ENHANCING SECURITY IN DISCORD'S SERVER: CREATING ANTI-SPAM & PHISHING DISCORD BOTS

Azri Bin Azmi
University of Kuala Lumpur
Kuala Lumpur, Malaysia
Azri.azmi01@s.unikl.edu.my

Dalilah Abdullah*
Cybersecurity & Technological Convergence (CTC)
University of Kuala Lumpur
Kuala Lumpur, Malaysia
dalilah@unikl.edu.my

*Correspondent author: dalilah@unikl.edu.my

***Abstract*— Online communities are increasingly threatened by spam and phishing attacks, compromising user experience and security. This project develops a specialized Discord bot using Python and Discord API to combat these issues. The bot uses Natural Language Processing (NLP) to detect and mitigate spam, including repetitive messages, unsolicited links, and suspicious user behavior. By analyzing message patterns and leveraging community feedback, the bot adapts and evolves its detection capabilities. Rigorous testing across different Discord servers assessed the bot's effectiveness, focusing on detection accuracy, false positive rates, and overall security impact. The results contribute to robust security measures for online communities, particularly on Discord, and offer insights for combating spam and phishing across digital platforms, fostering a safer online ecosystem.**

***Keywords*— Discord, Anti-Spam, Anti-Phishing, Security Bots, Natural Language Processing**

I. INTRODUCTION

Discord, launched in 2015, is a popular communication platform offering text, audio, and video interactions, widely used by students and online communities. Despite its advantages, Discord faces significant security challenges from spam and phishing attacks that compromise user experience and safety. Spam clutters channels with unsolicited messages, while phishing deceives users into revealing sensitive information. This project aims to enhance Discord server security by developing a specialized bot using Python and Discord's API. The bot utilizes Natural Language Processing (NLP) to detect and mitigate spam and phishing attempts. Through rigorous testing and community feedback, the bot will be refined for high detection accuracy and low false positive rates, contributing to a safer and more secure online environment for Discord users.

A. Background of Problem

Discord has become a widely used communication platform since its establishment in 2015, known for its versatility and cross-platform compatibility. It offers various communication mediums like text, audio, and video, making it popular among students for both formal and informal interactions. Despite its widespread adoption, Discord faces significant security challenges, particularly related to spam and phishing attacks that compromise user experience and safety.

B. Problem Statement

The primary challenge addressed in this project is the vulnerability of Discord accounts to security breaches, which often result in disruptive spam and phishing attacks. These issues not only disrupt communication but also pose serious security risks, particularly for young users and students. The project aims to enhance the security of Discord servers by developing a specialized bot to combat these threats.

C. Objective

1. To gain a comprehensive understanding of the operational intricacies, capabilities, and features of the Discord Bot Framework.
2. To create a proactive anti-spam and anti-phishing bot that blocks spam and harmful links within Discord servers, thereby improving the overall security posture.
3. To rigorously test the bot's performance in real-world scenarios, ensuring it effectively detects and mitigates spam and phishing attempts.

II. LITERATURE REVIEW

A. Detection of Spam Messages In E-Messaging Platform

Spam messages in electronic messaging platforms are a pervasive issue, causing disruption and security risks. They include unsolicited advertisements, phishing attempts, and other unwanted communications. Efficient detection and mitigation of spam are crucial to maintaining the integrity and usability of these platforms.

B. Detection of Phishing Websites

Phishing websites are malicious sites designed to deceive users into divulging sensitive information, such as usernames, passwords, and credit card details. Detecting these websites is crucial for protecting users from identity theft and financial loss.

C. Whitelisting and Black

Whitelisting and blacklisting are essential techniques used in cybersecurity to manage and control access to systems, networks, and applications. These methods help in preventing unauthorized access and protecting against malicious activities.

D. Spam Characteristic

Spam refers to unsolicited and often repetitive messages sent in bulk through electronic communication channels such as email, social media, and messaging platforms. These messages can disrupt communication, consume bandwidth, and pose significant security risks by potentially containing malicious links or attachments.

III. METHODOLOGY

This project will use an Agile approach for quick and effective software development. The process begins with gathering and analyzing requirements through stakeholder meetings and user feedback. Then, we move to designing the bot's architecture and interface using collaborative design tools for dynamic and iterative adjustments.

Next, the project enters the Development phase, where coding and implementation occur in iterative sprints using the chosen development framework. Each sprint focuses on developing specific features, integrating them, and refining them based on continuous feedback. Rigorous testing follows each sprint to identify and fix any issues promptly. To manage the project effectively, we'll use a Work Breakdown Structure (WBS) to divide the project into smaller tasks, aiding resource allocation, time estimation, and task prioritization. A Gantt chart will provide a visual timeline of the project, showing milestones and dependencies for efficient tracking.

Finally, in the Deployment and Response phase, the bot will be released to selected Discord servers, and user feedback will be integrated continuously to improve functionality and performance. By combining Agile principles, WBS, Gantt charts, and continuous user feedback, this project aims for systematic and successful execution, focusing on user-centric design and efficient development..

A. Agile Model

Fig. 1 Agile Model

B. Use Case Diagram

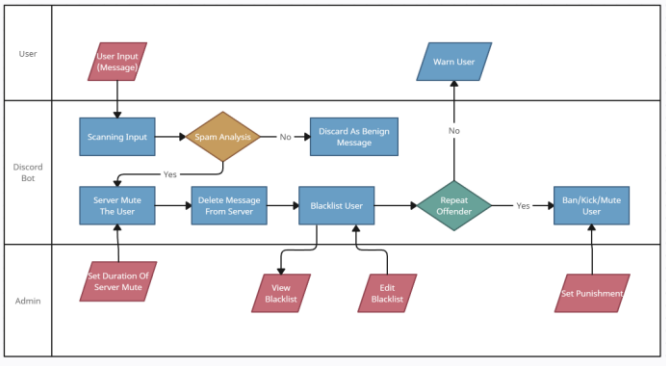

Fig.2 Anti-Spam Swimlane Bot Diagram

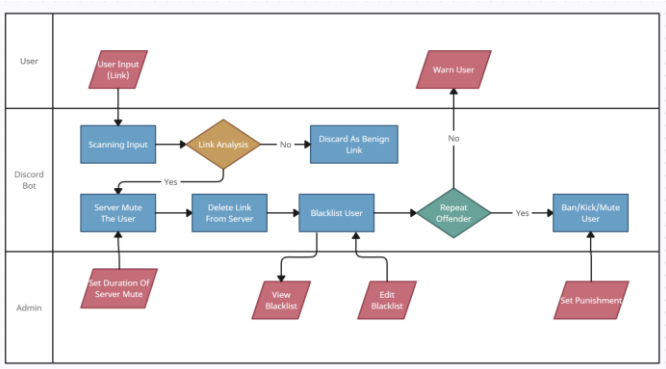

Fig.3 Anti-Phishing Swimlane Bot Diagram

C. Flowchart

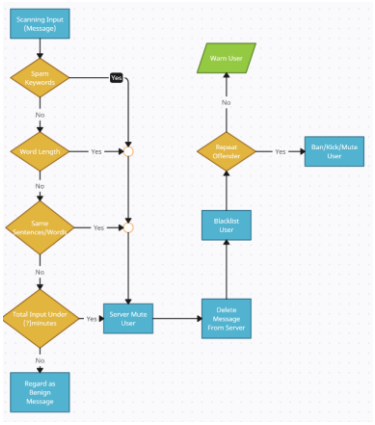

Fig.4 Anti-Spam Bot Flowchart

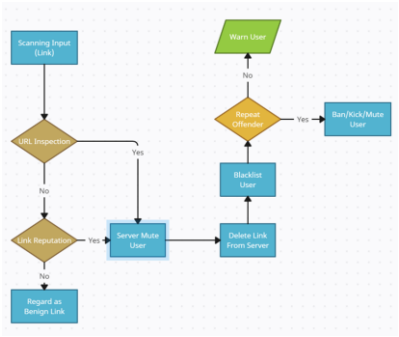

Fig.5 Anti-Phishing Bot Flowchart

D. Prototype Design

1. Discord Server Page

Fig.6 Discord Server Page

2. Discord Status

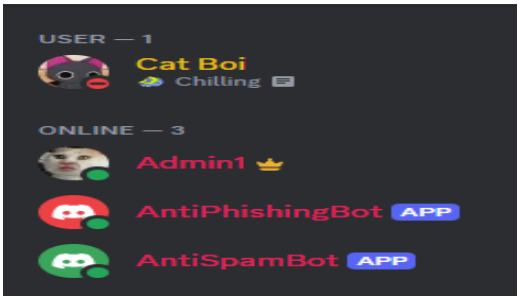

Fig.7 Discord Status

3. Instrument and tools used

In my project, I used several powerful tools to streamline the development and optimization of the anti-spam and anti-phishing Discord bot. Firstly, I used Python, a versatile and widely used programming language, to develop the core functionalities of the bot. Python's extensive libraries and frameworks, especially those for Natural Language Processing (NLP), were essential in implementing effective spam and phishing detection algorithms.

Additionally, I used Visual Studio Code, a free and open-source code editor from Microsoft. This versatile editor supports various programming languages and extensions, providing a customizable environment for coding, debugging, and version control. Features like IntelliSense enhance the coding experience by offering intelligent code completion and suggestions.

For version control and collaboration, I relied on GitHub, integrating changes seamlessly and maintaining a comprehensive history of code revisions. These tools collectively contributed to a streamlined development process, ensuring efficient coding, testing, and deployment of the Discord bot. Their capabilities facilitated a robust, user-centric approach, enhancing the overall functionality and performance of the application.

IV. SYSTEM DEVELOPMENT

This chapter focuses on the detailed development process of mobile applications. It describes the steps taken to develop the platform, including prototyping, user interface and designing the application. The chapter highlights the incorporation of permissions, best practices and features that identified the needs and preferences of the target users.

A. Interface project

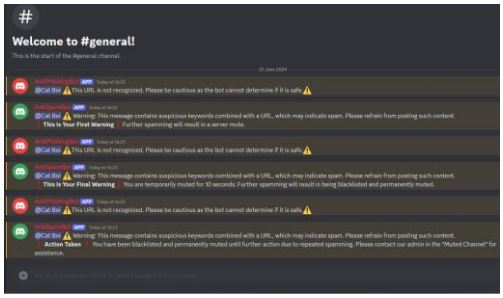

Fig.8 Spam Detection Warning and Server Muting User

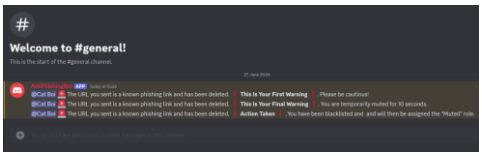

Fig.9 Phishing URL Detection Warning and Server Muting User

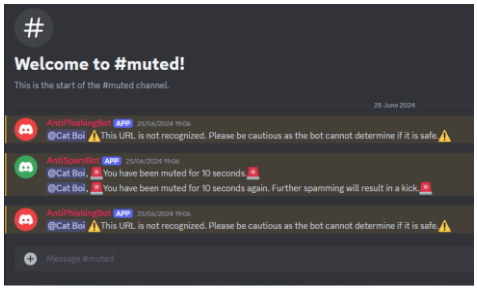

Fig.10 Muted Channel for Blacklisted User

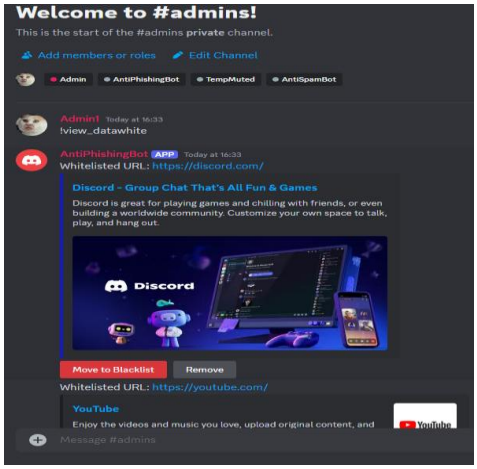

Fig.11 Admin Interface Whitelisted Database Management

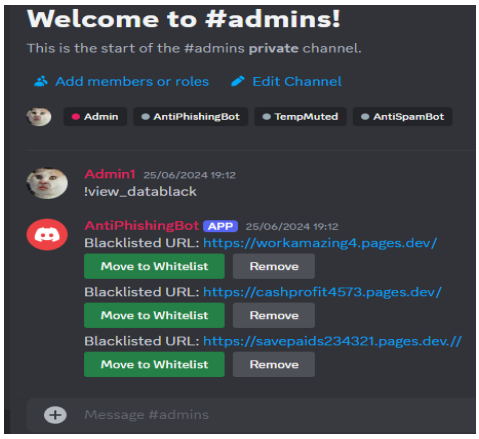

Fig.12 Admin Interface Blacklisted Database Management

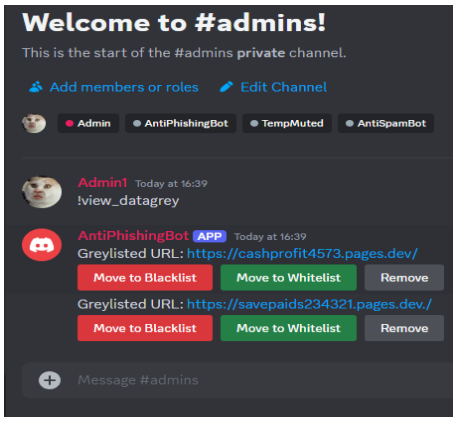

Fig.13 Admin Interface Greylisted Database Management

B. Conclusion

In conclusion, developing the anti-spam and anti-phishing Discord bot involved utilizing various powerful tools and following a structured Agile approach. The project implemented multiple features to enhance user convenience by effectively detecting and mitigating spam and phishing attempts. Moving forward, it is crucial to continuously refine these features and monitor user feedback to ensure the bot remains effective and responsive to evolving threats. By focusing on user-centric design and efficient development practices, the bot can provide a safer and more enjoyable experience for Discord users.

V. IMPLEMENTATION AND TESTING

A. Introduction

During this phase, the focus shifts from design concepts to the actual development and testing of the Discord bot. The goal is to ensure reliability, security, and user-friendliness by translating functional components from design specifications into a fully operational bot. This chapter describes the main tasks, procedures, and considerations involved in implementing and testing the bot.

B. Functional Testing

In this stage, functional testing is crucial to ensure that every part of the Discord bot works as intended and meets the specified requirements. The bot's functional aspects, such as its features, user interfaces, and system interactions, are thoroughly assessed during this testing phase.

Test ID	Description	Expected Outcome	Status
FT1	User Warning and Muting	Users receive warnings and get muted	Passed
FT2	URL Blacklisting	Blacklisted URLs are blocked	Passed
FT3	URL Whitelisting	Whitelisted URLs are allowed	Passed
FT4	URL Greylisting	Greylisted URLs prompt a warning	Passed
FT5	Spam Detection (High Frequency)	High frequency spam messages are detected	Passed
FT6	Spam Detection (Repeated Sentence)	Repeated sentence spam is detected	Passed
FT7	Spam Detection (Spam Keyword)	Messages with spam keywords are detected	Passed
FT8	Spam Detection (Long Words)	Long word spam messages are detected	Passed

Fig.14 User Interactions Functional Testing

The functional testing of user interactions verified the bot's ability to handle spam and manage URLs. Users received appropriate warnings and muting actions for policy violations. The bot effectively identified and managed blacklisted, whitelisted, and greylisted URLs, ensuring safe content. It demonstrated strong spam detection by accurately identifying high-frequency messages, repeated sentences, spam keywords, and long words. Overall, the bot performed as intended, providing a secure and user-friendly experience.

Test ID	Description	Expected Outcome	Status
FT9	URL Management (Whitelist, Greylist, Blacklist)	Admin can manage URLs successfully	Passed
FT10	User Management (Blacklisted User)	Admin can manage blacklisted users successfully	Passed
FT11	Admin Command Restriction	Admin commands are restricted	Passed

Fig. 15 Admin Interactions Functional Testing

The functional testing of admin interactions focused on the bot's capabilities for managing URLs and user restrictions. Admins were able to successfully manage whitelisted, greylisted, and blacklisted URLs, ensuring appropriate content filtering. Additionally, the tests confirmed that admins could effectively manage blacklisted users, enhancing server security. The restriction of admin commands was also verified, ensuring that only authorized actions could be performed. Overall, the admin interaction tests validated the bot's functionality in providing robust management tools for server administrators.

C. Non-Functional Testing

During this phase, non-functional testing ensures that the Discord bot meets the required performance, security, usability, and reliability standards. Non-functional testing encompasses various techniques and methods aimed at evaluating the bot's behavior under different conditions and ensuring it performs well in a real-world environment. This includes performance testing, load testing, stress testing, usability testing, and security testing to identify and mitigate potential vulnerabilities.

Test ID	Description	Expected Outcome	Status
NFT1	Bot Performance	Bot performs efficiently under load	Passed
NFT2	Bot Scalability	Bot scales well with increased load	Passed
NFT3	Bot Reliability	Bot remains reliable over time	Passed
NFT4	Bot Usability	Bot is user-friendly	Passed
NFT5	Bot Security	Bot is secure against vulnerabilities	Passed
NFT6	Bot Compatibility	Bot is compatible with various environments	Passed

Fig.16 Non-Functional Testing

Non-functional testing confirmed that the bot performs efficiently under load, scales well with increased demand, and remains reliable over time. Usability tests showed the bot is user-friendly, while security tests validated its robustness against vulnerabilities. Compatibility tests ensured the bot works seamlessly across different environments. Overall, the bot met all non-functional requirements, demonstrating its effectiveness and reliability in real-world conditions.

D. Conclusion

Ensuring the reliability, security, and usability of the Discord bot involves rigorous testing. Functional testing verifies that all features and user interfaces perform according to specifications, while non-functional testing assesses the bot's performance, security, and usability. By thoroughly testing both functional and non-functional aspects, potential vulnerabilities can be identified and addressed, enhancing the overall quality and trustworthiness of the bot. This chapter has outlined key tasks, procedures, and considerations essential for effectively deploying and testing the bot, emphasizing the importance of comprehensive testing to deliver a robust and secure application to users.

VI. CONCLUSIONS

A. Conclusions

This project has enhanced the security of Discord servers against spam and phishing attacks and increased user awareness about these threats. By implementing effective spam and phishing detection mechanisms using Natural Language Processing (NLP), the bot has significantly strengthened server security. Users were effectively educated on the importance of identifying and reporting suspicious activities. The project addressed common cybersecurity vulnerabilities and implemented robust security features to protect users from potential threats.

B. Recommendations

To maximize the Discord bot's effectiveness in enhancing user security, continuous updates, machine learning integration, and robust data security measures are essential. Regular updates will incorporate user feedback, address new security challenges, and introduce new features, maintaining the bot's effectiveness and user-friendliness. Machine learning can improve spam and phishing detection. Compliance with cybersecurity standards, strong encryption, and regular security audits will ensure a safer digital environment, build user trust, and protect user data. Focusing on these areas will significantly enhance server security and set new standards for transparency and safety in online communities.

VII. REFERENCES

[1] Vulnerability Details :

https://www.cvedetails.com/cve/CVE-2021-29466/

[2] Vulnerability Details : CVE-2020-15278

https://www.cvedetails.com/cve/CVE-2020-15278/

[3] N. Hadi, V. C. Mawardi and J. Hendryli, "Discord Bot Design for Hate Speech Sensor Using Convolutional Neural Networks (CNN) Method," *2023 International Conference on Computer Science, Information Technology and Engineering (ICCoSITE)*, Jakarta, Indonesia, 2023

https://ieeexplore-ieee-org.remotexs.unikl.edu.my/document/10127699

[4] Development, T. Lotlikar, S. Karekar, J. Kazi, S. Khamkar and M. Kulkarni, "The Spiffy – A Discord Chatbot," *2023 5th Biennial International Conference on Nascent Technologies in Engineering (ICNTE)*, Navi Mumbai, India, 2023

https://ieeexplore-ieee-org.remotexs.unikl.edu.my/document/10146616

[5] S. Vinothkumar, S. Varadhaganapathy, R. Shanthakumari, D. Ramkishore, S. Rithik and K. P. Tharanies, "Detection Of Spam Messages In E-Messaging Platform Using Machine Learning," *2022 Fifth International Conference on Computational Intelligence and Communication Technologies (CCICT)*, Sonepat, India, 2022

https://ieeexplore-ieee-org.remotexs.unikl.edu.my/document/9913577

[6] S. Vinothkumar, S. Varadhaganapathy, R. Shanthakumari, D. Ramkishore, S. Rithik and K. P. Tharanies, "Detection Of Spam Messages In E-Messaging Platform Using Machine Learning," *2022 Fifth International Conference on Computational Intelligence and Communication Technologies (CCICT)*, Sonepat, India, 2022

https://ieeexplore-ieee-org.remotexs.unikl.edu.my/document/9913577

[7] R. Zieni, L. Massari and M. C. Calzarossa, "Phishing or Not Phishing? A Survey on the Detection of Phishing Websites," in *IEEE Access*, vol. 11, pp. 18499-18519, 2023

https://ieeexplore-ieee-org.remotexs.unikl.edu.my/document/10049452

[8] eSecurity Planet. (n.d.). Whitelisting vs blacklisting: How are they different? Retrieved from https://www.esecurityplanet.com/products/whitelisting-vs-blacklisting-how-are-they-different/

[9] Reddy, S. K., & Padmaja, T. M. (2019). Non machine and machine learning spam filtering techniques. *International Journal of Recent Technology and Engineering (IJRTE), 7*(5S4), 131-135. Retrieved from https://www.ijrte.org