ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

# Risk Analysis of Android Apps and Fake ID Flaw Solution Model

**Shristi Pandey[1], Ravi Kant[2], Dr.Vijendra Singh[3]**

Dept. of Master of Technology in Software Engg., Indian Institute of Information Technology, Allahabad (U.P), India[1]

Department of Master of Science in Cyber Laws & Information Security, Indian Institute of Information Technology, Allahabad (U.P), India[2]

**Abstract**: In this era of modern technology where everyone is rely on the latest technical specifications specially concerning with the latest mobiles and apps. Keeping this in mind the various software company's has launched their own Operating System for mobile, and in the same field Google also launched their own mobile Operating System Android which has its own technicality and specification which is competing with the other companies Operating System. After using the Android in mobile people came to know that some problems has been faced by the android user and the problem is not hardware related but in fact it is the major flaw of the Android OS which is consider as the fake ID problem. This flaw allow hacker to impersonate the trusted application and potentially hijack the users mobile phone and extract all the critical data from the mobile and the result would be catastrophic. Various solution have been proposed by the researchers around the globe to overcome this problem. Here I am providing a specific method to overcome this critical flaw of android Operating System.

**Keywords**: APK, Android Apps, Certifying Authority, MinSDK, MaxSDK, TargetSDK, X.509, PKI, IPsec, TLS, PKIX, ASN.1, Malware, Spyware, Risk ware, Trojan, Jelly Beans, Kit Kat, Defense In Depth.

**Abbreviations:** DVM (Delvik Virtual Machine), GPS (Global Positioning System), MIPS (Million Instruction per Second), CA (Certifying Authority), PKI (Public Key Infrastructure), PKIX (Public Key Infrastructure X.509)

## I. INTRODUCTION

People are now start using mobile device as one of the most useful gadgets to work their day to day work as compare to the PC which are bulky in nature and not much handy as compare to the mobile devices of now days which are slim and agronomic in nature[1]. Even now days people are spending more time on mobile as compare to the TV the reason is their compactness and handy by the shape and size. People now days uses apps very frequently and they even more rely on apps day by day for their day to day activities like shopping, socialization, banking etc. etc. For most of the people the mobile device is more than just a device they keep their all critical information in mobile like personal images, videos, passwords, employers details social security number and other critical things related with them. While mobile devices are facing many threats and the app stores and app developers Constitutes at greatest risk. It may be possible that the app we have downloaded and their ensuring action have the possible potential to expose all the critical information On the mobile device. The Malicious app can able to get your VPN credentials could do unauthorized access to the bank accounts and can also copy and resend your personal e-mails. Moreover the Adware could also extract your personal information, benign apps unauthorized coordinate you're GPS and monitor all the installed apps. Sometime the developers make mistakes and unwittingly write flawed code that leave the app susceptible to attackers. Most of the app stores working on this to manage the authenticity of the app and reject the malicious app but the fact is that the anonymous developer of malicious app always keep himself one step ahead and make the breach most of the time. Malicious app can be prevented by most of the app store to not to make available for all. The app user, developer, store provider and the organization all together understand the behavior and the criticality of the malicious app and the risk pertaining to the malicious mobile app.

## II.RELATED WORK

Various research work has already been carried out in the field of android security and hence to enhancing the security in android is a primary task so that we are able to protect our clients from malicious applications of android so in the continuation of this process we are also proposing a model which is basically help in to eliminating the flaws of fake id and provide an appropriate solution.

[11] Wook Shin, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka In this particular paper they used permission mechanism of the android system using state machine based approach.They proposed security theorem to assure the security of the android system.

In another research work [12] Wei Tang, Guang Jin, Jiaming He, Xianliang Jiang have done another research work which is based on SD rules of the android security enforcement. ASESD a lightweight application security authentication service.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

In the other research work [13] Welderufael Berhane Tesfay, Todd Booth, and Karl Andersson proposed a methodology which is based on a reputation based security model for android applications.

In other research paper of [14] Lin Sun, ShuTao Huang, YunWu Wang, MeiMei Huo proposed a policy-based sandbox code access security designed to apply special policy on application

In another research paper [15] ] Parvez Faruki, Ammar Bharmal, Vijay Laxmi, Vijay Ganmoor, Manoj Singh Gaur,Mauro Conti, *Senior Member*, *IEEE*, and Muttukrishnan Rajarajan proposed a concept that has led to the use of behavior, anomaly, and dynamic-analysis-based methods. Since a single approach may be ineffective against the advanced techniques, multiple complementary approaches can be used in tandem for effective malware detection.

## RISK ANALYSIS OF ANDROID APPS

Various types of works has already been done by various researchers in the field of Android Security for fake ID solution but in this related work we have tried to implement the Certifying Authority Based solution using the different approach in which we used a special risk analysis of app approach[8].

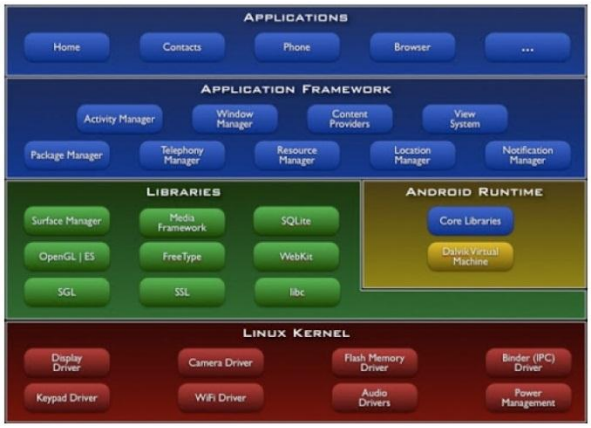



Fig.1 Android Architecture Framework

Android Architecture consist of following components:

1. Application: It is the top most layer of the model and it consist of the application installed by default and the user specified and installed applications.
2. Framework: It is the second layer consist of the Android systems key functions such as package manager which is used to enable installing/deleting of the Android Applications, Active Manager whose work is to controlling of the life cycle of every activity in each applications, etc.
3. Libraries: It use different components in the android system. It consist of the following libraries such as multimedia, graphic engines and SQLITE database engine.
4. Android Runtime: It consist of two components: Core Libraries and Dalvik Virtual Machine [10] or its successor, ART. Various Application are executed in the Virtual Machine so that the aim of providing the secure environment can be achieved in this model.
5. Kernel: Being the Last layer of the android architecture its main work is to work as an abstraction layer between the Hardware and Software. It also consist of some essential services like memory and process management, interaction with camera, Wi-Fi, audio etc.

## DEFENCE IN DEAPTH APPROACH

In order to provide more secure services google has proposed a more sophisticated security approach known as Defense in Depth. In this approach a layer by layer protection is being provided to the asset so that the unauthorized user is not able to penetrate the security in a single run [3].]The significance of using this approach is that when the attacker want to breach the security then by the mean time it reached to the core asset to exploit the asset the user get the enough time to get alarmed that something is going wrong and should be monitor properly.

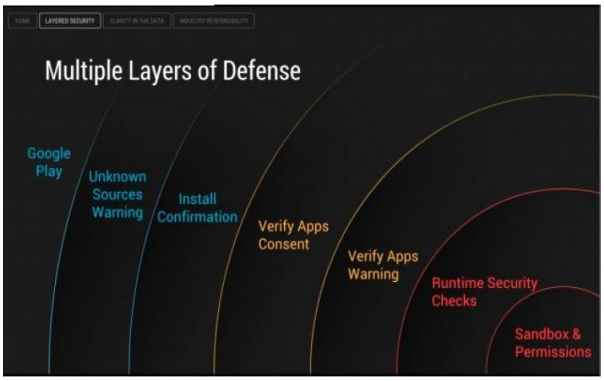



Fig.2 Android Defense In Depth Approach

In spite of providing the DiD Approach Android still have various loopholes. Various Loopholes are as follows

1. Apps are signed with self-signed certificate they do not have any Certifying Authority Third party to prove the authenticity of the genuine apps. It potentially increase the risk of being exploited.
2. Customized Permission may increase the risk of privacy risk.
3. There must be a need of more improvement in Bouncer‘s Security Controls
4. In Google Play Store user must be notified regarding certain changes if the application requested certain permission changes.

## APK FILE FORMAT

The Applications in the Android device are encapsulated in the specific file format known as APK(Application Package File) even malware also include the same file format in Android[4].For this platform the distribution and installation of the application is being done in the same file format.

Various Components of APK file format is as follows:

1. AndroidManifest.xml: It is mainly the application configuration file. It include different aspects such as unique identifier of application, its various components and the permission that application requires to work properly.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

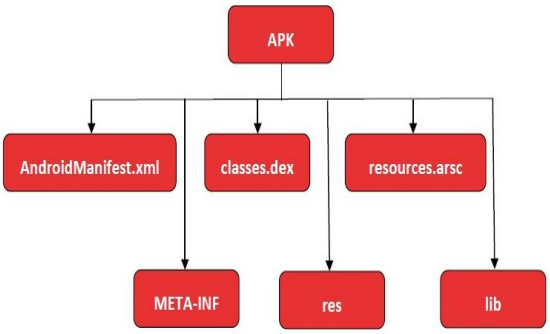



Fig.3 APK File Structure

2. classes.dex: It consist of compiled code of application in DEX format so that the Virtual Machine Dalvik or ART can able to interpret the code.
3.resources.arsc: It is a compiled resource of the corresponding file.
4. META-INF: It is the directory which stores the Digital Signature Information of corresponding application and contains the following files.
   MANIFEST.MF, CERT.SF, CERT.RSA
   MANIFEST.MF: It consist of complete list of APK's file along with its respective SHA-1 hash.
   CERT.SF: It consist of SHA-1 hash of every three lines that appear in the MANIFEST.MF.
   CERT.RSA: It stores the CERT.SF file signature,therefore consist of APK's signature.
5.res: It is basically a directory which stores the application used resources.(text, images, xml files etc.)
6. lib: This directory consist of the code in compiled form of various architectures.: x86 or mips, armeabi, armeabi-v7a.

## ANDROID VERSIONS

| Version | Codename | API | Distribution |
|---|---|---|---|
| 2.2 | Froyo | 8 | 0.4% |
| 2.3.3 - 2.3.7 | Gingerbread | 10 | 7.8% |
| 4.0.3 - 4.0.4 | Ice Cream Sandwich | 15 | 6.7% |
| 4.1.x | Jelly Bean | 16 | 19.2% |
| 4.2.x | | 17 | 20.3% |
| 4.3 | | 18 | 6.5% |
| 4.4 | KitKat | 19 | 39.1% |

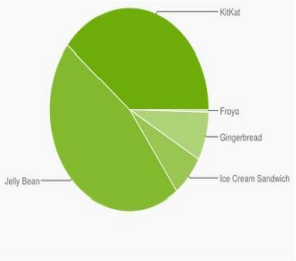



Fig.4 Distribution of Android Versions

Android Operating System is basically differentiated into various versions. After launching the google play service google tries to reduce this problem which enable upgrading of google and google play apps without undergoing the process of complete up gradation of the Operating System as a whole. The performance of the app is depend upon the version of the android operating system and is different for the different versions. That is why app in android has different versions.
MinSDK Version, MaxSDK version, TargetSDK version.

## TOP TEN FREQUENCTLY USED APP IN GOOGLE PLAY

We can measure the potential impact of malicious app have by calculating the how much amount of time these apps have been downloaded. Here in the graph we can see the top ten most frequent apps which have been downloaded from the google play and which contains the probability of having the malicious apps too.

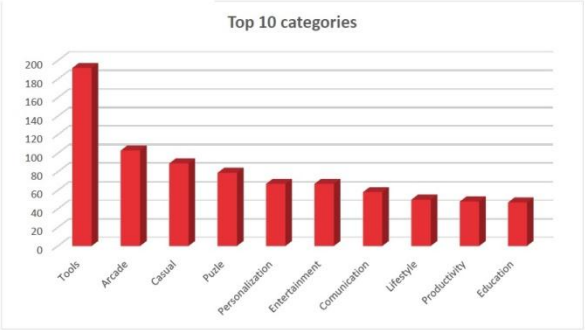



Fig. 5 Top ten categories of frequent used apps

## ANDROID PERMISSION MODEL

Android has its own permission model which help the user to grant the permission for wether the user want to install the particular app or not.Various danger level are there in the permission model of android depend upon the function they allow the app to perform and hence consequently have four protection level groups.Likewise it is possible to determine which app have access to the permission[2].

NORMAL, DANGEROUS, SIGNATURE, SIGNATURE/SYSTEM

NORMAL: Generally it does not represent a real threat to the device or the user and hence these permission are automatically accepted and when the installation is being performed they are not shown by default.Permissions within this category are viberate or allowing the state of network to be known.

DANGEROUS: They represent the real threats for the user or device such as accessing to the personal information or subscribing to the sms services and hence for the criticallity of this permission type the user is promt for the service before the installation or during the installation of the app and wether the service is accepted or not is subjected to the user concern. Some of the permissions are making calls, sms reception or sms submittion.

SIGNATURE: In this the permission is only granted to those apps which has been signed with the same certificate than the app that declears the permission.

SIGNATURE/SYSTEM: It is similar to the signature permission mode but can be used by the system also.

## THE MOST REQUESTED PERMISSION BY APP

Among the twenty most requested permission 15 percent is correspond to the normal protection level. 75 percent is to dangerous and 10 percent is to signature or system.And hence therefore after seeing the criticallity of the appps it is very ncessery to have focus on the permission that an application is being request when it is installed as it could be quite favaurable and signficant [5]. Due to the lack of awareness of majority of users the risk of being exploit is always been persist.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

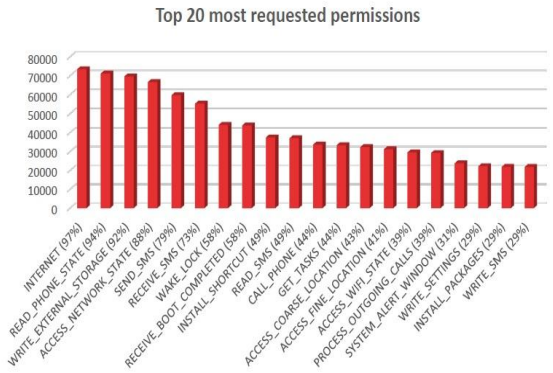



Fig. 6 Top 20 Most Requested Permissions

## ESTABLISHED CONNECTIONS BY MALACIOUS APPS

The world map shows here is the criticality of the android apps which has been constantly hit by the malicious apps and the most suffering country is the USA and China as shown in the map given below [9].

Fig.7 World map of the malicious app connection

The subsequent countries like Virgin Islands, Germany, Vietnam, Holland, Ireland etc. are the least affected as compare to the above mention countries.

## TYPES OF MALWARE

Malwares are of various forms like Trojan, Adware, Spyware, Risk ware, Monitor each malware has its own level of potential to exploit the weakness of the system. In the case of android the most dangerous malware is Trojan the percentage of affecting the android system is highest in the case of Trojan. After that Adware and Risk ware came then after that the Spyware and other malware came which affects the Android operating system.

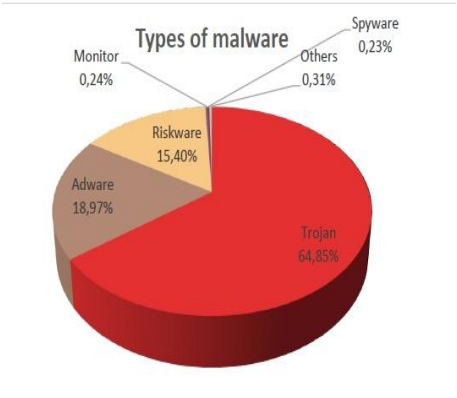



Fig.8 Types of Malware

In the graph given below shows the most active malware families which are prone to affect the android operating system mostly.

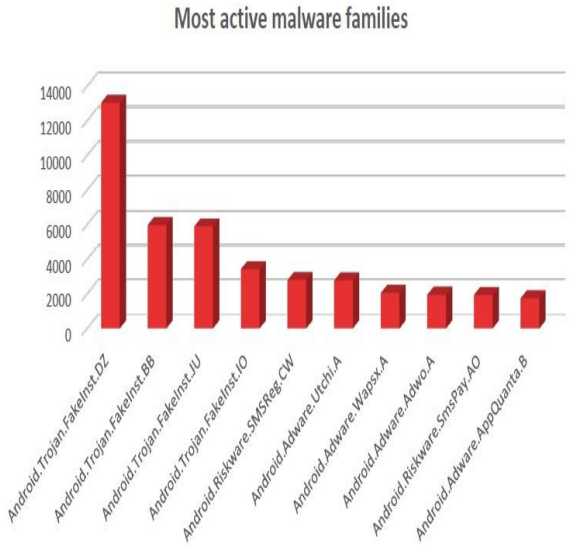



Fig.9 Active malware

POTENTIALLY DANGEROUS METHODS

Various APK methods have been analyzed and then compare with the list of potentially dangerous methods. So after comparing the APK with various methods we came to know that getId, getUrl, getPackageName and setpassword is considered as the most dangerous methods which means that these methods have the highest potential to affect the privacy of the users.

Other methods like getImage, getGender, getBirthday, getDeviceId, getPostalcode, and all these methods have potentially less impact as compare to the above mention methods. Most of the methods are related to the monetization libraries. To customized advertising on device these methods are used to obtain necessary information.

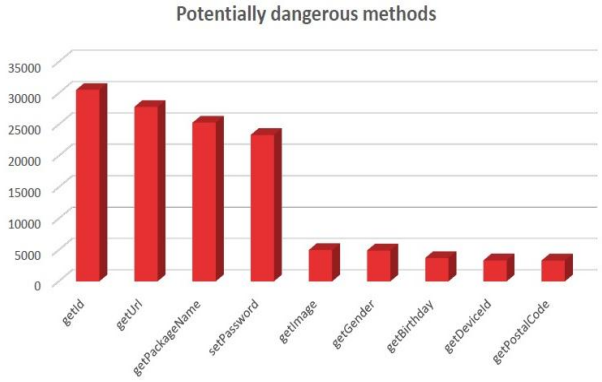



Fig.10 Potentially Dangerous Methods

## III. PROPOSED MODEL

In our proposed model I have tried to minimize the risk of the potentially harmful app used by the android operating system by providing the authentication between the legitimate app and the user. This authentication process is based upon the mechanism of cryptographic concept using X.509 Certifying Authority based model.

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

It uses a well-known protocol known as ASN.1 (Abstract Syntax Notation 1). In this model various apps which are available is being digitally signed by certifying authority and generate a certificate for the corresponding app and then this digitally signed app list is being published in open market for the users if some user want certain app then it just need to select the digitally signed app number and demand it by the app store and then that particular app is available for the user to use. Now during the use of the app if the user found that some malicious activity is being running out in their mobile then immediately it report to the certifying authority by sending them the malicious activity code and the digitally signed app number to the certifying authority and then the role of certifying authority takes place and it blacklist the particular app by the immediate effect and ready to take legal action against the developer of that particular app. So by using stern policy we are able to curb this situation. Fig.11 shows this.

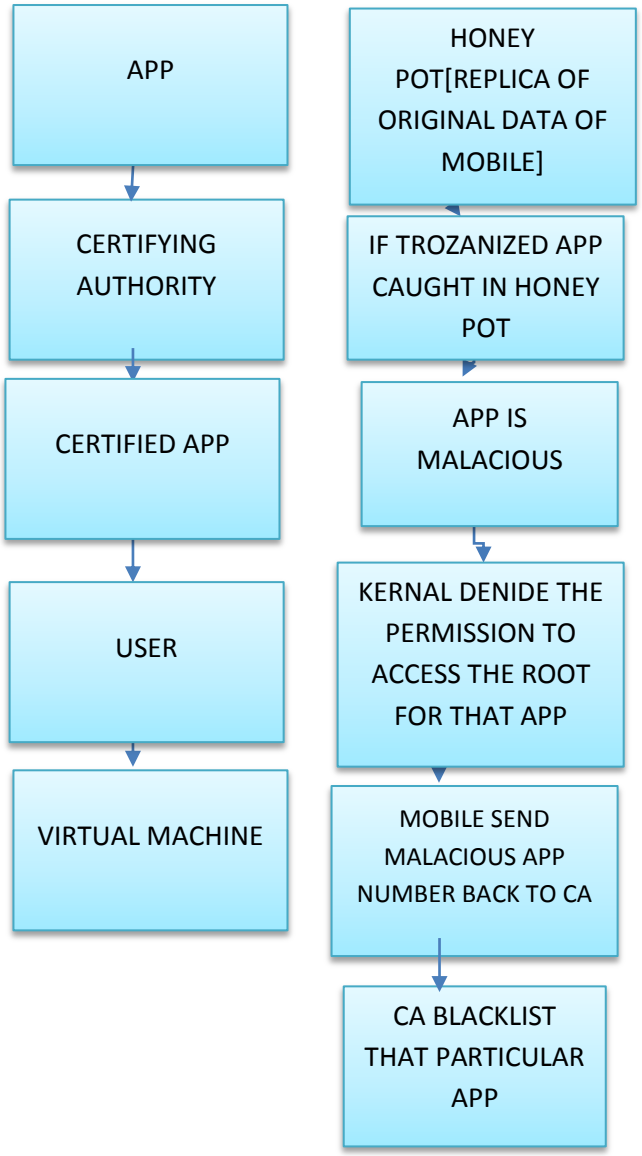



Fig.11. Block diagram of the proposed Model

In the above proposed model the block diagram depicts the flow of the processes. In this model initially the app has been certified by the certifying authority and then after that the certifying authority assigned a certificate number to the particular app and after that the user is free to use that app which has been populated by the certifying authority in public so that anyone can use this app by sending a request to the google play and as soon as the certified app is downloaded by the user through google play it is ready to installed on the client virtual machine. Here the Virtual machine plays a crucial role in this the certified app has been installed on this virtual machine and not on the real machine and this virtual machine contains honey pot, basically honey pot is the virtual representation of the critical data of the user information like account number, social security number etc. so once the app has been installed in this virtual machine and if after the installation of the app we monitor the activity of the app if the app is malicious and sows some malicious activity by attacking on the honey pot and after monitoring the level of disaster to the honey pot we can identify the criticality of the system and we come to know how dangerous is this application. Now if the malicious application has been trap in the honey pot then, we come to the conclusion that the application is malicious and hence there is a need to take the further steps and for that once it has been confirmed that the app is malicious then the kernel denied the permission to the particular app to access the root. And hence a mobile system sends a malicious app certificate number back to the certifying authority to take further step. As soon as the certifying authority received the details form any client about any application it start investigating and since the client has been facing some problem with that particular application or some malicious activity has been detected by the client for that particular application then the certifying authority has a responsibility to take action against the complaints of the client and for that the certifying authority as soon as received the details of that particular application the certifying authority blacklist that particular application from being distribution from the google play and hence in this way it quarantine the particular malicious application and hence prevent the other android client from being infected by the malicious application. The motive of provide this model is to prevent the android client from malicious application and this has been done by not installing the particular app in the real machine instead of that, that particular application has been installed on the virtual machine and this virtual machine has worked on the behalf of the real machine and when any malicious application has been pointed out then the corresponding preventive actions has been take place and if it has been detected that the app is malicious then the root permission has been denied and quarantine the particular application through certifying authority.

## TRUSTED MODEL FOR CERTIFYING AUTHORITY FOR ANDROID

For a Certifying Authority It is not possible for a single Certifying Authority to handle all the requests generated by all the app around the globe so it become very heavy for the single Certifying Authority so in order to overcome this problem a trusted model for the certifying authority is

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

being proposed particularly for the apps so that the load balancing can be done in a homogeneous manner and hence for this model is being proposed.

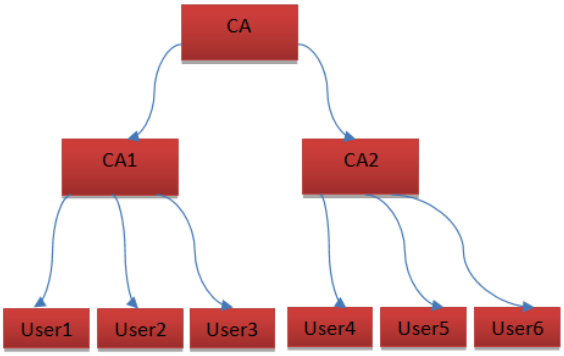



Fig.12 Certifying Authority Trusted Model

In this model we have taken a single certifying authority and demarcate it into the one two or n numbers of sub parts depend upon the load of the apps around the globe on the certifying authority and then each sub part of the certifying authority is ready to handle the request of the apps generated around the globe. For example here in this case the CA that is the certifying authority is further sub divided into the CA1 and CA2 such that the certifying authority 1 and certifying authority 2 and then further now each certifying authority is ready to handle the request for the user 1, user2, user3 or user n. The main purpose of this process is to reduce the load on the single Certifying Authority and hence the working of the CA is improved and the confidentiality Integrity and the availability of the Certifying Authority is also maintained. In the above figure it is also shown that that the Certifying Authority (root) has signed the certificates for CA1, CA2 and then the certifying authority1 signed certificates for user1, user2, and user3 and so on. Another Certifying Authority trusted model also named as the Mesh model in this model in this each root is connected to every other root. Mesh Model is used to root out the constrains of the previous model such that it can be serve for the large community which need several hierarchical structure connected together.

## PUBLIC KEY INFRASTRUCTURE (PKI) FOR ANDROID

The Public-Key Infrastructure (PKI) for android is a specific model for android system for the creating, revoking and distribution of certificates for the particular apps based on the X.509 .The main aim of this PKI android model is to key storage and updation for the particular app and also providing services to the other protocols based on the services used by the particular app, and also providing the access control to the system that is which app should provide permission and which not is decided by the particular access control mechanism .Various duties has been assign to the PKI so that the Confidentiality, Integrity and availability of the system can be intact. In this process X.509 defined some duties since PKIX is based on the X.509 and hence it need to manage all the duties related to the certificates.

Those who want to keep their private keys safe a PKI is the better place for storage and keep the keys safe, PKI also has the facility to update the keys on the demand of the member's. Few protocols of internet security such as IPsec and TLS are relying on the services provided by the PKI. Different level of access to the information stored in the database is also provided by the PKI so that the authenticated app can able to access the user protected data and the various level of access control keep the app restricted to access the unauthorised areas of user data in mobile and hence it provide a secured access by using either role based access control or rule based access control [6] and sometime it also have mandatory access control for the particular services.

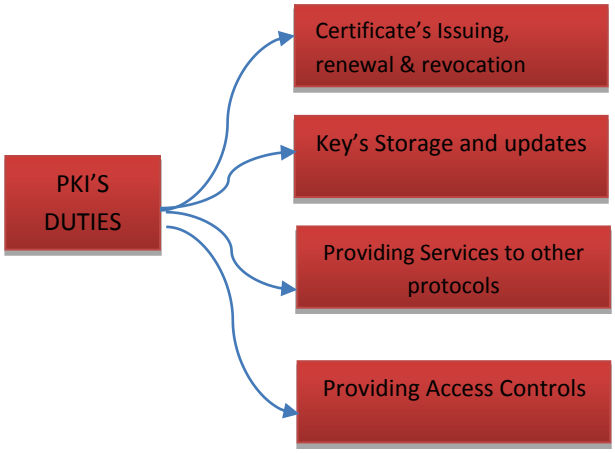
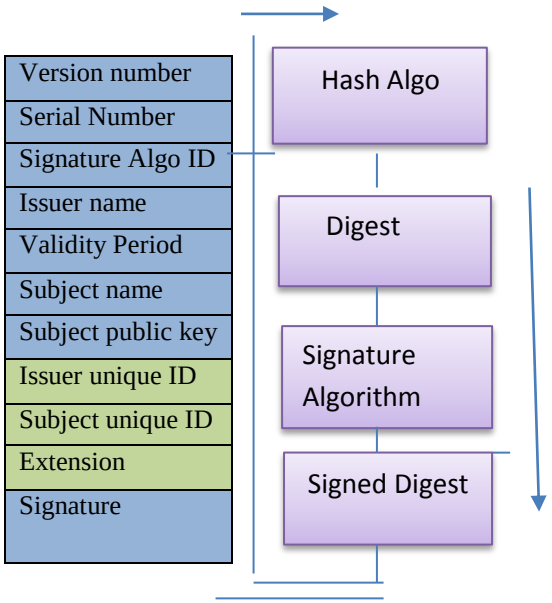



Fig. 13. PKI Duties

## X.509 CERTIFICATE FORMAT FOR ANDROID

For Increasing the confidence and reliability on the user and to provide the confidentiality a universally accepted format of the Certificate is necessary so in order to achive that ITU has designed X.509 a universally accepted certificate format. It describe the certificate in a structured way. It uses a well-defined rotocol known as ASN.1




Hash algorithm ID+ Cipher ID+ Parameter

Fig.14 X.509 Certificate Format for Android

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

*International Journal of Advanced Research in Computer and Communication Engineering*
*Vol. 4, Issue 9, September 2015*

Various Parameters of the Certificate is as follows:
**Version Number**: The version number is defines the version of the X.509 certificate. It generally started with 0.
**Serial Number**: This particularly defines the different different numbers assigned to the various certificate.
**Signature Algorithm ID**: This field particularly defines the algorithm used in to sign the certificate, other parameter required to sign the certificates is also included in this.
**Issuer name**: This parameter defines the authority which issue the certificate. It is basically in hierarchal format of country, state, organization, department etc. etc.
**Validity Period**: This parameter defines the earliest and the latest time for which the certificate is valid.
Subject Name: This parameter is used to define the entity to which the public key belongs.
**Subject Public Key**: This parameter defines the most important field sometime it is called the core of the certificate known as the Owners Public Key. It also defines the public key algorithm and its parameter of the corresponding certificate.
**Issuer unique identifier**: This is an optional field and used to differentiate between two issuer to have the same issuer field value then issuer unique identifier are different.
**Subject unique identifier**: It is also an optional parameter and used to allow to different subjects which have the same subject field, if the subject unique identifiers are different.
**Extensions**: This parameter is used to allow issuer to add any extra private information to the certificate if required.
**Signature**: This one is the most important parameter of Certificate, it consist of three sections. First include all other field in the certificate second include digest of the first section encrypted with the CA's public key, third section include the algorithm identifier which is used to create the second section of the signature part.

## IV. RESULT ANALYSIS

In this model after analysis we come to know that the android app is at high risk and most vulnerable for malicious activity and hence here in this model we proposed a solution which reduce the risk associated with the android app, malicious activity can be reduced at maximum extent after using this model and hence the confidentiality, integrity and availability of the android app can be improved and hence the risk associated with malicious app can be reduced. This model uses a trusted model a third party certifying authority, PKI duties for android apps and a certificate format for the android app and a strict and stern compliance associated with the implementation of the policy for the development and distribution of the android apps and hence all these things taking together prepare a defence in depth model for android app security at technical and managerial level for the safe and secure use of android apps and developing the confidence and trust on the clients of the android apps.

## V. CONCLUSION

After analysing the data we came to know that the development of malware is continuously growing day by day and hence the criticality of the problem is also growing and should be under great concern and hence for this reason a deep and rational analysis is required to successfully overcome this problem. Android malware is considered as one of the most successful malware which infect most of the android apps. The developers are continuously trying to mitigate the severity of the loophole which found day by day but few more intensive step should be needed. Most of the time the necessary steps are being avoided by the users which could prevent the damage which has been done by these android apps.

The most effective area is the financial section, most of the people used the various transaction apps used in mobile to monetary transition and here these malware do the most critical damage by sniffing the user's credentials and hence it would be catastrophic. Since most of the android apps have been developed for the windows operating system as they prefer windows as a platform and hence most of the apps have been compiled in windows operating system and hence it is most vulnerable. Almost all apps which have been used are of self-signed and hence the legitimacy is not been backed by any certifying authority which become a honey pot for the malwares. Most of the app which have been cloned and make malicious is being decided by which app have been downloaded frequently. Using this information we are able to improve the system involve in app development and distribution and hence we are able to categorize the app which are highly vulnerable and potentially malicious.

## REFERENCES

[1] Fire Eye Security Reimagined: Out of Pocket, a comprehensive mobile threat assessment report of 7 million iOS and Android Apps.
[2] Machigar Ongtang, Stephen McLaughlin, William Enck and Patrick McDaniel "Semantically Rich Application-Centric Security in Android"
[3] RSA Conference 2015 "Android Security Data from the Frontline"
[4] HISPASEC Spanish National Cyber Security Institute "Android Malware Situation"
[5] Ryan Johnson, Zhaohui Wang, Corey Gagnon, Angelos Stavrou "Analysis of Android Applications Permissions"
[6] Vanessa N. Cooper, Hossain Shahriar and Hisham M. Haddad "A Survey of Android Malware Characterisitics and Mitigation Techniques".
[7] Parvez Faruki, Ammar Bharmal, Vijay Laxmi, Vijay Ganmoor, Manoj Singh Gaur, Mauro Conti and Muttukrishnan Rajarajan "Android Security: A Survey of Issues, Malware Penetration and Defenses"
[8] Jinhua Liu Wenbo Pan, Jiahui Hu, Xianwei Zhou, Jianwei An "Research Of Secure Ecosystem Based On Android Platform"
[9] Xiaoyong Zhou_, Yeonjoon Lee , Nan Zhang , Muhammad Naveedy and XiaoFeng Wang "The Peril of Fragmentation: Security Hazards in Android Device Driver Customizations"
[10] Takamasa Isohara, Keisuke Takemori and Ayumu Kubota "Kernel-based Behavior Analysis for Android Malware Detection"
[11] Wook Shin, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka "Towards Formal Analysis of the Permission-based Security Model for Android"
[12] Wei Tang, Guang Jin, Jiaming He, Xianliang Jiang "Extending Android Security Enforcement with A Security Distance Model"
[13] Welderufael Berhane Tesfay, Todd Booth, and Karl Andersson "Reputation Based Security Model for Android Applications"
[14] Lin Sun, ShuTao Huang, YunWu Wang, MeiMei Huo" Application Policy Security Mechanisms of Android System"
[15] Parvez Faruki, Ammar Bharmal, Vijay Laxmi, Vijay Ganmoor, Manoj Singh Gaur,Mauro Conti, *Senior Member, IEEE*, and Muttukrishnan Rajarajan "Android Security: A Survey of Issues, Malware Penetration, and Defenses"