A Systematic Review of Android Malware Detection Techniques

Faris Auid Alharbi *faris-7000@hotmail.com*
Faculty of Computer Science and Engineering/Cybersecurity
University of Jeddah, Jeddah, 23468
P.O.Box: 4053, Kingdom of Saudi Arabia

Abdurhman Mansour Alghamdi *Ghamdiaman@gmail.com*
Faculty of Computer Science and Engineering/Cybersecurity
University of Jeddah, Jeddah, 21959
P.O.Box: 34, Kingdom of Saudi Arabia

Ahmed S. Alghamdi *ahmedg@uj.edu.sa*
College of Computer science and Engineering/Cybersecurity
University of Jeddah, Jeddah, 23465
Kingdom of Saudi Arabia

Abstract

Malware detection is a significant key to Android application security. Malwares threat to Android users is increasing day by day. End users need security because they use mobile device to communicate information. Therefore, developing malware detection and control technology should be a priority. This research has extensively explored various state of the art techniques and mechanisms to detect malwares in Android applications by systematic literature review. It categorized the current researches into static, dynamic and hybrid approaches. This research work identifies the limitation and strength current research work. According to the restrictions of current malware detection technologies, it can conclude that detection technologies that use statistical analysis consume more time, energy and resources as compare to machine learning techniques. The results obtained from this research work reinforce the assertion that detection approaches designed for Android malware do not produce 100% efficient detection accuracy.

Keywords: Malware Detection, Android, Static, Dynamic and Hybrid Detection.

1. INTRODUCTION

Nowadays, android is one of the top widespread Operating System (OS) in the world of mobile telephony with largest users in different parts of the world. Vast amount of financial applications such as mobile/Internet banking and online purchase/sell of products runs on this most popular mobile OS. Furthermore, sensitive information like health records, username and passwords are stored on android phones. Currently, mobile technology is being used widely [2]. The usage of mobile technology has been increasing rapidly since 2008 [3]. Everyone can simply store the private, and sensitive information in the mobile such as banking credentials and personal data like photos and videos etc. [4]. Statista provides statistic data about the total number of smart mobiles devices that are sold to end users around the world from 2007 to 2020 is shown in Fig.1. It shows that number of smartphone users are increasing every year. It also shows the in 2018 up to 1.56 billion smartphones were sold. In addition, 88% of smartphones that were sold in the first quarter of 2019 to the end users were smartphones with Android operating system [5].

Mobiles devices are available with various operating systems. The most famous operating system is Android. Android is open-source system software for smartphones and tablets. As stated by Google, 1.3 million android devices are being activated every single day [3]. According to

Gartner's report [6] Google's Android got a total of 82% of the market in 2016. There were 432 million smartphones sold out in the quarter of 2016 in which 352 million smartphones were Android based [7]. With the world's largest mobile phone OS, Android poses a greater risk of vulnerability and malware attacks. According to Google Android Security Report, 655 vulnerabilities were discovered in 2016 [8]. In 2017, 316 vulnerabilities were discovered in the Android OS, which is highest among all the mobile OS [9]. According to Cisco's report [2], 98% of the malware attacks were for Android OS. For that reason, many malware detection tools are being developed and gaining popularity for Android OS.

The technological progression of Android has created proportionate attraction by malware writers. Malware writers are advancing daily to gain financial benefits by creating malware applications. Such applications can directly break into Android OS security. As a result, the victims' personal data and financial credentials are compromised. Malware attack on android becomes critical issue. According to Symantec report [1], the mobile OS is also discovered as being ubiquitous.

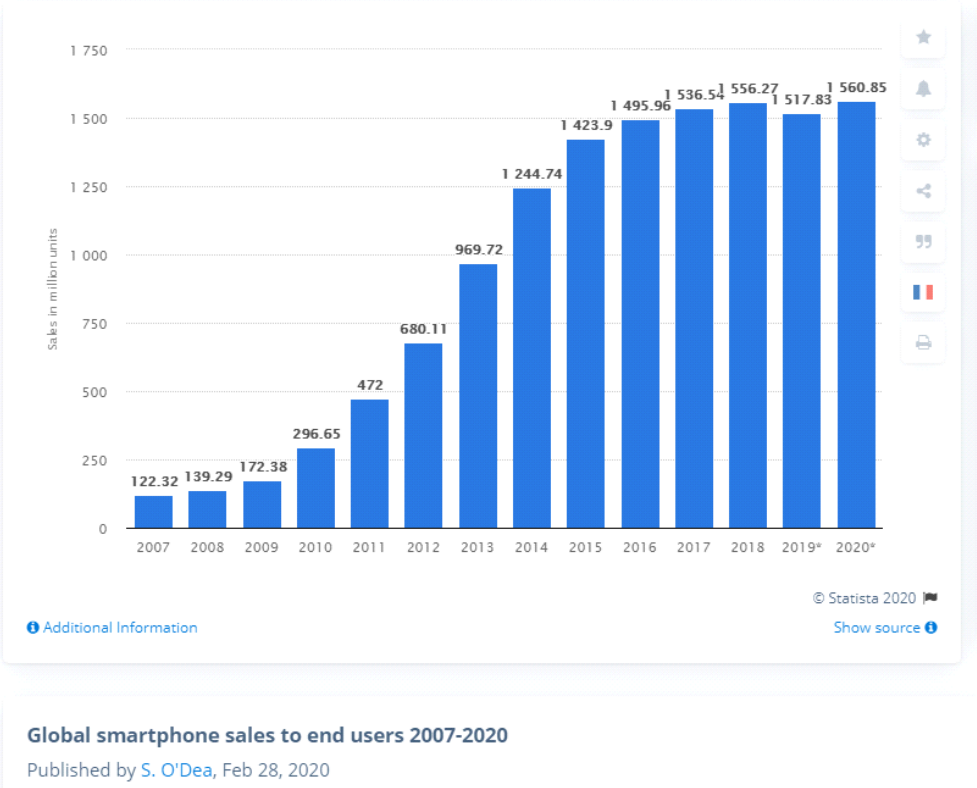

FIGURE 1: Number of smart mobiles sold to end users worldwide between 2007and 2020[5].

Malware threats are expected to increase as smartphone operations expand [10]. There are many malicious applications that include malware, which compromises the security of Android OS. These implementations include categories of malware such as Trojans, phishing applications, spyware etc. [8]. There are many malware detection technologies for Android OS.

Malware is a software that does many operations without the user's knowledge and permission [11]. The main goal of the malware is to steal the secret and sensitive data from the smartphones, sending SMS/MMS, locking the devices, doing calls to the specific numbers, and sharing data through GPS [2, 4]. Moreover, many research studies are conducted to distinguish different existing Android malware. Android Malware Genome was one of these projects that are conducted to characterize the existing Android malware [12]. Another study called Android Drebin was conducted to make a comparison of various malware detection methods. However, many data samples are available for Android Malware [13, 14]. According to the functionalities, Android

Malware can be classified into various categories, such as: Trojans, Virus, Rootkits, Phishing Apps, Spyware, Bot Process etc. [15].

In this paper, we have reviewed the state-of-the-art literature related to malware detection in Android OS. The research work from literature review is categorized into three categories (a) static (b) dynamic and (c) hybrid malware detection in Andriod OS. We have selected 2327 articles and then shorted listed 18 based on criteria mentioned in section 3. Lastly a detail comparison of the selected research work is performed.

This paper is organized as follows. Section 2 provide the literature review. Section 3 outline the methodology of this research work. Section 4 performs the comparative study of AMD techniques. Section 5 gives a fruitful discuss on comparative study and section 6 conclude the paper.

2. LITERATURE REVIEW

Mobile security research has become a growing concern every day. Similarly, research related to mobile technology from design, vulnerability, threats and detection methods is getting hot nowadays. Many industries spend billions in security of mobile OS particularly Android OS. There are numerous standard malware detection techniques proposed for Android in literature. A number of those leverage textual data from the application's description to study what an application can do. As an instance, tests this system to peer if the application behaves as advertised [17]. This section provides a review of malware detection techniques for Android OS. These techniques are categorized into static, dynamic and hybrid detection techniques.

2.1 Static Detection Techniques

The static malware detection technique may or may not execute malicious code. It depends only on malware compression. To detect malware using this technique, reliable detection features are extracted from or through the byte code of the application manifest file. Unlike a dynamic system that focuses on system calls and application pads. The Android apps are in Android Package (APK) format. This is usually called a postal package. All Android files, folders, and other resources are included in it. To find objects features, reverse engineering is often used for APKs. When looking for retrieving related features, the "AndroidManifest.xml" manifest file must first be considered.

In [18], researchers used Natural Language Processing (NLP) methods to analyze the market description of the application. It uses semantic authorization model to determine the reasons why the android applications use authorization. Meanwhile, another study [19] offers hidden malware detection in Android applications, analyzing inconsistencies between program behavior and the user interface. All these methods are based on textual information, declarations in the obvious file, or specific API calls, while their approach focuses on analyzing application behavior based on application code related to device-sensitive data.

To protect users from high levels of damage caused by Android malware, the researchers in [45] proposed model which is the most efficient and effective way to integrate access, sensitive APIs, monitor application events, and modify permission levels into a critical function (RF). This model is proposed to determine if an Android app is dangerous or not. In particular, a database of 2130 samples are used to test the effectiveness of the proposed method. Experimental results show that the proposed method achieves a high precision of 88.26%, a sensitivity of 88.40% and an accuracy of 88.16%. The test results also shows that the proposed model is very promising and can provide an inexpensive alternative for detecting Android applications.

In [66], researchers found that a permutation-based static function was informative than a program-based dynamic function. A strong analysis of this work shows that a small reduction in the number of authorizations can be of great benefit. Although the dynamic system control function is not very strong but it is powerful enough to serve a good purpose in combination with

other functions.

The researchers in [70] implement a new AMD program using the Deep Neural Network. Malware classification was performed based on a static analysis of the integrated system opcode sequence. Malware detection features are automatically detected in the order of network opcodes. In [71], researcher have developed a semantic decorative model of Android. They also developed a database called SMART, which automatically reads models from malware and uses an integrated machine learning and DSA deployment mechanism to detect and classify malware. Both studies [70, 71] show that it can reach an accuracy rate of 87%.

Early in 2016, [73] performed an offline mode in-depth analysis of network traffic logs and proposed a method called CREDROID. This method identifies malicious applications based on Domain Name Server (DNS) queries and data sent to remote servers. Instead of performing signature-based scans that cannot detect polymorphic malware, it does pattern-based detection. This study also focuses on the leakage of sensitive information sent to a remote server. They observed that 63% of malware has been the focus of research by researchers of applications in standard data sets.

The research study in [74] has used Machine learning where small malware files are extracted using the API call diagram, the detection accuracy achieved is 96.12% for nearest neighbor based, 98.7% for Random forest based. However, it cannot determine how malware processes the data affected and detected in virtual environment.

Finally, a machine learning file for detecting malware on Android devices has been introduced in [75]. In particular, four feature groups are published, including permissions to view Android apps (apps), in-app event tracking, critical APIs, and permission levels. If this happens accidentally in a diagnostic forest, they will learn to see if the usage is strong. The validity of the proposed method is evaluated by 10-fold real-time data checking. The test results show that the proposed method can achieve an overall accuracy of 89.91%.

A novel teachnique for detecting malware in Andriod OS is presented in [80]. It is based on feature engineering using machine learning. A static analysis is performed for matching Application Programming Interface (API) to specific features. This is called feature vector for an application. The experiments are performed on 972 and 1100 malicious and benign android applications. The accuracy achieved is 98.87%. after reducing the feature set by 75.9%, the accuracy achieved is 95.67%.

2.2 Dynamic Detection Techniques

Android-based applications communicate with the OS through system calls, allowing to track what's being exchanged between them. Dynamic detection technique monitors Android malware in a controlled runtime environment. Such techniques reporting malware directories that can model detection signatures. It checks for malware's interactions with mobile resources and services such as location, network, package, operating systems.

Machine learning is popular among the researchers to detect malicious Android applications. Moreover, most options educate the classifier solely on malware samples and can consequently be very high quality to discover different samples of the identical family. Such as, in [20], the approach primarily based on features extraction from malicious applications show up and disassembled code to educate the classifier. The place as Mobile Application Security Triage (MAST) [21] leverages permissions and feature elements are extracted from Android applications as elements to teach the classifier. These coarse elements are awesome mechanisms to filter many applications prior to leveraging methods, which require extra evaluation of the applications [21]. There are many different systems, such as Crowdroid, and DroidAPIMiner, that leverage computer to know strategies to analyze statistical aspects for detecting malware [22, 23].

An overview of dynamic X-ray detection and an approach to analyze how malware can be

detected in the Android Gingerbread version is presented in [30]. DroidScope has explorer and detected malware for Android DroidKungFu and DroidDream. The detection features obtained from the survey revealed during DroidDream and DroidKungfu accomplishes cellular exploit to gain unauthorized access to mobile security by encrypting IMSI and IMEI numbers in an XML string. The results of the study showed that the effectiveness of the technique used was resistant to malware confusion. The limitation observed is that mobile RAM analysis for forensic artifacts is not considered after operating the malware. In addition, basic properties such as basic logic, operating binaries and native libraries have not been analyzed by the technique.

In [35], researchers analyzed malware detection in mobile memory using a memory forensic method. A self-replicating Trojan was detected with 90% accuracy with 20% unclassified samples. This study has found that significant information about malware can be deduced from the analysis of the memory dump of Android mobile device. Hidden codes can easily be exposed to explosion under favorable conditions. So far, it is not possible to investigate the search for malware properties that are important for forensic and security analysis. There is extensive work on detection techniques, but there is no research that identifies and lists the limitations and strengths of these techniques. Identifying both limitations and strengths can help improve the effectiveness of these techniques and improve the detection of malware on Android devices.

In [65], researchers are developing and implementing computer-based methods (ServiceMonitor). This study categorizes applications as benign or malicious by powerful monitoring application performance behavior according to the proposed usage analysis technology and modeling these behaviors in the Markov chain. According to the results of the evaluation, ServiceMonitor was able to accurately and efficiently detect malicious Android software on mobile devices and achieved 96% accuracy when distinguishing between malicious and benign applications.

In [72], researcher used five machine learning classification techniques to evaluate a total of 11,000 samples (in which 6,971 are Android malware samples). Compared to Android's PUMA malware detection technology, which uses only 239 malware applications in the dataset, it achieves a maximum accuracy of 99.7% to detect malware with simple logistics technology.

The researcher in [79] uses various machine learning classifiers to determine which classifier gives higher accuracy. They have concluded that random forest provides higher accuracy as compared to SVM and Naïve Bayes classifier. ProDriod proposed in [80], is an android malware detection technique based on hidden Markov model. This research work uses behavioral based android malware detection technique. They have recompiled dataset to find out malicious activities and then generate the encoding. These encoding patterns are used to generate sequence signature of various malware types. Their proposed framework provides an accuracy of 94.5% for detecting malware in android OS.

2.3 Hybrid Detection Techniques

This technology combines the characteristics of dynamic and static technologies to provide the most consistent detection results in malware analysis. Hybrid malware detection methods mainly uses dynamic and static techniques to perform training and detection. The advantages of both systems are synergistic, providing faster detection speeds than dynamic and static techniques.

There are researchers that developed dynamic and static analyses techniques that help to detect the malware based on the known features. Apposcopy creates unique applications for flow control and data flow analysis [24]. RiskRanker runs many Android applications that are very dangerous for analysis and have been mentioned among lower risks [25]. Sebastian [26] analysis for malware detection of Android applications with dynamic load. These studies [27-29] sign malware and application detection based on signature-based virus detection method.

The study in [31] performed a comparative analysis of static, dynamic, and machine detection techniques used to detect malware in Android applications. Centralizing the market in mobile platform applications has made finding malware more difficult for most detection techniques, even

for machine learning techniques. For example, to test the legitimacy of multiple applications, Google plays the central market for Google Exit [32]. This security monitoring is inadequate as millions of Android developers connected to Google have not been checked before their apps are added to the Google Play Store. This is like the phone store and Windows or Apple's App Store. It does not guarantee 100% complete security on mobile devices, especially Android.

Emotions that are evaluated are mainly based on misuse and conflicts of use, unreliable data and the status of the system as key objectives of the analysis. Based on the results obtained, no technique provides a 100% detection rate of mobile malware. Anusha [33] uses a mobile API to detect malware and to change the behavior of malicious and mobile applications. They developed detection system that detects obscure mobile malware that is not detected by antivirus software. The detection approach uses FSAs to sample malware and test the method. Using runtime assays, they detected viruses before and after sample packing. Six antivirus software used was full of UBX and no one could find these mobile viruses. Most antivirus programs do not respond to malware detection because they are signature-based. This result is similar to [34], which is based on behavioral analysis of malware detection in Android applications. This study uses 216 and 278 samples for normal and malicious Android applications. Various trained mathematical algorithms have been applied to both harmful and malicious data sets. Using correlation analysis, the study achieved a detection accuracy of 97.16%.

In [69], researchers are reviewing two important features used to detect Android malware, which are permissions, and system calls that can differentiate between benign and malicious applications through machine learning algorithms. The results present that authorization data was better at detecting malware than system call data. When using permission data to check for malicious activity on Android devices, the average rating accuracy was 80%. So, it is a credible way to detect malware.

From another view, the main defense against Android malware is a commercial mobile security product that primarily uses signature-based methods of the malware detection. Nevertheless, attackers can easily create procedures like dimming and refilling to avoid detection, which may require new defensive techniques that are difficult to avoid. In [68], the study focuses on analyzing API calls extracted from small files. It categorizes API calls that belong to some of the methods in small code of block. Based on the created code blocks, and then apply a deep learning method to detect the unknown new malware. The method achieved accuracy at 92.66%. In [67], researchers implement Android-based malware detection technique called DroidDetector. It is based on deep learning and able to detect if the application is malicious or not. With the increasing numbers of Android apps, the researchers are testing an in-depth analysis of exploits to fully describe malware using DroidDetector. The results present that deep learning is appropriate to characterize malware and is particularly effective with more training data. DroidDetector able to achieve 96.76% detection accuracy compared to conventional machine learning technology.

Research work presented in [33] tested malware detection techniques, compared API calling for code sequencing and code processing. Model detection rates are based on Latent Markov models and on static and dynamic data. This study proposes a hybrid technique based on Deep Automated Code (DAC) with Conventional Neural Network (CNN) to increase the accuracy and efficiency of malware detection. To get better accuracy of malware detection, they reconstructed high quality features of Android (apps) and used several neural networks.

Relay studies of the monolithic convolutional neural network structure as a non-linear function, and open-source function, are a "breakthrough" to expand defects and prevent congestion. Unsatisfactory and composite layers combined together with the contact layer to increase output capacity. Under these circumstances, neural networks show a strong ability to remove malware and detect malware with an accuracy of 99.80%. Training time with the DAC-CNN model is also reduced by 83% compared to the CNN-S model.

Paper ID	Ref. No	Paper Title	Year	Database
S1	44	SAFEDroid: using structural features for detecting android malwares	2018	Springer
S2	45	DroidDet: Effective and robust detection of android malware using static analysis along with rotation forest model	2018	Elsevier
S3	46	Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach	2018	Elsevier
S4	47	Research on data mining of permissions mode for AMD	2019	Springer
S5	48	Effective AMD with a hybrid model based on deep autoencoder and convolutional neural network	2019	Springer
S6	49	Analytics on Malicious Android Applications	2018	Scopus
S7	33	A comparison of static, dynamic, and hybrid analysis for malware detection	2017	Springer
S8	65	AMD using markov chain model of application behaviors in requesting system services	2017	Google scholar /Cornell University
S9	66	Static and dynamic analysis of android malware	2017	The International Conference on Information Systems Security and Privacy
S10	67	Droiddetector: android malware characterization and detection using deep learning: Tsinghua Science and Technology	2016	IEEE
S11	68	Droiddelver: An AMD system using deep belief network based on api call blocks	2016	Springer
S12	69	A comparison of features for AMD	2017	ACM
S13	70	Deep AMD	2017	ACM
S14	71	Semantic modeling of android malware for effective malware comprehension, detection, and classification	2016	ACM
S15	72	Dynamic permissions-based AMD using machine learning 3. techniques	2017	ACM
S16	73	CREDROID: AMD by network traffic analysis	2017	ACM

S17	**74**	**AMD using network behavior analysis and machine learning classifiers**	**2017**	**Google scholar**
S18	**75**	**A highly efficient random forest-based malware detection framework for Android**	**2017**	**Springer**

TABLE 1: List of Finally Selected Research Papers.

3. METHODOLOGY

At the first place 2327 related articles are selected that are published from the year 2005 to 2020. Table 1 shows 18 shortlisted articles based upon the following criteria:

A. publication year 2016-2019
B. research area malware detection techniques and/or Malware on android devices
C. the shortlisted article should reflect the experimental studies of malware android software detection and results.
D . articles which are available in full text.

3.1 Techniques Used for Detecting Malware Attacks on Android Devices

According to the survey, several research studies are conducted to characterize many of the existing malware android programs. A project that called Android Malware Genome differentiate existing malware in Android OS. Another study called Android Drebin was also conducted and compared various malware detection methods [20, 40]. Many data samples from malware android are available in [13].

No	Malware Family	Characterization	Malicious Activities
1	FakeInst	Send premium SMS Messages	SMS (Send, process, delete)
2	OpFake	Send premium SMS Messages	SMS (Send, process, delete), Send Device Data to remote Server, Download, install, delete package
3	SNDApps	Steal various information such as device ID, email ID, device address and phone number and download it to a remote server	Information Stealing
4	Boxer	Send SMS messages to a premium rated numbers	Send SMS
5	GinMaster	Steals sensitive information from devices and sends it to remote servers.	Send (device information, installed applications, network information) to a remote server.
6	VDLoader	Steals the personal information	Root access and Information Stealing
7	FakeDolphin	Gives you dolphin browser and signs up a user for the services without their knowledge	Information Stealing
8	DroidKungFu	It steals the information like IMEI, device, OS version and dumps into a local file that is sent to the remote server	Send Device Information ,Network information, Phone data, SD card Data to Remote Server Root Access, Botnet and Information Stealing
9	BaseBridge	It sends the confidential details like IMEI, SMS, IMSI to a remote serve	SMS (Send, process, delete), Send Device Data to remote Server, Download, install, delete package, Dial Phone Numbers, terminate process, Botnet and Information Stealing
10	JIFake	Send premium rated SMS Messages	Send SMS

TABLE 2: The Top Ten Android Malware Families.

Depending on the functionality, android malware can be split into different categories, such as spyware, trojans, viruses, phishing applications, bot processes, root kits and etc [4]. Refer to the latest research, table 2 presents a list of the top 10 families of android malware with descriptions and functions [20, 41-43]. Table 2 shows the main Android malware families examined for the last known malware. However, the analysis methods for AMD can be divided into three types: static analysis, dynamic analysis and hybrid analysis. This section presents the analysis result for those types.

For dynamic analysis, the research in [65] applied a detection technique on two malware datasets 4034 and 10024. These dataset includes malware and harmless applications respectively. Using the Service Monitor method and a Random Forest rating algorithm malicious codes were found in about 96% of the applications. Using k-division verification and Markov chain, the classification module was led to extract the features of the sample. Detected information, such as malware IMEI calls and it proved to be 67% accurate for detecting malware. 17% of the applications were classified as premium services. It was observed that it had a payload connected to the device. Mobile utilities like CPU and memory infected 8% and 2% of overall performance, respectively. Some malicious programs remain active even after downloading and installing them on the device until the task is activated. Some malwares perform this task, while others load during download, installation, and execution.

In general, the permissions are granted by android users when downloading and installing apps make a huge difference to the attacker access on the device. However, the default permissions are always found during download and installation of applications. In this time, malware is associated with harmless applications. Important monitoring is required at this stage to improve the security of the mobile platform [72].

For static analysis techniques, the proposed detected malware software on Nexus 5 using 103 and 97 datasets of malware and harmless applications respectively [66]. As a result, high-profile malware attacks were detected at API level [19]. Naïve Bayes experiment was used. XML recipients claim the right to create malicious software stored in Formatted File Relation (ARFF) format. When the random forest algorithm was used, the result was a 96.6% detection rate, which was 0.069% different from the worst detection algorithm. During the review of the code that runs an application using a trained system, the appearance of a malicious application cannot be easily identified.

To analyze raw data processed with Dalvik bytecode [70], it is recommended to unzip the APK file and repair the opcode. Rebuilding this file provides assembly instructions for extracting and parsing additional android application files, such as XML and other resource files. This method is similar to the n-gram process to detect malware [71]. A large dataset of 5,560 malware samples was used to detect android malware using four different detection algorithms and to validate the experimental results obtained for detection accuracy. The various detection techniques from the trained algorithm is characterized by the strength of the DSA when applied to the input and extraction layers of the model. The random forest algorithm has 97% detection accuracy than others.

Finally, hybrid analysis techniques offer a better detection rate than dynamic and static techniques. Using the deep learning aspect of artificial intelligence [67], the DroidDetector model [74] with AMD algorithms was developed. The hybrid technology collected a total of 192 Android malware and harmless examples for educational purposes. This model gave an accuracy of the recognition result of 96.60% with a difference of 0.0021% between the algorithms used. In some complex cases where the malware example is unknown, training and detection cannot be performed simultaneously to avoid malfunctions. It has been shown that the Hidden Markov model has powerful functions for bidirectional improvement in malware detection. The hybrid technology enables a precise comparative analysis of the static and dynamic detection rates. Using the semantic approach of this method [33], a hidden example sequence was extracted from job code and API calls using a hidden Markov model. The threshold of the ROC curve was

determined by reproducibility, precision and specificity. The Android Buster sandbox was used as an analysis tool to define and determine the maliciousness and positivity of the application. However, it is not possible to solve the malware obfuscation problem by applying the API call sequence to detect android malware. The order in which the malware functions are monitored also has no relation to the individual states of the HMM. This method cannot generate the initial malware distribution status in the diagram or in the call sequence.

Similarly, there was a study in [74] in which small malware files were extracted using the API call diagram. A total of 1,022 extracts from 1,216 suspicious android apps were created. The detection accuracy was 96.12%. Android anti-malware attacks have been overcome in this way, but it cannot be used to Android addiction attacks through machine learning. A study in [68] focused on blocking API calls and used the Deep Belief Network algorithm to extract two semantic clues from known and unknown malware that was developed for search tools (Droiddelver). Boltzmann created a limited two-part graph from the malware probability distribution in the input level of the model.

In scenarios where the Print Vim malware is identical, an index of small programs is provided in most cases that the malware is likely to print on the android mobile kernel. Compressing and decompiling in android application before extracting the API call layer requires little code between the Dalvik virtual machine and the application interface. Some android malware programs are primarily used to collect information about the system's call. These images are captured by the camera on mobile tracking devices. This malicious application is vulnerable to physical and informative users of the device. It can easily track the system files that can be exploited for financial gain. This is shown in a study in [67] that uses a relatively small set of malicious Android data.

4. COMPARATIVE STUDY OF AMD TECHNIQUES

The comparisons result of AMD Techniques are presented in Table 3. It shows the detection accuracy, detection techniques, strength and limitation state of the art research work.

ID	Year	Detection Technique	Detection Approach	Detection Accuracy	Strength	limitations
S8	**2017**	**Dynamic**	**ServiceMonitor**	**86%**	**Overcame fitting problem**	**Susceptible to transformation and mimicry attacks**
S9	**2017**	**Static**	**Reverse engineering**	**96.6%**	**Overcomes issues of Bytecode Encryption**	**Fail to execute using Monkey Runner**
S10	**2016**	**Hybrid**	**AI, Deep learning DroidDetector**	**94.60%**	**High-level learning representation**	**Lopsided ratio, Little optimization**
S7	**2017**	**Hybrid**	**Hidden Markov Models (HMMs)**	**N/A**	**Known and unknown malware samples were detection**	**Problem of imbalance and obfuscation**
S11	**2016**	**Hybrid**	**Deep learning framework**	**92.66 %**	**Malware image recognition**	**Malware depth features were not extracted, assembly language is required**
S12	**2016**	**Hybrid**	**Permissions data flow**	**80%**	**N/A**	**Sample was not streamed-lined**
S13	**2017**	**Static**	**Opcode**	**87%**	**The need for**	**This could not**

			sequence		hand-engineered was removed	address the problem of malware encryption
S14	2016	static	CFG and Bigram using DSA	87.0%	Efficiency and scalability can be achieved with this approach	Susceptible to malware loading and replication
S15	2017	Dynamic	Machine learning and CFG	Simple Logistic 84%, Baiyes 67.64%	Overcame malware polymorphism	Some samples evaded detection
S16	2016	Content based	CREDROID and Web of Trust	63%	Fast execution	Fails when APK is not generating network traffic.
S11	2017	Emulation Based	Machine learning	Random forest 98%, Nearest neighbor 96% AdaBoost 99%	Detects Zeroday, privilege escalation malware	Cannot determine how malware processes the data affected, detected virtual environment
S12	2017	Static	Support vector machine	89.9%	Very fast and cost effective	Bias and variance in features detection
S2	2018	Static	ensemble Rotation Forest (RF)	88.26%	highest performance algorithm	Information in the feature set affect the accuracy of the classifier
S5	2019	Hybrid	deep autoencoder (DAE) and convolutional neural network	99.8%	powerful ability in feature extraction and malware detection	training time using DAE-CNN model is reduced by 83% compared with CNN-S model.

TABLE 3: A Comparative Analysis of Studied Papers.

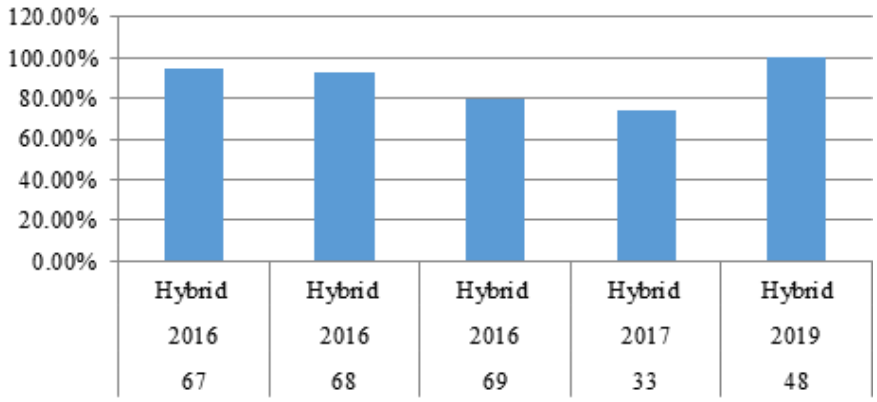

FIGURE 4: Hybrid Malware Detection Accuracy.

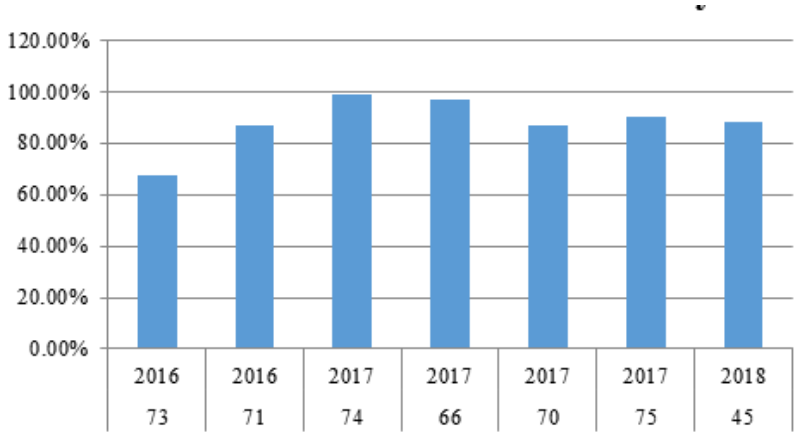

FIGURE 3: Static Malware Detection Accuracy.

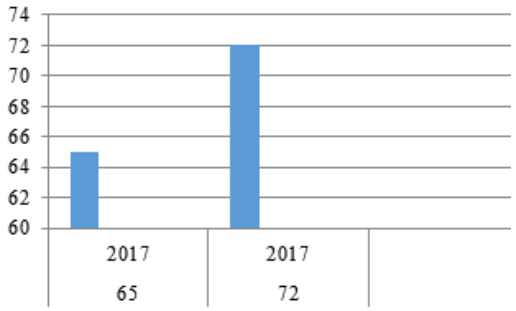

FIGURE 2: Dynamic Malware Detection Accuracy.

Figure 2 presents the dynamic malware detection accuracy result. Figure 3 present the static malware detection accuracy result. Figure 4 present the hybrid malware detection accuracy result. In all three figures the x-axis shows the papers reference no (see table 1) and y-axis show the accuracy percentage.

5. DISCUSSION

From the comparative analysis in previous section, a basic observation is observed in all detection techniques. That is, the use of small malware dataset. This hinders true evaluation of the detection efficiency since the sample size could not cover different edges of malware families. With such sample size, the technique might have seemed to perform proficiently but when implemented on a larger dataset, the result may not be the same. This could produce lopsided ratio with little optimization. Variation in detection rate by same algorithm in different detection scenarios is worrisome. Analysis of AMD techniques is significant to building an efficient detection tool by applying both the strengths and limitations identified in all the studied approaches.

5.1 Dynamic AMD Techniques

Dynamic analysis includes analyzing the behavior of the application at runtime. Dynamic capabilities include system calls, traffic, flows, and addresses in the network that monitors the operation of the system. Dynamic technique can overcome strings of detection issues such as malware fitting and oligomorphic form. The observed limitation is susceptibility to transformation attacks, vulnerability to mimicry attacks and its inability to run on un-rooted android devices. When opcode sequence approached, it overcomes the need for hand-engineering. This could not however address the problem of malware encryption.

The detection challenges range from known to unknown and simple to sophistication.

Transformation of the designed behavioral model could lead to malware obfuscation when the trained algorithm(s) and mutation approach are known by hackers or malware writers. From this research, it is clear that no detection technique developed and used by industries and individuals is 100% efficient in malware detection. As a result, occurrence of android malware has become a common attack threat to the users.

5.2 Static AMD Techniques

The static detection techniques are a lightweight computational method. It can quickly detect Android malware with low computational complexity and very high performance but still face some problems. As malware developers have started using many techniques to hide the malicious behavior of their applications, it has become a great challenge for detection methods based on static analysis. The studies conducted in this research showed that some of the proposed frameworks attempted to cover some of these techniques.

5.3 Hybrid AMD Techniques

The hybrid detection technique consists of static and dynamic methods that provide better detection accuracy. Other techniques studied are content and emulation-based detection. In this research it is observed that both dynamic and static techniques can be approached in different ways by applying diverse set of trained algorithms.

6. CONCLUSION

This research work presented a systematic literature review on various malware detection technologies for Android applications. This research identifies the limitations and strengths in each detection techniques through systematic literature review. The results obtained from this study reinforce the assertion that detection approaches designed for Android malware do not produce 100% efficient detection accuracy. This segment of the research presents a critical evaluation of the reviewed papers. The rationale behind making this comparative analysis is to give a well-defined understanding on the strengths and weaknesses that are identified in this research work. It is necessary to suggest a mixed method of malware detection using a machine learning approach that will overcome all limitations of static and dynamic analysis techniques. This method should be based on the detection of common malware. The hybrid method started by applying the static analysis technique on the local device then will execute the dynamic analysis on a remote server. Thus, it will be able to accurately detect more malware and consume less time, energy and resources. A comparative survey on detection techniques focusing primarily on identifying AMD techniques with their respective detection approaches, detection accuracy, and their corresponding strengths and limitations has not been explored before.

In addition, the study shows that many researchers still prefer to conduct informal literature reviews. However, it found that the quality of SLR become better, which indicates that researchers interested in this approach have become more effective in the methodology of SLR methods. The prevalence of topics covered by current DSLRs is somewhat limited.

7. REFERENCES

[1] Chien, E.: Motivations of recent android malware. Symantec Security Response, Culver City Press, California (2011).

[2] S.Birundha, Dr. V. Vanitha “Survey on Mobile Malware Detection Techniques in Android Operating System” International Journal on Applications in Information and Communication Engineering, vol. 2 issue. 4 Apr 2016.

[3] Saba Arshad, Abid Khan “Android Malware Detection and Protection: A Survey” International Journal of Advanced Computer Science and Applications, vol. 7 no. 2 2016.

[4] Nikita Rai, Dr, TriptiArjariya “A Survey on Detection Techniques of Android Malware” International Journal of Computer Security and Source Code Analysis (IJCSSCA), vol.1

issue.2 2015.

[5] https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-sers-since-2007/ last access on 2/3/2020

[6] “Gartner News Room”, Online Link: https://www.gartner.com/newsroom/id/3609817

[7] “99.6 percent of new smartphones run Android or iOS” 2016, Online Link: https://www.theverge.com/2017/2/16/14634656/android-ios-market-share-blackberry-2016

[8] “Android Security 2016 Year in Review”, Online Link: https://source.android.com/security/reports/Google_Android_Security_2016_Report_Final.pdf

[9] “Report: Top Android Security Problems in 2017” 2017, Online Link:https://dzone.com/articles/report-tojp-android-security-problems-in-2017

[10] PrajaktaSawle, A.B. Gadicha “Analysis of Malware Detection Techniques in Android” International Journal of Computer Science and Mobile Computing, vol. 3 issue. 3 Mar 2014.

[11] “Mobile Malware”, Online Link: http://www.webopedia.com/TERM/mobile_malware.html

[12] Yajin Zhou “Malgenome Project” 2011, Online Link: http://malgenomeproject.org

[13] “Koodous Beta APKs”, Online Link: https://koodous.com/apks

[14] “Research Gate Forum”, Online Link: https://www.researchgate.net/post/Where_can_I_get_Android_Malware_Sampe

[15] Vinisha Malik, Naveen Malik “Analysis of Android Malware and their Detection Techniques” International Conference on Parallel, Distributed and Grid Computing (PDGC) IEEE, 2016.

[16] 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.

[17] GORLA, A., TAVECCHIA, I., GROSS, F., AND ZELLER, A. Checking app behavior against app descriptions. In Proceedings of the 36th International Conference on Software Engineering (New York, NY, USA, 2014), ICSE 2014, ACM, pp. 1025–1035.

[18] PANDITA, R., XIAO, X., YANG, W., ENCK, W., AND XIE, T. Whyper: Towards automating risk assessment of mobile applications. In Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13) (Washington, D.C., 2013), USENIX, pp. 527–542.

[19] HUANG, J., ZHANG, X., TAN, L., WANG, P., AND LIANG, B. Asdroid: Detecting stealthy behaviors in android applications by user interface and program behavior contradiction. In Proceedings of the 36th International Conference on Software Engineering (New York, NY, USA, 2014), ICSE 2014, ACM, pp. 1036–1046.

[20] Daniel Arp, MichealSpreitzenbarth “DREBIN: Effective and Explainable Detection of Android in your pocket” NDSS, Feb 2014.

[21] CHAKRADEO, S., REAVES, B., TRAYNOR, P., AND ENCK, W. Mast: Triage for market-scale mobile malware analysis. In Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks (New York, NY, USA, 2013),WiSec ’13, ACM, pp. 13–24.

[22] BURGUERA, I., ZURUTUZA, U., AND NADJM-TEHRANI, S. Crowdroid: behavior-based

malware detection system for android. In Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices (New York, NY, USA, 2011), SPSM '11, ACM, pp. 15–26.

[23] AAFER, Y., DU, W., AND YIN, H. Droidapiminer: Mining api-level features for robust malware detection in android. In Security and Privacy in Communication Networks - 9th International ICST Conference, SecureComm 2013, Sydney, NSW, Australia, September 25-28, 2013, Revised Selected Papers (2013), pp. 86–103.

[24] FENG, Y., ANAND, S., DILLIG, I., AND AIKEN, A. Apposcopy: Semantics-based detection of android malware through static analysis. In Proceedings of the 22Nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (New York, NY, USA, 2014), FSE 2014, ACM, pp. 576–587.

[25] GRACE, M., ZHOU, Y., ZHANG, Q., ZOU, S., AND JIANG, X. Riskranker: Scalable and accurate zero-day android malware detection. In Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (New York, NY, USA, 2012), MobiSys '12, ACM, pp. 281–294.

[26] POEPLAU, S., FRATANTONIO, Y., BIANCHI, A., KRUEGEL, C., AND VIGNA, G. Execute This! Analyzing Unsafe and Malicious Dynamic Code Loading in Android Applications. In Proceedings of the ISOC Network and Distributed System Security Symposium (NDSS) (San Diego, CA, February 2014).

[27] ENCK, W., ONGTANG, M., AND MCDANIEL, P. On lightweight mobile phone application certification. In Proceedings of the 16th ACM Conference on Computer and Communications Security (New York, NY, USA, 2009), CCS '09, ACM, pp. 235–245.

[28] CHRISTODORESCU, M., JHA, S., SESHIA, S. A., SONG, D., AND BRYANT, R. E. Semanticsaware malware detection. In Proceedings of the 2005 IEEE Symposium on Security and Privacy (Washington, DC, USA, 2005), SP '05, IEEE Computer Society, pp. 32–46.

[29] GRIFFIN, K., SCHNEIDER, S., HU, X., AND CHIUEH, T.-C. Automatic generation of string signatures for malware detection. In Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection (Berlin, Heidelberg, 2009), RAID '09, Springer-Verlag, pp. 101–120.

[30] Lok-Kwong, Y., Yin, H.: DroidScope: Seamlessly Reconstructing the OS and Dalvik Semantic Views for Dynamic Android Malware Analysis. In: Symp. 2017. USENIX security symposium, pp. 569—584 (2017).

[31] Anastasia, S., Gamayunov, D.: Review of the mobile malware detection approaches: Parallel, Distributed and Network-Based Processing (PDP). In: Proc. 2015. IEEE 23rd Euro micro International Conference, pp. 600--603(2015).

[32] Wang, H., Li, H. and Guo, Y.: Understanding the Evolution of Mobile App Ecosystems: A Longitudinal Measurement Study of Google Play. In: Conf. 2019. ACM World Wide Web Conference. pp. 1988—1999 (2019).

[33] Anusha, D., Troia, F. D., Visaggio, C. A., Austin, T. H., Stamp, M.: A comparison of static, dynamic, and hybrid analysis for malware detection. Journal of Computer Virology and Hacking Techniques, vol. 13, no. 1, pp. 1--12 (2017).

[34] Shuaifu, D. Y., Liu, T., Wang, T., Zou, W.: Behaviorbased malware detection on mobile phone," In Wireless Communications Networking and Mobile Computing. IEEE International Conference, pp. 1—4 (2016).

[35] Latika, S., Hofmann, M.: Dynamic behaviour analysis of android applications for malware detection. In IEEE International Conference on Intelligent Communication and Computational Techniques (ICCT), pp. 1--7 (2017).

[36] Zhang, H., M.A. Babar & P. Tell. Identifying relevant studies in software engineering. Information and Software Technology 53(6): 625-637 (2011).

[37] Kitchenham, B. & S. Charters. Guidelines for performing Systematic Literature Reviews in Software Engineering. Keele University and Durham University Joint Report EBSE 2007-001(2007).

[38] Ilyas, M. & S.U. Khan. Software integration model for global software development. In: 5th International Multitopic Conference (INMIC), Islamabad, p. 452-457 (2012).

[39] Ashawa MA, Morris S. (2019) Analysis of Android malware detection techniques: a systematic review, International Journal of Cyber-Security and Digital Forensics. Volume 8, Issue 3, 2019, pp. 177-187

[40] “The Drebin Dataset” 2012, Online Link: https://www.sec.cs.tu-bs.de/~danarp/drebin/index.html

[41] “Current Android Malwares” 2016, Online Link: https://forensics.spreitzenbarth.de/android-malware/

[42] “Current Android Viruses List 2020” 2020, Online Link: https://drfone.wondershare.com/android-tips/top-android-virues-list.html

[43] “Trojan:Android/GinMaster.A”, Online Link: https://www.f-secure.com/v-descs/trojan_android_ginmaster.shtml

[44] S. Sen, A.I. Aysan, J.A. Clark, "SAFEDroid: Using Structural Features for Detecting Android Malwares" in in Security and Privacy in Communication Networks, Cham: Springer International, 2018.

[45] H.-J. Zhu et al., "DroidDet: Effective and robust detection of android malware using static analysis along with rotation forest model", Neurocomputing, vol. 272, pp. 638-646, 2018.

[46] S. Chen et al., "Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach", Computers & Security, vol. 73, pp. 326-344, 2018.

[47] Wang, C., Xu, Q., Lin, X., & Liu, S. (2019). Research on data mining of permissions mode for Android malware detection. Cluster Computing, 22(6), 13337-13350.

[48] W. Wang, M. Zhao, J. Wang, "Effective android malware detection with a hybrid model based on deep autoencoder and convolutional neural network", Journal of Ambient Intelligence and Humanized Computing, 2018.

[49] H. Abubaker, S.M. Shamsuddin, A. Ali, "Analytics on Malicious Android Applications", International Journal of Advances in Soft Computing & Its Applications, vol. 10, no. 1, 2018.

[50] S.Y. Yerima, S. Sezer, "DroidFusion: A Novel Multilevel Classifier Fusion Approach for Android Malware Detection", EEE Transactions on Cybernetics, pp. 1-14, 2018.

[51] Kirubavathi, G., & Anitha, R. (2018). Structural analysis & detection of android botnets using machine learning techniques. International Journal of Information Security, 17(2), 153-167.

[52] G. Tao et al., "MalPat: Mining Pattems of Malicious and Benign Android Apps via

Permission-Related APIs", EEE Transactions on Reliability, vol. 67, no. 1, pp. 355-369, 2018.

[53] M. Fan et al., "Android Malware Familial Classification and Representative Sample Selection via Frequent Subgraph Analysis", EEE Transactions on Information Forensics and Security, vol. 13, no. 8, pp. 1890-1905, 2018.

[54] Martín, A., Menéndez, H. D., & Camacho, D. (2017). MOCDroid: multi-objective evolutionary classifier for Android malware detection. Soft Computing, 21(24), 7405-7415.

[55] Feizollah, A., Anuar, N. B., Salleh, R., Suarez-Tangil, G., & Furnell, S. (2017). Androdialysis: Analysis of android intent effectiveness in malware detection. computers & security, 65, 121-134.

[56] Wang, X., Wang, W., He, Y., Liu, J., Han, Z., & Zhang, X. (2017). Characterizing Android apps' behavior for effective detection of malapps at large scale. Future generation computer systems, 75, 30-45.

[57] F. Idrees et al., "PIndroid: A novel Android malware detection system using ensemble leaming methods", Computers & Security, vol. 68, pp. 36-46, 2017.

[58] S. Alam et al., "DroidNative: Automating and optimizing detection of Android native code malware variants", Computers & Security, vol. 65, pp. 230-246, 2017.

[59] K. Sokolova, C. Perez, M. Lemercier, "Android application classification and anomaly detection with graph-based permission pattems", Decision Support Systems, vol. 93, pp. 62-76, 2017.

[60] Y. Du, J. Wang, Q. Li, "An Android Malware Detection Approach Using Community Structures of Weighted Function Call Graphs", EEE Access, vol. 5, pp. 17478-17486, 2017.

[61] Palumbo, P., Sayfullina, L., Komashinskiy, D., Eirola, E., & Karhunen, J. (2017). A pragmatic android malware detection procedure. Computers & Security, 70, 689-701.

[62] Yang, X., Lo, D., Li, L., Xia, X., Bissyandé, T. F., & Klein, J. (2017). Characterizing malicious android apps by mining topic-specific data flow signatures. Information and Software Technology, 90, 27-39.

[63] N. Milosevic, A. Dehghantanha, K.R. Choo, "Machine learning aided Android malware classification", Computers & Electrical Engineering, vol. 61, pp. 266-274, 2017.

[64] Karbab, E. B., Debbabi, M., Derhab, A., & Mouheb, D. (2017). Android malware detection using deep learning on API method sequences. arXiv preprint arXiv:1712.08996.

[65] Majid, S., Amini, M.: Android Malware Detection using Markov Chain Model of Application Behaviors in Requesting System Services" arXiv preprint arXiv:1711.05731 (2017).

[66] Ankita, K., Troia, F. D., Stamp, M.: Static and Dynamic Analysis of Android Malware: In ICISSP, pp. 653--662 (2017).

[67] Zhenlong, Y., Lu, Y., Xue Y.: Droiddetector: android malware characterization and detection using deep learning: Tsinghua Science and Technology, pp. 114-123. IEEE Press, (2016).

[68] Stamp, M. Anusha, D. F: A comparison of static, dynamic, and hybrid analysis for malware detection"Journal of Computer Virology & Hacking Techniques,vo.13, no.1, pp. 1--12 (2017).

[69] Shifu, H., Saas, A., Ye, Y., Chen, L.: Droiddelver: An android malware detection system using deep belief network based on api call blocks: In International Conference on Web-Age

Information Management, pp. 54--66, Springer, Cham (2016).

[70] Matthew, L., Atkison, T.: A comparison of features for android malware detection. In: Proc. 2017. ACM South East Conference, pp. 63--68. ACM (2017).

[71] Niall, M. D., Rincon, B., Kang, S.: Yerima, P. Miller, S. Sezer and Y. Safaei, “Deep android malware detection. In: Proc. 2017. ACM Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy, pp. 301—308 (2017).

[72] Guozhu, M., Yinxing, X., Zhengzi, X.: Semantic modelling of android malware for effective malware comprehension, detection, and classification. In: Proc. 2016. ACM Proceedings of the 25th International Symposium on Software Testing and Analysis, pp. 306-317 (2016).

[73] Arvind, M. and Singh, P.: Dynamic permissions-based Android malware detection using machine learning techniques. In: Proc. 2017. ACM Proceedings of the 10th Innovations in Software Engineering Conference, pp. 202-210 (2017).

[74] Jyoti, M., and Kaushal, R.: CREDROID: Android malware detection by network traffic analysis. In: Proc. 2016. Proceedings of the 1st ACM Workshop on Privacy-Aware Mobile Computing, pp. 28-36(2016).

[75] Gabriele, C., and Aria, H.: Android Malware Detection Using Network Behavior Analysis and Machine Learning Classifiers, pp. 25—32(2017).

[76] Hui-Juan, Z., Jiang, T., B., Shi, W., Cheng, L.: HEMD: a highly efficient random forest-based malware detection framework for Android.

[77] Agrawal, P., Trivedi, B., : Machine Learning Classifiers for Android Malware Detection, In Proceedings of ICDMAI 2020 Springer, Vol. 1, Aug 19, 2020.

[78] Sasidharan, S. K., Thomas, C., ProDroid – An Android malware detection framework based on profile hidden Markov model, Pervasive and Mobile Computing, Elsevier, Jan 21, 2021.

[79] Roy, A., Jas, D. S., Jaggi, G., Sharma, K.: Android Malware Detection based on Vulnerable Feature Aggregation, Procedia Computer Science, Elsevier, pp. 345-353, July 01, 2020.