IoT-Based Secure Access Control System with Real-Time Intrusion Detection and Alert Mechanism

Ritesh Ekbote(riteshekbote@gmail.com),

Chetan Dhumal(chetandhumal2005@gmail.com),

Bhagyashri Koli(bhagyashrikoli821@gmail.com),
Muskan Mulani(muskanmulani222@gmail.com)
Aishwarya Hosale(aishwaryakeshi@gmail.com)

Department of Computer Engineering, A.G. Patil Institute of Technology
Academic Year: 2025-26 | Guide: A. P. Hosale(aishwaryakeshi@gmail.com)|HoD: Prof. S. V. Kulkarni

Abstract

The increasing reliance on digital systems has created a critical need for robust access control and real-time intrusion detection mechanisms. This paper presents an IoT-Based Secure Access Control System that detects and responds to unauthorized login attempts in real time. The system employs JSON Web Token (JWT) authentication with IP binding, automatic IP-based blocking after repeated failed attempts, account-level locking to prevent credential-stuffing attacks, CSRF token protection, session fixation prevention, and real-time email alerts via PHPMailer over Gmail SMTP. A forgot-password module using time-limited secure tokens enables safe credential recovery. A centralized admin dashboard provides live monitoring of login history, blocked IPs, trusted IPs, and unknown device alerts. Physical intrusion alerts are delivered through an Arduino UNO connected to a relay module, triggered by a Python monitoring script that polls the server. The system is deployed on a live web server and demonstrates practical applicability for securing IoT-connected environments.

Keywords: *IoT Security, Access Control, JWT Authentication, Intrusion Detection, IP Blocking, Account Locking, CSRF Protection, Email Alert, Admin Dashboard, Arduino, Relay Module*

1. Introduction

The Internet of Things (IoT) has revolutionized the way devices interact and share data across diverse environments. However, this widespread connectivity has simultaneously introduced significant security challenges. Unauthorized access to IoT management interfaces can lead to data breaches, system disruptions, and severe privacy violations. Traditional username-password authentication mechanisms are often insufficient to address modern threats, particularly when systems are exposed to the public internet and subjected to automated brute-force attacks.

This paper proposes a comprehensive, multi-layered security system that integrates JWT-based session authentication with IP binding, real-time IP tracking and automatic blocking, account-level locking, CSRF token validation, session fixation prevention, secure password recovery via time-limited email tokens, and physical IoT alerting via an Arduino-controlled relay module. The system is designed to be lightweight, deployable on standard web hosting infrastructure, and accessible via any modern browser without requiring specialized hardware beyond a standard Arduino board.

The primary objective is to provide a practical, cost-effective, and academically demonstrable solution for securing access to IoT management interfaces. The system bridges software security mechanisms with tangible physical hardware responses, making it uniquely suitable for demonstrating IoT security principles in an educational setting.

The remainder of this paper is organized as follows: Section 2 reviews related work, Section 3 describes the system architecture, Section 4 details the implementation, Section 5 presents experimental results, and Section 6 concludes with future directions.

2. Literature Review

Numerous studies have addressed IoT security from different perspectives, each contributing important insights while leaving certain gaps that the proposed system addresses.

Kumar et al. (2020) proposed a role-based access control (RBAC) model for IoT gateways, demonstrating that fine-grained access policies reduce unauthorized access incidents by up to 60%. However, their system lacked real-time alert mechanisms and physical hardware integration, limiting its effectiveness in environments requiring immediate operator response.

Sharma and Patel (2021) investigated token-based authentication for IoT APIs and concluded that JWT offers superior stateless authentication compared to session cookies, particularly for resource-constrained environments. Their work, however, did not address IP-level intrusion tracking or account locking, leaving the system vulnerable to distributed brute-force attacks from multiple IP addresses.

Singh et al. (2022) developed an IP blacklisting mechanism for web-based IoT dashboards but relied on manual administrator intervention for unblocking and did not integrate physical alert mechanisms. The system proposed in this paper automates the blocking process with configurable time-based unblocking, adds account-level locking as a secondary defense layer, and bridges software detection with physical hardware response.

Existing systems commonly lack integration of all critical security components: token authentication, IP tracking, account locking, CSRF protection, secure password recovery, and physical alerting. The proposed system addresses this gap by combining all these mechanisms into a unified, deployable solution that serves both practical security and educational demonstration purposes.

3. System Architecture

The proposed system follows a three-tier client-server architecture consisting of a presentation layer, an application logic layer, and a database layer, with an additional physical IoT hardware layer for real-world alert delivery.

3.1 Presentation Layer

The presentation layer consists of a secure login page and a comprehensive admin dashboard, both implemented using HTML5, CSS3, and PHP. The login page features a dark-themed glassmorphism user interface with CSRF token embedding and real-time error feedback. The dashboard provides four monitoring panels: login history, blocked IPs, trusted IPs, and unknown device alerts with approve or block actions. An auto-refresh mechanism updates the dashboard every 30 seconds for near-real-time monitoring.

3.2 Application Logic Layer

The application layer is implemented entirely in PHP and handles authentication, session management, IP evaluation, CSRF validation, and alert dispatch. JWT tokens are generated upon successful login and stored in HTTP-only cookies with SameSite=Strict policy and in PHP sessions as a redundant verification layer. Each subsequent request to the dashboard verifies the token signature, expiry, and IP binding simultaneously. Session fixation attacks are mitigated by regenerating the session ID upon first initialization.

3.3 Database Layer

A MySQL database stores six tables: users, login_attempts, blocked_ips, trusted_ips, login_logs, and alert_flags. The users table includes is_locked and failed_attempts columns for account-level locking. A seventh table, password_resets, stores time-limited tokens for the forgot-password recovery flow. Indexed columns on ip_address and timestamp fields ensure fast query performance for real-time tracking operations.

3.4 IoT Hardware Layer

A Python monitoring script running on the administrator's laptop polls the check_alert.php endpoint every five seconds. When the alert flag is set by the PHP backend, the script sends a control signal over a serial connection to an Arduino UNO microcontroller. The Arduino drives a relay module, whose built-in LEDs provide immediate visual indication of the alert type. The green status LED remains active during normal operation, and the red alert LED activates upon intrusion detection. Three distinct hardware behaviors correspond to three threat levels: a single brief pulse for a wrong-password event, rapid continuous toggling for an unknown IP login, and a sustained five-second activation for a confirmed brute-force block.

This architecture is designed for future extension with an ESP32 microcontroller, which would eliminate the dependency on the administrator's laptop by enabling 24/7 independent WiFi-based polling of the server endpoint.

Figure 1: System Architecture Overview

Layer	Component	Technology
Presentation	Login Page + Admin Dashboard	HTML5, CSS3, PHP
Application Logic	Auth, IP Tracking, Alerts, CSRF	PHP, JWT, PHPMailer
Database	7 MySQL Tables	MySQL / PDO
IoT Hardware	Arduino UNO + Relay Module	Python, C++ (Arduino)
Future Extension	ESP32 WiFi Microcontroller	Arduino C++, HTTP

4. Implementation

4.1 Authentication Module

User authentication is implemented using PHP's password_verify() function with bcrypt hashing. Upon successful credential verification, a JWT token is generated using the firebase/php-jwt library with HS256 signing. The token payload includes the username, client IP address, user agent string,

issued-at timestamp (iat), and expiry set to one hour (exp = iat + 3600). The IP address and user agent are embedded in the token to prevent token reuse from different network locations or browsers, providing an additional layer of session binding beyond standard expiry enforcement.

4.2 CSRF Protection and Session Security

Each login form submission is validated against a server-generated CSRF token stored in the PHP session. The token is regenerated after every POST request to prevent token reuse attacks. Session fixation is mitigated by calling session_regenerate_id(true) upon session initialization, ensuring that an attacker who fixes a session ID before authentication cannot reuse it post-authentication. HTTP-only and SameSite=Strict cookie flags further harden session handling against cross-site request forgery and cookie theft.

4.3 IP Tracking, Blocking, and Account Locking

Every login attempt is recorded in the login_attempts table with the associated IP address and timestamp. The system queries failed attempts within a 15-minute sliding window using a time-based SQL filter. Upon reaching three consecutive failures, the IP is automatically inserted into the blocked_ips table with an unblock timestamp set 30 minutes in the future. Simultaneously, if the targeted username exists in the database, the corresponding user account is locked by setting is_locked = 1, preventing login even if the attacker subsequently changes their IP address. This dual-layer defense effectively defeats both IP-rotation and credential-stuffing attack strategies. Blocked IPs are evaluated at the beginning of every login request before any credential verification, ensuring computationally efficient denial of service to known attackers.

4.4 Unknown IP Detection

When a user authenticates successfully from an IP address not present in the trusted_ips table, the system classifies the login as an unknown IP event. The alert_flags table is updated to signal the Python monitoring script, and an email notification is dispatched to the administrator containing the username, IP address, and timestamp. An alert spam prevention mechanism limits email dispatch to a maximum of five alerts per IP per minute, preventing notification flooding during automated attacks. The administrator can approve or permanently block the unknown IP directly from the dashboard interface.

4.5 Forgot Password Module

The forgot-password feature enables secure credential recovery without exposing sensitive information. When a recovery request is submitted, the system queries the users table for the provided email address. If a matching record exists, a cryptographically secure 64-character hexadecimal token is generated using PHP's random_bytes(32) function and stored in the password_resets table alongside the email address and an expiry timestamp set 30 minutes in the future. A reset link containing the token is dispatched to the user's registered email address via PHPMailer. The reset page validates the token against the database, confirms it has not expired and has not been previously used, and then permits the user to set a new bcrypt-hashed password. Upon successful reset, the token is marked as used to prevent replay attacks.

4.6 Email Alert System

Email notifications are dispatched using PHPMailer with Gmail SMTP over TLS on port 587. The sendAlert() utility function is centrally defined in the configuration file and reused across multiple system components. Three alert categories are implemented: wrong-password events update the hardware flag without generating an email to prevent notification flooding; unknown IP login events generate an administrator alert with full contextual details; and confirmed brute-force blocks generate a high-priority intrusion alert. The forgot-password module uses a separate direct PHPMailer invocation to direct the reset link to the requesting user's email rather than the administrator's address.

4.7 Physical IoT Alert Mechanism

The Python alert_checker.py script executes on the administrator's laptop and polls check_alert.php every five seconds via HTTP GET. Upon receiving a non-zero flag value, the script transmits a single-character control signal over a USB serial connection to the Arduino UNO at 9600 baud. The Arduino firmware interprets three signal values corresponding to the three threat levels and drives the relay module accordingly. Flag value 1 produces one 300-millisecond relay pulse for a wrong-password event. Flag value 2 produces continuous 200-millisecond on-off toggling for five seconds to indicate an unknown IP login. Flag value 3 holds the relay energized for five full seconds to indicate a confirmed brute-force intrusion. The relay module's built-in green LED remains illuminated during normal standby, and the red LED activates whenever the relay coil is energized, providing unambiguous visual differentiation between alert states.

4.8 Security Hardening

The system incorporates multiple additional security measures. Directory listing is disabled via .htaccess configuration. Sensitive files including config.php and the vendor directory are blocked from direct browser access, returning 403 Forbidden responses to directory traversal attempts. All database interactions use PDO prepared statements with parameterized queries to eliminate SQL injection vulnerabilities. JWT tokens are verified against the originating IP address on every dashboard request to detect token theft and replay. Error messages presented to end users are deliberately generic to avoid leaking system internals to potential attackers.

5. Results and Discussion

The system was deployed on a live web hosting environment at adminpanel.genzctf.xyz and subjected to comprehensive functional testing across all implemented security scenarios.

5.1 Authentication and Session Security

JWT token generation and validation operated correctly across all tested scenarios. Tokens from different IP addresses were correctly rejected by the dashboard verification logic. The CSRF protection mechanism successfully blocked all forged POST requests. Session regeneration upon initialization confirmed that session fixation attacks were mitigated. The HTTP-only and SameSite cookie flags were verified via browser developer tools.

5.2 Intrusion Detection Performance

The IP blocking mechanism successfully triggered after exactly three consecutive failed login attempts within the 15-minute sliding window, with the blocking event logged and an email alert dispatched within two seconds in all tested cases. Account locking correctly activated alongside IP blocking, confirming the dual-layer defense behavior. Unknown IP detection correctly identified logins from new devices, including connections routed through VPN exit nodes, generating alerts in all tested cases. The hardware relay activated within the five-second polling interval in every test.

5.3 Password Recovery

The forgot-password flow was validated end-to-end. Reset emails were delivered to the requesting user's registered email address within 10 seconds of form submission. Expired tokens were correctly rejected. Previously used tokens were rejected on resubmission, confirming replay attack prevention. Incorrect tokens returned appropriate error messages without exposing database details.

5.4 Hardware Response Verification

All three hardware alert behaviors were verified against their corresponding threat events. The wrong-password single pulse, unknown IP rapid blink, and brute-force sustained activation were clearly distinguishable during testing, confirming the differentiated physical alerting design.

Table 1: Test Results Summary

Test Scenario	Expected Behavior	Result	Hardware Response
Wrong password (1x)	Flag=1, error message	PASS	Relay: 1 short pulse
Wrong password (3x)	IP blocked, account locked, email alert	PASS	Relay: 5s sustained ON
Unknown IP login	Flag=2, admin email alert	PASS	Relay: rapid blink 5s
Correct password, trusted IP	JWT issued, dashboard access	PASS	No alert
Expired JWT token	Redirect to login page	PASS	No alert
CSRF token mismatch	Request rejected	PASS	No alert
Forgot password (valid email)	Reset email delivered	PASS	No alert
Forgot password (invalid email)	Generic success message	PASS	No alert
Expired reset token	Error, request new link	PASS	No alert
Directory traversal attempt	403 Forbidden	PASS	No alert

6. Conclusion and Future Work

This paper presented a comprehensive IoT-Based Secure Access Control System integrating JWT authentication with IP binding, automatic IP blocking, dual-layer account locking, CSRF token protection, session fixation prevention, real-time email alerting, secure forgot-password recovery, and physical hardware alerting via an Arduino UNO relay module. The system successfully

demonstrates that robust, multi-layered security mechanisms can be implemented using open-source technologies on standard hosting infrastructure, making it both practically deployable and educationally demonstrable for small-scale IoT environments.

The physical hardware layer effectively bridges the software intrusion detection system with tangible real-world response, providing immediate visual differentiation between three distinct threat severity levels through the relay module's built-in LED indicators.

Future work will focus on the following enhancements:

- Integration of an ESP32 microcontroller to replace the Python script dependency, enabling 24/7 autonomous server polling without requiring the administrator's laptop to remain active.
- Implementation of HTTPS with SSL/TLS encryption to protect credentials and tokens in transit.
- Integration of two-factor authentication (2FA) via time-based one-time passwords (TOTP) for additional login security.
- Extension of the physical IoT layer with RFID-based proximity access control for physical door or device access management.
- Development of a mobile administrator application for push notification delivery of intrusion alerts.
- Implementation of device fingerprinting combining browser, operating system, screen resolution, and timezone parameters to strengthen unknown-device detection beyond IP-based tracking.

References

[1] Kumar, R., Sharma, A., & Gupta, P. (2020). Role-Based Access Control for IoT Gateway Systems. *International Journal of Network Security*, 22(4), 112-119.

[2] Sharma, N., & Patel, D. (2021). JWT-Based Stateless Authentication for IoT APIs: A Comparative Study. *IEEE Access*, 9, 45231-45242.

[3] Singh, V., Verma, K., & Mishra, S. (2022). IP Blacklisting Mechanisms for Web-Based IoT Dashboards. *Proceedings of the International Conference on IoT Security*, 88-95.

[4] OWASP. (2023). Authentication Cheat Sheet. Open Web Application Security Project. Available: https://cheatsheetseries.owasp.org

[5] Firebase. (2023). PHP-JWT Library Documentation. Available: https://github.com/firebase/php-jwt

[6] PHPMailer. (2023). PHPMailer SMTP Integration Guide. Available: https://github.com/PHPMailer/PHPMailer

[7] Arduino. (2023). Arduino UNO Reference Manual. Available: https://docs.arduino.cc/hardware/uno-rev3

[8] OWASP. (2023). Cross-Site Request Forgery Prevention Cheat Sheet. Available: https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html