associated data.
- + Examples:
 - + A ransomware attack encrypting files on multiple endpoints
 - + An attacker gaining unauthorized access to sensitive systems
 - + Exfiltration of customer *PII* data to an external server
 - + Credential theft and lateral movement within the network

Incidents

- + Key Characteristics:
 - + Incidents are actionable and require an IR response
 - + Can involve one or multiple alerts and events
 - + Must be documented, analyzed, and remediated

An incident is the confirmed threat that must be investigated, contained, and resolved.

Relationship Between Them



Relationship Between Them



Why It Matters in IR

- *Helps focus the analyst's attention on what truly matters*
- *Prevents alert fatigue by filtering noise from signal*
- *Enables proper incident classification, escalation, and documentation*
- *Ensures efficient use of IR resources*

A man with glasses and a beard is shown in profile, working on a computer in a dark room. The background is dark, and the man is wearing a dark shirt. The text is overlaid on the left side of the image.

Types of Log Sources: Endpoint, Network & Application

What is Logging?

Logging refers to the ***automatic recording*** of ***system and application events*** that occur on a device, server, or network.

These logs are generated by operating systems, applications, and network devices to capture details about activities, errors, requests, and behaviors.

Why Logging Exists

- + **Logging** plays a critical role in system administration, troubleshooting, and security operations. Its primary purposes include:
 - + Auditing: Track user actions and system changes for accountability
 - + Troubleshooting: Diagnose software or hardware failures
 - + Behavior Monitoring: Understand normal vs. anomalous activity
 - + Security Monitoring: Detect unauthorized access or malicious activity
 - + Compliance: Meet regulatory requirements (e.g., HIPAA, PCI-DSS)

Without logs, detecting incidents and understanding their root cause would be nearly impossible.

Types of Log Sources

In the context of security monitoring and incident response, logs generally come from three main categories of sources:

1. Endpoint Logs
2. Network Logs
3. Application Logs

1 - Endpoint Logs

- + These are Logs that are generated by individual systems or devices such as workstations, servers, laptops, and mobile devices.
- + Common Sources:
 - + Windows Event Logs (Security, Application, System)
 - + Sysmon Logs (detailed telemetry of system activity)
 - + Linux Logs (e.g., /var/log/syslog, auth.log, Auditd logs)
 - + Antivirus and EDR Logs
 - + Application crash logs or process monitoring tools

1 - Endpoint Logs

- + Applicability/Importance in IR:
 - + Detect user logins, failed logins, process execution, registry changes
 - + Investigate lateral movement, persistence, and privilege escalation

2 - Network Logs

- + These are Logs that are generated by network infrastructure and monitoring tools that provide visibility into traffic between systems.
- + Common Sources:
 - + Firewall Logs (allow/deny rules, NAT activity)
 - + Intrusion Detection/Prevention Systems (IDS/IPS) (e.g., Snort, Suricata)
 - + NetFlow/IPFIX (summarized metadata about network flows)
 - + Packet Captures (PCAPs) (full traffic inspection)
 - + VPN, DHCP, DNS, and Proxy Logs

2 - Network Logs

- + Applicability/Importance in IR:
 - + Detect command-and-control (C2) traffic, data exfiltration
 - + Trace lateral movement between hosts
 - + Identify unauthorized external connections

3 - Application Logs

- + These are Logs that are produced by software applications and services to track user activity, errors, and operations within the application.
- + Common Sources:
 - + Web Server Logs (e.g., Apache, Nginx)
 - + Database Access Logs
 - + Authentication Systems (e.g., LDAP, Active Directory)
 - + VPN Gateways and Email Gateways
 - + Custom Application Audit Logs

3 - Application Logs

- + Applicability/Importance in IR:
 - + Identify compromised user accounts or credentials
 - + Trace unauthorized access to sensitive data
 - + Detect application-layer attacks (e.g., SQLi, file uploads)

How These Sources Work Together (Correlation)

Effective incident detection and analysis depend on correlating data across these sources for full visibility and context.

Log Source Type	Example Use Case in IR
Endpoint	Identify malware executed on a workstation
Network	Detect outbound traffic to a known malicious IP
Application	Trace user login from an unusual location or time

Windows Event Logging

Windows Event Logging

Windows Event Logging is a ***built-in*** mechanism used by the **Microsoft Windows** operating system to ***record*** and ***store*** information about ***system activity***, ***application behavior***, ***security events***, and other operating system operations.

These logs are a critical component for system auditing, troubleshooting, and security monitoring.



How Logging Works on Windows

How Logging Works on Windows

1 - Event Generation

- + Whenever an **action occurs** on the system (e.g., a user logs in, a service starts, a file is modified), Windows generates an event.

This event includes:

- + A unique Event ID
- + A timestamp
- + The source (e.g., *Microsoft-Windows-Security-Auditing*)
- + A severity level (*Information, Warning, Error, Critical*)
- + Event data (user account, process, file path, IP address, etc.)

How Logging Works on Windows

2 - Event Logging Service

- + The Windows Event Log service (*eventlog*) captures these events and writes them to structured log files stored in the **.evtx** format in the system directory:

```
C:\Windows\System32\winevt\Logs\
```

How Logging Works on Windows

3 - Storage & Retention

- + Each log type (**Security**, **System**, **Application**, etc.) is stored in a separate **.evtx** file.
- + These logs can be rotated, archived, or overwritten based on configured settings (size, retention policy, etc.).

How Logging Works on Windows

4 - Accessing Logs

- + Logs can be viewed using:
 - + Event Viewer GUI (`eventvwr.msc`)
 - + PowerShell commands (e.g., `Get-WinEvent`, `Get-EventLog`)
 - + Command Line tools (e.g., `wevtutil`)
 - + SIEM solutions (e.g., logs shipped to Splunk, ELK)



Types of Windows Event Logs

Types of Windows Event Logs

1 - Security Logs

- + Related to system access, login/logout activity, account management, privilege use
- + Controlled via audit policies.
- + Managed by the **Security Auditing subsystem**.
- + Common Event IDs:
 - + 4624 – Successful logon
 - + 4625 – Failed logon
 - + 4670 – Permissions on object changed
 - + 4688 – Process created

Types of Windows Event Logs

2 - System Logs

- + Events logged by Windows system components and drivers (e.g., kernel, drivers, services)
- + Useful for troubleshooting hardware or OS-level issues
- + Common Event Sources: Service Control Manager, Ntfs, Kernel-General

Types of Windows Event Logs

3 - Application Logs

- + Events written by user-mode applications
- + May contain error messages, failures, warnings, or custom application logging
- + Example sources: MExchange, .NET Runtime, SQLServer

4 - Setup Logs

- + Installation-related logs, including Windows Updates or system setup components

Types of Windows Event Logs

5 - Forwarded Events

- + Centralized collection of events from remote Windows systems
- + Requires Windows Event Forwarding (WEF) and a configured subscription manager

Types of Windows Event Logs

6 - Additional Logs (via Microsoft or 3rd Party Tools)

- + Sysmon Logs – Deep process, network, and driver monitoring (via Sysinternals Sysmon)
- + Windows Defender Logs
- + PowerShell Operational Logs
- + Terminal Services Logs



Use Cases in Incident Response

Use Cases in Incident Response

- + User Behavior Monitoring: Login patterns, unauthorized access.
- + Process Tracking: Suspicious or unexpected program execution.
- + Privilege Abuse Detection: Admin group changes, token abuse.
- + Lateral Movement Tracing: Remote access via RDP, WinRM.
- + Persistence Mechanism Discovery: Scheduled tasks, service installation.

A man with glasses and a beard is shown in profile, looking at a computer monitor in a dark room. The monitor displays some code or data. The overall scene is dimly lit, with the primary light source being the screen.

Demo: Windows Event Logging

Linux Logging

Linux Logging Explained

Linux logging refers to the process by which Linux-based systems **record events**, **messages**, and **system activity**.

These logs are **essential** for system administration, troubleshooting, performance monitoring, and security analysis.

Linux relies on a combination of text-based log files, logging daemons, and kernel-level messages to capture and store logs.



How Logging Works on Linux

How Logging Works on Linux

1 - Event Generation

- + Whenever a system component, application, or user action occurs, ***an event message*** is generated.
- + These events can be:
 - + System-level messages (e.g., service start/stop, kernel messages)
 - + Application output (e.g., errors, access logs)
 - + Security-related activity (e.g., login attempts)

How Logging Works on Linux

2 - Logging Daemons

- + The logging system routes and stores these events using logging daemons, including:
 - + **rsyslog** (default on most modern distros)
 - + **syslog-ng**
 - + **journald** (used by systemd-based systems)
- + These daemons handle:
 - + Message formatting
 - + Priority/severity tagging
 - + Routing to files, remote servers, or other logging systems

How Logging Works on Linux

3 - Storage & Formatting

- + Logs are stored as plain text files under the **/var/log/** directory.

<code>/var/log/auth.log</code>	<code># Authentication and authorization logs</code>
<code>/var/log/syslog</code>	<code># General system activity logs</code>
<code>/var/log/kern.log</code>	<code># Kernel messages</code>
<code>/var/log/messages</code>	<code># General system messages (non-systemd)</code>

How Logging Works on Linux

4 - Accessing Logs

- + Use `cat`, `less`, or `tail` to view log files
- + Use `grep` or `awk` to search and filter
- + Use `journalctl` on `systemd` systems to query the system journal

```
# View recent logs with details
journalctl -xe

tail -f /var/log/auth.log
```



Types of Linux Log Files

Types of Linux Log Files

1 - Authentication Logs

- + Location(s):
 - + `/var/log/auth.log`
 - + `/var/log/secure`
- + Contains login attempts, SSH access, sudo usage.
- + Useful for detecting brute force attacks, privilege escalation, unauthorized logins.

Types of Linux Log Files

2 - System Logs

- + Location(s):
 - + `/var/log/syslog`
 - + `/var/log/messages`
- + Includes general system and daemon activity.
- + Useful for spotting service crashes, system errors, reboots.

3 - Kernel Logs

- + Location: `/var/log/kern.log`
- + Captures messages from the Linux kernel (e.g., hardware errors, driver issues)

Types of Linux Log Files

4 - Application Logs

- + Location(s): Varies by application, often in `/var/log/` or custom paths. For Example:
 - + `/var/log/apache2/access.log`
 - + `/var/log/mysql/error.log`
- + Custom/3rd Party applications may write to their own log.

Types of Linux Log Files

5 - Audit Logs

- + Tool: `auditd` (Linux Audit Daemon)
- + Location: `/var/log/audit/audit.log`
- + Tracks system calls, file accesses, user activity at a granular level.
- + Used for regulatory compliance and detailed forensics.

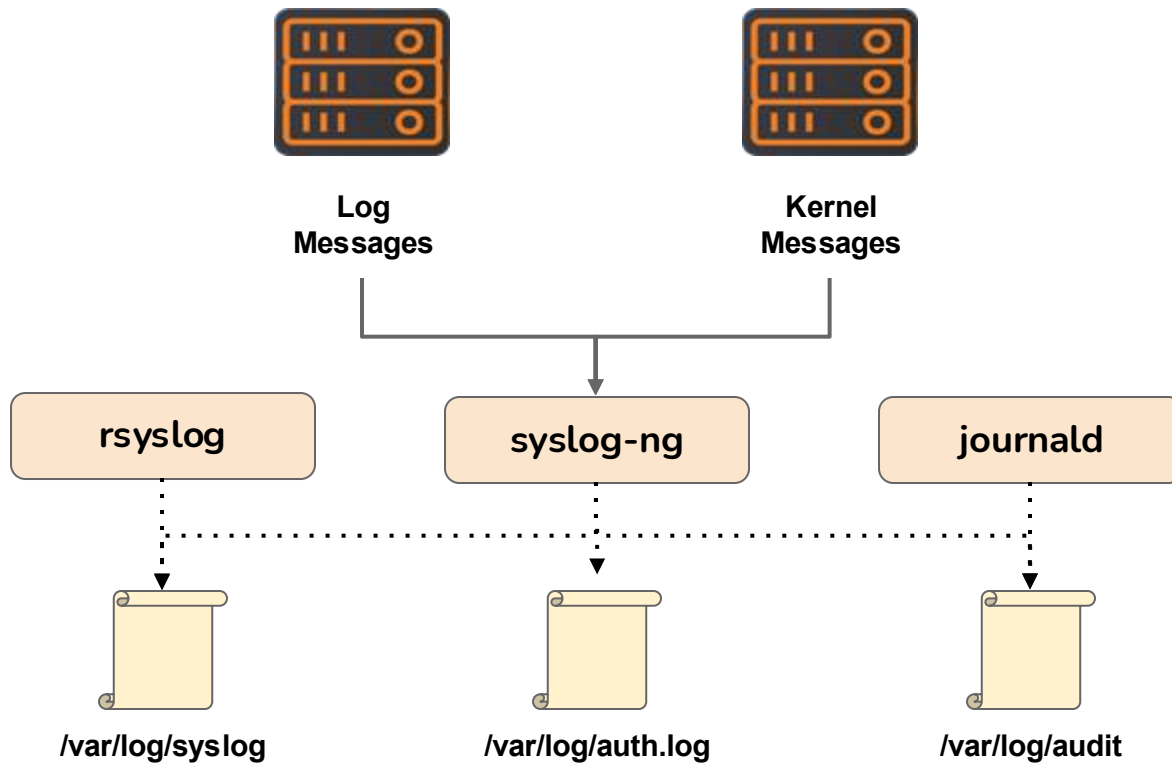
6 - Systemd Journal (if applicable)

- + Tool: `journald`
- + Command: `journalctl`
- + Captures kernel logs, service logs, authentication events, and more in a centralized binary journal



Reference: Linux Logging Architecture

Linux Logging Architecture





Use Cases in Incident Response

Use Cases in Incident Response

Use Case	Relevant Logs
Brute-force detection	<code>/var/log/auth.log</code>
Malware behavior	<code>/var/log/syslog</code> , <code>/var/log/messages</code> , <code>journalctl</code>
Root access tracking	<code>/var/log/auth.log</code> , <code>audit.log</code>
Lateral movement	SSH logs in <code>/var/log/auth.log</code>
Data exfiltration scripts	Custom script logs, process auditing

Why It Matters

- + *Linux logs provide deep visibility into system behavior and user actions*
- + *Critical for forensics, audit trails, and detection of compromise*
- + *Lightweight and customizable, logs can be centralized for SIEM integration*



Linux Log Files: Quick Reference

Use Cases in Incident Response

Log File	Location	Purpose
System Logs	<code>/var/log/syslog</code>	General system messages (Ubuntu/Debian)
General System Messages	<code>/var/log/messages</code>	System-wide logs (RHEL/CentOS)
Authentication Logs	<code>/var/log/auth.log</code>	SSH, sudo, and login activity
Secure Log (alt)	<code>/var/log/secure</code>	Auth & access logs on RHEL/CentOS
Kernel Logs	<code>/var/log/kern.log</code>	Messages from the Linux kernel
Audit Logs	<code>/var/log/audit/audit.log</code>	Detailed system call-level security events
Boot Logs	<code>/var/log/boot.log</code>	Boot-time system logs
Cron Logs	<code>/var/log/cron</code>	Scheduled task execution logs
Dmesg (Ring Buffer)	<code>dmesg</code>	Kernel messages from current boot
Application Logs	<code>/var/log/<app>/</code>	Varies (e.g., Apache: <code>/var/log/apache2/</code>)



Essential Linux Log Commands

Essential Linux Log Commands

+ View Logs

```
less /var/log/syslog  
less /var/log/auth.log
```

+ Real-Time Monitoring

```
tail -f /var/log/syslog  
journalctl -f
```

Essential Linux Log Commands

+ Search Logs

```
grep "Failed password" /var/log/auth.log  
journalctl | grep sshd
```

+ Filter by Time (journalctl)

```
journalctl --since "1 hour ago"  
journalctl --since "2024-05-01" --until "2024-05-15"
```

Essential Linux Log Commands

+ View Kernel Messages

```
dmesg | less
```



Log Collection, Shipping & Aggregation

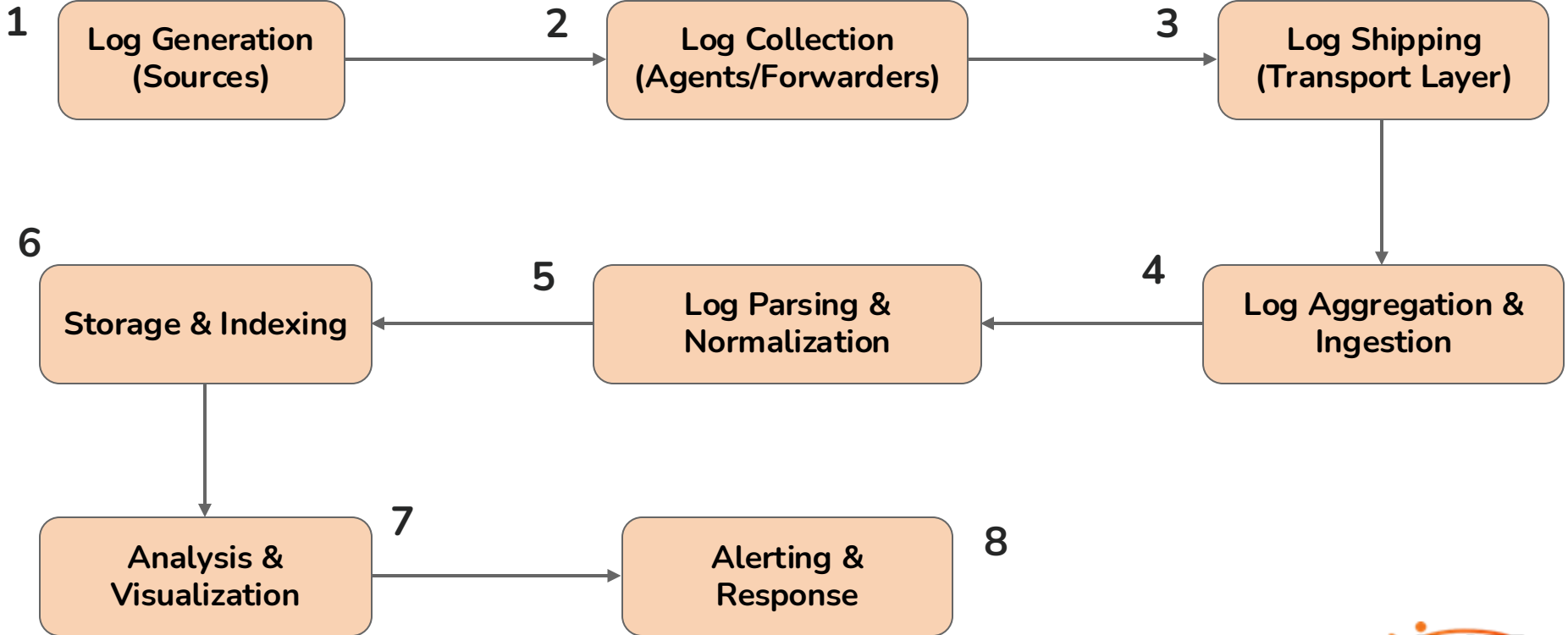
Centralized Logging

A **centralized logging** pipeline is the architecture and process through which logs from diverse systems are **collected, transported, processed, stored**, and **analyzed** in a **central location**.

This central location is typically a SIEM (Splunk) or a log analytics platform (ELK Stack).

It is essential for incident detection, threat hunting, auditing, and forensic investigation.

Core Components of a Centralized Logging Pipeline





Log Collection

What is Log Collection?

Log collection is the process of gathering log data from various systems, services, and devices within an IT environment.

These logs are generated by:

- + Endpoints (e.g., Windows, Linux)
- + Network devices (e.g., firewalls, routers)
- + Applications (e.g., web servers, databases)
- + Security tools (e.g., antivirus, IDS/IPS)

The primary goal of log collection is to centralize visibility, support monitoring, and prepare for analysis or alerting.



How Log Collection Works

Log collection is the first step in a centralized logging pipeline, crucial for monitoring, detection, alerting and forensic investigation.

- + Logs are either pulled (queried on demand) or pushed (sent automatically).
- + Collection can be agent-based (e.g., Filebeat, Winlogbeat) or agentless (e.g., syslog).
- + Logs may be collected in raw format or parsed/normalized during ingestion.

Push Based Collection

- + In a push model, the source system (e.g., a Linux server) actively sends (pushes) its logs to a central destination such as a log server or SIEM.
- + How it works:
 - + A log shipper/agent on the endpoint forwards logs.
 - + Sent over TCP/UDP, Beats, or HTTP protocols.
 - + The receiving system is passive, waiting for incoming logs.
- + Examples:
 - + rsyslog on Linux pushing logs to a remote server.
 - + Winlogbeat pushing Windows Event Logs to ELK.
 - + A firewall pushing logs via syslog to a SIEM.

Push Based Collection

+ Benefits:

- + Near real-time delivery
- + Minimal configuration on the central system

+ Challenges:

- + Source must be trusted to ship logs
- + Delivery can fail silently if network issues occur

Pull Based Collection

- + In a pull model, the central system (e.g., SIEM or collector) requests (pulls) logs from endpoints on a schedule or trigger.
- + How it works:
 - + Collector authenticates with the source system
 - + Uses SSH, WMI, API, or shared folders to retrieve logs
 - + Suitable for systems that can't run agents
- + Examples:
 - + A SIEM pulling logs via WMI from Windows machines
 - + Pulling Apache logs from a web server over SSH/SCP
 - + Using SNMP or APIs to collect logs from devices

Pull Based Collection

+ Benefits:

- + Centralized control of collection
- + Useful for legacy systems or those where agents are impractical

+ Challenges:

- + Higher network load
- + Pull intervals can create latency
- + Requires secure and reliable access to source systems

Agent-Based Collection

- + In this model, a lightweight agent is installed on the host system. It monitors logs in real-time and sends them to a log processor or SIEM.
- + How it works:
 - + Agent reads log files or system APIs
 - + Formats and forwards logs (e.g., JSON)
 - + Can add metadata or tags before shipping
- + Examples:
 - + Filebeat for Linux log files
 - + Winlogbeat for Windows Event Logs
 - + NXLog or Sysmon for advanced Windows telemetry

Agent-Based Collection

+ Benefits:

- + Efficient and real-time
- + Resilient (local buffering, retry mechanisms)
- + Can enrich or filter logs before shipping

+ Challenges:

- + Requires installation and maintenance on every endpoint
- + May raise concerns in tightly controlled environments

Agentless Collection

- + Agentless methods don't require installing anything on the source. Instead, they rely on native protocols and open ports to collect logs.
- + How it works:
 - + Uses WMI, SSH, syslog, or API polling
 - + Typically involves periodic access from a collector
- + Examples:
 - + Collecting logs from Windows using WMI
 - + Gathering logs from network devices via syslog
 - + Querying audit logs from a web app via REST API

Agentless Collection

+ Benefits:

- + No agent deployment overhead
- + Easier in environments with strict software policies

+ Challenges:

- + Less reliable or slower (compared to agent-based)
- + May miss transient logs due to polling delay
- + Complex to secure and scale



Log Shipping

What is Log Shipping?

Log shipping is the process of **transferring collected logs** from the source system to a **centralized log analysis platform or SIEM** (Security Information and Event Management) system such as ELK, Splunk.

Importance of Log shipping:

- + *Enables centralized analysis and correlation.*
- + *Prevents logs from being tampered with on compromised systems.*
- + *Supports real-time alerting and automated detection.*
- + *Ensures long-term storage and compliance.*

Log Collection vs. Log Shipping

Aspect	Log Collection	Log Shipping
Purpose	Gather logs from local/system sources	Send logs to centralized platform
Examples	Auditd, journald, Event Viewer	Filebeat, Winlogbeat, rsyslog, syslog-ng
Destination	Local store or temporary buffer	Logstash, Elasticsearch, Splunk, SIEM
Transport Layer	N/A (local process)	Syslog, Beats, Kafka, HTTP API, TCP/UDP

Common Log Shipping Tools

Tool	OS	Use Case
Filebeat	Linux	Ship flat log files (e.g., <code>/var/log/</code>)
Winlogbeat	Windows	Ship Windows Event Logs
rsyslog/syslog-ng	Linux	Ship logs using syslog format
Logstash	Cross-platform	Parse, enrich, forward logs
NXLog	Windows	Alternative to Winlogbeat, custom logs
Fluentd	Cross-platform	Flexible, plugin-based shipping

Security Best Practices

- + Use TLS encryption to secure logs in transit.
- + Buffer logs locally to avoid data loss during downtime.
- + Enable authentication and signature verification if supported.
- + Separate log collection infrastructure from production systems.



Log Aggregation

Log Aggregation

Log aggregation is the process of collecting and combining log data from multiple sources into a single, centralized system for easier search, correlation, analysis, and monitoring.

It acts as the consolidation phase in the log pipeline and is essential for building situational awareness and enabling effective incident detection and response.

Log Aggregation

Modern environments generate logs from a wide range of systems. These logs are often scattered, use different formats, and are stored in different locations.

Log aggregation solves this by unifying logs, allowing responders to:

- + View and analyze all logs from one place
- + Perform correlation across systems (e.g., login from X followed by alert on Y)
- + Detect patterns and anomalies across hosts/networks
- + Accelerate investigations and reduce detection time



Importance in IR

Importance in IR

Component	Definition	Why It Matters to IR
Log Collection	Gathering logs from endpoints, servers, apps, and devices	Ensures all relevant security events are captured for monitoring and analysis.
Log Shipping	Transmitting logs from source to central location (SIEM, ELK)	Enables reliable, timely delivery of logs for detection and investigation.
Log Aggregation	Centralizing logs from multiple sources for unified analysis	Supports correlation, visibility, and timeline reconstruction across the environment.



Key Terminology

Key Terminology

Term	Definition	Relevance to Incident Response
Log Source	The origin system or application that generates logs (e.g., Windows host, firewall, web server).	Knowing the log source helps responders trace the activity's origin and determine coverage.
Log Collector / Agent	A software component installed on a host that gathers log data (e.g., Filebeat, Winlogbeat).	Ensures log capture at the host level; critical for EDR visibility.
Log Aggregator	A system that receives and consolidates logs from multiple sources (e.g., Logstash, Fluentd).	Centralizes data for easier analysis and correlation.
Log Shipper	Component or agent that forwards collected logs to a centralized platform or SIEM.	Enables secure and reliable delivery of logs for detection and analysis.
Syslog	A widely used protocol for sending log messages, typically over UDP/TCP.	Common in network device logs and often used for infrastructure-level visibility.
Beats Protocol	A lightweight protocol used by Elastic Beats (e.g., Filebeat, Winlogbeat) to send logs.	Preferred for structured, reliable log transmission in ELK-based environments.

Key Terminology

Term	Definition	Relevance to Incident Response
Log Format	The structure in which a log message is recorded (e.g., JSON, Syslog, CEF).	Affects how easily logs can be parsed, indexed, and correlated.
Log Ingestion	The process of receiving and parsing logs into a storage or analysis system.	Crucial for ensuring that logs are stored, searchable, and usable.
Log Normalization	Converting different log formats into a standard structure (fields, labels).	Enables cross-source correlation and detection.
Log Parsing	Extracting structured fields from raw log data (e.g., timestamp, user, action).	Needed to create detection rules, dashboards, and alerts.
Log Retention	The policy for how long logs are stored before being archived or deleted.	Impacts historical investigations, compliance, and forensics.
Buffering	Temporary local storage of logs during network interruptions or load.	Prevents data loss during outages or SIEM downtime.

Key Terminology

Term	Definition	Relevance to Incident Response
Forwarder	Another term for a shipper or lightweight agent that sends logs (e.g., Splunk Universal Forwarder).	Ensures continuous log flow to central analysis tools.
Indexing	The process of organizing and storing logs in a searchable structure (e.g., Elasticsearch index).	Makes it possible to query and correlate logs quickly.
Time Synchronization	Ensuring all systems use the same time (via NTP).	Essential for timeline reconstruction and correlation across systems.
Log Enrichment	Adding context to logs (e.g., geolocation, hostname, user details).	Improves incident triage, detection accuracy, and investigation speed.



Log Shipping Protocols & Formats

Log Shipping Protocols & Formats

In the context of ***centralized logging***, **log shipping protocols** define ***how logs are transmitted*** from a source (e.g., Windows/Linux system) to a destination (e.g., SIEM, ELK stack).

Log formats define ***how the log data is structured*** during ***transport and ingestion***.

These two aspects ensure that logs are reliably transferred, securely delivered, and properly parsed for detection and analysis.

Common Log Shipping Protocols

Protocol	Transport Layer	Use Case	Notes
Syslog	<i>UDP / TCP / TLS</i>	Routers, switches, firewalls, Linux servers	Widely supported; standard for many appliances
Beats Protocol	<i>TCP (TLS optional)</i>	Filebeat, Winlogbeat → Logstash/Elasticsearch	Lightweight, reliable, event streaming format
HTTP(S)	<i>TCP</i>	API-based shipping to cloud/SaaS platforms	Used in Fluentd, custom collectors
Kafka	<i>TCP</i>	High-volume, scalable environments	Often used between Logstash nodes
gRPC / Protobuf	<i>TCP</i>	Efficient binary shipping (Velociraptor, EDRs)	Used in modern EDR or forensic tools

Security Considerations

- + Use TLS/SSL encryption (especially with syslog over TCP, Beats, or HTTP).
- + Authenticate agents/clients with keys or tokens.
- + Implement rate limiting and buffering to prevent data loss.

Common Log Formats

Format	Structure	Example Tools	Purpose
Syslog (RFC 5424)	<PRI> HEADER MSG	rsyslog, syslog-ng	Compact, text-based format for network devices
JSON	Key-value structured text	Filebeat, Fluentd, Logstash	Easy parsing; common in modern log pipelines
Plain Text	Unstructured line output	Apache logs, custom apps	Requires regex parsing; human-readable
CEF (Common Event Format)	key=value pairs	ArcSight, Fortinet, McAfee	Standardized for SIEM ingestion
LEEF (Log Event Extended Format)	Similar to CEF	IBM QRadar	Optimized for QRadar
Protobuf	Binary format	gRPC, Velociraptor	Compact and efficient for EDR/DFIR applications

Protocol + Format Pairings (Examples)

Source	Protocol	Format	Shipped To
Linux Syslog	TCP	Syslog	rsyslog → Logstash
Filebeat (Linux)	Beats	JSON	Logstash → Elasticsearch
Winlogbeat	Beats	JSON	ELK Stack
Firewall	UDP	Syslog	SIEM
Fluentd	HTTP/TCP	JSON	Splunk HEC, Graylog

Why This Matters for IR

- + Choosing the right protocol ensures reliability and security in log delivery.
- + Choosing the right format simplifies parsing, correlation, and analysis.
- + Misconfigured protocols or formats can result in dropped logs, parsing failures, or alerting delays.

A man with glasses and a beard is shown in profile, looking at a computer screen. The screen displays some code or data. The background is dark, and the overall tone is professional and technical.

Centralized Log Collection with Splunk

Lab Overview

In this lab, you'll configure **two servers** to **send log data to a Splunk instance** that resides on SERVER01.

DC01 will need specified Windows event logs, including some AD information sent, and LINUX01 will need specific auth log files sent.

What you'll do in this lab:

- + Collect logs from DC01 (Windows AD) and LINUX01 (Linux auth logs).
- + Ship logs to a centralized Splunk instance (SERVER01).
- + Analyze logs for visibility and incident response.

Lab Overview

You have been tasked with collecting log information from two servers in the organization.

Your director has asked for security event logs and basic AD information to be collected from DC01 and sent to the Splunk server on SERVER01.

In addition, you have been asked to have authorization logs from the Linux server indexed into Splunk as well.

Lab Objectives

- + Collect Security Event Logs and Active Directory logs from DC01.
- + Collect authorization logs (/var/log/auth.log) from LINUX01.
- + Configure all logs to be shipped to and indexed by Splunk on SERVER01.
- + Ensure reliable and secure delivery over port 9997.

Lab Topology

Windows AD
Domain Controller
IP: 172.31.115.100



DC01

Linux Server
IP: 172.31.115.111



LINUX01


In this lab, the log collection is push-based.

The Splunk Universal Forwarder installed on DC01 (Windows AD server) and LINUX01 (Linux server) is actively sending (pushing) logs to the central Splunk instance (SERVER01) over port 9997.



SERVER01

Splunk
IP: 172.31.115.110
Port: 9997



Lab Demo: Centralized Log Collection with Splunk

Introduction to SIEM

What is a SIEM?

A **SIEM** (*Security Information and Event Management*) is a **centralized platform/solution** that provides **real-time visibility, correlation, detection, and alerting** based on log and event data collected across an organization's infrastructure.

As the name suggests, a SIEM combines two core capabilities:

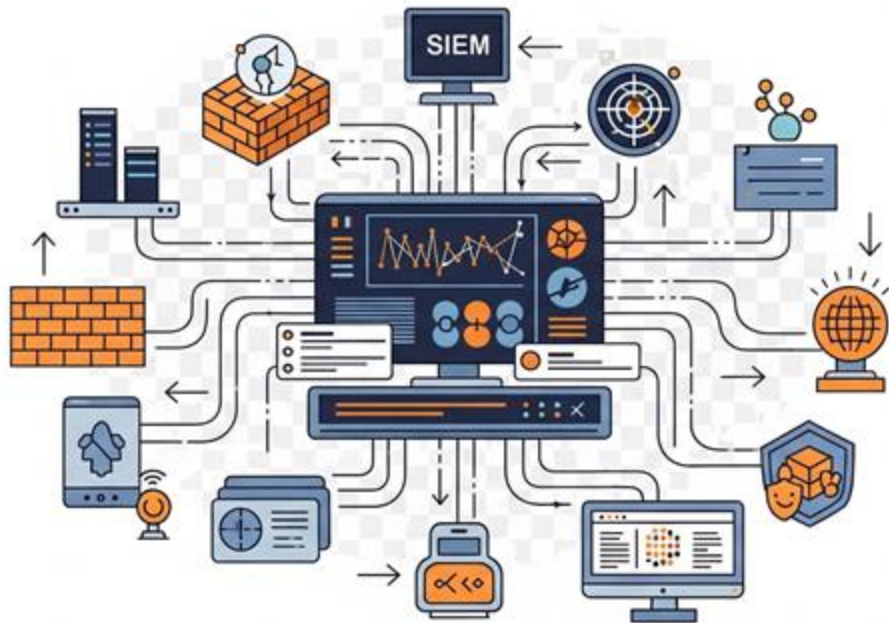
1. **Security Information Management (SIM)** – Historical data analysis and log management.
2. **Security Event Management (SEM)** – Real-time monitoring and alerting.

SIEM = SIM + SEM

What is a SIEM?

SIEM solutions are able to:

- + Analyze current or historical events and log data, perform event correlation and threat monitoring (**Security Event Management**)
- + Retrieve and index log data from disparate sources for analysis and reports (**Security Information Management**)



Popular SIEM Platforms

SIEM Solution	Open Source	Description
Splunk Enterprise Security	No	A leading commercial SIEM platform offering scalable log management, real-time monitoring, and powerful analytics capabilities. Widely used in enterprise environments.
IBM Security QRadar	No	IBM's flagship SIEM providing comprehensive threat detection, log correlation, and automated response. Integrates well with other IBM products.
LogRhythm NextGen SIEM	No	Combines log management, UEBA, and SOAR capabilities. Known for fast detection and response capabilities and strong compliance features.
Microsoft Sentinel	No	A cloud-native SIEM built on Azure. Offers AI-driven analytics, scalable data ingestion, and seamless Microsoft ecosystem integration.
Elastic Security (ELK Stack)	Yes	Built on Elasticsearch, Logstash, and Kibana (ELK), this SIEM offers open-source core components. Some advanced features require a commercial license under Elastic's dual licensing model.
Wazuh	Yes	A fully open-source security platform with log analysis, file integrity monitoring, intrusion detection, and compliance reporting. Highly extensible and community-driven.

Popular SIEM Platforms

SIEM Solution	Open Source	Description
OSSIM (Open Source SIEM)	Yes	Maintained by AT&T Cybersecurity, OSSIM integrates several open-source tools for event correlation and threat detection. A cost-effective, entry-level SIEM.
Security Onion	Yes	A free and open-source Linux distro combining Zeek, Suricata, Snort, Wazuh, and the ELK Stack for enterprise security monitoring and log management.
Graylog	Yes	An open-source log management solution with real-time search and alerting. It can be extended to function as a SIEM with additional features and plugins.
OpenSearch	Yes	A community-driven fork of Elasticsearch and Kibana. Offers open-source search, analytics, and visualization tools—often used as a foundation for custom-built SIEM solutions.

How a SIEM Works

A SIEM follows a log pipeline process with these core stages:

1. Log Collection

- + Gathers logs and events from across the environment:
 - + Windows/Linux servers
 - + Firewalls, IDS/IPS
 - + Applications, endpoints, authentication systems
 - + Cloud and on-prem infrastructure

2. Normalization & Parsing

- + Converts raw logs into a standardized format
 - + Extracts fields like IP addresses, usernames, timestamps



How a SIEM Works

3. Aggregation & Storage

- + Stores logs in indexed databases.
- + Allows for fast searching, reporting, and auditing.

4. Correlation & Detection

- + Analyzes patterns across multiple systems
- + Matches activity against detection rules (e.g., **brute force + suspicious login = alert**)

How a SIEM Works

5. Alerting & Dashboards

- + Raises alerts when suspicious activity is detected.
- + Presents visual dashboards for SOC analysts to monitor.

6. Search & Investigation

- + Enables deep-dive investigation and analysis.
- + Supports threat hunting and timeline reconstruction.

What Is a SIEM Used For?

Use Case	Description
Incident Detection	Identifies threats in real-time (e.g., malware, privilege abuse, C2 activity)
Log Analysis	Centralized analysis of logs from disparate systems
Alerting	Notifies analysts of high-risk events or rule matches
Threat Hunting	Enables proactive searches for indicators of compromise (IOCs)
Compliance & Auditing	Stores logs and proves security control effectiveness (e.g., PCI-DSS, HIPAA)
Forensic Investigations	Reconstructs events post-incident

Why SIEMs Are Critical for Incident Responders

A SIEM is often the primary tool used by incident responders and SOC analysts. It provides:

Centralized Visibility

- + All logs in one place.
- + Cross-system correlations (e.g., login + file access + external connection).

Fast Detection

- + Real-time alerts reduce time to detect (MTTD).
- + Triage queues help prioritize incidents.



Why SIEMs Are Critical for Incident Responders

Contextual Awareness

- + Enriched alerts include context: user, system, location, threat type.
- + Helps responders understand the scope and impact of an attack.

Investigation and Response

- + Timeline views, raw logs, and pivoting across data.
- + **Helps answer the question(s):**
 - + **What happened?**
 - + **Who was affected?**
 - + **What's the root cause?**



Introduction to the ELK Stack



Elasticsearch



Logstash



Kibana



What is the Elastic Stack (ELK)?

The **ELK Stack** is a powerful, *open-source suite of tools* used for *searching, analyzing*, and *visualizing* large volumes of **data**, especially **log data**, in real time.

ELK is an abbreviation of the term "**Elastic Stack**", more specifically, it is an abbreviation of the names of the components that make up the stack:

- + **E** – Elasticsearch
- + **L** – Logstash
- + **K** – Kibana

What is the Elastic Stack (ELK)?

These **tools/components** are developed by **Elastic.co** and are **commonly used together** to process and analyze massive amounts of data efficiently.

The combination of these components is the reason it is referred to as a stack.

However, today the term **Elastic Stack** is more accurate, as it now includes additional tools like **Beats** and the **Elastic Agent**, which **enhance data collection and monitoring capabilities**.

Why was ELK Created?

In modern IT environments, systems generate huge volumes of logs from:

- + Web servers
- + Applications
- + Operating systems
- + Security tools (e.g., firewalls, antivirus)

Managing, searching, and making sense of this data manually is nearly impossible. ELK was created to:

- + Centralize log data from multiple systems.
- + Index and search through data quickly.
- + Visualize trends, patterns, and anomalies.
- + Enable real-time monitoring and alerting.



Core Purpose of the ELK Stack

The best way to understand what the ELK stack is used for is to ***think of ELK as a data pipeline***:

1. **Ingest** logs from various sources (via Logstash or Beats).
2. **Store** and **index** the data efficiently (in Elasticsearch).
3. **Explore** and **visualize** the data (using Kibana).

The flexibility and compatibility between the components that make up the stack makes ELK a go-to solution for:

- + Log management
- + Security monitoring
- + Application performance monitoring (APM)
- + Business intelligence
- + Operational analytics



Where/How is ELK Used?

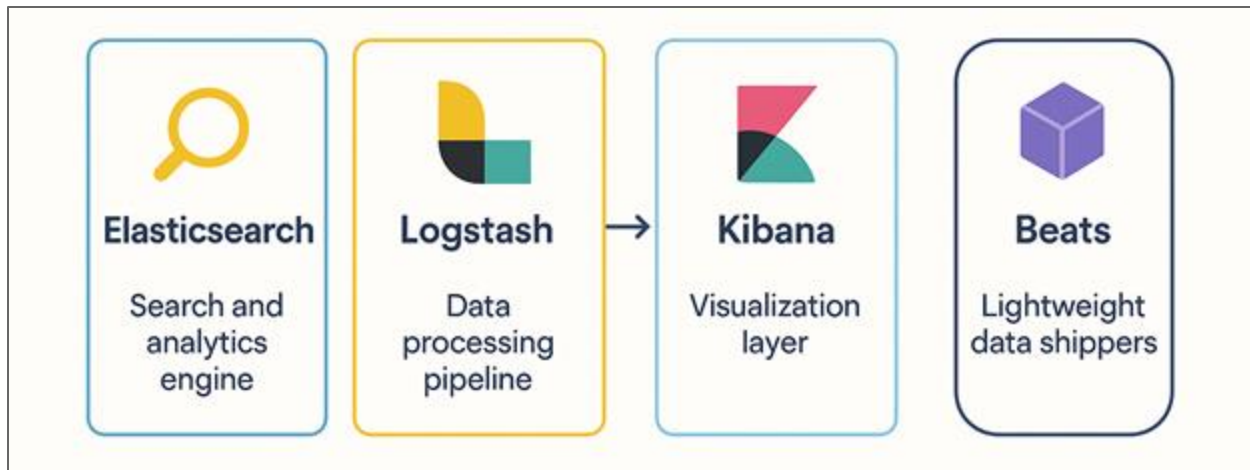
- + Enterprise IT departments for infrastructure monitoring.
- + DevOps teams for troubleshooting applications.
- + Security teams in Security Operations Centers (SOCs) for incident detection and response.
- + Cloud environments for centralized log management.



Core Components of the ELK Stack

Core Components of the ELK Stack

The **ELK Stack** is made up of **three core components**: Elasticsearch, Logstash, and Kibana. Furthermore, there are additional tools like Beats which are often integrated for enhanced log collection.



Core Components of the ELK Stack

Each component plays a specific role in the data processing pipeline. Let us take a closer look at each of them.

1 - Elasticsearch - The Search and Analytics Engine

Elasticsearch is a distributed, RESTful search and analytics engine designed for horizontal scalability and real-time performance.

Key Features:

- + Stores **structured and unstructured data** (logs, metrics, text, etc.).
- + Allows for **near-instantaneous search and filtering**.
- + Built on Apache Lucene, offering powerful full-text search.
- + Highly scalable and fault-tolerant.

Use in Security Operations:

It's used to index and retrieve logs quickly, enabling security analysts to search for Indicators of Compromise (IoCs), track user activity, and pivot across datasets rapidly.



Core Components of the ELK Stack

2 - Logstash - The Data Processing Pipeline

Logstash is a ***server-side data ingestion tool*** that collects, transforms, and sends data to Elasticsearch.

Key Features:

- + Supports multiple input sources: syslog, files, APIs, etc.
- + Can parse, filter, and enrich log data using grok patterns or conditionals.
- + Output data to Elasticsearch, files, or even message queues like Kafka.

Use in Security Operations:

It helps normalize logs from different sources, extract important fields (like usernames, IPs, process names), and ensure clean, searchable data in Elasticsearch.

Core Components of the ELK Stack

3 - Kibana - The Visualization Layer

Kibana is the **frontend interface** that allows users to visualize data stored in Elasticsearch.

Key Features:

- + Create interactive dashboards and data visualizations (charts, graphs, tables).
- + Use KQL (Kibana Query Language) to filter and search data.
- + Supports real-time exploration of data.
- + Enables alerting, detection rules, and even timeline-based investigations in Elastic Security.

Use in Security Operations:

- Monitor real-time dashboards for suspicious activity.
- Visualize trends in log data (e.g., spike in failed logins).
- Drill down into specific events for investigation.

Core Components of the ELK Stack

4 - Beats (Add-on) - Lightweight Data Shippers

Beats are *lightweight agents* installed on endpoints to collect specific types of data and forward it to Logstash or Elasticsearch.

Popular/Common Beats:

- + **Filebeat:** Reads log files (e.g., Apache logs, system logs)
- + **Winlogbeat:** Collects Windows event logs
- + **Packetbeat:** Captures network traffic
- + **Metricbeat:** Gathers system metrics (CPU, RAM, etc.)

Use in Security Operations:

Beats are often used on endpoints and servers to ensure continuous and structured log forwarding from a wide range of sources.



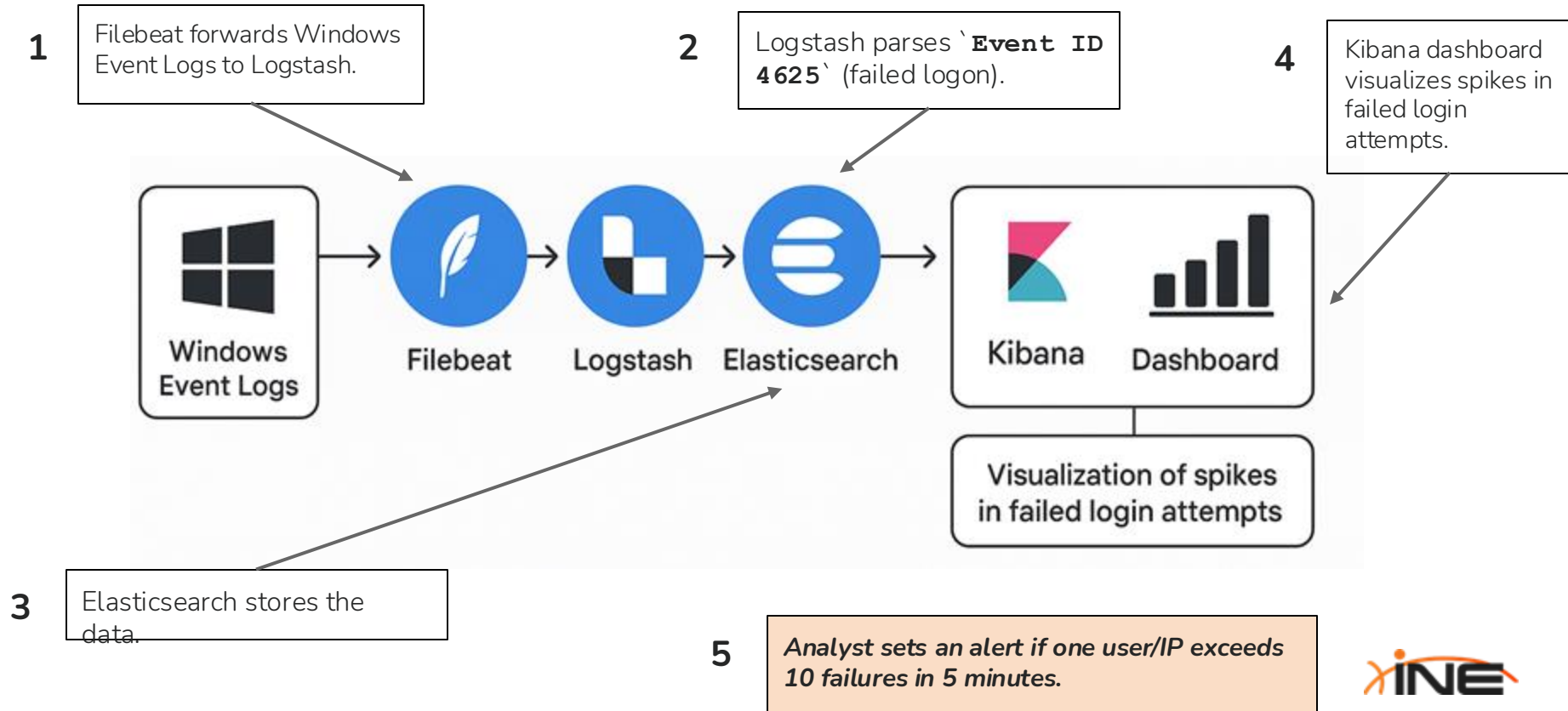
ELK Stack in Security Operations

ELK Stack in Security Operations

The **ELK Stack** plays a central role in modern SOC's. Its ability to ***ingest, store, analyze, and visualize log data*** makes it indispensable for monitoring and responding to cyber threats.

Use Case	Explanation
Threat Detection	<ul style="list-style-type: none">• ELK enables the detection of malicious behaviors by parsing and correlating logs across the environment.• Analysts can create detection rules (custom or using Elastic Security) to flag suspicious activities.
Incident Response	<ul style="list-style-type: none">• When an alert is triggered, responders can pivot quickly to search logs in Elasticsearch via Kibana.• Allows for deep-dive investigations, tracing the origin, method, and impact of attacks.
Security Monitoring	<ul style="list-style-type: none">• Real-time dashboards track critical security metrics like failed logins, abnormal outbound traffic, or changes in file integrity.• Alert thresholds can be set to escalate high-risk events automatically.
Threat Hunting	<ul style="list-style-type: none">• Analysts use Kibana's search and filter capabilities to proactively look for anomalies or Indicators of Compromise (IoCs).• Custom queries help identify patterns that don't match known signatures.

Example: Detecting Brute Force Login Attempts



Why ELK is Important for IR

The **ELK Stack** isn't just a tool for log management; it's a ***critical enabler*** for ***detecting, investigating, and responding to security incidents***. Here's why every incident responder should understand and master it.

For incident responders, mastering ELK means:

- + Faster detection and triage.
- + Better context during investigations.
- + More efficient threat hunting and reporting.
- + Reduced dwell time of adversaries in the network.

References

References & Further Reading

- + Elastic Official Documentation: <https://www.elastic.co/guide>
- + Elastic Security (SIEM on the ELK Stack):
<https://www.elastic.co/security>



Effectively Using ELK

Lab Scenario

The organization you work for is evaluating a customized ELK stack as a SIEM solution to enhance its intrusion detection capabilities.

The SOC manager tasked you with getting familiar with the ELK stack and its detection capabilities.

He also tasked you with translating common attacker behavior into ELK searches.

Lab Objective(s)

The learning objective of this lab, is to get familiar with the architecture and detection capabilities of the ELK Stack.

- + Task 1: Add any fields you see fit to enhance your understanding of the data
- + Task 2: Create an actionable visualization
- + **Task 3: Create a search to identify files that are named like system processes**
- + **Task 4: Create a search to identify suspicious services interacting with an executable from the Windows folder**
- + Task 5: Create a search to identify suspicious code injection
- + Task 6: Create a search to identify possible privilege escalation via weak service permissions
- + Task 7: Create a search to identify possible Windows session hijacking
- + **Task 8: Create a search to identify the whoami command being executed with System privileges**
- + Task 9: Create a search to identify LSASS loading a library not signed by Microsoft



Demo: Effectively Using ELK



Introduction to Splunk for Security Operations

What is Splunk?

Splunk is a *Security Information and Event Management (SIEM)* tool and *data analytics platform*.

It *ingests data from a variety of sources*; such as servers, network devices, applications, and security tools—and makes that data *searchable and actionable*.

Think of Splunk as a giant search engine for your machine data.



What is Splunk?

Splunk handles the following types of data:

- + Log files from operating systems.
- + Alerts and telemetry from firewalls, IDS/IPS, antivirus.
- + Application logs, including web servers, databases, and APIs.
- + Cloud infrastructure logs from AWS, Azure, GCP.

Basically, anything that generates logs or events can be ingested by Splunk.

Splunk Use Cases

1. IT Operations Monitoring

- + Troubleshooting performance issues
- + Monitoring uptime and application health

2. Security Monitoring

- + Detecting and investigating threats
- + Real-time alerting on suspicious behavior

3. Business Intelligence

- + Analyzing user behavior
- + Tracking KPIs using machine data

4. Compliance

- + Ensuring log retention and audit trails
- + Automating compliance reporting

How Splunk Works

1 - Data Ingestion

First, Splunk collects data from various sources. These sources include:

- + Servers (system logs)
- + Network devices (firewalls, routers)
- + Applications (web, database logs)
- + Cloud services (AWS CloudTrail, Azure Monitor)
- + Security tools (AV, EDR, IDS/IPS)

Forwarders:

- + Universal Forwarder (UF): Lightweight, for data forwarding only.
- + Heavy Forwarder (HF): Can parse and index data before sending it to the indexer.



How Splunk Works

2 - Indexing

Once data is received, Splunk parses it and stores it in indexes.

- + Parsing: Splunk breaks down raw data into events, applies time stamps, and extracts key-value pairs.
- + Indexing: Parsed data is stored in time-series indexes for fast retrieval.

Example:

- + A single firewall log becomes a searchable event with timestamp, source IP, destination IP, action taken, etc.

How Splunk Works

3 - Search & Visualization

- + SPL (Search Processing Language):
 - + Custom queries to filter, correlate, and analyze data.
 - + Examples: `index=firewall action=blocked` or `stats count by src_ip`
- + Dashboards:
 - + Real-time visuals using graphs, charts, and tables.
 - + Useful for monitoring system health, detecting anomalies, and reporting.
- + Alerts:
 - + Set thresholds to trigger alerts (e.g., "Alert when failed logins > 10 in 5 minutes").

Different Versions of Splunk

Splunk offers several editions tailored to different use cases, organization sizes, and deployment preferences.

1. Splunk Free

A limited version of Splunk Enterprise.

Key Features:

- + Single user access.
- + Up to 500MB of data indexing per day.
- + No user authentication or role-based access.
- + Best For: Small-scale testing, personal learning.



Different Versions of Splunk

2. Splunk Enterprise

The full-featured, on-premises solution.

Key Features:

- + Unlimited data indexing (based on license).
- + Supports clustering, scaling, and distributed search.
- + Role-based access control.
- + Supports all Splunk apps and premium solutions.

Best option for medium to large enterprises with complex environments.



Splunk Enterprise Security (ES)

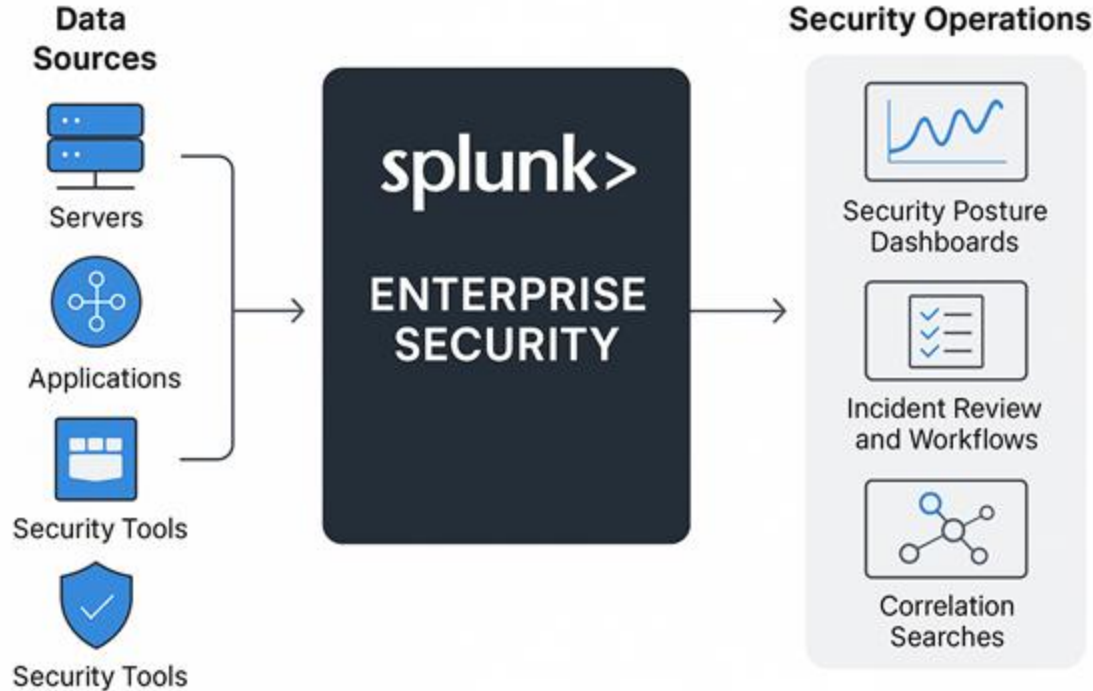
Splunk Enterprise Security (ES)

Splunk Enterprise Security, or **ES**, is a premium app that ***transforms Splunk into a full-featured SIEM platform.***

It's built specifically for security operations teams to detect, investigate, and respond to threats effectively.



Splunk Enterprise Security (ES)



Splunk Enterprise Security (ES) Features

1. Security Posture Dashboards

- + Provides a high-level overview of your organization's security status.
- + Includes panels for failed logins, malware detections, data exfiltration attempts, and more.
- + Visual, real-time view of threats across different domains.

2. Correlation Searches

- + Detects suspicious patterns by correlating multiple log sources and events.
- + Comes with prebuilt use cases (e.g., brute force, lateral movement, data exfiltration).
- + Analysts can create custom rules tailored to their environment.



Splunk Enterprise Security (ES) Features

3. Risk-Based Alerting (RBA)

- + Assigns risk scores to assets and users based on activity.
- + Reduces alert fatigue by prioritizing high-risk behavior.
- + Enables context-aware investigations.

3. Threat Intelligence Framework

- + Ingests and normalizes threat feeds (STIX/TAXII, MISP, VirusTotal, etc.).
- + Matches incoming data against known IoCs (IPs, domains, hashes).
- + Automates threat enrichment and triage.

Splunk Enterprise Security (ES) Features

4. Incident Review & Workflows

- + Central dashboard to triage, assign, and resolve security incidents.
- + Analysts can investigate events, add comments, and escalate cases.
- + Streamlines security operations and enables better team collaboration.

5. MITRE ATT&CK Integration

- + Maps detections and alerts to ATT&CK tactics and techniques.
- + Helps analysts understand attack progression and gaps in coverage.



Detection Rules for Incident Responders

Detection Engineering

Detection Engineering is the ***discipline of designing, developing, testing, and refining detection logic*** that identifies suspicious or malicious behavior in an environment.

In the context of Incident Response (IR), it is the ***strategic and tactical layer that bridges the gap*** between threat intelligence, real-world attacks, and actionable security alerts.

Detection Engineering ensures that incident responders are equipped with timely, relevant, and high-fidelity alerts, enabling them to:

- + Detect adversary behavior early in the attack chain.
- + Minimize dwell time.
- + Accelerate triage and containment efforts.



The Role of Detection Engineering in IR

1. Pre-Incident (Preparation)

- + Build and deploy detection rules based on known threat actor **TTPs** (e.g., MITRE ATT&CK).
- + Simulate attacks and refine rules through threat emulation or purple teaming.

2. During Incident (Detection & Analysis)

- + Alert logic triggers based on specific attacker behaviors.
- + Incident responders rely on these detections to guide investigations, validate alerts, and scope compromise.

3. Post-Incident (Lessons Learned)

- + Responders analyze gaps in detection and feed insights back into improving rule logic and alert coverage.

Why It Matters?

Detection Engineering is not just a backend task; *it's a collaborative function* that enables proactive, responsive, and accurate incident handling.

By understanding detection engineering, incident responders can:

- + Familiarise themselves with how alerts are built and what they truly represent.
- + Can contribute to detection tuning and validation.
- + Help prioritize and evolve detection strategies based on real attack scenarios.

Core Detection Engineering Skills

1. Writing and Testing Detection Rules

- + Craft detection logic using:
 - + SPL (Splunk), KQL (Sentinel), YARA, or vendor-specific DSLs.
 - + Validate and test detection rules using real or emulated data (e.g., Atomic Red Team).

2. Alert Triage and False Positive Reduction

- + Analyze alert quality: relevance, context, and noise.
- + Tune rules by adjusting filters, thresholds, or enrichment sources.

3. Incident-to-Detection Feedback

- + Translate incident learnings into new detection content.
- + Create retrospective queries to detect undetected past activity.

4. Collaboration with Threat Hunters and Engineers

- + Communicate clearly with detection engineers to propose rule improvements or request new logic.
- + Share TTPs and IR findings in actionable formats.

Types of Detection Rules

Detection Rules

In **Incident Response (IR) and Security Operations**, Detection Rules are ***logic-based instructions*** used by security platforms (e.g., SIEM, EDR, NDR) to ***identify suspicious or malicious activity*** across an environment. These rules help in automating the monitoring, alerting, and sometimes even initial triage of security events.

These rules are predefined logic-based expressions used to identify suspicious behavior, malicious activities, or deviations from expected baselines within an IT environment.

Detection rules enable automated monitoring and real-time alerting, allowing security teams to swiftly identify and triage potential threats. Whether deployed in a SIEM platform like Splunk, an EDR system, or a Network Detection and Response (NDR) solution, detection rules act as the first line of defense in threat detection.



Detection Rules

For incident responders, these rules are **critical**; *they provide visibility* into the "**what**", "**when**", and "**where**" of an incident, helping analysts **prioritize and investigate alerts** with context-rich data.

Well-crafted detection rules can mean the difference between detecting an adversary early or only realizing a breach has occurred after the fact.

Types of Detection Rules

1 - Signature-Based Detection

- + Matches known indicators (e.g., file hashes, IPs, strings) against observed data.
- + Used to detect known malware, exploits, or specific threat actor TTPs.

Example (YARA Rule):

```
rule Suspicious_Mimikatz
{
    strings:
        $a = "mimikatz"
    condition:
        $a
}
```

Types of Detection Rules

2 - Behavior-Based Detection

- + Behavior-Based Detection works by monitoring sequences of actions or unusual behavior patterns that deviate from baselines.
- + It is typically used to detect activity like: credential dumping, privilege escalation, or lateral movement.

Example (EDR Rule):

```
if process.name == "lsass.exe" and parent.name !=  
"winlogon.exe" then alert
```

Types of Detection Rules

3 - Heuristic/Anomaly-Based Detection

- + Uses statistical methods or machine learning to detect anomalies from normal behavior.
- + Used to detect outliers in user behavior, network traffic, or file access patterns.

Example:

Alert if data transfer exceeds 10GB/hour and user is not in IT group

Types of Detection Rules

4 - Threshold-Based Detection

- + Threshold-Based detection works by triggering alerts when a predefined threshold is exceeded.
- + Use Case: DDoS, brute-force attempts, or mass file deletion.

Example (SIEM Rule in KQL):

```
FailedLogons  
| where ResultType == "0x18"  
| summarize count() by AccountName, bin(TimeGenerated, 5m)  
| where count_ > 20
```

Types of Detection Rules

5 - TTP-Based Detection (MITRE ATT&CK Aligned)

- + Works by matching tactics, techniques, and procedures (TTPs) used by (attributed to) adversaries based on frameworks like MITRE ATT&CK.
- + Use Case: Coverage mapping for ATT&CK techniques.

Example (Splunk SPL):

```
index=wineventlog EventCode=4688  
| where New_Process_Name="*powershell.exe" and  
CommandLine="*EncodedCommand"
```

Types of Detection Rules

6 - Correlation Rules

- + Works by combining multiple detections/events over time to detect complex attacks.
- + Use Case: Multi-step attacks like phishing leading to credential theft and then lateral movement.

Example:

- Rule 1: Detect phishing email delivery.
- Rule 2: Detect unusual VPN login after email.
- ***Correlation Rule: Trigger if both occur within 30 minutes.***



Detection Rules vs. Alerts

Detection Rules vs. Alerts

In platforms like Splunk and Elastic, a ***detection rule defines the logic or pattern used to identify suspicious or malicious activity***, such as matching specific event IDs, process names, or behavioral indicators. It essentially describes what to look for.

An alert, on the other hand, is the response mechanism triggered when the conditions of a detection rule are met. It specifies what to do; such as sending an email, creating a ticket, or logging an incident.

While the detection rule is the analytical component, the alert operationalizes it by enabling timely notification and response, making both elements crucial for effective threat monitoring and incident response.

Detection Rules vs. Alerts

Concept	Detection Rule	Alert
Definition	A logic-based rule that defines what suspicious activity looks like	The action triggered when a detection rule condition is met
Function	Describes <i>what to detect</i>	Describes <i>what to do when it's detected</i>
Components	Query or pattern + condition (e.g., "EventCode=4732")	Notification method + threshold + response actions
Example	"Detect accounts added to Admin group"	"Send an email if above rule triggers"
Used In	Detection logic, threat hunting, SOC triage	Incident notification, automation, escalation

Summary

Detection Rule = What to look for.

Alert = What to do when it's found.



From Logs to Incidents: The Triage Process

Triage & Escalation - Introduction

Now that you've mastered the building blocks of detection; ***how logs are generated, collected, shipped, normalized, and hunted*** within a SIEM, plus the ***fundamentals of writing detection rules***, the next logical milestone is understanding **what happens once detection rules are triggered**.

In this video ***we'll shift focus from technology to workflow***: you'll learn how a raw log that matches a rule is transformed into an alert, how Tier-1 analysts triage and enrich that alert, and how a fully scoped incident is packaged and handed off to the Incident Responder with the evidence and context needed for rapid action.

Grasping this triage pipeline bridges the gap between detection engineering and incident response, showing you how the pieces you've already learned come together to protect the organisation in real time.



Triage & Escalation - Introduction

In this section of the course; **Triage & Escalation**, you will learn *how raw, machine-generated alerts* are transformed into *actionable, well-scoped incidents (cases, tickets)* that incident responders can immediately analyze/investigate.

By the end of this section of the course, you should have an understanding of:

1. The end-to-end process/workflow of how logs become alerts and how alerts become incidents.
2. The systematic triage process used to classify, enrich, and score alerts.
3. How incidents are escalated to incident responders in the form of evidence-rich tickets/cases that satisfy audit, SLA, and hand-off requirements.

Triage & Escalation - Introduction

Understanding this workflow/process is essential because an IR program's speed and accuracy hinge on ***disciplined triage***: efficient filtering prevents analyst fatigue, accelerates containment, and ensures critical threats are neither overlooked nor delayed by noise.

Furthermore, as an Incident Responder, it is very important to understand the process (flow) of **how a log becomes an alert**, and, more importantly, how an alert is investigated, triaged and classified as an incident.



The Triage Process

What “Triage” Means in a SOC

In cybersecurity operations, triage is ***the rapid, methodical filtering of raw security signals (logs)*** so that ***only real*** and ***high-impact threats*** reach an Incident Responder.

It plays the same role as medical triage: “decide who needs a doctor now, who can wait, and who is completely fine.”

In the context of IR, Triage is the ***process of analyzing and categorizing security alerts or events*** to determine whether they indicate a potential security incident, a false positive, or a benign activity.

The goal is to quickly assess the relevance and severity of an event and decide on the appropriate next steps.



How a Log Becomes an Incident - The Process

- + **Log Generation** – Endpoint, network device, cloud service generates raw logs.
- + **Ingestion & Normalization** – Logs are shipped (Syslog, Beats etc.) to SIEM; parsed into common schema (vendor, src_ip, process_name etc.).
- + **Correlation / Detection Rule** – SIEM evaluates normalized records against detection rules, signatures, behavioral analytics and threat-intel lists/databases.
- + **Event Creation** – A single rule match is stamped/categorized as an event with ***rule ID + metadata***.
- + **Alert Formulation** – The event is scored, enriched, and promoted to an alert that lands in the Tier 1 queue.
- + **Tier 1 Triage** – The analyst dismisses or escalates; escalation converts alert → case (investigation ticket).
- + **Tier 2 Investigation** – The Incident Responder takes ownership, performs full incident handling in the form of analysis, containment, eradication, recovery, and post-mortem.

How a Log Becomes an Incident - The Data Pipeline

1 - Log → Event

- + Devices, endpoints, and cloud services generate logs (e.g., a Windows Sysmon entry).
- + A SIEM or log-pipeline tool parses each log into a structured event (fields such as host, process_name, user, timestamp).

2 - Event → Alert

- + Detection logic/rules (signatures, behavioral rules, threat-intel matches, ML models) evaluates events.
- + A rule hit/match is stamped as an alert and placed in the SOC queue with initial metadata (rule ID, asset, time).

How a Log Becomes an Incident - The Data Pipeline

3 - Alert → Triage

Tier 1 analysts, or automated SOAR playbooks work the queue:

1. **Intake & Deduplication** – Collapse identical alerts, suppress known benign scanners.
2. **Initial Classification** – Tag alert with category (e.g., Phishing, Malware, Privilege Escalation) or MITRE ATT&CK TTP.
3. **Context Enrichment (≤ 30 s)** – Pull/Identify asset criticality, user privilege, IP reputation, recent alert history.
4. **Severity Scoring** – Apply a formula (**Impact** × **Likelihood**) to label **Critical**, **High**, **Medium**, or **Low**.
5. **Disposition Decision**
 - a. Close / Suppress – Confirmed false positive or policy-accepted activity.
 - b. Contain & Resolve – Quick scripted fix (e.g., isolate workstation) with no escalation needed.
6. **Escalate** – anything uncertain, high-severity, or requiring deeper forensics.

How a Log Becomes an Incident - The Data Pipeline

4 - Escalated Alert → Incident Case

- + ***For escalations***, the SOAR/SIEM opens an incident ticket (ServiceNow, TheHive, JIRA, etc.).
- + The ticket is populated with: ***enriched logs, artifact links, severity, containment actions already taken***, and ***SLA clocks***.
- + ***Ownership is reassigned from Tier 1 (Alert Monitor) to Tier 2 (Incident Responder)***.

The Triage Process - End-to-End Workflow

The table in the next set of slides will provide you with an ***end-to-end view*** of the triage workflow/process complete with the ***actions taken*** in each phase/step, ***ownership*** of the actions taken in each respective phase/step, and the ***deliverables/outputs*** of each phase/step of the Triage Process.

The workflow represented in the table is platform/tool-agnostic but maps cleanly to Splunk Enterprise Security, Elastic Security, Microsoft Sentinel, QRadar, or a home-grown (custom) stack.

The Triage Process: From Alert to Case Creation

Phase	Key Actions	Typical Owner	Deliverables / Artifacts
1. Alert Intake & Deduplication	<ul style="list-style-type: none">• Ingest alerts from SIEM rule triggers, EDR/IDS sensors, cloud security tools.• De-duplicate identical alerts (hash or alert-ID matching).• Suppress known benign signatures (noise filters).	SOC Tier-1 (automated pre-processing if possible)	Clean “working queue” with unique alert IDs and timestamps
2. Initial Classification	<ul style="list-style-type: none">• Identify alert type (malware, phishing, brute-force).• Map to MITRE TTP (if available).• Tag with asset owner / business unit.	Tier-1	Classification label (taxonomy tag)
3. Context Enrichment	<p>Pull quick, automated context:</p> <ul style="list-style-type: none">• Asset criticality (CMDB/API).• User identity & role (IdP/HR feed).• Geolocation & reputation for IPs / domains (TI feeds).• Previous alert history on entity.• Open-source threat intel (STIX/TAXII, VirusTotal).	Tier-1 assisted by SOAR	Enriched alert metadata blocks

The Triage Process: From Alert to Case Creation

Phase	Key Actions	Typical Owner	Deliverables / Artifacts
4. Severity Scoring / Prioritization	<p>Apply scoring formula (e.g., Impact × Likelihood or CVSS-like matrix):</p> <ul style="list-style-type: none">• External vs. internal asset?• High-value system?• Privileged account involved?• Detect stage in kill chain.• Confidence in rule (FP rate).	Tier-1 (auto, then human sanity check)	Numeric / categorical severity (Critical, High, Medium, Low)
5. Decision: Dismiss, Contain, or Escalate	<ul style="list-style-type: none">• Dismiss / Close – false positive or benign.• Contain & Resolve – straightforward fix (e.g., user clicked benign link that was block-paged).• Escalate – requires deeper analysis → moves to Tier-2 or IR team.	Tier-1	Disposition tag + notes
6. Creation of the Investigation Case	<p>For escalated alerts:</p> <ul style="list-style-type: none">• Open case in ticketing/IR platform (ServiceNow SecOps, JIRA, TheHive, etc.).• Auto-link artifacts (logs, PCAPs, malware samples).• Populate playbook template (tasks, SLAs, owners).• Notify on-call IR lead / Tier-2.	SOAR or Tier-1	Case ID, ownership, SLA clock starts

The Triage Process: From Alert to Case Creation

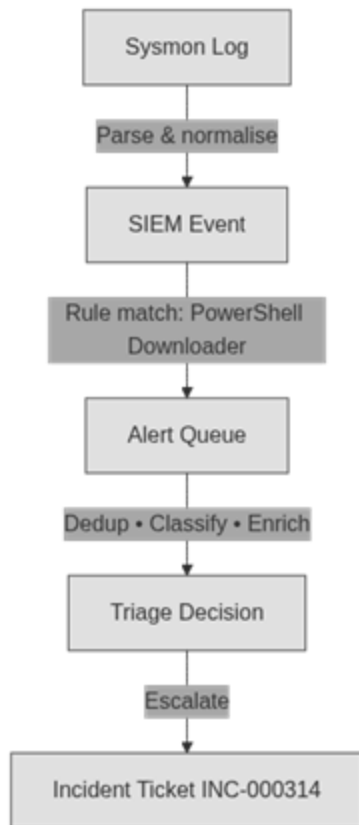
Phase	Key Actions	Typical Owner	Deliverables / Artifacts
7. Documentation & Handoff	<ul style="list-style-type: none">• Summarize triage findings (why escalated, severity rationale).• Attach enrichment evidence and initial queries.• Record containment already performed (e.g., blocked hash via EDR).• Hand off verbally or via workflow comment to Tier-2 analyst for deep dive.	Tier-1 → Tier-2 / IR	Fully populated case record ready for analysis

Why Each Step Matters to IR

Triage Checkpoint	Value for the Responder
Deduplication	Prevents duplicate work and alert fatigue.
Classification & Enrichment	Gives immediate context; critical server vs. test VM; privileged admin vs. guest.
Severity Scoring	Allows responders to triage their own workload, hitting the highest-risk incidents first.
Evidence-Rich Ticket	Minimises “re-triage”, letting Tier 2 jump straight to validation, analysis, scoping, and containment.

Examples

Scenario 1 – Malicious PowerShell on a Windows Workstation



1 - Log → Event

- + **Sysmon** on **WIN10-HR01** records **Event 1 (Process Create)** when a hidden PowerShell spawns with a **DownloadString** command-line argument.

2 - Event → Alert

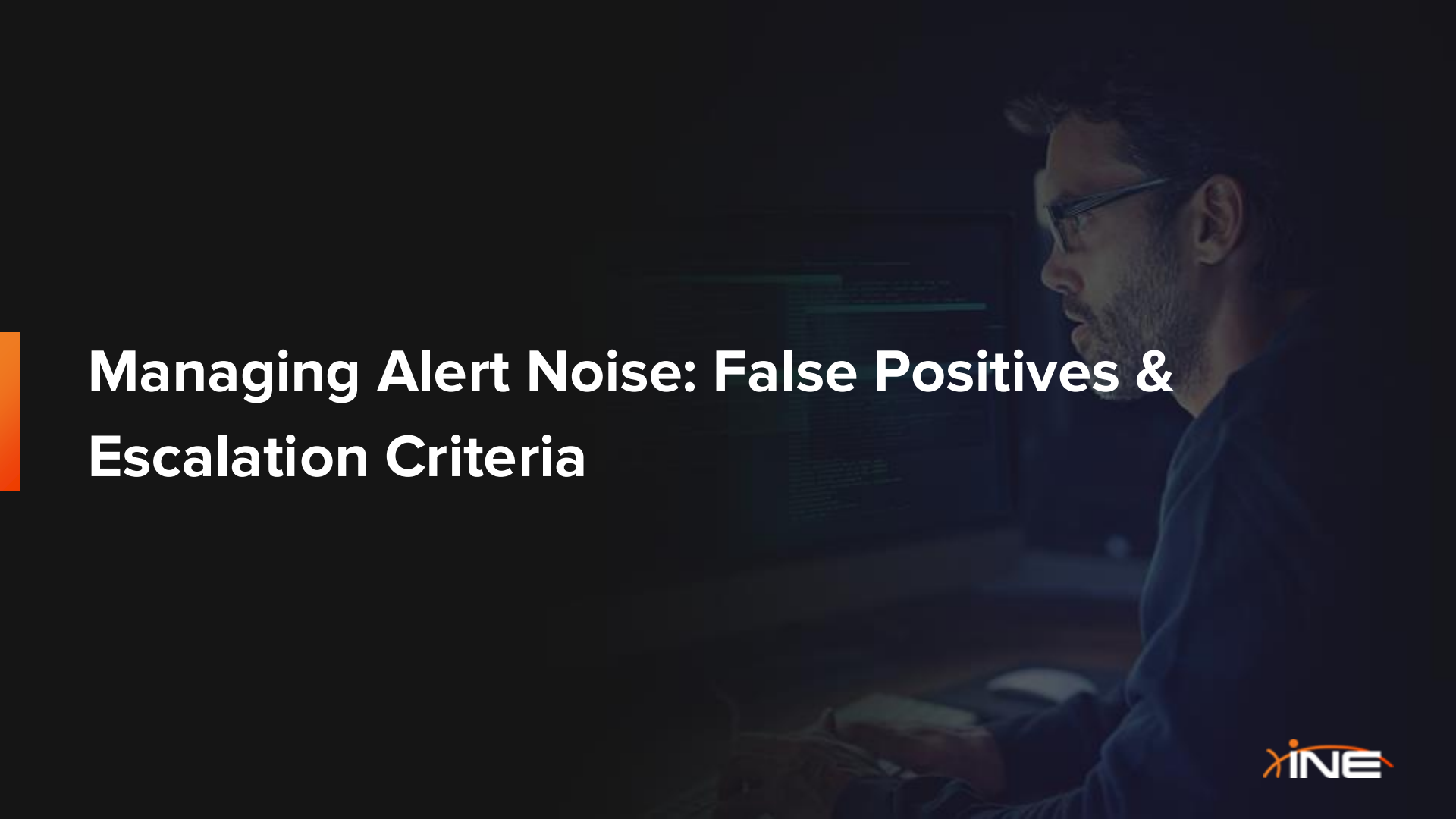
- + A Splunk correlation search (**process_name=powershell.exe AND command_line=*DownloadString***) fires and labels the match High severity.

3 - Alert → Triage

- + Tier-1 analyst enriches with asset criticality (business laptop), adds VirusTotal hash result, and sees no prior alerts on host → escalates.

4 - Case Creation

- + SOAR opens **ServiceNow** IR ticket **INC-000314**; all logs + hash attached; Tier-2 responder paged.



Managing Alert Noise: False Positives & Escalation Criteria

Alert Noise

Alert noise is the flood of *security alerts* that are *irrelevant, duplicate, low-value*, or *plainly false positives* in a SOC's monitoring queue.

It arises when *detection rules are too broad*, threat-intel feeds match benign traffic, or the *same event* reaches the SIEM multiple times.

The result is a queue packed with “chatter” that *obscures genuine threats*, drains analyst attention, and inflates metrics such as **Mean Time to Triage (MTTT)**.

Effective noise reduction through rule tuning, deduplication, suppression filters, and business-context enrichment is therefore critical to keep analysts focused on alerts that truly warrant investigation.



Alert Accuracy - Key Terms

- + **False Positive (FP):** Alert for activity that is benign or expected.
 - + **True Positive (TP):** Alert for genuine malicious or policy-violating activity.
 - + **True Negative (TN):** No alert because nothing bad occurred.
 - + **False Negative (FN):** Malicious activity happened but no alert fired (silent failure).
-
- + **Escalation Criteria:** Pre-defined conditions that must be met before Tier 1 promotes an alert to a Tier 2 incident case (e.g., Critical severity or High severity on crown-jewel asset).

Alert Accuracy - Terminology Explained

Term	What It Means	Simple Example	Why It Matters
True Positive (TP)	The detection system raises an alert and the activity is malicious .	A SIEM rule is triggered by a PowerShell DownloadString command, and the workstation is indeed infected with malware.	Confirms the rule works as intended; responder must act .
False Positive (FP)	The system raises an alert but the activity is benign .	The same rule fires when an admin legitimately uses PowerShell to download a corporate script.	Wastes analyst time; too many FPs create “alert-fatigue.”
True Negative (TN)	The system does not alert and nothing malicious happens .	Overnight backups run on servers; no security rule fires—and that’s fine, it’s expected traffic.	Indicates normal operations are correctly ignored.
False Negative (FN)	The system does not alert even though malicious activity occurs .	An attacker uses a novel LOLBin (living-off-the-land binary) that existing rules don’t cover, so no alert is generated while data is exfiltrated.	The most dangerous case; real threats slip through undetected; rules must be updated.

Escalation Criteria

Escalation Criteria - When Does an Alert Become an Incident?

Criterion Category	Typical Questions Asked	Practical Checks & Examples
Severity / Risk Score	Does the alert carry a <i>Critical</i> or <i>High</i> score after triage?	<ul style="list-style-type: none">• CVSS ≥ 9 exploited on internet-facing host.• SIEM rule confidence ≥ 90 %.
Asset Criticality	Is the target a <i>crown-jewel</i> system or privileged account?	Domain controller, payment-card environment, executive mailbox.
Kill-Chain Stage	Has the activity moved beyond <i>Recon / Initial Access</i> into <i>C2</i> , <i>Lateral Movement</i> , or <i>Exfiltration</i> ?	Beacon on port 443 to strange ASN → always escalate.
Regulatory Impact	Could the event trigger breach-notification laws?	GDPR Article 33 personal-data exposure, PCI DSS scope, HIPAA ePHI.
Rule Confidence & FP History	How often has this rule produced false positives?	“PowerShell Downloader” rule with < 1 % FP ⇒ auto-escalate on first hit.
Containment Feasibility	Can Tier 1 fully remediate within playbook?	If “yes” (e.g., block hash, isolate host) and risk is low, no escalation.
Business Hours & SLA	Does policy demand human review for Critical alerts 24 × 7?	After-hours <i>Critical</i> automatically escalates to on-call IR lead.



Indicators of Compromise (IOCs)

What are Indicators of Compromise (IOCs)?

An **Indicator of Compromise (IOC/IoC)** is a piece of *forensic data* that **suggests/indicates** that an *endpoint or network may have been breached*. Generally speaking IOCs are indicative of potentially malicious activity on a system or network.

IOCs serve as clues that security professionals **use to detect, investigate, and respond** to *malicious activity or cyber threats*, such as data breaches, ransomware attacks or insider threats.

They help answer questions like:

- + **Has this system been compromised?**
- + **What type of attack took place?**
- + **What other systems might be affected?**

What are Indicators of Compromise (IOCs)?

Detecting IOCs is reactive, in that; by the time an ***indicator is discovered***, it's often a sign that a compromise has already occurred.

However, if detection happens while the threat is still unfolding, ***timely recognition of an IOC*** can significantly mitigate damage by enabling faster containment and disruption of the attacker's operations.

With the advancement of threat actors, ***the ability to reliably detect IOCs has become increasingly challenging***.

Common indicators like: MD5 file hashes, command-and-control (C2) domains, static IP addresses, registry modifications, and known filenames are frequently rotated or obfuscated, diminishing their lifespan and complicating detection efforts.



What are Indicators of Compromise (IOCs)?

Just like ***breadcrumbs*** in a forest can ***lead you back to where someone has been (or where you came from)***, IOCs are the **digital traces that attackers leave behind during or after a compromise/breach.**

These breadcrumbs: IP addresses, file hashes, URLs, registry changes don't always tell the full story, ***but when followed, they can lead defenders to the source, impact, and scope of a breach.***

The Role of IOCs in Cybersecurity

IOCs are the digital footprints of threat actors. Their role in cybersecurity can be grouped into the following:

1 - Detection

- + IOCs enable automated tools (SIEMs, EDRs, firewalls) to flag or block known threats.
- + Example: If an endpoint connects to a known malicious IP, the security tool generates an alert.

2 - Investigation & Triage

- + Analysts use IOCs to triage alerts, determining if they are true positives.
- + Example: A file hash found in an alert is verified on VirusTotal and found to be a known RAT (Remote Access Trojan).

3 - Threat Hunting & Intelligence

- + Advanced teams use IOCs to search for hidden threats and pivot to discover related activity.
- + IOC enrichment can tie malicious activity to threat actors or campaigns.

4 - Incident Response & Remediation

- + During an incident, IOCs are crucial to scoping the extent of the compromise.
- + They also assist in blocking the threat, via firewall rules, AV signatures, etc.

Types of IOCs

1 - Network-Based IOCs

These indicators are observed in network traffic and are often the first signs of suspicious or malicious activity.

Examples:

- + IP addresses linked to threat actors or command-and-control servers.
- + Domain names or URLs used in phishing, malware delivery, or callback infrastructure.
- + Unusual port usage (e.g., C2 traffic over port 443 using non-HTTPS protocols).
- + Beaconsing patterns: regular, low-volume outbound traffic indicating C2 communication.

Use Cases:

- + Easily detected by firewalls, IDS/IPS, and proxy logs.
- + Useful for blocking outbound threats and identifying lateral movement.

Limitations:

- + Evasive techniques like domain fluxing, fast-flux hosting, or IP rotation reduce reliability.
- + Encrypted traffic may obscure payloads or specific indicators.

Types of IOCs

2 - File-Based IOCs

These relate to specific files or file characteristics associated with malicious software or activity.

Examples:

- + File hashes (MD5, SHA-1, SHA-256) of known malware or droppers.
- + Suspicious filenames or file paths (%APPDATA%\badfile.exe, C:\Temp\payload.ps1).
- + Digital certificates used to sign malware (especially reused or stolen ones).
- + File size or metadata anomalies (e.g., large hidden files).

Use Cases:

- + Useful for identifying and blocking known malware.
- + Can be scanned for at scale across endpoints or mail systems.

Limitations:

- + Highly volatile: hashes change with small file modifications.
- + Adversaries often use polymorphic or packed malware to bypass hash-based detection.

Types of IOCs

3 - Behavioral IOCs

These are patterns of activity or anomalies that suggest malicious intent, regardless of the specific tools or files used.

Examples:

- + Abnormal login activity (e.g., logins from new geolocations, at odd hours).
- + Unexpected process behavior (e.g., Word launching PowerShell).
- + Lateral movement behavior (e.g., use of **PSEXEC**, RDP to multiple systems).
- + Persistence techniques (e.g., registry modifications, scheduled tasks).

Use Cases:

- + Harder to evade, as they focus on how an attacker operates, not just what they use.
- + Ideal for detecting novel or sophisticated attacks and insider threats.

Limitations:

- + Require baselining and context (e.g., UEBA(User and entity behaviour analytics), EDR analytics).
- + More prone to false positives if not tuned to the environment.

Common Examples of IOCs

IOC Type	Description
IP Addresses	Malicious or suspicious IPs used for command and control (C2), data exfiltration, or scanning.
File Hashes	MD5, SHA1, or SHA256 hashes of known malicious files (e.g., malware samples, droppers).
Domains & URLs	Hostnames or specific web addresses used in phishing, payload delivery, or callback infrastructure.
Email Artifacts	Malicious sender addresses, email subjects, or attachments used in phishing attacks.
Registry Keys	Windows registry changes used for persistence or system manipulation by malware.
Filenames / Paths	Known malware or suspicious files stored in typical or obscure locations (e.g., temp.exe in unusual folders).
Process Names	Abnormal or masquerading process names like svch0st.exe or expl0rer.exe .
Mutexes	Unique identifiers used by malware to avoid multiple instances running.
Beaconing Patterns	Regular and repetitive network traffic that signals C2 communication.
Network Artifacts	Unusual port usage, encrypted traffic patterns, or protocol anomalies.

Identifying IOCs

Identifying IOCs

IOC identification is the process of *discovering* and *recognizing indicators of compromise* within your environment, often as part of alert triage, threat hunting, or incident response.

The goal is to extract and validate artifacts that may signify malicious activity. The process of identifying IOCs entails the following:

- + **Detection:** Security tools log or alert on suspicious behavior.
- + **Extraction:** Analysts identify observable data points; IP, URL, file hash, registry key.
- + **Correlation:** Match indicators across systems and time to build patterns.
- + **Enrichment:** Validate against known databases and add context.
- + **Validation:** Decide if it's truly malicious, benign, or unknown.

Identifying IOCs

Detection Sources

- + IOC identification typically begins with data collected from security telemetry. These sources capture suspicious behaviors, artifacts, and events.

IOC Correlation & Identification

- + Once a potential IOC is surfaced, it's subjected to correlation and contextual analysis to confirm its malicious nature.

Correlation → Link the IOC to:

- + Similar events or alerts across different hosts.
- + Known attack campaigns or malware families.

Example: An IP observed in outbound traffic is linked to a known phishing domain also detected in a recent email.



IOC Analysis & Enrichment

IOC Analysis

IOC analysis is the process of *investigating and understanding indicators of compromise* like suspicious IP addresses, file hashes, or **domains** in order to figure out if they are part of a cyberattack.

Think of it like being a digital detective:

- + You find a clue (an IOC) in an alert or log.
- + You look it up to see if it's known to be bad.
- + You check where it came from, who used it, and what it tried to do.
- + You decide if it's dangerous and if action is needed.

IOC Enrichment

IOC Enrichment is the process of **adding contextual information** to **raw Indicators of Compromise (IOCs)** to make them **more actionable, accurate, and meaningful** for detection, triage, investigation, and response.

While raw IOCs (like an IP or file hash) offer clues, **enrichment answers critical questions** like:

- + What does this indicator represent?
- + Is it known to be malicious?
- + What threat actors or campaigns is it associated with?
- + How urgent is the response?

IOC Enrichment Techniques

1 - Adding Contextual Data

This step *enhances the IOC* with *environmental* and *threat-relevant information* that helps analysts make better decisions.

Context Type	Purpose
Geo-Location	Identifies the geographic origin of an IP address (e.g., C2 traffic from a country with known threat activity).
ASN (Autonomous System Number)	Helps group IPs by ownership (e.g., a specific hosting provider or VPN service). Useful for identifying clusters of infrastructure.
WHOIS Information	Reveals domain registration data like: owner, date created, contact info. Red flags include newly registered domains or privacy-protected registrations.
Threat Actor Association	Links the IOC to known threat actors or campaigns based on threat intel reporting. Helps understand likely TTPs and motivations.

IOC Enrichment Techniques

2 - Leveraging Threat Intelligence Sources

External and **internal Threat Intelligence (TI)** helps **validate** and **enrich indicators** with global and organizational insight.

Source Type	Functionality
External Threat Feeds	Sources like VirusTotal, AbuseIPDB, AlienVault OTX, Recorded Future, or commercial TI vendors provide reputation scores, tagging, and relationships.
Sandbox Results	Dynamic analysis (e.g., ANY.RUN, Hybrid Analysis) simulates execution of files or URLs to observe behavior and extract secondary IOCs.
Passive DNS & Passive SSL	Observes historic domain-IP pairings and certificate usage to track infrastructure reuse.
Internal Reputation Databases	Correlate new IOCs with past incidents, hunt results, or false positive reports. Critical for environment-specific relevance.

IOC Enrichment Techniques

3 - Enrichment For Enhanced Decision Making

Enriched IOCs are not just informative, *they're strategic*. They feed directly into key decision-making areas:

Objective	How Enrichment Helps
Determine Severity	An IOC linked to known ransomware infrastructure increases the criticality of an alert.
Define Scope	Context such as multiple host observations, lateral movement evidence, or multiple users interacting with an IOC expands the scope.
Prioritize Response	IOC associated with a low-prevalence, high-impact threat (e.g., data exfiltration or hands-on-keyboard attack) warrants faster containment.
Triage Accuracy	Helps Tier 1 analysts quickly identify false positives or escalate real threats with confidence.

A man with glasses and a beard is shown in profile, looking at a computer monitor in a dark room. The monitor displays some code or data. The overall atmosphere is professional and focused.

Responsibilities in IOC Handling

Responsibilities in IOC Handling

Tier 1 Analyst Responsibilities in IOC Handling

- + Tier 1 SOC analysts are on the front lines of the incident response process.
- + Their role is to triage alerts, identify threats early, and determine if further action is needed, especially using IOCs.

Key Responsibilities

1 - Alert Triage and IOC Matching

- + Analysts review alerts generated by tools like SIEMs and EDR platforms.
- + They look for IOCs (e.g., suspicious IPs, domains, file hashes) in the alert data.
- + These IOCs are matched against "known bad" lists—internal blocklists or threat intelligence feeds.
- + ***The goal: Identify real threats quickly, discard false positives.***

Responsibilities in IOC Handling

2 - Basic IOC Enrichment

Tier 1 analysts perform lightweight research to add context to the IOCs:

- + Use VirusTotal to check if a file hash or URL is flagged by antivirus vendors.
- + Lookup **geoIP** data to identify suspicious IP origins (e.g., a login from North Korea).
- + Use WHOIS to check domain registration info (e.g., newly registered or anonymized domains).

This enrichment helps answer: Is this IOC really malicious or just unusual?

3 - Escalation Decision-Making

Based on the analysis and enrichment, Tier 1 analysts decide whether to:

- + Escalate to Tier 2 or IR for deeper investigation.
- + Close the alert if it's a false positive or non-malicious.
- + Decisions are based on evidence, context, and standard operating procedures.

Responsibilities in IOC Handling

Tier 2/Incident Responder Responsibilities in IOC Handling

- + Incident Responders (IRs) operate at a deeper, investigative layer in the SOC.
- + When Tier 1 or 2 analysts escalate an alert, IRs step in to determine the full scope of the threat, contain it, and enhance future detection, often using advanced IOC techniques.

Responsibilities in IOC Handling

Key Responsibilities

1 - Advanced IOC Correlation and Pivoting

- + **Link analysis:** Mapping connections between domains, IPs, and hashes to uncover infrastructure or attacker behavior patterns.
- + **Pivoting:** Starting from a known IOC (like a file hash) and searching across systems to identify:
 - + Multiple infected hosts
 - + Lateral movement
 - + Command-and-control relationships

This helps IRs uncover the full kill chain and eliminate blind spots.

Responsibilities in IOC Handling

2 - Use of Threat Hunting Platforms and Malware Sandboxes

- + **Threat Hunting Platforms:** Allow proactive search for threats across large datasets using behavioral logic and IOC matching.
- + **Malware Sandboxes:** Execute suspicious files in isolated environments to observe behaviors, extract secondary IOCs, and validate payload functionality.

3 - Generating New IOCs

From their investigations, IRs often discover new IOCs such as:

- + Additional domains, hashes, or URLs used in the campaign.
- + Behavioral indicators (e.g., specific registry changes, process trees).
- + Infrastructure reuse patterns.

These are then:

- + Fed back into detection logic (e.g., SIEM rules, EDR alerts, firewall blocks).
- + Shared with threat intelligence teams to enrich org-wide defenses.
- + Used to fine-tune alerting and reduce false positives in the future.

Incident Response: Detection & Analysis - Summary

Key Concepts - Recap

- + Threat Detection Foundations
- + SIEM Operations
- + Triage & Escalation
- + Endpoint/Host Analysis
- + Network Analysis



Learning Outcomes Recap

- + Explain the role and importance of the detection and analysis phase in the incident response lifecycle.
- + Differentiate between events, alerts, and incidents, and describe the process of alert triage.
- + Identify key log sources across endpoint, network, and application layers, and explain their relevance to incident response.
- + Collect and analyze logs from Windows and Linux systems using native and third-party tools.
- + Configure and utilize centralized logging solutions such as Splunk and the ELK stack for effective log analysis.
- + Develop and evaluate detection rules, manage alert noise, and prioritize incident response efforts.
- + Identify, enrich, and contextualize indicators of compromise (IOCs) using internal data and threat intelligence.
- + Perform initial response actions and deep analysis to assess the scope and impact of security incidents.
- + Conduct endpoint and log-based investigations using tools like Sysmon, wevtutil, EvtxECmd, and Chainsaw.
- + Analyze network traffic using PCAPs, flow data, and logs to detect scanning, attacks, and exfiltration.

Next Steps

- + Apply your skills in a lab environment (Course labs & Skill Dive)
- + Study real-world incident reports and postmortems:
 - + Analyze public breach reports (e.g., from Mandiant, CISA, or Microsoft) to see how professionals apply detection and analysis techniques.
- + Explore advanced incident response topics:
 - + Deepen your knowledge in areas like malware analysis, threat hunting, memory forensics, and digital forensics.

Next Steps

- + Stay updated on emerging threats and tools:
 - + Follow cybersecurity blogs, join communities (e.g., DFIR Slack, Reddit's /r/blueTeamSec), and attend webinars or conferences to remain current in the field.

THANKS FOR WATCHING!



EXPERTS AT MAKING YOU AN EXPERT

