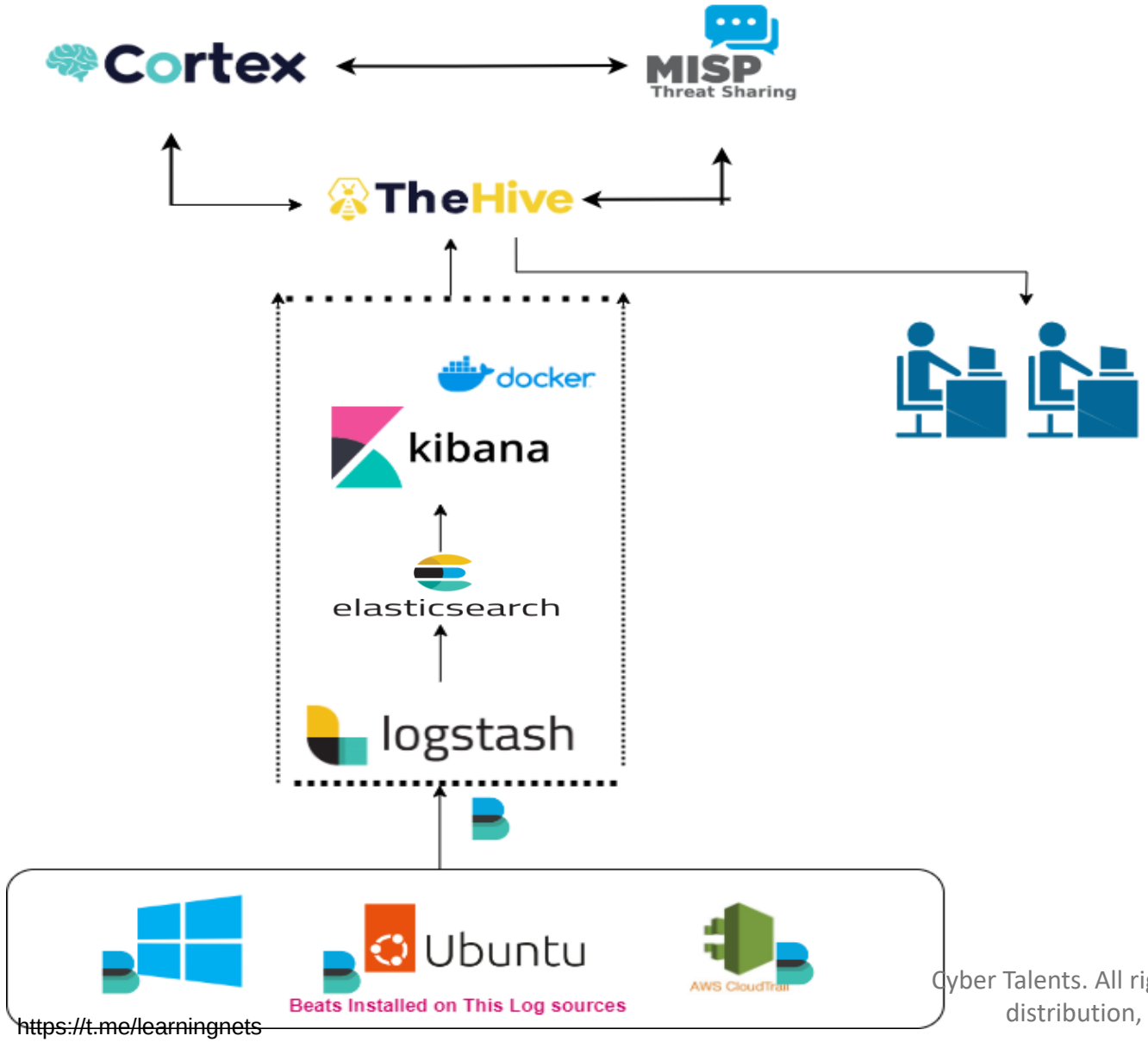
Build SOC With Open Source tools



Course Content

Section (A): Configuration and Installation of Elasticsearch

- Lecture 1: ELK Essentials: Exploring Elasticsearch Architecture and Components
- Lecture 2: Understanding the ELK Stack: A Step-by-Step Workflow Example
- Lecture 3: Introduction to Containerization and Docker Compose
- Lecture 4: Lab: Set up your AWS Account
- Lecture 5: Lab: Setting up EC2 for Elasticsearch: A Step-by-Step Guide
- Lecture 6: Lab: Installing Elasticsearch on EC2: A Step-by-Step Guide
- Lecture 7: Lab: Filebeat Essentials: Step-by-Step Configuration and Installation
- Section (B): Getting Started with MISP (Malware Information Sharing Platform)

Section (B): Getting Started with MISP (Malware Information Sharing Platform)

- Lecture 1: An Introduction to Malware Information Sharing Platform
- Lecture 2: Lab: MISP Installation Guide: Step-by-Step Setup
- Lecture 3: Lab: MISP: How to Add and Manage Threat Feeds
- Lecture 4: Lab: MISP: How to Create and Manage Events

Section (C): Getting Started with Cortex

- Lecture 1: An Introduction to Cortex
- Lecture 2: Lab: Cortex Installation and Configuration Guide: Step-by-Step Instructions
- Lecture 3: Lab: Cortex Analyzer Installation Guide: Step-by-Step Instructions

Section (D): Getting Started with TheHive

- Lecture 1: An Introduction to The Hive
- Lecture 2: The Hive Installation Guide: Step-by-Step Setup Instructions

Section 6: Integrating Tools

- Lecture 1: How to Integrate Hive with ELK: Setup and Configuration Guide
- Lecture 2: How to Integrate Hive and Cortex: Setup and Configuration Guide (Part 1)
- Lecture 3: How to Integrate Hive and Cortex: Setup and Configuration Guide (Part 2)
- Lecture 4: How to Integrate Hive and MISP: Setup and Configuration Guide

Explain ELK (Elastic search) architecture and Components

Beats: Lightweight data shippers that you install as agents on your servers to send data to Logstash or Elasticsearch.

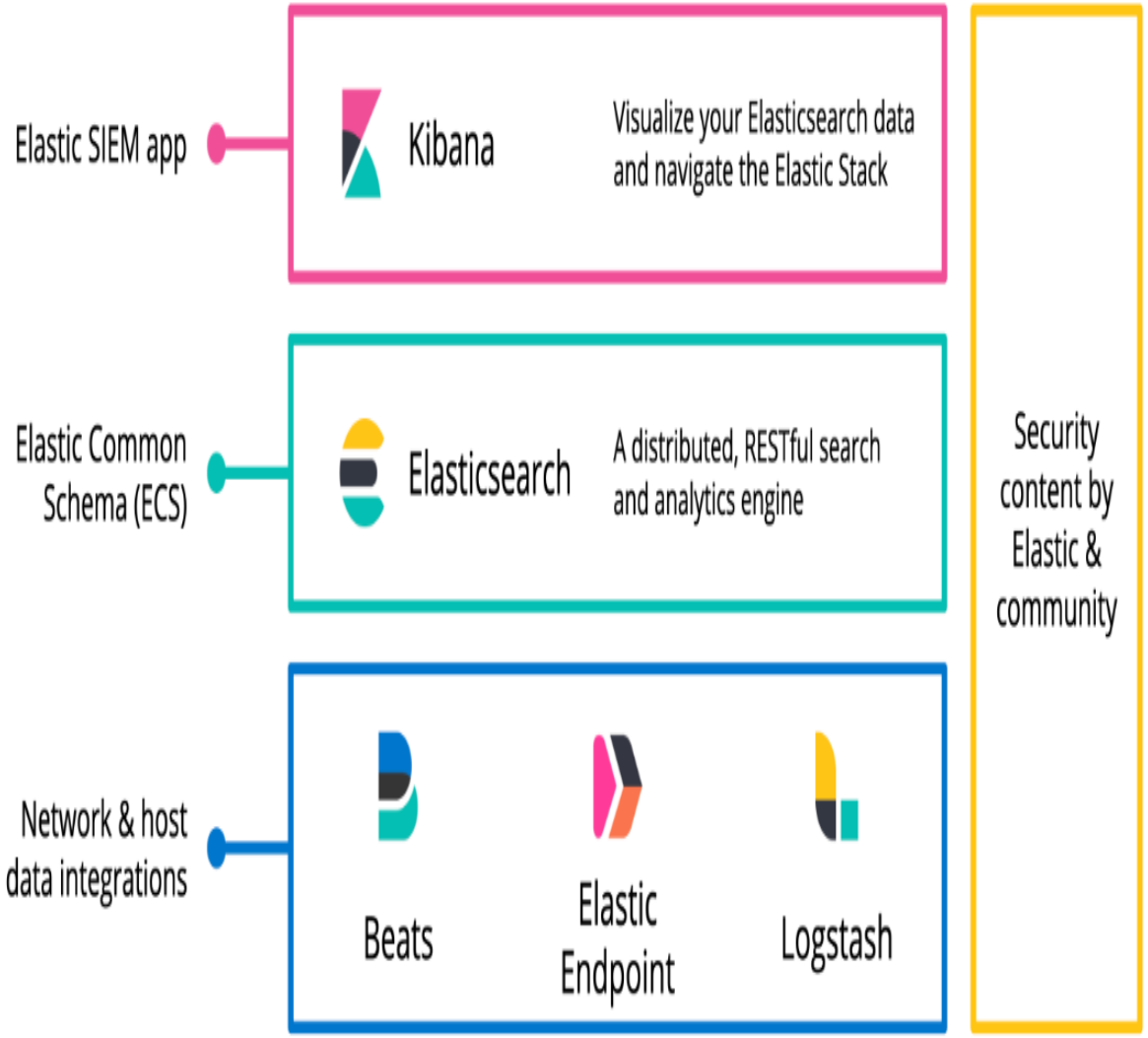
Logstash: A data processing pipeline that ingests data from multiple sources, transforms it, and then sends it to Elasticsearch.

Elasticsearch: A distributed search and analytics engine that stores and indexes the data, enabling quick searches and complex queries.

Kibana: A visualization tool that sits on top of Elasticsearch, providing a web interface for exploring and visualizing the data.

Elastic Common Schema (ECS): A standardized schema for the Elasticsearch SIEM module, facilitating the organization and correlation of data fields.

SIEM App in Kibana: A dedicated workspace for security analysts within Kibana, offering pre-built visualizations, dashboards, and tools for threat detection and incident response.



Understandingthe ELK Stack: A Step-by-Step Workflow Example

Example Workflow

- Let’s go through a simple example:

1. Collecting Data:

1. Imagine you have a web server and a database server. You install Filebeat on both servers to collect log files. Filebeat sends these logs to Logstash.

2. Processing Data:

1. Logstash receives the logs and processes them. It extracts useful information, like IP addresses, URLs, and error messages, and then sends the processed logs to Elasticsearch.

3. Storing and Indexing Data:

1. Elasticsearch stores the processed logs and indexes them. This allows you to quickly search for specific information, like how many users accessed a particular page on your website.

4. Visualizing Data:

1. In Kibana, you create a dashboard to visualize your web server and database logs. You can see charts showing the number of visitors, response times, and error rates.

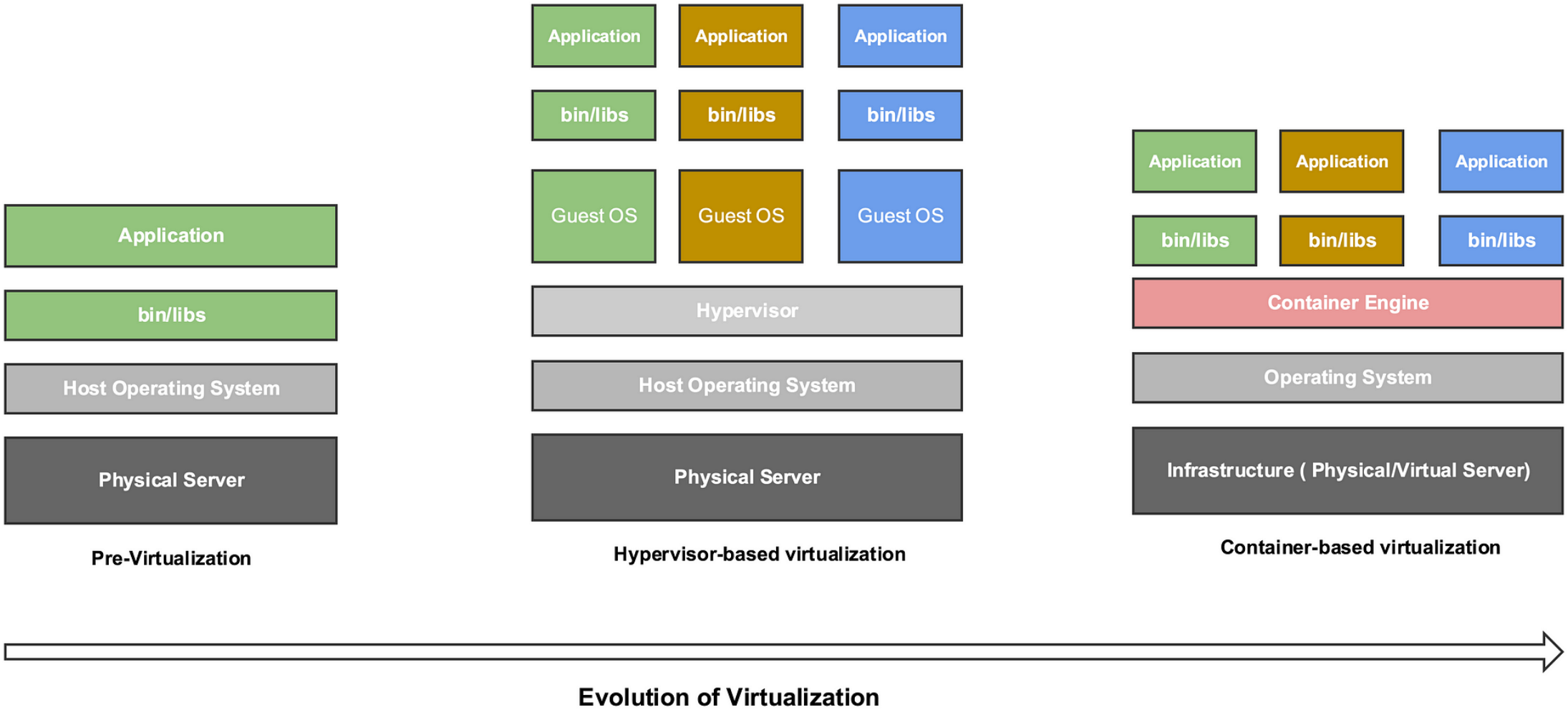
5. Analyzing Security:

1. Using the SIEM App in Kibana, you monitor your logs for unusual activity, like repeated failed login attempts or unexpected spikes in traffic. ECS ensures that the data is organized and easy to analyze.

Conclusion

- In summary, the ELK Stack is a powerful toolset for collecting, processing, storing, and visualizing data. By using Beats, Logstash, Elasticsearch, and Kibana, you can gain valuable insights from your data and monitor it effectively. The SIEM App and ECS enhance security analysis, making it easier to detect and respond to threats.

Introduction to CONTAINER and DOCKER



Introduction to CONTAINER and DOCKER

Shipping Container Analogy:

Imagine shipping goods worldwide in a container, ensuring safety and portability.

Software Containers:

Software containers are like shipping containers for applications, bundling code, dependencies, and configurations.

Docker Overview:

Docker is an open-source platform for creating, deploying, and running containers.
Ensures applications run consistently across different environments.

Docker Compose:

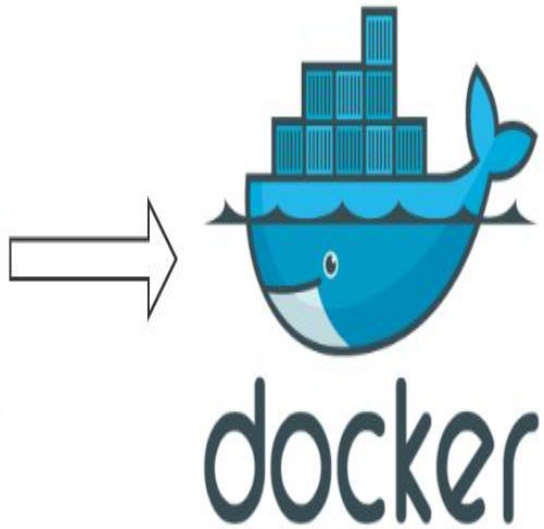
Acts as a logistics manager for Docker containers.
Manages multiple containers using a single configuration file.

Benefits:

Portability: Move containers between environments effortlessly.
Consistency: Ensure applications behave uniformly across development, testing, and production.

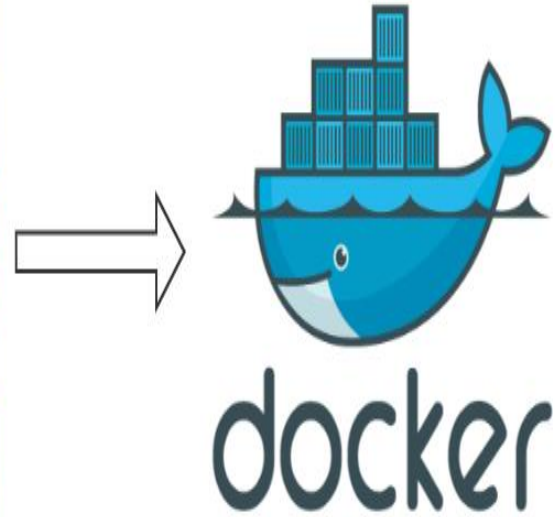
Conclusion:

Docker and Docker Compose simplify deployment and management of applications, akin to organizing and transporting goods in shipping containers.



Why Use Docker for ELK Stack?

Simplified Setup and Management: Docker containers simplify the setup of the ELK stack (Elasticsearch, Logstash, Kibana).



Advantages of Docker for ELK:

- Pre-configured Docker images for quick setup.
- Ensures the ELK stack runs the same in every environment.
- Each component runs in its own container, efficiently managing resources.
- Easily scale components by starting more containers.
- Docker containers can run on any system that supports Docker.

Advantages of Docker Compose for ELK:

- Define all services in a single docker-compose.yml file.
- Start all ELK stack components with a single command.
- Define configurations for different environments (development, testing, production).
- Ensures services start in the correct order and manage dependencies (e.g., Kibana waits for Elasticsearch).

Step-by-Step Guide to Installing and Configuring Elasticsearch and Kibana Using Docker

Step (1) sudo apt update : This command updates the list of available packages and their versions, but it does not install or upgrade any packages.

Step (2) sudo apt upgrade: This command installs the newest versions of all packages currently installed on your system. It's a good practice to keep the system up-to-date.

Step (3) sudo apt install docker-compose : Docker Compose is a tool for defining and running multi-container Docker applications. This command installs Docker Compose.

Step (4) sudo apt install docker. io : Docker is a platform used to develop, ship, and run applications inside containers. This command installs Docker.

Step (5) mkdir elastic : This command creates a new directory named elastic where we will store the configuration files for Elasticsearch and Kibana.

Step (6)cd elastic: This command changes the current directory to elastic.

Step (7) vi docker-compose. yml : This command opens the docker-compose.yml file in the vi editor. You need to download the code from the attachment section and paste it into this file.

Step (6) sudo docker-compose up - : This command starts the services defined in the docker-compose.yml file. The -d flag runs the containers in detached mode (in the background).

Step(7): Verify Services are Listening on Ports 9200 and 5601 using "netstat -1tpnd

Stpep(8): Access Kibana from this url : http://<<Public_IP_of_EC2>>:5601

By following these commands, you will set up Elasticsearch and Kibana on your VM using Docker and Docker Compose.

File beat: installation and configuration

Step(1) Download the Filebeat Debian package : **curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.15.0-amd64.deb**

Step (2) Installing Filebeat : **sudo dpkg -i filebeat-7.15.0-amd64.deb**

Step(3) Configuring Filebeat : **sudo vi /etc/filebeat/filebeat.yml (This is the Filebeat configuration file that needs to be edited.)**

Step (4) Setting Output to Elasticsearch:

```
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["http://x.x.x.x:9200"] #this is your Elasticsearch IP address
  # Protocol - either `http` (default) or `https`.
  protocol: "http"
  # Authentication credentials - either API key or username/password.
  #api_key: "id:api_key"
  username: "<elastic username>"
  password: "<elastic password>"
```

- Hosts: Set the IP address of your Elasticsearch server.
- Protocol: Set to http (default) or https.
- Authentication: Provide either an API key or a username and password.

Filebeat: installation and configuration (2)

Setting Up Kibana:

setup.kibana:
host: "http://x.x.x.x:5601" # this is your Kibana IP address

Enabling Filebeat Module:

cd /etc/filebeat/modules.d/

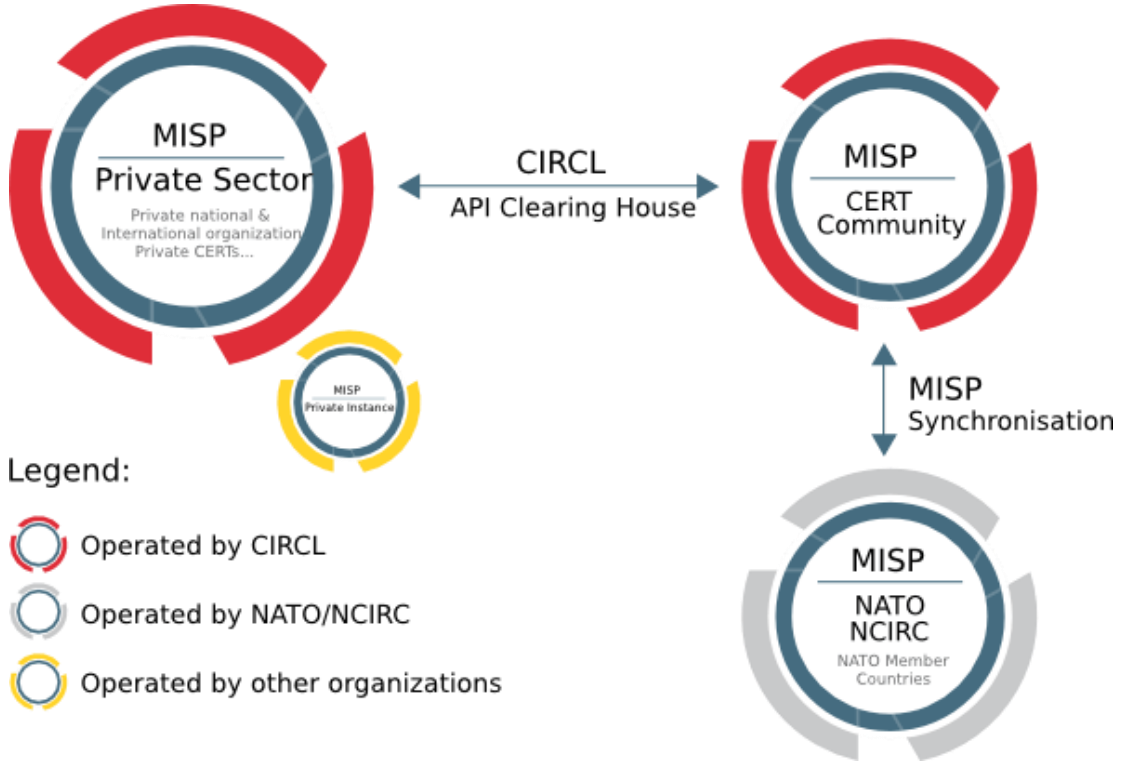
sudo filebeat modules enable system

- Command: cd modules.d/: Change directory to modules.d, where the Filebeat modules are located.
- Command: sudo filebeat modules enable system: Enable the system module, which collects and parses logs from the operating system

Starting Filebeat Service: sudo service filebeat start

Understanding MISP - Malware Information Sharing Platform

- MISP is an open-source platform for sharing threat intelligence.
- Stores structured data like IP addresses, domains, and email addresses.
- Enables collaborative defense measures within organizations and across communities.



MISP(Malware Information Sharing Platform)

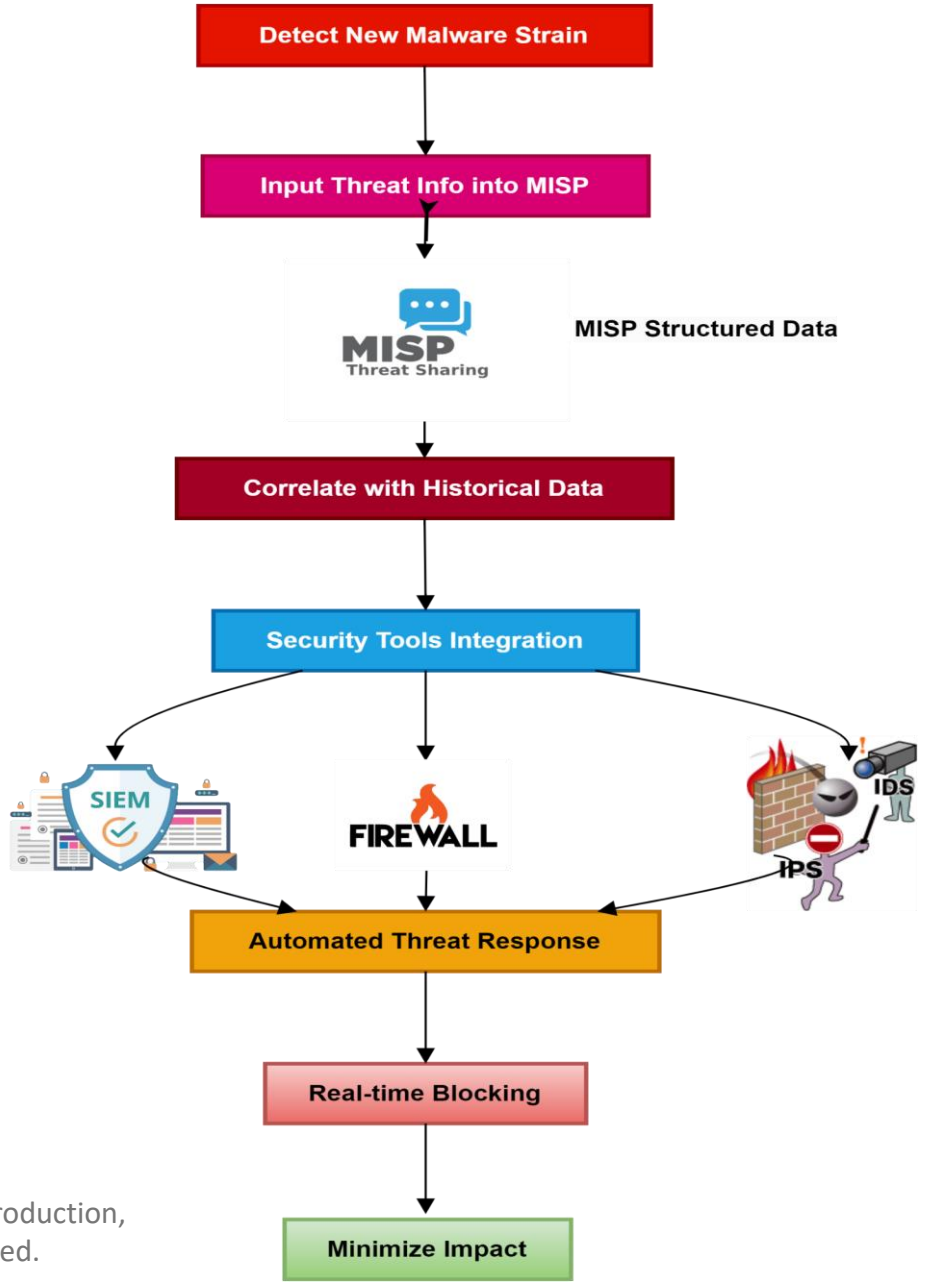
Imagine a scenario where a CyberShield member organization detects a new malware strain targeting their network. Using MISP, they quickly input detailed threat information such as file hashes, malicious URLs, and attack behaviors.

- **MISP's structured format ensures all data is interconnected with historical threat intelligence.**
- **Future detections automatically correlate with relevant historical data for comprehensive context.**

Now, automation comes into play. Security tools integrated with MISP—SIEM systems, firewalls, and IDS/IPS—directly ingest this structured threat data.

CyberShield configures their security infrastructure to automatically respond.

Example: Upon detecting a malicious file hash or URL from MISP, firewalls block access in real-time.



MISP –Installation Guide

To install MISP all you need to do is the following on a clean supported distribution.

Step (1) Please check the installer options first to make the best choice for your install

wget --no-cache -O /tmp/INSTALL.sh https://raw.githubusercontent.com/MISP/MISP/2.4/INSTALL/INSTALL.sh
bash /tmp/INSTALL.sh

Step (2) This will install MISP Core:

wget --no-cache -O /tmp/INSTALL.sh https://raw.githubusercontent.com/MISP/MISP/2.4/INSTALL/INSTALL.sh
bash /tmp/INSTALL.sh –c

Step (3):

MSSP Access bile via the Public IP of Your EC2 Instance
Use the Following default Credentials to login

User: admin@admin.test
Password: admin

Cortex

Cortex is an open-source security tool created by TheHive Project. It's designed to help SOC teams analyze and respond to security incidents efficiently. Think of Cortex as a highly skilled assistant that helps you understand and react to potential threats quickly.

Key Features:

Analyzer Management: Cortex supports various analyzers, which are like tools in a toolbox. Each analyzer performs specific tasks, such as extracting information from files or checking IP addresses against threat intelligence feeds.

Automated Analysis: Cortex can automatically analyze data, saving time and effort. For example, it can check a suspicious IP address against multiple threat databases at once.

Scalability: Cortex can grow with your needs. Whether you're a small team or a large organization, Cortex can handle increasing workloads by adding more instances.

API Integration: Cortex offers a REST API, which means other tools and systems can easily interact with it. This is useful for automating workflows.

Extensibility: You can create new analyzers to extend its capabilities. If you need a custom analysis tool, Cortex makes it easy to integrate it.

Security: Cortex includes features like access control and authentication to ensure only authorized users can interact with it.

Case study: Enhancing SOC Operations with Cortex

1) TechSecure Inc. Overview

- Industry: Cybersecurity
- Employees: 500
- SOC Team: 10 members
- Key Challenge: Managing the increasing number of security alerts and incidents efficiently.

2) The Challenge:

- Increasing volume of security alerts.
- Manual analysis is time-consuming and prone to errors.
- Difficulty in prioritizing incidents.

3) Real-World Example:

- Suspicious IP address detected.
- Multiple alerts triggered.
- Manual investigation required significant time and resources

4) Initial Response

- Manual investigation.
- Checking against threat intelligence feeds.
- Delayed response times.

5) Implementing Cortex:

- Automatic analysis of suspicious IP addresses.
- Multiple analyzers for comprehensive results.
- Integration with threat intelligence feeds.

6) Automation in Action:

- Cortex automatically analyzes the IP address.
- Checks against threat intelligence databases.
- Cross-references with past incidents.

7) Results and Analysis:

- Detailed report generated within minutes.
- Information on known malicious activity.
- Recommended actions provided.

8) Improved Response Time:

- Faster incident response times.
- Improved efficiency in handling alerts.
- Reduced manual workload for the SOC team.

9) Benefits and Outcomes:

- Higher volume of alerts managed.
- Reduced overwhelm for the SOC team.
- Prompt response to critical incidents

CORTEX Installation Guide

1. Update System and Install Java

sudo apt-get update
sudo apt install openjdk-8-jre-headless
java –version

2. Install Elasticsearch

Add Elasticsearch GPG Key: wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -

Add Elasticsearch Repository: echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list

Install Elasticsearch:

sudo apt-get update
sudo apt install elasticsearch

3. Configure Elasticsearch:

Edit the Elasticsearch configuration file: sudo vi /etc/elasticsearch/elasticsearch.yml /etc/elasticsearch/elasticsearch.yml

Uncomment and modify the following lines:

```
network.host: 0.0.0.0       # private address
http.port: 9200             # Uncomment this line
discovery.seed_hosts:       # Uncomment this line
node.name: node1            # Uncomment and set the node name
cluster.initial_master_nodes: ["node1"]  # Uncomment and configure accordingly
cluster.name: hive          # Set the cluster name
thread_pool.search.queue_size: 100000  # Adjust the queue size
```

CORTEX Installation Guide

4. Start Elasticsearch Service

sudo systemctl enable elasticsearch.service
sudo systemctl start elasticsearch.service
sudo systemctl status elasticsearch.service

5. Setup APT Configuration for Cortex

Add TheHive Project GPG Key: curl https://raw.githubusercontent.com/TheHive-Project/TheHive/master/PGP-PUBLIC-KEY | sudo apt-key add -

Add TheHive Project Repository:echo 'deb https://deb.thehive-project.org release main' | sudo tee -a /etc/apt/sources.list.d/thehive-project.list

Install Cortex:

sudo apt-get update
sudo apt install cortex

6. Configure Cortex

Create Configuration Directory: sudo cat /etc/cortex/application.conf

Generate and Add Secret Key to Configuration File:

```
(cat << _EOF_
# Secret key
# ~~~~~
# The secret key is used to secure cryptographic functions.
# If you deploy your application to several instances be sure to use the same key!
play.http.secret.key="$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 64 | head -n 1)"
_EOF_
) | sudo tee -a /etc/cortex/application.conf
```

CORTEX Installation Guide

Add the following lines:

sudo cat /etc/cortex/application.conf

```
## ElasticSearch
search {
  uri = "http://Cortex_vm_ip:9200"
}
```

7. Start Cortex Service

A
tail -f /var/log/cortex/application.log # Verify if Cortex is starting successfully

8. Access Cortex UI

http://Cortex_VM_IP:9001

CORTEX Installation Guide

Cortex, a key component of TheHive Project, allows security analysts to automate the analysis of observables and manage incident response workflows. Setting up analyzers and responders (referred to as neurons) in Cortex involves the following steps:

OverviewNeurons (Analyzers and Responders):

Previously, neurons were embedded in the Docker image, but this is no longer the case.The recommended method to run neurons is using Docker.Official neurons have their own Docker images available on DockerHub under the cortexneurons organization.The catalog of available neurons is provided to Cortex using the --analyzer-url and --responder-url parameters.Dependencies:

All analyzers and responders supported by TheHive Project are written in Python 2 or 3.They do not require a build phase but do require the installation of dependencies.

Steps to Set Up Analyzers and Responders :

Step (1) Install pip for Python 2Ensure pip for Python 2 is installed:

wget https://bootstrap.pypa.io/pip/2.7/get-pip.py
python2 get-pip.py

Step (2): Get python2
sudo apt install python2.7

Step (3) :Dependencies & setup tools:
sudo apt-get install -y --no-install-recommends python3-pip python2.7-dev python3- pip python3-dev ssdeep libfuzzy-dev libfuzzy2 libimage-exiftool-perl libmagic1 build-essential git libssl-dev

sudo apt-get install -y --no-install-recommends python3-pip python2.7-dev python3-pip python3-dev ssdeep libfuzzy-dev libfuzzy2 libimage-exiftool-perl libmagic1 build-essential git libssl-dev

sudo pip install -U pip setuptools && sudo pip3 install -U pip setuptools

Step (4) Clone the Cortex-Analyzers repository to your desired directory:
git clone https://github.com/TheHive-Project/Cortex-Analyzers

CORTEX Installation Guide

Install Analyzers:

```
for I in $(find Cortex-Analyzers -name 'requirements.txt'); do sudo -H pip install -r $I; done && \

for I in $(find Cortex-Analyzers -name 'requirements.txt'); do sudo -H pip3 install -r $I || true; done
```

Modify the application.conf file to point to the analyzers' directory: /etc/cortex/application.conf

```
analyzer {    # Directory that holds analyzers

path = [       "/path/to/default/analyzers",
               "/path/to/my/own/analyzers" # e.g., /opt/cortex/Cortex-Analyzers/analyzers
          ]
}
```

“HIVE” installation and configuration guide

Ref: Step by step guide - TheHive Project Documentation (thehive-project.org)

Step (1) Installing and Configuring OpenJDK 8 for The Hive Tool:

apt-get install -y openjdk-8-jre-headless

echo JAVA_HOME="/usr/lib/jvm/java-8-openjdk-amd64" >> /etc/environment

export JAVA_HOME="/usr/lib/jvm/java-8-openjdk-amd64”

sudo apt install open-jdk-8-jdk –y

Step (2) : Cassandra database Installation:

Adding the Apache Cassandra Repository : echo "deb https://debian.cassandra.apache.org 40x main" | sudo tee -a /etc/apt/sources.list.d/cassandra.sources
A
Step (3) : Adding Apache Cassandra GPG Keys: curl https://downloads.apache.org/cassandra/KEYS | sudo apt-key add –

Step(4): sudo apt-get update
Step (5) : sudo apt install Cassandra
- A

“HIVE” installation and configuration guide

Step(6):Configure Cassandra by modifying the /etc/cassandra/cassandra.yaml file

```
# content from /etc/cassandra/cassandra.yaml

cluster_name: 'thp'
listen_address: 'xx.xx.xx.xx' # address for nodes
rpc_address: 'xx.xx.xx.xx' # address for clients
seed_provider:
    - class_name: org.apache.cassandra.locator.SimpleSeedProvider
      parameters:
          # Ex: "<ip1>,<ip2>,<ip3>"
          - seeds: 'xx.xx.xx.xx' # self for the first node
data_file_directories:
  - '/var/lib/cassandra/data'
commitlog_directory: '/var/lib/cassandra/commitlog'
saved_caches_directory: '/var/lib/cassandra/saved_caches'
hints_directory:
  - '/var/lib/cassandra/hints'
```

Step(7); Save the config file and Restart the service : service cassandra restart

Part (2) This part contains instructions to install TheHive and then configure it.

Step (1) Adding TheHive Project PGP Public Key: curl https://raw.githubusercontent.com/TheHive-Project/TheHive/master/PGP-PUBLIC-KEY | sudo apt-key add –

Step(2) : sudo apt-get update

Step(3): sudo apt-get install thehive4

“HIVE” installation and configuration guide

Step(4):To use Cassandra database, TheHive configuration file (/etc/thehive/application.conf) has to be edited and updated with following lines:

Sudo vi /etc/thehive/application.conf

```
  ## Cassandra configuration
  # More information at https://docs.janusgraph.org/basics/configuration-reference/#stora
   backend: cql
   hostname: ["172.31.95.183"]
  # Cassandra authentication (if configured)
  cql {
    cluster-name: thp
    keyspace: thehive
  }
}
index.search {
  backend: lucene
  directory: /opt/thp/thehive/index
  # If TheHive is in cluster ElasticSearch must be used:
  // backend: elasticsearch
  // hostname: ["ip1", "ip2"]
  // index-name: thehive
}

## For test only !
# Comment the two lines below before enable Cassandra database
storage.backend: berkeleyje
storage.directory: /opt/thp/thehive/database
// berkeleyje.freeDisk: 200 # disk usage threshold


 Attachment storage configuration
orage {
## Local filesystem
 provider: localfs
 localfs.location:  /opt/thp/thehive/index
```

Step(5) Save configuration file and run the service:

service thehive start

Note: service may take some time to start. Once it is started, you may launch your browser and connect to http://YOUR_SERVER_ADDRESS:9000/.

Additional Commands : Monitoring TheHive Logs and Checking Network Status

1) **Monitoring TheHive Application Logs in Real-Time:**
   sudo tail -f /var/log/thehive/application.log

2) **Installing Network Tools on Your System: sudo apt install net-tools**

3) **Displaying Listening Ports and Associated Processes: sudo netstat –ltpnd**

Integration of Tools

Integrate TheHive with Cortex

Create Cortex Organization API User:

- Login to Cortex web UI as a specific organization administrative user and create an organization API user.
- Under **Organization**, click **Add user**
- Enter the login username, full name and the roles (read and analyze only).
- Click **Save user** to create the user.
- Next, click **Create API Key** against the user to generate the key.
- Once the key is created, click **Reveal** to show the key and copy it.

Integrate TheHive with Cortex :

- Next, open TheHive configuration and update Cortex connection details;
  sudo vim /etc/thehive/application.conf

- Update the configurations below as your setup.

- Cortex-Analyzer Configuration has to be change

Integrate TheHive with Cortex

```
## CORTEX configuration
cortex {
  servers: [
    {
      name = "Cortex-cybertalents"
      url: "http://Cortex_VM_IP:9001 # Replace with your VM Public VM
      auth {
        type = "bearer"
        key = "Paste your newly created API key here"
       }
     wsConfig {}
    }
  ]
}
```

- **Save the file and exit.**
- Restart TheHive; **sudo systemctl restart thehive**
- Login to TheHive web interface and confirm Cortex integration. (Go to About> You will see Cortex is OK)

Integrate TheHive and MISP

Generate MISP Auth key:

- Login to MISP web interface as administrative user and navigate to **Administration > List Auth Keys > Add authentication Key**;
- Select the User to create an API key for, comment, defined IPs allowed to query MISP using the API key, set expiration date or leave blank to set to not expire, choose whether to set the key as read only key.
- Click Submit to create MISP auth key;
- Next, copy and save the key somewhere you would be able to retrieve when needed. Once you click **take me back now**, you wont be able to see other parts of the key.
- Next, SSH to the EC2 where TheHive is running and adjust the configuration file here

  sudo vim /etc/thehive/application.conf

Integrate TheHive and MISP

```
misp {
 interval: 1m
 servers: [
 name: "MISP-Cybertalent"
 url: "http://MISP_VM_IP/"
 auth {
 type: "key"
 key: "PASTE YOUR NEWLY CREATED KEY Here"
 }
 wsConfig
 wsConfig.ssl.loose.acceptAnyCertificate: true # This line should add to bypass the cert check
 ]
}
```

- Restart TheHive service: **sudo systemctl restart thehive**
- Next, login to TheHive web UI and confirm the integration with MISP.

Integrate ELK and THEHIVE

Configure ELK Stack to Sent Alerts to TheHive

create a webhook destination in ELK

Please use the following path in ELK to create Webhook :

Management > Stack Management > Alerts & Insights > Rules & Connectors Create Connector > Webhook

- Connector Name: <<Please Write Name of your Connector>>
- POST URL: http://TheHiveIP:9000/api/case

Add HTTP Header :

- Content-Type - application/json
- Authorization - Bearer <<API-KEY>>

To generate an authorization key, follow these steps:

1.Access TheHive web application.
2.Log in as an admin.
3.Create a new user with the Org-Admin role.
4.Generate an API key for that user.
Once completed, please test the connector with the details provided below.

Integrate ELK and THEHIVE

Sample JSON alert that can be used to test TheHive and Elasticsearch integration

```
{
 "title": "Suspicious Activity Detected",
 "description": "Anomalous login detected from an unusual location.",
 "severity": 3,
 "type": "alert",
 "source": "SIEM",
 "date": 1672531200000,  // UNIX timestamp for the alert date
 "tags": ["login", "anomaly", "SIEM"],
 "observables": [
   {
    "dataType": "ip",
    "data": "192.168.1.100",
    "message": "Suspicious IP address"
   },
   {
    "dataType": "domain",
    "data": "malicious.example.com",
    "message": "Known malicious domain"
   }
 ],
 "artifacts": [
   {
    "dataType": "file",
    "data": "d41d8cd98f00b204e9800998ecf8427e",
    "message": "File hash detected in malware scan"
   }
 ],
 "sourceRef": "SIEM-2023-0001",
 "caseTemplate": "Default Case Template",
 "createdBy": "SIEM System"
}
```