A Survey on Android Ransomware and its Detection Methods

Usama Desai[1]

[1]*Student at Department of CE & IT, VJTI, Mumbai, India*

---***---

Abstract - *Ransomware attacks have been on a rise for a long time. Ransomware possesses a serious security threat to cyber security. Now-a-days mobile phones have become a necessity. People don't use mobile phones just for communication purpose but also to store personal files and many other different things. 85% of the mobile that are being used run on Android operating system. Apart from personal computers attackers have also targeted android smart phones. Ransomware extorts money from victim in order to regain the control by the user. Ransomware locks the entire system and encrypts user data. Due to the rapid growth seen in ransomware there is a need to develop effective solutions. Many studies have been carried out to detect ransomware on Android systems but they still remain inefficient as ransomware evolves. Ransomwares are detected using various approaches such as static, dynamic and hybrid approach which is combination of both static and dynamic approach.*

Key Words: Android, Ransomware, Static Analysis, Dynamic Analysis, Hybrid Analysis.

1. INTRODUCTION

Ransomware is one of the security buzzwords for the last few years. Apart from being a buzzword, ransomware possesses a serious threat to cyber security. Ransomware is a form of malware that locks your system or makes personal files inaccessible. In order to regain access ransomware demands ransom payments. Android based ransomware attacks are on the rise. Android accounts for around 85% of the global mobile OS market share [16]. Recent android ransomware attack was DoubleLocker, a ransomware that locks the victim's phone by changing the device pin and encrypts all the data stored on the device. This makes it nearly impossible for the victim to access their data without paying the ransom.

Android ransomware are disguised as legitimate android applications. Attackers lure user in downloading and installing ransomware applications by presenting themselves as hacking software, applications to boost your system performance, paid applications for free, etc. Global damages caused by ransomware attacks are predicted to reach $11.5 billion annually by 2019 [17]. Android allows user to install third party applications from sources other than Google Play Store without verifying them. This gives advantage to attackers to trick users to install malicious applications. Ransomware are classified into two forms such as:

- Locker ransomware
- Crypto ransomware

In Locker ransomware it locks the entire system and demands for payment in order to unlock the system. Crypto ransomware encrypts the personal files and informs the user that the data is encrypted and will not be decrypted unless the ransom is paid. Generally ransomware is detected using static and dynamic approaches. Static analysis is done in non-runtime environment.

In static approach we reverse engineer the android application and look for malicious codes and resources by analysing files retrieved after reverse engineering. Dynamic analysis is done during the runtime. In dynamic approach we try to monitor the behaviour of the application and find pattern similar to the malicious applications. Static approaches are less computationally intense than dynamic methods and that they do not need applications to be executed for identifying malware [12], but they are typically ineffective with obfuscated code as well as with run-time infections.

On the other hand, dynamic methods are effective in identifying new threats, outperforming static methods, but they need applications to be run to identify malicious behaviour, potentially infecting the device [10]. In addition dynamic methods are able to discriminate malware even when its code is obfuscated [11]. The ability to analyze and evaluate a suspected ransomware application by both static and dynamic approach is becoming increasingly important.

1.1 Android Ransomware

Mobile ransomware was introduced with the popular CryptoLocker and other similar families in 2014. From then ransomware was seen on a large scale on mobile devices. Mobile ransomware typically displays a message on screen that the device has been locked due to some type of illegal activity. The message that is displayed states that the phone will be unlocked after a ransom is paid. Mobile ransomware is often delivered via malicious apps, and requires that you boot the phone up in safe mode and uninstall the infected app in order to regain access to your mobile device. Some mobile ransomware encrypt the personal files on the smart phone and demand for ransom in order to decrypt those files. So booting your smart phone into safe mode won't work here.[19]

1.2 Ransomware Infection

Attackers use phishing techniques and other social engineering approach to trick users into downloading the Android ransomware onto their smart phone. A phishing mail is sent disguised as a legitimate mail from your co-worker, boss or some relative. The mail contains a malicious link that is redirected to ransomware download site that installs the application to user's smart phone automatically. Ransomware are also installed on user's smart phone through the adware banner on social networking websites. Attackers often imitate the ransomware malware as a useful application. The attacker lures the user by imitating ransomware as paid applications for free. Some ransomwares imitate themselves as hacking applications such as Wi-Fi hacker or password cracker, social media account hacking applications, tracking someone anonymously application, etc. The user carelessly downloads the ransomware malware considering it as a safe app. In order to function ransomware app requests various permissions. The user allows the ransomware application to access the required permission giving it the necessary permissions to exploit the smart phone. Once installed, ransomware locks down the device or encrypts personal files making the smart phone inaccessible to the user.

1.3 Android Ransomware Feature

1) Screen Locking: Locking the screen is the most commonly found technique that ransomware uses to extort the user. This type of ransomware is referred to as Locker Ransomware or Locking ransomware. Ransomware lock the smart phone screen by gaining administrative privileges. Some ransomware create a full screen Activity overlaying all other Activities. The full screen overlay is just a black background so that the device appears as if it was locked or switched off. There are some ransomwares that leverage the built-in Android PIN screen locking mechanism. It is able to set its own PIN on the device, or even change it if it was already set. It is able to change the PIN if the victim has granted the malicious app Device Administrator privileges.

2) File Encryption: In recent days the ransomware authors try to demand ransom from users by encrypting their personal files. This type of ransomware is referred to as crypto ransomware. To encrypt the user's personal files the attacker can use standard cryptosystem's or customized cryptosystem's. The customized cryptosystems are not reliable compared to the standard cryptosystem's provided by the android platform. The main reason for developing customized cryptosystem's by attacker is to decrease the probability of being detected by common malware analysis. Some ransomwares displays a ransom message and encrypts files in a separate program thread in the background. It scans the SD card for files with any of the following image, document or video extensions – JPEG, JPG, PNG, BMP, GIF, PDF, DOC, DOCX, TXT, AVI, MKV, 3GP, MP4 and then encrypts them using standard or customized cryptosystem. The encryption keys used are hardcoded inside the binary as plain text, so it is difficult to decode them. Modern ransomwares do not hardcode the encryption keys in the binaries as it is decodable by reverse engineering the apk file. Modern ransomware acquire the encryption keys by communicating with C&C (Command & Control) server. C&C server sends the encryption key once the information is exchanged.

3) C&C server communication: After a successful installation, most Android ransomware reports to a Command & Control (C&C) server. Modern ransomwares mainly use C&C server to get the encryption keys. In some cases, the reporting serves are only to track the infection, sending back basic device information such as the device model, IMEI number, device language, and so on. Alternatively, if a permanent C&C communication channel is established, the ransomware can listen to and execute commands sent by the ransomware operator. This creates a network of infected Android devices under the attacker's control. Some examples of commands that are supported by Android ransomware, outside its primary scope of locking the device and displaying a ransom message, include [18]:

- Open an arbitrary URL in the phone's browser
- Send an SMS message to any or all contacts
- Lock or unlock the device
- Access received SMS messages
- Access contacts
- Display a different ransom message
- Update to a new version
- Enable or disable mobile data
- Enable or disable Wi-Fi
- Track user's GPS location

4) Privileged permissions: In comparison with other types of malwares, ransomware has its own unique characteristics and traits. Taking into consideration the behaviour of malware, it attacks the user by unusual behaviour such as stealing private data, intensive data usage, decreasing performance, unwanted SMS charges. Most of these behaviours are uncommon to benign applications. In contradiction the ransomware shows behaviour that is very similar to that of benign applications such as opening popup or toast or file encryption. INTERNET, READ_PHONE_STATE and ACCESS_NETWORK_STATE are the common permissions that are widely requested by ransomwares and malwares. Because these permissions are necessary in order to communicate with the server. Ransomware tends to requests device related permissions such as WALK_LOCK,

DISABLE_KEYGUARD, SYSTEM_ALERT_WINDOW and RECEIVE_BOOT_COMPLETE, GET_TASK, KILL_BACKGROUND_PROCESSES. Ransomware presents the threatening text as soon as the smart phone restarts. Activity hijacking is main task to lock the device.

5) Ransom Payment: Electronic payment methods are used to pay the ransom. Most of the ransomware attackers demand ransom payment through crypto currencies such as Bitcoin. As crypto currencies are untraceable it is difficult to find the origin and final destinations of payments.

6) Threat Message: Most malware try to be sneaky, ransomware requires showing its presence for its malicious behaviours. Therefore, ransomware needs to display some kind of text to the victim, which is used to accuse him and then specify the payment information. A police themed ransomware intelligently presents its ransom demands as official looking warning messages from a local police. The most common allegations include retention of pornographic content; distribute copyrighted materials and possession of other illegal content.

2. LITERATURE SURVEY

In this section we will discuss about the techniques that are being currently used to detect Android Ransomwares. Various methods and techniques have been developed to detect android ransomware; all are based on three approaches:

- Static
- Dynamic
- Hybrid

2.1 Static Approach

Static approach is based on the non-executable code which is generated at the compile time. Static analysis is nothing but analyzing software before executing whether it is malicious or not. [13] In Android every program is composed in an apk file, which contains all the program's source code, resources, assets, certificates, and manifest file. These files are analyzed in static approach.

The approach proposed by Kanwal and Thakur [7] is based on three analysis i.e. permission analysis, text analysis and code analysis. Applications that were installed from Google play store were tagged as safe applications. Before doing the permission, code and text analysis the extraction process is done. First step in extraction process is to extract the dex file from apk. After that jar files are extracted which contain the class files and at last the java files are extracted from class files. When the apk file is decompiled manifest files are also obtained which contain the list of permissions required by the application. Based on the permission application can be tagged as vulnerable or non-vulnerable. After analyzing permissions text and code analysis is done.

In text analysis keywords such as Ransom, safety reasons, locked, action performed, money, accusation, law, etc. are searched. A custom file reader is used to open the java files. After that the text is converted into an array list of sentences and then keywords are searched in the sentences. On the occurrence of the keyword the score of the keyword is increased. On the basis of the keyword occurrence application is declared to be vulnerable or safe.

In code analysis they looked into the code to find if the application is trying to encrypt the user data or it can block the user from accessing other applications. Method such as onBackPressed() and onPause are looked up. If the onBackPressed() activity is left empty or the onPause() that means the app is stopping user from pausing the activity or killing the activity. Google firebase is used to store the information about the malicious and benign application.

Karimi and Moattar [2] proposed an approach in which they used reduced opcode sequence and image similarity to detect android ransomware. LDA algorithm is used for both feature selection and classification. At first the apk file is disassembled using androguard tool and after that opcode sequence is extracted. Opcodes are Dalvik bytecodes that are generated during compile time. Opcode sequence of length 2 is created for each apk. An image is created for each sample using opcode sequence with respect to the probability function. Dalvik has 256 opcodes which leads to creation of an image of 256 x 256 pixels for each sample. Feature selection is done to select the best opcodes that increases the accuracy. LDA is used for feature selection. Features are selected based on the ratio of the total within-class variability and between-class variance. Opcode sequences that are selected in feature selection phase are used to create a new image matrix. This lead to creation of decreased size images compare to previous one. In classification phase each image is converted into a vector so that each item represents the value of each pixel. As the images have the same size therefore lengths of the vectors are same too. LDA algorithm is used for classification. To classify by LDA the Scikit-learn library was used in python language.

Kanwal et al. [3] did an addition to their previous approach which can analyze the image for detecting keywords that are present in the image. To extract text from image Tesseract is use, which is an open source OCR (Optical Character Recognition) library. Pre-processing, character recognition and post-processing is done by Tesseract. Most of the ransomware show threatening message through images. So image analysis is also necessary.

Andronio et al. [15] proposed an approach in which three independent detectors are executed in parallel to detect ransomware. Only the static approach is discussed here. The three independent detectors are Threatening Text Detector, Encryption Detector and Locking Detector. Threatening Text Detector uses text classification to detect coercion attempts. If the result of only this classifier is positive that means the sample is Scareware. If encryption and/or locking detector are triggered that means that the application is actively performing either action on the infected device. In this case the sample is labeled as Ransomware.

Threatening Text Detector analysis consists of text extraction, text classification, localization and other sources of text. In text extraction static strings are extracted and analyzed by parsing the disassembled code and resource file. Text Classification uses a natural language processing (NLP) supervised classifier to estimate whether a string contains threatening sentences. Classifiers are trained using phrases labeled as threat, law, copyright, porn, and money, which typically appear in Scareware or ransomware threat message. NLP classifier supports localization transparently. It tells whether a given sentence is “threatening” in any of the languages on which it has been trained on. Text can be displayed by other means than strings i.e. through images. OCR is used to extract the text from images or videos.

In Encryption Detector we check whether the (disassembled) code of the sample under analysis contains traces of unsolicited file-encryption operations. getExternalStorageDirectory() and CipherOutputStream, delete() functions are used for unsolicited file-encryption process.

Locking Detector checks if the application under analysis is able to lock the device. Android ransomware require administration privileges to lock the device. It calls DevicePolicyManager lockNow() which forces the device to act as if the lock screen timeout expired. It starts from searching for the specific permission (BIND_DEVICE_ADMIN) in the manifest. If found, the Smali assembler code is parsed from the application until a call to the lockNow method is found.

2.2 Dynamic Approach

Dynamic methods are based on features that can only be obtained at runtime of the application and that represents' the behavior of applications.

Chen et al. [4] proposed a system called RansomProber which detects ransomware based on three analyses that are 1) Encryption Analysis, 2) Foreground Analysis and 3) Layout Analysis. RansomProber is implemented on top of Android Security Modules (ASM). Encryption analysis module is used by the RansomProber to detect whether any files have been encrypted. The RansomProber detects whether the encryption process belongs to the app that user is interacting with using Foreground analysis module. Layout analysis module analyses UI widgets of related activities and operation coordinates of the user.

In encryption analysis they predefined some directories that need to be protected i.e. /Android/data, /data/system/accounts.db, /data/.../contacts2.db, /data/.../mmsms.db, /sdcard/Pictures and /sdcard/Downloads. Instead of hooking APIs or system calls in predefined directories, RansomProber depends on the information entropy to measure the degree of data transformation. Encrypted files look like random information while a non-encrypted file looks like well structured information. Thus, the entropy of non-encrypted files is lower than that of encrypted files.

Once RansomProber detects that a file has been encrypted it determines whether the encryption behavior was result of user actions or abnormal. The purpose of the foreground is to make sure the whether the encryption process is triggered by the foreground application. Foreground analysis is done based on two system provided components i.e. widget and activity, both of them are needed for drawing graphical elements. In Android, activities are organized in a stack that is managed by the service called ActivityManager. The activity on top of the stack is shown to users, called top activity, and the corresponding app is called foreground application. Foreground analysis looks for system provided components in the foreground application while the application is executing the encryption process. If the foreground application is irrelevant to the encryption process then the application may be malicious.

The main intuition of layout analysis is that the ransomware hides itself in order to not display the encryption process to the user. Three UI indicators are commonly present in the activities during the file encryption i.e. File List, Hint Text and Button. These UI indicators are selected because they appear in the interface of a benign application when user encrypts the file voluntarily. None of the ransomware showed all these UI indicators while encryption process. During the file encryption, RansomProber records the user's click coordinates and related activities' layout information continually. If there exists no click operation during the encryption process, it is directly inferred that the operation is without users' intention. If there exists user's click behavior, UI widgets are further analyzed.

Song et al. [6] proposed a technique which is designed with three modules: Configuration, Monitoring, and Processing. The configuration module is the basic setup which is to be applied when the proposed technique detects a ransomware.

The role of configuration module is to specify the location of the files which need to be protected against ransomware. These areas of important files are called priority protection area (PPA). Information of PPA is collected and is registered to the watch list table for the monitoring module, and protects the corresponding files in real time. User's handling for the suspected process which is detected by the monitoring module are registered into the database and maintain the handling. Based on the user's feedback the process is automatically detected and deleted if the user determines the process as ransomware.

By monitoring the PPA area and the process the monitoring module detects the ransomware. The monitoring module is consists of two modules (file monitoring and process monitoring) based on the roles. In File Monitoring Module the status of the input/output events is continuously monitored such as reading, writing, copying, and deleting of a file belonging to a PPA which is set in the configuration module and detects the attacks of the ransomware. In Process Monitoring Module Processor share by Process, Memory usage, I/O count, Storage I/O count are continuously monitored to detect the ransomware.

In processing module the processes which are suspicious of being ransomware are forcibly stopped in the monitoring module and interrupts users about the appropriate handling of the process. Once the handling is determined, the information of the corresponding process is stored in the database and used in the configuration module subsequently. Through Android permission analysis the processing module warns users about the risk of the ransomware.

2.3 Hybrid Approach

The main objective behind using a hybrid approach is to have the advantages of both static and dynamic methods while reducing their disadvantages.

Ferrante et al. [1] proposed a hybrid method to detect android ransomware which includes both static and dynamic method. The static approach is based on the frequency of opcodes and the dynamic approach is based on the monitoring of memory, CPU, network and statistics on system calls. Hybrid approach is the combination of both static and dynamic approach. First static detection method is used when applications are installed. Applications that identified as malware are denied permission to run on device. All other applications are allowed to run for dynamic detection. In this way the malicious applications that are not detected by static method are detected by the dynamic method.

In static analysis the application is pre-processed in order to obtain the numeric values of frequencies of op-code sequences that are suitable to be processed by the classifier. After pre-processing, the classifier undergoes the learning phase in which it is trained by using a labeled dataset. After the learning phase, the classifier is used for the actual classification of the applications as ransomware or trusted.

In order to perform dynamic detection of ransomware, an effective method based on the observation of system behavior is used. Execution traces were obtained by running applications on android emulator and features were extracted from execution traces. Seven memory and CPU related features with addition to network usage and statistics on system calls are used in order to perform on-device detection at runtime, and that is based on a two-steps detection system. Similarly to static detection, the development of the dynamic detection method undergoes the two phases of pre-processing and learning, with the classification phase used at runtime to actually detect malware. Machine learning algorithms used were Naive Bayes, Decision tree (J48) and Logistic Regression. Experiments were done on a dataset containing 3,058 mobile applications from which 2,386 were Android trusted applications downloaded from the Google Play Store and 672 applications containing ransomware taken from the freely available HelDroid dataset.

Gharib and Ghorbani proposed DNA-Droid [16], a real-time hybrid detection framework. The DNA-Droid quickly assesses a sample using static analysis and if it is labeled suspicious, it will continuously monitor and profile the run-time behavior of the sample. Three major components of the architecture of the proposed framework are static analysis module, dynamic analysis module and detection module. The static module includes three sub-components for evaluating the apk file and decides whether it is benign, malicious or ransomware. The three sub-components are Text Classification Module (TCM), Image Classification Module and API calls and permissions Module (APM). In TCM the disassembled APK is parsed to extract the strings. TCM removes meaningless words/stop words (e.g., to, the, or) to clean the strings and remaining words (e.g., locking and locked are replaced with lock) are then lemmatized. Five scores which indicate the presence of each category in APK contents are based on the Cosine similarity. For example, for an APK the output of the TCM module would be {0.1, 0.9, 0.2, 0.1, 0.3} which shows that the APK content is 0.1 close to encrypt, 0.9 close to lock, 0.2 close to money, 0.1 close to porn and 0.3 close to threat.

ICM compares application images with this collection of logos such as banks, police, government, and famous brands using the Structural Similarity Index Measure algorithm (SSIM) and reports the number of detected images as a feature.

List of permissions are extracted from the AndroidManifest.xml file by the APM and by decompiling an APK, we obtain a list of API methods. There are large number of Android APIs and permissions, so the APM considers only APIs and permissions with the highest information gain between malware and benign apps.

In the dynamic analysis module the malware behavior are observed and its properties are analyzed by the execution of sample in a simulated environment. To differentiate benign and malicious samples the proposed system defines the dynamic behavior as an API call sequence. In dynamic analysis, samples should go through the following components to generate the DNAs i.e. Sandbox, pre-processing, and Multiple Sequence Alignment (MSA).

Sandbox component captures run-time behavior and produce API call sequences. The pre-processing component is responsible for refining the API call sequences to reduce noise and therefore increase accuracy. MSA helps in detecting malicious behavior which is caused by injecting malicious code in popular benign applications. Static analysis implementation of the DNA-Droid is done using shell and Python scripts. It uses Apktool to decompress and decode APKs and Natural Language Toolkit (NLTK) is used to extract linguistic features. Machine learning tasks including preprocessing, dimensionality reduction, training and testing phases are done through Scikit-learn and tensorflow libraries. Dynamic analysis implementation consists of a modified emulator, an Android application to hook API calls (written in Java), and python scripts to control the emulator and apply MSA and BSA techniques. Naive-Bayes,SVM, RF, AdaBoost (AB), and Deep Neural Networks (DNN) classifiers are used.

3. CONCLUSION

In this paper various methods for detecting ransomware are shown i.e. static, dynamic and hybrid method. Static methods show high accuracy in detecting ransomware compared to dynamic analysis. But they are ineffective with obfuscated code as well as with run-time infections. Hybrid method overcomes the shortcomings of both the static and dynamic approach. There is a little research work done on detecting and preventing mobile ransomware attacks. There is a need to develop methods to evaluate a ransomware application by both static and dynamic approach. We plan to propose a framework based on dynamic approach to detect ransomware by analyzing the communication made by the ransomware application with its C&C server and with ransom payment gateways i.e. crypto-currency gateways.

REFERENCES

[1] Ferrante A., Malek M., Martinelli F., Mercaldo F., Milosevic J. (2018) “Extinguishing Ransomware - A Hybrid Approach to Android Ransomware Detection”. In: Imine A., Fernandez J., Marion JY., Logrippo L., Garcia-Alfaro J. (eds) Foundations and Practice of Security. FPS 2017. Lecture Notes in Computer Science, vol 10723. Springer, Cham

[2] Alireza Karimi, Mohammad Hosein Moattar (2017), “Android Ransomware Detection Using Reduced Opcode Sequence And Image Similarity”, 7th International Conference on Computer and Knowledge Engineering, IEEE.

[3] Meet Kanwal ; Sanjeev Thakur ; Rishabh Lashkari (2017), “An App Based On Static Analysis For Android Ransomware”, 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), IEEE.

[4] Jing Chen, Chiheng Wang, Ziming Zhao, Kai Chen, Ruiying Du, and Gail-Joon Ahn (2017), “Uncovering the Face of Android Ransomware Characterization and Real-time Detection”, IEEE Transactions on Information Forensics and Security.

[5] Monika, Pavol, Zavarsky, Dale, Lindskog (2016), “Experimental Analysis Of Ransomware On Windows And Android Platforms: Evolution And Characterization”, Procedia Computer Science, Volume 94, Pages 465-472, Elsevier.

[6] Sanggeun Song, Bongjoon Kim, and Sangjun Lee, “The Effective Ransomware Prevention Technique Using Process Monitoring on Android Platform,” Mobile Information Systems, vol. 2016, Article ID 2946735, 9 pages, 2016.

[7] Meet Kanwal ; Sanjeev Thakur (2017), "An app based on static analysis for android ransomware", International Conference on Computing, Communication and Automation (ICCCA), IEEE

[8] Alexander Adamov ; Anders Carlsson (2017), "The state of ransomware. Trends and mitigation techniques", East-West Design & Test Symposium (EWDTS), IEEE

[9] Masarah Paquet-Clouston, Bernhard Haslhofer, Benoit Dupont (2018), "Ransomware Payments in the Bitcoin Ecosystem", 17th Annual Workshop on the Economics of Information Security (WEIS), arXiv:1804.04080

[10] Martinelli, F., Mercaldo, F., Saracino, A. (2017) ” Bridemaid: An hybrid tool for accurate detection of android malware”. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security. pp. 899{901. ACM

[11] Martinelli, F., Mercaldo, F., Saracino, A., Visaggio, C.A. (2016): “I find your behavior disturbing: Static and dynamic app behavioral analysis for detection of android malware”. In: Privacy, Security and Trust (PST), 2016 14th Annual Conference on. pp. 129{136. IEEE

[12] Rastogi, V., Chen, Y., Jiang, X. (2013): "Droidchameleon: evaluating android anti-malware against transformation attacks" In: Proceedings of the 8th ACM SIGSAC symposium on Information, computer and communications security. pp. 329{334. ACM

[13] P Ravi Kiran Varma, Kotari Prudvi Raj, K. V. Subba Raju (2017), "Android mobile security by detecting and classification of malware based on permissions using machine learning algorithms", International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), IEEE.

[14] Andronio N., Zanero S., Maggi F. (2015) HelDroid: Dissecting and Detecting Mobile Ransomware. In: Bos

H., Monrose F., Blanc G. (eds) Research in Attacks, Intrusions, and Defenses. RAID 2015. Lecture Notes in Computer Science, vol 9404. Springer, Cham

[15] Gharib A., Ghorbani A. (2017) DNA-Droid: A Real-Time Android Ransomware Detection Framework. In: Yan Z., Molva R., Mazurczyk W., Kantola R. (eds) Network and System Security. NSS 2017. Lecture Notes in Computer Science, vol 10394. Springer, Cham

[16] https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

[17] https://cybersecurityventures.com/ransomware-damage-report-2017-part-2/

[18] https://www.welivesecurity.com/wpcontent/uploads/2016/02/Rise_of_Android_Ransomware.pdf

[19] https://www.malwarebytes.com/ransomware/

[20] https://www.malwarebytes.com/malware/