Introduction

This project demonstrates a robust cybersecurity framework for Industrial Control Systems (ICS) using a simulated smart traffic management system powered by ESP32 microcontrollers. Key features include:

- **Intelligent Traffic Detection:** Dynamically adjusts traffic lights based on vehicle density, mimicking real ICS behavior.
- **Multi-layered Cybersecurity:** Includes a honeypot system to divert threats and protect critical components.
- **Secure Communication:** Uses TLS with Post-Quantum Cryptography for encrypted, future-proof data exchange.
- **Advanced IDS:** Combines Snort with a hybrid machine learning model (classical + quantum) for real-time threat detection.

The project is aimed at enhancing cybersecurity for Battery Energy Storage Systems (BESS) by demonstrating how advanced security measures can be effectively implemented on low-power IoT devices.

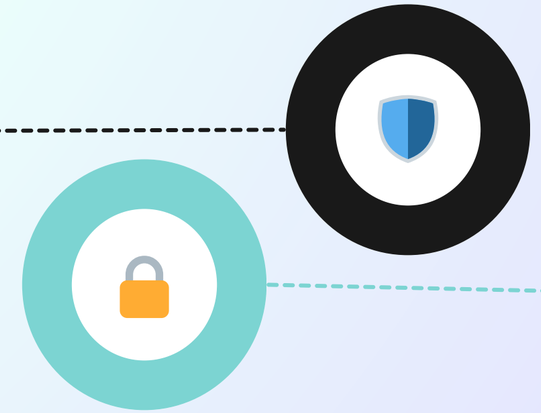
Motivation

From Current Threats to Quantum Resilience

The rise of Battery Energy Storage Systems (BESS) in critical infrastructure brings serious security challenges, from current cyber threats to future quantum attacks. Our innovative approach provides robust protection while ensuring adaptability for tomorrow's threats.

Protection:

Comprehensive solution addresses today's cyber risks while being future-ready for quantum computing threats

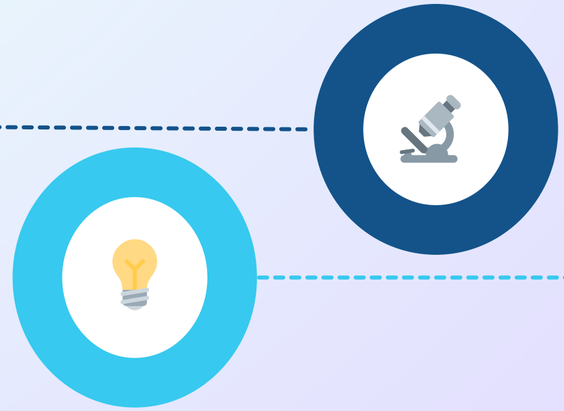


Security Gap:

Traditional systems lack adaptability and strong encryption methods needed for critical infrastructure

Validation:

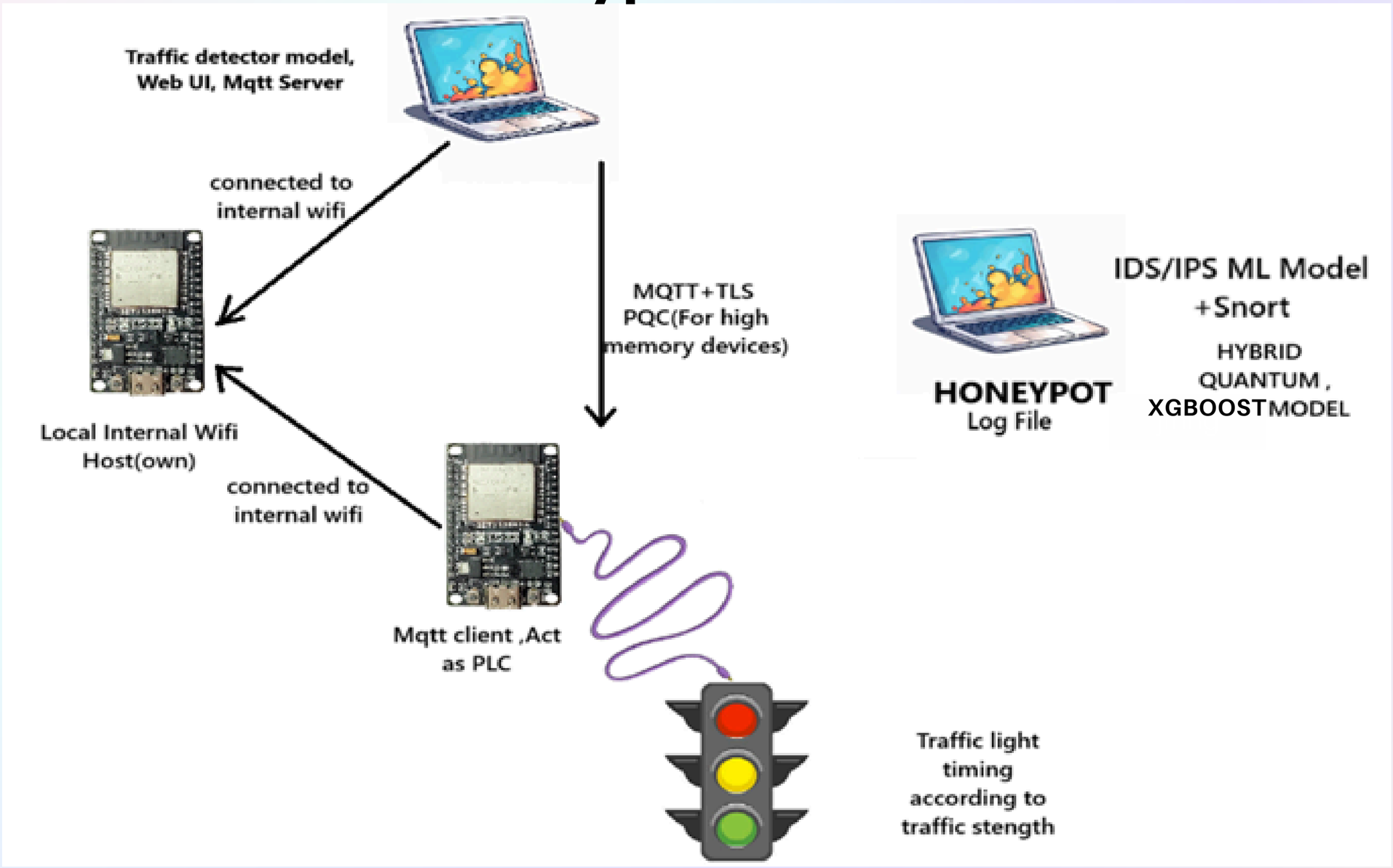
Successfully tested on a traffic management system with real-world constraints and challenges



Innovation:

Our proof-of-concept demonstrates that even low-power devices like ESP32 can enable secure, intelligent BESS operations

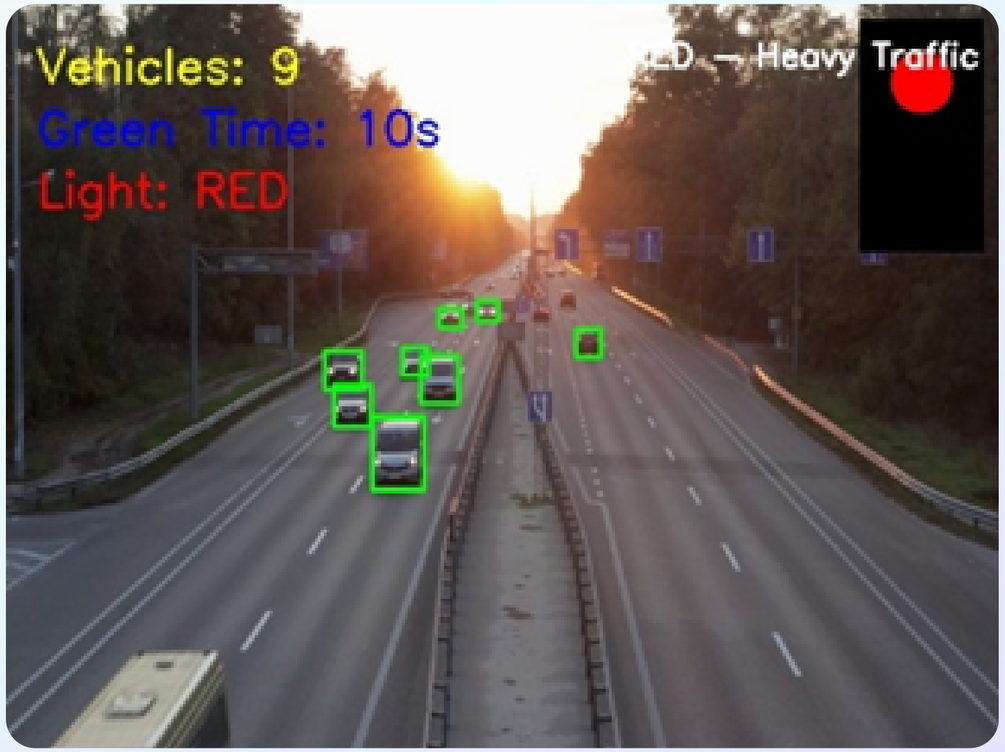
Secure IoT Traffic Control with PQC, TLS, ML-IDS, and Honeypot Defense

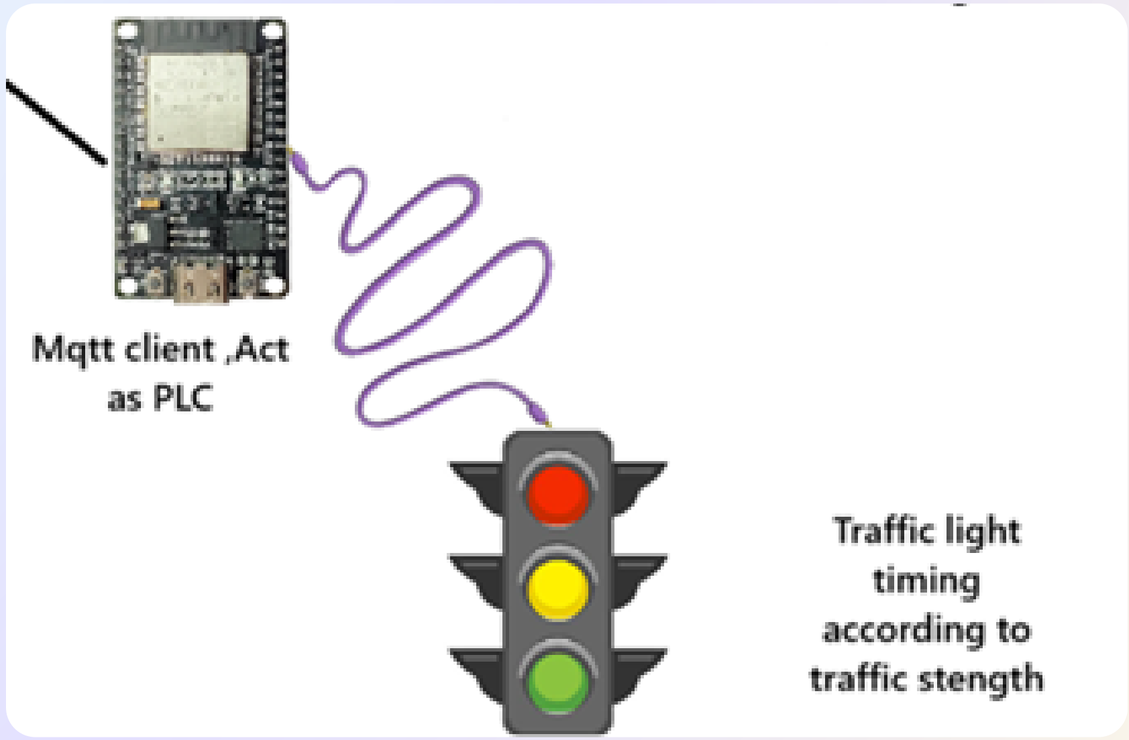


1.System Monitoring and Control:

- 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

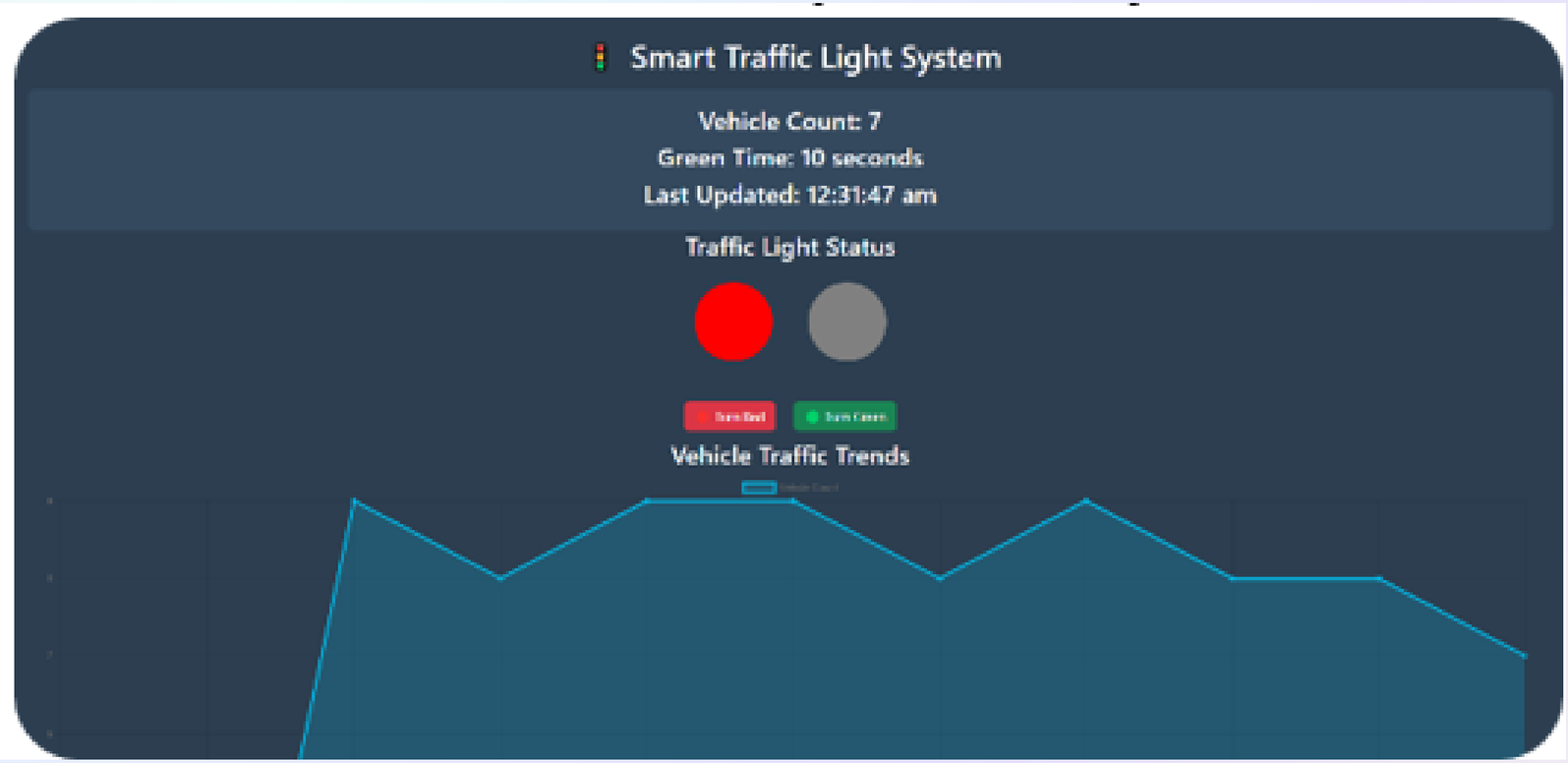
Flask-MQTT – to subscribe to traffic topics securely

MQTT (Mosquitto/ESP32) – for IoT communication

Chart.js – for dynamic data visualization

Bootstrap – for responsive, clean UI

TLS/SSL – for secure MQTT integration



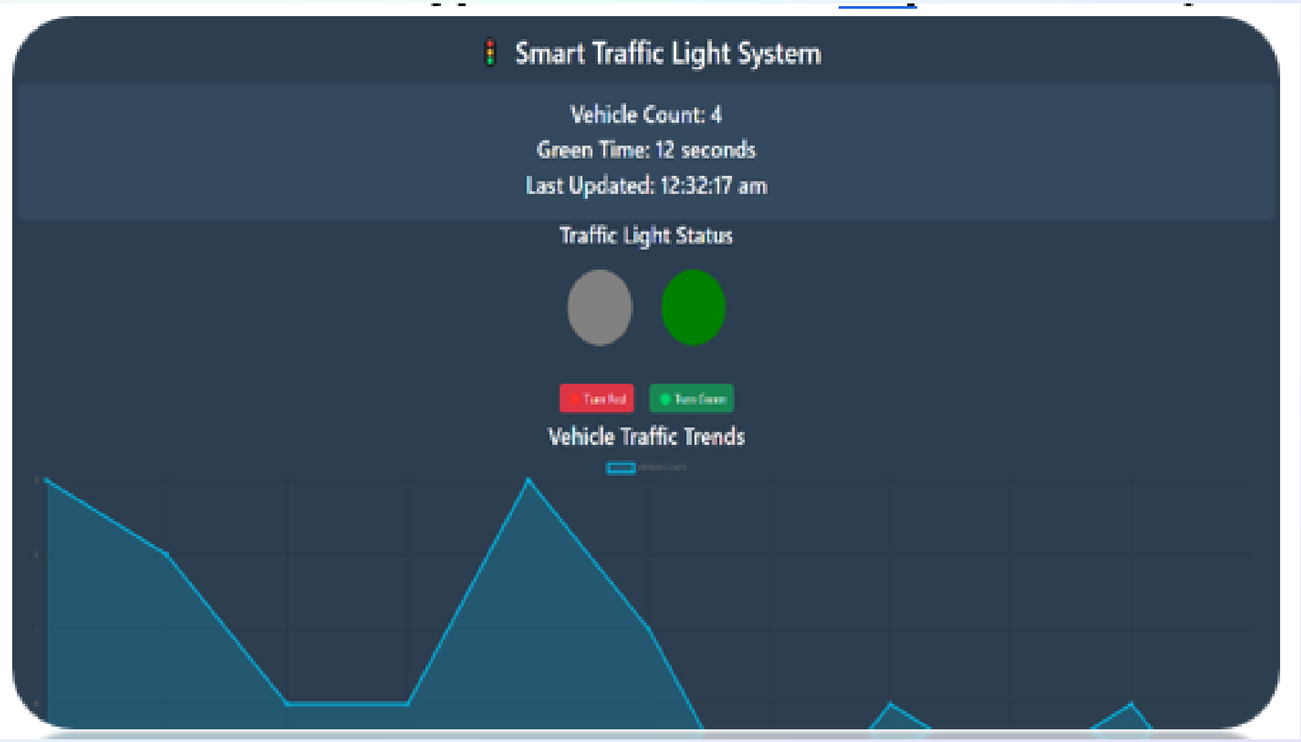
2.Security Infrastructure:

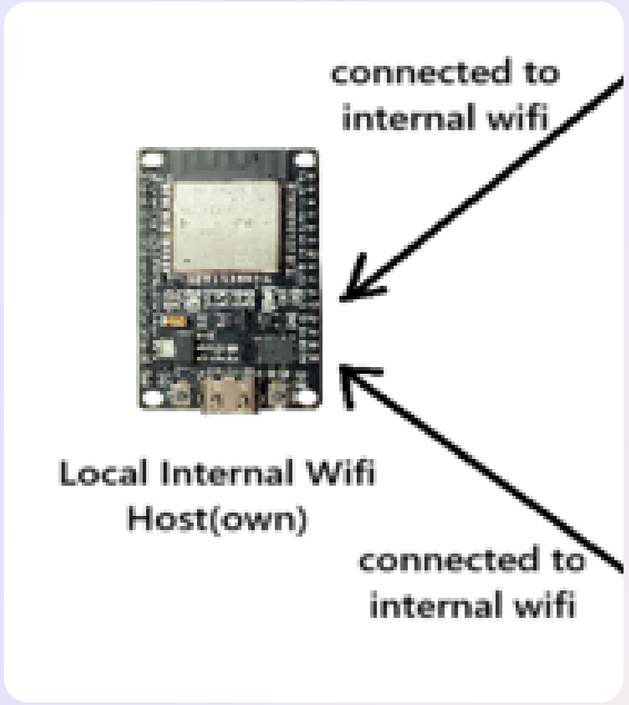
Honeypot System

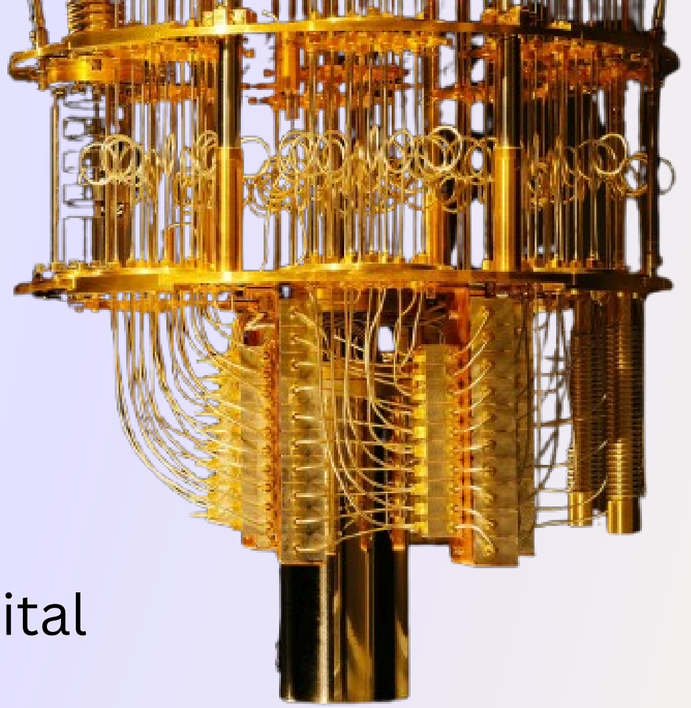
- Fake Virtual System: Redirects hackers to a decoy environment
- Web UI: Mimics a traffic light system
- Trap: Displays fake data to lure hackers
- Continuous Logging: Captures network activity

ESP32 Device #2

- Internal Wi-Fi Network: Hosts a secure, isolated network

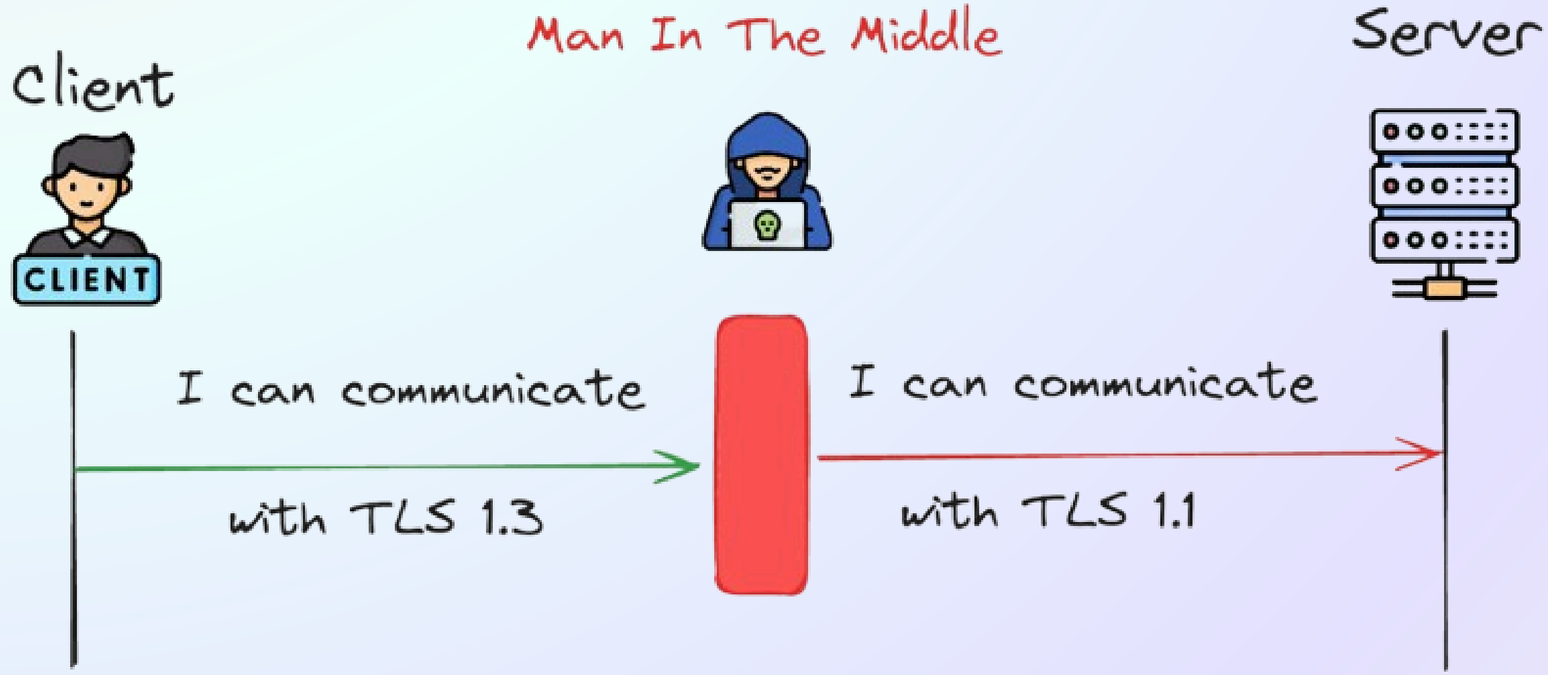






3.Secure Communication:

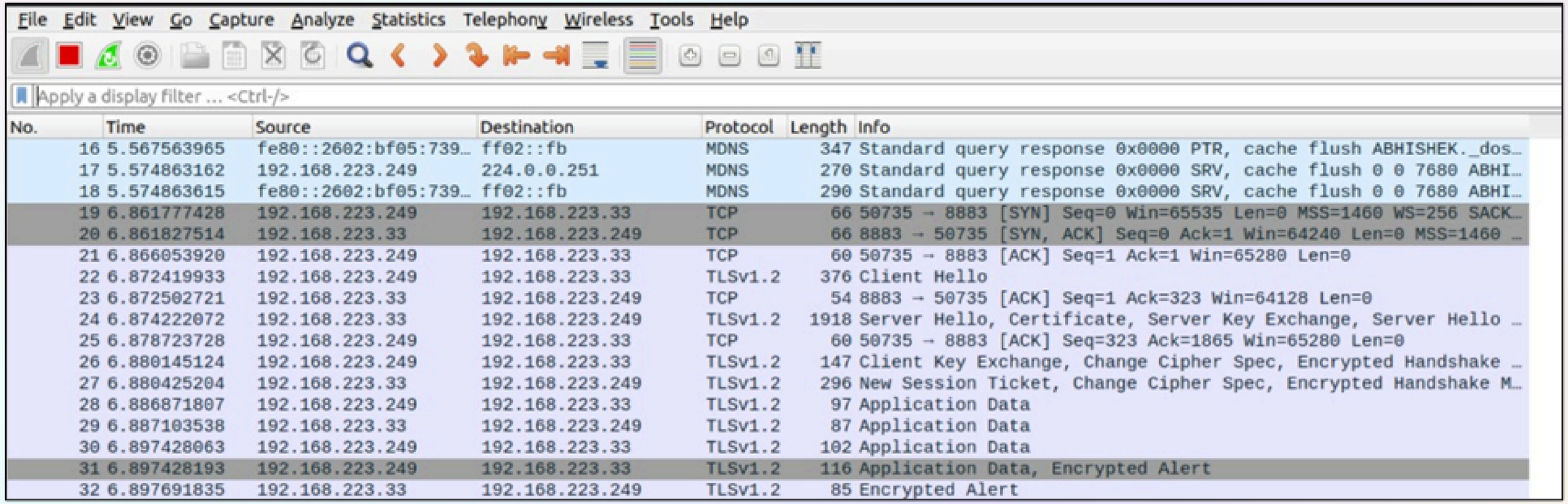
- TLS (Transport Layer Security) Implementation
  - Certificate generation using RSA (Rivest-Shamir-Adleman)
  - Widely used asymmetric cryptographic algorithm
- Post-Quantum Cryptography (PQC) Implementation
  - Certificate generation using p256_mldsa44
  - Hybrid approach: Classical ECDSA + MLDSA Post-Quantum digital signature algorithm



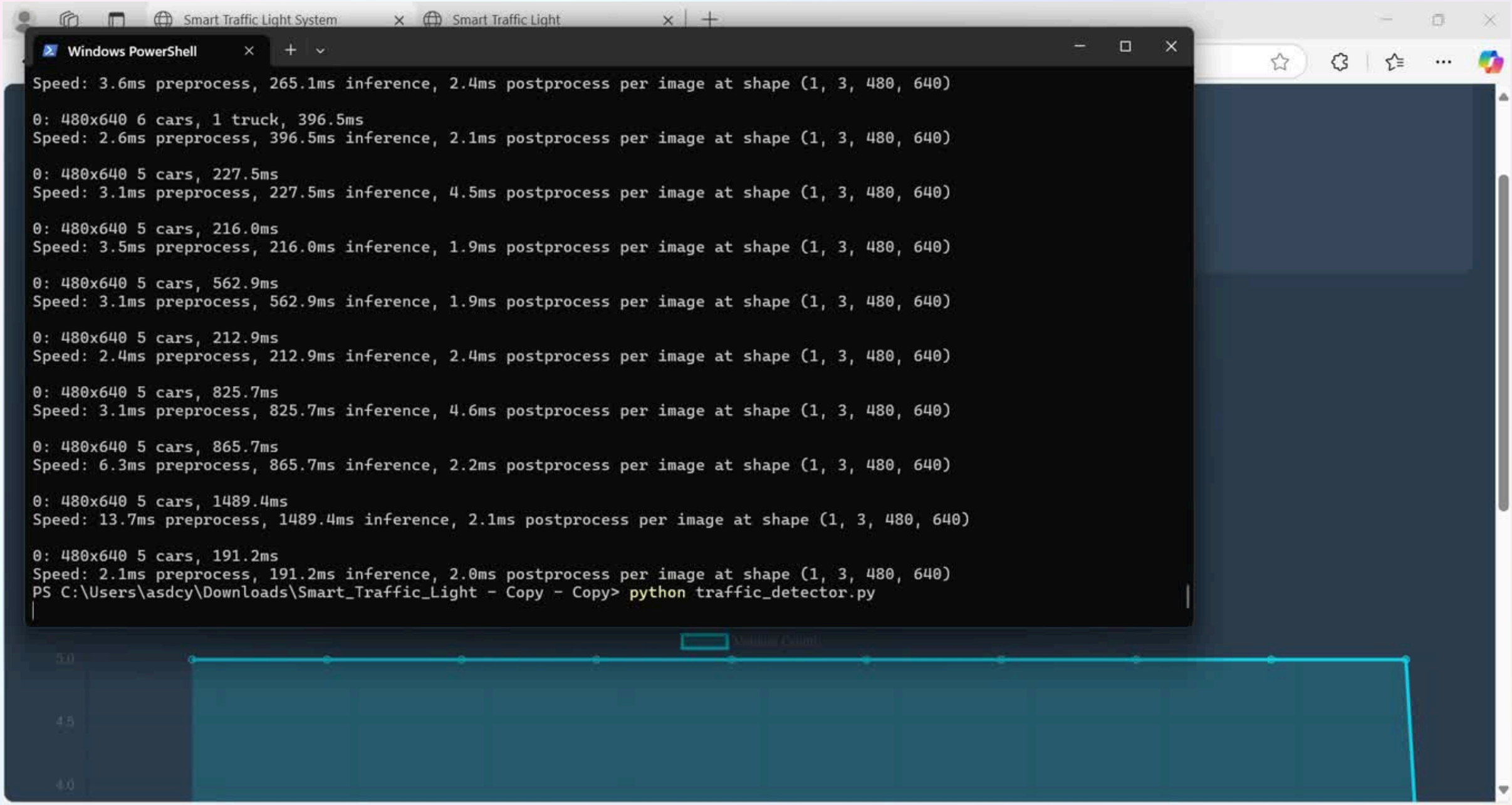
- Certificate-Based AuthenticationBetween ESP32 devices and Data Publisher

4.For monitoring security Wireshark, Snort/Suricata for Packet monitoring.

Wireshark



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


Smart Traffic Light System
Smart Traffic Light
Windows PowerShell
Speed: 3.6ms preprocess, 265.1ms inference, 2.4ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 6 cars, 1 truck, 396.5ms
Speed: 2.6ms preprocess, 396.5ms inference, 2.1ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 5 cars, 227.5ms
Speed: 3.1ms preprocess, 227.5ms inference, 4.5ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 5 cars, 216.0ms
Speed: 3.5ms preprocess, 216.0ms inference, 1.9ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 5 cars, 562.9ms
Speed: 3.1ms preprocess, 562.9ms inference, 1.9ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 5 cars, 212.9ms
Speed: 2.4ms preprocess, 212.9ms inference, 2.4ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 5 cars, 825.7ms
Speed: 3.1ms preprocess, 825.7ms inference, 4.6ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 5 cars, 865.7ms
Speed: 6.3ms preprocess, 865.7ms inference, 2.2ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 5 cars, 1489.4ms
Speed: 13.7ms preprocess, 1489.4ms inference, 2.1ms postprocess per image at shape (1, 3, 480, 640)
0: 480x640 5 cars, 191.2ms
Speed: 2.1ms preprocess, 191.2ms inference, 2.0ms postprocess per image at shape (1, 3, 480, 640)
PS C:\Users\asdcy\Downloads\Smart_Traffic_Light - Copy - Copy> python traffic_detector.py

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")
```

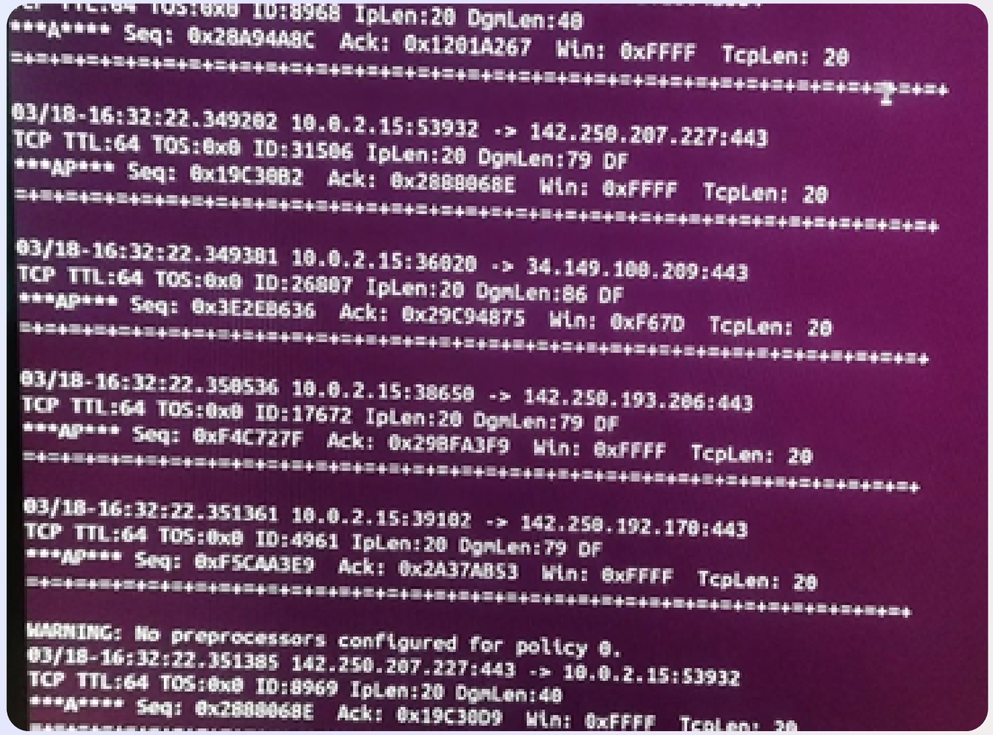
5.Intrusion Detection/Prevention System (IDS/IPS):

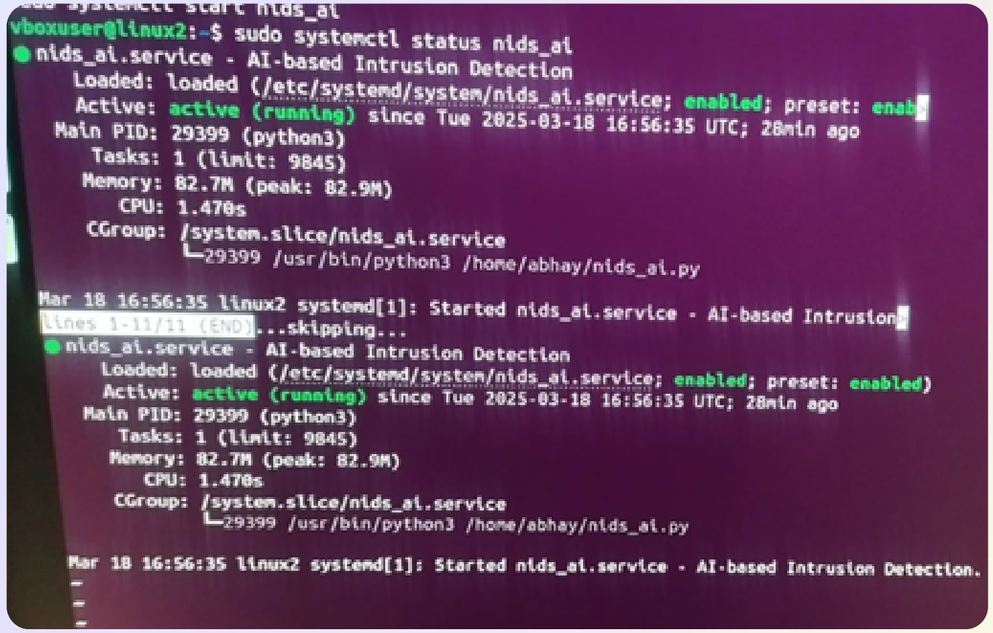
- **Snort/Suricata:**
  Used for real-time packet monitoring and logging.

- **ML Models:**
  Two models designed for real-time monitoring and blocking unauthorized access/attacks.

1. **Classical Model:**
- For classical, low-dimensional data.
- Suitable for current, less complex data sets.

1. **Hybrid Model (Classical + Quantum):**
- For future, high-dimensional data.
- Combines classical ML with quantum techniques to handle advanced, evolving threats.



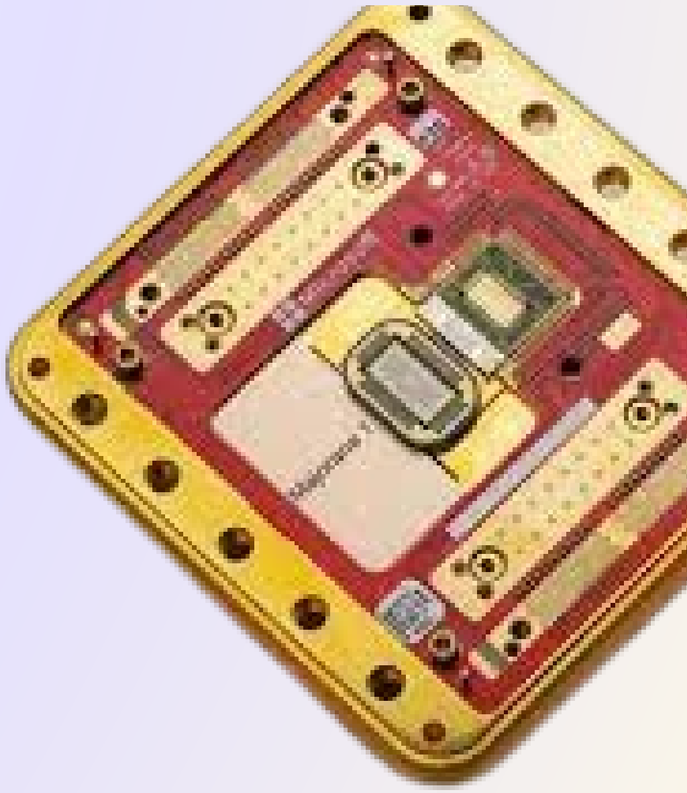


IDS/IPS Models :

First Model For Practical implementation Purpose

Core Pipeline:

Data Loading → Optimized Preprocessing *(Outlier Removal, Scaling)* →
Quantum Feature Mapping *(8-qubit circuit)* →
Feature Engineering *(Quantum + Classical)* →
PCA *(95% Variance Retained)* →
SMOTE Balancing →
Hybrid Ensemble Model *(XGBoost + RandomForest + LightGBM*) →
Evaluation



Second Model For Research Purpose

Core Pipeline

Data Loading → Stratified Splitting → Preprocessing *(Outlier Handling, Scaling)* →
Custom Quantum Feature Mapping →
Feature Extraction → Feature Combination/Selection →
SMOTE Balancing → Model Training *(4 Approaches)* →
Hyperparameter Tuning → Model Comparison → Evaluation

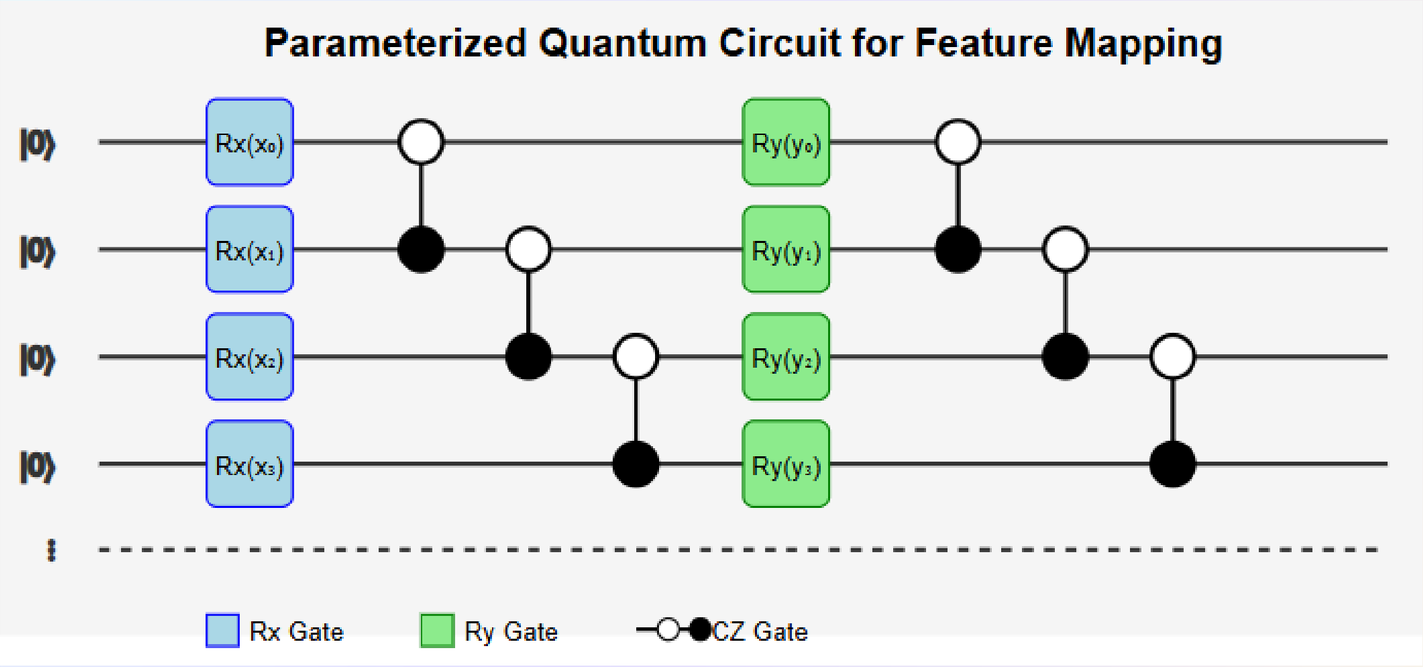
First Model Quantum Circuit

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.



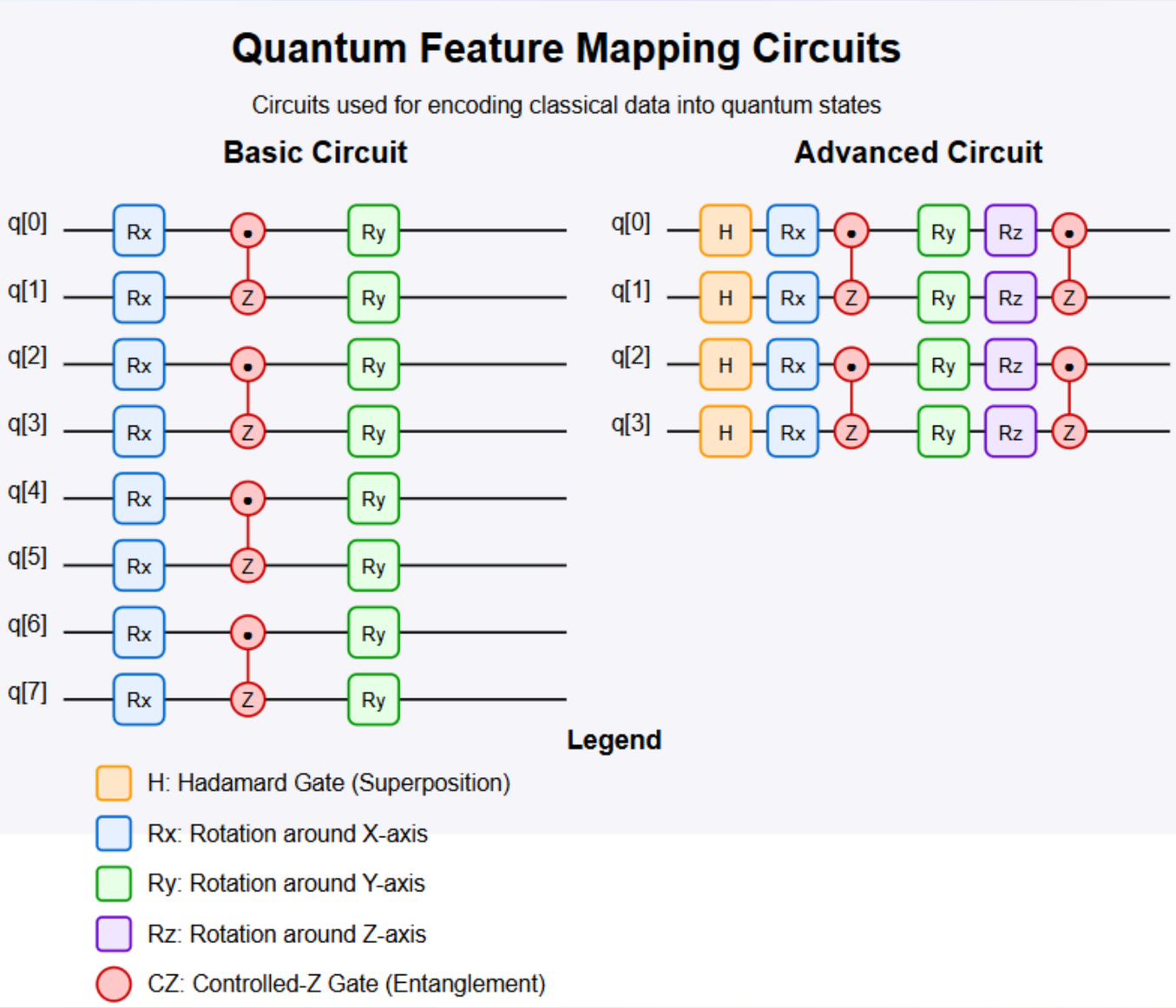
Second Model Quantum Circuit

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.

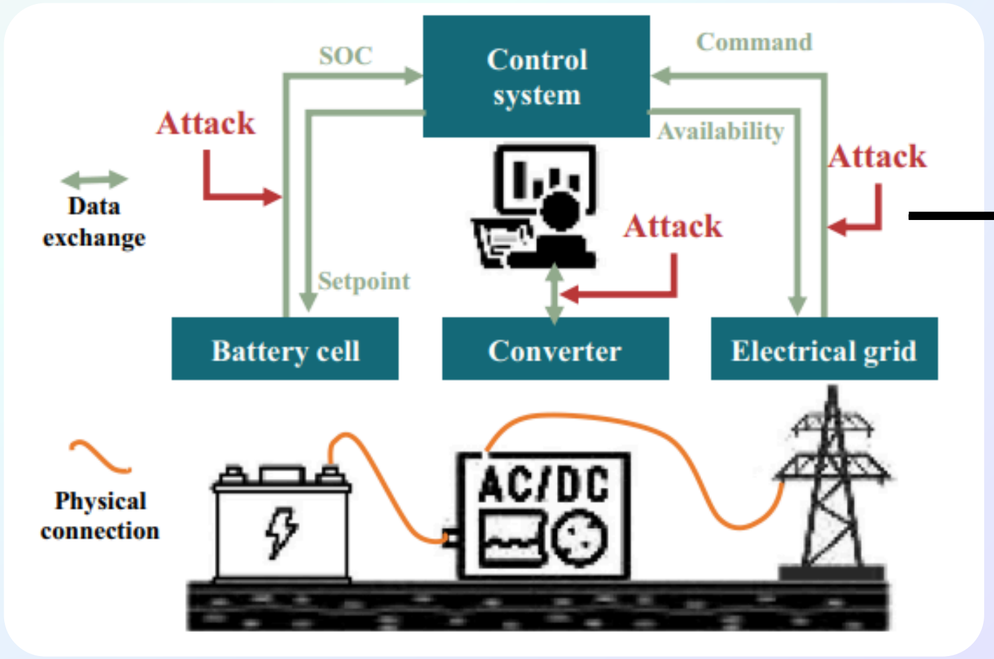
Advanced Circuit (4 Qubits):

- **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.

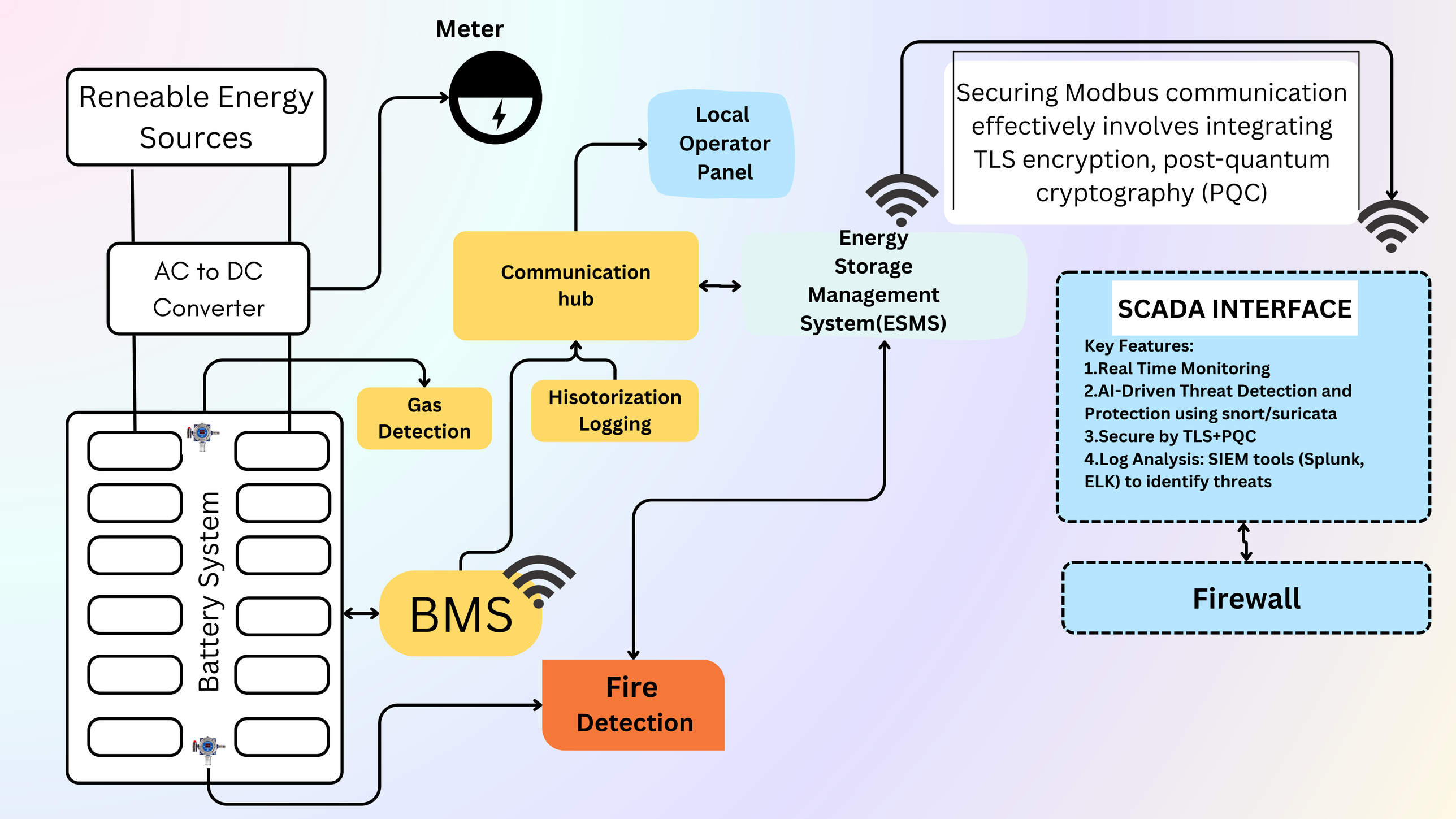


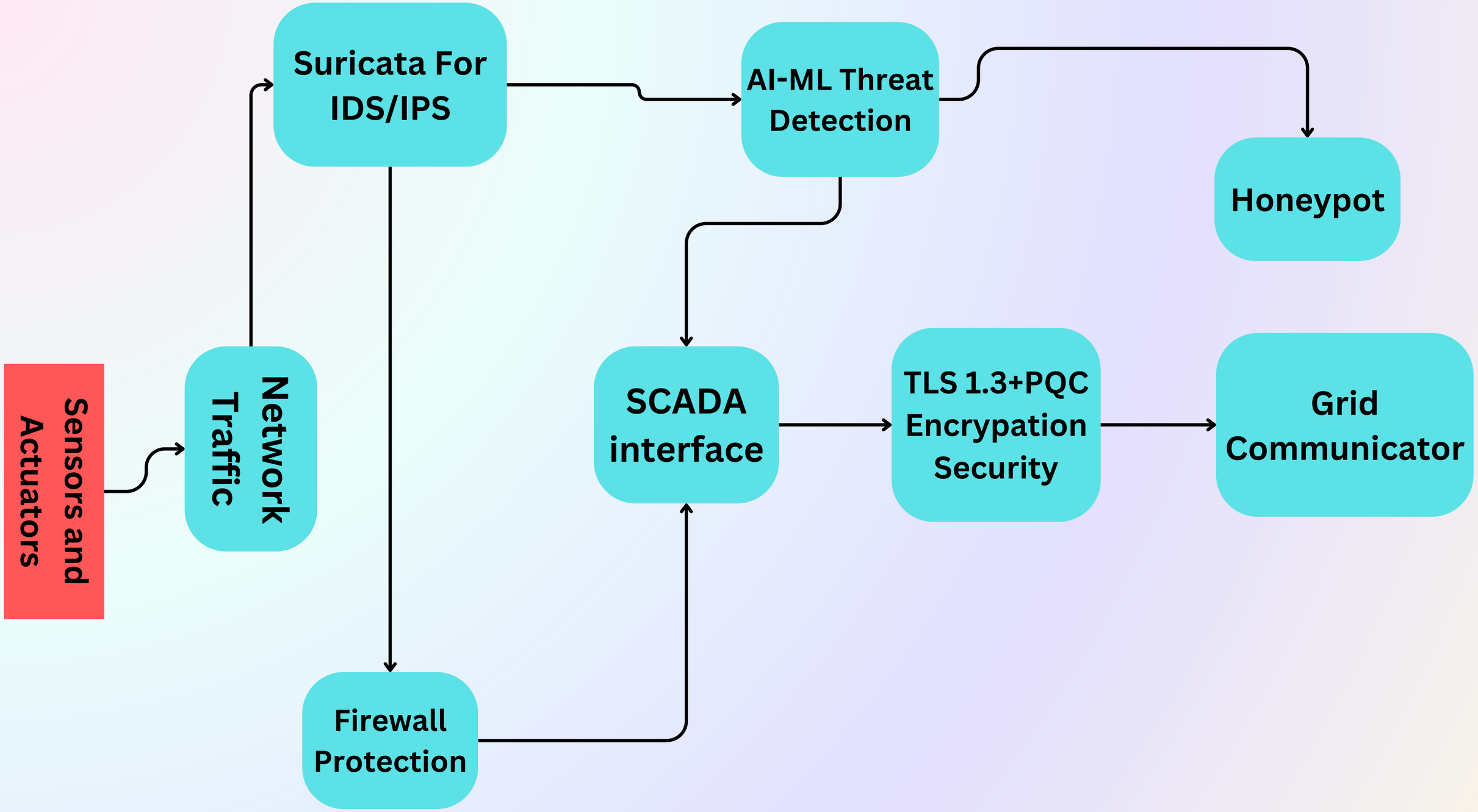
Future Direction

Renewable energy increases grid dependence on intermittent sources. BESS(Battery Energy storage system) ensures stability but faces growing cyber threats like ransomware and DoS. Attacks can disrupt operations and cause major losses. Future risks, including quantum computing, require strong security measures like detection and encryption.

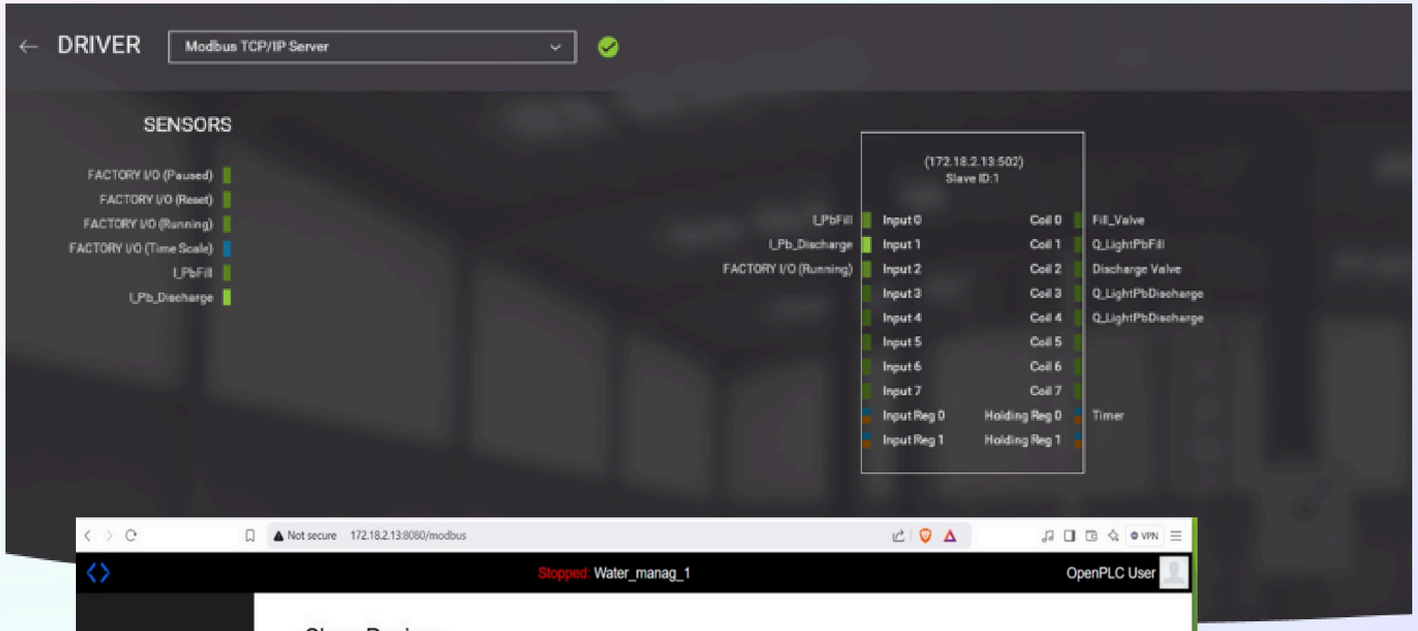


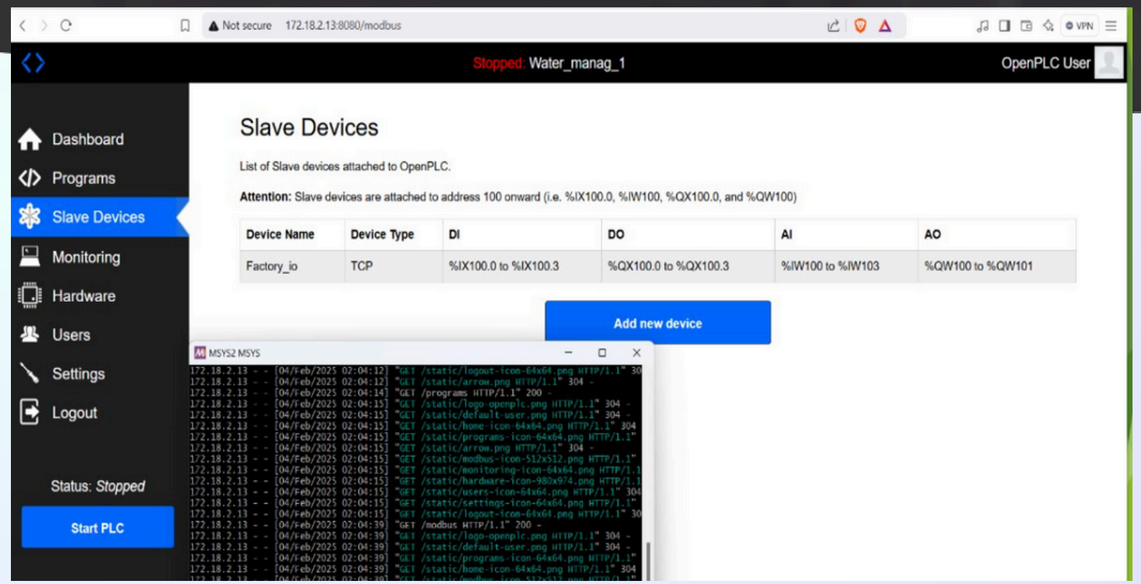
- False Data Injection Attack
- DDOS
- Ransomware
- Future quantum computing threatens


Meter
Reneable Energy Sources
AC to DC Converter
Local Operator Panel
Communication hub
Energy Storage Management System(ESMS)
Securing Modbus communication effectively involves integrating TLS encryption, post-quantum cryptography (PQC)
SCADA INTERFACE
Key Features:
1.Real Time Monitoring
2.AI-Driven Threat Detection and Protection using snort/suricata
3.Secure by TLS+PQC
4.Log Analysis: SIEM tools (Splunk, ELK) to identify threats
Firewall
Gas Detection
Hisotorization Logging
Battery System
BMS
Fire Detection


Suricata For IDS/IPS
AI-ML Threat Detection
Honeypot
Sensors and Actuators
Network Traffic
SCADA interface
TLS 1.3+PQC Encrypation Security
Grid Communicator
Firewall Protection

Factory IO Softtware for virtual scene creation and Drive Settings for making Communication

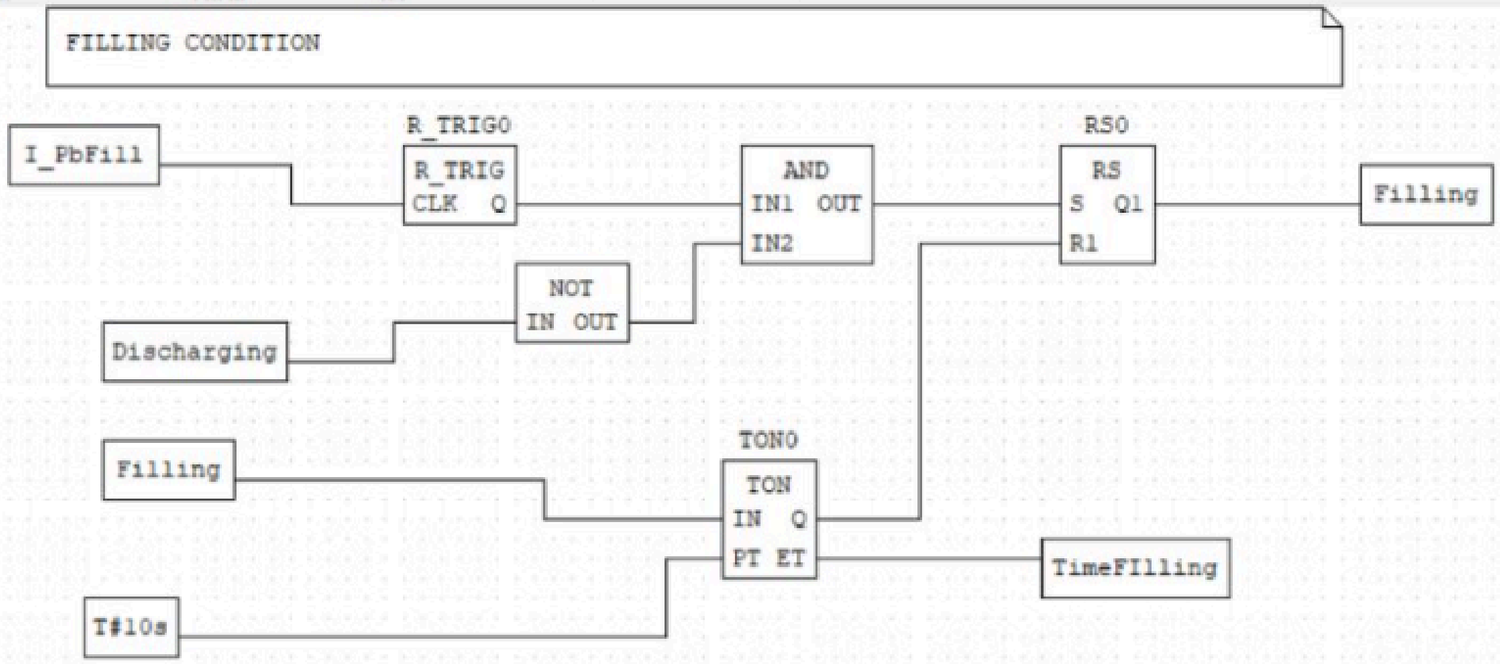




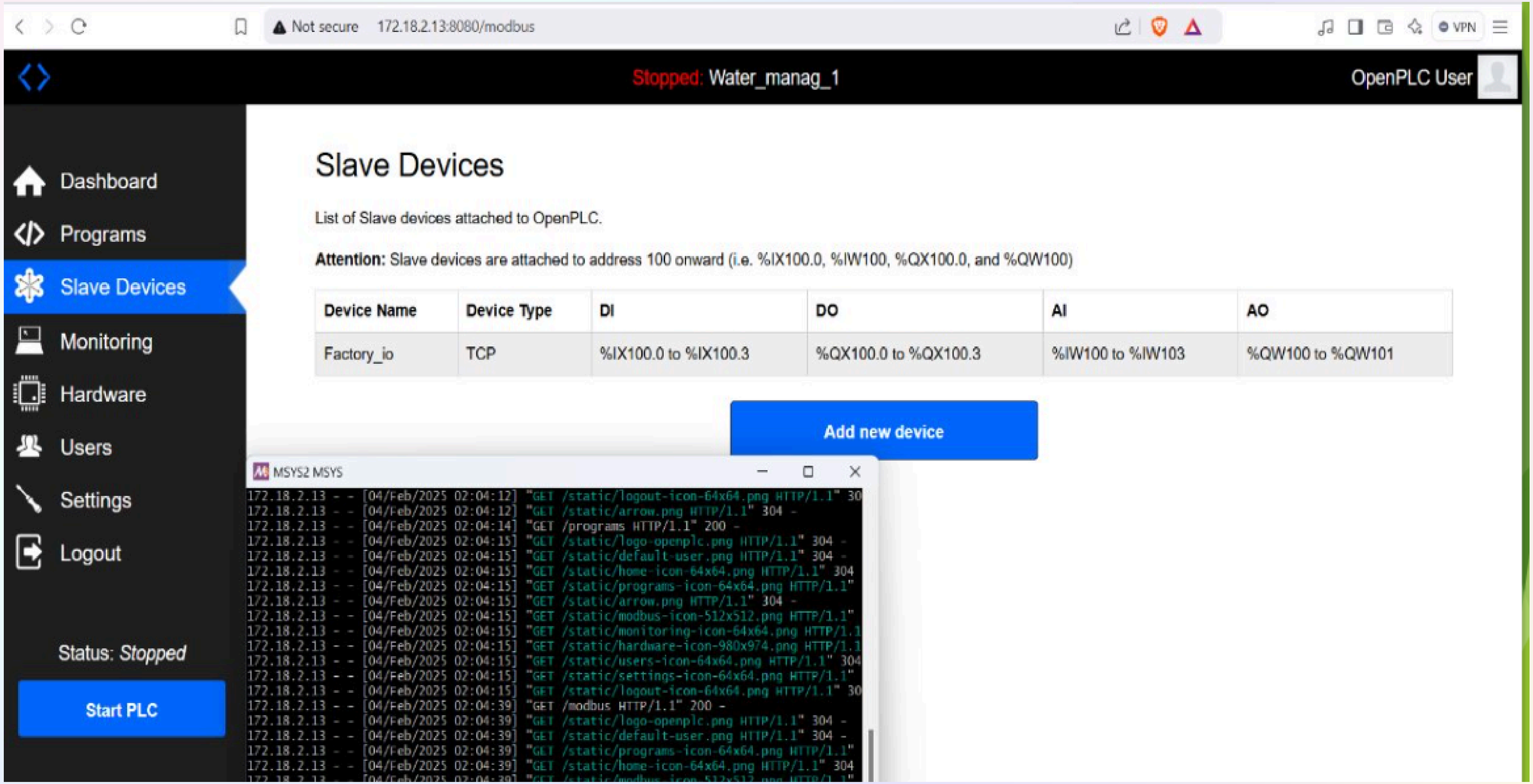
PLC Program for Water tank control system(FBD) in OPEN PLC Software

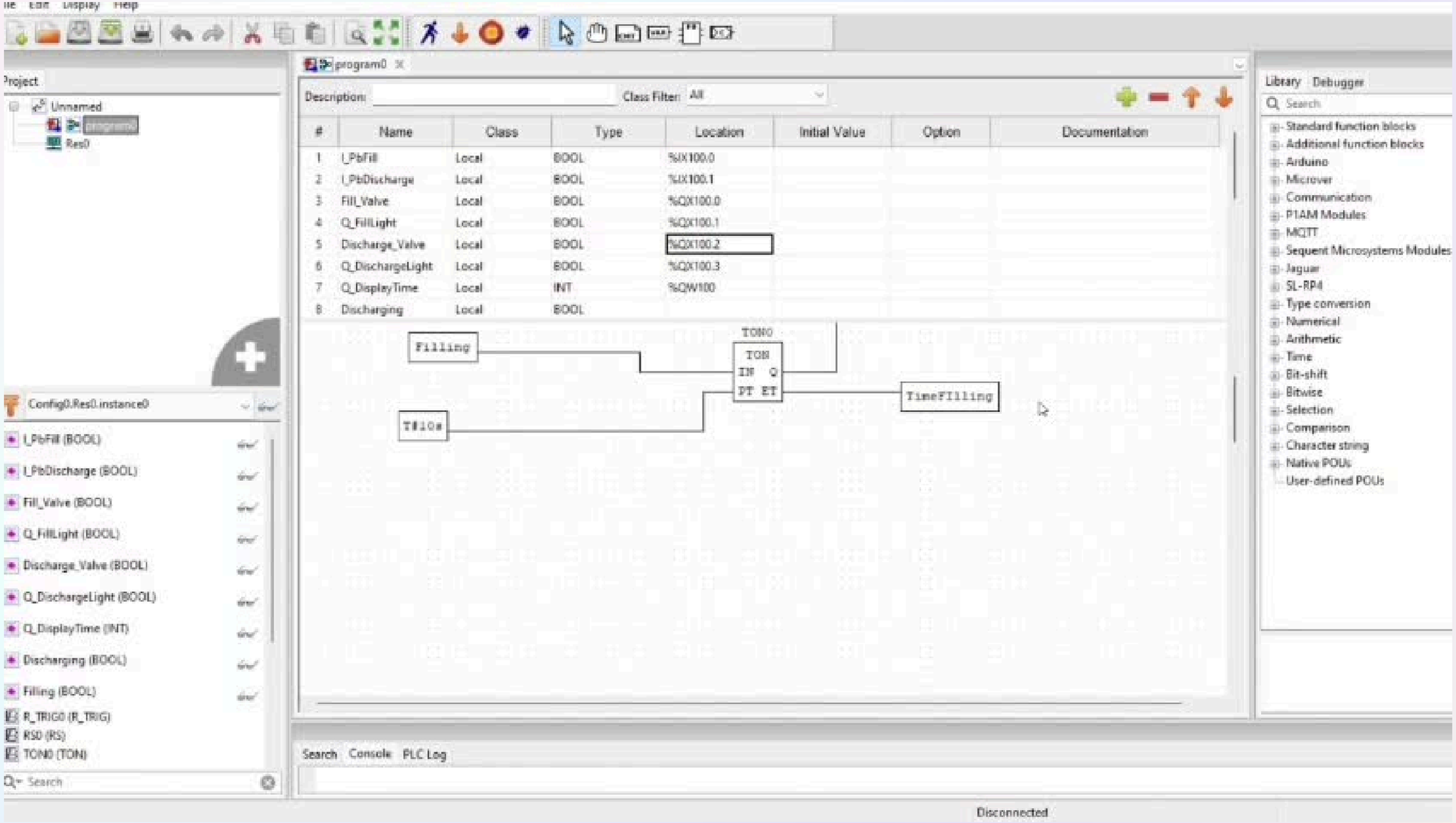
Description: Class Filter: All

#	Name	Class	Type	Location	Initial Value	Option	Documentation
1	I_PbFill	Local	BOOL	%IX100.0			
2	I_PbDischarge	Local	BOOL	%IX100.1			
3	Fill_Valve	Local	BOOL	%QX100.0			
4	Q_FillLight	Local	BOOL	%QX100.1			
5	Discharge_Valve	Local	BOOL	%QX100.2			
6	Q_DischargeLight	Local	BOOL	%QX100.3			
7	Q_DisplayTime	Local	INT	%QW100			
8	Discharging	Local	BOOL				
9	Filling	Local	BOOL				
10	R_TRIG0	Local	R_TRIG				
11	RS0	Local	RS				



Interface of Open PLC Runtime(Host) Making Communication between Factory IO and Open PLC for control remote




program0
Description:
Class Filter: All
Name Class Type Location Initial Value Option Documentation
1 I_PbFill Local BOOL %IX100.0
2 I_PbDischarge Local BOOL %IX100.1
3 Fill_Valve Local BOOL %QX100.0
4 Q_FillLight Local BOOL %QX100.1
5 Discharge_Valve Local BOOL %QX100.2
6 Q_DischargeLight Local BOOL %QX100.3
7 Q_DisplayTime Local INT %QW100
8 Discharging Local BOOL
TON0
Filling
IN Q
PT ET
TimeFilling
T#10s
Project
Unnamed
program0
Res0
Config0.Res0.instance0
I_PbFill (BOOL)
I_PbDischarge (BOOL)
Fill_Valve (BOOL)
Q_FillLight (BOOL)
Discharge_Valve (BOOL)
Q_DischargeLight (BOOL)
Q_DisplayTime (INT)
Discharging (BOOL)
Filling (BOOL)
R_TRIG0 (R_TRIG)
RS0 (RS)
TON0 (TON)
Search
Library Debugger
Search
Standard function blocks
Additional function blocks
Arduino
Microver
Communication
P1AM Modules
MQTT
Sequent Microsystems Modules
Jaguar
SL-RPi
Type conversion
Numerical
Arithmetic
Time
Bit-shift
Bitwise
Selection
Comparison
Character string
Native POUs
User-defined POUs
Search Console PLC Log
Disconnected