Random Forest based on Federated Learning for Intrusion Detection

Tijana Markovic[1] [0000−0002−4920−2012], Miguel Leon[1] [0000−0002−3425−3837], David Buffoni[2], and Sasikumar Punnekkat[1] [0000−0001−5269−3900]

[1] School of Innovation, Design and Engineering, Malardalen University, Vasteras, Sweden
{tijana.markovic,miguel.leonortiz,sasikumar.punnekkat}@mdu.se
[2] Tietoevry, Stockholm, Sweden
david.buffoni@tietoevry.com

Abstract. Vulnerability of important data is increasing everyday with the constant evolution and increase of sophisticated cyber security threats that can seriously affect the business processes. Hence, it is important for organizations to define and implement appropriate mechanisms such as intrusion detection systems to protect their valuable data. In recent years, various machine learning approaches were proposed for intrusion detection, where Random Forest (RF) is recognized as one of the most suitable algorithms. Machine learning algorithms are data-oriented and storing data for training on the centralized server can increase the vulnerability of the whole system. In this paper, we are using a federated learning approach that independently trains data subsets on multiple clients and sends only the resulting models for aggregation to a server. This considerably reduces the need for sending all data to a centralised server. Different RF-based federated learning versions were evaluated on four intrusion detection benchmark datasets (KDD,NSL-KDD, UNSW-NB15, and CIC-IDS-2017). In our experiments, the global RF on the server achieved higher accuracy than the maximum achieved with individual RFs on the clients in the case of two out of four datasets, and it was very close to the maximum for the third dataset. Even in the fourth case, the global RF performed better than the average accuracy, although it fell behind the maximum.

Keywords: Intrusion Detection · Random Forest · Federated Learning.

1 Introduction

Expanding use of computer networks has brought many security rules and preventive measures that have to be implemented by every organization whose data must not be compromised. Any malicious activity can seriously affect business processes and allowing unauthorized access to an organization's network can cause irreparable consequences. One of the necessary protective mechanisms is the Intrusion Detection System (IDS), and this research area has received a lot of attention during the past decade [18]. IDS is a software or hardware system that monitors the events occurring in a computer system or network and analyzes those events for signs of intrusion or violations of security polices [5]. IDSs can be divided based on the data sources (network-based NIDS and host-based HIDS), the analysis strategy (signature-based and anomaly-based) and the timing of information sources and analysis (interval-based and real-time) [5,14].

In recent years, artificial intelligence (AI) techniques are becoming widely used in the field of network security. Machine learning (ML) algorithms can learn from data how to distinguish between normal and abnormal activities, and this ability has proved to be very effective for the development of reliable IDSs. Various ML-based solutions were proposed by the researchers to improve the efficiency of IDSs [4]. Different ML algorithms were applied to Intrusion Detection (ID) problems and Random Forest (RF) is recognized as one of the most commonly used [7,22]. Many studies that compared the performances of different ML algorithms on different benchmark datasets concluded that the RF has the highest accuracy [23,2,9,11]. To evaluate the proposed ID techniques, researchers are using different publicly available datasets that are composed of normal traffic and different types of attacks [10]. The most frequently used public datasets are: KDD (24%), NSL-KDD (36%), UNSW-NB15 (18%) and CIC-IDS-2017&CSE-CIC-IDS-2018 (12%) [4].

Since the ML algorithms require a lot of data for training purposes, one of the main obstacles is the security of the provided data. Storing data and performing analysis and predictions on the centralized server can raise various security issues. These obstacles can be solved by the implementation of a collaborative learning approach, without the need of data sharing, and this approach is called federated learning (FL) [17,13]. FL is a decentralized learning technique that trains models locally on clients and transfers the parameters to the centralized server, which fits appropriately to the needs of IDSs [3,8]. Depending on the methodology of data distribution between different clients, FL can be [27]:

- Horizontal (HFL) - datasets have the same features but different instances.
- Vertical (VFL) - datasets have the same instances but different features.
- Transfer (TFL) - datasets are different in both instances and features.

In the literature, most of the models used for IDS in a FL setting are gradient-based, more specifically Deep Learning models. For example, Man et al. [20] proposed a FL mechanism for convolutional neural networks in Internet-of-Things applications. Contrary, we are considering a tree-based model which has been already successfully used in FL (e.g., FL based on Gradient Boosting trees [16]). Thanks to their great parallelism, RF is also a perfect candidate for FL. A federated version of RF has been proposed in [19], and it is proved to be as accurate as the non-federated version. In [25], authors proposed a way to create a decentralized federated forest in a ID scenario.

In this paper, we used a HFL approach to develop an anomaly-based intrusion detection system with a network-based data source (NIDS). The developed IDS is based on RF algorithm and it was evaluated on the most commonly used ID datasets. We summarize the contributions of this paper as follows:

- Comparison of RF hyper-parameters on four different ID benchmark datasets (KDD, NSL-KDD, UNSW-NB15, and CIC-IDS-2017).
- Comparison of different merging methods for aggregating independent RFs.
- Evaluation of RF HFL approach for ID.

This paper is organized as follows: Section 2 presents the used methodology, including the details about the used datasets and preprocessing techniques, the explanation of the regular and federated RF algorithm, and the presentation of the experimental setup. In Section 3 we present results of the conducted experiments, followed by the conclusion and plans for future work in Section 4.

2 Methodology

2.1 Datasets and preprocessing

The experiments presented in this paper were conducted on four existing ID datasets that are described in this section. For all of those datasets, data about network packets were preprocessed to create the features, and every entry was labeled either as normal activity, or as some type of network attack.

KDD99 [12] was created in a simulated environment, and it contains normal traffic and different types of network attacks. It has 41 features and the full training set contains 4 898 431 entries. In this paper, we used "kddcup.txt" file that contains 10% of data.

NSL-KDD dataset [1] is derived from the KDD99 dataset to solve some of the problems stated in [26]. Total number of entries in NSL-KDD is 148 517 and in this paper we used the data included in "KDDTrain+.txt" file.

UNSW-NB15 dataset [21] contains real normal traffic and simulated attacks. It has 42 features and the whole dataset has 2 540 044 entries. In this paper, we used a partition of data that is configured as a training set, which is included in "UNSW_NB15_ training-set.csv" file.

CIC-IDS-2017 dataset [24] was created in a simulated environment, and it contains normal traffic and different types of network attacks. CIC-IDS-2017 dataset has 78 features and 2 830 743 entries. The data capturing period was 5 days and the entries for each day and attack type are saved in a separate file. In this paper, we used all files except the files for Monday (that contains only normal traffic) and Thursday afternoon (that has an extremely low percentage of attacks - 0.01%). Two features from this dataset were removed (flow of bytes and flow of packets per second).

From the selected parts of the datasets, we have removed the instances that belong to classes with less than 800 cases. Additionally, we preprocessed each feature as follows:

- Features with binary values: The values were not changed.
- Features with numerical values: The values were normalized to a range between 0 and 1 using min/max approach.
- Features with categorical values: The values were one-hot encoded.

After applying above-mentioned preprocessing techniques, all datasets were divided with the goal to generate subsets to simulate a HFL setting. For each dataset, we selected a feature to be used as a division criteria, and removed it from the training data used by ML algorithms. We considered only subsets with at least 50 instances of both classes (normal and attack). Datasets were divided based on the value of the following feature:

- KDD99: "protocol" feature, resulting in 3 subsets
- NSL-KDD: "protocol" feature, resulting in 3 subsets

- UNSW-NB15: "service" feature, resulting in 6 subsets
- CIC-IDS-2017: "destination port" feature, resulting in 14 subsets

Table 1 presents a summary for the parts of the datasets that were used in this paper, as well as a summary for all subsets that were generated for federated learning. The number of features in parentheses is the actual number of features used by the ML algorithm after preprocessing.

Table 1: Information and distribution of the used instances for KDD99, NSL-KDD, UNSW-NB15, and CIC-IDS-2017 datasets.

DATASET	No. of Instances	No. of Features	Feature for Subsets	Subsets			
				Feature Value	No. of Instances	% of Instances	% of Attacks
KDD99	493 347	40 (115)	protocol	icmp	283 235	57.41%	99.5%
				tcp	189 786	38.47%	59.5%
				udp	20 326	4.12%	5.7%
NSL-KDD	125 597	40 (119)	protocol	icmp	8 090	6.44%	83.8%
				tcp	102 517	81.62%	47.7%
				udp	14 990	11.93%	17.1%
UNSW-NB15	79 924	41 (177)	service	-	45 516	56.95%	39.9%
				dns	21 367	26.73%	85.6%
				ftp	1 550	1.94%	51.1%
				ftp-data	1 396	1.75%	32%
				http	8 244	10.31 %	51.3%
				smtp	1 851	2.32%	65.7%
CIC-IDS-2017	1 543 535	75	destination port	21	11 781	0.76%	69.4%
				22	13 498	0.87%	45.5%
				53	643 986	41.72%	0.03%
				80	541 594	35.09%	70.6%
				88	3 920	0.25%	4.1%
				135	749	0.05%	21.4%
				139	1 921	0.12%	10.3%
				389	4 477	0.29%	3.6%
				443	312 986	20.28%	0.08%
				445	1 195	0.08%	15%
				465	2 656	0.17%	6%
				1124	259	0.02%	61.8%
				3268	1 903	0.12%	8.4%
				8080	2 610	0.17%	54.4%

2.2 Random Forest

Random Forest (RF) is an ensemble of different versions of the same machine learning algorithm, called Decision Tree (DT) [22].

DT is a predictive model based on constructing a tree that is composed of decision nodes. Beginning at the root node, the input parameters are tested at the decision nodes, with each possible outcome resulting in a branch that can lead either to another decision node or to a terminating leaf [15]. DT classifier employs splitting rules to construct a tree and in this paper we used following [28]:

- *gini* - attempts to find the largest homogeneous class and isolate it from the rest of the data,
- *entropy* - attempts to identify groups by minimizing the within-group diversity.

RF is an aggregation of DTs such that each tree uses a random subset from the examples in the training set. Those subsets are selected independently but with the same distribution for all DTs in the forest [6]. The number DTs that will be used for RF is an important hyper-parameter to achieve a high accuracy and avoid overfitting.

To determine the performance of the individual DT we used the following measures:

- *Accuracy (A)* - accuracy on the validation set
- *Weighed accuracy (WA)* - the weighted accuracy is equal to the accuracy of the DT for the predicted class (on the validation set) multiplied by the average accuracy of that DT for all different classes on the validation set.

In this paper, we used the following ensemble methods to determine the final prediction using the created DTs:

- *Simple Voting (SV)* - takes a majority vote as a predicted class.
- *Weighted Voting (WV)* - takes a majority vote as a predicted class but takes into account the weighted accuracy of the DTs.

2.3 Random Forest based on Federated Learning

Data distribution for our Federated Learning approach is horizontal, which means that the dataset is divided into a certain number of subsets where each subset has different instances but the same features. A RF model is trained on each client on a specific subset, then, all clients send the RF models to the centralized server, where they are combined in a global RF. Finally, the global RF is sent back to the clients for further usage. Architecture of the proposed approach is presented in Fig. 1.

Since each of the generated RFs is composed of the same number of DTs, different approaches were used to decide which DTs will be merged in the global RF:

- *Sorting DTs per RF based on Accuracy (S_DTs_A)* - DTs per RF are sorted based on the accuracy and the best ones from each RF are selected
- *Sorting DTs per RF based on Weighed Accuracy (S_DTs_WA)* - DTs per RF are sorted based on the weighted accuracy, and the best ones from each RF are selected
- *Sorting All DTs based on Accuracy (S_DTs_A_All)* - all DTs are assembled, sorted based on the accuracy and the best ones are selected
- *Sorting All DTs based on Weighed Accuracy (S_DTs_WA_All)* - all DTs are assembled, sorted based on the weighted accuracy and the best ones are selected

The maximum number of DTs (MaxDTs) that can be used for generating the global RF is the number of DTs per RF multiplied by the number of subsets. The number of the best DTs that will be included in the global RF is a hyper-parameter that may vary from 1 to MaxDTs for S_DTs_A_All and S_DTs_WA_All, and from the number of subsets to MaxDTs for S_DTs_A and S_DTs_WA.

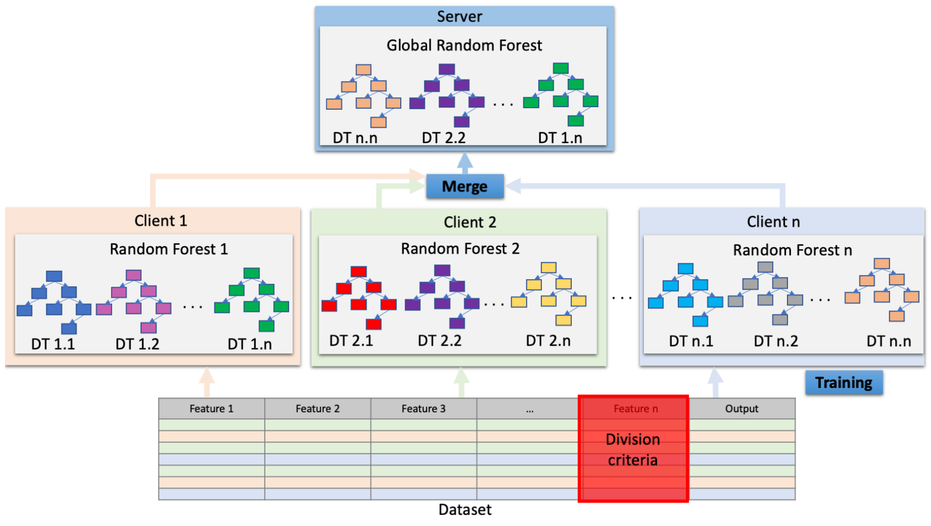


Fig. 1: Architecture of the Random Forest based Federated Learning

2.4 Experimental setup

As previously discussed, KDD99, NSL-KDD, UNSW-NB15 and CIC-IDS-2017 datasets have been used for the experiments. Three different experiments have been carried out:

1. *Study of hyper-parameters in RF* - The datasets have been divided into training set, validation set and testing set with a 70%-10%-20% distribution. The validation set was used to find the best combination of hyper-parameters in RF: number of DTs, splitting rule and ensemble method.
2. *Evaluation of RFs on different clients* - The training, validation and testing sets have been divided into subsets using one of the features, as explained in the Section 2.1. For each subset a independent RF was trained using the best combination of hyper-parameters from the first experiment.
3. *Federated Learning on the centralized server* - Different RFs were combined into a global one using different merging methods. The global RF is tested on the whole testing set created in the first experiment.

All experiments were performed five times and the mean value was used in all comparisons.

3 Results and Discussion

In this section, we present and discuss the results of our experiments. The best combination of hyper-parameters in RFs is presented in Section 3.1. An evaluation of RFs on different clients is shown in Section 3.2. Lastly, a study of different merging methods for aggregating independent RFs is presented in Section 3.3.

3.1 Study of hyper-parameters in RF

As indicated in Section 2.2, RFs have three important hyper-parameters: the number of DTs, splitting rule, and ensemble method. To find the best combination of the mentioned hyper-parameters, we conducted a comparative study with the number of DTs varying from 1 to 100 (since this is a binary classification problem, only odd numbers were used), two different splitting rules (gini and entropy) and two different ensemble methods (SV and WV).

The results are presented in Fig. 2. For three of the used datasets (KDD99, UNSW-NB15 and CIC-IDS-2017) entropy was the best splitting rule, while for NSL-KDD, gini was performing better. With respect to ensemble methods, SV outperformed WV in the older datasets (KDD and NSL-KDD), while WV gave better results in the newer ones. Finally, we hit a performance plateau with at least 11 DTs in KDD and NSL-KDD, while we have an increasing trend in UNSW-NB15 and CIC-IDS-2017. A summary of the best combination of hyper-parameters is given in Table 2.

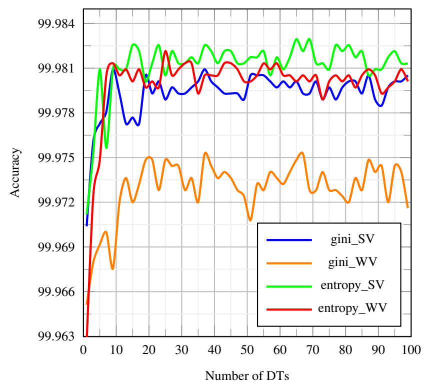
Table 2: Best combination of hyper-parameters in RF, per dataset.

Dataset	**KDD99**	**NSL-KDD**	**UNSW-NB15**	**CIC-IDS-2017**
Number of trees	65	65	93	77
Splitting rule	entropy	gini	entropy	entropy
Ensemble method	SV	SV	WV	WV

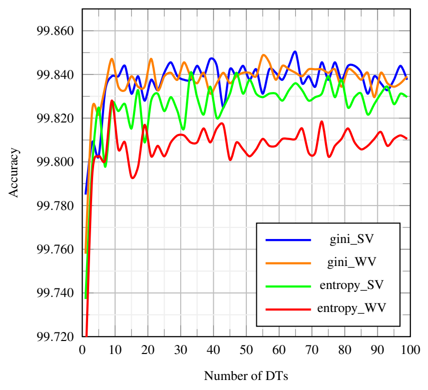
3.2 Evaluation of RFs on different clients

Different RFs have been trained and tested on different subsets using the best combination of hyper-parameters found in Section 3.1. The number of subsets differs between the datasets: KDD99 and NSL-KDD have 3 subsets each, UNSW-NB15 has 6, while CIC-IDS-2017 has 14 subsets. The results can be found in Fig. 3. Each RF has been trained and tested on the data from its own subset (blue bars in Fig. 3). Additionally, the same RFs have been tested on the entire testing set, independently on which subset the specific RF has been trained on (orange bar in Fig. 3).

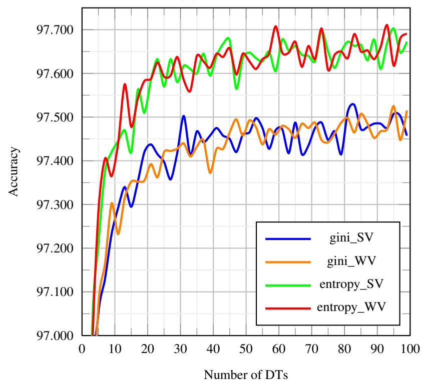
As can be observed, the performances are much lower when testing on the entire testing set than on the specific subset. The reason is that the different RFs have not been trained to classify those specific cases. This suggests that the different subsets have some differences between the instances. However, this is not always true, since some RFs manage to get high accuracy compared to the percentage of cases that belong to that subset (purple bar in Fig. 3). One good example is the subset with port 3268 in CIC-IDS-2017, where the accuracy of RF is 71.82% and the percentage of instances is 0.12% (Fig. 3d). This means that the RF is capable to correctly classify more instances than a random classifier. On the other hand, there are some RFs that perform much worse than a random classifier (e.g., port 21, 135 or 8080 in CIC-IDS-2017 Fig. 3d).



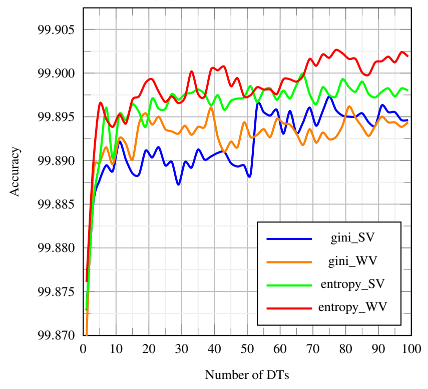
(a) KDD99



(b) NSL-KDD



(c) UNSW-NB15



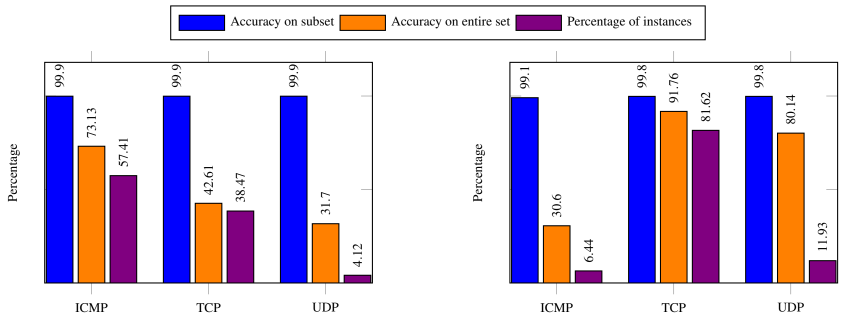
(d) CIC-IDS-2017

Fig. 2: Performance of RF on the validation set in (a) KDD99, (b) NSL-KDD, (c) UNSW-NB15, and (d) CIC-IDS-2017, for different combinations of hyper-parameters.

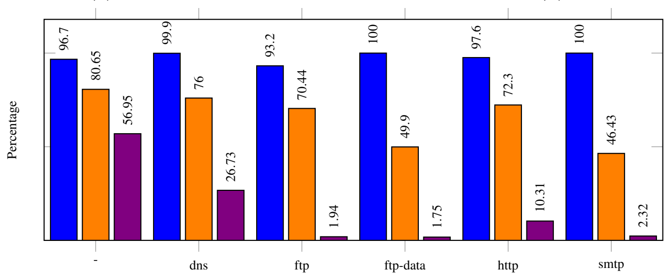
3.3 Federated Learning on the centralized server

As presented in Section 2.3 and further explained in Section 2.4, four different methods for merging RFs were compared. The best combination of splitting rule and ensemble method per dataset (found in Section 3.1) was used during the whole experimentation. The number of DTs for the global RF is studied in this experiment. For the methods S_DTs_A_All and S_DTs_WA_All, we tested all numbers between 1 and MaxDTs, while for the methods S_DTs_A and S_DTs_WA, we tested a multiple of the number of subsets (e.g., for KDD we tested 3,6,9... up to 195 n).

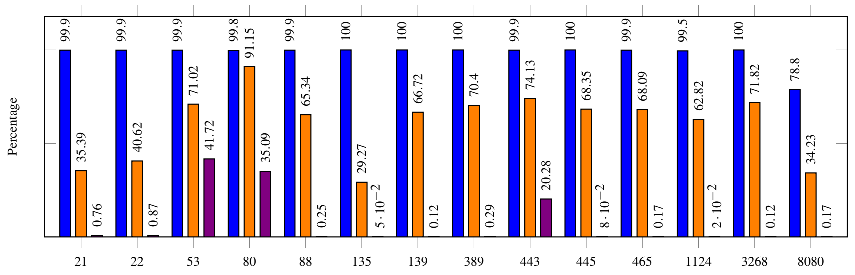
The results are presented in Fig. 4. We can see that combining all DTs and then selecting the best ones is the best option in UNSW-NB15 and CIC-IDS-2017. On the other hand, in KDD99 and NSL-KDD, the option of using the best DTs from each RFs and then combining them is a better option. When the sorting measurement is considered, there is not a big difference between A and WA, except for KDD99, where using A gives a considerable improvement (around 10%).



(a) KDD99 (b) NSL-KDD



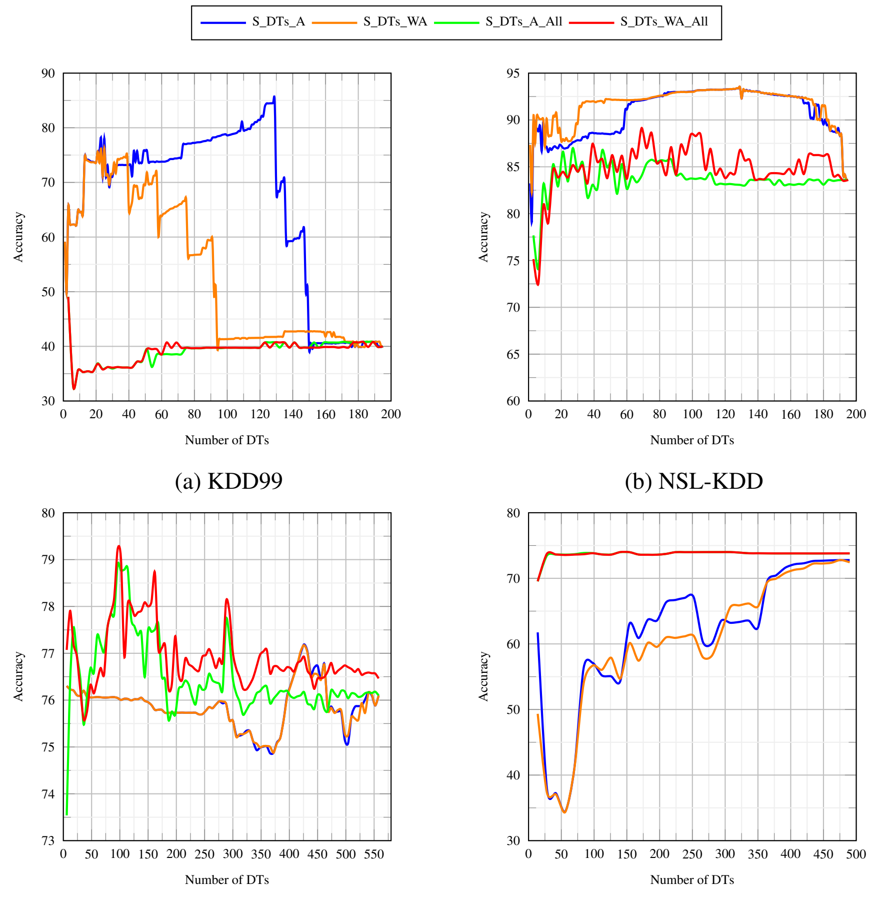
(c) UNSW-NB15



(d) CIC-IDS-2017

Fig. 3: Performance of independent RFs on different subsets from (a) KDD99, (b) NSL-KDD, (c) UNSW-NB15 and (d) CIC-IDS-2017 dataset.

The best combination of the number of DTs in global RF and the merging method per dataset are given in the upper part of Table 3. The lower part of Table 3 includes the performance of the global RF and the baselines that are used for the comparison. As the baselines, we used the maximum, average, and minimum accuracy of all independent RFs, that are presented in the orange bar of Fig. 3. From Table 3, we can see how the global RF improves the maximum accuracy of individual RFs for KDD99 and NSL-KDD, and it is very close for UNSW-NB15. For CIC-IDS-2017 it fell behind the maximum, but it performed better than the average accuracy.



(a) KDD99

(b) NSL-KDD

(c) UNSW-NB15

(d) CIC-IDS-2017

Fig. 4: Performance of the global RF on the testing set of (a) KDD99, (b) NSL-KDD, (c) UNSW-NB15, and (d) CIC-IDS-2017 using four different merging methods.

4 Conclusion and Future Work

In this paper, a RF-based federated learning approach for ID has been investigated in four different ID benchmark datasets. Different merging methods were evaluated to combine independent RFs, as well as the best combination of RF hyper-parameters including the number of DTs, splitting rule, and ensemble method.

The experiments have shown that combining independent RFs into a global one on the server outperforms the average accuracy of the RFs on the clients. This approach can be used in cases when the data cannot be centralized and in average it will provide a higher performance than using only the local RFs. Although this approach has lower accuracy than applying RF on the entire dataset, this approach is more realistic for real systems due to security restrictions and network overload.

Table 3: The upper part of the table shows the hyper-parameters needed to obtain the results at the lower part of the table.

Dataset		**KDD99**	**NSL-KDD**	**UNSW-NB15**	**CIC-IDS-2017**
No. of subsets (RFs)		3	3	6	14
No. of DTs per RF		65	65	93	77
No. of DTs in global RF		129	129	96	308
Merging method		S_DTs_A	S_DTs_A S_DTs_WA	S_DTs_WA_all	S_DTs_A_all S_DTs_WA_all
Accuracy of the global RF		84.77	93.51	79.13	73.26
Baseline	**Max accuracy**	74	91.76	80.65	91.15
	Avg. accuracy	49.15	67.5	65.95	54.82
	Min accuracy	31.7	30.6	46.43	29.27

As a part of future work, we plan to evaluate the proposed approach for the attack classification problem, as well as to study different methods for feature selection and reduction.

Acknowledgements This work has been partially supported by the H2020 ECSEL EU Project Intelligent Secure Trustable Things (InSecTT). InSecTT (www.insectt.eu) has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876038. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Sweden, Spain, Italy, France, Portugal, Ireland, Finland, Slovenia, Poland, Netherlands, Turkey.

The document reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains.

References

1. NSL-KDD. [`https://www.unb.ca/cic/datasets/nsl.html`] (2009)
2. Abedin, M., Siddiquee, K.N.E.A., Bhuyan, M., Karim, R., Hossain, M.S., Andersson, K., et al.: Performance analysis of anomaly based network intrusion detection systems. In: 43nd IEEE Conference on Local Computer Networks Workshops (LCN Workshops), Chicago, October 1-4, 2018. pp. 1–7. IEEE Computer Society (2018)
3. Agrawal, S., Sarkar, S., Aouedi, O., Yenduri, G., Piamrat, K., Bhattacharya, S., Maddikunta, P.K.R., Gadekallu, T.R.: Federated learning for intrusion detection system: Concepts, challenges and future directions. arXiv preprint arXiv:2106.09527 (2021)
4. Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., Ahmad, F.: Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies **32**(1), e4150 (2021)
5. Bace, R., Mell, P.: Intrusion detection systems. National Institute of Standards and Technology (NIST), Technical Report 800-31 (2001)
6. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
7. Buczak, A.L., Guven, E.: A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications surveys & tutorials **18**(2), 1153–1176 (2015)

8. Campos, E.M., Saura, P.F., González-Vidal, A., Hernández-Ramos, J.L., Bernabé, J.B., Baldini, G., Skarmeta, A.: Evaluating federated learning for intrusion detection in internet of things: Review and challenges. Computer Networks **203**, 108661 (2022)
9. Farnaaz, N., Jabbar, M.: Random forest modeling for network intrusion detection system. Procedia Computer Science **89**, 213–217 (2016)
10. Ghurab, M., Gaphari, G., Alshami, F., Alshamy, R., Othman, S.: A Detailed Analysis of Benchmark Datasets for Network Intrusion Detection System. Asian Journal of Research in Computer Science **7**(4), 14–33 (2021)
11. Hautsalo, J.: Using Supervised Learning and Data Fusion to Detect Network Attacks. [`urn:nbn:se:mdh:diva-54957`] (2021)
12. Hettich, S., Bay, S.D.: The UCI KDD Archive. [`http://kdd.ics.uci.edu`] (1999), irvine, CA: University of California, Department of Information and Computer Science.
13. Kairouz, P., McMahan, H.B., et al: Advances and open problems in federated learning (2021)
14. Khraisat, A., Gondal, I., Vamplew, P., Kamruzzaman, J.: Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity **2**(1), 1–22 (2019)
15. Larose, D.T., Larose, C.D.: Discovering knowledge in data: an introduction to data mining, vol. 4. John Wiley & Sons (2014)
16. Li, Q., Wen, Z., He, B.: Practical federated gradient boosting decision trees. Proceedings of the AAAI Conference on Artificial Intelligence **34**, 4642–4649 (04 2020)
17. Li, Q., Wen, Z., Wu, Z., Hu, S., Wang, N., Li, Y., Liu, X., He, B.: A survey on federated learning systems: Vision, hype and reality for data privacy and protection. IEEE Transactions on Knowledge and Data Engineering pp. 1–1 (2021)
18. Liao, H.J., Lin, C.H.R., Lin, Y.C., Tung, K.Y.: Intrusion detection system: A comprehensive review. Journal of Network and Computer Applications **36**(1), 16–24 (2013)
19. Liu, Y., Liu, Y., Liu, Z., Zhang, J., Meng, C., Zheng, Y.: Federated forest. CoRR **abs/1905.10053** (2019), `http://arxiv.org/abs/1905.10053`
20. Man, D., Zeng, F., Yang, W., Yu, M., Lv, J., Wang, Y.: Intelligent intrusion detection based on federated learning for edge-assisted internet of things. Security and Communication Networks **2021**, 9361348 (Oct 2021). `https://doi.org/10.1155/2021/9361348`
21. Moustafa, N., Slay, J.: 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). pp. 1–6. IEEE (2015)
22. Resende, P.A.A., Drummond, A.C.: A survey of random forest based methods for intrusion detection systems. ACM Computing Surveys (CSUR) **51**(3), 1–36 (2018)
23. Revathi, S., Malathi, A.: A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection. International Journal of Engineering Research & Technology (IJERT) **2**(12), 1848–1853 (2013)
24. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp **1**, 108–116 (2018)
25. de Souza, L.A.C., Antonio F. Rebello, G., Camilo, G.F., Guimarães, L.C.B., Duarte, O.C.M.B.: Dfedforest: Decentralized federated forest. In: 2020 IEEE International Conference on Blockchain (Blockchain). pp. 90–97 (2020)
26. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A.: A detailed analysis of the KDD CUP 99 data set. In: 2009 IEEE symposium on computational intelligence for security and defense applications. pp. 1–6. IEEE (2009)
27. Yang, Q., Liu, Y., Cheng, Y., Kang, Y., Chen, T., Yu, H.: Federated learning. Synthesis Lectures on Artificial Intelligence and Machine Learning **13**(3), 1–207 (2019)
28. Zambon, M., Lawrence, R., Bunn, A., Powell, S.: Effect of alternative splitting rules on image processing using classification tree analysis. Photogrammetric Engineering & Remote Sensing **72**(1), 25–30 (2006)