DATA-PRIVACY

FOCUSED

FEDERATED LEARNING FRAMEWORK

FOR

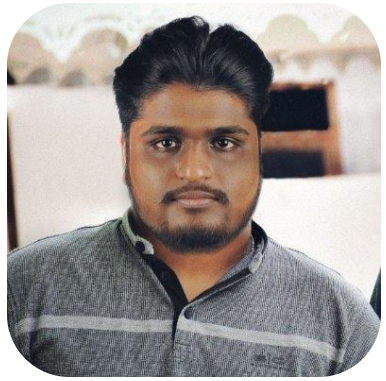
INDUSTRIAL IOT

R25 - 039

PROJECT GROUP

Team

Nanayakkara Y.D.T.D
IT21826368



Mendis H.R.M
IT21822612

Weerasinghe K.M
IT21831904

Dissanayaka K.D.A.R.A
IT21828348

Supervisors

Mr. Amila Nuwan Senarathne
Supervisor

Mr. Tharaniwarma Kyumaralingam
Co-Supervisor

RESEARCH FOCUS

PROBLEM

- Standard Fl systems fall short in IIoT, lacking a unified approach to security , Privacy and efficiency. They remain vulnerable to poisoning and byzantine attacks, and security failures in IIoT can have severe consequences

QUESTION

- How can modular enhancements strengthen FL against data leakage and security threats in industrial IoT environments

SOLUTION

- A modular framework enhancing FL with:
  - hybrid privacy preservation (DP & HE),
  - real-time attack detection
  - lightweight secure aggregation (Shamir's Secret Sharing)
  - Secure Communication and Protocol Enforcement Module

OBJECTIVES

MAIN OBJECTIVES

Design and implement a secure and efficient Federated Learning framework for Industrial IoT that ensures privacy, resilience, and reliability.

SUB OBJECTIVES

- Boost Resilience: Real-time detection & mitigation of poisoning/Byzantine threats.
- Secure Aggregation: Hide individual updates & ensure fault tolerance.
- Secure Communication: Prevent eavesdropping & tampering.
- Evaluate Framework: Measure security, privacy & efficiency in IIoT.

PROVEN GAPS AND LIMITATIONS

- **Data Leakage:** Standard FL is prone to inference attacks, compromising client data privacy.
- **Ineffective Defenses:** Lacks real-time detection for advanced threats like model poisoning and Byzantine attacks.
- **Insecure Communications:** channels often lack robust security, risking tampering.

FRAMEWORK COMPONENTS

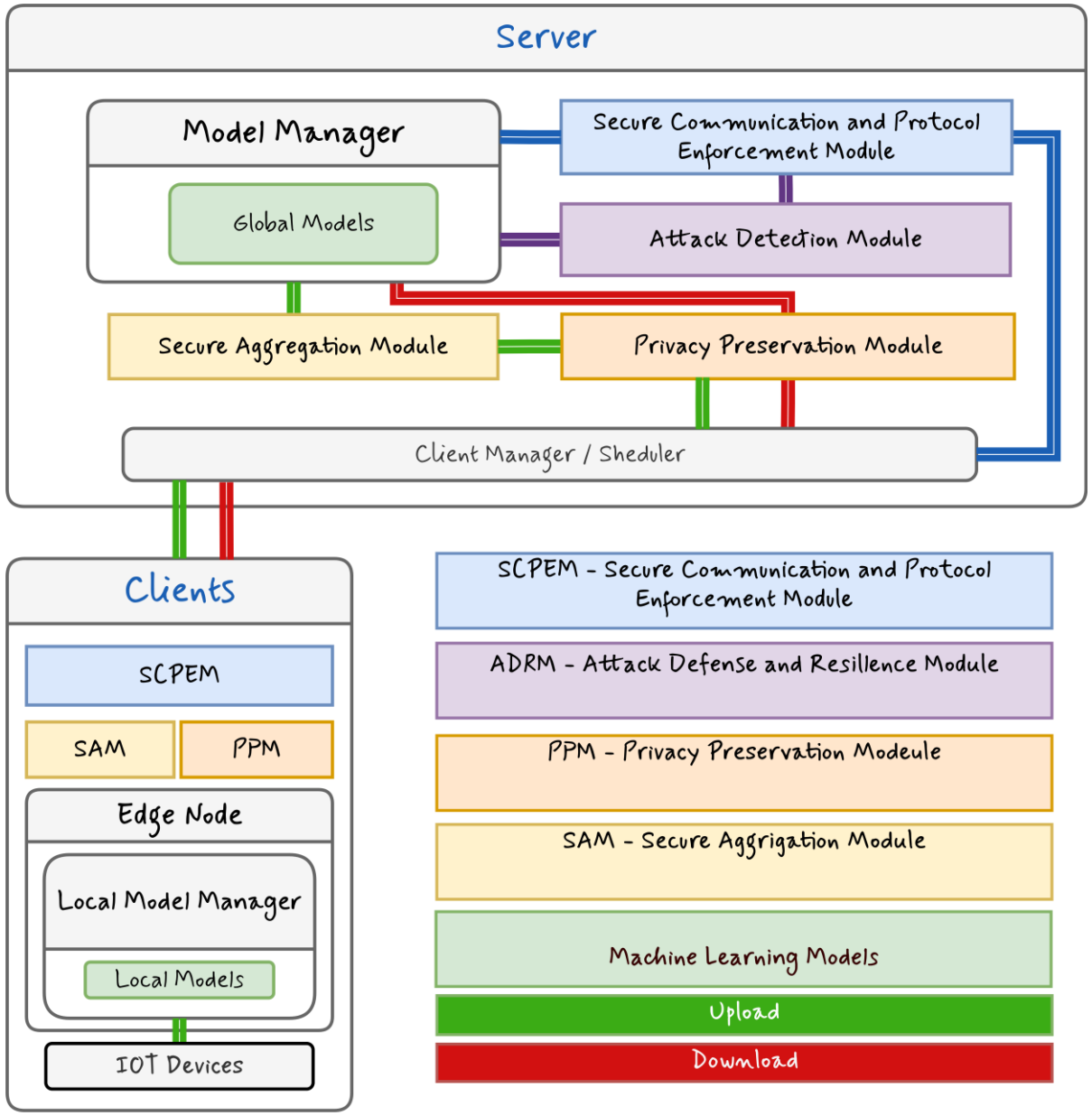
Attack Defence And Resilience Module

Privacy Preservation Module

Secure Aggregation Module

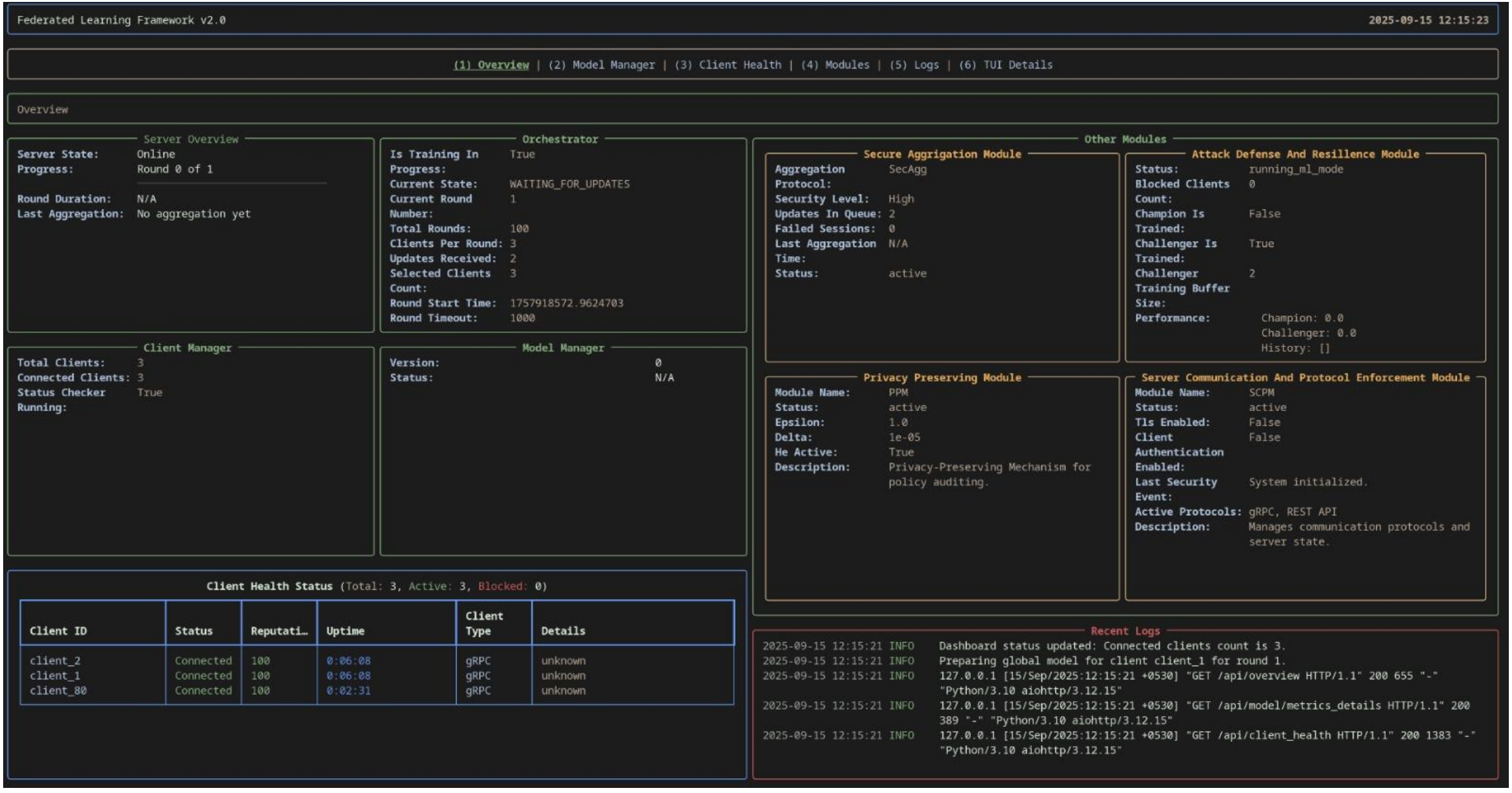
Secure Communication and Protocol Enforcement Module

OVERALL ARCHITECHTURE



- **Modular Design:** Security & privacy integrated in stages of FL
- **Core Components:** Server + Distributed Individual Clients
- **Server:** Orchestrates FL, manages models, coordinates clients, Logging & Monitoring
- **Clients:** Train locally; share only secure updates.
- Security Modules:
  - **SCPM** – Secure communication and protocol enforcement module
  - **ADRM** – Attack Detection & Resilience module
  - **PPM** – Privacy Protection module
  - **SAM** – Secure Aggregation module

Proposed System – Frontend Terminal User Interface(TUI)



Proposed System – Model Manager

Federated Learning Framework v2.0 2025-09-15 02:28:45

(1) Overview | (2) Model Manager | (3) Client Health | (4) Modules | (5) Logs | (6) TUI Details

Model Manager | p: Plot | ↑/↓: Scroll

Model Status

Model Version: 38
Convergence Status: Training
Last Update: 2025-09-15 02:27:08

Best Performance
Accuracy: 33.88%
Loss: 1.8766

Training Progress

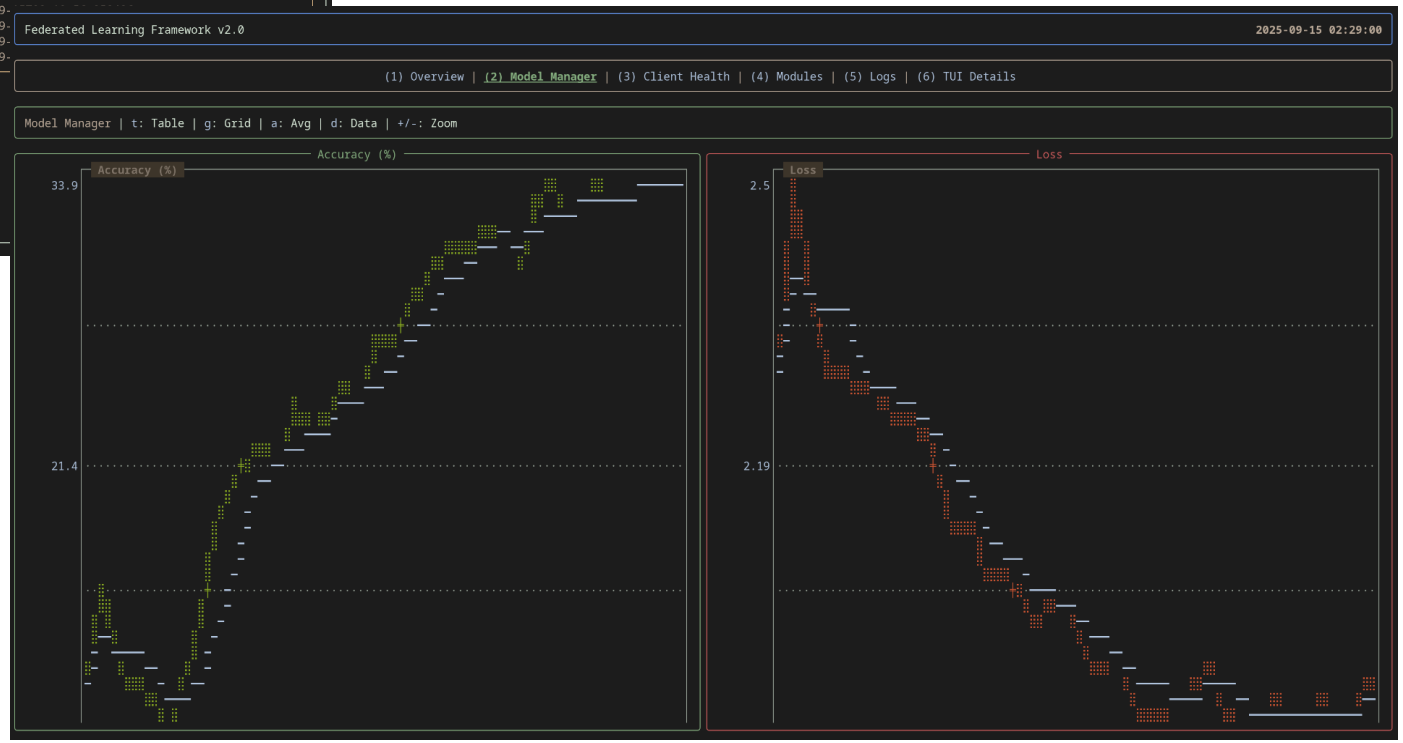
Round 38/100 (38.00%)

Round	Accuracy (%)	Loss	Agg. Method	Timestamp
1	11.26	2.2831	N/A	2025-09-15T02:16:57.462493
2	15.06	2.5019	N/A	2025-09-15T02:17:08.320216
3	11.73	2.3586	N/A	2025-09-15T02:17:18.971671
4	10.74	2.2818	N/A	2025-09-15T02:17:28.893967
5	10.58	2.2850	N/A	2025-09-15T02:17:39.329314
6	8.89	2.2720	N/A	2025-09-15T02:17:49.392638
7	11.27	2.2657	N/A	2025-09-15T02:17:59.638800
8	14.22	2.2307	N/A	2025-09-15T02:18:09.879189
9	18.33	2.2386	N/A	2025-09-15T02:18:20.727648
10	20.64	2.2072	N/A	2025-09-15T02:18:30.348283
11	21.21	2.1592	N/A	2025-09-15T02:18:41.717847
12	21.50	2.1091	N/A	2025-09-
13	20.87	2.1053	N/A	2025-09-
14	24.07	2.0629	N/A	2025-09-
15	22.56	2.0506	N/A	2025-09-

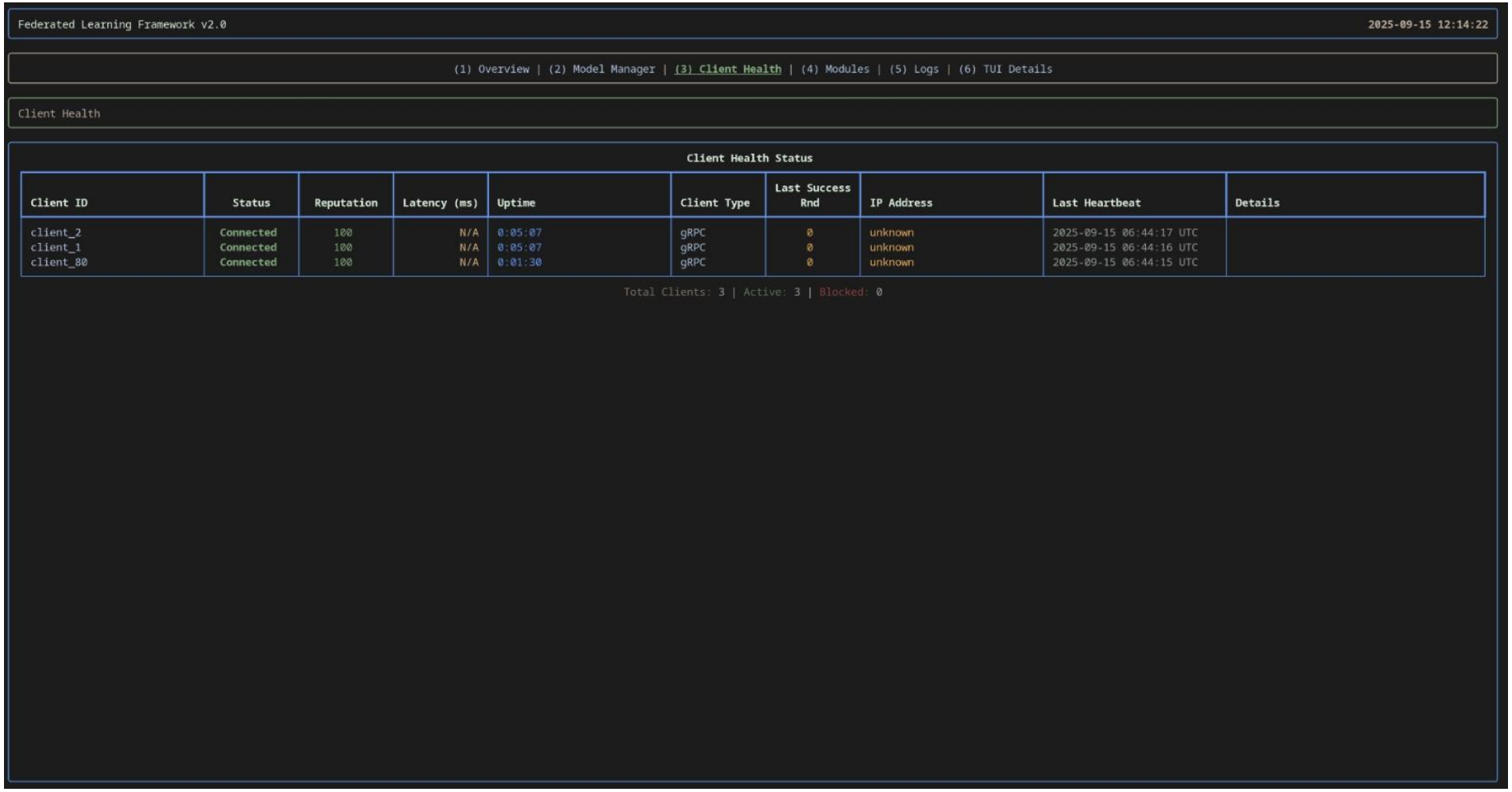
Showing rows 1-15 of 38. Use ↑/↓ to scroll.

Img 1: Global model table

Img 2: Global model plot

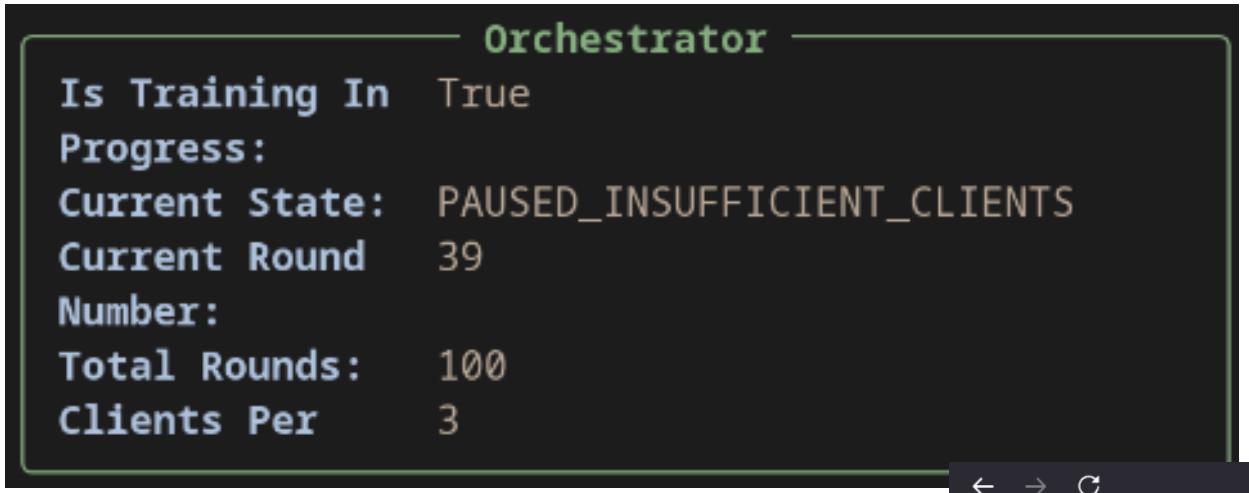


Proposed System – Client Manager



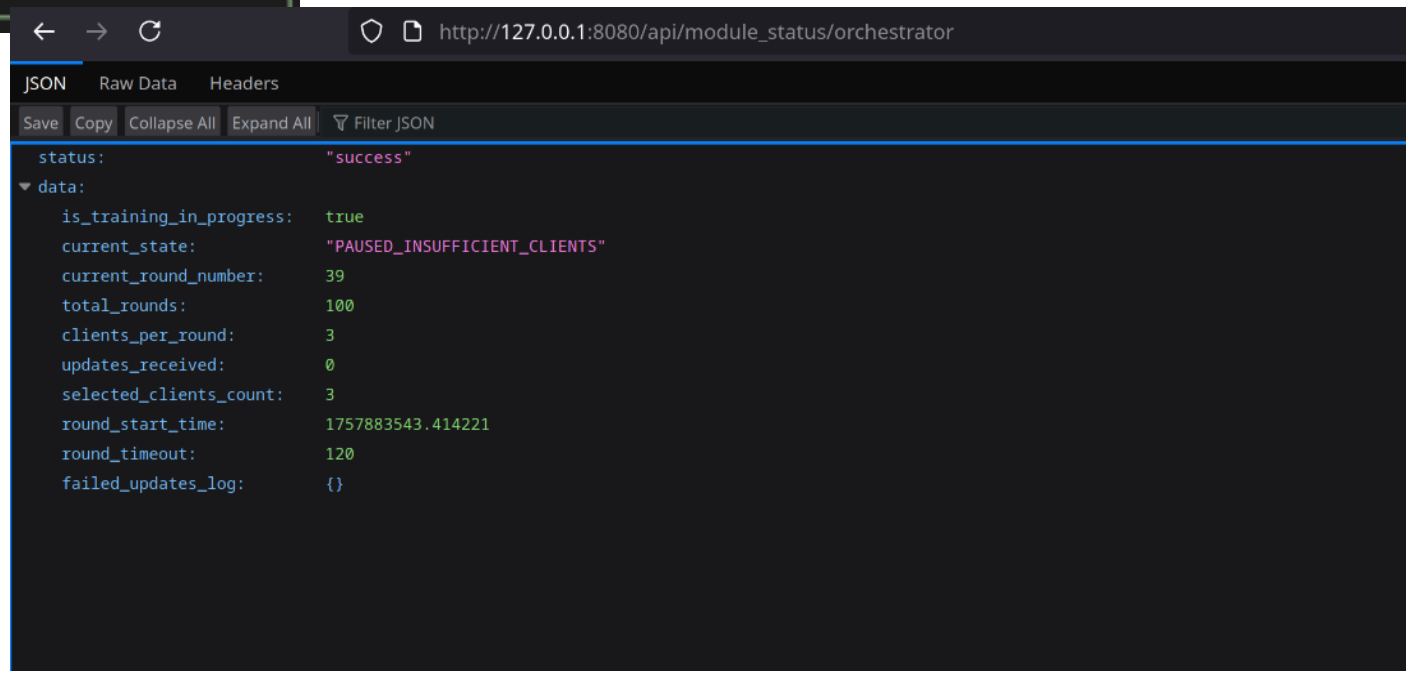
Img 3: Client Health table

Proposed System – Orchestration

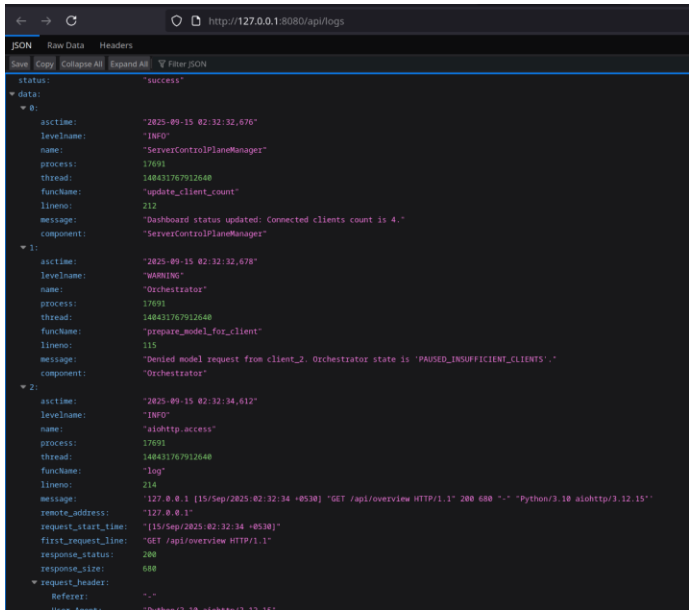


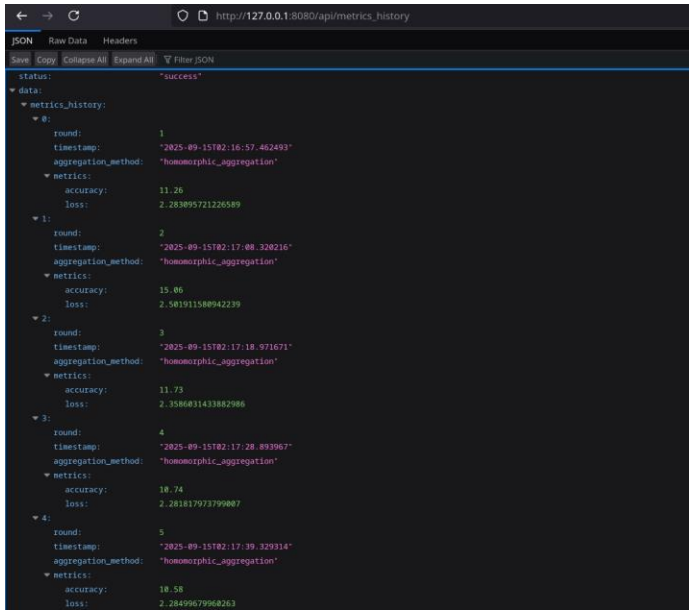
Img 4: training round details

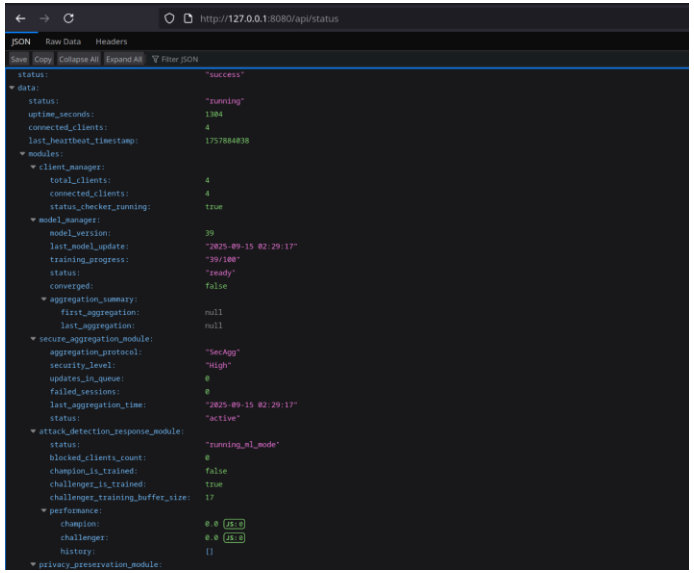
Img 5: Orchestrator API

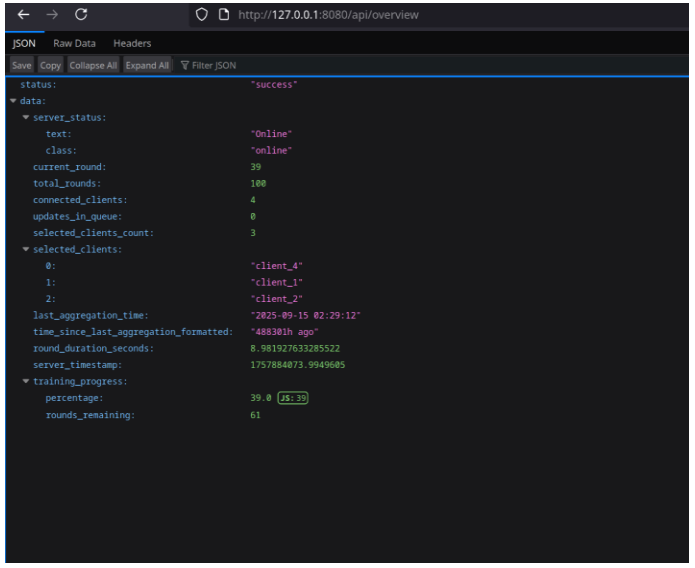


System process –Frontend – API calls










```
    ('GET', '/api/status', self.get_server_status),
    ('GET', '/api/overview', self.get_overview_data),
    ('GET', '/api/orchestrator_progress', self.get_orchestrator_progress),

    # Model & Metrics
    ('GET', '/api/metrics', self.get_model_metrics_data),
    ('GET', '/api/metrics_history', self.get_metrics_history),
    ('GET', '/api/model/metrics_details', self.get_model_metrics_details),
    ('GET', '/api/model', self.get_global_model_json),
    ('GET', '/api/model/bytes', self.get_global_model_bytes),
    ('POST', '/api/submit_update', self.submit_model_update),
    ('GET', '/api/evaluate_model', self.evaluate_model),

    # Client Status
    ('GET', '/api/client_health', self.get_client_health),
    ('GET', '/api/client_privacy_methods', self.get_client_privacy_methods),
    ('GET', '/api/client_heartbeat', self.client_heartbeat),

    # System Internals & Logs
    ('GET', '/api/logs', self.get_logs),
    ('GET', '/api/module_status/mm', self.get_mm_status),
    ('GET', '/api/module_status/sam', self.get_sam_status),
    ('GET', '/api/module_status/adrm', self.get_adrm_status),
    ('GET', '/api/module_status/ppm', self.get_ppm_status),
    ('GET', '/api/module_status/scpm', self.get_scpm_status),
    ('GET', '/api/module_status/orchestrator', self.get_orchestrator_status),

    # Admin Endpoints for ADRM
    ('POST', '/api/admin/adrm/unblock/{client_id}', self.admin_unblock_client),
    ('DELETE', '/api/admin/adrm/history/{client_id}', self.admin_reset_client_history),
    ('PUT', '/api/admin/adrm/config', self.admin_update_adrm_config),
]
```

Attack Detection and Resilience

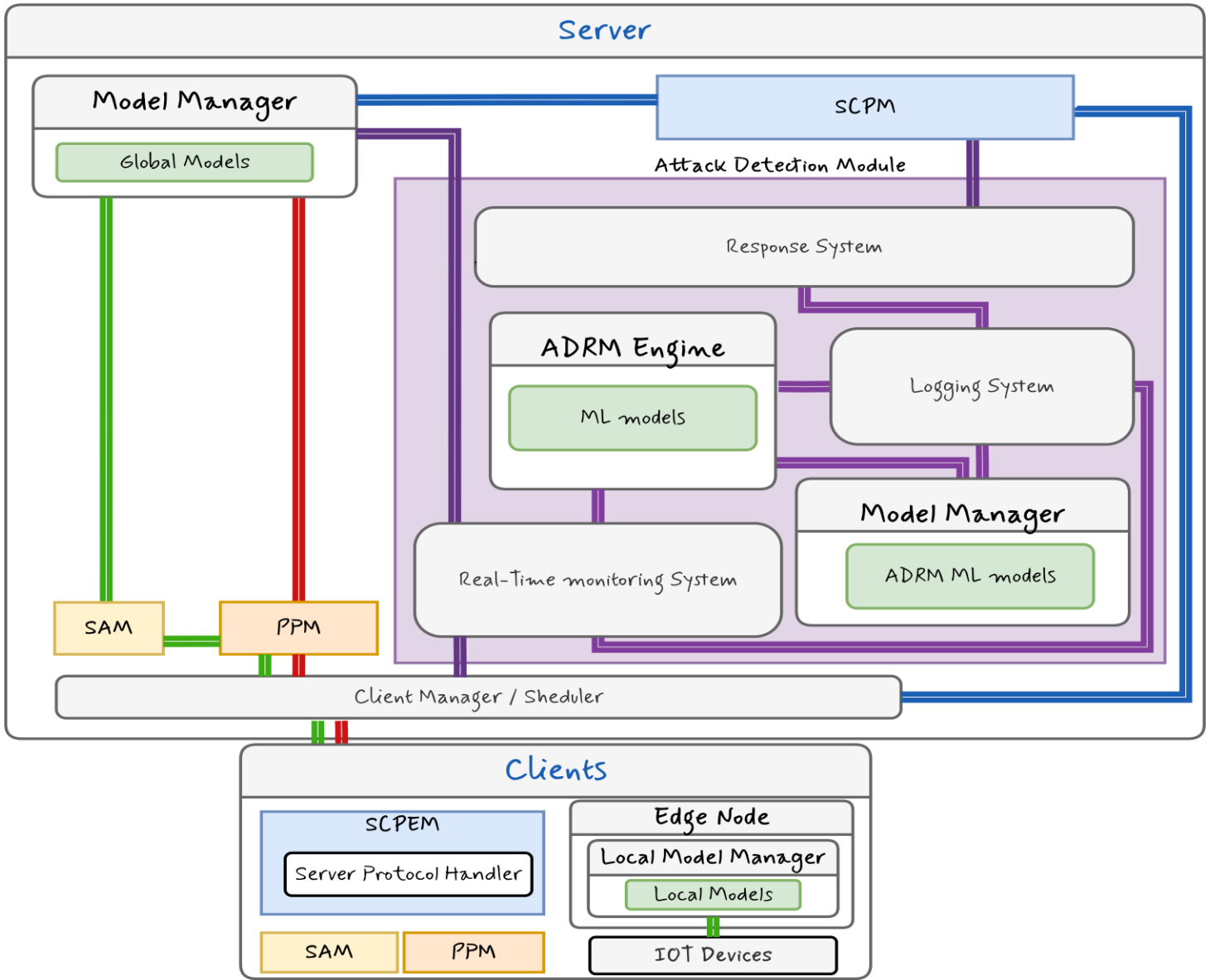
IT21826368 | Nanayakkara Y.D.T.D

- **Background**
  - Federated learning is a promising privacy preserving solution for IIoT security, but its decentralized nature create a new and significant attack surface for malicious actors
  - **Significance** : Without active defenses, Global model integrity is at risk, making robust security vital for industrial applications
- **RESEARCH GAP**
  - **Limitations**: Existing FL defenses are siloed, Targeting single threats, and its also too resource heavy for scalable IIoT
  - High Overhead makes many solutions impractical for IIoT. Also, There are not any unified lightweight approach
- **RESEARCH PROBLEM**
  - How can we design and integrate a lightweight, scalable, and effective attack detection and resilience system directly into the Federated Learning process to protect the global model's integrity in real-time?

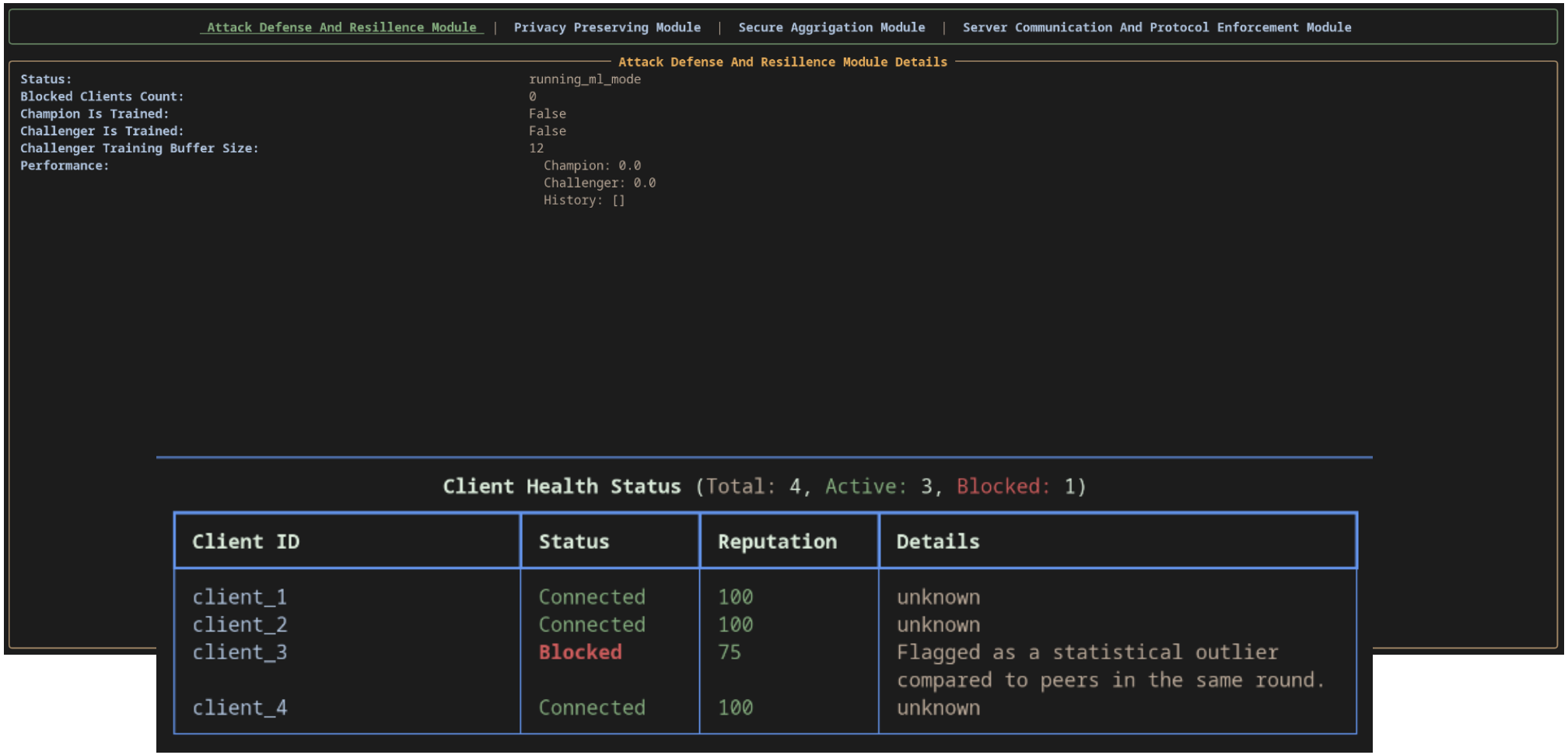
SOLUTION: ATTACK DETECTION & RESILIENCE MODULE

- **Main Objective:** To develop a scalable and lightweight module that enhances the security of FL in IIoT by detecting and mitigating malicious activities in real-time
- The Multi-Layered Approach:
  - **Real-Time Anomaly Detection:** Detects malicious updates with unsupervised learning
  - **Enhanced Resilience:** Ensures continuity with recovery mechanisms like rollbacks
  - **Client reputation grading system** – Reduce client points based on detection

DEVELOPED SOLUTION – MODULE ARCHTECTURE

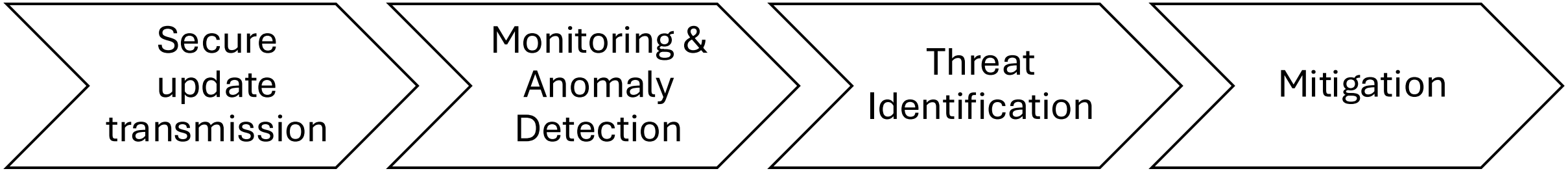


ADRM ARCHITECTURE



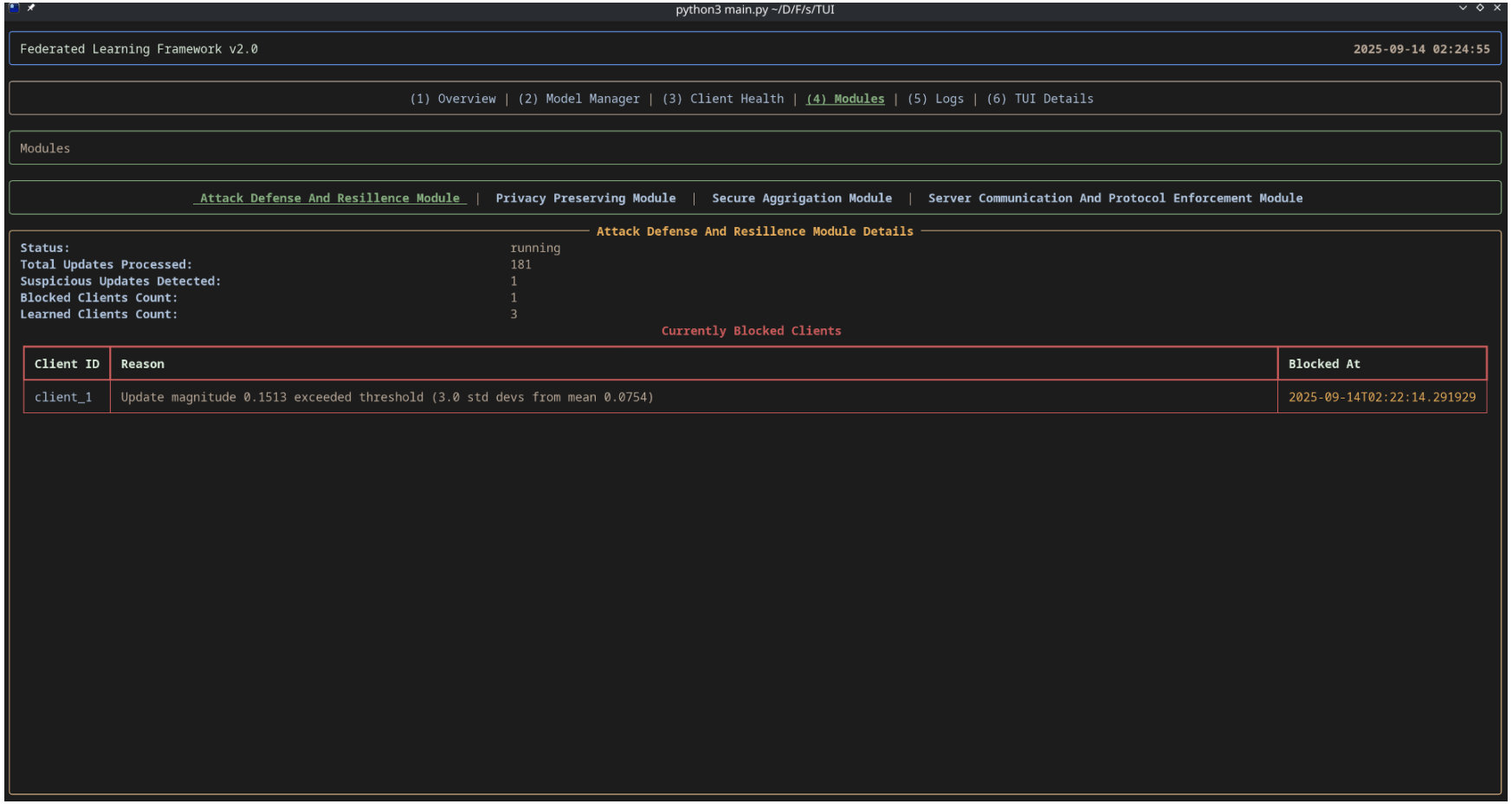
Img 1: reputation grading and client blocking

DETECTION & MITIGATION PROCESS FLOW

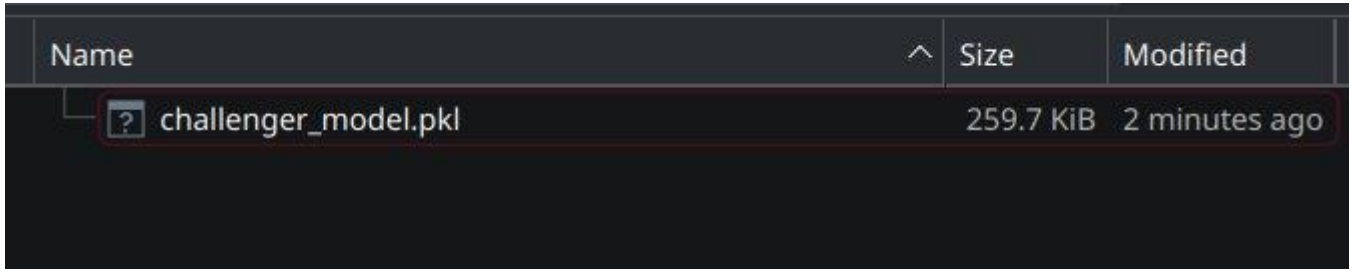


- Clients send encrypted updates via secure channels
- ADRM analyzes gradients, flags deviations
- Detects model anomalies
- Isolates/filter malicious clients; only verified updates aggregated
- Blocks and reduce reputation

Attack detection and Defense



DETECTION METHOD



Created ML Model for anomaly detection

Monitoring and Logging

```
2025-09-15 01:44:30,497 - ADRM_General - INFO - Initializing ADRM ML Model Manager...
2025-09-15 01:44:30,497 - ADRM_General - WARNING - Model file not found at database/adrm_models/champion_model.pkl. Creating a new, untrained model.
2025-09-15 01:44:30,498 - ADRM_General - WARNING - Model file not found at database/adrm_models/challenger_model.pkl. Creating a new, untrained model.
2025-09-15 01:44:30,498 - ADRM_General - INFO - ADRM ML Model Manager initialized.
2025-09-15 01:44:48,331 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:44:48,546 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:44:48,993 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:44:58,594 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:44:58,878 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:44:59,264 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:44:59,267 - ADRM_General - WARNING - Client 'client_3' blocked for 2 minutes. Reason: Flagged as a statistical outlier compared to peers in the same round.
2025-09-15 01:44:59,268 - ADRM_General - WARNING - High severity anomaly for 'client_3'. Update quarantined for review.
2025-09-15 01:46:59,268 - ADRM_General - INFO - Client 'client_3' unblocked. Duration expired.
2025-09-15 01:47:00,990 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:47:01,142 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:47:10,016 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:47:15,912 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:47:16,092 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:47:20,393 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:47:20,402 - ADRM_General - WARNING - Client 'client_2' blocked for 2 minutes. Reason: Flagged as a statistical outlier compared to peers in the same round.
2025-09-15 01:47:20,404 - ADRM_General - WARNING - High severity anomaly for 'client_2'. Update quarantined for review.
2025-09-15 01:49:20,401 - ADRM_General - INFO - Client 'client_2' unblocked. Duration expired.
2025-09-15 01:49:21,288 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:26,987 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:27,367 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:31,622 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:37,441 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:37,765 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:42,688 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:47,881 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:47,882 - ADRM_General - INFO - Challenger training buffer full. Triggering retraining.
2025-09-15 01:49:47,884 - ADRM_General - INFO - Training Isolation Forest on 20 samples...
2025-09-15 01:49:48,417 - ADRM_General - INFO - Training complete.
2025-09-15 01:49:48,550 - ADRM_General - INFO - Model saved successfully to database/adrm_models/challenger_model.pkl
2025-09-15 01:49:48,556 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:53,731 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:59,137 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:49:59,240 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:50:04,057 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:50:09,557 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:50:09,775 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:50:09,778 - ADRM_General - WARNING - Client 'client_1' blocked for 2 minutes. Reason: Flagged as a statistical outlier compared to peers in the same round.
2025-09-15 01:50:09,779 - ADRM_General - WARNING - High severity anomaly for 'client_1'. Update quarantined for review.
2025-09-15 01:52:09,778 - ADRM_General - INFO - Client 'client_1' unblocked. Duration expired.
2025-09-15 01:52:10,362 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:52:14,617 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:52:20,120 - ADRM_General - WARNING - Prediction skipped: model is not trained yet. Approving by default.
2025-09-15 01:52:20,128 - ADRM_General - WARNING - Client 'client_1' blocked for 2 minutes. Reason: Flagged as a statistical outlier compared to peers in the same round.
2025-09-15 01:52:20,130 - ADRM_General - WARNING - High severity anomaly for 'client_1'. Update quarantined for review.
2025-09-15 01:54:20,128 - ADRM_General - INFO - Client 'client_1' unblocked. Duration expired.
```

Model creation based on dataset in the current FL system

Attack detection

```
2025-09-15 01:44:59,267 - WARNING - CROSS-CLIENT outlier detected: client_3 with Z-Score=7.43
2025-09-15 01:44:59,267 - WARNING - Response triggered for 'client_3'. Severity: high. Reason: Flagged as a statistical outlier compared to peers in the same round.
2025-09-15 01:44:59,267 - WARNING - Client 'client_3' blocked for 2 minutes. Reason: Flagged as a statistical outlier compared to peers in the same round. | Reputation Penalty: -25 points.
2025-09-15 01:44:59,267 - INFO - Blocklist with 1 clients saved to file.
2025-09-15 01:46:59,269 - INFO - Client 'client_3' unblocked. Duration expired.
2025-09-15 01:46:59,271 - INFO - Blocklist with 0 clients saved to file.
2025-09-15 01:47:20,400 - WARNING - CROSS-CLIENT outlier detected: client_2 with Z-Score=5.22
2025-09-15 01:47:20,401 - WARNING - Response triggered for 'client_2'. Severity: high. Reason: Flagged as a statistical outlier compared to peers in the same round.
2025-09-15 01:47:20,403 - WARNING - Client 'client_2' blocked for 2 minutes. Reason: Flagged as a statistical outlier compared to peers in the same round. | Reputation Penalty: -25 points.
2025-09-15 01:47:20,404 - INFO - Blocklist with 1 clients saved to file.
2025-09-15 01:49:20,412 - INFO - Client 'client_2' unblocked. Duration expired.
2025-09-15 01:49:20,414 - INFO - Blocklist with 0 clients saved to file.
2025-09-15 01:50:09,778 - WARNING - CROSS-CLIENT outlier detected: client_1 with Z-Score=4.38
2025-09-15 01:50:09,778 - WARNING - Response triggered for 'client_1'. Severity: high. Reason: Flagged as a statistical outlier compared to peers in the same round.
2025-09-15 01:50:09,779 - WARNING - Client 'client_1' blocked for 2 minutes. Reason: Flagged as a statistical outlier compared to peers in the same round. | Reputation Penalty: -25 points.
2025-09-15 01:50:09,779 - INFO - Blocklist with 1 clients saved to file.
2025-09-15 01:52:09,779 - INFO - Client 'client_1' unblocked. Duration expired.
2025-09-15 01:52:09,779 - INFO - Blocklist with 0 clients saved to file.
2025-09-15 01:52:20,128 - WARNING - CROSS-CLIENT outlier detected: client_1 with Z-Score=4.98
2025-09-15 01:52:20,128 - WARNING - Response triggered for 'client_1'. Severity: high. Reason: Flagged as a statistical outlier compared to peers in the same round.
2025-09-15 01:52:20,129 - WARNING - Client 'client_1' blocked for 2 minutes. Reason: Flagged as a statistical outlier compared to peers in the same round. | Reputation Penalty: -25 points.
2025-09-15 01:52:20,129 - INFO - Blocklist with 1 clients saved to file.
2025-09-15 01:54:20,129 - INFO - Client 'client_1' unblocked. Duration expired.
2025-09-15 01:54:20,129 - INFO - Blocklist with 0 clients saved to file.
```

Privacy Preservation

IT21822612 | Mendis H.R.M

- **BACKGROUND**

Federated Learning enables decentralized model training but remains vulnerable to inference attacks from untrusted servers. Combining **Differential Privacy (DP)** for noise and **Homomorphic Encryption (HE)** for secure communication provides a hybrid solution to protect client data.

- **RESEARCH GAP**

Existing Federated Learning methods use either Differential Privacy or Homomorphic Encryption alone, failing to provide full protection without compromising accuracy or efficiency, especially in resource-limited IIoT settings.

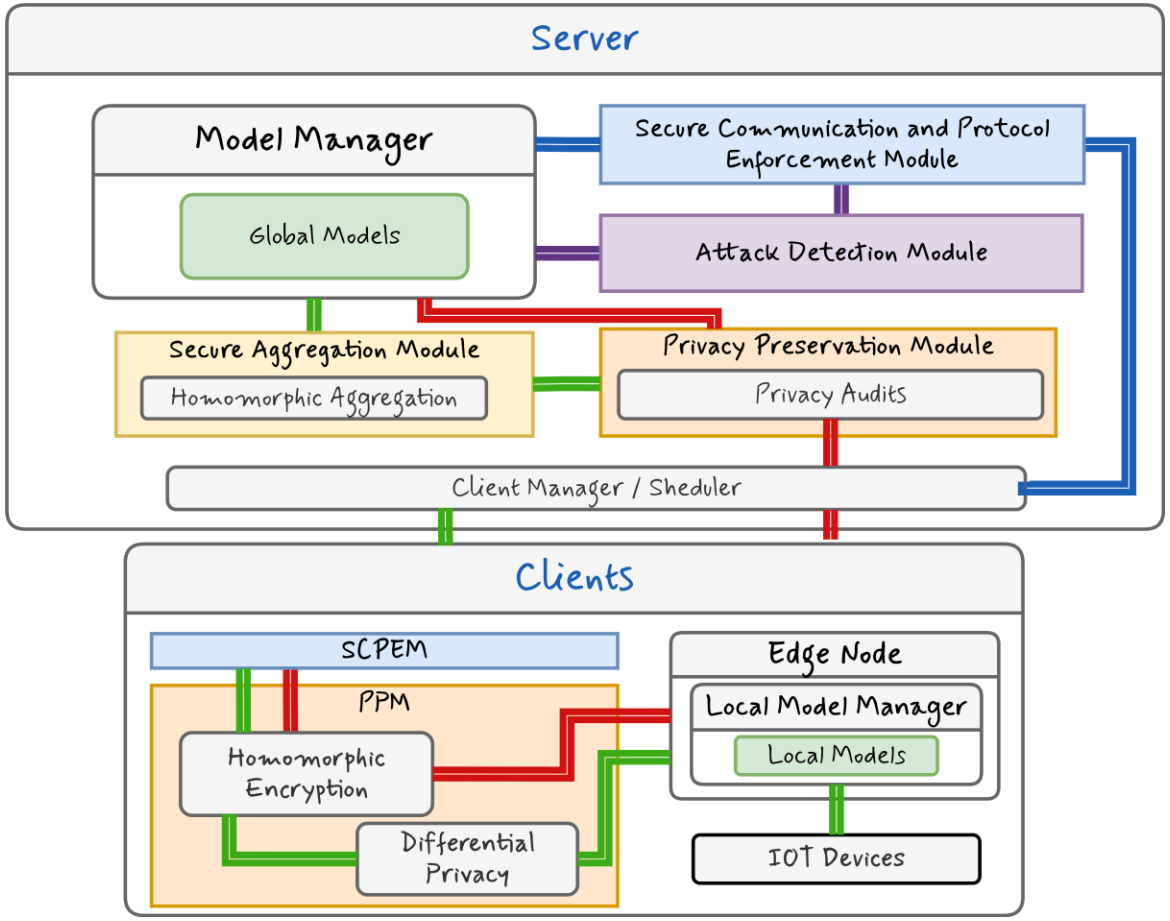
RESEARCH PROBLEM

How can we design a hybrid Federated Learning framework combining Differential Privacy and Homomorphic Encryption to protect client data from inference attacks by untrusted servers while maintaining model accuracy and communication efficiency?

SOLUTION:

- A hybrid privacy-preserving Federated Learning framework that combines **Differential Privacy (DP)** to add noise during training and **Homomorphic Encryption (HE)** to secure communication of model updates. This approach protects client data from inference attacks by untrusted servers while balancing model accuracy and communication efficiency, tailored for resource-constrained Industrial IoT environments.

PPM Architecture



METHODOLOGY

Approach:

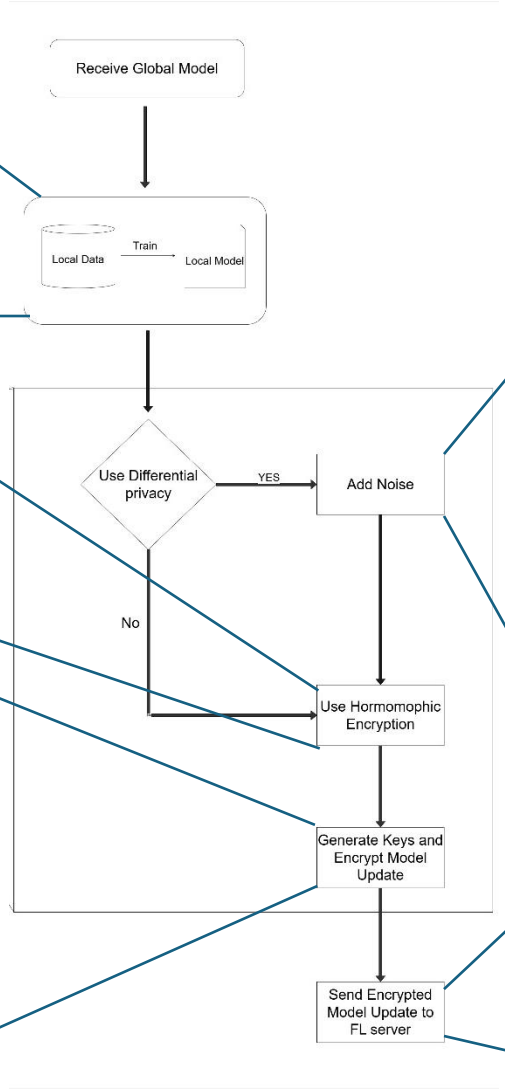
- Analyze existing FL privacy vulnerabilities.
- Combine HE and DP for enhanced privacy.
- Optimize techniques for IIoT-specific constraints.
- Validate Using real-world Datasets

Key Techniques:

- **Homomorphic Encryption (HE)**: Encrypts gradients, allowing computations on encrypted data without decrypting it. Prevents data leakage even if adversaries intercept communications.
- **Differential Privacy (DP)**: Ensure that individual data points cannot be separated by adding controlled noise to gradients. Balances model accuracy with privacy

FUNCTIONAL UNITS & IMPLEMENTATIONS (Client)

```python
def load_and_train_model(epochs=5):
    """Load data and train the model with the given number of epochs."""
    logging.debug("[MODEL] Loading data and starting training")
    x_train, y_train, x_test, y_test = load_data()
    model = get_model()
    model = train_local_model(model, x_train, y_train, x_test, y_test, epochs)
    logging.info("[MODEL] Training completed")
    return model, x_test, y_test
```




```python
        # Encrypt weights if HE is used
        if "HE" in mode:
            enc_weights = [encrypt(w, context) for w in weights]
            comm_bytes += sum([w.numel() * w.element_size() for w in weights])  # estimate comm
            client_weights.append(enc_weights)
        else:
            comm_bytes += sum([w.numel() * w.element_size() for w in weights])
            client_weights.append(weights)
```

```python
# Create a TenSEAL context (CKKS scheme)
def create_context():

    context = ts.context(
        ts.SCHEME_TYPE.CKKS,
        poly_modulus_degree=16384,
        coeff_mod_bit_sizes=[60, 40, 40, 60]
    )
    context.generate_galois_keys()
    context.global_scale = 2**40
    return context

# Encrypt a tensor
def encrypt(tensor, context):
    flat = tensor.detach().cpu().flatten().tolist()
    enc = ts.ckks_vector(context, flat)
    return enc, tensor.size()
```

```python
# DP functions for testing
class DifferentialPrivacyHandler:
    def __init__(self, noise_multiplier=0.2):
        """
        noise_multiplier : Controls scale of Gaussian Noise
        """
        self.noise_multiplier = noise_multiplier
        logging.debug(f"[DP INIT] Noise Multiplier set to {self.noise_multiplier}")

    def add_noise_to_weights(self, weights):
        """Add Gaussian noise to model weights"""
        logging.debug(f"[DP] Starting to add noise to weights")
        noisy_weights = []
        for idx, w in enumerate(weights):
            noise = np.random.normal(loc=0.0, scale=self.noise_multiplier, size=w.shape)
            logging.debug(f"[DP] Layer-{idx} | Original Weights Shape: {w.shape} | Noise Std: {self.noise_multiplier}")
            noisy_w = w + noise
            noisy_weights.append(noisy_w)
        logging.info("[DP] Successfully added noise to all model weights")
        return noisy_weights
```

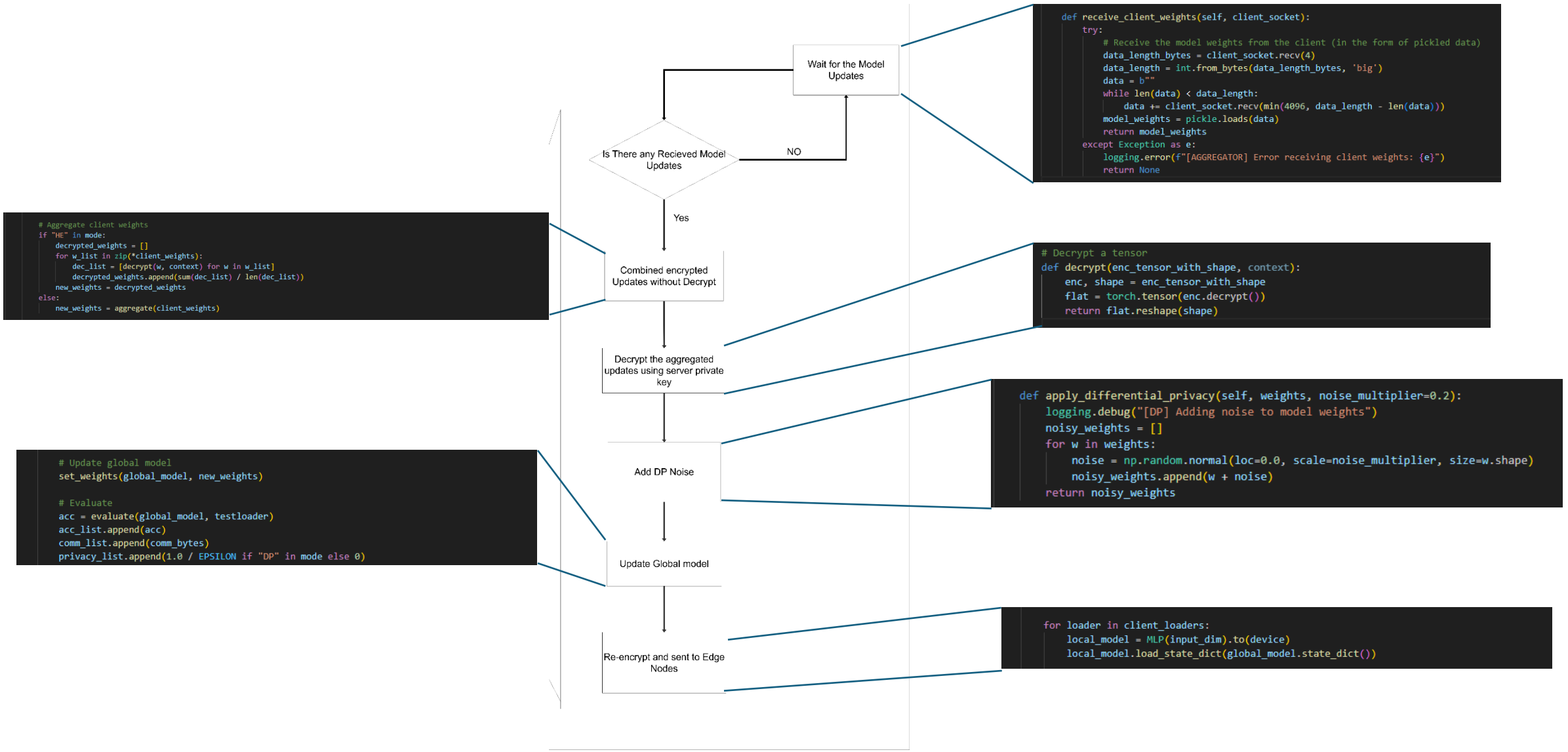
```python
def send_model_weights(model):
    """Serialize the model weights after adding DP noise"""
    try:
        logging.debug("[SEND] Extracting model weights")
        weights = model.get_weights()

        logging.debug("[SEND] Applying Differential Privacy to weights")
        noisy_weights = dp_handler.add_noise_to_weights(weights)

        logging.debug("[SEND] Serializing noisy weights for transmission")
        model_weights = pickle.dumps(noisy_weights)

        logging.info("[SEND] Model weights successfully noise-added and serialized")
        return model_weights
    except Exception as e:
        logging.error(f"[SEND] Error during weight processing: {e}")
        return None
```

FUNCTIONAL UNITS & IMPLEMENTATIONS (Server)




```python
def receive_client_weights(self, client_socket):
    try:
        # Receive the model weights from the client (in the form of pickled data)
        data_length_bytes = client_socket.recv(4)
        data_length = int.from_bytes(data_length_bytes, 'big')
        data = b""
        while len(data) < data_length:
            data += client_socket.recv(min(4096, data_length - len(data)))
        model_weights = pickle.loads(data)
        return model_weights
    except Exception as e:
        logging.error(f"[AGGREGATOR] Error receiving client weights: {e}")
        return None
```

```python
# Aggregate client weights
if "HE" in mode:
    decrypted_weights = []
    for w_list in zip(*client_weights):
        dec_list = [decrypt(w, context) for w in w_list]
        decrypted_weights.append(sum(dec_list) / len(dec_list))
    new_weights = decrypted_weights
else:
    new_weights = aggregate(client_weights)
```

```python
# Decrypt a tensor
def decrypt(enc_tensor_with_shape, context):
    enc, shape = enc_tensor_with_shape
    flat = torch.tensor(enc.decrypt())
    return flat.reshape(shape)
```

```python
def apply_differential_privacy(self, weights, noise_multiplier=0.2):
    logging.debug("[DP] Adding noise to model weights")
    noisy_weights = []
    for w in weights:
        noise = np.random.normal(loc=0.0, scale=noise_multiplier, size=w.shape)
        noisy_weights.append(w + noise)
    return noisy_weights
```

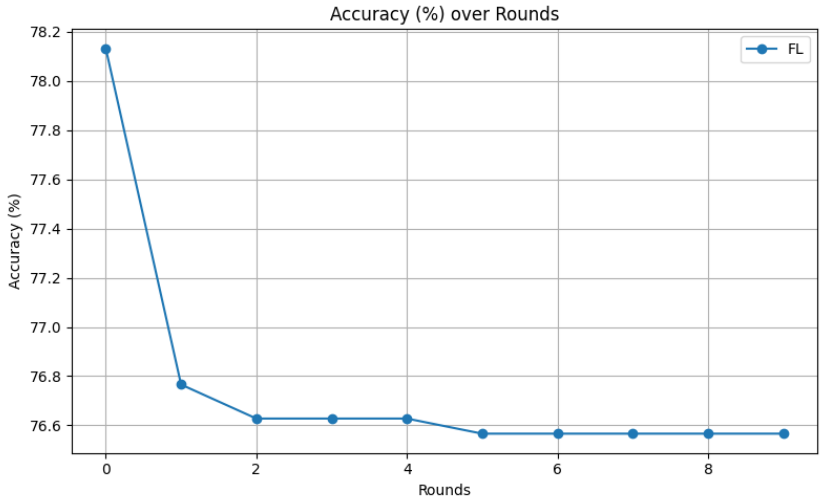
```python
# Update global model
set_weights(global_model, new_weights)

# Evaluate
acc = evaluate(global_model, testloader)
acc_list.append(acc)
comm_list.append(comm_bytes)
privacy_list.append(1.0 / EPSILON if "DP" in mode else 0)
```

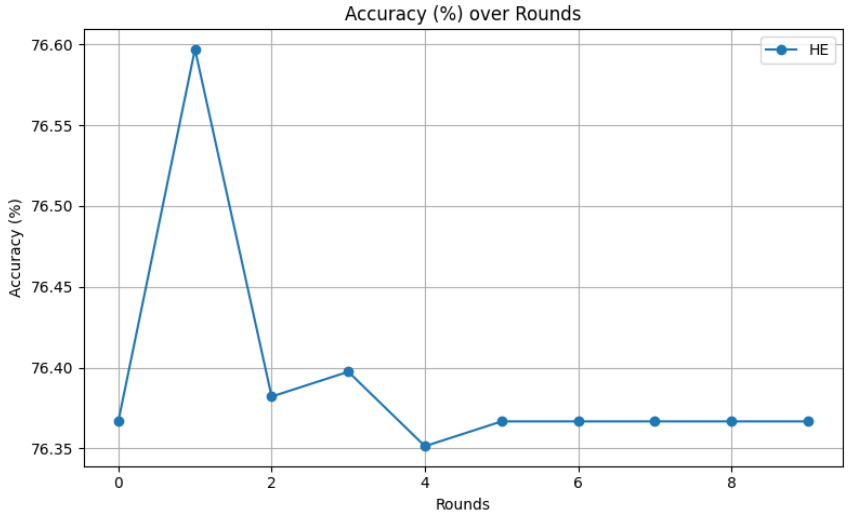
```python
for loader in client_loaders:
    local_model = MLP(input_dim).to(device)
    local_model.load_state_dict(global_model.state_dict())
```

PROOFS & RESULTS

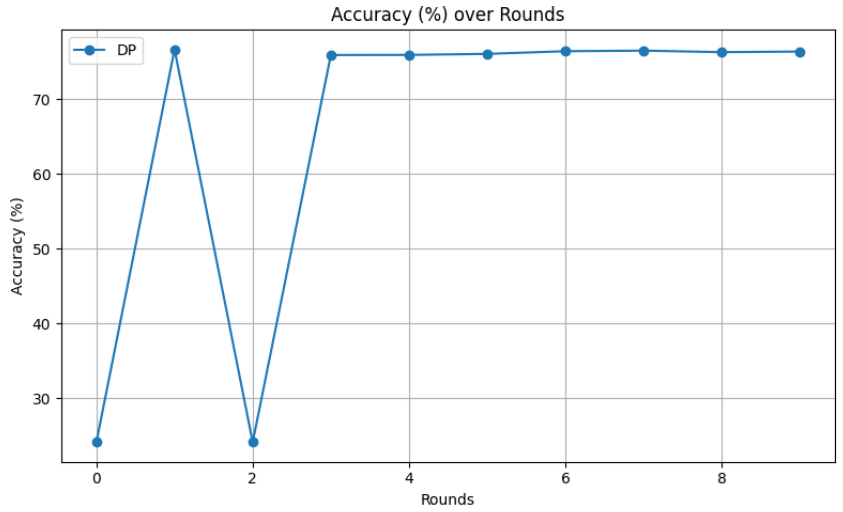
- **Accuracy Curves**



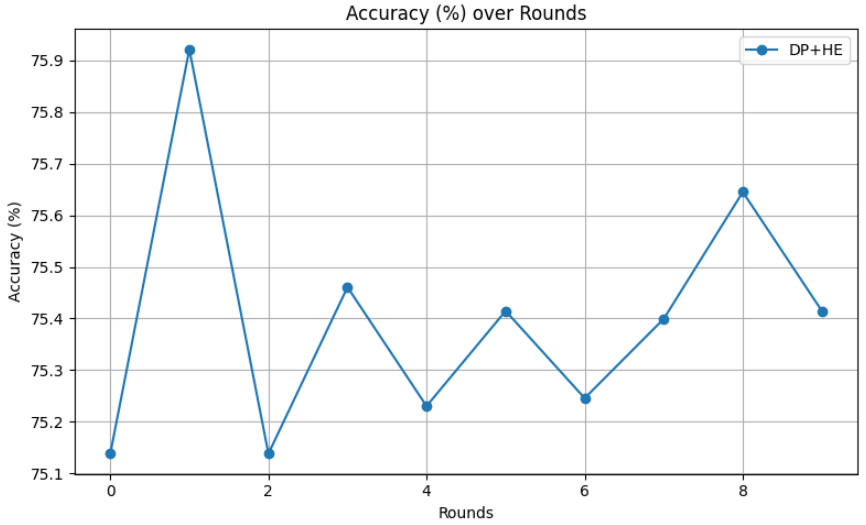
FL



FL + HE



FL + DP



FL + DP + HE

Note: Testing was done using UCI Adult Dataset.

PROOFS & RESULTS

Average Accuracy over Rounds

Mode	Average Accuracy (%)	Remarks
FL	76.81	Highest accuracy because no extra privacy mechanisms are applied.
FL + DP	65.87	~11% Lower accuracy due to Differential Privacy adds **noise** to gradients to protect individuals.
FL + HE	76.39	Differential Privacy adds **noise** to gradients/weights to protect individuals. Homomorphic Encryption protects updates **without changing the data** (no noise).
FL + DP + HE	75.34	Accuracy only slightly lower than FL (~1.5% drop)

FL + DP + HE is the best choice, balancing utility and privacy. FL alone has the highest accuracy but no privacy. DP offers strong privacy with lower accuracy, while HE protects only communication. Combining DP and HE gives strong privacy with minimal accuracy loss, making it both practical and secure.

Evidence of Completion

```
2025-04-07 00:54:03,110 - DEBUG - [DP INIT] Noise Multiplier set to 0.2
Connected to server at 127.0.0.1:5001 from static port 6001.
2025-04-07 00:54:15,729 - INFO - Waiting to receive initial model from the server.
2025-04-07 00:54:15,733 - INFO - Expecting 670665 bytes of model data.
2025-04-07 00:54:15,738 - INFO - Successfully received 670665 bytes of model data.
2025-04-07 00:54:15,739 - INFO - [RECEIVE] Successfully deserialized model weights (bytes)
2025-04-07 00:54:15,739 - DEBUG - [MODEL] Loading data and starting training
2025-04-07 00:54:17,836 - DEBUG - [MODEL] Creating new model instance
```

```
(venv) PS C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular> python main.py --mode FL

=== Running mode: FL ===
Round 1/10 | Mode: FL | Acc: 78.13% | Comm: 215080 bytes
Round 2/10 | Mode: FL | Acc: 76.77% | Comm: 215080 bytes
Round 3/10 | Mode: FL | Acc: 76.63% | Comm: 215080 bytes
Round 4/10 | Mode: FL | Acc: 76.63% | Comm: 215080 bytes
Round 5/10 | Mode: FL | Acc: 76.63% | Comm: 215080 bytes
Round 6/10 | Mode: FL | Acc: 76.57% | Comm: 215080 bytes
Round 7/10 | Mode: FL | Acc: 76.57% | Comm: 215080 bytes
Round 8/10 | Mode: FL | Acc: 76.57% | Comm: 215080 bytes
Round 9/10 | Mode: FL | Acc: 76.57% | Comm: 215080 bytes
Round 10/10 | Mode: FL | Acc: 76.57% | Comm: 215080 bytes
```

```
(venv) PS C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular> python main.py --mode HE

=== Running mode: HE ===
Round 1/10 | Mode: HE | Acc: 76.37% | Comm: 215080 bytes
Round 2/10 | Mode: HE | Acc: 76.60% | Comm: 215080 bytes
Round 3/10 | Mode: HE | Acc: 76.38% | Comm: 215080 bytes
Round 4/10 | Mode: HE | Acc: 76.40% | Comm: 215080 bytes
Round 5/10 | Mode: HE | Acc: 76.35% | Comm: 215080 bytes
Round 6/10 | Mode: HE | Acc: 76.37% | Comm: 215080 bytes
Round 7/10 | Mode: HE | Acc: 76.37% | Comm: 215080 bytes
Round 8/10 | Mode: HE | Acc: 76.37% | Comm: 215080 bytes
Round 9/10 | Mode: HE | Acc: 76.37% | Comm: 215080 bytes
Round 10/10 | Mode: HE | Acc: 76.37% | Comm: 215080 bytes
(venv) PS C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular> python main.py --mode DP+HE
```

```
(venv) PS C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular> python main.py --mode DP+HE

=== Running mode: DP+HE ===
C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular\venv\Lib\site-packages\opacus\privacy_engine.py:96: U
n as it allows for much faster training performance, but remember to turn it on and retrain one last t
  warnings.warn(
C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular\utils\train.py:31: UserWarning: Full backward hook is
inputs require gradients. See https://docs.pytorch.org/docs/main/generated/torch.nn.Module.html#torch.
  loss.backward()
Round 1/10 | Mode: DP+HE | Acc: 75.14% | Comm: 215080 bytes
Round 2/10 | Mode: DP+HE | Acc: 75.92% | Comm: 215080 bytes
Round 3/10 | Mode: DP+HE | Acc: 75.14% | Comm: 215080 bytes
Round 4/10 | Mode: DP+HE | Acc: 75.46% | Comm: 215080 bytes
Round 5/10 | Mode: DP+HE | Acc: 75.23% | Comm: 215080 bytes
Round 6/10 | Mode: DP+HE | Acc: 75.41% | Comm: 215080 bytes
Round 7/10 | Mode: DP+HE | Acc: 75.25% | Comm: 215080 bytes
Round 8/10 | Mode: DP+HE | Acc: 75.40% | Comm: 215080 bytes
Round 9/10 | Mode: DP+HE | Acc: 75.64% | Comm: 215080 bytes
Round 10/10 | Mode: DP+HE | Acc: 75.41% | Comm: 215080 bytes
(venv) PS C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular>
```

```
(venv) PS C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular> python main.py --mode DP

=== Running mode: DP ===
C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular\venv\Lib\site-packages\opacus\privacy_engine.py:96
n as it allows for much faster training performance, but remember to turn it on and retrain one las
  warnings.warn(
C:\Users\User\Videos\FL_DP_Tabular\FL_DP_Tabular\utils\train.py:31: UserWarning: Full backward hook
inputs require gradients. See https://docs.pytorch.org/docs/main/generated/torch.nn.Module.html#tor
  loss.backward()
Round 1/10 | Mode: DP | Acc: 24.12% | Comm: 215080 bytes
Round 2/10 | Mode: DP | Acc: 76.64% | Comm: 215080 bytes
Round 3/10 | Mode: DP | Acc: 24.12% | Comm: 215080 bytes
Round 4/10 | Mode: DP | Acc: 75.91% | Comm: 215080 bytes
Round 5/10 | Mode: DP | Acc: 75.92% | Comm: 215080 bytes
Round 6/10 | Mode: DP | Acc: 76.06% | Comm: 215080 bytes
Round 7/10 | Mode: DP | Acc: 76.41% | Comm: 215080 bytes
Round 8/10 | Mode: DP | Acc: 76.49% | Comm: 215080 bytes
Round 9/10 | Mode: DP | Acc: 76.29% | Comm: 215080 bytes
Round 10/10 | Mode: DP | Acc: 76.37% | Comm: 215080 bytes
```

```
Federated Learning Framework v2.0                                    2025-09-15 02:21:58

(1) Overview | (2) Model Manager | (3) Client Health | (4) Modules | (5) Logs | (6) TUI Details

Modules

Attack Defense And Resillence Module | Privacy Preserving Module | Secure Aggrigation Module | Server Communication And Protocol Enforcement Module

Privacy Preserving Module Details
Module Name:                    PPM
Status:                         active
Epsilon:                        1.0
Delta:                          1e-05
He Active:                      True
Description:                    Privacy-Preserving Mechanism for policy auditing.
```

Secure Aggregation

IT21831904 | Weerasinghe K.M

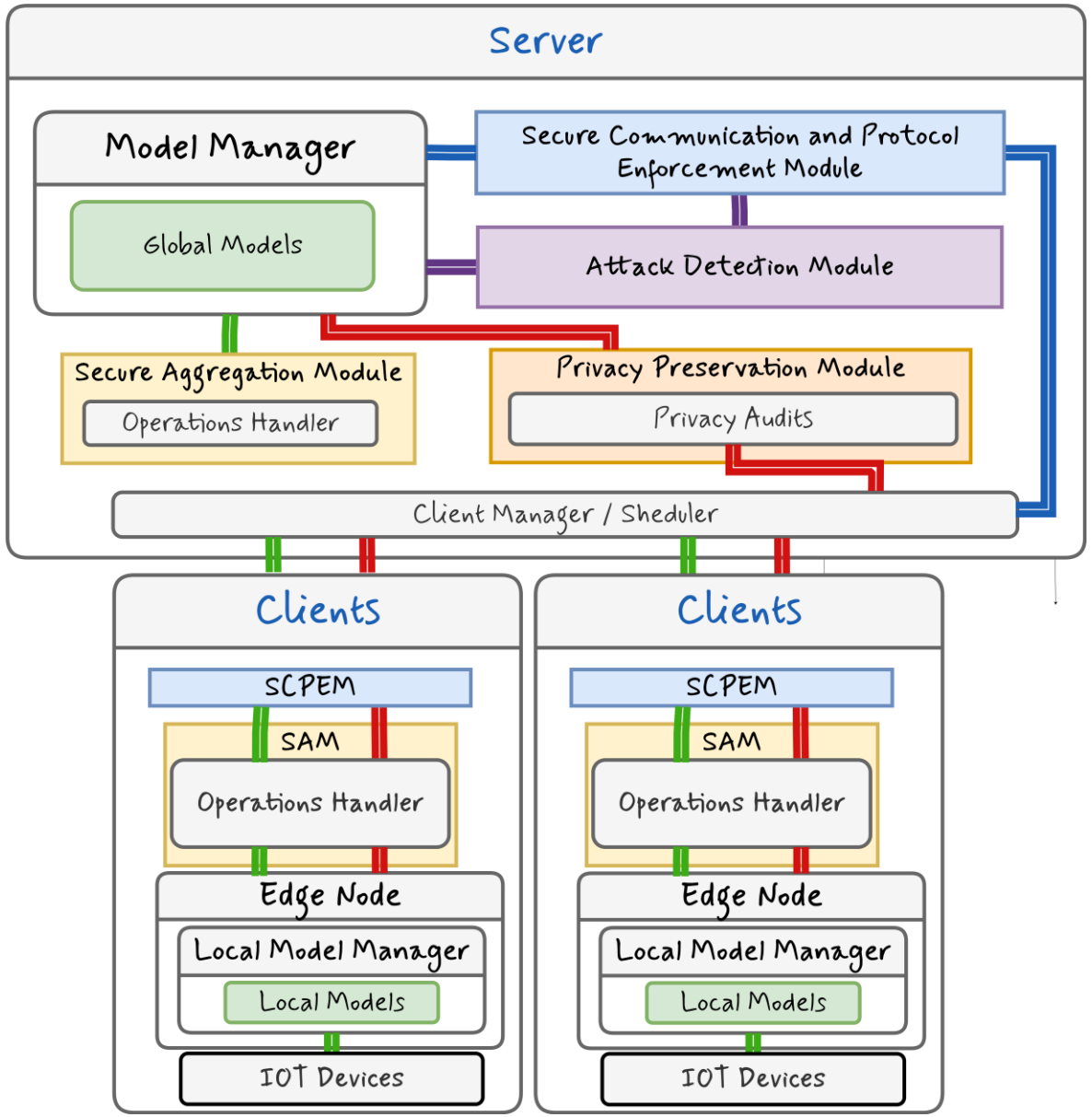
- **BACKGROUND**
  - **Main Function:** Secure aggregation ensures the server can compute an aggregate of client Updates without seeing any individual Private contribution
  - **Requirement**: It protects client data privacy during aggregation, even if the central server is curious or malicious
- **RESEARCH GAP**
  - **Limitations**: Many Secure aggregation protocols are either too heavy for resource constrained IIoT devices and some only exist in theory, and do not account for diverse IIoT environments
  - **Challenges**: They often lack robustness to network disruptions or client failures, which can cause and entire training round to collapse
- **RESEARCH PROBLEM**
  - How to design a secure Aggregation protocol that is both adaptive and robust enough for scalable IIoT deployments.

SOLUTION:

- **Main Objective:** To aggregate client updates securely, ensuring the server learns only the final sum, not individual contributions, while being resilient to client dropouts.
- **Our approach:**
  - **Shamir's Secret Sharing (SSS):** Each client's model update is split into multiple shares. No single share reveals any information.
  - **Threshold Cryptography:** The original secret (the aggregated sum) can only be reconstructed if a minimum number of shares (the "threshold") are combined.
- **Benefits:**
  - **server Blindness:** The server cannot reconstruct any individual update.
  - **Fault Tolerance:** The aggregation succeeds even if some client failures

Secure Aggregation Architechture

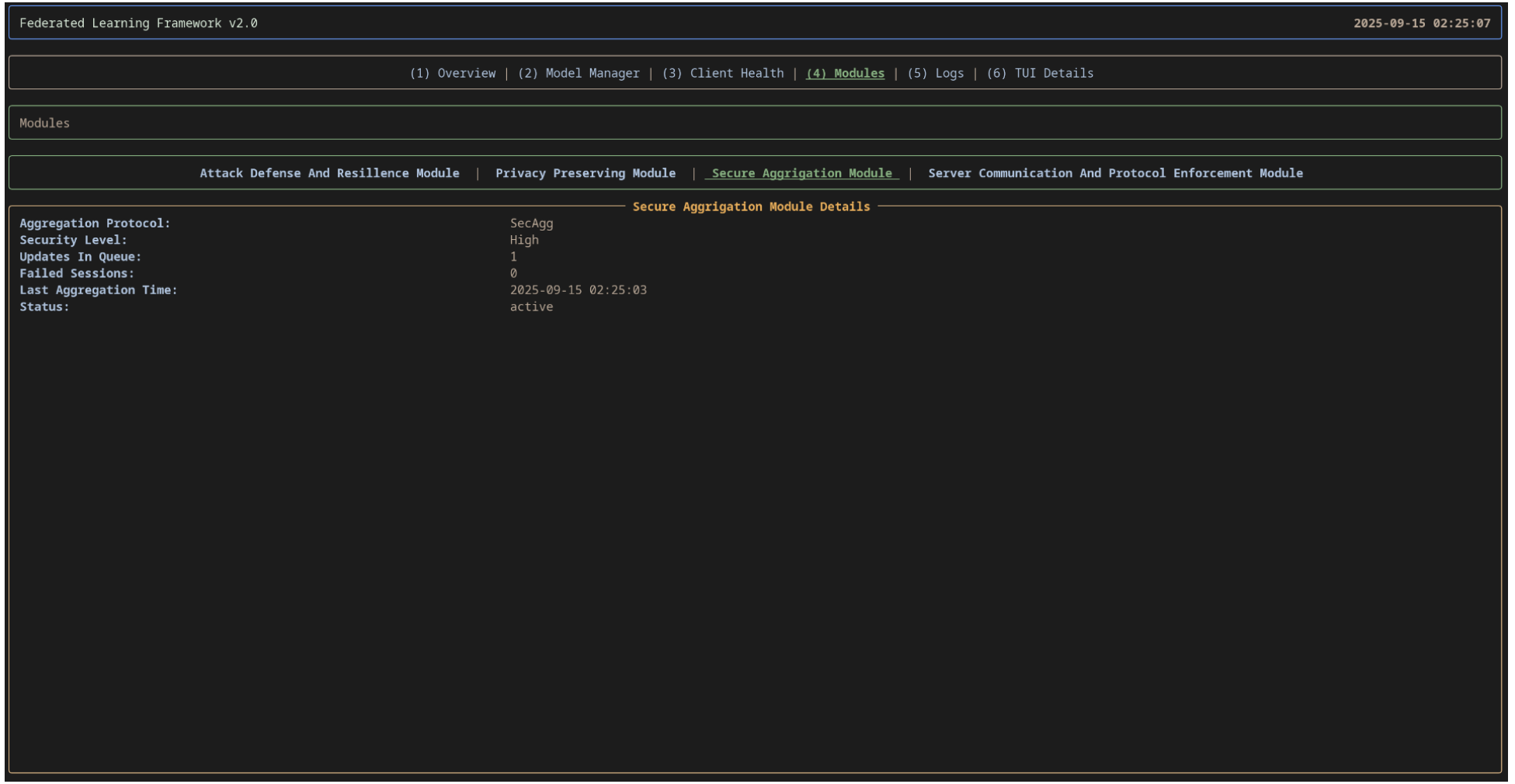
- **Client-Side Operations Handler:** Splits the local model update into encrypted shares using SSS.
- **Server-Side Operations Handler:** Receives and stores shares from all clients
- **Threshold Reconstructor:** A server function that combines shares to reconstruct the final aggregated model update once the required threshold is reached.



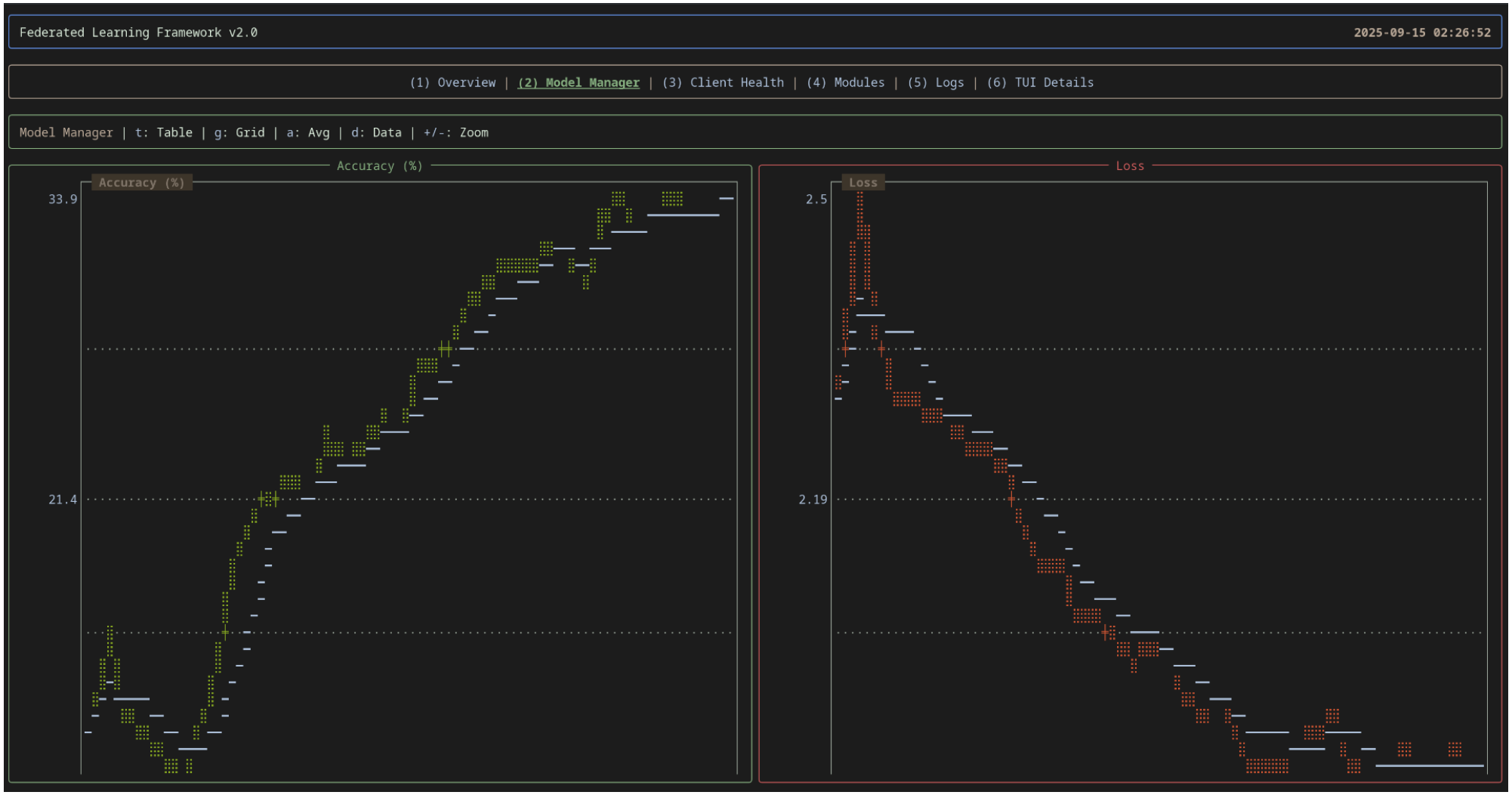
SECURE AGGREGATION PROCESS FLOW

- Each client splits its update into n no of shares.
- Shares are distributed securely to the server.
- The server aggregates the corresponding shares from all clients.
- Once at least t clients have submitted, the server reconstructs the final aggregated update.

SAM MODULE TUI INTERFACE



AGGREGATION PROCESS



Secure Communication and Protocol Enforcement

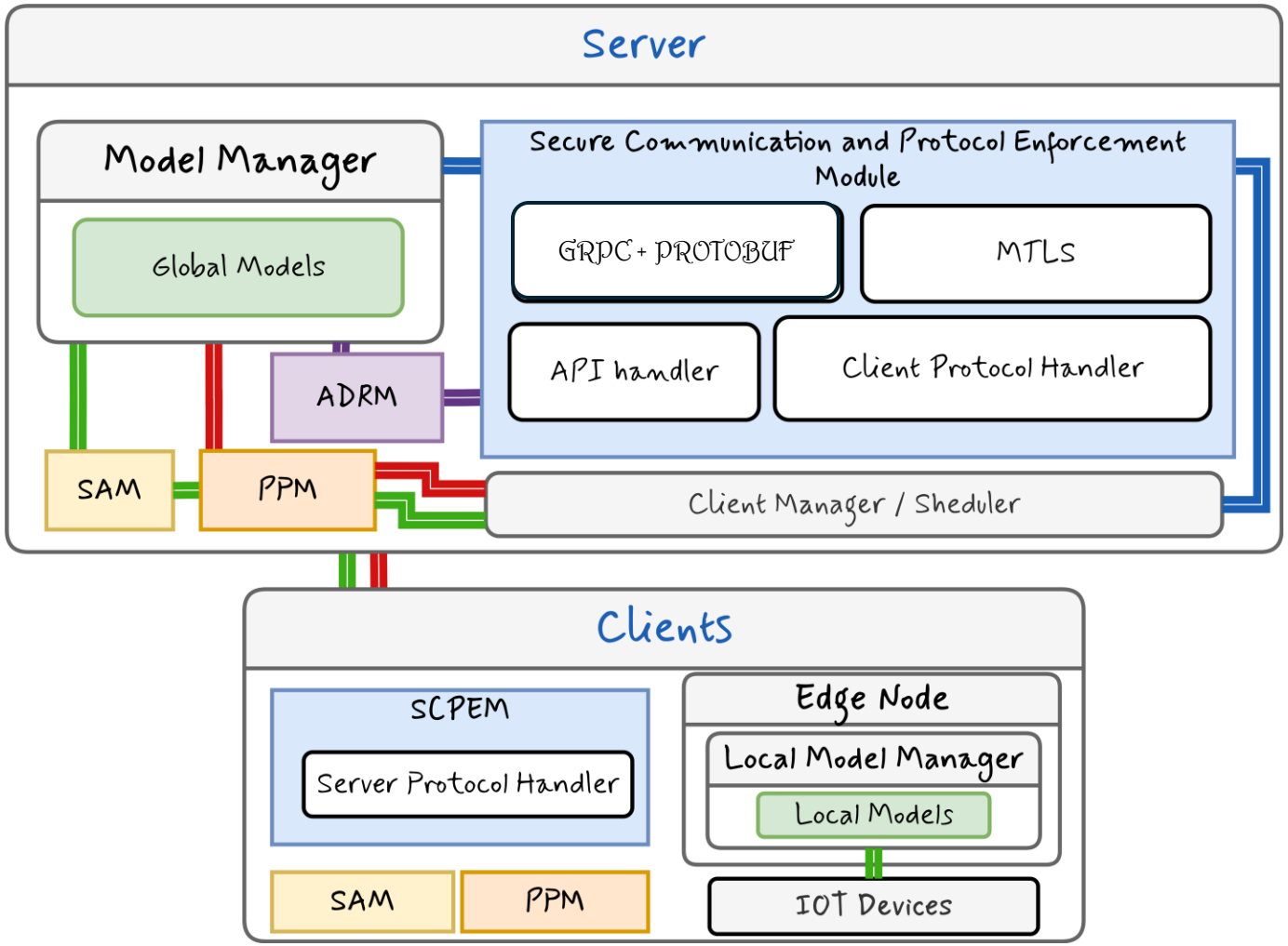
IT21828348 | Dissanayaka KDARA

- **BACKGROUND**
  - In IIoT, communication between clients and server is a prime target for threats like eavesdropping, tampering, and MITM attacks. Standard FL focused on efficiency but lacked strong defenses. TLS provides a baseline, but its one-sided authentication leaves gaps.
- **RESEARCH GAP**
  - Current solutions lack an integrated security framework for IIoT. Standard TLS often validates only the server, leaving no assurance that connected devices are legitimate. A stricter, two-way validation protocol is needed.
- **RESEARCH PROBLEM**
  - Insecure FL channels risk unauthorized access, tampered updates, fake servers, and undetected data changes.

SOLUTION:

- Implement a secure communication protocol that ensures only authorized IIoT devices can participate, protects model updates from tampering, verifies the legitimacy of the server, and adds integrity checks beyond encryption to guarantee unaltered data.

Module Architecture



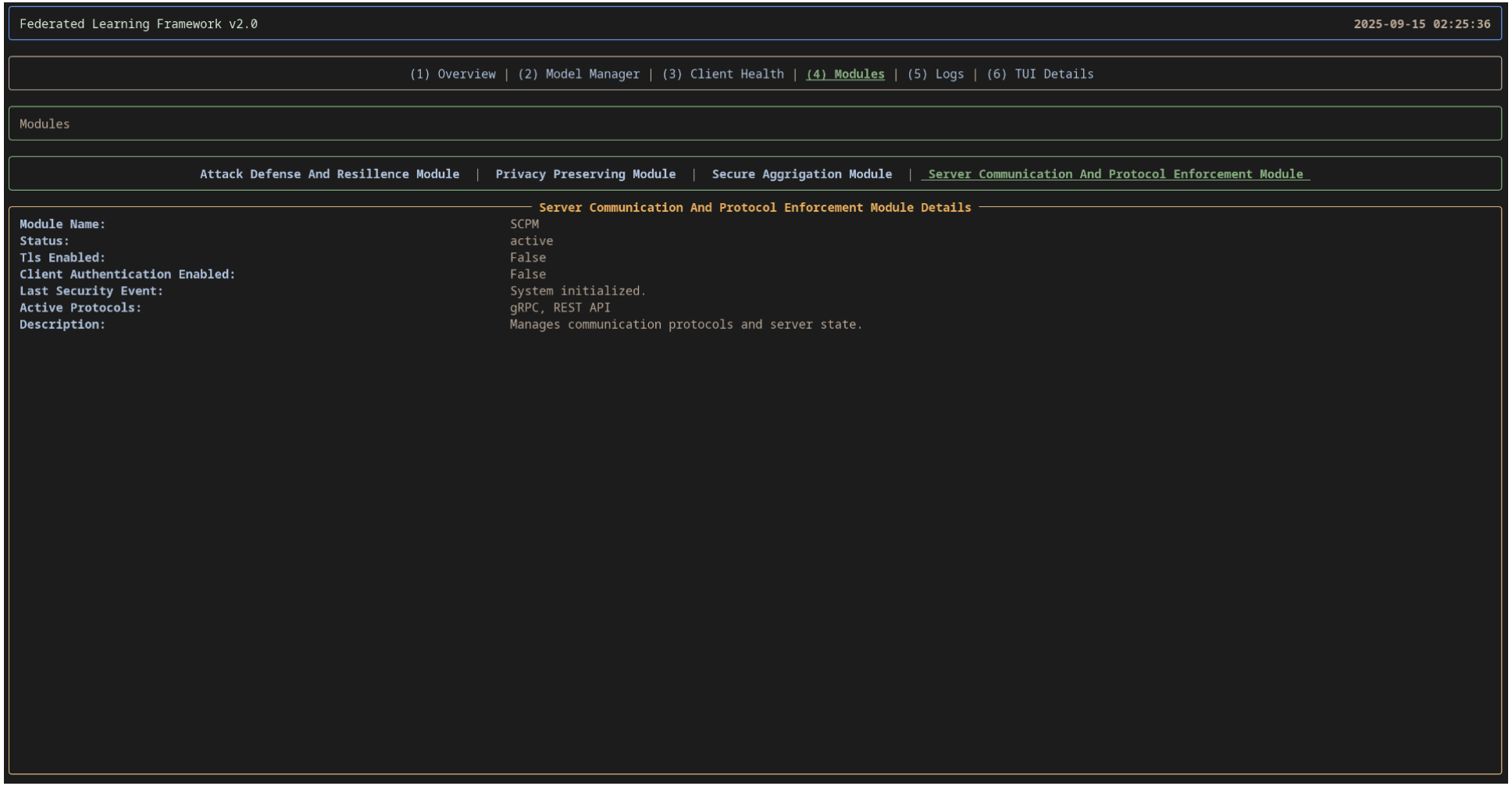
SERVER COMPONENTS

- **Model Manager:** Distributes and updates the global model.
- **SCPEM:** Secure channel using mTLS (two-way authentication) + HMAC for integrity.
- **Client Manager/Scheduler:** Selects and manages participating clients.

OVERALL WORKFLOW

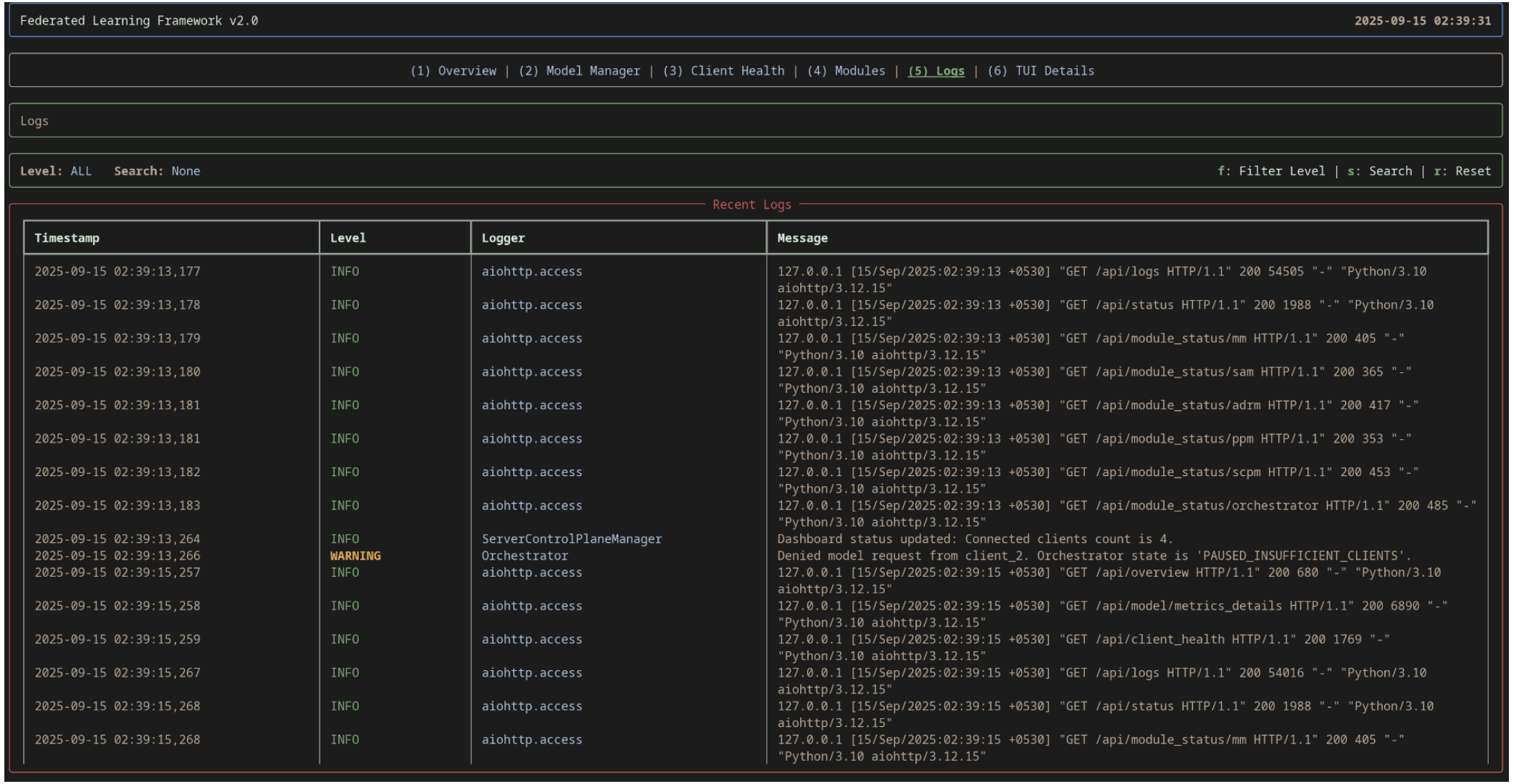
- Server selects clients for training.
- Global model sent via **SCPEM** (secure channel).
- Clients train locally on IoT data.
- Updates protected using **PPM & SAM**.
- Protected updates sent back via **SCPM**.
- Server validates updates with **ADRM**.
- **SAM** aggregates updates securely.
- **Model Manager** updates global model.
- Cycle repeats.

SCPM Architecture



Img 1: SCPM TUI frontned

Log manager



Img 2 : logging and monitoring of overall system

TECHNOLOGIES

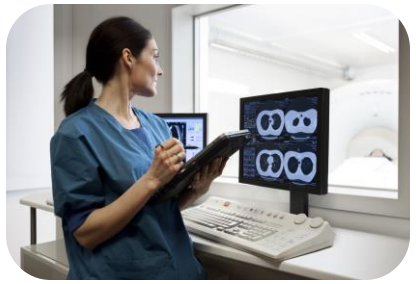
- BACKEND
  - Python
  - Asyncio
- COMMUNICATION
  - GRPC
  - Protocol buffers
  - AIOHTTP
  - AIOHTTP-CORS
- ML & DATA PROCESSING
  - PyTorch
  - Torch Vision
  - NumPy
- CRYPTOGRAPHY & SECURITY
  - PyCryptodrome
  - Secret-sharing
  - X.509 Certificates
- FRONTEND & VALIDATION
  - Rich
  - Prompt Toolkit
- LOGGING AND MONITORING
  - Python-JSON-Logger
  - Logstash

Commercialization

Target Industries

Manufacturing

Energy



Healthcare

Automobile

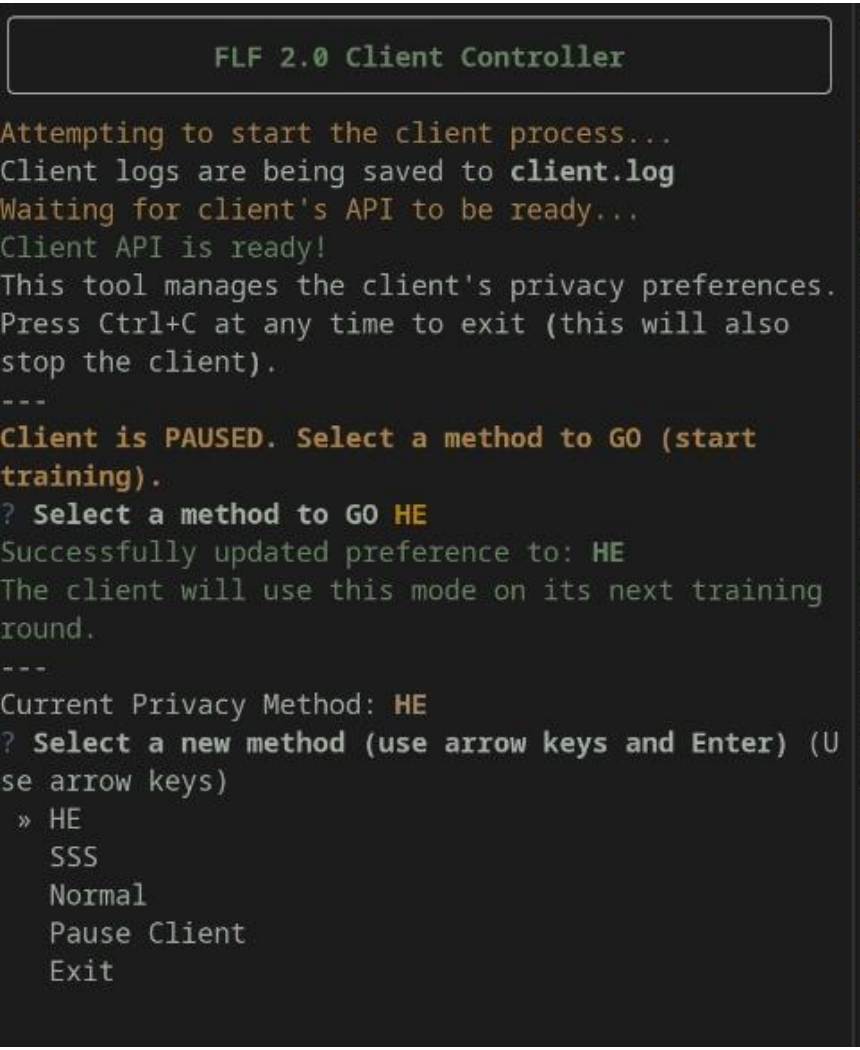
- Our product includes **4 modular components** that can be easily integrated into **any Federated Learning system** both **existing and new** to enhance **data privacy**.

Business Models

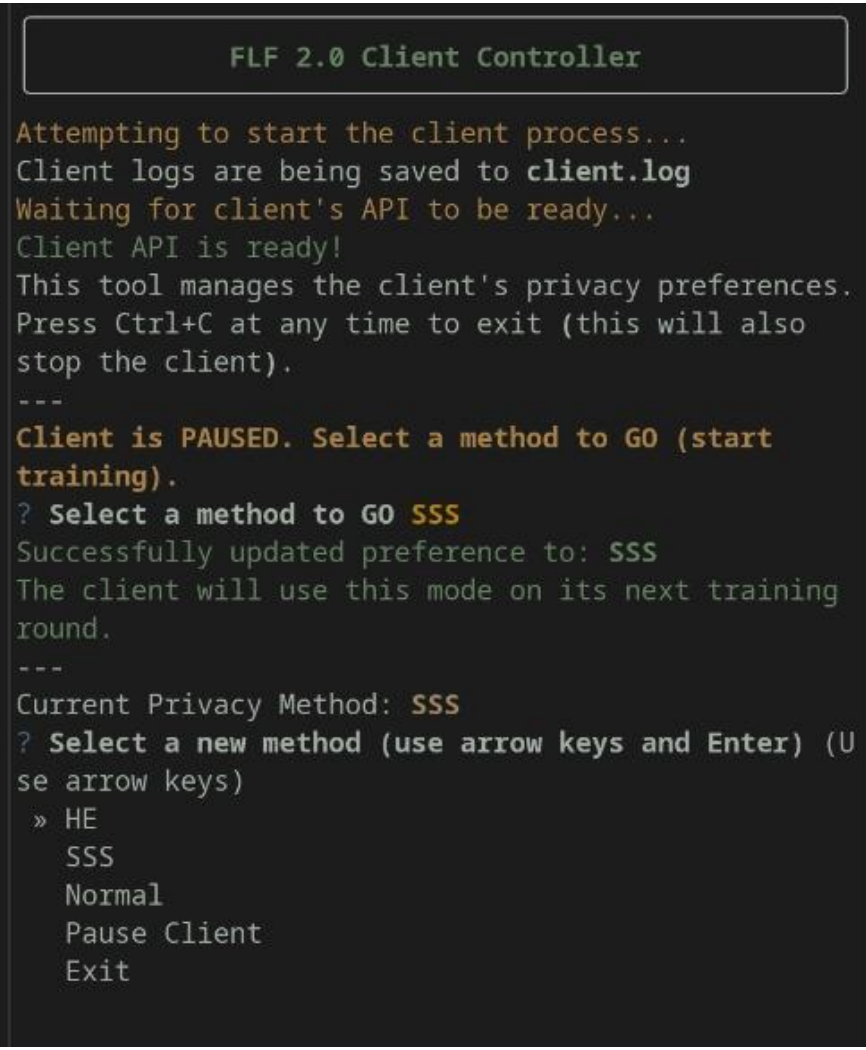
As a PaaS Model or Enterprise Solutions.

Demonstration

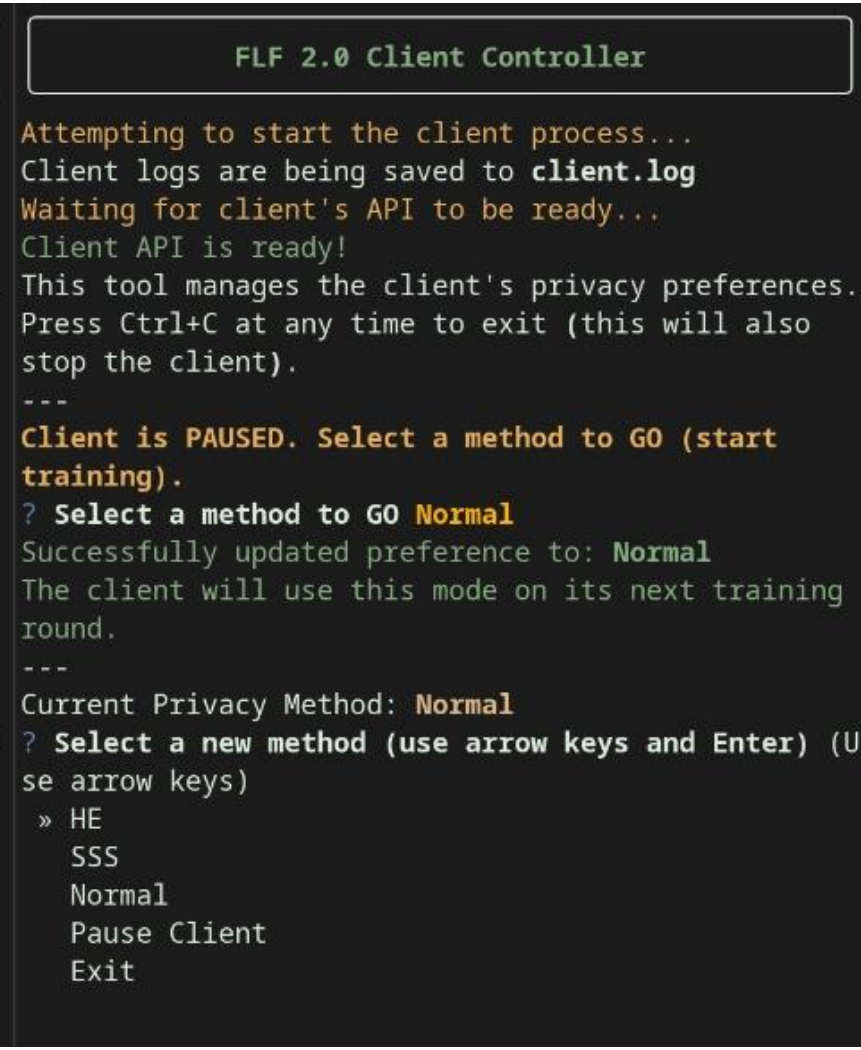
Clients – Running Methods



Method 2 – HE + DP method

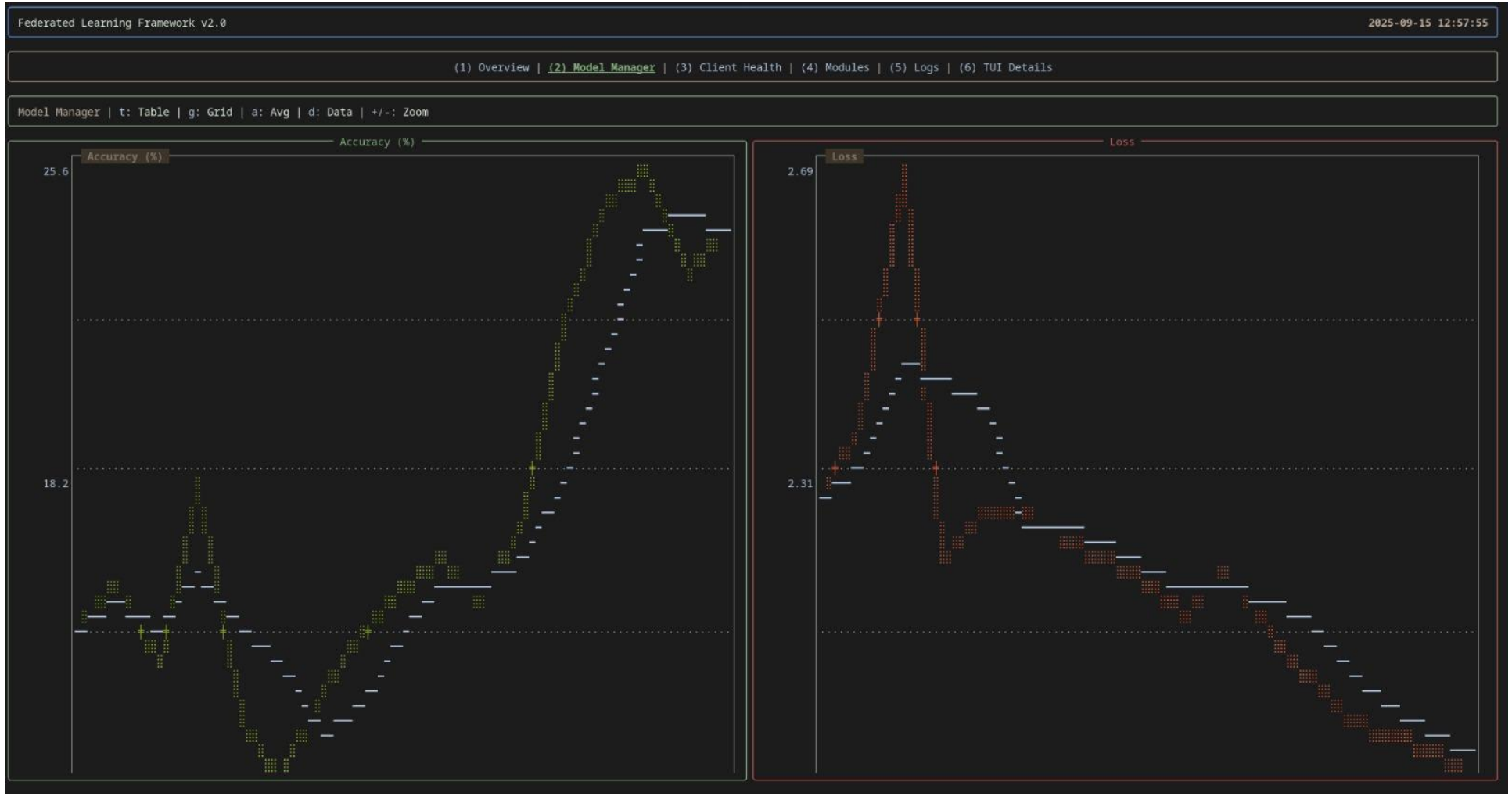


Method 3 – SSS+TH method

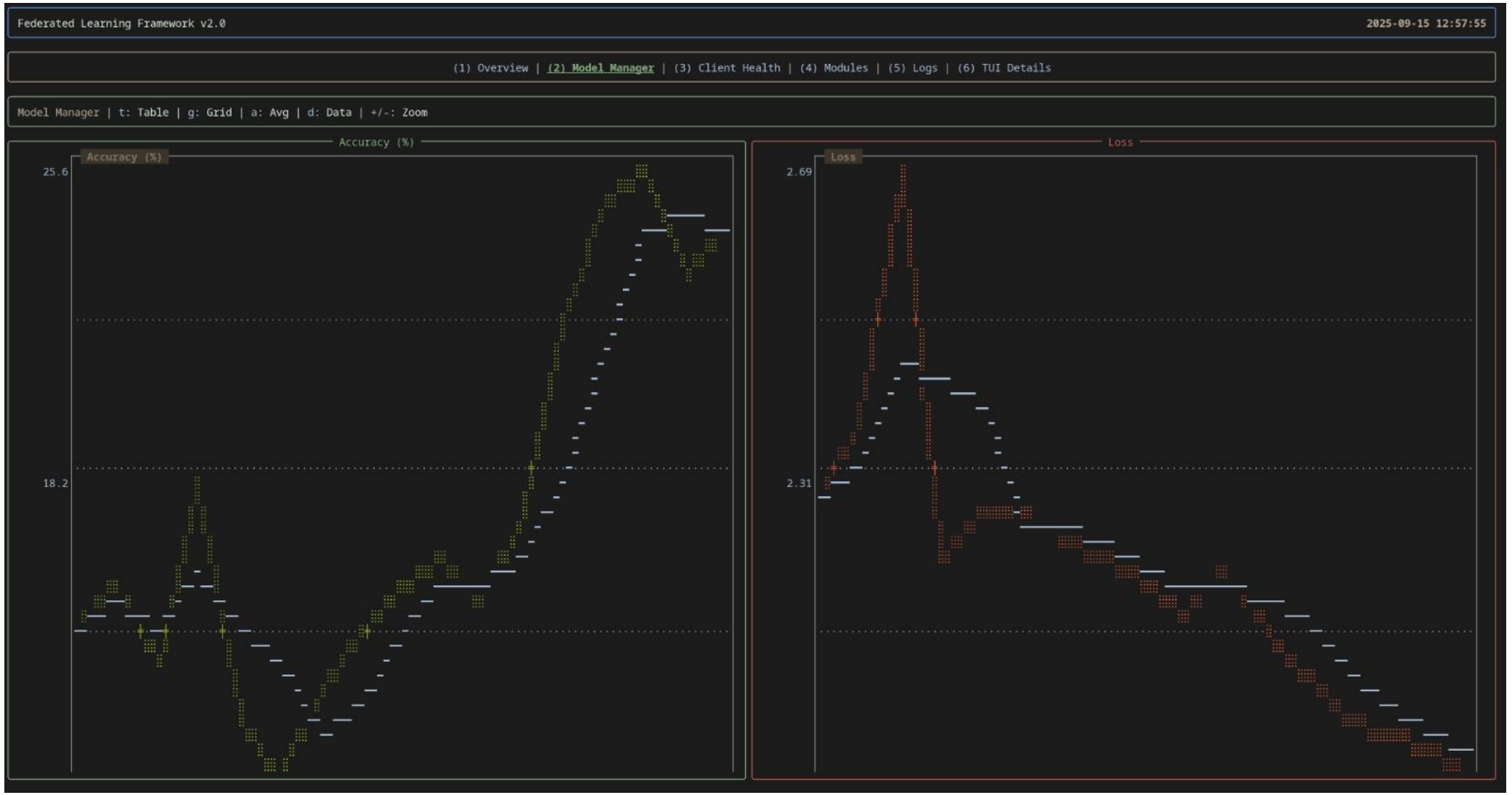


Method 1 – Traditional method

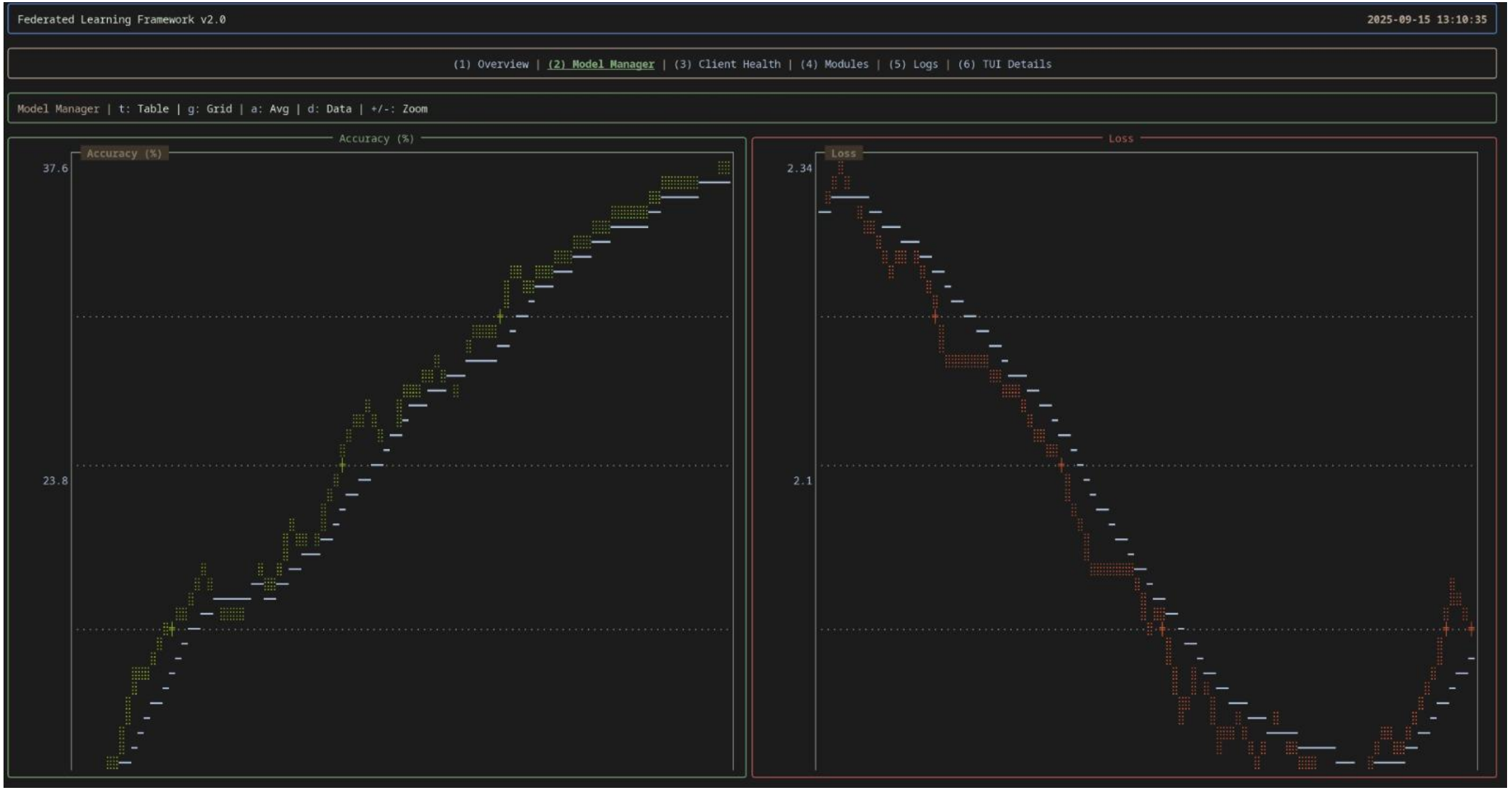
Method 1 – Normal FL method



Method 2 – HE + DP FL method



Method 3 – SSS + TC FL method



THE END

Thank You