DETECTION OF ZERO-DAY SECURITY THREAT USING MACHINE LEARNING

Manish A[1], Vivek S K[2], Vishal Kiran Gowda[3], Mahesh Shetty[4], Madhuri J[5]
[1-4]Student, Dept. of Computer Science and Engineering, Bangalore Institute of Technology, Bengaluru, Karnataka, India
[5]Assistant Professor, Dept. of Computer Science and Engineering, Bangalore Institute of Technology,
Bengaluru, Karnataka, India

Abstract: Nowadays highly-skilled attackers can find the vulnerabilities of many networked applications. Meanwhile, the risk of a data breach increases dramatically as a software or application vulnerability always remains without a patch. By exploiting such vulnerability (called zero-day), hackers gain entry to the target network and can steal sensitive data. It is challenging to detect zero-day with traditional defenses because signature information in zero-day attacks is unknown. Consequently, a novel security solution is required that will discover zero-day attacks and estimate the severity of identified zero-day vulnerability.
Machine Learning (ML) and Deep Learning (DL) have been used for building Intrusion Detection Systems (IDS). The increase in both the number and sheer variety of new cyber-attacks poses a tremendous challenge for IDS solutions that rely on a database of historical attack signatures. Therefore, the industrial pull for robust IDSs that are capable of flagging zero-day attacks is growing. Current outlier-based zero-day detection research suffers from high false-negative rates, thus limiting their practical use and performance.
Keywords-Machine Learning; Deep Learning; Random Forest; Naïve Bayes.

I. INTRODUCTION

Zero day (0-day) is a software or application vulnerability that can be exploited by a threat actor to gain entry to a target network. In this way, hackers or attackers can steal sensitive information such as legal documents and enterprises data.
Current organizations and enterprises have taken great care to secure their networks. However, they are still at risk even with responsible and sustained investment in their defenses. This happens because attackers can bypass organization's security through unknown vulnerabilities, which are not listed by security persons. In a well-guarded network, a loophole may be revealed by the persistent probing of a determined hacker. Attackers can leverage vulnerabilities which are present in network configuration to penetrate the target network. Cyber criminals are increasing the success rate of attacks by finding and exploiting zero-day vulnerabilities. Regularly, the information about vulnerability is not available until zero-day attacks have already taken place. As a result, it is difficult to identify and analyze attacks that use zero-day exploits. With zero-day vulnerability in hand, a hacker has two possible choices: He may help the software vendor by providing him with information about the discovered vulnerability; He may sell the crucial information to the black market broker, who may further sell the identified exploit at highest rate. Zero-day exploits have an element of surprise as they are previously unrevealed; an attacker incorporates the zero-day exploit into their charted list of vulnerabilities and once the penetration program process and payload are concocted, attack is launched.
Attackers are highly-skilled and the discovered vulnerability can remain unknown to the public

for months or even years. This fact provides plenty of time to attackers to cause irreparable harm. According to FireEye, a typical zero-day attack may last for 310 days on average. Therefore, dealing with zero-day is clearly a challenging task.

II. LITERATURE SURVEY

A. MALWARE CLASSIFICATION USING STATIC ANALYSIS

Several security researchers have applied domain level knowledge of portable executables (PE) for static malware detection. At present, analysis of byte n-grams and strings are the two most commonly used methods for static malware detection without domain level knowledge. However, the n-gram approach is computationally expensive and the performance is considerably very low .It is often difficult to apply domain level knowledge to extract the necessary features when building a machine learning model to distinguish between the malware and benign files. This is due to the fact that the windows operating system does not consistently impose its own specifications and standards. Due to constantly changing specifications and standards from time to time, the malware detection system warrants revisions to meet future security requirements. To address this, has applied machine learning algorithms (MLAs) with the features obtained from parsed information of PE file. They adopted formatting of agnostic features such as raw byte histogram, byte entropy histogram which was taken from and in addition employed string extraction. MalConv compared these classical machine learning models with the deep learning approach. They have made the dataset with features as well as raw files and the associated code publicly available since deep learning models require more exploration and require further research.

B. MALWARE CLASSIFICATION USING DYNAMIC ANALYSIS

Malware analysis methods based on Dynamic analysis are more robust to obfuscation methods as compared to Static analysis. In features from 5 minutes API calls were extracted and passed on to CNN for classification using Dynamic analysis. They used around 170 samples and obtained 0.96 for Area under Curve (AUC) as the quality measure. In shallow feed forward network with feature sets of API calls were obtained from a large number of samples of benign and malware that were collected privately. It performs well as compared to the existing approaches but it lacks the study on the speed of execution, which is important for real-time deployments. In experiments with echo state networks (ESNs) and RNN were conducted to learn the language of malwares. In most of the experiments, the ESNs performed well in comparison to RNN. In experiments were conducted to determine when to stop the malware execution with respect to the network communication. This method has reduced the total time taken by 67% compared with conventional methods. In the application of RNN and its variant long short term memory (LSTM) and CNN were employed for malware classification with API call long sequences as features.

C. MALWARE CLASSIFICATION USING IMAGE PROCESSING TECHNIQUES.

Malware attacks are on the rise and in recent days, new malwares are easily generated as variants of existing malware from a known malware family. To overcome this problem, it is important to learn the similar characteristics of malware that can help to classify it into its family. Several studies conducted in have taken advantage of the fact that most malware variants are similar in structure, with digital signal and image processing techniques used for malware categorization. They have transformed the malware binaries into gray scale images and report that malwares from the same malware family seem to be quite similar in layout and texture. Since image processing techniques require neither disassembly nor code execution, it is faster in comparison to the Static and Dynamic analysis.

III. DISADVANTAGES AND LIMITATION OF EXISTING SYSTEM

DISADVANTAGES:

Although dynamic analysis surpasses the static analysis in many aspects, dynamic analysis also has some drawbacks. Firstly, dynamic analysis requires too many resources relative to static analysis, which hinders it from being deploying on resource constraint smart phone.

On contrast to the above mentioned methods, anomaly detection engine in our proposed detection system performs dynamic analysis through Dalvik Hooking based on Xposed Framework. Therefore, our analysis module is difficult to be detected by avoiding repackaging and injecting monitoring code.

Overall, previous work focuses on detecting malware using machine learning techniques, which are either misuse-based detection or anomaly-based detection. Misuse based detector tries to detect malware based on signatures of known malware.

LIMITATIONS OF EXISTING SYSTEM

- Time consuming more
- Accuracy is high

IV. PROPOSED ARCHITECTURE

The proposed idea of this project is to provide a solution to detect zero-day malware by using hybrid technologies. There are 4 phases involved in our methodology namely Malware Data Sets (MDS), Analyzing MDS, Learning Algorithms, Detect and classify the attacks.

Firstly, we collect the sample of the known malware referred to data sets. In the first phase, the gathered data sets are analyzed. In the second phase, we use learning algorithms that correlates the relationships between the malware and would be able to detect if the network is secure or not.

In the third phase, we use learning algorithms that correlates the relationships between the malware and would be able to detect if the network is secure or not. In the final phase, we apply Detection Methods and train them so that they can be able to detect and classify the malware and give appropriate output.

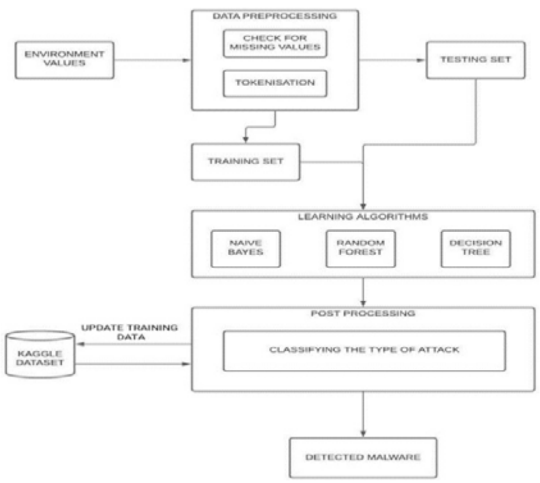


Fig 1: System Architecture

The architecture of a system describes its major components, their relationships, and how they interact with each other. Software architecture and design includes several contributory factors such as Business strategy, quality attributes, human dynamics, design, and IT environment. Initially we input the environmental values Where the data processing takes place which checks for the missing values while the tokenization takes place here we label, normalize since the dataset is too large we trim it. We split the data set into training and testing as we pass to the learning algorithms we have this in order to Correlate the relationships between the Malware and would be able to detect if the network is secure or not. Post processing of the system classifies the type of attack by comparing it with the Kaggle data set In return the type of malware shall be detected.

The interface design shows a pictorial representation of the main interfaces required for the different system components and actors to interact with each other. An interface is represented by a circular symbol as shown in the below figure.

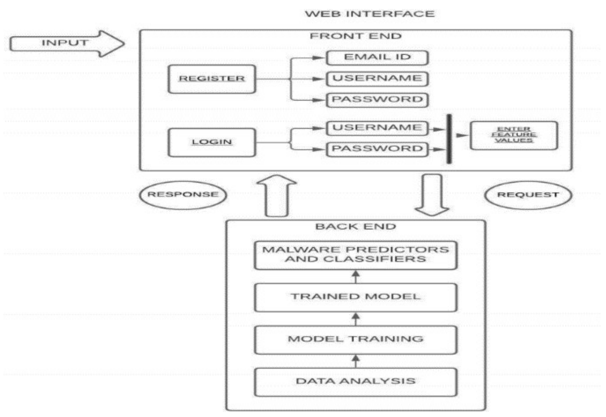


Fig 2: Interface design

User: The user is one of the important aspects of this and he runs the application to detect exploits. Upon running the application, he gets the information whether malware is detected or not.

System Interface: It takes care of the malware in the background. The malware is broken into tokens and then marched for precedence. If the precedence isn't matched then the data is acquired and sandboxed and it is modified for the procedure of static analysis. The algorithm is used to detect whether the acquired data is a malware or not.

Component Design/Module Decomposition

Modules are system structures that have well-established reliability evaluation algorithms. The technique of modular decomposition identifies such modules in complex systems and uses their efficient reliability evaluation algorithms for reliability evaluation of the whole system. In this article, we provide the formal definitions of modules and modular decomposition for system reliability analysis, illustrate the technique of modular decomposition, and outline reliability evaluation algorithms for several well-studied modules. References for further reading in this area will also be provided.

ADMINISTRATOR MODULE:

This module is the one who organizes all the user's data, allow the various users to create the id, manages dataset and maintains statistics.

Stimulus/Response Sequences:

Stimulus: Administrator logins with his/her user id and password.

Response: If it is validated user then it gets accessed or denied.

Stimulus: Administrator provides user id and password to all the users.

Response: Allots the relevant user id and password to the user. Module Description

USER MODULE:

A user is the one that creates an id and then logs in with the help of it and then enter the feature values to check the network connection status.

Stimulus/Response Sequences:

Stimulus: User logins with id and passwords.
Response: if it is validated then it gets accessed or denied.

Stimulus: User enters the related feature values.
Response: Gets information on the security status of network connection based on the input.

ZERO DAY DETECTOR/ CLASSIFIER:

In this module, we collect the data from Kaggle dataset. The dataset has abundant data pertaining to the different type of attacks that can take place over a network connection. We trim the dataset to a certain value as the size of the dataset is too big. We extract the feature names and the related values and check whether any fields are empty. We deal with the missing values by either deleting the related row or using mean, median to find the average value. We assign unique labels to normalize the column. We split the dataset to train the model. We apply learning algorithms (Naive Bayes, Decision tree & Random Forest) to classify and detect the security status of a network.

V. ALGORITHM

Random Forest Algorithm

We can understand the working of Random Forest algorithm with the help of following steps −

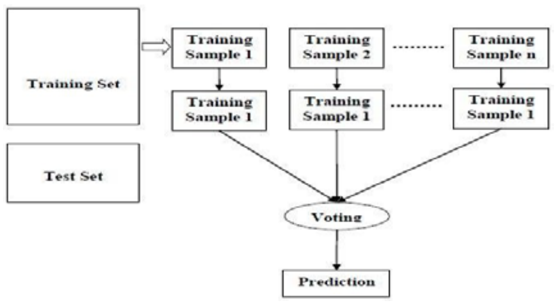
Step 1 − First, start with the selection of random samples from a given dataset.

Step 2 − Next, this algorithm will construct a decision tree for every sample. Then it will get the prediction result from every decision tree.

Step 3 − In this step, voting will be performed for every predicted result.

Step 4 − At last, select the most voted prediction result as the final prediction result.

Fig 3: Random Forest Algorithm



Naive Bayes

This Naive Bayes tutorial is broken down into 5 parts: Step 1: Separate By Class

We will need to calculate the probability of data by the class they belong to, the so-called base rate.

This means that we will first need to separate our training data by class.

Step 2: Summarize Dataset

We need two statistics from a given set of data.

We'll see how these statistics are used in the calculation of probabilities in a few steps. The two statistics we require from a given dataset are the mean and the standard deviation (average deviation from the mean).

Step 3: Summarize Data By Class

We require statistics from our training dataset organized by class.

Step 4: Gaussian Probability Density Function Calculating the probability or likelihood of observing a given real-value like X1 is difficult.

One way we can do this is to assume that X1 values are drawn from a distribution, such as a bell curve or Gaussian distribution.

Step 5: Class Probabilities

Now it is time to use the statistics calculated from our training data to calculate probabilities for new data.

Probabilities are calculated separately for each class. This means that we first calculate the probability that a new piece of data belongs to the first class, then calculate probabilities that it belongs to the second class, and so on for all the classes.

Decision Tree Algorithm

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).

Step-3: Divide the S into subsets that contains possible values for the best attributes.

Step-4: Generate the decision tree node, which contains the best attribute.

Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

VI. EXPERIMENTAL RESULTS

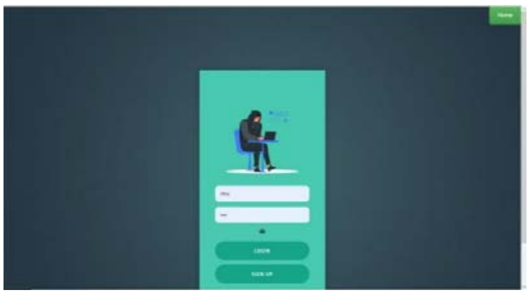


Fig 4: Home Page

Computer must be a safe space for users, user's data must be protected.

The sudden uproar of malware is problematic, different exlopits detection methods must be implemented in new and different ways.

Fig 5: Login Page where the user gets to register and sign in

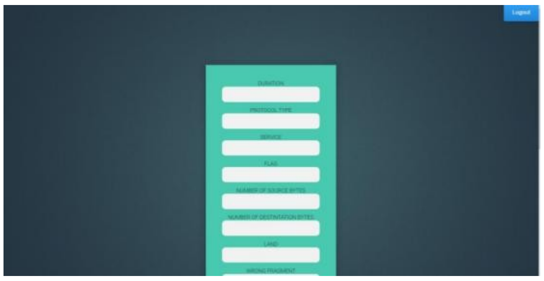


Fig 6: Input Page

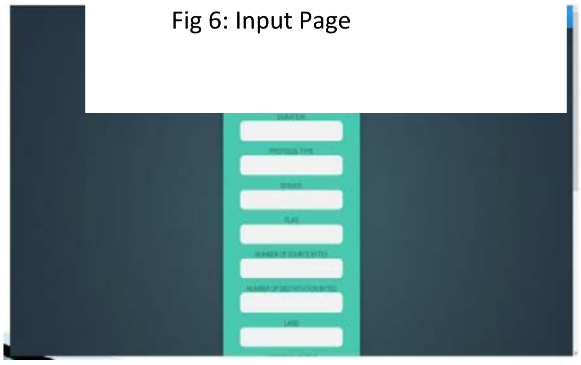


Fig 7: Result page displaying the type of attack

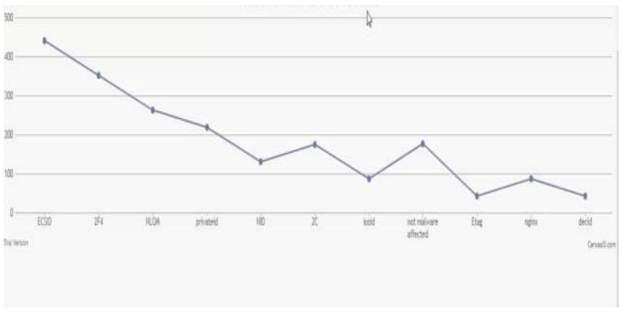


Fig 8: Graphical Representation

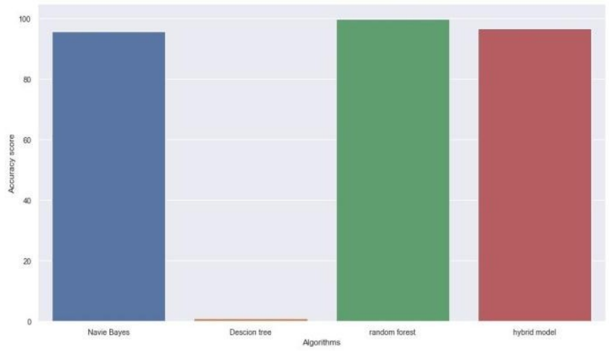


Fig 9: Graphical representation of accuracy rate for each algorithm

VII. CONCLUSION AND FUTURE WORK

It is necessary in order to detect zero-day attack malware which leads an opening of a platform to provide various malware to perform an attack or to overload the system. It play vital role in the field of security many attacks have been generated. Rapid growth of malware directs most researchers to implement new approaches to defeat the attacks and developed countermeasures. From a student using his computer at home to multinational companies with transactions worth millions everyone's devices store data important and in some cases extremely important to them. The scope of use of our application is very vast. It can be used be organizations scan all their devices on specific networks and manage or assess the security of their devices. It can be used by individuals at home to assess the security of the devices connected to their network at home. With the rise in number of cyber- attacks and other cyber related illegal activities there arises the need for better security and this form of network security is a significant concern. Our application has a role to play everywhere and anywhere anyone wants to pro-actively assess the security of their devices and pre- emotively avoid potential threats and the damage caused by them.

With further research and extensive efforts, the use of machine learning and artificial intelligence a more proactive approach or a more proactive and accurate application without human oversight can be expected in the future. Security of computer systems and networks is an evergreen issue great importance the threats evolve as technology evolves and this makes network security tools everlastingly relevant.

REFERENCES

[1] Kaur, R.; Singh, M., "Efficient hybrid technique for detecting zero-day polymorphic worms," Advance Computing Conference (IACC), 2014 IEEE International, pp.95-100, 2122 Feb. 2014.

[2] M. Rathor, D. M. Dakhane, ―Predicting Unknown Vulnerabilies in Network Using K-zero Day Safety Technique‖, International Journal of Advanced Research in Computer Science and Software Engineering 5 (4), pp. 221- 224, April- 2015.

[3] C. Joshi, U.K. Singh, ―A Review on Taxonomies of Attacks and Vulnerability in Computer and Network System‖. International Journal of Advanced Research in Computer Science and Software Engineering (IJRCSSE) Volume 5, Issue 1, January 2015, pp 742- 747.

[4] C. Joshi, U.K. Singh, ―ADMIT- A Five Dimensional Approach towards Standardization of Network and Computer Attack Taxonomies‖. International Journal of Computer Application (IJCA, 0975 – 8887), Volume 100, Issue 5,August 2014, pp

[5] C. Joshi and U. Singh, ―Analysis of Vulnerability Scanners in Quest of Current Information Security Landscape‖ International Journal of Computer Application (IJCA, 0975 – 8887), Volume 145 No 2, pp. 1-7, July 2016.

[6] C. Joshi, and U. K Singh, ―Performance Evaluation of Web Application Security Scanners for More Effective Defense‖ International Journal of Scientific and Research Publications (IJSRP), Volume 6, Issue 6, pp 660- 667, June 2016, ISSN 2250-3153.

[7] Z. Li, M. Sanghi, Y. Chen, ―Hamsa : Fast Signature Generation for Zero-day Polymorphic Worms with Provable Attack Resilience‖, Proceedings of the 2006 IEEE Symposium on Security and Privacy (S&P'06).

[8] M. Frigault, L. Wang, A. Singhal, and S. Jajodia,―Measuring network security using dynamic bayesian network,‖ in Proceedings of 4th ACM QoP, 2008.

[9] A. Lelli. (2010, Jan.) The trojan. hydraq incident: Analysis of the aurora 0-day exploit, Available:http://www.symantec.com/connect/blogs/trojanh ydraq-incidentanalysis-aurora-0day-exploit

[10] R. Goyal and P. Watters, ―Obfuscation of stuxnet and flame malware,‖ in Proc. 3rd Int. Conf. on Applied Informatics and Computing Theory, pp. 150–154, Barcelona, Oct. 2012.

[11] McAfee Labs 2017 Threats Predictions‖, Intel Security, November 2016.

[12] P. Ammann, D. Wijesekera, and S. Kaushik,
―Scalable, graph-based network vulnerability analysis,‖ in Proceedings of ACM CCS'02, 2002.

[13] A. AlEroud, G. Karabatis, ―Toward Zero-day Attack Identification Using Linear Data Transformation Techniques‖, IEEE 7th International Conference on Software Security and Reliability, pp 161-168, 18 - 20 Jun 2013.

[14] Symantec. (2011, Nov.) W32.duqu the precursor to the next stuxnet.