DESIGNING A SECURE INDUSTRIAL CONTROL SYSTEM

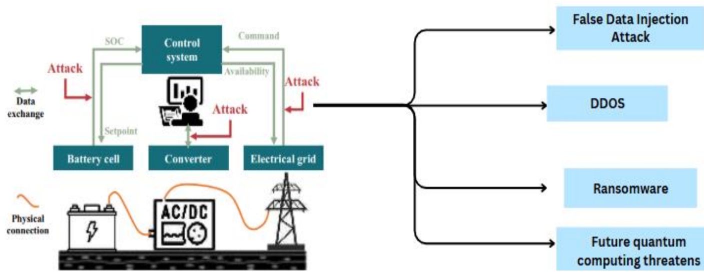
1. Introduction:

This project demonstrates the implementation of a comprehensive cybersecurity framework for **Industrial Control Systems (ICS)** using a simulated **traffic management system** as a testbed, using ESP32 microcontrollers. The objective is to showcase how advanced security mechanisms can be effectively deployed on **resource-constrained IoT devices** in critical infrastructure environments.
Intelligent Traffic Detection Module: Dynamically adjusts traffic light timing based on real-time vehicle density, simulating the behaviour of an ICS in a smart city environment.

- **Cybersecurity Architecture**: Features a multi-layered security approach, including the deployment of a **honeypot system** to detect and redirect malicious traffic, minimizing the risk to operational components.
- **Secure Communication Protocols**: Implements **TLS (Transport Layer Security)** augmented with **Post-Quantum Cryptography (PQC)** to ensure encrypted and future-proof data transmission between ESP32 nodes.
- **Intrusion Detection System (IDS)**: Integrates **Snort**, combined with a **Hybrid Machine learning model** (leveraging both classical and quantum algorithms), for real-time threat detection and anomaly analysis.

By leveraging the traffic control system as a representative ICS model, this project effectively illustrates the feasibility of deploying advanced cybersecurity strategies in **BESS (Battery Energy Storage Systems)**.

2. Motivation

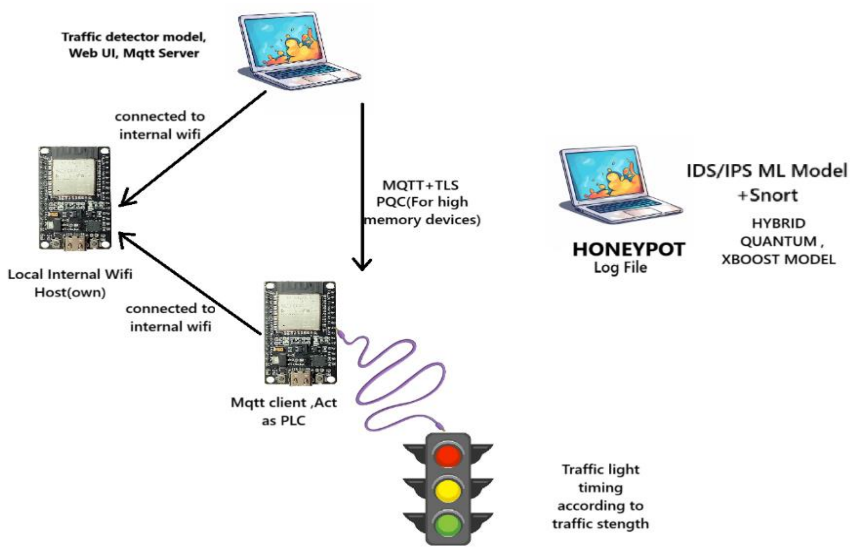


The rise of **BESS** in critical infrastructure brings serious security challenges, **from current cyber threats to future quantum attacks**. Traditional systems lack adaptability and strong encryption. We built a proof-of-concept showing that even low-power devices like the ESP32 can enable secure, intelligent BESS operations. Tested on a traffic management system, our solution addresses today's risks while being future-ready for the **quantum era.**

3. Literature Review

Prior studies support elements of our approach: Knowles et al. (2023) highlighted weak layered security in industrial systems; Vasilomanolakis et al. (2023) found industrial honeypots detect significantly more OT-specific attacks; Chen and Rodriguez (2024) applied honeypots to IoT but not BESS. The NIST PQC process (2024) standardized quantum-safe algorithms, with Seo et al. (2023) implementing them on ESP32s. Kumar et al. (2023) proposed effective hybrid IDS models, though resource-intensive for BESS. Zhang et al. (2024) identified BESS-specific threats, and Ramirez et al. (2023) documented real-world BESS incidents. Our work uniquely integrates adaptive management, honeypots, hybrid IDS/IPS, and post-quantum cryptography into a unified ESP32-based system, initially validated in traffic systems for future BESS deployment.

4. Methodology & Implementation Idea



Our system architecture consists of the following components:

1. System Monitoring and Control:
    - ESP32 Device #1 functions as a PLC (Programmable Logic Controller) Control Traffic light.

  Traffic Detector Model.

YOLOv8 (Ultralytics) — for object detection
OpenCV — for video processing and visualization
Paho MQTT — for sending data to IoT devices (like ESP32)
PyTorch — for deep learning model deployment
TLS/SSL — for secure MQTT communication

Web UI for Traffic Light Control System.

Flask – for backend web server
Flask-MQTT – to subscribe to traffic topics securely
MQTT (Mosquitto/ESP32) – for IoT communication
Chart.js – for dynamic data visualization
Bootstrap – for responsive, clean UI
TLS/SSL – for secure MQTT integration

2. Security Infrastructure:

   - Honeypot System:
     Redirect the unauthorized or hackers to fake virtual system.
     Web UI of this looks same like Traffic light system.
     Demonstrate **fake data on Web UI for trap the Hacker**.
     Continuously Save the **log file of Network**.
   - **ESP32 Device #2 hosts an internal Wi-Fi network.**

3. Intrusion Detection/Prevention System (IDS/IPS):
   - **Snort/Suricata** For real Time packets monitoring and save the log file.

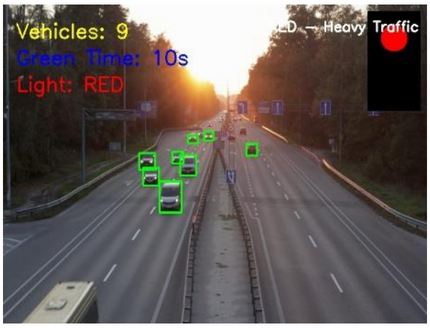
   ML models - Two models-For real Time Monitoring and Blocking Unauthorized access/attacks.
   1.**Classical model-For Classical and less Dimension Data.**
   2.**Hybrid (Classical + Quantum) model-For Future High Dimensional Data of Hackers.**

4. Secure Communication:
   - TLS (Transport Layer Security) implementation -Generating Certificates using Algorithm- **RSA (Rivest-Shamir-Adleman), a widely used asymmetric cryptographic algorithm.**
   - PQC (Post-Quantum Cryptography) implementation- Generating Certificates using Algorithm-p256_mldsa44(Hybrid **Classical Elliptic Curve Digital Signature Algorithm (ECDSA)+ Postquantum digital signature algorithm**
   - Certificate-based authentication between ESP32 devices and Data publisher.

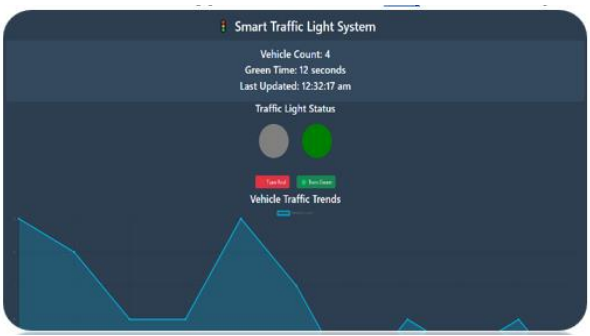
5.For monitoring security →Wireshark, Snort/Suricata for Packet monitoring.

5. Results

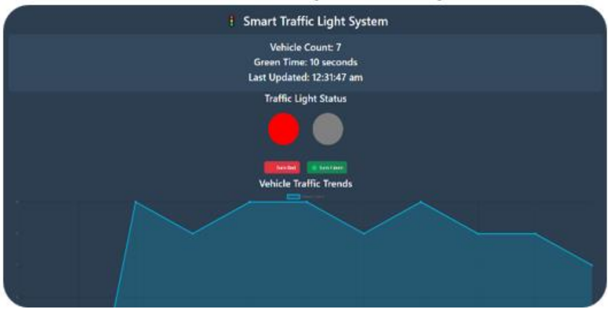
Traffic detector

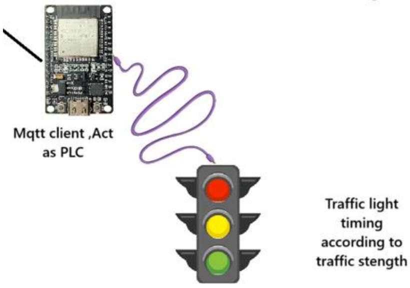


Honeypot Fake Web UI (Fake Data)

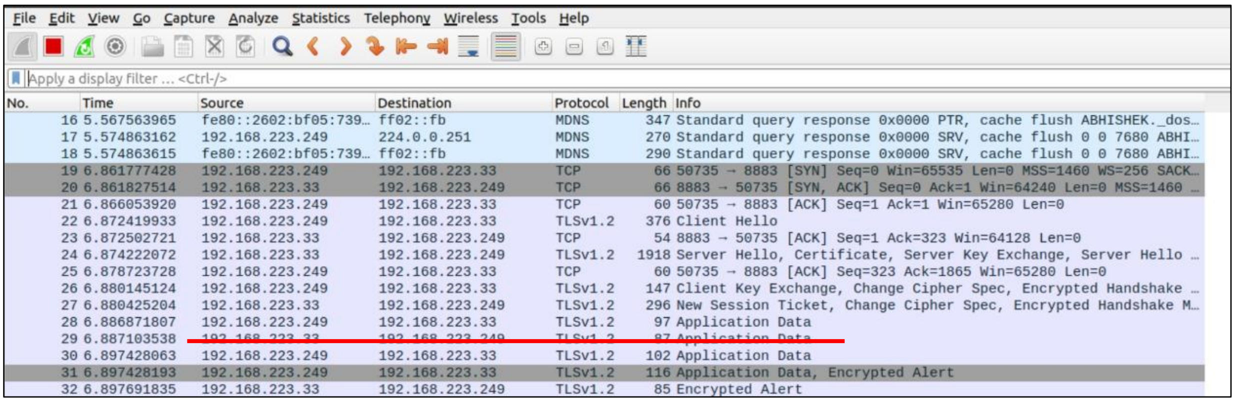


Real web UI (Real Data)



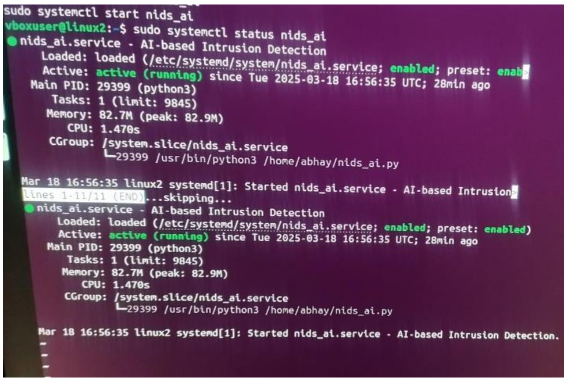


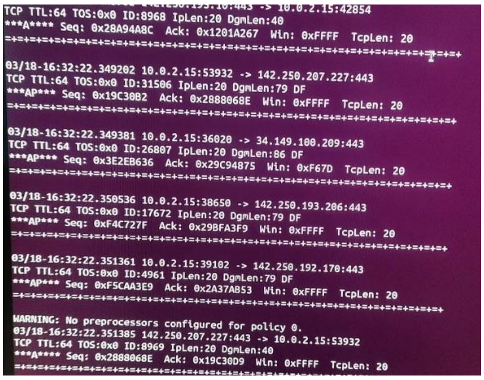
Wire Shark output - TLS 1.2 Encrypted Data



No.	Time	Source	Destination	Protocol	Length	Info
16	5.567563965	fe80::2602:bf05:739…	ff02::fb	MDNS	347	Standard query response 0x0000 PTR, cache flush ABHISHEK._dos…
17	5.574863162	192.168.223.249	224.0.0.251	MDNS	270	Standard query response 0x0000 SRV, cache flush 0 0 7680 ABHI…
18	5.574863615	fe80::2602:bf05:739…	ff02::fb	MDNS	290	Standard query response 0x0000 SRV, cache flush 0 0 7680 ABHI…
19	6.861777428	192.168.223.249	192.168.223.33	TCP	66	50735 → 8883 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK…
20	6.861827514	192.168.223.33	192.168.223.249	TCP	66	8883 → 50735 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 …
21	6.866053920	192.168.223.249	192.168.223.33	TCP	60	50735 → 8883 [ACK] Seq=1 Ack=1 Win=65280 Len=0
22	6.872419933	192.168.223.249	192.168.223.33	TLSv1.2	376	Client Hello
23	6.872502721	192.168.223.33	192.168.223.249	TCP	54	8883 → 50735 [ACK] Seq=1 Ack=323 Win=64128 Len=0
24	6.874222072	192.168.223.33	192.168.223.249	TLSv1.2	1918	Server Hello, Certificate, Server Key Exchange, Server Hello …
25	6.878723728	192.168.223.249	192.168.223.33	TCP	60	50735 → 8883 [ACK] Seq=323 Ack=1865 Win=65280 Len=0
26	6.880145124	192.168.223.249	192.168.223.33	TLSv1.2	147	Client Key Exchange, Change Cipher Spec, Encrypted Handshake …
27	6.880425204	192.168.223.33	192.168.223.249	TLSv1.2	296	New Session Ticket, Change Cipher Spec, Encrypted Handshake M…
28	6.886871807	192.168.223.249	192.168.223.33	TLSv1.2	97	Application Data
29	6.887103538	~~192.168.223.33~~	~~192.168.223.249~~	~~TLSv1.2~~	~~87~~	~~Application Data~~
30	6.897428063	192.168.223.249	192.168.223.33	TLSv1.2	102	Application Data
31	6.897428193	192.168.223.249	192.168.223.33	TLSv1.2	116	Application Data, Encrypted Alert
32	6.897691835	192.168.223.33	192.168.223.249	TLSv1.2	85	Encrypted Alert

Snort is running and capturing the network packets.





LOG FILE HONEYPOT BELOW.

```
2025-04-08 22:17:07,084 - 192.168.223.2 - - [08/Apr/2025 22:17:07] "GET /api HTTP/1.1" 200 -
2025-04-08 22:17:09,337 - [API REQUEST] IP: 192.168.223.2, Data: b'', Headers: {'Host': '192.168.223.249:8080', 'Connection':
'keep-alive', 'User-Agent': 'Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Mobile
Safari/537.36', 'Accept': 'application/json, text/javascript, */*; q=0.01', 'X-Requested-With': 'XMLHttpRequest', 'Referer':
'http://192.168.223.249:8080/', 'Accept-Encoding': 'gzip, deflate', 'Accept-Language': 'en-GB,en-US;q=0.9,en;q=0.8,hi;q=0.7'},
Location: {'ip': '192.168.223.2', 'bogon': True}
2025-04-08 22:17:09,337 - 192.168.223.2 - - [08/Apr/2025 22:17:09] "GET /api HTTP/1.1" 200 -
2025-04-08 22:17:09,653 - 192.168.223.2 - - [08/Apr/2025 22:17:09] "GET /toggle-light HTTP/1.1" 200 -
```

Formatted log file

```
[2025-04-08 22:11:53.418723] HACKER DETECTED: IP=192.168.223.2, UA=mozilla/5.0 (linux; android 10; k) applewebkit/537.36 (khtml,
like gecko) chrome/135.0.0.0 mobile safari/537.36
[2025-04-08 22:12:19.492729] HACKER DETECTED: IP=192.168.223.2, UA=mozilla/5.0 (linux; android 10; k) applewebkit/537.36 (khtml,
like gecko) chrome/135.0.0.0 mobile safari/537.36
```

Log file format store like this below--

```
2025-04-08 22:17:07,083 - [API REQUEST] IP: 192.168.223.2,
Data: b'',
Headers: {
 'Host': '192.168.223.249:8080',
 'Connection': 'keep-alive',
 'User-Agent': 'Mozilla/5.0 (Linux; Android 10; K)...',
 'Accept': 'application/json, text/javascript, */*; q=0.01',
 'X-Requested-With': 'XMLHttpRequest',
 'Referer': 'http://192.168.223.249:8080/',
 'Accept-Encoding': 'gzip, deflate',
 'Accept-Language': 'en-GB,en-US;q=0.9,en;q=0.8,hi;q=0.7'
},
Location: {
 'ip': '192.168.223.2',
 'bogon': True
}
```

Hacker or Unauthorized IP gets blocked

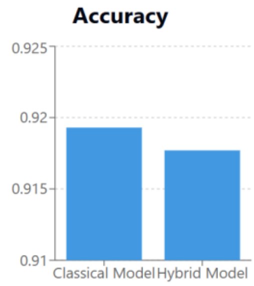
```
if prediction ==1
def block_ip(ip):
    print(f"Blocking IP: {ip}")
    os.system(f"sudo iptables -A INPUT -s {ip} -j DROP")
```

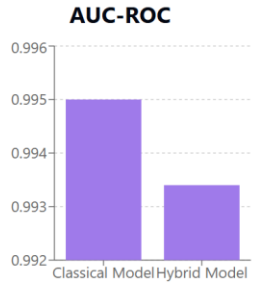
IDS/IPS Models Accuracy:

Core Pipeline

Data Cleaning → Quantum Circuit (8q) → [Quantum Features + Classical Features] → PCA (95% var) → SMOTE Balancing → Hybrid Ensemble (XGBoost + RF + LightGBM) → Evaluation

Metric	Classical Model	Hybrid Model	Difference
Accuracy	0.9193	0.9177	-0.16%
AUC.ROC	0.9950	0.9934	-0.12%



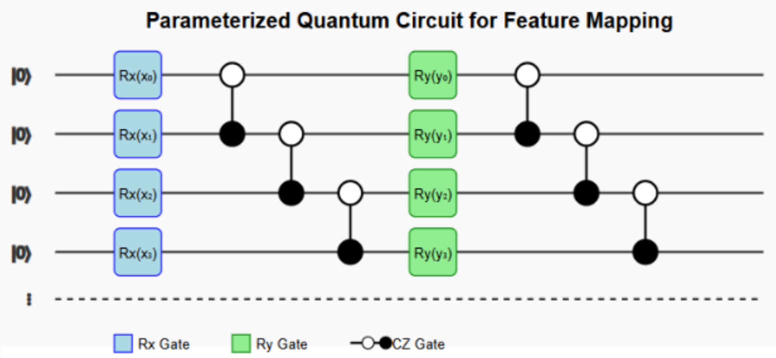


Hybrid Model Better for Future High Dimensional Data

Circuit 1(Basic)

Circuit Structure:

- First Layer:
  - Apply **Rx gates** to each qubit.
  - Rotation angles = normalized input features $x_i \in [-\pi, \pi]$.
- First Entanglement Layer:
  - Apply **CZ gates** between adjacent qubits.
- Second Layer:
  - Apply **Ry gates** to each qubit.
  - Rotation angles = $y_i = 0.8 \times x_i$.
- Second Entanglement Layer:
  - Apply another round of **CZ gates** between adjacent qubits.



Data Encoding:

- Normalize input features to the range **[-π, π]**.
- Use **scaled version (0.8×)** for the second rotation layer.

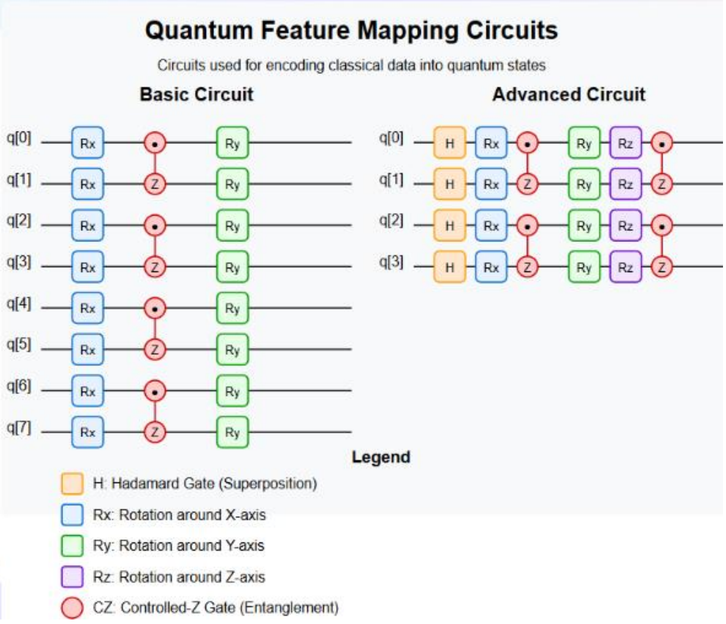
Circuit 2(More better)

Basic Circuit (8 Qubits):

- **Layer 1:** Rx gates (X-rotations) using normalized data.
- **Entanglement:** CZ gates between adjacent qubits.
- **Layer 2:** Ry gates (Y-rotations) with angles = 0.8 × input.

Advanced Circuit (4 Qubits):

- **Layer 1:** Hadamard gates (create superposition).
- **Layer 2:** Rx gates (X-rotations).
- **Layer 3:** CZ gates (entanglement).
- **Layer 4:** Ry gates (Y-rotations).
- **Layer 5:** Rz gates (Z-rotations), angles = 0.6 × input.
- **Layer 6:** Second CZ entanglement with new connectivity.



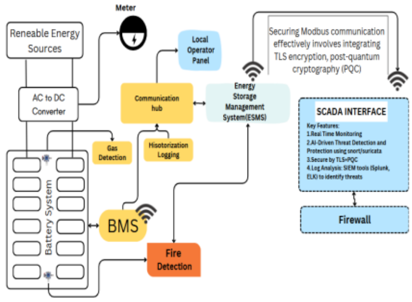
6. Conclusions / Achievements & Future Direction

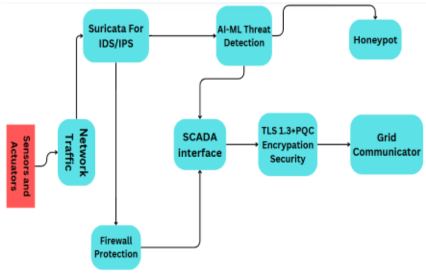
Achievements:

- Successfully demonstrated comprehensive security on resource-constrained ESP32 devices
- Integrated adaptive functionality with multi-layered security in a cohesive system
- Implemented PQC solutions to address quantum computing threats
- Hybrid(Classical+ Quantum) ML model for Future High Dimensional Data.

Future Direction:

1. **BESS Implementation:**
   Adapt security framework for Battery Energy Storage Systems
   Develop BESS-specific IDS/IPS rules and honeypot interfaces
   Integrate with grid control systems
   Model BESS-specific threat vectors
2. **Performance Optimization:**
   Reduce IDS/IPS false positives through improved ML training
   Optimize PQC implementation for efficiency
   Implement hardware acceleration for cryptographic operations
3. **Threat Detection Enhancement:**
   Deploy distributed sensing for coordinated threat monitoring
   Implement collaborative defence across ESP32 nodes
   Develop adaptive learning for energy system threat patterns
4. **Scalability:**
   Expand deployment to larger BESS networks
   Develop unified security monitoring interface
   Standardize deployment protocols for energy infrastructure





This project demonstrates that modern IoT devices can effectively implement sophisticated security alongside primary functions, potentially transforming protection for energy infrastructure against evolving cyber threats.