A Secure IoT-Based Smart Irrigation System Using NodeMCU and Raspberry Pi against Replay and SQL Injection Attacks

Bheemarasetti Amrutha
Master of Information Technology, Department of Information Technology and Computer Applications, Andhra University College of Engineering (A), Andhra University, Visakhapatnam, Andhra Pradesh, India

Abstract—**The rapid expansion of the Internet of Things (IoT) has introduced significant cybersecurity challenges, especially as more low-cost embedded devices connect to sensitive networks. This paper presents a practical IoT implementation that uses NodeMCU and Raspberry Pi to collect temperature and humidity data via a DHT11 sensor. The sensor readings are transmitted over the HTTP protocol and stored in a MariaDB database hosted on the Raspberry Pi. To evaluate the system's resilience, penetration testing was performed using Kali Linux, targeting two major threats: SQL Injection and Replay Attacks. These threats were executed individually on the Raspberry Pi server to expose and analyse real-world vulnerabilities. Following this, various security measures—such as SHA1-based signature validation, timestamp verification, and the use of prepared SQL statements—were applied to both the Arduino and server-side Python programs. The results clearly demonstrated that the applied security mechanisms effectively mitigated the identified vulnerabilities. This study highlights the importance of securing communication and data integrity in real-time IoT applications, and provides a reference model for developers and researchers aiming to deploy secure IoT systems using affordable, open-source tools.**

***Key Words*: DHT11, HTTP Protocol, Kali Linux MariaDB, NodeMCU, Raspberry Pi.**

I. INTRODUCTION

The Internet has become the central platform for information exchange, enabling people to communicate and share data globally without the restrictions of distance or time. This progress has given rise to the Internet of Things (IoT), which connects a wide range of devices such as sensors, controllers, and embedded systems to collect and transfer data seamlessly. IoT technology has found applications in areas like smart agriculture, healthcare, transportation, industrial automation, and environmental monitoring, making everyday systems more efficient and intelligent. Despite its advantages, IoT introduces major security concerns [1]. Many IoT devices are resource-constrained and often lack strong protective mechanisms, leaving them exposed to attacks. Cybercriminals can take advantage of these weaknesses to steal data, disrupt operations, or gain unauthorized control over systems [2]. As a result, securing IoT communications and protecting stored data has become an essential requirement. To address these risks, frameworks such as the OWASP Internet of Things Security Verification Standard (ISVS) define best practices and guidelines for building secure IoT systems. These standards help developers ensure confidentiality, integrity, and reliability in IoT deployments [3].

This project demonstrates a secure IoT model where a NodeMCU connected to a DHT11 sensor captures temperature and humidity readings and sends them to a Raspberry Pi server through the HTTP protocol**.** The data is stored in a MariaDB database for monitoring and analysis [4]. To evaluate the system's resilience, two common security threats—Replay Attack and SQL Injection—are implemented. The system is first tested against these attacks using Kali Linux penetration testing tools, and then secured with suitable defense techniques [5][6]. Finally, penetration testing is repeated to validate the effectiveness of the applied security measures [7][8]. Show in Fig.1.

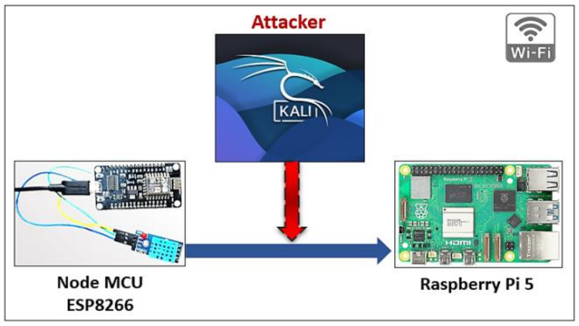

Fig.1. Testing Setup

Section II reviews the top 10 OWASP IoT Security threats and section III focuses. Specifically on two of the most significant and practically exploitable threats-Replay attack and SQL Injection-highlighting their impact, exploitation methods and mitigation strategies.

II. OWASP IOT SECURITY THREATS

The Open Web Application Security Project (OWASP) has identified a set of critical security concerns that affect IoT systems [4]. These issues highlight common weaknesses in IoT design, deployment, and maintenance, which attackers can exploit to compromise devices and networks. The major security threats are summarized below:

A. Weak, Guessable, or Hardcoded Passwords:

Many IoT devices are shipped with weak or default login credentials that remain unchanged by users. Such predictable authentication mechanisms allow attackers to easily gain unauthorized access and take control of the device.

B. Insecure Network Services:

Services running on IoT devices, particularly those exposed to the internet may be vulnerable to attacks that enable remote access, unauthorized data extraction, or manipulation of device behavior.

C. Insecure Ecosystem Interfaces:

IoT systems often interact with web portals, cloud platforms, mobile applications, or backend APIs. If these interfaces lack strong authentication, authorization, encryption, or proper input validation, they can be exploited to compromise the entire ecosystem.

D. Lack of Secure Update Mechanism:

Many IoT devices do not verify the authenticity of firmware updates or lack mechanisms to deliver them securely. This creates opportunities for attackers to install malicious updates or prevent devices from receiving critical patches.

E. Use of Insecure or Outdated Components:

Dependence on outdated libraries, software patches, or unverified third-party components exposes IoT devices to known vulnerabilities, making them easier to exploit.

F. Insufficient Privacy Protection:

Weak handling of user credentials or personal data in IoT ecosystems can lead to unauthorized access, identity theft, or leakage of sensitive information.

G. Insecure Data Transfer and Storage:

Without proper encryption and access control, sensitive data becomes vulnerable during transmission, processing, or storage. Attackers can intercept, modify, or misuse such information.

H. Lack of Device Management:

In many cases, deployed devices lack support for monitoring, patching, secure updates, or controlled decommissioning. This absence of lifecycle management increases the risk of exploitation over time.

I. Insecure Default Settings:

Devices often come preconfigured with insecure defaults such as open network ports, weak firewalls, or unnecessary services. Users rarely modify these settings, leaving devices exposed.

J. Lack of Physical Hardening:

Many IoT devices provide little resistance to physical tampering. Attackers with direct access to hardware can extract sensitive data, bypass controls, or gain full system control.

III. THREATS IMPLEMENTATION AND ANALYSIS

Among the threats discussed in the previous section, this project focuses on Replay Attack and SQL Injection, as both are highly relevant in lightweight IoT environments and can be effectively demonstrated using simple hardware and open-source tools. These attacks represent common vulnerabilities in IoT communication and data storage, and addressing them provides significant security improvements with relatively low complexity. This section presents the exploitation techniques and mitigation strategies for both threats in the context of the proposed system, where a NodeMCU with a DHT11 sensor transmits data to a Raspberry Pi server via HTTP, and the Raspberry Pi stores the readings in a MariaDB database.

3.1) Replay Attack: A Threat in IoT Communication

In this project, sensor readings are sent from the NodeMCU to the Raspberry Pi using HTTP requests. Without additional protection, these requests can be intercepted and reused by attackers. For instance, if a packet containing valid temperature and humidity values is captured, it can later be resent multiple times using tools like CURL or Burp Suite, causing the database to be filled with duplicate or misleading data. This manipulation compromises data integrity and can distort monitoring or decision-making processes in applications such as smart agriculture or smart home automation.

Common Exploitation Techniques include:

1. Capturing unencrypted HTTP requests containing sensor values.
2. Resending valid packets multiple times to mislead the server (Replay).
3. Automating repeated transmissions using scripts or penetration testing tools.

Mitigation Strategies applied in the project:

1. Timestamping requests: Each HTTP request from Arduino carries a timestamp to prove freshness.
2. Shared secret key: A predefined key is added to each transmission and verified on the Raspberry Pi server.
3. Request validation: The server checks that the timestamp is recent and has not been used before.
4. Rejection of expired or duplicate data: Ensures that replayed packets are ignored, preserving database accuracy.

With these measures, the system can distinguish between genuine data from the Arduino and replayed requests from an attacker.

3.2) SQL Injection: A Threat in IoT Data Storage

Since the Raspberry Pi server stores incoming sensor readings in a MariaDB database, improper handling of inputs can expose the system to SQL Injection attacks. If raw user input is inserted directly into SQL queries without validation, attackers can manipulate the query logic by injecting malicious code. For example, instead of sending normal sensor data, an attacker could insert an SQL payload such as:

```
{"temperature":"25'); DROP TABLE sensors; --","humidity":"60"}
```

If the server processes this without protection, the database table storing sensor data could be deleted, resulting in loss of critical information.

Common Exploitation Techniques include:

1. Injecting SQL commands through JSON or query parameters.
2. Manipulating WHERE clauses to bypass authentication or retrieve unauthorized data.
3. Using payloads to alter, drop, or dump database tables.

Mitigation Strategies applied in the project:

1. Parameterized queries: Implementing prepared statements in the Flask server ensures inputs are treated strictly as data.
2. Input sanitization: Rejecting unexpected characters or patterns before passing data to the database.
3. Least-privilege access: Database users are configured with only the required permissions, preventing attackers from escalating privileges.
4. Error message handling: Suppressing detailed SQL errors to avoid leaking database structure information.

By applying these measures, the project ensures that malicious SQL inputs are stored as harmless strings and cannot alter or damage the database.

If you are using Word, use either the Microsoft Equation Editor or the MathType add-on (http://www.mathtype.com) for equations in your paper (Insert | Object | Create New | Microsoft Equation or MathType Equation). ―Float over text‖ should not be selected.

IV. CONCLUSION

The rapid growth of the Internet of Things (IoT) has transformed fields such as agriculture, healthcare, and smart infrastructure by enabling real-time data collection and decision-making. However, this expansion has also created new opportunities for attackers to exploit weaknesses in communication protocols and data storage systems. This project implemented a practical IoT architecture where an NodeMCU with a DHT11 sensor transmits data to a Raspberry Pi server via HTTP, with the collected information stored in a MariaDB database. Two critical security threats were examined: Replay Attack and SQL Injection**.** Both attacks were demonstrated using penetration testing tools in Kali Linux, showing how unprotected IoT systems can be easily compromised. Replay attacks exploited unencrypted communication by resending captured requests, while SQL injection targeted weak query handling to manipulate the database. Security mechanisms were then integrated into the system to mitigate these vulnerabilities. Replay attacks were addressed through the use of secure keys and timestamp validation, ensuring that repeated or outdated requests were rejected. SQL injection was prevented by applying parameterized queries and input sanitization**,** which ensured that malicious inputs were treated as plain data instead of executable commands. Testing after these improvements confirmed that the system could effectively withstand the same attack scenarios.

The results highlight the importance of embedding security at the design stage of IoT systems rather than treating it as an afterthought. Lightweight but robust solutions, such as request validation and safe database practices, can significantly enhance the resilience of IoT applications without imposing heavy resource requirements. Future work may focus on extending this system with stronger encryption mechanisms, intrusion detection techniques, and scalable architectures to support larger IoT deployments.

REFERENCE

[1] Bakry, Batrisyia B. Mohd, Alisa Rafiqah Bt Adenan, and Yusnani Bt Mohd Yussoff. "Security attack on IoT related devices using Raspberry Pi and Kali Linux." 2022 International Conference on Computer and Drone Applications (IConDA). IEEE, 2022.

[2] Bella, Giampaolo, et al. "Petiot: Penetration testing the internet of things." *Internet of Things* 22 (2023): 100707.

[3] Gunawan, Teddy Surya, et al. "On the review and setup of security audit using Kali Linux." *Indonesian Journal of Electrical Engineering and Computer Science* 11.1 (2018): 51-59.

[4] Lazzaro, Sara, et al. "Is your kettle smarter than a hacker? a scalable tool for assessing replay attack vulnerabilities on consumer iot devices." *2024 IEEE international conference on pervasive computing and communications (PerCom)*. IEEE, 2024.

[5] Raoof, Mohammed Mohammed, Norah Aldaghmi, and Haifa Aljuaid. "IoT Security in Healthcare: A Recent Trend and Predictive Study of SQL Injection Attacks." The Journal of Engineering 2025.1 (2025): e70103.

[6] Ajasa, Ade Dotun, Hassan Chizari, and Abu Alam. "Database Security and Performance: A Case of SQL Injection Attacks Using Docker-

Based Virtualisation and Its Effect on Performance." Future Internet 17.4 (2025): 156.

[7] Fadlil, A., I. Riadi, and M. A. Mu'Min. "Mitigation from SQL Injection Attacks on Web Server using Open Web Application Security Project Framework." International Journal of Engineering 37.4 (2024): 635-645.

[8] Rajagopal, Swetha, and V. V. Akhil. "Guarding Against Injection Attacks: A Filtering Solution for Web Application Security." 2025 6th International Conference on Control, Communication and Computing (ICCC). IEEE, 2025.