SOAR Platform Deployment and Integration Guide

– Description :

- This document illustrates the deployment of an open-source **Security Orchestration, Automation, and Response (SOAR)** framework using **Wazuh**, **TheHive**, and **Cortex**.

- **Purpose of each Component in SOAR –**

  > .**Suricata (Network IDS/IPS) : -** Detects malicious or suspicious network traffic (e.g., port scans, DDoS, brute-force, exploits) by inspecting packets using rule-based signatures.

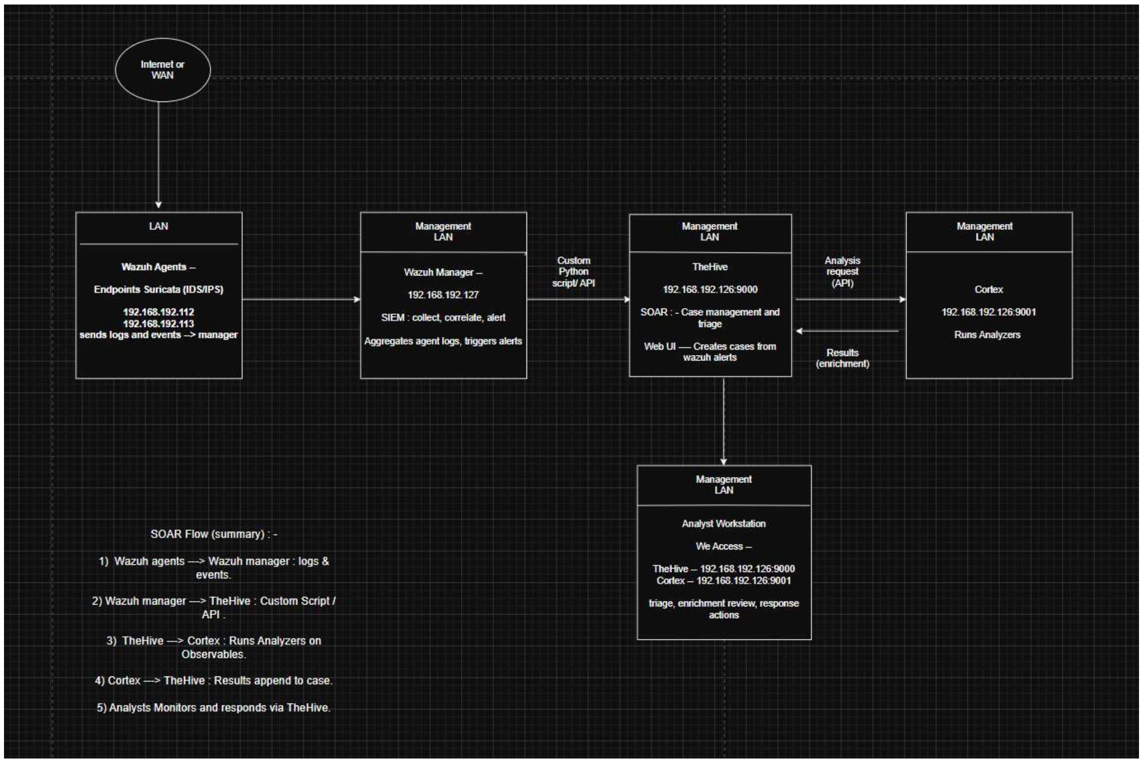
  > Wazuh (SIEM?Log Collector) : - Collects logs and security events from endpoints, servers, and network devices.

  > TheHive (SOAR/Incident Management Platform) : - Automatically creates and assigns cases when alerts are received from Wazuh.

  > Cortex (Analyzers and Responders Engine) : - Performs **automated analysis** and enrichment of observables (IPs, hashes, URLs, files) inside TheHive.

- **Suricata detects a threat → Wazuh collects and forwards the alert → TheHive receives and creates a case → Cortex analyzes and enriches the case → TheHive updates and automates the response.**
- The integration establishes an end-to-end workflow that automates the detection, case management, and enrichment of security incidents within a Security Operations Center (SOC).

– Network Architecture :


Internet or WAN
LAN
Wazuh Agents –
Endpoints Suricata (IDS/IPS)
192.168.192.112
192.168.192.113
sends logs and events –> manager
Management LAN
Wazuh Manager –
192.168.192.127
SIEM : collect, correlate, alert
Aggregates agent logs, triggers alerts
Custom Python script/ API
Management LAN
TheHive
192.168.192.126:9000
SOAR : - Case management and triage
Web UI — Creates cases from wazuh alerts
Analysis request (API)
Results (enrichment)
Management LAN
Cortex
192.168.192.126:9001
Runs Analyzers
Management LAN
Analyst Workstation
We Access –
TheHive – 192.168.192.126:9000
Cortex – 192.168.192.126:9001
triage, enrichment review, response actions

SOAR Flow (summary) : -

1) Wazuh agents —> Wazuh manager : logs & events.

2) Wazuh manager —> TheHive : Custom Script / API .

3) TheHive —> Cortex : Runs Analyzers on Observables.

4) Cortex —> TheHive : Results append to case.

5) Analysts Monitors and responds via TheHive.

--> Overall Flow of Alert to Case in the SOAR Setup :

1. **Wazuh Agent (Detection Layer) –**
   - The Wazuh Agent continuously monitors endpoints for suspicious activities (e.g., ICMP flood, SSH brute force).
   - When a security event matches a detection rule, the agent forwards the event to the **Wazuh Manager**.
2. **Wazuh Manager (Alert Generation & Integration) –**
   - The Wazuh Manager analyzes the logs and triggers a **rule-based alert**.
   - The alert (in JSON format) is then passed to the **custom integration script** configured under `/var/ossec/integrations/`.
3. **Custom Python Integration Script (Automation Bridge) –**

- The script receives the alert from Wazuh and uses **TheHive API** to automatically create a **new case** in TheHive.

- The case includes alert details like source IP, rule ID, description, and severity level.

4. **TheHive (Case Creation & Management) –**

   - TheHive receives the alert via API and creates a structured **incident case**.

   - Each alert becomes a ticket containing observables (e.g., IP addresses, hashes, domains).

   - Analysts review, assign, and begin investigation directly within TheHive.

5. **Cortex (Enrichment & Analysis) –**

   - When analysts run analyzers (like VirusTotal, AbuseIPDB, or URLhaus) on the observables, TheHive sends those observables to **Cortex**.

   - Cortex queries external **Threat Intelligence APIs** and returns reputation or enrichment data.

6. **TheHive (Receiving Enrichment Results) –**

   - TheHive automatically attaches Cortex's results to the observables within the same case.

   - Analysts can now see whether an IP or hash is malicious, previously seen, or associated with known attacks.

7. **Analyst Workstation (Investigation & Response) –**

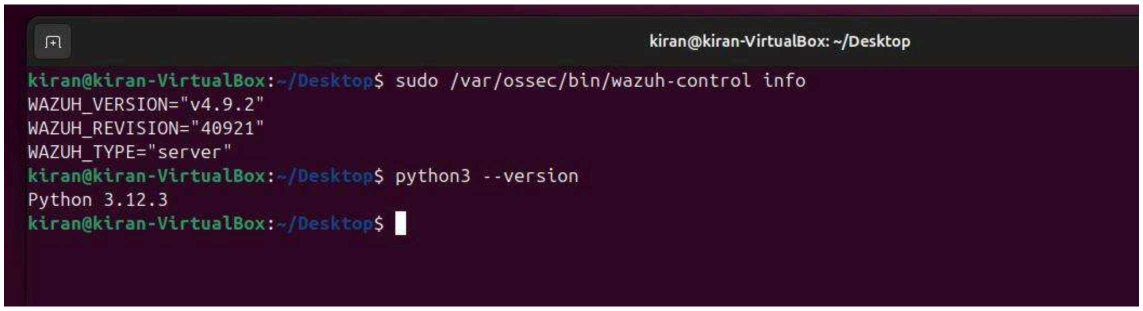
   - The analyst uses the enriched information to triage, correlate events, and plan response actions.

   - The case can then be marked as resolved, escalated, or linked to other related incidents.

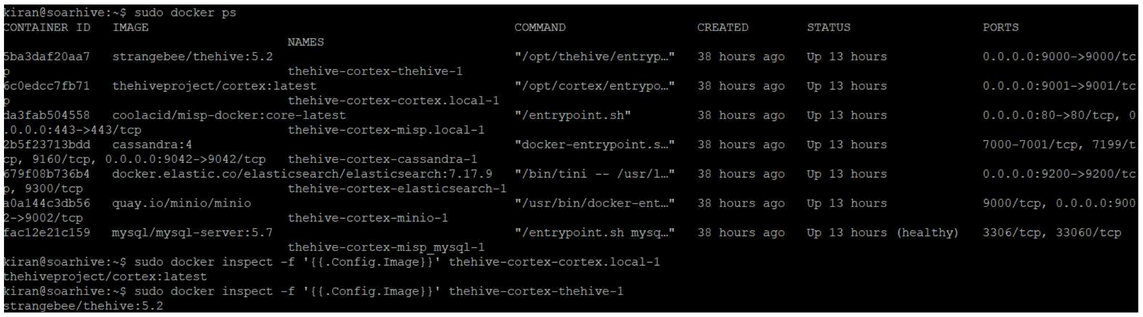
– Environment Setup :

- This section describes the environment used for deploying and testing the Wazuh–TheHive integration.Both tools were installed on same machine.

- The Wazuh Manager is responsible for event collection, rule-based alert generation, and integration execution, while TheHive serves as the incident response platform that receives alerts as cases.
- Before configuring the integration, it is essential to verify the versions of the operating system, Wazuh, TheHive, and Python to ensure compatibility.
- This also helps future maintenance and replication of the same environment.



```
kiran@kiran-VirtualBox:~/Desktop$ sudo /var/ossec/bin/wazuh-control info
WAZUH_VERSION="v4.9.2"
WAZUH_REVISION="40921"
WAZUH_TYPE="server"
kiran@kiran-VirtualBox:~/Desktop$ python3 --version
Python 3.12.3
kiran@kiran-VirtualBox:~/Desktop$
```



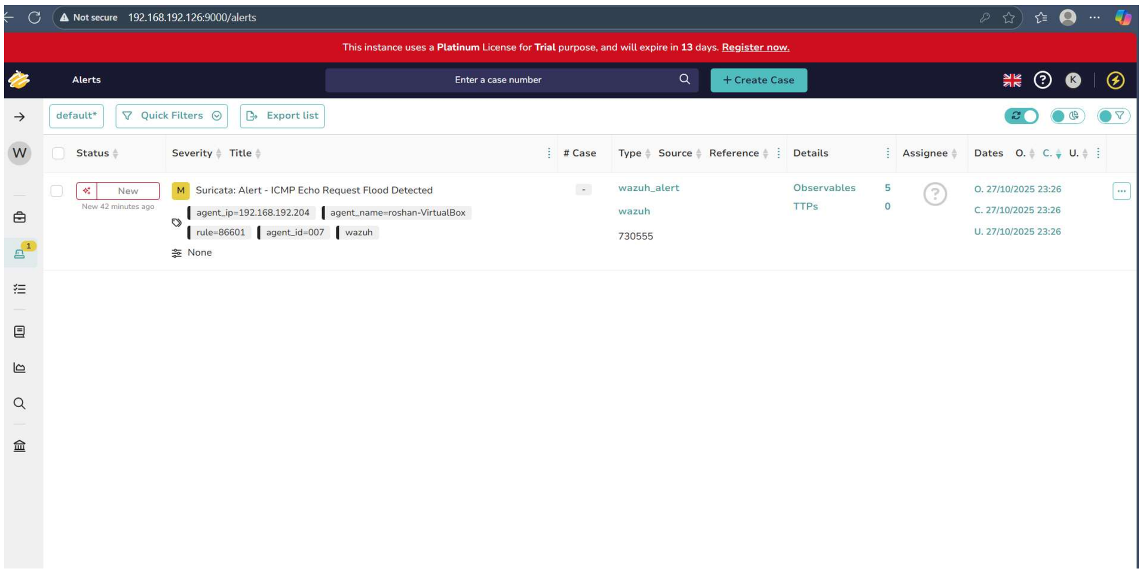
```
kiran@soarhive:~$ sudo docker ps
CONTAINER ID   IMAGE                                                  COMMAND                  CREATED        STATUS                  PORTS
                                       NAMES
5ba3daf20aa7   strangebee/thehive:5.2                                 "/opt/thehive/entryp…"   38 hours ago   Up 13 hours             0.0.0.0:9000->9000/tc
p                                      thehive-cortex-thehive-1
6c0edcc7fb71   thehiveproject/cortex:latest                           "/opt/cortex/entrypo…"   38 hours ago   Up 13 hours             0.0.0.0:9001->9001/tc
p                                      thehive-cortex-cortex.local-1
da3fab504558   coolacid/misp-docker:core-latest                       "/entrypoint.sh"         38 hours ago   Up 13 hours             0.0.0.0:80->80/tcp, 0
.0.0.0:443->443/tcp                    thehive-cortex-misp.local-1
2b5f23713bdd   cassandra:4                                            "docker-entrypoint.s…"   38 hours ago   Up 13 hours             7000-7001/tcp, 7199/t
cp, 9160/tcp, 0.0.0.0:9042->9042/tcp   thehive-cortex-cassandra-1
679f08b736b4   docker.elastic.co/elasticsearch/elasticsearch:7.17.9   "/bin/tini -- /usr/l…"   38 hours ago   Up 13 hours             0.0.0.0:9200->9200/tc
p, 9300/tcp                            thehive-cortex-elasticsearch-1
a0a144c3db56   quay.io/minio/minio                                    "/usr/bin/docker-ent…"   38 hours ago   Up 13 hours             9000/tcp, 0.0.0.0:900
2->9002/tcp                            thehive-cortex-minio-1
fac12e21c159   mysql/mysql-server:5.7                                 "/entrypoint.sh mysq…"   38 hours ago   Up 13 hours (healthy)   3306/tcp, 33060/tcp
                                       thehive-cortex-misp_mysql-1
kiran@soarhive:~$ sudo docker inspect -f '{{.Config.Image}}' thehive-cortex-cortex.local-1
thehiveproject/cortex:latest
kiran@soarhive:~$ sudo docker inspect -f '{{.Config.Image}}' thehive-cortex-thehive-1
strangebee/thehive:5.2
```

Figure 1 — Environment verification.

- *This screenshot confirms the versions of Ubuntu, Wazuh Manager, TheHive, and Python used in the deployment.*
- *This ensures compatibility between components before configuring integration.*

– TheHive Installation & Verification :

- TheHive was installed and configured as the Security Operations and Incident Response platform.
- After installation, the service was verified to ensure it runs on the default port 9000.
- TheHive provides an intuitive web interface for analysts to manage security incidents as cases, link observables, and correlate events with MITRE ATT&CK techniques.
- Once logged into the web interface, the administrator can generate an **API key** (under Profile → API keys).
- This API key is crucial because it allows the Wazuh Manager integration script to authenticate securely with TheHive API for automated case creation.



Not secure 192.168.192.126:9000/alerts

This instance uses a **Platinum** License for **Trial** purpose, and will expire in **13** days. Register now.

Alerts

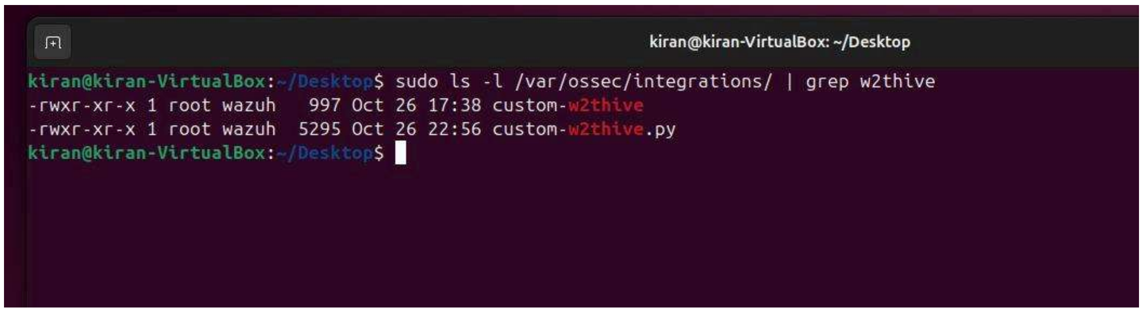
Status	Severity	Title	# Case	Type	Source	Reference	Details	Assignee	Dates
New New 42 minutes ago	M	Suricata: Alert - ICMP Echo Request Flood Detected agent_ip=192.168.192.204 agent_name=roshan-VirtualBox rule=86601 agent_id=007 wazuh None	-	wazuh_alert	wazuh	730555	Observables 5 TTPs 0		O. 27/10/2025 23:26 C. 27/10/2025 23:26 U. 27/10/2025 23:26

Figure 2 — TheHive successfully deployed.

- *The web interface indicates that TheHive is running correctly on port 9000 and ready to receive API requests from Wazuh.*
- *The administrator user can generate an **API key** under Profile → API keys, which will later be used in the Wazuh integration script. After the organization is created logon with that organization admin credentials.*

– Wazuh TheHive Integration :

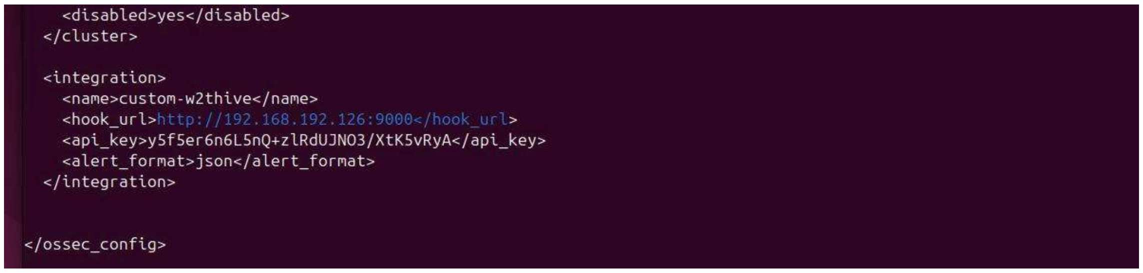
- In this phase, Wazuh is configured to automatically forward alerts to TheHive using a **custom integration script**.
- The integration is placed under /var/ossec/integrations/ where Wazuh looks for executable scripts matching names defined in the configuration file.
- The integration consists of two parts :
  1) A **Bash wrapper** (custom-w2thive) that Wazuh executes directly.
  2) A **Python script** (custom-w2thive.py) that uses thehive4py to send alerts to TheHive's API endpoint.
- And at last, Proper permissions (chmod 755) and ownership (root:wazuh or root:ossec) are assigned to ensure Wazuh can execute the files without permission errors.



```
kiran@kiran-VirtualBox:~/Desktop$ sudo ls -l /var/ossec/integrations/ | grep w2thive
-rwxr-xr-x 1 root wazuh   997 Oct 26 17:38 custom-w2thive
-rwxr-xr-x 1 root wazuh  5295 Oct 26 22:56 custom-w2thive.py
kiran@kiran-VirtualBox:~/Desktop$
```

Figure 3 — Custom integration scripts in Wazuh.

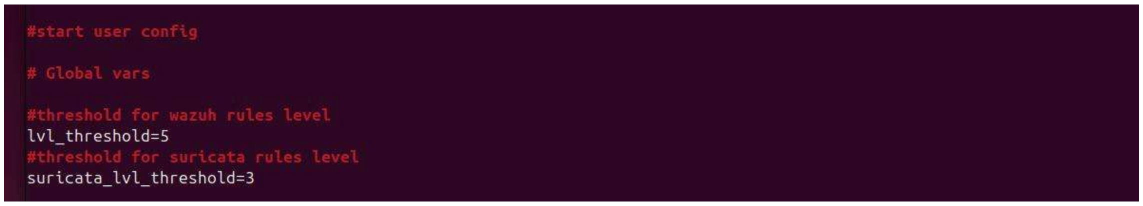
- *The files custom-w2thive (Bash wrapper) and custom-w2thive.py (Python script) are present and executable under /var/ossec/integrations/. These handle the communication between Wazuh and TheHive using the thehive4py API library.*



```
    <disabled>yes</disabled>
  </cluster>

  <integration>
    <name>custom-w2thive</name>
    <hook_url>http://192.168.192.126:9000</hook_url>
    <api_key>y5f5er6n6L5nQ+zlRdUJNO3/XtK5vRyA</api_key>
    <alert_format>json</alert_format>
  </integration>


</ossec_config>
```

Figure 4 — Wazuh Manager integration configuration.

- *The integration passes the alert JSON to the custom script for forwarding to TheHive.*



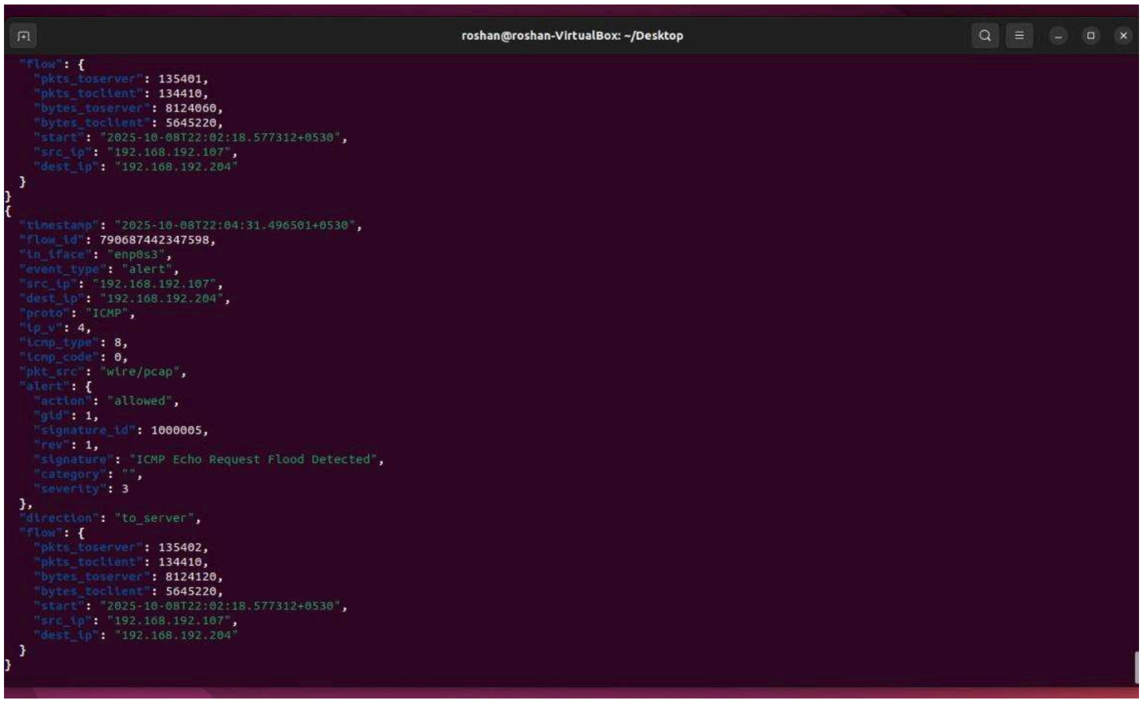
```
#start user config

# Global vars

#threshold for wazuh rules level
lvl_threshold=5
#threshold for suricata rules level
suricata_lvl_threshold=3
```

Figure 5 —thresold level of Wazuh

- *This snippet instructs Wazuh to trigger the custom-w2thive integration whenever an alert of level ≥ 3 is generated*

– Alert Generation and Forwarding :

- After the integration setup, the next step is to verify the flow of alerts from Wazuh to TheHive.
- When a detection rule is triggered — for example, during an ICMP flood or SSH brute-force test — Wazuh generates an alert in JSON format and passes it to the configured integration.
- The Bash wrapper captures this alert and calls the Python script, which then sends it to TheHive via REST API.
- The alert JSON contains all important details such as rule.id, rule.description, srcip, and agent.name, which are used to create a meaningful case title and description in TheHive



```
  "flow": {
    "pkts_toserver": 135401,
    "pkts_toclient": 134410,
    "bytes_toserver": 8124060,
    "bytes_toclient": 5645220,
    "start": "2025-10-08T22:02:18.577312+0530",
    "src_ip": "192.168.192.107",
    "dest_ip": "192.168.192.204"
  }
}
{
  "timestamp": "2025-10-08T22:04:31.496501+0530",
  "flow_id": 790687442347598,
  "in_iface": "enp0s3",
  "event_type": "alert",
  "src_ip": "192.168.192.107",
  "dest_ip": "192.168.192.204",
  "proto": "ICMP",
  "ip_v": 4,
  "icmp_type": 8,
  "icmp_code": 0,
  "pkt_src": "wire/pcap",
  "alert": {
    "action": "allowed",
    "gid": 1,
    "signature_id": 1000005,
    "rev": 1,
    "signature": "ICMP Echo Request Flood Detected",
    "category": "",
    "severity": 3
  },
  "direction": "to_server",
  "flow": {
    "pkts_toserver": 135402,
    "pkts_toclient": 134410,
    "bytes_toserver": 8124120,
    "bytes_toclient": 5645220,
    "start": "2025-10-08T22:02:18.577312+0530",
    "src_ip": "192.168.192.107",
    "dest_ip": "192.168.192.204"
  }
}
```

Figure 6 — Example Wazuh alert triggered by test attack.

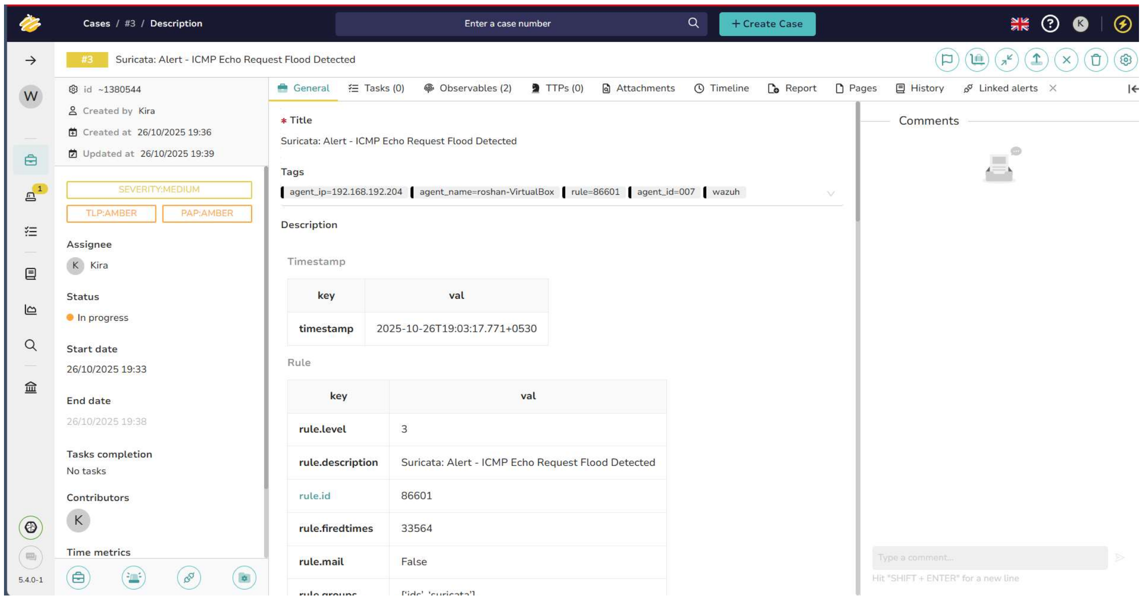
- *This shows the ICMP flood detection alert generated by Wazuh. The alert JSON includes key fields such as rule.id, rule.description, agent.name, and srcip, which are forwarded to TheHive through the integration script.*

– TheHive Automatic Case Creation Result :

- When TheHive receives the alert through its API, it automatically creates a new case based on the data provided by the integration script.
- Each case in TheHive includes :

> A **title** (derived from the Wazuh alert description),
> A **description** (the complete alert JSON),
> **Tags** for identification (e.g., wazuh, icmp_flood, ssh_bruteforce),
> And optionally, **observables** extracted from the alert (such as source IP or file hash).

- This automated case creation streamlines the workflow — security analysts can directly open TheHive, review alert details, add comments, and take response actions without manually checking Wazuh logs.



#3 Suricata: Alert - ICMP Echo Request Flood Detected

id ~1380544
Created by Kira
Created at 26/10/2025 19:36
Updated at 26/10/2025 19:39

SEVERITY:MEDIUM
TLP:AMBER PAP:AMBER

Assignee
Kira

Status
In progress

Start date
26/10/2025 19:33

End date
26/10/2025 19:38

Tasks completion
No tasks

Contributors
K

Time metrics

Title
Suricata: Alert - ICMP Echo Request Flood Detected

Tags
agent_ip=192.168.192.204 agent_name=roshan-VirtualBox rule=86601 agent_id=007 wazuh

Description

Timestamp

key	val
timestamp	2025-10-26T19:03:17.771+0530

Rule

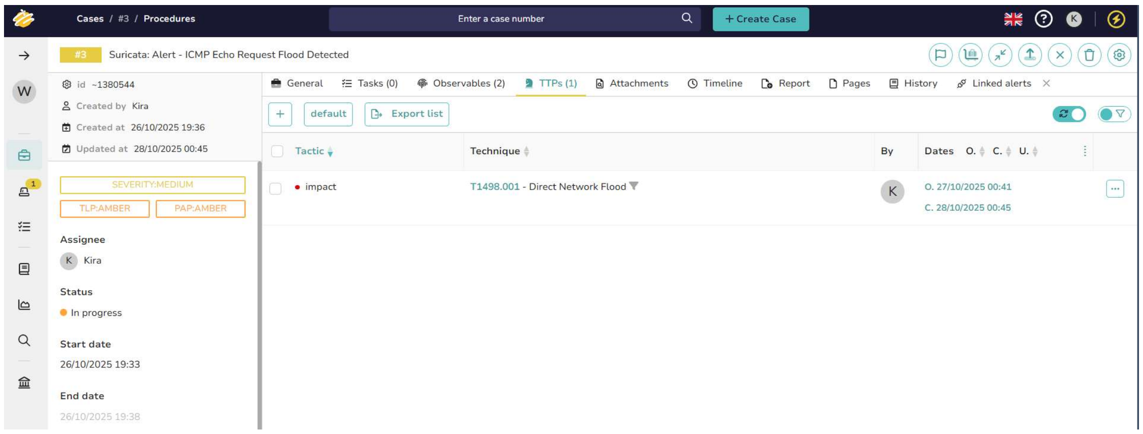
key	val
rule.level	3
rule.description	Suricata: Alert - ICMP Echo Request Flood Detected
rule.id	86601
rule.firedtimes	33564
rule.mail	False
rule.groups	['ids', 'suricata']

Figure 6 — Case automatically created in TheHive.

- *This confirms successful end-to-end integration: Wazuh detected the ICMP flood attack, forwarded the alert, and TheHive created a case with the full alert JSON in its description.*
- *Analysts can now triage, assign, and enrich the case directly within TheHive*

– MITRE ATT&CK Mapping :

- TheHive allows analysts to enrich and categorize cases according to the **MITRE ATT&CK framework**.
- By linking alerts or cases to specific ATT&CK techniques (e.g., T1498 – *Network Denial of Service*), SOC teams can quickly assess the adversary tactics involved and prioritize response efforts.
- During the integration process, MITRE technique identifiers can be added either automatically by the Python script (via a mapping file of rule IDs to techniques) or manually by analysts within TheHive UI.



#3 Suricata: Alert - ICMP Echo Request Flood Detected

id ~1380544
Created by Kira
Created at 26/10/2025 19:36
Updated at 28/10/2025 00:45

SEVERITY:MEDIUM
TLP:AMBER PAP:AMBER

Assignee
Kira

Status
In progress

Start date
26/10/2025 19:33

End date
26/10/2025 19:38

Tactic	Technique	By	Dates
impact	T1498.001 - Direct Network Flood	K	O. 27/10/2025 00:41 C. 28/10/2025 00:45

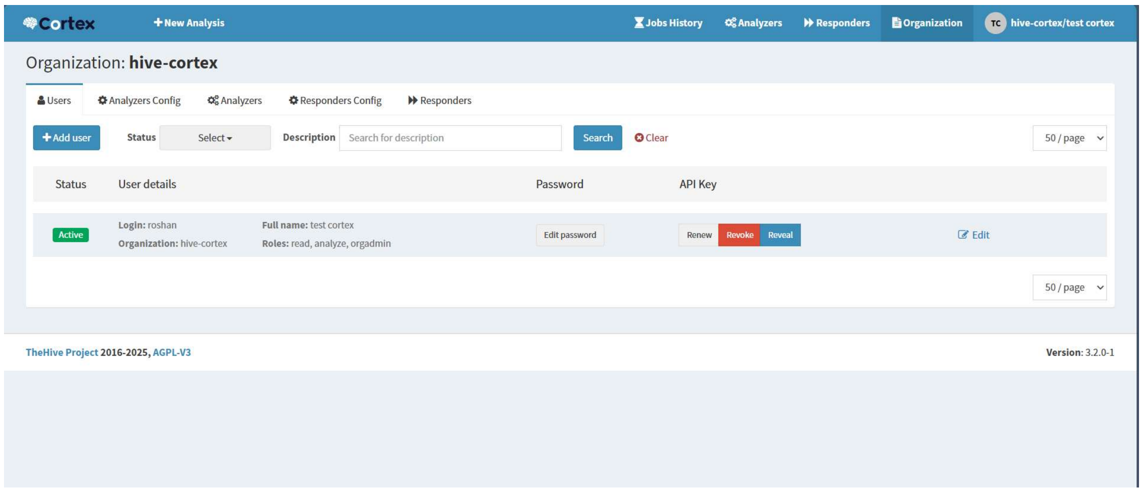
Figure 7 – Mapping the observable that extracted from the wazuh with the techniques in the MITRE ATT&CK Framework.

– Cortex Integration and Analyzer Configuration :

- Cortex is an open-source analysis and response engine designed to work alongside TheHive. It allows analysts to run hundreds of analyzers and responders on observables, such as IP addresses, domain names, file hashes, or URLs.
- When integrated with TheHive, Cortex automates **threat enrichment** and **incident validation**, reducing the manual effort needed by SOC analysts during investigations.
- In this setup, TheHive sends observables extracted from cases to Cortex, which performs external lookups and returns enriched intelligence results. This workflow helps analysts rapidly determine if an observable is malicious and provides context for decision-making.

– Cortex Web Interface Access :

- Cortex is hosted on the same server as TheHive (`192.168.192.126`) and operates on port **9001**.
- Accessing the Cortex web interface confirms that the service is running properly and available for integration.
- From here, administrators can manage analyzers, responders, API keys, and organization settings.



Organization: **hive-cortex**

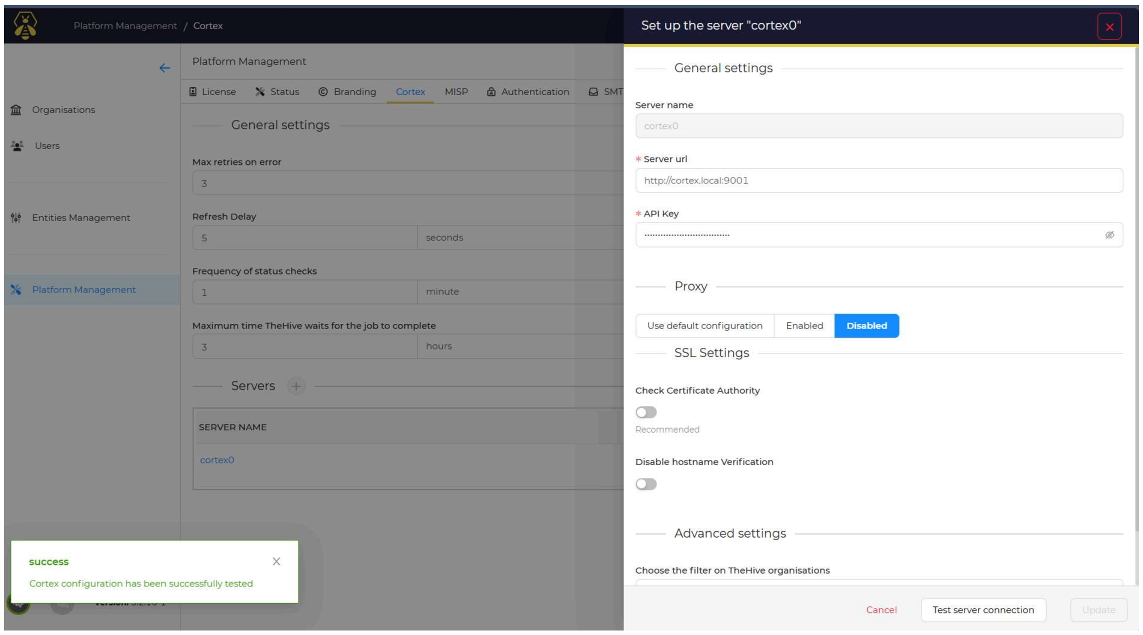
Status	User details		Password	API Key
Active	**Login:** roshan **Organization:** hive-cortex	**Full name:** test cortex **Roles:** read, analyze, orgadmin	Edit password	Renew Revoke Reveal

Figure 8 — Cortex web interface running on port 9001.

- This verifies that the Cortex service is successfully deployed and accessible on the network. It forms the core of automated analysis and response capabilities in the SOAR environment.

– TheHive–Cortex Connection Verification :

- Once Cortex is running, TheHive must be configured to communicate with it through the API endpoint.
- The integration is achieved by adding the Cortex instance details under TheHive’s administration settings and validating the connection.
- A successful connection ensures that TheHive can trigger Cortex analyzers when analysts request enrichment on observables.



Platform Management

General settings

Max retries on error: 3

Refresh Delay: 5 seconds

Frequency of status checks: 1 minute

Maximum time TheHive waits for the job to complete: 3 hours

Servers

SERVER NAME
cortex0

success

Cortex configuration has been successfully tested

Set up the server "cortex0"

General settings

Server name: cortex0

* Server url: http://cortex.local:9001

* API Key: ································

Proxy

Use default configuration | Enabled | [✓ Disabled]

SSL Settings

Check Certificate Authority

Recommended

Disable hostname Verification

Advanced settings

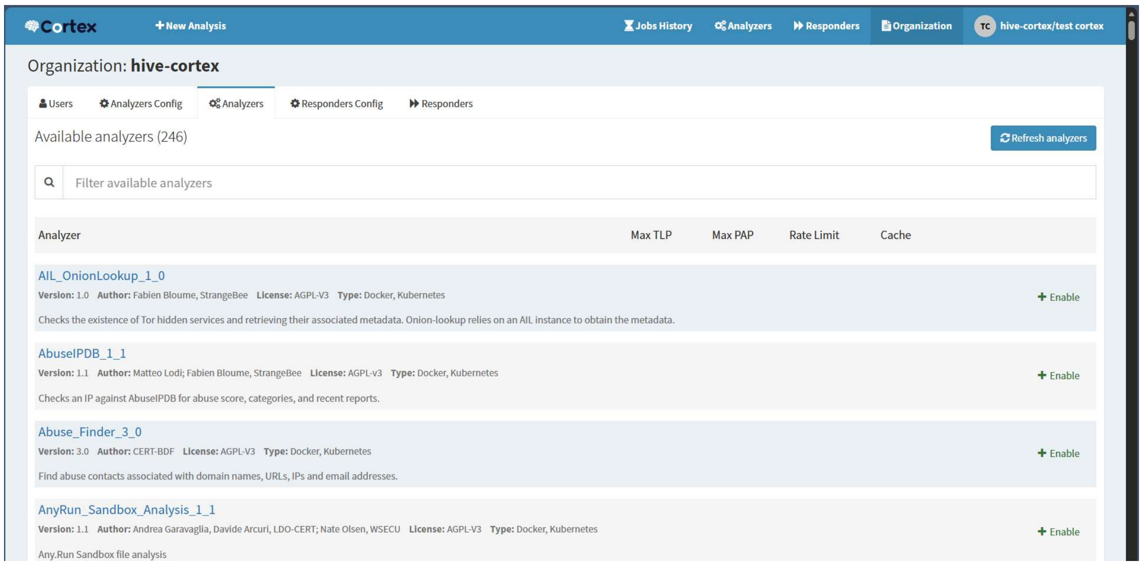
Choose the filter on TheHive organisations

Figure 9 — TheHive–Cortex integration test.

- TheHive connects to the Cortex instance at 192.168.192.126:9001 using the configured API key.

– Analyzer Configuration :

- Analyzers are modular plugins in Cortex that interact with third-party threat intelligence services. Each analyzer requires an API key or credentials, which are configured in its `config.json` file under `/opt/cortex/analyzers/`.
- For example, the **VirusTotal_GetReport_3_1** analyzer requires a VirusTotal API key. Once configured, Cortex can query these external services and send back detailed results to TheHive.

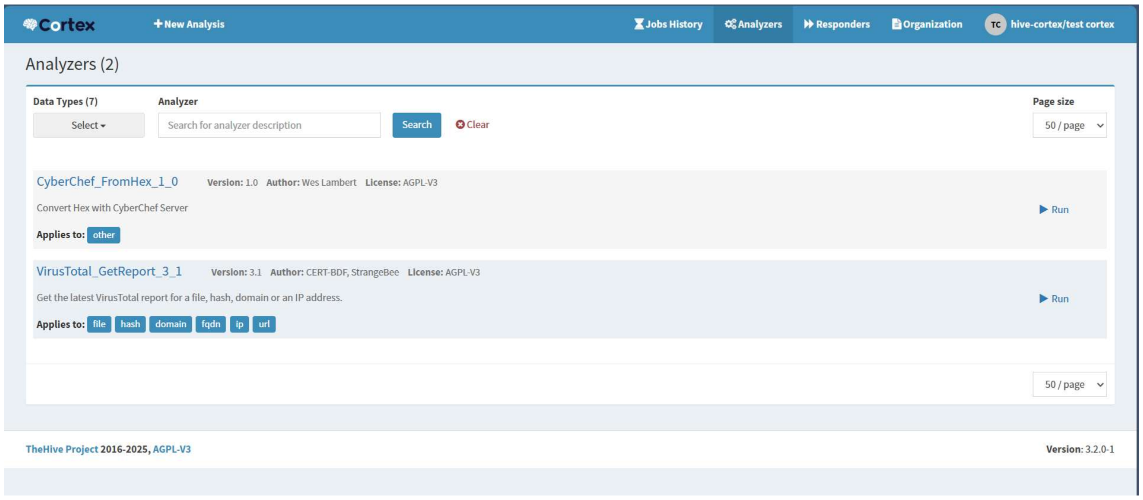


Organization: **hive-cortex**

Available analyzers (246)

Analyzer	Max TLP	Max PAP	Rate Limit	Cache	
AIL_OnionLookup_1_0 **Version:** 1.0 **Author:** Fabien Bloume, StrangeBee **License:** AGPL-V3 **Type:** Docker, Kubernetes Checks the existence of Tor hidden services and retrieving their associated metadata. Onion-lookup relies on an AIL instance to obtain the metadata.					Enable
AbuseIPDB_1_1 **Version:** 1.1 **Author:** Matteo Lodi; Fabien Bloume, StrangeBee **License:** AGPL-v3 **Type:** Docker, Kubernetes Checks an IP against AbuseIPDB for abuse score, categories, and recent reports.					Enable
Abuse_Finder_3_0 **Version:** 3.0 **Author:** CERT-BDF **License:** AGPL-V3 **Type:** Docker, Kubernetes Find abuse contacts associated with domain names, URLs, IPs and email addresses.					Enable
AnyRun_Sandbox_Analysis_1_1 **Version:** 1.1 **Author:** Andrea Garavaglia, Davide Arcuri, LDO-CERT; Nate Olsen, WSECU **License:** AGPL-V3 **Type:** Docker, Kubernetes Any.Run Sandbox file analysis					Enable

Figure 10 — available Analyzers list in Cortex.



Analyzers (2)

CyberChef_FromHex_1_0 **Version:** 1.0 **Author:** Wes Lambert **License:** AGPL-V3
Convert Hex with CyberChef Server
Applies to: other
Run

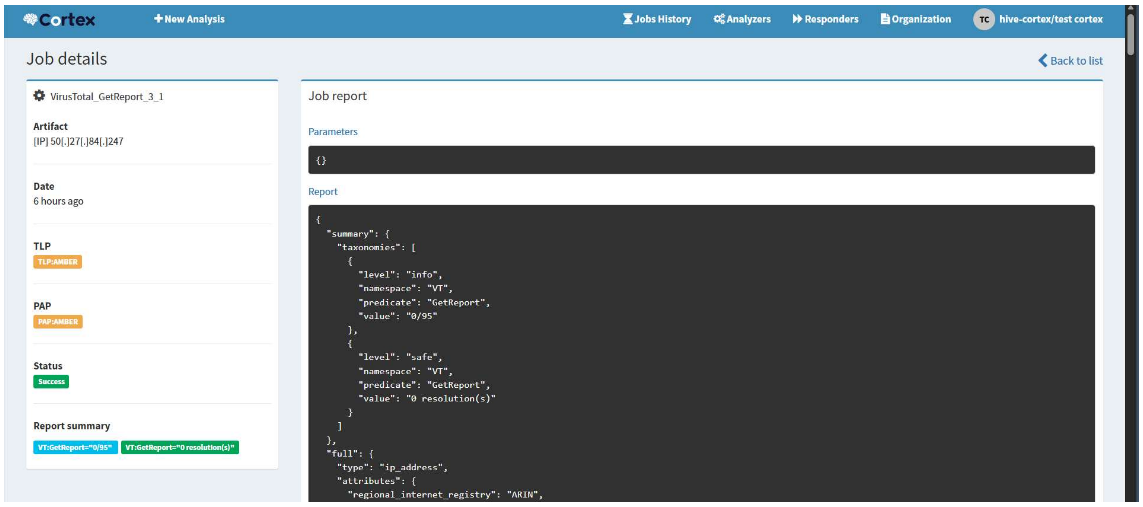
VirusTotal_GetReport_3_1 **Version:** 3.1 **Author:** CERT-BDF, StrangeBee **License:** AGPL-V3
Get the latest VirusTotal report for a file, hash, domain or an IP address.
Applies to: file hash domain fqdn ip url
Run

TheHive Project 2016-2025, AGPL-V3
Version: 3.2.0-1

Figure 11 — active Analyzers in the Organization.
The screenshot displays the active analyzers available in the Cortex platform.

– Analyzer Execution Result :

- Once analyzers are properly configured, analysts can execute them from within a case. For example, selecting an observable such as an IP address and running the **VirusTotal** analyzer triggers Cortex to fetch reputation data.
- TheHive then displays this information in the case as part of the enrichment results, giving immediate context for triage.



Job details

VirusTotal_GetReport_3_1

Artifact
[IP] 50[.]27[.]84[.]247

Date
6 hours ago

TLP
TLP:AMBER

PAP
PAP:AMBER

Status
Success

Report summary
VT:GetReport="0/95" VT:GetReport="0 resolution(s)"

Job report

Parameters

```
{}
```

Report

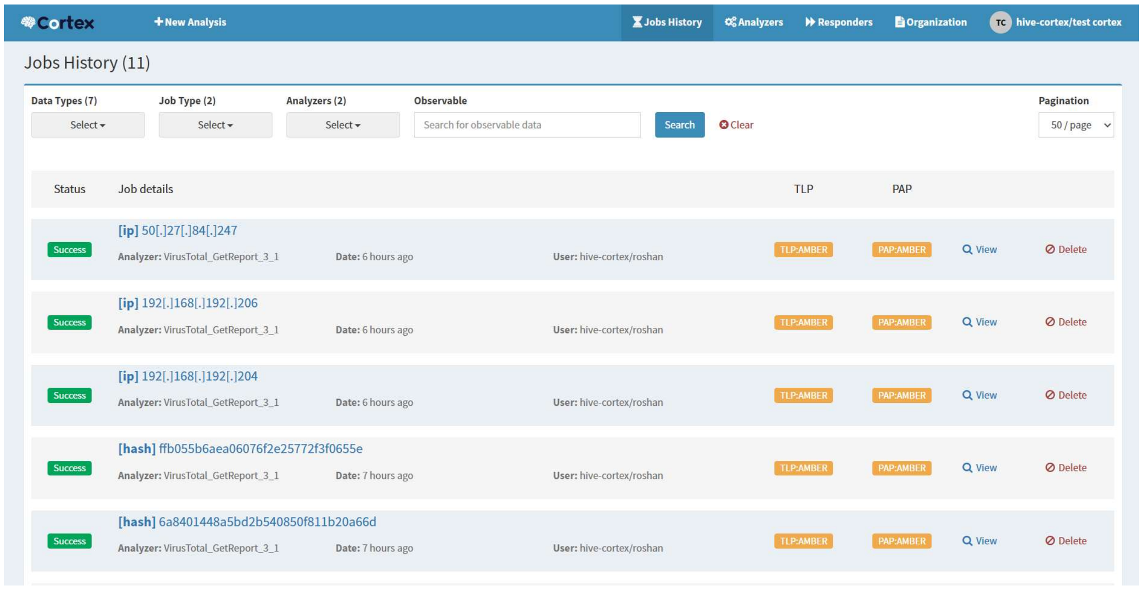
```
{
  "summary": {
    "taxonomies": [
      {
        "level": "info",
        "namespace": "VT",
        "predicate": "GetReport",
        "value": "0/95"
      },
      {
        "level": "safe",
        "namespace": "VT",
        "predicate": "GetReport",
        "value": "0 resolution(s)"
      }
    ]
  },
  "full": {
    "type": "ip_address",
    "attributes": {
      "regional_internet_registry": "ARIN",
```

Figure 12 — Cortex analyzer execution and result.

- The result shows successful enrichment of a case observable. Cortex retrieved threat data from external intelligence feeds, providing analysts with actionable insights for further investigation.

– Cortex Job history :

- Cortex maintains a record of all executed jobs, including their execution status and result data. This helps administrators monitor performance of analyzers. If a job fails, error logs in this section can be used for troubleshooting configuration or connectivity issues.



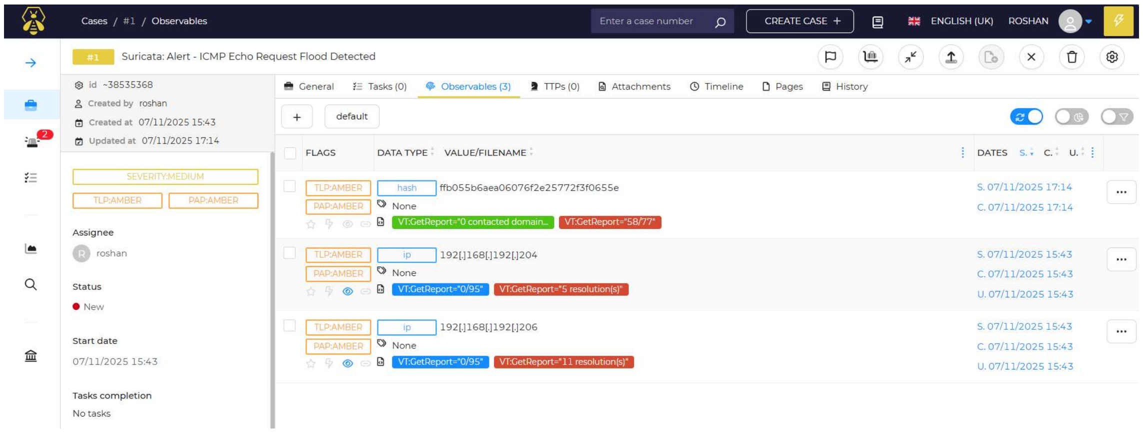
Jobs History (11)

Status	Job details			TLP	PAP		
Success	**[ip]** 50[.]27[.]84[.]247 **Analyzer:** VirusTotal_GetReport_3_1	**Date:** 6 hours ago	**User:** hive-cortex/roshan	TLP:AMBER	PAP:AMBER	View	Delete
Success	**[ip]** 192[.]168[.]192[.]206 **Analyzer:** VirusTotal_GetReport_3_1	**Date:** 6 hours ago	**User:** hive-cortex/roshan	TLP:AMBER	PAP:AMBER	View	Delete
Success	**[ip]** 192[.]168[.]192[.]204 **Analyzer:** VirusTotal_GetReport_3_1	**Date:** 6 hours ago	**User:** hive-cortex/roshan	TLP:AMBER	PAP:AMBER	View	Delete
Success	**[hash]** ffb055b6aea06076f2e25772f3f0655e **Analyzer:** VirusTotal_GetReport_3_1	**Date:** 7 hours ago	**User:** hive-cortex/roshan	TLP:AMBER	PAP:AMBER	View	Delete
Success	**[hash]** 6a8401448a5bd2b540850f811b20a66d **Analyzer:** VirusTotal_GetReport_3_1	**Date:** 7 hours ago	**User:** hive-cortex/roshan	TLP:AMBER	PAP:AMBER	View	Delete

Figure 12 — Cortex job execution details.

– Sending Analyzers results back to TheHive :

- **Cortex sends the analyzer results back to TheHive**, where they are automatically attached to the corresponding case as enrichment data. This allows analysts to review threat intelligence details (such as IP reputation, file hash analysis, or URL categorization) directly within the TheHive interface, enabling faster triage and informed decision-making.
- **TheHive → Cortex (analysis request) → Cortex → TheHive (enrichment results)**


Cases / #1 / Observables

#1 Suricata: Alert - ICMP Echo Request Flood Detected

id ~38535368
Created by roshan
Created at 07/11/2025 15:43
Updated at 07/11/2025 17:14

SEVERITY:MEDIUM
TLP:AMBER PAP:AMBER

Assignee
roshan

Status
New

Start date
07/11/2025 15:43

Tasks completion
No tasks

FLAGS	DATA TYPE	VALUE/FILENAME	DATES
TLP:AMBER PAP:AMBER	hash	ffb055b6aea06076f2e25772f3f0655e None VT:GetReport="0 contacted domain... VT:GetReport="58/77"	S. 07/11/2025 17:14 C. 07/11/2025 17:14
TLP:AMBER PAP:AMBER	ip	192[.]168[.]192[.]204 None VT:GetReport="0/95" VT:GetReport="5 resolution(s)"	S. 07/11/2025 15:43 C. 07/11/2025 15:43 U. 07/11/2025 15:43
TLP:AMBER PAP:AMBER	ip	192[.]168[.]192[.]206 None VT:GetReport="0/95" VT:GetReport="11 resolution(s)"	S. 07/11/2025 15:43 C. 07/11/2025 15:43 U. 07/11/2025 15:43

Figure 13 — Cortex sends Analyzers results back to the hive.

– Conclusion :

- The integrated SOAR environment successfully brings together **Wazuh**, **Suricata**, **TheHive**, and **Cortex** into a unified security operations ecosystem that enhances both visibility and response efficiency.
- Suricata provides real-time network intrusion detection, while Wazuh acts as the central SIEM, aggregating and correlating security events across the infrastructure.
- TheHive enables automated case management and structured incident handling, and Cortex strengthens the workflow by enriching alerts with powerful threat intelligence and analysis capabilities.
- This architecture not only automates alert triage and reduces analyst workload but also minimizes response time through seamless integration between detection, analysis, and orchestration layers.
- By aligning with modern SOC best practices, the setup establishes a scalable, open-source, and cost-effective foundation for proactive threat detection, investigation and response automation.