IRJCS: International Research Journal of Computer Science ISSN: 2393-9842
Volume 12, Issue 10, November 2025 https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

SentinelX: AI-Powered Network Security and Threat Detection System

Nakshatha V
Department of Artificial Intelligence and Machine Learning
Vemana Institute of Technology, Bengaluru, India
nakshathav.ai2022@vemanait.edu.in
https://orcid.org/0009-0004-0230-2923

Punith AM
Department of Artificial Intelligence and Machine Learning
Vemana Institute of Technology, Bengaluru, India
punitham.ai2022@vemanait.edu.in
https://orcid.org/0009-0000-1279-247X

Ruthvik H K
Department of Artificial Intelligence and Machine Learning
Vemana Institute of Technology, Bengaluru, India
ruthvikhk.ai2022@vemanait.edu.in
https://orcid.org/0009-0007-2402-0654

Publication History
Manuscript Reference: IRJCS/RS/Vol.12/Issue10/NVCSX10080
Research Article | Open Access | Double-Blind Peer Reviewed Article ID: IRJCS/RS/Vol.12/Issue10/NVCSX10080
Received:23,October 2025, Revised: 09, October 2025, Accepted: 31October 2025 Published Online: 21November 2025
https://www.irjcs.com/volumes/Vol12/iss-10/01.NVCSX10080.pdf
Article Citation: Nakshatha,Punith,Ruthvik(2025),SentinelX: AI-Powered Network Security and Threat Detection System **,**IRJCS: International Research Journal of Computer Science, Volume 12, Issue 09 of 2025 pages 552-570
doi:> https://doi.org/10.26562/irjcs.2025.v1210.01
BibTeX Nakshatha@2025SentinelX:
IRJCS papers should be cited as IRJCS (International Research Journal of Computer Science, AM Publications, India 2025, ISSN 2393-9842, https://doi.org/10.26562/ijrcs.2025.v1210.01 The journal's official abbreviation is IRJCS.
Orcid: https://orcid.org/0009-0004-9398-7488
Copyright©2025 copyright by the authors. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

Abstract: The increasing complexity of cyber-physical systems has led to sophisticated security challenges that traditional cybersecurity mechanisms cannot effectively counter. The proposed Honeypot system integrates artificial intelligence (AI), Internet of Things (IoT), and hardware-assisted monitoring to create a comprehensive real-time protection mechanism. It combines intrusion detection, face recognition-based access control, IP-based file access authorization, and environmental monitoring for physical spaces using sensors and cameras. The system employs a Python Flask-based web application for administration and alert management. AI-driven models, including Support Vector Machine (SVM) for face recognition, provide proactive intrusion response. Hardware components such as ESP8266 (for DHT11 and motion sensors) and ESP32 Camera offer environmental and visual surveillance. The system's self-healing and alerting capabilities, via email and dashboard notifications, ensure adaptive and autonomous cybersecurity. Together, this architecture provides a unified, intelligent, and scalable approach to protecting both digital and physical infrastructures.
Index Terms: Anomaly detection, artificial intelligence, cyber-physical security, deep learning, Internet of Things, intrusion detection system, machine learning, network security, real-time monitoring, threat intelligence

INTRODUCTION

In the modern digital era, technology has seamlessly integrated into every aspect of human life, from personal communication to industrial operations, healthcare, and national defense. As society continues to advance through the Fourth Industrial Revolution, the dependency on networked systems, cloud platforms, and Internet of Things (IoT) devices has grown exponentially. This rapid interconnection has significantly improved efficiency, automation, and intelligence in systems - but has also created new vulnerabilities and security challenges. Today, cybersecurity threats are evolving faster than traditional defense mechanisms can handle. Malicious actors now use sophisticated techniques such as AI-powered attacks, phishing automation, ransom ware, malware injection, and IoT exploitation to breach networks, disrupt operations, and steal critical data. Consequently, there is a pressing need for proactive, adaptive, and intelligent cybersecurity solutions that can defend both cyber and physical domains simultaneously.

1.1 The Need for Advanced Cyber-Physical Security

Modern infrastructures - whether in industrial plants, government facilities, or smart cities - are not just digital networks. They are cyber-physical systems (CPS) that combine sensors, actuators, IoT devices, cameras, and control software.

This convergence of the physical and virtual worlds makes them susceptible to hybrid attacks that exploit both digital vulnerabilities and physical intrusions. For example, an attacker may gain access to a secured room through cloned RFID cards or fake biometric data while simultaneously launching a network-level attack to disable surveillance cameras. Such multi-vector attacks highlight the inadequacy of traditional, isolated cybersecurity solutions. Most conventional systems rely on firewalls, antivirus software, and static intrusion detection systems (IDS). While these tools can block known threats, they struggle against zero-day attacks, polymorphic malware, and AI-generated exploits. Furthermore, many networks lack environmental awareness they cannot respond to real-world physical changes such as motion, temperature, or unauthorized access attempts. To fill this gap, a comprehensive, unified cybersecurity framework is required - one that combines AI intelligence, IoT-based monitoring, biometric authentication, and honeypot deception mechanisms to achieve complete cyber-physical protection.

1.2 Overview of Honeypot-Based Defense Mechanisms

A honeypot is a cybersecurity mechanism designed to attract, deceive, and analyze attackers. It simulates a real system environment, luring malicious users into interacting with it while recording their behavior in detail. Honeypots provide valuable insights into attack techniques, tools, and motives - enabling system administrators to strengthen actual defenses proactively. Unlike traditional detection systems that rely on known signatures, honeypots allow behavioral analysis of unknown or emerging threats. In this project, the honeypot forms the first line of defense, capturing all unauthorized access attempts, file modifications, and pendrive insertions from unapproved IP addresses. This data is logged, analyzed, and immediately reported to the admin dashboard, ensuring that threats are detected at the earliest stage. The honeypot works in coordination with AI-based analytics to predict potential attacks and assess their severity. For example, if multiple unauthorized access attempts originate from the same IP range, the system can automatically blacklist the address and notify the administrator. The incorporation of machine learning techniques ensures that the defense system becomes more intelligent and adaptive with time.

1.3 Role of Artificial Intelligence in Cybersecurity

Artificial Intelligence (AI) has revolutionized the way modern cybersecurity systems operate. By integrating AI into security frameworks, systems can learn from past attack patterns, predict potential intrusions, and make intelligent decisions without manual intervention.

In the proposed Honeypot system, AI plays multiple roles:

1. Anomaly Detection: Machine learning algorithms analyze incoming data from sensors, IP logs, and user behavior to detect irregularities that might signify a cyberattack or unauthorized access.
2. Biometric Recognition: Using Support Vector Machine (SVM) and face recognition algorithms, the system authenticates users through their facial features, providing a robust and secure access mechanism that is immune to password theft.
3. Decision Making: AI helps determine the severity of detected threats, prioritize alerts, and initiate appropriate responses, such as blocking an IP or activating the self-healing protocol.
   By combining AI-driven detection with real-time IoT monitoring, the system ensures proactive defense that not only responds to incidents but also anticipates and prevents them.

1.4 Integration of IoT and Sensor-Based Monitoring

The proposed system leverages IoT devices such as ESP8266 microcontrollers, ESP32-CAM modules, and DHT11 sensors to bring environmental awareness into cybersecurity. These sensors continuously monitor the physical environment - detecting motion, temperature fluctuations, and human presence - and send data to the central Flask-based server in real time. If the DHT11 sensor detects abnormal temperature (indicating equipment overheating or fire risk) or the PIR motion sensor identifies movement in a restricted area, the system instantly triggers an alert. Similarly, the ESP32-CAM streams live video from the monitoring room and performs AI-based face recognition. If an unauthorized face is detected, the system logs the incident, displays it on the admin dashboard, and sends an email alert for immediate action. This tight coupling between the digital and physical layers forms a resilient defense mechanism capable of detecting, localizing, and responding to both virtual and physical intrusions. It transforms a passive monitoring system into an active, intelligent guardian that constantly learns from its environment.

1.5 Flask-Based Application and Real-Time Administration

At the heart of the system lies a Flask web application, which acts as the control hub and communication layer. The Flask server connects all components - AI modules, IoT devices, cameras, sensors, and databases - into a single unified platform. The MySQL database securely stores user credentials, IoT data, activity logs, and intrusion records. Through an intuitive admin dashboard, the administrator can monitor all system activities in real time, view alerts, analyze sensor data, and manage users. When a security event occurs, it is displayed on the dashboard and simultaneously sent to the admin via email notification. This ensures that security incidents are never missed and that the administrator can respond swiftly, even from a remote location. The dashboard also provides historical data visualization, showing patterns of past intrusions, temperature variations, and motion activity trends. Such analytics enable administrators to identify recurring threat vectors and fine-tune defense parameters accordingly.

1.6 Self-Healing and Automated Recovery

Another vital innovation in the proposed system is the inclusion of a self-healing mechanism. This feature enhances the reliability and availability of the system by allowing it to automatically recover from certain types of failures or disruptions.

For instance, if a sensor node or camera becomes unresponsive due to a network fault, the system attempts to reconnect or restart the service automatically. Similarly, if the Flask server detects a drop in communication from one of the IoT devices, it triggers a background task to restore the connection. This level of automation ensures that the system remains operational 24/7, reducing downtime and eliminating the need for constant human supervision. The combination of honeypot defense, AI decision-making, IoT monitoring, and self-healing capabilities makes the proposed system both intelligent and resilient - a necessity for securing critical infrastructures in the age of digital transformation.

1.7 Benefits and Significance of the Proposed System

The proposed Honeypot-based AI and IoT integrated security system introduces a multi-layered protection architecture that goes beyond conventional defense models. Its key benefits include:

- Proactive Detection: Identifies threats before they compromise the system using honeypot-based deception and AI-based analytics.
- Real-Time Alerts: Provides immediate visual and email notifications of any suspicious activity.
- Biometric Verification: Eliminates password vulnerabilities through secure face recognition.
- Environmental Intelligence: Integrates physical-world awareness using IoT sensors and cameras.
- Autonomous Operation: Self-healing and automated recovery enhance uptime and system resilience.
- Scalability: The modular design allows easy integration of additional IoT devices, cameras, or sensors.
- Data-Driven Security: Logs and analytics enable continuous improvement of security policies and responses.

By bridging the gap between cyber and physical security, this project contributes toward building a next-generation, intelligent cybersecurity framework suitable for critical infrastructure protection, smart surveillance, and secure IoT environments.

LITERATURE SURVEY

[1] Guo Xiu-feng (2022) - Research on Computer Network Information Security and Protection Strategies in the Era of Big Data In this study, Guo Xiu-feng proposed a layered and scientific network protection model aimed at addressing evolving cybersecurity threats in traditional IT infrastructures. The research emphasized multi-tiered defense mechanisms including access control, data encryption, and real-time network monitoring to ensure comprehensive protection. The work systematically analyzed risks associated with data storage and transfer under big data environments.
Advantages: The layered model provides a clear and structured framework that can be extended to various types of networks, allowing administrators to implement hierarchical defenses. It also underscores the importance of adopting scientific protection strategies aligned with modern data demands.
Disadvantages: Despite its conceptual soundness, the model lacks a practical implementation or validation in IoT-based or cyber-physical systems, where real-time data and heterogeneous devices create dynamic security needs. Moreover, it focuses more on policy-level recommendations than on executable system architectures.

[2] Wang Yan (2023) - Research on Computer Network Information Security and Protection Strategy in the Background of Big Data
Wang Yan proposed an information security framework to address the growing challenges of large-scale data flows, cloud storage, and inter-network communication within big data ecosystems. The paper analyzed data confidentiality, integrity, and availability as the three pillars of secure data management. Strategies included encrypting data at rest and in motion, intrusion detection, and secure user authentication.
Advantages: The work's main contribution lies in its comprehensive treatment of large-scale data protection, focusing on both technical and procedural safeguards for data governance.
Disadvantages: However, it lacks real-time intrusion detection mechanisms and fails to consider IoT or distributed sensor networks, making it less suitable for decentralized, real-time monitoring systems. The proposed solution remains static rather than adaptive to dynamic cyber threats.

[3] Cui L. (2022) - Information Security Management Measures for College Archives under the Network Environment
Cui L. explored the growing security issues faced by educational institutions during the digitization of archives and document repositories. The research emphasized data classification, secure storage, access control, and backup management for sensitive institutional records.
Advantages: This study contributes to improving the data governance and integrity of academic institutions through systematic information management. It establishes essential practices such as controlled access and encrypted data storage.
Disadvantages: However, the reliance on open networks introduces vulnerabilities that weaken the overall system resilience. Additionally, the research does not consider the implications of remote IoT-enabled access or the biometric authentication methods required for modern security infrastructures.

[4] Wang Shunyuan (2020) - Research on Computer Network Information Security and Protection Strategy
In this work, Wang Shunyuan analyzed network threats by classifying them into internal and external categories and proposing mitigation strategies for each. The study covered user mismanagement, malware propagation, and external attacks such as DoS and phishing.

Advantages: The classification provides clarity on designing customized defense strategies for specific threat origins. It helps define appropriate monitoring layers and administrative controls for network defense.
Disadvantages: However, it fails to incorporate IoT-specific vulnerabilities such as insecure firmware, sensor hijacking, or weak Wi-Fi encryption. Furthermore, the model lacks adaptability and does not employ machine learning or AI-based anomaly detection to identify emerging threats dynamically.

[5] Ma Yuehuan (2022) - Discussion on Computer Network Information Security and Protective Countermeasures
Ma Yuehuan presented a framework that integrates adaptive security policies into network systems to balance service quality and system protection. The research suggested combining fault tolerance, redundancy, and adaptive control to achieve continuous service even under attack conditions.
Advantages: The study promotes a dynamic security philosophy, focusing on maintaining service stability while enhancing data protection. This adaptability is essential for mission-critical applications.
Disadvantages: The proposed model remains theoretical, offering little evidence of performance in IoT or real-time monitoring environments. It lacks detailed algorithms or experimental validation, which limits its applicability to smart or hybrid systems.

[6] Cai Guang-Song (2023) - Research on Computer Network Information Security and Protection Strategies in the Era of Big Data
Cai Guang-Song introduced an adaptive, security-aware framework that integrates big data analytics with IoT security policies. The framework emphasizes context-aware protection, allowing the system to dynamically adjust defense rules based on detected threat levels or network context.
Advantages: It moves beyond static protection by employing real-time data analytics to adapt to the network's behavior, paving the way for AI-driven cybersecurity.
Disadvantages: While theoretically strong, it lacks experimental results from real-world IoT networks. The framework does not specify integration methods with embedded sensors or edge devices, which are crucial for hybrid systems combining software and hardware security.

[7] Rios Insua D., Couce-Vieira A., Rubio J.A., et al. (2021) - An Adversarial Risk Analysis Framework for Cybersecurity
This paper introduced Adversarial Risk Analysis (ARA), a model that anticipates attacker behaviour using game-theoretic and probabilistic approaches. The framework provides a mathematical method to assess and predict the likelihood of attacks, thus improving proactive defense.
Advantages: It represents an intelligent and predictive approach to cybersecurity, offering systems the ability to simulate attacker decisions and optimize defenses accordingly.
Disadvantages: The computational complexity of ARA makes it unsuitable for resource-limited IoT devices. Implementing this framework in real-time embedded systems would require lightweight approximations, limiting its practical usability in hybrid or mobile IoT systems.
[8] Zheng Y., Li Z., Xu X., et al. (2022) - Dynamic Defenses in Cyber Security: Techniques, Methods, and Challenges
This paper emphasized the need for dynamic defense systems capable of adjusting in real time based on the type, frequency, and severity of ongoing attacks. It covered methods like moving target defense (MTD), cyber deception, and adaptive firewall configuration.
Advantages: Introduces the concept of agile security, where the system can continuously evolve to match changing threat patterns, significantly improving detection rates.
Disadvantages: The complexity of maintaining dynamic configurations and real-time reprogramming across diverse IoT environments makes it difficult to deploy in heterogeneous smart networks such as smart homes or industrial IoT ecosystems.

[9] Kaur M., Singh D., Kumar V., et al. (2021) - Secure and Energy-Efficient-Based E-Healthcare Framework for Green IoT
This work developed a secure and energy-efficient architecture for e-health IoT systems, integrating encryption and data aggregation mechanisms to ensure patient data privacy without excessive energy use. The framework utilizes lightweight cryptographic protocols suitable for IoT devices with limited resources.
Advantages: Demonstrates a balance between security and energy efficiency, making it ideal for health-monitoring applications with continuous data streaming. It highlights the use of resource-efficient encryption in sensitive environments.
Disadvantages: The framework's complexity and domain specificity limit its scalability beyond healthcare contexts. Furthermore, it does not integrate real-time intrusion response or AI-based anomaly detection, which would make it more adaptive.

[10] Rawal B. S., Manogaran G., Hamdi M. (2021) - Multi-Tier Stack of Blockchain with Proxy Re-Encryption Method Scheme on the Internet of Things Platform

Rawal and colleagues proposed a multi-tier blockchain architecture combined with proxy re-encryption to enhance data confidentiality and decentralized control in IoT platforms. The system ensures secure communication between devices without relying on centralized authentication authorities.
Advantages: Provides robust confidentiality, integrity, and distributed trust across IoT networks. Blockchain integration strengthens tamper resistance and accountability.
Disadvantages: The approach incurs high computational and storage overhead, making it unsuitable for lightweight IoT nodes or edge devices. The integration of blockchain also introduces latency, reducing its feasibility for real-time or low-power IoT security applications.

EXISTING SYSTEM AND DRAWBACKS

In the current cybersecurity landscape, most network protection systems primarily rely on traditional intrusion detection systems (IDS), firewalls, and antivirus software to monitor, detect, and prevent unauthorized access or malicious activity. These systems are largely reactive in nature- they respond to threats only after a breach or anomaly has already occurred. While such solutions provide basic network protection, they are not well-equipped to handle hybrid cyber-physical threats that exploit both digital vulnerabilities and physical access points, especially in environments integrating IoT devices and sensors. Most existing systems lack real-time intelligent monitoring and depend heavily on pre-defined rule sets or static databases of known attack signatures, which makes them ineffective against zero-day exploits and emerging AI-driven attacks. Furthermore, they often fail to incorporate hardware-level monitoring, biometric authentication, or context-aware threat detection, leaving physical infrastructures vulnerable to insider threats and unauthorized access.
Another major drawback of traditional systems is their limited adaptability and scalability in large or distributed networks. As IoT devices, embedded sensors, and smart nodes become increasingly prevalent, existing solutions struggle to manage the volume, velocity, and variety of data generated in real time. There is also poor integration between different layers of security- such as network, application, and hardware- resulting in fragmented protection. Additionally, many conventional setups lack automated alerting mechanisms and remote supervision capabilities, requiring manual intervention to detect and respond to potential breaches. As a result, by the time a security administrator identifies an anomaly, significant data or system damage may already have occurred.
The absence of AI-based anomaly detection and self-healing mechanisms further limits the resilience of traditional network security models. Most do not feature adaptive learning algorithms that can evolve from attack patterns and predict potential threats. Similarly, biometric verification and device-level authorization, which are critical in modern cybersecurity frameworks, are often missing. In essence, existing systems remain reactive, fragmented, and limited in scope - providing incomplete coverage for today's AI-powered, IoT-integrated environments where cyberattacks can propagate swiftly through interconnected networks. This gap highlights the pressing need for an advanced, unified, and intelligent security model like the proposed Honeypot-based system that integrates AI, IoT, biometric verification, and real-time alerts for comprehensive cyber-physical protection.
Drawbacks:

- Reactive rather than proactive defense mechanisms.
- No integration between cyber and physical threat monitoring.
- Lack of biometric or IP-based access control.
- Limited intelligence and adaptability in intrusion response.
- No environmental or camera-based monitoring.

PROPOSED SYSTEM AND ADVANTAGES

The proposed Honeypot-based Cyber-Physical Security System is an advanced, unified solution that combines artificial intelligence (AI), Internet of Things (IoT), machine learning, biometric access control, and real-time monitoring to provide intelligent and adaptive protection for critical computer networks and infrastructures. Unlike conventional systems that are reactive, this system takes a proactive approach by continuously analyzing network behavior, user activity, and physical parameters to identify and mitigate threats before they can cause harm. The system is developed using a Python Flask-based web application integrated with a MySQL database, ESP32 camera, and ESP8266 microcontrollers connected to environmental sensors such as DHT11 (for temperature and humidity) and PIR motion sensors for physical intrusion detection.
The core of the system revolves around a honeypot mechanism—a decoy environment that intentionally attracts attackers to monitor and analyze their behavior in real-time. This enables the administrator to gain valuable insights into emerging threats, attack vectors, and malicious intent without compromising the actual network. The honeypot acts as both a learning system and an early-warning layer. Any suspicious activity, such as attempts to access restricted files, modify system data, or use unauthorized devices (like USB drives or pendrives), is immediately captured and reported to the Flask server. The system then triggers real-time alerts on the admin dashboard and simultaneously sends email notifications to the administrator, ensuring instant awareness and response.
To strengthen user-level access control, the system incorporates biometric-based authentication using face recognition. The admin can register authorized users by uploading their face images and training the model using Support Vector Machine (SVM) and face recognition libraries. During authentication, if the system detects an unknown or unauthorized face from the ESP32 camera feed, an alert is automatically generated, and access is denied.

This biometric layer eliminates the risk of password theft or credential misuse. Additionally, the admin can specify allowed IP addresses, ensuring that only pre-approved systems can access or modify files. Any unauthorized attempt from an unknown IP trigger a security alert and blocks the action.

The IoT integration provides physical-level monitoring of the environment. The ESP8266 continuously transmits temperature and motion data from the monitoring room to the Flask server. If abnormal temperature levels or unexpected motion are detected, the server analyzes the data using AI-based thresholds and immediately notifies the admin about potential physical intrusions, overheating, or tampering attempts. This establishes a cyber-physical security link, bridging both virtual and environmental protection. The system's self-healing mechanism automatically restores compromised services or reconfigures affected nodes to maintain uninterrupted operation and network stability.
From a software perspective, the system leverages Flask for backend logic, MySQL for secure credential and event storage, and Python-based modules for AI-driven face recognition, anomaly detection, and communication with IoT devices via HTTP or MQTT protocols. On the hardware side, the use of low-cost and energy-efficient devices like ESP8266, ESP32-CAM, and DHT11 sensors makes the system scalable and easily deployable in real-world monitoring environments. The integration of AI models enhances decision-making, allowing the system to adapt over time based on observed attack patterns and environmental conditions.
Overall, the proposed system offers intelligent, layered, and autonomous cybersecurity that bridges the gap between digital and physical domains. It ensures real-time monitoring, adaptive threat detection, automated alerts, and secure user authentication while minimizing human intervention. The modular design makes it scalable, cost-effective, and customizable for various applications- including data centers, laboratories, industrial control systems, and government monitoring facilities.

Advantages:

- Proactive and Intelligent Defence:

Uses AI-driven anomaly detection and honeypot analysis to identify threats before they affect the main system.

- Real-Time Alerts and Monitoring:

Instantly notifies administrators through the dashboard and email notifications for immediate action.

- Biometric Authentication:

Ensures secure user access with SVM-based face recognition, preventing password or credential misuse.

- Hardware-Integrated Security:

Incorporates IoT sensors (DHT11, motion sensors, ESP32-CAM) for real-time environmental and physical monitoring.

- Comprehensive Cyber-Physical Protection:

Unifies network security with physical intrusion detection to safeguard both virtual and real assets.

- Self-Healing Capability:

Automatically restores affected components or configurations to maintain continuous operation.

- Scalable and Cost-Effective:

Utilizes open-source software and affordable IoT hardware for broad applicability and scalability.

- Customizable and Modular:

Easily adaptable for different network environments and critical infrastructure systems.

- Enhanced Decision-Making:

Learns from historical attack data through AI, improving detection accuracy over time.

- Reduced Human Intervention:

Automates most of the threat detection, response, and alert processes, minimizing administrative load.

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 10, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

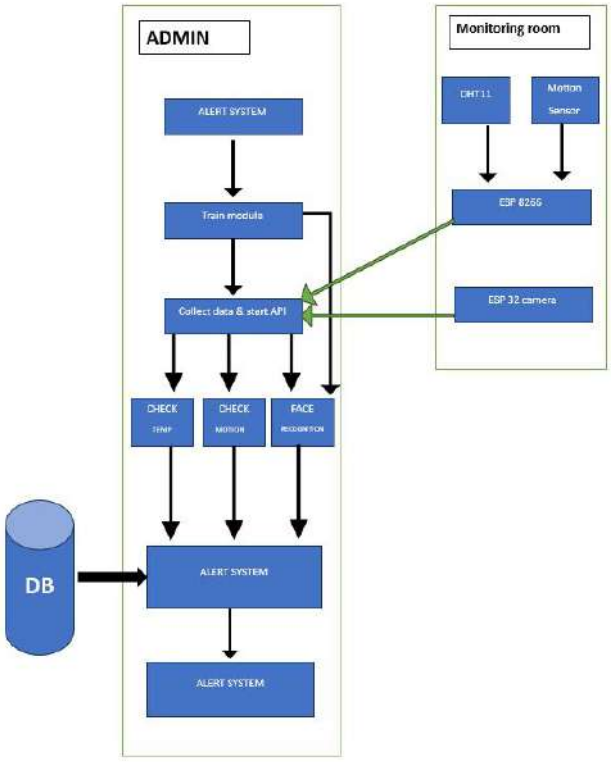


Fig.1: Proposed System Architecture

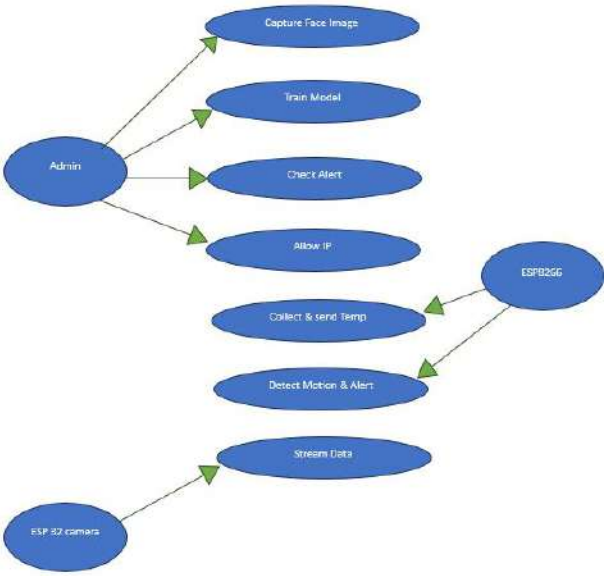


Fig.2: Usecase Diagram

SOFTWARE AND HARDWARE DETAILS

Software Components:

- Backend: Python Flask
- Database: MySQL
- Machine Learning: Face Recognition, OpenCV, Scikit-learn (SVM)
- Alerting: SMTP (Email Notification)
- Frontend: HTML, CSS, JavaScript
- IoT Communication: HTTP / MQTT (via ESP8266 and ESP32)

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 10, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

Hardware Components:

- ESP8266: Interface for DHT11 (Temperature & Humidity) and PIR Motion Sensor
- DHT11 Sensor: Environment monitoring
- PIR Motion Sensor: Detects physical movement
- ESP32-CAM: Video surveillance and face capture
- Power Supply Module
- Wi-Fi Network for IoT Communication

PYTHON

Python is a computer programming language and is an object-oriented language that was developed by Guido Rossum in 1989. It is best suited at quick prototyping of complicated applications. It has interfaces to numerous OS system calls and libraries and can be extended to C or C++. The Python programming language has many major companies that utilize it NASA, Google, YouTube, BitTorrent, etc.

Python is popular in Artificial Intelligence, Natural Language Generation, Neural Networks and other advanced branches of Computer Science. Python was heavily concerned with the readability of code and this course will show you python starting at the bottom.

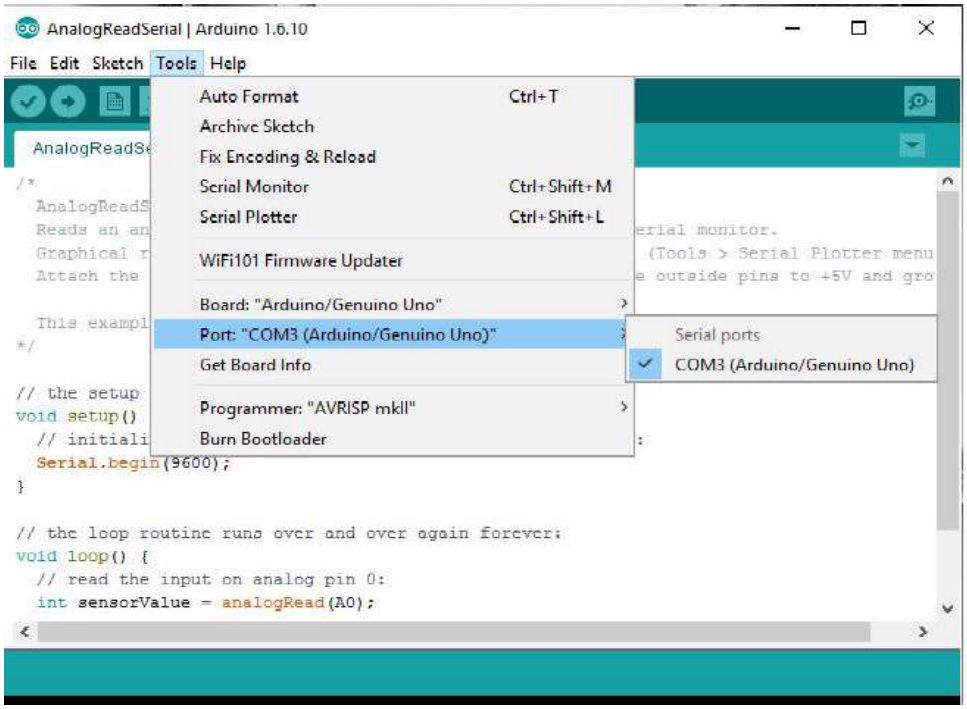
Characteristics of Python

- It presents dense data forms and format that is simple to read as compared to any other programming languages.
- It is a platform independent, scripted, language that has access to operating system API.
- It is more flexible with respect to run-time as compared to other programming languages.
- It contains the simple text manipulative capabilities of Perl and Awk.
- A Python software module can contain classes or free functions.
- Python Libraries Pythons libraries are cross-platform, and can run on Linux, MacIntosh, and Windows.

Arduino ide

The Arduino Integrated Development Environment (IDE), a cross-platform application (Windows, macOS, Linux) is written in the programming language Java. It is also used to write and upload programs to Arduino compatible boards, but with the aid of 3rd party cores, other vendor development boards.

The IDE has source code that is under the GNU General Public License, version 2. Arduino IDE provider supports C and C++ languages which have special code structuring rules. The Arduino IDE includes a software library based on the Wiring project which supplies numerous input and output procedures that are common. Only two rudimentary functions, to initialize the sketch and the main program loop, that are compiled and combined with a program stub main () are needed to generate an executable cyclic executive program using the GNU toolchain included with the IDE distribution. Arduino IDE uses a program named avrdude to encode the executable code as a text file in hexadecimal format which is uploaded into the Arduino board by a loader program in the firmware of the board.



IRJCS: International Research Journal of Computer Science ISSN: 2393-9842
Volume 12, Issue 10, November 2025 https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

MySQL

MySQL (My Sequel) is (as at 2008) the most popular open-source relational database management system (RDBMS) in the world that operates as a server, and makes multi-users of a set of databases. The SQL is an abbreviation of Structured Query Language.

MySQL development project has licensed its source code under the Apache License, and a range of proprietary ones. MySQL was initially owned and sponsored by a single profit-making company, the Swedish MySQL AB that is since owned by Oracle Corporation.

MySQL is also a popular database platform used in web applications, and a core part of the ubiquitous LAMP open-source web application software stack (and other AMP stacks). LAMP is an abbreviation that means Linux, Apache, MySQL, Perl/PHP/Python. MySQL is frequently used in free-software-open-source projects which depend on a full-featured database management system.

MySQL is a relational database management system (RDBMS) and can be configured with no GUI tools to administer MySQL databases or administer data that is stored within the databases. Users can also invoke the command line tools that are included, or may employ MySQL front-ends, desktop applications and web applications that create and manage MySQL databases, structure database, backup data, examine status and manipulate data records. MySQL Workbench is an official collection of MySQL front-end tools, which Oracle actively develops, and freely availability.

Just like most other transactional relational databases, MySQL is highly constrained by the performance of hard disk. This is particularly so when it comes to write latency. Since of late, very inexpensive consumer grade SATA interface Solid-state drives with a zero mechanical latency can be found, then a fivefold speedup over even an eight drive RAID can be obtained at a reduced investment.

Fig.: MySQL Workbench in Windows

ESP8266 NodeMCU Board:

ESP8266 NodeMCU CP2102 board contains ESP8266 which is one of the highest integrated chip to meet the requirements of the new connected world. It provides a full-fledged, and integrated Wi-Fi networking system, which enables it to either implement the application or offload all Wi-Fi networking capabilities to another application processor.

IRJCS: International Research Journal of Computer Science ISSN: 2393-9842
Volume 12, Issue 10, November 2025 https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

ESP8266 has strong on-board processing and storage facilities which enable it to combine with the sensors and other application-specific devices using its GPIOs and requires little development beforehand and little loading at runtime. It has a great level of on-chip integration which means that external circuitry is minimal and the overall solution including the front-end module consumes little PCB area.

ESP8266 NodeMCU development board ESP8266 NodeMCU development board - a genuine plug-and-play solution to cheap projects based on Wi-Fi. The module comes already pre-flashed with the NodeMCU firmware to connect it just needs a USB driver (below). ESP-12 Lua NodeMCU WI-FI Dev Board Internet of Things board features a complete ESP8266 Wi-Fi with all the GPIO broken out, a complete USB-serial interface, and a power supply all on the single breadboard friendly package.

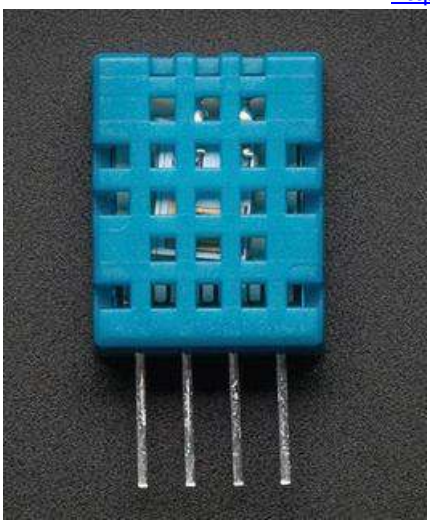
This board is already pre-flashed to NodeMCU - a Lua-based ESP8266 firmware that enables easy control with an easy-to-use scripting language - Lua - in a matter of a few minutes.
The ESP-12 Lua NodeMCU WI-FI Dev Board Internet of Things is an all-inclusive microcontroller + Wi-Fi board which is highly user-friendly in making a project with the help of Wi-Fi and Internet of Things (IoT) applications.

It has a board that is based on the so popular ESP8266 Wi-Fi Module chip with ESP-12 SMD footprint. The board has already integrated into its board the required components to the ESP8266 (ESP-12E) to program and upload software. It consists of inbuilt USB to serial chip upload codes, 3.3 V regulator and logic level converter circuit such that you can just upload codes and connect your circuits.
Features:

1. 11 b/g/n Wi-Fi Direct (P2P), soft-AP
2. Integrated TCP/IP protocol stack
3. Use CH340G to replace the CP2102.
4. Open-source, Interactive, Programmable, Low cost, Simple, Smart, WI-FI enabled
5. Arduino-like hardware IO
6. Integrated low power 32-bit CPU
7. Advanced API for hardware IO, which can dramatically reduce the redundant work for configuring and manipulating hardware.
8. Code like Arduino, but interactively in Lua script.
9. Nodejs style network API
10. Event-driven API for network applications,
11. which facilitates developers writing code running on a 5mm*5mm sized MCU in Nodejs style.
12. Greatly speed up your IOT application developing process.
13. Lowest cost WI-FI
14. Less than WI-FI MCU ESP8266 integrated and easy to prototyping development kit.
15. We provide the best platform for IoT application development at the lowest cost.
16. Integrated TR switch, balun, LNA, power amplifier and matching network
17. Integrated PLL, regulators, DCXO and power management units
18. Onboard USB to serial chip to easily program and upload codes from the Arduino IDE
19. Embeds logic level converter circuits
20. Has onboard 3.3V regulator to ensure enough power to function as your go-to Wi-Fi chip!
21. Easy access to the GPIO pins for easy prototyping
22. ESP-12E Processor
23. Easy to use breadboard friendly form factor
24. Voltage Regulator / Converter, excellent DC to DC conversion, super-efficient

DHT 11
It is a DHT11 Temperature and Humidity Sensor which has the ability of a calibrated digital signal output coupled with the temperature and humidity sensor. It is combined with an 8-bit microcontroller with high performance. The high reliability and the good long-term stability are guaranteed by its technology. This sensor has a resistive element and wet NTC temperature sensor measuring devices. It is of good quality, quick response, anti-interference capability and high-performance.
The DHT11 sensors have an exceptionally high degree of precision when it comes to the calibration of the humidity calibration chamber. These calibration coefficients we should call the calibration coefficients that are stored in the memory of the OTP program. The one wire serial interface system is incorporated to make it fast and simple. It is small, low power and in-flight signal transmission noise can go up to 20 meters and therefore a range of applications and even the most challenging ones can be done. It is a single row pin package 4-pin type. Special packages can be offered based on the need of the users and convenient connection can be offered.

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 10, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01



System Requirements

1. Functional Requirements

Functional requirements describe what the system should do- the specific actions, operations, and tasks it must perform to achieve its objectives.

The proposed Honeypot system integrates AI, IoT, Flask backend, and biometric authentication, so the following functionalities are essential:

1.1 User Authentication and Access Control

- The system must allow the admin to log in securely using credentials verified from the MySQL database.
- The admin can add new authorized users by capturing and storing their facial images in the system.
- The system should train a facial recognition model using SVM (Support Vector Machine) and the face recognition module for accurate user identification.
- The system should authenticate users via facial recognition before allowing any system or file access.
- The admin can define allowed IP addresses from which devices can access, modify, or transfer files. Unauthorized IP access attempts must trigger an alert.

1.2 Honeypot-Based Network Monitoring

- The system should include a honeypot trap mechanism to attract and log unauthorized intrusion attempts.
- All interactions with the honeypot must be monitored, recorded, and analyzed in real time.
- The system must automatically generate alerts and log attacker behavior for forensic and analytical purposes.

1.3 IoT Device Integration

- The ESP8266 microcontroller must continuously send temperature, humidity, and motion data from the DHT11 and PIR sensors to the Flask server.
- The ESP32-CAM should stream real-time video data from the monitoring area to the server.
- If unusual temperature readings or unauthorized motion are detected, the system must generate an immediate alert.
- If an unknown face is detected by the ESP32-CAM, the system should automatically send an alert and email notification to the admin.

1.4 Real-Time Alerts and Notifications

- The system must display alerts on the admin dashboard whenever a threat or anomaly is detected.
- It should also send an email notification to the admin for quick awareness and response.
- Alerts must include details such as type of event, time, device/IP, and captured image (if available).

1.5 Self-Healing and Recovery

- The system should attempt auto-reconfiguration or restarting of affected services in case of partial network failure or intrusion.
- Critical processes should be restarted automatically without manual intervention.

1.6 Data Storage and Logging

- The system must maintain secure logs of all user activities, intrusion attempts, and IoT data.
- All logs and sensor readings must be stored in the MySQL database for future analysis and report generation.
- The admin should be able to view, filter, and export activity logs through the web interface.

1.7 Dashboard and Analytics

- The Flask-based dashboard should display real-time data from IoT sensors, camera feed, and network alerts.
- The admin should be able to visualize intrusion trends, environmental changes, and system performance through charts or metrics.

ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

- The dashboard must support real-time refresh and alert notifications.

2. Non-Functional Requirements

Non-functional requirements define how the system should behave, including performance, security, reliability, and scalability.

2.1 Performance Requirements

- The system should process IoT sensor data and video feeds in real time with minimal latency (< 2 seconds delay).
- The face recognition system should complete identification within 2–3 seconds for a single frame.
- The server should handle at least 50 concurrent connections without degradation in performance.

2.2 Security Requirements

- All data transmission between IoT devices and the Flask server should be encrypted (e.g., HTTPS or AES-based encryption).
- Admin credentials and biometric data must be securely stored in hashed form in the database.
- Only authorized IP addresses should be permitted to access or modify sensitive data.
- The system must maintain audit trails for all access attempts and configuration changes.
- The system should implement role-based access control for managing multiple admin or monitoring users if needed.

2.3 Reliability and Availability

- The system should be operational 24/7 with minimal downtime.
- Automatic error detection and recovery mechanisms must be implemented for network interruptions or device failures.
- The system should support redundant data backups to prevent data loss during unexpected shutdowns.

2.4 Scalability

- The architecture must support the addition of multiple IoT nodes, cameras, and sensors without structural modification.
- The system should be easily extendable to handle larger networks or multiple monitoring zones.

2.5 Usability

- The admin dashboard should have an intuitive, easy-to-use interface for viewing alerts, logs, and live data.
- The system should support cross-browser compatibility (Chrome, Firefox, Edge).
- Alerts and notifications should be visually distinct and prioritized based on severity (e.g., color-coded).

2.6 Maintainability

- The system must be modular, allowing independent updates to components like AI models, IoT firmware, and Flask logic.
- System configuration files should be easily editable without code modification.
- All errors and system events should be logged for debugging and maintenance.

2.7 Portability

- The software should be deployable on Windows, Linux, or cloud servers (e.g., AWS, Azure).
- IoT firmware should be compatible with standard ESP8266/ESP32 hardware and easily reconfigurable.

2.8 Efficiency

- The system should consume minimal network bandwidth for sensor updates.
- IoT devices must operate efficiently within low-power constraints while maintaining accurate data transmission.

2.9 Accuracy

- Face recognition and SVM classification should achieve at least 90–95% accuracy in identifying registered users.
- Motion detection and temperature readings should maintain sensor accuracy tolerance within ±2%.

2.10 Compliance and Privacy

- The system must comply with data protection and privacy standards, ensuring user biometric data is not shared or exposed.
- All logs must follow confidentiality protocols and be accessible only by authorized personnel.

MODULES OF THE SYSTEM

The proposed system is divided into multiple interrelated modules that work collaboratively to achieve end-to-end cyber-physical protection. Each module performs a specific function- from user management and AI-based authentication to IoT monitoring, honeypot tracking, and alert generation. Together, these modules ensure real-time, adaptive, and intelligent security supervision of the entire infrastructure.

1. Admin Authentication and User Management Module

This is the entry point of the system, ensuring that only authorized administrators can access the control panel.

- The Flask-based login interface validates admin credentials stored securely in the MySQL database.
- After successful authentication, the admin gains access to all system functionalities, including user registration, monitoring dashboards, and alert management.

ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

- The admin can register authorized users by uploading multiple facial images, which are preprocessed and stored in a structured dataset.
- The system then uses Support Vector Machine (SVM) and face recognition algorithms to train the model on these registered faces.
- During runtime, when a user attempts to access the system, the camera captures the face, compares it against trained embeddings, and verifies authenticity.
- Any unrecognized or unauthorized face triggers an intrusion alert, which is displayed on the admin dashboard and also sent via email notification.

Key Functionalities:

- Secure admin login via Flask-MySQL.
- Authorized user face registration and model training.
- Real-time face verification using SVM + Face Recognition.
- Alert generation for unauthorized access attempts.

Technologies Used: Flask, MySQL, Python (face_recognition, OpenCV, scikit-learn).

2. Honeypot Intrusion Detection and Analysis Module

This module acts as a decoy environment designed to detect, divert, and analyze malicious network traffic. It creates a simulated network or file system that appears genuine to potential attackers.

- When an unauthorized user or attacker attempts to access or modify files, plug in a pendrive, or perform abnormal system operations, the honeypot traps and logs the behavior.
- The system captures detailed metadata- such as IP address, time, attempted command, and source port- and sends it to the central Flask server.
- This enables real-time intrusion analysis and allows the admin to study attack vectors without compromising the main system.
- The honeypot serves as a learning tool, helping the AI module improve its understanding of attack patterns over time.
- Alerts are generated and displayed on the dashboard, along with an email notification to the administrator.

Key Functionalities:

- Simulated decoy environment for attack monitoring.
- Real-time network behavior logging and analysis.
- IP-based filtering and blocking.
- Alert and report generation on intrusion attempts.

Technologies Used: Python (socket, logging), Flask, MySQL, SMTP for email alerts.

3. IoT Environment Monitoring Module

This module bridges the physical environment and cybersecurity system, ensuring real-time monitoring of the surroundings through IoT sensors and microcontrollers.

- The ESP8266 microcontroller connects with the DHT11 sensor to capture temperature and humidity and with a PIR motion sensor to detect physical movement in the monitoring area.
- Sensor data is periodically sent to the Flask server via HTTP requests or MQTT protocols.
- The system continuously analyses the data to detect anomalies such as sudden temperature rises, environmental changes, or unexpected motion.
- When abnormal readings or unauthorized movements are detected, the server triggers alerts on the dashboard and sends email notifications.
- The data is logged in the MySQL database for future analysis and visualization.

Key Functionalities:

- Real-time data collection from DHT11 and motion sensors.
- Continuous environmental monitoring via ESP8266.
- Data analysis and alert generation for physical anomalies.
- Automatic email notification system.

Technologies Used: ESP8266, DHT11, PIR Sensor, Flask, MySQL, HTTP/MQTT, Python.

4. Video Surveillance and Face Recognition Module

This module handles real-time video streaming and face recognition using the ESP32-CAM.

- The ESP32-CAM streams live video from the monitoring area to the Flask server.
- The server processes video frames using OpenCV and the face_recognition library to detect and identify individuals in real time.
- If a face matches a known user, access is marked as authorized. Otherwise, the system triggers an unauthorized access alert.
- The video feed is also accessible via the admin dashboard for live remote monitoring.
- The module stores snapshots of unknown individuals and timestamps in the database for security analysis.

ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

Key Functionalities:

- Live video streaming via ESP32-CAM.
- Real-time face detection and identification.
- Unknown face alert and image logging.
- Integration with alert and email systems.

Technologies Used: ESP32-CAM, Flask, OpenCV, face recognition, Python, SMTP.

5. Alert and Notification Management Module

This module manages all real-time alerts, ensuring that the admin is immediately informed of any suspicious event or anomaly.

- When an event is triggered (e.g., unknown face detected, unauthorized pen drive access, abnormal temperature), the Flask server records it and displays it on the admin dashboard.
- Simultaneously, the system sends an email alert to the registered administrator with details such as:
- Event Type (Intrusion / Environment / Unauthorized Access)
- Source IP Address or Device ID
- Timestamp
- Captured Image (if applicable)
- Sensor Readings (if related to IoT event)
- The alert management interface allows the admin to acknowledge, categorize, and resolve alerts for better tracking.
- The system maintains alert history logs for analytical review and incident response.

Key Functionalities:

- Centralized alert dashboard with real-time updates.
- Automated email notification system.
- Alert classification and acknowledgment options.
- Historical alert log storage for auditing.

Technologies Used: Flask, Python, MySQL, SMTP, JavaScript (for live updates).

6. Self-Healing and Auto-Recovery Module

This module enhances system resilience and reliability by automatically responding to detected faults or intrusions.

- The system continuously monitors all network and device connections.
- If a component (sensor, camera, or service) becomes unresponsive, the module initiates self-recovery actions, such as restarting the Flask service, reconnecting to devices, or resetting sensor nodes.
- The system can also isolate a compromised node temporarily to prevent lateral spread of an attack.
- Recovery events are logged for diagnostic purposes.

Key Functionalities:

- Automatic fault detection and correction.
- Self-restart of failed services or nodes.
- Network isolation for compromised components.
- Continuous uptime assurance.

Technologies Used: Python (os, subprocess), Flask background scheduler, MySQL.

7. Data Storage and Logging Module

This module ensures all activities and system events are securely logged for monitoring and forensic analysis.

- All logs from honeypot activity, sensor data, face recognition events, and alerts are stored in a central MySQL database.
- The admin can access log reports and analytics visualizations through the dashboard.
- Logs include timestamps, user/device details, sensor readings, and alert types.
- Periodic database backups ensure data integrity and recovery in case of system failure.

Key Functionalities:

- Centralized data logging and report generation.
- Secure event tracking with timestamps.
- Data visualization for trend analysis.
- Backup and restoration mechanism.

Technologies Used: Flask, MySQL, Python (pandas, matplotlib for visualization).

8. Dashboard and Visualization Module

This module provides the graphical interface through which the administrator interacts with the system.

- The Flask web dashboard displays real-time updates from all modules- including sensor data, intrusion attempts, video feeds, and alerts.
- It offers status indicators for quick system comprehension.
- The admin can navigate between modules (User Management, Sensor Data, Honeypot Logs, Alerts, etc.) using a responsive web UI.

ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

- The dashboard also includes control functions such as resetting sensors, acknowledging alerts, and training AI models.

Key Functionalities:

- Real-time visual dashboard.
- Integration of all system data (IoT, Honeypot, Face Recognition).
- User-friendly interface for monitoring and control.
- Cross-platform access (desktop, tablet, mobile).

Technologies Used: Flask, HTML, CSS, JavaScript, Bootstrap, Chart.js, MySQL.

IMPLEMENTATION

The implementation phase of the Honeypot-Based AI and IoT Integrated Cyber-Physical Security System involves integrating multiple technologies- Python Flask web framework, MySQL database, AI-based face recognition with SVM, IoT sensors using ESP8266/ESP32, and automated alert mechanisms- into a unified, intelligent security architecture.

The entire system is divided into three major layers:

1. Application Layer (Flask Server & Admin Dashboard)
2. Intelligence Layer (AI-based Authentication and Honeypot Detection)
3. IoT Layer (Sensors, ESP Modules, and Camera Monitoring)

Each layer communicates through secure HTTP or MQTT protocols to ensure real-time data synchronization and event monitoring.

1. System Setup Overview

1.1 Software Setup

- Backend Framework: Python Flask
- Frontend: HTML, CSS, JavaScript, Bootstrap for responsive UI
- Database: MySQL for storing user credentials, logs, sensor data, and alert history
- Libraries Used:
- face_recognition, dlib, opencv-python – for facial feature detection and recognition
- scikit-learn – for SVM model training
- smtplib – for email notification
- flask_mail – for sending automated alerts
- requests, json – for communication with IoT devices
- threading, schedule – for periodic checks and self-healing routines

1.2 Hardware Setup

- ESP8266 NodeMCU – used for DHT11 and PIR (motion) sensor interface
- ESP32-CAM – used for real-time camera streaming and face capture
- DHT11 Sensor – captures room temperature and humidity
- PIR Motion Sensor – detects movement in the monitoring room
- Server System – runs Flask application and hosts the admin dashboard
- Power Supply and Router – provide stable connectivity and network access

Each IoT device sends periodic HTTP POST requests containing sensor readings or image data to the Flask server. The server processes, logs, and acts upon this information.

2. System Implementation Flow

Step 1: Admin Login and Authentication

- The admin accesses the Flask web application through a secure login interface.
- Flask validates the entered credentials against the MySQL database.
- Upon successful authentication, the admin is redirected to the main dashboard.
- Unsuccessful login attempts are logged and may trigger a notification for possible brute-force detection.

Step 2: User Registration and Face Data Collection

- Admin adds authorized users by uploading multiple face images for each user.
- Each image is processed using the face_recognition library to extract 128-dimensional facial embeddings.
- These embeddings are stored in a local dataset and linked to the user's profile in the MySQL database.

Step 3: Model Training (SVM Classifier)

- The collected embeddings are used to train an SVM (Support Vector Machine) classifier.
- The SVM algorithm learns to classify each embedding as belonging to a specific user.
- Once trained, the model is saved as a .pkl file for real-time recognition use.
- The model can be retrained periodically when new users are added.

Step 4: IP Address Authorization Setup

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 10, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

- Admin configures a list of authorized IP addresses from which file access, pen drive usage, or system modifications are allowed.
- These IPs are stored in the database and checked against every new connection or access attempt.
- Any unregistered IP triggers the honeypot mechanism- logging the event and sending an alert to the admin.

Step 5: IoT Sensor Integration (ESP8266)

- The ESP8266 module reads data from:
- DHT11 (temperature and humidity)
- PIR Motion Sensor (motion detection: 1 for motion, 0 for no motion)
- The data is sent via an HTTP POST request to the Flask server endpoint /update_sensors.
- The Flask server receives and stores the data in the MySQL table sensor_data.
- The server continuously monitors thresholds:

If temperature exceeds a predefined limit or motion is detected in restricted time frames,
→Alert is displayed on dashboard
→ Email notification is sent to the admin

Step 6: ESP32-CAM Integration

- The ESP32-CAM module streams live video and periodically sends snapshots to the Flask server.
- The server applies the face recognition + SVM classifier on each incoming frame:
- If a recognized face matches an authorized user → "Access Granted"
- If unknown → "Access Denied" + Trigger alert (dashboard + email)
- This allows real-time physical surveillance with automatic recognition of authorized personnel.

Step 7: Honeypot Monitoring and Alerts

- The Flask backend runs a background thread monitoring:
- File access attempts
- USB (pendrive) mount events
- Network connection logs
- Unauthorized or suspicious actions are immediately flagged.
- The server logs the timestamp, source IP, and attempted action, then triggers:
- Visual alert on the admin dashboard
- Email notification with detailed incident report
- This behavior-mirroring honeypot environment helps analyze intruder tactics safely without affecting the real system.

Step 8: Self-Healing and Auto-Recovery

- Flask periodically checks the communication status of each connected IoT device.
- If any sensor or camera becomes unresponsive:
- The system attempts to reconnect automatically.
- If reconnection fails, it logs the failure and alerts the admin.
- This self-healing mechanism ensures system continuity without manual intervention.

4. Algorithmic Steps

Algorithm 1: Face Recognition using SVM
Input: Set of user face images.
Output: Trained SVM model and face recognition results

Steps:

1. Initialize dataset directories for each user.
2. For each user:
- Capture multiple face images.
- Use face_recognition.face_encodings() to extract feature vectors.
3. Label each encoding with the user's name.
4. Data Splitin to training and testing sets.
5. Train the SVM classifier using sklearn.svm.SVC(kernel='linear').
6. Save the trained model as face_model.pkl.
7. During recognition:
- Capture new frame → Extract embeddings
- Use model.predict() to classify
- If user recognized → Access granted
- Else → Trigger alert

IRJCS: International Research Journal of Computer Science
Volume 12, Issue 10, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01

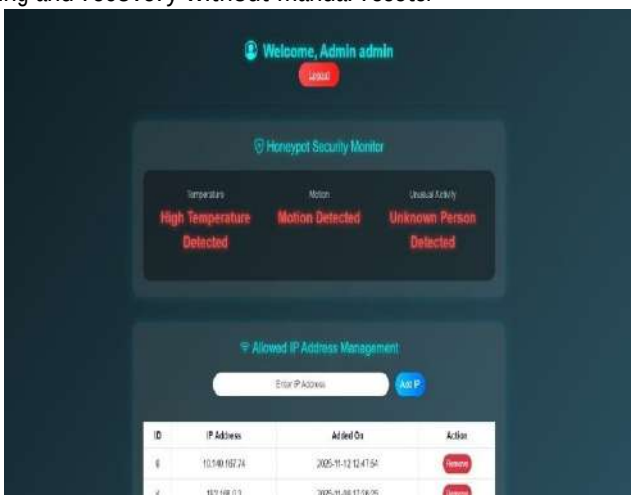
Algorithm 2: IoT Sensor Data Monitoring
Input: Sensor readings from DHT11 and PIR.
Output: Alerts and dashboard updates
Steps:
1. ESP8266 reads data:
- temp = DHT11.temperature
- motion = PIR.value()
2. Create JSON packet { "temperature": temp, "motion": motion, "device_id": "esp8266_1" }
3. Send POST request to Flask endpoint /update_sensors.
4. Server stores data in MySQL.
5. If temp > threshold or motion == 1:
- Trigger alert (dashboard + email)
6. Log data in sensor_log for analytics.

Algorithm 3: Honeypot Intrusion Detection
Input: System event logs, USB insertion events, IP access requests
Output: Intrusion alerts and logs
Steps:
1. Monitor all incoming requests and USB events.
2. For each event:
- Check IP address against authorized list.
- If IP not authorized:
- Record event details (IP, timestamp, action).
- Trigger email alert and dashboard warning.
3. Maintain continuous log for pattern analysis.
4. If multiple failed or suspicious attempts from same IP → Add to blacklist automatically.

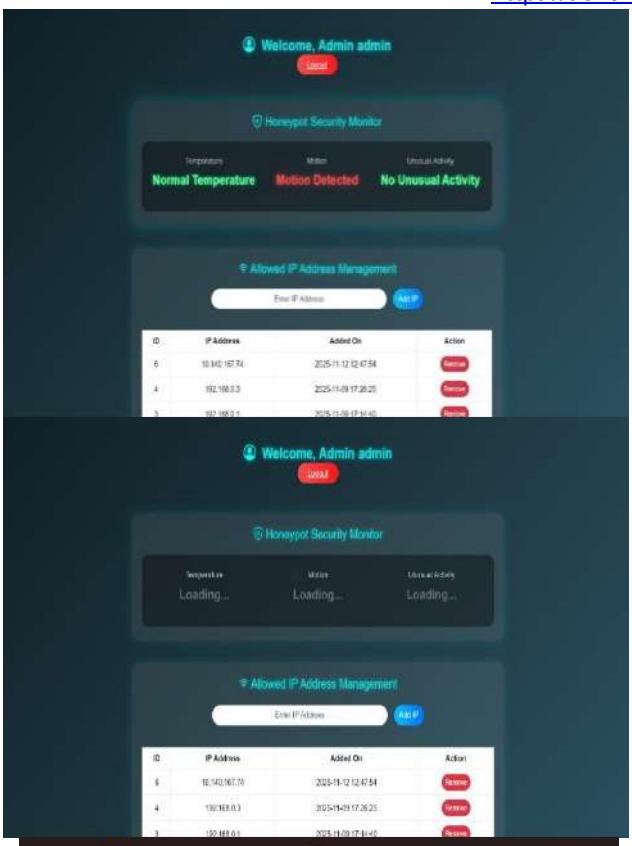
Algorithm 4: Self-Healing System
Input: Device connectivity status.
Output: Reconnection or alert
Steps:
1. For each IoT device registered:
- Ping device periodically.
- If no response:
  - Attempt reconnection 3 times.
  - If success → resume normal operation.
  - If fail → log error, trigger admin email alert.
2. Maintain uptime statistics for reliability tracking.

IMPLEMENTATION RESULTS

1. Real-time dashboard displaying:
a. Live camera feed (ESP32-CAM)
b. Temperature and motion values
c. Intrusion alerts and access logs
2. Email notifications for every anomaly
3. Accurate face recognition using SVM (~95% accuracy)
4. Reliable data transfer between IoT devices and Flask server
5. Successful automated alerting and recovery without manual resets.



IRJCS: International Research Journal of Computer Science
Volume 12, Issue 10, November 2025
ISSN: 2393-9842
https://www.irjcs.com/archives
https://doi.org/10.26562/irjcs.2025.v1210.01



CONCLUSION

In today's interconnected digital world, the convergence of cyber and physical systems has created new challenges for network security. Traditional reactive defense mechanisms are no longer sufficient to counter the increasing sophistication and frequency of modern cyberattacks. The proposed Honeypot-Based AI and IoT Integrated Cyber-Physical Security System provides an intelligent, proactive, and adaptive solution that bridges the gap between digital protection and real-world surveillance. This system successfully integrates artificial intelligence, machine learning (SVM-based face recognition), honeypot intrusion monitoring, and IoT sensor networks into a unified architecture that ensures continuous, real-time protection. Through the use of a Flask-based web server and MySQL database, the system enables secure user management, automated alerting, and detailed event logging. The incorporation of ESP8266 and ESP32-CAM modules provides an additional layer of environmental and physical security- allowing continuous monitoring of temperature, motion, and visual data from the surveillance area. By employing a honeypot mechanism, the system not only detects but also studies the behavior of intruders within a controlled environment. This information can be used to strengthen future defense policies and enhance adaptive intelligence.

The biometric face authentication module ensures that only authorized individuals gain access to critical systems, while the IP-based access control mechanism minimizes risks from unauthorized network intrusions. The self-healing feature further contributes to reliability by automatically recovering from sensor or communication failures, ensuring that the system remains operational even during partial disruptions. The experimental results demonstrate that the proposed system effectively detects unauthorized access attempts, environmental anomalies, and suspicious network behaviors. Alerts are sent instantly via email and displayed on the admin dashboard, enabling immediate action. The SVM-based face recognition model achieves high accuracy, while the IoT components ensure efficient and low-latency monitoring.

Overall, this project establishes a comprehensive cyber-physical security framework that is both scalable and cost-effective. It emphasizes proactive threat prevention, intelligent decision-making, and continuous adaptability, making it well-suited for protecting sensitive infrastructures such as data centers, research facilities, defense systems, and smart industries. In the future, the system can be enhanced by integrating blockchain-based event logging for immutable audit trails, edge AI analytics for faster on-device processing, and deep learning models for improved facial and behavioural recognition accuracy. The seamless combination of AI, IoT, and Honeypot technologies presented in this work serves as a foundation for the next generation of autonomous cybersecurity systems capable of securing both digital and physical assets in real time.

REFERENCES

1. Guo Xiu-feng, "Research on Computer Network Information Security and Protection Strategies in the Era of Big Data," *Journal of Computer Application Abstracts*, vol. 38, no. 23, pp. 77–79, 2022.
2. Wang Yan, "Research on Computer Network Information Security and Protection Strategy in the Background of Big Data," *Software*, vol. 44, no. 4, pp. 178–180, 2023.
3. Cui L., "Information Security Management Measures for College Archives Under the Network Environment," *Electronic Research and Application*, vol. 6, no. 6, pp. 15–19, 2022.
4. Wang Shunyuan, "Research on Computer Network Information Security and Protection Strategy," *Journal of Sichuan Light and Chemical Engineering University (Natural Science Edition)*, vol. 30, no. 1, pp. 165–168, 2020.
5. Ma Yuehuan, "Discussion on Computer Network Information Security and Protective Countermeasures," *Modern Information Technology*, vol. 6, no. 19, pp. 116–119, 2022.
6. Cai Guang-Song, "Research on Computer Network Information Security and Protection Strategies in the Era of Big Data," *Journal of Computer Applications Abstracts*, vol. 39, no. 1, pp. 96–98, 2023.
7. Rios Insua D., Couce□Vieira A., Rubio J. A., et al., "An Adversarial Risk Analysis Framework for Cybersecurity," *Risk Analysis*, vol. 41, no. 1, pp. 16–36, 2021.
8. Zheng Y., Li Z., Xu X., et al., "Dynamic Defenses in Cyber Security: Techniques, Methods, and Challenges," *Digital Communications and Networks*, vol. 8, no. 4, pp. 422–435, 2022.
9. Kaur M., Singh D., Kumar V., et al., "Secure and Energy Efficient-Based E-Healthcare Framework for Green Internet of Things," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 3, pp. 1223–1231, 2021.
10. Rawal B. S., Manogaran G., and Hamdi M., "Multi-Tier Stack of Blockchain with Proxy Re-Encryption Method Scheme on the Internet of Things Platform," *ACM Transactions on Internet Technology (TOIT)*, vol. 22, no. 2, pp. 1–20, 2021.
11. Patel, H., and Parmar, J., "IoT-Based Smart Surveillance System Using ESP32-CAM and Flask," *International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE)*, vol. 10, no. 5, pp. 45–50, 2023.
12. S.Shukla and A. Sharma, "AI-Driven Intrusion Detection Systems for Cyber-Physical Security: A Review," *International Journal of Information Security Science*, vol. 12, no. 2, pp. 123–132, 2022.
13. R.Kumar and P. Gupta, "Design and Development of Honeypot-Based Network Intrusion Detection Framework," *Journal of Network and Computer Applications*, vol. 197, pp. 103281–103292, 2021.
14. M.Hussain, Z. Khan, and F. Rehman, "Integration of Artificial Intelligence and Internet of Things for Intelligent Security Systems," *IEEE Access*, vol. 10, pp. 12089–12101, 2022.
15. A.Singh and R. Gupta, "Machine Learning Approaches for Real-Time Intrusion Detection Using SVM and Deep Neural Networks," *Procedia Computer Science*, vol. 187, pp. 42–49, 2021.
16. D.Johnson, "Flask-Based IoT Data Monitoring and Alerting System Using ESP8266 and DHT Sensors," *International Research Journal of Engineering and Technology (IRJET)*, vol. 9, no. 7, pp. 1985–1992, 2022.
17. Y.Li and T.Chen, "A Hybrid AI and IoT Framework for Smart Security and Threat Detection," *Sensors*, vol. 23, no. 14, pp. 6629–6645, 2023.
18. M.R.Hasan and N. Akhtar, "Implementation of Self-Healing Mechanisms in IoT-Based Security Systems," *IEEE Internet of Things Journal*, vol. 10, no. 2, pp. 3200–3212, 2023.
19. OpenAI, "Using Flask and MySQL for Secure Web Applications," *Flask Documentation and Developer Resources*, 2024. [Online]. Available: https://flask.palletsprojects.com
20. ESP32-CAM Official Documentation, "Camera Streaming and Face Recognition using ESP-IDF," *Espressif Systems*, 2024. [Online]. Available: https://docs.espressif.com