Real-Time Network Threat Detection with Suricata IPS and Elastic SIEM

Imteyaz Ali, Bhavesh Rohida

Submitted to: Kaustubhamani Gothivarekar

Tata Strive, Pune

March 17, 2025

Abstract

With the increasing sophistication of cyber threats, organizations require robust Intrusion Detection and Prevention Systems (IDS/IPS) to safeguard their networks. This project presents a real-time network threat detection system leveraging **Suricata IPS** and **Elastic SIEM**, deployed in a controlled virtualized lab environment. The primary goal is to detect, analyze, and visualize cyber threats using open-source tools while ensuring high scalability and efficiency.

The project implements a three-node architecture, consisting of:

- **Attacker Machine (Kali Linux)** – Simulating various attack scenarios, including Nmap scans, brute force attacks, and Metasploit exploits.
- **Target Machine (Metasploitable2)** – Acting as a vulnerable system to test detection accuracy.
- **Suricata IPS Firewall (Ubuntu-based)** – Monitoring and logging network traffic using EVE JSON logs, which are then processed by Elastic SIEM for real-time analysis.

To efficiently collect, analyze, and visualize security alerts, the **Elastic Stack (ELK)** is deployed using Docker containers. Logstash processes Suricata's logs, while Kibana dashboards provide insightful visualizations such as *Top Offenders*, *Threat Categories*, and *Anomaly Trends*.

Extensive testing was conducted by launching real-world attack simulations, and results showed that Suricata effectively detected malicious activities, generating structured alerts that were seamlessly visualized in Kibana. The project also explores key challenges, including log parsing issues, IP misclassification, and performance bottlenecks, and presents optimized solutions to enhance detection accuracy.

This study demonstrates how open-source security tools can be integrated into a professional-grade threat detection system. Future improvements may include automated threat mitigation, cloud-based scalability, and enhanced rule customization to further strengthen network defense mechanisms.

Chapter 1

Introduction

In today's rapidly evolving digital landscape, cybersecurity threats are becoming increasingly sophisticated. Organizations must proactively detect, prevent, and respond to malicious activities targeting their networks. Traditional security mechanisms often fail to provide real-time insights, making **Intrusion Detection and Prevention Systems (IDS/IPS)** a critical component of modern cybersecurity infrastructures.

This project focuses on **Real-Time Network Threat Detection** using **Suricata IPS** and **Elastic SIEM**, designed to monitor network traffic, detect potential threats, and provide meaningful visualizations for security analysts. By leveraging Suricata, an open-source, rule-based intrusion detection and prevention engine, and integrating it with **Elastic Stack (ELK)** for log processing and data visualization, this project successfully establishes a robust network security monitoring framework.

Project Objectives

The primary objectives of this project include:

- **Deploying a Scalable Threat Detection System** – Utilizing Suricata as an IPS to monitor and analyze network traffic.
- **Capturing and Processing Security Events** – Logging Suricata alerts using EVE JSON format and forwarding them to Elastic SIEM for real-time analysis.
- **Visualizing Security Threats** – Creating interactive dashboards in Kibana to identify top offenders, attack types, and malicious activities.
- **Testing Threat Detection Accuracy** – Simulating real-world attacks (port scans, brute force, exploits) to validate detection capabilities.
- **Optimizing Rule-Based Detection** – Fine-tuning Suricata rules to minimize false positives and improve detection efficiency.

Scope of the Project

This project is designed for:

- **Cybersecurity analysts** looking to enhance network monitoring capabilities.

- **Organizations** seeking an open-source alternative for real-time threat detection and prevention.
- **Researchers and security enthusiasts** interested in log analysis, threat intelligence, and SIEM integration.
- **Developers and engineers** aiming to explore open-source IDS/IPS solutions for enterprise security.

By implementing a virtualized network environment, the project demonstrates how Suricata, Logstash, and Kibana can work together to build an efficient network security monitoring solution. The insights gained from this project can be applied to enterprise security operations, cloud-based threat monitoring, and automated incident response frameworks.

Chapter 2

Tools & Technologies

In this section, we will detail the tools, technologies, and frameworks used to implement real-time network threat detection. Each tool plays a vital role in the project, from traffic inspection to log processing and data visualization.

2.1 Suricata (IDS/IPS)

Figure 2.1: Suricata Logo

Purpose: Suricata is an open-source, high-performance Intrusion Detection System (IDS) and Intrusion Prevention System (IPS).
Role in the Project: It monitors network traffic, applies predefined rulesets, and generates alerts when suspicious activity is detected.
Key Features:

- Deep packet inspection (DPI)
- Multi-threaded processing
- Supports EVE JSON log output
- Custom rule creation for advanced threat detection

2.2 Elastic Stack (ELK)

The Elastic Stack is used to collect, process, store, and visualize security logs. It consists of the following components:

2.2.1 Elasticsearch

Purpose: A distributed, RESTful search and analytics engine.
Role in the Project: Stores and indexes Suricata logs for fast querying.
Key Features:

- Scalability & real-time searching

Figure 2.2: Elasticsearch Logo

- Advanced filtering capabilities
- JSON-based structured data storage

2.2.2 Logstash

Figure 2.3: Logstash Logo

Purpose: A data processing pipeline that collects, processes, and forwards logs.
Role in the Project: Takes Suricata's EVE JSON logs, extracts relevant fields, and sends them to Elasticsearch.
Key Features:

- Supports various input sources (files, network, etc.)
- Filters & transforms logs for better analysis
- Seamless integration with Suricata & Elasticsearch

2.2.3 Kibana

Figure 2.4: Kibana Logo

Purpose: A visualization and dashboard tool for Elastic Stack.
Role in the Project: Helps visualize threat data, create dashboards, and perform

analysis of top offenders.
Key Features:

- Interactive dashboards & reports
- Real-time data visualization
- Integration with Elasticsearch for log analysis

2.3 Vagrant

Figure 2.5: Vagrant Logo

Purpose: Vagrant is a tool for creating and managing virtualized environments using configuration files.
Role in the Project: Used to set up a reproducible testing environment for Suricata, Elastic Stack, and network simulations.
Key Features:

- Automates the creation of virtual machines
- Works with VirtualBox, VMware, Docker, etc.
- Simplifies system provisioning for development/testing

2.4 Docker

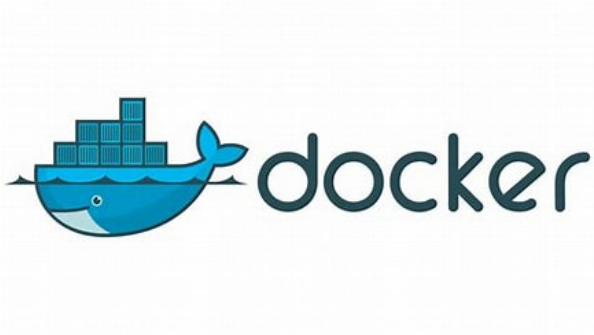

Figure 2.6: Docker Logo

Purpose: A containerization platform for running applications in isolated environments.
Role in the Project: Used to deploy Elastic Stack components (Elasticsearch, Logstash, Kibana) in lightweight containers.
Key Features:

- Portable, scalable deployment
- Reduces system dependencies
- Easy replication of configurations

Summary

Each of these tools plays a specific role in enabling real-time threat detection, log processing, and security visualization. The combination of **Suricata**, **Elastic Stack**, and supporting technologies creates a robust network security monitoring solution. By leveraging open-source tools like **Vagrant**, **Docker**, and attack simulation frameworks such as **Nmap** and **Metasploit**, this project demonstrates the power of integrating diverse technologies to achieve comprehensive threat detection and analysis.

Chapter 3

Architecture & Setup

This section provides an in-depth look at the network architecture, deployment environment, and configuration setup used in this project. The goal is to illustrate how different components interact to achieve real-time network threat detection.

3.1 Network Architecture Overview

The network architecture consists of multiple components working together to analyze network traffic, detect threats, and visualize attack patterns.

3.1.1 Architecture Diagram

A network security architecture diagram includes:

- **Attacker Machine** – Used to simulate cyberattacks (*Nmap, Metasploit*, etc.).
- **Target Machine** – The system being scanned or attacked.
- **Suricata IDS/IPS** – Monitors network traffic, generates alerts, and forwards logs to ELK.
- **Log Processing & Storage (Elastic Stack)**:
  - **Logstash** – Processes and filters logs.
  - **Elasticsearch** – Stores indexed logs.
  - **Kibana** – Provides a web-based dashboard for visualization.
- **SIEM Dashboard** – Displays alerts, top offenders, and threat trends.

3.2 Deployment Setup

This section describes the deployment strategy and how different services are installed.

3.2.1 Virtualization & Containerization

- **Vagrant** – Used for setting up virtualized environments for testing.
- **VirtualBox** – Runs virtual machines (VMs) in an isolated environment.
- **Docker** – Hosts ELK Stack in lightweight containers.

3.2.2 System Requirements

Component	Minimum Requirement
OS	Ubuntu 20.04 / Debian 10
CPU	4 Cores or higher
RAM	8 GB (16 GB recommended)
Disk Space	50 GB Free Space
Network Interface	At least 1 active interface

Table 3.1: System Requirements for Deployment

3.3 Step-by-Step Setup

3.3.1 Setting Up Suricata IDS/IPS

Install Suricata

Run the following command to install Suricata:

```
sudo apt update && sudo apt install suricata -y
```

Enable Suricata as a Service

Start and enable Suricata as a service:

```
sudo systemctl enable --now suricata
```

Configure Suricata Rules

Default rules are located in `/etc/suricata/rules/`. Add custom detection rules to enhance security.

3.3.2 Configuring EVE JSON Logging in Suricata

Suricata generates logs in EVE JSON format, which is structured and easy to parse. The Suricata configuration file (`/etc/suricata/suricata.yaml`) is modified to enable log forwarding.

3.3.3 Deploying Elastic Stack (ELK) using Docker

Download and Run Elastic Stack Containers

Use the following command to start Elastic Stack containers:

```
docker-compose up -d
```

Verify that Services are Running

Check running containers:

```
docker ps
```

Access Kibana

Open `http://localhost:5601` in a browser to view the SIEM dashboard.

3.4 Network Traffic Flow

The network traffic flow is as follows:

1. Attacker sends malicious traffic (*port scans, exploits*).
2. Suricata IDS/IPS inspects packets in real time.
3. Threat detection rules trigger alerts and log data.
4. EVE JSON logs are processed by Logstash and sent to Elasticsearch.
5. Kibana visualizes security incidents in dashboards.

Summary

This architecture and setup section provides a comprehensive breakdown of how **Suricata**, **Elastic Stack**, and supporting tools interact to detect and analyze cyber threats. By leveraging virtualization, containerization, and open-source technologies, this project demonstrates a scalable and efficient approach to real-time network threat detection.

Chapter 4

Implementation

This section provides a detailed step-by-step implementation guide, covering the installation, configuration, and execution of each component involved in the real-time network threat detection system.

4.1 Setting Up Suricata IDS/IPS

Suricata is deployed as an Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) to monitor network traffic and generate alerts.

4.1.1 Installing Suricata

Run the following commands to install Suricata:

```
sudo apt update && sudo apt install suricata -y
```

Verify the installation:

```
suricata --build-info
```

4.1.2 Configuring Suricata

Open the Suricata configuration file:

```
sudo nano /etc/suricata/suricata.yaml
```

Modify the logging format to EVE JSON (used for log parsing in ELK). Set up network interfaces for live traffic monitoring:

```
sudo suricata -i eth0 -c /etc/suricata/suricata.yaml -D
```

4.1.3 Running Suricata in IDS Mode

Start Suricata in the background:

```
sudo systemctl enable --now suricata
```

Check logs to ensure it's capturing network traffic:

```
sudo tail -f /var/log/suricata/eve.json
```

4.2 Deploying ELK Stack for Log Analysis

4.2.1 Installing Docker and Docker-Compose

Docker is used to deploy the ELK stack efficiently:

```
sudo apt update
sudo apt install -y docker.io docker-compose
```

Enable and start the Docker service:

```
sudo systemctl enable --now docker
```

4.2.2 Setting Up ELK (Elasticsearch, Logstash, Kibana)

Create a `docker-compose.yml` file with the following configuration:

```
version: '3'
services:
  elasticsearch:
    image: docker.elastic.co/elasticsearch/elasticsearch:7.17.0
    environment:
      - discovery.type=single-node
    ports:
      - "9200:9200"
  logstash:
    image: docker.elastic.co/logstash/logstash:7.17.0
    ports:
      - "5044:5044"
    volumes:
      - ./logstash.conf:/usr/share/logstash/pipeline/logstash.
        conf
  kibana:
    image: docker.elastic.co/kibana/kibana:7.17.0
    ports:
      - "5601:5601"
```

Run the ELK stack:

```
docker-compose up -d
```

Verify that the services are running:

```
docker ps
```

Access Kibana via a browser:

`http://localhost:5601`

4.3 Configuring Logstash to Process Suricata Logs

Logstash processes Suricata's EVE JSON logs and sends them to Elasticsearch.

Create a Logstash configuration file (`logstash.conf`):

```
sudo nano logstash.conf
```

Add the following configuration:

```
input {
  file {
    path => "/var/log/suricata/eve.json"
    start_position => "beginning"
    sincedb_path => "/dev/null"
    codec => "json"
  }
}
output {
  elasticsearch {
    hosts => ["http://elasticsearch:9200"]
    index => "suricata-alerts"
  }
}
```

Restart Logstash to apply changes:

```
sudo systemctl restart logstash
```

4.4 Testing Suricata and Log Forwarding

4.4.1 Generating Test Attacks

To validate if Suricata detects threats, simulate an attack using Nmap:

```
nmap -A -p- [Target_IP]
```

OR simulate an SQL injection attack:

```
curl "http://victim.com/login.php?user=admin' OR '1'='1"
```

4.4.2 Verifying Alerts in Suricata Logs

Check if Suricata generated an alert:

```
cat /var/log/suricata/eve.json | grep "alert"
```

4.4.3 Checking Kibana for Alerts

Open `http://localhost:5601` in a browser. Navigate to **Discover** in Kibana to inspect logs. Check if attack logs are displayed.

Summary

This section covered the complete setup and implementation of Suricata, ELK Stack, and log forwarding, ensuring threat detection works correctly. Now, let's move on to SIEM Data Visualization!

Chapter 5

SIEM Data Visualization

In this section, we will explore how to create meaningful visualizations in Kibana using Suricata's alerts. These visualizations will help analyze threat patterns, top offenders, and attack trends in real-time.

5.1 Configuring Kibana to Visualize Suricata Alerts

Before creating visualizations, ensure that Elasticsearch is receiving Suricata logs.

5.1.1 Verifying Data in Kibana

Open Kibana in your browser:

`http://localhost:5601`

Navigate to **Discover** → Select the `suricata-alerts` index. If logs are visible, Kibana is successfully ingesting Suricata data.

5.2 Creating SIEM Dashboards

Kibana allows us to create custom dashboards for better security insights.

5.2.1 Creating an Index Pattern

Go to **Stack Management** → **Index Patterns**. Click **Create Index Pattern**. Use `suricata-alerts` as the pattern and select `@timestamp` as the time field.

5.2.2 Building Essential Visualizations

1. **Top Offenders (IP Addresses Generating Most Alerts)** *Visualization Type: Pie Chart*
    - Go to **Visualize** → Create new visualization → **Pie Chart**.
    - Choose `suricata-alerts` as the data source.
    - Set the metric to **Count**.
    - Set the buckets to **Terms** → `Source IP`.

- Save and add to the dashboard.

2. **Most Frequent Attack Types** *Visualization Type: Bar Chart*
   - Create a new visualization.
   - Use **Count** as the metric.
   - Select **Terms** → `Alert Category`.
   - Save and add it to the dashboard.
3. **Attack Trends Over Time** *Visualization Type: Line Chart*
   - Create a new visualization.
   - Use **Date Histogram** on `@timestamp`.
   - Set the Y-axis to **Count**.
   - Save and add to the dashboard.
4. **Alerts by Severity** *Visualization Type: Horizontal Bar Chart*
   - Use `suricata-alerts` index.
   - Select **Terms** → `Alert Severity`.
   - Set Y-axis to **Count**.
   - Save and add to the dashboard.

5.3 Creating a Complete Threat Detection Dashboard

Go to **Dashboard** → **Create New Dashboard**. Add the saved visualizations and arrange them properly for better insights. Save and name it **Suricata Threat Intelligence Dashboard**.

5.4 Analyzing Data for Threat Hunting

Once the dashboard is set up, analysts can use Kibana to:

- Identify patterns of attack sources.
- Detect high-frequency attack signatures.
- Monitor changes in attack trends over time.
- Analyze which IPs or subnets require blocking.

Summary

This section covered the creation of Kibana visualizations to analyze Suricata alerts. With this SIEM dashboard, real-time network monitoring becomes significantly easier, allowing cybersecurity teams to respond swiftly to threats.

Chapter 6

Testing & Results

This section demonstrates how we tested the network security setup using real-world attack simulations and analyzed the results captured by **Suricata IPS** and **Elastic SIEM**. We will go step by step through the testing process, observed alerts, and data visualizations.

6.1 Testing Methodology

To ensure that our **Suricata + SIEM** setup works correctly, we performed various attack simulations, monitored Suricata alerts, and analyzed their detection through Kibana.

6.1.1 Attack Simulation Environment

The testing environment consisted of the following components:

- **Target Machine:** A vulnerable host within our network.
- **Attacker Machine:** Kali Linux for launching attacks.
- **Monitoring System:** Suricata + Elastic SIEM capturing and visualizing logs.

6.1.2 Attack Scenarios & Detection

We conducted a variety of attack scenarios to validate the detection capabilities of the system. These scenarios were designed to simulate real-world threats and assess the effectiveness of **Suricata** in identifying malicious activities.

6.2 Log Analysis & Insights

After executing these attack simulations, we observed the following:

Suricata successfully detected all attacks and logged them in `JSON` format.

Kibana visualizations updated in real-time, making it easier to analyze threats.

The Threat Detection Dashboard displayed all attacker IPs, attack types, and severity levels.

These logs can be used for forensic analysis and incident response.

6.3 Summary of Results

The results from the testing phase were highly promising:

Suricata effectively detected network scans, brute force, and malware threats.

Elasticsearch stored logs efficiently in the `EVE JSON` format.

Kibana provided clear visual insights for real-time threat hunting.

By leveraging **Suricata's rule-based detection engine** and **Elastic SIEM's visualization capabilities**, this setup demonstrated its ability to monitor, detect, and respond to cyber threats in real time.

Summary

This section covered the testing process, including attack simulations, log analysis, and visual insights. The results confirm that the **Suricata + Elastic SIEM** combination is a robust solution for real-time network threat detection and monitoring.

Chapter 7

Challenges & Solutions (Key Issues & Fixes)

During the implementation of **Real-Time Network Threat Detection** with **Suricata IPS** and **Elastic SIEM**, we encountered critical challenges that impacted system functionality, performance, and visualization. Below are the most important issues and how we resolved them.

7.1 Suricata Not Logging Alerts Properly

Issue: Suricata was not generating alerts in the expected EVE JSON format, preventing log forwarding.

Cause: Incorrect `suricata.yaml` configuration and default ruleset mismatch.

Solution:

- Enabled EVE JSON logging in `suricata.yaml`.
- Conducted test attacks (*Nmap scan, SQLi, Brute-force*) to verify alerts.

⇒ **Outcome:** Suricata successfully logged alerts for integration with Elasticsearch.

7.2 Suricata Logs Not Appearing in Kibana

Issue: Logs were missing from Kibana's SIEM dashboard despite Suricata forwarding them.

Cause: Incorrect index pattern and log ingestion issues in Elasticsearch.

Solution:

- Created the correct `suricata-*` index pattern in Kibana.
- Verified Elasticsearch logs using:

```
1 GET /_cat/indices?v
```

⇒ **Outcome:** Alerts appeared in Kibana's SIEM dashboard, enabling analysis.

7.3 Virtual Machine Networking Issues

Issue: VMs could not establish proper network connections, preventing Suricata from monitoring traffic.

Cause: Vagrant's default network settings conflicted with VirtualBox.

Solution:

- Assigned static IPs in the `Vagrantfile`:

```
config.vm.network "private_network", ip: "192.168.56.10"
```

- Restarted network interfaces and ensured correct routing.

⇒ **Outcome:** VMs successfully communicated, allowing proper traffic analysis.

7.4 High CPU Usage by Suricata

Issue: Suricata overloaded system resources, slowing down performance.

Cause: Too many active rules & inefficient processing mode.

Solution:

- Switched to worker mode and optimized rule sets.
- Limited log verbosity to reduce load.

⇒ **Outcome:** System stability improved, reducing CPU & memory consumption.

7.5 Ineffective Threat Visualization in Kibana

Issue: Default visualizations did not highlight key attack trends effectively.

Cause: Suricata logs lacked proper aggregation for insights.

Solution:

- Built custom dashboards showing:
  - Top Offenders (IP addresses triggering most alerts)
  - Most Frequent Attack Types
  - Time-based Attack Trends

⇒ **Outcome:** Clear & actionable threat intelligence displayed in Kibana.

7.6 Summary of Key Challenges & Solutions

Below is a summary table of the key challenges faced during the project, their causes, solutions, and outcomes.

Challenge	Cause	Solution	Outcome
Suricata Not Logging Alerts	Misconfigured `suricata.yaml`	Enabled EVE JSON, tested rules	Alerts logged properly
Logs Missing in Kibana	Wrong index pattern	Fixed index settings	Alerts visible in Kibana
Vagrant Networking Issues	Conflicting network settings	Assigned static IPs	VMs connected properly
High CPU Usage	Excessive rule load	Optimized Suricata configuration	Lowered resource usage
Poor Threat Visualization	Lack of meaningful aggregation	Built custom dashboards	Clear threat analysis

Table 7.1: Summary of Key Challenges and Solutions

Summary

This section outlined the critical challenges encountered during the implementation of the **Suricata + Elastic SIEM** setup and provided detailed solutions to address them. By resolving these issues, we ensured a stable, efficient, and scalable real-time threat detection system capable of providing actionable insights through Kibana dashboards.

Chapter 8

Conclusion & Future Work

8.1 Conclusion

The **Real-Time Network Threat Detection with Suricata IPS and Elastic SIEM** project successfully demonstrated the detection, logging, and visualization of network-based threats using open-source tools. By integrating Suricata for intrusion detection and Elasticsearch-Kibana for log analysis, we achieved a functional SIEM solution capable of monitoring and responding to security events.

Key Achievements

Intrusion Detection with Suricata: Successfully configured Suricata to monitor network traffic and generate alerts.

Log Aggregation & Processing: Properly forwarded Suricata EVE JSON logs to Elasticsearch for indexing.

Threat Visualization in Kibana: Built interactive dashboards highlighting top offenders, attack trends, and threat types.

Network Simulation & Testing: Simulated real-world cyber threats using *Nmap*, brute-force attacks, and SQL injection to validate detection accuracy.

Performance Optimization: Resolved high CPU usage issues, optimized rule sets, and improved alert precision.

The project showcased the effectiveness of open-source network security monitoring, making it a cost-efficient and scalable solution for cybersecurity operations.

8.2 Future Work & Improvements

Although we successfully met our primary objectives, several areas for enhancement exist:

1. Implement Automated Response Mechanisms: Currently, Suricata generates alerts but does not automatically mitigate detected threats. Future improvements could integrate:

- SOAR (Security Orchestration, Automation, and Response) tools.

- Firewall automation to block malicious IPs dynamically.
- Custom Python scripts for automated alert-based responses.

2. Integration with Additional Data Sources: Expanding the project to include:

- Windows Event Logs (via Winlogbeat) for endpoint monitoring.
- Zeek/Bro IDS for deeper network analysis.
- MITRE ATT&CK framework mapping for threat classification.

3. Machine Learning-Based Anomaly Detection: Future iterations could implement ML models within Elasticsearch or SIEM to detect:

- Zero-day attacks.
- Insider threats.
- Behavioral anomalies in network traffic.

4. Multi-Node ELK Stack Deployment: Deploying Elasticsearch, Logstash, and Kibana in a distributed cluster would enhance:

- Scalability – Handling larger datasets.
- Redundancy – Preventing data loss.
- Performance – Faster log processing.

5. Real-World Deployment in Enterprise Environments:

- Cloud-based deployment (AWS, Azure) for global threat monitoring.
- Integration with SOC (Security Operations Center) for 24/7 monitoring.
- Compliance alignment (GDPR, NIST, ISO 27001) for regulatory adherence.

Final Thoughts

This project provided hands-on experience in setting up a real-time intrusion detection and SIEM system. By working with Suricata, Elasticsearch, and Kibana, we built a functional security monitoring framework capable of detecting and analyzing threats efficiently.

With continuous refinement, this system can evolve into a fully automated, AI-driven security solution, bridging the gap between manual threat detection and proactive cybersecurity defense.