AI Driven Anomaly Detection in Kubernetes Cluster

Project Team

Muhammad Hani	21I-2595
Muhammad Usman Azam	21I-0653
Daniyal Ahmed	21I-2493

Session 2021-2025

Supervised by

Dr. Muhammad Arshad Islam

Department of Computer Science

National University of Computer and Emerging Sciences
Islamabad, Pakistan

September, 2024

Contents

List of Figures

Chapter 1

Introduction

Cyberattacks are becoming more frequent and sophisticated, which puts enterprises at serious risk of suffering major financial losses. An estimated 12.5 billion dollars were lost to cyberattacks in 2023[4]. Particularly in situations like Kubernetes clusters, where attack vectors can be broad and constantly growing, traditional methods cannot keep up with growing nature of cyber attacks. As attackers are evolving their methods to exploit vulnerabilities in the system, organization find it hard to secure their systems.

In particular, within the wider spectrum of cyber dangers, Distributed Denial of Service (DDoS)[2] attacks are a significant worry. The goal of DDoS attacks is to severely interrupt services by flooding systems with traffic. These assaults have the potential to cause significant financial harm; DDoS attack losses are estimated to cost companies millions of dollars worldwide each year. Attacker are now using botnets for DDos attacks which make it more of a challenge to mitigate it using traditional methods. Equally worrisome is the operational impact: DDoS attacks have the potential to completely destroy vital systems, resulting in protracted outages and a decline in customer confidence. Since businesses are depending more and more on digital services, it is essential to be able to identify and stop DDoS attacks in order to protect critical data and ensure company continuity.

Our project aims to address these challenges by developing an AI-based anomaly detection system tailored specifically for Kubernetes clusters. The system uses deep learning models, such as Transformer-based models [5] or Long Short Term Memory (LSTM) [6], to monitor network traffic and detect abnormal patterns that may indicate security risks. Because these are used to detect temporal trends in data, they are ideal for detecting long-term threats. Unlike traditional approaches, our AI-driven solution are able to recognize both known and unknown attack vectors by learning continuously, resulting in a more flexible and comprehensive protection mechanism. The system will adapt to evolving attacks by learning continuously from new attack patterns.

1.1 Problem Statement

Kubernetes[3], a widely adopted container orchestration platform, is an important component for deploying, managing, and scaling containerized applications with Docker. However, because of its widely adopted use, attackers target it for vulnerabilities. Many organizations find it difficult to efficiently monitor and analyze the massive amount of network traffic produced by Kubernetes clusters even with strong security measures. Modern cyber threats are sophisticated and can avoid detection by traditional detection methods, making it even more difficult to overcome. Kubernetes cluster is therefore still a prime target to a range of network anomalies, such as attempted intrusions, distributed denial-of-service (DDoS) attacks, and other malicious activities [1].

In addition, the increasing use of Kubernetes in different environments like cloud and standalone servers, has made its architecture more complex, thus increasing the range of attacks. The current anomaly detection tools of Kubernetes rely on static rules which makes attackers easier to bypass. This shows the need for a more intelligent solution which can detect patterns and behaviours.

1.2 Scope

- **In-Time Anomaly Detection:** KubeSecure will be monitoring and analysis the kubernetes clusters network traffic continuously to detect any unusual pattern which can be a potential harm for kubernetes security, such as DDos attacks.

- **AI-Driven Analysis:** KubeSecure will be using Deep Learning model to provide an adaptive and intelligent anomaly detection.

- **Automated Threat Mitigation:** KubeSecure will provide automated responses after detecting an anomaly like redirecting the traffic to Honeypots or rate limiting.

- **Static Rules Writing:** KubeSecure will be writing static network policy rules for cilium so that simple DDos attacks can be mitigated through predefined rules.

- **Testing and Validation:** KubeSecure will be testing our system by simulating various different attack scenarios using different computer system to ensure the effectiveness in real-world environments.

1.3 Modules

The suggested project is divided into several modules, each of which focuses on a different set of features that enhance the project's overall usability.

1.3.1 Model Selection and Training Module

In this module, we will setup Kubernetes cluster along with monitoring tools. Data preprocessing and Model training will begin.

1. Setup Kubernetes cluster with Grafana and Prometheus
2. Data Preprocessing and Deep Learning Model training

1.3.2 Network Setting Module

In this module we will setup Cilium and write Network policies. Training and testing of the selected model will be continued.

1. Setup Cilium
2. Network Policies in Cilium
3. Model Testing

1.3.3 Traffic Security and YAML Auditing Module

In this module we will integrate model in Kubernetes cluster to test it on streaming network traffic. We will also setup honeypots for traffic redirection and will make our yaml file misconfiguration tool.

1. Integrate model in Kubernetes
2. Setup TPOT Honeypot
3. Frontend website for yaml file misconfiguration
4. Backend logic for scanning yaml file

1.3.4 Application Development Module

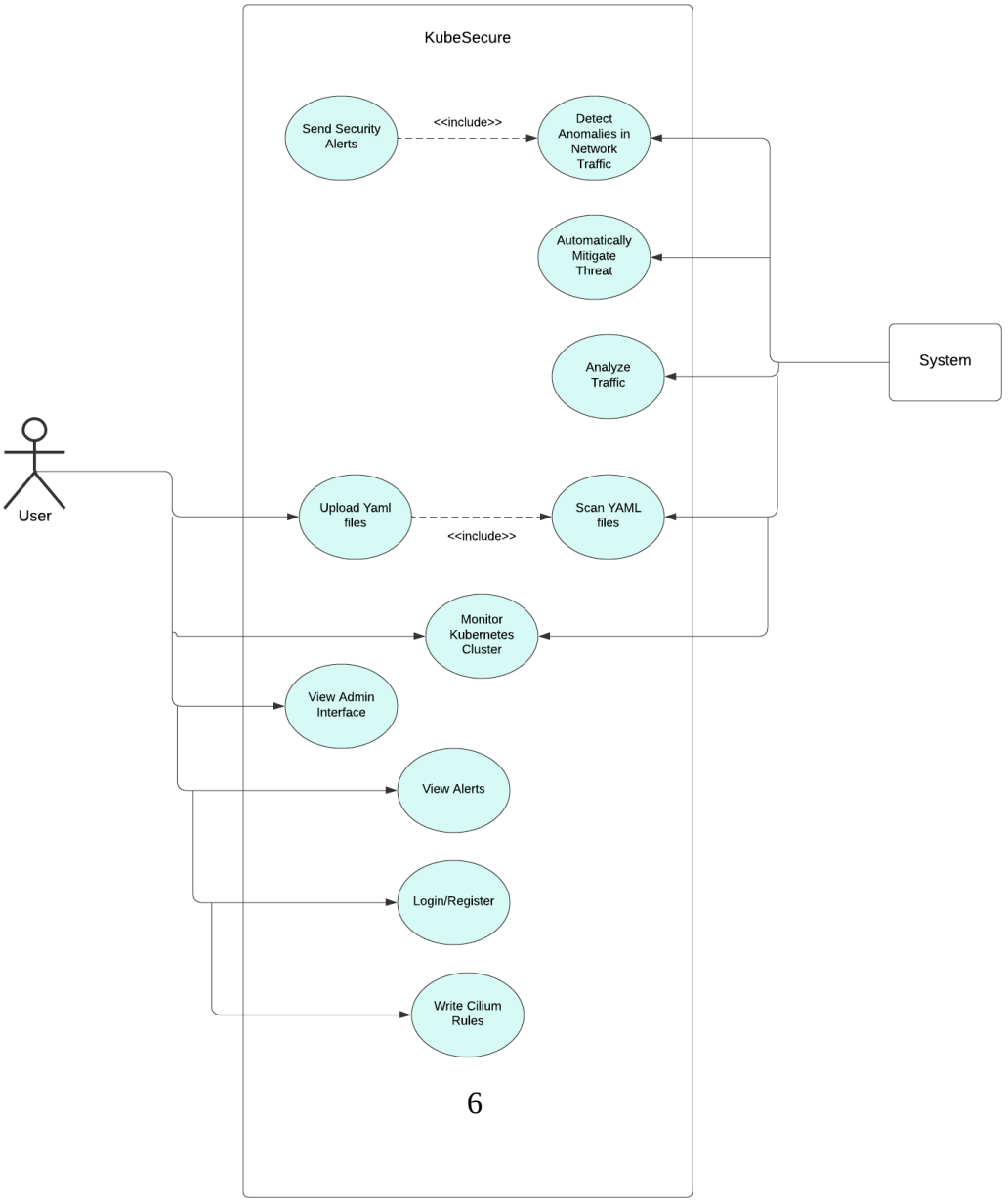
In this module we will develop android application for recieving alerts and will deploy our app on cloud.

1. Flutter App development
2. Deployment on Cloud

Chapter 2

Project Requirements

2.1 Use Case Diagram



2.2 Use Cases

2.2.1 Monitoring Kubernetes Cluster

Use Case	Monitoring Kubernetes Cluster	
Scope	KubeSecure	
Level	High Level	
Primary Actor	Security Team	
Stakeholders	Security Team, DevOps Engineers	
Precondition	The Kubernetes cluster is up and running, and Grafana and Prometheus are installed.	
Postcondition	Performance and anomaly measurements can be viewed and responded to.	
Main Success Scenario	**User Actions**	**System Response**
	1. User navigates to Grafana/Prometheus. 4. User chooses their desired dashboard to monitor the cluster. 6. User monitors cluster through the dashboard.	2. System opens Grafana/Prometheus interface. 3. System asks user to choose a dashboard to show. 5. System opens the selected dashboard.

2.2.2 Automatically Detect Anomalies in Network Traffic

Use Case	Automatically Detect Anomalies in Network Traffic	
Scope	KubeSecure	
Level	High Level	
Primary Actor	Security Team	
Stakeholders	Security Team, DevOps Engineers	
Precondition	The Kubernetes cluster is up and running, and the anomaly detection system is configured.	
Postcondition	Anomalies are detected and reported to the security team.	
Main Success Scenario	**User Actions**	**System Response**
	1. User initiates the monitoring process. 2. User reviews the detected anomalies.	3. System begins monitoring network traffic. 4. System detects anomalies based on configured parameters. 5. System alerts the user about the detected anomalies.

2.2.3 Automated Threat Mitigation

Use Case	Automated Threat Mitigation	
Scope	KubeSecure	
Level	High Level	
Primary Actor	Security Team	
Stakeholders	Security Team, DevOps Engineers	
Precondition	The anomaly detection system must be active and configured to mitigate threats.	
Postcondition	Threats are mitigated automatically without manual intervention.	
Main Success Scenario	**User Actions**	**System Response**
	1. User configures mitigation policies. 2. User enables automated threat mitigation. 6. User reviews logs of past mitigations.	3. System applies configured policies automatically. 4. System mitigates detected threats in in-time. 5. System logs all mitigation actions for review.

2.2.4 Scanning YAML Files for Misconfigurations

Use Case	Scan YAML Files	
Scope	KubeSecure	
Level	High Level	
Primary Actor	DevOps Engineers	
Stakeholders	Security Team, DevOps Engineers	
Precondition	User has access to the YAML files for scanning.	
Postcondition	Vulnerabilities and misconfigurations are reported.	
Main Success Scenario	**User Actions**	**System Response**
	1. User uploads YAML files for scanning. 2. User initiates the scanning process. 5. User reviews the scanning results.	3. System scans YAML files for vulnerabilities. 4. System generates a report on findings. 6. System alerts the user of critical vulnerabilities.

2.2.5 Writing Static Rules

Use Case	Write Cilium Rules	
Scope	KubeSecure	
Level	High Level	
Primary Actor	Admin User	
Stakeholders	Security Team, Network Engineers	
Precondition	User must be authenticated and authorized to write rules.	
Postcondition	Cilium rules are written and applied to the network policies.	
Main Success Scenario	**User Actions**	**System Response**
	1. User navigates to the Cilium rules section. 2. User writes or edits Cilium rules. 3. User saves the new or updated rules.	4. System validates the syntax of the rules. 5. System applies the rules to the network policies. 6. System confirms the successful application of the rules.

2.2.6 Upload YAML File

Use Case	Upload YAML Files	
Scope	KubeSecure	
Level	High Level	
Primary Actor	DevOps Engineers	
Stakeholders	Security Team, DevOps Engineers	
Precondition	User is authenticated and has permissions to upload YAML files.	
Postcondition	YAML files are successfully uploaded and ready for scanning.	
Main Success Scenario	**User Actions**	**System Response**
	1. User navigates to the upload section. 2. User selects YAML files to upload. 3. User confirms the upload.	4. System accepts the YAML files. 5. System confirms successful upload. 6. System prepares the files for further processing.

2.2.7 View Security Alerts

Use Case	View Security Alerts	
Scope	KubeSecure	
Level	High Level	
Primary Actor	Security Team	
Stakeholders	Security Team, Incident Response Team	
Precondition	The alert system must be active and logging alerts.	
Postcondition	Security alerts are displayed for review.	
Main Success Scenario	**User Actions**	**System Response**
	1. User navigates to the alerts section. 4. User reviews the list of security alerts.	2. System displays the alerts highlightng critical alerts for immediate attention. 5. System logs user activity for audit.

2.2.8 Traffic Analysis

Use Case	Send Security Alert	
Scope	KubeSecure	
Level	High Level	
Primary Actor	Security Team	
Stakeholders	Security Team, Incident Response Team	
Precondition	An alert has been generated and requires notification.	
Postcondition	The security alert is sent to designated recipients.	
Main Success Scenario	**User Actions**	**System Response**
	4. User navigates to Honeypot Dashboard. 5. User reviews Analysis Report.	1. System send the Traffic to Honeypot for Analysis. 2. Honeypot perform Analysis on the Traffic. 3. Honeypot generates Report on Analysis.

2.2.9 Access Admin Interface

Use Case	Access Admin Interface	
Scope	KubeSecure	
Level	High Level	
Primary Actor	Admin User	
Stakeholders	Security Team, Admins	
Precondition	User must be registered to access admin interface.	
Postcondition	Admin interface is displayed with all available options.	
Main Success Scenario	**User Actions**	**System Response**
	1. User enters their credentials. 4. User navigates through the available options. 5. User accesses specific functionalities like managing rules and monitoring dashboards.	2. System verifies user credentials and permissions. 3. System displays the admin interface with functionalities. 6. System allows access to the selected functionalities.

2.3 Functional Requirements

2.3.1 Anomaly Detection

2.3.1.1 Monitoring Kubernetes Cluster Using Prometheus and Grafana

- Network traffic metric will be collected from cilium
- Cluster and Application metrics will be collected from prometheus
- Grafana dashboards will show the metric collected from prometheus and cilium

2.3.1.2 Network Traffic Management

- The system will get streaming network traffic to a deployed application in Kubernetes Cluster.
- The incoming traffic will be duplicated to the Deep Learning model pod.
- Analysis will begin whenever the pod will get traffic.

2.3.1.3 Traffic Analysis through Deep Learning model

- Trained Deep Learning model pod will be deployed.
- It will get a traffic batch for analysis.
- When it will detect the traffic batch as malicious, it will issue an alert.

2.3.1.4 Automated Threat Mitigation

- When an alert will be issued by the model, prometheus will get the alert.
- Prometheus will call a web hook to run a python script to rate limit the incoming traffic.
- The python script will dynamically update cilium rules to rate limit or an EBPF script will be called.
- Alerts will be send to our Android application, slack and emails.

2.3.2 Network Policies

2.3.2.1 Mitigation Using Network Policies

- The system will detect simple DoS attacks, such as TCP Flooding patterns by keeping track of violations of defined static rules.
- Upong recognizing a violation of defined rules, it will rate limit the traffic coming to the cluster

2.3.3 Honeypot for Traffic Analysis

2.3.3.1 Traffic Identification

- According to its specific criteria of suspicious packet loads, aberrant traffic dynamics, and common attack origins, the framework will highlight harmful traffic amassings.

2.3.3.2 Traffic Forwarding

- When it spots malicious traffic, the system will pass along that batch for thorough investigation in the TPOT Honeypot.

2.3.3.3 Traffic Labeling

- Prior to sending the threatening traffic directed at TPOT Honeypot, the system will feature metadata indicating the source IP, the time of detection.

2.3.3.4 Detailed Analysis

- The TPOT Honeypot will generate enhanced analysis reports that contribute information on the behaviors of traffic, attack trends, and practical exploit vectors, all as part of improving mitigation and security practices.

2.3.4 Yaml File Checker

2.3.4.1 File Scanning

The system shall scan uploaded Kubernetes .yaml files for common configuration errors, including but not limited to:

- The findings indicate that a number of configurations in RBAC (Role-Based Access Control) are either left unenabled or improperly configured.
- Resource allocations can suffer from a problem that makes them either inadequate or excessive.
- Errors result from poor security statements regarding pods, inadequate configuration of network settings, or the configuration of volumes.
- Conditioned to obsolete API versions or choosing setups that do not provide security.

2.3.4.2 Error Identification

- The system will separate out misconfigurations found in .yaml files by evaluating their content against industry standards and security policies.

2.3.4.3 Send Notification

- The platform will issue emails, dashboard notifications, or system signals over the issues it finds.

2.3.5 Web Application

2.3.5.1 File Upload Functionality

- This platform will support web submissions of Kubernetes .yaml files for analysis.

2.3.5.2 Misconfiguration Detection

- Upon the submission of a .yaml file, the system will rapidly start its processing and will comb through for misconfigurations.

2.3.5.3 Admin Panel

- Admin can view and setup cilium rules
- Drop down menu for selecting values for network policies

2.4 Non Functional Requirements

User-Friendly: The system will be optimized whereby one can learn it easily and operate without much difficulty.

In-Time Data Processing: In order to manage all these flows the system will be able to process in time data with minimal delay.

Scalability: KubeSecure is optimistically designed to be scalable, and this means that it can scale up to support the increased network traffic and monitoring needs or requirements.

Reliability: The system shall be highly reliable, with redundancy and failover mechanisms in place to ensure continuous operation even during high traffic or under attack.

Interoperability: The system shall be compatible with various existing tools and platforms used in Kubernetes environments, such as different CI/CD pipelines, monitoring systems, and security tools, allowing for seamless integration.

Security: The traffic analysis deep learning pod is to be isolated of external access so that changes (may) not be possible by unauthorized actors. Under high level DDoS attack the pod will remain secure and stable while continuously monitoring and detecting anomalies without compromise.

2.5 Domain Model

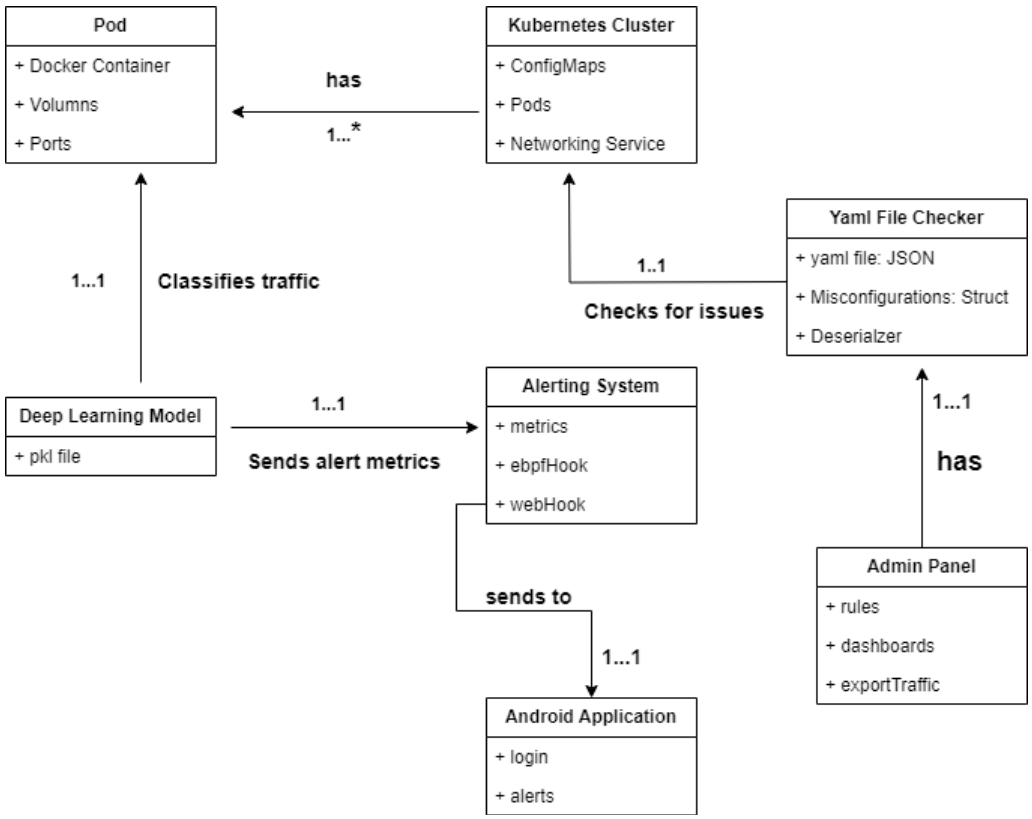


Figure 2.2: Domain Model

Chapter 3

System Overview

3.1 Architectural Design

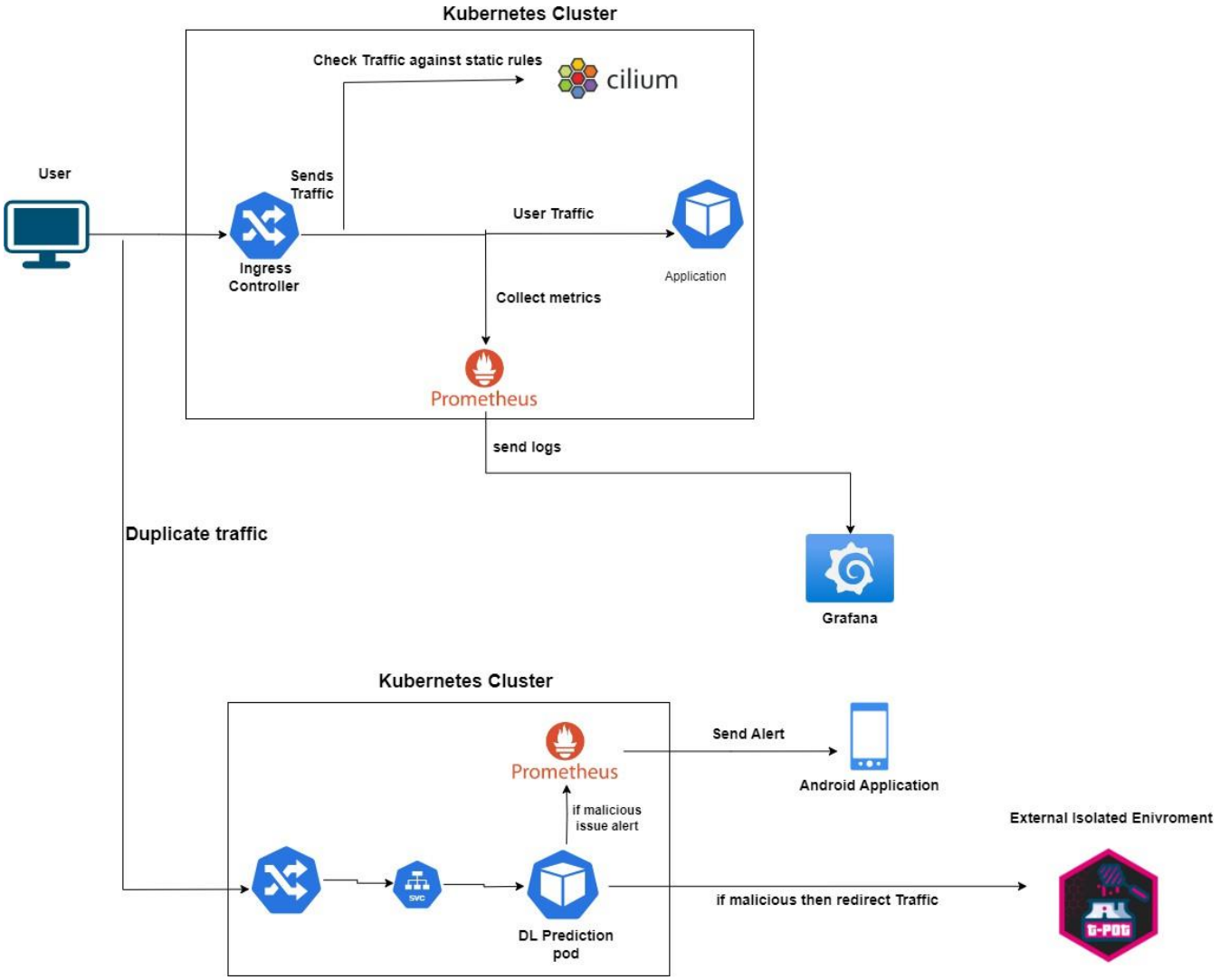


Figure 3.1: Architecture Diagram

3.1.1 Process flow

- User sends traffic to the application pod.
- The traffic goes to Kubernetes Ingress controller and it forwards traffic to application pod.
- It duplicates the traffic to Cilium and to Kubernetes Cluster which contains Deep Learning model for prediction.
- Cilium will check the traffic against predefined rules.
- If Cilium or Deep Learing model reports any threat. It will send alerts to our android application and the traffic batch will be given to honepot for traffic analysis.
- Prometheus will actively collect logs and show to graphs in Grafana.

3.2 Design Models

3.2.1 Activity Diagram

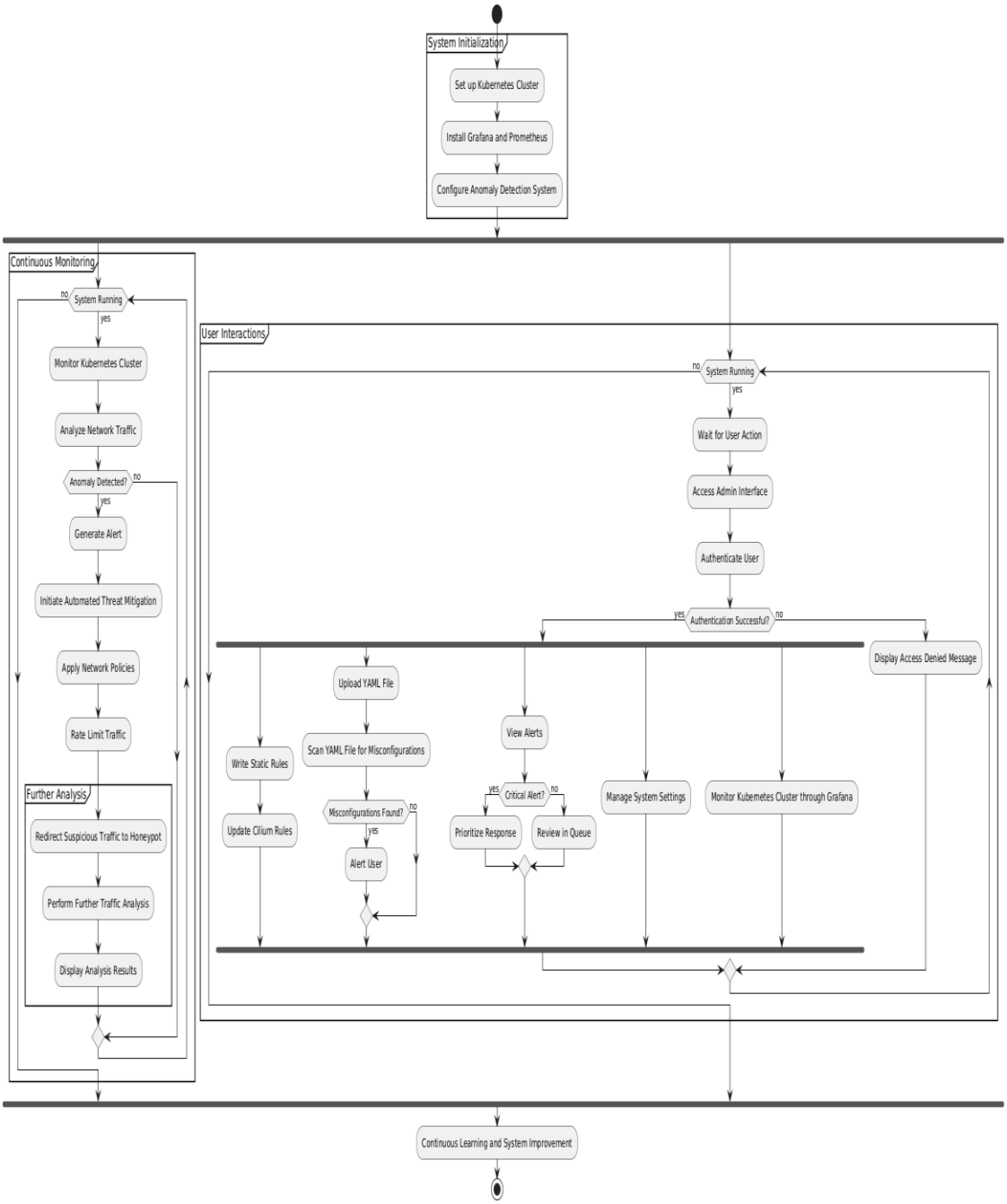


Figure 3.2: Activity Diagram

3.2.2 System Sequence Diagram

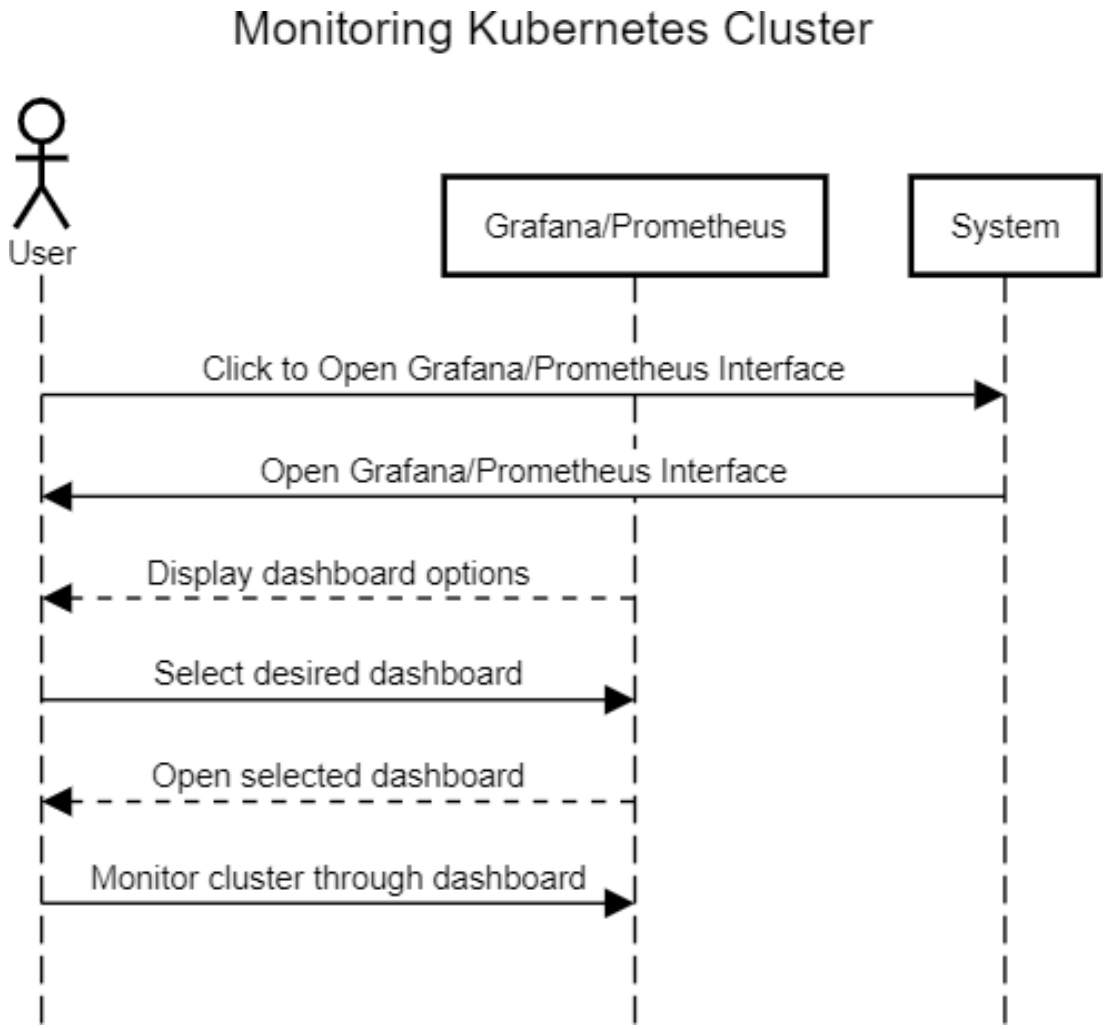


Figure 3.3: Monitoring Kubernetes Cluster

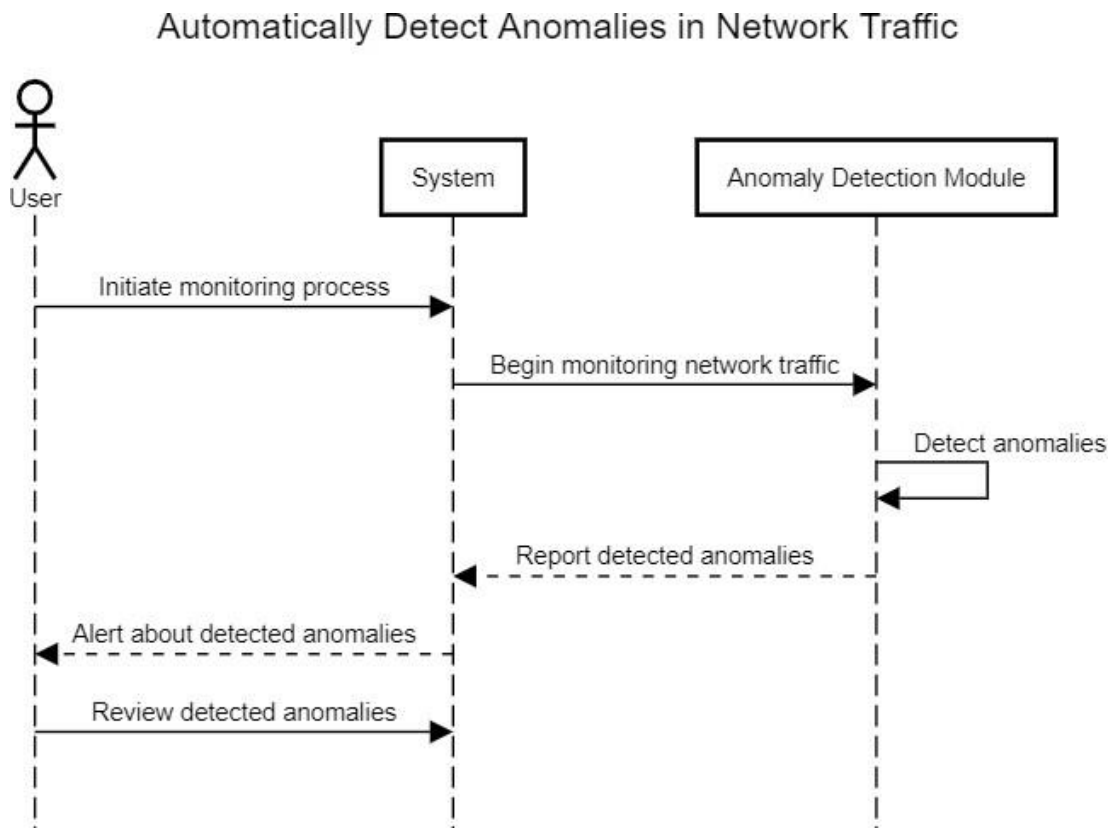


Figure 3.4: Automatically Detect Anomalies in Network Traffic

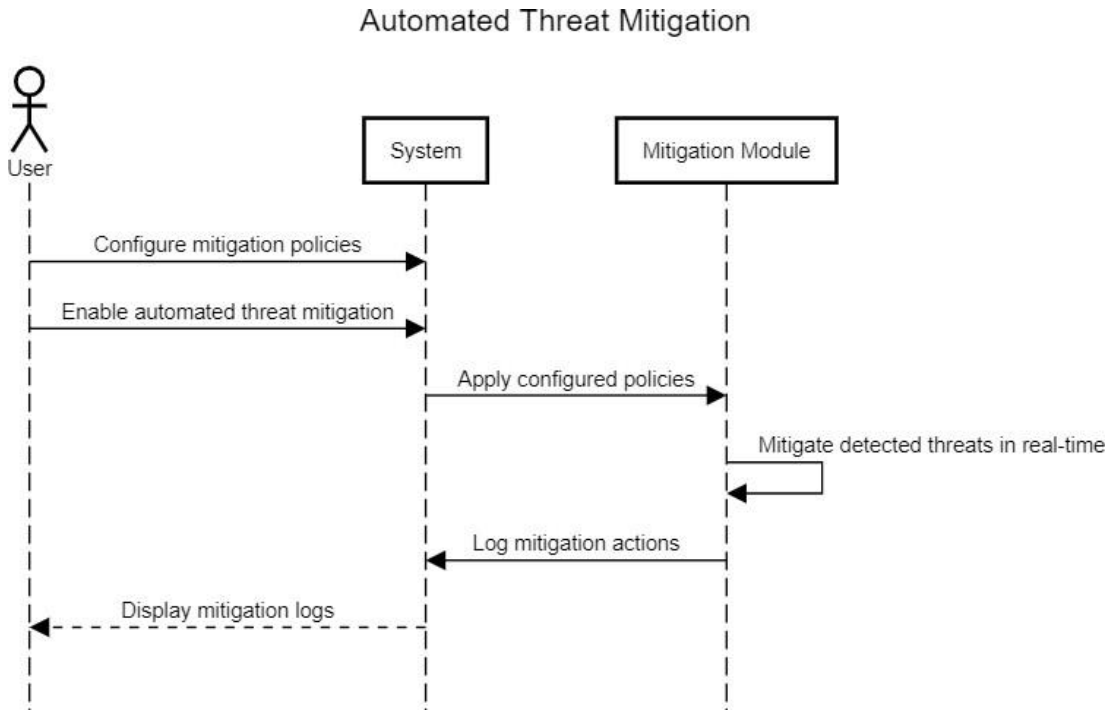


Figure 3.5: Automated Threat Mitigation

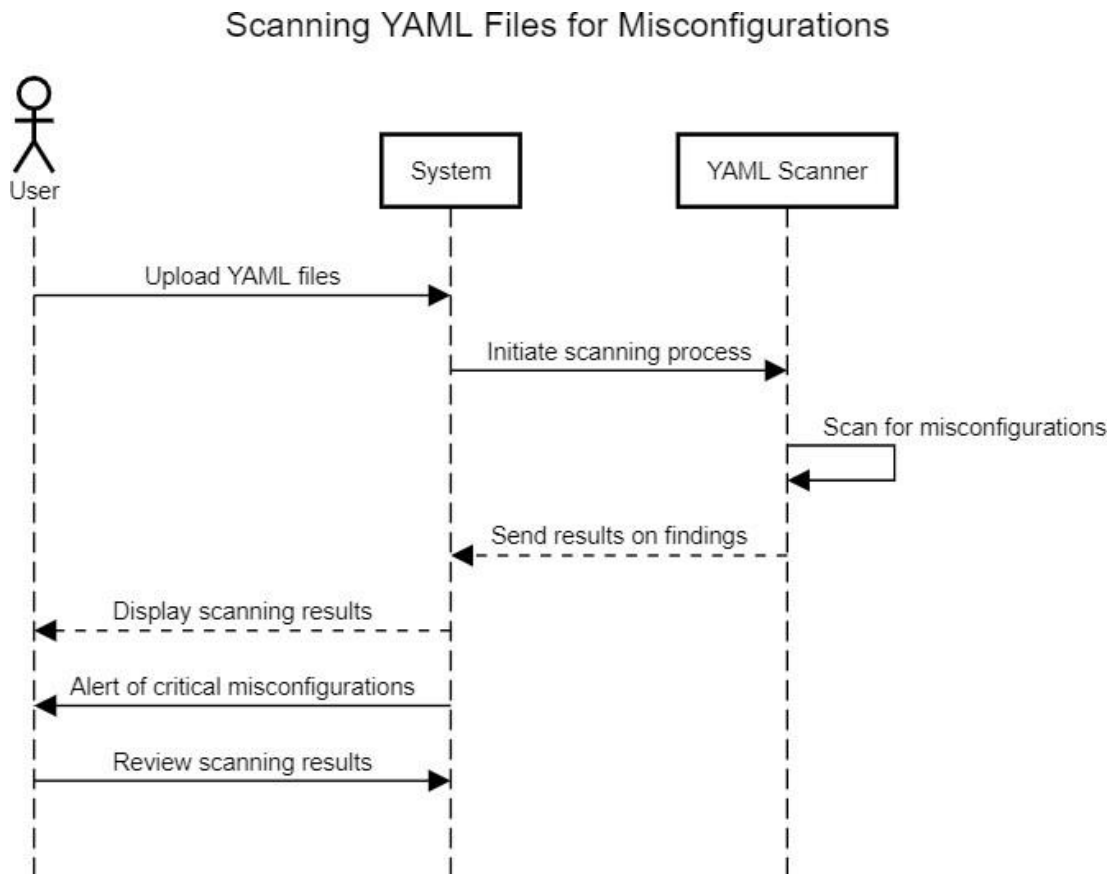


Figure 3.6: Scanning Yaml Files for Misconfigurations

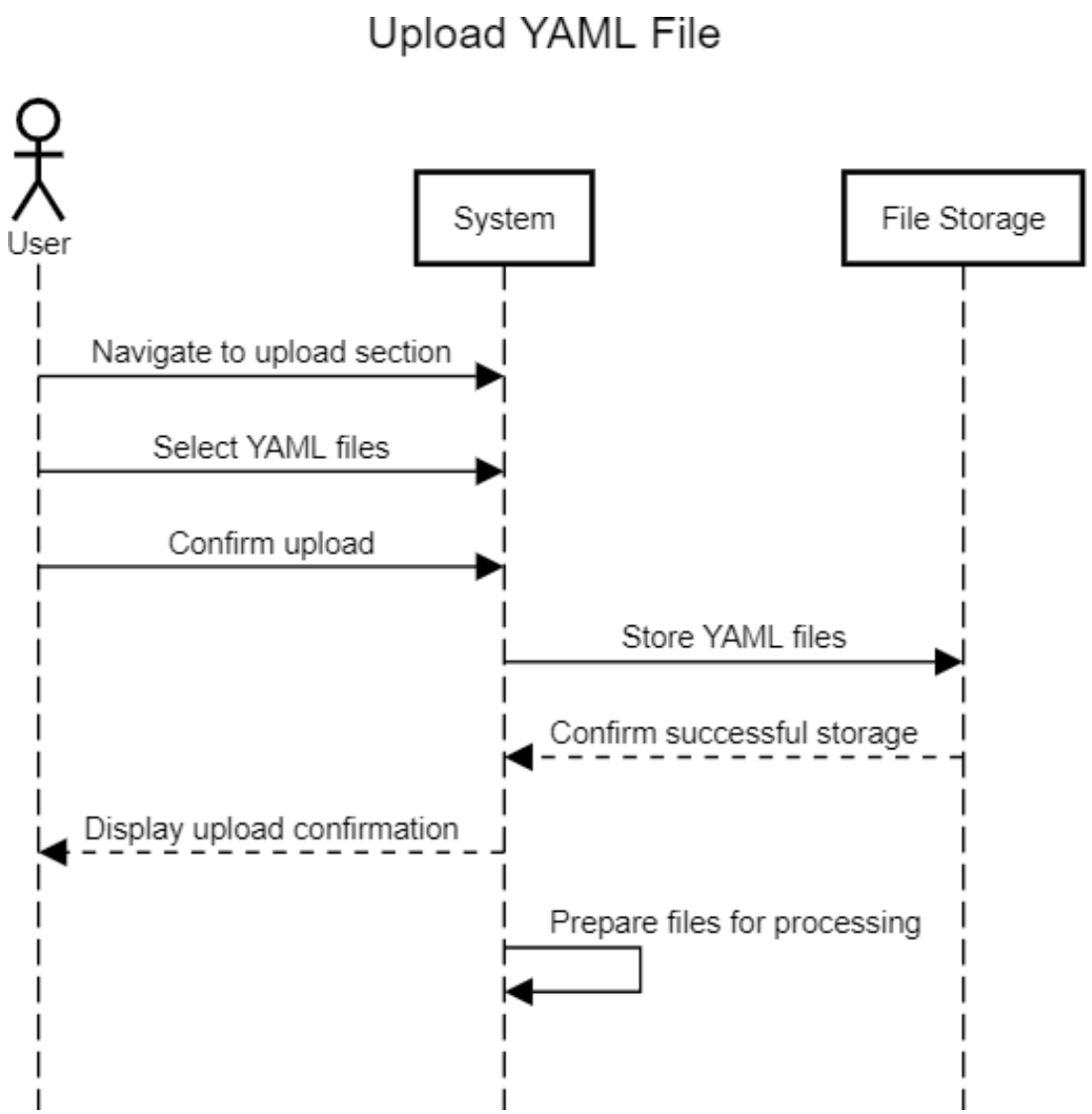


Figure 3.7: Upload Yaml File

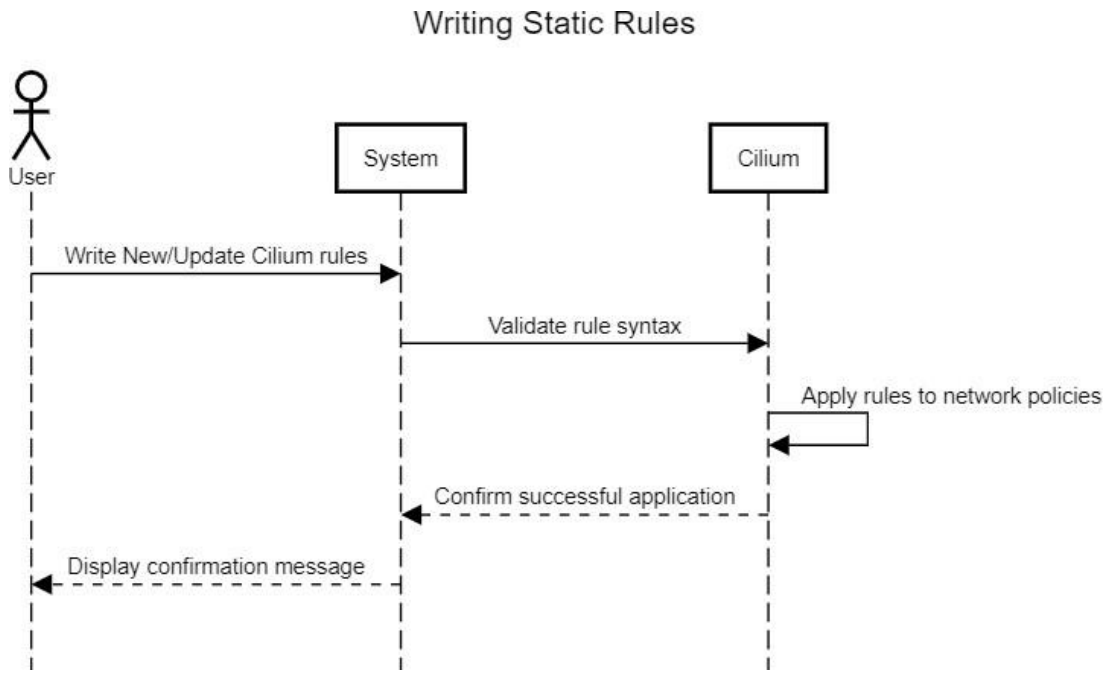


Figure 3.8: Writing Static Rules

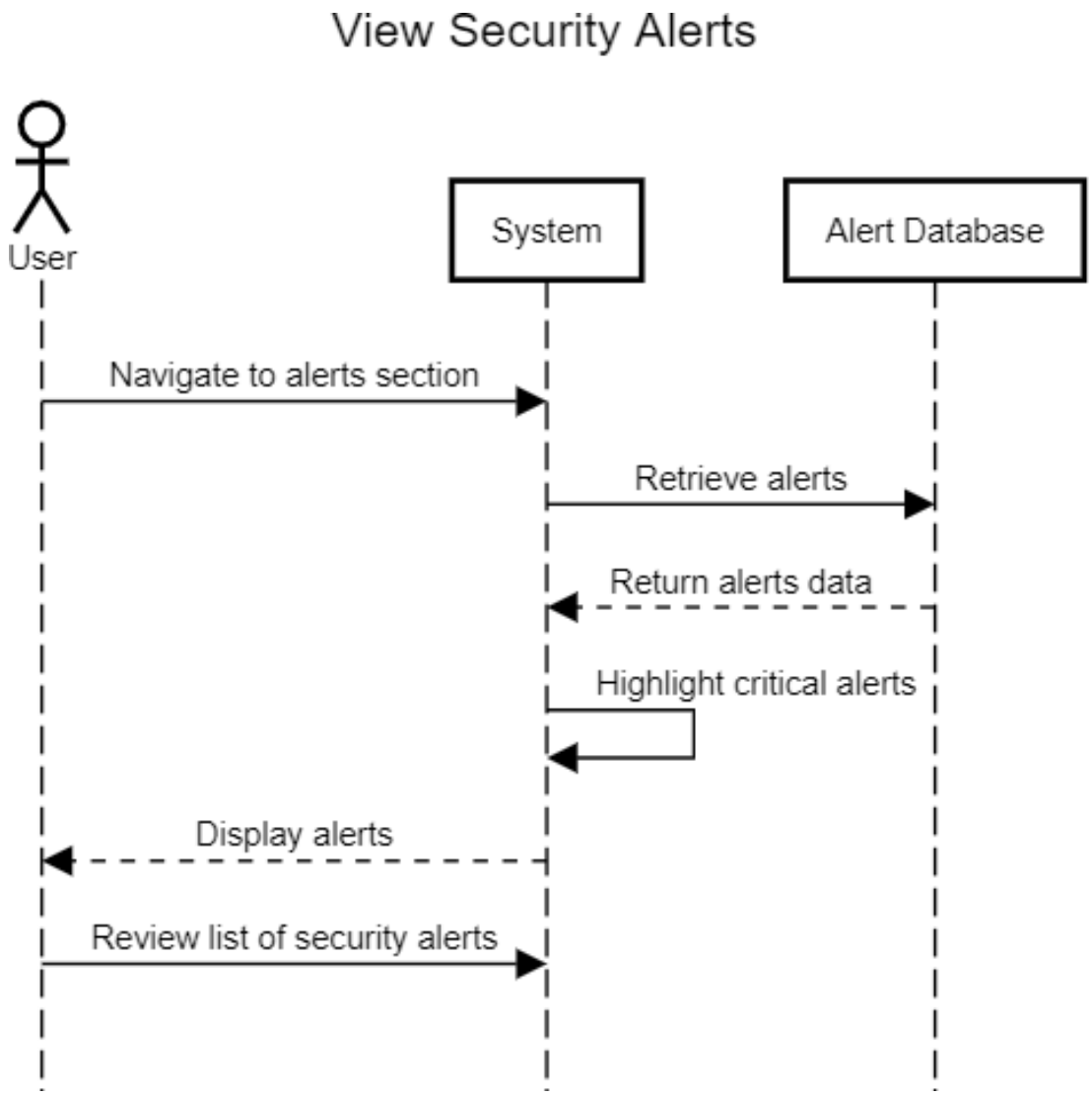


Figure 3.9: View Security Alert

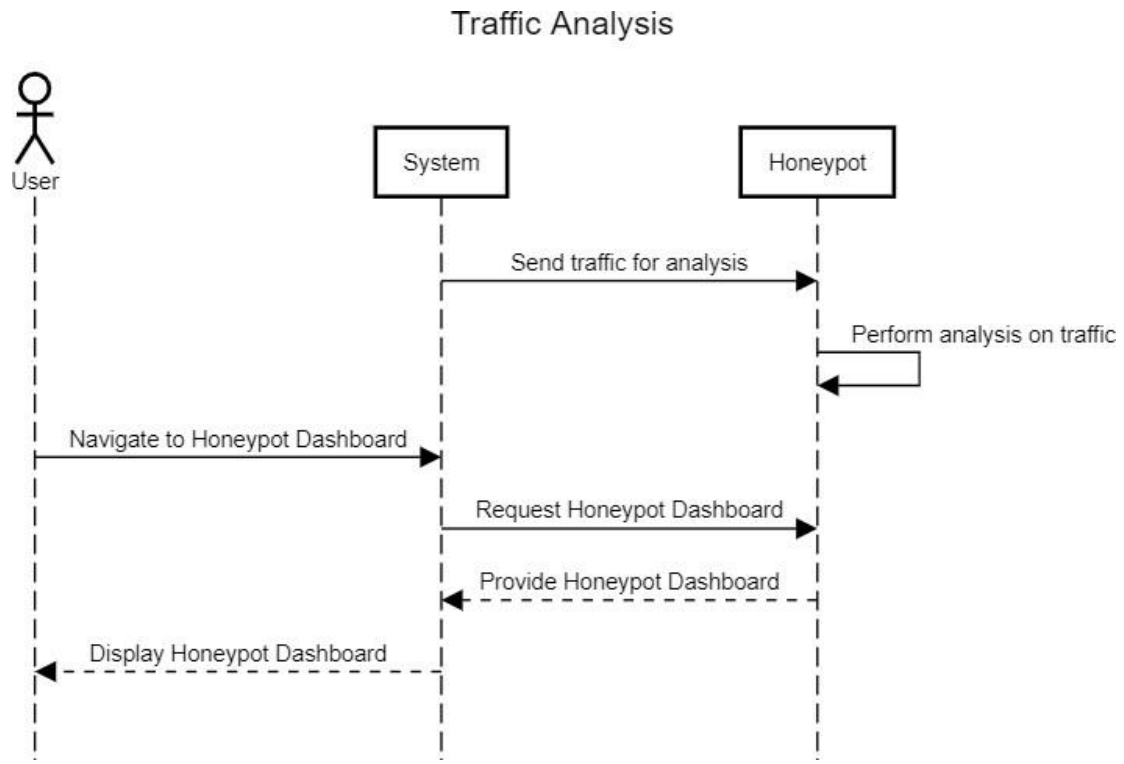


Figure 3.10: Traffic Analysis

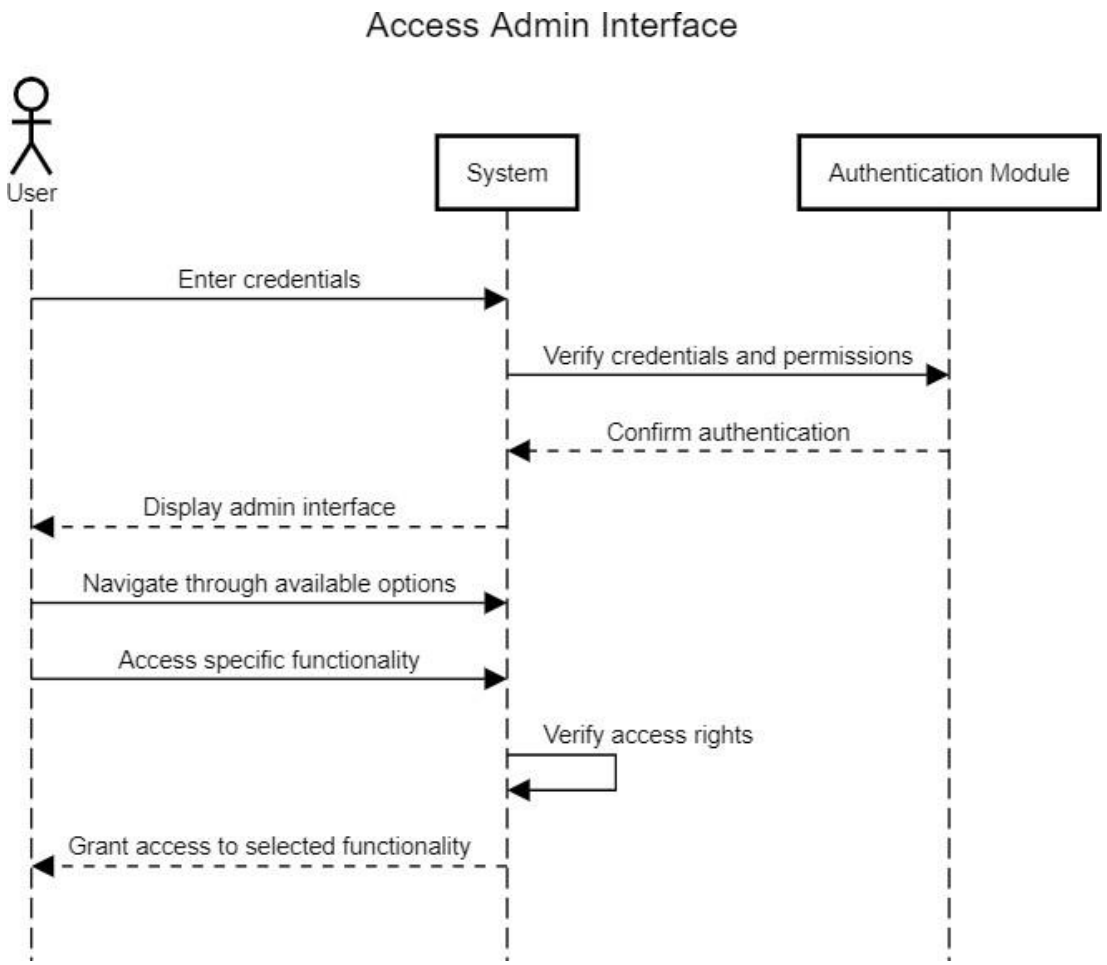


Figure 3.11: Access Admin Interface

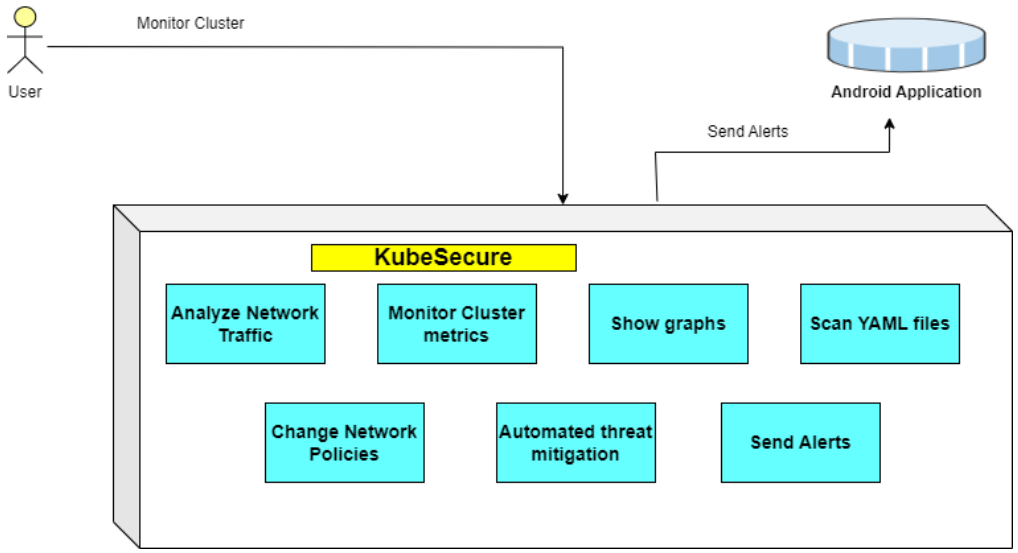


Figure 3.12: Data Flow Level 0

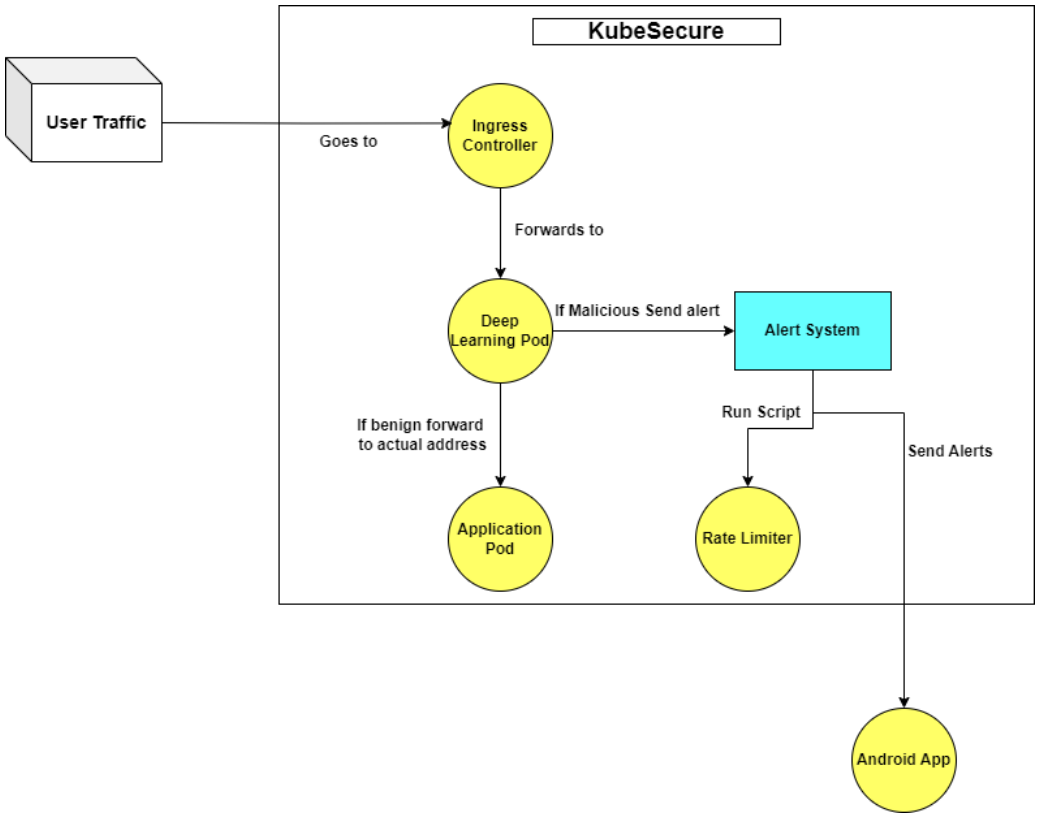


Figure 3.13: Data Flow Level 1

Bibliography

[1] Cyber attacks kubernetes. https://purplesec.us/breach-report/kubernetes-clusters-hacked/. Accessed: 2024-09-07.

[2] Distributed denial of service attack. https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/. Accessed: 2024-09-07.

[3] Kubernetes. https://kubernetes.io/docs/home/. Accessed: 2024-09-04.

[4] Loss due to cyber attacks. https://www.statista.com/chart/32341/worldwide-reported-losses-connected-to-cybercrime/. Accessed: 2024-09-04.

[5] Josue Genaro Almaraz-Rivera. An anomaly-based detection system for monitoring kubernetes infrastructures. *IEEE Latin America Transactions*, 21(3):457–465, 2023.

[6] Tae-Young Kim and Sung-Bae Cho. Web traffic anomaly detection using c-lstm neural networks. *Expert Systems with Applications*, 106:66–76, 2018.