iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** VI **Month of publication:** June 2024

DOI: https://doi.org/10.22214/ijraset.2024.63496

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue VI June 2024- Available at www.ijraset.com

Server-Assisted Security Model for Edge Computing: A Packet Tracking Approach

Mamlesh VA, Nishyanth Nandagopal, Mallavarapu Jonathan Vineeth, Sahithi Venkata Pokuri, Varun Venkat Saravanan, K Alwin Infant
Vellore Institute of Technology, India

Abstract: ***In current network systems, many applications run on edge computing devices, but these devices often lack safety and security measures to protect against hackers. To address this issue, we have developed a prototype model that enhances security. Our model connects the edge computing device to a main server via a socket connection. If the edge device detects any threats, it can alert the server immediately.***

Keywords: ***Edge computing, Packet tracking, Hacker protection, Threat detection, Socket Programming***

I. INTRODUCTION

Edge computing has become an increasingly popular technology in recent years, as it enables data processing and analysis at the edge of the network, closer to the source of the data. This approach can significantly reduce latency and improve real-time processing capabilities, making it particularly useful for applications such as IoT, autonomous vehicles, and smart cities. However, the widespread adoption of edge computing has also introduced new security challenges.

Traditional security measures, such as firewalls and intrusion detection systems, are often insufficient for protecting edge computing devices. These devices are frequently exposed to the public internet and are vulnerable to various types of attacks, including Distributed Denial of Service (DDoS) attacks, malware, and unauthorised access. The consequences of a successful attack can be severe, ranging from data breaches and intellectual property theft to disruption of critical infrastructure and even physical harm. In this context, a server-assisted security model utilising packet tracking has shown promise in enhancing network monitoring by using packet sniffers to capture and analyse data, ensuring efficient traffic management and security [1]. By setting network interfaces to promiscuous mode, packet sniffers can listen to all network traffic, providing comprehensive insights for intrusion detection and efficient network data transmission [1].

To address these security concerns, this research paper proposes a novel approach to enhancing the safety and security of edge computing devices. Our approach leverages packet tracking analysis to detect and respond to potential threats in real-time. By analysing network traffic patterns, our system can identify anomalies and suspicious activity, and alert the main server immediately. This allows for swift action to be taken to prevent or mitigate the attack, ensuring the continued operation of critical applications and services. Additionally, a prototype model that connects the edge device to a main server via a socket connection for immediate threat detection and alerting has been proposed [2]. This model leverages network traffic analysis and prediction, utilising various techniques such as neural networks and data mining for proactive network security [2]. By categorising predictions into long-term and short-term, the model can effectively manage dynamic bandwidth allocation, improve Quality of Service (QoS), and optimise resource management [2].

In addition to proposing a novel approach to bolstering the safety and security of edge computing devices, this research paper introduces eight distinct threat detection algorithms. These algorithms encompass a comprehensive range of potential security risks encountered by edge devices. They include detection mechanisms for unsecured storage of passwords (Plaintext Passwords), unauthorised access or security breaches (Intrusions and Breaches), exposure of Personally Identifiable Information (PII Leak), introduction of malicious software (Malware Infections), anomalies in communication protocols (Protocol Abnormalities), errors in network setup leading to vulnerabilities (Network Misconfigurations), unauthorised extraction of data (Data Exfiltration), and deliberate disruption of services through overwhelming systems (Denial of Service - DoS). Each algorithm contributes to a robust security framework aimed at safeguarding edge computing environments against evolving threats.

Furthermore, the increasing proliferation of IoE and smart devices has highlighted the necessity for effective security measures at the edge [4]. As edge computing processes data closer to its source, it reduces cloud dependency and enhances real-time performance, but also presents significant security vulnerabilities [4].

A comprehensive survey of key attacks, including DDoS, side-channel, malware injection, and authentication attacks, along with proposed defence mechanisms, provides an in-depth analysis of root causes, current security challenges, and future research directions in edge computing security [5]. By optimising feature selection methods, anomaly detection can be significantly improved, which is crucial for identifying novel attacks, misconfigurations, and network failures [3].

II. REVIEW

In the current landscape of network systems, edge computing devices are increasingly integral to various applications but often lack robust security measures, making them vulnerable to a range of cyber threats. Our paper addresses this critical issue by proposing a prototype model that enhances the security of edge computing devices through real-time threat detection and immediate alerting to a main server via a socket connection.

Primary security threats in edge computing, such as distributed denial of service (DDoS), side-channel attacks, malware injection, and authentication and authorization attacks, are extensively discussed in [6]. This comprehensive survey underscores the necessity of robust defense mechanisms, complementing our approach of real-time packet tracking and threat detection algorithms. Similarly, [5] and [4] provide thorough surveys of edge-based security solutions for IoT applications, addressing significant challenges such as resource constraints and insufficient security design. They review various security measures like firewalls, intrusion detection systems, and privacy-preserving mechanisms, emphasising the importance of robust security architectures in edge computing contexts.

The critical need for effective privacy-preserving methods in resource-constrained edge environments is highlighted in [7], which focuses on differential privacy (DP) in edge computing for smart city applications. This study aligns with our project's focus on enhancing security and real-time threat detection in edge computing. Furthermore, [16] presents an extensive survey on security and privacy issues within the EC-assisted IoT paradigm, providing a comprehensive overview of EC-assisted IoT, including its applications, architecture, advantages, and challenges. This paper reinforces the significance of our prototype model in addressing unique security risks introduced by this integration.

Security concerns in Edge and Fog Computing, particularly vulnerabilities in virtualization technologies like Containers, Real-Time Operating Systems (RTOS), and Unikernels, are emphasised in [8]. This highlights the current technology limitations and various attacks targeting these virtualization technologies, reinforcing the need for robust security measures and future research directions.

The role of Deep Packet Inspection (DPI) in network security and traffic analysis, as discussed in [9] and [24], is crucial for our model. The performance-limiting factors in DPI implementation and its integration with other network analysis techniques are essential for handling complex network behaviours. DPI applications in IoT and SDN architectures, especially its importance in Intrusion Detection Systems (IDS), and the challenges of inspecting encrypted traffic, are particularly relevant to our approach. The significance of tools like libtrace for simplifying trace analysis development and enhancing performance further supports our model's focus on real-time packet analysis for threat detection.

A novel approach combining Packet Capture (PCAP) analysis with Social Network Analysis (SNA) is proposed in [10] to overcome the limitations of traditional log data in network forensic analysis. This method enhances the reconstruction of communication patterns and identification of advanced security threats, aligning with our goal of developing comprehensive threat detection mechanisms. Similarly, [18] and [26] emphasise the increasing necessity for network monitoring tools due to the widespread use of computers and the internet, highlighting their role in diagnosing network issues and aiding in law enforcement activities by analysing data flow to detect and prevent crime. They explore packet analysis as a critical technique in network forensics, emphasising its role in detecting various network security threats such as data breaches, malware infections, and unauthorised access. In-depth analysis of packet sniffer tools like Wireshark, Ethereal, and TCP Dump is provided in [11], [12], and [22], emphasising their importance in collecting user data and analysing network performance, particularly in wireless communication technologies. These papers underscore the critical role of packet sniffing tools in enhancing network security and performance analysis, which is central to our project. They discuss the capabilities of tools in capturing and analysing network packets, highlighting their roles in monitoring network bottlenecks, detecting irregular behaviours, and capturing sensitive data such as passwords and VoIP transmissions.

The deployment of Intrusion Detection Systems (IDS) in IoT networks, with a focus on anomaly-based detection methods, is explored in [13]. The architectural aspects and challenges of deploying IDS in IoT environments are particularly relevant to our project, emphasising the importance of robust IDS solutions tailored to IoT networks. Complementing this, [20] introduces Tranalyzer 2 (T2), a robust flow-based traffic analyzer designed to address the complexities and challenges of modern IP-based networks, demonstrating its effectiveness in detecting anomalies and troubleshooting network issues.

RAPID, a real-time alert investigation system, is introduced in [14]. It addresses challenges in alert triage by improving space and time efficiency through task deduplication and dynamic prioritisation, providing a concurrent alert investigation platform with provenance analysis capabilities. This approach is relevant to our model's focus on efficient and timely threat detection.

The software-defined perimeter (SDP) framework proposed in [15] enhances security by authorising authenticated users at the edge to access cloud services. Its implementation within a mobile-edge LTE network and resilience testing against Denial-of-Service (DoS) attacks are pertinent to our goal of secure edge computing environments.

The innovative approach to federated learning (FL) leveraging edge computing in [17] alleviates the computational burden on mobile devices. By offloading part of the computation to edge servers, it reduces the need for extensive calculations on mobile devices and minimises global communication frequency, aligning with our focus on efficient security measures.

A framework for real-time wavelet-based analysis of network traffic anomalies in IDS, is introduced in [19]. Unlike traditional IDSs that primarily use signature or pattern matching techniques, Waveman applies signal processing techniques to detect unknown anomalous activities, which is crucial for our real-time threat detection approach.

The comprehensive survey on packet analysis and deep packet inspection in network forensics in [21] details AI-powered methods for advanced traffic classification and pattern identification. The review covers both hardware and software packet analyzers, emphasising their forensic capabilities and the admissibility of digital evidence in court, essential for our model's focus on robust network security measures.

Flow monitoring, integrating Deep Packet Inspection (DPI) for comprehensive traffic analysis, is discussed in [23]. IPFIX, serving as a versatile transport protocol for structured data, is particularly relevant for analysing data from various sensor devices in IoT environments, aligning with our project's goals.

Demonstrates how eliminating redundant and irrelevant features in a dataset can maintain detection performance while minimising computational overhead, emphasising resource optimization in securing network systems [3]. This practical benefit is critical for the efficiency of our security measures.

III. RESEARCH

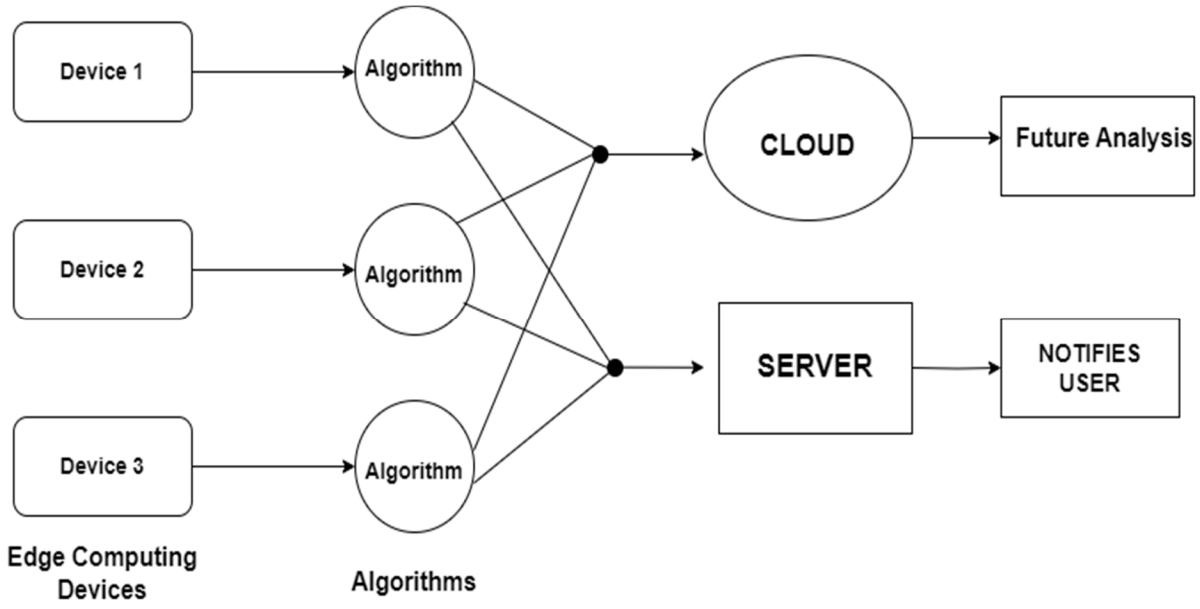


Fig 1: Architecture Diagram of Packer Tracking Analyser Tool

In the world of edge computing, many devices use Linux OS, which is open-source and doesn’t have a lot of built-in security features. These devices often connect to public networks at homes, workplaces, and other places, making them vulnerable to security threats. To solve this problem, we developed a packet tracking and analysis tool to improve the security of these devices. This tool uses eight unique threat detection algorithms to provide extra protection.

We built our tool from scratch using Python, incorporating a user-friendly interface created with Tkinter to display network packets in real time. The core functionality relies on the Scapy module, a powerful networking library in Python, which captures packets as they arrive. The interface provides a comprehensive view of important packet details, including IP addresses, port numbers, packet types (TCP or UDP), and destination addresses. This real-time display allows users to monitor network activity effectively. Furthermore, if a packet's message is not encrypted, our tool has the capability to show the message content directly within the interface, offering immediate insights into potential security issues. The combination of Scapy's robust packet handling and Tkinter's intuitive user interface ensures that our tool is both powerful and accessible, enabling users to maintain a secure network environment with ease.

Our tool's real-time monitoring capabilities and detailed packet information are crucial for identifying and mitigating security threats. By continuously analysing network traffic, the tool can detect unusual patterns and potential threats through its built-in algorithms. When it identifies suspicious activities, it promptly alerts users, enabling them to take immediate action. This proactive approach is essential for maintaining the security of edge computing devices, especially on unsecured networks. By providing instant visibility into network activity and potential threats, our tool helps ensure that edge devices remain protected against unauthorised access, data breaches, malware infections, and other cyber threats. This continuous vigilance and rapid response capability are key to maintaining a secure network environment in real-time.

Our packet tracking tool captures packets in real-time and stores them in Firebase as text files for future analysis, enabling us to review past network activities and identify any missed threats. Firebase, a cloud computing service, allows us to securely store the captured packets, ensuring that all relevant information is preserved. Each text file contains detailed packet information, including the TCP or UDP protocol type, source and destination port numbers, source and destination IP addresses, the number of bytes transferred, and the message content. This comprehensive data collection facilitates thorough future analysis, helping us to detect patterns and potential security threats that may have been overlooked during initial real-time monitoring. By leveraging Firebase's cloud storage capabilities, we ensure that all network activity data is easily accessible and organised for effective retrospective security analysis.

The eight unique threat detection algorithms we developed run in parallel using threading, ensuring that each algorithm operates independently without interference. Plaintext passwords pose a significant threat as they are stored without encryption, making them easy targets for attackers. Our program detects unencrypted passwords in packets, alerting users to secure them properly to prevent unauthorised access. Intrusions and breaches, which involve unauthorised access or security breaches, can compromise sensitive data. Our tool monitors network traffic for signs of intrusions, such as unusual login attempts or access from unknown IPs, helping to prevent breaches. The exposure of Personally Identifiable Information (PII) can lead to identity theft and privacy violations. Our tool scans for PII in network packets and flags any leaks, prompting immediate action to protect user privacy. Malware infections, caused by malicious software, can disrupt operations and steal data. Our tool analyses packet content for known malware signatures and suspicious behaviour, providing early detection and prevention of infections. Protocol abnormalities, which are anomalies in communication protocols, can indicate misconfigurations or malicious activity. Our tool checks for deviations from standard protocols, helping to maintain proper communication standards and detect potential threats. Network misconfigurations, such as errors in network setup, can create vulnerabilities. Our tool identifies misconfigurations in the network, such as incorrect IP settings or open ports, enabling timely corrections to enhance security. Data exfiltration, which involves the unauthorised extraction of data, can lead to significant losses. Our tool monitors for large or unusual data transfers, detecting and alerting users to potential data exfiltration attempts. Finally, denial of service (DoS) attacks, which are deliberate disruptions of services by overwhelming systems, can halt operations. Our tool detects abnormal spikes in traffic that may indicate a DoS attack, allowing for a quick response to mitigate the impact.

Resolving these threats is crucial for maintaining the security and integrity of edge computing devices. Our tool offers robust real-time monitoring and detection capabilities, which are essential for identifying and addressing potential security threats as they arise. By continuously analysing network traffic and capturing detailed packet information, the tool can detect unusual patterns and suspicious activities that may indicate security breaches or other malicious actions. This enables users to take prompt and informed action to mitigate these threats, ensuring that their edge computing devices remain secure and operational. The real-time alerts and comprehensive data logging provided by our tool help users to swiftly respond to any detected threats, maintaining the overall safety and functionality of their network environment. By proactively managing security threats, our tool plays a vital role in protecting sensitive data and maintaining the integrity of edge computing systems.

To ensure timely threat detection and response, our tool implements a robust 10-second timeout mechanism for each of the eight threat detection algorithms. This design allows the program to perform iterative checks every 10 seconds, continuously monitoring for any detected threats. During each iteration, if a threat is identified within the designated timeframe, the system promptly sends a notification message to the centralised server. This approach ensures swift communication of potential security issues, facilitating immediate actions to mitigate risks and enhance network security. In the absence of detected threats within the 10-second window, the program seamlessly continues its iterations, maintaining constant vigilance over network activities. This proactive monitoring strategy significantly enhances our ability to detect and respond to threats promptly, thereby bolstering overall security measures and minimising potential damage from security breaches.

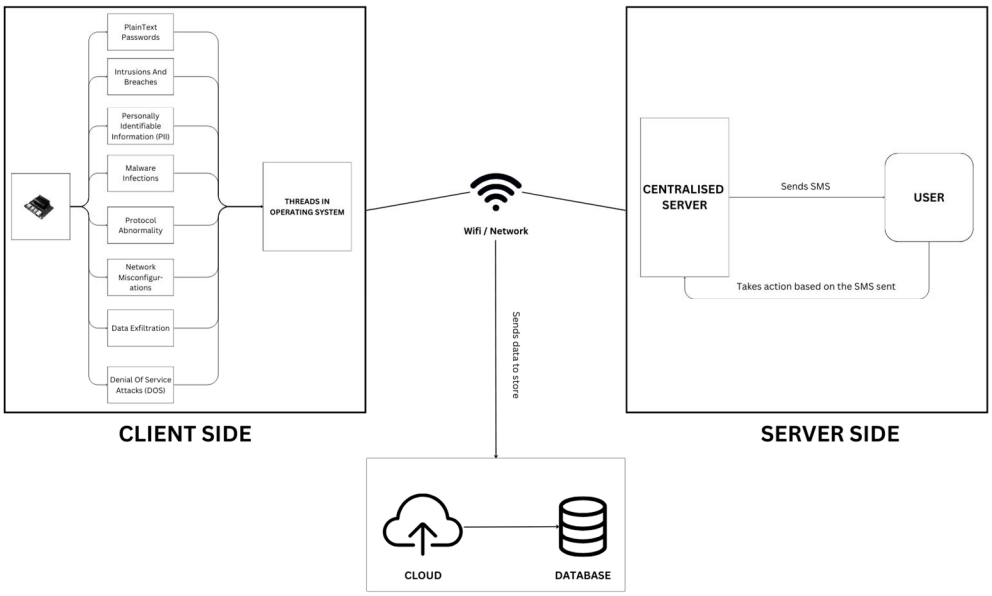


Fig 2: Detailed View of Tool

On the server side, we employ socket programming to establish robust communication channels between the edge computing devices and the centralised server. The server's static IP address and port number serve as the connection points, allowing edge devices to initiate socket connections for sending and receiving critical messages. By utilising a defined TCP protocol, we ensure that communication is both reliable and orderly, maintaining the integrity of data transmission between the server and the clients. To efficiently manage multiple connections concurrently, the server leverages threading, enabling it to handle incoming messages from various edge devices simultaneously. This multithreading capability ensures that the server can efficiently route messages between clients without delay, maintaining seamless communication and facilitating effective threat response across the network. By implementing these measures, we enhance the overall resilience and responsiveness of our threat detection system, ensuring that all edge devices remain in constant, synchronised communication with the centralised server.

To ensure seamless deployment and operation across all edge computing devices, we dockerized the entire program, encapsulating it into a containerized environment. Dockerization allows for consistent execution across diverse hardware configurations, simplifying deployment and management. With the program packaged as a Docker container, we deployed it onto each edge computing device effortlessly. Leveraging the container's isolated runtime environment, we ensured that the program runs consistently and reliably on each device, regardless of underlying system dependencies.

By configuring the Docker container to run as a backend process, we ensured that the program initiates automatically upon device boot-up, thus providing continuous threat monitoring without requiring manual intervention. This automated setup not only optimises resource utilisation but also guarantees persistent threat detection capabilities across all devices, enhancing overall network security.

The program's integration with the centralised server facilitates seamless communication and efficient threat reporting. Utilising socket programming and defined TCP protocols, the program reliably communicates detected threats to the server, enabling centralised monitoring and prompt response. This cohesive system ensures that all edge devices remain in constant communication with the central server, providing a robust and efficient framework for threat detection and response across the entire network.

The utilisation of static IP addresses eliminates the need for frequent configuration changes, significantly simplifying network management and enhancing system stability. By assigning static IP addresses to each edge device, we ensure a consistent and reliable connection to the centralised server, reducing the complexity of network configurations. Each edge device maintains a persistent socket connection with the server, allowing real-time threat notifications to be promptly relayed to users. This setup ensures that security incidents are communicated without delay, enabling timely awareness and response. The stability provided by static IP addresses, combined with the persistent socket connections, enhances the overall efficiency and reliability of our threat detection and response system.

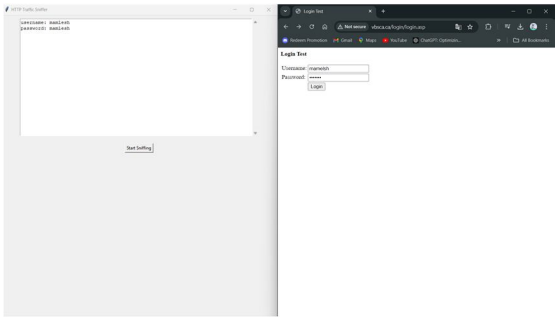
Table 1: List of Threat and How it Works

Threats	Description	Prevention	Detection	Result
Plaintext passwords	Storing passwords in readable format, vulnerable to unauthorised access.	Use strong, hashed passwords and implement multi-factor authentication (MFA).	This function identifies potential threats by monitoring SMTP traffic for instances where authentication credentials are sent in plaintext, specifically capturing payload data associated with the "AUTH LOGIN" command.	sends alerts to a server (server_address, server_port) regarding detected protocol abnormalities.
Intrusions and breaches	Unauthorised access into systems or networks, compromising security.	Regularly update software, employ intrusion detection systems (IDS), and conduct security audits.	This algorithm detects potential threats by identifying destination IPs that engage in scanning multiple ports, indicative of reconnaissance efforts typical before more severe security breaches.	sends alerts to a server (server_address, server_port) regarding detected protocol abnormalities.
Personally Identifiable Information (PII)	Personal data (e.g., SSN, DOB) exposed, risking identity theft.	Encrypt PII both at rest and in transit, limit access to necessary personnel only.	This algorithm identifies potential PII leaks by searching packet payloads for specific keywords associated with sensitive information, indicating potential data breaches or leaks.	sends alerts to a server (server_address, server_port) regarding detected protocol abnormalities.

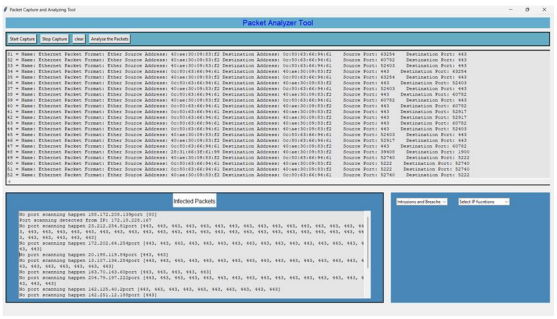
Malware infections	Software designed to harm or exploit systems, often covertly installed.	Install reputable antivirus software, keep systems patched, and educate users on safe browsing habits.	This algorithm identifies potential threats by flagging ports that are used less frequently than expected in network traffic, which may indicate unusual or unauthorised communication patterns.	sends alerts to a server (server_address, server_port) regarding detected protocol abnormalities.
Protocol abnormalities	Deviations from standard communication protocols, indicating potential attacks.	Monitor network traffic for anomalies, implement protocol-specific security measures (e.g., TLS for web traffic).	detect protocol abnormalities by comparing the observed frequency of each protocol with the expected frequency	sends alerts to a server (server_address, server_port) regarding detected protocol abnormalities.
Network misconfigurations	Incorrect network settings leading to vulnerabilities or operational issues.	Regularly audit network configurations, use automated tools for validation, and follow security best practices.	This algorithm identifies potential threats by inspecting network packets for insecure protocols (e.g., HTTP) and sensitive data keywords within packet payloads, alerting about them if found.	sends alerts to a server (server_address, server_port) regarding detected protocol abnormalities.
Data exfiltration	Unauthorised transfer of data outside of a network, often for malicious purposes.	Implement data loss prevention (DLP) tools, monitor outbound network traffic, and encrypt sensitive data.	This algorithm detects potential threats by identifying unusually large packets, suggesting possible data exfiltration activities occurring over the network.	sends alerts to a server (server_address, server_port) regarding detected protocol abnormalities.

denial of service (DoS) attacks	Overwhelming a network or system with traffic, rendering it unavailable.	Use DoS protection services, configure firewalls to filter out malicious traffic, and employ load balancers for traffic management.	This algorithm identifies potential DoS attacks by monitoring packet counts and identifying excessive requests from individual source IP and port combinations, indicative of a distributed or local DoS attack.	sends alerts to a server (server_address, server_port) regarding detected protocol abnormalities.

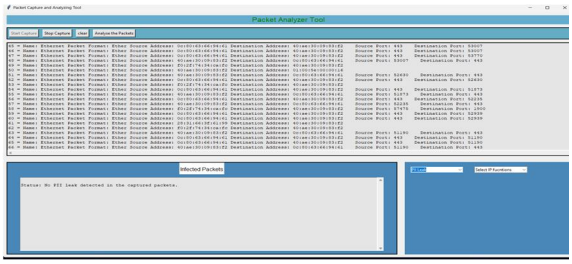
IV. RESULT



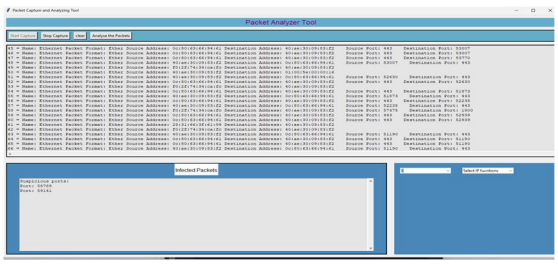
(a)



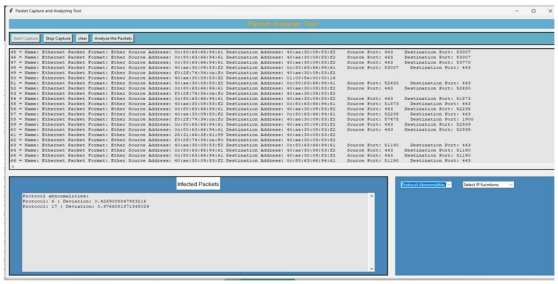
(b)



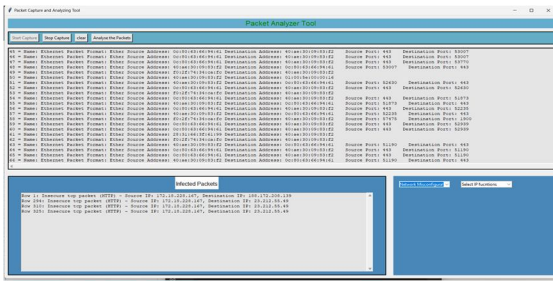
(c)



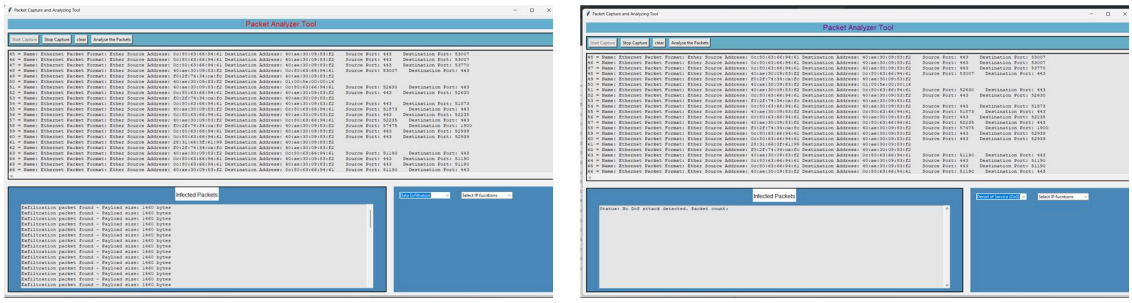
(d)



(e)



(f)



(g) (h)

Fig 3: Results of Each Threat Detection Algorithms (a) Plaintext passwords, (b) Intrusions and breaches, (c) Personally Identifiable Information (PII), (d) Malware infections, (e) Protocol abnormalities, (f) Network misconfigurations, (g) Data exfiltration, (h) Denial of Service (DoS) attacks

The results presented above are derived from a pcap file, showcasing our comprehensive approach to threat detection. Our system employs both real-time threat detection algorithms and post-capture analysis using Wireshark pcap files. This dual approach ensures robust monitoring and analysis of network activities.

In real-time, packets are captured and stored in Firebase, a scalable cloud-based database. To facilitate efficient threat detection, we developed an algorithm to retrieve these packets from Firebase and analyze them for potential threats. This retrieval process ensures that our system can promptly identify and respond to threats as they occur.

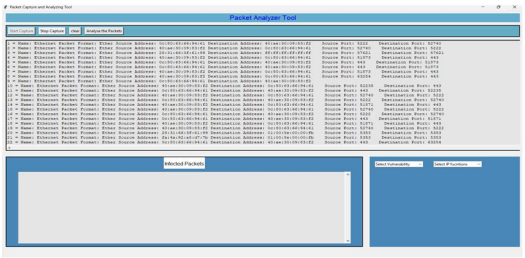


Fig 4: Real Time Packet Capture

We process network data stored in text files by slicing through the contents to extract the relevant information. This extracted data is then fed into our threat detection algorithms, enabling thorough analysis and identification of potential security issues. By combining real-time packet analysis with post-capture examination, our system provides a comprehensive solution for detecting and mitigating threats, ensuring the security and integrity of network activities. On the server side, we employ multithreading to manage multiple client connections simultaneously. This capability ensures that no matter how many clients are connected, the server can efficiently handle each one by assigning a unique identifier to every client. This unique number allows the server to easily identify and track individual clients. When a threat is detected, the server can display the unique identifier along with the details of the detected threat, ensuring clear and precise monitoring. Both the server and clients are connected to the same network, facilitating seamless data transfer between them. This network connectivity ensures that communication remains efficient and reliable, enabling the server to receive and analyze data from each client in real-time. Upon detection of a threat, the system is designed to automatically notify the user. This automated notification process enhances the responsiveness of our threat detection system, ensuring that users are promptly informed of any security issues without requiring manual intervention. By integrating these features, we have created a robust and efficient framework for continuous threat monitoring and rapid response, providing comprehensive security across all connected devices.

V. CONCLUSION

In conclusion, our research presents a significant step forward in addressing the pressing security concerns surrounding edge computing devices. By leveraging packet tracking analysis and introducing eight distinct threat detection algorithms, we have developed a prototype model capable of swiftly identifying and responding to potential security threats in real-time.

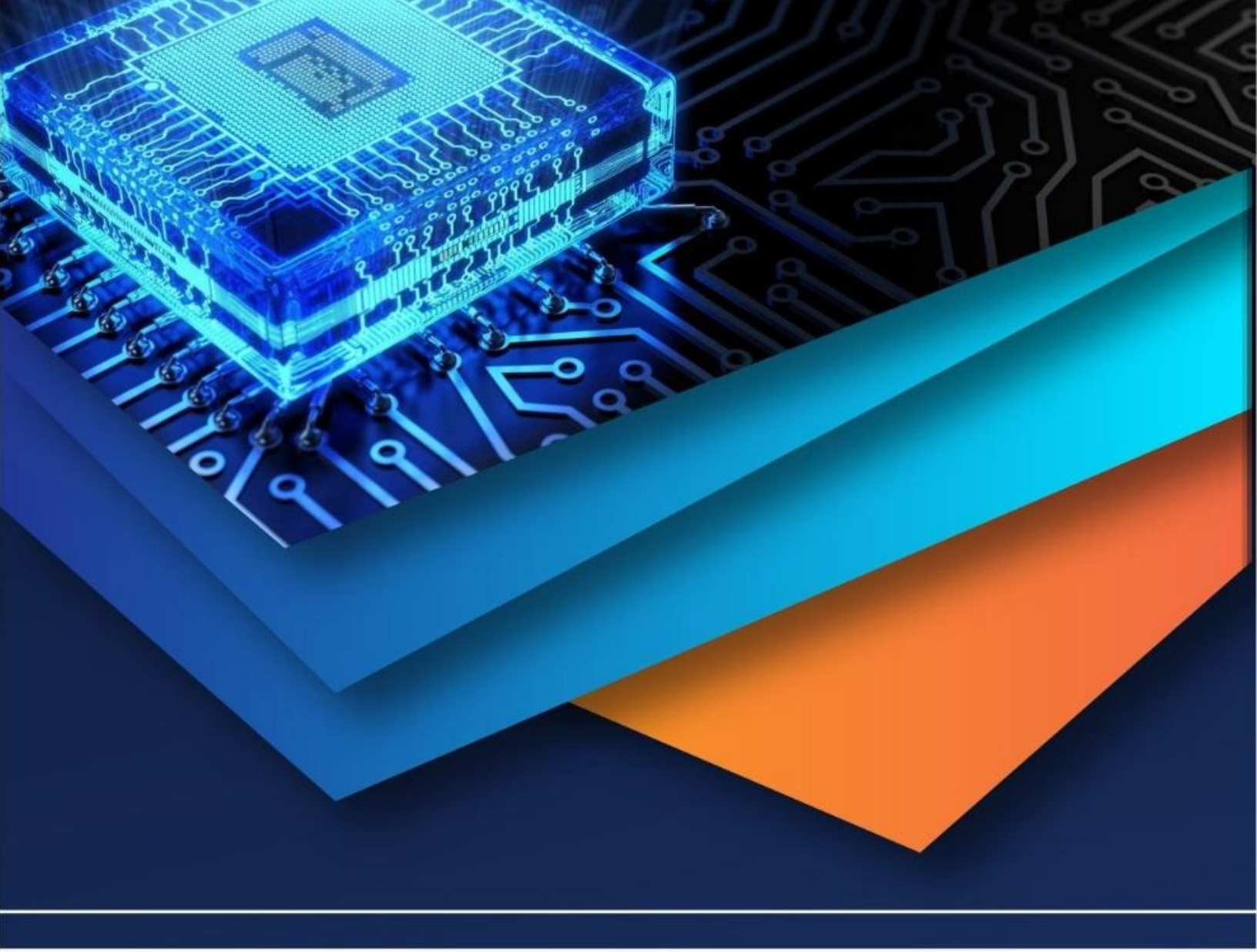
Our testing on several edge computing devices demonstrates the practicality and effectiveness of our approach. In the event of detecting a threat, our model efficiently communicates with a centralised server or device, providing detailed information about the identified threat and enabling prompt action to mitigate its impact. This proactive approach not only enhances the safety and security of edge computing environments but also empowers users to better protect their critical applications and services. As the adoption of edge computing continues to expand, our research lays a foundation for building resilient security frameworks that can adapt to the evolving threat landscape, ensuring the continued reliability and integrity of edge computing systems.

VI. ACKNOWLEDGMENT

I would like to express my heartfelt gratitude to all those who have taken the time to read and review this paper. Your valuable insights and feedback have significantly contributed to its quality and completeness. Thank you for your time and support.

REFERENCES

[1] Asrodia, P., & Patel, H. (2012). Network traffic analysis using packet sniffer. International journal of engineering research and applications, 2(3), 854-856.
[2] Joshi, M., & Hadi, T. H. (2015). A review of network traffic analysis and prediction techniques. arXiv preprint arXiv:1507.05722.
[3] Iglesias, F., & Zseby, T. (2015). Analysis of network traffic features for anomaly detection. Machine Learning, 101, 59-84.
[4] Cao, K., Liu, Y., Meng, G., & Sun, Q. (2020). An overview on edge computing research. IEEE access, 8, 85714-85728.
[5] Sha, K., Yang, T. A., Wei, W., & Davari, S. (2020). A survey of edge computing-based designs for IoT security. Digital Communications and Networks, 6(2), 195-202.
[6] Xiao, Y., Jia, Y., Liu, C., Cheng, X., Yu, J., & Lv, W. (2019). Edge computing security: State of the art and challenges. Proceedings of the IEEE, 107(8), 1608-1631.
[7] Yao, A., Li, G., Li, X., Jiang, F., Xu, J., & Liu, X. (2023). Differential privacy in edge computing-based smart city Applications: Security issues, solutions and future directions. Array, 100293.
[8] Caprolu, M., Di Pietro, R., Lombardi, F., & Raponi, S. (2019, July). Edge computing perspectives: Architectures, technologies, and open security issues. In 2019 IEEE International Conference on Edge Computing (EDGE) (pp. 116-123). IEEE.
[9] Çelebi, M., Özbilen, A., & Yavanoğlu, U. (2023). A comprehensive survey on deep packet inspection for advanced network traffic analysis: issues and challenges. Niğde Ömer Halisdemir Üniversitesi Mühendislik Bilimleri Dergisi, 12(1), 1-29.
[10] Sembiring, I., Iriani, A., Ginting, J. V. B., & Ginting, J. A. (2023). A Novel Approach to Network Forensic Analysis: Combining Packet Capture Data and Social Network Analysis. International Journal of Advanced Computer Science and Applications, 14(3).
[11] Pandit, Pooja. (2021). A Study of Packet Sniffer Tools. 10.13140/RG.2.2.31223.34729.
[12] Qadeer, M. A., Iqbal, A., Zahid, M., & Siddiqui, M. R. (2010, February). Network traffic analysis and intrusion detection using packet sniffer. In 2010 Second International Conference on Communication Software and Networks (pp. 313-317). IEEE.
[13] Spadaccino, P., & Cuomo, F. (2020). Intrusion Detection Systems for IoT: opportunities and challenges offered by Edge Computing and Machine Learning. arXiv preprint arXiv:2012.01174.
[14] Liu, Y., Shu, X., Sun, Y., Jang, J., & Mittal, P. (2022, December). RAPID: Real-Time Alert Investigation with Context-aware Prioritization for Efficient Threat Discovery. In Proceedings of the 38th Annual Computer Security Applications Conference (pp. 827-840).
[15] Singh, J., Bello, Y., Hussein, A. R., Erbad, A., & Mohamed, A. (2020). Hierarchical security paradigm for iot multiaccess edge computing. IEEE Internet of Things Journal, 8(7), 5794-5805.
[16] Alwarafy, A., Al-Thelaya, K. A., Abdallah, M., Schneider, J., & Hamdi, M. (2020). A survey on security and privacy issues in edge-computing-assisted internet of things. IEEE Internet of Things Journal, 8(6), 4004-4022.
[17] Ye, Y., Li, S., Liu, F., Tang, Y., & Hu, W. (2020). EdgeFed: Optimized federated learning based on edge computing. IEEE Access, 8, 209191-209198.
[18] Dodiya, B., & Singh, U. K. (2022). Malicious Traffic analysis using Wireshark by collection of Indicators of Compromise. International Journal of Computer Applications, 183(53), 1-6.
[19] Huang, C. T., Thareja, S., & Shin, Y. J. (2006, August). Wavelet-based real time detection of network traffic anomalies. In 2006 securecomm and workshops (pp. 1-7). IEEE.
[20] Burschka, S., & Dupasquier, B. (2016, December). Tranalyzer: Versatile high performance network traffic analyser. In 2016 IEEE symposium series on computational intelligence (SSCI) (pp. 1-8). IEEE.
[21] Sikos, L. F. (2020). Packet analysis for network forensics: A comprehensive survey. Forensic Science International: Digital Investigation, 32, 200892.
[22] Gandhi, C., Suri, G., Golyan, R. P., Saxena, P., & Saxena, B. K. (2014). Packet sniffer–a comparative study. International Journal of Computer Networks and Communications Security, 2(5), 179-187.
[23] Hofstede, R., Čeleda, P., Trammell, B., Drago, I., Sadre, R., Sperotto, A., & Pras, A. (2014). Flow monitoring explained: From packet capture to data analysis with netflow and ipfix. IEEE Communications Surveys & Tutorials, 16(4), 2037-2064.
[24] Alcock, S., Lorier, P., & Nelson, R. (2012). Libtrace: A packet capture and analysis library. ACM SIGCOMM Computer Communication Review, 42(2), 42-48.
[25] Mraz, L., Komosny, D., Cervenka, V., Moravek, P., & Simek, M. (2011, August). Open-packet analyser platform for wireless sensor networks based on IEEE 802.15. 4. In 2011 34th International Conference on Telecommunications and Signal Processing (TSP) (pp. 145-149). IEEE.
[26] Varanasi, A., & Swathi, P. (2016). Comparative Study of Packet Sniffing tools for HTTP Network Monitoring and Analyzing. International Journal of Science, Engineering and Computer Technology, 6(12), 406.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY