Attack detection using federated learning
IEEE Computer Society Conferences

Nicolas Bogenschultz*, Sami Benamar*, Martin Schaeffer*, Cedric Gattaz* and Hugo Ramond*
*School of Computer Science and Computer Engineering
CESI Engineering School, Strasbourg, France

1. Introduction

With the rapid increase in digitalization and interconnectedness, the volume of generated data has reached unprecedented levels. However, this abundance of data has brought forth new threats and vulnerabilities, particularly concerning the security of systems and the privacy of personal information.

The detection of cyber attacks is a critical challenge in safeguarding system security and protecting sensitive data. Traditional centralized approaches have been widely employed for attack detection, where all data is collected and analyzed on a central server. Nevertheless, this approach raises privacy concerns as it entails the massive transfer of personal data to third parties, potentially exposing users to risks.

In response to these challenges, a novel emerging approach known as federated learning has been developed. Federated learning is a distributed collaboration paradigm that enables the training of machine learning models on data distributed across multiple devices or entities, without the need for data centralization. This innovative approach offers a promising solution for attack detection while preserving data confidentiality.

In this paper, we extensively explore the fundamental concepts of federated learning and its potential application in the field of cyber attack detection. We examine the advantages and limitations of this approach, with a focus on privacy protection and data security. Additionally, we present various techniques and methodologies used in federated learning to effectively detect attacks while minimizing the disclosure of sensitive information.

Through an in-depth analysis of federated learning for attack detection, this paper sheds light on the potential of this approach to enhance the security and privacy of systems in the face of evolving cyber threats. The findings contribute to the ongoing research efforts in developing robust and privacy-preserving techniques for detecting and mitigating attacks in distributed environments.

2. State of the art

2.1. Attack detection methodologies

The detection of cyber attacks is a complex field that uses a variety of methods and techniques to identify malicious activity. These methods can be divided into two categories: signature-based approaches and anomaly-based approaches.

2.1.1. Signature-based approaches.

. These methods identify malicious activity based on known attack patterns called signatures. Alerts are generated when the activity matches an attack signature. The advantages of these approaches are their effectiveness in detecting known attacks and their ease of implementation and deployment. However, they are limited by their inability to detect new attacks that do not match known signatures.

2.1.2. Anomaly-based approaches.

. These methods identify abnormal and unusual behaviour that may indicate an attack. They are based on the idea that malicious activity is likely to differ from normal behaviour. The advantages of these approaches are their ability to detect new attacks. However, not all abnormal activity is necessarily malicious and can be limited by false positive rates.

2.2. Most common attacks

2.2.1. Denial of services.

. A denial of service attack, also known as Dos, is an attack designed to render a service unavailable by flooding it with network transmissions. unavailable by flooding it with network transmissions. However, this attack does not no advantage to the attacker over other types of attack that obtain or facilitate access. facilitate access. If this type of attack comes from several sources, it is called a distributed denial of service (DDoS). service (DDos).

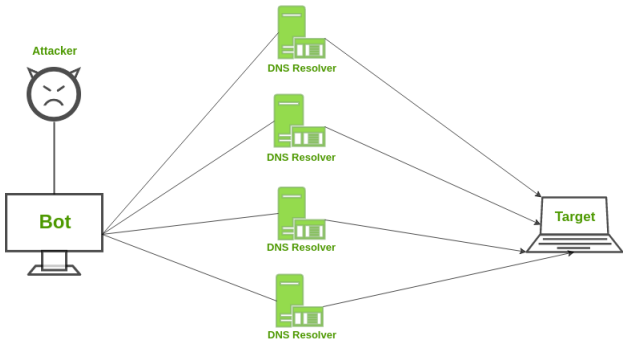

Fig.1 - Example of DDoS attack

2.2.2. Brute force attacks.

. This method is the oldest and simplest to perform. It can be used to crack a username encryption key, a password or even a hidden web page through several trial and error processes. several trial-and-error processes. This method is still effective today, successful attack depends on the complexity and length of the content to be obtained. content to be obtained. It can take anywhere from several seconds to several years.

2.2.3. Heart bleed attacks.

. The Heart Bleed flaw was introduced by Robin Seggelmann, a student at the University of Duisburg-Essen, who designed the Heartbeat extension for OpenSSL. This method is based on the fact that the server does not check that the integer sent actually corresponds to the size of the message. From therefore, the attacker can enter an integer greater than the message size, so that the server fills in the difference between the contents of its memory, which may contain passwords or private certification keys.

2.3. Federated learning

Federated learning is an approach to machine learning that allows multiple decentralised devices or servers to train a model on their own data, without having to share it directly with a central server. This approach is particularly useful in situations where data confidentiality is a major concern, as it allows the benefits of machine learning to be realised while minimising data exposure.

Federated learning can be used with both types of attack detection approaches. With signature-based approaches, federated learning can be used to train more robust and accurate models using data from multiple sources. With anomaly-based approaches, federated learning can be used to detect unusual behaviour in decentralised data, while respecting data confidentiality. Federated learning can be described in several stages:

1) Model initialization: A machine learning model is initialized on a central server.
2) Local training: Each device or decentralised server trains the model on its own data.
3) Model update: Updates to the model are sent to the central server. These updates are generally gradients or modifications to the model, not raw data, which helps to preserve data confidentiality.
4) Aggregation of updates: The central server aggregates model updates from all decentralised devices or servers to form a global model.
5) Model distribution: The global model is then sent back to all the decentralised devices or servers for a new local training iteration.

The benefits of federated learning include:

- Data confidentiality: As data remains on decentralised devices or servers and only model updates are shared, federated learning can help preserve data confidentiality.
- Federated learning allows decentralised data to be leveraged without having to centralise it, which can be beneficial in situations where data collection is expensive or difficult.

However, federated learning also has certain disadvantages:

- Coordination complexity: Coordinating model training across multiple devices or decentralised servers can be complex and requires effective communication management.
- Resistance to attack: Because federated learning involves multiple devices or decentralised servers, it can be vulnerable to a variety of attacks, such as Sybil or Byzantine attacks.

Despite these challenges, federated learning offers a promising approach for machine learning in situations where data confidentiality is a major concern.

2.4. Case studies

2.4.1. Attack detection : Detection Algorithm.

. The choice of algorithm in federated learning can drastically impact the efficiency and effectiveness of the model[1].

Starting with XGBoost, this is a gradient boosting algorithm that is known for its efficiency and effectiveness. It's well-suited to federated learning environments due to its built-in regularization parameters that penalize complex models, preventing over-fitting a common issue in distributed settings. XGBoost also excels in handling both regression and classification problems, offering versatility. However, its communication overhead can be quite high as gradient statistics need to be shared frequently during training, which can be a drawback in federated settings[9].

AdaBoost, short for Adaptive Boosting, is another powerful ensemble method. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. In federated learning, AdaBoost can effectively combine the models trained on each local data-set into a strong global model.

Nevertheless, like XGBoost, it suffers from high communication costs as it requires frequent model updates.

Neural Networks, specifically deep learning models, are a popular choice for federated learning due to their ability to learn complex patterns and high dimensional data. They are also naturally suited to the iterative nature of federated learning. However, they also have their own set of challenges. Training deep neural networks in a federated setting can be computationally expensive and time-consuming, given the need to back-propagate errors and update weights across all participating devices or servers. They also require a large amount of data to perform optimally, which might not always be available at each edge device in a federated setting.

Random Forests are an ensemble learning method that operates by constructing multiple decision trees during training and outputting the class that is the mode of the classes for classification or mean prediction of the individual trees for regression. They are inherently less prone to over-fitting and do not require scaling of data. However, in a federated learning context, Random Forests may struggle due to their nature of requiring access to global data statistics to optimally split nodes, which is not possible in a federated setting.

Algorithm	Approach	Accuracy	Recall	Precision	F1-Score	False Positive Rate	Time to Predict (sec)
Logistic Regression	NT Features	90.34%	70.45%	99.99%	81.67%	0.002%	0.042
	CN features	90.19%	70.09%	99.99%	82.41%	0.002%	0.044
Decision Tree	NT Features	92.26%	87.64%	88.57%	88.11%	5.49%	0.262
	CN features	96.78%	94.75%	95.39%	95.07%	2.23%	0.16
Random Forest	NT Features	93.06%	81.41%	96.87%	88.47%	1.27%	2.86
	CN features	97.37%	92.11%	99.85%	95.82%	0.066%	2.26
Neural Network	NT Features	93.19%	79.17%	99.99%	88.37%	0.002%	0.44
	CN features	94.13%	82.11%	99.95%	90.16%	0.016%	0.38
AdaBoost	NT features	93.32%	79.56%	99.99%	88.62%	0.002%	4.67
	CN features	96.99%	90.82%	99.99%	95.19%	0.004%	4.72
XGBoost	NT features	97.72%	93.02%	99.99%	96.38%	0.002%	0.64
	CN features	99.33%	97.97%	99.99%	98.97%	0.004%	0.618

Fig.2- Detection algorithms comparison

2.4.2. FL in attack detection : Aggregation Algorithm.

. Existing work has shown that federated learning can be used effectively to detect attacks in a variety of contexts, IoT, including wireless sensor networks, industrial cyber-physical systems and vehicular networks.

In the context of wireless sensor networks, federated learning has been used to train intrusion detection models from distributed data across multiple sensors. This approach has the advantage of not requiring data to be centralised, which can be beneficial in terms of confidentiality and efficiency. Here is a diagram showing how federation learning is used to detect attacks :

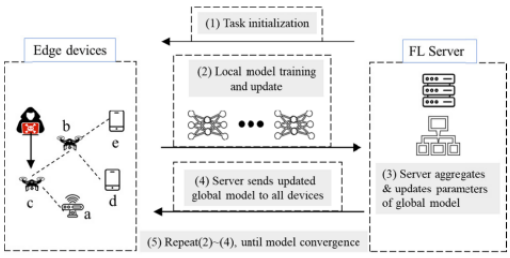

Fig.3 - Example of FL communication

In industrial cyber-physical systems, federated learning has been used to detect attacks that could compromise the security of these systems. Models trained using federated learning have shown good performance in detecting these attacks, while preserving data confidentiality.

In vehicle networks, federated learning was used to detect attacks that could compromise vehicle security and functionality. Models trained using federated learning performed well in detecting these attacks, while preserving data confidentiality.

However, despite these promising results, there are still many challenges to be overcome in using federated learning to detect attacks. For example, how can we ensure that the models trained using federated learning are robust to malicious attacks? How can we optimise the effectiveness of communication in federated learning to minimise the impact of denial of service attacks? How can we ensure that federated learning does not compromise data confidentiality?

Despite these challenges, federated learning offers many opportunities to improve attack detection. For example, federated learning could be used to train intrusion detection models from data distributed across a large number of devices, which could improve detection performance. In addition, federated learning could be used to preserve data confidentiality, which is particularly important in contexts where data is sensitive.

In the field of attack detection, federated learning has been used in several case studies, as shown by the following examples from the scientific literature.

Multiple scientist proposed an anomaly detection method based on deep learning for personnel identification and fire smoke detection in IoT equipment. In addition, a surveillance system called Read-IoT has been developed for the Tunisian army, which uses a network of heterogeneous objects to monitor security threats. Others proposed a framework that reduces the time needed to identify anomalous events in surveillance networks. This framework uses a combination of two-way LSTM and CNN features to identify and classify anomalous events.

Another notable example is the outlier detection algorithm called xStream, which deals with feature evolution in data streams.

One study proposed a solution in the form of FedAGRU, an intrusion detection method based on the federated learning framework.

This method does not require the transmission of original data to a central server, thus reducing the risk of data leakage while ensuring model accuracy. The FedAGRU algorithm uses the attention mechanism to improve its overall convergence speed and communication efficiency. The method was evaluated on three real network data-sets and was found to achieve better detection performance than the other centralised models tested.

It is important to note that the effectiveness of an algorithm can depend on the specific context in which it is used, including the type of data and the type of attacks it is designed to detect. Therefore, although the FedAGRU ap-

proach performed well in the study cited, other approaches may be more effective in other contexts.

To illustrate the operation of an attack detection system using federated learning, consider the following stages:

- The user uses the device, generating usage data.
- This data is collected and used by federated learning to train an attack detection model.
- The model is used to detect potential attacks.
- If an attack is detected, an alert is sent to the user.
- The model is constantly updated by federated learning to improve attack detection.

In addition, it is interesting to note that several research works have explored IoT vulnerabilities and feature-based security risks. For example, previous work has used deep learning for smart manufacturing, the Industrial Internet of Things and federated learning for anomaly detection. There is also research into the use of federated learning for anomaly detection. For example, one study used a VAE-LSTM model for anomaly detection, but found that data-sets from different sites can vary, increasing the bias of the model learned by the federated learning server.

.

Method	IID			non-IID		
	Accuracy (↑)	FAR (↓)	F1score (↑)	Accuracy (↑)	FAR (↓)	F1score (↑)
GRU-SVM(local)	99.84%	0.42 %	98.25%	98.84%	1.32 %	97.65%
GRU-softmax(local)	98.94%	1.12 %	97.96%	96.84%	3.15 %	95.46%
ICNN(local)	95.84%	5.16 %	94.55%	90.74%	9.89 %	90.55%
VAE(local)	97.84%	3.28 %	96.23%	95.84%	4.54 %	94.23%
FedAVG(GRU-SVM)	99.84%	1.82 %	97.44%	97.84%	2.08 %	97.02%
FedAGRU	**99.28%**	**0.73 %**	**98.12%**	**98.82%**	**1.43 %**	**97.12%**

Fig.4 - Aggregate algorithm comparison

Accuracy, False Alarm Rate (FAR), and F1 Score are key performance metrics utilized in statistics and machine learning. Accuracy represents the ratio of total correct predictions to the total number of predictions, providing an overall measure of a model's performance. However, it can be misleading in scenarios where the classes are highly imbalanced. In contexts where false positives can be costly or problematic, such as intrusion detection systems, the False Alarm Rate becomes a critical measure. This metric calculates the ratio of the number of false positives to the total number of actual negative observations. The F1 Score is another vital metric that combines both precision and recall, effectively taking into account both false positives and false negatives. It becomes particularly useful when classes are imbalanced and often offers more informative insights than accuracy alone[2].

On the other hand, "independently and identically distributed" or IID, and non-IID, are terms used to describe the nature of data within a data-set. IID implies that each data point in the data-set is generated by the same probability distribution process and is independent of all other data points, a common assumption in many statistical and machine learning models that simplifies analysis. However, non-IID is the exact opposite of IID. It indicates that the data points in the data-set are not all generated by the same probability distribution process or are not independent of each other. This situation can arise when data is collected over time or from different sources or populations, each with unique distribution characteristics. In federated learning scenarios, non-IID data is common due to the nature of data collection from a variety of users or devices, each with unique data or behavior patterns.

We can thus see that FedAGRU offers the best results compared to other algorithms.

. In conclusion, federated learning offers considerable potential for improving attack detection. However, many challenges remain to fully realise this potential. Future work should focus on addressing these challenges and exploring the many opportunities offered by federated learning.

2.5. Challenges and opportunities

Federated learning, a promising approach for detecting attacks in industrial control systems (ICS) and the Industrial Internet of Things (IIoT), presents both challenges and opportunities.

Among the challenges, data variability is a major obstacle. Data sets from different sites can vary considerably, increasing the bias of the model learned by the federated learning server. This variability can make it difficult to produce relevant predictions. In addition, the complexity of the classification, exacerbated by the lack of data, can also hamper the production of accurate results.

Another challenge lies in the security of IIoT devices. These devices, often vulnerable to a variety of attacks due to their optimal power consumption and micro-architecture less suited to deploying computationally heavy security firewalls, require special attention.

Despite these challenges, federated learning offers significant opportunities. It can be used to train an attack detection model, which is constantly updated to improve attack detection. Furthermore, despite the fact that the centralised learning model seems to perform better than their federated learning counterparts, the proposed federated learning approach nevertheless outperforms the centralised learning VAE-LSTM solution in most data-sets.

Threshold optimisation is another opportunity offered by federated learning. The paper suggests that the optimal threshold found by the KQE method can greatly improve the performance of an anomaly detection model and that their KQE is better and faster at finding the optimal threshold in most cases.

Finally, federated learning was used to deal with the problems of learning bias caused by the uneven distribution of data between different sites. This demonstrates the flexibility and adaptability of federated learning in the face of complex challenges.

2.6. Conclusion

The state of the art presented here has explored the use of federated learning in attack detection, focusing on attack detection methodologies, federated learning principles, relevant case studies and current challenges and opportunities.

It is clear that federated learning offers a promising new approach to attack detection, particularly in contexts where data confidentiality is a major concern. The case studies presented showed that federated learning can be used effectively to detect attacks in a variety of contexts, including wireless sensor networks, industrial cyber-physical systems and vehicular networks.

However, the application of federated learning to attack detection is not without challenges. Data variability, classification complexity and the security of IIoT devices all need to be addressed to realise the full potential of federated learning.

Despite these challenges, federated learning offers many opportunities to improve attack detection. Threshold optimisation, improved attack detection and dealing with learning bias issues are all opportunities that could be explored in future work.

In conclusion, federated learning represents a significant advance in the field of attack detection. Future research should focus on addressing the current challenges and exploring the opportunities offered by federated learning. In particular, further work is needed to develop effective methods for managing data variability, improving the security of IIoT devices and optimising the efficiency of communication in federated learning. With these advances, federated learning has the potential to transform the way we detect and respond to cyber attacks.

3. Try out

The study is divided into two parts:

- The first one is the attack detection algorithm.
- The second one is the federation algorithm

3.1. Attack detection algorithm

In order to validate the attack detection algorithm that we will be using, we have decided to employ the following three algorithms: Random Forest, XGBoost, and ADABoost. We have evaluated these three algorithms on our data-set based on various criteria such as accuracy, recall, precision, F1 Score, false positive rate, and execution time. To facilitate the algorithm validation process, we took the initiative not to divide the data-set into multiple parts but instead run the algorithm on a single virtual machine (VM). Below are the results obtained for each algorithm:

Algorithms	AdaBoost	XGBoost	Random Forest
Percentage of well classified	81.4159%	100%	99.9977%
Precision	0.7844	0.999996	0.999977
Recall	0.8142	0.999996	0.999977
F1 Score	0.7885	0.999996	0.999977
Overall false positive rate	0.0133	$2.99.10^{-7}$	$1,65.10^{-6}$
Algorithm execution time (in seconds)	710.2682	551.0632	100.7151

Fig.5 - Detection algorithm comparison

By evaluating these algorithms and their respective performance metrics, we aim to determine the effectiveness of the attack detection algorithm in order to proceed with the next steps of our study. We can observe that the XGBoost algorithm is significantly faster than ADABoost and more accurate in error detection than the other. Random Forest is five time faster than XGBoost and his results aren't this bad compare to XGBoost (less than 0,1% difference).

As with the previous evaluation, we will assess the performance of these algorithms on various metrics such as accuracy, recall, precision, F1 Score, false positive rate, and execution time. By analyzing the results obtained, we aim to determine the efficiency and reliability of the federation algorithm in the context of our study. First of all, we made the same tests, but we cut the data-set in 4 equal part and launch the 3 algorithm in 4 different VMs. And this is the results :

VM1 :

Algorithms	AdaBoost	XGBoost	Random Forest
Percentage of well classified	81.2542%	100%	99.9949%
Precision	0.7828	1.0	0.999949
Recall	0.8125	1.0	0.999949
F1 Score	0.7869	1.0	0.999949
Overall false positive rate	0.0133	0	$3.59.10^{-6}$
Algorithm execution time (in seconds)	131.5805	511.6711	21.1407

Fig.6 - Aggregate algorithm comparison

VM2 :

Algorithms	AdaBoost	XGBoost	Random Forest
Percentage of well classified	83.9545%	100%	99.9983%
Precision	0.8162	0.999974	0.999983
Recall	0.8395	0.999974	0.999983
F1 Score	0.8192	0.999974	0.999983
Overall false positive rate	0.0114	$1.79.10^{-6}$	$1.19.10^{-6}$
Algorithm execution time (in seconds)	127.1632	516.5346	16.9649

Fig.7 - Aggregate algorithm comparison

VM3 :

Algorithms	AdaBoost	XGBoost	Random Forest
Percentage of well classified	81.3128%	100%	99.9983%
Precision	0.7839	1.0	0.999983
Recall	0.8131	1.0	0.999983
F1 Score	0.7878	1.0	0.999983
Overall false positive rate	0.0133	0	$1.19.10^{-6}$
Algorithm execution time (in seconds)	127.9913	481.1338	16.2482

Fig.8 - Aggregate algorithm comparison

VM4 :

Algorithms	AdaBoost	XGBoost	Random Forest
Percentage of well classified	81.3279%	100%	99.9991%
Precision	0.7660	1.0	0.999991
Recall	0.8132	1.0	0.999991
F1 Score	0.7755	1.0	0.999991
Overall false positive rate	0.0133	0	$5.98.10^{-7}$
Algorithm execution time (in seconds)	125.9798	491.2915	21.1254

Fig.9 - Aggregate algorithm comparison

By dividing our dataset into 4 sub-VMs, we can observe that we do not significantly lose accuracy, recall, precision, F1 Score, and false positive rate. Only the execution time varies significantly for the Random Forest and ADABoost algorithms. Additionally, we can notice that XGBoost does not experience a significant change in execution time, making it less appealing in this context. Considering these observations, it becomes apparent that XGBoost, despite its consistent execution time, may not be as advantageous in terms of overall performance compared to the other algorithms. Consequently, we need to carefully evaluate the trade-offs between execution time and performance metrics

while considering the specific requirements of our study.

3.2. Federation algorithm

3.2.1. Approach used and set up. Regarding the federation algorithm, we have taken the initiative to once again utilize Random Forest. The method follows as follows: we combine multiple pre-trained (and saved) RandomForest models from various VMs. Then, the estimators (the individual decision trees that constitute the forest) from all loaded models are merged into a single model. This approach can be seen as a form of federated learning. In federated learning, models are trained on multiple devices or nodes, and then the models (or some parts of them) are combined into a single model.

The aggregation of individual models into one is a key characteristic of federated learning. The objective of this approach is to create a final model that is more performant and robust than the individual models, thanks to the aggregation of knowledge learned from different datasets or viewpoints.

By employing Random Forest in the federation algorithm, we aim to leverage the collective intelligence of multiple models trained on different VMs. This approach allows us to benefit from the diverse perspectives and datasets available across the sub-VMs.

The merging of the individual decision trees into a single model enables us to capture a broader range of patterns and improve the overall performance and robustness of the final model.

First of all, on our lab, we launch a python script who connect to each VMs execute the RandomForest script with 1/4 of our data-set, transfer the local model on the server and do this on all 3 VMs left.

After this, we use our aggregation script to merge all local model in one global model. We launch on the same DATA-set a RandomForest classic and one with the global model in entry.

To continue, this is the sequence diagram of the script use for this test :

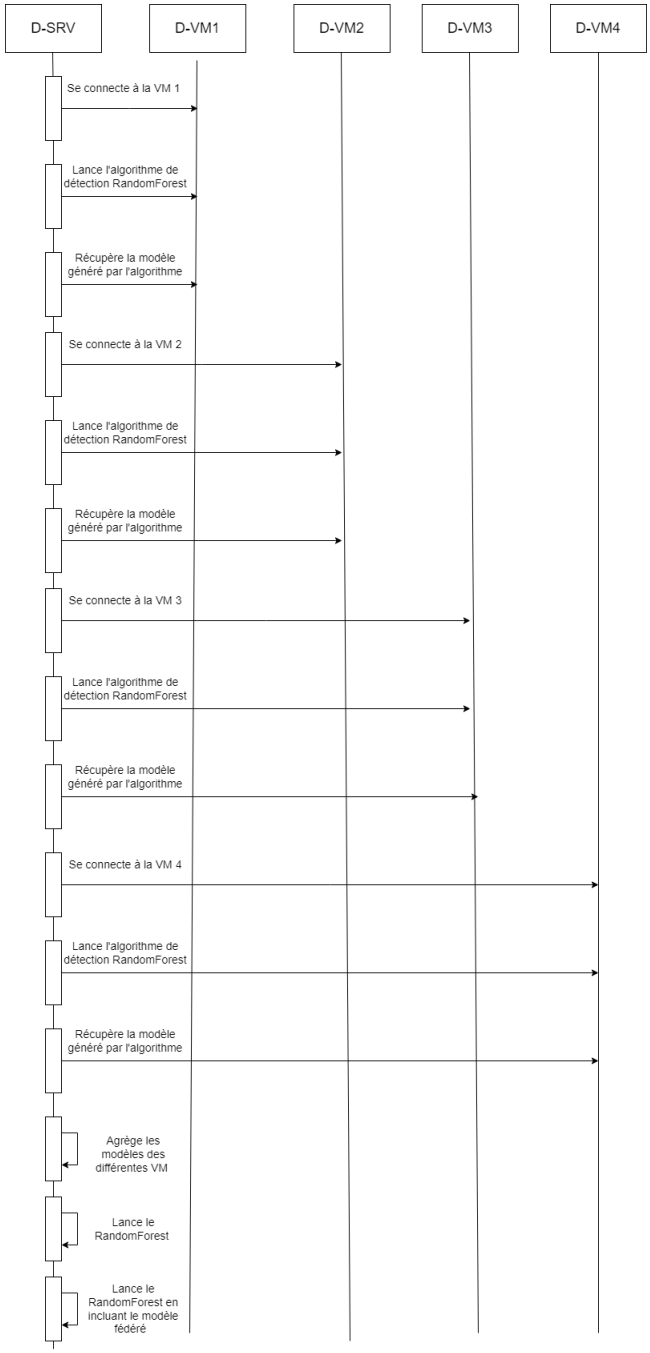

Fig.11 - Sequence Diagram

3.2.2. Result. Here how do we do for having the most accurate result possible ?

We will assess the performance of the algorithm using metrics such as accuracy, recall, precision, F1 Score, false positive rate, and execution time. By analyzing the results obtained, we aim to evaluate the effectiveness of the federated approach in enhancing the overall performance of the Random Forest model.

1) We first divided the complete dataset into 8 different parts.
2) We then took one of the dataset parts to train a model and make tests on this dataset, we recovered the model and the statistics (We repeated this manipulation on 4 different dataset parts).
3) We took all these statistics and averaged them, calling these results "Result1".
4) We then aggregated the 4 models
5) We tested the aggregated model on the remaining 4 datasets and averaged these results, calling them "Result2".
6) We aggregated the first 4 datasets to perform machine learning and testing in an unfederated way. We call these results "Result3".

Here's a table summarizing the results.

Algorithms	Result 1	Result 2	Result 3
Precision	0.999891	0.999958	0.999998
Recall	0.999891	0.999958	0.999998
F1 Score	0.999888	0.999957	0.999997
Overall false positive rate	$7.77.10^{-6}$	$2.99.10^{-6}$	$1.64.10^{-6}$
Algorithm execution time (in seconds)	6.9424	2.1082	108.1747

Fig.11 - Aggregate algorithm comparison

Although the results are all already very good, we can see that our federated learning works and does indeed improve the model compared to the model generated on the small dataset. However, machine learning without federation still performs better, which may seem logical. Federated learning is designed to handle situations where data is dispersed across multiple devices or locations, rather than centralised in a single location.

Unfederated machine learning, where all data is centralised, provides a complete and unified view of all data, and without the complexity of synchronisation between different nodes, the learning process can be more efficient and rapid.

4. Conclusion

In this research, we explored the use of various algorithms for attack detection and the implementation of a federation system. Our evaluation took into account several key criteria such as precision, recall, accuracy, F1 score, false positive rate and execution time. The results highlight the advantages and disadvantages of each algorithm.

Among the algorithms tested, XGBoost demonstrated significant execution speed and increased accuracy in error detection. However, when splitting the dataset across several virtual machines (VMs), we observed a sharp drop in execution time for the Random Forest and ADABoost algorithms, while XGBoost maintained its execution time.

Next, we implemented a federation algorithm based on the Random Forest model. The idea was to combine several pre-trained Random Forest models from different VMs into a single one. This method enabled us to take advantage of the diversity of perspectives and datasets available across the different VMs.

After testing the federation algorithm, we found that our federated learning approach improved the model well over the small dataset. However, unfederated machine learning still performed better, which seems logical given the concentrated nature of learning.

In conclusion, our study shows that while federated learning can improve model performance, particularly in a distributed system such as a network of VMs, traditional machine learning methods retain a slight edge. It's also important to note that the choice of algorithm must take into account the specific needs of the task in hand. As we look to the future, we plan to deepen our research into these concepts and continue to improve our methods to offer even more effective and efficient solutions.

References

[1] Redhwan Al-amri, Raja Kumar Murugesan, Mustafa Man, Alaa Fareed Abdulateef, Mohammed A. Al-Sharafi, and Ammar Ahmed Alkahtani. A review of machine learning and deep learning techniques for anomaly detection in IoT data. 11(12):5320.

[2] Mohamed Ali Ayed. MONTRÉAL, LE 5 FÉVRIER 2022.

[3] Ayan Chatterjee and Bestoun S. Ahmed. IoT anomaly detection methods and applications: A survey. 19:100568.

[4] Yao Chen, Yijie Gui, Hong Lin, Wensheng Gan, and Yongdong Wu. Federated learning attacks and defenses: A survey.

[5] Zhuo Chen, Na Lv, Pengfei Liu, Yu Fang, Kun Chen, and Wu Pan. Intrusion detection for wireless edge networks based on federated learning. 8:217463–217472.

[6] Elena Fedorchenko, Evgenia Novikova, and Anton Shulepov. Comparative review of the intrusion detection systems based on federated learning: Advantages and open challenges. 15(7):247.

[7] Rémi Gosselin, Loïc Vieu, Faiza Loukil, and Alexandre Benoit. Privacy and security in federated learning: A survey. 12(19):9901.

[8] Zakaria Abou El Houda. Renforcement de la sécurité à travers les réseaux programmables.

[9] Truong Thu Huong, Ta Phuong Bac, Dao Minh Long, Tran Duc Luong, Nguyen Minh Dan, Le Anh Quang, Le Thanh Cong, Bui Doan Thang, and Kim Phuc Tran. Detecting cyberattacks using anomaly detection in industrial control systems: A federated learning approach. 132:103509.

[10] Zengguang Liu, Cuiyun Guo, Deyong Liu, and Xiaochun Yin. An asynchronous federated learning arbitration model for low-rate DDoS attack detection. 11:18448–18460.

[11] Lingjuan Lyu, Han Yu, Xingjun Ma, Chen Chen, Lichao Sun, Jun Zhao, Qiang Yang, and Philip S. Yu. Privacy and robustness in federated learning: Attacks and defenses.

[12] Viraaji Mothukuri, Prachi Khare, Reza M. Parizi, Seyedamin Pouriyeh, Ali Dehghantanha, and Gautam Srivastava. Federated-learning-based anomaly detection for IoT security attacks. 9(4):2545–2554.

[13] Dinh C. Nguyen, Ming Ding, Pubudu N. Pathirana, Aruna Seneviratne, Jun Li, and H. Vincent Poor. Federated learning for internet of things: A comprehensive survey. 23(3):1622–1658.

[14] Ashneet Khandpur Singh, Alberto Blanco-Justicia, and Josep Domingo-Ferrer. Fair detection of poisoning attacks in federated learning on non-i.i.d. data.

[15] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. A survey on federated learning. 216:106775.