Enhancing Federated Learning with Homomorphic Encryption and Multi-Party Computation for improved privacy

Pedro Tomás*†, Samira Kamali Poorazad‡, Chafika Benzaïd‡, Luis Rosa*, Jorge Proença*,
Tarik Taleb§, and Luis Cordeiro*
*{pedro.tomas, luis.rosa, jorge.proenca, cordeiro}@onesource.pt *{samira.kamalipoorazad,chafika.benzaid}@oulu.fi
*tarik.taleb@rub.de
* OneSource, Consultoria Informática Lda., Portugal
† Department of Informatics Engineering, University of Coimbra, Portugal
‡ Faculty of Information Technology and Electrical Engineering, University of Oulu
§ Faculty of Electrical Engineering and Information Technology, Ruhr University Bochum, Bochum, Germany

***Abstract*—Federated Learning (FL) has demonstrated substantial promise in distributed machine learning by enabling collaborative model training without sharing raw data. However, FL systems still face challenges related to privacy and security. This paper evaluates the impact of incorporating Privacy-Enhancing Techniques (PETs) into a Federated Learning architecture conceived for Network Anomaly Detection.**

This work provides two-fold contributions: the impact of Homomorphic Encryption and Multi-Party Computation on model classification, considering various types of attacks, validated using a widely known dataset, the CIC-IDS2017 dataset, and the performance and resource analysis when compared to a classical Federated Learning architecture (without using PETs).

The results show that using Multi-Party Computation presents a residual increase in computational resources (from 0.998MB to 1.417MB of RAM consumption, on average) accompanied by a significant increase in privacy for the different agents involved and even a slightly increased performance achieved by the detection module. Such results offer insights into the trade-offs between security benefits and computational overhead, providing valuable guidelines for optimizing federated learning systems with integrated privacy protections.

***Index Terms*—federated learning; anomaly detection; enhance privacy; multi-party computation; homomorphic encryption;**

I. Introduction

Google introduced Federated Learning in 2016 as an innovative form of decentralised learning with a strong focus on the client's privacy. One of the main ideas behind this concept is that the original data never leaves the device and is solely used to train the Machine Learning (ML) model locally.

In an era of distributed systems, protecting such vast networks poses a serious challenge. Detecting security threats and network traffic anomalies is of paramount importance to promptly identify malicious traffic [1], [2]. The Holistic Security and Privacy Framework (HSPF) [3] is an FL-based tool for network traffic anomaly detection to identify anomalies in kubernetes-based scenarios.

Considering the importance of data security and privacy, this work proposes and evaluates the addition of two Privacy Enhancing Techniques to the HSPF: Homomorphic Encryption (HE) and Multi-Party Computation (MPC), specifically from resource consumption and performance points of view. The evaluation was divided into training and testing phases: the first focused on understanding the impact of adding HE and MPC on resource consumption; the second aimed to assess the impact of adding these techniques to the HSPF processes on its ability to detect network traffic anomalies. For the training phase, a simple scenario was created in a Kubernetes environment where agnostic requests were generated to an httpbin server, and the traffic was collected and the HSPF detection module learnt the characteristics of these requests. Next, in the testing phase, a dataset containing 60% of traffic collected from the training phase and 40% of malicious traffic (attacks) randomly retrieved from the CIC-IDS2017 [4] dataset was used to evaluate the performance of the trained models.

The experimentation showed that using HE significantly increased resource consumption (i.e., RAM) during the training phase. On the other hand, the testing phase showed better results when compared with a scenario without PETs. The addition of MPC resulted in a slight increase in resource consumption for the training and testing phases. The performance model yielded the best results when compared to the three other scenarios, with the best model achieving an f1-score of 56.28%.

II. Related Work

Motivated by their crucial role in enhancing data privacy and security, there has been a growing research interest in integrating encryption-based PETs, such as HE and MPC, with FL to protect shared model updates. This section provides an overview of existing studies, with a particular emphasis on the computation and accuracy degradation issues associated with the integration of encryption-based PETs.

A. HE-based FL Systems

Authors in [5] highlight the capability of HE to maintain model accuracy close to non-encrypted models while optimising communication and computational efficiency. Using Cheon-Kim-Kim-Song (CKKS) HE scheme with an aggregated public key in their method ensures that individual model updates remain confidential without significantly degrading classification performance. Authors in [6] propose a method named Paillier Federated Multi-Layer Perceptron (PFMLP). PFMLP integrates HE with FL to secure gradient data during transmission, while maintaining model accuracy. The optimised Paillier algorithm used in PFMLP enhances training efficiency, though the increased computational overhead due to encryption poses a challenge, potentially slowing down the training process. In [7], HE is combined with secure MPC to protect data privacy during training in IoT-enabled healthcare systems. The introduction of a dropout-tolerable mechanism and a weighted average algorithm based on data quality helps preserve model accuracy while reducing computation and communication overhead. However, the complexity of implementation and the reliance on multiple cryptographic techniques may pose challenges in practical deployment. The study in [8] introduces a secure FL framework that integrates HE and Verifiable Computing (VC) to protect the confidentiality and integrity of model training, particularly in scenarios involving a small number of reliable clients, such as cross-silo settings. While this approach maintains model accuracy and provides strong privacy and integrity guarantees, the increased complexity and resource demands make it less suitable for environments with limited computational power, especially when compared to simpler methods that do not offer comprehensive security. The PL-FedIPEC scheme proposed in [9] integrates HE with FL to enhance privacy and reduce computational overhead in edge computing, utilizing an optimized Paillier encryption algorithm. However, the scheme may face efficiency challenges in high communication or frequent model update scenarios. In [10], a Buffered Federated Learning (BFL) framework is presented for privacy-driven anomaly detection in IIoT environments. A key goal of BFL is to address the limitations of traditional FL methodologies, such as the straggler effect, communication bottlenecks, and privacy vulnerabilities. BFL achieves this by utilizing HE to enhance privacy, and implementing an innovative client selection strategy to balance the straggler effect and the communication bottleneck. As a result, BFL provides a more efficient and secure FL approach for IIoT applications than state-of-the-art methods in terms of accuracy, convergence speed, and privacy preservation.

HE consistently enhances privacy in FL systems across various domains. It effectively preserves model accuracy and offers strong privacy guarantees. However, the trade-offs include increased computational overhead, communication costs, and implementation complexity [11]. These factors vary depending on the specific application and the type of HE used, with some methods achieving better balance between privacy and performance than others. The choice of HE scheme ultimately depends on the specific needs of the deployment environment, particularly concerning resource availability and the frequency of model updates.

B. MPC-based FL Systems

The authors in [12] propose a Partially Encrypted MPC (PEMPC) method for FL, aimed at reducing the heavy communication and computation costs typically associated with standard MPC while preserving privacy. The proposed approach selectively encrypts critical model parameters/gradients, focusing on the first hidden layer to prevent data leakage during model aggregation. Despite its efficiency improvements, the method's complexity and potential challenges in dynamic environments with frequent updates are noteworthy. The CE-Fed framework proposed in [13] introduces a new hierarchical model aggregation method to tackle the high communication cost and scalability issues in MPC-enabled FL. By grouping clients based on geographical proximity and electing leaders for model aggregation, CE-Fed reduces communication overhead by $80 - 90\%$ while maintaining high model accuracy. Despite its merits, the proposed approach may face implementation challenges due to its complexity. The work in [14] proposes an augmented MPC method to enhance the security of FL against indirect gradient leakage. The proposed method consists in a two-round model decomposition that ensures the central server receives only a biased version of the model, protecting against data reconstruction. Despite the advantage of MPC in maintaining model accuracy while providing stronger privacy guarantees, it introduces additional communication and computation overhead.

The reviewed methods significantly enhance FL by improving communication efficiency, reducing computation costs, and strengthening privacy through secure MPC techniques. Each approach balances performance with security, though challenges like complexity and scalability persist. While these methods offer robust privacy protections, they often introduce trade-offs in resource consumption, highlighting the need for further optimisation in practical implementations.

III. Proposed Approach

Fig. 1 presents the proposed approach for integrating the PETs (i.e., Homomorphic Encryption and Multi-Party Computation) within the HSPF Architecture.

Tailored to cloud-native environments based on Kubernetes, the HSPF is composed of five major components: the HSPF Core, the Agent, the Collector, the Dashboard and the Policy Enforcer. Deployed as sidecars within the same pod as the protected application, the Collector and Agent components have distinct roles. The Collector captures inbound and outbound network traffic and is responsible for feature extraction, first by aggregation communications in network flows and then by computing various related statistics. The Agent performs network anomaly detection through inference on the extracted features and engages in continual learning to adapt the machine learning model to evolving network traffic patterns. In parallel, the Agent also participates in federated training rounds to ensure the model stays up to date with the most recent data patterns. The Dashboard offers comprehensive observability into the protected cloud environment, featuring graphical representations of components, network

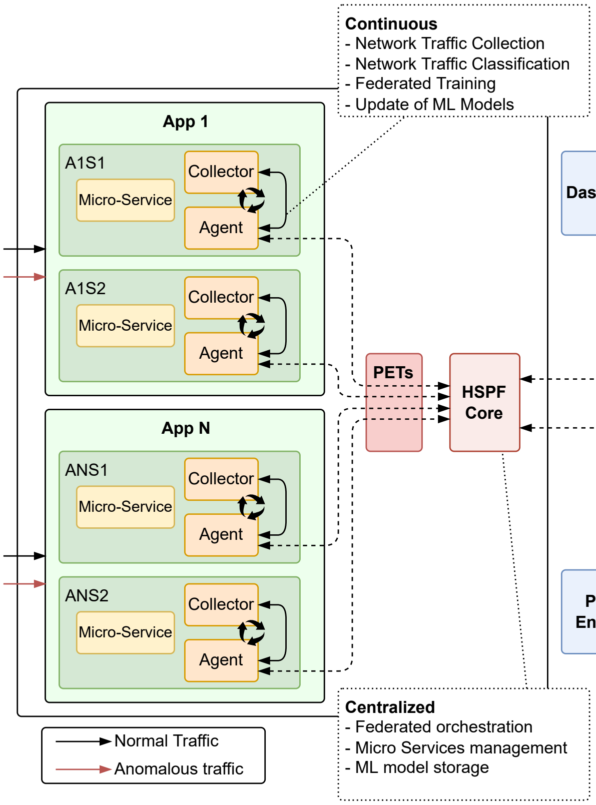

Fig. 1. HSPF components and functionalities

flows, and classification statistics. The Policy Enforcer enables the enforcement of policies over the protected environment using Istio Service Mesh [15] and Open Policy Agent [16]. For additional details on the baseline architecture and implementation of HSPF, please refer to [3].

A. HSPF with HE

Homomorphic Encryption (HE) has been implemented using the CKKS encryption schema [5], leveraging its compatibility with operations involving float numbers. Fig. 2 presents the implementation of HE within HSPF.

The first step involves using a trusted third party to generate a secure public-private key pair. These keys are then distributed to all agents protecting a specific application (in the figure, two agents protect two instances of the same application). During the federated learning (FL) process, after completing local training, each agent encrypts its model weights using the public key and sends the encrypted weights to the Aggregator (HSPF Core). The Aggregator then aggregates the encrypted weights without decrypting them and sends the aggregated result back to the agents. Finally, the agents decrypt the aggregated weights using their private keys, accessing the real values to continue FL iterations or receive the aggregated model. This process ensures the Aggregator never accesses the raw weights, preserving both the privacy of the model and the original data by preventing any attempt at data reconstruction.

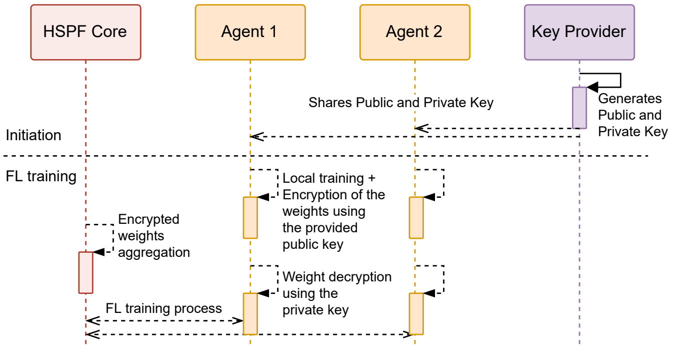
HSPF Core
Agent 1
Agent 2
Key Provider
Shares Public and Private Key
Generates Public and Private Key
Initiation
FL training
Local training + Encryption of the weights using the provided public key
Encrypted weights aggregation
Weight decryption using the private key
FL training process

Fig. 2. HE implementation within HSPF.

B. HSPF with MPC

MPC enables the secure aggregation of model weights. Secure aggregation is a technique where data is encrypted and aggregated locally before sharing, ensuring privacy. It improves traditional federated learning by ensuring that the model weights are not shared in raw format, thus preventing potential unwanted data breaches (e.g., where the attacker could potentially try to reconstruct the original data from the model weights). Fig. 3 presents the implementation of MPC within HSPF.

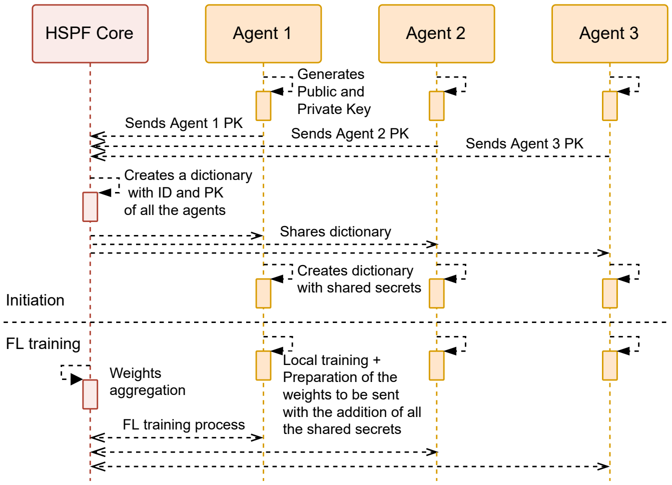
HSPF Core
Agent 1
Agent 2
Agent 3
Generates Public and Private Key
Sends Agent 1 PK
Sends Agent 2 PK
Sends Agent 3 PK
Creates a dictionary with ID and PK of all the agents
Shares dictionary
Creates dictionary with shared secrets
Initiation
FL training
Weights aggregation
Local training + Preparation of the weights to be sent with the addition of all the shared secrets
FL training process

Fig. 3. MPC implementation within HSPF.

Two phases were considered: the initiation and training phases. In the initiation phase: (i) each HSPF Agent generates Public and Private Keys using the Diffie-Hellman [17]; (ii) the HSPF Core elaborates a dictionary containing the IDs of each Agent and their respective public keys and then shares it with all the HSPF Agents; (iii) after receiving the dictionary, each agent calculates the shared secret which is used to mask their parameters during the training phase; (iv) all agents update their dictionary with the value of the shared secrets previously calculated. Each agent generates the shared secret by signing the public keys of the other Agents with its private key.

During the training phase, its Agent considers all the other

Agent IDs and shared secrets to add or subtract the value of its shared secret (usually an integer), considering if the other Agent IDs are greater or lower, respectively.

IV. Evaluation Methodology

The following aspects were considered to evaluate the impact of adding PETs into the HSPF architecture: federated training time, RAM consumption and overall algorithm performance. The first two were evaluated during the training phase, whilst the second was evaluated during the testing phase. The training and testing phases, described throughout this section, were repeated three times: the first, where no PET has been used; the second, where Homomorphic Encryption has been applied; and, the third, where Multi-Party Computation was implemented.

A. Training Phase

Fig. 4 presents the training scenario, which was prepared in a cloud-native environment using Kubernetes [18], [19]. Three different namespaces were considered: requests, server and hspf. The first (requests) uses a TCP random requests client to perform HTTP requests to the HTTP server, deployed in the second namespace (server), which was serving four HTTP-based websites. Following the sidecar pattern, the HSPF Agent and the HSPF Collector have been deployed next to the TCP random requests and next to the HTTP server. The third namespace (hspf) hosts the HSPF core components.

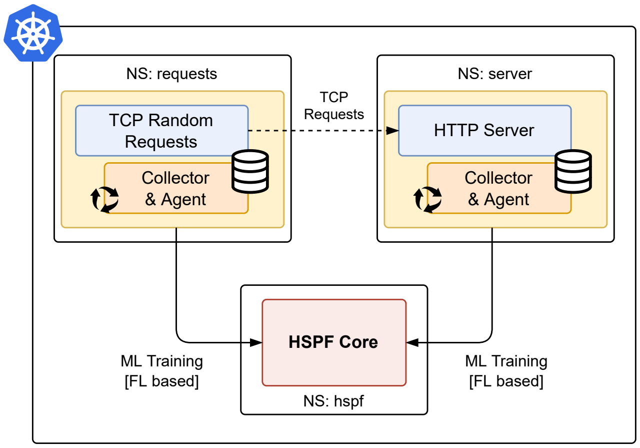

Fig. 4. Training Scenario.

Throughout the ML training iteration process, 100 models were trained, and the evaluation was later conducted with a sub-set of 10 models, selected in equidistant form (i.e., 10, 20, 30, 40, 50, 60, 70, 80, 90, 100). For each model, the threshold has been calculated as stated in formula 1, considering the Mean Squared Error (MSE) calculated between the original value of each feature and the correspondent value reconstructed by the Autoencoder. Such principle enables the use of Autoencoders to perform network anomaly detection, by taking into account the reconstruction error of the autoencoder, assumed to be higher whenever compared with an unseen sample (i.e., network anomaly). Before considering the threshold and aiming to reduce the effect of potential outliers, the up-and-lower fence strategy was applied – as an outlier removal approach – by considering only the MSE values that fall within the first and third quartile of the cumulative distribution formed by the MSE values.

$$t = \overline{x}_{MSE} + 3\sigma_{MSE} \tag{1}$$

As previously stated, the focus during the training phase was to analyse the impact of adding Homomorphic Encryption (HE) and Multi-Party Computation (MPC), considering the FL training time and the RAM consumption. As such, the same training scenario was used for the three evaluated scenarios (without any PET, with HE and with MPC), with the respective differences in the implementation level of the HSPF components and respective communication channels.

B. Testing Phase

Fig. 5 presents the testing methodology. The models trained and the network flows generated during the training phase were extracted from the training environment.

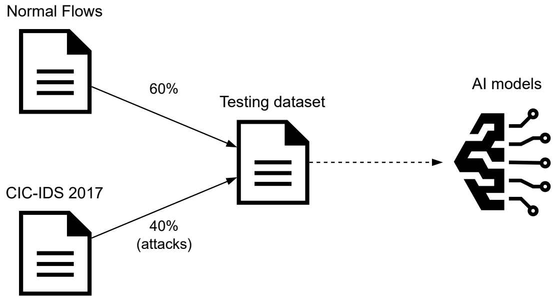

Fig. 5. Testing Methodology.

The 10 algorithms trained for each PET scenario, were later tested with a custom dataset, which was prepared containing a ratio of 60% of normal flows, generated during the training phase, and 40% of attack samples, randomly extracted from the CIC-IDS2017 dataset. For each performance evaluation, the following metrics were registered: accuracy, precision, recall and f1-score.

V. Results

This section presents the results, separated into training and testing phases.

A. Training Phase

The training results are divided into three categories: without PETs, with HE, and finally with MPC. Each result details the threshold, time, and RAM consumed for each model.

Table I presents a summary of the results obtained during the training phase in the absence of PETs (baseline) in comparison with HE and MPC methods.

The results show a decreasing tendency for the threshold throughout the models part of the same sub-set, for the three scenarios, which is aligned with the assumption that the MSE value decreases over time as per the model continually evolves

TABLE I
MODEL PERFORMANCE COMPARISON BETWEEN BASELINE, HE AND MPC METHODS.

#	Threshold			Time(s)			RAM (MB)		
	Base	HE	MPC	Base	HE	MPC	Base	HE	MPC
1	0,49	0,40	0,47	2,26	5,26	5,23	1,61	174,25	3,17
2	0,09	0,08	0,07	1,97	4,64	3,39	1,18	174,36	2,46
3	0,11	0,32	0,07	2,08	5,05	2,14	1,04	174,32	1,21
4	0,19	0,06	0,08	1,93	4,60	1,38	0,65	174,37	0,90
5	0,07	0,08	0,10	2,04	4,64	1,67	1,12	174,35	1,28
6	0,08	0,06	0,07	1,15	4,70	1,30	0,85	174,38	0,92
7	0,09	0,04	0,06	1,72	5,45	1,41	0,91	174,44	0,91
8	0,04	0,06	0,06	1,28	5,86	2,18	0,89	174,49	0,95
9	0,04	0,05	0,08	1,34	5,24	1,95	0,90	174,35	1,37
10	0,06	0,05	0,07	1,22	4,88	1,61	0,73	174,25	1,01
$\overline{x}$	0,13	0,12	0,11	1,70	5,03	2,23	0,99	174,36	1,42

throughout the different training iterations. The training time is higher for the scenarios with HE and MPC, compared to the one where no PET was used. The RAM consumption is significantly higher for the scenario with HE, which may be explained by the extra steps involved in each training iteration when using this technique.

B. Testing Phase

Table II presents the results obtained during the testing phase for the algorithms trained without using PETs.

TABLE II
BASELINE EVALUATION

Model	Accuracy	Precision	Recall	f1-score
1	0,6001	0,0000	0,0000	0,0000
2	0,6172	0,6159	0,5788	0,5472
3	0,6788	0,7412	0,4875	0,5419
4	0,6002	0,5592	0,0794	0,0919
5	0,4994	0,4272	0,5737	0,4727
6	0,5264	0,4465	0,5634	0,4819
7	0,5815	0,5557	0,4452	0,4642
8	0,3601	0,3446	0,6804	0,4526
9	0,3454	0,3288	0,6250	0,4273
10	0,5416	0,4823	0,5880	0,5079
$\overline{x}$	0,5351	0,4501	0,4622	0,3988

Table III presents the results obtained during the testing phase for the algorithms trained with the use of Homomorphic Encryption.

For the scenario without the use of PETs, the model with the best performance achieved an f1-score of 54.72%, while the model with the worst performance achieved an f1-score of 0% (which occurs when the model did not identify any of the malicious samples). Such took place for the first model of the sub-set, and later improved for the remaining models, while the models continued to train and adapt to the network traffic characteristics.

Table IV presents the results obtained during the testing phase for the algorithms trained using Multi-Party Computation.

TABLE III
EVALUATION WITH HE.

Model	Accuracy	Precision	Recall	f1-score
1	0,4218	0,4104	0,7155	0,5022
2	0,5829	0,5536	0,6772	0,5696
3	0,6000	0,0000	0,0000	0,0000
4	0,5198	0,4724	0,7360	0,5510
5	0,5327	0,4675	0,5581	0,4835
6	0,4732	0,3968	0,6063	0,4695
7	0,3769	0,3368	0,5903	0,4263
8	0,4531	0,3794	0,5873	0,4514
9	0,4222	0,3687	0,6153	0,4539
10	0,4944	0,4374	0,6036	0,4891
$\overline{x}$	0,4877	0,3823	0,5690	0,4396

For the scenario with the HE, the results show that the model with the best performance achieved an f1-score of 56.96%, with an average f1-score for the 10 sub-set of models close to 44%.

TABLE IV
EVALUATION WITH MPC.

Model	Accuracy	Precision	Recall	f1-score
1	0,6239	0,2000	0,0597	0,0876
2	0,5122	0,4797	0,7647	0,5628
3	0,5259	0,4710	0,6290	0,5109
4	0,5542	0,5196	0,5181	0,4829
5	0,6229	0,6315	0,5102	0,5156
6	0,4799	0,4029	0,6007	0,4711
7	0,4691	0,3817	0,5272	0,4385
8	0,4532	0,3793	0,5864	0,4511
9	0,6091	0,5612	0,5255	0,5097
10	0,6378	0,5870	0,5297	0,5368
$\overline{x}$	0,5488	0,4614	0,5251	0,4567

For the scenario with the MPC, the considered 10 models achieved an average f1-score of 45.67%, with the best performance model reaching 56.28% and the worst model achieving 8.76%.

VI. DISCUSSION

Adding Homomorphic Encryption and Multi-Party Computation to the HSPF architecture reflects into an increase in the federated training time and RAM consumption. Despite this, the cost of these two techniques is not the same. Using HE is far more expensive in resource consumption when compared to MPC, which is mainly explained by the overhead caused by the encryption and decryption of the traffic shared between the HSPF Agent and the HSPF aggregation mechanism.

The results attained during the testing phase revealed that the models trained using PETs yielded better results, with the most significant difference rounding to 5.79% (on average), between the performance of the models trained with MPC and those that were trained without any PET.

Given the need to improve the privacy of federated learning approaches, such as HSPF, MPC exhibits the best performance-cost balance, showing a slight increase in resource consumption during the training phase and an improved performance during the testing phase.

VII. Conclusion

Federated Learning has changed the paradigm of training ML algorithms in distributed systems. The ability to train ML models locally, ensuring data privacy and security concerns, has proven paramount.

Nevertheless, the standard versions of FL present some vulnerabilities that may allow unwanted access to the model weights, enabling malicious reconstruction of the original clients' data from the ML model weights shared throughout the network. This paper reports on the application of two PETs (MCP and HE) into the HSPF framework, a FL-based network anomaly detection framework.

The obtained results reveal the feasibility of using PETs next to the HSPF, both from a resource consumption point of view, as well as from a detection module performance (with the best module presenting 56.28% of f1-score during the testing phase, and an average of 1.42MB of RAM consumption, during the training phase, with the application of MPC).

Acknowledgment

This work was supported in part by the EU's HE research and innovation programme HORIZON-JU-SNS-2022 under the RIGOUROUS project (Grant No. 101095933). The views expressed in this contribution are those of the authors and do not necessarily represent the project nor the Commission.

References

[1] O. Hireche, C. Benzaïd, and T. Taleb. Deep Data Plane Programming and AI for Zero Trust Self-Driven Networking in Beyond 5G. *Elsevier Journal on Computer Networks*, page 108668, Feb. 2022.

[2] T. Taleb, C. Benzaïd, R. Addad, and K. Samdanis. AI/ML for Beyond 5G Systems: Concepts, Technology Enablers & Solutions. *Elsevier Journal on Computer Networks*, page 110044, Dec. 2023.

[3] Pedro R. Tomas, Pedro Felix, Luis Rosa, Andre S. Gomes, and Luis Cordeiro. A novel approach for continual and federated network anomaly detection. London, United Kingdom, Nov. 2024. *FTC 2024 - Future Technologies Conference 2024*, accepted.

[4] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *International Conference on Information Systems Security and Privacy*, 2018.

[5] Jing Ma, Si-Ahmed Naas, Stephan Sigg, and Xixiang Lyu. Privacy-preserving federated learning based on multi-key homomorphic encryption. *International Journal of Intelligent Systems*, 37(9):5880–5901, 2022.

[6] Haokun Fang and Quan Qian. Privacy preserving machine learning with homomorphic encryption and federated learning. *Future Internet*, 13(4), 2021.

[7] Li Zhang, Jianbo Xu, Pandi Vijayakumar, Pradip Kumar Sharma, and Uttam Ghosh. Homomorphic encryption-based privacy-preserving federated learning in iot-enabled healthcare system. *IEEE Transactions on Network Science and Engineering*, 10(5):2864–2880, 2023.

[8] Abbass Madi, Oana Stan, Aurélien Mayoue, Arnaud Grivet-Sébert, Cédric Gouy-Pailler, and Renaud Sirdey. A secure federated learning framework using homomorphic encryption and verifiable computing. In *2021 Reconciling Data Analytics, Automation, Privacy, and Security: A Big Data Challenge (RDAAPS)*, pages 1–8, 2021.

[9] Chunrong He, Guiyan Liu, Songtao Guo, and Yuanyuan Yang. Privacy-preserving and low-latency federated learning in edge computing. *IEEE Internet of Things Journal*, 9(20):20149–20159, 2022.

[10] S. Kamali Poorazad, C. Benzaïd, and Tarik Taleb. A novel buffered federated learning framework for privacy-driven anomaly detection in IIoT. In *Proc. of the IEEE Global Communications Conference (Globecom)*, Cape Town, South Africa, Dec. 2024.

[11] C. Benzaïd and T. Taleb. AI for Beyond 5G Networks: A Cyber-Security Defense or Offense Enabler? *IEEE Network Magazine*, 34(6), Nov./Dec. 2020.

[12] Ekanut Sotthiwat, Liangli Zhen, Zengxiang Li, and Chi Zhang. Partially encrypted multi-party computation for federated learning. In *2021 IEEE/ACM 21st International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pages 828–835, 2021.

[13] Renuga Kanagavelu, Qingsong Wei, Zengxiang Li, Haibin Zhang, Juniarto Samsudin, Yechao Yang, Rick Siow Mong Goh, and Shangguang Wang. Ce-fed: Communication efficient multi-party computation enabled federated learning. *Array*, 15:100207, 2022.

[14] Chi Zhang, Sotthiwat Ekanut, Liangli Zhen, and Zengxiang Li. Augmented multi-party computation against gradient leakage in federated learning. *IEEE Transactions on Big Data*, pages 1–10, 2022.

[15] Istio. https://istio.io/. Accessed: 2024-07-20.

[16] Open policy agent. https://www.openpolicyagent.org/. Accessed: 2024-07-20.

[17] Ueli M. Maurer and Stefan Wolf. The diffie–hellman protocol. *Designs, Codes and Cryptography*, 19(2):147–171, Mar 2000.

[18] Amir Javadpour, Forough Ja'fari, Tarik Taleb, Chafika Benzaid, Luis Rosa, Pedro Tomas, and Luis Cordeiro. Deploying testbed docker-based application for encryption as a service in kubernetes. In *Proc. of the 32nd Int'l Conf. on Software, Telecommunications and Computer Networks (SoftCOM 2024)*, 2024. Split - Bol, Croatia.

[19] T. Theodoropoulos, L. Rosa, P. Gray C. Benzaid, E. Marin, A. Makris, L. Cordeiro, F. Diego, P. Sorokin, M. Di Girolamo, P. Barone, T. Taleb, and K. Tserpes. Security in cloud-native services: A survey on key features. *J. Cyber Security and Privacy*, 3(4):758–793, 2024.