International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 11 Issue 3 ǁ March 2023 ǁ PP. 94-100

Random Forest Algorithm based Security in IoT

Rajani S. Pujar
Electronics and Communication Engineering Department
Basaveshwar Engineering College
Bagalkote - 587102, Karnataka, India

Abstract: *Recent developments in programmable networks, particularly the programmability of data planes in switches and routers, have opened up new channels for the detection of such assaults. Taking advantage of this newly discovered capability, this paper suggests using Random Forests, a Machine Learning approach, to help identify DoS assaults in a programmable switch fast and reliably. Many procedurally generated classification trees are used in random forests, and each of them classifies an input as one of a number of classes on its own. Each decision tree will then categorize a network flow as either potentially dangerous, such as a component of a Denial of Service (DoS) assault, or a legal user flow. Random forests are very lightweight since only a small number of straightforward calculations are needed for each classification tree, despite the fact that several classification trees are used to increase accuracy. With their limited resources and need for quick processing to function at line rate, programmable switches are a strong choice for using this approach due to the simplicity of the operations carried out in each tree.*

Keywords: Denial of Service (DoS), IoT, Machine Learning, Support Vector Machine, KNN.

Date of Submission: 25-02-2023 Date of acceptance: 06-03-2023

I. Introduction

Future IoT systems must incorporate privacy and security measures because to the ease with which the physical world and computer communication networks may be integrated by the IoT and because of applications (apps) like infrastructure management and environmental monitoring. IoT systems, which combine radio-frequency identifications (RFIDs), wireless sensor networks (WSNs), and cloud computing, must safeguard user data privacy and deal with security problems like malware, eavesdropping, spoofing attacks, intrusions, distributed denial-of-service attacks (DDoS), and DoS attacks. For instance, personal information leakage must be prevented by wearable devices that gather and communicate user health data to a linked smartphone. IoT devices typically lack the processing power, memory, radio bandwidth, and battery life necessary to complete computationally demanding and latency-sensitive security tasks, especially when dealing with high data volumes.

Nevertheless, the majority of security solutions now in use place a significant computational and communication burden on IoT devices, and outdoor IoT devices like inexpensive sensors with weak security defences are typically more open to intrusion than computer systems. IoT devices can detect source nodes and defend against identity-based attacks like spoofing and Sybil attacks with the use of authentication. Unauthorized users are prevented from accessing IoT resources through access control. IoT devices may utilize servers' and edge devices' computing and storage resources for computationally heavy and latency-sensitive operations thanks to secure offloading mechanisms. Malware detection guards against malware including viruses, worms, and Trojans to prevent data loss, battery drain, and network performance deterioration on IoT devices. IoT devices must decide on a defensive strategy and the essential parameters in the security protocols for the trade-off in the heterogeneous and dynamic networks due to the growth of ML and smart assaults. This task is tough because an IoT device with limited resources typically struggles to predict the present network and attack condition accurately and on time.

II. Related Works

The examination of Internet of Things (IoT) bots against malicious distributed denial-of-service attacks by hackers is the main emphasis of the article referred in [1]. They employed the K-NN method, which uses a potentially non-parametric technique for classification called k-nearest neighbors. Whether KNN is used for regression or classification will affect the results. The benefit of using KNN is that it can handle multiclass scenarios and offer adjustable distance selection. As a result, it may offer security through biometric authentication. The future result may be expanded to include blocking and identifying suspect port entrance traffic. Author had discussed keystroke dynamics authentication using KNN in work [2]. We can distinguish between authorized and unauthorized users by collecting user behaviors

and utilizing that data to train the model. Large data storage and database-based data management present issues. KNN is an algorithm that is employed as a classifier. The benefits of doing this include Gaining access to a computing system via biometric behavioral and physiological characteristics offers further security, but there is no set protocol for a keystroke system, which is one of the disadvantages. less accurate results are produced. The result might be implemented in a system that needs security, such a financial system.

The authors of article [3] focused on a KYC (Know Your Customer)-based authentication approach for online financial services. The primary difficulties in this situation are securing the information for that Challenge Question (CQ) when login using user ID and password in order to more thoroughly verify the person. KYC must then be privatized with extensive dynamic user input. The initial data from account opening, user engagement, and dynamic updating all contribute to the KYC database's enrichment; on the other hand, users can submit more private information or arbitrary questions with answers to the KYC database to strengthen and secure the authentication process. The author of study [4] focused on user authentication using mouse movements. In this case, authentication is done once at the beginning of each session. A re-authentication system's goal is to continuously track the user's actions during the session and flag any "anomalous" behavior. Re-authentication of users aims to confirm that the present user is the legitimate user. A computer system may be vulnerable to insider assaults in the absence of a user re-authentication procedure. In particular, an unauthorized person might get access to an account either through initial authentication (for example, by stealing a password) or by simply taking advantage of an authorized user's open account who neglected to log out before leaving a computer station. The proposed approach included a decision tree and supervised learning. The benefit is that authentication makes it easier to forecast movement. The project's output authenticates computer equipment using mouse clicks and use the best machine algorithm to spot unusual behavior. The effort can be increased by detecting replay attackers who exploit vulnerabilities and by expanding the data collection for API events.

The challenges of the work discussed in [5] include identifying intruder activities that lead to IoT device malfunctions and gathering dynamic data from a large user base in order to provide the best possible authentication, verification, and identification. The Support Vector Machine Algorithm is utilized by the model (SVM). The benefit of utilizing this is that the machine learning algorithm perfectly separates the identities of users and, unlike other methods, its continuous process makes it simple to spot an intruder. Nevertheless, the drawback is that when the user doesn't utilize a mouse to operate, it becomes ineffective. The results of this initiative include Use the enrollment stage and the authentication stage, two ways that can enable two-step authentications, to distinguish between legitimate users and imposters. Future work may improve the continuous authentication system's performance or dependability, concentrate on various system configurations, and pay particular attention to emotional behavior. The work in [6] entails the Python implementation of a toll box for image processing. A numerical package offers a collection of modules that Python may quickly include. The only approach currently used to identify plant diseases is expensive, simple naked eye inspection. A genetic algorithm and MFA system are used to segment an image into its many components. The project output can be improved in the future with algorithm support for massive data collecting. Author in [7] talks about user authentication via keystroke biometrics. To create a keystroke dynamic system that offers recommendations to raise the KB system's accuracy rate. The Multilayer Preparation Neural Network (MLP-NN) approach is what we are doing. being affordable, open to the user, inventive, and offering the capability of continuous monitoring system Moreover, improved interface selection and feature fusion will be used in future work to increase KB system performance. Beyond device identification, it will broaden the approach to the boarder class of cyber attack. We can create future pathways for potential mobile IoT device research directions.

The authors in [8] discussed pressure factors and security weaknesses for massive IoT data in healthcare solutions. The most difficult assaults, including denial of service (DOS), machine middle, and dynamic instructions, are the main emphasis of this paper. SVM and KNN Adoption of IoT are the two techniques we are applying in this case. With potential benefits for healthcare diagnostics, remote monitoring, and wearable, big data in healthcare is on the rise. Health care IoT device authentication and main security analysis based on survey data has been given. The work in [9] discusses machine learning DDOS detection for consumer IoT devices. Here, collecting DOS traffic presents significant difficulties. We are employing Decision Tree and Random Forest as our two approaches. aids in increasing accuracy. It may be adaptive to both classification and regression issues to find and display DOS attack traffic coming from a smart home device. Their prior knowledge may be applied to further language machine learning research.

III. Proposed Scheme

IoT security researchers must devote a lot of attention to authentication, one of the most important security features. Currently, the majority of IoT implementations use centralized client-server architecture to connect to the cloud through the Internet. Cloud servers with significant processing and storage capabilities identify, authenticate, and connect all devices. Even when they are close to one another, IoT devices will still need to communicate through the cloud. Such a design is vulnerable to congestion points, outages, and planned assaults that might compromise network performance as a whole. Due to IoT devices' limited resource availability, the issue is made worse. Because they were not created for devices with limited resources, the existing best security practices cannot be used to safeguard the IoT ecosystem, leaving billions of vulnerable devices. For many IoT-related topics, including security, conventional methods—such as clustering, protocols, applications, data aggregation, services, architectures, and resource allocation—work effectively. This exemplifies one of the authentication network scenarios in which the IoT device authentication is provided after the malware assault is detected by the software.

We may use supervised learning to assess the behaviour of the programmes while they are running in malware detection. Because IoT devices are susceptible to virus, malware, and DoS assaults, we use the Network layer of TCP/IP to give them the protection they need. The Internet of Things (IoT) devices filter TCP packets and choose features from a variety of parameters, such as frame numbers, frame length, etc. The feature extractor module, in contrast to the previously described modules, is situated in a P4-enabled switch's programmable data plane. This module does computations for each incoming packet, starting with the flow to which it belongs in the first place. Next, in order to extract characteristics to be used by the RF, it makes use of metadata data, such as the packet's ingress time and packet length, as well as any pertinent header fields, such the urgent and push flags of an IPv4 header or the window size value of a TCP header. The characteristics are labeled and saved in the database after selection and extraction. The model is created by training the data with the Random Forest Machine Learning Classifier Algorithm using the supplied data. Certain features, including PSH flags, are depending on values from the IPv4 header. The block diagram for the proposed scheme is shown in Figure 1.

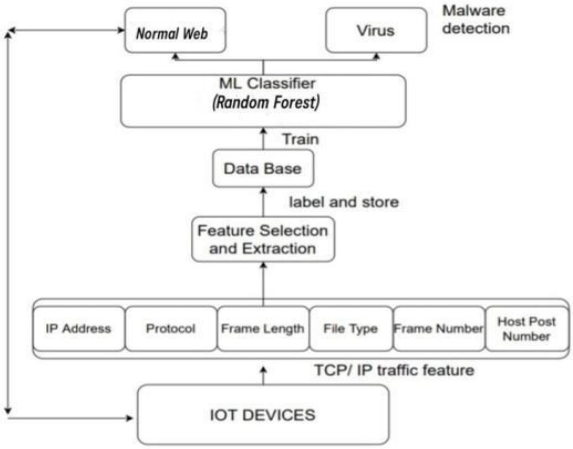

Figure 1. IoT Security Analysis

Popular machine learning algorithm Random Forest is a part of the supervised learning methodology. It may be applied to ML issues involving both classification and regression. It is built on the idea of ensemble learning, which is a method of integrating many classifiers to address difficult issues and enhance model performance. Random Forest, as the name implies, is a classifier that uses a number of decision trees on different subsets of the provided dataset and averages them to increase the dataset's prediction accuracy. Instead than depending on a single decision tree, the random forest uses forecasts from each tree and predicts the result based on the votes of the majority of predictions. The greater number of trees in the forest leads to higher accuracy and prevents the problem of over fitting. The Random Forest algorithm is explained in Figure 2.

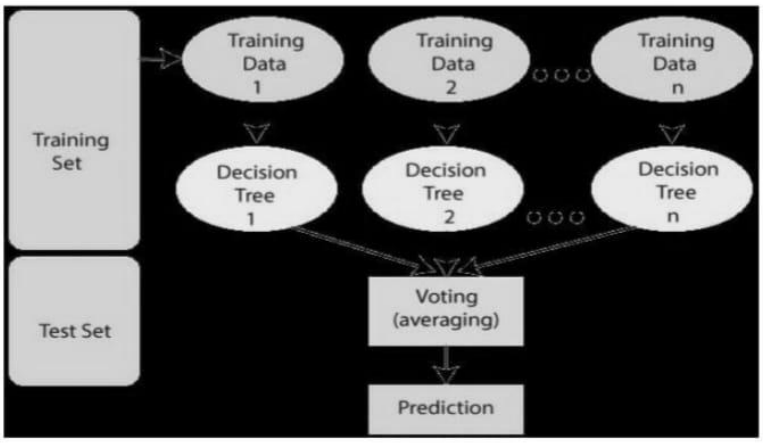

Figure 2. Random Forest algorithm

Some decision trees may predict the proper output, while others may not, since the random forest mixes numerous trees to forecast the class of the dataset. Yet when all the trees are combined, they forecast the right result. For the dataset's feature variable to predict true outcomes rather than a speculated result, there should be some real values in the dataset. Each tree's predictions must have extremely low correlations. First, N decision trees are combined to generate the random forest, and then predictions are made for each tree that was produced in the first phase. The steps and picture below can be used to demonstrate the working process. Choose K data points at random from the practice set. Create the decision trees linked to the chosen data points (Subsets). For any decision trees you intend to construct, choose N. Find each decision tree's forecasts for any new data points, then place them in the category that receives the most votes. Random Forest is mostly utilized in four industries: Banking: This algorithm is mostly used in the banking sector to identify loan risk. Medicine: This method may be used to identify illness patterns and risk factors. Land Use: With this technique, we can locate places with comparable land uses. Marketing: This algorithm may be used to find marketing trends. C. Training of convolutional neural networks. Python is an interpreted, high-level, general-purpose programming language that was first introduced in 1991 and was created by Guido Van Rossum. The following characteristics of the Python language: Easy to Learn and Use Python is an easy-to-learn and use language. It is an efficient high-level programming language for programmers.

Table 1. Features Implemented

Destination Port	
Flow Duration	
Packet Count	
Header Length Sum	
Initial Window	
ACT Data Count	
PSH	Flag Count
URG	
Packet Length	Total
	Minimum
	Maximum
	Mean
Inter Arrival Time	Total
	Minimum
	Maximum
	Mean
Segment Size	Total
	Minimum
	Maximum
	Mean

The length of a flow is the amount of time that has passed since the first packet in that flow arrived up until the present moment. We use a register to keep the timestamp of the first packet in each flow. We use the current packet's standard metadata ingress timestamp to estimate the time; the number of packets in that flow that the switch has seen is represented by packet count. Header length sum is the total, in bytes, of the header lengths of each packet in that flow. This value is kept in a register and is increased by the header length of every incoming packet from that flow. If a TCP header is present, the initial window is taken from the first packet in the flow. After the first packet in that flow, this value is kept in a register and is never altered; ACT The number of packets in that flow that included at least one byte of payload is known as the data count. PSH Flag Count is the number of packets in that flow that had the PSH flag set, as determined by the IPv4 header, and is kept in a register and increased by one for each arriving packet from that flow that has a payload length of at least one byte. For each incoming packet from that flow that has the PSH flag of the IPv4 header set (not zero), this value is put in a register and increased by one. URG Flag Count is the total number of packets from that flow that had the URG flag set, as determined by the IPv4 header. For each packet that arrives and is part of that flow and has the URG flag of the IPv4 header set (not zero), this value is placed in a register and increased by one. Packet length is determined by the number of bytes in each packet that is part of that flow.

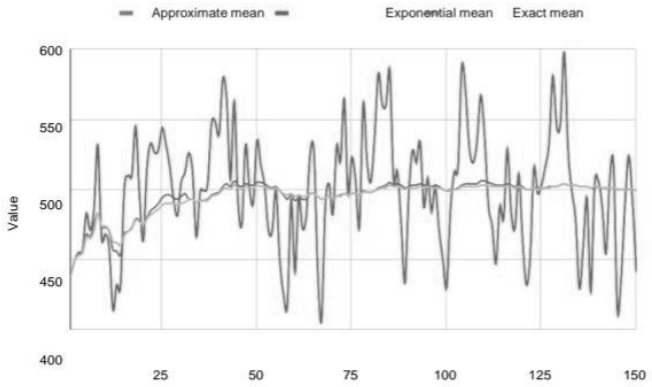

Figure 3. Mean values computed by different approaches

We plot the precise value as shown in Figure 3, EWMA, and the computed value of our technique in blue and red, respectively (in yellow). As we can see, our method (in blue) comes closer to the actual number (in yellow). EWMA's sensitivity to new values that happen to be outliers is an often unwanted attribute. When employing a weight of 0.5, as in the example given in (Busse-Grawitz et al., 2019). We also used the weight of 0.5 for EWMA in our testing to compare that behaviour with our strategy. This behaviour is evident in the way the EWMA value (shown in red) quickly increases, much beyond the precise amount seen in the yellow curve.

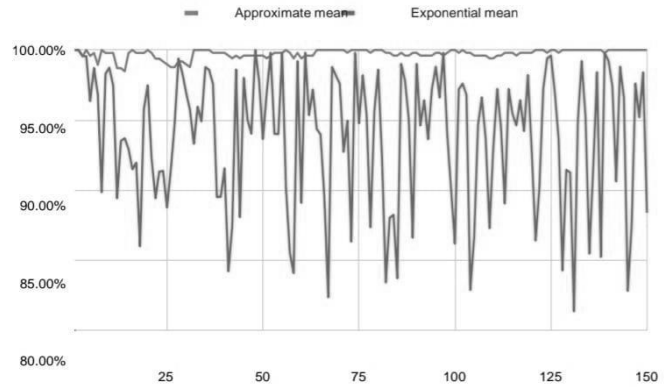

Figure 4. Comparison between proposed approach and EWMA approach

With respect to the precise value, we depict the accuracy attained by our method (in blue) and EWMA (in red). The Figure 4. demonstrates how much more accurate our method (in blue) is than EWMA (in red). Furthermore, we observe sharp drops in accuracy, which, as already said, is probably a result of the great sensitivity to outliers that EWMA experiences when using a weight of 0.5.

IV. Results and Discussions

We contrast the various arrangements of the meta-parameters that make up a Random Forest model. We specifically highlight each model's F1-Score, a well-liked statistic for gauging the effectiveness of ML students. In addition, taking into account legitimate traffic as the positive class, we also evaluate each model's accuracy, precision, and recall. With legitimate traffic as the positive class, accuracy refers to the percentage of samples that the classifier correctly predicted. The F1-score each of the trained RF models produced may be shown. Every model, including those with fewer trees and shallower woods, has an F1-Score higher than 0.85, as can be seen in the image. Yet, even with a modest number of classifiers per forest, raising the maximum depth gives trained forests a significantly higher F1-score. As a consequence, even a modest number of trees can produce respectable results while simultaneously restricting the maximum depth. contrasts the results each of the models was able to provide in terms of accuracy, precision, and recall. As we can see, trained models often have a high degree of accuracy. This shows that the model accurately categorizes the majority of predicted flows as valid, given that legitimate traffic is the positive class. The taught learners often attain the greatest parameter, which is accuracy, but the other metrics are not considerably worse. The worst-case scenario for processing time on the data plane is always the number of table applications equal to the maximum depth of layers times the number of trees. A forest with 5 trees that restricts the maximum depth of each tree by 6 will, at worst, make 5 6 comparisons, or 30 in this example. Hence, the processing time is constrained by O(NM), where NM is the maximum depth and N is the number of trees. The comparison of different matrices between best trained models is shown in Figure 5.

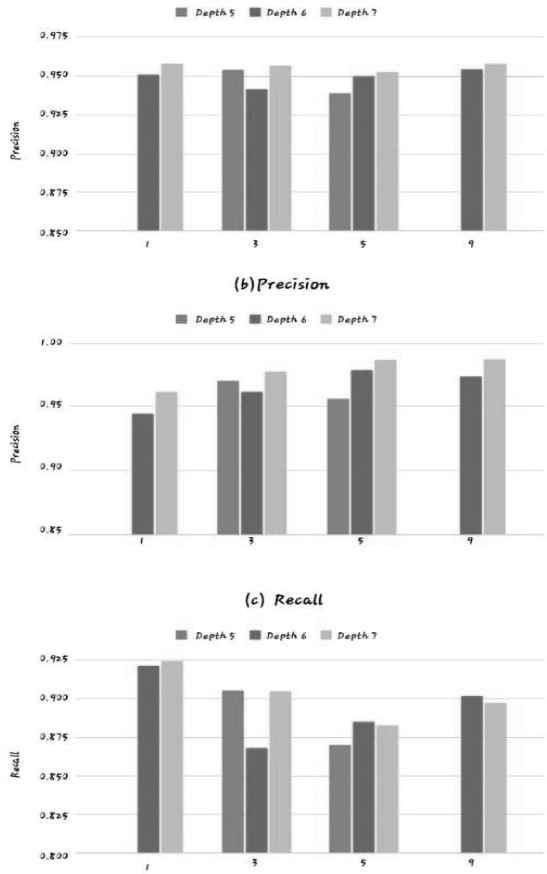

Figure 5. Comparison of different matrices between best trained models

Each node is mapped into a single match action table item in terms of memory. A tree with one layer will thus only have one node (the root), but a tree with two complete layers would have three nodes, a tree with three full layers will have seven nodes, and so on. The trees produced by induction algo-rhythms may include leaf nodes in any layer, not only the last one, hence not every layer will have as many nodes as it might. As a result, in the worst-case situation, each tree can only use up to O (2M) bytes of memory, where M is the tree's maximum depth. Because there are N trees, the overall memory use is capped at O (N (2M)). In terms of the number of trees in the forest and the maximum number of layers per tree, we first show the worst-case scenario analysis of the processing and memory utilization.

V. Conclusion

The conclusion of this study is presented in this chapter, along with an overview of the contributions and future work. This study introduced BACKORDERS, a technique for categorizing network flow in programmable data planes. A Random Forest classifier is implemented by the system, and its structure is mapped to fit in a P4-enabled switch. The information present in each node is translated into match action tables using this technique. We are able to evaluate nodes sequentially by translating the data structure that is prone to recursion into a collection of table entries. We determine the properties of each flow being watched once the learner has been mapped into the programmable data plane.

These characteristics are later used by Classification Trees to classify network flows and determine whether they are malicious or genuine. We effectively classify a sample using a previously trained Random Forest by leveraging match action tables, a structure that is well suited for programmable switches. We also suggest, put into practice, and assess a technique for roughly estimating mean values in the data plane. Although though our technique just makes use of additions, subtraction, and bitshifts, it produces an approximate result that is quite near to the real number. In order to replace moving averages in systems that rely on mean values as a component of their decision logic, exponentially weighted moving averages have been proposed. We intend to optimize several system components in further development. More specifically, we must reduce the amount of memory used by each network flow in order to classify many flows simultaneously. According to analysis, our currently implemented functionalities may use too much RAM. Because that programmable switches typically have dozens to hundreds of thousands of flows, each feature must only require a small amount of memory in order to be implemented for each flow. As a result, we think there are methods that may be used to cut the amount of bits used for each flow's various characteristics. In subsequent work, we intend to develop the creation of code for the feature extractor module using a more intelligent technique in order to significantly reduce the memory used by our system. We will only include registers, metadata variables, and actions for features that were used by the inserted RF rather than using a register and metadata for every feature that is accessible. This strategy would save memory use by removing some registers for features that the forest did not use. Therefore, it would be challenging to introduce a new forest into the data plane while the P4-enabled switch is running. As a result, in our future work, we want to assess the trade-offs between our already-implemented strategy and this suggested optimization.

REFERENCES

[1]. K. Gurulakshimi, A. Nasrani, "Analysis of IoT bots against DDOS attack using Machine Learning algorithm", Proceedings of the 2nd IEEE International conference on trends in Electronics and Informatics **(2018),** pp. 13500-13503.

[2]. Amol Pilgaonkar, Ankita Slunk, Nikita Gupta, Vikrant Sharma. "Authentication through key stroke dynamics using KNN". Journal of Multidisciplinary Engineering Science and Technology (JMEST),vol. 4, issue 5, **(2017)** May,pp. 7265-7267.

[3]. Prakash Chandra Modal, Rupam Mohammad Deb and Mohammad Nurul Huda, "KYC based authentication method for financial service through the interne". Proceedings of the IEEE 19th International Conference on Computer and Information Technology, December 18-20, **(2016),** North South University, Dhaka, Bangladesh, pp. 235-24.

[4]. Maja Psara, Carla E. Broadley. "User Re-Authentication via Mouse Movements". Department of Computer Science Tuffs University , pp. 234-242.

[5]. Suhail jaded Quraishi, Sorabjee Singh Beedi. "On mouse dynamics as continuous user authentication". International journal of scientific and technology research volume 8, pp. 1052-1057.

[6]. Mohamad Amine Farrago, Leandro's Mulgaras and Abdelhamid drab." Authentication for mobile IoT Devices using Bio features" Volume **(2019),** pp. 519 - 539.

[7]. Md Liaqat Ali, John V. Monaco, Charles C.T. and Mekong Quit. "Keystroke biometric system for user authentication "Springer Science and Business media New York, **(2016),** pp. 617-635.

[8]. Madhu Sharma Gaur, Sanjeev Kumar, Navneet K Gaur and Prem saggar Sharma." Persuasive Factors and Weakness for Security vulnerabilities in big IoT data in Health care Solution". Journal of Physics, conference series. **(2021),** pp.67-81.

[9]. Rohan Doshi, Noah Calthorpe, Nick Feemster. "Machine Learning DDoS Detection for Consumer IoT Devices". 2018 IEEE Security and Privacy workshop (SPW), **(2018),** pp. 29-35.