جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

أكاديمية كاوست
KAUST ACADEMY

DAY 2

LOG MONITORING: COLLECTION, MANAGEMENT & ANALYSIS

Dr. Ali Hassan
Instructional Professor, CEMSE Division,
King Abdullah University of Science and Technology,
Contact: ali.hassan.1@kaust.edu.sa

Site Manager:
Mahmoud Ellouh
Teacher Assistants:
Mohammed Alqurashi
Osama Aldossary
Jumanah Alharbi
Ritaj Alghamdi
Salman Aldhalaan

ACKNOWLEDGEMENT

Some of the material presented here is prepared by:

Dr. Rashid Tahir,
Assistant Professor,
CSFC Department
University of Prince Mugrin, Madinah

LECTURE OUTLINE

- What are log files?
- Types of log files
- Log file creation and storage
- How do log files help?
- Log management & analysis
  - Syslog standard
  - Common log management functions
  - Analysis techniques
  - SIEM and SOC

WHAT ARE LOG FILES?

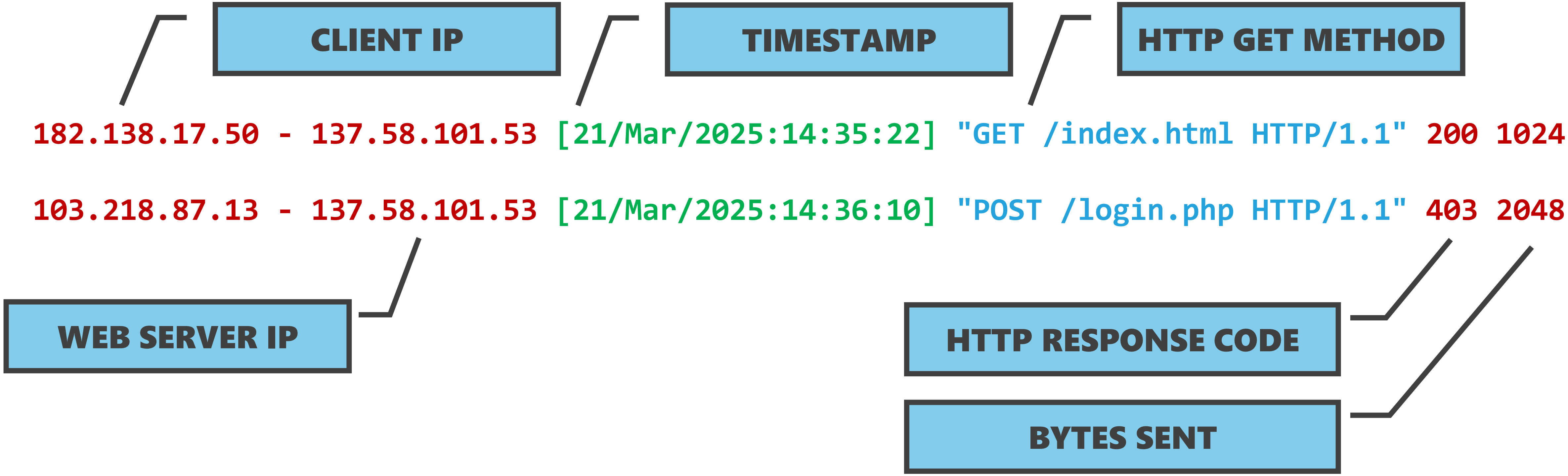
Understanding event recording and tracing via log files

LOG FILES – AN INTRODUCTION

- Log files are **records** of events, activities, incidents and transactions stored in a file
- Generated by systems, applications, network appliances, middleboxes, security devices, etc.
- Provide **critical visibility** into system operations, user actions, and potential security incidents

EXAMPLE 1 – WEB SERVER LOG

SCENARIO: A web server receives one **HTTP GET request** and one **HTTP POST request** from two clients on the Internet. For the first request, the resource is available and returned (**code 200 – OK**). For the second request, the client attempted an unauthorized action and hence, is denied (**code 403 – Forbidden**)



```
182.138.17.50 - 137.58.101.53 [21/Mar/2025:14:35:22] "GET /index.html HTTP/1.1" 200 1024

103.218.87.13 - 137.58.101.53 [21/Mar/2025:14:36:10] "POST /login.php HTTP/1.1" 403 2048
```

EXAMPLE 2 – WINDOWS SECURITY LOG

```
Source: Microsoft-Windows-Event-Log
Log Type: Security
Event ID: 4625
Task Category: Logon
Level: Information
User: admin
Computer: SERVER21
Date/Time: 2025-03-21 14:32:10
Description: An account failed to log on.
    - Account Name: admin
    - Workstation Name: DESKTOP-KU21
    - Source IP: 192.168.1.100
    - Failure Reason: Unknown username or bad password
```

SCENARIO: A user attempted to log into a Windows machine but provided incorrect credentials. The **authentication request failed**, triggering a security event in the Windows Event Log under the Security category. This log entry records details such as the username, source IP, timestamp, and failure reason

TYPES OF LOG FILES

What type of information can be recorded and stored?

TYPES OF LOG FILES

- Understanding different **types of logs** and their **sources** is critical for **effective log monitoring** and **analysis**
  - System Logs (e.g., Windows Event Logs, Linux Syslog)
  - Network Logs (e.g., Firewalls, IDS/IPS, Load Balancers, Routers)
  - Application Logs (e.g., Web Servers, Databases, Cloud Services)
  - Security Logs (e.g., SIEM, Antivirus, Honeypot, Endpoint Detection & Response)
  - Operational Technology (OT) Logs (e.g., SCADA, Data Historian, HMI logs)
- Each log type provides **unique insights** into system behavior, security incidents, and operational performance

HOW ARE LOG FILES CREATED & WHERE CAN WE FIND THEM?

Windows vs Linux

LOG CREATION – WINDOWS SYSTEMS

- In **Windows** systems, logs are created by the **Windows Event Logging** service
  - Collects, stores and manages logs from various system components (OS, services, apps)
- Categorizes records into **four different types**:
  - *Security Logs* (Records any security related events)
  - *System Logs* (OS events like driver failures)
  - *Application Logs* (Software and application events)
  - *Setup Logs* (Installation and update-related logs)
- Logs are stored in two directories:
  - `C:\Windows\System32\winevt\Logs` (new location)
  - `C:\Windows\System32\config` (old location but still used)
- Logs can be viewed & analyzed in the **Windows Event Viewer** utility
- Users can also perform **targeted security logging** through **Windows Security Auditing** feature
  - Takes in a user-specified **auditing policy** to track certain types of events and activities

LOG CREATION – LINUX-BASED SYSTEMS

- In **Linux** systems, logging is generally performed through a **Syslog-based** utility, such as *rsyslog*, *syslog-ng* or *Graylog*
  - Syslog captures a wide range of system, application, and security events
  - Well-defined and widely-used logging standard
  - Syslog will be covered in more detail in the subsequent slides

- Logs are stored in **/var/log/** directory (most apps/utilities share this directory for storing logs of different kinds)
- For **targeted logging** of security events and incidents, **Linux Audit Framework** (*AuditD*) is used
  - Equivalent to the *Windows Security Auditing* feature
  - Tracks security events across the system based on audit policies
  - Logs are stored in **/var/log/audit/audit.log**

WHY ARE LOGS IMPORTANT AND HOW DO THEY HELP

Understanding the role of log files in cybersecurity & digital forensics

ROLE OF LOG FILES

- Logs play a critical role in both cybersecurity and digital forensics
  - Provide a **recorded history** of system, network, and user activity
  - Important **source of evidence** in investigating incidents
- Help answer key questions about **attack timeline** and **attribution**
  - Who accessed the system and when?
  - What commands or actions were performed on the system?
  - Was any sensitive data stolen or exfiltrated?
  - Were there any security policies violated?
  - Did the infection spread to other machines in the network?
  - And many others!!

GENERAL BENEFITS OF LOG MONITORING

- Log monitoring refers to the continuous **collection**, **analysis**, and **real-time tracking** of log data generated by systems, networks, applications, and security devices
  - Supports **troubleshooting** performance-related problems, slow response times & crashes
  - Ensures **system integrity** by tracking changes to configuration files and registry settings
  - Helps **detect anomalies**, security incidents, and operational issues
  - Facilitates the process of **addressing cyber threats** before they escalate
  - Essential for **incident response** and **compliance requirements**
  - Heavily used to **monitor infrastructure state** via Security Information and Event Management (SIEM) and Security Operations Center (SOC)
- Let’s see some more details of log monitoring and its applications

EXAMPLE USE CASES & APPLICATIONS

- **Threat Detection and Incident Response:**
  - User Authentication logs help detect brute force attacks and unauthorized logins
  - Firewall and IDS/IPS logs reveal suspicious network traffic (e.g., port scans, DDoS attacks)
  - Endpoint Security logs detect malware infections, unauthorized software installations, and suspicious command executions
- **Security Monitoring and Anomaly Detection:**
  - By combining logs from various sources (e.g., firewalls, servers, endpoint devices), organizations can detect anomalies that might indicate an attack
- **Compliance and Regulatory Requirements:**
  - **GDPR & HIPAA:** Require logs to track access to personal or sensitive data
  - **PCI-DSS:** Mandates logging of all access to cardholder data

LOG MANAGEMENT & ANALYSIS

Centralized vs Decentralized

LOG MANAGEMENT APPROACHES

CENTRALIZED

Focus Here

- All logs are collected and stored in a **central repository** (e.g., SIEM solutions)
- Enables correlation across different systems for better insights
- Allows for efficient long-term storage and retrieval

DECENTRALIZED

- Logs are stored **locally on devices** and are analyzed independently
- Common in legacy or air-gapped environments (e.g., ICS/OT networks)
- Devices retain control over log data but makes correlation harder

INTRODUCING THE SYSLOG STANDARD

The gold standard of centralized logging

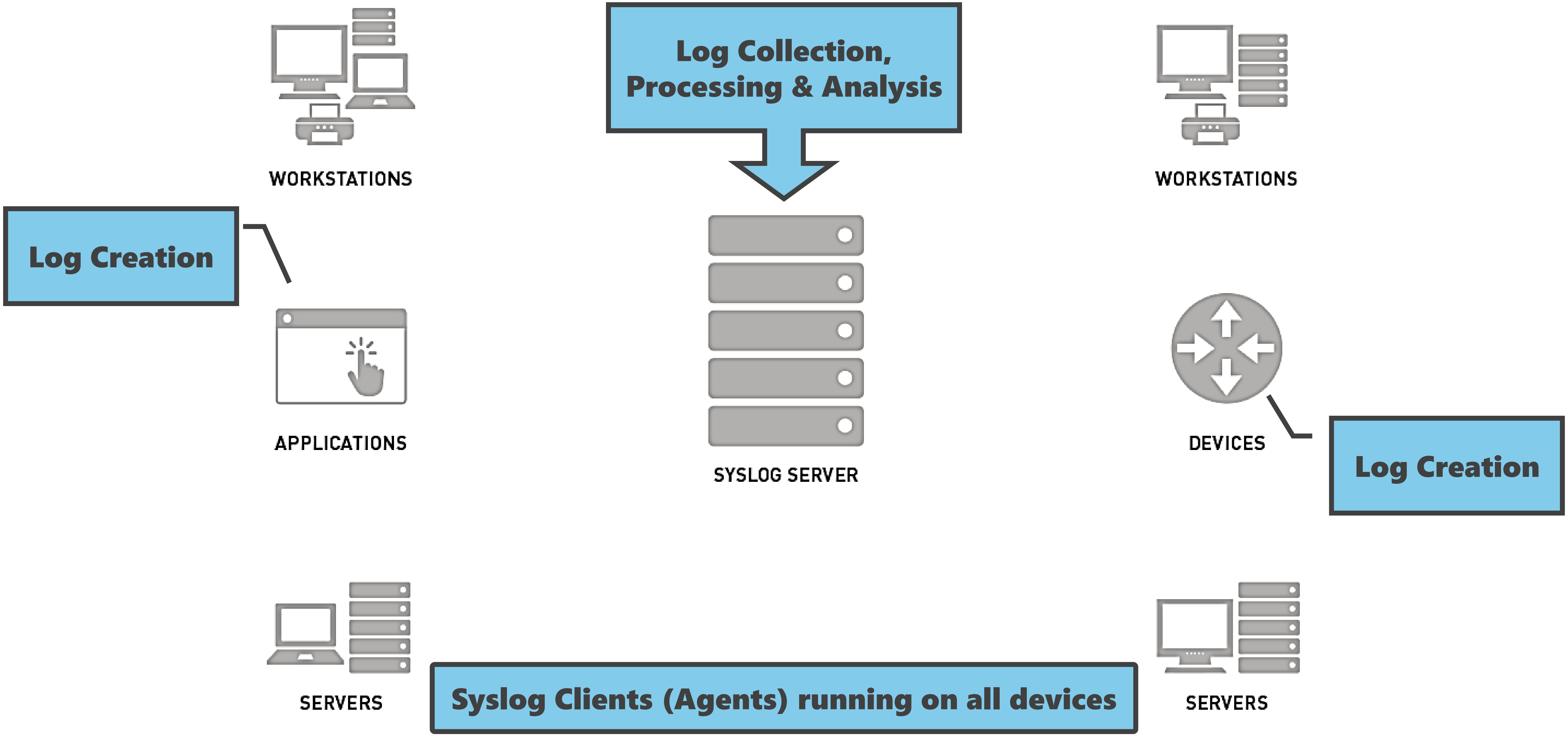
WHAT IS SYSLOG

- Syslog is a **comprehensive logging standard** for centralized message logging
- Modular design allows for the separation of the software that generates messages, the system that stores them, and the software that reports and analyzes them
  - Frees programmers from **managing** log files
  - Gives sysadmins **control** over log management
- Each message includes a:
  - **Facility Code** (what is the source of a message or where did a certain event take place)
  - **Severity Level** (what is the criticality of a message or how serious is an event)
- Admins and devs may use syslog for **system management** and **security auditing** as well as general informational, analysis, and debugging messages
- A wide **variety of devices**, such as printers, routers, middleboxes, etc., across many platforms use the Syslog standard
- Consolidates logging data from different types of systems into a **central repository** for processing and analysis

SYSLOG – ARCHITECTURE

- Syslog Client
  - **Daemon** that does the **actual logging**
  - Can be configured to track and record **events** of different **types** at different **granularity**
  - **Shares** the log data with the server
- Syslog Server
  - Also known as the Syslog **Collector/Receiver/Listener**
  - **Collects** all Syslog messages sent by the network devices in a **database**
  - Responsible for **filtering** the data and **generating alerts** (or appropriate response)
- In a typical network, numerous Syslog clients are simultaneously sending log data to the Syslog server

CENTRALIZED LOGGING – SYSLOG



Log Collection, Processing & Analysis

Log Creation

Log Creation

Syslog Clients (Agents) running on all devices

SYSLOG – FACILITY CODES

A **facility value** is used to specify the **type of system** that generated an event. Is also used to compute the priority of the event (PRI).

NUMBER	FACILITY DESCRIPTION
0	Kernel messages
1	User-level messages
2	Mail system
3	System daemons
4	Security and authorization-related messages
…	…
15	Clock daemon
16-23	Eight local levels for other programs

SYSLOG – SEVERITY LEVELS

A **severity code** is used to define the **severity level** (or criticality) of an event that is being logged.

CODE	SEVERITY	DESCRIPTION
0	***Emergency***	System is unusable, panic situations (hardware failure, crash)
1	***Alert***	Urgent situations, immediate action required
2	***Critical***	Critical situations or conditions
3	***Error***	Non-critical errors
4	***Warning***	Warnings
5	***Notice***	Might merit investigation
6	***Informational***	Informational messages
7	***Debug***	Debugging (typically enabled temporarily)

SYSLOG – PRIORITY VALUE (PRI)

- The two values (**Facility** value and **Severity** code) are combined to produce a **Priority Value** (PRI) sent with the message
- The Priority Value is calculated by **multiplying** the Facility value by eight and then **adding** the Severity code to the result
- PRI = (Facility Value x 8) + Severity Code
- The **lower** the **PRI**, the **higher** the **priority**
  - Higher priority items require immediate attention
  - Lower priority items can be deferred

SYSLOG – MESSAGE FORMAT

- The Syslog message consists of three parts:
  - **HEADER** (with identifying information)
  - **STRUCTURED DATA** (machine readable data in “**key=value**” format)
  - **MSG** (the message itself or the payload)
- **FORMAT (**RFC5424**):** *HEADER* **+** *STRUCTURED DATA* **+** *MSG*
  - **OLD FORMAT (**RFC3164**):** *PRI + HEADER + MSG*
- Some messages are simple, readable text, others may be quite long and contain fine-grained details covering every aspect of an event

LET'S LOOK AT EACH SYSLOG COMPONENT INDIVIDUALLY

Header + Structured Data + Msg

SYSLOG – HEADER COMPONENT

- **HEADER**
  - Priority Value (PRI)
  - Version
  - Timestamp
  - Hostname
  - Application
  - Process ID
  - Message ID
- **EXAMPLE:**

Follows the ISO 8601 format (YYYY-MM-DDThh:mm:ss±ZONE)

Process ID is missing

SYSLOG – HEADER COMPONENT

- **HEADER**
  - Priority Value (PRI)
  - Version
  - Timestamp
  - Hostname
  - Application
  - Process ID
  - Message ID

Follows the ISO 8601 format (YYYY-MM-DDThh:mm:ss±ZONE)

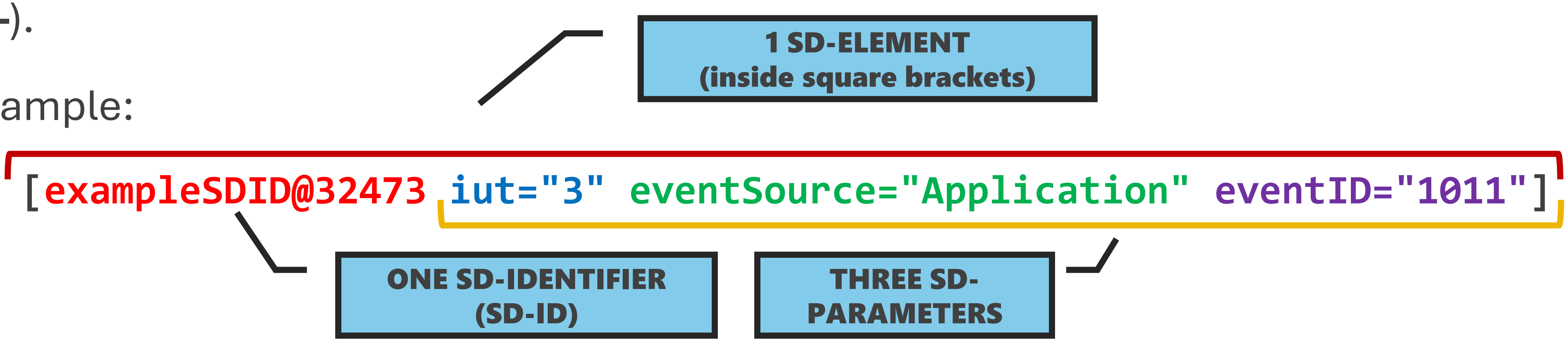
Process ID is missing

- **EXAMPLE:**

```
<34>1 2022-10-11T22:14:15.003Z mymachine.example.com su — ID47
```

SYSLOG– STRUCTURED DATA COMPONENT

- **STRUCTURED DATA**
  - Provides a mechanism to express information in a **well-defined**, easily **parseable** and **interpretable** data format in the form of **key=value pairs**.
- Can contain zero, one, or multiple structured data elements (SD-Elements)
- In case of zero SD-Elements, the STRUCTURED DATA field MUST contain the NILVALUE (—).
- Example:



```
[exampleSDID@32473 iut="3" eventSource="Application" eventID="1011"]
```

1 SD-ELEMENT (inside square brackets)

ONE SD-IDENTIFIER (SD-ID)

THREE SD-PARAMETERS

- This example has one SD-Element with an SD-ID that has value "exampleSDID@32473", which has three further parameters (one in blue, one in green and one in purple).

SYSLOG – MSG COMPONENT

- **MSG**
  - The MSG part (also called the payload) contains a **free-form** message that provides information about the event.
- If a Syslog application encodes the message body in UTF-8 encoding, the string MUST start with the Unicode Byte Order Mask or Mark (**BOM**)
  - The hex representation of UTF-8 BOM is EF BB BF
  - For other encodings, the BOM will be different
- The MSG component is often used to describe the event being recorded, for example:
  - Failed login attempt by remote user
  - Configuration settings changed
  - Patch C157 installed by admin user

SYSLOG – EXAMPLE

```
<165>1 2025-02-11T22:14:15.003Z kaust.server123.com evntslog 1187 ID47 [sampleSDID@786 interface=“eth1” eventSource=“NginX” protocol=“TCP”] [SDID@KAUST471 severity=“warning”] An Application event log entry was deleted unexpectedly
```

- In this example, we have the following information:
  - **HEADER** is in **red** font, **STRUCTURED DATA** elements are in **blue** font and **MSG** is in **green** font
  - The PRI value is 165
  - The Syslog version is 1
  - The message was created on 11 February 2025 at 10:14:15pm UTC, 3 milliseconds into the next second
  - The message originated from the host “kaust.server123.com"
  - The name of the application that generated the message is “NginX“
  - The process ID is 1187
  - The message ID is ID47
  - There are two structured data elements in the STRUCTURED DATA component. The first has SD-ID “sampleSDID@786” and three parameters and the second has SD-ID “SDID@KAUST471” with only one parameter
  - The message or payload is "An application event log entry was deleted unexpectedly”

LOG MANAGEMENT PLATFORMS

A necessity in the *age of data*

WHAT IS A LOG MANAGEMENT PLATFORM?

- Logs constitute **large amounts** of data
  - Once aggregated, logs can be gigabytes or terabytes of data
  - Makes management and analysis very challenging and time-consuming
- Log management platforms help deal with this challenge
- Provide several desirable functions to make dealing with log data manageable:
  - Collection & Aggregation
  - Log Storage
  - Log Analysis & Reporting
  - Log Disposal
- Multiple components work together to **generate**, **transmit**, **store**, **analyze** and **dispose** of log data

LOG MANAGEMENT – FUNCTIONS

- **Collection & Aggregation**
  - Log Parsing
  - Event Filtering
  - Event Aggregation
- **Storage**
  - Log Rotation
  - Log Archiving
  - Log Compression
  - Log Reduction
  - Log Normalization / Conversion
  - Log File Integrity Checking
- **Analysis**
  - Event Correlation
  - Log Viewing
  - Log Reporting
- **Disposal**
  - Log Clearing

LOG MANAGEMENT – FUNCTIONS

Collection & Aggregation

- Log Parsing
  - Extracts specific data fields from raw log entries, **transforming unstructured logs** into **structured data** that can be easily analyzed or used in other logging processes.
- Event Filtering
  - Not all log entries are valuable. Event filtering identifies and **suppresses log entries** that are deemed **low-priority** or **irrelevant**, reducing noise and optimizing storage.
- Event Aggregation
  - When multiple log entries describe the same event, aggregation **merges** them into a **single record** while maintaining a count of occurrences. This minimizes redundancy and reduces size of data.

LOG MANAGEMENT – FUNCTIONS

Storage

- Log Rotation
  - To prevent logs from growing indefinitely, log rotation **closes** an active **log file** and **starts** a **new one** based on a predefined schedule (e.g., hourly, daily) or when a file reaches a set size.
- Log Archiving
  - Security logs often need to be stored long-term to meet legal, regulatory, or forensic requirements. Logs may be **moved** to **external** or **secondary storage** (e.g., SAN, cloud storage, or dedicated log servers) for future reference.
- Log Compression
  - To conserve storage, log compression **reduces file size** without altering content. This is commonly applied during log rotation or archiving.
- Log Reduction
  - Log reduction is **removing unneeded entries** from a log to create a new log that is smaller. A similar process is event reduction, which removes unneeded data fields from all log entries.

LOG MANAGEMENT – FUNCTIONS

- ## Storage
  - Log Normalization / Conversion
    - Logs often exist in different formats. Conversion **translates** logs from **one format** to **another** (e.g., from a database format to a structured XML file) to ensure compatibility across tools and systems.
  - Log File Integrity Checking
    - To detect tampering, integrity checks **compute** and **store cryptographic hashes** (message digests) of log files. Any unauthorized modification is flagged as a security concern.

LOG MANAGEMENT – FUNCTIONS

Analysis

Event Correlation

- This technique **connects related log entries** to detect patterns, anomalies, or security incidents. Rule-based correlation is commonly used to link events based on timestamps, IPs, or user actions.

Log Viewing

- Raw logs can be complex. Log viewers **format** and **display logs** in a **human-readable way**, often with search, filtering, and aggregation capabilities.

Log Reporting

- Reports **summarize log data** over a defined period, highlighting critical security events, trends, or compliance insights. These reports are essential for audits and incident investigations.

Disposal

Log Clearing

- When logs are no longer needed, log clearing **removes old entries** while ensuring important data has been archived. This prevents unnecessary log buildup and optimizes system performance.

Event Correlation & Root Cause Analysis

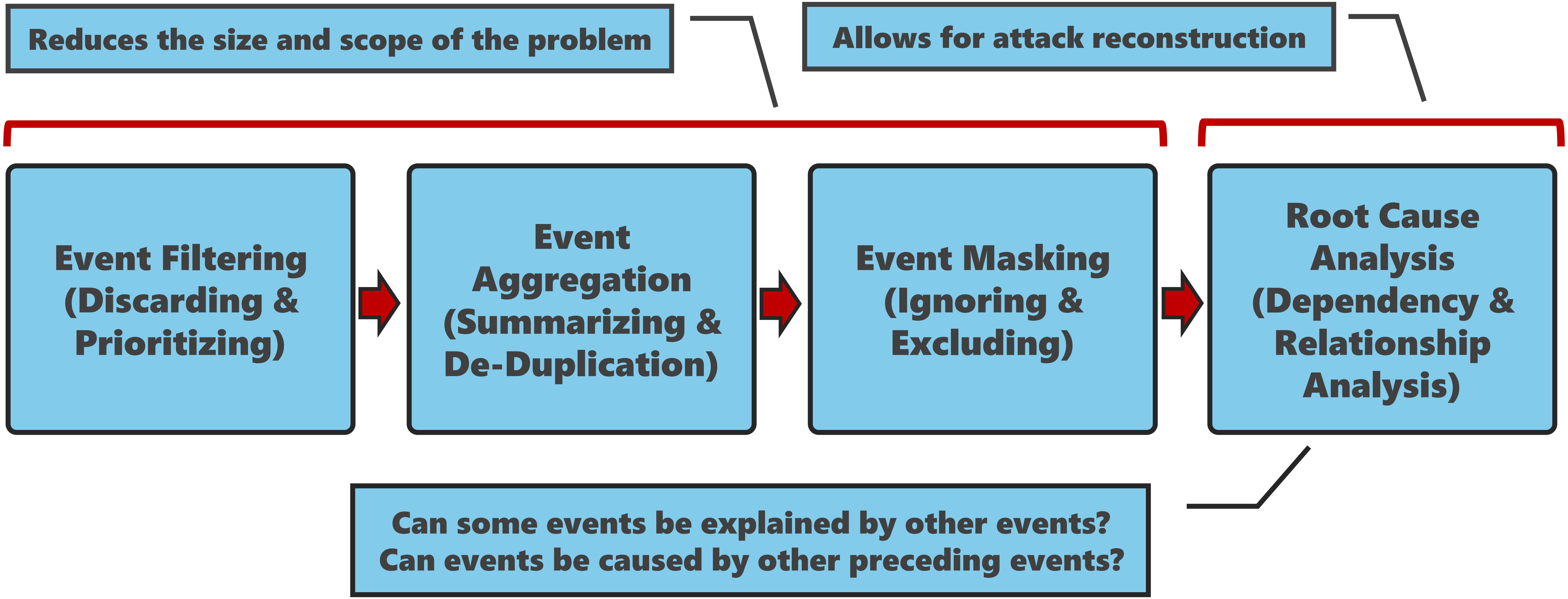
From raw data to insights: Analyzing log files

UNDERSTANDING EVENT CORRELATION

- Event correlation is a technique that relates or links various events across logs to identify **relationships** and **attack patterns** and determine the **cause** and **methodology** of an attack
- Events can be linked or correlated based on several **attributes**:
  - Similar IP addresses, usernames/accounts, hostnames, etc.
  - Events triggered by the same process, application or executable
  - Close physical proximity or geolocation
  - Temporally sequential events (log entries occurring in quick succession having close timestamps)
  - Events originating from the same device, service, or cloud provider
- Used for making sense of a **large number of events** and **pinpointing** the few events that are really important in a mass of information
- **Root Cause Analysis** (RCA) is a major component of event correlation
  - Method of problem solving used for identifying the root causes (or **primary causes**) of faults or problems

EVENT CORRELATION & ROOT CAUSE ANALYSIS

In log analysis, event correlation is usually a four-step process carried out on a **Log Management Platform**:



Reduces the size and scope of the problem

Allows for attack reconstruction

Event Filtering (Discarding & Prioritizing)

Event Aggregation (Summarizing & De-Duplication)

Event Masking (Ignoring & Excluding)

Root Cause Analysis (Dependency & Relationship Analysis)

Can some events be explained by other events?
Can events be caused by other preceding events?

LET'S LOOK AT AN EXAMPLE

Understanding the process of extracting meaningful insights from log files

EVENT CORRELATION – EXAMPLE

- **Scenario Overview:**
  - A cybersecurity incident has occurred where an attacker gained access to an enterprise network through a phishing attack. The attacker then escalated privileges, moved laterally (pivoted), and exfiltrated sensitive data.
- **Phishing Email** → **PowerShell Execution** → **C2 Communication** → **Credential Theft** → **Lateral Movement** → **Data Exfiltration**
- We have logs from different network devices and security systems
- We will analyze the logs and correlate the events

DEVICE	LOG SOURCE	RELEVANT LOG ENTRIES
EMAIL GATEWAY	Email security logs	A phishing email with a malicious attachment was sent to user1@company.com.
USER WORKSTATION	Windows Event Logs (Security)	User1 opened the attachment, which spawned powershell.exe, indicative of a malicious script execution (Event ID 4688).
FIREWALL	Network logs	An outbound connection was established to attacker.com over port 443, indicating a possible C2 communication.
EDR	Host-based logs	mimikatz.exe was executed, suggesting credential dumping.
ACTIVE DIRECTORY	Domain Controller Logs	User1’s credentials were used to attempt multiple authentication requests on different machines. Several failed logins followed by a successful login were recorded (Event ID 4624 and 4625).
SIEM	Aggregated logs	Multiple login attempts from User1’s workstation to high-privilege admin accounts.
FILE SERVER	File Access Logs	Large file transfers of sensitive data were initiated from a newly created account.
DLP SYSTEM	Data Exfiltration Logs	Unusual outbound file transfer to an external cloud storage service detected.

EVENT CORRELATION – EXAMPLE

Step 1: Initial Compromise (Phishing & Malware Execution)

- *Email Security Gateway (Phishing & Malware Delivery):*

  ```
  <134> 1 2025-03-08T10:15:23Z mailGW1 EmailSecurity 5432 MSG001 [eventSDID@137 email_category="phishing“ user_inbox="user1"] ALERT: Suspicious Email detected - Subject: "Urgent Invoice - Open ASAP", From: "attacker@evil.com", To: "user1@company.com", Attachment: "invoice.docm"
  ```

- *User Workstation (Malicious Execution):*

  ```
  <54> 1 2025-03-08T10:16:45Z winPC1 Sysmon 872 MSG205 [eventSDID@76 process="powershell.exe" user="user1"] EVENT ID 4688 - New Process Created - Process: C:\Windows\System32\powershell.exe -ExecutionPolicy Bypass -File C:\Users\user1\AppData\Local\temp\malicious.ps1
  ```

CORRELATION: Email Security Logs → Windows Event Logs

EVENT CORRELATION – EXAMPLE

Step 2: Persistence & C2 Communication

- *Firewall (Outbound Connection to C2 Server):*

  ```
  <61> 1 2025-03-08T10:17:10Z firewall1 Firewall 3201 MSG013 [eventSDID@99 connection_type="outbound" src_ip="192.168.1.100" dst_ip="203.0.113.50" dst_port="443"] ALERT: Outbound connection detected - Action: Allowed
  ```

- *Endpoint Detection & Response (Credential Dumping via Mimikatz):*

  ```
  <98> 1 2025-03-08T10:18:55Z winPC1 edrAGENT 7854 MSG009 [eventSDID@06 process="mimikatz.exe" user="user1"] ALERT: Suspicious process detected - mimikatz.exe executed (PID 6789) - Possible credential theft
  ```

CORRELATION: Email Security Logs → Windows Event Logs → Network Logs → Host Logs

EVENT CORRELATION – EXAMPLE

Step 3: Privilege Escalation

- *Active Directory (Failed & Successful Logins) – 2 Entries:*

  -
    ```
    <103> 1 2025-03-08T10:19:30Z activeDC1 SecurityAgent 5123 MSG115 [eventSDID@33 user="user1" src_ip="192.168.1.100"] EVENT ID 4625 - Failed Logon Attempt - Reason: Invalid Credentials
    ```

  -
    ```
    <74> 1 2025-03-08T10:20:15Z activeDC1 SecurityAgent 5124 MSG116 [eventSDID@136 user="user1" src_ip="192.168.1.100" auth_method="NTLM"] EVENT ID 4626 - Successful Logon
    ```

CORRELATION: Email Security Logs → Windows Event Logs → Network Logs → Host Logs → Domain Controller Logs

EVENT CORRELATION – EXAMPLE

Step 4: Lateral Movement

- *Firewall (Lateral Connection Attempt):*

  ```
  <184> 1 2025-03-08T10:22:30Z firewall1 Firewall 3202 MSG014 [eventSDID@316 src_ip="192.168.1.100" dst_ip="192.168.1.200" protocol="RDP" dst_port="3389"] ALERT: Internal connection detected - Status: Successful
  ```

- *SIEM (Related or Matching Alert: Unauthorized Access):*

  ```
  <138> 1 2025-03-08T10:23:45Z siem1 SIEM 6902 MSG328 [eventSDID@26 user="user1" src_ip="192.168.1.100"] ALERT: Suspicious Lateral Movement - User1 accessed multiple devices within 5 minutes
  ```

CORRELATION: Email Security Logs → Windows Event Logs → Network Logs → Host Logs → Domain Controller Logs → Network Logs → SIEM Logs

EVENT CORRELATION – EXAMPLE

Step 5: Data Exfiltration

- *File Server (Unusual File Access):*

  ```
  <114> 1 2025-03-08T10:25:50Z fileSRV1 FileAudit 4398 MSG119 [eventSDID@88 user="user1" file="/sensitive_data/financials.xlsx" action="COPY"] ALERT: Large file transfer detected - Destination: C:\Temp\exfil_data.zip
  ```

- *DLP System (External Upload Detected):*

  ```
  <44> 1 2025-03-08T10:27:10Z dlp1 DLP 5551 MSG016 [eventSDID@155 user="user1" src_file="C:\Temp\exfil_data.zip" dst="cloudstorage.com" file_size="150MB"] ALERT: Unauthorized Data Transfer
  ```

CORRELATION: Email Security Logs → Windows Event Logs → Network Logs → Host Logs → Domain Controller Logs → Network Logs → SIEM Logs → File Access Logs → Data Exfiltration Logs

EVENT CORRELATION – TYPES

AI/ML-Based Approach

A neural network is constructed and trained to detect the anomalies in the event stream. It can also highlight root causes and various other indicators of interest.

Graph-Based Approach

A graph is constructed with each node as a system component and each edge as a dependency/relation among two components. The graph is then searched for peculiar patterns and sub-graphs indicative of a problem.

Rule-Based Approach

Events are correlated according to a set of rules and conditions. The system can take appropriate actions based on which rules and conditions are triggered.

DIGGING FOR IOCs IN LOG FILES

Indicators of Compromise

WHAT ARE IOCs

- **IOC:** Artifact or sign that indicates a system or network may have been breached
- Common types of IoCs:
  - **File Hashes** (MD5, SHA-1) of malware samples
  - **IP Addresses / Domains** used for command-and-control (C2)
  - **File Paths / Registry Keys** modified by malware
  - **Malicious Email Addresses** or **URLs** in phishing campaigns
  - A few others (unusual ports or services, suspicious cron jobs, malicious macros, etc.)
- Very important for monitoring an organization’s infrastructure for malicious activity
  - Enable early detection of threats
  - Help in incident response and containment
  - Support threat hunting and intelligence sharing

USING SIGMA & YARA TO FIND IOCs

- Logs contain a lot of information pertaining to different kinds of malicious activities, which leaves behind IoCs in the records
- Sigma and YARA are **YAML-based** detection languages (or tools) that search for **malicious patterns** or **indicators** in log files via **user-defined rules**
  - Sigma was designed specifically to scan and search through log data
  - YARA is mostly used for scanning files and executables/binaries but can also be used for log files
- Provide rich searching capabilities to analyze log files, fish out relevant data that matches the search criteria and raise alerts
- **Technology agnostic** standards with large **open-source repositories** containing thousands of "*ready to go*" rules

LET’S LOOK AT SOME YARA & SIGMA EXAMPLES

How to look for *Indicators of Compromise*

HUTING FOR IOCs – YARA

- In YARA, each rule contains a textual or binary pattern to match a particular malware family
  - This is called a **signature** (a binary value that indicates the presence of the malware)
- Specifically, each rule has three sections:
  - **Meta Section**
    - General description and meta-level information about the rule
  - **Strings Definition Section**
    - Specific strings to be searched in file or memory
  - **Condition Section**
    - Logic of the rule goes here
    - Usually refers to strings defined in the Strings section

SIGNATURE MATCHING – YARA EXAMPLE 1

```
rule kaust_trojan
{
  meta:
    description = "This is just an example"
    threat_level = 3
  strings:
    $a = {6A 40 68 00 30 00 00 6A 14 8D 91}
    $b = {8D 4D B0 2B C1 83 C0 27 99 6A 4E 59 F7 F9}
    $c = "UVODFRYSIHLNWPEJXQZAKCBGMT"
  condition:
    $a or $b or $c
}
```

Signature of malicious hexadecimal string

Signature of malicious hexadecimal string

Signature of malicious textual string

SIGNATURE MATCHING – YARA EXAMPLE 2

```
rule NCA_trojan
{
   strings:
      $hex_string = { E2 34 ?? C8 A? FB }

   condition:
      $hex_string
}
```

Full wildcard byte

Wildcard nibble (4 bits)

A wildcard means that YARA can ignore this value & only check the rest of the signature

SIGNATURE MATCHING – YARA EXAMPLE 3

```
rule UPM_trojan
{
   strings:
      $hex_string = { F4  23  [4-6]  62  B4 }

   condition:
      $hex_string
}
```

Arbitrary sequence of 4 to 6 bytes

Captures a jump in the malicious code

```
F4 23 01 02 03 04 62 B4
F4 23 00 00 00 00 00 62 B4
F4 23 15 82 A3 04 45 22 62 B4
```

YARA REAL EXAMPLE – EMOTET MALWARE

```
rule win_emotet_w1
{
  meta:
     description = "This rule targets a modified Emotet binary discovered on the 26th of
     January 2021."
  strings:
     $key = { c3 da da 19 63 45 2c 86 77 3b e9 fd 24 64 fb b8 07 fe 12 d0 2a 48 13 38 48 68
     e8 ae 91 3c ed 82 }
  condition:
     filesize >  300KB and
     filesize < 700KB and
     uint16(0) == 0x5A4D and
     $key
}
```

YARA REAL EXAMPLE – TRICKBOT TROJAN

```
rule win_trickbot_w0
{
  meta:
     description = "Detects mailsearcher module from the Trickbot Trojan“
  strings:
     $str_01 = “mailsearcher”
     $str_02 = “handler”
     $str_03 = “conf”
     $str_04 = “ctl”
     $str_05 = “SetConf”
     $str_06 = “file”
     $str_07 = “needinfo”
     $str_08 = “mailconf”
  condition:
     all of ($str_*)
}
```

HUNTING FOR IOCs – SIGMA

- Sigma rules contain information required to detect odd, bad or malicious behavior when inspecting log files (usually within the context of a SIEM – coming later)
- Rules are similar to YARA in appearance as both are YAML-based
- Each rule is separated into three main components:
  - **Detection**
    - What malicious behavior the rule should search for
    - Most important component of any Sigma rule as it specifies exactly what the rule is looking for across relevant logs
  - **Logsource**
    - What types of logs this detection should search over
  - **Metadata**
    - Other information about the detection

LET'S LOOK AT A SIGMA RULE EXAMPLE

A rule to raise an alert whenever a PowerShell process is launched on a Windows machine

```
title: Simple PowerShell Execution
id: simple-powershell-001
description: Detects when PowerShell is launched on a Windows system.
logsource:
    category: process_creation
    product: windows
detection:
    selection:
        Image | endswith: ‘\powershell.exe’
    condition: selection
fields:
    - Image
    - CommandLine
level: low
```

These YAML tags are all metadata of the Sigma rule

Which fields from the log entry should be included in the alert

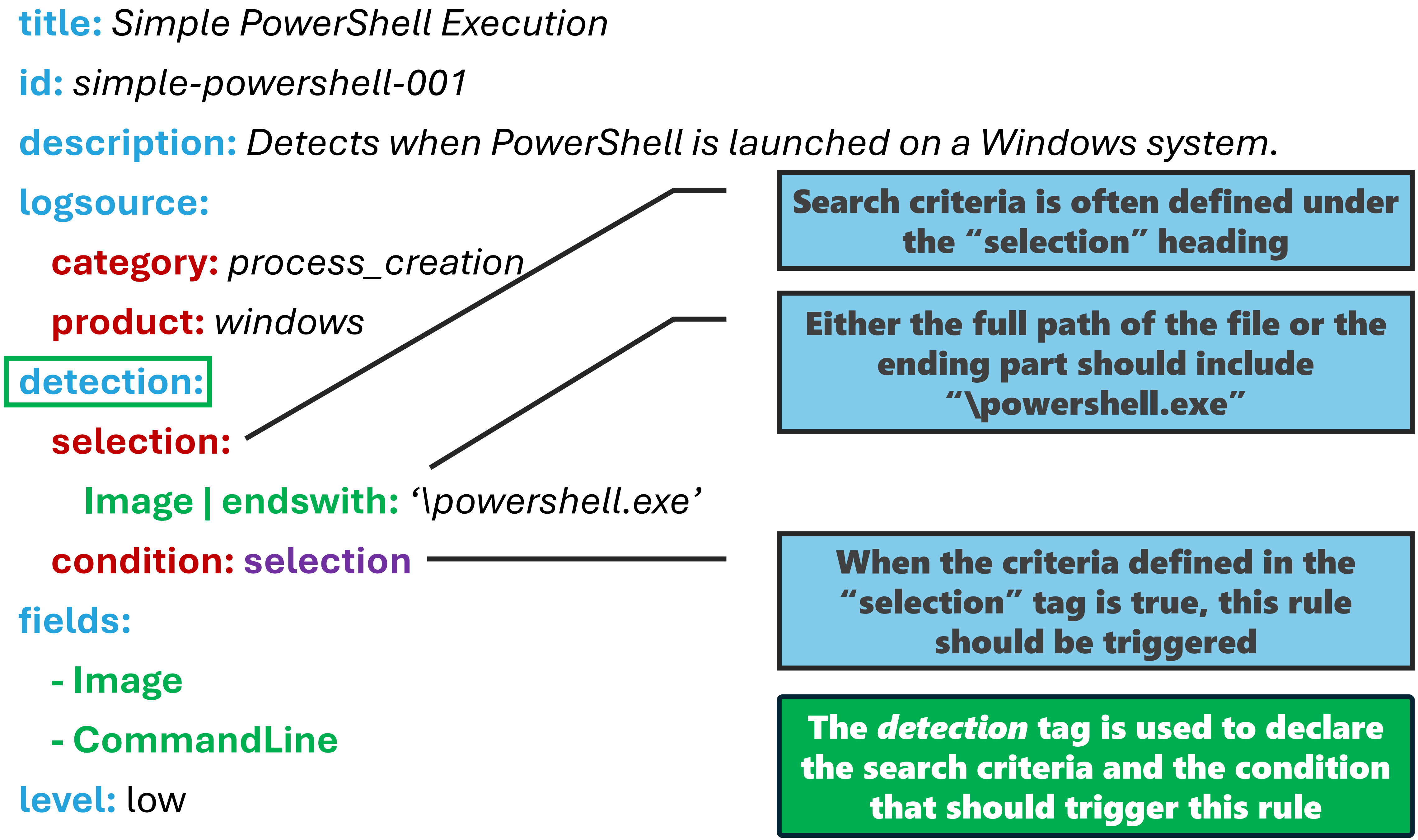
Image = The full path of the executable
CommandLine = The complete command along with all the arguments

```
title: Simple PowerShell Execution
id: simple-powershell-001
description: Detects when PowerShell is launched on a Windows system.
logsource:
  category: process_creation
  product: windows
detection:
  selection:
    Image | endswith: ‘\powershell.exe’
  condition: selection
fields:
  - Image
  - CommandLine
level: low
```

The log file for all created processes

The platform is Windows

The *logsource* tag is used to declare the exact log file on which this Sigma rule should be applied



```
title: Simple PowerShell Execution
id: simple-powershell-001
description: Detects when PowerShell is launched on a Windows system.
logsource:
  category: process_creation
  product: windows
detection:
  selection:
    Image | endswith: ‘\powershell.exe’
  condition: selection
fields:
  - Image
  - CommandLine
level: low
```

Search criteria is often defined under the “selection” heading

Either the full path of the file or the ending part should include “\powershell.exe”

When the criteria defined in the “selection” tag is true, this rule should be triggered

The *detection* tag is used to declare the search criteria and the condition that should trigger this rule

INTEGRATING ALL THAT WE HAVE LEARNED SO FAR!!

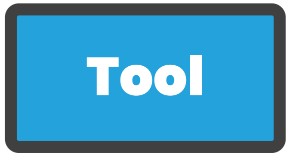
Security Information and Event Management (SIEM)

CYBER SECURITY PLATFORM – SIEM

- Security Information and Event Management (SIEM)
  - **Collects** and **aggregates** data from various devices and performs **correlation**
  - **Examines** and **analyzes** data for IoCs and signs of compromise using YARA/Sigma rules & user queries
- **ELK stack** is the most popular open-source log analysis and management platform used to build custom SIEM solutions (OSSEC Wazuh, Azure Sentinel, Apache Metron, etc.)
  - **E – Elasticsearch**
    - A search and analytics engine
    - Stores and indexes massive amounts of log data quickly
    - Think of it as the brain that lets you query everything fast
  - **L – Logstash (**often combined with Beats**)**
    - A data processing pipeline
    - Collects logs from various sources, processes them (e.g., filtering, conversion, etc.,), and ships to Elasticsearch
    - Like a smart conveyor belt for logs
  - **K – Kibana**
    - A visualization tool
    - Let's you explore, plot (e.g., extrapolation, trend lines, etc.,), and dashboard your log data
    - The UI of the stack used for user inputs/outputs and alerting

BEYOND SIEMs – SOCs

SIEM


Tool

- Think log collection + detection + correlation + dashboards
- Like a security camera system

SECURITY OP CENTER

People + Process + Tools

- The operational team uses tools (like SIEM) to defend the organization through structured processes
- Like a security guard team monitoring the infrastructure via cameras

QUESTIONS!!!