Federated Learning Based Intuition Detection Systems for IoT

Nathanael Bowley, Hongwei Zhang, Raham Moghaddam, Han Yang, Ehssan Mousavipour

DALHOUSIE UNIVERSITY

Outline

1. Nguyen et al.:
   - Introduction
   - Background and Related Work
   - System and
   - Experiment and Results
   - Conclusion and Future Work
2. Popoola et al.:
   - Extensions and Changes

What are IoT Devices?

- Sensors and Actuators: IoT devices are equipped with sensors that gather data from the environment, such as temperature, humidity, and light, and actuators that can perform actions based on that data
- IoT devices are connected to the internet, which allows them to exchange data and communicate with other devices and systems
- IoT devices generate large amounts of data, which can be analyzed using machine learning algorithms to extract insights and optimize processes, such as predicting equipment failures or optimizing energy usage.

Figure 1. Blink Mini- Compact indoor plug-in smart security camera

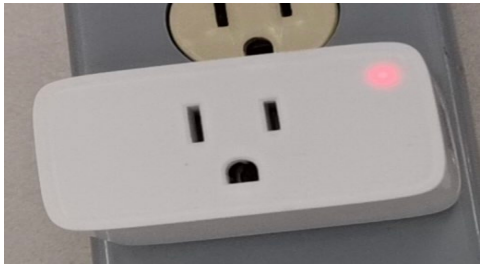


Figure 2. MyLink Mini Wi-Fi Smart Plug

What is Federated Learning?

- Federated Learning (FL) is a distributed machine learning scheme that coordinates many clients for training models through one or more central servers

- Compared with other model aggregation methods, FL provides better security data aggregation and is suitable to use when data is distributed.

DÏoT: A Federated Self-learning Anomaly Detection System for IoT

T. D. Nguyen, S. Marchal, M. Miettinen,
H. Fereidooni, N. Asokan
and A. -R. Sadeghi

Background and Related Work

- IoT Devices released with security vulnerabilities are targeted by malware [19] - [22].
- Delays or absence of automated security updates introduce vulnerabilities for devices and leads to reactive security methods such as intrusion detection systems (IDS) to detecting if a device has been compromised or not [15] - [18].
- Similarly, anomaly detection profiles the normal device behavior to detect previously unknown attacks when the device deviates from its normal behavior profile. [16] - [19].

- Unusable in practice due to:
  - High false alarm rate and high IoT device heterogeneity from high device variability.
  - Limited communication functions (LCFs) in these devices corresponds to low generation of network traffic. LCFs lead to inaccurate, uncomprehensive, and overgeneralized training models for anomaly detection in IoT devices.

Background and Related Work (1)

Detection and prevention of intrusions in IoT networks

- [13] outlines the taxonomy of IoT attacks on Smart Applications and creates ContexIoT to improve context integrity support.
- [14] presents the first IoT intrusion detection system for IPv6-connected IoT devices but suffers from a poor false alarms rate when detecting malicious nodes.
  - In contrast, DÏoT reports no false alarms in a real-world smart home deployment setting.
- Doshi et al. [15], Used an anomaly detection pipeline approach for feature extraction and binary classification using multiple approaches to differentiate normal traffic from DoS attack traffic.
  - In DÏoT legitimate traffic is modelled to allow dynamic detection of unknown malicious traffic.

Background and Related Work (2)

Detecting anomalies in network traffic

- Procedural analysis of individual packets [8] or clustering of multiple packets [9] - [10] was the previous standards in the literature.
- [11] - [12] set the foundation for DÏoT by being the first to model communication as a language to be able to detect anomalies.
- [12] used finite state automatons to model short sequences of packets to detect anomalies and attacks. DÏoT follows a similar approach but uses GRU to both model protocol specifications and allow for modeling longer sequences of packets.

Background and Related Work (3)

Recurrent Neural networks (RNN)

- A more recent novel approach for anomaly-detection which has been used by [4] – [7].
- Oprea et al. [8], used deep belief networks; a type of RNN; for detecting infection of enterprise networks. GRU which is used in DÏoT is also a RNN but is considered to allow for faster training, real time operations, and operates in real time unlike in [8].

Motivation

- Differentiating typical and atypical IoT device behavior crucial for anomaly detection
- Numerous device types and infrequent communication complicate learning and updating processes
- Difficulty in accurately detecting abnormal behavior without causing false alarms
- Device-type-specific detection models for accurate detection of anomalous behavior
- Novel anomaly detection approach using network packets as language symbols
- Federated learning for aggregating anomaly-detection profiles
- Attack dataset with real IoT malware (Mirai) for system evaluation
- DÏoT: A self-learning, distributed system for IoT device security monitoring

System Overview

Security Gateway

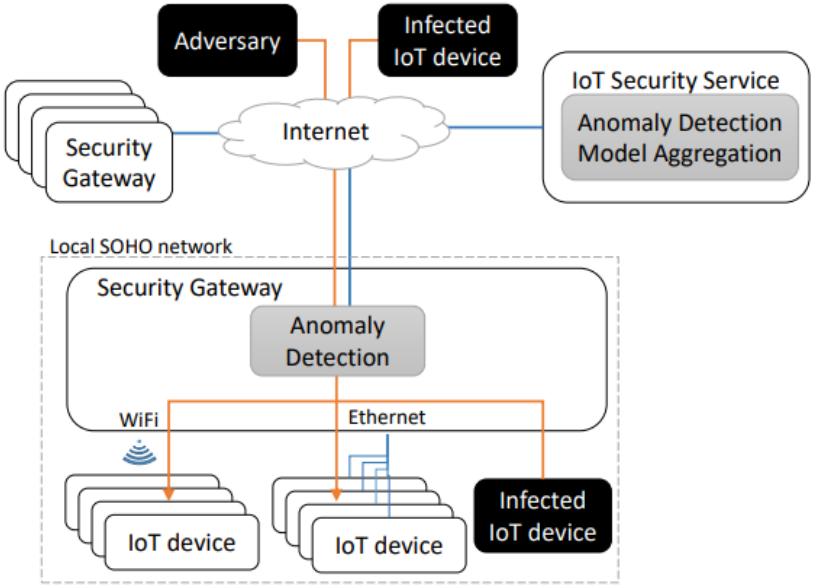
- Anomaly Detection component
- Identifies type of IoT devices

IoT Security service

- Aggregate local model to update global model
- Maintaining a repository of device type specific anomaly detection models

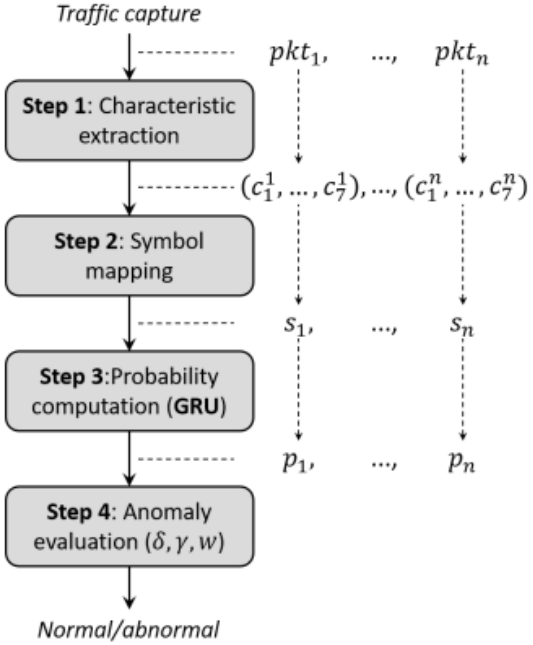
System Architecture

- Adversary
  - The adversary is IoT malware performing attacks against, or launching attacks from, vulnerable devices in the small office home office (SOHO) network.
- Defense goal
  - The primary goal of DÏoT is to detect attacks on IoT devices to take appropriate countermeasures



Device-Type-Specific Anomaly Detection

Modelling Packet Sequences



ID	Characteristic	Value
c_1	direction	$1 =$ incoming, $0 =$ outgoing
c_2	local port type	bin index of port type
c_3	remote port type	bin index of port type
c_4	packet length	bin index of packet length
c_5	TCP flags	TCP flag values
c_6	protocols	encapsulated protocol types
c_7	IAT bin	bin index of packet inter-arrival time

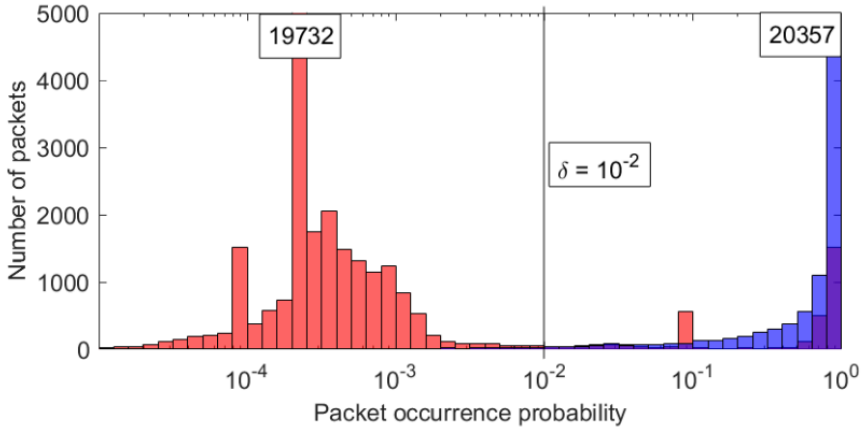
Device-Type-Specific Anomaly Detection

Detection Process

- Probability computation

  They used the GRU model to calculate an occurrence probability p_i for each packet symbol s_i given the sequence of k preceding symbols

$$p_i = P(s_i | < s_{i-k}, s_{i-k+1}, \ldots, s_{i-1} >)$$



Device-Type-Specific Anomaly Detection

Anomaly Evaluation

How to trigger the alarm to reduce False Positive Rate (FPR)?

An anomaly is triggered only significant number of packets in a window of consecutive packets are anomalies

The condition to trigger the anomaly alarm is:

$$\frac{|\{s_i \in S | p_i < \delta\}|}{w} > \gamma$$

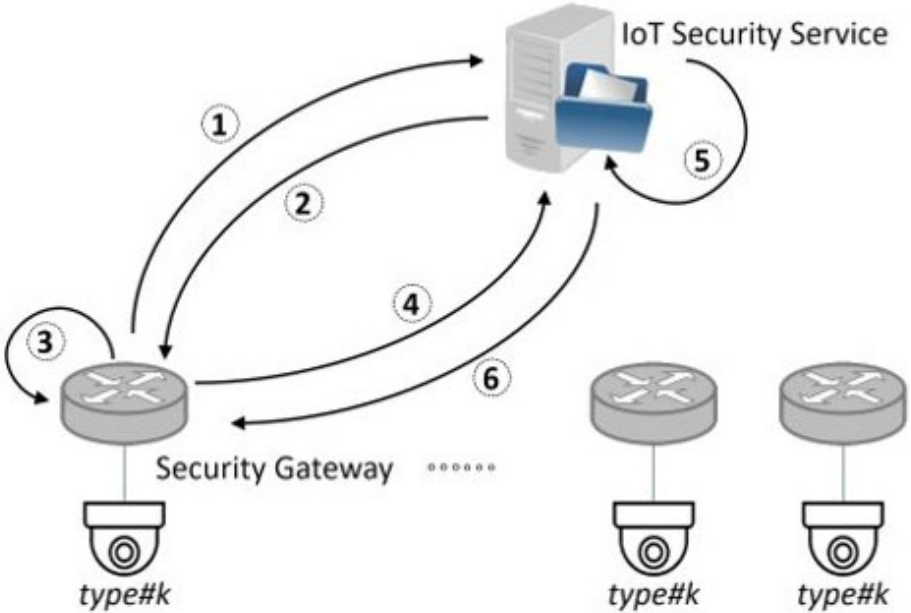
Device-Type-Specific Anomaly Detection

Detection Process

- Initialize GRU model
- Train by local data
- Update global model
- Send back global model

Global model G which is aggregated from those local models as follows:

$$G = \sum_{i=1}^{n} \frac{s_i}{s} W_i \quad (\text{where } s = \sum_{i=1}^{n} s_i)$$



Experimental Setup (1)

Datasets

- Activity Dataset
  - The operations of the user invoking the IoT device (On/Off)
  - The dataset contains records of the operations of 33 IoT devices
- Deployment Dataset
  - Smart home IoT devices were installed in several different scenarios
  - Communication traces of these devices were collected under real-life usage conditions
- Attack Dataset
  - Dataset of malicious traffic from IoT devices infected with Mirai malware in four different attack phases (pre-infection, infection, scanning, and DoS attack)
  - Dataset of traffic when Mirai is in standby mode

Experimental Setup (2)

Device-type	Identifier	Device model	WiFi	Ethernet	Other	Activity	Deployment	Attack
type#01	ApexisCam	Apexis IP Camera APM-J011	•	•	○	•	○	○
type#02	CamHi	Cooau Megapixel IP Camera	•	•	○	•	○	○
type#03	D-LinkCamDCH935L	D-Link HD IP Camera DCH-935L	•	○	○	•	○	○
type#04	D-LinkCamDCS930L	D-Link WiFi Day Camera DCS-930L	•	•	○	•	•	•
	D-LinkCamDCS932L	D-Link WiFi Camera DCS-932L	•	•	○	•	•	•
type#05	D-LinkDoorSensor	D-Link Door & Window sensor	○	○	•	•	○	○
	D-LinkSensor	D-Link WiFi Motion sensor DCH-S150	•	○	○	•	•	○
	D-LinkSiren	D-Link Siren DCH-S220	•	○	○	•	○	○
	D-LinkSwitch	D-Link Smart plug DSP-W215	•	○	○	•	•	○
	D-LinkWaterSensor	D-Link Water sensor DCH-S160	•	○	○	•	○	○
type#06	EdimaxCamIC3115	Edimax IC-3115W Smart HD WiFi Network Camera	•	•	○	•	○	○
	EdimaxCamIC3115(2)	Edimax IC-3115W Smart HD WiFi Network Camera	•	•	○	•	○	○
type#07	EdimaxPlug1101W	Edimax SP-1101W Smart Plug Switch	•	•	○	•	•	•
	EdimaxPlug2101W	Edimax SP-2101W Smart Plug Switch	•	○	○	•	•	•
type#08	EdnetCam	Ednet Wireless indoor IP camera Cube	•	•	○	•	○	○
type#09	EdnetGateway	Ednet.living Starter kit power Gateway	•	○	•	•	•	○
type#10	HomeMaticPlug	Homematic pluggable switch HMIP-PS	○	○	•	•	○	○
type#11	Lightify	Osram Lightify Gateway	•	○	•	•	•	○
type#12	SmcRouter	SMC router SMCWBR14S-N4 EU	•	•	○	•	○	○
type#13	TP-LinkPlugHS100	TP-Link WiFi Smart plug HS100	•	○	○	•	○	○
	TP-LinkPlugHS110	TP-Link WiFi Smart plug HS110	•	○	○	•	○	○
type#14	UbnTAirRouter	Ubnt airRouter HP	•	•	○	•	○	•
type#15	WansviewCam	Wansview 720p HD Wireless IP Camera K2	•	○	○	•	○	○
type#16	WeMoLink	WeMo Link Lighting Bridge model F7C031vf	•	○	•	•	○	○
type#17	WeMoInsightSwitch	WeMo Insight Switch model F7C029de	•	○	○	•	○	○
	WeMoSwitch	WeMo Switch model F7C027de	•	○	○	•	○	○
type#18	HueSwitch	Philips Hue Light Switch PTM 215Z	○	○	•	•	•	○
type#19	AmazonEcho	Amazon Echo	•	○	○	•	•	○
type#20	AmazonEchoDot	Amazon Echo Dot	•	○	○	•	○	○
type#21	GoogleHome	Google Home	•	○	○	•	•	○
type#22	Netatmo	Netatmo weather station with wind gauge	•	○	•	•	•	○
type#23	iKettle2	Smarter iKettle 2.0 water kettle SMK20-EU	•	○	○	•	•	○
	SmarterCoffee	Smarter SmarterCoffee coffee machine SMC10-EU	•	○	○	•	•	○

Table 2: 33 IoT devices used in the datasets and their connectivity technologies

Category (count)	Typical actions
IP cameras (6)	START / STOP video, adjust settings, reboot
Smart plugs (9)	ON, OFF, meter reading
Sensors (3)	trigger sensing action
Smart lights (4)	turn ON, turn OFF, adjust brightness
Actuators (1)	turn ON, turn OFF
Appliances (2)	turn ON, turn OFF, adjust settings
Routers (2)	browse amazon.com
Hub devices (6)	no actions

Table 4: Actions for different IoT device categories

Experimental Setup (3)

Parameter Selection

- A lookback history of k=20 symbols is sufficient to capture most communication interactions with sufficient accuracy
- A GRU network with three hidden layers of size 128 neurons each
- Learned 23 anomaly detection models, each corresponding to a device type identified
- Each anomaly detection model was trained with, and respectively tested on, communication from all devices matching the considered type

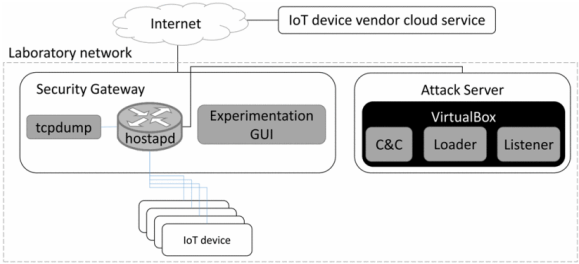


Figure 6: Laboratory network setup

Dataset (Number of devices)	Time (hours)	Size (MiB)	Flows	Packets
Activity (33)	165	465	115,951	2,087,280
Deployment (14)	2,352	578	95,518	2,286,697
Attack (5)	84	7,734	8,464,434	21,919,273

Table 3: Characteristics of used datasets

Experimental Setup (4)

Evaluation Metrics

- False Positive Rate (FPR): Normal communications are classified as abnormal
- True Positive Rate (TPR): Attacks Classified as abnormal
- Minimize FPR and keep users from being inundated with false positives
- Maximize TPR so that as many attacks as possible are detected

	Actually Positive	Actually Negative
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

Experimental Setup (5)

Evaluation Metrics

- Testing FP by four-fold cross-validation for device types in the Activity and Deployment datasets (three folds for training and one for testing)
  - FP: any triggered anomaly alarm for packets of the window
  - TN: windows without alarms
- Testing TP using the Activity and Deployment datasets as training data and the Attack dataset for testing
  - TP: windows of w=250 packets
  - FN: windows without triggered alarms

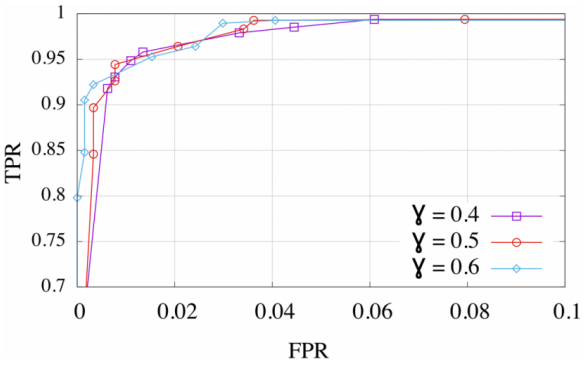


Figure 7: ROC curve of TPR and FPR in dependence of detection threshold δ and anomaly triggering threshold γ.

Experimental Results

Accuracy

- When Detection threshold was 0.01 and anomaly triggering threshold was 0.5 at w = 250 achieved 94.01% TPR at <0.01 FPR [1].
- Attack detection rate of 95.60% TPR and 0% FPR during one week of evaluation [1].
- DÏoT detects attacks in pre-infection stage after 223 packets while Mirai generates more than 900 packets during pre-infection [1].
- The detection rate for DoS attacks is 88.96% TPR [1].

Efficiency of Federated Learning

- Federated models with more participating clients achieve better FPR, while TPR deteriorates only slightly [1].
- A small decrease in TPR as we increase the number of clients by the system [1].
- A single model for all devices has a higher FPR (0.67%) and lower TPR (97.21%) compared to using device-type specific anomaly detection models [1].

Performance

- GRU performance evaluation without specific optimizations on a laptop and a desktop computer [1].

Experimental Results

TABLE V: Average detection times of analyzed Mirai attacks

Attack	packets/s.	det. time (ms.)	TPR
standby	*0.05*	*4,051,889*	*33.33%*
Pre-Infection	426.66	524	100.00%
Infection	721.18	272	93.45%
Scanning	752.60	166	100.00%
DoS	1,412.94	92	88.96%
Average	866.88	257±194	95.60%

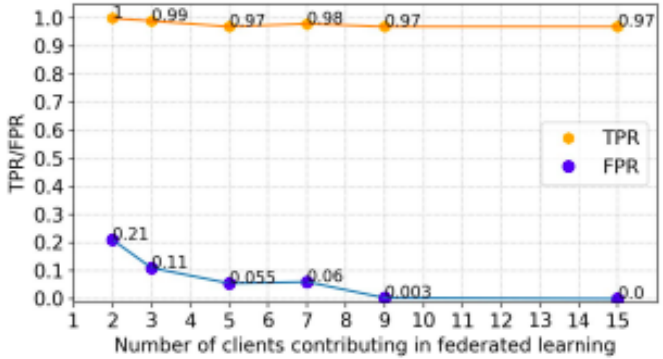


Fig. 8: Evolution of TPR and FPR as we increase the number of clients in federated learning. TPR decreases slightly (-3%) while FPR reaches 0 (-21%) when using 15 clients.

TABLE VI: Effect of using federated learning comparing to centralizing approach

Type	Centralized learning	Federated learning		
		5 clients	9 clients	15 clients
FPR	0.00%	0.00%	0.00%	0.00%
TPR	95.60%	95.43%	95.01%	94.07%

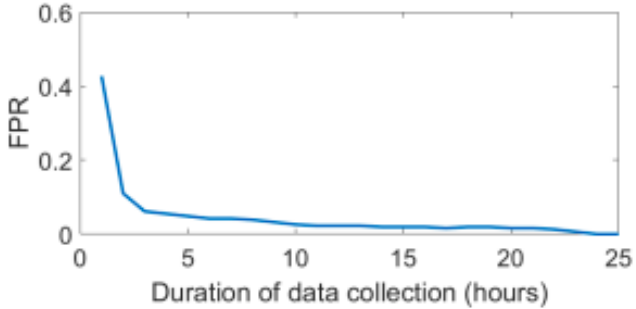


Fig. 9: Effect of training data size in time to FPR

while FPR is not deteriorated (remaining constant at 0.00%). This small drop in TPR is not a concern since a large number of packet windows would still trigger an alarm for any attack stage.

Conclusions and Future Work

- DÏoT relies on novel automated techniques for device-type specific anomaly detection
- No human intervention or labeled data is required for DÏoT to operate
- DÏoT learns anomaly detection models autonomously, using unlabeled crowdsourced data captured in client IoT network
- Efficacy of anomaly detection in detecting a large set of malicious behavior from devices infected by the Mirai malware was demonstrated
- DÏoT detected 95.6% of attacks in 257 milliseconds on average
- DÏoT detects attacks without raising any alarms

Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT-Edge Devices

S. I. Popoola, R. Ande, B. Adebisi,
G. Gui, M. Hammoudeh
and O. Jogunola

Differences in Motivation

[2] outlines their motivations in 3 comparison areas to past research and found that the DÏoT paper in [2] lacked in:

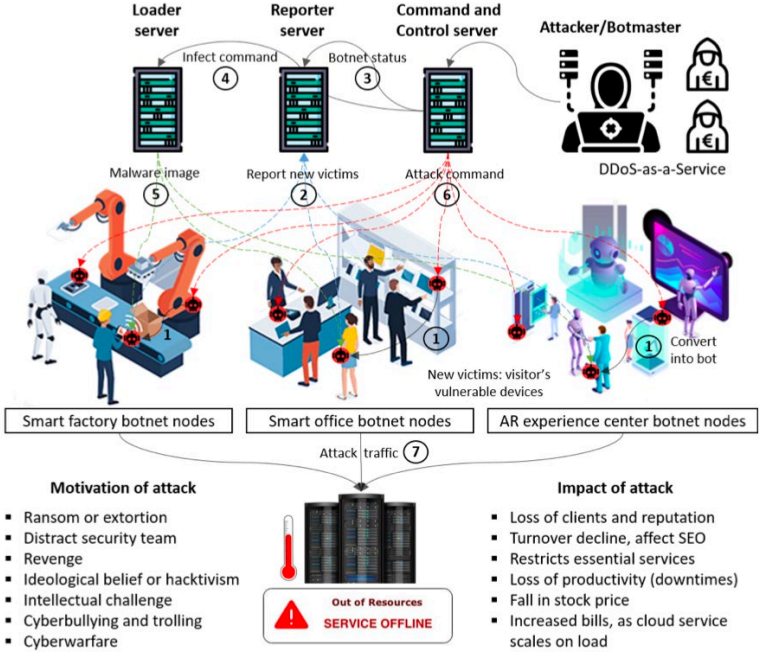
a. Botnet Attacks
b. Zero-day Scenarios
c. IoT Traffic

TABLE I
COMPARISON WITH RELATED WORKS

Ref.	Year	Model	Dataset	IoT traffic	Botnet attacks	Zero-day scenario	Main contribution
[22]	2019	GRU	Private	✗	✗	✗	Nguyen *et al.* proposed FL approach for device-type-specific anomaly detection in IoT networks.

Botnet Attacks

- Marai is a botnet that infected IoT devices to launch a DDoS reflection attack on the major DNS distributor Dyn.
  - Resulted in widespread connectivity issues for uptime sensitive websites such as Twitter, Spotify, and PayPal [3].
- Marai could launch Scan attacks, Ack-Syn, and UDP flood attacks. Marai is opensource posing a risk of future modification and extension into new IoT botnets.



Zero-Day Attacks

- Zero-day botnet attacks involve hackers exploiting unknown vulnerabilities in IoT systems
- Compromised computing devices form a network for these attacks
- Detection is challenging due to lack of prior knowledge
- FDL model aims to detect zero-day botnet attacks in IoT-edge devices

IOT Traffic

- Modern IoT networks are increasingly scalable
- Network constraints make offloading large IoT traffic data to remote cloud servers difficult
- Real-life use cases face challenges in processing data centrally due to these constraints

Proposed FDL Method

- Input parameters
  - R (Communication round)
  - E (Number of training iterations)
  - B (Batch size of training data)
  - K (Total number of DNN)
- Local Updates
  - Select some batches of data to perform gradient descent.
  - Update local weight matrix W_k
- Center server
  - Calculate average weight matrix W_r for each round

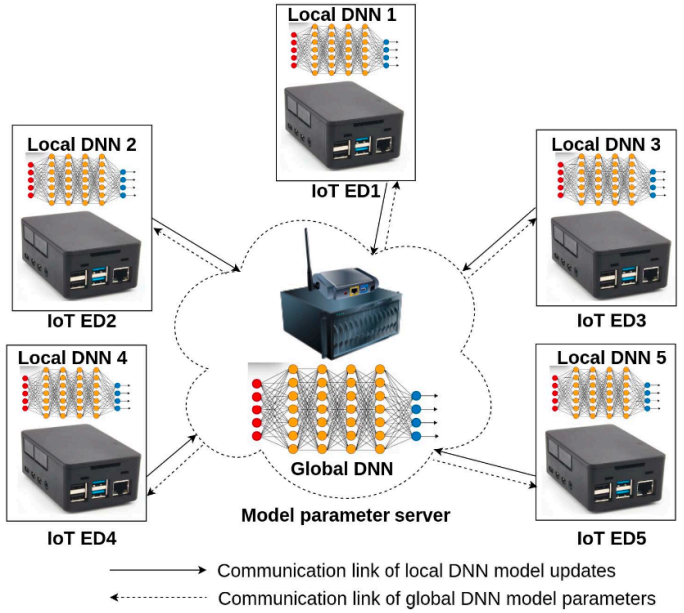
Algorithm 1: FDL Algorithm

Input: R, E, N, B, K
Initialization: $W = W_0$
Output: W_r

function `localUpdate(`W, k`):`
  for $e = 1$ *to* E **do**
    for $b = 1$ *to* $\frac{N}{B}$ **do**
      $W_{k,b} = W_{k,b-1} - \gamma \Delta L(b, W_k)$
    end
  end
  return W_k
end function
for $r = 1$ *to* R **do**
  for $k = 1$ *to* K **do**
    $W_{r,k}$ = localUpdate(W_{r-1}, k)
  end
  $W_r = \sum_{k=1}^{K} \frac{n_k}{N} W_{r,k}$
end

Model Development and Experiments

- Five IoT edge devices connect to Model parameter server
- Each edge IoT devices store localized data traffic within same network
- For each commination round, the five edge devices will update the local weight
- Model parameter server will aggregate the global weight matrix and send back to devices



Model Development and Experiments

In this study 4 models are used to test between:

- Central Deep Learning (CDL): A type of deep learning where data is collected and stored in a central location, usually a single server or cluster of servers.
- Localized Deep Learning (LDL): A type of deep learning where data is stored and processed locally, on individual devices such as smartphones or laptops.
- Distributed Deep Learning (DDL): A type of deep learning where data and processing are distributed across multiple devices or servers.
- Federated Learning (FDL):
  - In federated learning, the data remains on the devices, and the model is trained locally on each device.
  - The model is then sent to a central server, where the updates are combined to create a global model.

Model Development and Experiments

Popoola et al., used 477 benign IoT samples and 3,668,045 botnet attack samples as network traffic for their unique dataset.

They removed six redundant features from their dataset:

- 1) pkSeqID; 2) flgs; 3) proto; 4) state; 5) saddr; and 6) daddr.

Normalization is done across the 37 features that Popoola et al. measured.

$$\mathbf{x}_{\text{norm}} = \frac{\mathbf{x} - \mathbf{x}_{\text{min}}}{\mathbf{x}_{\text{max}} - \mathbf{x}_{\text{min}}} \quad (5)$$

Four classifications of performance were used to compare CDL, LDL, DDL, and FDL models:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \quad (6)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (7)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (8)$$

$$F1 = \frac{2 \times TP}{(2 \times TP) + FP + FN} \quad (9)$$

Results and Discussion

- The DNN model with four hidden layers and 100 hidden neurons per layer had the best overall classification performance for botnet attack detection in IoT-edge devices [2].
- The CDL model achieved excellent performance because it was trained with a large and diverse dataset; however, it lacks preserving privacy, has high training time, high communication cost, and high memory space usage [2].
- The LDL model preserved privacy, has low training time, low communication cost, and low memory space usage; however, performance is low because it was not trained with insufficient dataset and fewer botnet attack scenarios [2].
- The DDL model achieved a lower classification performance, since communication of local model updates from IoT-edge devices was limited to a single round [2].
- The FDL model outperformed the other models in terms of classification accuracy, training time, communication cost, and memory space requirement, while also preserving data privacy and reducing network latency [2].

Results and Discussion

Model	Classification Performance	Training Time	Data Privacy Preservation	Communication Cost	Memory Space Requirement	Network Latency
CDL	Good for DDoS, DoS, and Reconnaissance. Poor for Normal and Theft due to class imbalance.	High	No	High	High	High
LDL	Good for DoS, and Reconnaissance. Poor for DDoS and Theft due to class imbalance.	Medium	Yes	Medium	Medium	Medium
DDL	Good for DDoS. Poor for DoS, Reconnaissance, and Theft due to class imbalance.	Medium	Yes	Medium	Medium	Medium
FDL	Good for all classes.	Low	Yes	Low	Low	Low

Table 1 Comparison of different Deep Learning Models, adopted from [1; 2]

Conclusions and Future Work

- FDL model was developed with the Bot-IoT and N-BaIoT data sets
- Effectiveness of FDL was compared with CDL, LDL, and DDL models
- CDL model achieved high classification performance but did not preserve privacy and security of network traffic data
- LDL and DDL models overcame limitations of CDL, but their classification performance was low
- FDL model outperformed CDL, LDL, and DDL models, except for the long training time
- FDL method is most efficient for zero-day botnet attack detection in IoT-edge devices
- The future plan include exploring how advanced FL algorithms can improve the classification performance of attack detection models

References

1. 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," *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, Dallas, TX, USA, 2019, pp. 756-767, doi: 10.1109/ICDCS.2019.00080.
2. S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh and O. Jogunola, "Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT-Edge Devices," in *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3930-3944, 1 March1, 2022, doi: 10.1109/JIOT.2021.3100755.
3. "Financial Impact of Mirai DDoS Attack on Dyn Revealed in New Data." Corero, 03-Aug-2017. [Online]. Available: https://www.corero.com/financial-impact-of-mirai-ddos-attack-on-dyn-revealed-in-new-data/. [Accessed: 15-Mar-2023].
4. P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, “Long short term memory networks for anomaly detection in time series,” in Proceedings. Presses universitaires de Louvain, 2015, p. 89.
5. A. Nanduri and L. Sherry, "Anomaly detection in aircraft data using Recurrent Neural Networks (RNN)," *2016 Integrated Communications Navigation and Surveillance (ICNS)*, Herndon, VA, USA, 2016, pp. 5C2-1-5C2-8, doi: 10.1109/ICNSURV.2016.7486356.
6. M. Du, F. Li, G. Zheng, and V. Srikumar, “Deeplog: Anomaly detection and diagnosis from system logs through deep learning,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS ’17. ACM, 2017, pp. 1285–1298, http://doi.acm.org/10.1145/3133956.3134015.
7. A. Oprea, Z. Li, T.-F. Yen, S. H. Chin, and S. Alrwais, “Detection of early-stage enterprise infection by mining large-scale log data,” in Dependable Systems and Networks (DSN), *2015 45th Annual IEEE/IFIP International Conference on. IEEE*, 2015, pp. 45–56.
8. C. Krugel, T. Toth, and E. Kirda, “Service specific anomaly detection ¨ for network intrusion detection,” in *Proceedings of the 2002 ACM symposium on Applied computing*. ACM, 2002, pp. 201–208.
9. L. Portnoy, E. Eskin, and S. Stolfo, “Intrusion detection with unlabeled data using clustering,” in *Proceedings of ACM CSS Workshop on Data Mining Applied to Security*, 2001.
10. S. Rajasegarar, C. Leckie, and M. Palaniswami, “Hyperspherical cluster based distributed anomaly detection in wireless sensor networks,” *Journal of Parallel and Distributed Computing*, vol. 74, no. 1, pp. 1833–1847, 2014.
11. A. Kleinmann and A. Wool, “Automatic construction of statechartbased anomaly detection models for multi-threaded scada via spectral analysis,” in *Proceedings of the 2Nd ACM Workshop on Cyber-Physical Systems Security and Privacy, ser. CPS-SPC ’16.* ACM, 2016, pp. 1–12.
12. R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, and S. Zhou, “Specification-based anomaly detection: a new approach for detecting network intrusions,” in *Proceedings of the 9th ACM conference on Computer and communications security*. ACM, 2002, pp. 265–274.
13. Y. J. Jia, Q. A. Chen, S. Wang, A. Rahmati, E. Fernandes, Z. M. Mao, and A. Prakash, “ContexIoT: Towards providing contextual integrity to appified IoT platforms,” in *24th Annual Network & Distributed System Security Symposium (NDSS)*, feb 2017.
14. S. Raza, L. Wallgren, and T. Voigt, “Svelte: Real-time intrusion detection in the internet of things,” Ad hoc networks, vol. 11, no. 8, pp. 2661– 2674, 2013.
15. R. Doshi, N. Apthorpe, and N. Feamster, “Machine learning ddos detection for consumer internet of things devices,” CoRR, vol. abs/1804.04159, 2018. [Online]. Available: http://arxiv.org/abs/1804.04159
16. C. Krugel, T. Toth, and E. Kirda, “Service specific anomaly detection ¨ for network intrusion detection,” in Proceedings of the 2002 ACM symposium on Applied computing. ACM, 2002, pp. 201–208.
17. L. Portnoy, E. Eskin, and S. Stolfo, “Intrusion detection with unlabeled data using clustering,” in In *Proceedings of ACM CSS Workshop on Data Mining Applied to Security*, 2001.
18. S. Rajasegarar, C. Leckie, and M. Palaniswami, “Hyperspherical cluster based distributed anomaly detection in wireless sensor networks,” *Journal of Paralle*
19. M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, “Understanding the mirai botnet,” in *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017, pp. 1093–1110.
20. S. Edwards and I. Profetis, “Hajime: Analysis of a decentralized internet worm for IoT devices,” *Rapidity Networks, Tech. Rep*., 2016.
21. C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, “DDoS in the IoT: Mirai and other botnets,” *Computer*, vol. 50, no. 7, pp. 80–84, 2017.
22. Radware, “BrickerBot results in PDoS attack,” https://security.radware.com/ddos-threats-attacks/brickerbot-pdospermanent-denial-of-service/, 2017, [Online; accessed 8-April-2019]


Thanks for watching.
Questions?