Archives available at journals.mriindia.com

International Journal on Advanced Computer Engineering and Communication Technology

ISSN: 2278-5140

Volume 15 Issue 01, 2026

PLUTON: A Discord-Based Command-and-Control (C2) Framework with Offensive Simulation and Detection Capabilities

[1]Mrs. A Haseenath Jaseela, [2]Shyam J., [3]Guru Moorthi R. A., [4]Parthiban S., [5]Thomas Allwin D.

[1]M.E., Assistant Professor, Department of Artificial Intelligence & Data Science, Loyola Institute of Technology and Science, Thovalai, Kanniyakumari, Tamil Nadu, India

[2-5] UG Student, Department of Artificial Intelligence & Data Science, Loyola Institute of Technology and Science, Thovalai, Kanniyakumari, Tamil Nadu, India

Peer Review Information

Submission: 08 March 2026

Revision: 26 March 2026

Acceptance: 05 April 2026

Keywords

Discord-based C2, Remote Administration Tool (RAT), Command and Control, Malware Simulation, Cybersecurity Education, Threat Detection, Evasion Techniques, System Monitoring, Remote Command Execution, Defensive Security

Abstract

In modern cybersecurity, Command-and-Control (C2) frameworks play a critical role in remote system administration and the operation of advanced malware. This project, titled PLUTON: A Discord-Based Command-and-Control (C2) Framework with Offensive Simulation and Detection Capabilities, presents a novel approach to designing a C2 system by leveraging the Discord platform as a communication medium in a controlled, safe testing environment. The proposed system integrates both offensive and defensive components, enabling users to simulate real-world cyberattack scenarios while also developing mechanisms for detecting such activities. The framework consists of a Discord bot acting as the C2 server, a client module capable of executing commands, and a builder module that converts Python scripts into executable files with added obfuscation. It supports functionalities such as remote command execution, file management, system monitoring, and multimedia capture. Additionally, the project incorporates detection techniques to identify suspicious behaviours associated with advanced evasion strategies. By combining practical implementation with security analysis, the system serves as an educational tool for understanding modern C2 architectures, malware behaviour, and defensive strategies in cybersecurity research.

Introduction

The PLUTON system is a multi-functional Remote Administration Tool (RAT) developed as a Python-based platform to provide advanced system monitoring and control capabilities. Unlike traditional RAT frameworks, PLUTON integrates multiple modules into a unified architecture, providing enhanced session management, task automation, and system awareness. This integration enables efficient coordination between various functionalities, making the system both flexible and scalable for cybersecurity research and educational purposes.

The primary objective of this system is to design and develop a Discord-based Command and Control (C2) framework that facilitates secure and controlled remote communication. Through this framework, the system demonstrates key capabilities such as remote command execution, file management, system monitoring, and webcam streaming, thereby simulating real-world attacker techniques within a controlled and ethical environment.

In addition to offensive capabilities, the system includes a Builder Module to simplify converting Python scripts into executable files, enabling easier deployment and testing. To further

enhance its research value, PLUTON simulates modern malware evasion techniques, enabling a deeper understanding of how advanced threats bypass traditional security mechanisms.

A significant focus of this work is on defensive cybersecurity. The system includes the development of a detection tool designed to identify suspicious C2 activity, thereby bridging the gap between attack simulation and threat detection. This dual approach supports the creation of an integrated offensive-defence framework aimed at improving cybersecurity education and practical learning.

Furthermore, the project contributes to the development of new C2 detection solutions capable of addressing advanced evasion strategies. By combining both attack simulation and defensive analysis within a single platform, PLUTON serves as a comprehensive tool for studying modern cyber threats and enhancing detection methodologies in a safe and controlled environment.

Methodology

The methodology of the PLUTON system is based on the design and implementation of a Discord-based Command and Control (C2) framework that operates within a controlled and ethical environment. The system follows a modular architecture consisting of a Discord bot acting as the C2 server, client agents deployed on target systems, and supporting modules for execution, monitoring, and detection.

Initially, the Discord bot is configured using the Discord API to receive and process commands issued by the operator. The client module, developed in Python, establishes communication with the bot and continuously listens for incoming instructions using asynchronous event handling techniques [1]. Upon receiving commands, the execution engine processes system-level operations such as file handling, command execution, and data collection.

To ensure efficient data transmission, captured outputs such as screenshots or system information are processed using image compression and serialisation techniques before being transmitted via Discord channels [2]. The Builder Module simplifies deployment by converting Python scripts into executable files with optional obfuscation for simulation purposes.

Additionally, the system incorporates detection mechanisms that monitor suspicious activities such as abnormal API usage and unauthorised command execution. This dual approach enables both offensive simulation and defensive analysis, making the framework suitable for cybersecurity experimentation and education.

Existing System

Traditional Command and Control (C2) frameworks and Remote Administration Tools (RATs) rely heavily on centralised infrastructure such as dedicated servers, static IP addresses, or cloud-hosted platforms. These systems typically communicate using standard protocols such as HTTP/HTTPS or TCP sockets [3].

In such architectures, client machines periodically connect to the server to fetch commands and execute them locally, returning the results through predefined communication channels. While effective, these systems are increasingly monitored and detected by modern security tools due to their predictable communication patterns and identifiable signatures.

Moreover, existing systems do not effectively utilise widely used real-time communication platforms such as Discord, resulting in limited adaptability to modern application-layer communication environments.

Demerits of the Existing System

Existing Command and Control (C2) frameworks exhibit several limitations that reduce their effectiveness in modern cybersecurity environments. These systems primarily rely on signature-based detection techniques, making them ineffective against newly emerging or modified threats [4]. Furthermore, they cannot detect encrypted or covert communication channels such as HTTPS, DNS tunnelling, and application-layer protocols.

Another major limitation is the absence of behavioural analysis, resulting in poor identification of anomalous activities. This leads to increased rates of false positives and false negatives. Additionally, existing systems struggle to adapt to evolving attacker strategies and lack real-time monitoring and alerting mechanisms.

Importantly, these frameworks are not designed to analyse malicious activities occurring within legitimate platforms such as Discord or Slack, which are increasingly being exploited for covert communication.

Proposed System

The proposed PLUTON framework introduces a Discord-based Command and Control (C2) architecture that replaces traditional centralised server infrastructure with a modern communication platform. In this approach, Discord functions as the communication medium, allowing commands and responses to be transmitted through standard messaging channels.

Each client dynamically establishes a communication channel, enabling individualised

interaction between the controller and remote systems. The framework supports multiple functionalities, including remote command execution, file management, system monitoring, screenshot capture, and webcam streaming.
Experimental observations from system implementation demonstrate effective real time communication and execution capabilities. The Builder Module further enhances usability by converting Python scripts into executable files, enabling simplified deployment. Additionally, the framework integrates detection mechanisms that monitor suspicious behavioural patterns, thereby bridging offensive simulation with defensive cybersecurity analysis.

Merits of the Proposed System

The proposed system offers several advantages:

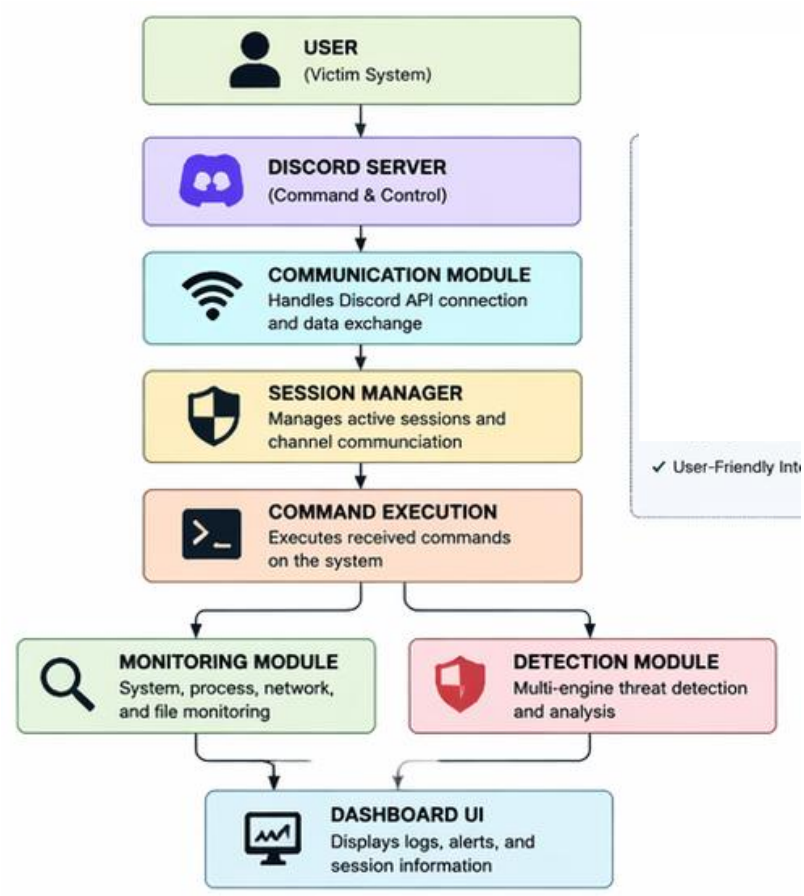
- Adaptive communication using Discord
- Reducing detectability
- Asynchronous execution for handling multiple operations simultaneously [1]
- Modular architecture enabling scalability and flexibility
- Real-time monitoring and command execution
- Integrated logging for tracking activities
- Secure and efficient data transmission
- Failover mechanisms for maintaining connectivity
- Low resource consumption with continuous background operation

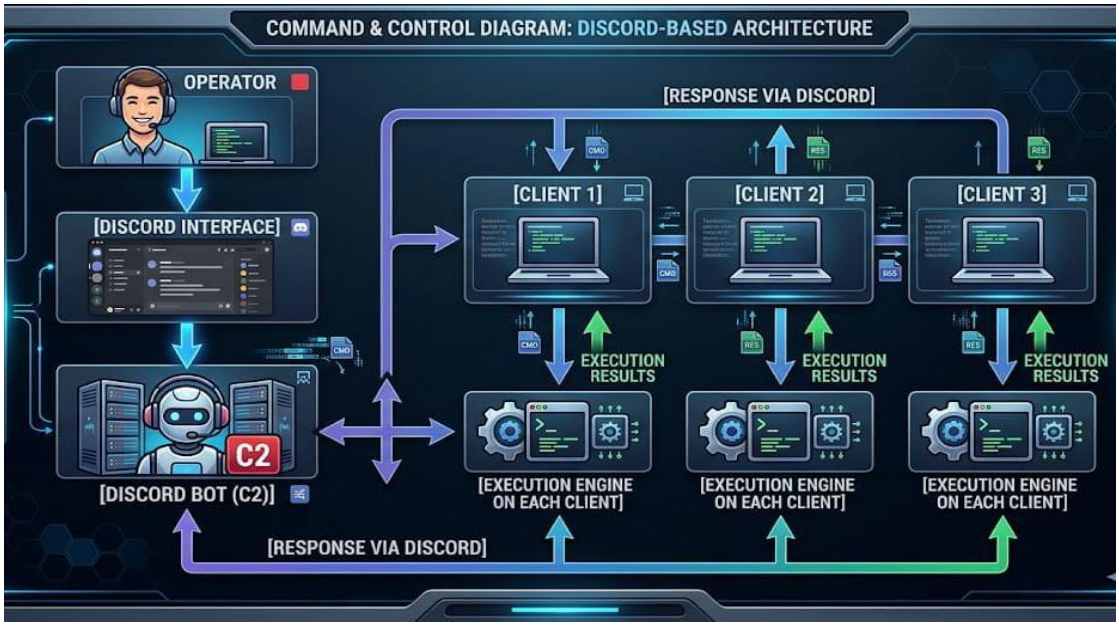
System Architecture

The PLUTON system follows a modular architecture consisting of four primary components:

- Discord Bot (C2 Server) Handles command reception, session management, and communication control.
- Client Module (Agent) Runs on the target system, connects to Discord, and executes received commands.
- Execution Engine Processes system-level operations such as command execution, file handling, and network tasks.
- Builder Module, Converts Python scripts into executable files with optional obfuscation.

DIAGRAMMATIC REPRESENTATION (CONCEPTUAL)





Discord Bot Interaction Flow

The interaction flow of the system follows a structured communication pattern:

Command → Discord Bot → Client → Execution → Response.

The operator sends a command through Discord. The bot receives and processes the command. The client retrieves the command via API The execution engine performs the task. Results are sent back to the bot and displayed in Discord. This continuous loop enables real time remote control and monitoring.

Modules and Their Functional Use

The system utilises several Python libraries for implementation:

- discord.py – Enables communication with Discord APIs and acts as the backbone of the C2 framework. It allows commands to be sent and received via Discord channels.
- requests – Used for HTTP communication, such as downloading payloads or interacting with external services. pyautogui – Captures screenshots and automates GUI interactions on the client system.
- opencv-python (cv2) – Handles webcam access and video/image processing for surveillance functionalities. numpy – Processes image and video data as multi-dimensional arrays, improving computational efficiency.
- psutil – Monitors system processes, CPU usage, memory, and other system-level information.
- customtkinter – Provides a graphical interface for configuring the builder or control panel.
- rich – Enhances terminal output with formatted and readable displays.
- subprocess – Executes system commands and retrieves outputs.
- threading – Enables concurrent execution of multiple tasks without interrupting system flow.

Algorithms Used

The system incorporates several key algorithms:

- **Asynchronous Event Loop Algorithm**
- Ensures continuous communication between client and server without blocking execution [1]
- **Cryptographic Hashing (HWID Generation)**
- Generates unique identifiers for each client system using hashing techniques such as SHA-256 [5]
- **Image Compression Algorithm**
- Reduces the file size of screenshots using transformation techniques for efficient transmission
- **Data Serialisation and Encoding**

Converts complex data into JSON and Base64 formats for safe communication [2]

Formulas and Explanation

- **RSA Encryption:** Used for secure key exchange
- **AES Encryption:** Ensures secure data transmission
- **XOR Obfuscation:** Used to hide payload signatures
- **Jitter Formula:** Randomises communication timing to avoid detection
- **SHA-256 Hashing:** Ensures data integrity

- **Base64 Encoding:** Converts binary data into a safe text format

Conclusion

The PLUTON system demonstrates an innovative approach to Command and Control architecture by leveraging Discord as a communication platform. By replacing traditional server-based models, the system achieves reduced detectability, improved scalability, and cost efficiency.

The integration of both offensive simulation and defensive detection provides a comprehensive platform for understanding modern cybersecurity threats. The project serves as a valuable educational and research tool for analysing real-world attack strategies and developing effective detection mechanisms

Future Enhancements

Role-Based Access Control for multiple operators, Enhanced end-to-end encryption mechanisms, Cross-platform compatibility (Linux and macOS), AI-based command automation, Web-based visualisation dashboard, Integration with advanced detection systems (EDR tools)

Results and Discussion

The PLUTON framework was evaluated in a controlled virtual environment to assess its functionality and communication efficiency. The system successfully established stable connections between the Discord bot and multiple client instances, enabling real-time command execution and response handling.

Functional validation confirmed that key features, including remote command execution, file management, system monitoring, and multimedia capture, operated as intended. The use of Discord as a communication platform allowed the framework to blend with normal network traffic, reducing immediate detection by basic monitoring systems.

The Builder Module effectively generated executable payloads, simplifying deployment across test environments. Additionally, the detection module identified certain suspicious patterns, such as repeated command execution and abnormal API usage. However, the detection mechanism remains limited to rule-based analysis and does not incorporate advanced machine learning techniques.

Although the system demonstrates practical feasibility, it has limitations. No quantitative performance metrics, such as detection accuracy or false positive rates, were measured. Furthermore, testing was limited to controlled environments and did not include enterprise-level security systems.

References

Python Software Foundation, “asyncio — Asynchronous I/O,” Python Documentation. [Online]. Available: https://docs.python.org/3/library/asyncio.html

Python Software Foundation, “json — JSON encoder and decoder,” Python Documentation. [Online]. Available: https://docs.python.org/3/library/json.html

Palo Alto Networks, “What is Command and Control (C2)?,” Cybersecurity Resource Guide.

P. Wang, S. Sparks, and C. C. Zou, “An Advanced Hybrid Peer-to-Peer Botnet,” IEEE Trans. Dependable Secure Comput., vol. 7, no. 2, pp. 113–127, 2010.

B. AsSadhan, J. M. F. Moura, and D. Lapsley, “Detecting Botnets Using Command and Control Traffic,” in Proc. IEEE Int. Conf. Network Applications (NCA), 2009.

National Institute of Standards and Technology (NIST), “Secure Hash Standard (SHA-256),” FIPS PUB 180-4, 2015.

Discord Inc., “Discord Developer Documentation,” [Online]. Available: https://discord.com/developers/docs

OpenCV Organisation, “OpenCV: Open Source Computer Vision Library,” [Online]. Available: https://opencv.org

PyInstaller Development Team, “PyInstaller Documentation,” [Online]. Available: https://pyinstaller.org

HackerOne, “Hacker-Powered Security Platform,” [Online]. Available: https://www.hackerone.com

TryHackMe, “Cybersecurity Training Platform,” [Online]. Available: https://tryhackme.com

Hack The Box, “Penetration Testing Labs,” [Online]. Available: https://www.hackthebox.com