privacy of the user data. In [15] a multi-step Distributed Denial of Service (DDoS) attack prediction method that uses Hidden Markov Model within a centralized FL architecture using Reinforcement Learning is presented and tested against a global algorithm, while in [16] a multi-class classifier for FL-based IDS considers multiple data distributions, and in [17], a self-learning distributed approach is developed to detect IoT devices compromised by Mirai malware. Similarly, [18] presents an anomaly detection approach based on centralized FL to classify and identify attacks in IoT networks, and has tested it on a dataset consisting of Man-in-the-Middle (MitM) and flood attacks. In [19], an architecture was proposed to mitigate DDoS in industrial IoT networks offering reduced mitigation delay.

2) Decentralized Federated Learning: In order to defend only against gradient attacks, Reference [20] proposed a decentralized FL framework, which is based on a peer-to-peer network for sending, aggregating, and updating local models. Another study [21] used decentralized FL to detect anomalies in network traffic generated by IoT devices, when all federated IDSs are shared with each distinct participant to obtain a weighted average, while in [22] blockchain-based FL was used as a decentralized architecture to specifically detect poisoning attacks.

B. Contributions of This Paper

In this paper, we propose a novel Decentralized Online Federated Learning Intrusion Detection (DOF-ID) architecture for improved online learning of ML-based IDS, that uses the Deep Random Neural Network [23]. The DOF-ID architecture hosts many IoT or IP node,s each of which utilizes an instance of a common IDS, learns directly from its local data, and collaborates with other nodes to incorporate their up-to-date knowledge into its IDS. This architecture improves the overall security of all collaborating nodes with online learning between nodes, by taking advantage of the experience of each node, while preserving the confidentiality of the local data at each of these nodes.

The DOF-ID architecture uses a learning procedure that combines Local Learning, and Decentralized Federated Updates (DFU) with concurrent parameter updates taking place on the collaborating nodes with local data. Therefore the proposed DOF-ID architecture with the DFU algorithm contrasts sharply with recent work on federated learning IDS.

We then evaluate the performance of the DOF-ID architecture and compare it with the performance of four benchmark methods, on three different types of cyberattacks obtained from two well-known public datasets: Kitsune [24], [25] and Bot-IoT [26].

The results show that DOF-ID provides significant performance gains compared to learning from local data alone and outperforms other state-of-the-art federated learning methods.

The remainder of this paper is organized as follows: Section II presents the novel DOF-ID architecture with DRNN-based IDS (Section II-A), local learning algorithm (Section II-B), and the DFU algorithm (Section II-C). Section III evaluates the performance of the proposed DOF-ID architecture on public datasets. Section IV summarizes the paper and presents some insights towards future work.

II. Intrusion Detection with Decentralized and Online Federated Learning

In order to improve the performance of an ML-based IDS, we now present a novel Decentralized and Online Federated Learning Intrusion Detection Intrusion Detection (DOF-ID) architecture, which is based on the collaboration of N nodes (denoted by set $\mathcal{N}$) using separate local instances of the same IDS. Figure 1 displays the proposed architecture from the perspective of a particular node n, where the nodes are represented as computer networks (e.g. Internet of Things (IoT) networks). From the application perspective, one may consider DOF-ID as a subscription based service, where each subscriber (i.e. node) receives the updates of other subscribers to improve its local security level.

As seen in Figure 1, each node n directly communicates with other nodes in $\mathcal{N}$ (i.e. peers) to send locally learned parameters of IDS and to receive those learned by other nodes. That is, locally learned IDS parameters are Peer-to-Peer (P2P) shared between every node in the DOF-ID architecture distributing the knowledge over collaborating nodes in $\mathcal{N}$ to improve their security (subsequently the global security) while the confidentiality of local data in every node is assured.

The DOF-ID architecture operates over time windows each with a length of T seconds, where time windows are considered to be synchronized among collaborating nodes. We also assume that the first time window (denoted by $l = 0$) starts with the use of DOF-ID architecture. Accordingly, at the beginning of each time window l, each node n updates its local IDS if no intrusion is detected in the previous window $l-1$. That is, if the intrusion decision of node n in window $l-1$, denoted by $y_n^{l-1} \in \{0, 1\}$, equals zero, node n executes the following steps as part of the learning procedure for the current window l:

1) It learns from training data containing local *benign* network traffic for windows up to beginning of l, denoted by $\mathcal{D}_n^l$, such that $\mathcal{D}_n^l = \{\, k : \quad y_n^k = 0, \forall k \in \{1, \ldots, l-1\}\,\}$. When the learning is completed, an up-to-date locally trained IDS of node n, denoted by I_n^l, is obtained to use for detection in window l.
2) It shares the parameters of I_n^l with other collaborating nodes in $\mathcal{N}$ and receives the local updates of those nodes, i.e. $\{I_{n'}^l\}_{n' \in \mathcal{N} \setminus n}$. In this paper, it is assumed that the P2P parameter exchange is instantaneous; however, future work shall analyse the time, bandwidth and energy requirements of the proposed DOF-ID architecture regarding P2P parameter exchange.
3) As the final step, node n updates the local IDS I_n^l by merging its parameters with $\{I_{n'}^l\}_{n' \in \mathcal{N} \setminus n}$ via the proposed DFU.

Following the training procedure in window l, each node n estimates the intrusion probability y_n^l through the following steps:

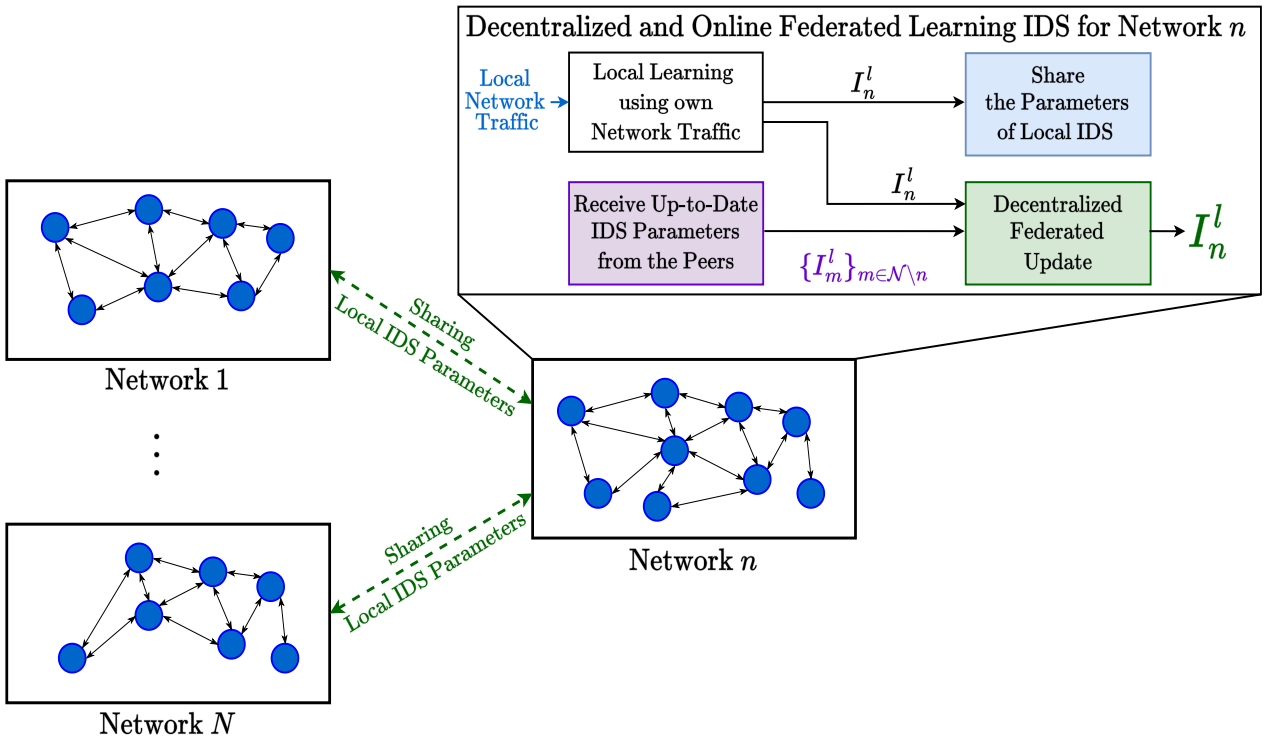

Fig. 1. Schematic system representation of the Decentralized and Online Federated Learning Intrusion Detection (DOF-ID) architecture.

5) The inputs of the utilized IDS are considered to be statistics calculated from the traffic of node n. Thus, node n first calculates traffic statistics as a vector of IDS inputs, denoted by x_n^l, in time window l.
6) Using the up-to-date IDS I_n^l, the final intrusion decision $y_n^l \equiv I_n^l(x_n^l)$ for traffic statistics x_n^l is calculated.

In the rest of this section, we respectively present our methodology for the particular ML-based IDS utilized in DOF-ID architecture as well as the local and federated learning algorithms.

A. IDS Utilized in the DOF-ID Architecture

Within our DOF-ID architecture, we use an IDS which is the modified version of the one presented in [13] and comprised of DRNN and Statistical Whisker-based Benign Classifier (SWBC) as shown in Figure 2. At each window l, this IDS estimates y_n^l that indicates whether the traffic of the considered node n in window l is malicious based on the input vector of traffic statistics, x_n^l.

1) Traffic Statistics: Let p_n^t denote the packet with length $|p_n^t|$ generated in node n at instantaneous time t, and let P_n^l be the set of all packets generated in n within time window l whose length equals T_n:

$$P_n^l = \{\, p_n^t : \ (l-1)\,T_n \le t < l\,T_n \,\}. \tag{1}$$

In each time window l, node n calculates three main statistics that represent the overall density of the network traffic as the average packet length in bytes (μ_n^l), the average number of packets per second (λ_n^l), and the average traffic in bytes per second (ρ_n^l):

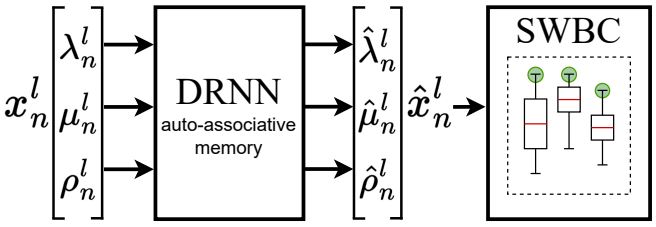

Fig. 2. Structure of the IDS utilized in the DOF-ID architecture

$$\mu_n^l = \frac{\sum_{p\in P_n^l}|p|}{|P_n^l|}, \quad \lambda_n^l = \frac{|P_n^l|}{T_n}, \quad \rho_n^l = \frac{\sum_{p\in P_n^l}|p|}{T_n}. \tag{2}$$

In order to use these statistics with DRNN, each element i of $x_n^l = [\mu_n^l, \lambda_n^l, \rho_n^l]$, denoted by $x_{n,i}^l$, is normalized to have values in $[0, 1]$.

2) Deep Random Neural Network to Create Auto-Associative Memory: In order to create an auto-associative memory, we use the well-known lightweight deep learning model DRNN [23], which is a Random Neural Network [11] model with feed-forward and clustered structure. As a result of its unique architecture presented in [23], each neuron at hidden layers of DRNN utilizes the following activation function,

which is specific to this model:

$$\Psi(\Lambda) = \frac{p\,(r+\lambda^+)+\lambda^-+\Lambda}{2\,[\lambda^-+\Lambda]} - \sqrt{\Big(\frac{p\,(r+\lambda^+)+\lambda^-+\Lambda}{2\,[\lambda^-+\Lambda]}\Big)^2 - \frac{\lambda^+}{\lambda^-+\Lambda}}\,, \quad (3)$$

where Λ is the input of the given cluster, p is the probability that any neuron received trigger transmits a trigger to some other neuron, and λ^+ and λ^- are respectively the rates of external Poisson flows of excitatory and inhibitory input spikes to any neuron. On the other hand, the neurons at the output layer of DRNN utilize linear activation functions.

As we consider three different network statistics, the DRNN model that we use in this paper consists of $H = 3$ fully connected layers with three neurons each. Accordingly, from the input vector x_n^l, DRNN estimates vector $\hat{x}_n^l$ of the statistics that are expected to be observed when the network traffic is benign:

$$\hat{x}^l_{(n,1)} = \Psi([x^l_n, 1]\, W^l_{(n,1)}) \quad (4)$$

$$\hat{x}^l_{(n,h)} = \Psi([\hat{x}_{(n,h-1)}, 1]\, W^l_{(n,h)}) \;\; \forall h \in \{2,\dots,H-1\}, \quad (5)$$

$$\hat{x}^l_n = [\hat{x}^l_{(n,h-1)}, 1]\, W^l_{(n,H)}, \quad (6)$$

where $\hat{x}^l_{(n,h)}$ is the output of layer h, and $\hat{x}^l_n$ is the final output of DRNN for node n in window l. In addition, the term $[x^l_n, 1]$ or $[\hat{x}^l_n, 1]$ indicates that 1 is added to the input of each layer as a multiplier of the bias, and $W^l_{(n,h)}$ is the connection weight matrix between layers $h-1$ and h of DRNN in I^l_n.

3) Statistical Whisker-based Benign Classifier: As the second operation in I^l_n (IDS of node n in time window l) that makes a decision on an intrusion, SWBC is used to measure the significance of the difference between the actual statistics measured from the network traffic and the expected statistics estimated by DRNN. SWBC is originally proposed in [13] and calculates the decision y^l_n as follows:

$$\zeta^l_n = \sum_{i\in\{1,2,3\}} \mathbf{1}(|x^l_{n,i} - \hat{x}^l_{n,i}| > w^l_{n,i}), \quad (7)$$

$$y^l_n = \mathbf{1}(\zeta^l_n > \theta^l_n), \quad (8)$$

where $x^l_{n,i}$ is the i–th element of vector x^l_n corresponding the traffic statistic i, and $\{w^l_{n,i}\}_{i\in\{1,2,3\}}$ and θ^l_n are the only parameters of the decision maker which are computed (learned) during training along with the connection weights of DRNN.

B. Local Learning

We now present the methodology of the local learning procedure that node n executes to learn parameters of I^l_n only using local data. In this procedure, node n respectively learns the DRNN weights and SWBC parameters for window l based on the available data $\mathcal{D}^l_n$.

1) Learning DRNN Weights: Using the local data of node n, DRNN in I^l_n is trained to create an auto-associative memory for the normal – benign – network traffic. To this end, first the connection weights of each hidden layer $h \in \{1,\dots,H-1\}$ are calculated by minimizing a square cost with L1 regularization via Fast Iterative Shrinkage-Thresholding Algorithm (FISTA) with the following objective:

$$W^l_{(n,h)} = \underset{\{W:\, W\geq 0\}}{\operatorname{argmin}} \Big(\Big|\Big| [adj(\Psi(\hat{X}^l_{(n,h-1)} W_R\ )), \mathbf{1}_{|\mathcal{D}^l_n|}]\, W - \hat{X}^l_{(n,h-1)} \Big|\Big|_2^2 + ||W||_1 \Big)\,, \quad (9)$$

where $\hat{X}^l_{(n,h-1)}$ is the matrix of $\hat{x}^k_{(n,h-1)}$ collected for $k \in \mathcal{D}^l_n$ for $h \geq 1$, $\hat{X}^l_{(n,0)} = X^l_n$ which is the matrix of x^k_n collected for $k \in \mathcal{D}^l_n$, $\mathbf{1}_{|\mathcal{D}^l_n|}$ is a column vector of ones with length $|\mathcal{D}^l_n|$, and W_R is randomly generated $(H \times H)$ matrix with elements in the range $[0,1]$. In addition, $adj(A)$ linearly maps the elements of matrix A to the range $[0,1]$ then applies z-score, and adds a positive constant to remove negativity.

For each layer $h \in \{1,\dots,H-1\}$, after FISTA is executed, we normalize each resulting weight matrix $W^l_{(n,h)}$:

$$W^l_{(n,h)} \leftarrow 0.1 \frac{W^l_{(n,h)}}{\max_{k\in\mathcal{D}^l_n}\big(\hat{X}^k_{(n,h)}\big)}\,. \quad (10)$$

The connection weights of the output layer H are calculated via an extreme learning machine as

$$W^l_{(n,H)} = (\hat{X}^l_{(n,H-1)})^+ X^l_n, \quad (11)$$

where A^+ denotes the pseudo-inverse of matrix A.

2) Computing SWBC Parameters: Using the training data, $\mathcal{D}^l_n$, which consists of only benign traffic features, we determine the values of θ^l_n and $w^l_{n,i}$ for each statistic i. To this end, for each i, the value of the absolute difference $z^k_{n,i} = |x^k_{n,i} - \hat{x}^k_{n,i}|$ is computed for all $k \in \mathcal{D}^l_n$.

Then, we compute the lower quartile $Q^L_{n,i}$ and upper quartile $Q^U_{n,i}$ of $\{z^k_{n,i}\}_{k\in\mathcal{D}^l_n}$. Using $Q^L_{n,i}$ and $Q^U_{n,i}$, the upper whisker $w^l_{n,i}$ is calculated as

$$w^l_{n,i} = Q^U_{n,i} + \frac{3}{2}(Q^U_{n,i} - Q^L_{n,i}) \qquad \forall i \in \{1,2,3\} \quad (12)$$

Since the training data contains only benign traffic, θ^l_n must be selected to classify training samples as benign traffic. Meanwhile, we should also consider that the training data may include false negative samples. Therefore, we determine θ^l_n to classify the majority but not all of the training samples as benign traffic, and we set the value of θ^l_n to the mean of ζ^l_n (i.e. the average number of abnormal statistics) plus two standard deviations of ζ^l_n in $\mathcal{D}^l_n$:

$$\theta^l_n = \text{mean}_{\mathcal{D}^l_n}(\zeta^l_n) + 2\,\text{std}_{\mathcal{D}^l_n}(\zeta^l_n) \quad (13)$$

C. Decentralized Federated Update

We now present the Decentralized Federated Update (DFU) algorithm that is performed as the last step of our DOF-ID architecture. In the DFU algorithm, the parameters of node n are updated using the parameters of other nodes in DOF-ID, whose data is unknown by node n. To this end, at each window l in this algorithm, node n performs three main operations: 1) select the set of concurring nodes, denoted by $\mathcal{C}_n^l$ that achieve decisions similar to those of node n, 2) update the value of each parameter segment in I_n^l using the corresponding segment with closest value to that segment among all nodes in $\mathcal{C}_n^l$, 3) recalculate the output layer weights of DRNN via extreme learning machine in order to fully adapt updated parameters to the local network traffic.

1) Selecting a Set of Concurring Nodes: In the current window l, node n first selects a set of nodes that concur with it for most of its decisions regarding local data. In order to select the concurring nodes, node n evaluate the performance of each node $m \in \mathcal{N} \setminus n$ on the local data of node n over all time windows up to current window l:

$$\mathcal{C}_n^l = \{m : \quad \frac{1}{l}\sum_{k=1}^{l} \mathbf{1}\big(I_m^l(x_n^k) = y_n^k\big) \geq \Theta, \quad \forall m \in \mathcal{N} \setminus n\} \tag{14}$$

2) Updating IDS Parameters: Using the IDSs of the concurring nodes, the parameters of I_n^l are updated separately for each segment of the IDS (such as each DRNN layer, each SWBC whisker, and the SWBC threshold) averaging with the closest one for that segment among all the concurring nodes.

To this end, first, for each layer $h \in \{1, \ldots, H-1\}$ of DRNN in I_n^l, the node m_h^* that has the closest connection weights $W_{(n,h)}^l$ with node n in window l is obtained:

$$m_h^* = \underset{m \in \mathcal{C}_n^l}{\arg\min} \left(\left|\left| W_{(n,h)}^l - W_{(m,h)}^l \right|\right|_1 \right). \tag{15}$$

Then, the connection weights of this layer, $W_{(n,h)}^l$, are updated as

$$W_{(n,h)}^l \leftarrow c\, W_{(n,h)}^l + (1-c)\, W_{(m_h^*,h)}^l, \tag{16}$$

where $0.5 \leq c \leq 1$ is a coefficient of weighted averaging that prioritizes locally learned weights over federated weights.

Similarly, for each whisker $w_{n,i}^l$ of SWBC in I_n^l, the node m_i^* with the whisker value $w_{m_i^*,i}^l$ closest to $w_{n,i}^l$ is obtained among concurring nodes in window l:

$$m_i^* = \underset{m \in \mathcal{C}_n^l}{\arg\min} \left(\left| w_{n,i}^l - w_{m,i}^l \right| \right), \tag{17}$$

and each whisker $w_{n,i}^l$ of SWBC in I_n^l is updated:

$$w_{n,i}^l \leftarrow c\, w_{n,i}^l + (1-c)\, w_{m_i^*,i}^l. \tag{18}$$

The decision threshold θ_n^l is also updated as

$$\theta_n^l \leftarrow c\, \theta_n^l + (1-c)\, \theta_{m_\theta^*}^l \tag{19}$$

for

$$m_\theta^* = \underset{m \in \mathcal{C}_n^l}{\arg\min} \left(\left| \theta_n^l - \theta_m^l \right| \right). \tag{20}$$

3) Adapting the Updated IDS to Local Network Traffic: Finally, the output layer weights of DRNN in the IDS I_n^l are updated to fully adapt I_n^l to local *benign* network traffic of node n. To this end, (11) is repeated:

$$W_{(n,H)}^l = (\hat{X}_{(n,H-1)}^l)^+ X_n^l. \tag{21}$$

III. Experimental Results

We now evaluate the performance of the proposed DOF-IDS architecture. To this end, from two publicly available datasets Kitsune [25] and Bot-IoT [26], we use three attack data each of which corresponds to a single node in DOF-IDS architecture. That is, we consider three collaborating nodes each of which is an IoT network whose data is obtained from a public dataset.

We perform the experiments on a computer with 16 GB of ram and M1 Pro 8-core 3.2 GHz processor. The performance of DOF-IDS is also compared against four benchmark methods.

A. IoT Traffic Datasets and Their Processing

As the first node in DOF-IDS architecture, we use "Mirai Botnet" attack data from the Kitsune dataset [25], which is the collection of $764,137$ individual traffic packets, which are transmitted by 107 unique IP addresses within 7137 seconds (approximately 2 hours).

As the second and third nodes in DOF-IDS, we use "DoS HTTP" and "DDoS HTTP" attacks from Bot-IoT dataset [26]. The DoS HTTP attack data contains $29,762$ packets transmitted in 49 minutes, and the DDoS HTTP attack data contains $19,826$ packets transmitted in 42 minutes. Since these two types of attacks start this data with an attack traffic and the presented system requires cold-start with only benign traffic, we use each of these data by flipping it on the time axis.

During our experimental results in order to obtain approximately the same number of time windows from each dataset, we set the values of T_n as follows: 23 for Mirai, 9 for DoS HTTP, and 8 for DDoS HTTP. One should also note that both datasets include a binary ground truth $a(p_n^t)$ for each packet p_n^t, which is determined by the providers, stating whether the packet is a normal "benign" packet or "malicious" corresponding to an ongoing attack. Accordingly, based on the individual packet ground truths, we determine an overall ground truth, denoted by g_n^l, for each node n in each time window l:

$$g_n^l = \mathbf{1}\left(\frac{\sum_{p \in P_n^l} a(p)}{|P_n^l|} > 0.5 \right) \tag{22}$$

B. Benchmark Methods

1) No Federated: In this method, the contributions of the other nodes are not considered in the learning of the IDS parameters. This is the conventional training approach, which is equivalent to the local learning procedure of our DOF-IDS architecture.

2) Average over All Collaborating Nodes: The rest of the benchmark methods are used in place of the DFU algorithm to update the IDS parameters after local learning.

In this method called "Average", the parameters of IDS in node n (i.e. I_n^l) are updated as the average of all connection weights over all collaborating nodes in the DOF-ID architecture. To this end, connection weights for each layer h of DRNN in I_n^l are updated as

$$W_{(n,h)}^l \leftarrow \frac{1}{N} \sum_{m \in \mathcal{N}} W_{(m,h)}^l, \tag{23}$$

Subsequently, SWBC parameters are also updated in the same way:

$$w_{n,i}^l \leftarrow \frac{1}{N} \sum_{m \in \mathcal{N}} w_{m,i}^l \quad \forall i, \quad \text{and} \quad \theta_n^l \leftarrow \frac{1}{N} \sum_{m \in \mathcal{N}} \theta_n^l \tag{24}$$

One should note that the parameters updated using this method become the same for all nodes. That is, at the end of this method, $I_n^l = I_m^l, \ \forall n, m \in \mathcal{N}$.

3) Average with Closest Node: In this method called "Average with Closest Node (ACN)", the parameters of I_n^l are updated by taking their average with the closest parameters among all nodes in the DOF-ID architecture. To this end, first, the node m^* that has the closest parameters with node n at time window l is obtained:

$$m^* = \arg\min_{m \in \mathcal{N} \setminus n} \left(\sum_{h=1}^{H} \left|\left| W_{(n,h)}^l - W_{(m,h)}^l \right|\right|_1 + \sum_{i=1}^{3} \left| w_{n,i}^l - w_{m,i}^l \right| + \left| \theta_n^l - \theta_m^l \right| \right) \tag{25}$$

Then, in time window l, the parameters I_n^l are updated taking their average with the parameters of $I_{m^*}^l$ as

$$W_{(n,h)}^l \leftarrow \frac{W_{(n,h)}^l + W_{(m^*,h)}^l}{2}, \quad \forall h \tag{26}$$
$$w_{n,i}^l \leftarrow \frac{w_{n,i}^l + w_{m^*,i}^l}{2}, \quad \forall i \qquad \theta_n^l \leftarrow \frac{\theta_n^l + \theta_{m^*}^l}{2}$$

4) Average with Closest Node per Layer: In the last benchmark method called "Average with Closest Node per Layer (ACN-L)", the parameters of I_n^l are updated for each parameter segment of the IDS (such as a layer of DRNN, a whisker of SWBC, and the threshold of SWBC) individually taking the average with the same part of the closest node.

For each layer h of DRNN in I_n^l, the connection weights of this layer are updated using (16) for a value of m_h^* calculated using (15). Then, each whisker $w_{n,i}^l$ of SWBC in I_n^l is updated (18) for a value of m_i^* calculated using (17). The decision threshold θ_n^l is updated by subsequently using (20) and (19).

C. Performance Evaluation

We now present the performance evaluation results of our DOF-ID architecture, where we set $c = 0.75$ and $\Theta = 0.65$. In addition, we used the DRNN model that has 10 neurons in each cluster and has the following parameter settings: $p = 0.05$, $r = 0.001$, and $\lambda^+ = \lambda^- = 0.1$.

Figure 3 displays the average performance of DOF-ID with respect to Accuracy, True Positive Rate (TPR), and True Negative Rate (TNR). The results in this figure show that each node (i.e. Mirai, DoS HTTP and DDoS HTTP) achieves above 0.86 detection performance with respect to all metrics. One may also see that although the nodes suffer from some false positive alarms (shown by the TNR metric), all nodes detect local intrusions with a considerably high performance (shown by the TPR metric).

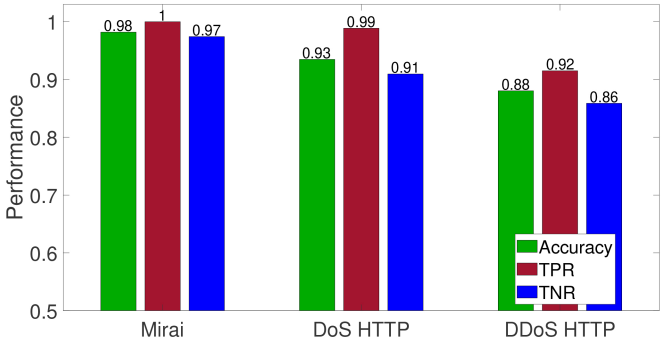

Fig. 3. Performance of the DOF-ID architecture for each node among Mirai, DoS HTTP, and DDoS HTTP with respect to Accuracy, TPR, and TNR

We also compare the performance of DOF-ID with benchmark methods in Figure 4. The results in Figure 4 (top) show that the proposed method has the best accuracy among all methods compared. Another important observation of this figure is the poor performance of the averaging over all collaborating nodes. This is an expected result as network traffic across nodes varies considerably.

The evaluation results further show that FL-based methods (i.e. DOF-ID, Average, ACN, and ACN-L) significantly improve the detection performance measured by TPR in Figure 4 (middle), while they mostly tend to raise more false positive alarms compared to local learning as shown in Figure 4 (bottom). On the other hand, the proposed DOF-ID method appears to have a small decrease in TNR (i.e. a slight increase in false alarms) but a significant improvement in the detection rate, TPR.

Finally, we measure the training time of the proposed and compared methods. We especially measure the time required for federated update and present it in Table I since the local learning time is the same (with negligible random deviations) for all models, which equals $19.2\ ms$ on average.

The federated update time measurements in Table I show that the time spent by DOF-ID in addition to local learning is about $30\ ms$ for each node. This time is significantly larger than other methods as its operations are more advanced and detailed.

On the other hand, a method is considered to be acceptable for a real-time application as long as the total operation time spent on local and federated learning and detection is shorter than the window length T_n. For the DOF-ID architecture, the total operation time is $48.91\ ms$ on average as the sum of local

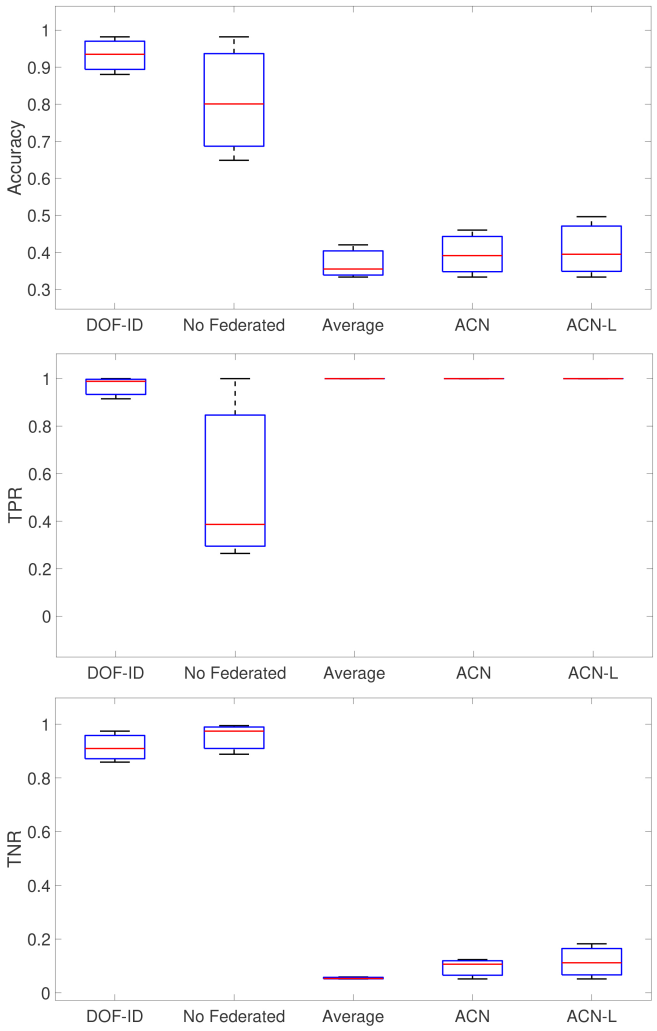

Fig. 4. Performance comparison of DOF-ID with benchmark methods (No Federated, Average, ACN, and ACN-L) with respect to accuracy (top), TPR (middle), and TNR (bottom) presented as a box-plot across all nodes, namely Mirai, DoS HTTP and DDoS HTTP

TABLE I
AVERAGE TIME SPENT IN MICROSECONDS (μs) BY EACH METHOD FOR THE FEDERATED UPDATE IN A SINGLE WINDOW

	Mirai	**DoS HTTP**	**DDoS HTTP**
DOF-ID	29.6×10^3	29.7×10^3	29.7×10^3
Average	34.3	33.3	32.8
ACN	74.5	66.4	66.3
ACN-L	106.8	101.1	113.2

learning time of 19.2 ms, federated learning time of 29.6 ms, and detection time of 0.11 ms.

IV. CONCLUSIONS

This paper proposed a novel Decentralized and Online Federated Learning Intrusion Detection (DOF-ID) architecture to improve the detection performance of anomaly-based IDS using a DRNN model and SWBC decision maker, both of which learn using only normal "benign" network traffic. The presented DOF-ID architecture provides a collaborative learning system that enables each node to learn from the experiences of other collaborating nodes without violating data confidentiality. In this way, DOF-ID improves both local and global security levels of all collaborating nodes simultaneously, quickly and effectively eliminating the requirement for a large learning data.

This paper also evaluates the performance of DOF-ID and compares it against the benchmark methods using two public well-known datasets, Kitsune and Bot-IoT. During the performance evaluation, the impacts of FL on intrusion detection performance are also investigated. Our experimental results revealed that the proposed DOF-ID method significantly improves the detection performance with a small increase in false positive alarms compared to the same IDS structure learning only from local traffic. In addition, the proposed method has significantly superior performance (at least 15% accuracy difference) over benchmark methods with higher computation time.

Future work shall primarily expand the experimental setup and evaluate the performance of the proposed DOF-ID architecture for large networked systems such as smart grids or large IoT networks. It would also be interesting to address the performance and security issues regarding the parameter exchange within the proposed DOF-ID architecture. Accordingly, we shall analyse the time, bandwidth and energy requirements of this architecture due to P2P parameter exchange and investigate the security breaches that may aim to leak or corrupt IDS parameters during their transfer. Another important issue that will have to be considered is that FL itself may come under attack in a distributed system of systems [27], so that this aspect will also require further research and attention.

REFERENCES

[1] M. A. Alsoufi, S. Razak, M. M. Siraj, I. Nafea, F. A. Ghaleb, F. Saeed, and M. Nasser, "Anomaly-based intrusion detection systems in iot using deep learning: A systematic literature review," *Applied sciences*, vol. 11, no. 18, p. 8383, 2021.

[2] P. Maniriho, E. Niyigaba, Z. Bizimana, V. Twiringiyimana, L. J. Mahoro, and T. Ahmad, "Anomaly-based intrusion detection approach for iot networks using machine learning," in *2020 international conference on computer engineering, network, and intelligent multimedia (CENIM)*. IEEE, 2020, pp. 303–308.

[3] E. Gelenbe and M. Nakıp, "Traffic based sequential learning during botnet attacks to identify compromised IoT devices," *IEEE Access*, vol. 10, pp. 126 536–126 549, 2022.

[4] A. Nisioti, A. Mylonas, P. D. Yoo, and V. Katos, "From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3369–3388, 2018.

[5] I. H. Sarker, "Cyberlearning: Effectiveness analysis of machine learning security modeling to detect cyber-anomalies and multi-attacks," *Internet of Things*, vol. 14, p. 100393, 2021.

[6] P. Kairouz, et al., "Advances and open problems in federated learning," 2019. [Online]. Available: https://arxiv.org/pdf/1912.04977.pdf

[7] S. R. Pokhrel and J. Choi, "Federated learning with blockchain for autonomous vehicles: Analysis and design challenges," *EEE Transactions on Communications*, vol. 68, no. 8, p. 4734–4746, 2020.

[8] S. Savazzi, M. Nicoli, and V. Rampa, "Federated learning with cooperating devices: A consensus approach for massive iot networks," *IEEE Internet of Things Journal*, no. 5, p. 4641–4654, 2020.

[9] Z. Xu, F. Yu, J. Xiong, and X. Chen, "Helios: Heterogeneity-aware federated learning with dynamically balanced collaboration," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, 2021, p. 997–1002.

[10] E. Gelenbe, "G-networks with instantaneous customer movement," *Journal of Applied Probability*, vol. 30, no. 3, pp. 742–748, 1993.

[11] ——, "Random neural networks with negative and positive signals and product form solution," *Neural Computation*, vol. 1, no. 4, pp. 502–510, 1989.

[12] O. Brun, Y. Yin, and E. Gelenbe, "Deep learning with dense random neural network for detecting attacks against iot-connected home environments," *Procedia Computer Science*, vol. 134, pp. 458–463, 2018.

[13] E. Gelenbe and M. Nakıp, "G-networks can detect different types of cyberattacks," in *2022 30th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE, 2022, pp. 9–16.

[14] R. Taheri, M. Shojafar, M. Alazab, and R. Tafazolli, "Fed-iiot: A robust federated malware detection architecture in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8442–8452, 2020.

[15] Z. Li, X. Wu, and C. Jiang, "Efficient poisoning attacks and defenses for unlabeled data in ddos prediction of intelligent transportation systems," *Security and Safety*, vol. 1, p. 2022003, Jun 2022.

[16] E. M. Campos, P. F. Saura, A. González-Vidal, J. L. Hernández-Ramos, J. B. Bernabé, G. Baldini, and A. Skarmeta, "Evaluating federated learning for intrusion detection in internet of things: Review and challenges," *Computer Networks*, vol. 203, p. 108661, 2022.

[17] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "Dïot: A federated self-learning anomaly detection system for iot," in *2019 IEEE 39th International conference on distributed computing systems (ICDCS)*. IEEE, 2019, pp. 756–767.

[18] V. Mothukuri, P. Khare, R. M. Parizi, S. Pouriyeh, A. Dehghantanha, and G. Srivastava, "Federated-learning-based anomaly detection for iot security attacks," *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2545–2554, 2021.

[19] J. Li, L. Lyu, X. Liu, X. Zhang, and X. Lyu, "Fleam: A federated learning empowered architecture to mitigate ddos in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 6, pp. 4059–4068, 2021.

[20] G. Lu, Z. Xiong, R. Li, N. Mohammad, Y. Li, and W. Li, "Defeat: A decentralized federated learning against gradient attacks," *High-Confidence Computing*, p. 100128, 2023.

[21] Z. Lian and C. Su, "Decentralized federated learning for internet of things anomaly detection," in *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, 2022, pp. 1249–1251.

[22] R. Al Mallah and D. López, "Blockchain-based monitoring for poison attack detection in decentralized federated learning," in *2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*. IEEE, 2022, pp. 1–6.

[23] E. Gelenbe and Y. Yin, "Deep learning with random neural networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*, 2016, pp. 1633–1638.

[24] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *The Network and Distributed System Security Symposium (NDSS) 2018*, 2018.

[25] "Kitsune Network Attack Dataset," August 2020. [Online]. Available: https://www.kaggle.com/ymirsky/network-attack-dataset-kitsune

[26] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[27] P. Liu, X. Xu, and W. Wang, "Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives," *Cybersecurity*, vol. 5, no. 4, 2022. [Online]. Available: https://doi.org/10.1186/s42400-021-00105-6