Federated Learning For Cyber Security: SOC Collaboration For Malicious URL Detection

Ekaterina Khramtsova
SNT
University of Luxembourg
Luxembourg
ekaterina.khramtsova@uni.lu

Christian Hammerschmidt
Department of Intelligent Systems
TU Delft
Delft, The Netherlands
0000-0003-2460-1997

Sofian Lagraa
SNT
University of Luxembourg
Luxembourg
sofian.lagraa@uni.lu

Radu State
SNT
University of Luxembourg
Luxembourg
radu.state@uni.lu

***Abstract*—Managed security service providers increasingly rely on machine-learning methods to exceed traditional, signature-based threat detection and classification methods. As machine-learning often improves with more data available, smaller organizations and clients find themselves at a disadvantage: Without the ability to share their data and others willing to collaborate, their machine-learned threat detection will perform worse than the same model in a larger organization. We show that Federated Learning, i.e. collaborative learning without data sharing, successfully helps to overcome this problem. Our experiments focus on a common task in cyber security, the detection of unwanted URLs in network traffic seen by security-as-a-service providers. Our experiments show that i) Smaller participants benefit from larger participants ii) Participants seeing different types of malicious traffic can generalize better to unseen types of attacks, increasing performance by 8% to 15% on average, and up to 27% in the extreme case. iii) Participating in Federated training never harms the performance of the locally trained model. In our experiment modeling a security-as-a service setting, Federated Learning increased detection up to 30% for some participants in the scheme. This clearly shows that Federated Learning is a viable approach to address issues of data sharing in common cyber security settings.**

***Index Terms*—cyber-security, federated-learning, machine-learning**

I. Introduction

Managed security service providers (MSSPs) with offers ranging from security operating centers (SOCs) as a service to managed detection and response (MDR) services enable smaller organisations to keep up with the increasing threat surface brought by the ongoing digital revolution across industries. Their goal is to protect mission-critical data and assets, respond to cyber emergencies, and provide continuity and efficient recovery. Increasingly, machine-learning methods are key factors in surpassing traditional, signature-based threat detection and classification methods. As machine learning often improves with more data available, smaller organizations and clients find themselves at a disadvantage: Without the ability to share their data and others willing to collaborate, their machine-learned threat detection will perform worse than the same model in a larger organization. Unfortunately, sharing raw data of different customers is problematic: this data can contain confidential information about users such as personally identifiable information (PID). Legal and contractual restrictions, as imposed in the European Union (GDPR [1]) make it difficult to build a common shareable model to detect threats.

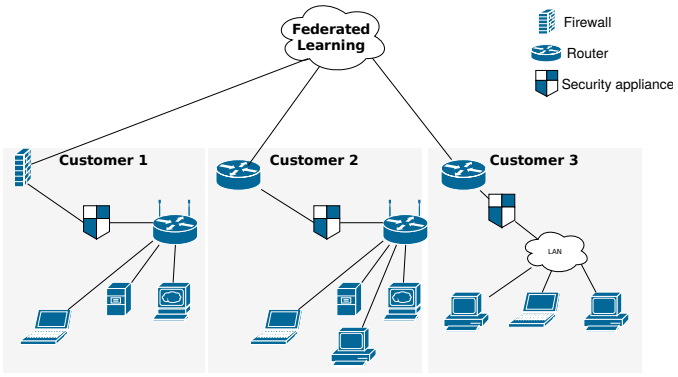

Fig. 1. The scenario considered here: A SOC/MSSP with three clients running on a range of services on slightly different infrastructure. The goal is to build a shared threat detection model that improves security over learning separate models for each customer, but without sharing data from different customers.

How can we collectively learn a better model than the one a customer/organization could learn on its own, without having to share data or other confidential information? We show that Federated Learning is a valid approach to resolve this question. We focus on a scenario where a managed security service provider with multiple customers cannot share the customers' data to build a shared model from data directly, but can share models learned for each customer separately. Figure 1 shows an example of this scenario.

We apply our Federated Learning approach to malicious URL detection. The goal is to distinguish URLs leading to phishing sites, defacement sites, malware-spreading sites, or spam sites from URLs leading to legitimate sites. Classifying malicious URLs serves as a prototype of the problems faced in cyber security because it shares a number of important properties commonly encountered in cyber security: It is strongly imbalanced, contains several classes with distinct characteristics, is hard to solve purely by blacklisting and rule-based methods, and is non-stationary over time. Within cyber security finding malicious URLs is a problem of value in itself: by identifying potentially dangerous sites, users can

be proactively protected.

The contributions of this paper are as follows:

1) We collected an extensive URL dataset and extracted features from URLs. We have made this dataset publicly available[1]. 2) We proposed a Federated Learning-based approach for malicious URL detection in a managed security service provider setting. We modeled several scenarios taking into consideration the characteristics of the data, such as class balance and class distribution, together with various sizes of customer infrastructure. 3) We showed that the small providers can achieve up to 30% performance improvement when trained in collaboration, while the big providers become more stable and generalize better with no cost compared to a locally-trained model.

The rest of the paper is organized as follows. Section II introduces Federated Learning, malicious URL detection and related literature. Section III explains our setup, data splits, and experiments. Section IV summarizes the obtained results. Section V concludes our paper and describes potential future work.

II. Background and related work

Federated Learning (FL). FL is a Machine Learning concept for decentralizing training. It helps to preserve the participants privacy by allowing them to collaboratively train a model without ever sharing their data. This approach is particularly appealing to the holders of highly sensitive data, such as medical records, financial information, private conversations, and other PID.

One of the most common optimisation methods for FL is the Federated Averaging algorithm [2], where each client (*agent*) trains a *local* model on his *local* data for several epochs, using Stochastic Gradient Descent (SGD). Note that the structure of the model across agents is the same. Then the central server selects a subset of agents, collects their local models and averages them. This aggregated model is then distributed back and used for the next local training at each agent. Local training, agent selection, model gathering, model aggregation and model distribution form one round (*global epoch*) of federated training.

Various modifications of the aggregation mechanism exist. For example, in Federated Averaging (FedAvg) the priority is given to the agents holding the most data. [3] introduces Attentive Aggregation method. q-Fair FL [4] encourages a fairer accuracy distribution across agents. [5] proposes Probabilistic Federated Neural Matching to either match local parameters and aggregate them, or dynamically extend the network structure.

The complexity and convergence of FL has been analyzed thoroughly. We refer to [6] for details.

In cyber security, research on Federated Learning often focuses on the vulnerability of Federated Learning itself, e.g. via data and model poisoning, or on privacy benefits of Federated Learning schemes [7], [8]. For the sake of simplicity, we note that the solutions proposed in these works also apply to our model, even though we do not present a system incorporating these, as the focus of our work is to show the benefit of cooperation itself.

Recently, numerous FL-based applications have been proposed for healthcare [9], autonomous driving [10], mobile applications [2], keyword spotting [11], The authors of [12] propose to apply FL-based malware defense mechanism by analysing individual logs in cloud ecosystems.

Malicious URL detection. Within cyber security, the task of distinguishing malicious URLs from benign URLs is a well studied problem. Existing proposals relying on centralized learning use several techniques. These range from pattern mining-based [13] and deep learning-based approaches [14] to semantic information extraction from URLs [15], [16]. Using machine-learning outperforms rule-based approaches, such as static blacklisting of domains. In practice, lists and rule-based systems are often used in combination with machine-learning approaches. The former capture well-known threats from threat-intelligence providers and the latter detects yet unknown but suspicious cases.

The survey [17] provides an overview of existing approaches. In general, research on machine-learning approaches takes the following steps:

1) URL collection and feature extraction. Features typically include lexical information, extracted host information, and content extraction of the target. Lexical features such as URL length, the words used in the URL, and bi-grams are most commonly used.
2) Algorithm design. Both supervised and unsupervised approaches in batch and streaming-setting exist.

Learning with an imbalanced distribution of data is a classic problem in the field of cyber security [18].

III. Federated Learning for malicious URL classification

As outlined in Figure 1, we propose a collaboration scheme for SOCs, MSSP, and small organisations to collaboratively classify malicious URLs using Federated Learning.

The main goal of the paper is to showcase how Federated Learning can be beneficial for the customers.

Our proposed solution does not take into account adversarial actors and other deployment considerations. Instead, we design experiments to answer the following questions:

- How much performance is lost by distributing training instead of learning a single centralized model on the data from all the participants?
- How does the number of agents affect the training of a model?
- What happens if agents have very different data?
- What would be the performance in a realistic environment?

The rest of this section is organised as follows:
We begin with a description of our Federated Learning setup

[1] https://github.com/khramtsova/url_feature_extractor/raw/master/urls_final_complete.tar.xz

used in all the experiments. We further proceed with defining 3 types of data partitioning, that reflect different scenarios and expose the strengths and weaknesses of FL.

A. Federated Learning setup

We introduce the following modification to the Federated Averaging algorithm described in Section II. We assume that all agents behave fairly and follow the protocol. Therefore the server does not perform agent selection - everyone can participate.

For the structure of the model we chose a fully connected feed-forward neural network with 72 input neurons. It consists of 2 hidden Linear layers with Relu with the output dimensions equal 64 and 32 respectively. We apply a dropout after the second layer with a keep probability of $p = 0.2$. Finally, the last output layer is Linear with Logarithmic softmax.

The network is trained with Cross Entropy loss, SGD optimizer with the learning rate equal 0.01.

The evaluation of the FL is done through the accuracy calculated over a test set, that has not been seen by any agent during training. The accuracy is calculated *after* the aggregation of the models by the server, which means that all agents have the same test accuracy.

B. Data partitioning

We conducted several experiments to assess the capabilities and limitations of Federated Learning.

Assume that $x \in X$ are the feature vectors, extracted from URLs; $\hat{y} \in \{0, 1\}$ are the labels to be predicted - benign or malicious. Even though the main task represents a binary classification problem, the malicious class can be further categorised based on the type of attack: $y \in \{$Defacement, Phishing, Malware, Spam$\}$. y is not needed for training, however it might be used as one of the parameters that defines the federated environment.

Our Federated Learning setup consist of k agents, where each of the agents draws samples from it's local data distribution: $(x, y) \sim P_i(x, y)$. One of the main difficulties of FL training lies in the uneven data distribution among participants. In order to mimic real-world scenarios, we performed multiple data splits, including independent and identical distribution (IID) and non-independent and identical distribution (non-IID). The experiments are grouped into three categories based on the splitting criteria:

- ***Category 1: IID***
  The data is evenly split between workers and the label distribution is preserved. In reality, identical distribution is almost never achievable; it serves as a best case benchmark of Federated Learning and the starting point of the analysis.
- ***Category 2: Label-based split***
  The dataset is split by the type of attack, while the amount of benign data in each agent is balanced. For example, one agent has only malicious instances of the type malware, while another agent has only phishing attacks. This models scenarios where the security provider has clients experiencing different types of threats. Private consumers, public institutes, banks, and corporations experience different kinds of attacks.
- **Category 3: Size-based split**
  We define small, medium and large agents based on the size of the dataset they possess. For all the agents, the samples are drawn independently on sub-labels y. This models a security provider with various customer sizes.

A more detailed description of the proposed partitions and their relevance for our use-case is provided in the following section.

IV. Experimental Results

We performed our experiments on University of Luxembourg HPC Iris cluster [19] using 1 GPU Tesla V100 16G and 1 CPU Intel Xeon Gold 6132. Our solution has been implemented in Python3 with the help of PyTorch and PySyft library [20] for federated communication.

A. Dataset overview

A large dataset of more than 700K malicious and benign URLs was collected from various sources. We began with a dataset from [21] consisting of 110 000 URLs, equally distributed between malware, defacement, phishing, spam and benign classes. This equal size distribution did not reflect a real-world scenario. Therefore, we decided to augment some classes. In particular, the benign class was enlarged with Hacker News post URLs[2]; data from [16] and [22]. Moreover, 70K samples of malware URLs were taken from URLHaus[3]. Finally, the phishing class was extended with the samples from OpenPhish[4] and with valid phishes from PhishTank [5] gathered for 4 consecutive days.

A detailed overview of the final constructed dataset is shown in Figure 2.

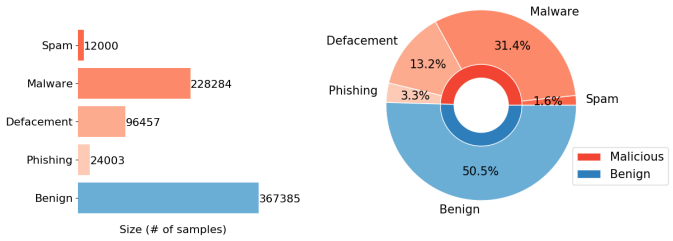

Fig. 2. Overview of the dataset. Total number of items per class (left) and relative distribution (right).

During pre-processing, we reproduced the feature extraction mechanism from [21] and retrieved 72 basic lexical features from the entire URL string and separately from each URL component, such as hostname, path, directory, filename, extension and query. Retrieved features include:

[2]https://kaggle.com/hacker-news/hacker-news-posts
[3]https://urlhaus.abuse.ch/
[4]https://openphish.com/
[5]https://www.phishtank.com/

- Number of letters, symbols, digits, dots, delimiters, dashes and tokens
- Length and length ratios between URL components, e.g. length(path)/length(query)
- Number of letter-digit-letter (ldl) and digit-letter-digit (dld) sequences
- Character entropy and character continuity rate
- Binary features, e.g. IsPortEighty; IsIPAddressInDomain

Note that all the features are static and dictionary and language-independent. Thus feature extraction can be easily parallelized between agents without introducing any ambiguity. For reproducibility purposes, we publish the feature extraction script.[6]

We split our dataset of benign and malicious URLs into three categories to model scenarios in which a SOC combines data from customers with a range of different characteristics.

B. Category 1: IID

We started with the identically distributed setting and a small number of participants, where the data is equally split between workers and the label distribution is preserved (Figure 3).

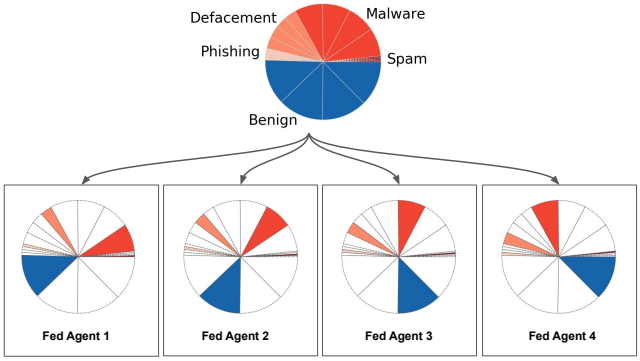

Fig. 3. Showing the distribution of samples in the label-based data split

We evaluated the training loss and accuracy on each agent after 2 local epochs. We further performed aggregation of the gradients, followed by testing on a subset of the initial dataset that had not been seen by any of the participants. The test set follows the same uniform label distribution as the training subsets and represents 10% of the whole dataset.

In order to evaluate the advantages of collaborative learning, federated training was compared to centralized and local trainings. Due to the data homogeneity across the agents, they all received similar results while training locally without participation in collaborative learning. For this reason, we only report the mean of their local accuracy. The results are shown in Fig.4.

In the case where all the available data is assembled by the centralized agent, an accuracy of 89% is obtained after only one training epoch. The federated training shows worse results

[6]https://github.com/khramtsova/url_feature_extractor

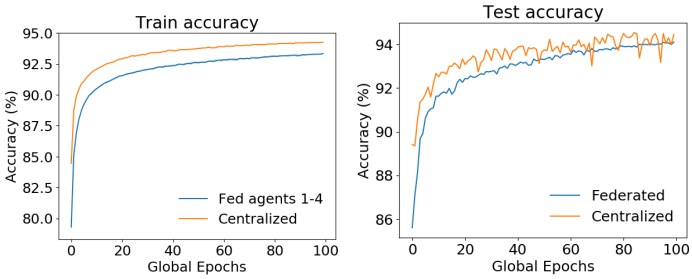

Fig. 4. Comparing the performance of a model with an equal-sized IID data split, 4 agents. *Centralized* shows the performance of a learner with access to all data.

in the beginning, however it reaches the same performance after approximately 80 epochs. In other words, in the case where each agent has a relatively large amount of data, distributing the training does not harm the performance of the algorithm. Moreover, the graph reveals that Federated training is more stable compared to centralized setting.

In the next step, we increased the number of participants from 4 to 50. This means that each agent had access to much smaller parts of the initial dataset. The resulting loss and accuracy can be found in Figure 5.

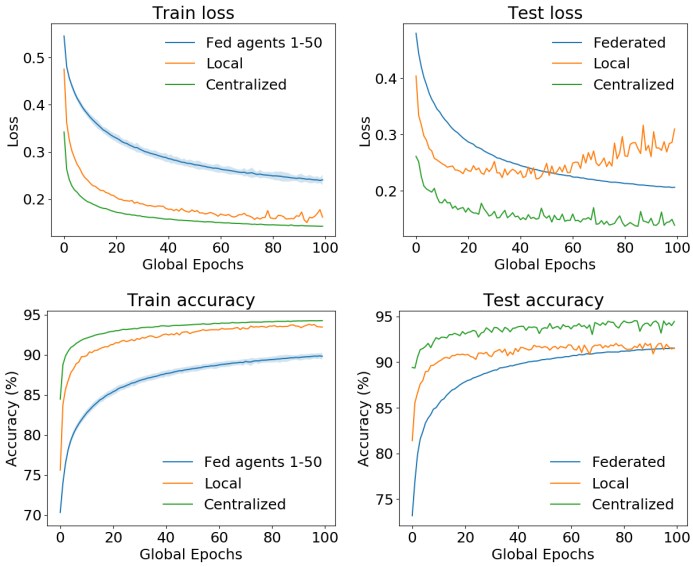

Fig. 5. Comparing the performance of a model with an equal-sized IID data split, 50 agents. *Centralized* shows the performance of a learner with access to all data. *Local* shows the performance of a learner with access to only his own data.

The test accuracy of FL is worse that centralized training, however it reaches the same accuracy as the local trainings. Moreover, the test loss reveals an interesting property of FL: it is less prone to overfitting. This is not surprising, seeing that Federated Averaging plays the role of label smoothing, that has been shown to be efficient against overfitting [23].

All in all, FL does not harm the performance of the agents in idealistic IID scenario, however it does not provide any benefits either.

C. Category 2: Label-based split

After studying the behaviour of the system in the ideal case of independent and identically distributed data, we proceeded by splitting the dataset by the type attack. In other words, each worker had an exclusive possession of one of the attack types, while the benign data was divided equally amongst them (Fig.6).

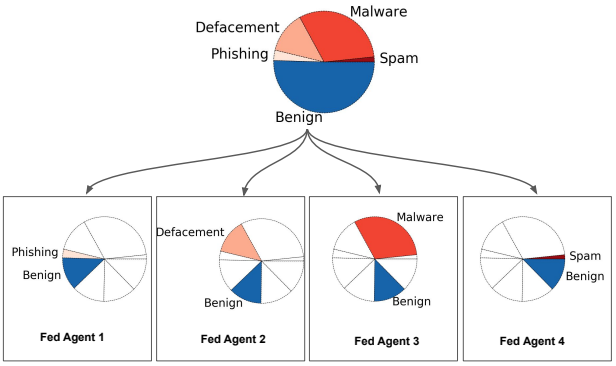

Fig. 6. Showing the distribution of samples in the label-based data split

When the data was split by attack, agents got a high local accuracy, which means that they learned well how to detect their particular type of attack. In this set of experiments our goal was to estimate how well their model generalizes to other attack types and what is the gain if they decide to collaborate with other agents.

At first, we selected 10% of each attack type and 10% of the benign data and used it as the test set. The rest of the data was distributed between 4 agents, as shown in Fig.6. We then run the federated training for 40 global epochs with 1 local epoch per round. We compared the resulting test accuracy to the accuracy on the *same* test set, achieved by each agent from training exclusively on their attack. Additionally, we noticed that the performance of FL in this setting varies a lot depending on the weight initialisation of the default models. Therefore both federated and local training were ran 10 times. We report the mean and the standard deviation of the resulting accuracy in Figure 7.

The test accuracy shows that in the case where the test label distribution from the initial dataset is kept (10 % of each class regardless the size), most of the participants benefit from collaborative training and only the agent with Malware reaches the same accuracy as FL. This is due to the fact that more than a half of the test data belongs to the Malware type. However, if we balance the test label distribution by selecting 1000 samples of each attack and 4000 samples of benign class, all the agents show improvment in collaborative learning (Figure 8, left).

Moreover, if we further exclude their own attacks from the test subsets, the advantage of Federated Learning is even bigger (Figure 8, right). In other words, if agents participate in FL, they become significantly better at detecting attacks they have not seen previously. In particular, the agent with Malware

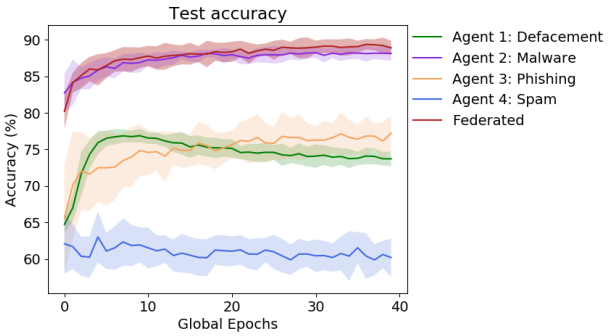

Fig. 7. Average performance when each agent only sees one class compared with the federated performance after combining the models. The shaded area indicates standard deviation over 10 runs with 40 epochs each.

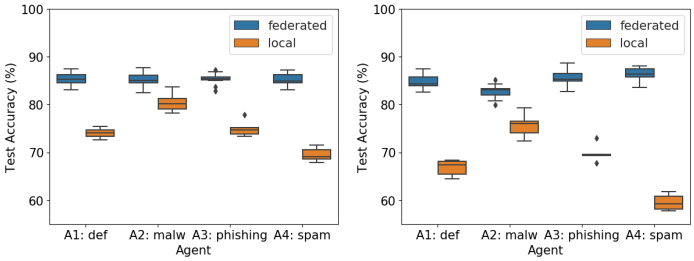

Fig. 8. Evaluating accuracy on: equally distributed test set (left) and test set with one attack excluded (right)

gains on average 8% in accuracy, the agent with Phishing 15%, while the agent with Spam boosts from 59% to 86%.

D. Category 3: Size-based split

The final set of experiments reflects a more realistic scenario where all agents have access to various attacks, but differ in the size of the datasets they posses.

The estimation of the agents size was made based on the average number of the security alerts received by SOCs on a daily basis during the year 2018 as reported in [24].

We created a Gaussian kernel density estimate (KDE) over the distribution, presented in aforementioned report. While not all of the alerts received in a SOC come from malicious URLs, it helped us to establish the differences in traffic volume amongst SOCs. We therefore scaled our agents to reflect this size distribution, as shown in Figure 9.

The sizes $s_1, ...s_{30}$ were sampled from the KDE and were allocated to 30 agents: $a_i : \{(x_1, y_1), ..., (x_{s_i}, y_{s_i})\}$. Each agent randomly selected his training samples from the training set T regardless of the label. Note that $\sum_{i=1}^{30} s_i > |T|$, which means that agents are partially overlapped. It reflects the fact that some URLs, especially benign, can be seen by many SOCs.

Similarly to the first experiment, 10% of the data from each class was reserved for the test set, the rest of the data belonged to the training set. We performed federated training for 30 epochs with 1 local epoch per round.

For evaluation, agents were grouped into small, medium, and large along their dataset size s as follows:

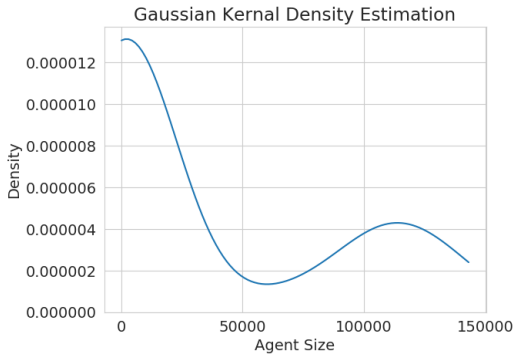

Fig. 9. Distribution of customer sizes obtained by a kernel density estimation of typical alert counts published in industry white papers.

$0 < s < 50000$ small, $50000 < s < 100000$ medium, and $100000 < s < 150000$ large. Figure 10 shows the the results.

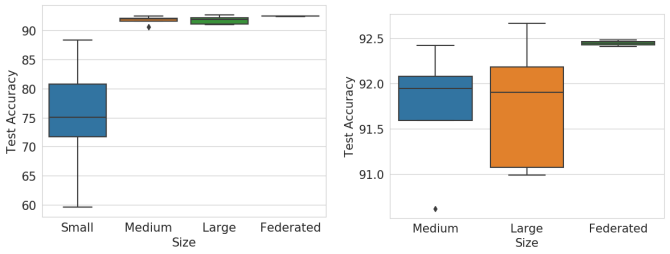

Fig. 10. Left: Testing accuracy for the three categories separately and the federated model performance for the size-based splitting scenario with 30 agents. Right: Small companies gain the most, while large companies only slightly improve with FL.

As expected, the small agents benefit the most from collaborative learning, gaining up to 30% in accuracy. Medium-sized and large-sized agents perform slightly worse than FL (only by 0.5%). However, they are less stable.

V. Discussion and Conclusion

In this paper, we examined how Federated Learning can help problems with data sharing in cyber security, in particular in malicious URL detection. We considered several scenarios with respect to the data partitioning between agents and showed, across all scenarios, that Federated Learning can only improve the detection performance. To provide a realistic analysis, we collected a huge dataset of labeled URLs with unbalanced classes. In experiments with different types of malicious URLs, the federated model can improve detection rates up to 27%. This also shows that it is difficult to generalize from one malicious class to another. Finally, we made an estimation of the real-world scenario and demonstrated that the overall performance increase exceeded 30% for some clients.

In future work, we would like to increase the complexity of the features and experiment with different federated aggregation mechanisms. Another interesting venue of research would be to apply FL in other domains of cyber security, e.g. in edge computing.

References

[1] General data protection regulation (GDPR) – official legal text. [Online]. Available: https://gdpr-info.eu/

[2] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data." [Online]. Available: http://arxiv.org/abs/1602.05629

[3] S. Ji, S. Pan, G. Long, X. Li, J. Jiang, and Z. Huang, "Learning private neural language modeling with attentive aggregation."

[4] T. Li, M. Sanjabi, and V. Smith, "Fair resource allocation in federated learning." [Online]. Available: http://arxiv.org/abs/1905.10497

[5] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, T. N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks." [Online]. Available: http://arxiv.org/abs/1905.12022

[6] X. Li, K. xuan Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," *ArXiv*, vol. abs/1907.02189, 2019.

[7] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning."

[8] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens." [Online]. Available: http://arxiv.org/abs/1811.12470

[9] J. Xu and F. Wang, "Federated learning for healthcare informatics," *CoRR*, vol. abs/1911.06270, 2019.

[10] X. Liang, Y. Liu, T. Chen, M. Liu, and Q. Yang, "Federated transfer reinforcement learning for autonomous driving," 2019.

[11] D. Leroy, A. Coucke, T. Lavril, T. Gisselbrecht, and J. Dureau, "Federated learning for keyword spotting," 2018.

[12] J. Payne and A. Kundu, "Towards deep federated defenses against malware in cloud ecosystems," *2019 First IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pp. 92–100, 2019.

[13] D. Huang, K. Xu, and J. Pei, "Malicious URL detection by dynamically mining patterns without pre-defined elements," *World Wide Web*, vol. 17, no. 6, pp. 1375–1394, 2014.

[14] vinayakumar R, S. S, S. KP, and M. Alazab, "Malicious URL Detection using Deep Learning," 1 2020.

[15] S. Marchal, K. Saari, N. Singh, and N. Asokan, "Know your phish: Novel techniques for detecting phishing sites and their targets," in *2016 IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2016, pp. 323–333.

[16] S. Marchal, J. François, R. State, and T. Engel, "PhishStorm: Detecting phishing with streaming analytics," vol. 11, no. 4, pp. 458–471.

[17] D. Sahoo, C. Liu, and S. C. H. Hoi, "Malicious URL detection using machine learning: A survey," *CoRR*, vol. abs/1701.07179, 2017.

[18] G. Kaiafas, G. Varisteas, S. Lagraa, R. State, C. D. Nguyen, T. Ries, and M. Ourdane, "Detecting malicious authentication events trustfully," in *2018 IEEE/IFIP Network Operations and Management Symposium, NOMS 2018, Taipei, Taiwan, April 23-27, 2018*, 2018, pp. 1–6.

[19] S. Varrette, P. Bouvry, H. Cartiaux, and F. Georgatos, "Management of an academic hpc cluster: The ul experience," in *Proc. of the 2014 Intl. Conf. on High Performance Computing & Simulation (HPCS 2014)*. Bologna, Italy: IEEE, July 2014, pp. 959–967.

[20] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, and J. Passerat-Palmbach, "A generic framework for privacy preserving deep learning," *CoRR*, vol. abs/1811.04017, 2018. [Online]. Available: http://arxiv.org/abs/1811.04017

[21] M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhanova, and A. A. Ghorbani, "Detecting malicious URLs using lexical analysis," in *Network and System Security*, J. Chen, V. Piuri, C. Su, and M. Yung, Eds. Springer International Publishing, vol. 9955, pp. 467–482. [Online]. Available: http://link.springer.com/10.1007/978-3-319-46298-1$_$30

[22] O. K. Sahingoz, E. Buber, O. Demir, and B. Diri, "Machine learning based phishing detection from URLs," vol. 117, pp. 345–357. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417418306067

[23] W. Li, G. Dasarathy, and V. Berisha, "Regularization via structural label smoothing." [Online]. Available: http://arxiv.org/abs/2001.01900

[24] Survey: 27 percent of IT professionals receive more than 1 million security alerts daily | imperva. [Online]. Available: https://www.imperva.com/blog/27-percent-of-it-professionals-receive-more-than-1-million-security-alerts-daily/