Asian Journal of Research in Computer Science

7(4): 59-69, 2021; Article no.AJRCOS.68044
ISSN: 2581-8260

Efficiency of Malware Detection in Android System: A Survey

Maria A. Omer[1*], Subhi R. M. Zeebaree[1], Mohammed A. M. Sadeeq[1], Baraa Wasfi Salim[2], Sanaa Mohsin[3], Zryan Najat Rashid[4] and Lailan M. Haji[5]

[1]Duhok Polytechnic University, Duhok, Kurdistan Region, Iraq.
[2]Nawroz University, Duhok, Kurdistan Region, Iraq.
[3]University Information Technology and Communication, Baghdad, Iraq.
[4]Sulaimani Polytechnic University, Sulaimani, Kurdistan Region, Iraq.
[5]University of Zakho, Duhok, Kurdistan Region, Iraq.

Authors' contributions

This work was carried out in collaboration among all authors. Authors MAO and SRMZ prepared the detailed review of previous works related to Efficiency of Malware Detection in Android System. Both of authors MAMS and BWS wrote the first draft of the manuscript. In addition, authors SM, ZNR and LMH managed the analyses and discussion parts of the study. All authors read and approved the final manuscript

Article Information

DOI: 10.9734/AJRCOS/2021/v7i430189
Editor(s):
(1) Dr. Xiao-Guang Lyu, Huaihai Institute of Technology, China.
Reviewers:
(1) H. K. Shreedhar, Global Academy of Technology, India.
(2) Marco Antonio Campos Benvenga, Universidade Paulista- UNIP, Brazil.
Complete Peer review History: http://www.sdiarticle4.com/review-history/68044

Review Article

Received 20 February 2021
Accepted 24 April 2021
Published 26 April 2021

ABSTRACT

Smart phones are becoming essential in our lives, and Android is one of the most popular operating systems. Android OS is wide-ranging in the mobile industry today because of its open-source architecture. It is a wide variety of applications and basic features. App users tend to trust Android OS to secure data, but it has been shown that Android is more vulnerable and unstable. Identification of Android OS malware has become an emerging research subject of concern. This paper aims to analyze the various characteristics involved in malware detection. It also addresses malware detection methods. The current detection mechanism utilizes algorithms such as Bayesian algorithm, Ada grad algorithm, Naïve Bayes algorithm, Hybrid algorithm, and other algorithms for machine learning to train the sets and find the malware.

**Corresponding author: E-mail: Mariaayob997@gmail.com;*

Keywords: Android operating system; malware detection; dynamic analysis; static analysis; hybrid analysis.

1. INTRODUCTION

The Android operating system (OS) for smart mobile devices has been the most popular since it was launched in 2008. In 2019 around 86.6 % of worldwide smart phone sales were based on Android. As the most popular mobile operating system rising, mobile malware has taken the Android platform as their target. Researches have shown that personal privacy theft has recently become a significant form of attack among these malware attacks [1]. Nearly half of Android malware is multifunctional Trojans that steal personal information stored on the user's phone. By the end of Dec 2020, over 3 million apps were on Google Play, the official store for Android apps [2]. Because of numerous reasons like Android applications' available environmental method, management of its coarse-grained permission control and the Capability to summon code from the third party, there are several Surfaces for security attacks available that seriously threatens Android apps integrity [3]. Statistics indicate that only in 2016, more than 3.25 million malware-infected Android apps were discovered, indicating that a new malware of Android app was discovered approximately each10 seconds [4]. Many technologies, including platform strengthening, malware prevention, developer feedback, and vulnerability recognition, ensure the stability of the Android ecosystem [5]. Amongst the different protection choices that can prevent malware from being released into or installed and used in the Android apps marketplace, Android malware detection is a commonly utilized security defense tool [6]. The Android malware detection method could be categorized into static, dynamic, and hybrid detection [7, 8].

Without the Android app, static detection is focused on the review of suspect code. It can achieve high code coverage but faces multiple countermeasures, such as dynamic code loading and code obfuscation [9]. Dynamic identification, on the other hand, involves the Software application being tested by running the code [10]. This will expose threats that are not easy to recognize by static analysis but are available for dynamic detection with relatively high computing resources and time costs [11]. Hybrid detection is a method that incorporates dynamic detection and static detection to achieve a balance between detection performance and effectiveness [12]. Compared to traditional techniques, such as signature-based malware detection, machine learning-based recognition, which is based on the detection of unusual characteristics in identified malware, can recognize previously unknown types of malware and continue with the detection of Android malware [13,14]. Whether based on static, dynamic or hybrid processing strategies, machine learning theory has widely used to detect Android malware [15].

The organization of this paper consists of six sections. Section 1 produces the introduction, while the malware detection addressed in section 2. Android operating system is explained in section 3. Details of previous works produced in literature review at section 4. Section 5 related to the discussion and comparison of reviewed works in section 5. Finally, the conclusion produced in section 6.

2. MALWARE DETECTION

Malware is any program with a mischievous intent (malicious software) [16]. It can be written to interrupt regular activity, gather confidential information, control the device without the user's awareness, circumvent access controls, or show inappropriate advertising [17]. Furthermore, malware and unintentionally damaging applications are referred to collectively as malware [18]. The key categories in which malware can be categorized are viruses, Trojans, ransom ware, worms, botnets, root kits, etc. It has been noted that the evolution of Android malware is at an accelerated speed. Since the first-ever virus [11], malware for mobile devices has evolved enormously [19].

2.1 Static or Structural Analysis Architecture

It is a technique of inspecting apps by verifying without executing the application code [20]. The procedure provides an understanding of the code structure and can help to understand the features it will execute [21]. In this approach, code coverage has maximized since it only requires the study of source code [22]. Another advantage of static analysis is that it discloses malicious intentions without paying the price of being detected and facing losses inaccurate execution [23]. Due to this feature, code obfuscation is the main downside of this method, so pattern matching is not difficult to detect the behavior of

the application's malicious acts [24]. This technique will detect logical inconsistencies, runtime errors and scheduled security breaches [25]. API calls and permissions are the most commonly used static functions [26]. However, this technique is highly ineffective in the presence of code obfuscation and dynamic loading of code, with many apps suffering from challenges such as event-driven Android application in the static investigation process [27]. The approach of Static analysis includes Signature-Based Approach and Permission-Based Analysis [15].

2.2 Dynamic Analysis

The application needs to be executed on an independent framework to track its behavior [28]. The reverse to static analysis, the dynamic analysis of the code execution process is necessary so that the usual behaviors of the application. Observation Network activity, API calls and trying, file changes to Identify the system calls have performed by dynamic analysis. The approach of Dynamic analysis includes Taint Analysis and Anomaly-Based Detection [29].

2.3 Hybrid Analysis

The hybrid analysis consists of integrating both dynamic and static analysis to overcome their respective shortcomings and achieve the primary goal of achieving the best detection results. Resources with a hybrid approach to analysis includes Andrus and Mobile Sandbox [30].

3. ANDROID OPERATING SYSTEM

The mobile market has taken off, rising rapidly in the past ten years. Android has become the most prominent smart OS [31]. More than 80 percent of the entire smartphone market has so far accounted for the number of Android devices [32]. The Android OS also becomes a favorite target for malicious criminals because it is competitive and popular with many computer users and developers [33]. One of the Android malware detected in September 2010 was first discovered on a smartphone. In the future, the number of malicious applications would steadily rise shortly afterwards [34].

Security researchers identified thousand of malware apps over the years that followed [35]. According to an analysis done by G data 2, the amount of malware for Android devices reached 3.2 million in the third quarter of 2018 and rose by 40 % year-on-year. According to Google Play, Android apps have become 'popular in real life' because more people own Android devices than iOS ones [36]. In September 2018, the number of Android apps was nearly 2.6 million, but many malicious apps are still hidden on the Android market and pose a danger to users [37]. Without the consent of the consumers, Android malware has begun to install and work on smartphones. This malware can do different things [38,39]. They include browser hijacking and taking over the targeted machine, capturing personal information about the owner, stealing data from the machine, modifying settings on the device, and downloading other non-intended programs [40]. These acts would seriously violate the individual's right to do with their land. Because of this, there would be a loss of interest for users [41].

Android malware can be classified into four groups based on these behaviors, including malware installation (e.g. update attack, repacking, and drive-by download), malware activation, malicious payloads (e.g. privilege escalation, finance fee, data collection, and remote control), and misuse of Permissions [42,43]. Practitioners and researchers suggest various approaches, primarily static analysis and dynamic analysis, to counter these threats, as mentioned earlier. The APK for malware detection is executed and enforced through dynamic analysis [44].

4. LITERATURE REVIEW

This section summarizes the numerous research papers on malware detection on smartphones:

Paolo palumbo et al. [45] proposed using a fine-grained malware detection mechanism to use the Naïve base algorithm for malware detection. They developed an ensemble strategy to detect malware on Android smartphones as input for support vector machine on various APKs, and it uses the automatic Naïve Bayes classifier. It focuses only on the organized mining and activity patterns of malware families. The result of this model was that 1,20,000 samples were successful.

In order to cover all analyses, shahid Alam et al. [46] use Automatic labelling. The tag starts when the input is given. It operates based on semantic-based signatures of particular control flow patterns and allows the Malware detection system in native code or byte code. The AOT compiler has used to translate byte code into machine code. This work results in 94.48 %. The

downside to this method is parallelization to maximize run time.

Similar to the work above, Venkatesh Gowri Shankar et al.[47] suggested an androTaint model that operates based on dynamic taint analysis. It categorizes the conduct of the features and app. to detect the malware, they use four ways, they are automatic fragmentation and tagging process and anatomy detection, core framework and DDI hooking and taint tag identification, it manages all user events and inputs to determine the Vulnerabilities, the results of this model have detected 86% of malicious applications, 94% of aggressive application 90% of the benign application, and risky apps are 95%. The downside of this model requires much time to analyzing the malware.

A method Fg detection of Fine-Grained Android Malware was suggested by Dongfanghi et al. [48]. It extracts the app's features, translates vector features into low-dimensional features, and detects whether it is malicious. The same applies to classification algorithms like a decision tree and naive Bayes. Also, they use transformation techniques to decrease the problem of the multi-binary problem from the multi-class problem. To solve address class problems, the adaptation approach has used to broaden. As a result of this work, 89.2% of the malware has been detected. The demerit is less accurate, and more models based on machine learning may have been used to detect more malware.

A zero-day malware detection approach using machine learning was proposed by Gurdit Singh et al. [49]. In both dynamic and static analysis, it detects more binary malware. The Waikato knowledge analysis environment implemented by the machine learning algorithm is used (WEKAA). Static and dynamic features include the section count that the file size packer uses, the order that is executed on the network activities and services register. To set up and construct system report review, JavaScript object notation was used. The outcome of this model is the execution of 3130 portable executable files containing 1410 benign and 1720 malicious files. In this method, the risky and benign apps are not classified. They have displayed only offensive apps.

Like the work above, the AMDEC Anomaly detection malware detection of the android model was suggested by Fariba Ghaffar et al. [50] that ensemble classifier works on combining Merge classifiers to detect malware, the ensemble's classifiers consist of different heterogeneous one-class classifiers. This focuses primarily on knowledge from the network used by the app. It will be divided into two groups. These are detections based on entropy. This utilizes variables such as hardware features that have needed suspicious permission from the API91.73% of the detection rate is the product of this process. The downside to this strategy is that it has a decreased accuracy rate by triggering more false alarms to detect only zero-day malware.

Based on the work of Tong et al.[51] proposed a model using a hybrid approach to perform static and dynamic analysis rather than a machine learning process. This technique built a standard set of standard patterns and negative patterns and then compared these patterns to detect malware and benign applications. To produce frequent malware patterns and to detect malware, this hybrid approach has used. Without the user's awareness, the data flow analyzer shielded the user is sensitive data transfer to the third person. It then used the centroid device and lightweight combining technique to classify malicious apps. The scope of this work is unique to specific mobile operating systems.

Hangliang Liang et al. [52] Suggested end-to-end detection method malware of Android. It operates on apps that run and retrieve data from those apps and determines without any manual interference if the sequence is malicious. They use the ADA GRAD algorithm optimization to identify and train malware patterns. They chose several convolution layers after extracting the list of API calls. They use an android virtual device to map the system call chain. They track the process of implementation. The SoftMax Layer is used to finish the classification task in the application phase. As a result, the algorithm yields 93 per cent precision. The downside to this algorithm is that malicious fragments have not been eliminated from the call chain system, leading to decrease precision in detecting malware.

Ali feizollah et al. [53] study of android intention efficacy in malware detection has proposed a method of Andro analysis. The efficacy of Android is tested intensively in this approach. Some of the powerful features are semantically rich, with distinct strength compared to other app features. To learn the malware pattern, it uses the Bayesian network algorithm, and a dual-process algorithm has often used to learn both

network structure and probability tables. The Genetic Search Algorithm has used to maximize the outcome of the Bayesian network. For evaluating the Bayesian network structure, the Mill Climber search algorithm has used. An accuracy rate of 90.3% is the outcome of this strategy. The accuracy rate has found to be lower, even though effective methods have used.

Mathur et al. [54] presented the new Android malware identification system NATICUSdroid, benign and malware were examined and classified using native and customized Android permissions statistically chosen features different machine-learning (ML) classifications. The necessary permissions based on the pattern are analyzed in more than 29,000 benign or malware acquired during 2010–2019. These defined licenses, which include both indigenous and custom permits, are then collected. Lastly, the author uses select functional methods and test 8 ML NATICUSdroid algorithms to differentiate between benign and malware applications. Experimental findings show that 97 percent accurate, 3.32 percent false-positive and 0.96 f-measuring worked better in Random Forest classification models.

Gao et al. [55] Explored the malware classification application of the graphical neural network. The author is developing a GDroid prototype device. Experiments reveal that GDroid successfully detects 98.99% of Android malware with a less than 1% low false-positive rate, exceeding current approaches. In the Malware family classification mission, the author also achieves an overall accuracy of nearly 97 percent by exceeding the guidance. Please work with the QI-ANXIN Technology Research Institute to assess its effect on the natural world, and GDroid also delivers good real-world results.

Wang et al. [56] First, a multi-dimensional, kernel weight-based detection (WBD) intended to classify and understand the characteristics of Android malware and benign applications. Besides, our program agent automatically orchestrates and implements thousands of malicious and benevolent applications for data processing and storage. We test 112 kernel properties of the execution of the Android task data structure in a series of datasets of different sizes for identification accuracy. We think that memory and signal features lead to a more accurate ranking than schedule and other task state descriptors mentioned in our article. Memory-related characteristics, in particular, include fine-grain grade classification policies to maintain better grade accuracy than those relating to signals and others. We also analyze and test 80 newly infected attributes in the task layout of the Android kernel, prioritizing 70 critical features based on dimensional reductions to maximize high-dimensional classification performance. Our second contribution is to show that our approach can achieve 94-98 % precision and 2 percent -7 percent false-positive rates relative to current techniques with a shortlist of task-structure functionalities (16 or 32 functions). In contrast, malware applications with reduced dimensional characteristics are being found to abbreviate malware detections online and advance them appropriately.

Şahin et al. [57] A malware identification framework built on machine learning has proposed differentiating Android from benign apps. The objective is to eliminate redundant functionality during the feature selection stage in the proposed Malware Detection Scheme using a linear feature selection method based on regression. This reduces the size of the function vector, reduces training times and can be used in real-time malware detection systems in the classification model. The maximum features reduction ratio as (0.961) is achieved by depending at least 27 features based on the F-measurement metric when the findings have assessed.

Cai et al. [58] proposed a new Android malware identification scheme based on the weighting and classifier weighting function, known as JOWMDroid. First, features in eight types are removed from the Android device kit and then a range of critical features have chosen for malware detection with an information advantage. The first weight for each chosen feature is then determined using three computer models, followed by five weight maps to map the first weights to the end weights. Finally, with the differentially evolving algorithm, the weight mapping function and classifier parameters are jointly optimized. The experimental findings show that the suggested procedure exceeds four advanced approaches for weighting features and allows weight-conscious classification more competitive.

5. COMPARISON AND DISCUSSION

Depending on the reviewed research in the literature review section, a summarized comparison has extracted as shown in Table-1. Hence, the following points can be highlighted:

Table 1. Comparison of various research works in malware detection

Authors	Contributions	Attributes Used	Demerits/ Future directions	Accuracy obtained
Gurdit et.al. [49], 2016	Zero-Day Malware Detection, With Fewer Variables, Malware	Network activity, Part count, operations of the file system.	In this work, there are fewer parameters used to detect the Apps for Malware	90%
Shahid et.al. [46], 2016	Detecting Android Malware from Native Code Variant by Automated Droid Variant Taint	Native code, identifier, permission, tag	APPLN does not run in the background so that the malware can crash the computer.	94.48%
Tong et.al. [51], 2016	Hybrid method on Mobile Detection of malware inside Android	Count of Section, identifier, operators of the network, permission	This method does not extend to all operating systems.	90%
Ali et.al. [53], 2016	Method of Andro It creates multiple error warnings and detects only zero-day malware.	Filtered intent, services, permission.	The method causes more false alerts, and fewer variables have used.	90.3%
Venkate et.al. [47], 2017	Works in Dynamic Analysis under the Definition the Taint Mechanism for androids.	Broadcast recipients, native code, Services, identifiers, network operator, tag, activities, and permissions.	Malware detection takes more time.	86%
Feng et.al. [59], 2017	Complex Flow Android Environment Detection	Permission, native code, Identifier, API calls	Machine learning approaches are used minimal attributes to classify malware.	89%
Dongfang et.al. [48],2017	Fine-Grained Android Malware Detection	Permission, purpose filtered, calls to the API, identifier	Due to the simple machine learning method, it produces a lower accuracy score.	89.2%
Fariba et.al. [50], 2017	Anatomy Based Android Malware Detection	Service, permission, broadcast, network operator	It creates multiple error warnings and detects only zero-day malware.	91.7%
Paolo et.al. [45], 2017	There is a pragmatic way for testing malware for Android phones.	Permission, activities, identifier, network operators	It safeguards only ransomware attacks	89%
Hangliang et.al. [52], 2017	End to End of Detection of malware Android in Platform of Android	Identifiers, services, permission, tag	Benign and dangerous apps are not adequately categorized	93%

Authors	Contributions	Attributes Used	Demerits/ Future directions	Accuracy obtained
Mathur et al. [54], 2021	Presented the new Android malware identification system NATICUSdroid.	The author uses select functional methods and tests 8 ML NATICUSdroid algorithms to differentiate between benign and malware applications.	3.32 percent false-positive and 0.96 f-measuring worked better in Random Forest classification models.	97%
Gao, Han et al. [55], 2021	developing a GDroid prototype device	Explored the malware classification application of the graphical neural network	Please work with the QI-ANXIN Technology Research Institute to assess its effect on the natural world, and GDroid also delivers good real-world results.	98.99%
Wang et al. [56], 2021	-First, a multi-dimensional, kernel weight-based detection (WBD) intended. -show that our approach can achieve 94 percent -98 percent precision and 2 percent -7 percent false positive	Classify and understand the characteristics of Android malware and benign applications.	Malware applications with reduced dimensional characteristics are being found to abbreviate malware detections online and advance them appropriately.	94-98%
Şahin, et al. [57], 2021	A malware identification framework built on machine learning is proposed.	The objective is to eliminate redundant functionality during the feature selection stage in the proposed Malware Detection Scheme.	The maximum features reduction ratio as (0.961) is achieved by depending at least 27 features based on the F-measurement metric when the findings have assessed.	------
Cai et al. [58], 2021	proposed a new Android malware identification scheme based on the weighting and classifier weighting function, known as JOWMDroid	the Android device kit and then a range of critical features have chosen for malware detection with the information advantage	Procedure exceeds four advanced approaches for weighting features and allows weight-conscious classification more competitive.	97-97.5%

- Many of the research work in the manual form to detect malware, and the scanning process must also be initiated at all times. Any time the new apps begin, the detection apps close.
- In the android platform, more new types of malware have occurred that the digital signature does not recognize the pattern-based malware.
- Some of the methods of detection yield a greater false alarm rate. The simple machine language strategy also recognizes the innocuous apps as hostile behaviour and does not detect poisoning attack and mimicry.
- Often, malicious fragments have not been deleted from the device's call sequence scheme, so detecting the malware has a lower accuracy rate.
- The parameters/factors used are restricted for malware detection. Research works with limited attributes that are not enough to detect malware showing a more precise pace.
- Some strategies could detect only zero-day malware, creating more false alarms and a lower rate of accuracy.
- Few approaches detections of the platform utilized in smartphones.
- Machine learning has used at an early stage.

6. CONCLUSION

With the Internet of Things popularized, 5G and Mobile intelligent machines build other innovations quickly. The size of Android installed applications Smart devices, including cell phones and laptops, are now available. However, there was a rise in the number of Platform-focused malware. This attracted many people. Much research into Android device detection Malware is influenced. The Artificial Introduction Methods of knowledge like machine learning are very much Enhanced opportunities for Android malware detection.

This paper has discussed core attributes/parameters as well as different types of Android malware identification techniques. Since Android is open, many malware has hidden in many seemingly harmless applications in Android markets. Such malware badly jeopardizes Android's protection. The intruder has access to user data such as notes, emails, bank mTANs, locations, and so on. It also included an update on recent research work on established parameters aimed at malware detection.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

1. Liu K, Xu S, Xu G, Zhang M, Sun D, Liu H. A review of android malware detection approaches based on machine learning. IEEE Access. 2020;8:124579-124607.
2. Yasin HN, Hamid SHA, Yusof RJR, Hamzah M. An empirical analysis of test input generation tools for android apps through a sequence of events. Symmetry. 2020;12:1894.
3. Sadeeq MA, Zeebaree SR, Qashi R, Ahmed SH, Jacksi K. Internet of things security: A survey. In International Conference on Advanced Science and Engineering (ICOASE). 2018;162-166.
4. Li J, Sun L, Yan Q, Li Z, Srisa-An W, Ye H. Significant permission identification for machine-learning-based android malware detection. IEEE Transactions on Industrial Informatics. 2018;14:3216-3225.
5. Tan DJ, Chua TW, Thing VL. Securing android: a survey, taxonomy, and challenges. ACM Computing Surveys (CSUR). 2015;47:1-45.
6. Sadeeq MJ, Zeebaree SR. Semantic search engine optimisation (sseo) for dynamic websites: A review. International Journal of Science and Business. 2021;5:148-158.
7. Lopes J, Serrão C, Nunes L, Almeida A, Oliveira J. Overview of machine learning methods for Android malware identification. In *2019 7th* International Symposium on Digital Forensics and Security (ISDFS). 2019:1-6.
8. Choudhary M, Kishore B. HAAMD: hybrid analysis for Android malware detection. In International Conference on Computer Communication and Informatics (ICCCI). 2018:1-4.
9. Salih AA, Zeebaree SR, Abdulraheem AS, Zebari RR, Sadeeq MA, Ahmed OM. evolution of mobile wireless communication to 5g revolution. Technology Reports of Kansai University. 2020;62:2139-2151.
10. Alzakholi O, Shukur H, Zebari R, Abas S, Sadeeq M. Comparison among cloud

technologies and cloud performance. Journal of Applied Science and Technology Trends. 2020;1:40-47.
11. Qamar A, Karim A, Chang V. Mobile malware attacks: Review, taxonomy & future directions. Future Generation Computer Systems. 2019;97:887-909.
12. Abdulazeez AM, Zeebaree SR, Sadeeq MA. Design and implementation of electronic student affairs system. Academic Journal of Nawroz University. 2018;7:66-73.
13. Samann FEF, Zeebaree SR, Askar S. IoT provisioning QoS based on cloud and fog computing. Journal of Applied Science and Technology Trends. 2021;2:29-40.
14. Ageed ZS, Ibrahim RK, Sadeeq MA. Unified ontology implementation of cloud computing for distributed systems. Current Journal of Applied Science and Technology. 2020;82-97.
15. Poeplau S, Fratantonio Y, Bianchi A, Kruegel C, Vigna G. Execute this! analyzing unsafe and malicious dynamic code loading in android applications. in NDSS. 2014:23-26.
16. Ageed Z, Mahmood MR, Sadeeq M, Abdulrazzaq MB, Dino H. Cloud computing resources impacts on heavy-load parallel processing approaches. IOSR Journal of Computer Engineering (IOSR-JCE). 2020;22:30-41.
17. Sallow AB, Sadeeq M, Zebari RR, Abdulrazzaq MB, Mahmood MR, Shukur HM, et al. An investigation for mobile malware behavioral and detection techniques based on android platform. IOSR Journal of Computer Engineering (IOSR-JCE). 22;14-20.
18. Shukur H, Zeebaree SR, Ahmed AJ, Zebari RR, Ahmed O, Tahir BSA. et al. A state of art survey for concurrent computation and clustering of parallel computing for distributed systems. Journal of Applied Science and Technology Trends. 2020;1:148-154.
19. Sadeeq M, Abdulla AI, Abdulraheem AS, Ageed ZS. Impact of electronic commerce on enterprise business. Technol. Rep. Kansai Univ. 2020;62:2365-2378.
20. Abdulraheem AS, Salih AA, Abdulla AI, Sadeeq MA, Salim NO, Abdullah H, et al. Home automation system based on IoT; 2020.
21. Abdullah SMSA, Ameen SYA, Sadeeq MAM, Zeebaree S. Multimodal Emotion Recognition using Deep Learning. Journal of Applied Science and Technology Trends. 2021;2:52-58.
22. Abdulrahman LM, Zeebaree SR, Kak SF, Sadeeq MA, Adel AZ, Salim BW. et al. A State of Art for Smart Gateways Issues and Modification. Asian Journal of Research in Computer Science. 2021;1-13.
23. Sadeeq MAM, Zeebaree S. Energy management for internet of things via distributed systems. Journal of Applied Science and Technology Trends. 2021;2:59-71.
24. Ibrahim IM. Task Scheduling Algorithms in Cloud Computing: A Review. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 2021;12:1041-1053.
25. HamaAli KW, Zeebaree SR. Resources allocation for distributed systems: A review. International Journal of Science and Business. 2021;5:76-88.
26. Ageed ZS, Zeebaree SR, Sadeeq MM, Kak SF, Yahia HS, Mahmood MR, et al. Comprehensive survey of big data mining approaches in cloud systems. Qubahan Academic Journal. 2021;1:29-38.
27. Mohammed SM, Jacksi K, Zeebaree SR. A state-of-the-art survey on semantic similarity for document clustering using GloVe and density-based algorithms. Indonesian Journal of Electrical Engineering and Computer Science. 2021;22:552-562.
28. Sulaiman MA, Sadeeq M, Abdulraheem AS, Abdulla AI. Analyzation study for gamification examination fields. Technol. Rep. Kansai Univ. 2020;62:2319-2328.
29. Shen YC, Chien R, Hung SH. Toward efficient dynamic analysis and testing for Android malware. IT CoNvergence PRActice (INPRA). 2014;2:14-23.
30. Arshad S, Shah MA, Wahid A, Mehmood A, Song H, Yu H. Samadroid: a novel 3-level hybrid malware detection model for android operating system. IEEE Access. 2018;6:4321-4339.
31. Abdulraheem AS, Zeebaree SR, Abdulazeez AM. Design and implementation of electronic human resource management system for duhok polytechnic university.
32. Maulud DH, Zeebaree SR, Jacksi K, Sadeeq MAM, Sharif KH. State of art for semantic analysis of natural language processing. Qubahan Academic Journal. 2021;1.

33. Hasan DA, Hussan BK, Zeebaree SR, Ahmed DM, Kareem OS, Sadeeq MA. The Impact of Test Case Generation Methods on the Software Performance: A Review. International Journal of Science and Business. 2021;5:33-44.
34. Pan Y, Ge X, Fang C, Fan Y. A systematic literature review of android malware detection using static analysis. IEEE Access. 2020;8:116363-116379.
35. Sallow AB, Zeebaree SR, Zebari RR, Mahmood MR, Abdulrazzaq MB, Sadeeq MA. Vaccine tracker/sms reminder system: Design and implementation.
36. Zeebaree SSR, Mohammed AS. Social media networks security threats, risks and recommendation: A case study in the kurdistan region. International Journal of Innovation. Creativity and Change. 2020;13:349-365.
37. Saleem SI, Zeebaree S, Zeebaree DQ, Abdulazeez AM. Building smart cities applications based on iot technologies: A review. Technology Reports of Kansai University. 2020;62:1083-1092.
38. Husain BH, Zeebaree SR. Improvised distributions framework of hadoop: A review. International Journal of Science and Business. 2021;5:31-41.
39. Yazdeen AA, Zeebaree SR, Sadeeq MM, Kak SF, Ahmed OM, Zebari RR. FPGA implementations for data encryption and decryption via concurrent and parallel computation: A review. Qubahan Academic Journal. 2021;1:8-16.
40. Abdulla AI, Abdulraheem AS, Salih AA, Sadeeq MA, Ahmed AJ, Ferzor BM, et al. Internet of Things and Smart Home Security. Technol. Rep. Kansai Univ. 2020;62:2465-2476.
41. Zhou W, Zhou Y, Jiang X, Ning P. Detecting repackaged smartphone applications in third-party android marketplaces. In Proceedings of the second ACM conference on Data and Application Security and Privacy. 2012;317-326.
42. Sadeeq MM, Abdulkareem NM, Zeebaree SR, Ahmed DM, Sami AS, Zebari RR. IoT and Cloud computing issues, challenges and opportunities: A review. Qubahan Academic Journal. 2021;1:1-7.
43. Rashid ZN, Zeebaree SR, Sengur A. Novel remote parallel processing code-breaker system via cloud computing.
44. Zhou Y, Jiang X. Dissecting android malware: Characterization and evolution. In 2012 IEEE symposium on security and privacy. 2012;95-109.
45. Palumbo P, Sayfullina L, Komashinskiy D, Eirola E, Karhunen J. A pragmatic android malware detection procedure. Computers & Security. 2017;70:689-701.
46. Alam S, Qu Z, Riley R, Chen Y, Rastogi V. DroidNative: Automating and optimizing detection of Android native code malware variants. Computers & Security. 2017;65:230-246.
47. Shankar VG, Somani G, Gaur MS, Laxmi V, Conti M. AndroTaint: An efficient android malware detection framework using dynamic taint analysis. In ISEA Asia security and privacy (ISEASP). 2017:1-13.
48. Li D, Wang Z, Li L, Wang Z, Wang Y, Xue Y. FgDetector: fine-grained android malware detection. In IEEE Second International Conference on Data Science in Cyberspace (DSC). 2017;311-318.
49. Singh G, Bansal D, Sofat S. A smartphone based technique to monitor driving behavior using DTW and crowdsensing. Pervasive and Mobile Computing. 2017;40:56-70.
50. Ghaffari F, Abadi M, Tajoddin A. AMD-EC: Anomaly-based android malware detection using ensemble classifiers. In Iranian Conference on Electrical Engineering (ICEE). 2017:2247-2252.
51. Tong F, Yan Z. A hybrid approach of mobile malware detection in Android. Journal of Parallel and Distributed computing. 2017;103:22-31.
52. Liang H, Song Y, Xiao D. An end-to-end model for android malware detection. In IEEE International Conference on Intelligence and Security Informatics (ISI). 2017:140-142.
53. Feizollah A, Anuar NB, Salleh R, Suarez-Tangil G, Furnell S. Androdialysis: Analysis of android intent effectiveness in malware detection. Computers & Security. 2017;65:121-134.
54. Mathur A, Podila LM, Kulkarni K, Niyaz Q, Javaid AY. NATICUSdroid: A malware detection framework for Android using native and custom permissions. Journal of Information Security and Applications. 2021;58:102696.
55. Gao H, Cheng S, Zhang W. GDroid: Android malware detection and classification with graph convolutional network. Computers & Security. 2021;102264.

56. Wang X, Li C. Android malware detection through machine learning on kernel task structures. Neurocomputing. 2021;435:126-150.
57. Şahin DÖ, Kural OE, Akleylek S, Kılıç E. A novel permission-based Android malware detection system using feature selection based on linear regression. Neural Computing and Applications. 2021;1-16.
58. Cai L, Li Y, Xiong Z., JOWMDroid: Android malware detection based on feature weighting with joint optimization of weight-mapping and classifier parameters. Computers & Security. 2021;100:102086.
59. Shen F, Del Vecchio J, Mohaisen A, Ko SY, Ziarek L. Android malware detection using complex-flows. IEEE Transactions on Mobile Computing. 2018;18:1231-1245.

Peer-review history:
The peer review history for this paper can be accessed here:
http://www.sdiarticle4.com/review-history/68044