Zero Trust Security for Web Applications in Microservice-Based Environments

Jayaraj Viswanathan
Department of Computer Science and Engineering,
Amrita Vishwa Vidyapeetham,
Chennai, India,
ch.en.u4cys21026@ch.students.amrita.edu

Dinesh Kumar N
Department of Computer Science and Engineering,
Amrita Vishwa Vidyapeetham,
Chennai, India,
ch.en.u4cys21014@ch.students.amrita.edu

***Abstract*—With the rapidly shifting landscape in cybersecurity, cyber threats have made the old perimeter-based models obscurate. With the increasing proliferation of cloud-native applications and microservice architectures into organizations, a sound security model shall be a basic necessity that shall withstand these threats. This paper introduces a Zero Trust Security Model for web applications in Kubernetes clusters, utilizing key technologies such as JSON Web Token (JWT) authentication and mutual Transport Layer Security (mTLS). Unlike traditional models that rely on predefined trust zones, this approach enforces continuous identity verification, dynamic access control, and fine-grained micro-segmentation to secure sensitive data. The model incorporates Zero Trust principles, including encrypted communications, strict user and device authentication, and real-time threat detection. It provides robust policy enforcement to protect APIs, containers, databases, and other critical resources in a microservices environment. The persistent observation of user activity, along with the implementation of machine learning algorithms for the identification of threats, improves the system's capability to recognize and address potential incidents in real-time. Our experimental validation shows that this model effectively minimizes lateral movement and unauthorized access, significantly reducing attack surfaces. The proposed Zero Trust architecture offers a scalable, secure, and resilient framework suited to modern containerized applications, ensuring data integrity and security.**

Keywords*— *Zero Trust Security, JWT Authentication, Mutual TLS, Cybersecurity, Web Application Security, Adaptive Access Control, Micro-segmentation, Data Protection, Cloud-Based Architecture, Kubernetes Clusters.

I. INTRODUCTION

Organizations are facing a wide range of cyber threats in the modern digital environment that reflect the weaknesses of traditional perimeter-based security protocols. These kinds of frameworks - often called "castle-and-moat" defenses - operate on the assumption that threats come from outside a network, thus leaving internal resources open to vulnerabilities. At present, this oversimplification approach has led to serious violations, such as the 2017 Equifax hack, which exposed the personal information of approximately 147 million people and had an estimated cost of $4 billion in remediation and settlements. The rapid adoption of cloud computing, agile methodologies, and containerization has significantly expanded the attack surface, making organizations more susceptible to sophisticated cyber threats. Incidents associated with phishing, ransomware, and insider threats have increased, with ransomware attacks having risen by 150% in 2020, with an estimated damage of about $20 billion. Further, the Verizon 2020 Data Breach Investigations Report indicated that 86% of breaches were financially motivated, highlighting the critical need for adaptive and proactive security measures that can evolve alongside emerging threats. To counter these challenges, the Zero Trust Security Model is emerging as the new paradigm shift in the more comprehensive approach to cybersecurity. Here, unlike the traditional models wherein trust zones are predefined and based on network locations, Zero Trust requires no inherent trust for any entity-be it a user, device, or application. All access requests are screened through continuous authentication and authorization irrespective of origin. In the philosophy of "never trust, always verify" which has a very significant relevance in environments that are characterized with microservice architectures with dispersed data and services on many containers, accessible to users from diversified locations. This paper reports on the conceptualization and implementation of a Zero Trust framework in a modern web application with a microservice architecture. By employing key technologies such as JSON Web Tokens (JWT) for stateless authentication, mutual TLS (mTLS) for secure communication, and micro-segmentation to isolate services, the proposed framework enhances security by continuously verifying all entities and enforcing strict access controls.

This significantly mitigates risks associated with lateral movement attacks, potentially leading to widespread data breaches. The motivation for adopting Zero Trust in Kubernetes clusters arises from the complexities of securing modern distributed systems. Traditional models often grant access based on network location, which is inadequate in environments where applications are highly decoupled and operate across various cloud providers. In these settings, vulnerabilities can be exploited, necessitating a model that safeguards distributed resources and sensitive data effectively. It presents noteworthy contributions toward the domain of cybersecurity by dealing with the constraints inherent in conventional security models and adapting to the fluid characteristics of cloud-native applications. The proposed framework provides a scalable and flexible approach to safeguarding contemporary, containerized settings to ensure the integrity and confidentiality of sensitive information at all places across an organization's application ecosystem.

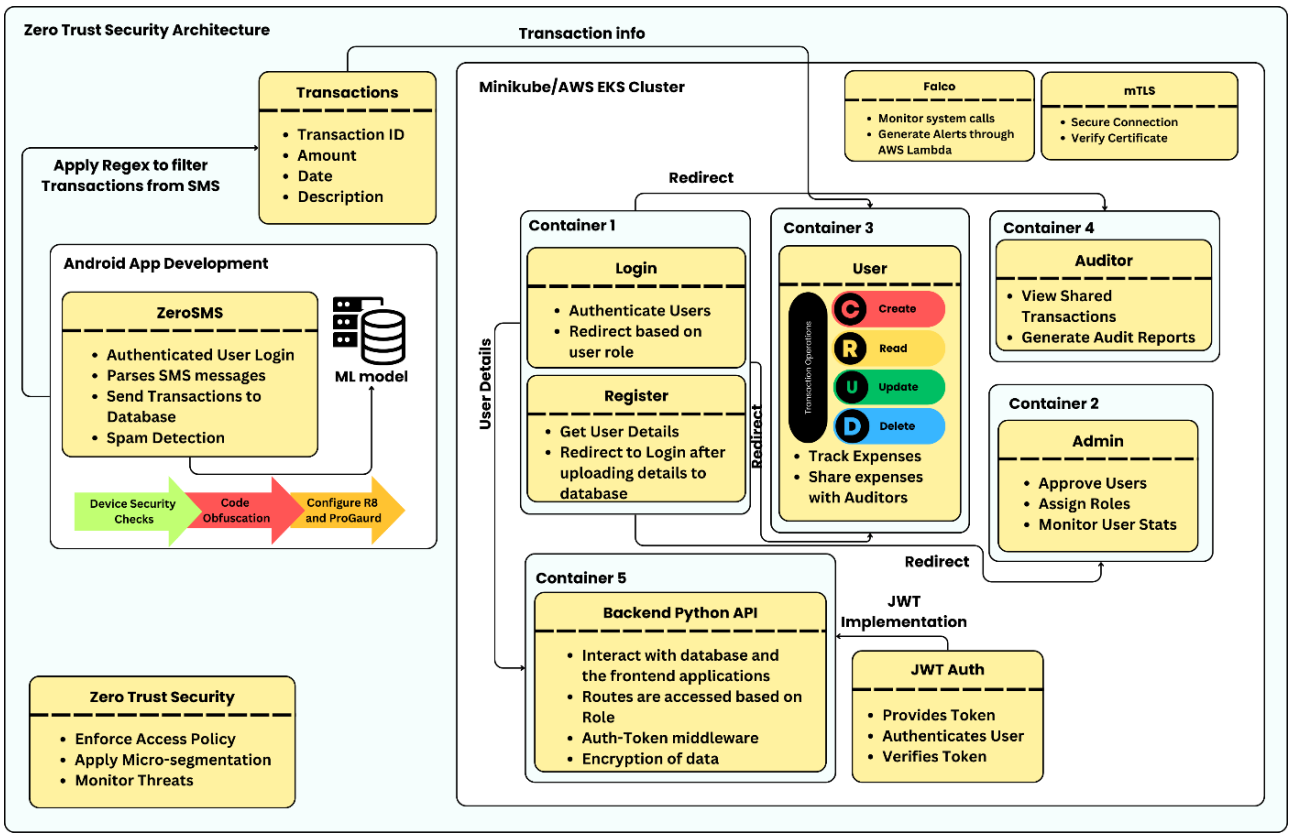

Fig. 1. Architecture diagram of the implementation of Zero Trust in the Web Application

II. Related Works

D'Silva et al. researched the adoption of Zero Trust Architecture (ZTA) in Kubernetes environments [7], emphasizing its relevance in a cloud computing context. They emphasized how ZTA continuously verified users, devices, and applications, challenging traditional trust models. The study discussed the use of technologies such as Kubernetes, Docker and RBAC/ABAC for enhanced access control. Additionally, they critiqued traditional security frameworks, noting the inadequacies of perimeter-based models in modern cloud infrastructures. Their proposed architecture improved security through continuous verification and logging, making it adaptable to decentralized systems.

Varalakshmi et al. [8] explored the enhancement of JSON Web Token (JWT) authentication within Software Defined Networks (SDNs). They identified vulnerabilities in traditional JWT implementations and proposed a modified authentication approach that improves security without sacrificing performance. Their analysis highlighted the importance of securing API endpoints and user sessions, ultimately enhancing the overall integrity of network communications in SDN environments. Bucko et al. [9] assessed how the JWT authentication and authorization process could be improved in web applications using user behavior history. Their study analyzed how behavior-based metrics could improve the reliability of authentication processes. They proposed a dynamic authentication mechanism that adapts based on user activity, significantly reducing unauthorized access and enhancing user security in web applications.

Jánoky et al. [10] conducted an analysis of revocation mechanisms for JSON Web Tokens (JWTs). They identified the challenges associated with effectively revoking tokens without degrading system performance. Their research emphasized the need for efficient revocation strategies to ensure the security of applications relying on JWTs, contributing valuable insights into the management of token lifecycles. Achary and Shelke [11] have analyzed fraud detection in banking transactions by applying some machine learning methodologies. They proposed a framework in order to reduce the fraudulent activities by using multiple algorithms in real time. This showed the effectiveness of machine learning for the enhancement of financial transaction security, providing banks with the same skills to prevent fraud. Thus, the authors underscore the adaptive models that learn from fraud patterns, which are dynamic in nature, making credit card security protocols much more effective.

Fraudulent Transaction Detection in Credit Card by Applying Ensemble Machine Learning Techniques by Prusti et al. [12]: Researches ensemble machine learning models for fraudulent credit card transaction detection. By combining multiple algorithms, it is an effort to enhance precision, accuracy, and speed of detection. Key findings show that ensemble methods outperform individual classifiers in identifying fraudulent transactions, reducing false positives, and increasing overall efficiency in real-time fraud detection systems. Jaculine Priya and Saradha [13] reviewed machine learning algorithms for fraud detection and prevention. Their comprehensive analysis encompassed various techniques, assessing their strengths and weaknesses in different contexts. The paper provided a critical overview of the current state of fraud detection systems, emphasizing the potential of machine learning in enhancing predictive accuracy and operational efficiency in fraud management.

A.I. Weinberg and others [14] discussed Zero Trust Architecture (ZTA) in relation to application and network

security, outlining relevance in modern paradigms of cybersecurity. The research detailed how ZTA eliminates implicit trust and enforces strict access controls, providing a robust model for securing sensitive data in dynamic environments. Kang and others [15] provided an abridged version of the Zero Trust security, from theoretical perspectives to practice. They discussed several implementations and looked into the possibility that Zero Trust principles can have an augmenting effect on the overall security posture. Their findings illustrated the applicability of Zero Trust in diverse settings, emphasizing its importance in mitigating modern cybersecurity threats.

Ashfaq et al. [16] conducted an in-depth review of the Zero Trust security framework. This study investigated existing literature while outlining fundamental principles and strategies applicable for Zero Trust in various scenarios. The authors pointed out gaps in related studies, thus providing ideas for future research to deepen understanding and applicability of Zero Trust principles. Liu et al. [17] evaluated the relevant literature for Zero Trust, seeing its potential in IoT systems. They have identified major challenges as well as approaches towards Zero Trust principles in IoT environments. Consequently, they concluded that strict access controls with continuous verification must be in place to solve security issues. The authors provided a comprehensive overview of current research trends and highlighted areas requiring further investigation to enhance IoT security through Zero Trust frameworks.

Mehraj and Banday [18] proposed a Zero Trust framework with cloud computing contexts to overcome security-related issues such as identity theft, data breaches, and complications in trust management. The model proposed by the authors emphasizes the necessity of strong access controls coupled with continuous verification owing to the dynamic and shared nature of cloud services. The authors explained the inadequacy of traditional security approaches within the context of cloud. Muddinagiri et al. [19] presented a method for deploying Kubernetes locally using Minikube to manage Docker containers, emphasizing the advantages of containerization as a lightweight alternative to virtual machines. The paper highlights the importance of local Kubernetes testing for industries like finance and healthcare, which require secure, scalable applications without relying on cloud-based deployments. Their approach was demonstrated using a Python-based web server built with a DockerFile. Pace's thesis [20] focuses on the implementation of Zero Trust Networks using Istio in Kubernetes environments. Istio, as a service mesh, manages traffic, observability, and security, offering features like mutual TLS, JWT-based authentication, and seamless integration with Kubernetes, enhancing microservices security and operational efficiency.

III. Methodology

The proposed containerized web application is designed to provide users with a comprehensive expense tracking system that adheres to Zero Trust security principles as illustrated in Figure 1. This application uses Docker for containerization, Kubernetes for Container Orchestration and Helm to dynamically handle configurations of Kubernetes manifests. The complete application is then deployed to AWS EKS cluster remotely enforced with strict IAM policies and VPC.

A. Web Application Overview

At the core of this application is the user interface, which enables users to easily perform CRUD operations (Create, Read, Update, Delete) on their financial transactions. Users can manually input their expenses, savings, and investments through a form-based system that adds each entry as a card component. A distinguishing feature of this application is its integration with the ZeroSMS mobile app. The ZeroSMS app automatically parses SMS messages related to transactions, which are often sent by banks or payment services, and synchronizes them with the web application. This functionality is especially beneficial for users who may forget to log some expenses or are too busy to manually enter transaction details Another key feature is to the ability for users to share their transaction data with a selected list of auditors. The auditors, who are granted read-only access to user transactions, can view the data to prepare financial reports. The auditor's interface displays list of users who have shared their transaction data, allowing them to access the data without modifying it, which aligns with Zero Trust's principle of minimal privileges for authorized users.

A. Admin and Auditor Interfaces

The admin interface is critical for managing user roles and access within the application. New users must first register through a registration page, but their access to the system is only granted upon approval by the admin. Role-based access control guarantees that only the resources required for completing a task in an assigned role are accessible to a given user, which supports the core principle of Zero Trust: least privilege. The admin panel has as well behavior analytics dashboard, along with user management functionality. This feature enables the admin to monitor important logs, such as user login and logout times, system activity, and other behaviors that may indicate suspicious activity. Since the application continuously monitors these behaviors, it will be possible to detect anomalies that can be useful in preventing unauthorized access or wrong usage of the system. This functionality enhances the Zero Trust principle of "never trust, always verify" because the users will have continuous authentication and their behavior examined to reveal possible threats.

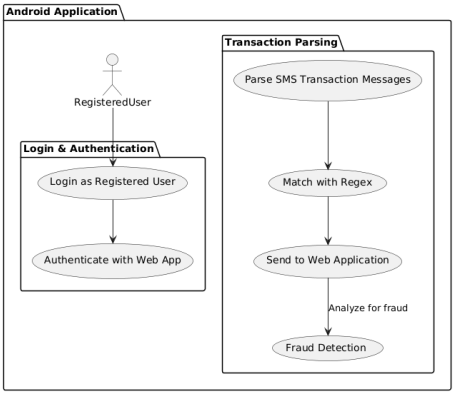

Fig. 2. Activity Diagram for the ZeroSMS Android Application

B. Access Tokens functionality and Implementation

Each request to access sensitive data is validated JSON Web Tokens (JWT), which provide secure, time-bound access tokens that validate the user's identity and role. If a user's JWT token is invalid or expired, access is immediately denied, ensuring that no unauthorized requests are processed. JWTs are compact, URL-safe tokens that enable secure information exchange between parties,

comprising three elements as illustrated in Figure 3: a header, payload or payload with user claims, and a signature to validate the authenticity. This structure allows for seamless authentication flows, where the server can validate the token without maintaining session information.

The Python APIs to generate and manage and rigorously validate JWT are containerized and built to run as a separate pod inside the Kubernetes cluster which ensures continuous authentication of requests from the clients. Upon successful authentication, the system generates a JWT containing essential user-specific claims, such as roles and permissions, which is then transmitted to the user. For subsequent requests, the user includes this token in the authorization header, allowing the application to verify the token's integrity and validity before granting access to protected resources. This implementation does not only follow the least privilege principle by limiting access based on existing permissions granted to the user but also adds security layers as it uses transient access tokens, thus decreasing threats against token theft or misuse.

C. Spam Detection in Financial Transactions

The ZeroSMS app parses the SMS messages and uses reges to classify them as either transaction-related or non-transaction-related. This functionality not only helps users track their expenses more effectively but also mitigates the risk of overlooking minor transactions such as small purchases and pocket money. As, illustrated in Figure. 2, the workflow demonstrates how SMS parsing feeds into the user's expense management process, emphasizing the seamless integration between the mobile and web platforms. To further enhance the functionality of the ZeroSMS app, a Machine Learning (ML) model was developed to detect spam messages. The model was created using a spam SMS dataset sourced from Kaggle, and constructed through a synthesis of supervised learning strategies along with ensemble methodologies, utilizing algorithms such as Random Forest, Gradient Boosting, and SVM. Through rigorous testing procedures, the model arrived at an accuracy of above 96%, thereby proving to be an efficient tool for finding fraudulent activities. As shown Table 1, the performance metrics highlight the model's precision and recall rates, emphasizing its capability to minimize false positives and negatives in spam detection.

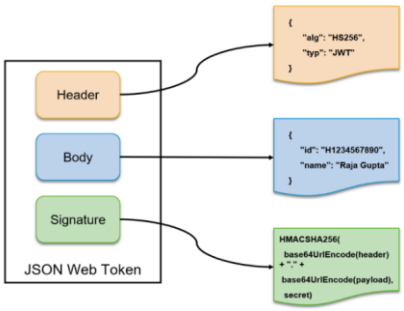

Fig. 3. JWT Architecture

D. Policy Enforcement Point for RBAC

A Policy Enforcement Point (PEP) is one of the key components in security enforcement within web applications within a Zero Trust Architecture (ZTA). The PEP acts as a gatekeeper that intercepts all incoming requests before they reach the application's core services, ensuring that only authorized users or services gain access. When implemented in microservice based web applications, the PEP evaluates each request based on predefined security policies. These policies determine who can access which service and resources, at what time, and under what conditions. The PEP typically collaborates with the Policy Decision Point, which decides access according to such policies as RBAC or ABAC. Once a decision is made by the PDP, the PEP either grants or denies access.

In practical terms, the PEP is embedded at critical points within the microservice architecture, such as API gateways or middleware layers. Every incoming request to the service of application containers passes through the PEP, where it is authenticated, and authorization policies are applied. If the user or service is unauthorized or violates policies, the PEP blocks access. For an instance, in our application a, the PEP ensures that an auditor is allowed to access only the transactions and data related to their client. Also, the routes behind the services are restricted and are specific to the entities (user, admin, auditor) to access. It could also enforce least privilege access, allowing users or services to only perform the actions necessary for their role. Additionally, PEPs can be deployed as sidecar containers in microservices, ensuring that access controls are enforced at each service endpoint. This is especially important when services communicate internally within distributed environments. Overall, PEP strengthens security by ensuring consistent policy enforcement, providing granular access control, and mitigating potential attack vectors.

E. mTLS within the Kubernetes Clusters

Mutual TLS (mTLS) is a foundational security mechanism in implementing Zero Trust Architecture (ZTA) for microservices [21]. It provides robust authentication and encrypted communication between microservices by requiring both the client and server to present valid certificates, ensuring mutual trust before communication occurs. This mutual verification process is key to the "never trust, always verify" principle of ZTA. In mTLS, each pod of the cluster gets a unique cryptographic certificate that is issued by a trusted CA. When two services communicate with each other, each party to the communication-up the client and the server upload their certificates in the handshake protocol. Each uses the public key of the CA to verify the certificate of the other party. After verification, the certificates can set up an encrypted communication channel using TLS, or Transport Layer Security.

This ensures that data exchanged between services is not only authenticated but also protected from eavesdropping and tampering. mTLS is integrated into our application service meshes using a tool named *Istio* which automate certificate management and enforce mTLS at the network layer. In these environments, mTLS is applied transparently to microservices, without requiring each service to implement its own TLS logic. The service mesh handles certificate issuance, renewal, and rotation, ensuring seamless mTLS enforcement. Unlike traditional systems that assume trust once inside the network, mTLS requires that every interaction between services involves identity verification [22]. This eliminates the risk of unauthorized services accessing resources, as only those with valid certificates are allowed to communicate. When combined with Policy Enforcement Points (PEPs), mTLS strengthens the overall ZTA. The PEP verifies that requests originate from authorized services with valid certificates before granting access to resources.

IV. RESULTS

With the Zero Trust application, some of the fundamental principles of Zero Trust Architecture were fully

implemented: strict access control, continuous authentication, verification of end-users and devices, and proper security protection of sensitive resources by integrating role-based access control and least-privilege policies. The integration of Kubernetes for container orchestration further enhanced scalability and resource management, allowing the system to adapt to the changing security requirements in real-time. Overall, the implementation exhibited robust security postures with minimal performance overhead. The web application was deployed in an EKS cluster, utilizing node groups belonging to the EKS Cluster. As illustrated in Figure 4., the architecture ensures scalability and fault tolerance, with seamless communication between services for a smooth user experience.

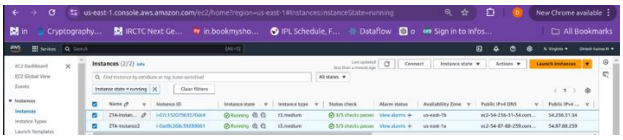

Fig 4. Node Groups of EKS Cluster deployed as EC2 instances

Security within the application is strengthened through AWS Lambda and the Falco tool, which monitors the cluster for anomalies and triggers alerts for real-time threat detection, as illustrated in Figure 5. showcases the alerts from Lambda as logs captured in AWS Cloudwatch. These logs contain valuable information such as the violated rule, container name, container ID, pod name, and namespace where the attack occurred, along with the system call used to perform the attack. This collected data shows the deployment of the Zero Trust strategy and assists in identifying the cause of suspicious activity.

Additionally, session logs provide a thorough audit trail, tracking user activities and identifying suspicious behavior. These logs ensure compliance with Zero Trust principles by ensuring all user actions are continuously monitored and recorded.

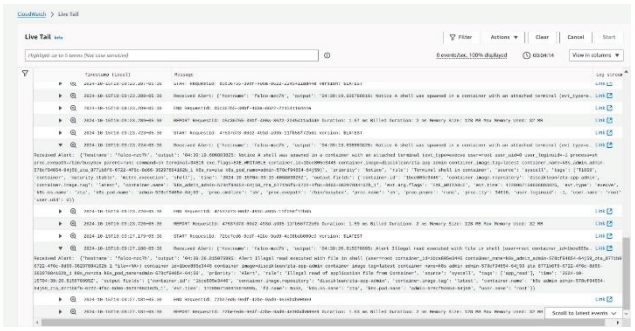

Fig 5. AWS Lambda Alerts triggered by Falco tool deployed in the cluster

In Spam detection model, Naive Bayes was the best model with F1 score 0.918 and accuracy 0.978 as illustrated in Table 1, showing excellent precision and recall balance. As for the precision, the best accuracy was achieved by SVM with a value of 1.0, and for this model, ROC AUC was also the highest at 0.984. The Logistic Regression model was excellent as well, reporting the F1 score of 0.909. Other models had a bit lower F1 scores but maintained high accuracy across all models in Random Forest and Gradient Boosting.

To prepare for production release, the app leverages the JailMonkey library to detect rooting and unsafe environments, while utilizing R8 and ProGuard tools for code optimization and obfuscation to enhance performance and security as shown in Figure 6. The APK is securely signed with a private key from a trusted keystore, ensuring authorized distribution. Due to R8 and ProGuard obfuscation, reverse engineering ZeroSMS is significantly more challenging as illustrated in Figure 7.

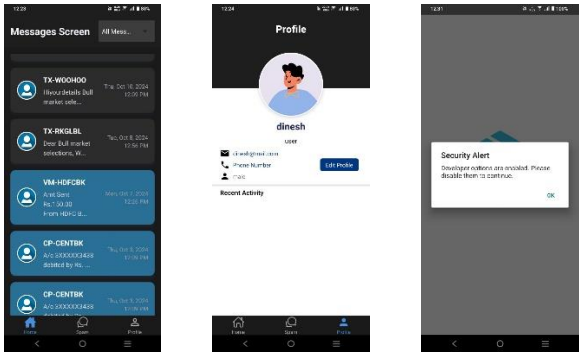

Fig. 6. ZeroSMS in production release with security features

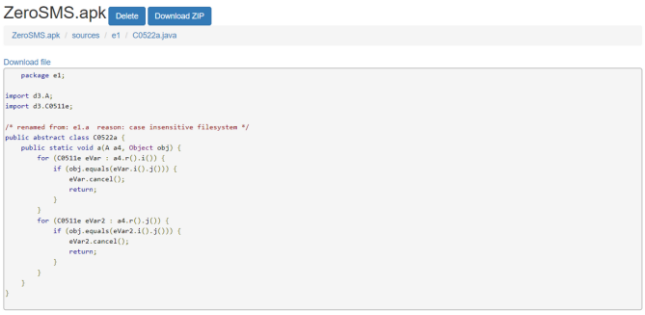

Fig. 7. ZeroSMS with class name and code Obfuscation

The MobSF scan illustrated in Figures 8 a) & b) shows ZeroSMS achieving a security score of 57 (Grade B), 83.87% higher than a standard APK, which scored 31 (Grade C). ZeroSMS has SMS permissions and allows "http" traffic for backend connectivity, impacting its security score. In contrast, the standard APK is merely a "Hello World" application built in Android Studio, lacking any additional privileges.

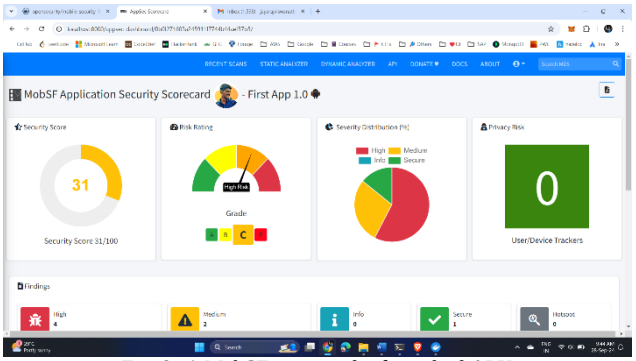

Fig. 8. a) MobSF scan result of standard APK

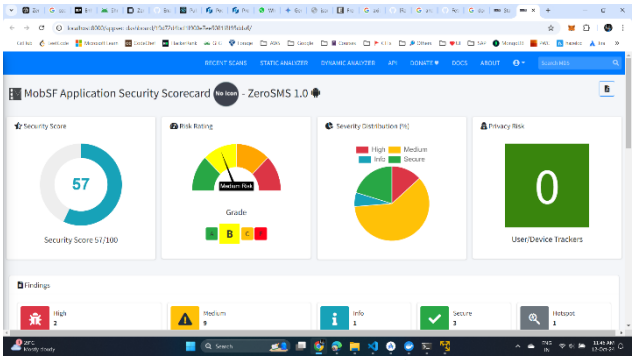

Fig. 8. b) MobSF scan result of ZeroSMS

Table 1. Comparison of Model Performance Metrics

Model	F1 Score	Recall	Precision	Accuracy	Mean Squared Error	ROC AUC Score	Mean Absolute Error	Specificity
Naive Bayes	0.918	0.918	0.918	0.978	0.022	0.975	0.022	0.988
SVM	0.9505	0.826	1.0	0.977	0.023	0.984	0.023	1.0
Logistic Regression	0.909	0.845	0.984	0.978	0.022	0.983	0.022	0.998
Random Forest	0.905	0.826	1.0	0.977	0.023	0.981	0.023	1.0
Gradient Boosting	0.87	0.781	0.983	0.969	0.031	0.973	0.031	0.998

V. CONCLUSION AND FUTURE WORK

The implementation of Zero Trust Architecture (ZTA) marks a significant shift in cybersecurity strategies, particularly for cloud-based and decentralized systems. Traditional perimeter-based security models have proven inadequate against sophisticated cyber threats and the challenges posed by remote work. By eliminating implicit trust and enforcing continuous verification of users and devices, ZTA addresses vulnerabilities in legacy frameworks. Technologies like Kubernetes and JSON Web Tokens (JWT) are vital in deploying Zero Trust principles, enabling fine-grained access control and dynamic policy enforcement. Continuous logging and monitoring enhance the threat detection and response capabilities of an organization while establishing a security-aware culture. In digital transformation, Zero Trust models will form an important countermeasure against risks in contemporary computing environments. Future studies should focus on the challenges encountered in the application of ZTA in different contexts such as IoT and decentralized identity management contexts. Enhancements to the Zero Trust application could include advanced behavioral analytics for detecting anomalous activities and integrating machine learning algorithms for adaptive risk scoring to adjust access privileges in real-time.

VI. REFERENCES

[1] O. E. Imokhai, T. E. Emmanuel, A. A. Joshua, E. S. Emakhu, and S. Adebiyi, "Zero Trust Architecture: Trend and Impact on Information Security," International Journal of Emerging Technology and Advanced Engineering, vol. 12, no. 7, pp. 87-92, July 2022.

[2] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, NIST Special Publication 800-207 Zero Trust Architecture. Gaithersburg, MD: National Institute of Standards and Technology (NIST), 2020.

[3] Y. He, D. Huang, L. Chen, Y. Ni, and X. Ma, "A Survey on Zero Trust Architecture: Challenges and Future Trends," Wireless Communications and Mobile Computing, vol. 2022, Article ID 6476274, 13 pages, 2022. DOI: https://doi.org/10.1155/2022/6476274.

[4] A. Mustyala and S. Tatineni, "Advanced Security Mechanisms in Kubernetes: Isolation and Access Control Strategies," ESP Journal of Engineering & Technology Advancements, vol. 1, no. 2, pp. 45-52, 2021.

[5] B. Pranoto, "Threat Mitigation in Containerized Environments," Applied Research in Artificial Intelligence and Cloud Computing, vol. 6, no. 8, pp. 142-151, 2023.

[6] S. Bagheri, H. Kermabon-Bobinnec, S. Majumdar, Y. Jarraya, L. Wang, and M. Pourzandi, "Warping the Defence Timeline: Non-disruptive Proactive Attack Mitigation for Kubernetes Clusters," in Proc. IEEE International Conference on Communications (ICC 2023), Rome, Italy, 28 May - 01 June, 2023, pp. 567-574.

[7] D. D'Silva and D. D. Ambawade, "Building a Zero Trust Architecture Using Kubernetes," in Proc. 2021 6th International Conference for Convergence in Technology (I2CT), Pune, India, Apr. 2021, pp. 1-5.

[8] P. Varalakshmi, B. Guhan, P. V. Siva, T. Dhanush, and K. Saktheeswaran, "Improvising JSON Web Token Authentication in SDN," in Proc. 2022 International Conference on Communication, Computing and Internet of Things (IC3IoT), Mar. 2022. DOI: 10.1109/IC3IOT53935.2022.9767873.

[9] A. Bucko, K. Vishi, B. Krasniqi, and B. Rexha, "Enhancing JWT Authentication and Authorization in Web Applications Based on User Behavior History," Computers, vol. 12, no. 4, pp. 78-85, 2023. DOI: https://doi.org/10.3390/computers12040078.

[10] L. V. Jánoky, J. Levendovszky, and P. Ekler, "An Analysis on the Revoking Mechanisms for JSON Web Tokens," International Journal of Distributed Sensor Networks, vol. 14, no. 9, Aug. 2018. DOI: 10.1177/1550147718801535.

[11] R. Achary and C. J. Shelke, "Fraud Detection in Banking Transactions Using Machine Learning," in Proc. 2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), Jan. 2023.

[12] D. Prusti and S. K. Rath, "Fraudulent Transaction Detection in Credit Card by Applying Ensemble Machine Learning Techniques," in Proc. Jun 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT),

[13] G. J. Priya and S. Saradha, "Fraud Detection and Prevention Using Machine Learning Algorithms: A Review," in Proc. 2021 7th International Conference on Electrical Energy Systems (ICEES), Feb. 2021. DOI: 10.1109/ICEES51510.2021.9383631.

[14] A. I. Weinberg and K. Cohen, "Zero Trust Implementation in the Emerging Technologies Era: A Survey," Complex Engineering Systems, vol. 4, pp. 16-28, 2024. DOI: http://dx.doi.org/10.20517/ces.2024.41.

[15] H. Kang, G. Liu, Q. Wang, L. Meng, and J. Liu, "Theory and Application of Zero Trust Security: A Brief Survey," Entropy, vol. 25, no. 12, 2023.

[16] S. Ashfaq, S. A. Patil, S. Borde, P. Chandre, and P. M. Shafi, "Zero Trust Security Paradigm: A Comprehensive Survey and Research Analysis," Journal of Electrical Systems, vol. 19, no. 2, pp. 28-37, 2023.

[17] C. Liu, R. Tan, Y. Wu, Y. Feng, Z. Jin, F. Zhang, Y. Liu, and Q. Liu, "Dissecting Zero Trust: Research Landscape and Its Implementation in IoT," Cybersecurity, vol. 7, no. 20, pp. 1-10, 2024. DOI: https://doi.org/10.1186/s42400-024-00212-0.

[18] S. Mehraj and M. T. Banday, "Establishing a Zero Trust Strategy in Cloud Computing Environment," in Proc. 2020 International Conference on Computer Communication and Informatics (ICCCI 2020), Coimbatore, India, Jan. 2020.

[19] R. Muddinagiri, S. Ambavane, and S. Bayas, "Self-Hosted Kubernetes: Deploying Docker Containers Locally with Minikube," in Proc. 2019 International Conference on Innovative Trends and Advances in Engineering and Technology (ICITAET).

[20] M. Pace, Zero Trust Networks with Istio, Master's thesis, Politecnico di Torino, Turin, Italy, 2020.

[21] A. Kurbatov, Design and Implementation of Secure Communication Between Microservices, Master's thesis, Aalto University, Espoo, Finland, 2020.

[22] Y. Yang, W. Shen, B. Ruan, W. Liu, and K. Ren, "Security Challenges in the Container Cloud," in Proc. 2021 Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA), Dec. 2021. DOI: 10.1109/TPS-ISA52974.2021.