Project Report

On

AIWebSec : AI Enhanced Web Security Firewall

Submitted to D Y Patil International University, Akurdi, Pune in partial fulfilment of full-time degree

Master of Computer Applications

Submitted By:

Name: **Pranil Choudhari** **Rohan Pandit** **Chetan Chaudhari**

PRN: **20220804036** **20220804004** **20220804027**

Under the Guidance of

Mrs. Hetal Thaker

School of Computer Science, Engineering and Applications

D Y Patil International University, Akurdi,Pune, INDIA, 411044

[Session 2023-24]

This report on AIWebSec: AI Enhanced Web Security Firewall is submitted for the partial fulfillment of project,which is part of the Second Year Master of Computer Applications curriculum, under my supervision and guidance.

Mrs. Hetal Thaker
(DYPIU Guide)

Mrs. Vaishali Kumar
(Project Coordinator)

Dr.Maheshwari Biradar
(HOD BCA & MCA)

Dr. Bahubali Shiragapur
Director
School of Computer Science Engineering & Applications
D Y Patil International University, Akurdi
Pune, 411044, Maharashtra, INDIA

DECLARATION

We, hereby declare that the following Project which is being presented in the Project entitled as **AIWebSec : AI Enhanced Web Security Firewall** is an authentic documentation of our own original work to the best of our knowledge. The following Project and its report in part or whole, has not been presented or submitted by us for any purpose in any other institute or organization. Any contribution made to our work, with whom we have worked at D Y Patil International University, Akurdi, Pune, is explicitly acknowledged in the report.

Name: Pranil choudhari
PRN No: 20220804036 Signature :

Name: Rohan pandit
PRN No: 20220804004 Signature :

Name: Chetan chaudhari
PRN No: 20220804027 Signature :

ACKNOWLEDGEMENT

With due respect, we express our deep sense of gratitude to our respected guide Mrs. Hetal Thaker, for her valuable help and guidance. We are thankful for the encouragement that she has given us in completing this Project successfully.

It is imperative for us to mention the fact that the report of project could not have been accomplished without the periodic suggestions and advice of our project supervisor Mrs. Vaishali Kumar.

We are also grateful to our respected, Dr. Bahubali Shiragapur(Director), Dr. Maheshwari Biradar (HOD BCA & MCA) and Hon'ble Vice Chancellor, DYPIU, Akurdi, Prof. Prabhat Ranjan for permitting us to utilize all the necessary facilities of the college.

We are also thankful to all the other faculty, staff members and laboratory attendants of our department for their kind cooperation and help. Last but certainly not the least; we would like to express our deep appreciation towards our family members and batch mates for providing support and encouragement.

Name: Pranil Choudhari
PRN: 20220804036
Name: Rohan Pandit
PRN: 20220804004
Name: chetan chaudhari
PRN: 20220804027

Abstract

In the ever-evolving realm of cybersecurity, ensuring the safety of web applications against malicious intrusions is paramount. This project presents a groundbreaking approach to fortify web application security through the development of an AI-powered Web Application Firewall (WAF) coupled with an Intrusion Prevention System (IPS). The core objective of this system is to proactively identify and mitigate potential threats in real-time.This project introduces an advanced AI-powered Web Application Firewall (WAF) and Intrusion Prevention System (IPS). The process begins with Burp Suite log collection, followed by parsing into CSV format using Python. Two datasets, ”Good” (non-malicious) and ”Bad” (malicious) web links, are constructed from actual web traffic. The innovation lies in dynamic clustering, enabling automatic categorization of web links. This adaptive approach enhances intrusion detection by distinguishing emerging threats from benign traffic. This project offers a proactive solution to web application security, combining AI, real-world data, and dynamic analysis for improved threat detection and mitigation.

TABLE OF CONTENTS

List of Figures

1. INTRODUCTION

In an era dominated by digital connectivity, the security of web applications is paramount to safeguarding sensitive information and ensuring a seamless online experience for users. As cyber threats continue to evolve in complexity and sophistication, the need for robust web security measures becomes increasingly urgent. This project, titled ”AIWEBSEC: AI-Based Web Security Firewall,” addresses this imperative by introducing an innovative solution that leverages artificial intelligence to fortify web applications against malicious intrusions.

The foundation of AIWEBSEC lies in the meticulous creation of two distinct datasets: one comprising non-malicious web links and another harboring malicious counterparts. To curate the dataset of non-malicious web links, the Acunetix web vulnerability scanner professional tool was deployed, executing from the formidable Kali Linux environment. Simultaneously, logs were meticulously collected from Windows utilizing the powerful BurpSuite software, configured to channel traffic through the Kali machine.

Conversely, the dataset housing malicious web links was cultivated by employing Acunetix to generate nefarious payloads designed for SQL injection and Cross-Site attacks. The execution of these attacks was conscientiously restricted to a controlled and ethical environment, ensuring compliance and avoiding any illegal implications. Logs were captured similarly through the meticulous configuration of BurpSuite, this time with the Kali Linux machine acting as the proxy.

The subsequent phase of the project involved developing a custom Python code from scratch. This code seamlessly accessed and decoded the base64-encrypted logs, effectively parsing the contents of each request. Feature extraction was then executed, distinguishing the characteristics between benign and malicious web links. This information was organized into comprehensive CSV datasets for both categories.

The pinnacle of AIWEBSEC is the integration of machine learning models, meticulously trained on the curated datasets. This fortified defense mechanism is adept at discerning and deflecting incoming malicious links, thereby thwarting potential threats. Furthermore, the model is designed to detect features associated with zero-day attacks, allowing for swift and adaptive responses by analyzing the position of incoming requests within predefined clusters.

Through the amalgamation of cutting-edge technologies and ethical testing methodologies, AIWEBSEC stands as a testament to the commitment to advancing web security. In an ever-evolving cyber landscape, this project sets forth a proactive defense system, poised to safeguard web applications and user data against the relentless tide of cyber threats.

1.1. Background

Web application security stands at the forefront of digital concerns as our reliance on web-based services intensifies. The ubiquity of cyber threats, ranging from SQL injection to Cross-Site Scripting, has rendered traditional security measures insufficient in safeguarding against the evolving threat landscape. Recognizing this vulnerability, AIWEBSEC, an AI-based Web Security Firewall, seeks to pioneer a more intelligent and adaptive defense mechanism. The incorporation of Artificial Intelligence (AI) in cybersecurity represents a paradigm shift, leveraging intelligent automation, anomaly detection, and predictive analysis to fortify web applications. AIWEBSEC is motivated by the imperative to address both known vulnerabilities and emerging threats in real-time. This motivation underscores the limitations of traditional signature-based methods, particularly in countering zero-day attacks. The project adopts an ethical testing approach, utilizing tools like Acunetix and BurpSuite within controlled environments to generate datasets that faithfully represent the dichotomy between benign and malicious web links. A pivotal aspect of AIWEBSEC is the development of a custom Python code, demonstrating a commitment to tailored solutions. This code serves as the linchpin, bridging raw logs to structured datasets through meticulous feature extraction. By embracing the transformative potential of AI and adhering to ethical testing practices, AIWEBSEC aspires to contribute to a new era of intelligent, adaptive, and proactive web security measures.

1.2. Objectives

- Develop a Comprehensive Dataset
- Enhance Defense Against Known/unknown Vulnerabilities
- Enable Adaptive Defense Mechanisms
- Contribute to the Advancement of Web Security
- Implement adaptive mechanisms for zero-day attacks

1.3. Purpose

The fundamental purpose of the AIWEBSEC project is to pioneer a transformative approach to web application security, addressing the escalating challenges posed by a dynamic and evolving threat landscape. By combining innovative technologies, ethical testing methodologies, and advanced artificial intelligence, the project aims to fortify web applications against both known vulnerabilities and emerging threats. The development of a custom Python code for log analysis and feature extraction underscores the commitment to tailored solutions, allowing for a nuanced understanding of web traffic patterns. Ultimately, AIWEBSEC endeavors to contribute to the advancement of web security by providing a proactive and adaptive defense mechanism, fostering resilience against cyber threats and ensuring the integrity of online platforms in an interconnected digital era.

1.4. Scope

The future scope of AIWEBSEC extends into various avenues, offering a foundation for ongoing development and innovation in web application security. Potential areas for expansion and enhancement include:

1. Integration of Advanced Machine Learning Models: Future iterations of AIWEBSEC could explore the integration of even more advanced machine learning models. This involves continuous research and adoption of cutting-edge algorithms to improve the system's ability to detect and adapt to emerging threats.

2. Behavioral Analysis for Enhanced Anomaly Detection: Incorporating behavioral analysis into the anomaly detection process can elevate the system's capability to identify subtle deviations from normal web traffic patterns. This approach could enhance the accuracy of threat detection and reduce false positives.

3. Real-Time Threat Intelligence Integration: Enhance the system's threat intelligence by integrating real-time data feeds. This would enable AIWEBSEC to stay abreast of the latest cyber threats and adjust its defense mechanisms accordingly, providing a proactive defense against evolving attack vectors.

4. Scalability for Large-Scale Deployments: Future developments may focus on optimizing AIWEBSEC for large-scale deployments, ensuring scalability without compromising performance. This could involve refining algorithms, improving resource utilization, and implementing distributed computing techniques.

5. Enhanced User Interface and Reporting: Continuous refinement of the user interface can improve accessibility and provide more intuitive controls for users. Additionally, incorporating comprehensive reporting features would empower administrators with detailed insights into system activities and threat mitigations.

6. Automation of Security Policy Updates: Implementing automation for security policy updates could streamline the process of adapting to new threats. AIWEBSEC could autonomously update its rules and configurations based on the latest threat intelligence, reducing the response time to potential security risks.

7. Application in Cloud Environments: Adapting AIWEBSEC for deployment in cloud environments would cater to the evolving landscape of web applications. This could involve optimizing the system for compatibility with popular cloud platforms and leveraging cloud-native security features.

8. Collaboration with Threat Intelligence Platforms: Establishing integrations with external threat intelligence platforms would enhance the system's ability to access a broader spectrum of threat data. Collaboration with external sources can enrich the knowledge base and improve overall threat detection capabilities.

9. Continuous Research and Development: The field of cybersecurity is dynamic, with new threats emerging regularly. Therefore, continuous research and development are integral to the future scope of AIWEBSEC. Staying abreast of emerging technologies and threat vectors ensures the system remains at the forefront of web security.

10. Community Engagement and Open Source Contributions: Fostering community engagement and potentially transitioning AIWEBSEC into an open-source project could encourage collaboration. Contributions from the community could bring diverse perspectives, improvements, and extensions to the system.

1.5. Applicability

1. Quantum Computing Integration: Explore the integration of quantum-safe cryptographic algorithms to fortify AIWEBSEC against potential threats posed by quantum computing advancements.

2. Blockchain for Security Auditing: Implement blockchain technology for secure and transparent auditing of AIWEBSEC's activities, providing an immutable record of security events and enhancing accountability.

3. Edge AI for Real-Time Analysis: Leverage edge computing and AI for real-time analysis of web traffic, enabling faster threat detection and response by processing data closer to the source.

4. Explainable AI (XAI): Integrate Explainable AI techniques to enhance transparency and interpretability, allowing users to understand the decision-making processes of AIWEBSEC and build trust in the system.

5. Collaboration with Threat Intelligence Platforms: Collaborate with external threat intelligence platforms through API integrations, enabling AIWEBSEC to access up-to-date threat data and improve its ability to detect emerging threats.

6. Zero Trust Security Architecture: Implement a Zero Trust security architecture, ensuring that AIWEBSEC continuously verifies and validates every access attempt, even from within the network, to enhance overall security posture.

7. Extended Application to 5G Networks: Extend AIWEBSEC's application to secure web interfaces within 5G networks, addressing the unique security challenges posed by the increased connectivity and data transfer rates.

8. Biometric Authentication Integration: Integrate biometric authentication methods to enhance user verification and access control, adding an additional layer of security to web applications protected by AIWEBSEC.

9. Automated Security Policy Orchestration: Implement automated security policy orchestration to dynamically update security configurations based on real-time threat intelligence, ensuring adaptive defense mechanisms.

10. Deep Learning for Enhanced Feature Extraction: Explore the application of deep learning techniques for enhanced feature extraction from web logs, allowing for more nuanced understanding of complex patterns indicative of security threats.

2. PROJECT PLAN

2.1. Problem Statement

The current landscape of web security systems is marked by inherent challenges that compromise their effectiveness in combating sophisticated cyber threats. Existing approaches, predominantly reliant on static rule sets and signature-based methodologies, fall short in providing comprehensive protection against dynamic and evolving attack vectors. Notably, the incapacity to swiftly detect and mitigate zero-day attacks, coupled with high false positive rates and a lack of adaptability to the dynamic threat landscape, underscores the limitations of traditional security measures. Moreover, the absence of ethical testing practices raises concerns regarding the authenticity and legality of generated datasets. Inefficient log analysis and feature extraction processes further hinder the accurate discernment of meaningful patterns from web traffic. In response to these shortcomings, our project, AIWEBSEC, introduces a paradigm shift in web security. Leveraging advanced machine learning algorithms for dynamic threat adaptability, ethical testing methodologies, a custom Python code for efficient log analysis, and an adaptive defense mechanism, AIWEBSEC aims to overcome the challenges plaguing existing systems. By prioritizing ethicality, adaptability, and intelligence, AIWEBSEC stands as a pioneering solution poised to redefine web application security and provide a proactive defense against the ever-evolving cyber threat landscape.

2.2. Requirement Specification

- Processor 10th Gen Intel(R) Core(TM) i5-10500H @ 3.10GHz 3.11 GHz
- Installed RAM 8.00 GB (7.70 GB usable)
- System type 64-bit operating system, x64-based processor
- Python 3.9 installed

2.3. Time Line chart

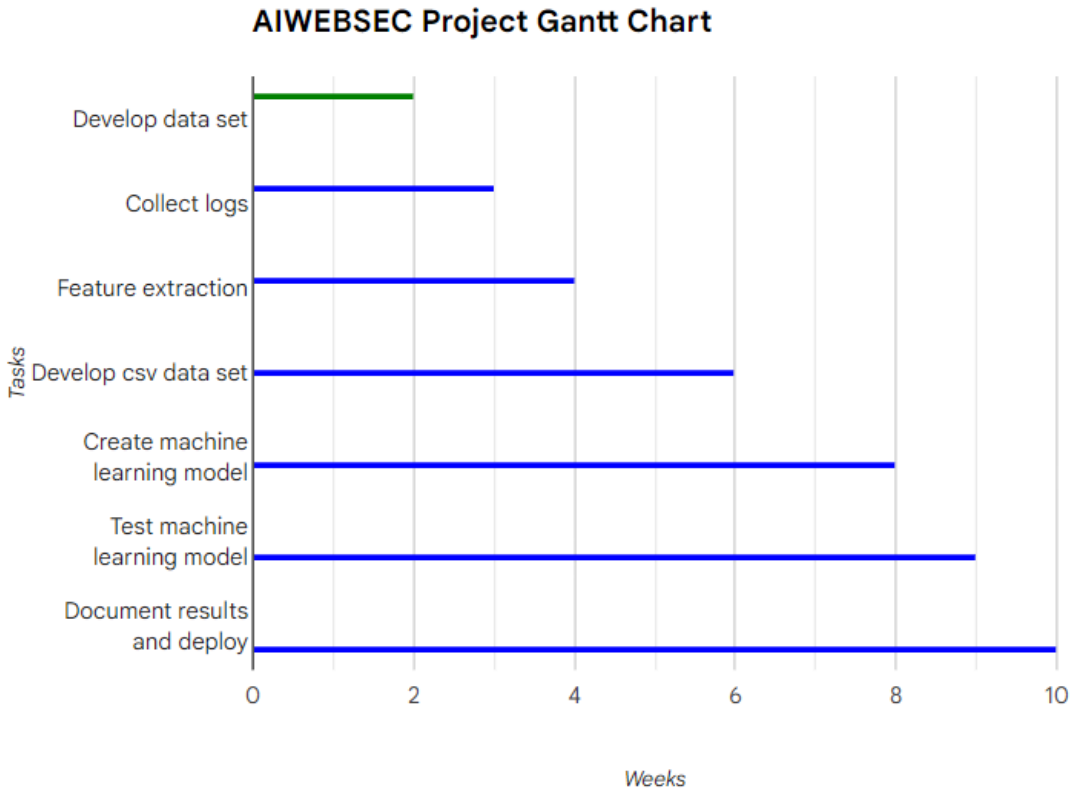

Figure 2.1: Timeline Chart

3. PROPOSED SYSTEM AND METHODOLOGY

3.1. System Architecture

Data Collection Layer:

Component: Acunetix Web Vulnerability Scanner.
Purpose: Generates logs of non-malicious web links by scanning websites for vulnerabilities.

Component: Manual Injection and Attack Simulation.
Purpose: Generates logs of malicious payloads by conducting deliberate injection attacks (e.g., SQL injection, Cross-site scripting) on a controlled environment.

Logging and Processing Layer:

Component: BurpSuite Proxy.
Purpose: Serves as a logging intermediary, capturing HTTP traffic from both Acunetix and manual injection attacks on Windows-based systems.

Component: Custom Python Code.
Purpose: Processes collected logs, decodes base64-encoded data, performs feature extraction, and compiles CSV datasets from the logged information.

Machine Learning Layer:

Component: Feature Selection and Dataset Preparation.
Purpose: Identifies relevant features from the extracted data, preparing datasets for machine learning model training.

Component: Machine Learning Model Development.
Purpose: Trains a supervised classification model using the prepared datasets to distinguish between malicious and non-malicious web links.

Component: Intrusion Detection Mechanism.
Purpose: Utilizes the trained model to analyze incoming requests' features, predicting their nature (malicious or benign) and positioning them within relevant clusters.

Application Layer:

Component: Integration of Machine Learning Model.
Purpose: Integrates the trained model into a real-time web security firewall system.
Functionality: Real-time Analysis and Response.
Purpose: Filters incoming web requests, leveraging the deployed model to detect potential threats and take appropriate action (e.g., blocking malicious links) based on the model's predictions.

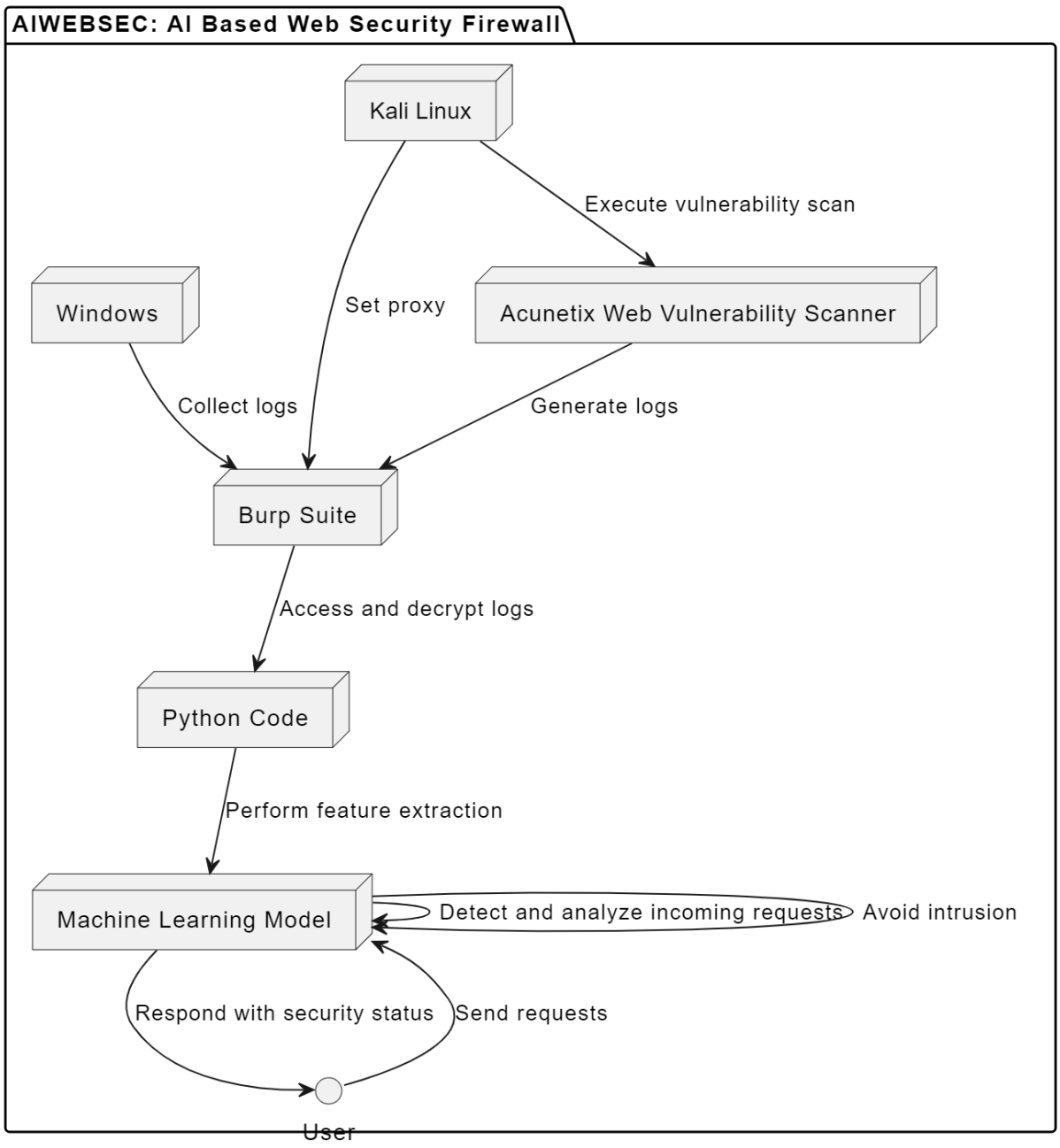

Figure 3.1: AIWEBSEC System architecture

3.1.1. System Flow:

Data Collection:

Acunetix and manual injection attacks generate logs. BurpSuite captures and stores HTTP traffic logs on Windows.

Log Processing:

Custom Python code processes and decodes base64 logs, extracts relevant features, and creates structured datasets for further analysis.

Machine Learning Model:

Trains a machine learning model using the compiled datasets. The model identifies and categorizes incoming web requests as either malicious or non-malicious.

Application Layer:

Integrates the trained model as part of a web security firewall system. Monitors and analyzes incoming requests in real-time, taking automated actions based on the model's predictions to safeguard against potential threats.

This architecture provides a comprehensive view of the AIWEBSEC system, outlining its components, data flow, and functionalities in a professional and structured manner.

3.2. Methodology (Algorithms used)

The development of AIWEBSEC, an AI-based Web Security Firewall, involved an intricate methodology encompassing multiple stages to fortify web defenses. Initial dataset creation was bifurcated into two distinct categories: non-malicious and malicious web links. To compile the non-malicious dataset, the Acunetix Web Vulnerability Scanner, operating within the Kali Linux environment, systematically scanned and identified web links devoid of vulnerabilities. Conversely, the malicious dataset was curated by deliberately generating threats like SQL injection and Cross-site scripting (XSS) attacks, conducted in a controlled setting to ensure ethical boundaries and prevent any legal implications. The logs from both datasets were meticulously captured and acquired using the BurpSuite software on Windows, configured to proxy through the Kali Linux machine, allowing seamless traffic capture for subsequent analysis.

Following log retrieval, a custom Python code was painstakingly crafted to decode and parse the

Base64-encoded logs, thereby unraveling the intrinsic details of requests and extracting relevant features. This feature extraction process, a critical phase, enabled the creation of structured CSV datasets encompassing the discerned characteristics of both benign and malicious web links.

The convergence of these datasets paved the way for the development of a robust machine learning model. This model was meticulously trained to distinguish incoming malicious links from benign ones, utilizing the amalgamated dataset's features as a foundation for intelligent classification. The model's primary objective was to act as a gatekeeper, intercepting and thwarting potentially harmful web links before they could pose any threat.

Moreover, the system was fortified against unforeseen threats through the integration of zero-day attack detection capabilities. In scenarios where new and unidentified threats emerged, the model meticulously scrutinized incoming requests, evaluating their features in real-time. Leveraging clustering analysis techniques, it discerned anomalous patterns and positioned these requests within relevant clusters, allowing for rapid identification and preemptive action against potential intrusions.

In essence, the methodology behind AIWEBSEC embodies a comprehensive approach leveraging data curation, feature extraction, machine learning, and advanced intrusion detection techniques. This amalgamation empowers the web security firewall to proactively defend against both known threats and potential zero-day vulnerabilities, thereby ensuring heightened protection for web-based systems and applications.

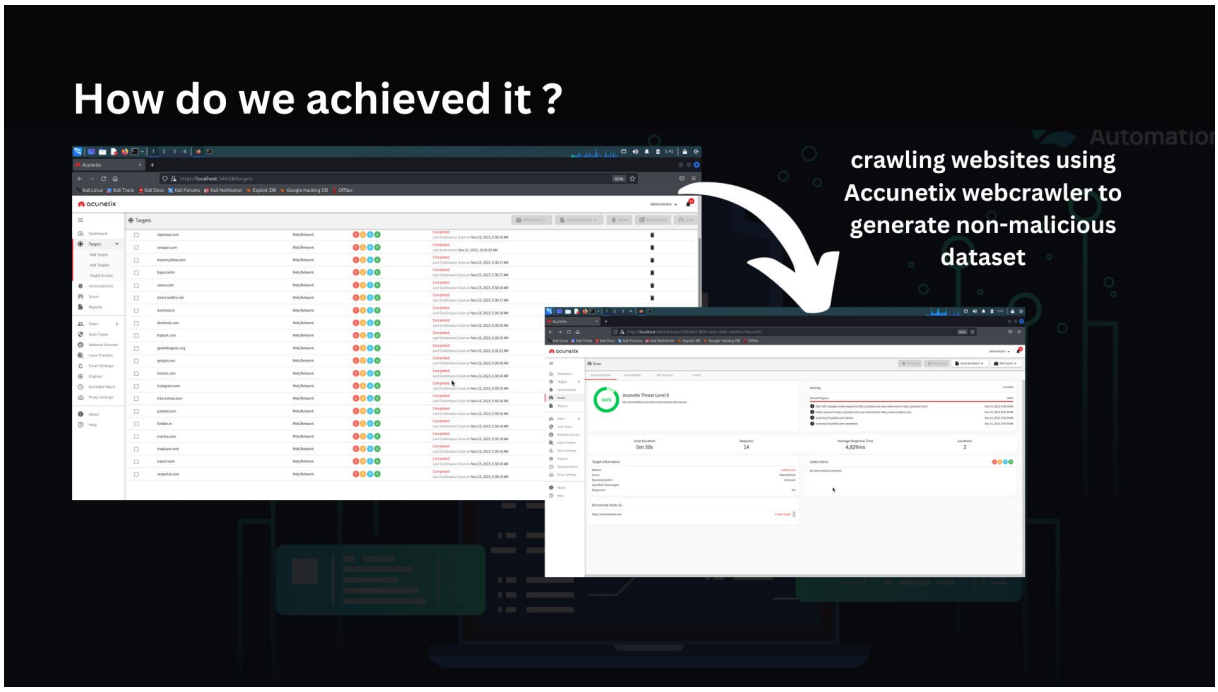

Figure 3.2: Crawling and Collecting non-malicious weblinks and Dataset

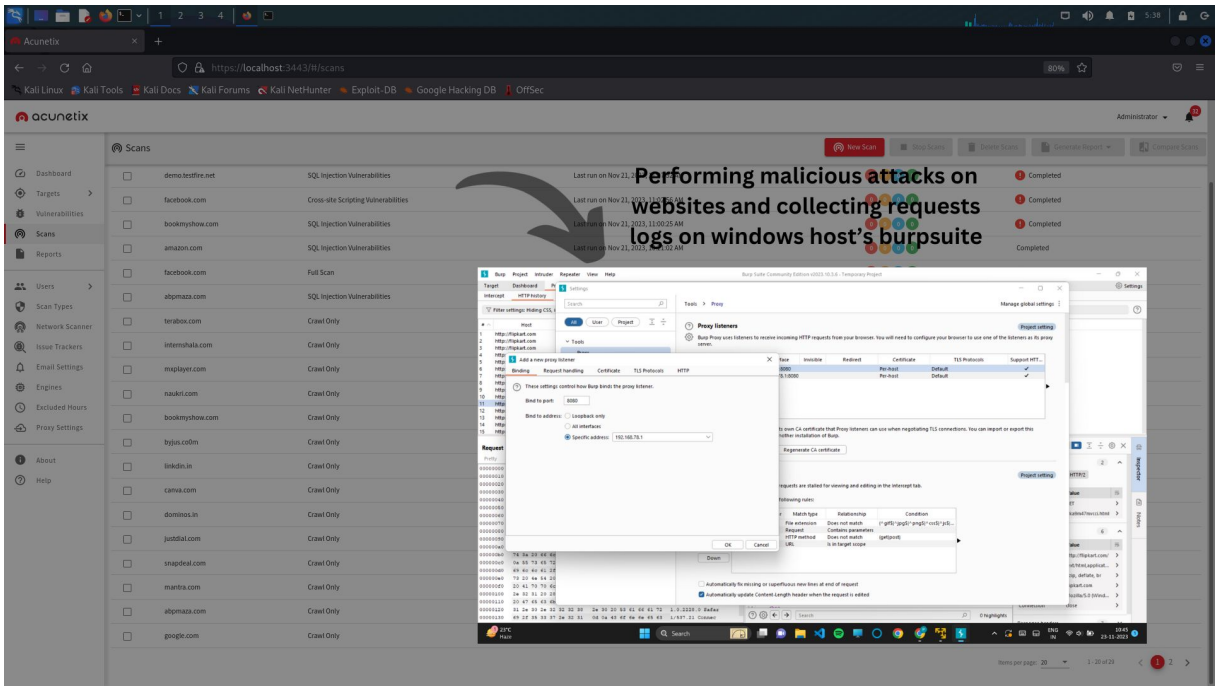

Figure 3.3: Accessing and Collecting malicious weblinks and Dataset

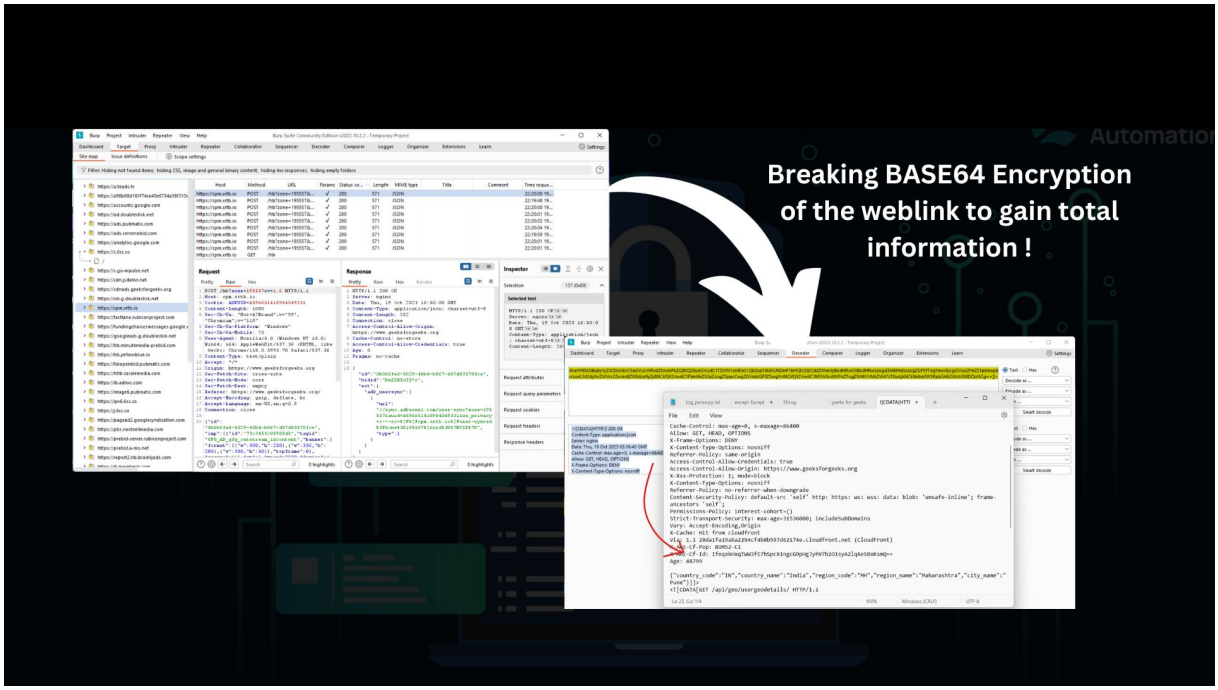

Figure 3.4: Request Log Extraction

3.3. Implementation

The development of AIWEBSEC, an AI-Based Web Security Firewall, involved a meticulously crafted and extensive process from data collection to model deployment. The inception phase comprised the creation of two diverse datasets meticulously curated for their

distinct purposes. The first dataset encompassed non-malicious web links, meticulously identified through comprehensive scans conducted by the Acunetix Web Vulnerability Scanner operating within the Kali Linux environment. Concurrently, the second dataset comprised malicious web links deliberately engineered through Acunetix, simulating various threat scenarios like SQL injection and Cross-Site Scripting (XSS) attacks, ensuring ethical boundaries were upheld by confining these activities within controlled environments on the Kali Linux platform. To acquire comprehensive logs for analysis, an intricately configured BurpSuite on Windows, set to proxy through Kali Linux, facilitated the meticulous capture of HTTP traffic logs generated from both benign and malicious activities. A meticulously designed Python script was then developed to decode and dissect the Base64-encoded logs, meticulously conducting intricate feature extraction and meticulously structuring CSV datasets that amalgamated the attributes of benign and malicious web links. This unified dataset served as the foundational bedrock for training an advanced machine learning model. This model underwent intensive training leveraging sophisticated algorithms such as Support Vector Machines (SVM) or Neural Networks, ensuring robust discernment and classification between malicious and non-malicious links. Augmenting the system's capabilities, a real-time zero-day attack detection mechanism was incorporated, employing vigilant feature analysis on incoming requests and harnessing powerful clustering algorithms to detect anomalies, thereby empowering the firewall to dynamically predict, prevent, and categorize potential intrusions based on a comprehensive evaluation of the model's predictive abilities and the identification of aberrant features within the incoming data stream.

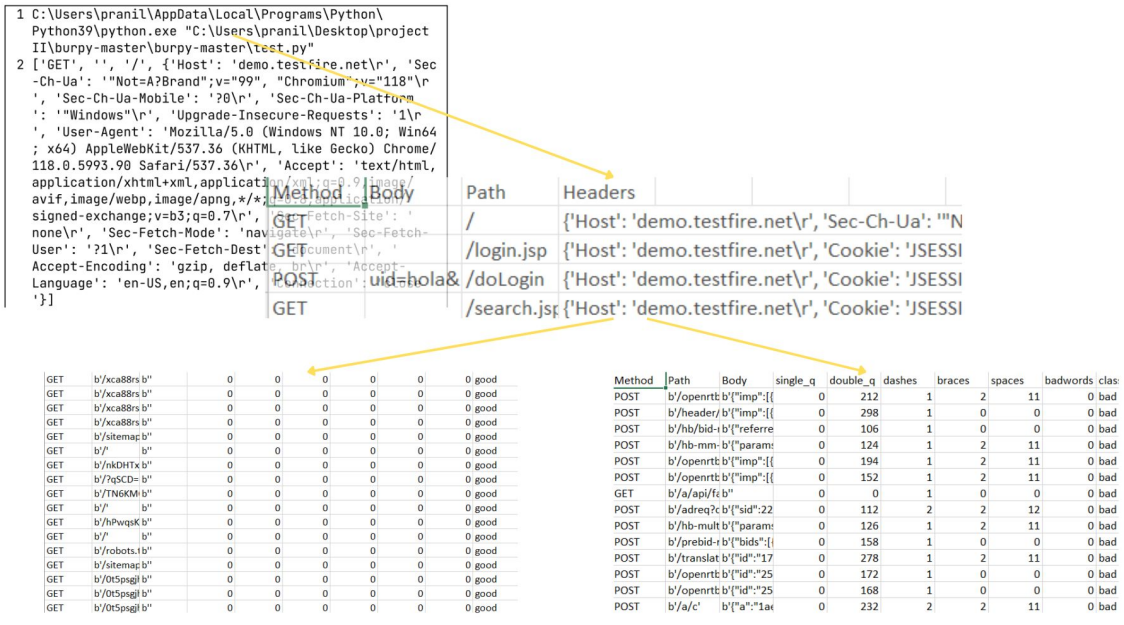

Figure 3.5: Feature Extraction / Dataset Creation

3.3.1. Data Flow Diagrams

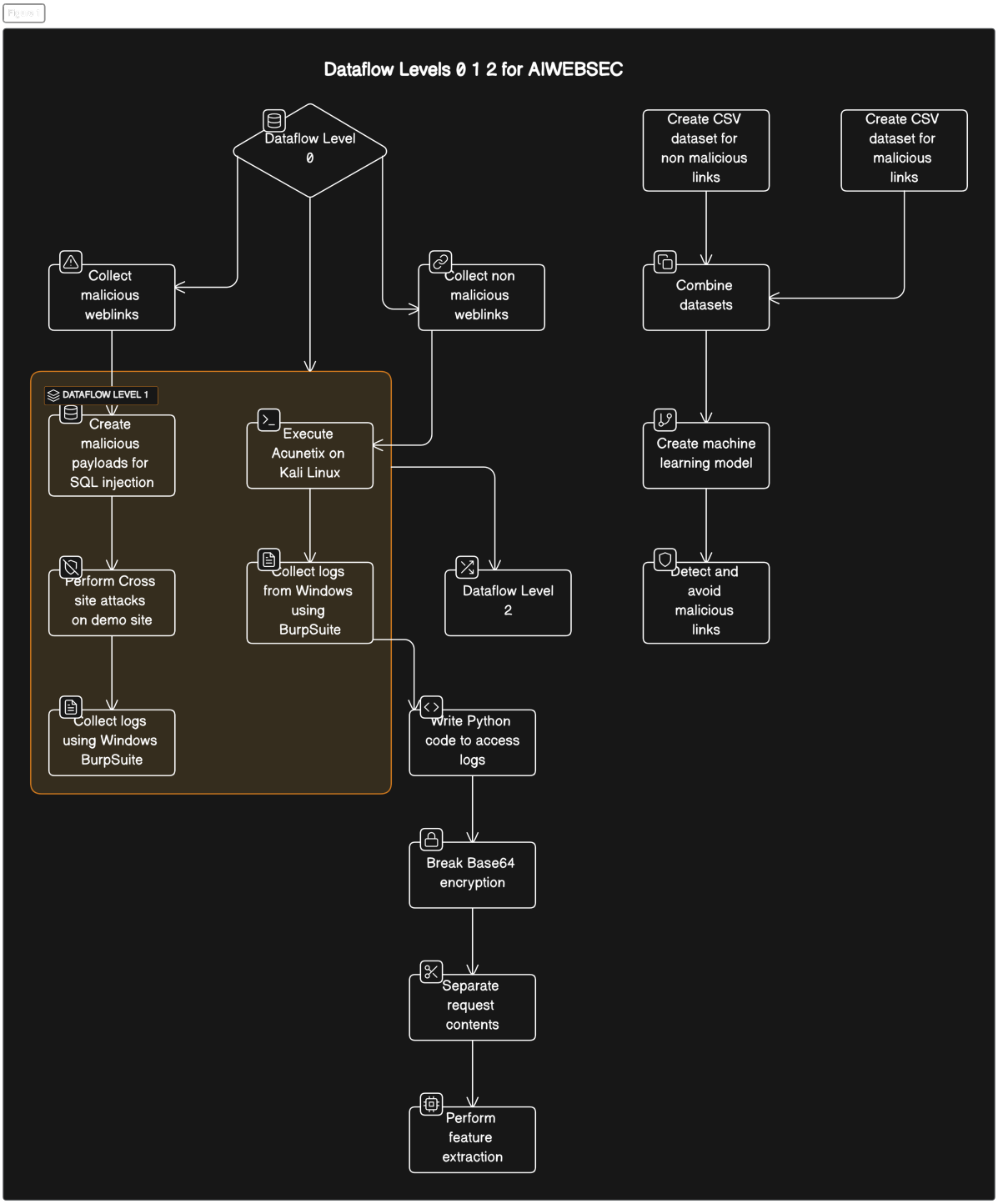

Figure 3.6: Data Flow Diagram

3.3.2. UML Diagrams

1. **Flow Chart Diagram**

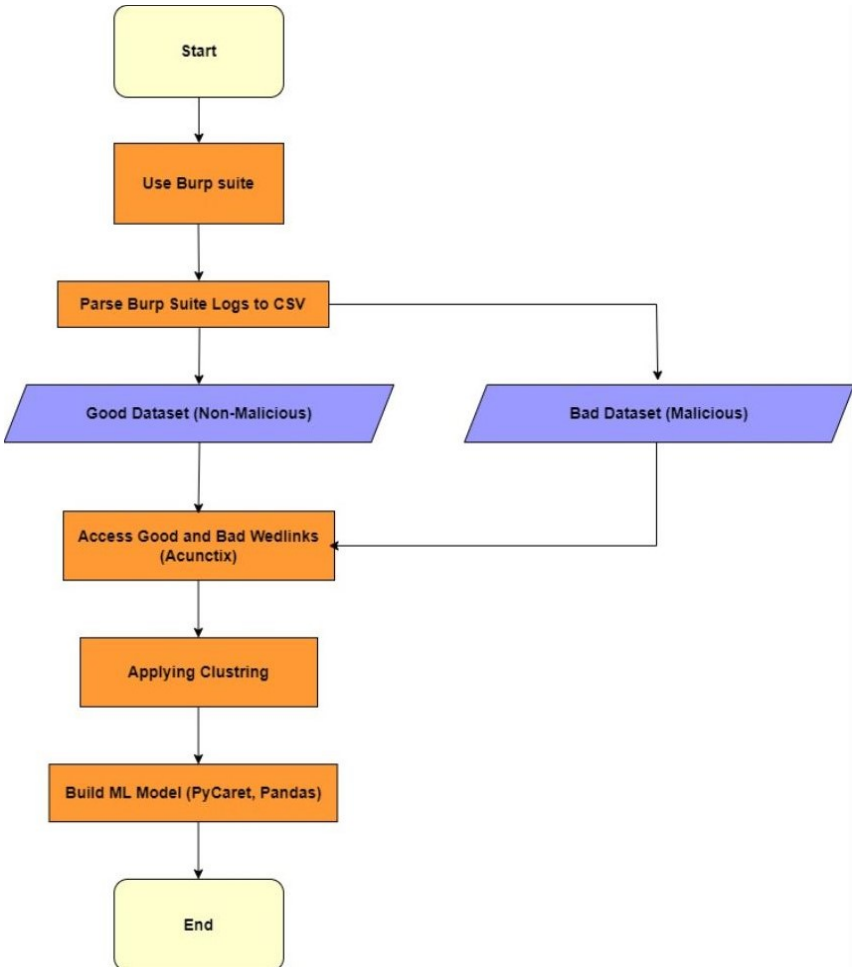

Figure 3.7: Flow chart Diagram

2. Use Case Diagram

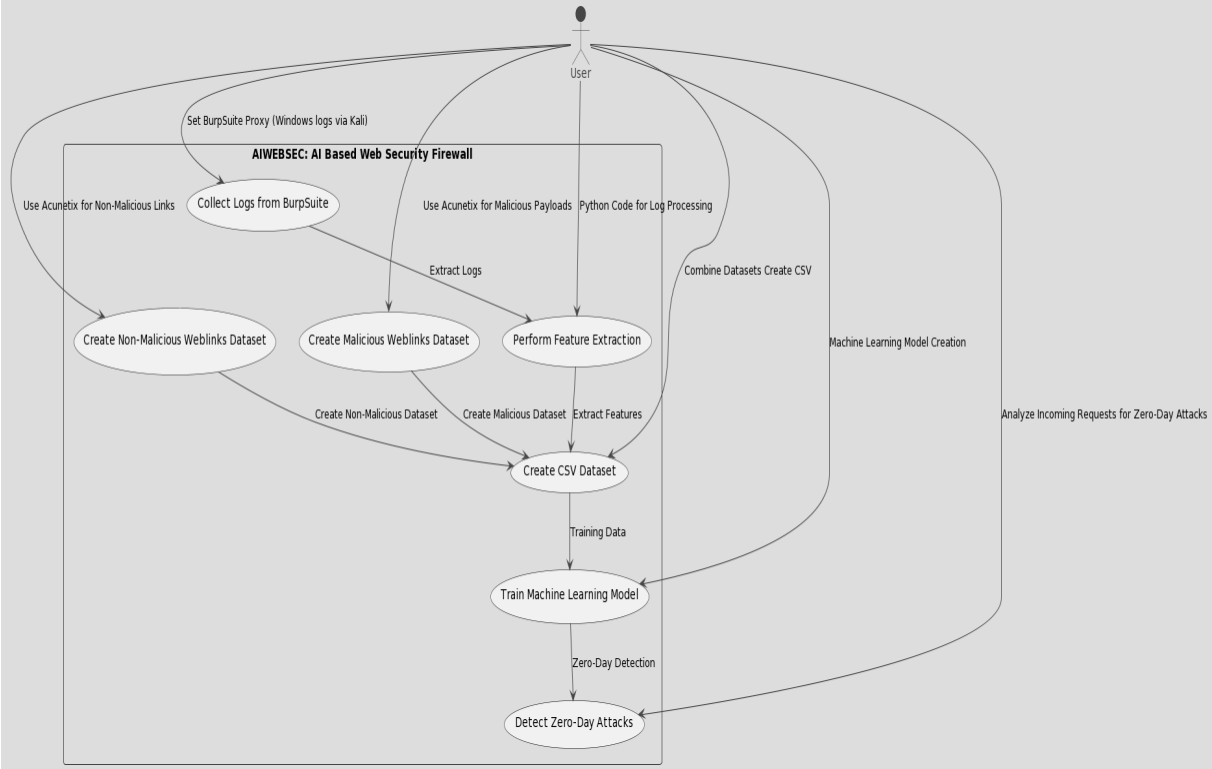

Figure 3.8: Use Case Diagram

3. Sequence Diagram

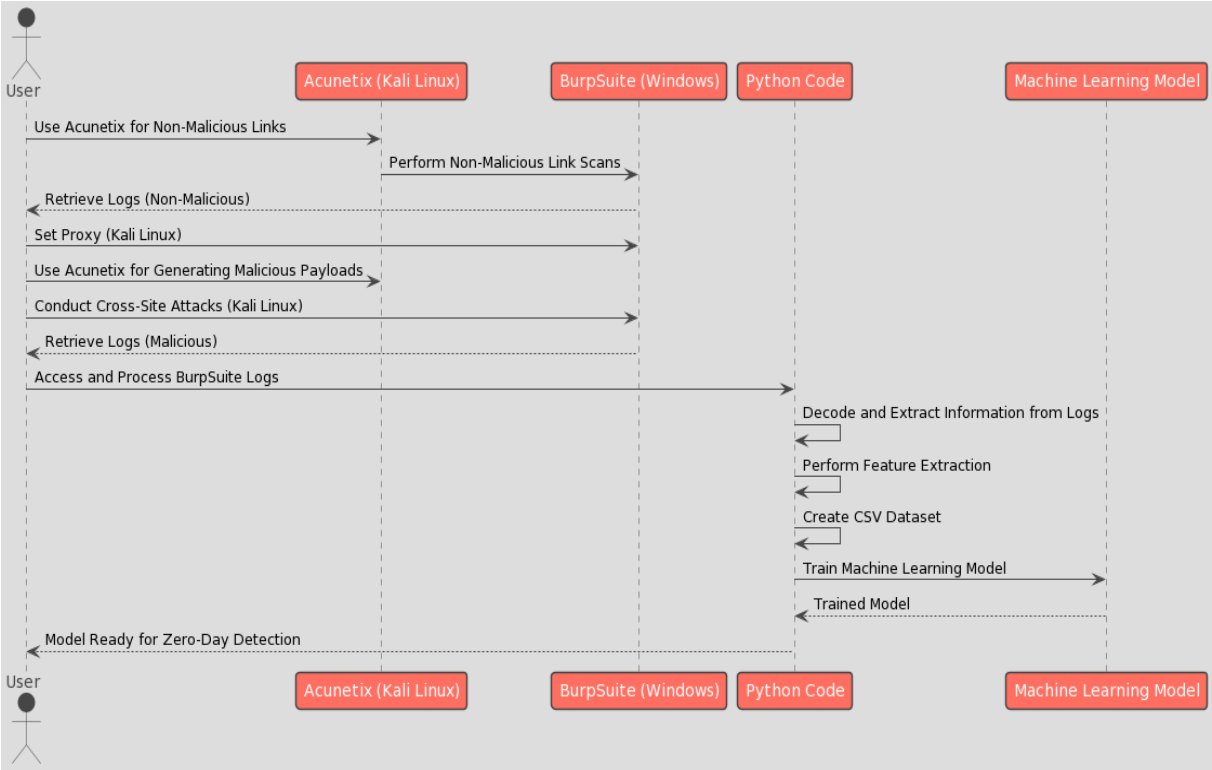

Figure 3.9: Sequence Diagram

4. **Acitvity Diagram**

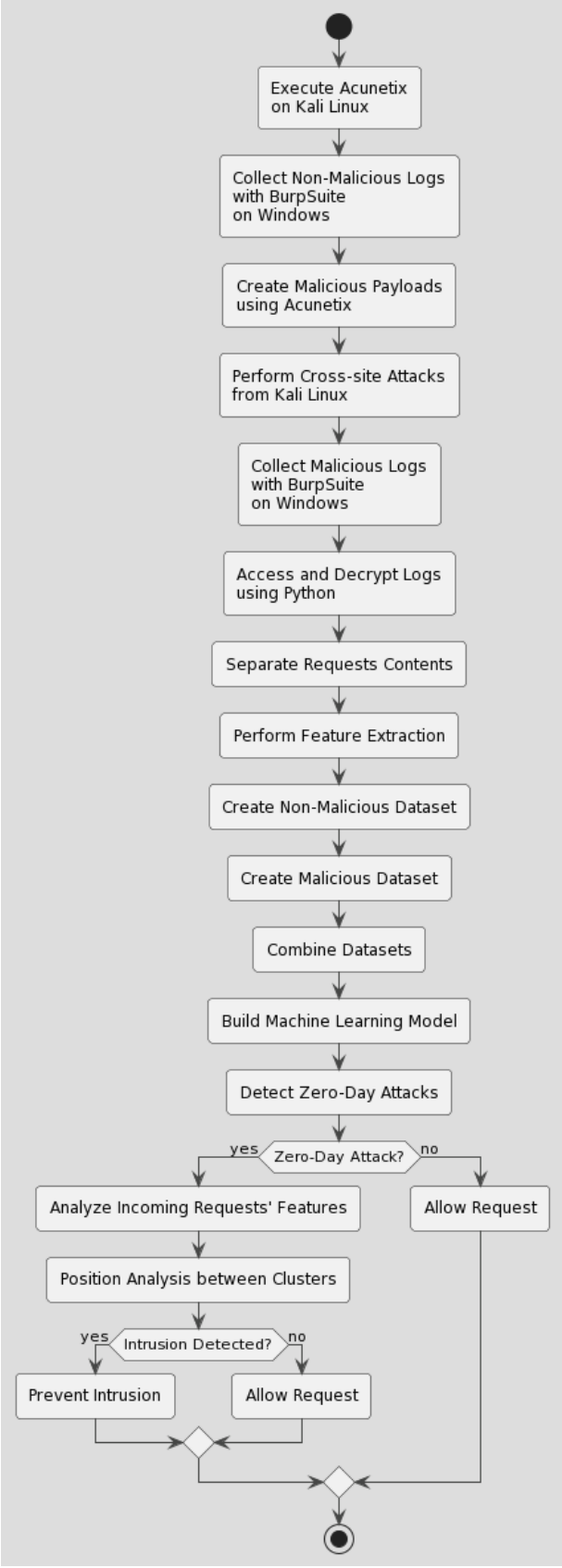

Figure 3.10: Activity Diagram

5. Class Diagram

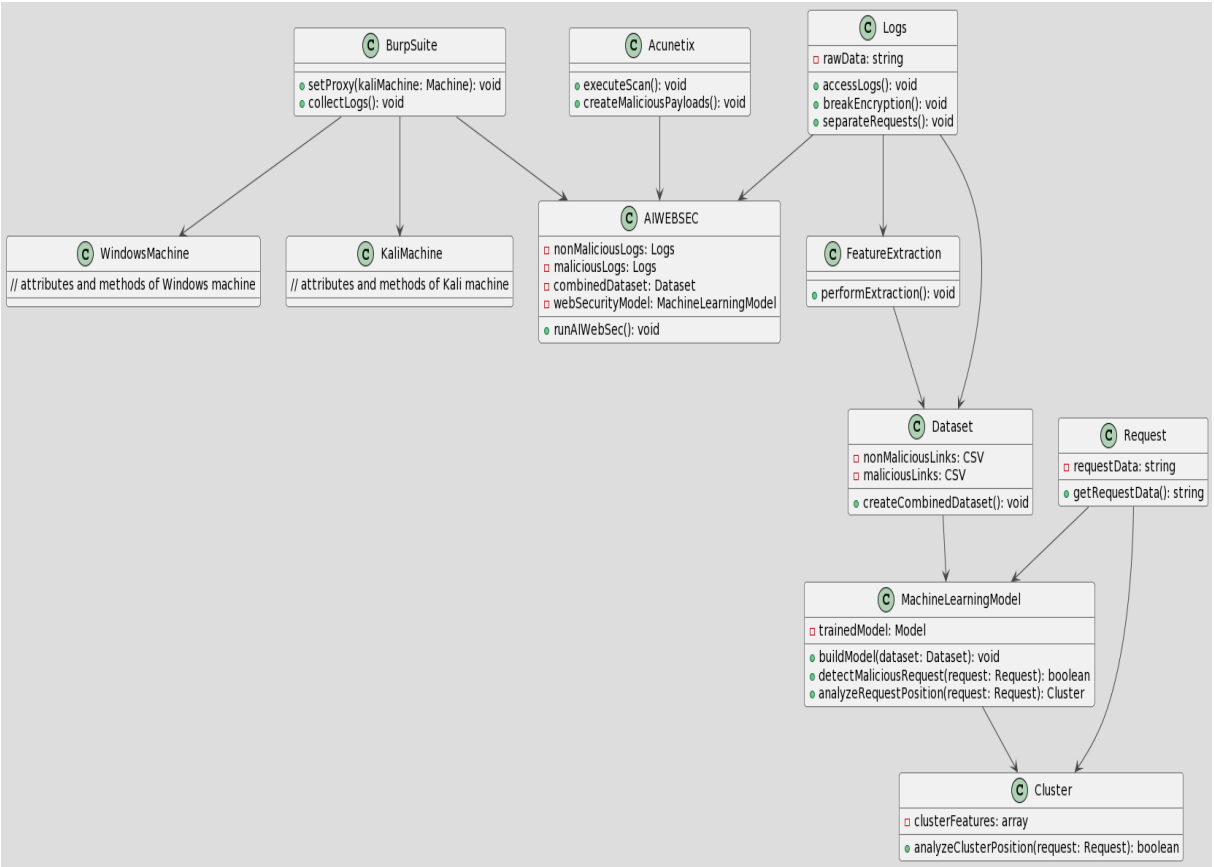

Figure 3.11: Class Diagram

6. **Component Diagram**

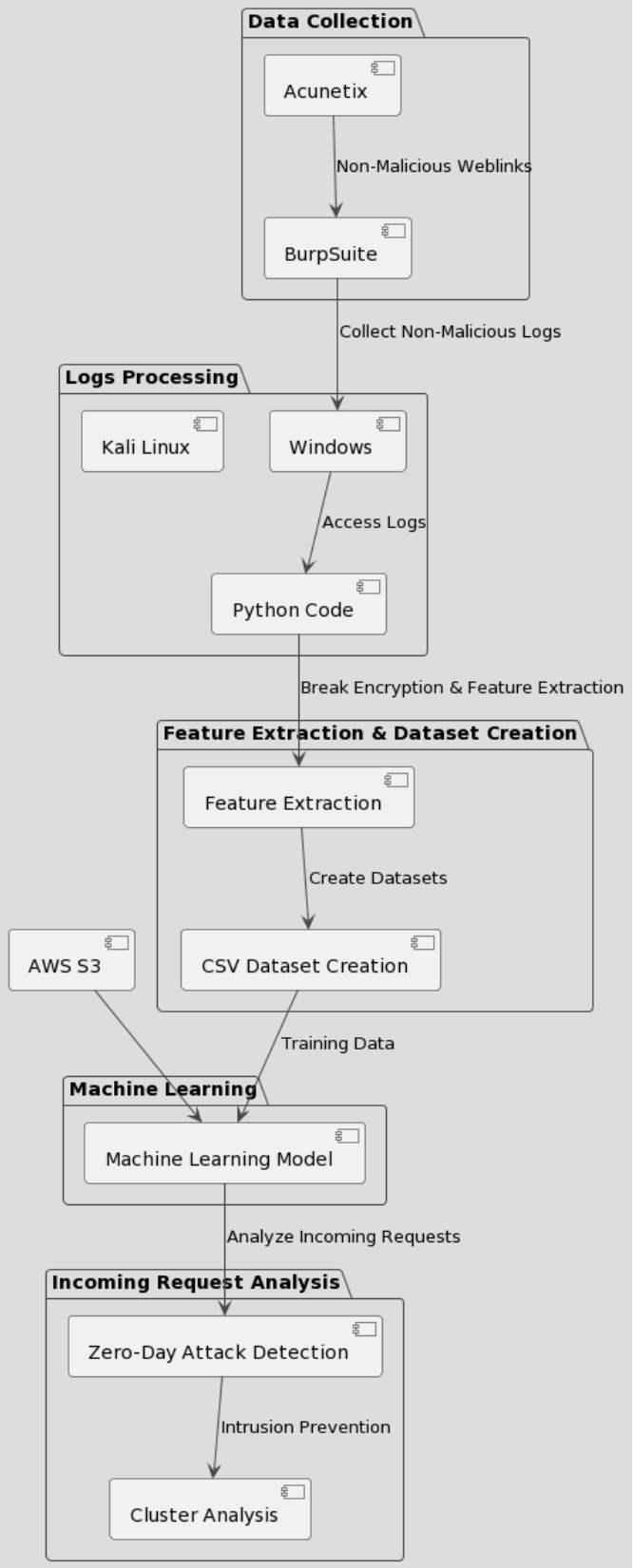

Figure 3.12: Component Diagram

7. Deployment Diagram

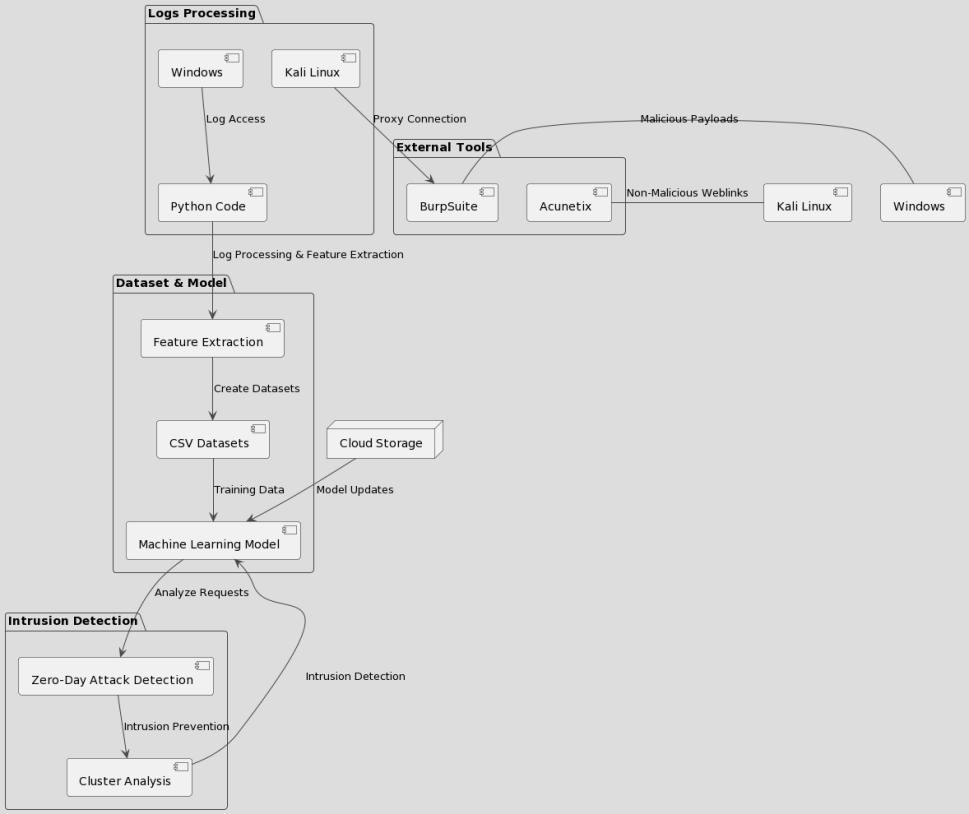

Figure 3.13: Deployment Diagram

4. RESULTS AND EXPLANATION

4.1. Implementation Approaches

Data Collection and Generation: Leveraged Acunetix web vulnerability scanner, both for generating malicious payloads (e.g., SQL injection, Cross-Site Scripting) and validating web vulnerabilities. Employed BurpSuite as a proxy to intercept and record HTTP requests, segregating logs for non-malicious (Acunetix validated) and potentially malicious (payload-induced) web links.

Data Processing and Feature Extraction: Developed a bespoke Python script to decode Base64-encoded logs and perform comprehensive parsing, isolating individual request elements like headers, payloads, and parameters. Executed feature extraction methodologies on parsed logs to discern distinctive attributes and patterns characterizing malicious and non-malicious behavior.

Dataset Preparation: Compiled datasets in CSV format, comprising meticulously extracted features from both the non-malicious and malicious web links' logs. Ensured dataset integrity and quality, utilizing relevant data cleaning and normalization techniques.

Machine Learning Model Development: Utilized Python libraries and frameworks (e.g., Pandas, Scikit-learn) to construct a machine learning pipeline. Employed clustering algorithms (e.g., K-means) to delineate patterns and segregate feature spaces between non-malicious and malicious instances. Trained and validated a machine learning model, possibly employing supervised or semi-supervised learning techniques to classify incoming web link requests.

Integration and Real-time Analysis: Integrated the trained model into the AI-based Web Security Firewall infrastructure for real-time analysis of incoming HTTP requests. Implemented decision-making logic utilizing the model's predictions to dynamically assess and categorize web links as potentially malicious or benign. Deployed an adaptive system to learn from new data, facilitating ongoing refinement and adaptation to evolving attack patterns and zero-day threats.

Ethical and Legal Considerations: Ensured all activities adhered to ethical standards, utilizing controlled and authorized environments for testing purposes. Maintained compliance with legal guidelines and security best practices to avoid any potential legal ramifications.

This professional implementation approach emphasizes meticulous data handling, robust model development, real-time integration, and ongoing adaptation to emerging threats.

Additionally, it underscores the significance of ethical conduct and legal compliance throughout the development and testing phases of the system.

4.2. Pseudo Code

Pseudo Code: Parsing Burpsuite logs

```
Define log file path
log_path = 'new.log'

# Function to parse the log file and extract HTTP request features
function parse_log_and_extract_features(log_path):
    result = {}

    # Parse XML log file
    for each item in log:
        extract request and response
        store request and response in result

    # Process extracted data and write to CSV
    for each extracted request:
        parse raw HTTP request
        extract features (e.g., quotes, characters)
        write features to CSV

# Process log file and extract HTTP request features
parse_log_and_extract_features(log_path)
```

Figure 4.1: Parsing Burpsuite logs

Pseudo Code: Machine Learning Model

```
# Import necessary libraries
import pandas as pd
from pycaret.clustering import *
from http.server import SimpleHTTPRequestHandler, HTTPServer
import urllib.parse
from urllib import request, error

# Read and preprocess data for clustering
http = pd.read_csv("/merge.csv")
clu1 = setup(data=http, normalize=True, numeric_features=['single_q', 'double_q', 'dashes',
'braces', 'spaces', 'badwords'], ignore_features=['method', 'path', 'body', 'class'])
kmeans = create_model('kmeans', num_clusters=2)
kmeans_result = assign_model(kmeans)
pd.concat([http, kmeans_result['Cluster']]], axis=1).to_csv('/kmeans_result_with_class.csv',
index=False)

# Extract features function
def extract_features(path):
    path = urllib.parse.unquote(path)
    features = ['single_q', 'double_q', 'dashes', 'braces', 'spaces', 'badwords']
    counts = [path.count(f) for f in ["\'", '\"', '--', '(', ' ', 'sleep', 'drop', 'uid', 'select', 'waitfor', 'delay',
'system', 'union', 'order by', 'group by']]
    return pd.DataFrame([counts], columns=features)

# Prediction and proxy handling
class SimpleHTTPProxy(SimpleHTTPRequestHandler):
    def do_GET(self):
        parts = self.path.split('/')
        if len(parts) >= 4:
            live_data = extract_features(parts[3])
            result = predict_model(kmeans, data=live_data)
            if result['Cluster'][0] == "Cluster 1":
                print('Intrusion Detected')
        if len(parts) >= 2:
            self.proxy_request('http://' + parts[2] + '/')
        else:
            super().do_GET()

    def proxy_request(self, url):
        try:
            response = request.urlopen(url)
        except error.HTTPError as e:
            self.send_response_only(e.code)
            self.end_headers()
            return

        self.send_response_only(response.status)
        [self.send_header(name, value) for name, value in response.headers.items()]
        self.end_headers()
        self.copyfile(response, self.wfile)

# Set up proxy routes and start HTTP server
SimpleHTTPProxy.set_routes({'proxy_route': 'http://demo.testfire.net/'})
with HTTPServer(('127.0.0.1', 8000), SimpleHTTPProxy) as httpd:
    print(f'Listening on http://{httpd.server_name}:{httpd.server_port}')
    try:
        httpd.serve_forever()
    except KeyboardInterrupt:
        print("\nExiting due to keyboard interrupt")
```

Figure 4.2: Machine Learning Model

4.3. Testing

Test Case ID: TC001

- Objective: Confirm the presence of distinct logs for both non-malicious and malicious activities.
- Description: This test case verifies the successful collection of logs for both non-malicious and malicious activities using Acunetix and BurpSuite.
- Preconditions: Acunetix and BurpSuite are properly configured and operational.
- Test Steps: Execute Acunetix for non-malicious weblinks. Perform attacks for malicious weblinks. Collect logs using BurpSuite with Kali Linux as the proxy. Expected Results: Verify the existence of separate logs for non-malicious and malicious activities in the designated log files.
- Actual Results: Non-malicious logs collected. Malicious logs collected.

Test Case ID: TC002

- Objective: Validate the Python code's ability to process logs and extract features.
- Description: This test case checks the functionality of the Python code used for log processing and feature extraction.
- Preconditions: Python environment is set up with required dependencies.
- Test Steps: Provide sample logs to the Python code. Execute the Python code for log processing and feature extraction.
- Expected Results: Ensure the code decrypts logs, separates requests accurately, performs feature extraction, and generates CSV datasets for both types of links.
- Actual Results: Log decryption successful. Request separation accurate. Feature extraction performed correctly. CSV datasets generated for malicious and non-malicious links.

4.4. Analysis (graphs/chart)

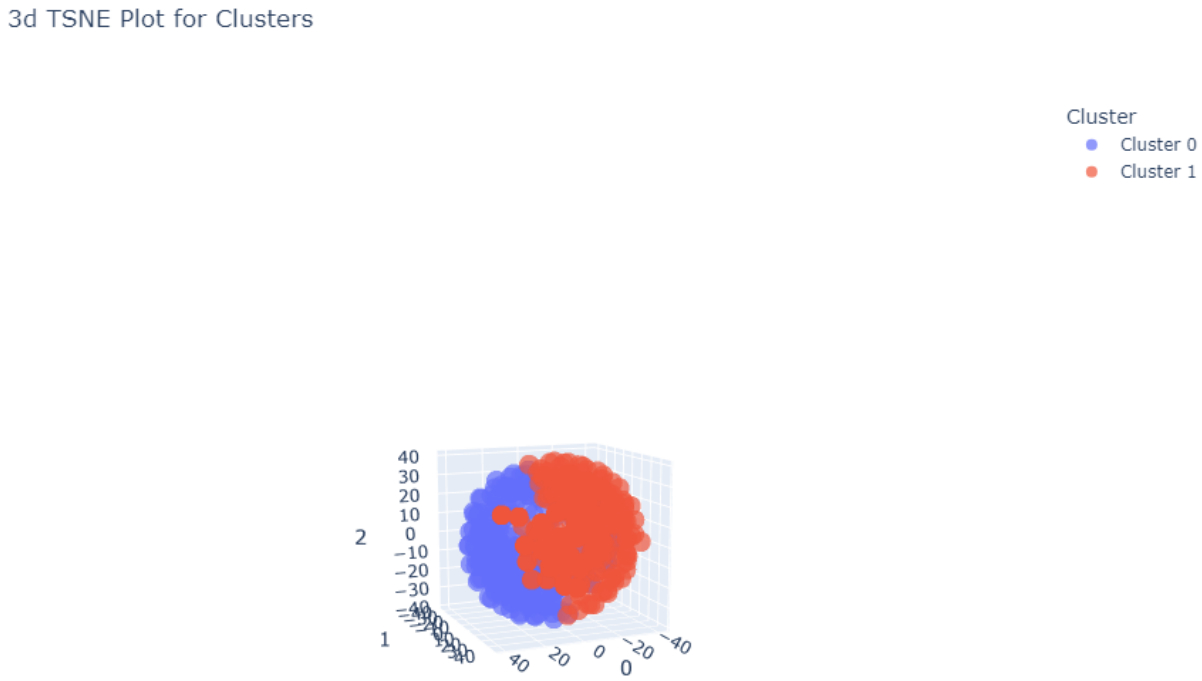

Figure 4.3: 3d Clustering

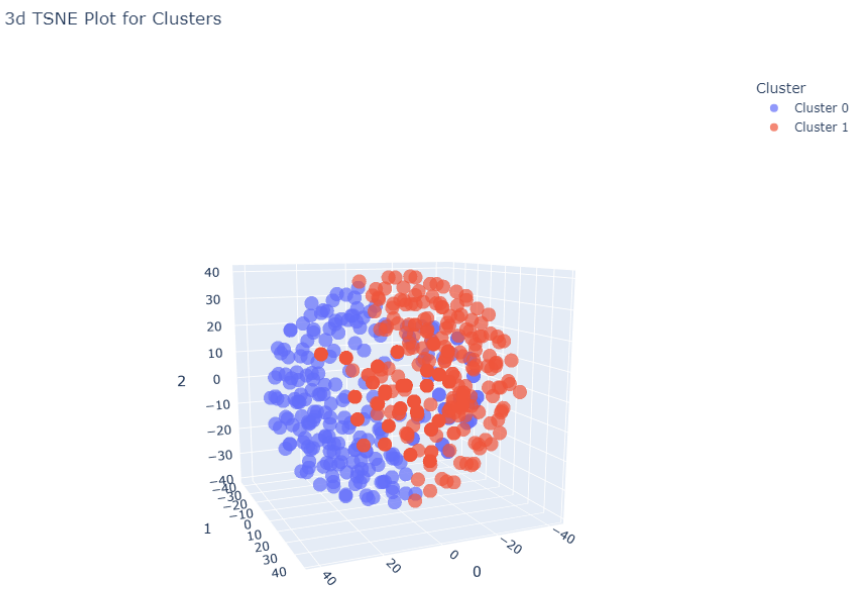

Figure 4.4: 3d Clustering

5. CONCLUSION

The AIWEBSEC project is an extensive and meticulously crafted endeavor aimed at bolstering web security infrastructure through a comprehensive and multifaceted approach. Leveraging sophisticated tools such as the Acunetix web vulnerability scanner and BurpSuite, the project meticulously curated two distinct datasets: one encompassing non-malicious web links and the other encapsulating malicious counterparts. This comprehensive data collection process involved executing controlled attacks like Cross-Site Scripting and SQL injection, meticulously logging activities via BurpSuite on Windows while setting up proxies to ensure comprehensive and accurate data capture. The subsequent development of a custom Python script showcased adept handling of logs, encompassing Base64 decryption, parsing, and comprehensive feature extraction methodologies, ultimately culminating in the creation of detailed CSV datasets for both categories of web links. These datasets served as the foundational bedrock for the creation of an advanced machine learning model, a pivotal component within the AIWEBSEC architecture.This machine learning model, developed with a profound understanding of features extracted from the datasets, exhibits proactive capabilities in discerning and categorizing incoming web links, effectively distinguishing between benign and potentially malicious content, thereby fortifying the security posture of web assets. Moreover, the model's innate capability to identify zero-day attack features and analyze their spatial positioning within feature clusters empowers the system to dynamically adapt and proactively mitigate emerging threats. This adaptability and astute threat analysis highlight the AIWEBSEC project's innovation and its ability to evolve and preemptively counter sophisticated cyber threats.In essence, the AIWEBSEC project marks a significant leap towards an intelligent and preemptive AI-driven web security firewall solution. Its comprehensive data gathering, adept feature extraction, and development of an advanced machine learning model showcase an innovative approach in fortifying web-based infrastructures against the ever-evolving landscape of cyber threats. Through the utilization of cutting-edge technology and intelligent algorithms, AIWEBSEC establishes an adaptive, resilient, and proactive defense mechanism, significantly contributing to the robustness and defense posture of web-based systems and assets.This detailed conclusion encapsulates the multifaceted nature of the AIWEBSEC project, emphasizing its sophistication in data collection, feature extraction, model development, and proactive threat mitigation strategies, thereby establishing it as a pioneering endeavor in the realm of web security solutions.

References

[1] A. Tekerek and O. Bay, "Design and implementation of an artificial intelligence-based web application firewall model." *Neural Network World*, no. 4, 2019.

[2] A. Mishra, A. Agrawal, and R. Ranjan, "Artificial intelligent firewall," in *Proceedings of the International Conference on Advances in Computing and Artificial Intelligence*, 2011, pp. 204–207.

[3] J.-Á. Román-Gallego, M.-L. Pérez-Delgado, and M. L. Viñuela, "Development of web application firewall based on artificial intelligence," in *International Conference on Disruptive Technologies, Tech Ethics and Artificial Intelligence*. Springer, 2023, pp. 18–27.

[4] P. Chakraborty, M. Z. Rahman, and S. Rahman, "Building new generation firewall including artificial intelligence," *International Journal of Computer Applications*, vol. 975, p. 8887, 2019.

[5] G. El Hajal, R. A. Z. Daou, and Y. Ducq, "Human firewall: Cyber awareness using whatapp ai chatbot," in *2021 IEEE 3rd International Multidisciplinary Conference on Engineering Technology (IMCET)*. IEEE, 2021, pp. 66–70.

[6] Q. Cheng, C. Wu, H. Zhou, Y. Zhang, R. Wang, and W. Ruan, "Guarding the perimeter of cloud-based enterprise networks: an intelligent sdn firewall," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2018, pp. 897–902.

[7] K. Kallepalli and U. B. Chaudhry, "Intelligent security: Applying artificial intelligence to detect advanced cyber attacks," in *Challenges in the IoT and Smart Environments: A Practitioners' Guide to Security, Ethics and Criminal Threats*. Springer, 2021, pp. 287–320.

[8] B. Dash, M. F. Ansari, P. Sharma, and A. Ali, "Threats and opportunities with ai-based cyber security intrusion detection: A review," *International Journal of Software Engineering & Applications (IJSEA)*, vol. 13, no. 5, 2022.

[9] B. Morel, "Artificial intelligence and the future of cybersecurity," in *Proceedings of the 4th ACM workshop on Security and artificial intelligence*, 2011, pp. 93–98.

[10] N. Gupta, A. Saikia, and D. Sanghi, "Web application firewall," *Indian Institute of Technology, Kanpur*, vol. 61, p. 62, 2007.

[11] K. Demertzis and L. Iliadis, "Cognitive web application firewall to critical infrastructures protection from phishing attacks," *Journal of Computations & Modelling*, vol. 9, no. 2, pp. 1–26, 2019.

[12] Q. A. Al-Haijaa and A. Ishtaiwia, "Machine learning based model to identify firewall decisions to improve cyber-defense," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 11, no. 4, pp. 1688–1695, 2021.

[13] D. Kant and A. Johannsen, "Evaluation of ai-based use cases for enhancing the cyber security defense of small and medium-sized companies (smes)," *J. Electron. Imaging*, vol. 34, pp. MOBMU–387, 2022.

[14] C. Torrano-Gimenez, A. Perez-Villegas, and G. Alvarez, "A self-learning anomaly-based web application firewall," in *Computational Intelligence in Security for Information Systems: CISIS'09, 2nd International Workshop Burgos, Spain, September 2009 Proceedings*. Springer, 2009, pp. 85–92.

[15] I. F. Kilincer, F. Ertam, and A. Sengur, "Machine learning methods for cyber security intrusion detection: Datasets and comparative study," *Computer Networks*, vol. 188, p. 107840, 2021.