Indonesian Journal of Electrical Engineering and Computer Science
Vol. 20, No. 1, October 2020, pp. 353~360
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v20.i1.pp 353-360

A computational forensic framework for detection of hidden applications on Android

Tahira Rasul[1], Rabia Latif[2], Nor Shahida Mohd Jamail[3]
[1]Department of Information Security, College of Signals, National University of Sciences and Technology (NUST), Pakistan
[2,3]College of Computer and Information Sciences, Prince Sultan University, Saudi Arabia

Article Info

Article history:

Received Jan 22, 2020
Revised Mar 27, 2020
Accepted Apr 11, 2020

Keywords:

Android
Data hiding
Forensic investigation
Physical and logical
Rooted and non-rooted

ABSTRACT

Smartphones, since their emergence has become a significant part of our lives and Android is popular of all. They are successful due to the increasing availability of user applications to answer every possible need, so it is of great importance to ensure security and privacy when handling personal and sensitive information of the user. To secure the data on mobile devices, users use applications available on the Google Play store, which help to hide data on their devices known as Hidden Applications. Hidden applications are categorized as one of the major applications used for data hiding and storing. These applications can be used to hide date from snooping, intrusion and against the data theft. Therefore, the proposed framework in this research helps to find either they store and hide data in efficient manner or not and if they do so either it is encrypted or not. In this paper, main objective is to identify the privacy threats which end users face by using such applications, analyse these application's behaviour, working, their code to understand how data is hidden and if the information is encrypted, it can be retrieved or not. The work not only focuses on the identification of hidden data/apps; it also provides a mechanism to recover and reconstruct the data from these hidden parts of the memory. In the end, present the results obtained by using the proposed framework in a case file so that it can be used in a criminal court case.

Corresponding Author:

Tahira Rasul,
Department of Information Security,
National University of Sciences and Technology (NUST), Islamabad, Pakistan
Email: tahirarasul.msis14@students.mcs.edu.pk

1. INTRODUCTION

Smartphones, being repositories for photos, messages, e-commerce, and social existence, are required to store the desired information securely [1]. There are several applications which provide security and safety to the user, among which hidden applications are commonly used [2]. They are readily available on Play Store and provide the purpose of hiding storage data at different locations either online or other folders / offline from snooping eyes and intruders and to conceal the valuable data from being spied. Hidden applications are prone to vulnerabilities, frauds and criminal activities and claim to offer a higher level of security than the already available on an android [3]. Hidden Application in this research are referred to as those applications which can hide images, videos, media files and other documents as well. Sometimes these applications also hide other applications. These hidden applications come in variety of forms, few of them hide their icon in application tray while few come disguised as another form at front end and when some specific password is typed original application is opened for example clock, volume fixer, calculator etc.

In past, there had been cases when criminal activities were performed by the use of smartphones. Upon the investigation by the law enforcement agencies, manufacturers refused to provide the information and resultantly there was no way to get the hidden information. Therefore, it is progressively vital for law enforcement agencies to remain vigilant regarding the latest developments and know how data is concealed

[4]. Google Play Store [5] is a significant source of applications for Android OS based Smartphones. Hidden applications are selected on the following criteria: (a) Hides User Data (b) Disguised as other applications (c) Strong password setting (d) Number of downloads (e) User's reviews, (f) Customer support and (g) "Data Hide" keyword. In case of criminal activity, these applications prove as a hindrance for forensic investigations as they hide storage files, relevant documents, phone calls and messages. For example, in 2015, high school students exchanged questionable images among them and hid those images using a calculator like hidden applications. In an investigation, it was found that such applications exist and raised the concern how to recover data [6] as these applications use extra permissions to access other memory areas of device [7] and investigators were failed to find out the actual data which caused the concern. One of widely trusted and used paid forensic tool is Cellebrite [8], it can detect all the installed applications. Even though Cellebrite provides many features but it is not able to detect any suspicious application behaviour which can be used to hide information [9].

The main objective of this research is to identify hidden applications and their respective behaviour that how they hide the data [10], which will provide a foundation for proposing a computational forensics framework. The study aims to analyse artefacts in these applications as they offer evidentiary data in the form of photos, messages, contact lists etc. The focus is to propose a remedial framework to enhance the security of an Android device, to check the loopholes where these applications miss-use the information or present the data as it is, and to help the investigators in future to get the information from an android device immediately if it is suspected.

Already a lot of work has been reviewed in three significant areas: types of android applications, selection of hidden applications, and Android forensic techniques. Several studies have been carried out in the domain of the steganography, forensic analysis of android and retrieval of sensitive information from images. Thus contributing towards the detection of hidden information, content in steganography and general android forensics discussing the techniques of how to carry out the device forensics. Several forensic tools including Androphsy [11], Droidwatch [12] and Amandroid [13] have also been proposed earlier that cover the android investigation procedure and give guidance to investigators, but no specific framework related to the investigation of hidden applications and data retrieval from these data hiding applications has been proposed.

The contribution of this paper include: (1) Acquisition of hidden applications, comprehensive forensic analysis and compilation of results, (2) Forensic analysis of android devices with/ without hidden applications and (3) Proposal of a computational forensic framework for detection of hidden applications on an android. Rest of the paper is organized as follows: Section 2 discusses literature review. Section 3 research method. Section 4 discusses results and discussions and Section 5 discusses conclusion.

2. LITERATURE REVIEW

2.1. Types of android applications

There are three types of Android Applications. Table 1 below shows Android OS applications comparison [14]. Table 1 shows that native applications are high on development cost, performance and user interface is better as compared to hybrid and third party applications which are fair/low in performance.

Table 1. Android operating system (os) applications comparison

	Native Applications	Hybrid Applications	Third Party / Web Applications
Development Cost	High	Low	Low
Performance	High (Data on device)	Low (Data on Web Server)	Low (Data on Web Server)
User Interface	Better	Fair	Fair

2.2. Android forensic techniques

Primarily there are four significant methods to do forensics of an Android device [15]:

a) *Acquiring Physical Image:* Physical Image is a bit-by-bit copy of the Android device. Data residing on a device also on the unallocated space and the deleted data files are all copied through this method.
b) *Acquiring Logical Image*: There are a number of tools available for this method but logical image cannot recover the data from the deleted/ unallocated data space. Backup of device in known as logical image [16].
c) *Imaging Memory Card:* Memory card is removed safely, and data is copied and examined.
d) *Manual Method:* When an Android device is brought for the forensics investigation, screenshot of every action performed on the device is taken. This is time-consuming and also not accurate.

2.3. Apparatus setting and test scenario creation

Table 2 shows the android devices that are used in this research along with their versions and root options.

Table 2. Apparatus for forensic investigation

	Android Device	Version	Root Option
1	Samsung Grand Prime	Lollipop	Rooted
2	Huawei Mate 10	Oreo	Non-Rooted

Samsung Grand Prime and Huawei Mate 10 were used to hide the data and perform forensics. Selected Hidden Applications are installed and images are hidden using these applications. For Rooted devices, image is taken, and for non-rooted backup is taken. Both image and backup are taken twice as before and after the installation of hidden applications to examine.

2.4. Analysis methods

Mentioned below is the list of analysis carried out for this research purpose [17]:

a) *Configuration Analysis* – On Android Device
b) *Static Analysis* – Reverse Engineered the Application to observe/analyze the code
c) *Rooted Android Device* – Physical Acquisition / Image File
d) *Non-Rooted Device* – Logical Acquisition / Backup File
e) *File System Analysis* – Android device internal file examination

Hidden Applications for this research purpose are analyzed, both dynamically and statically [18]. For dynamic analysis, applications are installed and analyzed on an android device. For static analysis, APK of application has been converted to source code and code has been analyzed extensively [19].

2.5. Analysis of hidden applications

This section explains the dynamic analysis, static analysis, rooted android device analysis, non-rooted android device analysis and android file system analysis carried out in this research [20].

Dynamic Analysis: Applications were thoroughly examined after installation and login settings. The major features that are analyzed include: available in recent applications list or not, Easy to use, Vault camera available, Password protected, Password recovery option available, Fingerprint available, Pattern available, Facedown closes the app, Create new folders and text files, Private Cloud available, Loss of data on deleting and uninstalling the app and Administrative rights are required to uninstall the application.

Static Analysis: After performing reverse engineering, code obtained from the APK has been investigated to find out the critical functions on which these applications work. Code obtained was obfuscated as well, but mostly all the applications had "hide function" which upon investigation found out that it contained the information in plain letters for example user password and login details. Hide functions, locations (where data is stored) and login functions are retrieved in the research.

Rooted Android Device Analysis: An Android device which has root access to the Android OS is known as "Rooted Device".

a) **Image Creation:** To examine the rooted device for research, ADB (Android Debug Bridge) is used from Android Studio to take the "Image" of the android's memory [21]. Further, Autopsy is used to read the image obtained from the device.
b) Forensic Investigation analysis without Hidden Applications Installed.
c) Forensic Investigation with analysis Hidden Applications Installed.

Non-Rooted Android Device Analysis: All the available android devices comes with certain restrictions from manufacturers, they are non-rooted.

a) **Backup Creation:** ADB (Android Debug Bridge) is used in a command terminal to access the device using ADB commands and take the backup. After that, Android Backup Extractor is used to read the .ab file obtained from backup and extract it to readable format obtained in the .zip folder [22].
b) Forensic Investigation without Hidden Applications Installed.
c) Forensic Investigation with Hidden Applications Installed.

Android File System Analysis: After analysing android rooted and non-rooted devices, another aspect of investigating and detecting hidden application is observing Android File Structure [23]. In previous sections, we have seen that few apps add **".hide"** or **".hidden"** after the file extension to hide the data, few of them move the files from internal device storage to their folder by changing name and few properly encrypt the data and hide [24].

File structure can also be used to observe an android device. Several hidden applications use files structure to manipulate the data files and place it at the same location. 10/18 applications were accessed using file structure analysis. Therefore, there is a chance of getting accurate data by only observing internal storage.

Comparison of Image and Backup File: Table 3 shows the comparison of results found from image and backup file analysis carried out in research.

Table 3. Analysis of image and backup file

Android Device		Size	Total Applications
Rooted Device (Physical Acquisition / Image)	*Without Hidden Applications*	6.50 GB of 8 GB	24
	With Hidden Applications	7.28 GB of 8 GB	42 (18 detected)
Non-Rooted Device (Logical Acquisition / Backup)	*Without Hidden Applications*	470 MB of 64 GB	34
	With Hidden Applications	609 MB of 64 GB	44 (10 detected)

In the table above, results for rooted device shows that this case study has helped in a way that no hidden application is missed in the physical acquisition of android device.

Results for non-rooted devices shows *that ten out of eighteen hidden applications installed are shown in the backup file*. Therefore for forensic investigators, this case study of observing backup file of non-rooted Android devices do not show the complete applications installed from third party sources. So, while examining the non-rooted android device, internal storage must be checked bit by bit as well to find out clues to hidden data in android device.

3. RESEARCH METHOD

This section discusses the proposed framework architecture, methodology, framework step-by-step guide and a computational forensic framework for detection of hidden applications pseudocode based on the analysis carried out in methodology section. Figure 1 shows the architecture of the proposed computational forensic framework for the detection of hidden applications in this research:

Figure 1 of framework architecture above perform the following functions to gather the information from android device and find out the desired information:

a) Forensic investigator gets the android device for analysis.
b) For safety purpose the device has been put in Farady cage.
c) Investigator kills all the processes running on the device to stop network communication or external interference.
d) Backup copies are created so that data cannot be compromised.
e) One of the copy is brought to the workstation for analysis.
f) Backup files are also available on cloud if collaboration is needed with other investigators.
g) Results are gathered, stored in database and reports are generated.

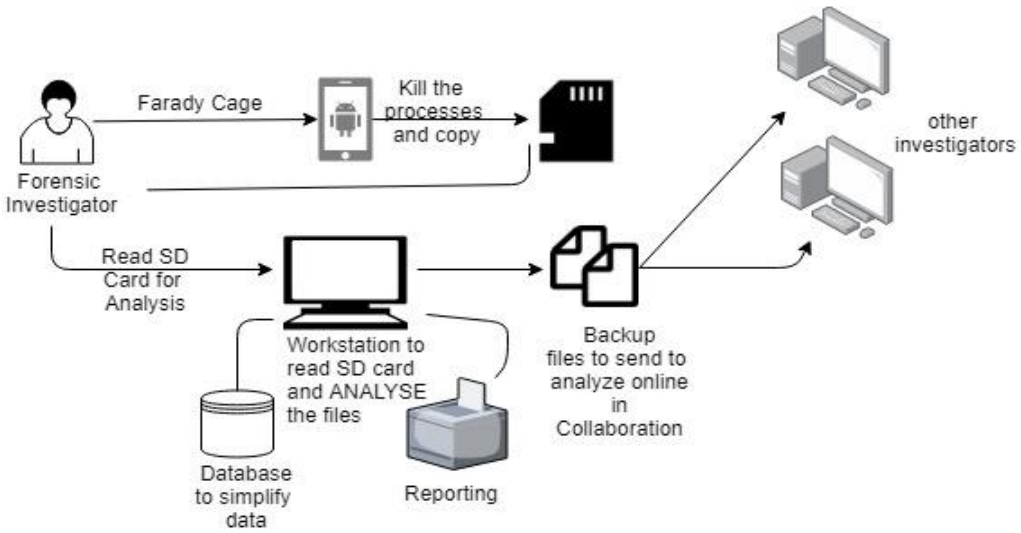

Figure 1. Framework architecture

Framework architecture defined clearly in framework methodology. It includes android device acquisition, data acquisition and data analysis:

a) Android Device Acquisition: The analysis includes checking the *device's state* either it is off or on, if on then either locked or not, connected to the network or not and SD card slot available or not. After that *power of the device*. As the examination might be power-intensive, therefore the device can be plugged. After that copying the data from the device's to the investigator's external storage so that backup is also created right after the acquisition of device. Perform further analysis on investigator's copy of data, and thus if any changes are made, original data is not compromised that is present on the internal device storage.
b) Data Acquisition: Data acquisition can be time taking, as android devices are coming in large storage space options, and if the device has SD card option as well, then it can take more time to copy data. First of all, if the device is locked, the forensic investigator is required to unlock it with any unlocking tool available with him, and after that, Android forensic techniques can be followed to acquire data.
c) Data Analysis: Data Analysis is divided into manual and technical analysis discussed below. Analyze the information that is acquired previously in the following steps:

Manual Data Analysis Procedure:

a) ***List down applications*** installed on the device by going to settings -> apps and notifications.
b) Check if any application has ***name or words including obvious hints*** to hiding data.
c) Check those applications which have ***separate password*** and which are ***present twice***.
d) Try to ***double tap the application's icon*** as it opens the original application.
e) ***Check if media files and documents*** are not located in device's original storage location.
f) Check if there is any ***folder*** related to media with same name.

Technical Data Analysis Procedure:

1) ***Connect*** Android device with forensic work station.
2) ***Allow*** "File Transfer" option.
3) ***Run search*** related to hint words so that any folder, file and application can be refined.
4) Check if any file type ***extension is manipulated*** as 2/18 applications add words after files extension and 2/18 remove the file extension and add hex value at the end of file name.
5) Check if media files and documents have ***"."*** at the start of their name.
6) ***List down*** all third party packages installed on device using command: "adb shell pm list packages -3"
7) Check if mobile device is ***rooted or not*** [25].
   a) If an Android device is ***not rooted***: Take Logical Acquisition of device by taking Backup, Analyse the Backup file, Compare the list of third party packages with the applications in backup file, Check "SP" folder in application, this is the location where these applications hide user login credentials, and finally, Check "Data" folder in application.
   b) If an Android device is ***rooted***: Take Physical Acquisition of device by taking Image Analyse the Image file, Compare the list of third party packages with the applications in Image file, All the similar applications must be analysed, Check "Shared_Prefs" folder in application, this is the location where these apps hide user login credentials and finally, Check "Data" folder in application.
8) If after all these, no results are found and no hidden application is detected then applications must be reversed engineered to get the source code.
9) Observe the source code ***"main activity"*** class and get the hint to main functionality the application performs. From there, hidden data can be found. As location to the path where hidden data is stored is mentioned, and password reset functions can also be found.

The framework guide will provide overall steps of the proposed forensic framework and how methodology is incorporated:

Device acquisition from the spot.

1) Shield the device with Farady Cage.
2) Kill all the processes running in the device.
3) Copying of data from internal storage to external (SD card or hard drive).
4) Forensic investigator's workstation with external data card reader (SD card reader or hard drive reader).
5) Data analysis ***(following the data analysis mentioned before)***
6) If hidden files exist then store data in separate directory.
7) When hidden data is stored, create database to store data in respective tables with defined hidden applications attributes, else simple store the data to analyse.
8) Online backup of analysis to compute results.
9) Open collaboration with online forensic investigators to carry out analysis in parallel.
10) Result based reporting.

Proposed pseudo-code for the framework is discussed below. Pseudocode explains the acquisition of device, copying of data and investigation with online collaboration with multiple investigators and finally explains reporting and data storage as retrieved. It involves the killing of processes so that no external communication can be made with the device.

Acquire device from spot
Shield the device with Farady Cage
If exists alive process **then**
Kill all process
end if
Copying of data from internal storage to external
Prepare forensic workstation to work with external data reader
Data Analysis
If hidden data exist **then**
Store data in separate directory
Else
Take image of device data
End if
If hidden data stored **then**
Create database
Store data in respective tables with defined hidden applications attributes
Else
Simply store the data to analyse
Online backup of analysis to compute the results
If working in team **then**
Open collaboration with online forensic investigators to carry out analysis in parallel
End if
Result based reporting.
Else
Analyse Data to find clue
End if

4. RESULTS AND DISCUSSIONS

Performance of chosen hidden applications is analyzed from the proposed framework and summary is gathered in Table 4. Most of the applications serve the purpose of hiding data, but they are also prone to weak programming, as data and login credentials can easily be found. Table 4 shows the framework evaluation and analysis results.

Table 4. Evaluation of results

S.No	Application	Code Analysis	Image (Photo)	Video	Password	Phone State
1	Phone Dialer	M	U	U	C	R , NR
2	File Hider	N	U	U	C	R , NR
3	File Hide Expert	N	E	E	E	R , NR
4	Gallery Vault	O	U, Ek	U, Ek	E	R , NR
5	Folder Hider Expert	-	E	E	C	R , NR
6	Vault	O , N	U	U	E	R , NR
7	Vaulty	-	E	E	E	R
8	Hide it Pro	M	U	U	C	R , NR
9	Clock	N	U	U	C	R , NR
10	Calculator	O	U	U	C	R
11	Timer	N	U	U	C	R
12	Calculator	O, N	U	U	C	R , NR
13	Calculator	N	U	U	E	R, NR
14	Keepsafe	O , N	U	U	C	R
15	Calculator	N	U	U	C	R , NR
16	TimeLock	M	U	U	C	R
17	Apps Lock and Gallery Hider	M	E	E	C	R
18	zCalculator	-	U	U	C	R
	Result	O = 5/18 M = 3/18 N = 7/18	U = 14/18 E = 4/18 Ek = 1/18	U=14/18 E= 4/18 Ek=1/18	C = 13/18 E = 5/18	R = 7/18 R, NR= 11 /18

O = Code was obfuscated
U = Unencrypted File
Ek = Encryption Key available
C = Clear File
M = Main Function has critical data and pointers to other functions
N = Functions in Native Library
E = Encrypted File
NR = Non-Rooted, R = Rooted

Code analysis and Table 4 shows that 5/18 applications had obfuscated data, 3/18 had the main function having critical data, and 7/18 had normal native library functions. Image analysis and video analysis with respect to the proposed framework shows that 14/18 applications stored data unencrypted, 4/18 apps stored data using encryption and 1/18 application had encryption key available. Password analysis shows that 13/18 applications saved the password in clear text and 3/18 saved password in an encrypted format. Phone state analysis showed that during all the investigation and finding 7/18 applications were observed and retrieved data in a rooted format and 11/18 retrieved data in both rooted and non-rooted thus showing that clear images and data can be retrieved from such applications.

5. CONCLUSION

The hidden applications serve the purpose of hiding data accurately and also save information from misuse in case of theft and data leakage. Beside major features are available in pro versions, the basic version of all the applications provide hiding media accurately by encrypting the data. Thus, they prove to be a hindrance in forensic investigation. In this research, by the comparison of rooted and unrooted devices and forensic analysis, it has been seen that rooted devices offers more clues. If any device is subjected to be forensically investigated in case of criminal activities or involved in trying to find out the hidden data; the forensic framework presented in this research, can cover all the aspects to detect the hidden applications. Even it helps to get the clue of an application from a single word, data hidden using such application and user login credentials as well. Forensic investigators can acquire and analyse the device easily using this framework to understand the clues necessary for court hearings.

ACKNOWLEDGEMENTS

This work was partially supported by the Artificial Intelligence Data Analytics Lab (AIDA), Prince Sultan University, Riyadh, Saudi Arabia.

REFERENCES

[1] R. R. S. N. H. A. Nor Afifah Shafin, "Implementation of persuasive design principles in mobile application development: a qualitative study," *Indonesian Journal of Electrical Engineering and Computer Science,* Vol. 18, no 3, pp. 1464-1473, June 2020.

[2] U. K. Michaila Duncan, "Detection and Recovery of Anti-Forensic (VAULT) Applications on Android Devices," Annual *ADFSL Conference on Digital Forensics, Security and Law, 6*, 2018.

[3] W. Y. H. L. M. S. a. S. A. S. Azadegan, "Novel Anti-forensics Approaches for Smart Phones," *2012 45th Hawaii International Conference on System Sciences, Maui, HI,* nº 10.1109/HICSS.2012.452, pp. 5424-5431, 2012.

[4] M. A. S. S. A. L. A. H. M. Tehseen Mehraj, "A critical insight into the identity authentication systems on smartphones," *Indonesian Journal of Electrical Engineering and Computer Science,* Vol.13, no 3, pp. 982-989, 2019.

[5] Y. Zhou, Z. Wang, W. Zhou, and X. Jiang, "Hey, You, Get Off of My Market: Detecting Malicious Apps in Official and Alternative Android Markets", *In Proceedings of the 19th Annual Symposium on Network and Distributed System Security (NDSS 2012),* Feb. 2012.

[6] B. F. B. Xiaolu Zhang, "Breaking into the vault: privacy, security and forensic analysis of android vault applications," *Computers & Security (2017),* http://dx.doi.org/doi: 10.1016/j.cose.2017.07.011

[7] S. M. S. a. S. H. Howida Abubaker, "Exploring permissions in android applications using ensemble-based extra tree feature selection," *Indonesian Journal of Electrical Engineering and Computer Science,* Vol.19, no 1, pp. 548-557, 2019.

[8] W. B. G. Karl-Johan Karlsson, "Android Anti-forensics: Modifying CyanogenMod," *2014 47th Hawaii International Conference on System Science,* Vol.10, 1109, p. 593, 2014.

[9] T. B. Tajuddin and A. A. Manaf, "Forensic investigation and analysis on digital evidence discovery through physical acquisition on smartphone," *2015 World Congress on Internet Security (WorldCIS),* Dublin, 2015, pp. 132-138.

[10] S. P. Mylam Chinnappan Babu, "Protecting sensitive information utilizing an efficient association representative rule concealing algorithm for imbalance dataset," *Indonesian Journal of Electrical Engineering and Computer Science,* Vol.15, No 1, pp. 517-524, 2019.

[11] B. P. a. A. A. I. U. Akarawita, "ANDROPHSY - forensic framework for Android," *2015 Fifteenth International Conference on Advances in ICT for Emerging Regions (ICTer), Colombo,* 2015.

[12] J. Grover, "Android forensics: Automated data collection and reporting from a mobile device," *Digital Investigation,* pp. S12-S20, 2013.

[13] S. R. X. O. a. R. Fengguo Wei, "Amandroid: A Precise and General Inter-component Data Flow Analysis Framework for Security Vetting of Android Apps," *ICC (inter-component communication), Android app security analysis, Static analysis,* vol. 14, pp. 2471-2566, 2018.

[14] G. D. H. D. S. S. H. Anirban Sarkar, "Android Application Development: A Brief Overview of Android Platforms and Evolution of Security Systems," *2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, pp. 73-79, 2019.
[15] K. K. L. A. Nihar Ranjan Roy, "Android Phone Forensic: Tools and Techniques," International *Conference on Computing, Communication and Automation (ICCCA2016)*, 2016.
[16] Hoog, *Android Forensics: Investigation, Analysis, and Mobile Security for Google Android*. Syngress, 2011.
[17] V. P. S. J. Uma Narayanan, "A novel approach to big data analysis using deep belief network for the detection of android malware," *Indonesian Journal of Electrical Engineering and Computer Science,* Vol.16, No 3, pp. 1447 - 1454, 2019.
[18] F. M. A. N.S. Selamat, "Comparison of malware detection techniques using machine learning algorithm," *Indonesian Journal of Electrical Engineering and Computer Science,* Vol.16, No 1, pp. 435-440, 2019.
[19] R. T. H. M. Satish Bommisetty, em *Practical Mobile Forensics,* 2014.
[20] N.-T. C. S. K. a. S. J. Long Nguyen-Vu, "Android Rooting: An Arms Race between Evasion and Detection," *Wiley, Security and Communication Networks,* vol. 2017, p. 13, 2017.
[21] K. S. M. S. G. C. K. C. Soudamini Patil, "Data Extraction Techniques for Android Based Devices," *International Journal of Computer Science Trends and Technology (IJCST),* Vol. 5, Issue 2, Mar – Apr 2017.
[22] N. E. A. B. Yun Shen, "Insights into rooted and non-rooted Android mobile devices with behavior analytics," *In Proceedings of the 31st Annual ACM Symposium on Applied Computing (SAC '16),* NY,USA, 2016.
[23] M. A. Darren Quick, "Forensic analysis of the android file system YAFFS2," *Australian Digital Forensics Conference,* 1-1-2011.
[24] T. X.-h.,. W. J. CHANG Xu, "Forensic research on data recovery of android smartphone," *Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE 2013),* 2013.
[25] D. R. V. D. D. V. M. T. Rizwan Ahmed, "Efficient Generalized Forensics Framework for extraction and documentation of evidence from mobile devices," *International Journal Of Enhanced Research In Management And Computer Applications,* vol. 2, no 1, 2013.

BIOGRAPHIES OF AUTHORS

Tahira Rasul has received her BS - Information Technology from School of Electrical Engineering and Computer Science – NUST Islamabad, Pakistan in 2013. In 2015, she pursued her MS - Information Security National University of Sciences and Technology - Islamabad, Pakistan. Her research interests include Wireless networks, Databases and Android architecture. She is currently associated with Department of CS & IT, Women University of Bagh. AJK, Pakistan.

Dr. Rabia Latif received her MS in Information Security (2010) and PhD in Information Security (2016) from National University of Sciences and Technology, Pakistan. She is currently working as Assistant Professor in Prince Sultan University, Riyadh, Saudi Arabia. Her Research interest includes Cloud Computing Security, Web Security, Cyber Security and Network Security. Her professional career consists of activities including Conference Chair, Technical Program Committee Member and reviewer for several international journals and conferences. She is an active member of Artificial Intelligence Data Analytics Research Lab at Prince Sultan University, Riyadh.

Dr. Nor Shahida Mohd Jamail is currently in Prince Sultan University, Riyadh, Saudi Arabia as an Assistant Professor. She obtained her PHD in Software Engineering from Universiti Putra Malaysia. Her specialized are purely in Software Engineering, Software Process Modelling, Software Testing and Cloud Computing Services. She had involved in Machine Learning Research Group in Prince Sultan University and also involved in research project which collaborated with National and International University. Email: njamail@psu.edu.sa