Demo: P4 Based In-network ML with Federated Learning to Secure and Slice IoT Networks

Chamara Madarasingha*, Thilini Dahanayaka†, Kanchana Thilakarathna†, Suranga Seneviratne†, Young Choon Lee‡, Salil S Kanhere*, Albert Y. Zomaya†, Aruna Seneviratne*, and Phil Ridley§

*University of New South Wales, Sydney, Australia
†The University of Sydney, Sydney, Australia
‡University of Macquarie, Sydney, Australia
§IoT Factory, Sydney, Australia

***Abstract*—Recent cyberattacks have increasingly targeted distributed networking environments like IoT networks. To detect these attacks, hidden under network traffic encryption, many centralized Machine Learning (ML) based solutions have been introduced, which are not well suited for IoT networks. This work proposes PIFL a practical approach to secure IoT networks by combining federated learning, in-network ML using P4-enabled devices, software-defined networks, and binarized neural networks. PIFL detects compromised edge devices and isolates them into separate network slices based on trust parameters derived from their behavior. We demonstrate the feasibility of PIFL using an experimental testbed with three intelligent network devices and seven IoT devices implemented on Raspberry Pi devices.**

I. Introduction

With the advances in network systems, including Internet-of-Things (IoT), various network attacks (e.g., DDoS, Sybil etc.) have also emerged. With end-to-end encryption of network traffic, detecting such malicious behavior has become more challenging. While numerous machine learning (ML)-based solutions have been proposed for encrypted traffic analysis, most focus on centralized systems, rendering them less suited to IoT networks [2].

Federated learning (FL) is a decentralized machine learning paradigm where model training occurs locally on edge nodes, with only model weights aggregated and shared across the network. FL addresses the lack of training data at individual nodes through weight sharing and offers enhanced privacy by eliminating the need for data transmission. However, in IoT environments where edge devices are resource-constrained, conducting computationally and memory-intensive training of ML models is challenging.

Binarized Neural Networks (BNNs) are a variant of neural networks that constrain weights to binary values which simplifies operations to bitwise operations, enabling efficient implementation on hardware platforms with reduced memory and computational requirements. For instance, the utilization of hardware-friendly BNNs with P4[1] enabled routing devices can create an intelligent edge that can train and deploy ML models for IoT networks.

Hence, we propose PIFL[2] which aims to develop ML solutions tailored for IoT networks to detect malicious edge devices and dynamically isolate them into separate network slices based on trust parameters derived based on their behavior. By assigning devices with less trust to a separate network slice rather than simply quarantining, the less trusted devices can operate in the network without affecting devices with high trust. Also, data collected at such devices can still be used after being weighted with the trust level of the device.

While previous work [4], [5] explored FL/ML and BNNs to detect anomalous devices and P4 for in-network ML, they used previously collected datasets instead of using real-time data and lacked an end-to-end setup that can leverage true benefits of SDN such as network slicing. To the best of our knowledge, we are the first to explore the practicality of leveraging FL and BNNs with P4-enabled routing at resource-constrained devices (e.g. Raspberry Pi (RPi) mini-computers) to build end-to-end systems to deploy ML models for anomaly detection in IoT networks. More specifically, this work presents a prototype platform that encompasses, training BNNs on real RPi devices with FL, real-time, in-network inference, and agile trust level assignment and network slicing.

II. PIFL System design

PIFL consists of two key components: *i*) Binarized FL which utilizes BNNs with FL, *ii*) Agile slicing which segments the network based on the IoT device trust level. In both training and inference phases, network architecture comprises of IoT sensor networks (as in Local Area Networks: LANs), L2/L3 switches and related servers. We convert RPi 4 devices to an L2/L3 switch using P4Pi library [3], which is built upon the P4 programming language specifically designed for network packet handling [1]. Each switch has *Data plane* where the network packets are handled and routing tables are deployed and *Control plane* which control the *Data plane* behaviour through `P4-runtime controller`.[3] Each switch provides a WiFi access point (AP) to a LAN, enabling communication with the IoT server on the internet.

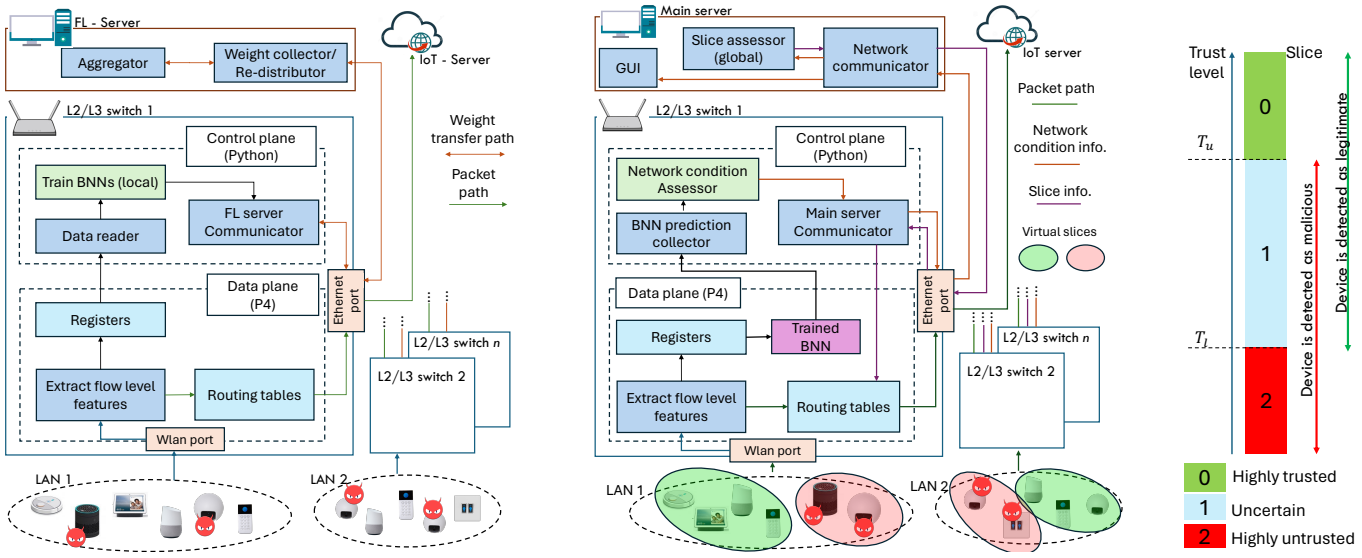
A. Binarized FL: Training Phase

Fig. 1a shows the FL training process. At the *Data plane* of the P4 switch, we first extract flow-level features and

[1]Programming Protocol Independent Packet Processors

[2]**P**4 Based **I**n-Network ML with **F**ederated **L**earning to Secure and Slice IoT Networks

[3]https://p4.org/p4-spec/p4runtime/main/P4Runtime-Spec.html



(a) FL training phase (b) FL inference phase (c) Assigning slice

Fig. 1. Overall system architecture of PIFL for training and inference phases and the mechanism for slice assigning in the inference phase.

store each feature data in a register dedicated to that feature. Registers have a predefined number of slots where each slot will be allotted to a identified flow in the LAN. Note that each IoT device creates one major network flow that can be used to uniquely represent device behaviour. These registers are periodically read by the *Data Reader* in the *Control plane* to train the BNNs locally at the switch. Before the local training, the FL-server shares the number of global FL rounds (R_f) with each P4-switch connected to FL-server.

In BNN training backward propagation, local weights (W_f^l) are represented by floating point arithmetic, where $W_f^l \in [-1, 1]$. During forward pass, W_f^l is converted to its binary representation W_b^l, where $W_b^l \in \{-1, 1\}$. Once the local training is completed after a pre-defined number of epochs, W_f^l is shared with the FL-server. *Weight Collector* in FL-server keep waiting until it receives W_f^l from all the connected switches in each FL round, R. Then, *Aggregator* takes the average of them generating the global weights, W_f^g, which is shared with the P4 switches by *Re-distributor*. Once the $R = R_f$, we stop the FL-training process and deploy the corresponding global binary weights, W_b^g, in the BNNs in the inference phase. We pre-configure routing tables to transfer all the traffic from IoT sensors to IoT servers on the internet.

B. Binarized FL: Inference phase

During the inference phase, we follow the same feature extraction method in the training phase. We also develop BNNs at *Data plane* leveraging P4 language [5] and deploy W_b^g from the training phase. For each flow, we provide BNN prediction every T seconds and share the prediction information with the *Control plane* via *BNN prediction collector*.

At the *Control plane*, we run *Network Condition Assessor*, which generates three measures. *i*) Average prediction: averaging the prediction across w_s samples for each flow. *ii*) Trust level (TL): based on average prediction, the model assesses the TL (between 0 – 100%) of a given device. Here, if an IoT device is predicted as malicious, we reduce the trust by α factor and increase the trust by $1 - \alpha$ factor otherwise. By setting α at higher value, we drastically reduce the TL if a device is detected as malicious and slowly build the TL if the device exhibits legitimate behavior again. *iii*) Slice: based on the TL of IoT device, we assign them to three main network slices, as in Fig. 1c, where the communication between devices between different slices are restricted. According to Fig. 1c, if a device is predicted to be malicious and the trust level is less than a predefined threshold, T_l, we assign the slice to be *Highly malicious*. With the same prediction, if the trust level is above T_l, we assign the device to the *Uncertain* slice. We follow a similar approach for benign devices considering TL threshold, T_u.

C. Agile slicing of network

Network communicator collects the above data (i.e., Average prediction, TL, Slice, and IPv4 addresses of the IoT devices) from each switch. The collected *Slice* information is used to decide which devices can communicate with each other both at intra-LAN and inter-LAN levels, allowing virtual slicing of the entire network. Here, we share the IPv4 address pairs of the IoT device found from different slices that cannot communicate with each other and update routing tables accordingly. Also, if a device is in the highly untrusted slice, we provide restricted access by blocking the device from the IoT server for a pre-defined period and unblock it after that.

III. Demonstration

A. Experiment Setup

Our demonstration setup comprises three P4 switches and seven IoT devices and is arranged into three LANs. IoT devices in the network are built on RPi 0 devices, which

can be connected to different types of sensor devices (e.g., temperature, motion, etc.) and periodically send data to a dedicated IoT server at a rate of 80Kbps. In this demonstration, we set a malicious device to send out bursts at rates over 160Kbps by changing the data reporting interval and amount of data sent. Note that since we do not control the bandwidth of the entire network, both malicious and legitimate devices can show the same network traffic behavior in certain periods. Therefore, a heuristic approach such as simple thresholding is not applicable to detect anomaly behavior accurately, requiring a sophisticated ML model. We empirically set $R_f = 5$, $T = 4s$, $T_l = 25\%$ and $T_u = 75\%$.

B. Demonstration

1) Dashboard for visualization: During inference, PIFL consists of an interactive dashboard that can be used by an operator to get a detailed view of a device they are interested in. The dashboard is built using two open-source tools, Prometheus[4] which keeps track of time series information collected by *Network communicator* and Grafana[5], which visualizes those details under three levels. Those are, **i) System level overview**: System level details, including the number of devices connected, the history of slice distribution of devices, traffic behaviour of devices, and alerts for devices detected as malicious. **ii) P4 switch level overview**: Overview of all IoT devices connected to the selected P4 switch, including traffic behavior, slice history, and prediction history. **iii) IoT device level overview**: Details of a selected IoT device, such as traffic behavior from the device and the slice, prediction, and trust level history (see Fig. 2b).

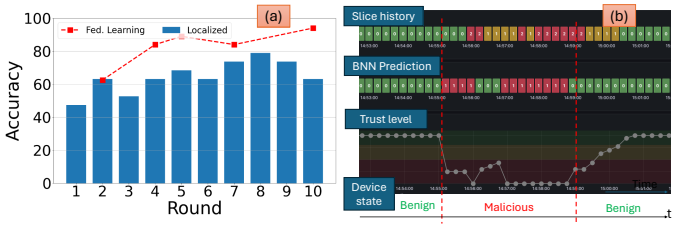


Fig. 2. (a) Accuracy gain achieved by FL based training and (b) extracted IoT device level overview of GUI

2) Evalution scenarios: We demonstrate the PIFL functionality using three different scenarios.

Scenario 1: FL training with BNNs In this scenario, training a BNN via FL following steps in Section II-A. The main features we extract are *Throughput uplink/downlink* and *interpacket gap*. During the training phase, at least one device in each LAN is configured to be malicious and the operator can interact with the process by selecting the sensor devices to be malicious. During the process, FL server will display global FL accuracy in each round. Fig. 2a shows experimental results comparing the convergence of the model for the federated setting vs a localized training setting. We observe that the model trained with federated learning converges better to reach a maximum accuracy of 95% which is 15% higher than the maximum accuracy of localized training setting.

Scenario 2: Inference phase with malicious devices. We let the operator to configure any IoT device(s) in the network to be malicious and period of being malicious. As we explained in Section II-B, PIFL, predicts whether the device is malicious or not, the trust level and the network slice to be assigned which are displayed on the dashboard (see Section III-B1). Fig. 2b shows IoT device-level components for a sample malicious device. We observe how the model accurately detects the malicious behavior and how the trust level falls drastically, immediately after the prediction turns malicious and the device is assigned to Slice 2 (highly untrusted). Similarly, when IoT 3 reverts to benign behavior, the trust level gradually increases, causing the device to be assigned to Slice 0. If there are fluctuations in the predictions, device is assigned to Slice 1.

Scenario 3: Agile slicing of the network. In PIFL we expect devices in the same network slice can communicate with each other. To demonstrate this, we use `ping` command to send ICMP packets between two selected devices. Initially, both devices will be in the same slice (e.g., highly trusted) and share ICMP packets. We move one device to a different slice, for example by converting it to malicious, and show how the ICMP packets between two devices are dropped. Also, if a device is set to be highly untrustworthy, the missing data reporting period on the IoT server will indicate the blocked communication between the server and devices.

IV. Conclusion

This paper proposed PIFL, a platform for in-network malicious node detection on resource-constrained edge devices using BNNs and federated learning. PIFL also calculates a trust value per device at each inference and enables slicing the network based on the trust level. Additionally, the platform also comprises a dashboard at the FL server which can be used to monitor the status of the overall system. We have built a prototype that implements the proposed platform and demonstrated the functionality of the platform in both the training and inference phases.

Acknowledgement

This work was funded by the NSW Department of Industry for Defence Innovation Network, Australia.

References

[1] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese *et al.*, "P4: Programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014.

[2] V. Gugueoth, S. Safavat, and S. Shetty, "Security of internet of things (iot) using federated learning and deep learning — recent advancements, issues and prospects," *ICT Express*, 2023.

[3] S. Laki, R. Stoyanov, D. Kis, R. Soulé, P. Vörös, and N. Zilberman, "P4pi: P4 on raspberry pi for networking education," *ACM SIGCOMM Computer Communication Review*, vol. 51, no. 3, pp. 17–21, 2021.

[4] Q. Qin, K. Poularakis, K. K. Leung, and L. Tassiulas, "Line-speed and scalable intrusion detection at the network edge via federated learning," in *2020 IFIP Networking Conference (Networking)*. IEEE, 2020, pp. 352–360.

[5] C. Zheng, M. Zang, X. Hong, R. Bensoussane, S. Vargaftik, Y. Ben-Itzhak, and N. Zilberman, "Automating in-network machine learning," *arXiv preprint arXiv:2205.08824*, 2022.

[4]https://prometheus.io/

[5]https://grafana.com/