



Research Repository UCD

Title	Federated Learning based Anomaly Detection as an Enabler for Securing Network and Service Management Automation in Beyond 5G Networks
Authors(s)	Jayasinghe, Suwani, Siriwardhana, Yushan, Porambage, Pawani, Liyanage, Madhusanka, Ylianttila, Mika
Publication date	2022-06-10
Publication information	Jayasinghe, Suwani, Yushan Siriwardhana, Pawani Porambage, Madhusanka Liyanage, and Mika Ylianttila. "Federated Learning Based Anomaly Detection as an Enabler for Securing Network and Service Management Automation in Beyond 5G Networks." IEEE, June 10, 2022. https://doi.org/10.1109/eucnc/6gsummit54941.2022.9815754 .
Publisher	IEEE
Item record/more information	http://hdl.handle.net/10197/13092
Publisher's statement	© 2022 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Publisher's version (DOI)	10.1109/eucnc/6gsummit54941.2022.9815754

Downloaded 2025-12-04 23:05:58

The UCD community has made this article openly available. Please share how this access benefits you. Your story matters! (@ucd_oa)



© Some rights reserved. For more information

Federated Learning based Anomaly Detection as an Enabler for Securing Network and Service Management Automation in Beyond 5G Networks

Suwani Jayasinghe*, Yushan Siriwardhana*, Pawani Porambage*, Madhusanka Liyanage^{†*}, Mika Ylianttila*,

*Centre for Wireless Communications, University of Oulu, Finland. email: {firstname.lastname}@oulu.fi

[†]School of Computer Science, University College Dublin, Ireland. email: madhusanka.liyanage@ucd.ie

Abstract—Network automation is a necessity in order to meet the unprecedented demand in the future networks and zero touch network architecture is proposed to cater such requirements. Closed-loop and artificial intelligence are key enablers in this proposed architecture in critical elements such as security. Apart from the arising privacy concerns, machine learning models can also face resource limitations. Federated learning is a machine learning-based technique that addresses both privacy and communication efficiency issues. Therefore, we propose a federated learning-based model incorporating the ZSM architecture for network automation. The paper also contains the simulations and results of the proposed multi-stage federated learning model that uses the UNSW-NB15 dataset.

Index Terms—5G, beyond 5G, Network automation, Security, Federated learning, ZSM

I. INTRODUCTION

Fifth generation (5G) mobile networks are already evolving while demanding diverse requirements, including bandwidth, latency, and reliability. As the networks become more complex, security is another aspect that has gained the most attention in the networking and telecommunication research community. Intelligent security and automated security management are key features considered in the vision of beyond 5G or 6G mobile networks [1]. Myriads of unprecedented challenges will arise during the deployment of such automated security solutions and their operations with little or no human intervention while catering to high-performance expectations.

The European Telecommunications Standards Institute (ETSI) proposed the Zero-touch Network and Service Management (ZSM) framework to meet the automation requirements of the network management [2]. The ZSM architecture, where the network is divided into domains based on the different requirements they have, such as operational, business, and technological, also incorporates Artificial Intelligence (AI) and Machine Learning (ML) algorithms along with closed loop operations [3]. The security considerations can be directly applicable to the general ZSM architecture and the security closed loop operations for attack detection, security analytics, cyber threat intelligence for attack mitigation or prevention, security orchestration, and security policy updates [4].

As the entry point to the security closed loop (i.e., for security monitoring and security data collection tasks), there should be a sufficient number of detectors empowered by

ML algorithms in the network inspired by the ZSM architecture. However, detectors will encounter constraints such as processing power and storage. We identify that federated learning (FL) is a relatively novel form of ML algorithm that allows decentralized processing with higher privacy and communication efficiency [5]. In FL, the models are trained on decentralized devices, which are then aggregated. FL methods can be directly applicable in networks with a ZSM architecture due to the domain-based definition with the end-to-end (E2E) service management domain overlooking all the other domains.

Therefore, in this paper, we use FL to create a hierarchical anomaly detection mechanism that can be applied to the ZSM architecture. In particular, we consider the reference security architecture proposed by the INSPIRE-5G plus project [6] to develop the anomaly detection mechanism that consists of two stages for network traffic analysis. Each stage has an FL-based detector that removes the network flows that are identified as anomalies. The complexity of the model and the size of the detector's database change depending on the stage. To the best of our knowledge, this is the first research article that proposes a hierarchical FL-based anomaly detection mechanism and validates it with simulations to support the automation of security management in the ZSM architecture. We train our model using the UNSW-NB 15 data set and test the trained model with pre-separated testing data from the same dataset. We conduct simulations for varying anomaly percentages in both stages 1 and 2 and demonstrate that the model reaches a minimum accuracy of 93.6%.

The remainder of the paper is organized as follows. Section II outlines the existing literature on FL-based anomaly detection that supports ZSM. Section III describes the proposed system model, and Section IV explains the simulations in detail. V discusses the simulation results and Section VI concludes the paper with future research directions.

II. BACKGROUND AND RELATED WORK

Security automation for network and service management is an emerging research area which is made possible by AI/ML related techniques together with closed-loop operation. Each management domain in the ZSM architecture has incorporated a data analytic service as a support function for closed-loop

operations for network optimization and security. The project funded by the European Union called "Intelligent security and pervasive trust for 5G and beyond" (INSPIRE-5Gplus) has proposed a security architecture for the ZSM architecture [6]. As a high-level security architecture, the INSPIRE-5G plus framework presents a fully automated network and service security management in the domain-based environment.

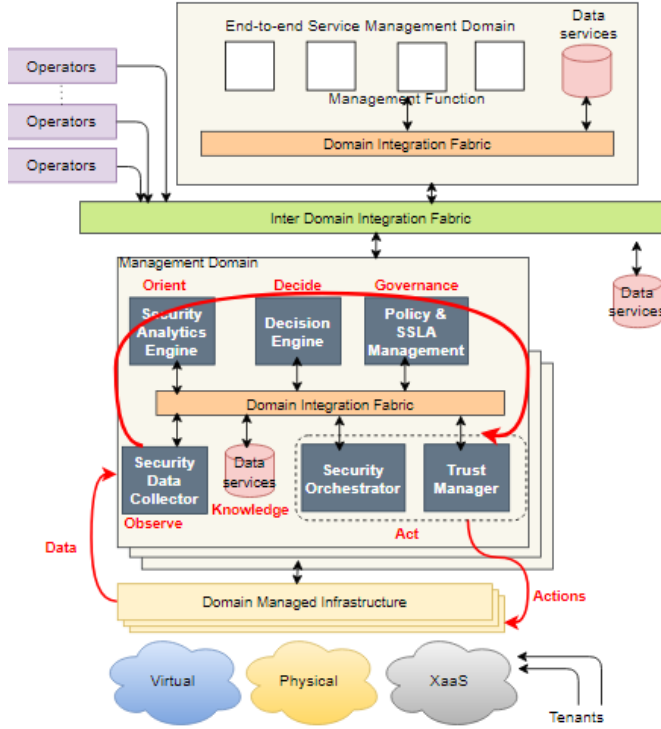


Fig. 1: INSPIRE-5G plus Security Framework - High Level Architecture (HLA) [6]

In accordance with the ZSM architecture, a management domain in Figure 1 consists of services and resources federated together based on different constraints such as operational, deployment, and functional. The end-to-end management domain manages the service or resource requests among such domains. The domain and inter-domain integration fabrics respectively facilitate the communication inside the domains and between domains and the E2E management domain. As given in Figure 1, the INSPIRE-5G plus security architecture of a management domain contains a security data collector, a security analytics engine, a security orchestrator, a decision engine, a SLA manager, and a trust manager.

In the ZSM architecture, closed loop operations are a critical enabler of network automation, network optimization, and behavior adaptation [2]. As illustrated in Figure 1, the domain level security closed loop has five stages called observe, orient, decide, act, and knowledge [6]. In a closed loop, data is observed and collected from relevant resources, which are then oriented so that their meaning and significance can be understood. Analysing or orientation may include filtering, correlation, or any other similar mechanism based on application. Decisions are taken using AI/ML mechanisms

with their capabilities to recognize patterns or trends and make predictions. The decision-making component also makes plans, determines root causes, or does similar processes based on the circumstances, which are then implemented. It is also possible to collaborate with other closed loops based on the situation [2].

As shown in Figure 1, the security agents residing in multiple locations within the network collect data and submit it to the security data collector, which observes the data. The security analytic service performs data processing and data analysis. The inferred information is then provided to the decision engine, which has the capability to decide the course of action required to operate the network optimally, mitigating any risk signalled by the data. Subsequently, the security orchestrator will enforce the recommendations of the decision engine on the relevant entities. The consequences will then be observed by the data analytic engine through the data collected by security agents and security data collectors [6]. As Benzaid et al. have mentioned, open API security threats, intent-based interface security threats, security threat-driven network automation, artificial intelligence, ML-based attacks, and security threats due to the adaptation of programmable network technologies are the threat surfaces in the ZSM architecture.

According to the proposed architecture, AI models are also deployed for network resource optimization and proactive and reactive incident analysis services in the domain analytic service, in addition to security. Service management can be empowered by AI and Rizwan et al. [7] demonstrate methodology that employs AI-enabled CDR analysis for service management in ZSM networks. They have used k-means clustering, support vector mechanisms, and their own algorithm for identifying suboptimal network performances and root causes.

Wang et al. [8] have presented a hierarchical FL anomaly detection mechanism for industrial IoT devices, and the local model at each device is trained by deep reinforcement learning. Nguyen et al. [9] also propose a FL-based solution for anomaly detection in IoT applications, and their mechanism is based on calculating the probabilities related to the communication patterns of devices. Yadav et al. [10] also present an unsupervised FL-based IoT intrusion detection system. They have used an auto encoder and an ANN model for detection and the CICIDS 2017 data set for testing and training the proposed model. Apart from IoT applications, Wei et al. [11] present an FL-based anomaly detection solution for 5G heterogeneous networks that consists of three main parts: user end devices, the edge, and the cloud. They have also used reinforcement learning based anomaly detection in devices deployed in the mentioned locations, and the parameter server is located in the cloud. Liu et al. [12] use a FL framework to collaboratively train a Deep Anomaly Detection (DAD) model in industrial IoT applications. Li et al. [13] present DeepFed, which is a system for utilizing federated deep learning for intrusion detection in industrial cyber-physical systems. They have used the Paillier encryption mechanism and FL to develop a scheme that collaborates with multiple industrial CPS owners to build

a deep learning intrusion detection model while preserving privacy. The proposed model in [14] is a deep learning-based self-adaptive algorithm for detecting anomalies with two stages. A simple anomaly classification is done by a combination of a flow collector and an ML-empowered classifier. If it suspects that an anomaly has happened, network flow will be redirected to a network anomaly detection system placed in the evolved packet core. Monitors placed in the network for anomalies are utilized for their proposed self-adaptive algorithm.

As discussed here, most of the studies have been conducted extensively to develop FL-based anomaly detectors for IoT-based applications. There are very few applications introduced for areas concerning the security of automated networks or the ZSM architecture. However, FL-based solutions can be effectively used for security applications while adhering to privacy and communication efficiency. To fill the void left by the lack of an FL-based detector for security in automated networks, we have presented a detector that effectively employs the FL-based model for detecting network anomalies.

III. PROPOSED MODEL

Our proposed model is an FL-based hierarchical security scheme that detects anomalies in network traffic in two stages. Each of the two stages has a different ML model, and the idea is inspired by two reference security schemes presented in [11], [14]. Each anomaly detector is accompanied by an FL-based model and a database, along with two types of parameter aggregating servers, which are positioned for detection. As illustrated in Figure 2 and in accordance with the ZSM architecture, the aggregation server for anomaly detectors is performed at the management domain level and the E2E management domain level.

The network flow that enters the network with the goal of reaching its destination will pass through two anomaly detectors. The first anomaly detector in the security management domain has a simple ML model with a relatively smaller database, allowing for a sufficient number of detectors to be placed throughout the network while meeting any network resource constraints. All of the A-type anomaly detectors are linked to the aggregation server located in the security management domain. The portion of the network flows that are confirmed to be anomalous will be removed by stage 1, anomaly detector A, and the remaining data will be analyzed at stage 2, detector B. For the purpose of the simulation, a neural network was used for both detectors, as it generated sufficient accuracy. However, the proposed system imposes no restrictions on other network types.

In stage 2, detector B has a more complex ML model and a relatively larger database compared with stage 1, detector A. Stage 2, detector B, placed higher in the hierarchy of the proposed system, serves a group of lower-level anomaly detectors (stage 1, detector A), making it possible to function with a low number of detectors. The aggregation servers of stage 1 and stage 2 detectors are respectively placed in the management domains and the E2E management domain. The filtered flow from stage 1 will undergo another filtering

mechanism in stage 2. If there is any flow that is confirmed to be an anomaly, it will be immediately removed and only the remaining flow will be continued to its destination.

A training mechanism, in addition to the detection mechanism, will be implemented within the network to generate the network model used at each detector. This procedure is similar to that of an FL-based model, in which model-generated parameters are aggregated at the servers according to a predefined criterion. Initially, each anomaly detector will store a set of data in the database, and the model parameters will be computed by training the model, which will be shared with the aggregation server. As proposed in the algorithm, the aggregation server will collect the parameters from each detector and calculate the average of the parameters, which will then be sent back to the detectors as a new set of parameters that can be used for detection.

The ZSM architecture is expected to experience unprecedented events, and it is possible that it will be subjected to unknown security threats. In the event that an unknown anomaly is observed that is filtered through both the anomaly detectors, we propose a mechanism to update the detectors accordingly within the closed-loop cycle such that recurrence does not take place.

The security data analytics component has the potential to generate network alarms and notifications on the network based on predefined criteria. If the anomaly detector fails to identify a disturbance, an alarm will be generated in the security data analytic component in the domain where the event has taken place. The security data analytics component will notify both the stage 1 aggregation server and the E2E security orchestration engine. The aggregation server will instruct the detectors to capture the network flows within the duration of the event that occurred and to train the model again. The E2E security orchestration engine will notify both the security analytics engines of other domains and the stage 2 aggregation server. The stage 2 aggregation server will request the detectors to train the model. The security analytics engine in other domains will scan the domain for anomalies and begin training if an anomaly is identified.

IV. SIMULATION

This section describes the setup we used for the simulation of a two-stage anomaly detection mechanism. The dataset UNSW-NB 15 was used for model training and testing in simulation [15]. Each stage in the system needs an FL-based model that will be used for detection, and first, these models need to be developed and trained. Then the generated models were used for the detection process. The parameters, such as the anomaly percentage of the data used for training the models, were varied to observe the effect of each on the overall accuracy of the proposed system. Tensorflow federated libraries were used for the simulation.

There are 257673 data flows in the UNSW-NB 15 dataset. Before starting the simulation, a randomly selected 10,000 samples of the data set were allocated for testing, and the rest of the data was used for training the two models. The

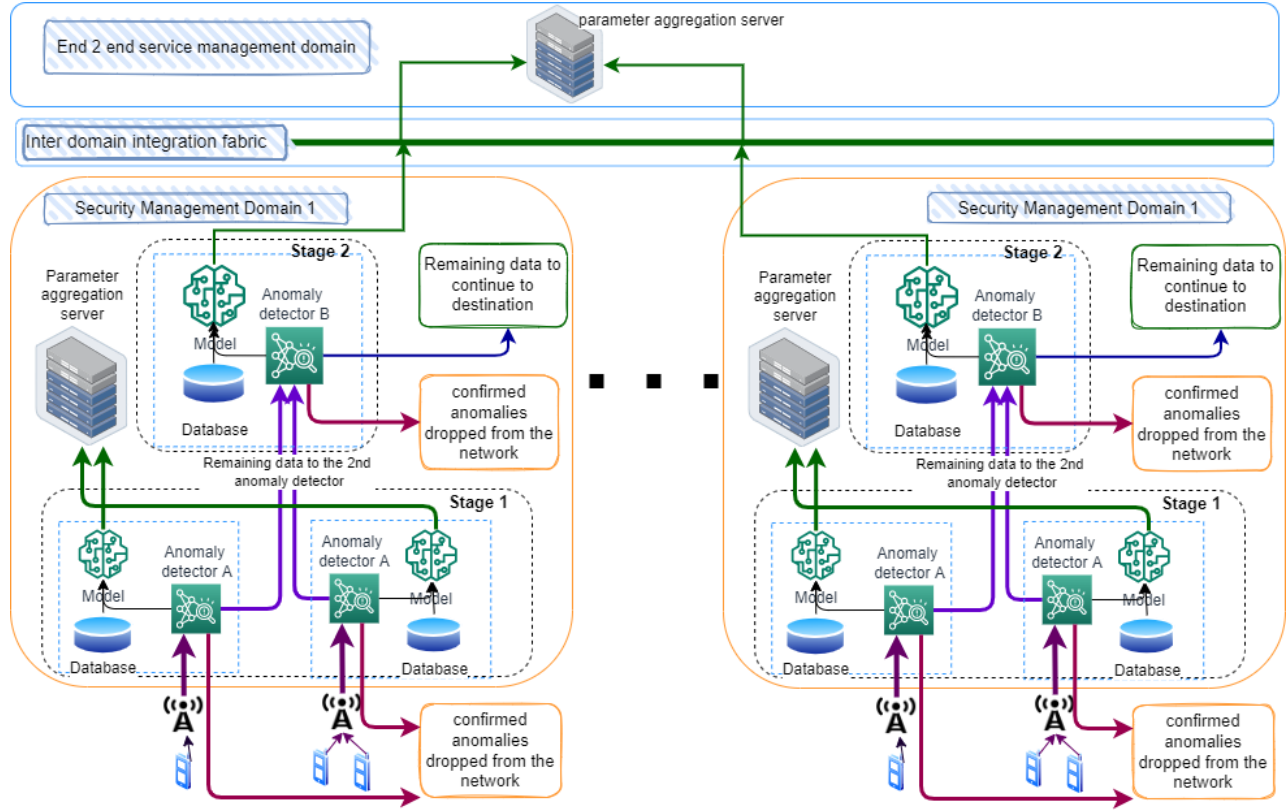


Fig. 2: Proposed hierarchical FL based anomaly detection mechanism and its mapping with the ZSM architecture.

number of samples used for training stage 1 is in the range of 6000–14000, and around 99,000 samples were used for training the stage 2 model. The data set has 42 features of nine types of attacks along with normal traffic. The features include source and destination IP addresses, source and destination port numbers, protocols, destination to source and source to destination transaction bytes, etc.

FL was used to train the models in stages 1 and 2. Devices (also known as clients) have a small data set and train models for a small number of local rounds. The number of rounds is also known as epochs. Then each client shares parameters with a central server, which aggregates the model parameters and redistributes them to the client. Clients train the model again, starting from the model created based on the parameters sent by the central server, and they again share the parameters with the central server. A simulation round is defined as the process of generating model parameters from clients, sharing them with the central server, and aggregating them. 100 such rounds were used for each case.

The ML model used for stage 1 has four layers, with two hidden layers. For training model 1, the number of data samples was independently and identically divided into ten groups. Each client trains a model upon receiving a set of data, and the parameters generated are shared with a central server. Each client uses five epochs for training. The central server aggregates the parameters and calculates the average of the parameters and sends them back to the devices. We have recorded the model parameters for 100 such rounds for

each case. Cases are generated by varying the percentage of anomalies in the training data set for stages 1 and 2.

Stage 2 was approached in a similar manner. The second-stage model has six layers with four hidden layers, and 99,200 samples are used for training. Therefore, each client gets around 9920 data flows, which are used for training with five epochs to generate model parameters. In the next step, it will be shared with the central server for aggregation. 100 such rounds are used for each case, which is generated by changing the anomaly percentage of the training data.

In the final stage, after generating the model for both stages for different cases, test data was sent through both stages for detection. A model for each stage was chosen from parameter sets generated during the training. A set of data that is confirmed to be anomalies will be dropped from stage 1, and only the data that is categorized as normal traffic will be sent to the second stage. In the second stage, the data is again categorized as an anomaly or normal. The table I shows how the overall testing accuracy varies with the anomaly percentage in the training data on stages 1 and 2. Table II demonstrates how the overall system accuracy changes from stage 1 to 2.

Figure 3 shows how the overall system accuracy after 100 rounds behaves when the anomaly percentage of the training data in stage 1 is varied. Anomaly percentages in the training data of stages 1 and 2 in all cases used for the Figure 3 and figure 4 are shown in the table III.

The figure 4 shows how the overall accuracy of the model changes when the percentages of anomalies in the training data

TABLE I: Overall Testing accuracy against anomaly data percentage in training data stage 1 and stage 2.

Anomaly data percentage in training data of Stage 2	Anomaly data percentage in training data-Stage 1					
	20%	30%	40%	50%	60%	70%
20%	90.5%	91.0%	92.0%	92.2%	92.4%	91.9%
30%	91.9%	92.3%	92.4%	92.8%	92.8%	91.3%
40%	92.7%	92.6%	92.7%	92.6%	92.7%	91.9%
50%	93.2%	93.4%	93.2%	93.1%	92.7%	91.9%
60%	93.6%	93.5%	93.4%	93.2%	93.0%	91.7%
70%	93.0%	93.1%	93.0%	92.8%	92.7%	91.6%

93.1-:94.0%
 92.1-93.0%
 90.0-92.0%

TABLE II: Overall Testing accuracy increase after stage 2 against anomaly data percentage in training data stage 1 and stage 2.

Anomaly data percentage in training data-Stage 2	Anomaly data percentage in training data-Stage 1					
	20%	30%	40%	50%	60%	70%
20%	3.6%	1.7%	1.5%	0.8%	0.4%	0%
30%	5.7%	3.6%	2.3%	1.3%	0.7%	0.3%
40%	6.4%	3.6%	2.7%	1.6%	0.8%	0.3%
50%	6.5%	3.6%	2.2%	1.9%	0.9%	0.6%
60%	5.5%	4.4%	3.1%	1.5%	1.1%	0.4%
70%	6.3%	4.3%	2.6%	1.5%	0.1%	-0.5%

-1.0-1.0%
 1.1-3.0%
 3.0-5.0%
 5.0-7.0%

of the stage 2 model are varied.

The figure 5 shows how the testing accuracy changes along with the number of rounds when the number of training samples in stage 1 is varied.

V. DISCUSSION

As per the data shown in the table I, when the amount of anomaly in stage 1 training data is varied from 20–70%, accuracy has peaked when the percentage is around 50–60% for most of the cases. However, when the stage 2 anomaly percentage is set at 60%, the accuracy value initially increases, and it gradually reduces when the stage 1 percentage is increased. We have included Figure 3 to show how the overall accuracy of the model changes along with the number of rounds when only the anomaly percentage of training data in stage 1 is varied. It shows behavior similar to the pattern shown in the table I throughout 100 rounds. When the stage 1 anomaly data rate is maintained at a fixed value and the anomaly ratio of training data in stage 2 is varied, accuracy shows a maximum value of 60% and it drops slightly afterwards. The figure 4 shows how the accuracy behaves during 100 rounds. The same patterns observed in the table I can be seen in the figure 4 as well.

Testing data was selected randomly from the set of data, of which 50 – 60% of the data were anomalies. Therefore, testing data is expected to have the same distribution. We believe that accuracy has increased when the training data for stage 1 has the same composition. Maximum accuracy is shown

TABLE III: Set of cases shown in Figure 3 and 4

Case	Anomaly data percentage in training data	
	Stage 1	Stage 2
Case ST1:20 ST2:60	20%	60%
Case ST1:30 ST2:60	30%	60%
Case ST1:40 ST2:60	40%	60%
Case ST1:50 ST2:60	50%	60%
Case ST1:60 ST2:60	60%	60%
Case ST1:70 ST2:60	70%	60%
Case ST1:20 ST2:20	20%	20%
Case ST1:20 ST2:30	20%	30%
Case ST1:20 ST2:40	20%	40%
Case ST1:20 ST2:50	20%	50%
Case ST1:20 ST2:60	20%	60%
Case ST1:20 ST2:70	20%	70%

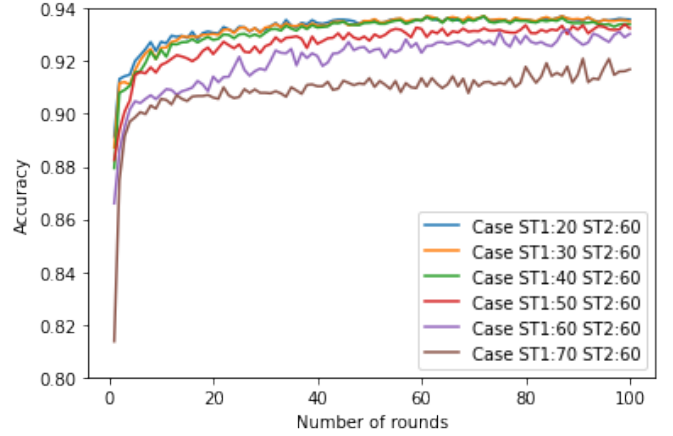


Fig. 3: Overall Testing accuracy against anomaly data percentage in training data stage 1

when stage 2 percentage is around 60%. Number of training samples in stage 2 is 99,200, which is 10 times the number of data samples in stage 1. We believe that when 60% anomalies are taken, the majority of the anomalies are included in the training data, making it possible to detect a maximum number of anomalies accurately. When the anomaly percentage is 70%, not enough normal data is captured in the training data. Hence, the accuracy decreases. The Table II shows that overall system accuracy has increased after the two stages, and the increment takes a higher value when the anomaly data ratio of stage 1 is lower.

As per the results, an adequate number of cases of different anomaly cases should be included in the training data. The method proposed allows having a larger database at the second stage, which allows covering all possible scenarios. Therefore, even if any abnormal flow is sent to the 2nd stage marked as normal data, the 2nd stage is capable of detecting it. Thus, the two-stage model enhances the detector accuracy while preserving the privacy of the data and maintaining communication efficiency. Apart from the advantages inherited from using the FL model, it allows catering to inevitable resource constraints as well.

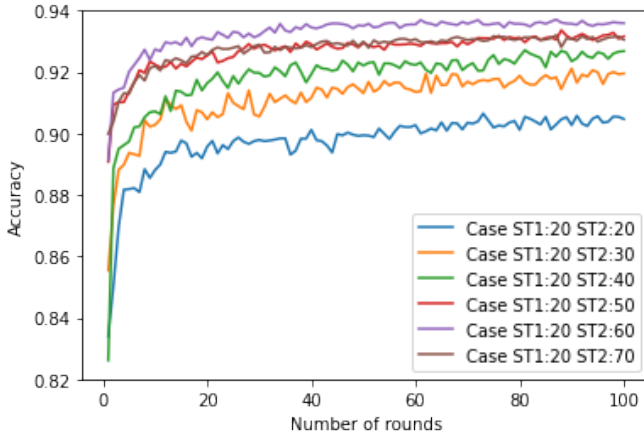


Fig. 4: Overall Testing accuracy against anomaly data percent in training data stage 2.

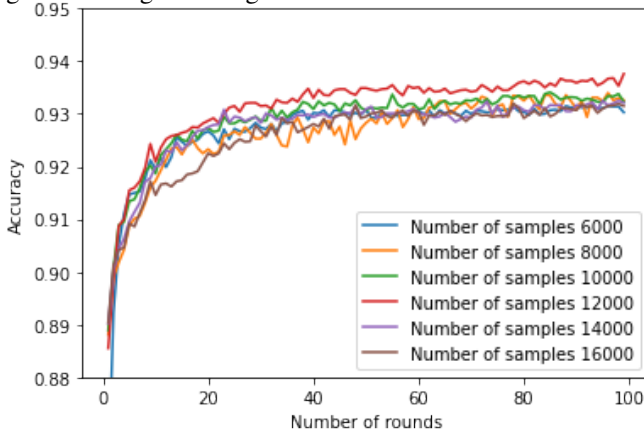


Fig. 5: Testing accuracy vs Number of rounds obtained from varying the number of Training samples in stage 1

VI. CONCLUSION

FL is a distributed ML technique with the features of enabling privacy and enhancing communication efficiency. In this paper, we presented a multi-stage model developed with FL to identify the anomalies and their applicability in the ZSM architecture. Moreover, we simulated the proposed system using the networking dataset UNSW-NB 15. The presented results demonstrate how the system accuracy has changed when the composition of the training data changes. It is also visible from the results how the two-stage model improves the accuracy. In particular, the composition of training data of the used model should cover enough cases to facilitate adequate detection. In the current setup, the maximum accuracy we received was around 94%. In future work, we expect to increase the accuracy of the model and apply it in a security analytics framework in the ZSM security architecture.

ACKNOWLEDGMENT

This work is supported by 6Genesis Flagship (grant 318927) project. The research leading to these results partly received funding from European Union's Horizon 2020 research and innovation programme under grant agreement no 871808 (5G PPP project INSPIRE-5Gplus) and 101021808 (H2020 SPATIAL project). The paper reflects only the authors' views. The Commission is not responsible for any use that may be made of the information it contains.

REFERENCES

- [1] P. Porambage, G. Gür, D. P. M. Osorio, M. Liyanage, A. Gurtov, and M. Ylianttila, "The roadmap to 6g security and privacy," *IEEE Open Journal of the Communications Society*, vol. 2, pp. 1094–1122, 2021.
- [2] ETSI, "Zero-touch Network and Service Management (ZSM)," ETSI GS ZSM 002 - Reference Architecture, Aug 2019.
- [3] Y. Siriwardhana, P. Porambage, M. Liyanage, and M. Ylianttila, "AI and 6G Security: Opportunities and Challenges," *2021 IEEE Joint European Conference on Networks and Communications (EuCNC) 6G Summit*, 2021, pp. 1–6, 2019.
- [4] C. Benzaid and T. Taleb, "ZSM security: Threat Surface and best Practices," *IEEE Network*, vol. 34, no. 3, pp. 124–133, 2020.
- [5] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang, and D. Niyato, "Federated Learning for 6G Communications: Challenges, Methods, and Future directions," *China Communications*, vol. 17, no. 9, pp. 105–118, 2020.
- [6] C. Benzaid, P. Alemany, D. Ayed, G. Chollon, M. Christopoulou, G. Gür, V. Lefebvre, E. M. de Oca, R. Muñoz, J. Ortiz, A. Pastor, R. Sanchez-Iborra, T. Taleb, R. Vilalta, and G. Xilouris, "White paper intelligent security architecture for 5g and beyond networks," 2020. [Online]. Available: https://zenodo.org/record/4288658#Y1_KY9pBxPY
- [7] A. Rizwan, M. Jaber, F. Filali, A. Imran, and A. Abu-Dayya, "A zero-touch network service management approach using ai-enabled cdr analysis," *IEEE Access*, vol. 9, pp. 157 699–157 714, 2021.
- [8] X. Wang, S. Garg, H. Lin, J. Hu, G. Kaddoum, M. J. Piran, and M. S. Hossain, "Towards accurate anomaly detection in industrial internet-of-things using hierarchical federated learning," *IEEE Internet of Things Journal*, pp. 1–1, 2021.
- [9] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A.-R. Sadeghi, "Dfot: A federated self-learning anomaly detection system for iot," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, 2019, pp. 756–767.
- [10] K. Yadav, B. Gupta, C.-H. Hsu, and K. T. Chui, "Unsupervised federated learning based iot intrusion detection," in *2021 IEEE 10th Global Conference on Consumer Electronics (GCCE)*, 2021, pp. 298–301.
- [11] Y. Wei, S. Zhou, S. Leng, S. Maharjan, and Y. Zhang, "Federated learning empowered end-edge-cloud cooperation for 5g hetnet security," *IEEE Network*, vol. 35, no. 2, pp. 88–94, 2021.
- [12] Y. Liu, N. Kumar, Z. Xiong, W. Y. B. Lim, J. Kang, and D. Niyato, "Communication-efficient federated learning for anomaly detection in industrial internet of things," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.
- [13] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "Deepfed: Federated deep learning for intrusion detection in industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5615–5624, 2021.
- [14] L. Fernandez Maimo, A. L. Perales, Gomez, F. J. Garcia Clemente, M. Gil Perez, and G. Martinez Perez, "A self-adaptive deep learning-based system for anomaly detection in 5g networks," *IEEE Access*, vol. 6, pp. 7700–7712, 2018.
- [15] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6.