Detection of Mobile Keyloggers Using Deep Learning

Ch. Madhuri, K. Rama Siritha, S. Sai Ram, G. Rama Koteswara Rao

Abstract: Keylogger is a tool which is used to record every keystroke made on the machine. It is used for gaining sensitive information without the knowledge of the owner by the attacker or any cybercriminal. By identifying the keyloggers we can prevent huge amount of data loss including sensitive information like personal details, credit card data, login credentials, passwords of any online banking and e-commerce websites, etc. The Keyloggers intrude our privacy by keeping track of the websites we visit, media files opened, passwords we type on the website. The deep learning algorithm gradient descent is used for detecting mobile keyloggers in the APK files.

Index Terms: Keylogger, Deep Learning, Gradient Descent, APK files.

I. INTRODUCTION

The invasion of keyloggers into a system causes a loss of sensitive information. A keylogger is a tool which records the keystrokes on a computer. They are induced into the machine in order view or monitor the user activity by logging keystrokes and intentionally delivering them to an outsider or a third party. The basic level keyloggers looks like they are absolutely harmless. If it is used by a hacker or any cybercriminal, it is a serious threat to the user's data as they record the keystrokes to know the passwords and sensitive information. By using the keyloggers, the hackers have the benefit of accessing the account numbers and PIN of our bank accounts, email ids, passwords of any online banking and e-commerce websites, etc. Keyloggers can be used as a spying tool to compromise business and company's data. Keyloggers can be sent through email, chats, text messages or even social networks. Keyloggers are hard to detect and can be deployed remotely via a software vulnerability attack. They are easy to write and work on all computing platforms.

Types of Keyloggers:

Software Keyloggers: These keyloggers track the system by collecting the keystroke data within the particular operating system, store them on any memory location and send them to the attacker who injected the keyloggers. The software program of keylogger consists of two files, a dynamic link library file that records and an executable file that installs and triggers the DLL file to function.IT organizations uses keyloggers to trace and correct the technical problems related to computer networks and business networks. Families and business people can use keyloggers legally in order to track the network usage without their user's knowledge. Even the Microsoft team has publicly admitted that the latest version of its operating system has the built-in keylogger to improve writing and typing skills.

There are different types of software keyloggers:

Hypervisor-based: This type of keyloggers can be present directly in the operating system and it remains intangible or untouched. It will become a virtual machine. Blue Pill is considered as one of the examples of hypervisor based keylogger.

API-based: These keyloggers are attached to the keyboard APIs inside a running application. It acts like a normal application instead of having key logging malware. Whenever the user gives a keystroke, the keylogger receives an event and records it.

Kernel-based: The keylogging program obtains the administration access on the machine by hiding itself in the operating system and records the keystrokes that are going through the kernel. These type of keyloggers are difficult to identify for user-based applications because they do not have administration approach. The operating system gains unauthorized access to the hardware keyloggers as they are implemented as rootkits. This type of keyloggers can act as keyboard device drivers.

Memory-Injection-based: In these types of keyloggers, the memory tables are related with the viewer and various other system functions are altered by their logging function. This technique is used by attackers to bypass Windows User Account Control by patching the memory tables or injecting directly into the memory.

Hardware Keyloggers:

These are connected in between the keyboard and the computer. The hardware keyloggers come in three types.[6]

• Inline devices: These are linked to the cable of the keyboard.

•Devices that can be installed within the regular keyboards.

•Replacement keyboards containing the already installed keylogger.

These keyloggers does not depend on software that has to be installed as they exist in a system at a hardware level.

Revised Manuscript Received on June 3, 2019

Ch.Madhuri, Information Technology, VR Siddhartha Engineering College, Vijayawada, India.

K.Rama Siritha, Information Technology, VR Siddhartha Engineering College, Vijayawada, India.

S.Sai Ram, Information Technology, VR Siddhartha Engineering College, Vijayawada, India.

Figure1:Inline devices

Firmware-based: The events of keyboard has to be changed to record these events because they are handled and managed by the BIOS level firmware. Physical access is required to the machine in-order to create separate hardware that it is going to run on.

Keyboard hardware: Hardware circuit acts as medium in between the keyboard and the computer for keystroke logging. Some of the USB connectors acts as the hardware keyloggers for computers as well as laptops. The hardware keyloggers are not detected by any software as they are not installed on the user computer's operating system. But if it is installed as an inline device between computer and keyboard, physical presence can be detected.

Problem Statement:

Due to the vulnerability of mobile devices than computers, mobile devices are the main targets for malicious applications. Cyber criminals make use of the malicious or malware software to make use of mobile devices like Smartphones. Mobile users can email, take advantage of online banking, buy goods, and use social networking sites. The attackers are attracted to steal the information because of the money transactions through the mobile phones. They may use the information for mischievous activities.

By downloading the harmful applications unknowingly the mobile devices are malware contaminated. The destructive applications are designed to look like genuine applications by the cyber criminals placed online, for example free market applications. While downloading the applications from these free application markets, users should be careful. Taking these issues into account, mobile devices should have an appropriate mobile security application to detect the problem of keyloggers present in it.

II. RELATED WORK

Stefano Ortolani, Cristiano Giuffrida, Bruno Crispo had done a milestone work for detection of user-space keyloggers in the background process which registers operating system supported hooks to record every keystroke issued by the user into the current foreground application using Black-box approach[1]. In this work, their goal is to prevent user-space keyloggers from stealing the confidential data originally intended for a legitimate foreground application. It constantly monitors the input and output activities generated by the keyloggers. The keystroke patterns can be artificially injected into the systems and then the best input pattern is chosen to improve our detection rate.

Hugo Gascon, Fabian Yamaguchi, Daniel Arp, Konrad Rieck evaluated that Detection of android malware applications are based on permission and API usage or the identification of expert features[5]. It deals with how recent developments in machine learning classification of graphs can be effectively applied to this particular problem. The proposed method for malware detection is based on embedding's of function call graphs with an explicit feature map inspired by a linear-time graph kernel or neighborhood hash graph kernel.

Irfan Bulut, A.Gokhan Yavuz looked over a novel model based on deep learning for prediction of mobile malware without requiring execution in a sandbox environment[4]. Application permissions were used as features. After optimizing their weights with automatic encoder and they were classified with a multilayer perceptron with an accuracy of 93.67%.

Shifu Hou, Lifei Chen, Yanfang Ye gave a dynamic analysis method named Component Traversal that can automatically execute the code routines of each given android app as completely as possible[3]. Based on the extracted system calls for the linux kernel, we further build the weighted directed graphs and then apply a deep learning framework based on graph-based features for newly unknown detection of Android malware. This System has also been integrated into a commercial Android antimalware software.

Zarni Aung, Win Zaw have worked on a framework is provided for classifying Android applications using machine learning techniques (K-means) whether they are malware or normal applications[2]. Android-based smartphone users can get free apps from the Android app market and these apps are not certified by legitimate organizations and may contain malware apps that can steal sensitive information from users. There are three types of mal-ware detection techniques namely attack or invasion detection, misuse detection and anomaly detection which consist of some pros and cons.

III. PROPOSED METHOD

The overall system design of the proposed method is shown in Fig 2. Dataset collection phase includes the extraction of features from the APK files.

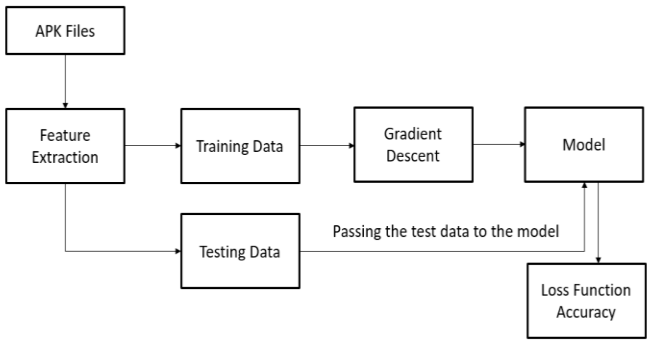

Figure2:Design Methdology

The features are extracted from the APK files and are saved in a csv file format. The dataset is in binary format. If the APK file consists of that particular feature then it is given the value 1, otherwise 0.The type also is obtained in the dataset, if the apk file is a benign file the type is given as 1 and if the apk file is a keylogger file then the type is given as 0. The APK files are taken as input files. Benign files and keylogger files are used for extracting parameters. The extracted parameters are used to form the dataset. The gradient descent algorithm is applied on the dataset. In deep learning we have automatic tuning, in which the machine tunes itself when the new additional values are given to the system. Then the accuracy and the loss function values are obtained from the model using the test data.

Gradient Descent Algorithm

Gradient descent is an algorithm of first order iterative optimization to find the minimum of a function. It is also called as steepest descent algorithm. The first derivative is taken into account when updating the parameters. We update the parameters on each iteration in the opposite direction of the objective function's gradient with respect to the parameters where the gradient gives the direction of the steepest ascent. The learning rate determines the size of the step we take to build the local minimal level for each iteration. Therefore, we follow the path of the downward slope until the local minimum is attained. To update our model's parameters, we use gradient descent. Parameters refer to weights in neural networks. The learning rate is called the size of the steps. A low learning rate is more accurate, but time-consuming gradient calculation.

Gradient Descent algorithm are majorly of three primary types.

The main reason for these variations is computational efficiency.

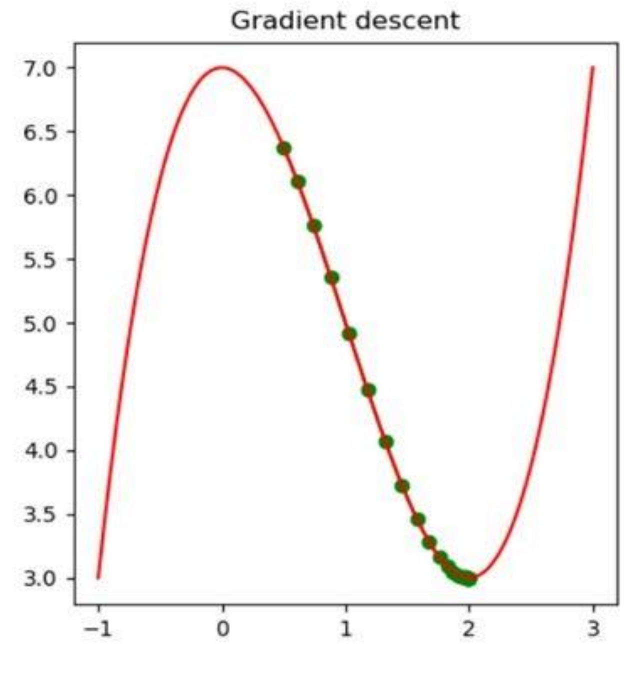

Figure3:Gradient Descent

Batch Gradient Descent

The most straightforward type is the Batch Gradient Descent. It calculates the error in the training set for each example. It updates the model parameters after evaluating all training examples. This process is often called an epoch of training. Batch gradient descent benefits from being computationally efficient and producing a stable gradient of error and a stable convergence. One drawback is that the gradient is stable of error can erratically lead to a convergence position that is not the best that the model can be successful. It also calls for the whole training set to be in memory and provide the algorithm.

Stochastic Gradient Descent

This algorithm updates the parameters for a single example of training according to the gradient of the error. This is contrary to the above algorithm, which, after evaluating all training examples, updates the parameters. This can make Stochastic gradient descent faster depending on the problem than Batch gradient descent. One advantage is that there is a detailed rate of improvement in frequent updates. A drawback is that frequent updates are more expensive than batch gradient descent computationally. The frequency of updates can also lead to noisy curves and can end up causing the failure rate to change dramatically instead of slowly decreasing.

Mini Batch Gradient Descent

This algorithm is a technique that is often preferred as it uses a combination of stochastic gradient descent and batch gradient descent. It simply breaks down the training set into small lots and changes each of these lots. Therefore, it creates a balance between Batch Gradient Descent's efficiency and Stochastic Gradient Descent's robustness. Common numbers of examples ranging from 30 to 500 per batch. But there is no well-defined rule, as with any other machine learning technique, because the optimal number can vary with different issues. Mini batch gradient descent is commonly used for deep learning problems.

IV. RESULT ANALYSIS

By applying stochastic gradient descent algorithm the keylogger is detected from the dataset which contains the features of the APK files. The loss function and the accuracy values are calculated.

Dataset Collection:

The dataset is obtained from the benign and keylogger files. The features of apk files and the previous dataset columns are used to form the new dataset. The dataset contain 31 samples containing 331 attribute fields in it. The apk files are taken and the features are extracted.

Gradient Descent:

The features obtained are sent to a csv file and the type label is given to the files. The apk files taken are of two types, benign and keylogger apk files. By extracting these features, the dataset is formed. All the features obtained from the benign and keylogger apk files are taken in binary format and are placed in the csv file. The type label is also obtained in the dataset. If the file has a particular permission, the

*Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication*

value is given as 0, otherwise the value is given as 1.

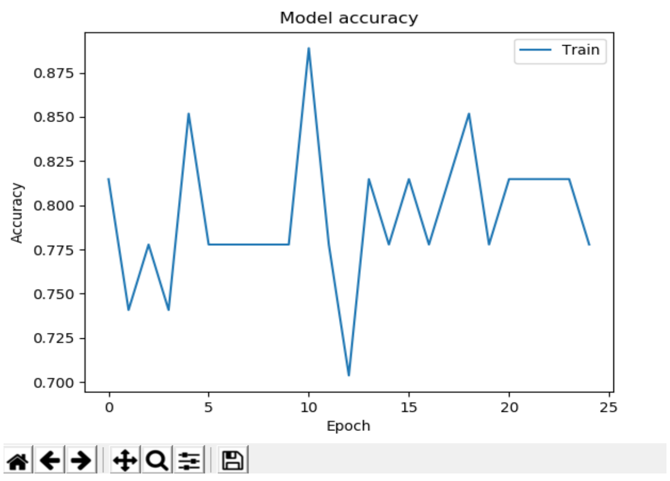

Figure4:Accuracy

The unnecessary features are removed using the dropout layer. For the remaining high weighted necessary features, an epoch value (number of iterations) is given. So, according to the given epoch value the number of iterations occur. The accuracy is obtained after the training and testing are done and the unnecessary attributes are removed using the dropout layer. The network model and the weights are saved to the files. The loss function and accuracy values are obtained after the iterations.

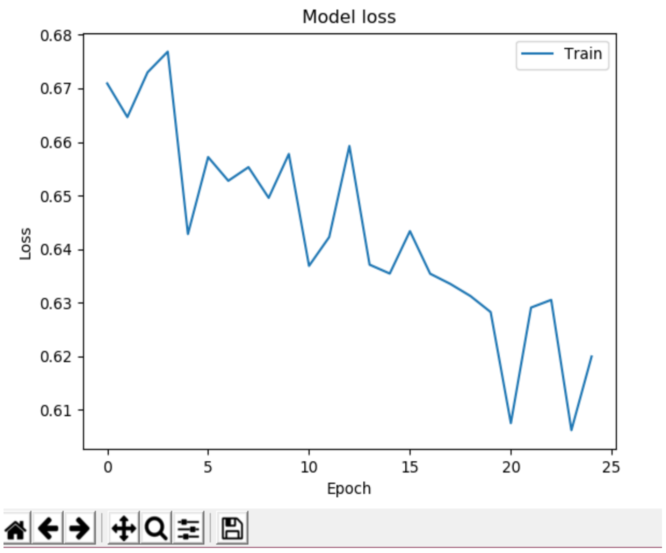

Figure5:Loss Function

V. CONCLUSION

The APK files are taken which are keylogger and benign files. The features are extracted from those files and a dataset is formed on which the algorithm is applied. The dataset obtained is in binary Format. In the dataset, the files having that particular feature are given the value 1, otherwise 0.We also obtain a type label which is used to know whether the files a keylogger file or a benign file. If the file has value 0 for type then it is keylogger file, otherwise it is a benign file. The stochastic gradient descent algorithm is used on the dataset and the unnecessary features are removed using the dropout layer. The accuracy is calculated and also the graphs are generated for loss function values and the accuracy values.

ACKNOWLEDGMENT

First and foremost, I sincerely salute our esteemed institution **V.R SIDDHARTHA ENGINEERING COLLEGE** for giving me this opportunity for fulfilling my project. I am grateful to our principal **Dr. A.V.RATNA PRASAD**, for his encouragement and support all through the way of my project.

On the submission of this Project report, I would like to extend my honour to **Dr. M. Suneetha**, Professor and Head of the Department, IT for her constant motivation and support during the course of my work.

I feel glad to express my deep sense of gratefulness to my project guide **Dr. G. Rama Koteswarao, Professor** for **his** guidance and assistance in completing this project successfully.

I would also like to convey my sincere indebtedness to all faculty members, including supporting staff of the Department, friends and family members who bestowed their great effort and guidance at appropriate times without which it would have been very difficult on my part to finish the project work.

REFERENCES

1. Stefano Ortolani, Cristiano Giuffrida, Bruno Crispo, "Unprivileged Black-Box Detection of User-Space Keyloggers",IEEE,2013
2. Zarni Aung, Win Za, "Permission-Based Android Malware Detection", International Journal of Scientific&Technology Research Vol 2,Issue 3,2013.
3. Shifu Hou, Lifei Chen, Yanfang Ye, "Deep4Maldroid: A Deep Learning Framework for Android Malware Detection Based on Linux Kernel System Call Graphs", IEEE,2016.
4. Irfan BULUT, A.Gokhan YAVUZ, "Mobile Malware Detection Using Deep Neural Network", IEEE, 2017.
5. Hugo Gascon, Fabian Yamaguchi, Daniel Arp, "Structural
6. Detection of Android Malware using Embedded Call Graphs",2013
7. Types of Keyloggers,"https://www.slideshare.net/AnkitAMG/keyloggers-and-spywares"
8. S. Shang, N. Zheng,J. Xu,M. Xu, and H. Zhang," Detecting malware variants via function-call graph similarity". In Proc. of the International Conference on Malicious and Unwanted Software (MALWARE), 2010.
9. S. Gunalakshmii, P. Ezhumalai, "Mobile Keylogger Detection using Machine learning technique",IEEE, 2014.
10. Suleiman Y.Yerima, Sakir Sezer, Gavin McWilliams Igor Muttik, "A New Android Malware Detection Approach Using Bayesian Classification",IEEE,2013
11. Y. Zhou and X. Jiang, "Dissecting android malware: Characterization and evolution" In Proc. of IEEE Symposium on Security and Privacy, pages 95–109, 2012.
12. B. Anderson, D. Quist, J. Neil, C. Storlie, and T. Lane, "Graph-based malware detection using dynamic analysis. Journal in Computer Virology, 2011.

AUTHORS PROFILE

Madhuri Chanumolu, B. Tech, Information Technology, Velagapudi Ramakrishna Siddhartha Engineering College, Kanuru, Vijayawada – 520007, Andhra Pradesh, India.

Rama Siritha Kantamaneni, B. Tech, Information Technology, Velagapudi Ramakrishna Siddhartha Engineering College, Kanuru, Vijayawada – 520007, Andhra Pradesh, India.

Sai Ram Sangepu, B. Tech, Information Technology, Velagapudi Ramakrishna Siddhartha Engineering College, Kanuru, Vijayawada – 520007, Andhra Pradesh, India.

Dr. Rama Koteswararao .G, M. Tech,
Professor, Information Technology, Velagapudi Ramakrishna Siddhartha Engineering College, Kanuru, Vijayawada – 520007, Andhra Pradesh, India.