|  |
| --- |
| Close-up image showing the leaf-sides of two oversized books side-by-side on a bookshelf, with additional books in soft focus background |
| Development Individual Project  Secure CPS Communication Simulation Report |
| |  |  |  | | --- | --- | --- | | Milad Chowdhury | 6/5/25 | Secure Systems Architecture April 2025 | |

Contents

[1. Introduction and Research Objective 2](#_Toc200057320)

[2. System Design and Architecture 2](#_Toc200057321)

[3. Implementation Techniques and Security Architecture 3](#_Toc200057322)

[4. Vulnerability Analysis and CWE Mapping 3](#_Toc200057323)

[5. Testing Methodology and Results 4](#_Toc200057324)

[6. Critical Evaluation 5](#_Toc200057325)

[7. Functional Test Cases 6](#_Toc200057326)

[8. Conclusion 6](#_Toc200057327)

[References 7](#_Toc200057328)

## 1. Introduction and Research Objective

Cyber-Physical Systems (CPS) increasingly depend on distributed Internet of Things (IoT) architectures, where autonomous sensors communicate with centralised control logic. While this model offers greater scalability and responsiveness, it also exposes systems to a broader attack surface, particularly regarding **data confidentiality, integrity, and availability (CIA)**. As the National Institute of Standards and Technology (NIST, 2015) noted, communication pathways between distributed components represent one of the most critical vulnerabilities within industrial control systems.

This project adopts the **ABCDE framework** proposed by Boardman and Sauser (2006), with specific emphasis on:

* **Connectivity**: Establishing secure and reliable communication between independent sensor nodes and the controller.
* **Emergence**: Understanding how security properties and risks arise from the combined behaviour of modular system components.

**Research Hypothesis :**   
Client-side encryption, combined with simulated Quality of Service (QoS) constraints, improves delivery reliability and mitigates key security vulnerabilities (e.g. CWE-319, CWE-400, CWE-770, CWE-693) in distributed IoT systems.

This project evaluates whether local encryption and QoS-aware transport behaviours can strengthen CPS resilience by reducing message loss, preventing cleartext transmission, and hardening the system against malformed or excessive input.

All outputs were reviewed using log\_controller.txt and results.csv to validate consistency, traceability, and evidence-based testing.

## 2. System Design and Architecture

The system includes six Python-based simulated clients, each representing a typical IoT sensor:

* Temperature
* Motion
* Humidity
* Door
* Smoke
* Energy Monitor

These devices send encrypted messages to a central controller over TCP sockets. All encryption uses Fernet (AES) with a pre-shared key (shared\_key.key), which is identical across the system. A qos\_utils.py module introduces realistic transport challenges:

* Artificial latency: 0.1s to 0.5s delay
* 10% simulated packet drop probability

The controller decrypts messages, validates format, logs messages, and returns acknowledgements. Failed messages (e.g. dropped or malformed) are logged separately, simulating security logging practices in modern IDS tools.

## 3. Implementation Techniques and Security Architecture

Each IoT client script:

* Encrypts a structured message: [Sensor] value at TIMESTAMP
* Applies random delay and drop simulation via simulate\_qos()
* Sends messages via TCP to the controller

The controller:

* Decrypts the incoming payload using Fernet
* Validates basic format
* Logs successes and malformed inputs separately in log\_controller.txt

**Security Architecture Highlights:**

* **Confidentiality**: Ensured through Fernet-based symmetric encryption (CWE-319)
* **Availability simulation**: Drop and delay test system robustness (CWE-400)
* **Message structure enforcement**: Limits parsing vulnerabilities (CWE-693)
* **Allocation control**: Prevents overuse of controller memory or crash risk (CWE-770)

Encryption was a baseline mitigation because CPS devices often lack hardware-backed key exchange or full TLS stacks. The project assumes key distribution occurs out-of-band and would be upgraded in future iterations using TLS or X.509.

## 4. Vulnerability Analysis and CWE Mapping

The vulnerabilities addressed in this simulation were drawn directly from the AD Tree created in Part One. Each was mapped to a corresponding **Common Weakness Enumeration (CWE)** reference.

|  |  |  |
| --- | --- | --- |
| **CWE ID** | **Description** | **Mitigation** |
| CWE-319 | Cleartext Transmission | Encrypted all communication using Fernet |
| CWE-400 | Uncontrolled Resource Consumption | QoS delay and drop simulation; message pacing |
| CWE-770 | Allocation Without Limits | Input size limits and formatted message expectations |
| CWE-693 | Improper Protection Mechanism Handling | Logs all malformed inputs; safe fallback used |

These CWE threats are real-world and frequently exploited in IoT contexts (MITRE, 2025). They represent passive and active attack vectors — from eavesdropping (CWE-319) to malformed packet denial-of-service (CWE-770).

## 5. Testing Methodology and Results

A structured test routine involved all six client devices to validate the system's reliability and security controls. Each client transmitted encrypted messages to the controller under varying QoS conditions.

**Key testing objectives:**

* Measure **delivery success rates** under simulated packet drop
* Observe **latency variance** per message
* Verify controller behaviour under **malformed or missing input**

**Data Capture:**

* **results.csv** records: timestamp, client, value, latency, encryption status, drop status, and CWE mapping.
* **log\_controller.txt** contains only **successfully decrypted** messages logged at runtime.

**Results Snapshot (from results.csv):**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Timestamp** | **Client** | **Value** | **Latency (s)** | **Dropped** | **CWE** |
| 2025-06-05T21:50:23 | Energy | Consumption 1200W | 0.457 | False | CWE-319 |
| 2025-06-05T21:47:24 | Motion | Movement detected | 0.494 | False | CWE-770 |
| 2025-06-05T21:48:15 | Temperature | 22.5°C | 0.416 | False | CWE-319 |
| 2025-06-05T21:50:02 | Humidity | 45% RH | 0.480 | False | CWE-400 |
| 2025-06-05T21:50:40 | Door | Door opened | 0.349 | False | CWE-693 |

*Note: Drop and delay values were simulated via qos\_utils.py.*

**Log Evidence (log\_controller.txt)**

|  |  |  |
| --- | --- | --- |
| **Client** | **Message Status** | **Logged?** |
| Motion | Delivered & decrypted | Yes |
| Temperature | Delivered & decrypted | Yes |
| Humidity | Delivered & decrypted | Yes |
| Energy | Delivered & decrypted | Yes |
| Door | Delivered & decrypted | Yes |
| Smoke | Dropped (QoS simulation) | Not Logged |

**Observations:**

* Delivery success was >85% across multiple runs
* Dropped packets triggered no controller crashes and were reflected in CSV
* Malformed payloads were manually simulated by disabling encryption in select test runs. The controller logged the errors without crashing, confirming correct handling of CWE-693.

## 6. Critical Evaluation

The solution demonstrates strong alignment with secure-by-design principles:

**Strengths:**

* Modular architecture; easily extensible to real protocols like MQTT
* Encrypted transmission with structured validation
* CWE-to-code mapping supports both security and academic objectives
* Reusable for multiple use cases: home automation, industrial monitoring, edge computing

**Limitations:**

* **Static Key**: Pre-shared key lacks forward secrecy. Real-world deployments would require TLS or X.509-based exchange (NIST, 2015).
* **No Runtime Intrusion Detection**: Although logs exist, no IDS logic or alerting framework is implemented.
* **Limited Threat Coverage**: Only 4 CWEs were tested. Broader coverage would improve risk modelling.
* **No Resilience to Replay Attacks**: Message replay is possible without nonce or token mechanisms.
* **No Mutual Authentication**: Clients trust the controller by default. Zero trust architecture principles are not enforced.

**Real-World Relevance:**

The **Mirai botnet** exploited weak input validation and open ports in IoT devices (Kaspersky Labs, 2014). By simulating malformed messages and enforcing logging, this project shows how such risks can be mitigated even in constrained environments.

**OWASP Alignment:**

Mapped to the **OWASP Proactive Controls**:

* C1: Define Security Requirements
* C6: Implement Digital Identity
* C7: Enforce Access Control
* C9: Security Logging and Monitoring

## 7. Functional Test Cases

The following test cases were executed to validate system behaviour under simulated security and delivery conditions:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Outcome** | **Status** |
| TC-01 | The encrypted client message is sent and decrypted | Message logged and ACK returned by controller | Passed |
| TC-02 | Packet dropped by QoS delay simulation | No message logged; drop recorded in results.csv | Passed |
| TC-03 | Malformed message payload sent | Error logged; system does not crash | Passed |
| TC-04 | Message sent during artificial latency window | Increased latency recorded; successful delivery | Passed |
| TC-05 | Message missing timestamp or improperly structured | Logged as rejected message (format validation) | Passed |

These cases reflect CWE threat responses and are evidenced through log\_controller.txt and results.csv.

## 8. Conclusion

The project confirms that client-side encryption paired with simulated QoS characteristics improves message security and resilience for distributed CPS environments. While simplified, this simulation reflects real-world challenges and aligns with recognised security frameworks and vulnerability databases.

The methodology of threat enumeration via AD Tree, CWE mapping, and structured logging demonstrates how secure systems can be designed, tested, and analysed. While future work should address key rotation, protocol upgrades, and live detection systems, the current prototype validates the core hypothesis. It serves as a strong baseline for academic and professional security evaluations.

**Word Counts: 1268**

## References

Boardman, J. and Sauser, B. (2006). *System of Systems – the Meaning of*. IEEE Conference on System of Systems Engineering.

Jain, S.M. (2023). *Linux Containers and Virtualisation*. Apress.

Mishra, A.D. and Mustafa, K. (2022). 'A review on security requirements specification by formal methods'. *Concurrency and Computation*, 34(5), e6702.

MITRE Corporation (2025). *Common Weakness Enumeration (CWE) Database*. Available at: <https://cwe.mitre.org>

NIST (2015). *SP 800-82r2: Guide to ICS Security*. <https://nvlpubs.nist.gov>

OWASP Foundation (2023). *OWASP Proactive Controls*. <https://owasp.org>

Kaspersky Labs (2014). *A Brief History of Computer Viruses and Botnets*. <https://www.kaspersky.com>