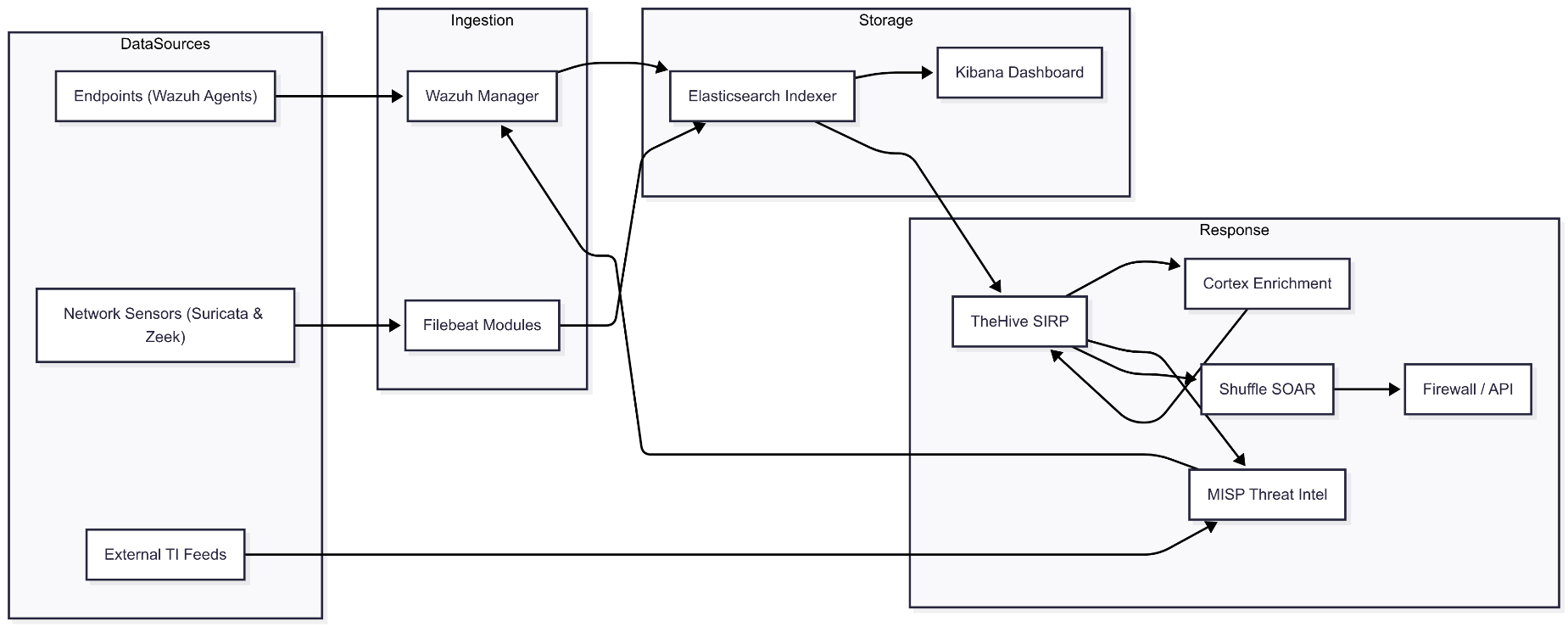
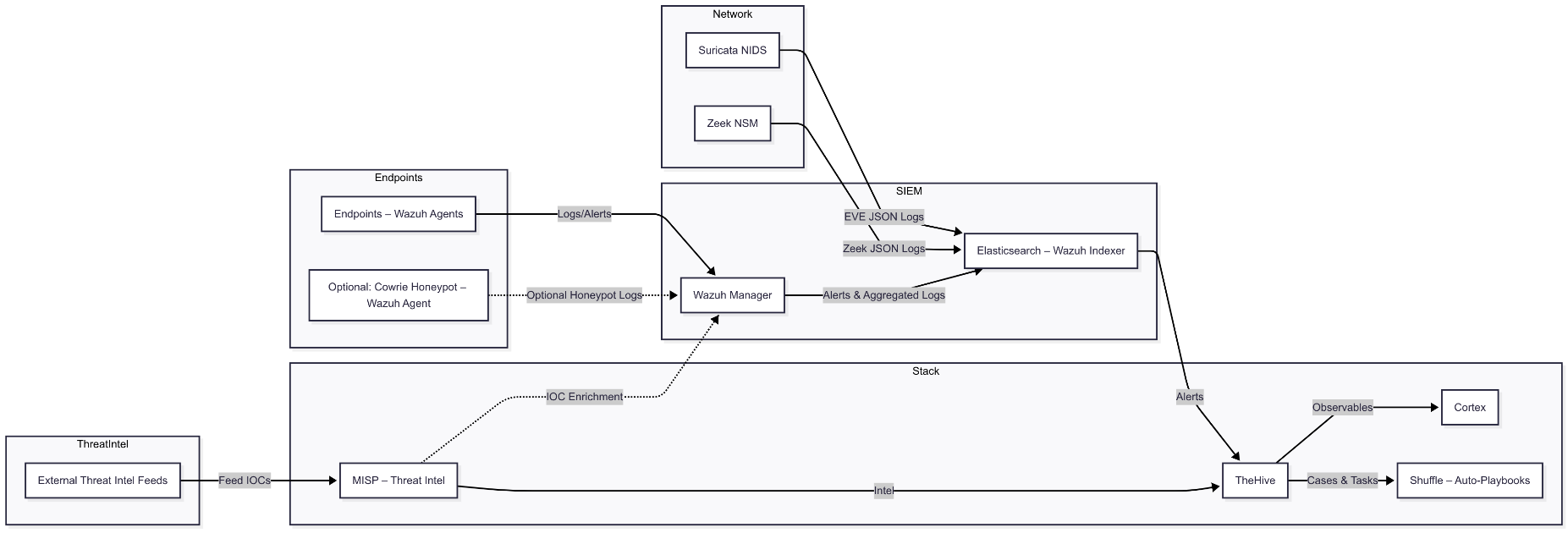
**Design of a secure Open-Source SOC framework**

Building an open-source Security Operations Centre (SOC) involves integrating multiple tools to collect logs, detect threats, manage incidents, and automate responses. A typical open-source SOC stack combines a host-based SIEM/XDR engine (like Wazuh) with network-based visibility tools (such as Suricata and Zeek), a case-management platform (TheHive and Cortex), threat intelligence sharing (e.g. MISP), and orchestration (e.g. Shuffle). The diagram below illustrates such an architecture: endpoint logs (via Wazuh agents), network traffic (via Suricata and Zeek), and honeypots (e.g. Cowrie) feed into a central log management backend (Elasticsearch), which supports search, visualization, and alert generation. Alerts are passed to TheHive for investigation, Cortex automatically analyzes observables, and MISP (along with external feeds) provides IOC correlation. Analysts view data in Kibana and collaborate on cases in TheHive; Shuffle workflows automate triage, enrichment, and response across the stack.

*Proposed Open-Source SOC flow diagram*



*Proposed Open-Source SOC Block Architectural diagram*

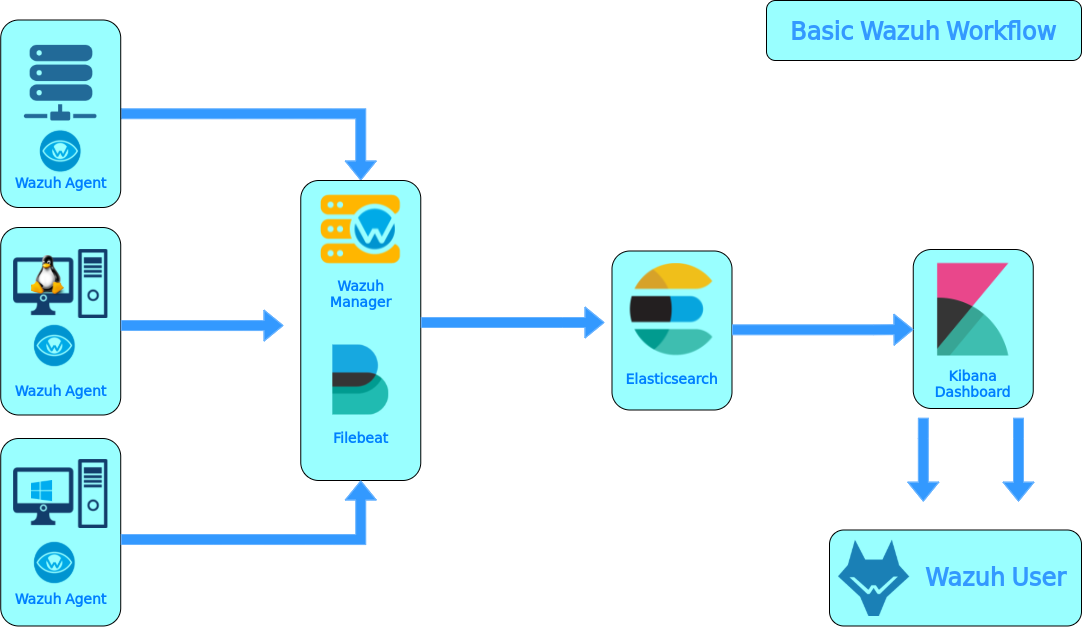
1. **Wazuh (SIEM/XDR Platform)**

**Wazuh** is an open-source **SIEM/XDR** platform for log collection and threat detection. It is a fork of the OSSEC HIDS, providing agents for endpoints and a central manager/analysis server. Wazuh agents collect logs (system logs, application logs, Windows Event logs, etc.) and perform file-integrity monitoring, vulnerability assessment, and compliance checks on each host. The Wazuh manager aggregates these logs, correlates events against detection rules, and sends alerts to an Elasticsearch index (or Graylog/Logstash) for searching and visualization.

**Key Wazuh capabilities include:**

* **Log Analysis & Correlation:** In-depth log parsing and correlation rules trigger alerts on suspicious activity (malware, rootkits, anomalies).
* **File Integrity Monitoring:** Detects unauthorized file changes to OS/config files, with immediate alerts.
* **Vulnerability & Compliance:** Integrated vulnerability scanner and policy compliance checks (e.g. PCI-DSS, GDPR) allow auditors to track issues.
* **Active Responses:** Automated remediations (scripts) can be triggered on endpoints when alerts fire.

Wazuh’s **agent-based architecture** supports Windows, Linux, macOS and cloud workloads. Its free, open nature means no license costs or vendor lock-in, with a large community providing continuous updates.

Deploying a basic wazuh **SIEM** solution in VirtualBox  


**Wazuh Manager** collects logs from **Wazuh Agents**. **Filebeat** then sends logs from **Wazuh Manager** to **Elasticsearch**. **Kibana Dashboard** visualize the logs from **Elasticsearch**. In this deployment, we'll use the following nodes.

| **Deployment** | **OS** | **RAM** |
| --- | --- | --- |
| Wazuh Manager & Filebeat | Ubuntu Server 20.04 | 1 GB |
| Elasticsearch | Ubuntu Server 20.04 | 1GB |
| Kibana Dashboard & Wazuh Agent-1 | Kali Linux | 2 GB |
| Wazuh Agent-2 | Ubuntu Server 20.04 | 1 GB |
| Wazuh Agent-3 | Windows XP | 512 MB |

**Wazuh agent - Wazuh server communication**

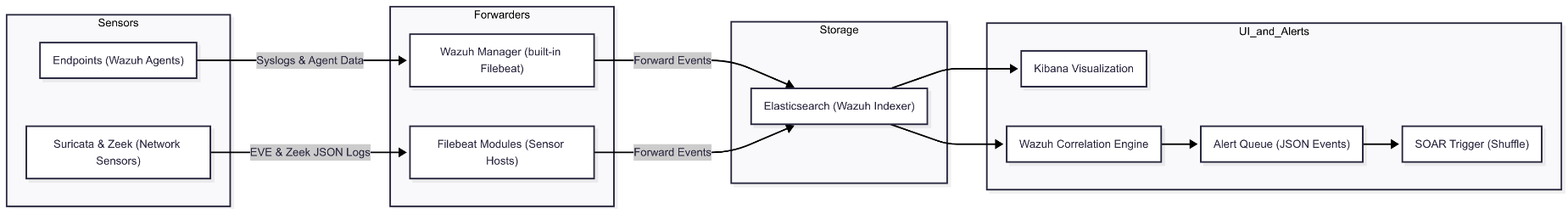
The Wazuh agent continuously sends events to the Wazuh server for analysis and threat detection. To start shipping this data, the agent establishes a connection with the server service for agent connection, which listens on port 1514 by default (this is configurable). The Wazuh server then decodes and rule-checks the received events, utilizing the analysis engine. Events that trip a rule are augmented with alert data such as rule ID and rule name. Events can be spooled to one or both of the following files, depending on whether or not a rule is tripped:

* *The file****/var/ossec/logs/archives/archives.json****contains all events whether they tripped a rule or not.*
* *The* ***file /var/ossec/logs/alerts/alerts.json****contains only events that tripped a rule with high enough priority (the threshold is configurable).*

The Wazuh messages protocol uses AES encryption by default, with 128 bits per block and 256-bit keys. Blowfish encryption is optional.

**Wazuh server - Wazuh indexer communication**

The Wazuh server uses Filebeat to securely transmit alert and event data to the Wazuh indexer via TLS encryption. Filebeat monitors output data from the Wazuh server and forwards it to the Wazuh indexer, which listens on port 9200/TCP by default. Once indexed, you can analyze and visualize the data through the Wazuh dashboard. The Vulnerability Detection module updates the vulnerability inventory. It also generates alerts, providing insights into system vulnerabilities. The Wazuh dashboard queries the Wazuh RESTful API (by default listening on port 55000/TCP on the Wazuh server) to display configuration and status-related information of the Wazuh server and agents. It can also modify agents or server configuration settings through API calls. This communication is encrypted with TLS and authenticated with a username and password.

With Suricata and Zeek added to the pipeline, network-based alerts (e.g. signature hits or suspicious traffic patterns) also feed into TheHive via the SIEM and automation layer, giving analysts unified visibility into both host and network threats.

| **Component** | **Port** | **Protocol** | **Purpose** |
| --- | --- | --- | --- |
| **Wazuh server** | **1514** | **TCP (default)** | **Agent connection service** |
| **1514** | **UDP (optional)** | **Agent connection service (disabled by default)** |
| **1515** | **TCP** | **Agent enrollment service** |
| **1516** | **TCP** | **Wazuh cluster daemon** |
| **514** | **UDP (default)** | **Wazuh Syslog collector (disabled by default)** |
| **514** | **TCP (optional)** | **Wazuh Syslog collector (disabled by default)** |
| **55000** | **TCP** | **Wazuh server RESTful API** |
| **Wazuh indexer** | **9200** | **TCP** | **Wazuh indexer RESTful API** |
| **9300-9400** | **TCP** | **Wazuh indexer cluster communication** |
| **Wazuh dashboard** | **443** | **TCP** | **Wazuh web user interface** |

**Alternatives:** Other SIEM/log platforms include OSSIM (AlienVault), Splunk, IBM QRadar or Elastic Stack (ELK). Splunk is commercial and powerful, whereas Wazuh is free and extensible. Compared to pure log managers like Graylog, Wazuh adds built-in detection rules and EDR capabilities. We chose Wazuh for its unified approach (SIEM+EDR/XDR) and vibrant open-source ecosystem, though one could also build a SOC on Graylog, ELK, or a distribution like Security Onion (which bundles Wazuh/Suricata).

1. **Suricata and Zeek Network Sensors**

Suricata and Zeek add network-level detection and context to our SOC. Suricata is an open-source, high-performance **signature-based NIDS** that inspects traffic against known threat rulesets. It analyzes packets (even encrypted SSL/TLS fingerprints and HTTP requests) in real time and emits alerts whenever traffic matches a malicious rule. Zeek (formerly Bro) is a powerful **network analysis framework** that passively monitors all traffic and logs rich protocol metadata. Zeek records every connection and protocol event (e.g. TCP/UDP/ICMP sessions, DNS queries, HTTP requests, TLS handshakes, etc. – see examples) into structured logs. In practice, Suricata provides real-time signatures and alerting, while Zeek provides detailed behavioral logs for investigation. Together they greatly enhance visibility: a Suricata alert on a suspicious session can be correlated with Zeek’s HTTP/DNS/TLS logs for that same session, or with host events collected by Wazuh/XDR.

**Roles in SOC**

Suricata’s role in the SOC is **threat detection by signatures**. It watches mirrored traffic and compares packets to known IOCs or exploit patterns. When a rule is violated, Suricata generates an alert (in its EVE JSON log). These alerts are forwarded to the SIEM for correlation. Zeek’s role is **passive monitoring and metadata logging**. It does not rely on predefined rules; instead it “peels the onion” of traffic, logging connection summaries and protocol details. Zeek outputs multiple log types (e.g. conn.log for all connections, http.log, dns.log, ssl.log, notice.log, etc. for specific protocols). These logs give analysts context for what happened on the network – for example, Zeek’s dns.log will show all domain queries, and http.log will show full HTTP URLs requested, along with Suricata’s alert for any malicious HTTP URI. Zeek can even extract files from traffic or generate its own “notice” alerts for anomalies. In summary: Suricata alerts identify **known threats** in traffic, and Zeek logs provide a full-picture record of network activity around those alerts.

**Deployment and Network Placement**

Suricata and Zeek should be deployed on dedicated sensor hosts attached to strategic network taps or SPAN/mirror ports. They must see all relevant traffic to/from the endpoints. In practice, this means placing sensors on network taps or configured mirror ports at core switches, DMZ gateways or data centre aggregation points. For example, a span port on the Internet gateway or a switch that bridges east-west traffic can feed both a Suricata node and a Zeek node. Suricata can run in IDS mode (passive) on this mirror, or in inline IPS mode if desired (requiring bidirectional taps). Zeek typically runs in passive mode on a machine (“Zeek sensor”) dedicated to traffic capture, with a NIC in promiscuous/tap mode. Proper placement is critical: as one guide notes, “deploy Suricata on a SPAN/mirror port or network tap to monitor all relevant traffic”. Similarly, Zeek users are advised to identify “a single location… to instrument with a network tap or switch span port that provides the maximum visibility”. Optional sensors like honeypots (e.g. Cowrie SSH honeypot) can also be run on the network; these too forward logs (via Wazuh agents) into the SOC pipeline. In all cases, the key is that Suricata and Zeek see the same traffic of interest as the Wazuh agents on the endpoints, enabling cross-correlation.

**Integration with Wazuh/ELK Stack**

Network sensor logs are collected and forwarded into the Wazuh/Elastic stack for indexing and correlation. Suricata’s JSON output (the EVE log, typically at */var/log/suricata/eve.json*) can be read directly by a Wazuh agent or shipped by Filebeat. For example, the Wazuh agent can be configured with a *<localfile>* for */var/log/suricata/eve.json* (with *<log\_format>json</log\_format>*), allowing the manager to automatically parse Suricata alerts. Wazuh’s built-in rules can then generate correlated alerts in the dashboard. Alternatively, Elastic Filebeat provides a **Suricata module**: enabling the Suricata module in Filebeat will automatically read the EVE JSON and apply an ingest pipeline to structure the data for Elasticsearch. Similarly, Zeek’s logs (which can be output in JSON via Zeek’s scripts) can be collected by Filebeat’s **Zeek module**, which parses Zeek’s JSON logs into Elasticsearch fields. In practice, one could deploy Filebeat on the Suricata and Zeek sensor hosts, or a Wazuh agent that tails their log files, to ship all network data to the Wazuh indexer (Elasticsearch). Once in Elasticsearch, these logs become searchable and displayable alongside host logs.

**Correlation and Alert Forwarding**

With Suricata and Zeek logs in the indexer, they can be correlated with other events. For example, a Suricata rule hit on a workstation’s IP can trigger a Wazuh alert linked to that endpoint. An analyst can then use Zeek’s logs to “pivot” on the same network connection for more detail (e.g. what HTTP or DNS records were involved). Zeek logs also feed threat intel analysis: some deployments subscribe to Zeek logs, extract IOCs (file hashes, IPs, domains), and cross-check them against MISP feeds. In one SOC prototype, a service forwarded Zeek events to MISP, and any matches (“malicious IoCs”) were written back into Wazuh as alerts. Similarly, Suricata’s own rule alerts (via its community signature sets) enrich host alerts. Both sensors’ data are visible in the Wazuh dashboard: indeed, the Wazuh/Kibana interface can display Suricata and Zeek events for threat hunting. From there, alerts can be pushed into TheHive for incident tracking and into Cortex analyzers for enrichment. Automated workflows in Shuffle can be triggered by these network alerts (for example, creating a case in TheHive or querying threat intel). In short, adding Suricata and Zeek brings network-based detections into the Wazuh-driven SOC pipeline, supplying signature alerts and contextual logs that augment endpoint alerts for comprehensive correlation.

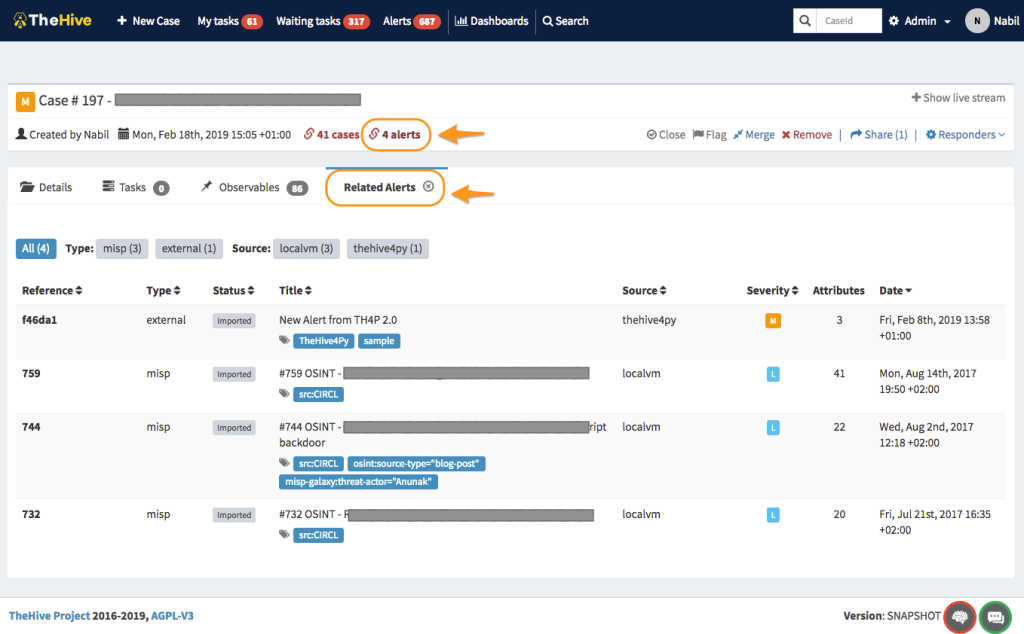
1. **TheHive (Incident Response / Case Management)**

**TheHive** is a free, open-source Security Incident Response Platform (SIRP) for managing investigations. It provides a **collaborative case management** interface where analysts can triage alerts, attach evidence, assign tasks, and document findings. TheHive lets multiple analysts work simultaneously on cases (real-time updates, tagging, threaded tasks) and supports alert ingestion from external sources (SIEMs, emails, Threat Intel feeds).

* **Use:** When Wazuh (or another SIEM) generates an alert, TheHive can automatically create an incident/case. Analysts then *collaborate* on the case, adding observables (IPs, files, domains, etc.) and tracking response actions. TheHive’s UI shows the status of all cases and tasks in one place, reducing alert fatigue and duplication.
* **Features:** TheHive supports *Case/Task Management*, letting you template investigations, attach evidence, set deadlines, and maintain an audit trail. It also has **alert triage** – imported alerts can be reviewed and either dismissed or escalated to full cases.
* **Scale:** It is designed for SOCs/CERTs of all sizes. The current TheHive 5 release is fully open-source and scales via multi-instance clustering.

TheHive integrates deeply with Cortex (below) for automated analysis and with MISP for intelligence sharing. For example, analysts can run Cortex analyzers on case observables directly from TheHive, or push new IOCs back to MISP.

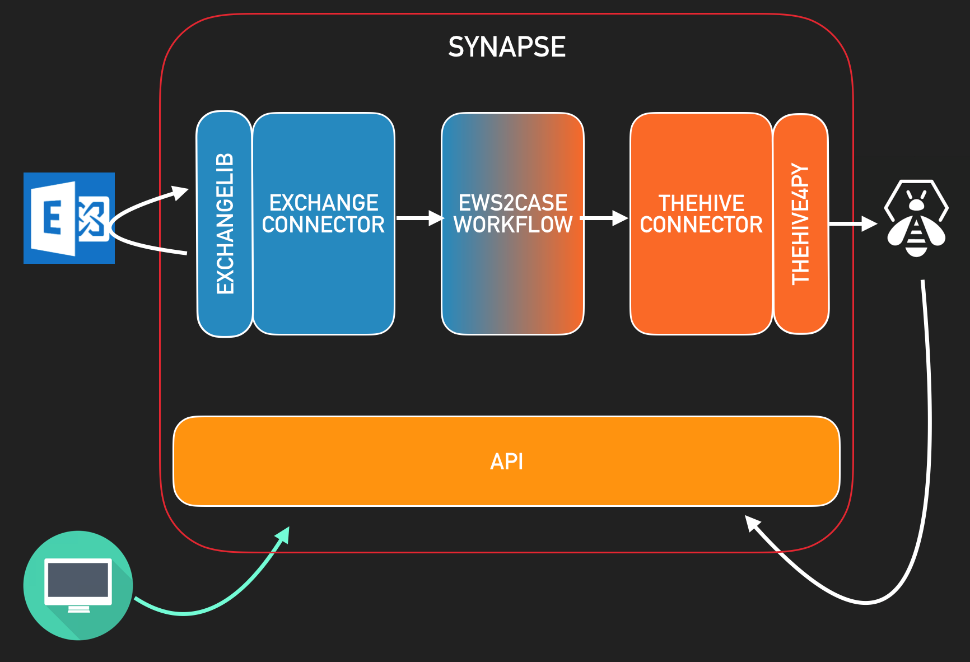
**Alternatives:** Commercial platforms like Palo Alto XSOAR (Demisto) or Splunk SOAR exist, but TheHive/Cortex is free and self-hosted. Other open SIRP options include RTIR (Request Tracker for Incident Response) or simple ticketing systems, but TheHive is specialized for security workflows. Because it’s built by and for security teams, TheHive handles common IR needs out-of-the-box.



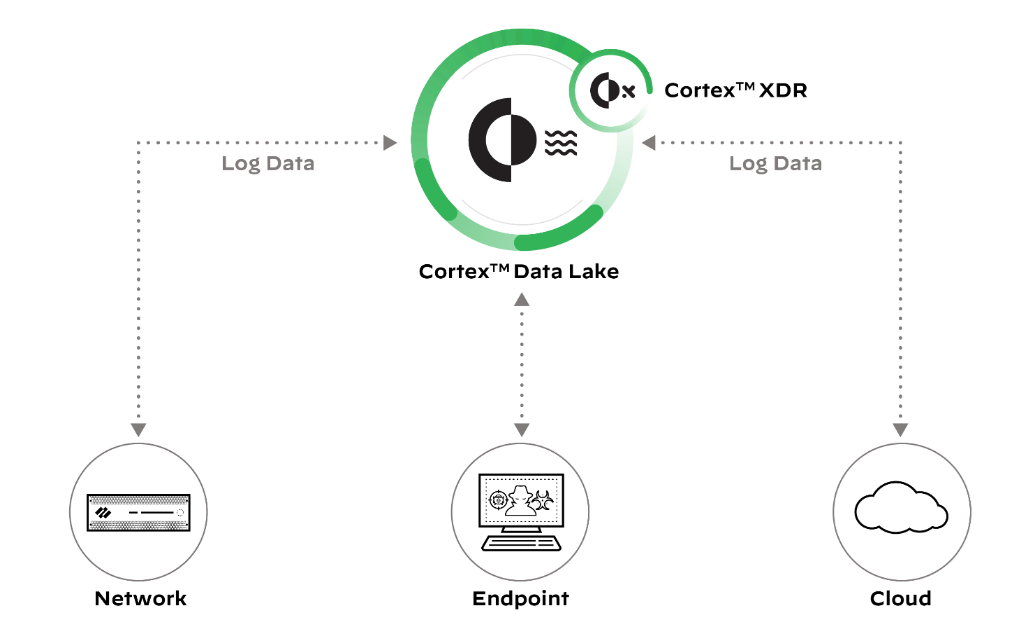
*Overview of the dashboard*

**Automation in TheHive**

Synapse is a Python 3 app that bridges incident sources (like firewalls, IDS, SIEM) and TheHive, automating repetitive tasks in incident response (IR) to improve analyst efficiency. It uses connectors to interact with different devices and workflows (Python scripts) to automate tasks such as case creation, assignment, and log management. An API allows users to trigger workflows and listen for real-time events. Synapse is open-source and free, with connectors for Exchange and upcoming ones for QRadar. Its goal is to reduce the manual effort involved in incident handling and keep analysts engaged.



1. **Cortex (Analysis & Enrichment Engine)**

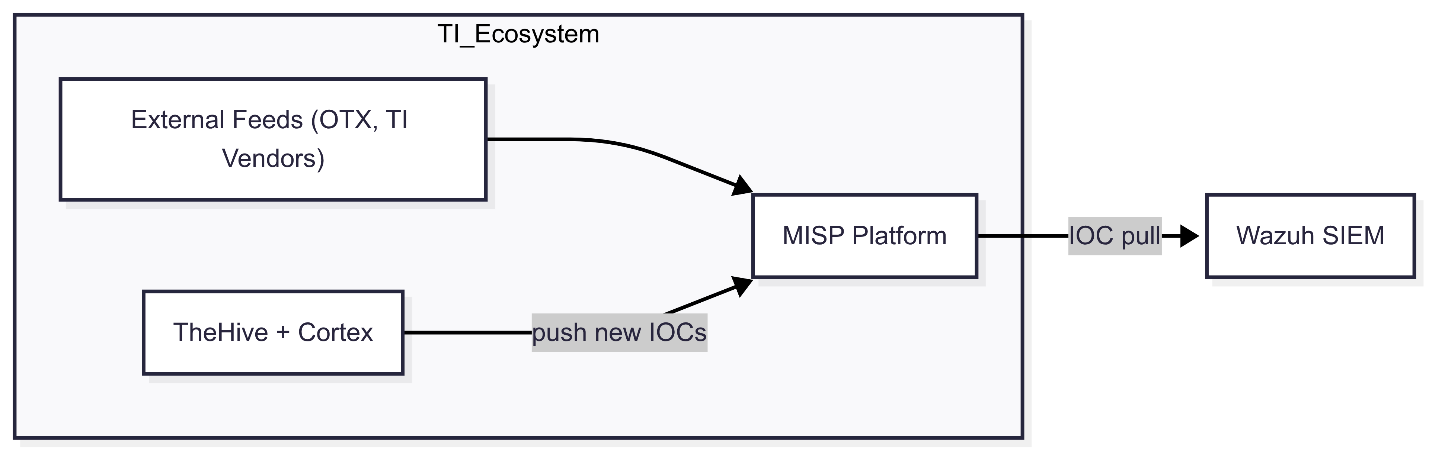


**Cortex** is the automated analysis engine behind TheHive. It is free software that lets you analyze “observables” (IPs, URLs, hashes, etc.) at scale by querying threat intelligence and analysis services. In practice, when a case is open in TheHive, analysts send observables to Cortex, which runs dozens of analyzers (VirusTotal, Whois, Shodan, etc.) and returns results. Cortex can operate in single-query mode or bulk mode via API.

* **Role:** Cortex automates the grunt work of enrichment. For example, it can take an IP address, run it through WHOIS, VirusTotal, AbuseIPDB, etc., then return a summary (reputation, history, related domains) to TheHive. This accelerates investigations by surfacing context quickly.
* **Integration:** TheHive calls Cortex via its REST API. Analysts can launch analyses manually or automate them via playbooks (e.g. a Shuffle workflow can trigger Cortex on new alerts). Observables extracted from network sensors (Suricata alerts, Zeek logs) also flow into cases, and Cortex enriches those IPs, domains, and file hashes just as it does for host-derived data**.** Cortex results are formatted by TheHive’s reporting engine for easy reading. Unlike general-purpose analysis tools, Cortex is tailored for security use: it comes with *dozens* of specialized analyzers (file scanners, IP reputation, malware sandboxes, etc.) and can be extended with custom Python scripts. Since Cortex is open-source, new analyzers or connectors (e.g. to proprietary APIs) can be added as needed.

**Alternatives:** Other observables-analysis tools include commercial offerings or home-grown scripts. In many SOCs, analysts would manually query services; Cortex standardizes this. No direct free alternative matches its integration with TheHive. Note Cortex was made by TheHive Project for this purpose, and it remains open-source.

1. **MISP (Threat Intelligence Platform)**

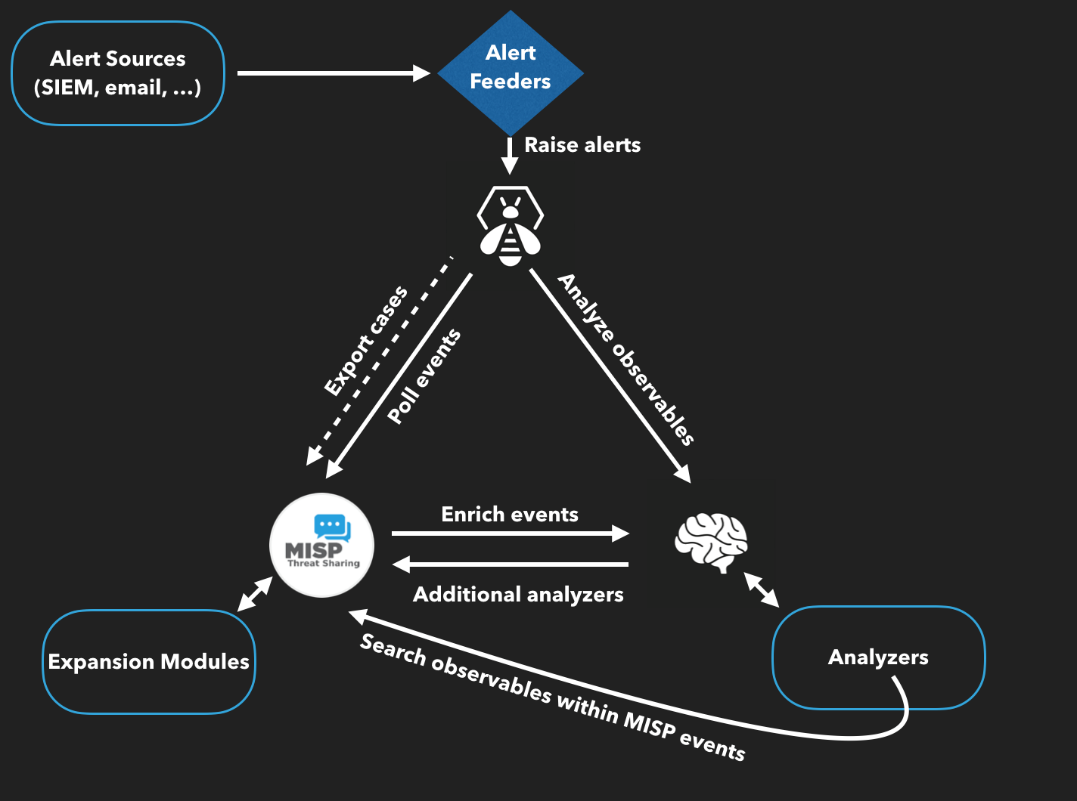
****

**MISP (Malware Information Sharing Platform)** is an open-source threat intelligence platform for sharing, storing, and correlating indicators of compromise. In a SOC, MISP serves as the repository of threat data (malicious IPs, domains, file hashes, TTPs, etc.) that the team has collected or received from partners.

* **Use:** MISP lets you create “events” that group related IOCs. These events can be shared with trusted communities. In TheHive, MISP integration allows importing events as cases or sending validated IOCs back into MISP for distribution. For example, a phishing URL found in an incident can be pushed to a MISP event and shared with peers.
* **Sources:** MISP can ingest feeds from other communities, as well as internal intelligence from network sensors (Suricata EVE alerts, Zeek-extracted IOCs) via automation (e.g. Shuffle or Wazuh). You can also run Cortex analyzers directly on MISP event fields (Cortex has built-in MISP modules). This bi-directional flow enriches both tools”.
* **Collaboration:** By using MISP, organizations contribute to and benefit from collective threat data. It’s licensed under open terms, so anyone can run a MISP instance.

Alternatives include free services like AlienVault OTX or commercial TIPs. Unlike a passive feed, MISP is a full platform for structured sharing and analysis. We chose MISP because it is widely adopted in open-source SOC stacks and integrates natively with TheHive/Cortex. It differs from a simple feed in that it maintains metadata (taxonomy, clusters, confidence levels) and fosters two-way sharing.

[**TheHive, Cortex and MISP: How They All Fit Together**](https://blog.thehive-project.org/2017/06/19/thehive-cortex-and-misp-how-they-all-fit-together/)



**TheHive**

TheHive is a Security Incident Response Platform (SIRP). It can receive alerts from different sources (SIEM, IDS, email. etc.) via its REST API. This is where alert feeders come into play.

**Alert Feeders**

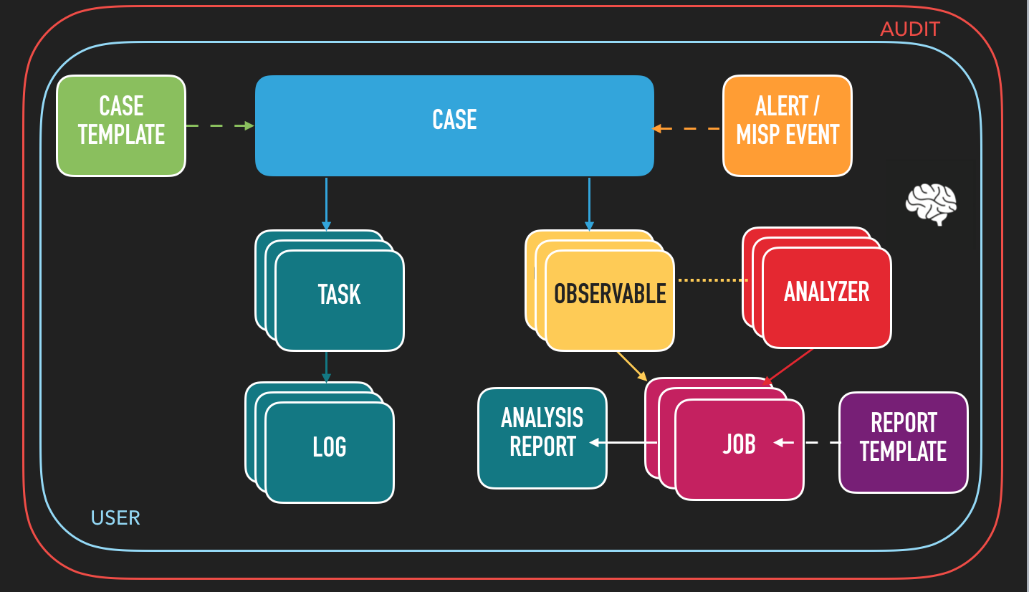
Think of an alert feeder as a specialized program which consumes a security event (SIEM alert, email report, IDS alert, and so on), parses it and outputs an alert that its sends to TheHive through TheHive4py, the Python library we provide to interact with TheHive’s REST API.

We do not supply such feeders but developing them should be straightforward. If not, let us know and we’ll do our best to help you out. These feeders can pull events not only from SIEMs, email systems, and MISP, but also from network-sensor alerts (Suricata EVE JSON, Zeek notice logs) via the Wazuh event stream or direct Filebeat-to-TheHive feeders.

**Alerts**

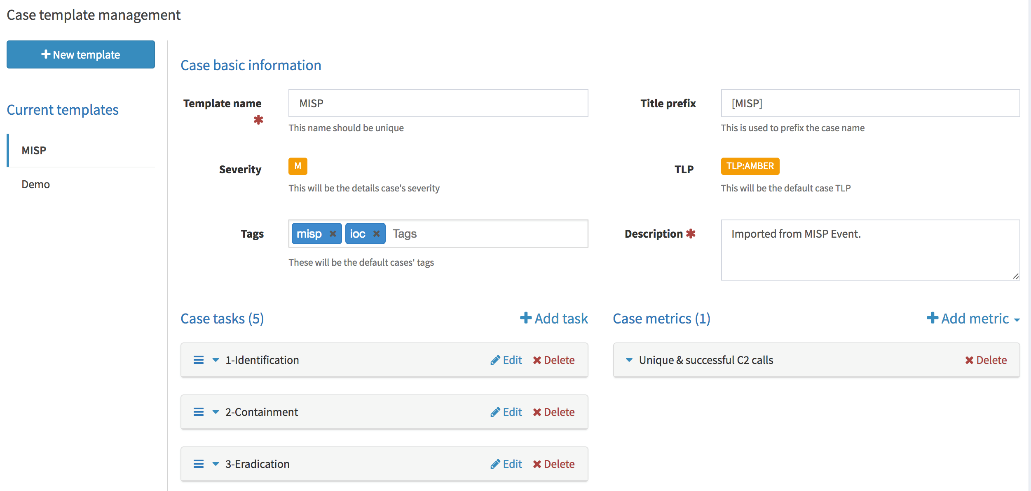
Any alert sent to TheHive whether from Wazuh, network sensors (Suricata/Zeek), email, or other sources will show up in its Alerts pane. In addition to the sources mentioned above, new or updated MISP events will show up as well in that area if you configured TheHive to connect to one or several MISP instances. If so, TheHive will poll those MISP instances at every interval looking for new or updated events. If there are any, TheHive will generate an alert which will end up in the Alerts pane. Alerts can be ignored, marked as read, previewed, and imported. When an alert is imported, it becomes a case that needs to be investigated.

**Cases**



A case can be generated from an alert or created from scratch. It is subdivided into tasks (think identification, containment, eradication, check proxy logs, and so on) and observables (IP addresses, hashes, email addresses, domain names, URLs…). When analysts are working on tasks, they add logs as they go. In TheHive’s terminology, logs are text entries which may contain attachments to help analysts record what they have been doing. Logs can be written using Markdown or a rich-text editor.

**Case Templates**



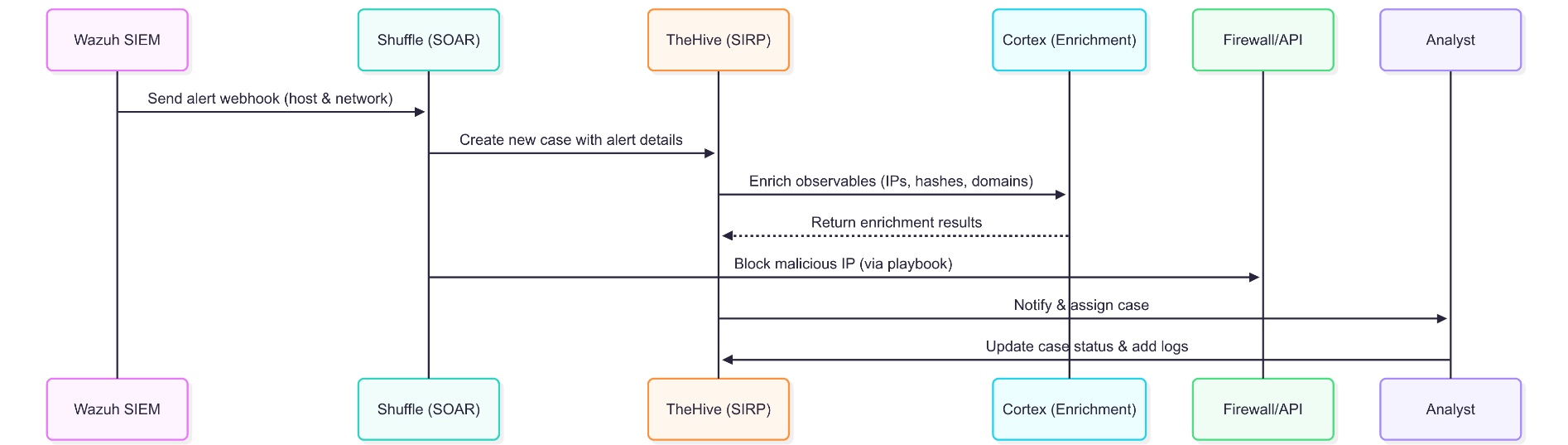
You don’t need to add the same tasks over and over when working on cases belonging to a given category (DDoS, Malspam, APT, …). You can create custom templates to which you add tasks as shown.

1. **Shuffle (SOAR / Workflow Automation)**

**Shuffle** is an open-source SOAR (Security Orchestration, Automation and Response) platform. It provides a visual workflow builder and connectors (apps) for many security tools. In practice, Shuffle can listen for alerts (e.g. via webhook from Wazuh) and execute playbooks: it can create TheHive cases, run Cortex analyses, invoke MISP updates, send Slack notifications, etc. Shuffle’s GUI lets non-programmers drag-and-drop these integrations into end-to-end automations.

Key points about Shuffle:

* **Workflows:** You build “flows” of actions (e.g. on alert: query VirusTotal, update ticket, block IP). The example screenshot below (from Shuffle’s docs) shows an email-phishing workflow: it fetches emails, analyzes attachments (YARA, VirusTotal), creates alerts, and updates TheHive cases. This is a fraction of what’s possible.



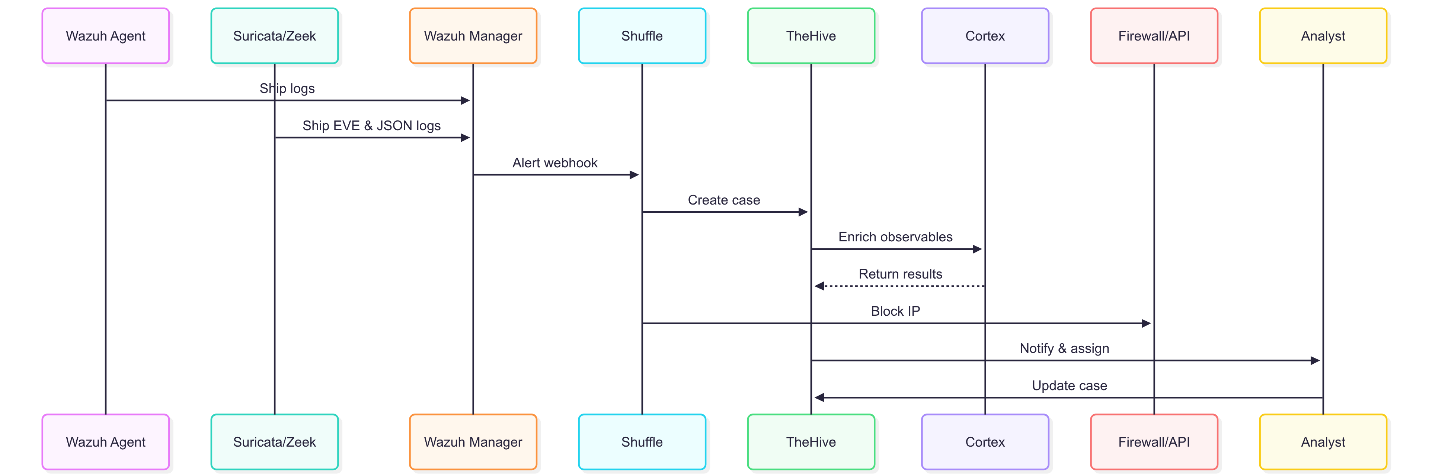
* **Integration:** Shuffle has prebuilt “apps” (connectors) for TheHive, Cortex, MISP, VirusTotal, SIEMs, network sensors (Suricata, Zeek), ticketing systems, etc. This lets it orchestrate between our tools. For instance, a Suricata or Zeek alert (delivered via Wazuh or direct webhook) could trigger Shuffle to gather context (using Cortex/MISP) and then file an official incident report.
* **Collaboration:** Shuffle supports multi-tenancy and sharing. Many security teams share playbooks in the Shuffle community. Because it’s open-source (AGPL) and modular, organizations can extend it or self-host with full access.

**Alternatives:** Commercial SOAR tools (Palo Alto XSOAR, Splunk Phantom, etc.) are feature-rich but expensive. Shuffle offers much of this functionality at no cost. Other automation platforms (like StackStorm or even Zapier) can be adapted, but Shuffle is built specifically for security use-case.

1. **Workflow Summary**

Putting it all together, a day in our SOC might flow like this:

1. **Log Collection:**  
   Endpoints and devices send logs via Wazuh agents into the Wazuh Manager. **Network sensors (Suricata, Zeek) feed EVE JSON and Zeek logs (via Filebeat or Wazuh agents) into Elasticsearch.** Optional honeypots (Cowrie) can also forward logs via Wazuh.
2. **Detection & Alerting:**
   * **Wazuh rules** trigger alerts on host anomalies (malicious login, known malware hash).
   * **Suricata rules** trigger alerts on known network threats.
   * **Zeek scripts/notices** generate metadata-based alerts for anomalous traffic.  
     All alerts index into Elasticsearch/Kibana for searching.
3. **Case Creation:**  
   A high-severity alert from Wazuh, Suricata, or Zeek triggers Shuffle via a webhook. Shuffle automatically creates a case in TheHive with the alert details.
4. **Enrichment:**  
   TheHive assigns the case to analysts. Observables from the alert (IPs, file hashes, domains) are automatically sent to Cortex. Cortex returns threat intelligence (from VirusTotal, AbuseIPDB, etc.) and MISP. TheHive displays these results to the team.
5. **Analysis & Response:**  
   Analysts review the case in TheHive, marking tasks (kill processes, block IPs). They may use Shuffle playbooks to run active responses (e.g. update firewall rules, quarantine host). Throughout, they update the case log.
6. **Intelligence Sharing:**  
   If new IOCs are found (malicious URL, C2 IP), analysts export them to MISP, enriching community intelligence.
7. **Reporting:**  
   Dashboards in Kibana update with the incident statistics; periodic reports are generated from TheHive for management.



*End-to-End Sequence Diagram*