Federated Learning-Based Intrusion Detection System with Device-Specific Modeling for IoT Networks

CS 549 : Computer and Network Security

Anubhab Dutta 220101119
Aditya Kumar 220101005

01

Introduction

IoT Growth and Security Challenges

- IoT has expanded rapidly, connecting billions of smart devices.
- These devices generate and process massive amounts of sensitive data.
- Limited computational resources make them vulnerable to attacks.
- Common threats: botnets, DDoS, and protocol-specific exploits.
- Machine Learning can help detect intrusions by recognizing attack patterns.

Limitations of ML-based IDS

- Centralized ML-based IDS require sharing raw data → privacy concerns.
- IoT devices vary in computational power → one-size models don't fit all.
- Static models struggle to adapt to new or evolving threats.
- Centralized processing causes communication bottlenecks in large networks.
- FLIDS (Federated Learning-based IDS) addresses these issues using local training and knowledge sharing

Prior Work

- Several studies have explored Federated Learning (FL) for IoT intrusion detection..
- Works like Bimal & Rawat, Campos et al., and Abdul Rahman et al. analyzed FL for IoT security and compared deployment strategies.
- Research has applied deep learning models (CNN, LSTM, GRU, Autoencoders) in FL-based IDS to enhance detection accuracy.
- These models are too heavy to be efficiently deployed in Low power IoT devices
- Assumed homogeneous models across all IoT devices.

Identified Research Gaps

- Existing FL-IDS approaches rarely support heterogeneous model architectures across devices.
- Knowledge transfer between different model classes remains largely unexplored.
- Many systems focus on accuracy but overlook scalability for resource-limited IoT devices.
- Our work fills these gaps with heterogeneous model support and teacher–student knowledge distillation

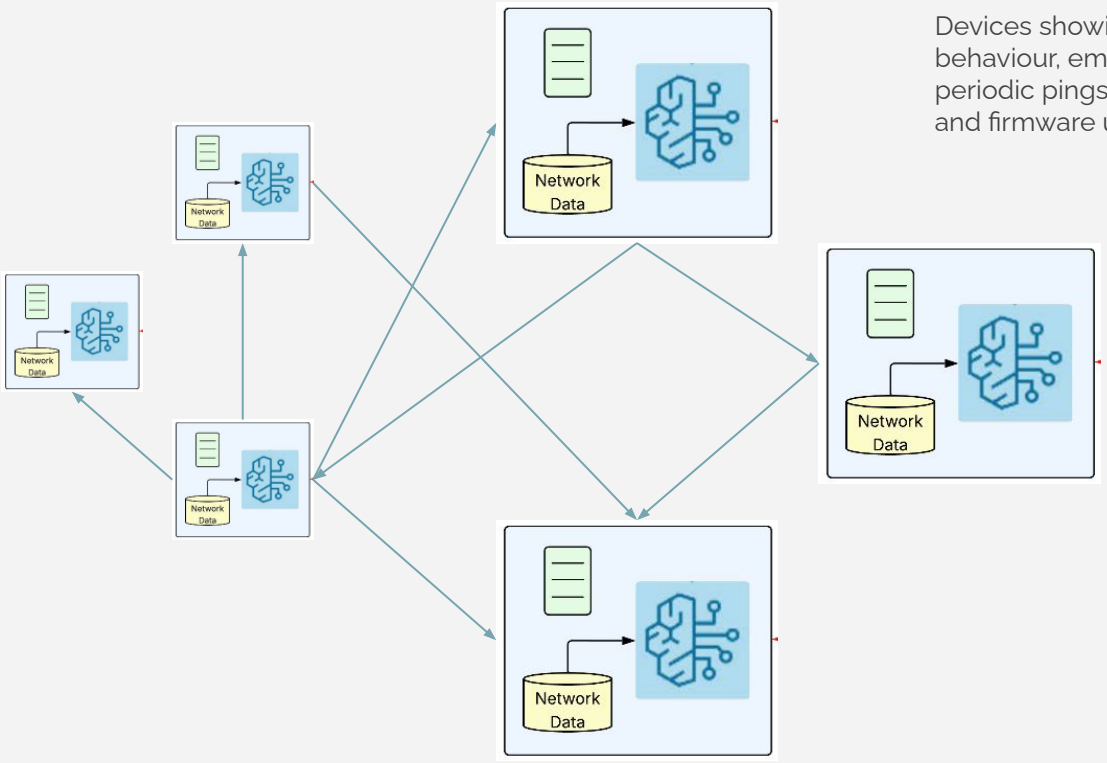
02

Architecture

Experiment Design

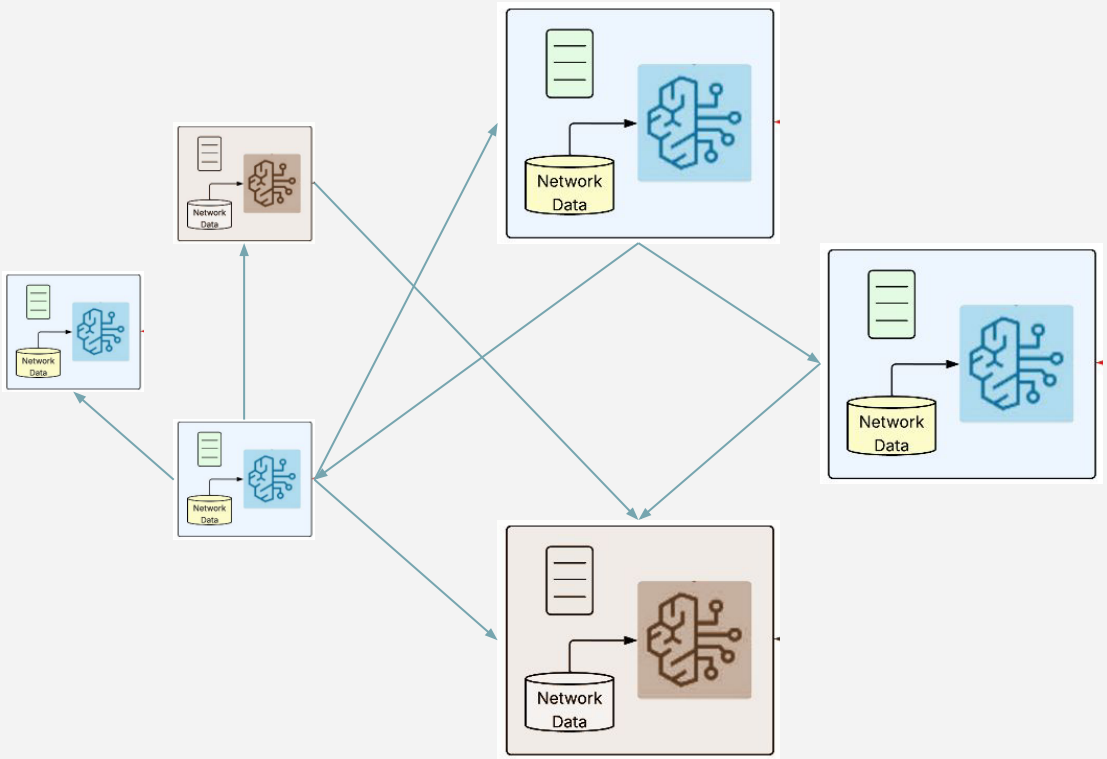
- Simulation using six IoT devices.
- Divided into two groups of three devices, one running the lightweight models, and another running the heavyweight.
- Simulates normal traffic, attack scenarios (e.g., SYN flooding, bursty malware), and queue management for packet analysis.
- Uses SimPy to imitate parallel devices running on a network, with transmission delays and queue management, and collecting data from the packets captured.

Network of IOT devices



Devices showing normal behaviour, emulating periodic pings, streaming and firmware updates.

Attack State



Two devices got infected and are running a BotNet attack, flooding the network with packets

Data Collection & Processing

- Simulation runs for 1000 seconds, producing 6 device logs capturing network activity.
- Each device records packet-level and time-aggregated traffic data.

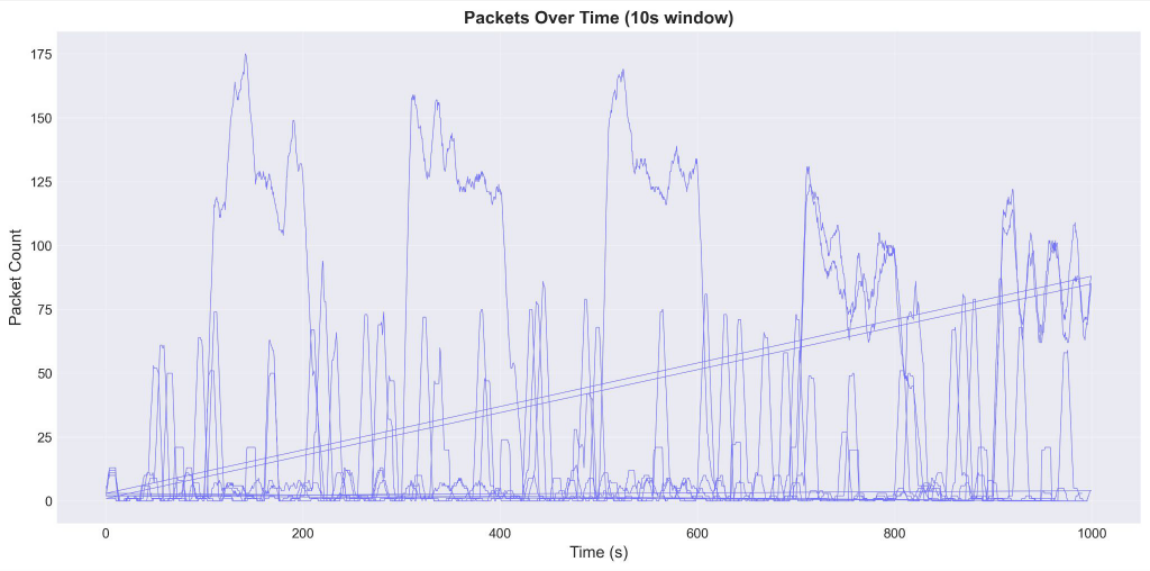
Packet level Features:

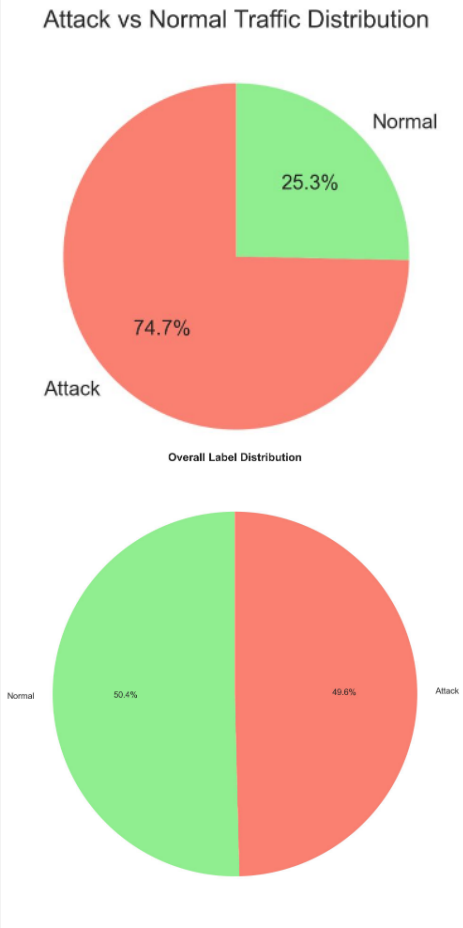
- Timestamp, src/dest, protocol type (TCP/UDP)
- Packet size and queue utilization
- Attack indicator
- Protocol flags (SYN, ACK etc.)

Time-Windowed Aggregation

- Aggregated every 0.5s, over 5s, 15s, 30s windows.
- Metrics: packets sent/received, ang and variance of packet size, Counts per protocol, queue occupancy stats.

Data Analysis





Number of packets in the network increase dramatically when under attack

Local Training & Federation

- **Heterogeneous Architecture:** Supports two model families for diverse IoT devices.
  - Lightweight: Logistic Regression, SVM
  - Heavyweight: Multilayer Perceptron
- **Federated Aggregation:** A central server aggregates models from each family separately using the FedAvg algorithm

$$w_{global} = \sum_{k=1}^{K} \frac{n_k}{n} w_k$$

where w(k) are model parameters of device k and n(k) are number of samples at device k.

Knowledge Transfer & Model Update

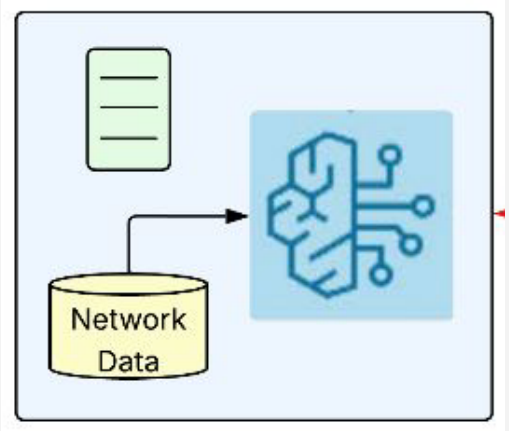
- **Bidirectional Knowledge Transfer**: Enables lightweight and heavyweight models to learn from each other.
- **Teacher-Student Distillation**: Uses a shared global dataset to transfer knowledge.
- Unlike standard Federated learning, each local model is updated using the global models using

$$w_{local}^{new} = 0.7 \times w_{local}^{old} + 0.3 \times w_{global}$$

- Trained models are sent back to devices to perform real-time intrusion detection.

In-Device model

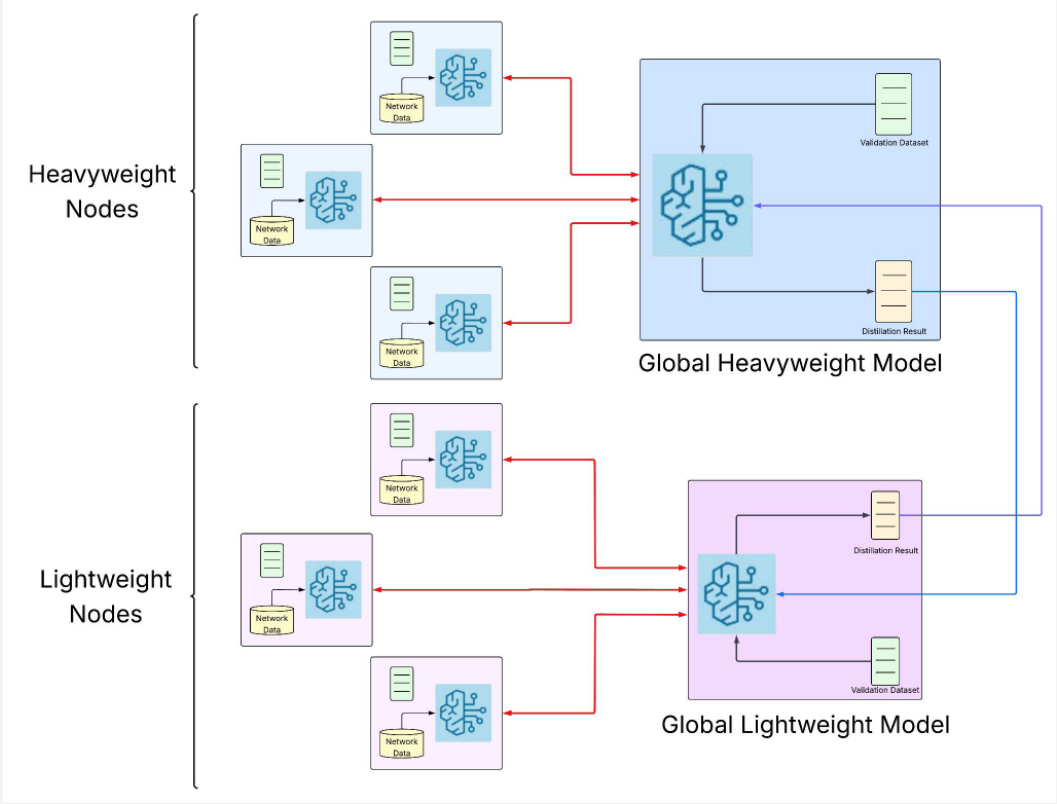
Tested on a separate dataset after each training round



Runs either a lightweight model or a heavyweight model.

Recently captured data is used for training the model

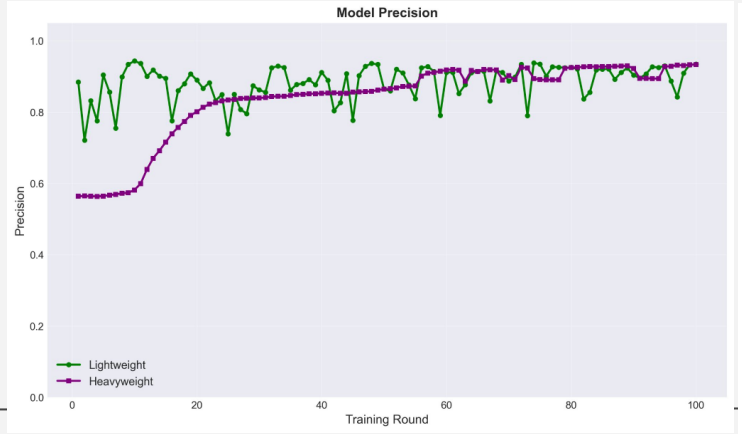
Model Architecture



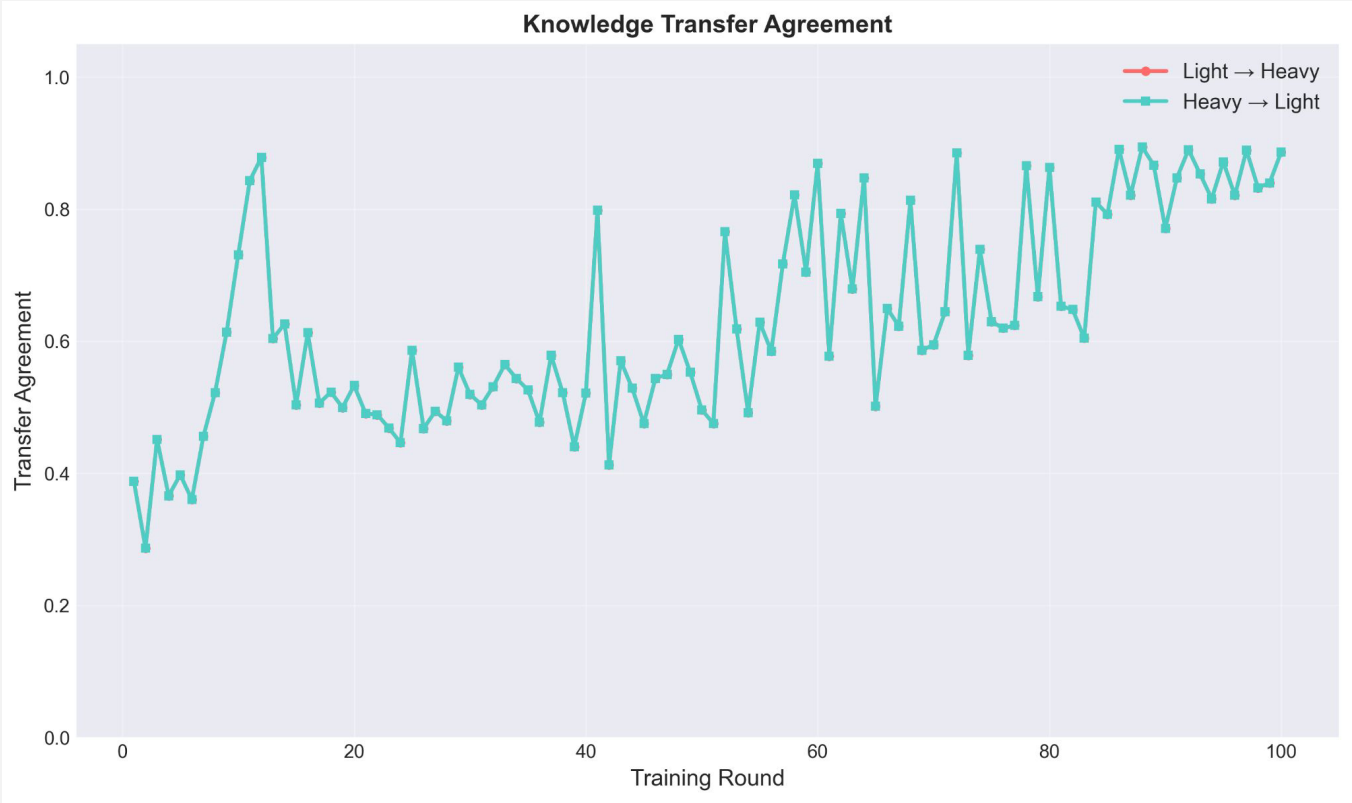
03

Results

Training Metrics



Knowledge Transfer



Thanks!