Federated Learning Based Detection of Ransomware

Bereket Getnet Teshome

A Thesis Submitted to the Graduate Faculty of

GRAND VALLEY STATE UNIVERSITY

In

Partial Fulfillment of the Requirements

For the Degree of

Master of Science

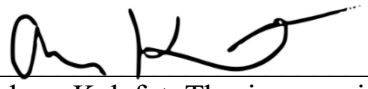
Cybersecurity

December 2022

Thesis Approval Form

GRAND VALLEY STATE UNIVERSITY

The signatories of the committee members below indicate that they have read and approved the thesis of Bereket Teshome in partial fulfillment of the requirements for the degree of Master of Science in Cybersecurity.

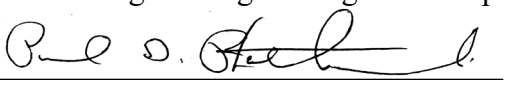


12/15/2022
Andrew Kalafut, Thesis committee chair — Date

12/15/2022
Vijay Bhuse, Committee member — Date

12/15/2022
Rahat Rafiq, Committee member — Date

Accepted and approved on behalf of the Padnos College of Engineering and Computing



Dean of the College

December 15, 2022
Date

Accepted and approved on behalf of the Graduate Faculty

Associate Vice-Provost for the Graduate School

1/12/2023
Date

Abstract

Ransomware is one of the top threats in the world of cyber security. The ransomware landscape is growing in sophistication and maturity. The latest developments in ransomware, such as Ransomware as a service (RaaS), have exacerbated the problem by offering would-be criminals ransomware services, lowering the technical barrier to entry. Private and public sector stakeholders are currently investing heavily in ransomware detection. Ransomware detection benefits private businesses and government organizations by reducing the hefty financial cost of a ransomware attack. It is therefore crucial that ransomware detection is accurate and efficient. There are shortcomings in machine learning (ML) models and datasets when working with ransomware detection. Specifically, there is a need for monitoring UDP traffic. One alternative that remains to be properly tested is federated learning. This thesis aims to demonstrate the viability of federated learning as a solution to detect ransomware, by testing speed and accuracy (using metrics such as accuracy, precision, and recall) in a virtual network environment. In addition to the main benefits of federated learning (distributed datasets and privacy), the research will also analyze if federated learning offers performance advantages in Malware detection compared to other machine learning models. The main focus of the research will be analyzing UDP traffic. UDP is not given much attention by organizations since it's a stateless protocol.

Table of Contents

List of Tables

List of Figures

Abbreviations

C&C	Command and Control
CPU	Central Processing Unit
FL	Federated Learning
GPU	Graphics Processing Unit
ML	Machine learning
SDN	Software Defined Networks
TFF	Tensorflow Federated
TTL	Time to Live
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Chapter 1 Introduction

(*2022-Unit42-Ransomware-Threat-Report-Final.Pdf*, n.d.) According to a survey in the ransomware threat report, by Paloalto networks, 58% of businesses hit by ransomware took more than a month to recover and 29% of businesses took more than three months to recover. In comparison, 9% said it took them more than six months. The ancillary costs of a ransomware attack can't also be neglected: costs associated with downtime, the impact of the breach on a company's brand reputation, and data loss that triggers any number of follow-on impacts.

The landscape of ransomware is also evolving. according to an IBM report, in 2021, there was an increase in Multi-extortion techniques (where attackers not only encrypt files but name and shame the victims and/or threaten more attacks), Ransomware as a service (which offers "startup kits" and "support services" to would-be cybercriminals, significantly lowering the technical barrier to entry and accelerating the speed with which attacks can be introduced and spread), and rapid weaponization of vulnerabilities(as long as organizations fail to patch known critical vulnerabilities, attackers will exploit them to their advantage).

Despite the relatively nascent nature of software-defined networks (SDN), they are not exempt from ransomware attacks. There are several machine learning (ML) based solutions proposed to mitigate ransomware (and malware) attacks against SDN, by the scientific community. Most of these studies analyze a dataset of network traffic and classify network traffic as benign or malicious, based on the predictors in the datasets such as Source Port, Destination Port, and TTL..

There were two issues with the ML mentioned above analysis. The first one was, that UDP traffic was not considered part of the classifiers. The datasets only included TCP traffic properties. Ransomware is not limited to TCP traffic. For example, the Cry ransomware uses UDP protocol to encrypt victims' files with a ".cry" file extension. In addition, traditional ML approaches require centralizing the training data on one machine or in a data center. There are a few issues with such approaches. The first one is, that the dataset stays the same unless otherwise updated. This requires constantly updating the dataset so the models improve their performance (a common issue known as concept drift). Another problem is that a centralized data set requires a complex hardware setup to run models that use a large memory volume and processing speed. Federated learning could solve this problem. This paper proposes using federated learning to detect ransomware targeting software-defined networks, embedded in UDP traffic, using federated learning.

1.1. Federated Learning

Standard machine learning approaches require centralizing the training data on one machine or in a data center. Federated Learning enables devices to collaboratively learn a shared prediction model while keeping all the training data on the device, decoupling the ability to do machine learning from the need to store the data in the cloud. This goes beyond the use of local models that make predictions on mobile devices (like the Mobile Vision API and On-Device Smart Reply) by bringing model training to the device as well.

Federated Learning allows for smarter models, lower latency, and less power consumption, all while ensuring privacy. And this approach has another immediate benefit: in addition to providing an update to the shared model, the improved model on your phone can also be used immediately, powering experiences personalized by the way you use your phone.

To implement FL, we will be using Tensorflow Federated (TFF). TensorFlow Federated (TFF) is an open-source framework for machine learning and other computations on decentralized data. It enables developers to simulate the included federated learning algorithms on their devices and data, as well as to experiment with novel algorithms.

Chapter 2 Review of Literature

Before writing this paper, a survey of relevant works was conducted. Due to the rapidly evolving nature of ransomware, it was beneficial to focus on recent papers. For the literature review, the sourced papers are from 2018 onwards. The papers are from the following sources: IEEE, ACM, Springer, and Science Direct. Since this paper touches on Ransomware, SDN security, and federated learning, the literature review is grouped accordingly.

2.1. Ransomware detection

(Cusack et al., 2018) presented a method for detecting ransomware via its network traffic signature by utilizing the high processing rate of new hardware-based flow generators in combination with RaftLib's high performance and parallel framework to process rich flow records, extract flow features, and classify ransomware. Since malware communication is moving towards HTTPS for delivery and control, it utilizes the unencrypted features of HTTPS traffic for model creation. When monitoring the communication between the infected machine and the C&C server, they were able to significantly reduce their initial feature set and achieve a detection accuracy rate of almost 87%, while maintaining a strong false negative rate of close to 10%. This work, however, doesn't cover UDP traffic. Some ransomware variants utilize UDP traffic to communicate with the C&C server.

(Beaman et al., 2021) Recent advances in ransomware analysis, detection, and prevention were explored. It was found that the focus of the state-of-the-art ransomware detection techniques mostly revolves around honeypots, network traffic analysis, and machine learning-based approaches.

Prevention techniques mostly focused on access control, data and key backups, and hardware-based solutions. However, it seems that there is a trend in using machine learning-based approaches to detect ransomware. They have conducted several experiments on ransomware samples, through which it was observed that there is a need for more intelligent approaches to detect and prevent ransomware. Through the experiments, it was also observed that ransomware could be easily created and used. In the end.

2.2. Security in SDN

(Yang et al., 2021) This paper proposes an ensemble classifier and evaluates its performance in traffic classification in SDN networks. Since SDN can read packet header fields only, the number of data features available to classification algorithms in real time is limited. Instead of relying on various statistical features of traffic traces for training the model, It uses the easily collected destination and source port numbers for training the classifier. To compensate for the lack of classification features, a two-tier classification workflow that combines the advantages of multiple classifiers is used. Experimental results show that the proposed ensemble classifier not only achieves an overall high classification accuracy but also improves the classification performance for individual traffic types.

(Cabaj & Mazurczyk, 2016) propose a solution for mitigating ransomware attacks using SDN-based solutions. The solution relies on an up-to-date database of malicious ransomware proxy servers. Even though the results of the hypothesis show promising results, there is the problem of constantly updating the database to keep up with new ransomware threats.

(Thapa et al., 2021) proposed an integrated clinical environment (ICE), called fedDICE (federated distributed integrated clinical environment). In this work, we presented FedDICE, which is federated learning distributed integrated clinical environment. It enables data privacy by leveraging federating learning. The results demonstrated that FedDICE effectively detects ransomware spread detection of WannaCry, Petya, BadRabbit, and PowerGhost with a testing accuracy of around 99% in the distributed integrated clinical environment (DICE). FedDICE is a generic framework and can be potentially used for other applications such as AI-supported medical decisions with data privacy. This work demonstrated the proof of concept of FedDICE and its application in ransomware spread detection and mitigation. Studies considering a large-scale DICE with a large number of ransomware types are interesting avenues for further exploration.

(Alotaibi & Vassilakis, 2021) The focus of this project is on blocking BadRabbit's attempts at self-propagation. An SDN-based IDPS consists of five modules to detect and block self-propagating ransomware, such as BadRabbit. There are several methods implemented. Specifically, deep packet inspection to look for specific values, and monitor ARP scanning. Three other modules rely on novel methods in SDN-based ransomware detection. They include inspecting the packet header to block SMBv1 access attempts, an SMB packet size checker, and finally using a honeypot in the network to detect any attempts to access port 80 or 445 of the honeypot system.

(Bhatia et al., 2019) This study shows that Autoencoder-based (a type of neural network that can be used to learn a compressed representation of raw data) unsupervised classifiers, when trained on benign traffic data, are effective at modeling network behavior and detecting anomalies and attacks in industrial networks.

They also outperform some supervised ML classifiers at detecting new, unfamiliar attacks. It also presents an ML-based approach to identify compromised sources under IP spoofing. The flows studied here were limited to TCP. Re-training methods and source detection were touched upon in this paper but need to be explored further, as well as comparisons with other unsupervised methods, such as one-class SVM and clustering.

2.3. Federated learning

(Rey et al., 2022) This work proposes a privacy-preserving framework for IoT malware detection that leverages FL to train and evaluate both supervised and unsupervised models without sharing sensitive data. This framework is designed to be deployed on the network nodes providing access to the IoT devices in Wifi, 5G, or B5G networks, offloading the computation from the IoT device itself. To demonstrate its feasibility in a realistic IoT scenario, the N-BaIoT dataset has been used due to its heterogeneity and divisibility in terms of IoT devices and malware samples. Using N-BaIoT, we compared the performance of i) a federated approach, where all device owners train their model, which is periodically aggregated in a server, and ii) a non-privacy-preserving setup, in which the whole dataset is centralized and trained by the server, and iii) a local setup where each device owner trains one isolated. and individual models.

This comparison has shown that the use of more diverse and more extensive data, as done in the federated and centralized methods, has a considerable positive impact on the model performance both in a supervised and an unsupervised scenario. Besides, it has been demonstrated that the privacy of the data can be preserved without losing model performance by following the federated approach.

(Li et al., 2020) Provides an overview of federated learning. It discusses the unique properties and associated challenges of federated learning compared with traditional distributed data center computing and classical privacy-preserving learning. The paper provides a broad survey of classical results as well as more recent work precisely focused on federated settings. Finally, it outlines a handful of open problems worth future research efforts.

(Duy et al., 2021) This paper leverages federated learning to construct a collaborative framework for training intrusion detection systems (IDS) in the context of SDN-based IoT. The traffic flow from each involved network is gathered and processed, then labeled by a security network tool before using as training input. Through experiments, the FL approach with local training on each security gateway server proves that the framework can achieve privacy preservation and maintain high-rate accuracy in anomaly detection without exposing the sensitive network data of involved parties.

Chapter 3 Methodology

3.1. Data

The dataset (Dheeru & Casey, n.d.) used for this research is obtained from Kaggle. The dataset suggests *real* traffic data, gathered from 9 commercial IoT devices authentically infected with malware. It was generated using anomaly detection techniques. However, as the malicious data can be divided into 10 attacks, the dataset can also be used for multi-class classification: 10 classes of attacks, plus 1 class of 'benign'. For each of the 9 IoT devices, a deep autoencoder was trained and optimized on 2/3 of its benign data (i.e., the training set of each device). This was done to capture normal network traffic patterns. The test data of each device comprised the remaining 1/3 of benign data plus all the malicious data. On each test set, we applied the respective trained (deep) autoencoder as an anomaly detector. The detection of anomalies (i.e., the cyberattacks launched from each of the above IoT devices) concluded with 100% TPR (true positive rate).

There are three steps involved in making a dataset ready to be consumed by machine learning models. It involves three steps: data acquisition, data preprocessing and creating a federated dataset.

I. Data acquisition

   The dataset is about 2GB and is collected from several devices in an IoT network. Since this dataset contains traffic information from various devices, it suits our paradigm of federated learning. the devices in the IoT setup will be considered clients when we construct our federated models.

The dataset is split into two CSV files. one file contains the benign traffic and the other contains the malicious traffic. we merge these two files to create a data frame, as follows:

```
benign = pd.read_csv('../benign.csv')
malicious = pd.read_csv('../malacious.csv')
benign['type'] = 'benign'
malicious['type'] = 'malicious'
data = pd.concat([benign, malicious], axis=0, sort=False, ignore_index=True)
#Show how many instances of each class in the dataset
data.groupby('type')['type'].count()

output:
type
benign          40154
malicious       104011
Name: type, dtype: int64
```

TABLE 1 Statistics of the dataset

Feature Name		Number of Instances, %
IoT device types	Security Cameras	1
	Webcam	1
	Smart Baby Monitor	1
	Thermostat	1
	Smart Doorbell Devices	2
General Features	Total number of instances	144, 165
	# of features in the dataset	115
	Time Windows	100 ms, 500 ms, 1.5s, 10 s, and 1 min
Distribution of data (2 classes)	# of “Benign” records	40,154 (7.23%)
	# of “malicious” records	104,011 (92,77%)

II. Data preprocessing

Standard data pre-processing was applied before working with the dataset. This includes removing empty values, selecting only relevant columns, and normalizing integer values. We divide the dataset into two categories during training: clean (label 1) and ransomware (label 0). We consider binary classification because our main task is to detect the ransomware irrespective of its specific type. In FL, the training and validation dataset are local, so it is different for different clients, whereas the test dataset is global and checked on the global model.

Shuffle the data:

Shuffling data serves the purpose of reducing variance and making sure that models remain general and overfit less.

```
#Shuffling rows of the dataframe
sampler = np.random.permutation(len(df))
df = df.take(sampler)
```

Dataset Normalization:

We use standard scaler algorithm to normalize our dataset. StandardScaler standardizes a feature by subtracting the mean and then scaling to unit variance. StandardScaler results in a distribution with a standard deviation equal to 1.

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
data = scaler.fit_transform(features)
```

Cross Validation:

Since there is a big class imbalance (70% of the data is malicious), we need to consider this when splitting our dataset into training and testing. For such cases, cross-validation is the way to go, specifically KFold cross-validation. KFold Provides train/test indices to split data into train/test sets. Split the dataset into k consecutive folds. Each fold is then used once as validation while the k - 1 remaining folds form the training set.

III. creating federated data

A key concept in federated learning is "federated data", which refers to a collection of data items hosted across a group of devices in a distributed system (eg. client datasets, or the server model weights). The entire collection of values across all devices is represented as a single *federated value*. In FL, the training and validation dataset are local, so it is different for different clients, whereas the test dataset is global and checked on the global model... There are three steps involved to use a normal dataset as a federated one.

1. partition the dataset into per-client subsets
2. create a data per-client subset
3. pass a list of all, or subset, of the database, objects to the federated optimizations

3.2 Implementation

To get a clear picture of the performance of our federated model, we need to compare it to a non-federated model. This will give us an insight into how FL stacks compared to non-federated ones. For this purpose, We use logistic regression to compare it to our FL model. Our FL model is Fedavg (Xing et al., 2022). The training/testing implementation is done in python programming by leveraging the PyTorch library. All the programs were executed on a Dell laptop with an Intel Core i5-8350U CPU, 8GB RAM, and an x64-based processor.

3.3 Metrics for performance measurement

We consider accuracy, precision, recall, F1-score, and false-negative rate (FNR) for performance analysis. These metrics are defined in the appendix section. In our dataset setup, the ransomware class is labeled 0, and it is a positive class, whereas the normal class is labeled 1, and it is a negative class. Thus, we consider only FNR (not false-positive rate) in our analyses. Moreover, precision, recall, and F1-score have averaged figures (average = ‘macro’).

Chapter 4 Results

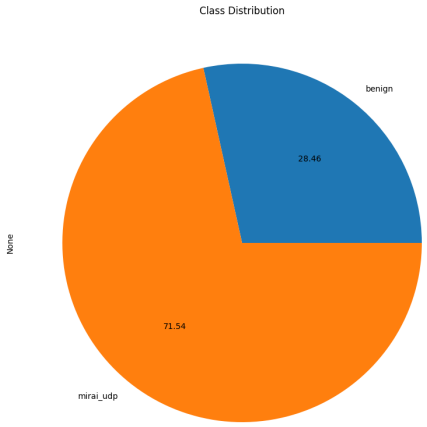


Fig 1: Class distribution of our dataset

Table 2: Performance results of fedAvg with several clients

Clients	Precision	Recall	F1- score
0	0.989759180734459	0.995173356930255	0.992458884877657
1	0.762988541830168	0.791951775822744	0.777200415700695
2	0.550168111838613	0.503074433656958	0.525568421942355
3	0.984731385485391	0.891619730329408	0.935865281261197
4	0.496504826667935	0.99904315376519	0.663341804320203
5	0.706444967628016	0.396416480885146	0.507854233881631
6	0.897483777601419	0.993232072742302	0.942933516443093
7	0.982795317488471	0.842801733972165	0.907430910951894
8	0.785249636318722	0.91548876304795	0.845382149136415
9	0.823467875512161	0.833274523641496	0.828342175647827
accuracy	0.751400526404752	0.751400526404752	0.751400526404752
macro avg	0.725417601918669	0.742006841976879	0.720579799469361
weighted avg	0.687838223049343	0.751400526404752	0.703986178905453

Fig 2: Training Loss graph

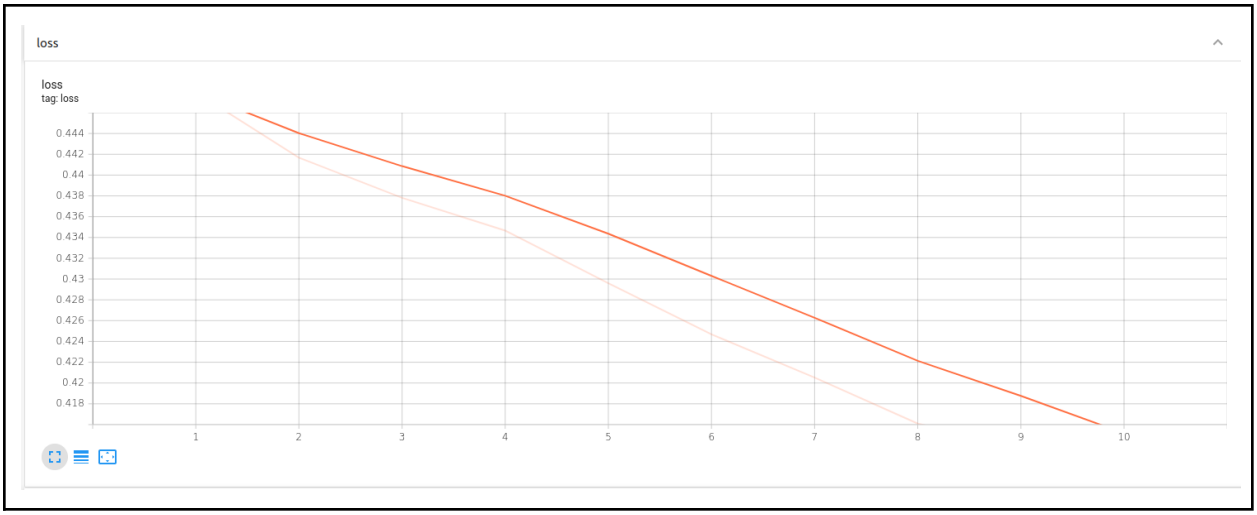


Fig 3: Sparse categorical accuracy graph

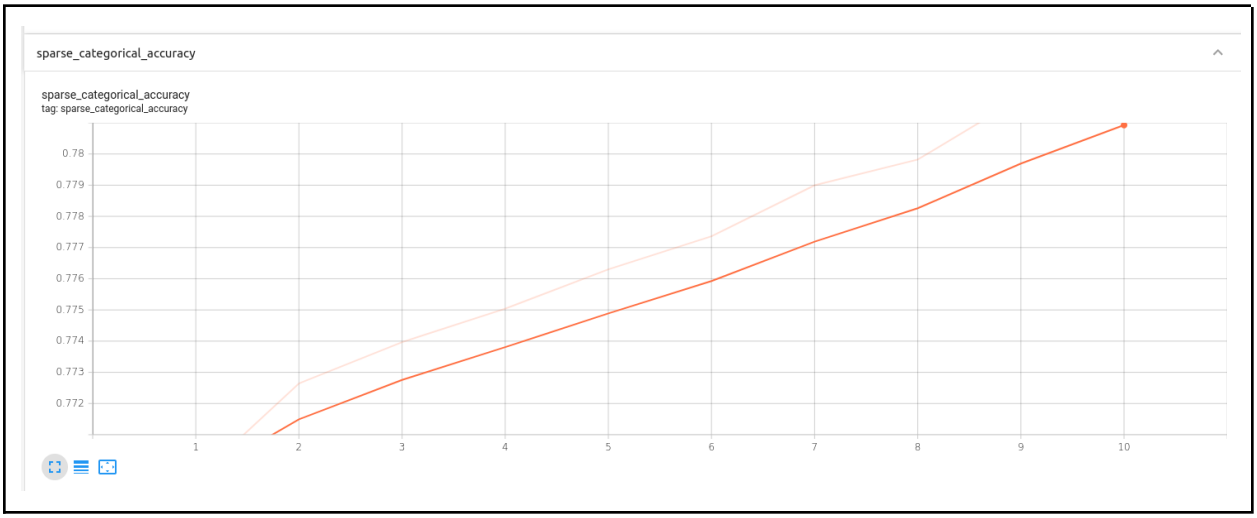


Table 3: Results summary of centralized learning (Logistic regression)

Precision	Recall	F-score
0.9523044206520952	0.9448900840209669	0.9483838797097701

Chapter 5 Discussion and Conclusions

As we can see from diagram 1, 71.54% of the dataset is malicious whereas the remaining 28.46% is benign. This is a big class imbalance and might affect the results of our models. Specifically, when considering matrices such as accuracy. we should put more emphasis on other matrices, such as F-score.

Each client had a different precision, recall, and f1 score. This result is aggregated at the end, which leads to relatively lower performance. especially compared to our logistic regression model. This could be rectified by performing multiple iterations of averaging or introducing a more diverse dataset. This finding could nullify our initial assumption that federated learning results in a better performance than centralized learning.

This work demonstrated the proof of concept of federated learning and its application in ransomware spread detection and mitigation. Studies considering large-scale clients with a large number of ransomware types are interesting avenues for further exploration. Other federated learning algorithms can be explored as well (such as FedSGD). In addition, a GPU-oriented solution to federated learning could be an interesting area to explore. The resource-intensive computations lend themselves well to the utilization of GPU.

Appendices

Metrics for performance measurement Before introducing accuracy, precision, recall, F1-score, and false-negative ratio (FNR), we define the following terms:

– **True Positive (TP)**: Number of positive classes that are correctly classified as positive.

– **False Positive (FP)**: Number of negative classes that are incorrectly classified as positive.

– **True Negative (TN)**: Number of negative classes that are correctly classified as negative.

– **False Negative (FN)**: Number of positive classes that are incorrectly classified as negative.

- **Accuracy**: Accuracy indicates the ability of the classifier to correctly classify all classes. It is calculated as

  Accuracy = $\frac{TP + TN}{TP + TN + FP + FN}$ $X\ 100\%$

- **Precision**: Precision indicates the ability of the classifier not to classify positive class if the data is actually of negative class. It is calculated as

  Precision = $\frac{TP}{TP + FP}$

  Its value ranges from zero to one, and high precision indicates better classifier/detection from the FP perspective.

- **Recall**: Recall (also known as sensitivity) indicates the ability of the classifier to correctly classify positive class It is calculated as

  Recall = $\frac{TP}{TP * FN}$

- **F1-score**: F1-score is the harmonic mean of Precision and Recall. It takes both FP and FN into consideration. It is calculated as

  F1-score = 2 X $\frac{Precision\ X\ Recall}{Precision + Recall}$

Bibliography

"2022-unit42-ransomware-threat-report-final.pdf." Accessed: Aug. 04, 2022. [Online]. Available: https://www.paloaltonetworks.com/content/dam/pan/en_US/assets/pdf/reports/2022-unit42-ransomware-threat-report-final.pdf?utm_source=marketo&utm_medium=email&utm_campaign=Global-DA-EN-22-03-17-7014u000001hKM8AAM-P3-Unit42-2022-unit-42-ransomware-threat-report&mkt_tok=NTMxLU9DUy0wMTgAAAGFbmQr974xd6jRB3QCbeDK2wubsMP5FRn7-1oEGz8elogIprhw8W2dP6ddX_iUAIBxPPG4QsEWOO_D5SSPSsBdZfIQgCOe-_tmtb2IGQsKykFkTN8YISg

G. Cusack, O. Michel, and E. Keller, "Machine Learning-Based Detection of Ransomware Using SDN," in *Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*, New York, NY, USA, Mar. 2018, pp. 1–6. doi: 10.1145/3180465.3180467.

C. Beaman, A. Barkworth, T. D. Akande, S. Hakak, and M. K. Khan, "Ransomware: Recent advances, analysis, challenges and future research directions," *Comput. Secur.*, vol. 111, p. 102490, Dec. 2021, doi: 10.1016/j.cose.2021.102490.

T. Yang, S. Vural, P. Qian, Y. Rahulan, N. Wang, and R. Tafazolli, "Achieving Robust Performance for Traffic Classification Using Ensemble Learning in SDN Networks," in *ICC 2021 - IEEE International Conference on Communications*, Jun. 2021, pp. 1–6. doi: 10.1109/ICC42927.2021.9500571.

K. Cabaj and W. Mazurczyk, "Using Software-Defined Networking for Ransomware Mitigation: The Case of CryptoWall," *IEEE Netw.*, vol. 30, no. 6, pp. 14–20, Nov. 2016, doi: 10.1109/MNET.2016.1600110NM.

C. Thapa, K. K. Karmakar, A. H. Celdran, S. Camtepe, V. Varadharajan, and S. Nepal, "FedDICE: A ransomware spread detection in a distributed integrated clinical environment using federated learning and SDN based mitigation," vol. 402, 2021, pp. 3–24. doi: 10.1007/978-3-030-91424-0_1.

F. M. Alotaibi and V. G. Vassilakis, "SDN-Based Detection of Self-Propagating Ransomware: The Case of BadRabbit," *IEEE Access*, vol. 9, pp. 28039–28058, 2021, doi: 10.1109/ACCESS.2021.3058897.

R. Bhatia, S. Benno, J. Esteban, T. V. Lakshman, and J. Grogan, "Unsupervised machine learning for network-centric anomaly detection in IoT," in *Proceedings of the 3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks*, New York, NY, USA, Dec. 2019, pp. 42–48. doi: 10.1145/3359992.3366641.

V. Rey, P. M. S. Sánchez, A. H. Celdrán, G. Bovet, and M. Jaggi, "Federated Learning for Malware Detection in IoT Devices," *Comput. Netw.*, vol. 204, p. 108693, Feb. 2022, doi: 10.1016/j.comnet.2021.108693.

T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated Learning: Challenges, Methods, and Future Directions," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 50–60, May 2020, doi: 10.1109/MSP.2020.2975749.

P. T. Duy, T. V. Hung, N. H. Ha, H. D. Hoang, and V.-H. Pham, "Federated learning-based intrusion detection in SDN-enabled IIoT networks," in *2021 8th NAFOSTED Conference on Information and Computer Science (NICS)*, Dec. 2021, pp. 424–429. doi: 10.1109/NICS54270.2021.9701525.

D. Dheeru and G. Casey, “N-BaIoT Dataset to Detect IoT Botnet Attacks.” https://www.kaggle.com/datasets/mkashifn/nbaiot-dataset (accessed Nov. 13, 2022).

S. Xing, Z. Ning, J. Zhou, X. Liao, J. Xu, and W. Zou, “N-FedAvg: Novel Federated Average Algorithm Based on FedAvg,” in *2022 14th International Conference on Communication Software and Networks (ICCSN)*, Jun. 2022, pp. 187–196. doi: 10.1109/ICCSN55126.2022.9817607.