Detection and Response

Module 1: Intro to Detection and Incident Response

Roles in Response

Command, control, and communication

A computer security incident response team (CSIRT) is a specialized group of security professionals that are trained in incident management and response. During incident response, teams can encounter a variety of different challenges. For incident response to be effective and efficient, there must be clear command, control, and communication of the situation to achieve the desired goal.

- Command refers to having the appropriate leadership and direction to oversee the response.
- Control refers to the ability to manage technical aspects during incident response, like coordinating resources and assigning tasks.
- Communication refers to the ability to keep stakeholders informed.

Establishing a CSIRT organizational structure with clear and distinctive roles aids in achieving an effective and efficient response.

Roles in CSIRTs

CSIRTs are organization dependent, so they can vary in their structure and operation. Structurally, they can exist as a separate, dedicated team or as a task force that meets when necessary. CSIRTs involve both nonsecurity and security professionals. Nonsecurity professionals are often consulted to offer their expertise on the incident. These professionals can be from external departments, such as human resources, public relations, management, IT, legal, and others. Security professionals involved in a CSIRT typically include three key security related roles:

1. Security analyst
2. Technical lead
3. Incident coordinator

Security analyst

The job of the security analyst is to continuously monitor an environment for any security threats. This includes:

- Analyzing and triaging alerts
- Performing root-cause investigations
- Escalating or resolving alerts

If a critical threat is identified, then analysts escalate it to the appropriate team lead, such as the technical lead.

Technical lead

The job of the technical lead is to manage all of the technical aspects of the incident response process, such as applying software patches or updates. They do this by first determining the root cause of the incident. Then, they create and implement the strategies for containing, eradicating, and recovering from the incident. Technical leads often collaborate with other teams to ensure their incident response priorities align with business priorities, such as reducing disruptions for customers or returning to normal operations.

Incident coordinator

Responding to an incident also requires cross-collaboration with nonsecurity professionals. CSIRTs will often consult with and leverage the expertise of members from external departments. The job of the incident coordinator is to coordinate with the relevant departments during a security incident. By doing so, the lines of communication are open and clear, and all personnel are made aware of the incident status. Incident coordinators can also be found in other teams, like the SOC.

Other roles

Depending on the organization, many other roles can be found in a CSIRT, including a dedicated communications lead, a legal lead, a planning lead, and more.

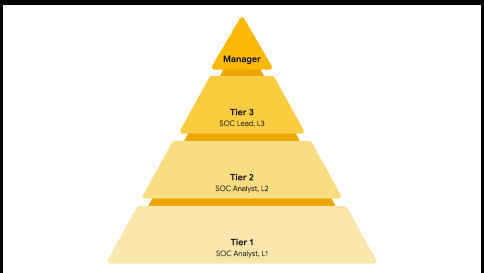
Note: Teams, roles, responsibilities, and organizational structures can differ for each company. For example, some different job titles for incident coordinator include incident commander and incident manager.

Security operations center

A security operations center (SOC) is an organizational unit dedicated to monitoring networks, systems, and devices for security threats or attacks. Structurally, a SOC (usually pronounced "sock") often exists as its own separate unit or within a CSIRT. You may be familiar with the term *blue team*, which refers to the security professionals who are responsible for defending against all security threats and attacks at an organization. A SOC is involved in various types of blue team activities, such as network monitoring, analysis, and response to incidents.

SOC organization

A SOC is composed of SOC analysts, SOC leads, and SOC managers. Each role has its own respective responsibilities. SOC analysts are grouped into three different tiers.



Tier 1 SOC analyst

The first tier is composed of the least experienced SOC analysts who are known as level 1s (L1s). They are responsible for:

- Monitoring, reviewing, and prioritizing alerts based on criticality or severity
- Creating and closing alerts using ticketing systems
- Escalating alert tickets to Tier 2 or Tier 3

Tier 2 SOC analyst

The second tier comprises the more experienced SOC analysts, or level 2s (L2s). They are responsible for:

- Receiving escalated tickets from L1 and conducting deeper investigations
- Configuring and refining security tools
- Reporting to the SOC Lead

Tier 3 SOC lead

The third tier of a SOC is composed of the SOC leads, or level 3s (L3s). These highly experienced professionals are responsible for:

- Managing the operations of their team
- Exploring methods of detection by performing advanced detection techniques, such as malware and forensics analysis
- Reporting to the SOC manager

SOC manager

The SOC manager is at the top of the pyramid and is responsible for:

- Hiring, training, and evaluating the SOC team members
- Creating performance metrics and managing the performance of the SOC team
- Developing reports related to incidents, compliance, and auditing
- Communicating findings to stakeholders such as executive management

Other roles

SOCs can also contain other specialized roles such as:

- Forensic investigators: Forensic investigators are commonly L2s and L3s who collect, preserve, and analyze digital evidence related to security incidents to determine what happened.
- Threat hunters: Threat hunters are typically L3s who work to detect, analyze, and defend against new and advanced cybersecurity threats using threat intelligence.

Note: Just like CSIRTs, the organizational structure of a SOC can differ depending on the organization.

Resources for more information

Here are some resources if you'd like to learn more about SOC organization or explore other incident response roles:

- The security operations ecosystem
- Cyber career pathways tool
- Detection and Response at Google: Episode 2 of the Hacking Google series of videos

Intrusion Detection Systems

Intrusion Detection Systems (IDS)

- An IDS monitors system and network activity to identify potential intrusions.
- It works by analyzing system information for abnormal activities and sending alerts when something unusual is detected.

Intrusion Prevention Systems (IPS)

- An IPS has all the capabilities of an IDS but can also take action to stop intrusions.
- It monitors for intrusions and actively intervenes to prevent unauthorized access or malicious activity.

Common Tools

- Many tools combine the functionalities of both IDS and IPS.
- Examples of popular tools include Snort, Zeek, Kismet, Sagan, and Suricata.

Why you need detection tools

Detection tools work similarly to home security systems. Whereas home security systems monitor and protect homes against intrusion, cybersecurity detection tools help organizations protect their networks and systems against unwanted and unauthorized access. For organizations to protect their systems from security threats or attacks, they must be made aware when there is any indication of an intrusion. Detection tools make security professionals aware of the activity happening on a network or a system. The tools do this by continuously monitoring networks and systems for any suspicious activity. Once something unusual or suspicious is detected, the tool triggers an alert that notifies the security professional to investigate and stop the possible intrusion.

Detection tools

As a security analyst, you'll likely encounter IDS, IPS, and EDR detection tools at some point, but it's important to understand the differences between them. Here is a comparison chart for quick reference:

Capability	IDS	IPS	EDR
Detects malicious activity	✓	✓	✓
Prevents intrusions	N/A	✓	✓
Logs activity	✓	✓	✓
Generates alerts	✓	✓	✓
Performs behavioral analysis	N/A	N/A	✓

Overview of IDS tools

An intrusion detection system (IDS) is an application that monitors system activity and alerts on possible intrusions. An IDS provides continuous monitoring of network events to help protect against security threats or attacks. The goal of an IDS is to detect potential malicious activity and generate an alert once such activity is detected. An IDS does *not* stop or prevent the activity. Instead, security professionals will investigate the alert and act to stop it, if necessary.

For example, an IDS can send out an alert when it identifies a suspicious user login, such as an unknown IP address logging into an application or a device at an unusual time. But, an IDS will not stop or prevent any further actions, like blocking the suspicious user login.

Examples of IDS tools include Zeek, Suricata, Snort®, and Sagan.

Detection categories

As a security analyst, you will investigate alerts that an IDS generates. There are four types of detection categories you should be familiar with:

1. **A true positive** is an alert that correctly detects the presence of an attack.
2. **A true negative** is a state where there is no detection of malicious activity. This is when no malicious activity exists and no alert is triggered.
3. **A false positive** is an alert that incorrectly detects the presence of a threat. This is when an IDS identifies an activity as malicious, but it isn't. False positives are an inconvenience for security teams because they spend time and resources investigating an illegitimate alert.
4. **A false negative** is a state where the presence of a threat is not detected. This is when malicious activity happens but an IDS fails to detect it. False negatives are dangerous because security teams are left unaware of legitimate attacks that they can be vulnerable to.

Overview of IPS tools

An intrusion prevention system (IPS) is an application that monitors system activity for intrusive activity and takes action to stop the activity. An IPS works similarly to an IDS. But, IPS monitors system activity to detect and alert on intrusions, *and* it also takes action to *prevent* the activity and minimize its effects. For example, an IPS can send an alert and modify an access control list on a router to block specific traffic on a server.

Note: Many IDS tools can also operate as an IPS. Tools like Suricata, Snort, and Sagan have both IDS and IPS capabilities.

Overview of EDR tools

Endpoint detection and response (**EDR**) is an application that monitors an endpoint for malicious activity. EDR tools are installed on endpoints. Remember that an endpoint is any device connected on a network. Examples include end-user devices, like computers, phones, tablets, and more.

EDR tools monitor, record, and analyze endpoint system activity to identify, alert, and respond to suspicious activity. Unlike IDS or IPS tools, EDRs collect endpoint activity data and perform *behavioral analysis* to identify threat patterns happening on an endpoint. Behavioral analysis uses the power of machine learning and artificial intelligence to analyze system behavior to identify malicious or unusual activity. EDR tools also use *automation* to stop attacks without the manual intervention of security professionals. For example, if an EDR detects an unusual process starting up on a user's workstation that normally is not used, it can automatically block the process from running.

Tools like Open EDR®, Bitdefender™ Endpoint Detection and Response, and FortiEDR™ are examples of EDR tools.

Overview of SIEM Technology

SIEM advantages

SIEM tools collect and manage security-relevant data that can be used during investigations. This is important because SIEM tools provide awareness about the activity that occurs between devices on a network. The information SIEM tools provide can help security teams quickly investigate and respond to security incidents. SIEM tools have many advantages that can help security teams effectively respond to and manage incidents. Some of the advantages are:

- **Access to event data**: SIEM tools provide access to the event and activity data that happens on a network, including real-time activity. Networks can be connected to hundreds of different systems and devices. SIEM tools have the ability to ingest all of this data so that it can be accessed.
- **Monitoring, detecting, and alerting**: SIEM tools continuously monitor systems and networks in real-time. They then analyze the collected data using detection rules to detect malicious activity. If an activity matches the rule, an alert is generated and sent out for security teams to assess.
- **Log storage**: SIEM tools can act as a system for data retention, which can provide access to historical data. Data can be kept or deleted after a period depending on an organization's requirements.

The SIEM process

The SIEM process consists of three critical steps:

1. Collect and aggregate data
2. Normalize data
3. Analyze data

By understanding these steps, organizations can utilize the power of SIEM tools to gather, organize, and analyze security event data from different sources. Organizations can later use this information to improve their ability to identify and mitigate potential threats.

Collect and aggregate data

SIEM tools require data for them to be effectively used. During the first step, the SIEM collects event data from various sources like firewalls, servers, routers, and more. This data, also known as logs, contains event details like timestamps, IP addresses, and more. Logs are a record of events that occur within an organization's systems. After all of this log data is collected, it gets aggregated in one location. **Aggregation refers to the process of consolidating log data into a centralized place**. Through collection and aggregation, SIEM tools eliminate the need for manually reviewing and analyzing event data by accessing individual data sources. Instead, all event data is accessible in one location—the SIEM.

Parsing can occur during the first step of the SIEM process when data is collected and aggregated. *Parsing* maps data according to their fields and their corresponding values. For example, the following log example contains fields with values. At first, it might be difficult to interpret information from this log based on its format:

```
April 3 11:01:21 server sshd[1088]: Failed password for user nuhara from 218.124.14.105 port 5023
```

In a parsed format, the fields and values are extracted and paired making them easier to read and interpret:

- host = `server`
- process = `sshd`
- source_user = `nuhara`
- source ip = `218.124.14.105`
- source port = `5023`

Normalize data

SIEM tools collect data from many different sources. This data must be transformed into a single format so that it can be easily processed by the SIEM. However, each data source is different and data can be formatted in many different ways. For example, a firewall log can be formatted differently than a server log.

Collected event data should go through the process of normalization. *Normalization* converts data into a standard, structured format that is easily searchable.

Analyze data

After log data has been collected, aggregated, and normalized, the SIEM must do something useful with all of the data to enable security teams to investigate threats. During this final step in the process, SIEM tools analyze the data. Analysis can be done with some type of detection logic such as a set of rules and conditions. SIEM tools then apply these rules to the data, and if any of the log activity matches a rule, alerts are sent out to cybersecurity teams.

Note: A part of the analysis process includes correlation. *Correlation* involves the comparison of multiple log events to identify common patterns that indicate potential security threats.

SIEM tools

There are many SIEM tools. The following are some SIEM tools commonly used in the cybersecurity industry:

- AlienVault® OSSIM™
- Chronicle
- Elastic
- Exabeam
- IBM QRadar® Security Intelligence Platform
- LogRhythm
- Splunk

Maintain Awareness of Network Traffic Flows

Understanding Network Traffic

- Definition: Network traffic refers to the amount of data moving across a network, while network data is the information transmitted between devices.
- Volume: Large organizations can generate a significant volume of network traffic due to numerous employees sending and receiving data.

Identifying Abnormal Network Behavior

- Normal vs. Abnormal: Just like traffic on a road, understanding typical network traffic patterns helps identify unusual activity.
- Indicators of Compromise (IoC): Abnormal traffic can signal a potential security incident, such as data exfiltration.

Detecting Data Exfiltration

- Data Exfiltration: This is the unauthorized transfer of data from a system, often used by attackers to steal sensitive information.
- Detection: Observing large volumes of outbound traffic from a host can be an indicator of data exfiltration, requiring further investigation.

Know your network

As you've learned, networks connect devices, and devices then communicate and exchange data using network protocols. Network communications provide information about connections such as source and destination IP addresses, amount of data transferred, date and time, and more. This information can be valuable for security professionals when developing a baseline of normal or expected behavior.

A baseline is a reference point that's used for comparison. You've probably encountered or used baselines at some point. For example, a grocery amount for a personal budget is an example of a baseline that can be used to help identify any patterns or changes in spending habits. In security, baselines help establish a standard of expected or normal behavior for systems, devices, and networks. Essentially, by knowing the baseline of *normal* network behavior, you'll be better able to identify *abnormal* network behavior.

Monitor your network

Once you've determined a baseline, you can monitor a network to identify any deviations from that baseline. Monitoring involves examining network components to detect unusual activities, such as large and unusual data transfers. Here are examples of network components that can be monitored to detect malicious activity:

Flow analysis

Flow refers to the movement of network communications and includes information related to packets, protocols, and ports. Packets can travel to ports, which receive and transmit communications. Ports are often, but not always, associated with network protocols. For example, port 443 is commonly used by HTTPS which is a protocol that provides website traffic encryption.

However, malicious actors can use protocols and ports that are not commonly associated to maintain communications between the compromised system and their own machine. These communications are what's known as command and control (**C2**), which are the techniques used by malicious actors to maintain communications with compromised systems.

For example, malicious actors can use HTTPS protocol over port 8088 as opposed to its commonly associated port 443 to communicate with compromised systems. Organizations must know which ports should be open and approved for connections, and watch out for any mismatches between ports and their associated protocols.

Packet payload information

Network packets contain components related to the transmission of the packet. This includes details like source and destination IP address, and the packet payload information, which is the actual data that's transmitted. Often, this data is encrypted and requires decryption for it to be readable. Organizations can monitor the payload information of packets to uncover unusual activity, such as sensitive data transmitting outside of the network, which could indicate a possible data exfiltration attack.

Temporal patterns

Network packets contain information relating to time. This information is useful in understanding time patterns. For example, a company operating in North America experiences bulk traffic flows between 9 a.m. to 5 p.m., which is the baseline of normal network activity. If large volumes of traffic are suddenly outside of the normal hours of network activity, then this is considered *off baseline* and should be investigated.

Through network monitoring, organizations can promptly detect network intrusions and work to prevent them from happening by securing network components.

Protect your network

In this program, you've learned about security operations centers (SOC) and their role in monitoring systems against security threats and attacks. Organizations may deploy a network operations center (**NOC**), which is an organizational unit that monitors the performance of a network and responds to any network disruption, such as a network outage. While a SOC is focused on maintaining the security of an organization through detection and response, a NOC is responsible for maintaining network performance, availability, and uptime.

Security analysts monitor networks to identify any signs of potential security incidents known as indicators of compromise (**IoC**) and protect networks from threats or attacks. To do this, they must understand the environment that network communications travel through so that they can identify deviations in network traffic.

Network monitoring tools

Network monitoring can be automated or performed manually. Some common network monitoring tools can include:

- Intrusion detection systems (IDS) monitor system activity and alert on possible intrusions. An IDS will detect and alert on the deviations you've configured it to detect. Most commonly, IDS tools will monitor the content of packet payload to detect patterns associated with threats such as malware or phishing attempts.

- Network protocol analyzers, also known as packet sniffers, are tools designed to capture and analyze data traffic within a network. They can be used to analyze network communications manually in detail. Examples include tools such as tcpdump and Wireshark, which can be used by security professionals to record network communications through packet captures. Packet captures can then be investigated to identify potentially malicious activity.

Resources

- If you would like to learn more about network components organizations can monitor, check out network traffic - MITRE ATT&CK®
- Attackers can leverage different techniques to exfiltrate data, should you like to learn more, check out data exfiltration techniques - MITRE ATT&CK®

Packets and Packet Captures

Packets

Previously in the program, you learned that a data packet is a basic unit of information that travels from one device to another within a network. Detecting network intrusions begins at the packet level. That's because packets form the basis of information exchange over a network. Each time you perform an activity on the internet—like visiting a website—packets are sent and received between your computer and the website's server. These packets are what help transmit information through a network. For example, when uploading an image to a website, the data gets broken up into multiple packets, which then get routed to the intended destination and reassembled upon delivery.

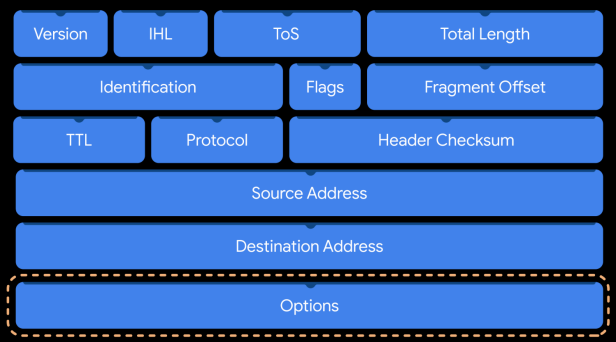
In cybersecurity, packets provide valuable information that helps add context to events during investigations. Understanding the transfer of information through packets will not only help you develop insight on network activity, it will also help you identify abnormalities and better defend networks from attacks.

Packets contain three components: the header, the payload, and the footer. Here's a description of each of these components.

Header

Packets begin with the most essential component: the header. Packets can have several headers depending on the protocols used such as an Ethernet header, an IP header, a TCP header, and more. Headers provide information that's used to route packets to their destination. This includes information about the source and destination IP addresses, packet length, protocol, packet identification numbers, and more.

Here is an IPv4 header with the information it provides:



Payload

The payload component directly follows the header and contains the actual data being delivered. Think back to the example of uploading an image to a website; the payload of this packet would be the image itself.

Footer

The footer, also known as the trailer, is located at the end of a packet. The Ethernet protocol uses footers to provide error-checking information to determine if data has been corrupted. In addition, Ethernet network packets that are analyzed might not display footer information due to network configurations.

Note: Most protocols, such as the Internet Protocol (IP), *do not* use footers.

Network protocol analyzers

Network protocol analyzers (packet sniffers) are tools designed to capture and analyze data traffic within a network. Examples of network protocol analyzers include tcpdump, Wireshark, and TShark.

Beyond their use in security as an investigative tool used to monitor networks and identify suspicious activity, network protocol analyzers can be used to collect network statistics, such as bandwidth or speed, and troubleshoot network performance issues, like slowdowns.

Network protocol analyzers can also be used for malicious purposes. For example, malicious actors can use network protocol analyzers to capture packets containing sensitive data, such as account login information.

Here's a network diagram illustrating how packets get transmitted from a sender to the receiver. A network protocol analyzer is placed in the middle of the communications to capture the data packets that travel over the wire.

How network protocol analyzers work

Network protocol analyzers use both software and hardware capabilities to capture network traffic and display it for security analysts to examine and analyze. Here's how:

1. First, packets must be collected from the network via the Network Interface Card (NIC), which is hardware that connects computers to a network, like a router. NICs receive and transmit network traffic, but by default they only listen to network traffic that's addressed to them. To capture all network traffic that is sent over the network, a NIC must be switched to a mode that has access to all visible network data packets. In wireless interfaces this is often referred to as monitoring mode, and in other systems it may be called promiscuous mode. This mode enables the NIC to have access to all visible network data packets, but it won't help analysts access all packets across a network. A network protocol analyzer must be positioned in an appropriate network segment to access all traffic between different hosts.
2. The network protocol analyzer collects the network traffic in raw binary format. Binary format consists of 0s and 1s and is not as easy for humans to interpret. The network protocol analyzer takes the binary and converts it so that it's displayed in a human-readable format, so analysts can easily read and understand the information.

Note: Enabling promiscuous can expose your device to potential attacks because it allows sensitive information like passwords and other confidential data to be captured. It's important to use tools that operate in promiscuous mode responsibly and with caution.

Capturing packets

Packet sniffing is the practice of capturing and inspecting data packets across a network. A packet capture (p-cap) is a file containing data packets intercepted from an interface or network. Packet captures can be viewed and further analyzed using network protocol analyzers. For example, you can filter packet captures to only display information that's most relevant to your investigation, such as packets sent from a specific IP address.

Note: Using network protocol analyzers to intercept and examine private network communications without permission is considered illegal in many places.

P-cap files can come in many formats depending on the packet capture library that's used. Each format has different uses and network tools may use or support specific packet capture file formats by default. You should be familiar with the following libraries and formats:

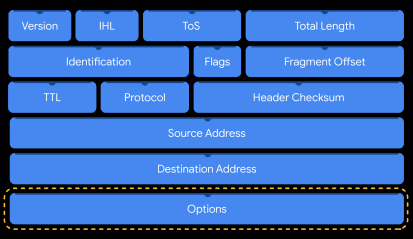
1. **Libpcap** is a packet capture library designed to be used by Unix-like systems, like Linux and MacOS®. Tools like tcpdump use Libpcap as the default packet capture file format.
2. WinPcap is an open-source packet capture library designed for devices running Windows operating systems. It's considered an older file format and isn't predominantly used.
3. **Npcap** is a library designed by the port scanning tool Nmap that is commonly used in Windows operating systems.
4. **PCAPng** is a modern file format that can simultaneously capture packets and store data. Its ability to do both explains the "ng," which stands for "next generation."

Investigate Packet Details

Internet Protocol (IP)

Packets form the foundation of data exchange over a network, which means that detection begins at the packet level. The Internet Protocol (IP) includes a set of standards used for routing and addressing data packets as they travel between devices on a network. IP operates as the foundation for all communications over the internet.

IP ensures that packets reach their destinations. There are two versions of IP that you will find in use today: IPv4 and IPv6. Both versions use different headers to structure packet information.



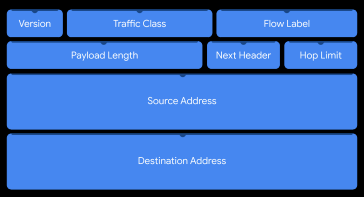
IPv4

IPv4 is the most commonly used version of IP. There are thirteen fields in the header:

- **Version**: This field indicates the IP version. For an IPv4 header, IPv4 is used.
- Internet Header Length (**IHL**): This field specifies the length of the IPv4 header including any Options.
- Type of Service (**ToS**): This field provides information about packet priority for delivery.
- **Total Length**: This field specifies the total length of the entire IP packet including the header and the data.
- **Identification**: Packets that are too large to send are fragmented into smaller pieces. This field specifies a unique identifier for fragments of an original IP packet so that they can be reassembled once they reach their destination.
- **Flags**: This field provides information about packet fragmentation including whether the original packet has been fragmented and if there are more fragments in transit.
- **Fragment Offset**: This field is used to identify the correct sequence of fragments.

- Time to Live (**TTL**): This field limits how long a packet can be circulated in a network, preventing packets from being forwarded by routers indefinitely.
- **Protocol**: This field specifies the protocol used for the data portion of the packet.
- **Header Checksum**: This field specifies a checksum value which is used for error-checking the header.
- **Source Address**: This field specifies the source address of the sender.
- **Destination Address**: This field specifies the destination address of the receiver.
- **Options**: This field is optional and can be used to apply security options to a packet.

IPv6



IPv6 adoption has been increasing because of its large address space. There are eight fields in the header:

- **Version**: This field indicates the IP version. For an IPv6 header, IPv6 is used.
- **Traffic Class**: This field is similar to the IPv4 Type of Service field. The Traffic Class field provides information about the packet's priority or class to help with packet delivery.
- **Flow Label**: This field identifies the packets of a flow. A flow is the sequence of packets sent from a specific source.
- **Payload Length**: This field specifies the length of the data portion of the packet.
- **Next Header**: This field indicates the type of header that follows the IPv6 header such as TCP.
- **Hop Limit**: This field is similar to the IPv4 Time to Live field. The Hop Limit limits how long a packet can travel in a network before being discarded.
- **Source Address**: This field specifies the source address of the sender.
- **Destination Address**: This field specifies the destination address of the receiver.

Header fields contain valuable information for investigations and tools like Wireshark help to display these fields in a human-readable format.

Wireshark

Wireshark is an open-source network protocol analyzer. It uses a graphical user interface (GUI), which makes it easier to visualize network communications for packet analysis purposes. Wireshark has many features to explore that are beyond the scope of this course. You'll focus on how to use basic filtering to isolate network packets so that you can find what you need.

Display filters

Wireshark's display filters let you apply filters to packet capture files. This is helpful when you are inspecting packet captures with large volumes of information. Display filters will help you find specific information that's most relevant to your investigation. You can filter packets based on information such as protocols, IP addresses, ports, and virtually any other property found in a packet. Here, you'll focus on display filtering syntax and filtering for protocols, IP addresses, and ports.

Comparison operators

You can use different comparison operators to locate specific header fields and values. Comparison operators can be expressed using either abbreviations or symbols. For example, this filter using the `==` equal symbol in this filter `ip.src == 8.8.8.8` is identical to using the `eq` abbreviation in this filter `ip.src eq 8.8.8.8`.

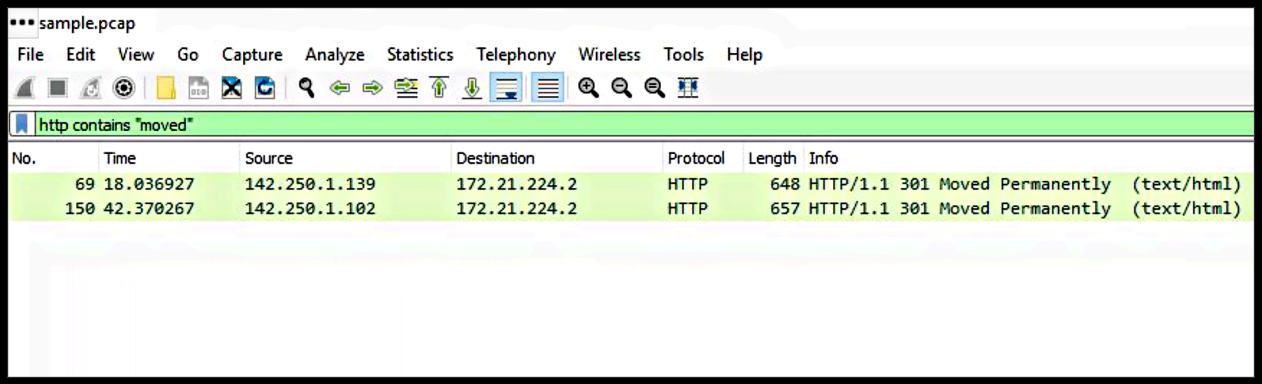
This table summarizes the different types of comparison operators you can use for display filtering.

Operator type	Symbol	Abbreviation
Equal	`==`	eq
Not equal	`!=`	ne
Greater than	`>`	gt
Less than	`<`	lt
Greater than or equal to	`>=`	ge
Less than or equal to	`<=`	le

Pro tip: You can combine comparison operators with Boolean logical operators like `and` and `or` to create complex display filters. Parentheses can also be used to group expressions and to prioritize search terms.

Contains operator

The `contains` operator is used to filter packets that contain an exact match of a string of text. Here is an example of a filter that displays all HTTP streams that match the keyword `"moved"`.

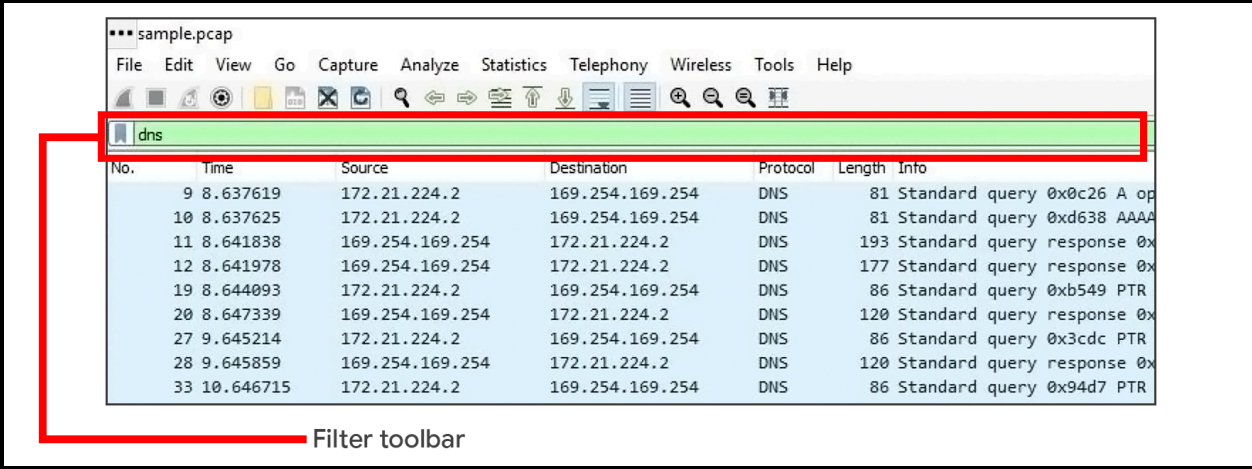


Matches operator

The `matches` operator is used to filter packets based on the regular expression (regex) that's specified. Regular expression is a sequence of characters that forms a pattern. You'll explore more about regular expressions later in this program.

Filter toolbar

You can apply filters to a packet capture using Wireshark's filter toolbar. In this example, `dns` is the applied filter, which means Wireshark will only display packets containing the DNS protocol.



Pro tip: Wireshark uses different colors to represent protocols. You can customize colors and create your own filters.

Filter for protocols

Protocol filtering is one of the simplest ways you can use display filters. You can simply enter the name of the protocol to filter. For example, to filter for DNS packets simply type `dns` in the filter toolbar. Here is a list of some protocols you can filter for:

- dns
- http
- ftp
- ssh
- arp
- telnet
- icmp

Filter for an IP address

You can use display filters to locate packets with a specific IP address.

For example, if you would like to filter packets that contain a specific IP address use `ip.addr`, followed by a space, the equal `==` comparison operator, and the IP address. Here is an example of a display filter that filters for the IP address `172.21.224.2`:

```
ip.addr == 172.21.224.2
```

To filter for packets originating from a specific source IP address, you can use the `ip.src` filter. Here is an example that looks for the `10.10.10.10` source IP address:

```
ip.src == 10.10.10.10
```

To filter for packets delivered to a specific destination IP address, you can use the `ip.dst` filter. Here is an example that searches for the `4.4.4.4` destination IP address:

```
ip.dst == 4.4.4.4
```

Filter for a MAC address

You can also filter packets according to the Media Access Control (MAC) address. As a refresher, a MAC address is a unique alphanumeric identifier that is assigned to each physical device on a network.

Here's an example:

```
eth.addr == 00:70:f4:23:18:c4
```

Filter for ports

Port filtering is used to filter packets based on port numbers. This is helpful when you want to isolate specific types of traffic. DNS traffic uses TCP or UDP port 53 so this will list traffic related to DNS queries and responses only.

For example, if you would like to filter for a UDP port:

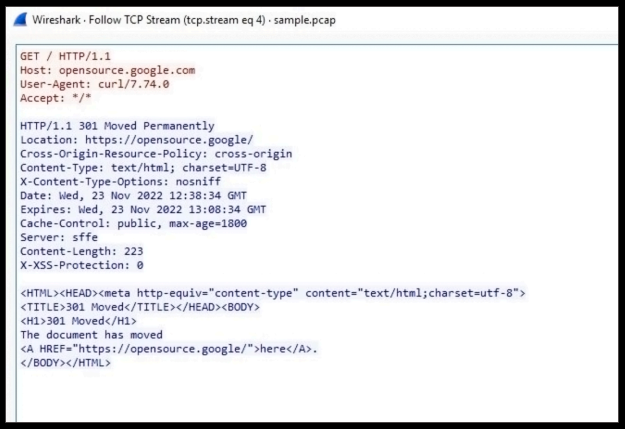
```
udp.port == 53
```

Likewise, you can filter for TCP ports as well:

```
tcp.port == 25
```

Follow streams

Wireshark provides a feature that lets you filter for packets specific to a protocol and view streams. A stream or conversation is the exchange of data between devices using a protocol. Wireshark reassembles the data that was transferred in the stream in a way that's simple to read.



Following a protocol stream is useful when trying to understand the details of a conversation. For example, you can examine the details of an HTTP conversation to view the content of the exchanged request and response messages.

Resources

- To learn more about Wireshark's full features and capabilities, explore the Wireshark official user guide.

Overview of tcpdump

What is tcpdump?

Tcpdump is a command-line network protocol analyzer. Recall that a command-line interface (CLI) is a text-based user interface that uses commands to interact with the computer.

Tcpdump is used to capture network traffic. This traffic can be saved to a packet capture (p-cap), which is a file containing data packets intercepted from an interface or network. The p-cap file can be accessed, analyzed, or shared at a later time. Analysts use tcpdump for a variety of reasons, from troubleshooting network issues to identifying malicious activity. Tcpdump comes pre-installed in many Linux distributions and can also be installed on other Unix-based operating systems such as macOS®.

Note: It's common for network traffic to be encrypted, which means data is encoded and unreadable. Inspecting the network packets might require decrypting the data using the appropriate private keys.

Capturing packets with tcpdump

Previously in this program, you learned that a Linux root user (or superuser) has elevated privileges to modify the system. You also learned that the sudo command temporarily grants elevated permissions to specific users in Linux. Like many other packet sniffing tools, you'll need to have administrator-level privileges to capture network traffic using tcpdump. This means you will need to either be logged in as the root user or have the ability to use the sudo command. Here is a breakdown of the tcpdump syntax for capturing packets:

```
sudo tcpdump [-i interface] [option(s)] [expression(s)]
```

- The `sudo tcpdump` command begins running tcpdump using elevated permissions as sudo.
- The `-i` parameter specifies the network interface to capture network traffic. You must specify a network interface to capture from to begin capturing packets. For example, if you specify `-i any` you'll sniff traffic from all network interfaces on the system.
- The `option(s)` are optional and provide you with the ability to alter the execution of the command. The `expression(s)` are a way to further filter network traffic packets so that you can isolate network traffic. You'll learn more about `option(s)` and `expression(s)` in the next section.

Note: Before you can begin capturing network traffic, you must identify which network interface you'll want to use to capture packets from. You can use the `-D` flag to list the network interfaces available on a system.

Options

With tcpdump, you can apply options, also known as flags, to the end of commands to filter network traffic. Short options are abbreviated and represented by a hyphen and a single character like `-i`. Long options are spelled out using a double hyphen like `--interface`. Tcpdump has over fifty options that you can explore using the manual page. Here, you'll examine a couple of essential tcpdump options including how to write and read packet capture files.

Note: Options are case sensitive. For example, a lowercase **`-w`** is a separate option with a different use than the option with an uppercase **`-W`**.

Note: tcpdump options that are written using short options can be written with or without a space between the option and its value. For example, **`sudo tcpdump -i any -c 3`** and **`sudo tcpdump -iany -c3`** are equivalent commands.

-w

Using the **`-w`** flag, you can write or save the sniffed network packets to a packet capture file instead of just printing it out in the terminal. This is very useful because you can refer to this saved file for later analysis. In this command, tcpdump is capturing network traffic from all network interfaces and saving it to a packet capture file named **`packetcapture.pcap`**:

```
sudo tcpdump -i any -w packetcapture.pcap
```

-r

Using the **`-r`** flag, you can read a packet capture file by specifying the file name as a parameter. Here is an example of a tcpdump command that reads a file called **`packetcapture.pcap`**:

```
sudo tcpdump -r packetcapture.pcap
```

-v

As you've learned, packets contain a lot of information. By default, tcpdump will not print out all of a packet's information. This option, which stands for verbose, lets you control how much packet information you want tcpdump to print out.

There are three levels of verbosity you can use depending on how much packet information you want tcpdump to print out. The levels are **`-v`**, **`-vv`**, and **`-vvv`**. The level of verbosity increases with each added v. The verbose option can be helpful if you're looking for packet information like the details of a packet's IP header fields. Here's an example of a tcpdump command that reads the **`packetcapture.pcap`** file with verbosity:

```
sudo tcpdump -r packetcapture.pcap -v
```

-c

The **`-c`** option stands for count. This option lets you control how many packets tcpdump will capture. For example, specifying **`-c 1`** will only print out one single packet, whereas **`-c 10`** prints out 10 packets. This example is telling tcpdump to only capture the first three packets it sniffs from **`any`** network interface:

```
sudo tcpdump -i any -c 3
```

-n

By default, tcpdump will perform name resolution. This means that tcpdump automatically converts IP addresses to names. It will also resolve ports to commonly associated services that use these ports. This can be problematic because tcpdump isn't always accurate in name resolution. For example, tcpdump can capture traffic from port 80 and automatically translates port 80 to HTTP in the output. However, this is misleading because port 80 isn't always going to be using HTTP; it could be using a different protocol.

Additionally, name resolution uses what's known as a reverse DNS lookup. A reverse DNS lookup is a query that looks for the domain name associated with an IP address. If you perform a reverse DNS lookup on an attacker's system, they might be alerted that you are investigating them through their DNS records.

Using the `-n` flag disables this automatic mapping of numbers to names and is considered to be best practice when sniffing or analyzing traffic. Using `-n` will not resolve hostnames, whereas `-nn` will not resolve *both* hostnames or ports. Here's an example of a tcpdump command that reads the `packetcapture.pcap` file with verbosity and disables name resolution:

```
sudo tcpdump -r packetcapture.pcap -v -n
```

Pro tip: **You can combine options together. For example, `-v` and `-n` can be combined as `-vn`. But, if an option accepts a parameter right after it like `-c 1` or `-r capture.pcap` then you can't combine other options to it**.

Expressions

Using filter expressions in tcpdump commands is also optional, but knowing how and when to use filter expressions can be helpful during packet analysis. There are many ways to use filter expressions.

If you want to specifically search for network traffic by protocol, you can use filter expressions to isolate network packets. For example, you can filter to find only IPv6 traffic using the filter expression `ip6`.

You can also use boolean operators like `and`, `or`, or `not` to further filter network traffic for specific IP addresses, ports, and more. The example below reads the `packetcapture.pcap` file and combines two expressions `ip and port 80` using the `and` boolean operator:

```
sudo tcpdump -r packetcapture.pcap -n 'ip and port 80'
```

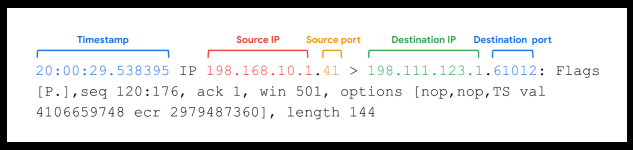
Pro tip: You can use single or double quotes to ensure that tcpdump executes all of the expressions. You can also use parentheses to group and prioritize different expressions. Grouping expressions is helpful for complex or lengthy commands. For example, the command `ip and (port 80 or port 443)` tells tcpdump to prioritize executing the filters enclosed in the parentheses before filtering for IPv4.

Interpreting output

Once you run a command to capture packets, tcpdump will print the output of the command as the sniffed packets. In the output, tcpdump prints one line of text for each packet with each line beginning with a timestamp. Here's an example of a command and output for a single TCP packet:

```
sudo tcpdump -i any -v -c 1
```

This command tells tcpdump to capture packets on `-i any` network interface. The option `-v` prints out the packet with detailed information and the option `-c 1` prints out only one packet. Here is the output of this command:



1. **Timestamp**: The output begins with the timestamp, which starts with hours, minutes, seconds, and fractions of a second.
2. **Source IP**: The packet's origin is provided by its source IP address.
3. **Source port**: This port number is where the packet originated.
4. **Destination IP**: The destination IP address is where the packet is being transmitted to.
5. **Destination port**: This port number is where the packet is being transmitted to.

The remaining output contains details of the TCP connection including flags and sequence number. The `options` information is additional packet information that the `-v` option has provided.

Resources for more information

- Learn more with tcpdump's tutorials and guides, which includes additional educational resources.
- Learn more about using expressions to filter traffic with this tcpdump tutorial by Daniel Miessler.

Module 3: Incident Investigation and Response

Cybersecurity Incident Detection Methods

Methods of detection

During the Detection and Analysis Phase of the incident response lifecycle, security teams are notified of a possible incident and work to investigate and verify the incident by collecting and analyzing data. As a reminder, detection refers to the prompt discovery of security events and analysis involves the investigation and validation of alerts.

As you've learned, an intrusion detection system (IDS) can detect possible intrusions and send out alerts to security analysts to investigate the suspicious activity. Security analysts can also use security information and event management (SIEM) tools to detect, collect, and analyze security data.

You've also learned that there are challenges with detection. Even the best security teams can fail to detect real threats for a variety of reasons. For example, detection tools can only detect what security teams configure them to monitor. If they aren't properly configured, they can fail to detect suspicious activity, leaving systems vulnerable to attack. It's important for security teams to use additional methods of detection to increase their coverage and accuracy.

Threat hunting

Threats evolve and attackers advance their tactics and techniques. Automated, technology-driven detection can be limited in keeping up to date with the evolving threat landscape. Human-driven detection like threat hunting combines the power of technology with a human element to discover hidden threats left undetected by detection tools.

Threat hunting is the proactive search for threats on a network. Security professionals use threat hunting to uncover malicious activity that was not identified by detection tools and as a way to do further analysis on detections. Threat hunting is also used to detect threats before they cause damage. For example, fileless malware is difficult for detection tools to identify. It's a form of malware that uses sophisticated evasion techniques such as hiding in memory instead of using files or applications, allowing it to bypass traditional methods of detection like signature analysis. With threat hunting, the combination of active human analysis and technology is used to identify threats like fileless malware.

Note: Threat hunting specialists are known as threat hunters. Threat hunters perform research on emerging threats and attacks and then determine the probability of an organization being vulnerable to a particular attack. Threat hunters use a combination of threat intelligence, indicators of compromise, indicators of attack, and machine learning to search for threats in an organization.

Threat intelligence

Organizations can improve their detection capabilities by staying updated on the evolving threat landscape and understanding the relationship between their environment and malicious actors. One way to understand threats is by using threat intelligence, which is evidence-based threat information that provides context about existing or emerging threats.

Threat intelligence can come from private or public sources like:

- Industry reports: These often include details about attacker's tactics, techniques, and procedures (TTP).
- Government advisories: Similar to industry reports, government advisories include details about attackers' TTP.
- Threat data feeds: Threat data feeds provide a stream of threat-related data that can be used to help protect against sophisticated attackers like advanced persistent threats (APTs). APTs are instances when a threat actor maintains unauthorized access to a system for an extended period of time. The data is usually a list of indicators like IP addresses, domains, and file hashes.

It can be difficult for organizations to efficiently manage large volumes of threat intelligence. Organizations can leverage a *threat intelligence platform* (TIP) which is an application that collects, centralizes, and analyzes threat intelligence from different sources. TIPs provide a centralized platform for organizations to identify and prioritize relevant threats and improve their security posture.

Note: Threat intelligence data feeds are best used to add context to detections. They should not drive detections completely and should be assessed before applied to an organization.

Cyber deception

Cyber deception involves techniques that deliberately deceive malicious actors with the goal of increasing detection and improving defensive strategies.

Honeypots are an example of an active cyber defense mechanism that uses deception technology. Honeypots are systems or resources that are created as decoys vulnerable to attacks with the purpose of attracting potential intruders. For example, having a fake file labeled *Client Credit Card Information - 2022* can be used to capture the activity of malicious actors by tricking them into accessing the file because it appears to be legitimate. Once a malicious actor tries to access this file, security teams are alerted.

Resources for more information

If you would like to explore more on threat hunting and threat intelligence, here are some resources:

- An informational repository about threat hunting from The ThreatHunting Project
- Research on state-sponsored hackers from Threat Analysis Group (TAG)

Ongoing Monitoring of CI/CD

Automation for Finding Threats

CI/CD pipelines help you release software faster, but they can also open up new vulnerabilities for attackers. If someone breaks into your pipeline, they could add code, steal private information, or stop your software from working. So, ongoing monitoring that automatically finds unusual pipeline activity is critical. Effective CI/CD monitoring uses automation to do more than just collect logs. It uses monitoring tools to automatically find unusual things happening in build processes, code, or deployment steps that may indicate potential security threats. When these threats are found, security teams can respond quickly and limit the damage. This automated threat detection is a main goal of strong CI/CD security.

Common Indicators of Compromise (IoCs) in CI/CD Pipelines

Understanding common CI/CD IoCs helps you monitor effectively and quickly find security incidents. Here are some examples:

- Unauthorized Code Changes:
    - Code changes from people who shouldn't be making changes.
    - Code changes made at unusual times or from unexpected locations.
    - Code changes that look suspicious, like confusing code, very large deletions without a good reason, or code that doesn't follow coding rules.
- Suspicious Deployment Patterns:
    - Deployments to unusual or unapproved systems (for example, production deployments started directly from developer branches).
    - Deployments happening at unexpected times or too often (deployments outside of planned release times).
    - Deployments started by unusual user accounts or automated accounts that shouldn't be releasing to production.
- Compromised Dependencies:
    - Finding known vulnerabilities (CVEs) in dependencies during automated checks in the CI/CD pipeline.
    - Suddenly adding new, unexpected dependencies to build settings.
    - Attempts to download dependencies from unofficial or untrusted sources.
- Unusual Pipeline Execution:
    - Pipeline steps that normally work fine suddenly failing.
    - Pipelines takeing much longer to run for no clear reason.
    - Changes in the order or way pipeline steps run without approved changes being made.
- Secrets Exposure Attempts:
    - Logs showing attempts to get to secrets from unapproved places in the pipeline.
    - Finding private secrets hardcoded in code changes (ideally prevented earlier, but monitoring can catch mistakes).

Proactive Security Through Monitoring for IoCs

Ongoing monitoring of CI/CD pipelines, focusing on automated anomaly detection and finding IoCs, makes your security stronger and more proactive. By using monitoring tools to continuously check pipeline activity for these indicators before serious damage occurs, you can:

- Respond to Incidents Quickly: Finding IoCs early helps security teams respond rapidly to potential attacks, stopping problems before attackers reach their goals.
- Limit the Damage: Responding quickly based on IoC detection reduces the possible impact of a security issue by limiting how long attackers are in the pipeline.
- Improve Threat Knowledge: Checking IoCs gives valuable information about how attackers are targeting your CI/CD, which helps improve security and threat hunting in the future.

Using Automation to Find Anomalies and IoCs

To monitor CI/CD pipelines and automatically find threats, you can use these methods:

Comprehensive Logging and Auditing

Detailed logs are the bases of monitoring. Logs provide the raw data that monitoring tools check for unusual activity and potential Indicators of Compromise (IoCs). The most common logs for finding anomalies include:

- Pipeline Execution Logs: To effectively leverage pipeline execution logs for security monitoring, specialized tools employ automated baselining techniques. These tools analyze logs from successful, typical CI/CD pipeline runs to establish a profile of normal operation. This baseline encompasses key performance indicators such as the standard duration of each pipeline stage and expected success and failure rates. By continuously monitoring execution logs and comparing them against this established baseline, the tools can automatically detect anomalous activities. Deviations from the norm, including pipeline steps exceeding typical execution times, unexpected error occurrences, or alterations in the usual step order, are flagged as potential Indicators of Compromise (IoCs), warranting further security scrutiny.
- Code Commit Logs: Keep track of code changes for each pipeline run. Unusual code changes, such as changes from people who shouldn't be making changes, changes made late at night, or changes with suspicious content (like very large deletions or confusing code), are important IoCs to monitor.
- Access Logs: Monitoring tools can learn who usually accesses CI/CD. Unusual logins, like logins from different countries, failed login attempts followed by a successful login, or login attempts to change important pipeline settings, are strong indicators of compromise.
- Deployment Logs: Tools can learn how often deployments usually happen and what those deployments look like. Unusual deployments, such as deployments at odd times or deployments to unexpected places, can be IoCs.

Security Information and Event Management (SIEM) Integration

Connecting your CI/CD logs to a SIEM tool can help automatically find anomalies at a large scale. SIEM platforms are made to:

- Automatically Find Anomalies: SIEMs use machine learning and analytics to automatically find unusual patterns in CI/CD logs, which are possible IoCs to investigate.
- Use Rules to Alert for Known IoCs: You can set up specific rules in the SIEM to find known CI/CD IoCs. For example, rules can send alerts when:
    - Detection of specific malicious file hashes (related to known CI/CD attacks) are found in build results.
    - CI/CD servers connect to known malicious command and control (C2) servers (using threat intelligence data).
    - Someone tries to download or access private secrets outside of approved pipeline steps.

Real-time Alerting and Notifications

Automated alerts make sure security teams are notified right away about unusual activity and possible IoCs, so they can respond quickly. Alerts should be set up for:

- Unusual Build Failures: Pipeline steps failing repeatedly, especially after code changes that shouldn't cause failures.
- Suspicious Code Changes (Based on Anomalies): Alerts sent by code analysis tools that find highly unusual code changes based on size, author, or confusing content.

- Attempts to Expose Secrets: Alerts sent by security tools when someone tries to access or steal secrets from unapproved parts of the pipeline.
- Unusual Network Traffic: Alerts for unusual network traffic from CI/CD servers, especially traffic going out to unknown or suspicious locations.

Performance Monitoring to Find IoAs and Discover IoCs

Performance monitoring, while mainly used to make sure things are running smoothly, can also indirectly help find IoCs. Performance issues (Indicators of Attack - IoAs) like sudden slowdowns or CI/CD servers running out of resources can lead to deeper checks that may uncover IoCs.

Continuous Vulnerability Scanning

Regularly checking the CI/CD infrastructure for weaknesses can proactively find vulnerable parts. This includes Common Vulnerabilities and Exposures (CVEs) in CI/CD tools, plugins, and containers. These weaknesses are potential IoCs. They highlight areas that need to be patched right away to prevent attacks and possible pipeline compromise.

Resources:

1. Optimizing logs for a more effective CI/CD pipeline [Best Practices]. https://coralogix.com/blog/optimizing-logs-for-a-more-effective-ci-cd-pipeline/
2. Streamline Your CI/CD: Hand-on Anomaly Detection with AI. https://www.latesttechinsights.com/2024/04/streamline-your-cicd-hands-on-anomaly.html
3. What is CI/CD? - Continuous Integration, Delivery, and Deployment. https://www.threatintelligence.com/blog/continuous-integration-continuous-delivery
4. CI/CD & DevOps Pipelines: An Introduction. https://www.splunk.com/en_us/blog/learn/ci-cd-devops-pipeline.html

Indicators of Compromise

Indicators of compromise

Indicators of compromise (IoCs) are observable evidence that suggests signs of a potential security incident. IoCs chart specific pieces of evidence that are associated with an attack, like a file name associated with a type of malware. You can think of an IoC as evidence that points to something that's already happened, like noticing that a valuable has been stolen from inside of a car.

Indicators of attack (IoA) are the series of observed events that indicate a real-time incident. IoAs focus on identifying the behavioral evidence of an attacker, including their methods and intentions.

Essentially, IoCs help to identify the *who* and *what* of an attack after it's taken place, while IoAs focus on finding the *why* and *how* of an ongoing or unknown attack. For example, observing a process that makes a network connection is an example of an IoA. The filename of the process and the IP address that the process contacted are examples of the related IoCs.

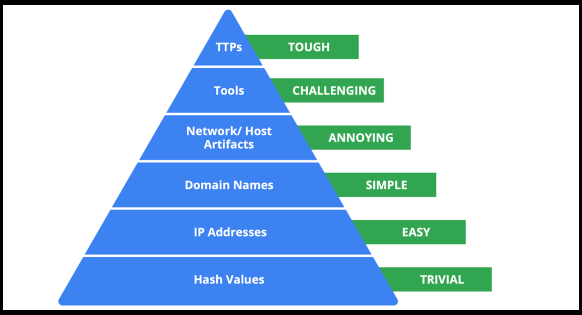
Note: Indicators of compromise are not always a confirmation that a security incident has happened. IoCs may be the result of human error, system malfunctions, and other reasons not related to security.

Pyramid of Pain

Not all indicators of compromise are equal in the value they provide to security teams. It's important for security professionals to understand the different types of indicators of compromise so that they can quickly and effectively detect and respond to them. This is why security researcher David J. Bianco created the concept of the Pyramid of Pain, with the goal of improving how indicators of compromise are used in incident detection.

The Pyramid of Pain captures the relationship between indicators of compromise and the level of difficulty that malicious actors experience when indicators of compromise are blocked by security teams. It lists the different types of indicators of compromise that security professionals use to identify malicious activity.

Each type of indicator of compromise is separated into levels of difficulty. These levels represent the "pain" levels that an attacker faces when security teams block the activity associated with the indicator of compromise. For example, blocking an IP address associated with a malicious actor is labeled as easy because malicious actors can easily use different IP addresses to work around this and continue with their malicious efforts. If security teams are able to block the IoCs located at the top of the pyramid, the more difficult it becomes for attackers to continue their attacks. Here's a breakdown of the different types of indicators of compromise found in the Pyramid of Pain.



1. Hash values: Hashes that correspond to known malicious files. These are often used to provide unique references to specific samples of malware or to files involved in an intrusion.
2. IP addresses: An internet protocol address like 192.168.1.1
3. Domain names: A web address such as www.google.com
4. Network artifacts: Observable evidence created by malicious actors on a network. For example, information found in network protocols such as User-Agent strings.
5. Host artifacts: Observable evidence created by malicious actors on a host. A host is any device that's connected on a network. For example, the name of a file created by malware.
6. Tools: Software that's used by a malicious actor to achieve their goal. For example, attackers can use password cracking tools like John the Ripper to perform password attacks to gain access into an account.
7. Tactics, techniques, and procedures (**TTPs**): This is the behavior of a malicious actor. Tactics refer to the high-level overview of the behavior. Techniques provide detailed descriptions of the behavior relating to the tactic. Procedures are highly detailed descriptions of the technique. TTPs are the hardest to detect.

Analyze Indicators of Compromise with investigative tools

You've learned about the Pyramid of Pain which describes the relationship between indicators of compromise and the level of difficulty that malicious actors experience when indicators of compromise are blocked by security teams. You also learned about different types of IoCs, but as you know, not all IoCs are equal. Malicious actors can manage to evade detection and continue compromising systems despite having their IoC-related activity blocked or limited.

For example, identifying and blocking a single IP address associated with malicious activity does not provide a broader insight on an attack, nor does it stop a malicious actor from continuing their activity. Focusing on a single piece of evidence is like fixating on a single section of a painting: You miss out on the bigger picture.

Security analysts need a way to expand the use of IoCs so that they can add context to alerts. Threat intelligence is evidence-based threat information that provides context about existing or emerging threats. By accessing additional information related to IoCs, security analysts can expand their viewpoint to observe the bigger picture and construct a narrative that helps inform their response actions.

By adding context to an IoC—for instance, identifying other artifacts related to the suspicious IP address, such as suspicious network communications or unusual processes—security teams can start to develop a detailed picture of a security incident. This context can help security teams detect security incidents faster and take a more informed approach in their response.

The power of crowdsourcing

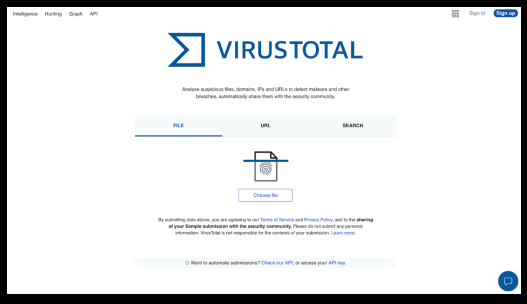
Crowdsourcing is the practice of gathering information using public input and collaboration. Threat intelligence platforms use crowdsourcing to collect information from the global cybersecurity community. Traditionally, an organization's response to incidents was performed in isolation. A security team would receive and analyze an alert, and then work to remediate it without additional insights on how to approach it. Without crowdsourcing, attackers can perform the same attacks against multiple organizations.

With crowdsourcing, organizations harness the knowledge of millions of other cybersecurity professionals, including cybersecurity product vendors, government agencies, cloud providers, and more. Crowdsourcing allows people and organizations from the global cybersecurity community to openly share and access a collection of threat intelligence data, which helps to continuously improve detection technologies and methodologies.

Examples of information-sharing organizations include Information Sharing and Analysis Centers (ISACs), which focus on collecting and sharing sector-specific threat intelligence to companies within specific industries like energy, healthcare, and others. Open-source intelligence (OSINT) is the collection and analysis of information from publicly available sources to generate usable intelligence. OSINT can also be used as a method to gather information related to threat actors, threats, vulnerabilities, and more.

This threat intelligence data is used to improve the detection methods and techniques of security products, like detection tools or anti-virus software. For example, attackers often perform the same attacks on multiple targets with the hope that one of them will be successful. Once an organization detects an attack, they can immediately publish the attack details, such as malicious files, IP addresses, or URLs, to tools like VirusTotal. This threat intelligence can then help other organizations defend against the same attack.

VirusTotal



VirusTotal is a service that allows anyone to analyze suspicious files, domains, URLs, and IP addresses for malicious content. VirusTotal also offers additional services and tools for enterprise use. This reading focuses on the VirusTotal website, which is available for free and non-commercial use.

It can be used to analyze suspicious files, IP addresses, domains, and URLs to detect cybersecurity threats such as malware. Users can submit and check artifacts, like file hashes or IP addresses, to get VirusTotal reports, which provide additional information on whether an IoC is considered malicious or not, how that IoC is connected or related to other IoCs in the dataset, and more.

1. Detection: The Detection tab provides a list of third-party security vendors and their detection verdicts on an IoC. For example, vendors can list their detection verdict as malicious, suspicious, unsafe, and more.
2. Details: The Details tab provides additional information extracted from a static analysis of the IoC. Information such as different hashes, file types, file sizes, headers, creation time, and first and last submission information can all be found in this tab.
3. Relations: The Relations tab provides related IoCs that are somehow connected to an artifact, such as contacted URLs, domains, IP addresses, and dropped files if the artifact is an executable.
4. Behavior: The Behavior tab contains information related to the observed activity and behaviors of an artifact after executing it in a controlled or sandboxed environment. This information includes tactics and techniques detected, network communications, registry and file systems actions, processes, and more.
5. Community: The Community tab is where members of the VirusTotal community, such as security professionals or researchers, can leave comments and insights about the IoC.
6. Vendors' ratio and community score: The score displayed at the top of the report is the vendors' ratio. The vendors' ratio shows how many security vendors have flagged the IoC as malicious overall. Below this score, there is also the community score, based on the inputs of the VirusTotal community. The more detections a file has and the higher its community score is, the more likely that the file is malicious.

Note: Data uploaded to VirusTotal will be publicly shared with the entire VirusTotal community. Be careful of what you submit, and make sure you do not upload personal information.

Example: Paste this hash into VirusTotal to see it in action: 287d612e29b71c90aa54947313810a25

Other tools

There are other investigative tools that can be used to analyze IoCs. These tools can also share the data that's uploaded to them to the security community.

Jotti malware scan

Jotti's malware scan is a free service that lets you scan suspicious files with several antivirus programs. There are some limitations to the number of files that you can submit.

Urlscan.io

Urlscan.io is a free service that scans and analyzes URLs and provides a detailed report summarizing the URL information.

MalwareBazaar

MalwareBazaar is a free repository for malware samples. Malware samples are a great source of threat intelligence that can be used for research purposes.

Best Practices for Effective Documentation

Documentation benefits

You've already learned about many types of security documentation, including playbooks, final reports, and more. As you've also learned, effective documentation has three benefits:

1. Transparency
2. Standardization
3. Clarity

Transparency

In security, transparency is critical for demonstrating compliance with regulations and internal processes, meeting insurance requirements, and for legal proceedings. Chain of custody is the process of documenting evidence possession and control during an incident lifecycle. Chain of custody is an example of how documentation produces transparency and an audit trail.

Standardization

Standardization through repeatable processes and procedures supports continuous improvement efforts, helps with knowledge transfer, and facilitates the onboarding of new team members. Standards are references that inform how to set policies.

You have learned how NIST provides various security frameworks that are used to improve security measures. Likewise, organizations set up their own standards to meet their business needs. An example of documentation that establishes standardization is an incident response plan, which is a document that outlines the procedures to take in each step of incident response. Incident response plans standardize an organization's response process by outlining procedures in advance of an incident. By documenting an organization's incident response plan, you create a standard that people follow, maintaining consistency with repeatable processes and procedures.

Clarity

Ideally, all documentation provides clarity to its audience. Clear documentation helps people quickly access the information they need so they can take necessary action. Security analysts are required to document the reasoning behind any action they take so that it's clear to their team why an alert was escalated or closed.

Best practices

As a security professional, you'll need to apply documentation best practices in your career. Here are some general guidelines to remember:

Know your audience

Before you start creating documentation, consider your audience and their needs. For instance, an incident summary written for a security operations center (SOC) manager will be written differently than one that's drafted for a chief executive officer (CEO). The SOC manager can understand technical security language but a CEO might not. Tailor your document to meet your audience's needs.

Be concise

You might be tasked with creating long documentation, such as a report. But when documentation is too long, people can be discouraged from using it. To ensure that your documentation is useful, establish the purpose immediately. This helps people quickly identify the objective of the document. For example, executive summaries outline the major facts of an incident at the beginning of a final report. This summary should be brief so that it can be easily skimmed to identify the key findings.

Update regularly

In security, new vulnerabilities are discovered and exploited constantly. Documentation must be regularly reviewed and updated to keep up with the evolving threat landscape. For example, after an incident has been resolved, a comprehensive review of the incident can identify gaps in processes and procedures that require changes and updates. By regularly updating documentation, security teams stay well informed and incident response plans stay updated.

The Triage Process

Triage process

Incidents can have the potential to cause significant damage to an organization. Security teams must respond quickly and efficiently to prevent or limit the impact of an incident before it becomes too late. Triage is the prioritizing of incidents according to their level of importance or urgency. The triage process helps security teams evaluate and prioritize security alerts and allocate resources effectively so that the most critical issues are addressed first.

The triage process consists of three steps:

1. Receive and assess
2. Assign priority
3. Collect and analyze

Receive and assess

During this first step of the triage process, a security analyst receives an alert from an alerting system like an intrusion detection system (IDS). You might recall that an IDS is an application that monitors system activity and alerts on possible intrusions. The analyst then reviews the alert to verify its validity and ensure they have a complete understanding of the alert.

This involves gathering as much information as possible about the alert, including details about the activity that triggered the alert, the systems and assets involved, and more. Here are some questions to consider when verifying the validity of an alert:

- Is the alert a false positive? Security analysts must determine whether the alert is a genuine security concern or a false positive, or an alert that incorrectly detects the presence of a threat.
- Was this alert triggered in the past? If so, how was it resolved? The history of an alert can help determine whether the alert is a new or recurring issue.
- Is the alert triggered by a known vulnerability? If an alert is triggered by a known vulnerability, security analysts can leverage existing knowledge to determine an appropriate response and minimize the impact of the vulnerability.
- What is the severity of the alert? The severity of an alert can help determine the priority of the response so that critical issues are quickly escalated.

Assign priority

Once the alert has been properly assessed and verified as a genuine security issue, it needs to be prioritized accordingly. Incidents differ in their impact, size, and scope, which affects the response efforts. To manage time and resources, security teams must prioritize how they respond to various incidents because not all incidents are equal. Here are some factors to consider when determining the priority of an incident:

- Functional impact: Security incidents that target information technology systems impact the service that these systems provide to its users. For example, a ransomware incident can severely impact the confidentiality, availability, and integrity of systems. Data can be encrypted or deleted, making it completely inaccessible to users. Consider how an incident impacts the existing business functionality of the affected system.
- Information impact: Incidents can affect the confidentiality, integrity, and availability of an organization's data and information. In a data exfiltration attack, malicious actors can steal sensitive data. This data can belong to third party users or organizations. Consider the effects that information compromise can have beyond the organization.
- Recoverability: How an organization recovers from an incident depends on the size and scope of the incident and the amount of resources available. In some cases, recovery might not be possible, like when a malicious actor successfully steals proprietary data and shares it publicly. Spending time, effort, and resources on an incident with no recoverability can be wasteful. It's important to consider whether recovery is possible and consider whether it's worth the time and cost.

Note: Security alerts often come with an assigned priority or severity level that classifies the urgency of the alert based on a level of prioritization.

Collect and analyze

The final step of the triage process involves the security analyst performing a comprehensive analysis of the incident. Analysis involves gathering evidence from different sources, conducting external research, and documenting the investigative process. The goal of this step is to gather enough information to make an informed decision to address it. Depending on the severity of the incident, escalation to a level two analyst or a manager might be required. Level two analysts and managers might have more knowledge on using advanced techniques to address the incident.

Benefits of triage

By prioritizing incidents based on their potential impact, you can reduce the scope of impact to the organization by ensuring a timely response. Here are some benefits that triage has for security teams:

- Resource management: Triaging alerts allows security teams to focus their resources on threats that require urgent attention. This helps team members avoid dedicating time and resources to lower priority tasks and might also reduce response time.
- Standardized approach: Triage provides a standardized approach to incident handling. Process documentation, like playbooks, help to move alerts through an iterative process to ensure that alerts are properly assessed and validated. This ensures that only valid alerts are moved up to investigate.

Business Continuity Considerations

Business continuity planning

Security teams must be prepared to minimize the impact that security incidents can have on their normal business operations. When an incident occurs, organizations might experience significant disruptions to the functionality of their systems and services. Prolonged disruption to systems and services can have serious effects, causing legal, financial, and reputational damages. Organizations can use business continuity planning so that they can remain operational during any major disruptions.

Similar to an incident response plan, a business continuity plan (**BCP**) is a document that outlines the procedures to sustain business operations during and after a significant disruption. A BCP helps organizations ensure that critical business functions can resume or can be quickly restored when an incident occurs.

Entry level security analysts aren't typically responsible for the development and testing of a BCP. However, it's important that you understand how BCPs provide organizations with a structured way to respond and recover from security incidents.

Note: Business continuity plans are not the same as *disaster recovery plans*. Disaster recovery plans are used to recover information systems in response to a major disaster. These disasters can range from hardware failure to the destruction of facilities from a natural disaster, like a flood.

Consider the impacts of ransomware to business continuity

Impacts of a security incident such as ransomware can be devastating for business operations. Ransomware attacks targeting critical infrastructure such as healthcare can have the potential to cause significant disruption. Depending on the severity of a ransomware attack, the accessibility, availability, and delivery of essential healthcare services can be impacted. For example, ransomware can encrypt data, resulting in disabled access to medical records, which prevents healthcare providers from accessing patient records. At a larger scale, security incidents that target the assets, systems, and networks of critical infrastructure can also undermine national security, economic security, and the health and safety of the public. For this reason, BCPs help to minimize interruptions to operations so that essential services can be accessed.

Recovery strategies

When an outage occurs due to a security incident, organizations must have some sort of a functional recovery plan set to resolve the issue and get systems fully operational. BCPs can include strategies for recovery that focus on returning to normal operations. Site resilience is one example of a recovery strategy.

Site resilience

Resilience is the ability to prepare for, respond to, and recover from disruptions. Organizations can design their systems to be resilient so that they can continue delivering services despite facing disruptions. An example is site resilience, which is used to ensure the availability of networks, data centers, or other infrastructure when a disruption happens. There are three types of recovery sites used for site resilience:

- Hot sites: A fully operational facility that is a duplicate of an organization's primary environment. Hot sites can be activated immediately when an organization's primary site experiences failure or disruption.
- Warm sites: A facility that contains a fully updated and configured version of the hot site. Unlike hot sites, warm sites are not fully operational and available for immediate use but can quickly be made operational when a failure or disruption occurs.
- Cold sites: A backup facility equipped with some of the necessary infrastructure required to operate an organization's site. When a disruption or failure occurs, cold sites might not be ready for immediate use and might need additional work to be operational.

Post-incident activity

The Post-incident activity phase of the NIST Incident Response Lifecycle is the process of reviewing an incident to identify areas for improvement during incident handling.

Lessons learned

After an organization has successfully contained, eradicated, and recovered from an incident, the incident comes to a close. However, this doesn't mean that the work of security professionals is complete. Incidents provide organizations and their security teams with an opportunity to learn from what happened and prioritize ways to improve the incident handling process.

This is typically done through a lessons learned meeting, also known as a post-mortem. A lessons learned meeting includes all involved parties after a major incident. Depending on the scope of an incident, multiple meetings can be scheduled to gather sufficient data. The purpose of this meeting is to evaluate the incident in its entirety, assess the response actions, and identify any areas of improvement. It provides an opportunity for an organization and its people to learn and improve, not to assign blame. This meeting should be scheduled no later than two weeks after an incident has been successfully remediated.

Not all incidents require their own lessons learned meeting; the size and severity of an incident will dictate whether the meeting is necessary. However, major incidents, such as ransomware attacks, should be reviewed in a dedicated lessons learned meeting. This meeting consists of all parties who participated in any aspect of the incident response process. Here are some examples of questions that are addressed in this meeting:

- What happened?
- What time did it happen?
- Who discovered it?
- How did it get contained?
- What were the actions taken for recovery?
- What could have been done differently?

Besides having the opportunity to learn from the incident, there are additional benefits to conducting a lessons learned meeting. For large organizations, lessons learned meetings offer a platform for team members across departments to share information and recommendations for future prevention.

Pro tip: Before a team hosts a lessons learned meeting, organizers should make sure all attendees come prepared. The meeting hosts typically develop and distribute a meeting agenda beforehand, which contains the topics of discussion and ensures that attendees are informed and prepared. Additionally, meeting roles should be assigned in advance, including a moderator to lead and facilitate discussion and a scribe to take meeting notes.

Recommendations

Lessons learned meetings provide opportunities for growth and improvement. For example, security teams can identify errors in response actions, gaps in processes and procedures, or ineffective security controls. A lessons learned meeting should result in a list of prioritized actions or actionable recommendations meant to improve an organization's incident handling processes and overall security posture. This ensures that organizations are implementing the lessons they've learned after an incident so that they are not vulnerable to experiencing the same incident in the future. Examples of changes that can be implemented include updating and improving playbook instructions or implementing new security tools and technologies.

Final report

Throughout this course, you explored the importance that documentation has in recording details during the incident response lifecycle. At a minimum, incident response documentation should describe the incident by covering the 5 W's of incident investigation: *who*, *what*, *where*, *why*, and *when*. The details that are captured during incident response are important for developing additional documents during the end of the lifecycle.

One of the most essential forms of documentation that gets created during the end of an incident is the final report. The final report provides a comprehensive review of an incident. Final reports are not standardized, and their formats can vary across organizations. Additionally, multiple final reports can be created depending on the audience it's written for.

Here are some examples of common elements found in a final report:

- Executive summary: A high-level summary of the report including the key findings and essential facts related to the incident
- Timeline: A detailed chronological timeline of the incident that includes timestamps dating the sequence of events that led to the incident
- Investigation: A compilation of the actions taken during the detection and analysis of the incident. For example, analysis of a network artifact such as a packet capture reveals information about what activities happen on a network.
- Recommendations: A list of suggested actions for future prevention

Pro tip: When writing the final report, consider the audience that you're writing the report for. Oftentimes, business executives and other non-security professionals who don't have the expertise to understand technical details will read post-incident final reports. Considering the audience when writing a final report will help you effectively communicate the most important details.

Module 4: Network Traffic and Logs using IDS and SIEM Tools

Best Practices for Log Collection and Management

Logs

Data sources such as devices generate data in the form of events. A log is a record of events that occur within an organization's systems. Logs contain log entries and each entry details information corresponding to a single event that happened on a device or system. Originally, logs served the sole purpose of troubleshooting common technology issues. For example, error logs provide information about why an unexpected error occurred and help to identify the root cause of the error so that it can be fixed. Today, virtually all computing devices produce some form of logs that provide valuable insights beyond troubleshooting.

Security teams access logs from logging receivers like SIEM tools which consolidate logs to provide a central repository for log data. Security professionals use logs to perform log analysis, which is the process of examining logs to identify events of interest. Logs help uncover the details surrounding the 5 W's of incident investigation: *who* triggered the incident, *what* happened, *when* the incident took place, *where* the incident took place, and *why* the incident occurred.

Types of logs

Depending on the data source, different log types can be produced. Here's a list of some common log types that organizations should record:

- **Network**: Network logs are generated by network devices like firewalls, routers, or switches.
- **System**: System logs are generated by operating systems like Chrome OS™, Windows, Linux, or macOS®.
- **Application**: Application logs are generated by software applications and contain information relating to the events occurring within the application such as a smartphone app.
- **Security**: Security logs are generated by various devices or systems such as antivirus software and intrusion detection systems. Security logs contain security-related information such as file deletion.
- **Authentication**: Authentication logs are generated whenever authentication occurs such as a successful login attempt into a computer.

Log details

Generally, logs contain a date, time, location, action, and author of the action. Here is an example of an authentication log:

```
Login Event [05:45:15] User1 Authenticated successfully
```

Logs contain information and can be adjusted to contain even more information. Verbose logging records additional, detailed information beyond the default log recording. Here is an example of the same log above but logged as verbose.

```
Login Event [2022/11/16 05:45:15.892673] auth_performer.cc:470 User1 Authenticated successfully from device1 (192.168.1.2)
```

Log management

Because all devices produce logs, it can quickly become overwhelming for organizations to keep track of all the logs that are generated. To get the most value from your logs, you need to choose exactly what to log, how to access it easily, and keep it secure using log management. Log management is the process of collecting, storing, analyzing, and disposing of log data.

What to log

The most important aspect of log management is choosing what to log. Organizations are different, and their logging requirements can differ too. It's important to consider which log sources are most likely to contain the most useful information depending on your event of interest. This might be configuring log sources to reduce the amount of data they record, such as excluding excessive verbosity. Some information, including but not limited to phone numbers, email addresses, and names, form personally identifiable information (PII), which requires special handling and in some jurisdictions might not be possible to be logged.

The issue with overlogging

From a security perspective, it can be tempting to log everything. This is the most common mistake organizations make. Just because it can be logged, doesn't mean it *needs* to be logged. Storing excessive amounts of logs can have many disadvantages with some SIEM tools. For example, overlogging can increase storage and maintenance costs. Additionally, overlogging can increase the load on systems, which can cause performance issues and affect usability, making it difficult to search for and identify important events.

Log retention

Organizations might operate in industries with regulatory requirements. For example, some regulations require organizations to retain logs for set periods of time and organizations can implement log retention practices in their log management policy.

Organizations that operate in the following industries might need to modify their log management policy to meet regulatory requirements:

- Public sector industries, like the Federal Information Security Modernization Act (**FISMA**)
- Healthcare industries, like the Health Insurance Portability and Accountability Act of 1996 (**HIPAA**)
- Financial services industries, such as the Payment Card Industry Data Security Standard (**PCI DSS**), the Gramm-Leach-Bliley Act (**GLBA**), and the Sarbanes-Oxley Act of 2002 (**SOX**)

Log protection

Along with management and retention, the protection of logs is vital in maintaining log integrity. It’s not unusual for malicious actors to modify logs in attempts to mislead security teams and to even hide their activity.

Storing logs in a centralized log server is a way to maintain log integrity. When logs are generated, they get sent to a dedicated server instead of getting stored on a local machine. This makes it more difficult for attackers to access logs because there is a barrier between the attacker and the log location.

Overview of Log File Formats

Common log file formats:

- JSON
- Syslog
- XML
- CSV
- CEF

JavaScript Object Notation (JSON)

JavaScript Object Notation (JSON) is a file format that is used to store and transmit data. JSON is known for being lightweight and easy to read and write. It is used for transmitting data in web technologies and is also commonly used in cloud environments. JSON syntax is derived from JavaScript syntax. If you are familiar with JavaScript, you might recognize that JSON contains components from JavaScript including:

- Key-value pairs
- Commas
- Double quotes
- Curly brackets
- Square brackets

Key-value pairs

A key-value pair is a set of data that represents two linked items: a key and its corresponding value. A key-value pair consists of a key followed by a colon, and then followed by a value. An example of a key-value pair is `"Alert": "Malware"`.

Note: For readability, it is recommended that key-value pairs contain a space before or after the colon that separates the key and value.

Commas

Commas are used to separate data. For example: `"Alert": "Malware", "Alert code": 1090, "severity": 10`.

Double quotes

Double quotes are used to enclose *text* data, which is also known as a string, for example: `"Alert": "Malware"`. Data that contains numbers *is not* enclosed in quotes, like this: `"Alert code": 1090`.

Curly brackets

Curly brackets enclose an object, which is a data type that stores data in a comma-separated list of key-value pairs. Objects are often used to describe multiple properties for a given key. JSON log entries start and end with a curly bracket. In this example, `User` is the object that contains multiple properties:

```
"User":

{

    "id": "1234",

    "name": "user",

    "role": "engineer"

}
```

Square brackets

Square brackets are used to enclose an array, which is a data type that stores data in a comma-separated ordered list. Arrays are useful when you want to store data as an ordered collection, for example: `["Administrators", "Users", "Engineering"]`.

Syslog

Syslog is a standard for logging and transmitting data. It can be used to refer to any of its three different capabilities:

1. Protocol: The syslog protocol is used to transport logs to a centralized log server for log management. It uses port 514 for plaintext logs and port 6514 for encrypted logs.
2. Service: The syslog service acts as a log forwarding service that consolidates logs from multiple sources into a single location. The service works by receiving and then forwarding any syslog log entries to a remote server.
3. Log format: The syslog log format is one of the most commonly used log formats that you will be focusing on. It is the native logging format used in Unix® systems. It consists of three components: a header, structured-data, and a message.

Syslog log example

Here is an example of a syslog entry that contains all three components: a header, followed by structured-data, and a message:

```
<236>1 2022-03-21T01:11:11.003Z virtual.machine.com evntslog - ID01 [user@32473 iut="1"
eventSource="Application" eventID="9999"]

This is a log entry!
```

Header

The header contains details like the timestamp; the hostname, which is the name of the machine that sends the log; the application name; and the message ID.

- Timestamp: The timestamp in this example is `2022-03-21T01:11:11.003Z`, where `2022-03-21` is the date in YYYY-MM-DD format. `T` is used to separate the date and the time. `01:11:11.003` is the 24-hour format of the time and includes the number of milliseconds `003`. `Z` indicates the timezone, which is Coordinated Universal Time (UTC).
- Hostname: `virtual.machine.com`
- Application: `evntslog`
- Message ID: `ID01`

Structured-data

The structured-data portion of the log entry contains additional logging information. This information is enclosed in square brackets and structured in key-value pairs. Here, there are three keys with corresponding values: `[user@32473 iut="1" eventSource="Application" eventID="9999"]`.

Message

The message contains a detailed log message about the event. Here, the message is `This is a log entry!`.

Priority (PRI)

The priority (PRI) field indicates the urgency of the logged event and is contained with angle brackets. In this example, the priority value is `<236>` . Generally, the lower the priority level, the more urgent the event is.

Note: Syslog headers can be combined with JSON, and XML formats. Custom log formats also exist.

XML (eXtensible Markup Language)

XML (eXtensible Markup Language) is a language and a format used for storing and transmitting data. XML is a native file format used in Windows systems. XML syntax uses the following:

- Tags
- Elements
- Attributes

Tags

XML uses tags to store and identify data. Tags are pairs that must contain a start tag and an end tag. The start tag encloses data with angle brackets, for example `<tag>`, whereas the end of a tag encloses data with angle brackets and a forward slash like this: `</tag>`.

Elements

XML elements include *both* the data contained inside of a tag and the tags itself. All XML entries must contain at least one root element. Root elements contain other elements that sit underneath them, known as child elements.

Here is an example:

```
<Event>

    <EventID>4688</EventID>

    <Version>5</Version>

</Event>
```

In this example, `<Event>` is the root element and contains two child elements `<EventID>` and `<Version>`. There is data contained in each respective child element.

Attributes

XML elements can also contain attributes. Attributes are used to provide additional information about elements. Attributes are included as the second part of the tag itself and must always be quoted using either single or double quotes.

For example:

```
<EventData>

    <Data Name='SubjectUserSid'>S-2-3-11-160321</Data>

    <Data Name='SubjectUserName'>JSMITH</Data>

    <Data Name='SubjectDomainName'>ADCOMP</Data>

    <Data Name='SubjectLogonId'>0x1cf1c12</Data>

    <Data Name='NewProcessId'>0x1404</Data>

</EventData>
```

In the first line for this example, the tag is `<Data>` and it uses the attribute `Name='SubjectUserSid'` to describe the data enclosed in the tag `S-2-3-11-160321`.

CSV (Comma Separated Value)

CSV (Comma Separated Value) uses commas to separate data values. In CSV logs, the position of the data corresponds to its field name, but the field names themselves might not be included in the log. It's critical to understand what fields the source device (like an IPS, firewall, scanner, etc.) is including in the log.

Here is an example:

```
2009-11-24T21:27:09.534255,ALERT,192.168.2.7, 1041,x.x.250.50,80,TCP,ALLOWED,1:2001999:9,"ET MALWARE BTGrab.com Spyware Downloading Ads",1
```

CEF (Common Event Format)

Common Event Format (CEF) is a log format that uses key-value pairs to structure data and identify fields and their corresponding values. The CEF syntax is defined as containing the following fields:

```
CEF:Version|Device Vendor|Device Product|Device Version|Signature ID|Name|Severity|Extension
```

Fields are all separated with a pipe character `|`. However, anything in the `Extension` part of the CEF log entry must be written in a key-value format. Syslog is a common method used to transport logs like CEF. When Syslog is used a timestamp and hostname will be prepended to the CEF message. Here is an example of a CEF log entry that details malicious activity relating to a worm infection:

```
Sep 29 08:26:10 host CEF:1|Security|threatmanager|1.0|100|worm successfully stopped|10|src=10.0.0.2 dst=2.1.2.2 spt=1232
```

Here is a breakdown of the fields:

- Syslog Timestamp: `Sep 29 08:26:10`
- Syslog Hostname: `host`
- Version: `CEF:1`
- Device Vendor: `Security`
- Device Product: `threatmanager`
- Device Version: `1.0`
- Signature ID: `100`
- Name: `worm successfully stopped`
- Severity: `10`
- Extension: This field contains data written as key-value pairs. There are two IP addresses, `src=10.0.0.2` and `dst=2.1.2.2`, and a source port number `spt=1232`. Extensions are not required and are optional to add.

This log entry contains details about a `Security` application called `threatmanager` that `successfully stopped a worm` from spreading from the internal network at `10.0.0.2` to the external network `2.1.2.2` through the port `1232`. A high severity level of `10` is reported.

Note: Extensions and syslog prefix are optional to add to a CEF log.

Resources for more information

- To learn more about the syslog protocol including priority levels, check out The Syslog Protocol.
- If you would like to explore generating log formats, check out this open-source test data generator tool.
- To learn more about timestamp formats, check out Date and Time on the Internet: Timestamps.

Security Monitoring with Detection Tools

- Detection systems rely on data from various sources, including logs generated by different devices.
- Telemetry is the collection and transmission of data for analysis, with packet captures being an example of network telemetry.

Intrusion Detection Systems (IDS)

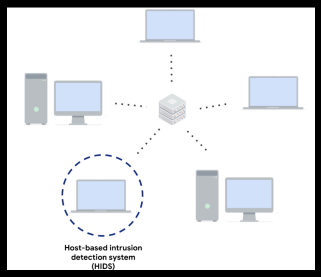
- An IDS monitors activity and alerts on possible intrusions, acting as a crucial tool for security professionals.
- Host-based intrusion detection systems (HIDS) are installed on individual devices (endpoints) to monitor their activity and detect suspicious behavior.

Network-based Intrusion Detection Systems (NIDS) and Detection Methods

- Network-based intrusion detection systems (NIDS) analyze network traffic and data, often deployed at multiple points for comprehensive visibility.
- Signature analysis is a common detection method where the IDS uses predefined rules (signatures) to identify and alert on events of interest.

Detection Tools and Techniques

Host-based intrusion detection system



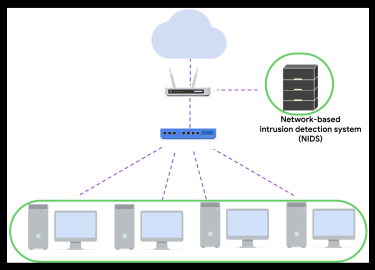
A host-based intrusion detection system (HIDS) is an application that monitors the activity of the host on which it's installed. A HIDS is installed as an agent on a host. A host is also known as an endpoint, which is any device connected to a network like a computer or a server.

Typically, HIDS agents are installed on all endpoints and used to monitor and detect security threats. A HIDS monitors internal activity happening on the host to identify any unauthorized or abnormal behavior. If anything unusual is detected, such as the installation of an unauthorized application, the HIDS logs it and sends out an alert.

In addition to monitoring inbound and outbound traffic flows, HIDS can have additional capabilities, such as monitoring file systems, system resource usage, user activity, and more.

This diagram shows a HIDS tool installed on a computer. The dotted circle around the host indicates that it is only monitoring the local activity on the single computer on which it’s installed.

Network-based intrusion detection system



A network-based intrusion detection system (NIDS) is an application that collects and monitors network traffic and network data. NIDS software is installed on devices located at specific parts of the network that you want to monitor. The NIDS application inspects network traffic from different devices on the network. If any malicious network traffic is detected, the NIDS logs it and generates an alert.

This diagram shows a NIDS that is installed on a network. The highlighted circle around the server and computers indicates that the NIDS is installed on the server and is monitoring the activity of the computers.

Using a combination of HIDS and NIDS to monitor an environment can provide a multi-layered approach to intrusion detection and response. HIDS and NIDS tools provide a different perspective on the activity occurring on a network and the individual hosts that are connected to it. This helps provide a comprehensive view of the activity happening in an environment.

Detection techniques

Detection systems can use different techniques to detect threats and attacks. The two types of detection techniques that are commonly used by IDS technologies are signature-based analysis and anomaly-based analysis.

Signature-based analysis

Signature analysis, or signature-based analysis, is a detection method that is used to find events of interest. A signature is a pattern that is associated with malicious activity. Signatures can contain specific patterns like a sequence of binary numbers, bytes, or even specific data like an IP address.

Previously, you explored the Pyramid of Pain, which is a concept that prioritizes the different types of indicators of compromise (IoCs) associated with an attack or threat, such as IP addresses, tools, tactics, techniques, and more. IoCs and other indicators of attack can be useful for creating targeted signatures to detect and block attacks.

Different types of signatures can be used depending on which type of threat or attack you want to detect. For example, an anti-malware signature contains patterns associated with malware. This can include malicious scripts that are used by the malware. IDS tools will monitor an environment for events that match the patterns defined in this malware signature. If an event matches the signature, the event gets logged and an alert is generated.

Advantages

- Low rate of false positives: Signature-based analysis is very efficient at detecting known threats because it is simply comparing activity to signatures. This leads to fewer false positives. Remember that a false positive is an alert that incorrectly detects the presence of a threat.

Disadvantages

- Signatures can be evaded: Signatures are unique, and attackers can modify their attack behaviors to bypass the signatures. For example, attackers can make slight modifications to malware code to alter its signature and avoid detection.
- Signatures require updates: Signature-based analysis relies on a database of signatures to detect threats. Each time a new exploit or attack is discovered, new signatures must be created and added to the signature database.
- Inability to detect unknown threats: Signature-based analysis relies on detecting known threats through signatures. Unknown threats can't be detected, such as new malware families or zero-day attacks, which are exploits that were previously unknown.

Anomaly-based analysis

Anomaly-based analysis is a detection method that identifies abnormal behavior. There are two phases to anomaly-based analysis: a training phase and a detection phase. In the training phase, a baseline of normal or expected behavior must be established. Baselines are developed by collecting data that corresponds to normal system behavior. In the detection phase, the current system activity is compared against this baseline. Activity that happens outside of the baseline gets logged, and an alert is generated.

Advantages

- Ability to detect new and evolving threats: Unlike signature-based analysis, which uses known patterns to detect threats, anomaly-based analysis *can* detect unknown threats.

Disadvantages

- High rate of false positives: Any behavior that deviates from the baseline can be flagged as abnormal, including non-malicious behaviors. This leads to a high rate of false positives.
- Pre-existing compromise: The existence of an attacker during the training phase will include malicious behavior in the baseline. This can lead to missing a pre-existing attacker.

Examining Suricata Logs

EVE JSON Format for Suricata Logs

- Suricata outputs alerts and events in EVE JSON format, which uses key-value pairs for easy searching and text extraction.
- EVE stands for Extensible Event Format, and JSON stands for JavaScript Object Notation.

Types of Suricata Log Data

- Alert Logs: These logs contain security-relevant information, typically triggered by signatures detecting suspicious activity, such as malware.
- Network Telemetry Logs: These logs record general network traffic flows, like connections to specific ports, and are not always security-relevant.

Log Examples

- Alert Log Example: An alert log shows details like IP addresses, protocol, and a signature message indicating malware detection.
- Network Telemetry Log Example: An HTTP log provides details about a web request, including the accessed hostname, user agent (e.g., Mozilla 5.0), and content type (e.g., HTML text).

Overview of Suricata

Suricata is an open-source intrusion detection system, intrusion prevention system, and network analysis tool.

Suricata features

There are three main ways Suricata can be used:

- Intrusion detection system (**IDS**): As a network-based IDS, Suricata can monitor network traffic and alert on suspicious activities and intrusions. Suricata can also be set up as a host-based IDS to monitor the system and network activities of a single host like a computer.
- Intrusion prevention system (**IPS**): Suricata can also function as an intrusion prevention system (IPS) to detect and block malicious activity and traffic. Running Suricata in IPS mode requires additional configuration such as enabling IPS mode.
- Network security monitoring (**NSM**): In this mode, Suricata helps keep networks safe by producing and saving relevant network logs. Suricata can analyze live network traffic, existing packet capture files, and create and save full or conditional packet captures. This can be useful for forensics, incident response, and for testing signatures. For example, you can trigger an alert and capture the live network traffic to generate traffic logs, which you can then analyze to refine detection signatures.

Rules

Rules or signatures are used to identify specific patterns, behavior, and conditions of network traffic that might indicate malicious activity. The terms rule and signature are often used interchangeably in Suricata. Security analysts use signatures, or patterns associated with malicious activity, to detect and alert on specific malicious activity. Rules can also be used to provide additional context and visibility into systems and networks, helping to identify potential security threats or vulnerabilities.

Suricata uses signatures analysis, which is a detection method used to find events of interest. Signatures consist of three components:

- Action: The first component of a signature. It describes the action to take if network or system activity matches the signature. Examples include: alert, pass, drop, or reject.
- Header: The header includes network traffic information like source and destination IP addresses, source and destination ports, protocol, and traffic direction.
- Rule options: The rule options provide you with different options to customize signatures.

Here's an example of a Suricata signature:

Rule options have a specific ordering and changing their order would change the meaning of the rule.

Note: The terms rule and signature are synonymous.

Note: Rule order refers to the order in which rules are evaluated by Suricata. Rules are processed in the order in which they are defined in the configuration file. However, **Suricata processes rules in a different default order: pass, drop, reject, and alert**. Rule order affects the final verdict of a packet especially when conflicting actions such as a drop rule and an alert rule both match on the same packet.

Custom rules

Although Suricata comes with pre-written rules, it is highly recommended that you modify or customize the existing rules to meet your specific security requirements.

There is no one-size-fits-all approach to creating and modifying rules. This is because each organization's IT infrastructure differs. Security teams must extensively test and modify detection signatures according to their needs.

Creating custom rules helps to tailor detection and monitoring. Custom rules help to minimize the amount of false positive alerts that security teams receive. It's important to develop the ability to write effective and customized signatures so that you can fully leverage the power of detection technologies.

Configuration file

Before detection tools are deployed and can begin monitoring systems and networks, you must properly configure their settings so that they know what to do. A configuration file is a file used to configure the settings of an application. Configuration files let you customize exactly how you want your IDS to interact with the rest of your environment.

Suricata's configuration file is `suricata.yaml`, which uses the YAML file format for syntax and structure.

Suricata Log files

There are two log files that Suricata generates when alerts are triggered:

- **eve.json**: The eve.json file is the standard Suricata log file. This file contains detailed information and metadata about the events and alerts generated by Suricata stored in JSON format. For example, events in this file contain a unique identifier called flow_id which is used to correlate related logs or alerts to a single network flow, making it easier to analyze network traffic. The eve.json file is used for more detailed analysis and is considered to be a better file format for log parsing and SIEM log ingestion.

- **fast.log**: The fast.log file is used to record minimal alert information including basic IP address and port details about the network traffic. The fast.log file is used for basic logging and alerting and is considered a legacy file format and is not suitable for incident response or threat hunting tasks.

The main difference between the eve.json file and the fast.log file is the level of detail that is recorded in each. The fast.log file records basic information, whereas the eve.json file contains additional verbose information.

Key takeaways

In this reading, you explored some of Suricata's features, rules syntax, and the importance of configuration. Understanding how to configure detection technologies and write effective rules will provide you with clear insight into the activity happening in an environment so that you can improve detection capability and network visibility. Go ahead and start practicing using Suricata in the upcoming activity!

Resources for more information

If you would like to learn more about Suricata including rule management and performance, check out the following resources:

- Suricata user guide
- Suricata features
- Rule management
- Rule performance analysis
- Suricata threat hunting webinar
- Introduction to writing Suricata rules
- Eve.json jq examples

Example:


```
{
  "timestamp": "2022-09-11T20:18:28.575413-0700",
  "flow_id": 1785579587298036,
  "pcap_cnt": 11,
  "event_type": "http",
  "src_ip": "2601:0644:8f00:254e:4f09:244b:a715:2634",
  "src_port": 43202,
  "dest_ip": "2001:4998:0044:3507:0000:0000:0000:8001",
  "dest_port": 80,
  "proto": "TCP",
  "tx_id": 0,
  "community_id": "1:x+bWOP2v5tbbdxaJa9l22exz414=",
  "http": {
    "hostname": "www.yahoo.com",
    "url": "/",
    "http_user_agent": "Mozilla/5.0",
    "http_content_type": "text/html",
    "http_method": "GET",
    "protocol": "HTTP/1.1",
    "status": 301,
    "redirect": "https://www.yahoo.com/",
    "length": 8
  }
}
```

Log Sources and Log Ingestion

Data is required for SIEM tools to work effectively. SIEM tools must first collect data using log ingestion. Log ingestion is the process of collecting and importing data from log sources into a SIEM tool. Data comes from any source that generates log data, like a server.

In log ingestion, the SIEM creates a copy of the event data it receives and retains it within its own storage. This copy allows the SIEM to analyze and process the data without directly modifying the original source logs. The collection of event data provides a centralized platform for security analysts to analyze the data and respond to incidents. This event data includes authentication attempts, network activity, and more.

Log forwarders

There are many ways SIEM tools can ingest log data. For instance, you can manually upload data or use software to help collect data for log ingestion. Manually uploading data may be inefficient and time-consuming because networks can contain thousands of systems and devices. Hence, it's easier to use software that helps collect data.

A common way that organizations collect log data is to use log forwarders. Log forwarders are software that automate the process of collecting and sending log data. Some operating systems have native log forwarders. If you are using an operating system that does not have a native log forwarder, you would need to install a third-party log forwarding software on a device. After installing it, you'd configure the software to specify which logs to forward and where to send them. For example, you can configure the logs to be sent to a SIEM tool. The SIEM tool would then process and normalize the data. This allows the data to be easily searched, explored, correlated, and analyzed.

Note: Many SIEM tools utilize their own proprietary log forwarders. SIEM tools can also integrate with open-source log forwarders. Choosing the right log forwarder depends on many factors such as the specific requirements of your system or organization, compatibility with your existing infrastructure, and more.

Resources

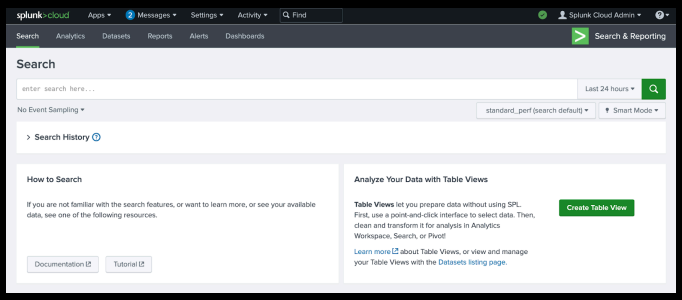
Here are some resources if you'd like to learn more about the log ingestion process for Splunk and Google SecOps (Chronicle) :

- Guide on getting data into Splunk
- Guide on data ingestion into Google Security Operations

Pro tip: Later on in this module, you'll use Splunk Cloud to upload data, perform basic searches on the data, and answer a series of questions. Follow the instructions in the Follow-along guide for Splunk sign-up to complete the following before you begin using Splunk. To get started with Splunk quickly, sign up now! The verification email might take a while to arrive.

Search Methods with SIEM Tools

Splunk searches



As you've learned, Splunk has its own querying language called Search Processing Language (**SPL**). SPL is used to search and retrieve events from indexes using Splunk's Search & Reporting app. An SPL search can contain many different commands and arguments. For example, you can use commands to transform your search results into a chart format or filter results for specific information.

Here is an example of a basic SPL search that is querying an index for a failed event:

```
index=main fail
```

- `index=main`: This is the beginning of the search command that tells Splunk to retrieve events from an `index` named `main`. An index stores event data that's been collected and processed by Splunk.
- `fail`: This is the search term. This tells Splunk to return any event that contains the term `fail`.

Knowing how to effectively use SPL has many benefits. It helps shorten the time it takes to return search results. It also helps you obtain the exact results you need from various data sources. SPL supports many different types of searches that are beyond the scope of this reading. If you would like to learn more about SPL, explore Splunk's Search Reference.

Pipes

Previously, you might have learned about how piping is used in the Linux bash shell. As a refresher, piping sends the output of one command as the input to another command.

SPL also uses the pipe character `|` to separate the individual commands in the search. It's also used to chain commands together so that the output of one command combines into the next command. This is useful because you can refine data in various ways to get the results you need using a single command.

Here is an example of two commands that are piped together:

```
index=main fail| chart count by host
```

- `index=main fail`: This is the beginning of the search command that tells Splunk to retrieve events from an `index` named `main` for events containing the search term `fail`.
- `|`: The pipe character separates and chains the two commands `index=main` and `chart count by host`. This means that the output of the first command `index=main` is used as the input of the second command `chart count by host`.
- `chart count by host`: This command tells Splunk to transform the search results by creating a `chart` according to the `count` or number of events. The argument `by host` tells Splunk to list the events by host, which are the names of the

devices the events come from. This command can be helpful in identifying hosts with excessive failure counts in an environment.

Wildcard

A wildcard is a special character that can be substituted with any other character. A wildcard is usually symbolized by an asterisk character `*`. Wildcards match characters in string values. In Splunk, the wildcard that you use depends on the command that you are using the wildcard with. Wildcards are useful because they can help find events that contain data that is similar but not entirely identical. Here is an example of using a wildcard to expand the search results for a search term:

```
index=main fail*
```

- `index=main`: This command retrieves events from an `index` named `main`.
- `fail*`: The wildcard after `fail` represents any character. This tells Splunk to search for all possible endings that contain the term `fail`. This expands the search results to return any event that contains the term `fail` such as “failed” or “failure”.

Pro tip: Double quotations are used to specify a search for an exact phrase or string. For example, if you want to only search for events that contain the exact phrase `login failure`, you can enclose the phrase in double quotations `"login failure"`. This search will match only events that contain the exact phrase `login failure` and not other events that contain the words `failure` or `login` separately.

Google Security Operations (Chronicle) searches

In Google SecOps (Chronicle), you can search for events using the Search field. You can also use Procedural Filtering to apply filters to a search to further refine the search results. For example, you can use Procedural Filtering to include or exclude search results that contain specific information relating to an event type or log source. There are two types of searches you can perform to find events in Google SecOps (Chronicle), a Unified Data Model (UDM) Search or a Raw Log Search.

Unified Data Model (UDM) Search

The UDM Search is the default search type used in Google SecOps (Chronicle). You can perform a UDM search by typing your search, clicking on “Search,” and selecting “UDM Search.” Through a UDM Search, Google SecOps (Chronicle) searches security data that has been ingested, parsed, and normalized. A UDM Search retrieves search results faster than a Raw Log Search because it searches through indexed and structured data that’s normalized in UDM.

A UDM Search retrieves events formatted in UDM and these events contain UDM fields. There are many different types of UDM fields that can be used to query for specific information from an event. Discussing all of these UDM fields is beyond the scope of this reading, but you can learn more about UDM fields by exploring Google Security Operations UDM field list. Know that all UDM events contain a set of common fields including:

- Entities: Entities are also known as nouns. All UDM events must contain at least one entity. This field provides additional context about a device, user, or process that's involved in an event. For example, a UDM event that contains entity information includes the details of the origin of an event such as the hostname, the username, and IP address of the event.
- Event metadata: This field provides a basic description of an event, including what type of event it is, timestamps, and more.
- Network metadata: This field provides information about network-related events and protocol details.
- Security results: This field provides the security-related outcome of events. An example of a security result can be an antivirus software detecting and quarantining a malicious file by reporting "virus detected and quarantined."

Here's an example of a simple UDM search that uses the event metadata field to locate events relating to user logins:

```
metadata.event_type = "USER_LOGIN"
```

- `metadata.event_type = "USER_LOGIN"`: This UDM field `metadata.event_type` contains information about the event type. This includes information like timestamp, network connection, user authentication, and more. Here, the event type specifies `USER_LOGIN`, which searches for events relating to authentication.

Using just the metadata fields, you can quickly start searching for events. As you continue practicing searching in Google SecOps (Chronicle) using UDM Search, you will encounter more fields. Try using these fields to form specific searches to locate different events.

Raw Log Search

If you can't find the information you are searching for through the normalized data, using a Raw Log Search will search through the raw, unparsed logs. You can perform a Raw Log Search by typing your search, clicking on "Search," and selecting "Raw Log Search." Because it is searching through raw logs, it takes longer than a structured search. In the Search field, you can perform a Raw Log Search by specifying information like usernames, filenames, hashes, and more. Google SecOps (Chronicle) will retrieve events that are associated with the search.

Pro tip: Raw Log Search supports the use of regular expressions, which can help you narrow down a search to match on specific patterns.

Resources for more information

Here are some resources should you like to learn more about searching for events with Splunk and Google SecOps (Chronicle):

- Splunk's Search Manual on how to use the Splunk search processing language (SPL)
- Google Security Operations quickstart guide on the different types of searches