International Journal of Computer Applications Technology and Research
Volume 12–Issue 12, 17 - 22, 2023, ISSN:-2319–8656
DOI:10.7753/IJCATR1212.1004

Building Comprehensive Dataset: Android File and Unstructured Data Collection from APKs for Enhanced Vulnerability Detection

Jigna Rathod
Assistant Professor
Babu Madhav Institute of Information Technology
UTU
Bardoli, India

Dr. Dharmendra Bhatti
Director and Professor
Asha M. Tarsadia Institute of Computer Science
Engineering, UTU
Bardoli, India

Abstract: Google's Android platform grew to become one of the world's largest mobile systems. An Android application can use any functionality that the platform provides. Malware may also be present in Android apps. That is why our objective in this research is to collect files from wide malware families and use them to help us and other researchers run vulnerability detection tests. From the most well-known Android malware projects, we gathered both benign and malicious files. This process yielded 20832 malicious files and 10856 benign files. In this study, we also disclosed the Android file collection step as well as our vulnerability identification technique. Machine learning approaches are frequently used to identify whether an APK file is tainted, with the goal of detecting malicious apps. We have been concentrating on machine learning approaches to discover the unknown vulnerability. To identify malware, the malware researcher must create his or her own dataset. As part of our dataset production process, we collect Android files, do dynamic analysis, and then extract their characteristics. We described the technique of producing data sets in this paper. The android file contains a lot of unformatted data in the form of text or XML files, which is difficult to analyze and store. Our objective in this context is to provide a dynamic analysis of these obtained Android files, allowing you to access the underlying information, such as system calls, network traffic, and permission requests made by specific apps. Using Dynamic Analysis, we are attempting to manage massive amounts of data and assure correct processing in the context of Android files. In this study, we offer MalwareDefender, a dynamic analytical tool that handles the challenging issues of evaluating and processing massive volumes of data.

Keywords: Android; Smartphones; Malware; Vulnerability Detection; APK Files, Dataset Generation, File Collection

1. INTRODUCTION

Nowadays, the process of detecting Android Vulnerability is based on a variety of methods, such as signatures, patterns, resources, and components that were statically analyzed to find known vulnerabilities [1][16], whereas machine learning techniques are capable of discovering previously unknown or recently discovered vulnerabilities [2]. Machine learning methods are increasingly being used to find vulnerabilities [1]. Machine learning approaches are proven to be less time demanding and resource costly than nonmachine learning methods [2].

An APK file can be detected as malicious by a researcher. To that purpose, a researcher must develop his or her standard set of methods. In this situation, a researcher would like to have access to the same dataset. The construction of a dataset is an important aspect of the procedure since it provides a list of features that will assist you in determining whether this APK file is malicious or not. It is also critical to choose the properties of the dataset since various malicious files exploit these aspects. This assists researchers in identifying a malicious file using exploited characteristics. Our dataset generation procedure consists of Android file collecting, dynamic analysis, and data extraction. During the dynamic analysis phase, we created MalwareDefender, an analytical tool that would examine APK files and collect raw data like as network traffic, system calls, and permission in real time. The chosen characteristics are derived from the unstructured data received during the data extraction phase via dynamic analysis.

As part of an Android File Collection phase, we collected 20832 malicious files from the most well-known Android malware projects, as well as 10856 benign files from the Google Play Store and other sites. In this article, we discussed android file collections, the dynamic analysis of files, and collecting unstructured data from APK files. All characteristics are extracted from files, and the data extraction step will result in the creation of our final dataset. The dataset will be utilized to train and test the models throughout the Machine Learning phase in order to research and deliver improved outcomes. Other articles will go over the various stages of feature extraction and machine learning.

Big Data comprises huge volumes of uncategorized data that can be stored in a number of formats such as text files, photos, audio, video, and so on [3]. Unstructured data has no explicit structure; instead, it consists of its internal structure [4]. The Android files are made up of text and XML files, each with its unique structure. There are 20832 malicious files and 10856 benign files in the Android file gathering phase. As a result, keeping and storing a huge amount of Android files is a big data problem, and executing dynamic analysis is a task in and of itself. In this part, we are attempting to manage a big quantity of data regarding Android files and ensure that they are appropriately handled by using dynamic analysis.

There is no dynamic analytic tool available to do this study. To that end, we've developed MalwareDefender, a sophisticated analysis tool that will be used by each application to perform tasks such as running it, keeping track of its current operation, and collecting raw data such as

system calls generated by an application, network traffic captured by applications, and permission requests from each functioning application. We discussed how our dynamic analysis tool works in our prior publication [5].

This document includes the sections listed below. Section 2 of our Generalized Detection System describes the full vulnerability detection procedure. Section 3 describes the android file gathering step. Section 4 describes the many difficulties we experienced while collecting data. Section 5 describes the limitations of the file gathering procedure. Section 6 describes the architecture flow of the dynamic analysis tool. Section 7 contains the paper's conclusion.

2. Overall Process of Vulnerability Detection

The overall process for detecting vulnerabilities is explained in this section. The architecture of vulnerability detection for android is explained in Figure 1. There are four main phases of this process.

- Android File Collection
- Monitoring Component (Dynamic Analysis)
- Data Extraction and Generation
- Machine Learning

The first and second phases of this paper are the collection of android files and the monitoring component. The next set of papers will deal with further phases. All android files are collected during the Android File Collection phase. Many popular android malware projects such as Drebin [6], AndroZoo [7], AndroPRAGuard [8] and MalDroid2020 [10] are used to collect these malicious files. Collection of benign files are available from the Canadian University of Cybersecurity [9] and Google Play, as well as ApkPure.com [11].

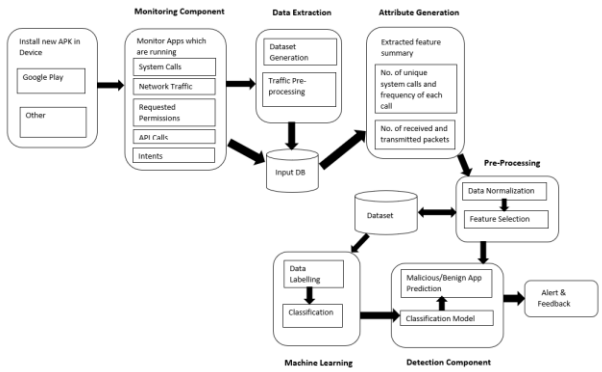

Figure 1 Proposed Architecture of Android Vulnerability Detection

The APK file format is an Android Package file type that is used to launch programs on Android-powered devices. The APK file includes the Java code required to run the program, as well as AndroidManifest.xml, resources, assets, and other files. Java code is compiled into bytecodes, which are subsequently stored as Dalvik executable files (.dex files). The Android operating system uses the Dalvik Virtual Machine (DVM) to run .dex files.

For the Monitoring component, we created an Android app named MalwareDefender. MalwareDefender is an Android software that monitors and records the state and performance of the system. It is written in Java and uses the Monkey Runner tool to perform 5000 random events each app launch. The Strace and tcpdump utilities have been introduced to gather system call logs and monitor network traffic. We obtained a total of 10066 sets of original data by doing dynamic analysis on all active apps. A component responsible for attribute creation will gather data from permissions, system calls, and network traffic. Selected characteristics are used as input in the data extraction and creation process. A dataset of chosen characteristics is built from the unstructured data acquired through dynamic analysis. The final dataset includes key aspects such as permissions, system calls, and network traffic. This dataset is utilized in the machine learning phase to categorize malware and benign programs, where different supervised algorithms, feature reduction and extraction approaches, and ensembling techniques are used. We will train a dataset using machine learning, and then apply all algorithms to the tests based on the taught data to produce more precise findings for vulnerability detection. This will help us understand the vulnerability detection system better.

3. Android Files Collection

The overall collection process for the android files is explained in this section. The first step in preparation of the dataset is to collect android files, which is explained here.

3.1 File Collection Procedure

This section describes the architectural flow of the Android file gathering process. The user opens the Android File collection module, inputs a URL, and requests a malicious or benign file download. This module connects to a website via a specified URL, and if the connection is successful, the website connects to the server and sends the user-supplied file request for download. The server responds with the website's file path, and lastly, the malicious or benign file is downloaded and saved to a physical place. Malware or benign files are selected based on global Android Malware databases and website data. In our article, we evaluated and detailed some well-known datasets of Android malware as well as a few benign datasets. Other researchers quote, use, and refer to the majority of those data sets. Furthermore, because these databases include a wide spectrum of malware families, our study is extremely dependable, implying that the results we collected are as well. The Android malware dataset may be acquired from the Drebin dataset, which is a well-known initiative [6]. The Androzoo dataset has a vast number of applications that are constantly growing [7] [12]. AndroPRAGuard's database contains 10479 applications from various families, as well as examples from the MalGenome and ContagioMinidump datasets [8]. The old samples may be found in Drebin, AndroZoo, and AndroPRAguard. So, for recent samples, we gathered samples from the MalDroid2020 [10] dataset. The preceding procedure resulted in the collection of roughly 20832 Malware programs. Drebin has been used to download about 5490 programs [6]. AndroZoo has around 4000 programs downloaded [7]. AndroPRAGuard has been used to download 5953 apps [8]. From MalDroid2020 [10] around 5389 applications are downloaded. In view of the variety of malware families included in this dataset, as well as a number of recently discovered malicious files, we chose these popular Android Malware Datasets to run our process.

Benign applications were acquired from databases, the Google Play Store, and other places. We collected around 6500 benign applications from the Canadian University of

Cyber Security dataset [9]. The Google Play store has received 356 app downloads. We gathered around 4000 benign applications from websites [11]. We choose the official Google Play Store, various well-known datasets, and well-known internet domains for benign file download. Table 1 summarizes the total number of Android applications downloaded from each market place.

Table 1 Android Files Summary

Popular Datasets	No. of files download	File Type	Total Files
Drebin	5490	Malicious	20832
Androzoo	4000		
AndroPRAGuard	5953		
MalDroid2020	5389		
Google Play Store	356	Benign	10856
Canadian University of Cyber Security	6500		
Websites	4000		

4. Challenges in Data Collection

During the collection of APK files from various android marketplaces, we have encountered a number of challenges and obstacles. Some general issues include the following:

1. **Request to grant access to the Datasets:** The data for Android Malware and Benign APK are easily accessible, however they cannot be downloaded directly. To be authorized for this reason, we must request a dataset by sending an email, and researchers of the datasets will enable us to obtain them in exchange for API keys, passwords, or downloading URLs. Requests are used to retrieve datasets from Drebin [6], AndroZoo [7], AndroPRAGuard [8], MalDroid2020 [10], and The Canadian University of Cyber Security [9]. These datasets, which include many different types of malwares and a large number of files, are being used by researchers all around the world in their study.
2. **Manual Download of Files:** When we obtained the researcher's API key to download the files, we were not permitted to download the bulk files at the same time in Androzoo [7] [12]. Androzoo [7] gives an excel file with all of the infected files' details, including SHA256, Package Name, File Size, Date, and Time. The SHA256 value is unique to each file. Androzoo [7] allows you the freedom to download any malicious file you choose. You may also download new malicious files as well as from the past. A collection of malware files can be created based on one's preferences and needs. To download each file manually, we have followed the following steps.

    1. File Download link:

       https://androzoo.uni.lu/api/download?apikey=${APIKEY}&sha256=${SHA256}

    2. Use the API key acquired from AndroZoo [7][12].

    3. Select the SHA256 value from the Excel file and replace it in the link.

    4. Paste the entire URL into the browser.

       For instance, https://androzoo.uni.lu/api/download?apikey=f60e30400dbecb3eda905972548fb8b971ef07710bf96b8d358567d01a741ce8&sha256=0000003B455A6C7AF837EF90F2EAFFD856E3B5CF49F5E271914

    5. The API Key will remain the same, but the SHA256 value for each APK file will be different. So, for each APK, change the SHA256 value and copy-paste the URL to download it.

5. Limitations of File Collection Process

The total amount of Malware Files acquired by this method imposes some limitations also. They are as follows

1. It is not possible for a researcher to identify a Malware Family from a single Malware file. We do not disclose any more information on the Malware Files.
2. The Malware Files we have collected span the years 2011 to 2021.
3. We have only gathered files that are openly available.

6. Dynamic Analysis of Android Files

The different challenges to unstructured data extraction and the dynamic analysis phase are discussed in this section.

6.1 Challenges of Unstructured Data Mining

In the realm of big data, there is a massive volume of unstructured and heterogeneous data to deal with [13-15]. This is because unstructured data does not have specified schema or models and has its own internal structure, rendering traditional RDBMS unsuitable for storing it [3][4]. Big data is concerned with three Vs: 1) Volume, 2) Velocity, and 3) Variety [13][15].

- Volume: This is a large number of datasets, which are complex in their data structure. The challenge with volume is to handle the complexity of the data structure [13][15].
- Velocity: The need to handle the pace with which new datasets are created or old datasets are updated is referred to as velocity. This aspect pertains to data generated by machines, such as mobile device sensors. The issue with velocity is coping with the streaming system's limited capacity while obtaining valuable information from continual fresh dataset production [13] [15].
- Variety: Datasets come from a range of sources and can be in a variety of formats, including text, audio, video, graphs, sensors, and so on. [13][15]. The variety of data provides more information for

problem solutions. The issue with diversity is merging numerous systems to accept different sorts of data [15].

The framework for large data analysis and mining is depicted in Figure 2. Tier 1 data mining procedures for large datasets necessitate the use of expensive computer units and clusters for data processing and comparison [13]. To achieve high speed, big data processors rely on cluster computers, and data mining activities are done using parallel programming tools such as Map-Reduce [13]. Tier 2 focuses on data security and subject expertise. Data privacy may be achieved by restricting data access such that sensitive data is only available to particular persons. The second way to preserve data privacy is to anonymize data fields so that sensitive information is not connected with a specific record [13]. Tier 3 is focused with algorithms for massive data mining. Deep analysis, which is quite difficult, is necessary for the creation of the algorithms [15]. Mining algorithms are self-contained and work decentralized [13]. Big data mining necessitates the use of machine learning and data mining technologies that need a large amount of computational power and resources [13]. As a result, in big data analysis and mining, each layer covers all of the complicated issues.

Tier 1: Data Accessing and Computing
Tier 2: Data Privacy and Domain Knowledge
Tier 3: Big Data Mining Algorithms

Figure 2 The Framework for Big Data Analysis and Mining

To solve all of the challenges connected with unstructured data analysis and mining, we proposed a dynamic analysis tool (MalwareDefender) that does dynamic file analysis, creates unstructured data in.xls,.csv, and .pcap files, and saves the files on the device. These data are then sent to a local system with enough capacity for further processing. It solves all of the issues involved with dealing with data volume, velocity, and variety.

6.2 Dynamic Analysis Phase

MalwareDefender in Java is a program that does a dynamic examination of all APKs that are now executing. It looks for features like permission, system calls, and network traffic in the present program. These attributes are saved in a variety of file formats, including .xls,.csv, and .pcap, and Wireshark is then used to analyze the .pcap files to extract additional network traffic information. This allows for the effective extraction of certain features from Android apps.

Figure 3 depicts the architectural flow of dynamic analysis tool and its operation, which entails the sequential implementation of the following steps:

- The release of MalwareDefender, an Android application built by us, will signal the beginning of the data extraction approach based on dynamic analysis.
- When MalwareDefender is launched, it will provide a complete list of all programs presently installed on the device.
- Run the required application to extract features from the one that is already executing.
- If the app is correctly launched, Monkey Runner will generate 5000 random actions; else, a list of installed applications will be displayed for the user to pick and run a new one.
- During a 20-minute random event produced by Monkey Runner, MalwareDefender will monitor and collect system calls, app permissions, and network activity.
- The app should be terminated and deleted from the background after 20 minutes.
- After exiting the current app, MalwareDefender saves permission, system calls, and network traffic data to the device in.xls, .cls, and .pcap formats, respectively.
- If you have access to all three files on your smartphone, save them to your local system for additional investigation. If this is not the case, restart the application to extract the data.
- The data extraction app will go through each application and extract all of its attributes.

Once all this unstructured data is gathered, data extraction and mining will be carried out. The data extraction and mining phase will be discussed in another paper.

7. CONCLUSION

To determine if an APK file is malware or not, a researcher needs to first create a dataset. We discussed the dataset production process, which included Android File Collection, Monitoring Component (Dynamic Analysis), Data Extraction and generation, and the Machine Learning Phase. We investigated the Android File collection phase in this paper and presented a dynamic analysis tool (MalwareDefender) that we designed to do dynamic analysis of running applications. During the Android File Collection phase, we collected 20832 malicious files and 10856 benign ones. We also discussed some of the issues faced during data collection as well as some of the data collection limits.

We finished the collecting of Malware and Benign files for dataset construction during the Android File gathering Phase. We also reviewed the architectural flow of our recommended dynamic analysis tool and illustrated the whole Android File Collection procedure. Mining

characteristics from APK files and generating the dataset needs dynamic file analysis. We performed a dynamic analysis on all 10066 APK files, implying that our final dataset would contain 10066 records that machine learning algorithms will use to train the model. The next papers will go into the stages of data extraction and generation, as well as machine learning.

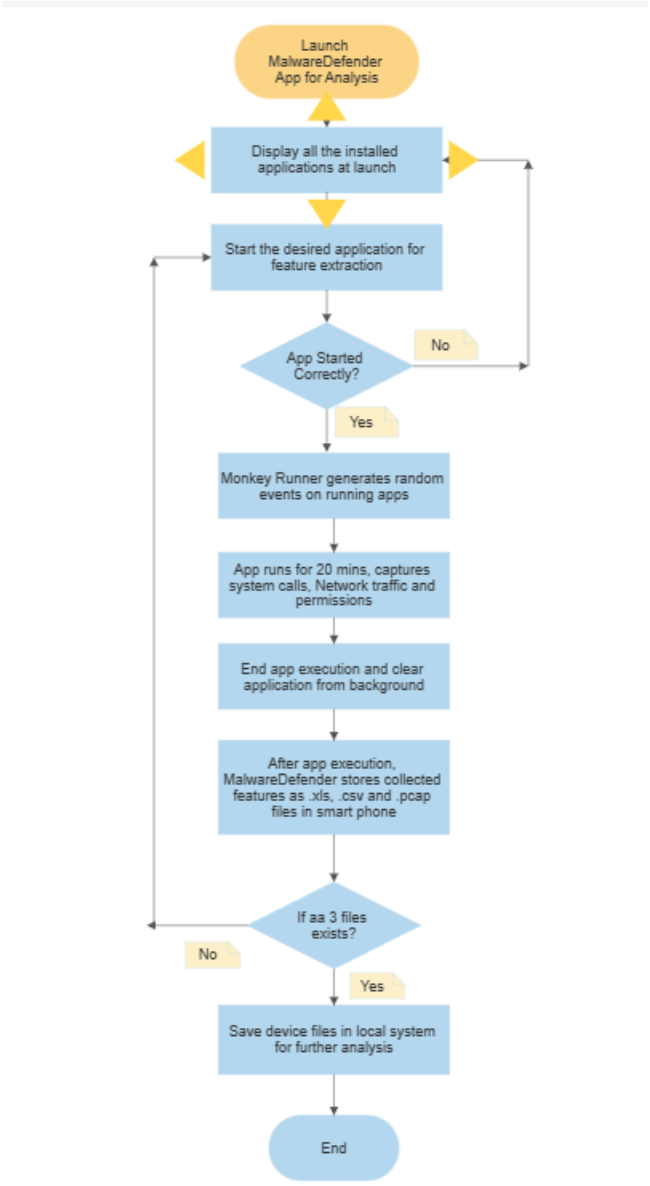

Figure 3 MalwareDefender – Architectural Flow of Dynamic Analysis Tool

8. ACKNOWLEDGMENTS

We would like to acknowledge our two students Mr. Bhargav Patel and Mr. Parth Desai of BMIIT, UKA Tarsadia University who helped us in manually downloading the files from AndroZoo [7] and performing dynamic analysis of each APK file using our implemented tool. They downloaded around 4000 files from AndroZoo [7] and perform dynamic analysis of around 1000 APKs dedicating their time for this task.

9. REFERENCES

[1] Agrawal, P., & Trivedi, B. (2019, February). A survey on android malware and their detection techniques. In 2019 IEEE International conference on electrical, computer and communication technologies (ICECCT) (pp. 1-6). IEEE.

[2] Alqahtani, E. J., Zagrouba, R., & Almuhaideb, A. (2019, June). A survey on android malware detection techniques using machine learning algorithms. In 2019 Sixth International Conference on Software Defined Systems (SDS) (pp. 110-117). IEEE.

[3] Kanimozhi, K. V., & Venkatesan, M. (2015). Unstructured data analysis-a survey. International Journal of Advanced Research in Computer and Communication Engineering, 4(3), 223-225.

[4] Chen, M., Mao, S., & Liu, Y. (2014). Big data: A survey. Mobile networks and applications, 19, 171-209.

[5] Rathod, J., & Bhatti, D. (2022, March). Vulnerability detection in the android application based on dynamic analysis. In Proceedings of Third International Conference on Intelligent Computing, Information and Control Systems: ICICCS 2021 (pp. 287-302). Singapore: Springer Nature Singapore.

[6] Arp, D., Spreitzenbarth, M., Hubner, M., Gascon, H., Rieck, K., & Siemens, C. E. R. T. (2014, February). Drebin: Effective and explainable detection of android malware in your pocket. In Ndss (Vol. 14, pp. 23-26). Brown, L. D., Hua, H., and Gao, C. 2003. A widget framework for augmented interaction in SCAPE.

[7] University of Luxembourg. (n.d.). Androzoo home. Retrieved April 9, 2019, from https://androzoo.uni.lu/

[8] Android PRAGuard Dataset | PRA Lab. (n.d.). Retrieved May 19, 2019, from http://pralab.diee.unica.it/en/AndroidPRAGuardDataset

[9] CICMalAnal 2020 | Datasets | Research | Canadian Institute for Cybersecurity | UNB. (n.d.). Retrieved July 15, 2020, from MalDroid 2020 | Datasets | Research | Canadian Institute for Cybersecurity | UNB

[10] Retrieved July 15, 2020, from MalDroid 2020 | Datasets | Research | Canadian Institute for Cybersecurity | UNB

[11] Android Files Download. (n.d.). Retrieved Feb 19, 2019, from https://apkpure.com

[12] Allix, K., Bissyandé, T. F., Klein, J., & Le Traon, Y. (2016, May). Androzoo: Collecting millions of android apps for the research community. In Proceedings of the 13th international conference on mining software repositories (pp. 468-471).

[13] Wu, X., Zhu, X., Wu, G. Q., & Ding, W. (2013). Data mining with big data. IEEE transactions on knowledge and data engineering, 26(1), 97-107.

[14] Chen, J., Chen, Y., Du, X., Li, C., Lu, J., Zhao, S., & Zhou, X. (2013). Big data challenge: a data management perspective. Frontiers of computer Science, 7, 157-164.

[15] Fan, W., & Bifet, A. (2013). Mining big data: current status, and forecast to the future. ACM SIGKDD explorations newsletter, 14(2), 1-5.

[16] Solanky, M. J., & Bhatti, D. (2019). A review on several vulnerabilities' detection techniques in Android Mobile.