**Splunk from Elango video**

Splunk is a powerful platform designed to handle and analyze large volumes of machine-generated data. It collects, indexes, and correlates real-time data, providing insights through dashboards, alerts, reports, and visualizations

**Problems Splunk Solves**

1. **Data Overload**
   * **Problem**: Imagine you have hundreds of servers, applications, and devices all generating logs and data every second. It's overwhelming to manually sift through all this information.
   * **Solution**: Splunk collects all this data in one place and indexes it, making it searchable. For example, if you want to find out why a server crashed, you can quickly search through the logs in Splunk to pinpoint the issue.
2. **Security Monitoring**
   * **Problem**: Cyber threats are constantly evolving, and it's challenging to detect and respond to them quickly.
   * **Solution**: Splunk can monitor your systems in real-time for suspicious activity. For instance, if there's an unusual login attempt from a different country, Splunk can alert your security team immediately.
3. **IT Operations**
   * **Problem**: Keeping IT systems running smoothly requires constant monitoring and quick troubleshooting when issues arise.
   * **Solution**: Splunk provides dashboards that show the health of your systems in real-time. If a website goes down, Splunk can help you identify whether it's due to a server issue, a network problem, or something else.
4. **Business Analytics**
   * **Problem**: Businesses generate a lot of data, but it's often hard to extract meaningful insights from it.
   * **Solution**: Splunk can analyze sales data, customer behavior, and other business metrics. For example, a retail company can use Splunk to track which products are selling the most and adjust their inventory accordingly.

**Examples**

* **Example 1**: A bank uses Splunk to monitor transactions for fraud. If Splunk detects a pattern that looks like fraudulent activity, it can alert the bank's security team to investigate further.
* **Example 2**: An e-commerce website uses Splunk to track user activity. If the website starts running slowly, Splunk can help the IT team quickly find out if the issue is with the server, the database, or the network.

A diagram of a company's security and compliance

Description automatically generated with medium confidence

**1. Splunk Forwarder**

**What It Does**: The forwarder is responsible for collecting data from various sources and sending it to the Splunk Indexer. Think of it as a data collector.

**Types**:

* **Universal Forwarder**: Sends raw data without any processing.
* **Heavy Forwarder**: Processes data before sending it, which can reduce the load on the indexer.

**Example**: Imagine you have a web server generating logs. You install a Splunk Universal Forwarder on the server to collect these logs and send them to the Splunk Indexer.

**2. Splunk Indexer**

**What It Does**: The indexer takes the data from the forwarder, processes it, and stores it in a way that makes it easy to search and analyze. It transforms raw data into searchable events.

**Functions**:

* **Parsing**: Breaks down the data into individual events.
* **Indexing**: Stores the parsed data in an organized manner.

**Example**: Continuing from the previous example, the logs collected by the forwarder are sent to the indexer. The indexer processes these logs, extracting useful information like timestamps and error messages, and stores them for easy retrieval.

**3. Splunk Search Head**

**What It Does**: The search head provides the user interface for searching, analyzing, and visualizing the data stored by the indexer. It allows users to run queries and create reports and dashboards.

**Example**: If you want to find out how many errors occurred on your web server in the last 24 hours, you would use the search head to run a query on the indexed data. The search head retrieves the relevant data from the indexer and displays it in a user-friendly format.

**Ports (can be customizable)**

Web Interface: 8000

Data Forwarding: 9997

Index Replication: 9887

Key-value Replication: 8191

Event Collector: 8088

Custom-Syslog Port: 514

**Types of Forwarders**

The main difference between a Heavy Forwarder and a Universal Forwarder in Splunk lies in their processing capabilities.

**Heavy Forwarder:** A Heavy Forwarder is a full Splunk Enterprise instance with most features enabled. It can perform several processing tasks before forwarding data to the indexer:

1. **Data Parsing**: It can break down raw data into individual events, extracting useful information like timestamps and metadata.
2. **Data Filtering**: It can filter out unwanted data, ensuring only relevant information is forwarded.
3. **Data Routing**: It can route data to different indexers based on specific criteria, such as the source or type of event.
4. **Local Indexing**: It can index data locally if needed, allowing for local searches and analysis.
5. **Data Transformation**: It can modify data, such as masking sensitive information or reformatting fields.

**Universal Forwarder:** A Universal Forwarder, on the other hand, is a lightweight version designed solely for forwarding data. It has limited processing capabilities:

1. **Basic Data Collection**: It collects raw data from various sources and forwards it to the indexer.
2. **Minimal Parsing**: It performs minimal parsing, mainly for metadata stamping, but does not break down data into individual events.
3. **No Filtering or Routing**: It does not filter or route data based on content.
4. **No Local Indexing**: It cannot index data locally or perform searches.

**Example Scenario:** Imagine you have a network of servers generating logs:

* **Universal Forwarder**: You install Universal Forwarders on each server to collect logs and send them directly to the indexer. These forwarders do not process the logs; they simply forward them as-is.
* **Heavy Forwarder**: You use a Heavy Forwarder as an intermediary. It collects logs from the Universal Forwarders, parses the logs to extract useful information, filters out irrelevant data, and routes the processed logs to different indexers based on the type of log.

**Why do we use heavyweight forwarder as an intermediary with universal, when we can directly collect logs from heavyweight?**

**Scalability**

* **Universal Forwarders** are lightweight and can be deployed on numerous endpoints. They efficiently collect and send data to a central Heavy Forwarder.
* **Heavy Forwarder** can handle the processing load from multiple Universal Forwarders, making it easier to scale data collection across a large number of devices.

**2. Centralized Processing**

* **Data Parsing and Filtering**: By centralizing data parsing and filtering at the Heavy Forwarder, you ensure consistent data processing rules across all collected data. This reduces the processing burden on individual endpoints.
* **Example**: If you have hundreds of servers generating logs, each with a Universal Forwarder, the Heavy Forwarder can centrally parse and filter these logs, ensuring uniformity and reducing redundancy.

**3. Efficient Resource Utilization**

* **Resource Management**: Universal Forwarders are designed to be resource-efficient, minimizing the impact on the systems they are installed on. Heavy Forwarders, being more resource-intensive, are better suited for centralized locations where they can leverage more powerful hardware.
* **Example**: On a critical application server, you might not want the additional load of a Heavy Forwarder. Instead, a Universal Forwarder sends data to a Heavy Forwarder located on a dedicated machine.

**4. Data Routing and Load Balancing**

* **Routing**: Heavy Forwarders can route data to different indexers based on predefined rules.
* **Example**: If you have different types of logs (e.g., application logs, security logs), the Heavy Forwarder can route them to specialized indexers for more efficient processing and storage.

**5. Security and Compliance**

* **Data Masking and Transformation**: Heavy Forwarders can perform data masking and transformation to ensure sensitive information is protected before it reaches the indexers.
* **Example**: If logs contain sensitive user information, the Heavy Forwarder can mask this data to comply with privacy regulations before forwarding it.

**How are Events Stored?**

A diagram of a bucket diagram

Description automatically generated

**Indexers**

**What They Do**: Indexers are the Splunk components responsible for receiving, processing, and storing data. **They take raw data, parse it into events, and store these events in a structured way so they can be easily searched and analyzed**.

**Example**: Imagine you have a server generating logs. The Indexer receives these logs, processes them to extract useful information (like timestamps and error messages), and stores them for future searches.

**2. Indexes**

**What They Are**: An index is a collection of data that Splunk has processed and stored. Each index is like a database that holds a specific type of data.

**Example**: You might have different indexes for different types of data, such as:

* main: The default index for general data.
* security: An index for security-related logs.
* web: An index for web server logs.

**3. Buckets**

**What They Are**: Buckets are directories within an index that store data. Each bucket contains both the raw data and the index files that make the data searchable. Buckets are organized by the age of the data they contain.

**Stages of Buckets**:

* **Hot Buckets**: These are actively being written to. When data first arrives, it goes into a hot bucket.
* **Warm Buckets**: Once a hot bucket is full, it is closed and becomes a warm bucket. Warm buckets are still searchable but no longer written to.
* **Cold Buckets**: As data ages, warm buckets are moved to cold storage. Cold buckets are also searchable but stored in a different location to optimize storage costs.
* **Frozen Buckets**: Eventually, data in cold buckets is either archived or deleted. Frozen buckets are not searchable unless they are thawed (restored).

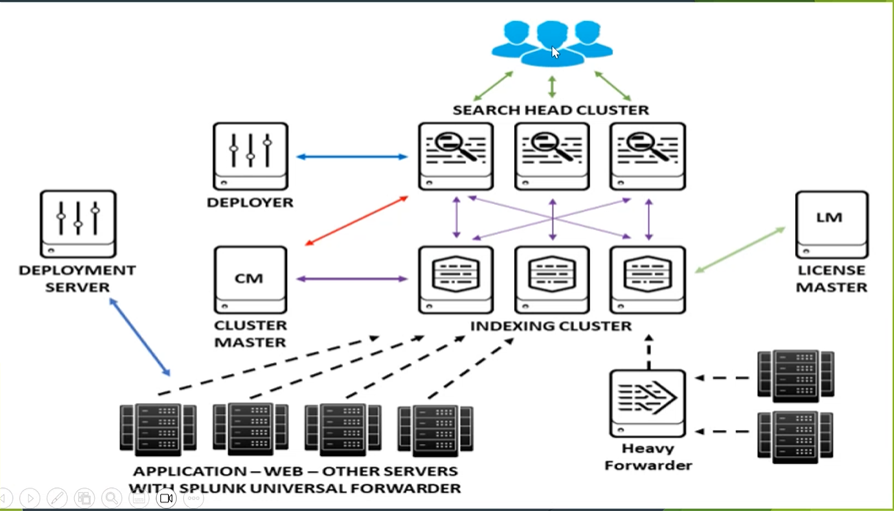
**Example**: Let's say your web server generates logs continuously. These logs are first stored in a hot bucket. After a certain period or when the bucket reaches a size limit, it becomes a warm bucket. Over time, as more data comes in, older warm buckets are moved to cold storage. Finally, very old data might be archived or deleted as frozen buckets.

**Visual Example:** Imagine you run an online store:

* **Indexer**: Receives logs about user activity, transactions, and errors.
* **Indexes**: You have separate indexes for transactions, user\_activity, and errors.
* **Buckets**: Within the transactions index, today's logs are in a hot bucket, last week's logs are in warm buckets, and last month's logs are in cold buckets. Logs from last year might be archived as frozen buckets.

**Does heavyweight forwarders process the logs or the indexers**

* **Heavy Forwarders**: Perform initial processing, such as parsing, filtering, and transforming logs before forwarding them.
* **Indexers**: Handle the final parsing, indexing, and storage of logs, making them searchable and ready for analysis.

****

**fig: Splunk Architecture**

**Deployment server, cluster master, deployer, license Master, search head master**

**1. Deployment Server (manages configuration on all forwarders)**

**What It Does**: The Deployment Server manages and distributes configurations (like apps, input.conf, props.conf, etc) to multiple Splunk instances, especially forwarders.

**Example**: If you have 100 servers with Splunk forwarders, then deployment server makes sure all forwarders are updated with the latest input.conf and output.conf file

**2. Cluster Master (Index Cluster Master)**

**What It Does**: The Cluster Master manages indexer clusters, which are group of indexers, working together to provide data replication and high availability

**Example**: The Cluster Master makes sure that data is spread across all of them. If one indexer fails, the data is still available on the others. Data like important transaction logs of financial organization.

**3. Deployer**

**What It Does**: The Deployer ensures that all search heads in a cluster have the same configurations and apps.

**Example**: If your company wants all search heads in a cluster to display the same set of dashboards. The deployer ensures the dashboards are same on all search heads

**4. License Master**

**What It Does**: The License Master manages Splunk licenses and monitors how much data is being indexed across all Splunk Components.

**Example**: If a company has 100GB daily indexing licence. The license master ensures that all data indexed by multiple indexers stay within the limit and warns if the limit is exceeded.

**5. Search Head Cluster (SHC)**

**What It Does**: A Search head cluster is a group of search heads that work together to distribute and coordinate search jobs across multiple users for high availability.

**Example**: A company where multiple analysts are running searches and creating dashboards for sales data. A search head cluster ensures that if one search head fails, analysts can continue working without any downtime

**Data Onboarding**

1. **HTTP Event Collector**

HEC is a feature in Splunk that allows you to send data directly to Splunk over HTTP or HTTPS. This is particularly useful for **collecting data from applications, cloud services, and IoT devices without needing a Splunk forwarder.**

**How Does HEC Work?**

1. HEC Token: A unique authentication key (Token) is generated in splunk to allow secure data ingestion. The splunk will send this token to the app. Application include this token in their HTTP requests when sending data to splunk.
2. Data Ingestion: Data is sent to splunk in JSON format, through an HTTP Post request. This data include logs.
3. Indexing: Splunk processes the incoming data and stores it in the specified index based on the HEC configuration or events
4. Search and Analye: Once the data is indexed, users can search, visualize and analyze it in splunk dashboard.
5. **DB Onboarding (We cannot use universal forwarder here, so we use DBConnect add on app)**

Database onboarding in Splunk involves bringing data from databases into Splunk for indexing, searching, and analysis. This is typically done using the Splunk DB Connect app, which allows you to connect to various databases, run queries, and import data into Splunk.

**Drawback:** DBConnect can work only for a specific time range like 5-10 mins. If you have large number of jobs, then don’t use DBConnect because it will use more resources.

**Key Steps in Database Onboarding with an example**

**Example:** Imagine you have a MySQL database that stores sales data for your online store. You want to bring this data into Splunk to analyze sales trends and generate reports.

**1. Install Splunk DB Connect**

**What It Is**: Splunk DB Connect is an app that you install in Splunk to enable database connectivity. It supports various databases like MySQL, PostgreSQL, Oracle, and Microlsoft SQL Server.

**Example**: You download and install the Splunk DB Connect app from Splunkbase, which adds database connectivity features to your Splunk instance.

**2. Configure Database Connection**

**What It Involves**: Setting up a connection to your MySQL database by providing necessary details.

**Example**:

* **Database Type**: MySQL
* **Host**: db.example.com
* **Port**: 3306
* **Username**: db\_user
* **Password**: db\_password

You enter these details in the Splunk DB Connect app to establish a connection to your MySQL database.

**3. Create Database Inputs**

**What It Involves**: Defining how data will be fetched from the database and imported into Splunk.

**Types of Inputs**:

* **Rising Column Input**: Tracks a column (like a timestamp) to fetch only new records.
* **Batch Input**: Fetches a complete set of data at regular intervals.

**Example**:

* **SQL Query**: SELECT \* FROM sales WHERE sale\_date > ?

This query fetches new sales records based on the sale\_date.

* **Input Type**: Rising column input, which tracks the last value of sale\_date to fetch only new records.
* **Target Index**: sales\_data

You configure this input in Splunk DB Connect to regularly fetch new sales data from your database.

**Bucket Lifecycle**

**A diagram of a cycle

Description automatically generated**

**Splunk Conf. files (opt/splunk/etc)**

* **deploymentclient.conf: Purpose**: This file is used by the deployment server to manage updates for deployment clients (other splunk instances). Think of it as a way to tell client machines how often they should check in for updates. Example:

[deployment-server]

repositoryLocation = /opt/splunk/etc/deployment-apps

phoneHomeIntervalInSecs = 300

Here,

repositoryLocation: Where the apps to be deployed are stored.

phoneHomeIntervalInSecs: Clients check every 5 minutes for updates.

* **inputs.conf: Purpose**: This file defines where and how Splunk collects data from various sources. It contains list of instructions to tell splunk what data sources, how to collect data and where to collect data.
* **outputs.conf: Purpose:** This file is used to forward data from one Splunk instance to another, like from a forwarder to an indexer. If output.conf is not present then splunk don’t know where to send the data.
* **server.conf: Purpose:** This file contains general settings for the Splunk server, including its roles and general behavior.
* **serverclass.conf:** This file is used by the deployment server to define which clients get which apps. You set up rules based on client characteristics. Deploying AWS app on certain hostnames working in cloud.
* **props.conf: Purpose:** This file tells Splunk how to handle and process incoming data, like defining data types or extracting timestamps. It tells how to break data (maybe timestamp), how to assign metadata and how to normalize data.
* **transforms.conf: Purpose**: This file is used alongside `props.conf` to transform or filter data. This file ensures that data is clean. It contains rule to modify data. For example, you can mask sensitive information or decide where specific data goes..
* **indexes.conf: Purpose**: This file configures index settings, such as where data is stored and how long it is retained. This file is also used to manage the bucket to store data. It stores the current data in db and older data in colddb. It tells how long to store data, how much data index can store, where the data is saved in index, how long to keep data before deleting it.
* **authorize.conf: (Access Control) Purpose**: This file manages user roles and capabilities. This file manages roles and their permissions. A role determines what a user can do in Splunk. Types of access: User: Basic access, Power: Limited access and Admin.
* **authentication.conf: Purpose**: This file configures how users log in to Splunk. For example, you can set up LDAP authentication. It tells splunk how to verify users.
* **savedsearches.conf: Purpose:** This file defines saved searches, alerts, and reports. It is used to save the searches that you run frequently.

**Splunk Role Creation**

**You can create roles in splunk Enterprize or by editing the authorize.conf file**

In Splunk, roles determine what users can or cannot do. A role is a collection of permissions and settings that control access to data. Parameters

**Role Name:** Unique name for the role.

**Inherit from other roles:** Choose if this role should inherit permissions from existing roles. Example: A "Manager" role might inherit all permissions from a "User" role.

**Capabilities**: Select actions the role can perform (e.g., create alerts, edit dashboards). **Examples:**

search: Allows users to run searches.

edit\_dashboards: Allows users to edit dashboards.

admin\_all\_objects: Grants full administrative access.

**Indexes**: Assign what data this role can access. Specify which indexes a role can access.

Example: srchIndexesAllowed = web\_logs, sales\_data restricts access to these two indexes

**Search Restrictions**: Add filters to limit data visibility (e.g., restrict to a subset of data). Restrict what data users can search for. ‘Example: srchFilter = index=main NOT confidential ensures users can’t see confidential data.

**Resource Limits:** Control resource usage to ensure fair access.

**Splunk Enterprise and Splunk Cloud**

**Splunk Enterprise**

Splunk Enterprise is typically deployed on-premises or in a private cloud. This means you install and manage the software on your own servers or infrastructure.

* **Full Control**: You have complete control over your data and infrastructure, which is beneficial for organizations with specific compliance or security requirements.
* **Customization**: You can customize the deployment to fit your specific needs, including hardware configurations and network settings.

**Splunk Cloud**

Splunk Cloud is a fully managed cloud-based service provided by Splunk. It eliminates the need for managing infrastructure, as Splunk handles all the backend operations.

**Key Differences**

1. **Deployment Model**:
   * **Splunk Enterprise**: On-premises or private cloud.
   * **Splunk Cloud**: Fully managed cloud service.
2. **Scalability**:
   * **Splunk Enterprise**: Requires manual scaling by the IT team.
   * **Splunk Cloud**: Automatically scales to meet data demands.
3. **Menus**

* Splunk Enterprise: The menus are displayed on the right of the UI
* Splunk Cloud: The menus are displayed on the left side of the UI

1. **Maintenance and Upgrades**:
   * **Splunk Enterprise**: Managed by your IT team.
   * **Splunk Cloud**: Managed by Splunk, including updates and maintenance.
2. **Security and Compliance**:
   * **Splunk Enterprise**: Full control over security settings and compliance measures.
   * **Splunk Cloud**: Splunk ensures security and compliance, suitable for many regulatory environments.
3. **Cost and Pricing**:
   * **Splunk Enterprise**: Costs include hardware, software, and maintenance.
   * **Splunk Cloud**: Subscription-based pricing, with costs covering the managed service.

**Splunk UI**

1. Index = \_internal : It considers splunks internal logs
2. Prerequisites before onboarding data in splunk:

* Server should have splunk installed
* Splunk should have read permissions for files
* You should have a sample log
* Check the format of log and proceed.

1. Parameters of Indexes

* Index Name
* Max raw data size
* Retention Days (No. of days data is searchable)

**Splunk (from udemy)**

Splunk is a powerful platform designed to search, monitor, and analyze machine-generated data (like logs from servers, applications, and devices). It helps organizations make sense of large volumes of data by turning it into valuable insights.

**Key Components of Splunk**

1. **Forwarder**: This component collects data from various sources and sends it to the Splunk indexer. There are two types:
   * **Universal Forwarder**: Lightweight, used for collecting and forwarding data.
   * **Heavy Forwarder**: Can parse data before forwarding it.
2. **Indexer**: This component processes the incoming data, indexes it, and stores it. It also handles search requests and retrieves the data.
3. **Search Head**: This is the user interface where you can search, analyze, and visualize the data. It allows you to create dashboards, reports, and alerts.

**How Splunk Works**

1. **Data Collection**: Data is collected from various sources using forwarders.
2. **Indexing**: The indexer processes and stores the data in a searchable format.
3. **Searching and Reporting**: Users can search the indexed data using the Search Processing Language (SPL) and create visualizations.

**Example Use Case:** Imagine you are managing a large e-commerce website. You want to monitor the performance of your servers and track user activity. Here’s how Splunk can help:

1. **Data Collection**: Install a universal forwarder on each server to collect logs.
2. **Indexing**: The forwarders send the logs to the Splunk indexer, which processes and stores them.
3. **Searching**: Use the search head to query the data. For example, you can search for error logs to identify issues or analyze user activity to understand peak usage times.
4. **Reporting**: Create dashboards to visualize server performance metrics and user activity trends.

**Splunkbase:** One of Splunk’s powerful features is its marketplace, known as Splunkbase. Here, users can find and install apps and add-ons developed by others. This allows for out-of-the-box solutions for various use cases. For example, if you use AWS, you can find apps on Splunkbase that provide pre-built dashboards and reports for AWS services.

**(Udemy)**

**Importing Data into Splunk**

**1. Add Data Wizard:** This is the most straightforward method, especially for beginners.

1. **Go to Settings**: In the Splunk web interface, click on **Settings**.
2. **Select Add Data**: Choose **Add Data** from the dropdown menu.
3. **Upload Your File**: Click on **Upload** and select the file you want to import. For example, if you have a CSV file named sales\_data.csv, you can drag and drop it here.
4. **Set Source Type**: Splunk will try to automatically detect the source type (e.g., CSV, JSON). You can also manually set it if needed.
5. **Input Settings**: Configure settings like the host field value and the index where you want to store the data.
6. **Review and Submit**: Review your settings and click **Submit**. Your data will be indexed and ready for searching.

**Example:** Imagine you have a CSV file with sales data. You would:

* Go to **Settings > Add Data > Upload**.
* Select sales\_data.csv.
* Let Splunk detect the source type as CSV.
* Set the index to sales\_index.
* Click **Submit**.

**2. Using Universal Forwarder:** This method is used for continuously monitoring and forwarding data from a server to Splunk.

1. **Configure Receiving Indexer**: Set up your Splunk instance to receive data on a specific port (e.g., 9997).
2. **Install Universal Forwarder**: Download and install the Splunk Universal Forwarder on the server where your data resides.
3. **Start Forwarder**: Start the forwarder and configure it to run on boot.
4. **Add Forward Server**: Add the address and port of your receiving Splunk instance.
5. **Monitor Data Source**: Specify the data source you want to monitor (e.g., log files).
6. **Verify Data**: Check that the data is being indexed in your Splunk instance.

**Example:** If you want to monitor a log file on a server:

* Install the Universal Forwarder on the server.
* Configure it to forward data to your Splunk instance at splunk-server:9997.
* Add the log file path (e.g., /var/log/syslog) to the forwarder configuration.
* Verify that the data appears in Splunk.

**Parsing Authentication Logs**

**Understanding Log Parsing in Splunk:** When you upload a log file to Splunk, it’s crucial to set the correct **source type**. The source type tells Splunk what kind of data it is dealing with, so it can use the appropriate parser to extract meaningful information from the logs.

**Setting the Source Type:** When you upload a log file, you can specify its source type. For example, if you have a log file from an Apache web server, you might set the source type to access\_combined. This helps Splunk understand the format of the log file and apply the correct parsing rules.

**Using Add-Ons for Additional Parsing:** Sometimes, the default parsers in Splunk might not be enough, especially for specific types of logs like Linux authentication logs. In such cases, you can install add-ons from the Splunkbase marketplace to enhance Splunk’s parsing capabilities

**Verifying the Parsing:** After setting the source type and installing the necessary add-ons, you can verify that the logs are being parsed correctly by searching for the logs in Splunk and checking the extracted fields.

**Basic Search in Splunk**

**1. Searching for a Specific String:** One of the simplest ways to search in Splunk is by entering a specific string in the search bar. For example, if you want to find all logs containing the string “200”, you would type 200 in the search bar and press search. This will return all events that include “200”.

**2. Using the Time Range Picker:** When you search for a string, Splunk allows you to specify the time range for your search. For example, if you want to see logs containing “200” from the last 24 hours, you can set the time range to “Last 24 hours” instead of “All time”.

**3. Highlighting Search Terms** Splunk highlights the search terms in the results, making it easier to spot the relevant events. For example, if you search for “fail”, all occurrences of “fail” in the events will be highlighted.

**Advanced Search Techniques**

**1. Using Wildcards:** You can use wildcards to search for variations of a term. For example, searching for fail\* will match “fail”, “failed”, “failing”, etc. This is similar to using wildcards in file searches on an operating system.

**2. Boolean Expressions:** Splunk supports Boolean expressions like AND, OR, and NOT to refine your searches.

* **AND**: To find events that contain both “root” and “failed”, you can use root AND failed. By default, a space in SPL is considered as AND
* **OR**: To find events that contain either “admin” or “root”, you can use admin OR root.
* **NOT**: To exclude events containing “root”, you can use NOT root.

**Search Mode in Splunk**

**Fast Mode: Fast Mode** returns only **essential event data**, such as default fields and fields extracted at index time. This mode is useful when you need quick results and don’t require detailed field information.

**Key Features:**

* Extracts only default fields (e.g., host, source, sourcetype).
* Does not perform search-time field extraction unless explicitly specified.
* Ideal for quick, high-level searches.

**Example:** If you want to quickly count the number of events in a specific index:

index=web\_logs | stats count

In Fast Mode, this search will run quickly, providing a count of events without extracting additional fields.

**2. Smart Mode: Smart Mode** is the default mode and provides a balance between speed and completeness. It dynamically switches between Fast and Verbose modes based on the type of search you are running.

**Key Features:**

* For event searches, it behaves like Verbose Mode, extracting all fields.
* For transforming searches (e.g., those using stats or chart), it behaves like Fast Mode, prioritizing speed.
* Suitable for most searches, offering a good balance.

**Example:**

If you search for error events and want to see detailed information:

index=web\_logs error

In Smart Mode, this search will extract all relevant fields and provide detailed event information.

**3. Verbose Mode: Verbose Mode** returns the most comprehensive results, including all possible field and event data. This mode is useful when you need detailed information for in-depth analysis, but it can be slower.

**Key Features:**

* Extracts all fields, including default, automatic, and user-defined fields.
* Displays a full event list and timeline.
* Best for detailed investigations and when you need to see all available data.

**Example:** If you need to investigate specific user activities in detail:

index=web\_logs user=john\_doe

In Verbose Mode, this search will extract all fields related to john\_doe, providing a complete view of the user’s activities.

**Splunk Reports (See udemy video for example)**

**What are Splunk Reports?**

Reports in Splunk are saved search results that can be run on demand or scheduled to run at specific intervals. They can include statistics, visualizations, and other data insights, making it easier to analyze and share information.

**Creating a Report**

**Perform a Search**: Start by running a search that retrieves the data you need.

**Format the Results**: You can format the search results to display specific fields

**Save the Search as a Report**

**Editing a Report:** Once a report is created, you can edit it to change the search query, update the visualization, or modify the report settings.

* **Edit the Search Query**: Open the report and modify the search query as needed. Save the changes to update the report.
* **Change Visualization**: You can change how the data is visualized by selecting different chart types or adjusting the display settings.

**Scheduling a Report:** To automate the running of a report, you can schedule it to run at specific intervals. Like on daily or weekly basis.

**Splunk Apps and Add-on**

**Splunk Apps: App-like environment**

**Apps** in Splunk are comprehensive packages that provide a user interface and various functionalities to help you work with your data. They often include dashboards, reports, alerts, and other tools to visualize and analyze data.

* **Example**: **Splunk App for Enterprise Security**
* **Purpose**: This app is designed to help organizations monitor and respond to security threats. It provides a comprehensive security operations suite, including dashboards, reports, and alerts tailored for security use cases.
* Carnival Corporation, the world's largest leisure travel company uses Splunk app for enterprise security.

**Splunk Add-Ons: like Plugins of jenkins**

**Add-ons** are components that extend Splunk’s capabilities, typically focusing on data collection, parsing, and enrichment. They do not provide a user interface but are used by apps to process and manage data.

* **Example**: The **Splunk Add-on for Microsoft Windows** collects and parses Windows event logs. It provides the necessary data inputs and field extractions that can be used by other apps to create dashboards and reports.
* **Apps**: Provide user interfaces and are designed for specific use cases.
* **Add-Ons**: Extend Splunk’s capabilities by collecting and processing data, typically used by apps.

**Dashboard and Panels**

A **dashboard** in Splunk is a collection of visualizations, such as charts, tables, and single value displays, that provide insights into your data.

A **panel** is a single visualization within a dashboard. Each panel can display data in different formats, such as bar charts, line charts, pie charts, tables, and more

**Adding Panels to a Dashboard**

**Inline Panels**: Create a panel directly within the dashboard.

**Panels from Reports**: Use saved reports to create panels.

**Example: Creating a Dashboard with Panels**

Let’s create a simple dashboard to monitor web server logs.

1. **Create a Dashboard**:
   * Name: Web Server Monitoring
   * Description: Dashboard to monitor web server logs
2. **Add Panels**:
   * **Panel 1**: HTTP Status Codes
     + Search: index=web\_logs | stats count by status
     + Visualization: Bar Chart
   * **Panel 2**: Error Logs Over Time
     + Search: index=web\_logs status=500 | timechart count
     + Visualization: Line Chart
   * **Panel 3**: Top 10 IP Addresses
     + Search: index=web\_logs | top limit=10 clientip
     + Visualization: Table

**Dashboard Inputs**

**Adding a Time Range Picker:** A **time range picker** allows users to select a time range for the data displayed in the dashboard. This is useful for viewing data from different periods, like the last 24 hours or the last 7 days.

**Text Based Input:** A **text-based input** in Splunk allows users to enter text that can be used to dynamically update the data displayed in a dashboard. This is useful for making dashboards interactive and customizable based on user input.

**Dropdown Input:** A **dropdown input** in Splunk allows users to select from a predefined list of options. This is useful because it ensures that users can only choose valid fields or values, reducing the chance of errors.

**Splunk Directory Structure**

**Main Directories in Splunk**

1. **Bin**: It contains the primary splunk binary data as well as other tools like btool
2. **Etc:** contains all the configuration files as well as all the apps and add-ons that you install in splunk
3. **Var:** contains all the data that gets indexed as well as log files
4. **Lib**: contains necessary libraries needed for splunk to run.

**Splunk Configuration Directories**

**Splunk Configuration Directories:** Splunk uses a layered configuration system where the same configuration file can exist in multiple directories. This system helps determine which settings take precedence. The main directories involved are:

1. **default**
2. **local**
3. **app**

**Key Directories and Their Roles**

**1. default Directory**

* **Location**: $SPLUNK\_HOME/etc/system/default
* **Purpose**: Contains default configuration files provided by Splunk.
* **Important Note**: **Do not edit** files in this directory because they will be overwritten during upgrades.

**2. local Directory**

* **Location**: $SPLUNK\_HOME/etc/system/local
* **Purpose**: Contains custom configuration files that override the default settings.
* **Important Note**: This is where you should make your changes.

**3. app Directory**

* **Location**: $SPLUNK\_HOME/etc/apps
* **Purpose**: Contains configuration files for individual apps and add-ons.
* **Structure**:
  + **default**: Default settings for the app.
  + **local**: Custom settings for the app that override the default settings.

**Configuration File Precedence:** Splunk uses a precedence system to determine which configuration settings to apply when the same setting is defined in multiple places. The order of precedence is:

1. **System local**: $SPLUNK\_HOME/etc/system/local
2. **App local**: $SPLUNK\_HOME/etc/apps/app\_name/local
3. **App default**: $SPLUNK\_HOME/etc/apps/app\_name/default
4. **System default**: $SPLUNK\_HOME/etc/system/default

**Example: Understanding Configuration Precedence:** Let’s say you have a configuration setting for serverName in different directories:

* **default**: serverName=default
* **app**: serverName=app
* **local**: serverName=king

Splunk will use the value from the **local** directory (serverName=king) because it has the highest precedence.

**Indexes**

An **index** in Splunk is a repository where Splunk stores data. When data is ingested into Splunk, it is processed and stored in an index. This indexed data can then be searched and analyzed. Splunk transforms incoming data into events, which it stores in indexes.

**Key Components of an Index**

1. **Indexer**: The Splunk component that processes and stores data in indexes.
2. **Buckets**: Directories within an index that store data files. Buckets are organized by age.
3. **Raw Data**: The original data ingested into Splunk, stored in a compressed format.
4. **Index Files (tsidx files)**: Files that point to the raw data and contain metadata to facilitate fast searching.

**How Indexing Works**

When data is ingested into Splunk, the indexer processes it in several steps:

1. **Data Input**: Data is collected from various sources (e.g., logs, metrics).
2. **Parsing**: Splunk breaks the data into individual events and extracts timestamps.
3. **Indexing**: The parsed data is stored in the index, with raw data and index files created.
4. **Searching**: Users can search the indexed data using Splunk's search language.

**Example: Searching an Index**

To search data in an index, you use the index keyword in your search query.

**Example Query:** index=web\_logs error

**Bucket Lifecycle**

**What is a Bucket in Splunk?**

A **bucket** in Splunk is a directory that stores indexed data. Buckets are used to organize and manage data based on its age and state. Each bucket contains raw data and index files that help Splunk search and retrieve data efficiently.

**Bucket Lifecycle Stages:** Buckets go through several stages in their lifecycle:

**1. Hot Buckets**

* **Description**: Hot buckets store newly indexed data. They are actively written to and are the most recent data.
* **Characteristics**:
  + Data is being actively indexed.
  + Multiple hot buckets can exist at the same time.
* **Example**: When you ingest new log data, it is first stored in a hot bucket.

**2. Warm Buckets**

* **Description**: When a hot bucket is full or Splunk restarts, it is rolled to a warm bucket. Warm buckets are read-only and contain slightly older data.
* **Characteristics**:
  + Data is no longer being written to.
  + Warm buckets are still searchable.
* **Example**: After a day of indexing, yesterday's data might be in a warm bucket.

**3. Cold Buckets**

* **Description**: When warm buckets reach a certain age or size, they are rolled to cold buckets. Cold buckets store older data and are typically stored on slower, less expensive storage.
* **Characteristics**:
  + Data is read-only and stored on slower storage.
  + Cold buckets are still searchable.
* **Example**: Data from last month might be stored in cold buckets.

**4. Frozen Buckets**

* **Description**: When cold buckets reach their retention limit, they are rolled to frozen buckets. Frozen buckets are not searchable and are typically archived or deleted.
* **Characteristics**:
  + Data is no longer searchable.
  + Frozen buckets can be archived or deleted to save space.
* **Example**: Data older than a year might be moved to frozen buckets and archived.

**5. Thawed Buckets**

* **Description**: If you need to search data from frozen buckets, you can thaw them. Thawed buckets are temporarily restored to a searchable state.
* **Characteristics**:
  + Data is restored from frozen storage.
  + Thawed buckets are searchable again.
* **Example**: If you need to investigate an incident from two years ago, you can thaw the relevant frozen buckets.

**Splunk Workflow Actions**

**Workflow Actions** in Splunk allow you to add interactivity to your data by linking fields in your indexed data to external web resources or other Splunk searches. This feature is particularly useful for quickly accessing additional information related to your data.

**Types of Workflow Actions**

1. **GET Workflow Actions**: Open a URL in a new window using the value of a field.
2. **POST Workflow Actions**: Send data to a URL using the HTTP POST method.
3. **Search Workflow Actions**: Run a new search based on the value of a field.

**Example: Creating a GET Workflow Action**

Let's create a workflow action that allows us to look up information about an IP address using an external website like abuseipdb.com.

**Splunk Forwarder Management**

**Challenges in Managing Splunk Universal Forwarders**

1. **Manual Installation and Configuration**: Installing and configuring the Universal Forwarder on hundreds of servers manually is time-consuming and prone to errors.
2. **Scalability Issues**: Making changes to the configuration (e.g., adding a new log file to monitor) requires updating and running scripts on all servers, which is not scalable. **Example**: If you need to monitor an additional log file on 600 servers, you have to update the configuration on each one.

**Solution: Splunk Forwarder Management:** Splunk provides an in-built feature called **Forwarder Management** to address these challenges. Here's how it works:

1. **Centralized Management**:
   * **Feature**: Forwarder Management allows you to manage all your Universal Forwarders from a central Splunk instance.
   * **Benefit**: You can configure and update settings for all forwarders from one place, eliminating the need for manual updates on each server.
2. **Automated Configuration**:
   * **Feature**: You can define rules and configurations centrally, which are then automatically applied to the forwarders.
   * **Benefit**: Changes are propagated to all relevant forwarders automatically, ensuring consistency and saving time.

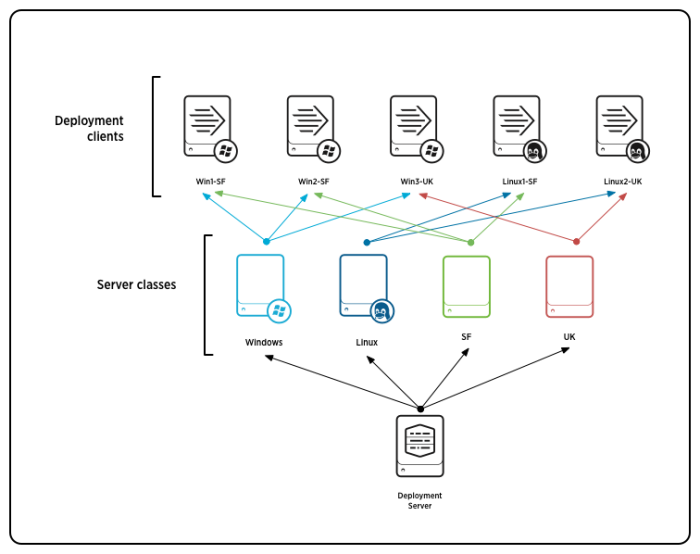
**Deployment Server**

A Deployment Server in Splunk is a central server that manages the configuration of multiple Splunk instances, such as Universal Forwarders and other Splunk components.

**Why Use a Deployment Server?**

In large organizations, managing the configuration of hundreds or thousands of Splunk instances manually is impractical. A Deployment Server automates this process, ensuring consistency and saving time.

**Forwarder Managememt is a GUI built on top of deployment server** that provides an easy way to configure the deployment server and monitor the status of deployment updates.



**How Does It Work?**

1. **Deployment Server**: The central server that holds the configuration files and apps.
2. **Deployment Clients**: The Splunk instances (like Universal Forwarders) that receive configurations from the Deployment Server. **Deployment Clients** are the universal Forwarder which are installed on the specific server
3. **Server Classes**: Groups of deployment clients that share the same configuration.

**Example Scenario:** Imagine you have a network of 500 servers, each running a Splunk Universal Forwarder. You want to ensure they all have the same configuration for monitoring log files.

1. **Setup Deployment Server**: Install and configure a Splunk instance to act as the Deployment Server.
2. **Create Configuration**: On the Deployment Server, create a configuration file to monitor /var/log/secure.
3. **Define Server Classes**: Group the 500 servers into a server class called all\_servers.
4. **Deploy Configuration**: Assign the configuration file to all\_servers server class.
5. **Clients Pull Configuration**: Each Universal Forwarder (deployment client) checks in with the Deployment Server and pulls the configuration to monitor /var/log/secure.

**Input.conf and output.conf (Not done)**

**What is inputs.conf?**

inputs.conf is a configuration file used by Splunk to specify which data sources (like log files) should be monitored and indexed. It tells Splunk where to look for data and how to handle it.

**Key Sections in inputs.conf**

1. **[monitor]**: This stanza is used to specify files or directories that Splunk should monitor.
2. **[tcp]** and **[udp]**: These stanzas are used to specify network ports that Splunk should listen to for incoming data.

**What is outputs.conf?**

outputs.conf is a configuration file used by Splunk to specify where the collected data should be sent. It defines the destination Splunk instances (like indexers) that will receive the data.

**Key Sections in outputs.conf**

1. **[tcpout]**: This stanza is used to define the default group of indexers to which data should be sent.
2. **[tcpout:]**: This stanza specifies the details of the group of indexers.
3. **[tcpout-server]**: This stanza lists the individual indexers within a group.

**Putting It All Together:** When you configure a Universal Forwarder to send logs to a Splunk instance, you need both inputs.conf and outputs.conf.

**Step-by-Step Example**

1. **Create the Deployment App**: On the Deployment Server, create a new directory for your app:

mkdir /opt/splunk/etc/deployment-apps/linux\_servers

cd /opt/splunk/etc/deployment-apps/linux\_servers

1. **Create inputs.conf**: Create and edit the inputs.conf file: nano inputs.conf. Add the following content:

[monitor:///var/log/auth.log]

disabled = false

index = main

sourcetype = auth\_log

1. **Create outputs.conf**: Create and edit the outputs.conf file: nano outputs.conf. Add the following content:

[tcpout]

defaultGroup = default-autolb-group

[tcpout:default-autolb-group]

server = 170.77.0.5:9997

\ [tcpout-server://170.77.0.5:9997]

1. **Deploy the App**:
   * In the Splunk web interface, go to **Settings > Forwarder Management**.
   * Create a new server class named linux\_servers.
   * Add the deployment clients (Universal Forwarders) to this server class.
   * Assign the linux\_servers app to the server class.
2. **Verify the Deployment**:
   * Check the /opt/splunkforwarder/etc/apps directory on the client servers to ensure the app is deployed.
   * Open the Splunk web interface and go to the **Search & Reporting** app to verify that logs from the backend server are being indexed.

**Regular Expressions?**

Regular expressions (regex) are sequences of characters that define a search pattern. They are used for matching, extracting, and manipulating text. In Splunk, regular expressions are used to extract fields, filter events, and perform complex searches.

**Key Commands Using Regular Expressions in Splunk**

1. **rex Command**: Extracts fields using regular expression named groups or replaces/substitutes characters in a field using sed expressions.
2. **regex Command**: Filters search results using a regular expression.

**Basic Syntax**

* **Literal Characters**: Match themselves (e.g., abc matches "abc").
* **Metacharacters**: Have special meanings (e.g., . matches any character except a newline).

**Common Metacharacters**

* .: Matches any single character except newline.
* \*: Matches 0 or more of the preceding element.
* +: Matches 1 or more of the preceding element.
* ?: Matches 0 or 1 of the preceding element.
* ^: Matches the start of a string.
* $: Matches the end of a string.
* []: Matches any one of the characters inside the brackets.
* |: Acts as an OR operator.

**Two Ways to Parse Data in Splunk**

1. **Custom Add-ons with Custom Regex**: You can write your own regex to parse data.
2. **Pre-built Add-ons from Splunkbase**: You can use add-ons built by Splunk or third-party vendors that already contain regex for parsing specific types of logs. Instead of writing your own regex, you can use pre-built add-ons. For example, the Splunk Add-on for NGINX

**Understanding Source Types in Splunk**

In Splunk, the **source type** is a critical field that defines how data is parsed and indexed. It helps Splunk understand the format of incoming data and apply the correct field extractions and transformations.

**Importance of Source Types**

1. **Field Extractions**: Field extractions are often defined at the source type level. This means that if the source type is incorrect, Splunk may not extract fields properly.
2. **Built-in Source Types**: Splunk comes with built-in source types for common log formats (e.g., Apache logs, syslogs). These source types have predefined regex patterns for field extraction.
3. **Custom Source Types**: You can create custom source types if your log format does not match any of the built-in types.

**Interactive Field Extractor (IFX)**

The Interactive Field Extractor (IFX) in Splunk is a tool that allows you to extract fields from your log data without writing complex regular expressions (regex). It's particularly useful for quickly setting up field extractions for logs with simple patterns.

**When to Use IFX?**

* **Simple Log Patterns**: When your log data has a simple and consistent format.
* **Quick Analysis**: When you need to quickly analyze data without spending time on writing regex.
* **Non-Production Use**: While IFX is great for quick setups, regex is recommended for production environments due to its flexibility and precision.

**Props.conf and transform.conf (Not Done)**

**What is props.conf?**

props.conf is a configuration file in Splunk that defines how data should be parsed and indexed. It specifies how to handle different source types, including how to break events, apply timestamp extraction, and perform field extractions.

**Key Sections in props.conf**

1. **[source::...]**: Defines settings for data from a specific source.
2. **[host::...]**: Defines settings for data from a specific host.
3. **[sourcetype]**: Defines settings for a specific source type.

**Common Settings in props.conf**

1. **TIME\_PREFIX**: This is a pattern that helps Splunk find the timestamp in your data.
2. **TIME\_FORMAT**: This tells Splunk the format of the timestamp.
3. **MAX\_TIMESTAMP\_LOOKAHEAD**: This tells Splunk how many characters to look ahead to find the timestamp.
4. **SHOULD\_LINEMERGE**: This decides if multiple lines should be combined into one event.
5. **LINE\_BREAKER**: This pattern tells Splunk where to break events into separate lines.
6. **REPORT**: This points to the field extraction rules defined in transforms.conf.

**What is transforms.conf?**

transforms.conf is a configuration file in Splunk that defines how to transform data, including field extractions, lookups, and data masking. It works in conjunction with props.conf to apply these transformations.

**Key Sections in transforms.conf**

1. **[stanza\_name]**: This is the name of the rule.
2. **REGEX**: This is the pattern used to extract fields.
3. **FORMAT**: This tells Splunk how to format the extracted fields.
4. **DEST\_KEY**: This specifies where to store the extracted fields.

**Event Types in Splunk**

Event types in Splunk are a way to categorize and label events based on specific search criteria. They help you quickly identify and analyze events that match certain conditions without having to write complex queries repeatedly.

**How Event Types Work**

1. **Define a Search Query**: Create a search query that matches the events you are interested in.
2. **Save as Event Type**: Save this query as an event type with a meaningful name.
3. **Use Event Type**: Use the event type in future searches to quickly find matching events.

**Example of Pre-built Event Types:** Splunk comes with several pre-built event types for common log sources. For example, Windows logs might have event types for free disk space, login failures.

**Limitations of Event Types**

* **No Pipe Operator**: Event types cannot include a pipe (|) operator after a simple search.
* **No Subsearches**: Event types cannot include subsearches.

**Understanding Tags in Splunk**

Tags in Splunk allow you to assign meaningful names to specific field and value combinations. This can include event types, hosts, sources, or source types. Tags make it easier to categorize and identify data, enhancing the analysis process.

**Example Use Case: I**magine you have network logs with various IP addresses belonging to different subnets. These subnets represent different environments (e.g., development, staging, production). By tagging these IP addresses, you can easily identify which environment they belong to.

**Subnets and Tags**

* **192.160.10.0**: Tag as Singapore\_Production
* **10.77.00.16**: Tag as Mumbai\_Staging
* **10.66.00.16**: Tag as Oregon\_Development

**Splunk Lookups (See from chatgpt)**

Lookups in splunk helps you add extra information to your data by pulling details from an external source like a file (eg: CSV), a key-value store, or a script. Instead of searching raw logs, lookups enrich your data by adding meaningful details, making it easier to understand and analyze.

**Types of Lookups**

1. **CSV Lookups**: Use CSV files to map fields in your events to additional data.
2. **External Lookups**: Use scripts or external databases to fetch additional data.
3. **KV Store Lookups**: Use Splunk's key-value store to store and retrieve data.

**Example Scenario:** Imagine you have a CSV file called users.csv with the following data:

uid, username, department

1066, Claudia Garcia, Engineering

1690, Rutherford Sullivan, Engineering

1815, Vanya Patel, IT

1862, Wei Zhang, Engineering

1916, Alex Martin, Personnel

You also have event data in Splunk that looks like this:

\_time host action uid

2024-11-18 10:52:41 mailsv1 Failed password 1815

2024-11-18 06:23:48 mailsv3 Session closed 1916

2024-11-18 08:18:36 mailsv1 Session closed 1690

You want to add the username and department fields from the users.csv file to your event data based on the uid field.

**Steps to Perform a Lookup**

1. **Upload the Lookup Table**: First, you need to upload the users.csv file to Splunk.
2. **Define the Lookup**: Define the lookup in Splunk so it knows how to use the users.csv file.
3. **Use the Lookup Command**: Use the lookup command in your search to add the additional information to your events.

**Example Search:** Here's how you can use the lookup command to enrich your event data:

index=main | lookup users.csv uid OUTPUT username, department

**Explanation**:

* index=main: Searches the main index for events.
* lookup users.csv uid OUTPUT username, department: Uses the lookup command to match the uid field in your events with the uid field in the users.csv file. It then adds the username and department fields from the CSV file to your events.

**Result:** After running the search, your enriched event data will look like this:

\_time host action uid username department

2024-11-18 10:52:41 mailsv1 Failed password 1815 Vanya Patel IT

2024-11-18 06:23:48 mailsv3 Session closed 1916 Alex Martin Personnel

2024-11-18 08:18:36 mailsv1 Session closed 1690 Rutherford Sullivan Engineering

**Understanding Splunk Alerts**

A Splunk alert is a notification triggered by a specific condition in your data. Alerts help you monitor your data in real-time or on a schedule and notify you when certain conditions are met. This can be useful for detecting issues, monitoring performance, or responding to security threats.

**How Alerts Work**

1. **Saved Search**: An alert is based on a saved search. This search runs periodically or in real-time to check for specific conditions in your data.
2. **Trigger Conditions**: You define conditions that, when met, will trigger the alert. These conditions can be based on the number of search results, specific field values, or custom criteria.
3. **Alert Actions**: When an alert is triggered, Splunk can perform various actions, such as sending an email, running a script, or creating a ticket in a ticketing system.

**Types of Alerts**

1. **Scheduled Alerts**: These alerts run at regular intervals (e.g., every hour, daily) and check for conditions in the data collected during that period.
2. **Real-Time Alerts**: These alerts continuously monitor data as it comes in and trigger immediately when the conditions are met.

**Example 1: Scheduled Alert**

**Use Case**: Monitor for errors in your application logs and send an email if there are more than five errors in the last 24 hours.

**Steps**:

1. **Create a Search**: index=main sourcetype=app\_logs error

This search looks for events with the word "error" in the app\_logs sourcetype.

1. **Save the Search as an Alert**:
   * Go to **Save As > Alert**.
   * **Title**: Errors in the last 24 hours
   * **Alert Type**: Scheduled
   * **Time Range**: Last 24 hours
   * **Schedule**: Run every day at 10:00 AM
   * **Trigger Condition**: Number of results > 5
   * **Alert Action**: Send an email to the IT team

**Result**: Every day at 10:00 AM, this alert runs the search. If there are more than five errors in the last 24 hours, it sends an email notification to the IT team.

**Splunk Access Control**

Splunk access control is a system that manages who can access what data and perform which actions within Splunk. It uses users, roles, and permissions to control access.

**Key Components**

1. **Users**: Individual accounts that people use to log into Splunk.
2. **Roles**: Groups of permissions that can be assigned to users.
3. **Permissions (Capabilities)**: Specific actions that users can perform, such as searching data or managing indexes.

**How It Works**

1. **Create Users**: Users are created and assigned to roles.
2. **Define Roles**: Roles are defined with specific permissions.
3. **Assign Permissions**: Permissions are assigned to roles, and users inherit these permissions through their roles.

**Example Scenario:** Imagine you have a team of analysts who need access to search data but should not have administrative privileges.

1. **Create Analyst Role**:
   * Name: analyst
   * Capabilities: search, rtsearch
   * Indexes: main, logs
2. **Create Users**:
   * User: analyst1, Role: analyst
   * User: analyst2, Role: analyst
3. **Verify Access**:
   * Log in as analyst1 and verify that the user can search data but cannot perform administrative tasks.

**Distributed Splunk Architecture**

Distributed Splunk architecture is designed to handle large volumes of data efficiently by distributing the workload across multiple components. This architecture allows Splunk to scale horizontally, providing better performance, reliability, and manageability.

**Key Components of Distributed Splunk Architecture**

1. **Forwarders**: Collect and forward data to indexers.
2. **Indexers**: Index and store the data, making it searchable.
3. **Search Heads**: Provide the interface for searching and analyzing the data.
4. **Deployment Server**: Manages configurations and apps for forwarders.
5. **Cluster Master**: Manages indexer clusters.
6. **Deployer**: Manages search head clusters.

**How It Works**

1. **Data Collection**:
   * **Forwarders** collect data from various sources (e.g., logs, metrics) and send it to **indexers**.
   * Example: A forwarder on a web server collects access logs and sends them to the indexers.
2. **Data Indexing**:
   * **Indexers** receive data from forwarders, index it, and store it in a searchable format.
   * Example: Indexers process the web server logs, making them searchable by fields like IP address, timestamp, and status code.
3. **Data Searching**:
   * **Search Heads** provide a user interface for querying and analyzing the indexed data.
   * Example: An analyst uses the search head to query the web server logs for all 500 status codes in the last 24 hours.

**Example Scenario**

Imagine you have a large organization with multiple data sources, such as web servers, application servers, and databases. You want to collect, index, and analyze this data using Splunk.

1. **Forwarders**:
   * Installed on each data source to collect logs and metrics.
   * Example: Universal forwarders on web servers collect access logs.
2. **Indexers**:
   * Receive data from forwarders and index it.
   * Example: Indexers store and index the access logs from all web servers.
3. **Search Heads**:
   * Provide a web interface for users to search and analyze the data.
   * Example: Analysts use search heads to create dashboards and alerts based on the indexed data.

**License Master**

The License Master (also known as the License Manager) in Splunk is a centralized server that manages and monitors the licensing for your entire Splunk deployment.

**Key Points to Remember**

1. **Data Ingestion**: Splunk Enterprise ingests external data, indexes it, and stores it on disk.
2. **License Limits**: Your license specifies how much data you can index per day. For example, a 1GB license allows you to index up to 1GB of data per day.
3. **License Requirement**: All Splunk Enterprise instances require a license.
4. **Standalone vs. Distributed**: In a standalone setup, you can install the license locally on the Splunk instance. In a distributed environment, you need a License Master.
5. **License Master**: Manages licenses for multiple Splunk instances, ensuring compliance and monitoring usage.

**License Master Architecture:** In a distributed Splunk environment, you have multiple Splunk instances (e.g., indexers, search heads). Instead of installing the license on each instance, you use a License Master to manage the licenses centrally.

**Components**

1. **License Master**: A Splunk Enterprise instance where the license is installed.
2. **License Slaves**: Other Splunk instances (indexers, search heads) that connect to the License Master to report their data usage.

**Example Scenario:** Imagine you have a distributed Splunk deployment with multiple indexers and search heads. You want to ensure that your deployment stays within the licensed data volume limits.

1. **License Master**:
   * Install and configure a License Master on a central server.
   * Upload the Splunk Enterprise license file.
2. **License Slaves**:
   * Configure all indexers and search heads as license slaves.
   * Point them to the License Master for license management.
3. **Monitor Usage**:
   * Regularly check the license usage reports on the License Master.
   * Set up alerts to notify you if the data volume approaches the licensed limit.

**Splunk License Pools (See from Chatgpt)**

**1. License Stack and Default Pool**

* **License Stack**: When you install a Splunk license, it resides in a license stack, also known as a Splunk enterprise stack.
* **Default Pool**: This stack has a default license pool called the "auto-generated pool enterprise." Any server (referred to as a slave) that connects to the license master can access this default pool.

**2. Volume Capacity**

* The default pool has a volume capacity, for example, 10 GB. This means any connected server can use up to 10 GB of data from this pool.

**3. Problem with Default Pool**

* **Shared Usage**: If multiple teams (e.g., Development, SRE, Security) connect their Splunk instances to the license master, they all share the same 10 GB pool.
* **Resource Contention**: If one team (e.g., Development) uses 8 GB, only 2 GB is left for the other teams. This can lead to issues where some teams might not have enough capacity to index their data.

**4. Creating Specific Pools**

To avoid the problem of resource contention, you can create specific pools for each team:

* **Development Pool**: Allocate 3 GB.
* **SRE Pool**: Allocate 3 GB.
* **Security Pool**: Allocate 4 GB.

This way, each team has a dedicated portion of the total license volume, ensuring fair usage.

**Masking Sensitive Data**

**Masking Sensitive Data in Splunk**

**1. Why Mask Sensitive Data?**

* **Sensitive Information**: Log files might contain sensitive information like credit card numbers or social security numbers.
* **Privacy**: To protect privacy, you need to mask this information so that analysts cannot see the original data.

**2. When to Mask Data?**

* **Before Indexing**: Masking should be done before the data gets indexed. This ensures that the sensitive information is transformed before it is stored in Splunk.

**3. Stages in Data Processing**

* **Parsing Stage**: This is where data is prepared for indexing. It includes actions like extracting default fields (host, source, source type), character normalization, and masking sensitive data.
* **Indexing Stage**: This is where the data is indexed and stored. Masking should be completed before this stage.

**4. Example Scenario**

* **Sample Log File**: Let's say you have a log file with credit card information.
* **Transformation**: You want to transform this information so that it appears masked (e.g., XXXX-XXXX-XXXX-1234) when indexed.

**Search Head**

**Search Head**: A component in Splunk Enterprise responsible for managing search-related functions. It sends search requests to indexers, retrieves the results, merges them, and sends the final results back to the user.

**How Does a Search Head Work?**

* **Request and Response**: The search head sends search requests to multiple indexer nodes. Each indexer processes the request and sends back the results. The search head then merges these results and presents them to the user.
* **No Data Storage**: The search head does not store any data itself. It only manages and coordinates the search process.

**Functions of a Search Head**

* **Search Management**: Handles search queries and coordinates with indexers to retrieve results.
* **Dashboards and Reports**: Allows users to build and view dashboards and reports.
* **Data Models**: Supports building and accelerating data models.
* **Alerting**: Manages alerting functionalities to notify users of specific conditions or events.

**Splunk Monitoring Console**

* **Monitoring Console**: A tool in Splunk Enterprise that provides detailed information about your Splunk deployment. It helps monitor server performance, indexing performance, search performance, and more.

**Why is Monitoring Important?**

* **Performance Issues**: If users report slow searches or lag, the monitoring console helps identify and troubleshoot these issues.

**Features of the Monitoring Console: Pre-built Dashboards**: The console comes with pre-built dashboards that provide visibility into various aspects of your Splunk deployment, such as indexing performance, resource usage, and license usage.

**Key Areas Monitored**

* **Indexing Performance**: Monitors how well data is being indexed.
* **Search Performance**: Tracks the performance of search queries.
* **Resource Usage**: Monitors CPU, memory, and disk usage.
* **License Usage**: Keeps track of how much of your Splunk license is being used.
* **HTTP Event Collector Performance**: Monitors the performance of data collection via HTTP.
* **TCP Performance**: Tracks the performance of data collection over TCP.

**Indexer Clustering**

* **Indexer Clustering**: A method to ensure high availability and reliability of your data in Splunk. It involves multiple indexers working together to store and manage data, providing redundancy and load balancing.

**2. Why Use Indexer Clustering?**

* **High Availability**: Ensures that your data is always accessible, even if one or more indexers fail.
* **Data Redundancy**: Copies of data are stored across multiple indexers, preventing data loss.
* **Load Balancing**: Distributes the indexing load across multiple indexers, improving performance.

**3. Components of Indexer Clustering**

* **Cluster Master**: The central component that manages the cluster. It coordinates the activities of the peer nodes (indexers) and ensures data replication.
* **Peer Nodes (Indexers)**: The indexers that store and manage the data. They work together to replicate data and handle search requests.
* **Search Head**: Sends search requests to the peer nodes and retrieves the results. It does not store data itself.

**4. How Indexer Clustering Works**

1. **Data Ingestion**: Data is ingested into the cluster through forwarders.
2. **Data Replication**: The cluster master ensures that data is replicated across multiple peer nodes. For example, if you have a replication factor of 3, each piece of data will be stored on three different indexers.
3. **Search Requests**: When a search request is made, the search head sends the request to the peer nodes. Each peer node processes its portion of the data and sends the results back to the search head.
4. **Result Merging**: The search head merges the results from the peer nodes and presents them to the user.

**Example Architecture**

* **Master Node**: Coordinates the cluster and manages configurations.
* **Peer Nodes**: Store and replicate data. The number of peer nodes can vary based on the size of your data. For example, you might have 2, 10, or even 20 peer nodes.
* **Search Head**: Sends search requests to the peer nodes and retrieves the results.

Important Points:

* **Replication Factor**: Determines how many copies of data the cluster maintains. For example, a replication factor of 3 means three identical copies of data are stored on separate nodes.
* **Search Factor**: Determines the number of immediately searchable copies of data. Searchable copies include both raw data and index files, making searches faster.
* **Security Key**: Used for authenticating communication between the master node and peer nodes.
* **Cluster Label**: A name to identify the cluster.

**Configuration Bundles in Splunk Indexer Clustering**

* **Configuration Bundle**: A set of configurations that you want to apply to all peer nodes in a Splunk indexer cluster. This ensures consistency across the cluster.

**2. Why Use Configuration Bundles?**

* **Consistency**: Applying configurations via the master node ensures that all peer nodes have the same settings.
* **Efficiency**: It is more efficient to manage configurations centrally rather than manually updating each peer node.

**3. Where to Store Configuration Bundles?**

* **Master Node**: Store the configuration bundle in the ETC/master-apps directory.
* **Peer Nodes**: Once pushed, the configurations will appear in the ETC/slave-apps directory on each peer node.

**Rolling Back Changes**

* **Rollback**: If a configuration change causes issues, you can roll back to the previous configuration.

splunk rollback cluster-bundle

**Indexer Discovery in Splunk**

**Indexer Discovery**: A method where the Universal Forwarder dynamically discovers the available peer indexers from the master indexer. This avoids the need to hard-code IP addresses of peer indexers in the forwarder's configuration.

**2. Why Use Indexer Discovery?**

* **Scalability**: Automatically adapts to changes in the cluster, such as adding or removing peer indexers.
* **Maintenance**: Reduces the need for manual updates to the forwarder's configuration.

**3. How Does Indexer Discovery Work?**

* **Query to Master Indexer**: The Universal Forwarder queries the master indexer for a list of available peer indexers.
* **Load Balancing**: The forwarder uses the list to distribute data across the peer indexers using load balancing.

**Search Head Clustering**

**Search Head Cluster**: A group of Splunk search heads that work together to manage search requests, distribute search jobs, and provide high availability and load balancing for search operations.

**2. Why Use a Search Head Cluster?**

* **High Availability**: Ensures that search functionality remains available even if one or more search heads fail.
* **Load Balancing**: Distributes search queries across multiple search heads to optimize performance.
* **Scalability**: Allows you to handle more search requests by adding more search heads to the cluster.

**3. Components of a Search Head Cluster**

* **Search Heads**: Nodes that handle search requests and manage search jobs.
* **Cluster Master (Deployer)**: Manages the configuration and coordination of the search head cluster.
* **Search Peers (Indexers)**: Nodes that store and index data. They work with the search heads to execute search queries.

**4. How Search Head Clustering Works**

* **Search Distribution**: When a user submits a search query, the search head distributes the query to the appropriate indexers.
* **Result Aggregation**: Indexers process the query and send the results back to the search head, which aggregates and presents the results to the user.
* **Configuration Management**: The cluster master (deployer) manages and distributes configuration updates to all search heads in the cluster.