# Analyzing Data Leakage Using Third Party Connections in Mobile Applications

*Thesis submitted in partial fulfillment of the requirements for the award of degree of*

**Master of Engineering**

in

**Information Security**


*Submitted By*

**Pradeep Kumar**

**(Roll No. : 801333016)**


Under the supervision of:


**Dr. Maninder Singh**
Associate Professor

**Dr. V.P. Singh**
Assistant Professor




COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004


**May 2015**

## CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Analyzing Data Leakage Using Third Party Connections in Mobile Applications*", in partial fulfillment of the requirements for the award of degree of Master of Engineering in *Information Security* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Dr. Maninder Singh & Dr. V.P. Singh and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Pradeep Kumar

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Dr. Maninder Singh
(Associate Professor, CSED)

Dr. V.P. Singh
(Assistant Professor, CSED)

**Countersigned by**

**(Dr. Deepak Garg)**
Head
Computer Science and Engineering Department
Thapar University
Patiala

**(Dr. S. S. Bhatia)**
Dean (Academic Affairs)
Thapar University
Patiala

## ACKNOWLEDGEMENT

I would like to express my deepest appreciation to Dr. Maninder Singh and Dr. V.P. Singh, my mentors and thesis supervisors for their constant support and motivation. They had been instrumental in guiding me throughout the thesis with his valuable insights, constructive criticisms and interminable encouragement.

I would like to thank all the faculty members and staff of Computer Science and Engineering Department who were always there at the need of hour and provided all the help and facilities, which I required, for the completion of this work.

I offer my deepest gratitude to my family for their support and affection and believing in me always. I also want to thank my colleagues, who have given me moral support and their relentless advice throughout the completion of this work.

Pradeep Kumar
ME (Information Security)
801333016

## ABSTRACT

In previous few years, an incredible growth is witnessed in the popularity and pervasiveness of smart phones. And also it has seen that new mobile applications are built day by day. These applications provide users functionality like social networking apps, games, and many more. Some of the mobile applications might be having a direct purchasing cost or be free but having an ad-support for revenue and in return these apps provide users' private data to ad provider without users consent. Worryingly, some of ad libraries ask for permissions beyond the requirement and additional ones listed in their documentation. Some apps also track users by a network sniffer across ad providers and its applications. Though from security point of view, users have right to know if someone is taking their private data. It is often ineffective at conveying meaningful, useful information on how a user's privacy might be impacted by using an application.

In this research work, the effect on user privacy of some grossing Android apps was examined that includes apps which provide private data of user without their permission. Android device is firstly rooted for this work and then made some useful changes to collect the required traffic from different applications. On basis of certain parameters, traffic is analyzed and effective results concluded. Using third party connections that an app makes, a threshold is defined to know about legitimacy of application. Some other parameters were also added to check whether an app is stealing users' private information.

# Table of Contents

# List of Figures & Tables

**List of Figures** **Page No.**



**List of Tables**

# Chapter 1

# INTRODUCTION

## 1.1. Information Security:

Information security aka InfoSec is branch of security which defends the information in various aspects like unauthorized access, use, disruption, inspection, modification, disclosure, perusal, recording or destruction. Information technology security or information security is securing the computing devices either it is a desktop, router, data center or server etc. the data must be safe either it is natural disaster or any security breach or system failure. As almost every information is kept on computers in today's world so security specialist is needed to deal with such issues. The most common method of providing information assurance is to have an off-site backup of the data. The definition of information security is "Preservation of confidentiality, integrity and availability of information in addition with authenticity, accountability, non-repudiation and reliability" [1]. The basic principles of information security are:

- Confidentiality.
- Integrity.
- Availability.

### 1.1.1. Confidentiality:

It means maintaining the privacy. Confidentiality is to giving access only to the authorized persons. No unauthorized person should be able to get access to information.

### 1.1.2. Integrity:

Integrity means maintaining and preserving the consistency and accuracy of data. Data should not be affected in the way or transmission by unauthorized or protected manner.

### 1.1.3. Availability:

The information should be available when needed. No unauthorized person should be able to un-avail the information. Attacker tries to do this by DoS and DDos attacks which stops the availability of information to the users.

The above mentioned are the main and basic principles of information security. In addition to these, there are two more principles which have equal importance. Both are explained below:

### 1.1.4. Authenticity:

In information security, it is necessary that the parties who are communicating are genuine. Using digital signatures one can authenticate what they claim to be. It may so happen that an unauthentic person may get included in transaction. It is duty of security personal to authenticate in advance.

### 1.1.5. Non-repudiation:

Non-repudiation implies one's intention to fulfill the regulations of contract. It also assures that if one party made a transaction cannot deny having received a transaction nor can the other party deny having sent a transaction.

## 1.2. Network Security:-

It is a special branch of computer networking that involves protecting a computer network infrastructure. Network security is the job of network administrator who implements network hardware, software and security policies to secure a network and its resources accessed in unauthorized manner. And also ensure that authorized persons have adequate access to network and resources. Network security is a strategy and provisions of an organization for ensuring security of its assets and network traffic [2]. To maintain security of an organization a strategy is followed which includes:

- Policy.
- Enforcement.
- Auditing

### 1.2.1. Policy:

IT security policy is the principle document of network security. Its goal is to figure out the rules for ensuring the security in organization. All the rules are mentioned in this document, how and what ways to use the organizational assets and resources. It is made according to organization's culture and use. The auditing and enforcement procedures for any regulatory compliance of an organization are required to meet the security policy [3].

### 1.2.2. Enforcement:

Enforcement mainly concerns about analyzing network traffic flows and aim to preserve the CIA triad (confidentiality, integrity, and availability) of all systems and information on the network. It also enforces the employees to follow the IT security policy strictly [3].

### 1.2.3. Auditing:

The main aim of auditing is to check on enforcement measures to determine how exactly they are aligned with the IT security policy. It also encourages the contiguous improvement in the security policy consistently. It also gives the opportunity to the organization to adjust and align their policy and enforcement strategy [3].

## 1.3. Type of attacks in network security:

There are several type attacks possible in network. But these can be divided two categories:

### 1.3.1. Passive attacks:

In these attacks, attacker only monitors the traffic and looks for unencrypted passwords and other useful information. These are attacks in which attacker play no active role and chances of getting revealed the identity of attackers are less. Wiretapping, sniffing and eavesdropping come under this category [2].

### 1.3.2. Active attacks:

In these attacks, attacker tries to break into the secure system or network. This can be done by sending virus, Trojan or any malicious packet and there are chances of

attacker get traced back. Denial of Service , DNS spoofing, Man in the middle attack, ARP poisoning, Buffer overflow, Heap overflow and SQL Injection are some of the popular active type attacks [2].

## 1.4. Mobile Security:

The term digital is mainly used in computers. And digital security comes in area of securing the information that is on computer or internet. In broader view one can also call it as internet security that comprises the information, resources and your personal internet identity. As it has seen in previous decade the number of users of digital technology has increased in a dramatic way. Around 3 billion users are on internet [4] [5] and 156 million users are of smartphone approx. [6].Digital security is protection of digital identity. It includes taking necessary precautions to secure identity, assets and technology in online and mobile world. Any digital product that is online or connected to any network can be hacked or stolen, so a hacker may get personal information of any digital product. It can be stealing of your credit card number, social security number (only in US) or passwords of bank accounts. So in a short way, digital Security is simply about being secure about everything that is connected to any network. Ex., when one begin to learn about computers and internet, it is taught to not reveal any personal information on internet, such as full name and contact numbers. Then after, one learns to use antivirus software and other security measures that keep one safe from cyber or digital attacks. Now specifically about digital security regarding smartphones, the platform used for smartphone are not pretty much secure and very prone to attack. Also the smartphone users are not that much aware about the security of their phone.

The main platforms used in smartphone are android, iOS, Windows, blackberry, Symbian, Bada etc. The share of android in global market as per 2014 statistics is 81.5%, iOS has 14.8%, windows reaches up to 2.7% and blackberry got only 0.4% [7]. Some of above given operating systems are obsolete now like Bada, Symbian etc. The android has become so popular because it's cheaper than its rival iOS and also allow to users creating applications for fun or revenue and let it go register on google play store. With that android has reached the unbelievable level of use in market but left us prone to a huge number of insecurities as the applications created by a user in play store are not

necessarily have a proper and secure mechanism to use. This lures attacker or hacker to attack. In past Gmail app product of google was hacked by hackers. So if a google application can be hacked then how could a naïve user who is building an application for the first time can make his application secure and save its users from attacks. Its new operating system still needs changes and vulnerable to attacks.

A wide range of applications are being made like online banking, health monitoring apps, social networking applications etc. Many developers release apps (short for applications) for free and gain revenue from ads. In return they provide users' sensitive data to ads providers. Based on the information provided, ads companies provide only those ads to users which are most relevant to them. The more specific information ad provider get, the more suitable and profitable ads will be. Whenever an ad library embedded in a mobile app on the user's smartphone requests an advertisement and after that it sends private information about the user and device to the ad servers, thus it raises concern about users' digital privacy. An online mobile application can store the passwords and credit/debit card numbers of users and that can be powerful breach in their account. Someone having health monitoring application installed get stored all details about his health on server and many related other information that is totally complete breach in someone's personal life. These applications are intended to watch on customers' private information. A Malicious applications that steal customers' data may also behave in the same way, like transmitting private information to cloud or any third party server. So, transmission of sensitive information by itself may not stipulate true privacy leakage; a better demonstration should be whether the information transmission is on users' intention or not.

- ***User-intended data transmission***: To use an application on the extended mode a user often tolerates when private data is taken from his/her device using some communication channels. Some applications ask for the registration using some email id or phone number. For example, like WhatsApp which is a kind of social networking app which reads users' personal contacts, images, messages and other related information. If user doesn't allow app to access to these information, he/she won't be able to use the app, so users is bounded to provide such information. Another example, GPS service if one

wishes to use this app he/she knows very well that [8] his/her location will be sent out to get intended contents tailored to the particular location. Since to use application user must have to provide sensitive data for functional use and that to be with his/her own intention. So one cannot treat such type of transmission of data as privacy leakage.

- ***Unintended data transmission*:** The unscrupulous transmission of sensitive information performed by an application that is not user intended and not relevant to the function user demanded or application providing, is elucidated as unintended data transmission or in other words privacy leakage. In the most cases, the malicious apps usually do this in a stealthy manner and without prior knowledge of users.

In a recent study it has shown that Android applications frequently send private data of user to unknown destinations without consent [9]. Thus, to stop this happening and save users data leakage, strong analysis tools that Android applications are needed that can identify and remove malicious applications. Some state-of-art approaches to detect privacy leakage on smartphones has been introduced that focus on detection of sensitive data transmission [10, 11, 12, 13, 14]. However, most of mobile applications also use cloud computing, because of what privacy leakage by mobile applications is a subject that needs to be reconsidered. Many of these applications provide services to end users from cloud and to do so these applications normally collect sensitive data from users such as location, calendar, contact, to transmit to the cloud. Malicious application that steals user data also exhibits the same behavior like any other legitimate application.

## 1.5. Challenges of Mobile security:

As above discussed smartphone users are exposed to different threats. According to Google's android security report 2014, 26% increment in third quarter. These threats may disrupt operation of smartphone and transmit user data. The main threat targets are:

1. **Data**: It is the main target of any attacker. Some sensitive information like passwords or Debit/Credit card number may get stolen.
2. **Identity:** Each smartphone has a different identity like IMEI, IMSI or UDID. Apps may transmit all these information to steal the identity of owner.

3. **Availability**: one can deprive the services or limit the access by attacking a smartphone.

## 1.6. Types of attacks in mobiles:

As the popularity grew of mobile phone, it becomes easy for attacker to attack as they have now a larger platform to exploit. And also the users have no education regarding how to use and what to use. With the invention of smartphones and new technology android, the market extended exponentially. There are various ways in which one can segregate the type of attacks in mobiles. Some of them are explained as follows:

### 1.6.1. SMS and MMS based attacks:

Some mobile phones have shortcoming and it becomes difficult for them to handle an unaware situation if any occur. Some phones have problems in dealing with the binary SMS messages. It is quite possible that by sending an ill-formed message can cause phone to restart and leading to denial of service attack. Like in any Siemens S55 if one sends a text message in Chinese font, the phone may get crashed. Similarly in Nokia mobile, mobile may be dysfunction if it receives a mail from mail id of having length more than 32 characters. Another way to attack is sending an MMS with having an attachment with it. This attachment might be infected with virus. Whenever receiver receives the MMS, the attachment is gets downloaded and opened and with that phone gets infected [15].

### 1.6.2. Attack based on GSM Networks:

The attacker attacks the encryption mechanism followed by the GSM network i.e. A5. There were two variants of this algorithm: A5/1 and A5/2. Both of these were made public and due to that it was possible to break the encryption. It is easy to do eavesdropping and cryptanalysis in these algorithm. By eavesdropping one can attack on mobile radio networks using a fake base station called IMSI catcher. When the encryption algorithm of GSM is broken, anyone can intercept all unencrypted communications done by the victim's mobile [15].

### 1.6.3. Wi-Fi based attacks:

An attacker by doing eavesdropping intercepts the Wi-Fi communication. The security in WLAN is more vulnerable. Firstly WEP encryption system was used and it

was vulnerable as the length of key was short or limited. It was possible if an attacker can break the password then he may easily get in the local network of the victim. And if attacker succeeds in breaking identification key, it becomes possible to attack not only the phone but the entire network [15].

### 1.6.4. Bluetooth based attacks:

There are certain issues with the security in Bluetooth system. Any unregistered service does not require any authentication and virtual serial port is used by the vulnerable applications to control phone. To attack through the Bluetooth service attacker must be in range and also the Bluetooth is in discoverable mode. Through sending a file via Bluetooth and if recipient accepts, a malicious file is transmitted. Cabir is a type of worm that spreads via Bluetooth connection [15].

### 1.6.5. Web browser based attack:

The mobile web browsers are new and attracted area for attacker as like other they include plug-ins and widgets. And the most important thing they are in developing stage. In web browser based attacks, attacker uses the leverages like stack based overflow and other vulnerabilities in libraries. It is possible in all kind of operating system either Android or iOS. Smartphones are also vulnerable to phishing and other malicious web site based attacks and there is a big problem with smartphones is they don't have strong antivirus protection yet [15].

### 1.6.6. Operating System based attacks:

One may apply any number of secure mechanisms but if there is vulnerability in operating system, it might be going to affect one day surely. There are several loopholes in operating systems of smartphones as these are in earlier stages and developers are not much aware about the kind of attacks possible. It was possible to circumvent the security of operating system and bypass the bytecode verifier and access underlying operating system. Similarly in windows mobile OS, it possible to edit its pointer from general configuration file to a modifiable file. Also it is possible one using the malicious certificate may do changes in the directory whenever an application is installed as at that time it has root privileges [15].

# Chapter 2

# LITERATURE SURVEY

Security is the prior thing in today's world. Nothing is secure if it is connected to any network. Day by day the attacks are increasing as the programmers making code is having loop holes and that lures attackers. The way the applications are made is not secure, even in software industry security is taken as last step. But security is needed everywhere in design, modelling and programming. With the use of smartphones, security becomes more crucial as the platform is new and it is in its developing stage. Programmers need more proactive approach to save their apps and users get affected from different attacks. It is ridicule to see that more than 80% of users have no knowledge to use the internet and apps. Users are not aware about their security; it never comes in their mind to read the terms and conditions of use any application. As it may so happen that developer of application may reading its private information and using it for some personal benefit or purpose. With the idea of Internet of Things, it becomes individuals' responsibility to be secure as almost everything will be on internet from household things to business. This puts everything on edge as things becomes more vulnerable. More the things get online, more they get prone to attack. To have a better view on the application interaction with our smartphones, rooting of android device was done that allows having root privileges to users to see what it is going through.

## 2.1. Mobile Operating Systems:

There are numerous operating systems for mobile phones like Android, iOS, Windows, Symbian, Bada, Blackberry etc. Each operating system has its major features and with that some of the shortcomings. In this section, discussion about every operating system is given briefly and their popularity.

### 2.1.1. Android:

Android is Linux kernel based mobile operating system developed by the Google in 2003 by Andy Rubin, Rich Miner, Nick Sears and Chris White. Its default interface is based on direct touch using touching pads. ARMv7 and ARMv8 with x86 and MIPS are officially supported hardware for Android operating system. It evolved in years from

version **Froyo** released in 2008 to the latest version **Lollipop** that is released in May 2015. Android's source code is released under open source license by Google. Android is pretty much popular as it is ready-made, customizable and low cost operating system. Android operating system is written in combination of C, C++ and Java. The core part is written in c and UI part is mainly developed in C++ and java. It supports the monolithic type of kernel. It is widely available in 68 languages. It supports the application made for supporting android platform and those are widely published under Google Play Store where any one can register and publish his/her application. With the upbringing of Android it gave more challenges and fun to developers to get recognition and revenue through creating different application [16].

### 2.1.2. iOS:

The most popular operating system is iOS that was designed by Apple in 2007 and it supports only the Apple hardware including iPhone, iPad and iPod touch. This one is developed by the Steve Jobs and Scott Forstall. It belongs to Unix- like (BSD) and OS X family and supports a hybrid type kernel. It is written in C, C++ and Swift languages. This was the proprietary of Apple not released under any open source community like the Android. It is available in 34 major languages. It supports the ARM architecture of 64 and 32 bit. The default user interface for iOS is Cocoa touch. The operating system evolved through the years from OS X to iOS 9 by adding multiple and lucrative features in it. Due to the constant evolution in it, it remained the popular in market. The application for the iOS can be found on Apple Store and one can create the iOS app using the SDK released by the Apple only and can get revenue and recognition. It has the most secure feature to keep itself safe from the carious intrusion in market [17].

### 2.1.3. Blackberry OS:

Blackberry OS came at earlier than other OS like Android, iOS. It came 1998 and firstly used blackberry pager. As the name indicates it was developed by Blackberry. It follows the JVM type of kernel and support only proprietary of Blackberry. It is released in multiple languages. It supports the graphical interface. It is developed in C++ language by Blackberry developers. It evolved from version Blackberry 1.0 to Blackberry 10 but after that it discontinued the service but only provide the support to the existing system. Blackberry was the most popular in corporate world as its' native support to corporate

email and synchronization with Microsoft Exchange, calendar, notes and tasks etc. used with Blackberry Enterprise Server. Anyone could develop the blackberry use using third party SDK and get it digitally signed [18].

### 2.1.4. Symbian:

Symbian is the popular operating system developed by the Nokia Company in 1997. But it was actually developed by Accenture Company for Nokia. It follows the RTOS family. It's a proprietary of Nokia but in earlier stages it was released as an open source. It also supports the multiple languages as the popularities of Nokia mobiles in earlier years of 21$^{st}$ Century. It was developed in completely C++ language. It has real time micro kernel EKA2. The platform supported by Symbian is ARM and x86. It remained in service till 2010 after that Nokia adopted the Windows OS. It supports the graphical interface as well as Mini QWERTY keypad. The application can be created in C++, python, Adobe flash light and Java ME. It features the multi-tasking and memory management like all other popular operating systems [19].

### 2.1.5. Windows OS:

Windows came into mobile family in 2000 and developed by Microsoft. Before that Microsoft only dealt with the desktop related OS and applications. It was name as Windows mobile in 2003 with the idea dealing with enterprise consumers and business. By 2007, it became the most popular smartphone in U.S. It was written in C++ language. It has evolved from Windows CE released in 2000 almost 15 years back to windows 10. After the release of Windows 10, it discontinued the services as facing the competition from rivalries like iOS and Android. Like all other operating system it also supports the graphical user interface. It has a hybrid kernel. Earlier it was used by Nokia smartphones but later it was used in Windows mobile phones [20].

### 2.1.6. Bada:

Bada is the operating system developed by the Samsung Electronics in April 2010. The name was derived from Korean word which name Ocean. Samsung released it under open source license. It was completely developed in C++ language. It comes into POSIX family. It supports multiple languages. The kernel type was RTOS. It uses graphical and Touch wiz user interface. The applications are found at Samsung Kies. It

discontinued service in 2013 when Samsung started using the Android version of OS. It had released two versions of it. It shared almost 5% of market in its popularity time [21].

## 2.2. Application Security:

Application security is a branch of security that considers mainly about the security of application created for various platforms like web application, system application and mobile application. This ensures that there is no flaw in design, development, deployment and maintenance of application. It checks the complete code's life cycle of an application to avoid the security gaps in policy. OWASP and WASC define the provisions that one considers before building any application and also keep updating with latest upgrades. OWASP mainly provide solution to web applications. In application security, one learn what the known threats are and how to secure your application from attackers [22].

### 2.2.1. Types of Threats possible:

There are several threats possible in applications but here in this section only the most popular and important threats are discussed that one should take care before building any application.

#### 2.2.1.1. Input validation:

This is the most dangerous and general type of attack. Generally developers forget to put validation on user what they can feed into. Taking advantage of this some malicious user provides unauthorized input that leads to crash or infect the application. Before publishing of any application, its responsibility of developer that he/she puts limitations that what type and of what length a user can provide the input. Some popular attacks in this category are Buffer overflow, Cross-site scripting and SQL injection [22].

#### 2.2.1.2. Authentication:

Authentication is that one should provide the credentials to prove the legitimacy to use application. Authentication mechanism should be applied to some sensitive application so that on unknown or illegitimate person can do interference. Some attacks in this category are: Brute force attack, Cookie reply, Dictionary attack and Credential theft [22].

#### 2.2.1.3. Authorization:

Authorization is having right permission to use the application. One may have access to application but not allowed to access the critical and sensitive features and for that authorization should be applied. Attacks possible: Elevation of privileges, data tempering and disclosure of sensitive information [22].

#### 2.2.1.4. Session Management:

It is increasing area of possible attacks as the numbers of users are increasing exponentially in digital world. In this, whenever a connection is made to any web application a session ID is created. It may be possible that by luring the users attacker may hijack their session Id and can steals the private and sensitive information. For that proper security measures should be taken like HTTPS, TLS/SSL layer security. Attacks possible: Session Hijacking, man in the middle attack and session replay [22].

#### 2.2.1.5. Parameter Manipulation:

In this type attacks, attacker tries to manipulate the fields that provided the leverage to user to run certain tasks. In this, attackers do its malicious activity by providing some malicious command or query. Attacks possible: HTTP header manipulation, Form field manipulation, query string manipulation and cookie manipulation [22].

#### 2.2.1.6. Exception Management:

Exception management comes into picture whenever any organization is using a legacy application that needs to be updated or it may get breached but organization is not in situation to do that. Another scenario is organization using third party services to run the, application and the third party not having the proper security measures. These scenarios lures attackers to attack the application as they have the enough time to attack the application because of inability of organization to update the package. Attacks possible are: Denial of Service and information disclosure [22].

These are the certain type of attacks possible to an application. OWASP generated a report and shows the top 10 risks to mobile application security. These top 10 risks are lined as follows [23]:

- Weak Server side Control.

- Insecure data storage.
- Insufficient transport layer security.
- Unintended data leakage.
- Poor authentication and authorization.
- Broken Cryptography.
- Client side injection.
- Security decisions via untrusted inputs.
- Improper session handling.
- Lack of binary protection.

## 2.3. Rooting (Android):

Rooting is a process of allowing users of smartphones and other handheld devices to attain privileged control or root access. Android is having Linux like Kernel so rooting the Android device will give us similar access to administrative permissions as on Linux or any other system [24].Rooting is performed with goal of overcoming limitations that carriers and hardware manufacturers put on devices. Thus, rooting gives ability to modify or replace system settings and applications. With this one can also run specialized applications that require administrator or root level permissions. With the help of rooting, one can do removal or replacement of operating system with latest release.

### 2.3.1. Advantages of rooting:

There are several advages of rooting as it gives complete control over device. Some of popular adantages are given below:

1. Full control on theming capabilities like changing and theming from color of battery indicator to the look of the contacts and dialer pad.
2. Full control of CPU and kernel.
3. Full control on applications that includes ability to backup, restore. Also remove the bloatware that comes pre-loaded on various smartphones.
4. Automate the processes on device using applications like Tasker.

5. Allow user to install a custom firmware that provides control on a rooted device at an additional levels. Also Android is an open source operating system, anyone having proper skills can create their own customized version.

### 2.3.2. Disadvantages of rooting:

With great advantages rooting has some disadvantages also. With great joy of rooting one may have to pay a price of losing control over phone. Some of disadvantages are given below:

1. Phone might get bricked.
2. End up voiding phone's warranty.
3. Can become vulnerable to malwares.

## 2.4. Proposed Approaches:

There has been proposed a lot of approaches to the leakage of digital information in smartphones. Some of popular approaches are described below:

C. Mann et al. [25] have presented the framework to detect the privacy violating information flow in Android based applications using static analysis of byte code of an application. This framework was designed in such a way that it supports the Dalvik virtual machine instruction set. They identified the set of sources and sinks of private information which used to indicate the privacy policy to be enforced by framework via analyzing the Android API. System presented was mainly to track flow of explicit information only. The main focus of theirs is to detect the implicit information leaks due to program control flow for Dalvik byte code as implicit leaks in application sinks are easily detectable. The security system checks whether Dalvik bytecode implementation of android application confirms the privacy policy. In this, they completely analyzed the Android API for possible sinks and sources of private data. And regarding this some of the novel research results exhibited like systematization of Dalvik VM instruction set (218 instructions) into abstract instruction set (61 instructions) by capturing pertinent information flow aspects, Set of sinks and sources are carefully identified in the android API of private information which is used to define a private policy and detecting the explicit information leaks in security type system.

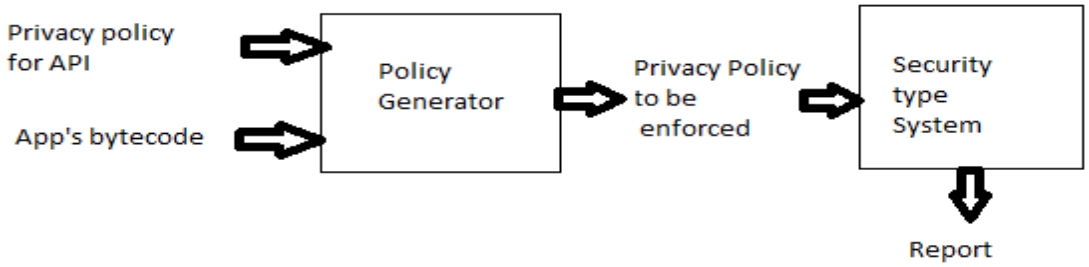



**Figure.2.1** Framework for static analysis [25]

Z. Yang et al. [26] presented a new approach i.e. AppIntent. It detects the privacy leakage in android applications. AppIntent framework is designed to know if the data transmission is user intended or not. For this event-space constraint guided symbolic execution technique is proposed and in that it extract the app inputs that were present in some user interactions and using dynamic analysis platform it institutively display context information of sensitive data that is transmitted. With this wonderful work, AppIntent has some limitations also that any kind of native code is not supported by it because of what it can't capture the privacy leakage in native code.

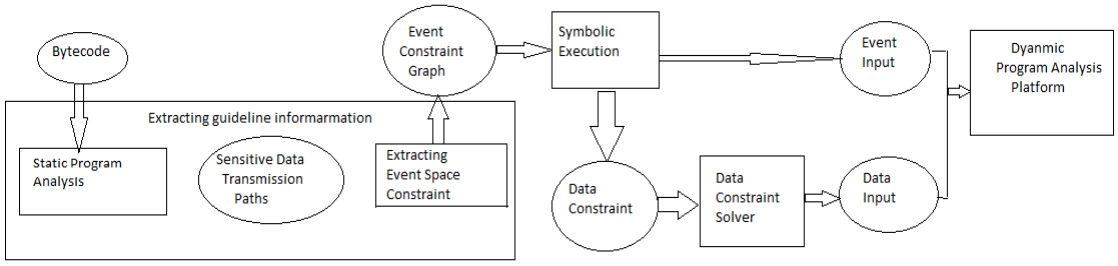



**Figure.2.2** Framework of AppIntent [26]

S. Rosen et al. [27] proposed a system i.e. AppProfiler that detects the application behavior through well-defined framework. This system mainly works in two steps which are: building knowledge base of API calls and that to be of their privacy relevant behavior and second step is using this knowledge base to provide behavior profiles for apps. For this around 80,000 applications were used in making knowledge base and then

defining the behavior profile. The goal of this application is to provide users the ability to make some informed decisions about applications they install.

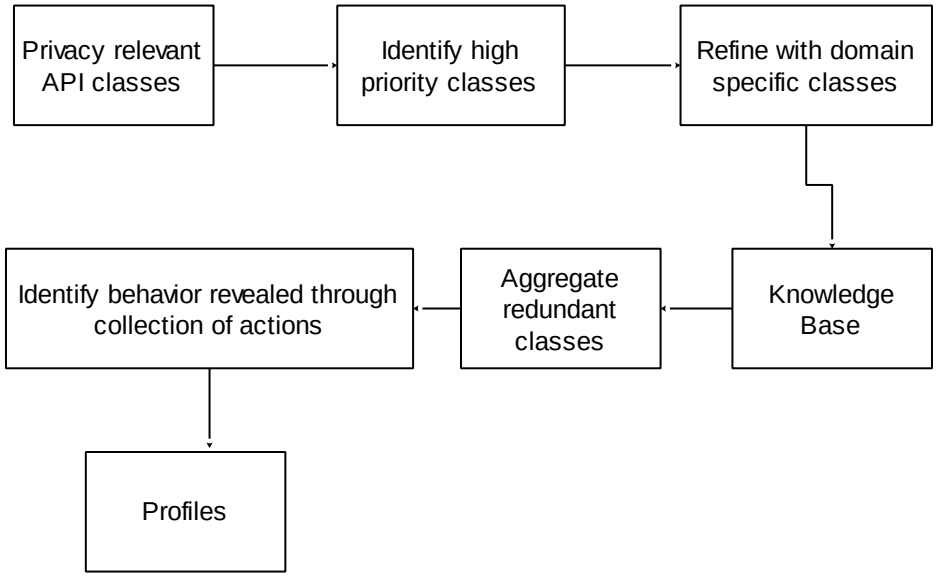



**Figure2.3** Framework of AppProfiler [27]

R. Stevens et al. [28] proposed a solution that was to analyze the Ad libraries those were embedded in smartphone applications. In this, it has been found that many of the ad libraries extract and use private information of users that these are not intended to be. To have a clear vision on this they chose a set of ad providers from top 500 applications on android platform and with some specific tools they realized that some of the ad provider breaching the privacy policy. In search of dig the deep, they manually analyzed each and every ad libraries, its functions and permissions. They found a certain set of ad providers that taking data that is optional and some are transmitting data that they are not supposed to be. Also it was found that this data is transmitted over network with least security measures that means users' private information may get leaked anywhere.

Although not a single ad provider provides complete private user profile but it has seen that UDID field present in nearly all app ad requests. Using UDID fields, it allows the observer to create a long-term user profile that includes targeting information and GPS locations. Finally, a solutions is proposed to several common ad libraries privacy

vulnerabilities that includes failure to user data protection in ad requests, misuse of UDIDs and lack of privileges separation between ad and application code on Android.

L. Btyuk et al. [29] proposed a static analysis for automatic assessment of android application and also the mitigation of unwanted and malicious activities provided. The process was folded in two steps: first, a system was designed that is able to assess android applications for static analysis and generate readable reports to the user. The next step comes to mitigate security and privacy threats. And to provide these automated reverse-engineering on binary application packages id done according to users' security preferences. This Deep static analysis of smartphone apps with comprehensive reports and privacy and security threat mitigation gives an end user great benefit in numerous ways. With this user gets an insight in privacy and security related information of any application.

Having this comprehensive reach to internal, user gets ability to decide whether particular functionality is malicious or not and also they have provided the proof of concept prototype of the implementation.

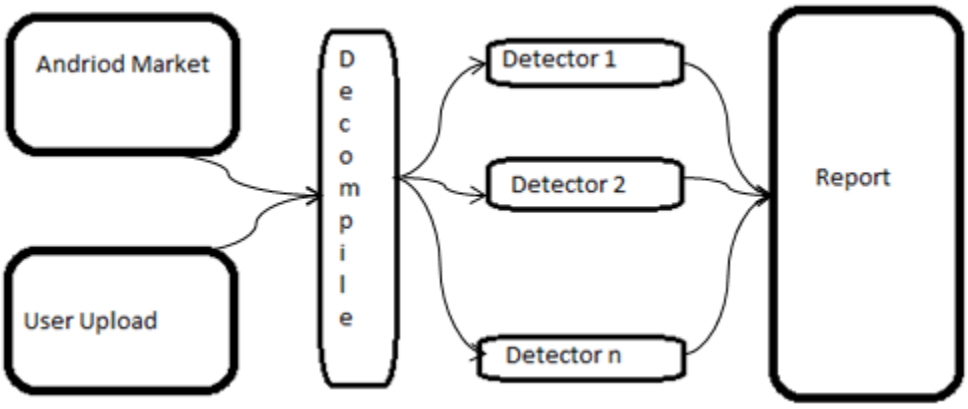



**Figure.2.4** Static analysis for automatic assessment of android application [29]

I. Bilogrevic et al. [30] presented a novel approach to preserve private information-sharing system that checks in (semi)automated way whether to share contextual information and also to what extent. They used the active machine learning for this. Their system adapts accordingly user's behavior and decides level if sharing information

without making any harm to private information to any third party. This system was evaluated over 70 participants and it gave the stunning results on sharing decisions. Its accuracy is up to 90% .


**Figure.2.5** The interface of application to detect privacy leak [30]

W. Zhou et al. [31] presented PiggyApp approach, to detect the piggybacked applications existing in Android market. Their presented system was fast and scalable. In this approach, they decoupled the primary and non-primary modules of a piggybacked app. Also they developed a fingerprint technique that extracts the various semantic features from the primary module and used this to construct featured vector. A metric space was made and a linear arithmetic algorithm of complexity O (nlogn) was proposed to detect piggybacked applications scalable and efficiently. The system was tested on around 84767 application collected from Android market and analyzed and results were very promising.

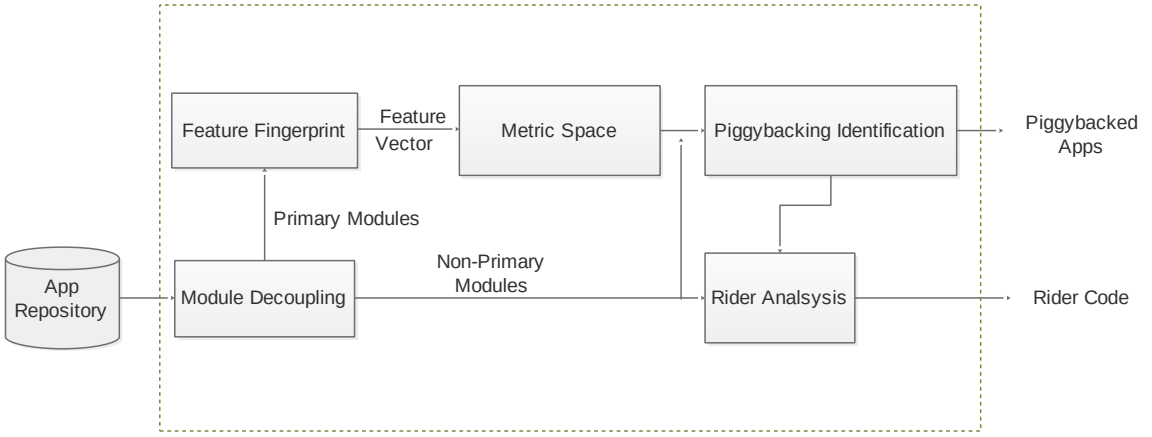



**Figure.2.6** System Architecture of PiggyApp [31]

J. I. Hong et al. [32] presented a toolkit Confab, for development of private information sensitive ubiquitous computing application. To develop this toolkit, all end user and application developer needs were collected using several surveys and various research papers. This toolkit provides support to ubiquitous computing apps and provide framework to customize privacy mechanism. It also provides mechanism securing the location privacy. Using all these standards one toolkit was developed to spectrum the privacy needs and trust levels. As all know ubiquitous computing is mostly criticized for the privacy risks it has. However many designing and architecture changes are made to keep it effective and mange privacy. To develop the Confab the end user requirements were clear value proposition, plausible deniability, decentralized control, limited retention of data, special exception for emergencies and simple and appropriate control and feedback [32].

P. Pearce et al. [33] presented an approach AdDroid, to detect privacy and security threats due to Advertising. Keeping in mind that, they integrated the advertising API i.e. AdDroid into Android platform. To protect the API, AdDroid used the privilege separation to distinguish sensitive information advertising network activity from the apps and provided other advertising permissions. PoC (Proof of Concept) of implementation of advertising API and AdDroid was also given. With the involvement of this Advertising API system into Android application not only Android system can evolve but privacy and security can also be increased. It is economically beneficial for developers, advertisers and platform itself. A study was performed of Android market and it was found that

almost 46% applications supporting advertisement were suffering from over privilege security permissions and also 34% apps of remaining block request for the access like location just for the adverting purpose.

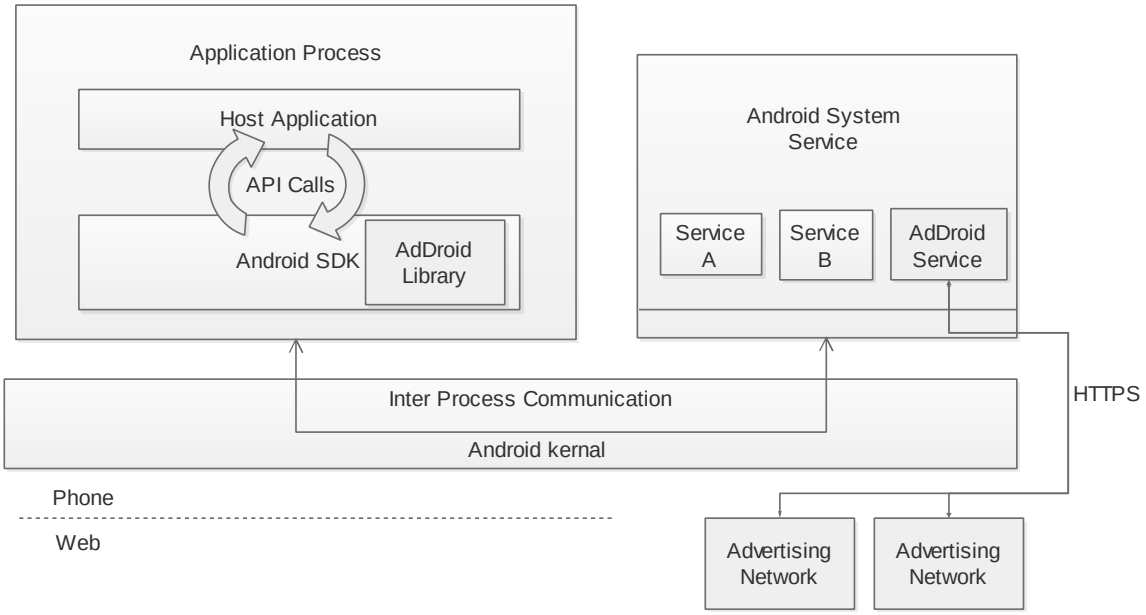



**Figure.2.7** AdDroid Design [33]

R. Balebako et al. [34] presented an approach where they calculated number private information sent. For this they first discussed the misconceptions regarding smartphone applications regarding the private information shared. To measure this gap into privacy leakage and the understanding of users' two parameters are taken into considerations: JIT (just in time) information that demonstrate at the moment data is shared and finally visualizing the all data shared through a session. It has seen that most of users' were not aware by the information sharing and the damage it can lead to them. But their approach alarms the user at the time if any application tries to steal any private information. This also demonstrated that finally what and where the data is transmitted by visual. These troubles come mainly because of the app and ad industry. To avoid this, there is a need to explore interface and tools that can improve users' awareness on leakage of private of information.

I. Leontiadis et al. [35] presented an approach prospectively look in the problem of privacy protection for smart phones. In this, they targeted the applications of Android market to identify significance of advertisement in a free application for mobile phones. The solutions is designed to detect the link between ad supported apps and private information sent about users that can bear risks regarding the breaking business model that provides the financial support. The provided prototype solution was aimed to introduce a mechanism that protects privacy that is interlinked with the free apps that supports ads to get revenue. This was done by entrenching a feedback control loop and to maintain equilibrium within the ad revenue and private information privacy controls are dynamically adjusted.

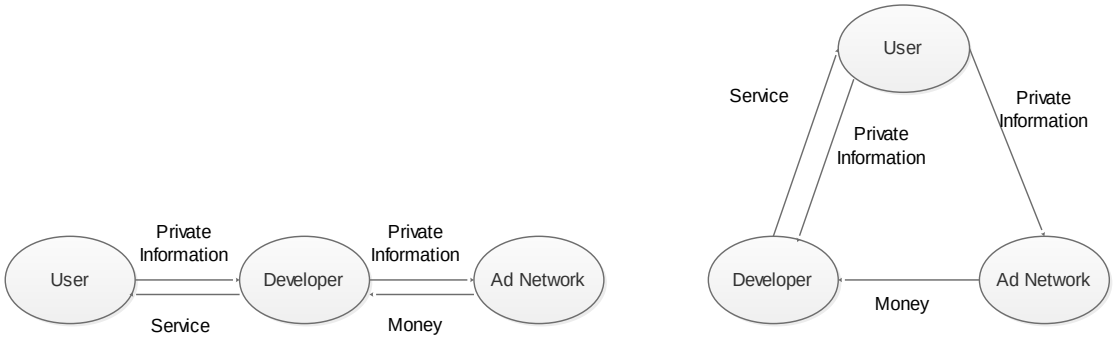



**Figure.2.8** Mobile Advertisement Models [35]

E. Chin et al. [36] have given idea that mobile phone ecosystem that includes application market, used pattern and application vendors are quite different and new from desktop computing. So to provide a security and improve computing, there is need to understand the privacy and security implications of users' behaviors and their perceptions at different step of the ecosystem. By understanding it comprehensively, solution can be provided to protect from the potential attacks and threats offered by mobile platforms. It was found in a study that users are feared of four main factors: data loss and fear of theft, misunderstanding about the security provisions, fears of touching or clicking accidently the wrong application and less trustworthiness nature of smartphone applications. To avoid these, an approach was proposed that contains new security measures that ensure privacy and security of users and it also guide users to avoid using untrusted applications. Also it is recommend that devices that come readymade services in which it is to remote

lock, use data backup and remote wipe services. It is also believed that mistrust applications could be eradicated by building up centralized markets with the information provided by users and application reviews.

S. Ickin et al. [37] have presented research on mobile's Quality of Experience. They presented the approach that considers both procedures quality and quantity where users are an active part of research. Firstly, information is gathered from different applications that with the directly in interaction with users and then is employed to ESM (Experience Sampling Method). Second, backtracked analysis of application's user experience and factors of employing are required. Then it is employed to DRM (Day Reconstruction method) to sync with the collection of past 24 hours. It was seen that DRM was very supporting for validating gathered data thorough CSS app. An analysis was presented with collected data and some factoring by impacting user's QoE. The novelty of this approach was that it concludes the influence of phone features, app performance, app interface design and user routines, usability and many more. It was seen that increased RTT and SRT values to study the QoS.

N. li et al. [38] presented an approach that is used only for location sharing applications. In this they have given a brief description of LSNs (Location Sensing Networks) that are used to share the location. It is used to connect the users socially based on their location. A user's location is very private and sensitive information. They studied different location sharing locations sensing mechanism and did a trace based analysis to get the actual way of real world sharing of location. It was also found that there is huge similarity between the information transmitted based on gender, age, and geographical locations. And almost similar protection measures are taken in friends. An extensive research is done and a model is designed on basis of friendship and user classification to trace the data. Finally it was concluded that it was the first large scale empirical study of LSNs.

# Chapter 3

# PROBLEM STATEMENT

## 3.1. Gaps in Study:

Mobile applications are now everywhere. Mobile applications now days are used for business, entertainment, bill payment, shopping and much more. But because of the time constraint and high competition among app developers, very less stress is laid on the security aspect while developing the applications. So there is need to encourage and educate the app developers to consider the security aspect and attacks associated with the mobile applications. The security flaws will not only affect the business and reputation of the company but it can also lead to loss in terms of money for example the app vulnerabilities can allow an attacker to transfer funds from one account to another. There are a very few automated vulnerabilities assessment tools that are available in market that aims to analyze the applications for presence of security vulnerabilities in mobile applications. One of the major problems in mobile applications is that these make connection to third party servers for the transmission of private information related to user. But because of unawareness users are still using such application those are stealing their private information. The main problem is that still there are no tools that can detect or prevent such data leaks and provide security to users.

## 3.2. Problem Definition:

After studying the problem in the existing domain, this conclusion has been derived that the mobile platform still has a number of flaws and developers are not much aware about these. The applications made by developers are somewhere damaging the user's privacy and stealing the private information that is not good as per security perspective. Developers embed the Ad libraries in their application code for information gathering purpose and when users install the app these libraries steal user information and send it to the Ad server to understand the user needs and providing the specific ads to increase their revenue. Similarly developers also get paid by embedding these Ad libraries and from this they generate revenue. Some of the gathered information is sold to some other organization and this may lead to breach in privacy of the user. If this information gets

caught in wrong hands it can lead to huge damage to the user. Thus seeing mentioned gap, it was decided to look into the problem of making third party connection and made a novel approach to reveal the secrets behind this. For this detection mechanism was proposed that detects the illegitimate applications which make connections to third party servers.

## 3.3. Objectives:

1. To detect third party connections made by an application on an Android device.
2. To define the threshold that derives the legitimacy of mobile applications.
3. To verify and validate apps based on experimentation performed using the test bed scenarios.

# Chapter 4

# IMPLEMENTATION AND EXPERIMENTAL RESULTS

This section describes the design of framework and methodologies. This work used different approaches to derive the desired results and those are explained in this section. An approach was proposed that is using the third party server connections made by a particular application. Also there are certain other parameters that were taken to provide the trust value to an application. The main focus of ours was to set a threshold value that defines the legitimacy of an application based on the trust value that it gets on different parameters. The framework to detect the legitimacy of an application is given below

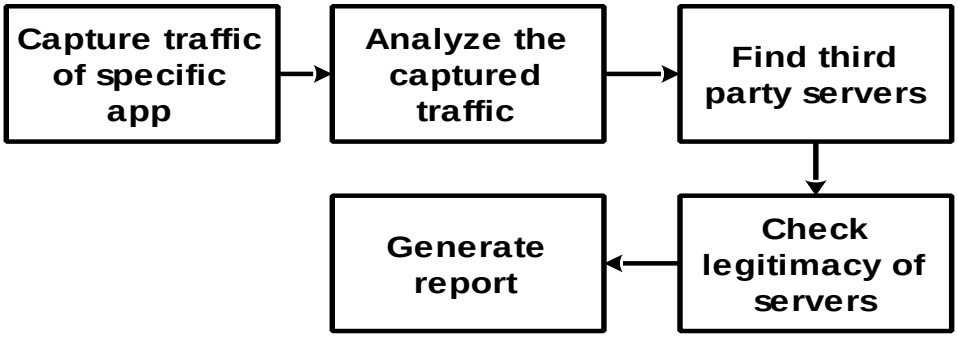



**Figure.4.1** Framework to find third party servers

The main focus was to set a threshold value that defines the legitimacy of an application based on the trust value that it gets on different parameters. The parameters taken are as follows:

- Permission request.
- Number of third party server connections.
- Size of a particular application.
- Number of connection to a particular third party server and data uploaded to it.

Keeping above parameters in mind, different mechanisms was used to detect the private data leakage in mobile applications. The approaches used are given following:

## 4.1. Implementation using AirPcap:

AirPcap was used to capture the Wi-Fi traffic of the Android phone and Wireshark & Network Miner is used to analyze it. Step by step procedure is given below:

a) Connect AirPcap to a windows machine.

b) Setup Windows machine with

- Wireshark
- Microsoft Network Monitor

c) Connect Android Phone of user1 to Wi-Fi network.

d) Launch mobile apps one by one to communicate with internet.

e) Capture packets and analyze.

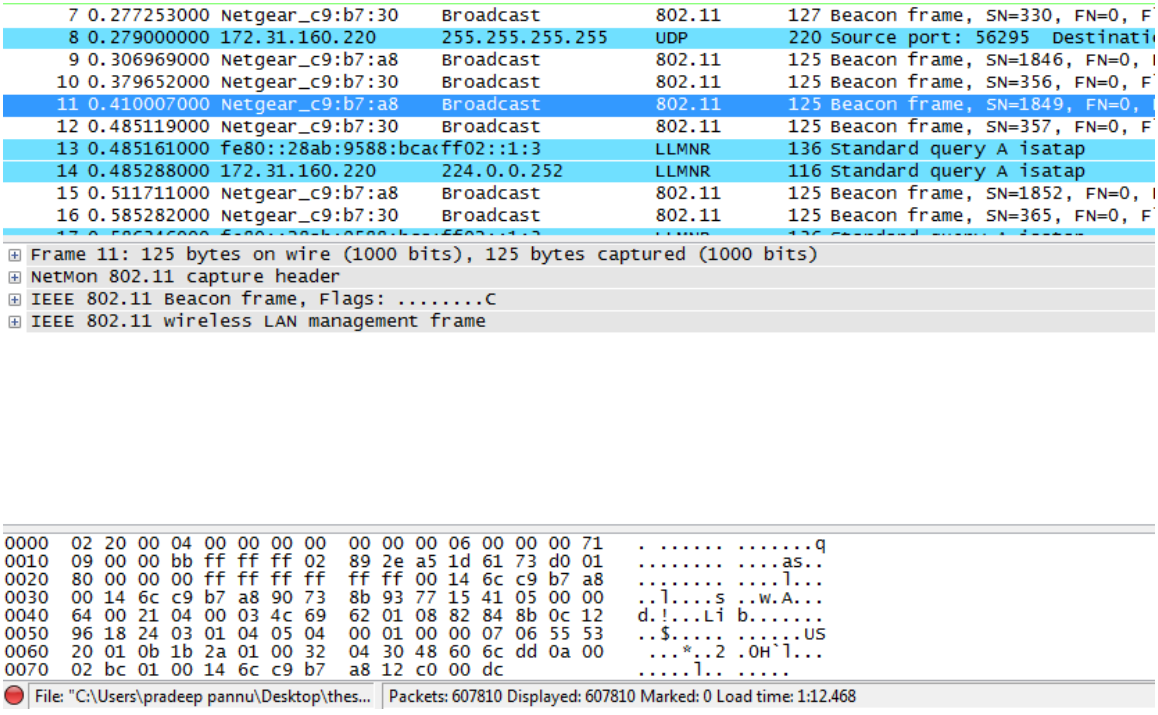



**Figure 4.2** Screenshot of Wireshark for management traffic

**Results:**

AirPcap captured Wi-Fi traffic between android and DSL modem. The following results are concluded:

a. Only management traffic, control traffic and packet with TLS encryption were present in captured packets. The private key of individual app was required for decrypting the data frames.

b. This method was not suitable for the case study and therefore not all desired results were drawn.

c. This approach was unable to see the encrypted TLS packets.

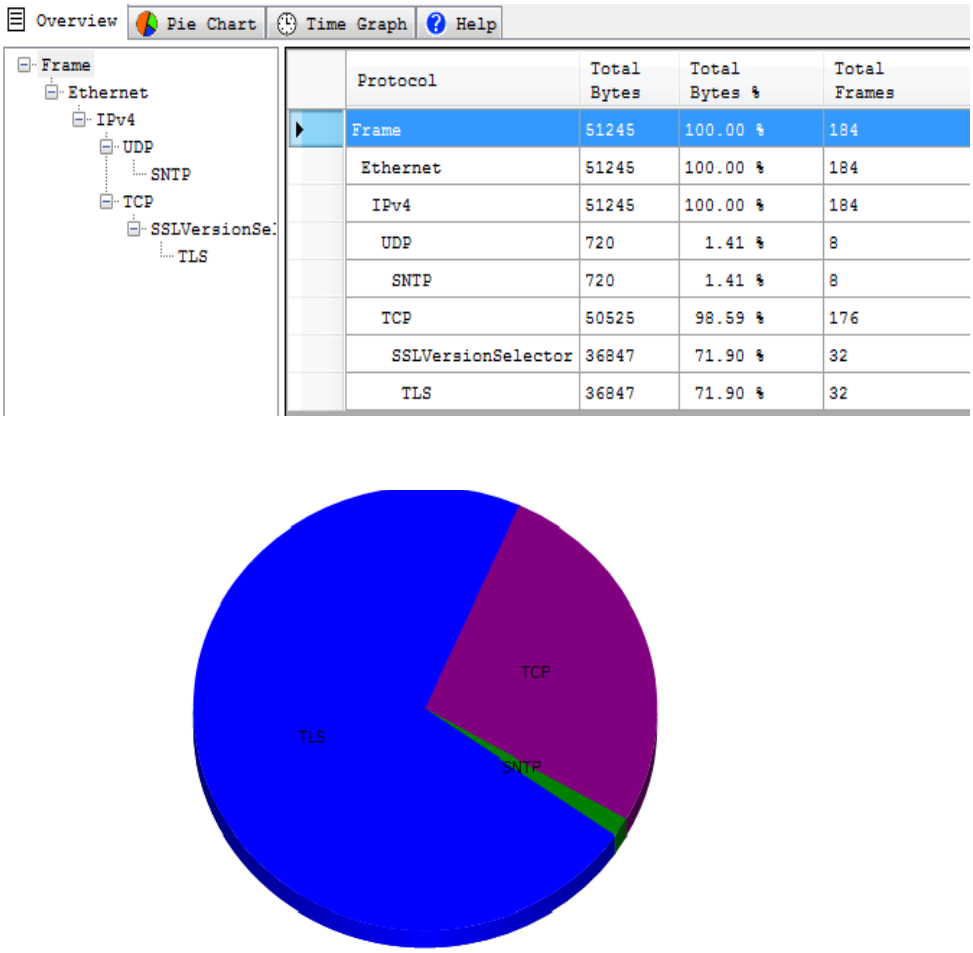



**Figure 4.3** Screenshot of Network Monitor for TLS encrypted data frames

## 4.2. Implementation using tPacketCapture pro:

tPacketCapture pro was used to capture the traffic for specific app. Step by step procedure is given below:

a) tPacketCapture pro was installed from Play Market on Android phone.

b) Launch mobile apps one by one and traffic is collected of one app at a time.

c) Captured packets were saved in pcap format and analyzed using different tools:

- Network Investigator
- Microsoft Network Miner
- Microsoft Visual Round Trip Analyzer

d) Number of connections made to third party servers was calculated for particular application. For this a windows shell script was created which detect the illegitimate FQDN (Fully Qualified Domain Name) for a particular app.

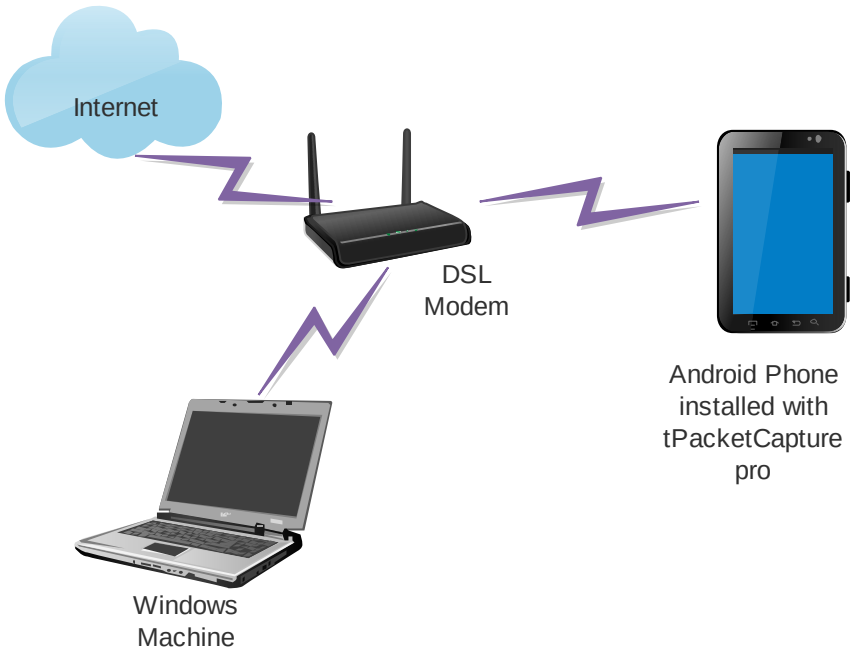
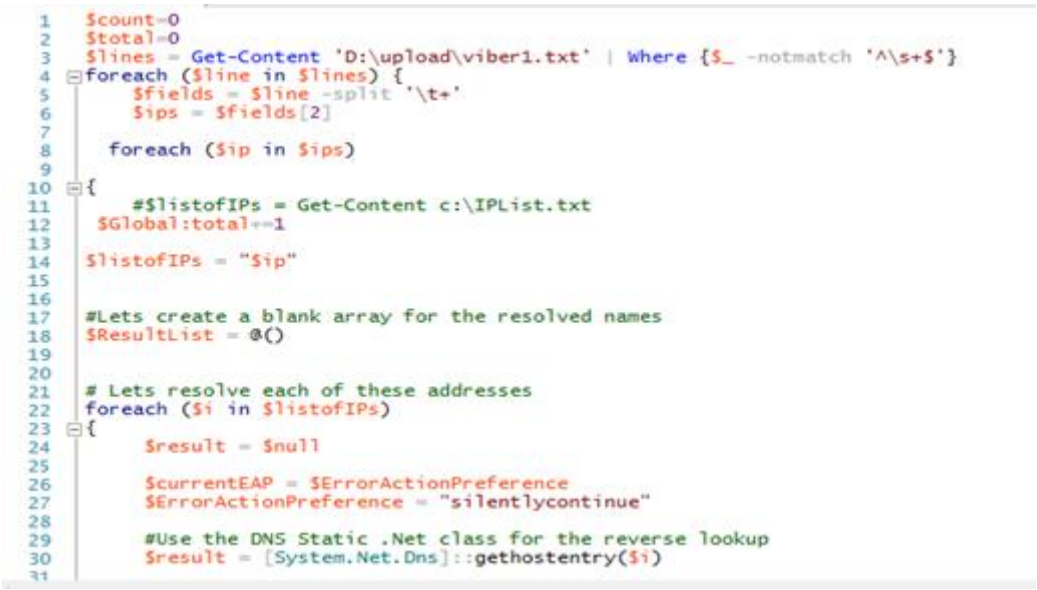



**Figure 4.4** Network setup using tPacketCapture pro



```
$count=0
$total=0
$lines = Get-Content 'D:\upload\viber1.txt' | Where {$_ -notmatch '^\s+$'}
foreach ($line in $lines) {
    $fields = $line -split '\t+'
    $ips = $fields[2]

  foreach ($ip in $ips)

{
    #$listofIPs = Get-Content c:\IPList.txt
 $Global:total+=1

$listofIPs = "$ip"


#Lets create a blank array for the resolved names
$ResultList = @()


# Lets resolve each of these addresses
foreach ($i in $listofIPs)
{
     $result = $null

     $currentEAP = $ErrorActionPreference
     $ErrorActionPreference = "silentlycontinue"

     #Use the DNS Static .Net class for the reverse lookup
     $result = [System.Net.Dns]::gethostentry($i)
```

**Figure.4.5** Code snippet to match FQDN

**Results:**

**a.** *Third Party Connections:* In this scenario, if an application is connecting to the particular third party servers many times. It may so happen that it is transmitting the users' private data. An app has to make connection to its legitimate server more to provide the functionality, rather to a third party server. It is also seen that apps transfer bulk of data to third party servers that indicates in breaching the security policies. To support this parameter some apps were analyzed and their top connected third party servers are shown in table given below:

**TABLE 4.1** Top third party connection to particular app in social networking domain

| **Application Name** | **Server Name (third party)** | **No. of connections** |
|---|---|---|
| App 1 | ec2-**-254-***-132.ap-*******-1.compute******.com | 1133 |
| | server-**-230-***-90.**m2.r.*******.net | 585 |
| | server-**.192.***.35.**m2.r.*******.net | 377 |
| | server-**-230-***-90.**m2.r.*******.net | 447 |
| | | |
| App 2 | a***-51-***-234.deploy.static.*********.com | 2514 |
| | **2-**.73.***.221.compute-1.*******.com | 150 |
| | **2-54.***.251.***.compute-1.*******.com | 174 |

| | | |
|---|---|---|
| App 3 | ***.192.***.18-static.revrese.******.com | 72 |
| | ***.168.***.121-static.revrese.******.com | 76 |
| | ***.168.***.227-static.revrese.*******.com | 196 |
| | | |
| App 4 | **2-**.169.***.8.ap-*******-1.compute.*******.com | 193 |
| | *3-ap-southeast-1.*******.com | 117 |
| | **2-**.169.191.3.ap-*******-1.compute.******.com | 562 |
| | | |
| App 5 | edge-star-****-01-***1.*****.com | 1536 |
| | edge-mqqt-***-01-***1.*****.com | 1392 |
| | a***-56-***-215.deploy.**********.com | 287 |
| | | |
| App 6 | server-**-230-***-100.a**50.r.******.net | 106 |
| | a***.56.***.41.deploy.**********.com | 148 |
| | server-**.231.***.41.a**50.r.*******.net | 155 |
| | edge-star-***-01-***1.********.com | 197 |
| | *3-eu-west-1.*******.com | 722 |
| | a***.56.***.40.deploy.**********.com | 347 |
| | | |
| App 7 | **2-***.22.***.119.compute-1.******.com | 324 |
| | edge-star-***-01-***1.********.com | 321 |
| | ***01*06-i*-*13.**100.net | 553 |
| | server-**-230-***-133.***2.r.******.net | 153 |
| | **2-50.**.207.1***.compute-1.********.com | 765 |
| | ***01*06-i*-*15.1*00.net | 987 |

**Due to confidentiality name of applications and third party servers are not revealed*

**b.** *Permission Request:* Whenever users install an app, it requests a set of access permissions and in those permissions, application request like read contacts, IMEI, phone name, operating system, camera, messages and many more. To run an application a certain set of permission is necessary but in addition to that application request for extra permissions those are for information gathering purpose for third party server and ad servers. When captured packet using tPacketCpature pro was analyzed then it was seen that some apps asking more than the permissions it actually requires to provide functionality. The screenshot of flashlight application permission justifies our argument.

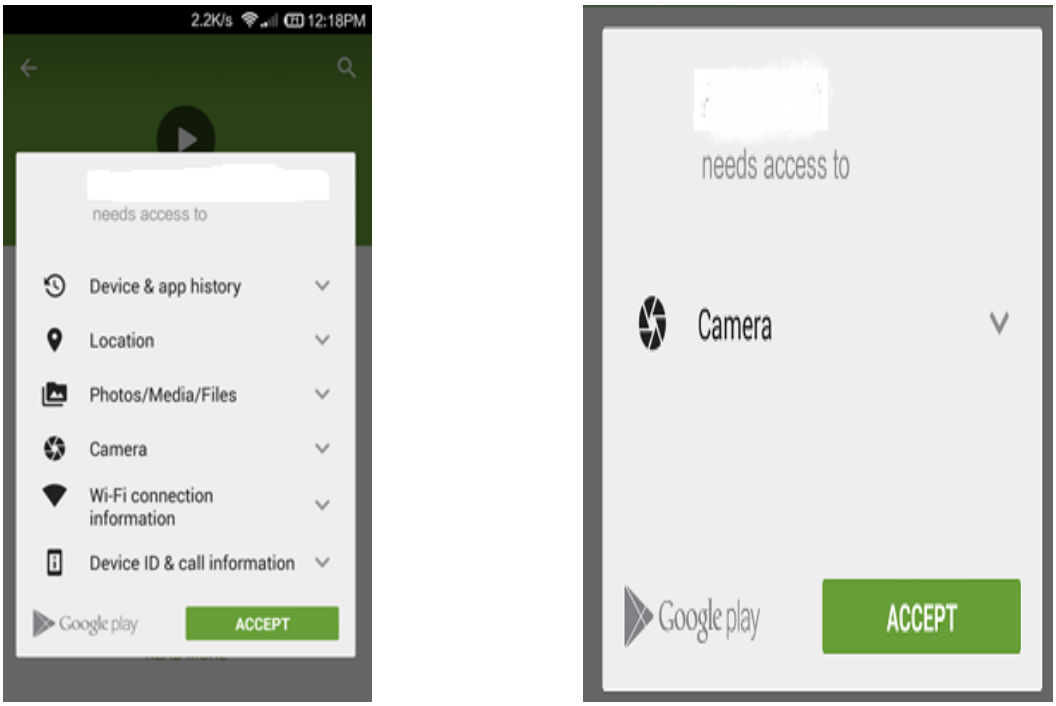



**Figure 4.6** Comparison between the permission requests of flashlight apps

**c.** *Size of app:* Almost all applications of a domain provide the similar functionality and hope to be of same size probably. But it generally happens that these applications vary in size a lot. Thus it shows that an application of larger size having some other extra package in it and that can be of ad libraries for the information gathering. As the size grows more doubtful the app becomes with respect to third party connections. For example, the social networking domain was analyzed and certain measuring difference were seen in their size which leave some doubt in mind that it may contains some extra and unusual functionalities.

**TABLE 4.2** Size variations in social networking domain apps

| Application | Size (MB) | Messaging | Call | Media |
|---|---|---|---|---|
| App1 | 24 | Y | Y | Y |
| App2 | 29 | Y | Y | Y |
| App3 | 32 | Y | Y | Y |
| App4 | 50 | Y | Y | Y |
| App5 | 56 | Y | Y | Y |

## 4.3. Implementation Using Fiddler as Man-in-the-Middle:

The following setup is built in which fiddler was installed on a windows machine and Man-in-the Middle setup was established to capture the traffic in Wi-Fi environment. Step by step procedure is given below:

a) Install fiddler on windows machine and connect that machine to DSL modem.
b) Setup a wireless hotspot on windows machine using netsh command.
c) Connect the Android phone to Wi-Fi on hotspot of windows machine.
d) Start fiddler on windows machine. All traffic of android phone is now routed through windows machine.
e) Download the fiddler certificate to phone and add it to certificates.
f) Launch mobile apps one by one from Android phone to communicate with outside world.
g) Traffic was captured on windows machine and analyzed.


**Figure 4.7** Network Setup Using Fiddler as proxy on Windows Machine

**Results:**

The following observations were made from the captured traffic:

a. A large number of requests per host were initiated. A number of Ad sites and analytics were connected. This confirms the third party transfers.

b. Since no advertisements were offered on app while using, all traffic being generated is suspicious.
c. A large number of data packets, basically javascripts were downloaded to mobile devices.
d. Possibly app has downloaded its malicious function onto device and may trigger it later on depending on the suitability of parameters or time triggered.
e. A number of apps were found to transmit user identity in terms of name, age, gender etc. Mobile operator information was also transmitted by some of apps.
f. Some of applications were seen to capture the phone book of users.

## 4.4. Implementation Using TaintDroid:

It is a system that provides dynamically taint tracking and analysis of applications simultaneously via tracking various sensitive data sources. By having leverage in Android's virtualized environment, one can do real-time analysis in TaintDroid. It is an extension to Android platform that helps in tracking private data through third party apps. The goal of ours is to detect what data is leaked and which app is helping it to do so. This system labels or taints the data that is leaked from sensitive sources. Whenever this tainted packet leaves the system, destination address, application responsible and type of data gets logged in TaintDroid system. Users get a better insight into what apps are actually working. Multiple granularity approach is followed by the TaintDroid system: file-level, method-level, variable-level and message-level. The multilevel approach architecture for taint tracking is given below and also procedure to do multi-level tracking is as follows:

1. VM interpreter is used for variable-level tracking in third party applications using variable semantics and only data is tainted.
2. Secondly, message-level tracking is performed, in which messages were tainted not the data in the messages.

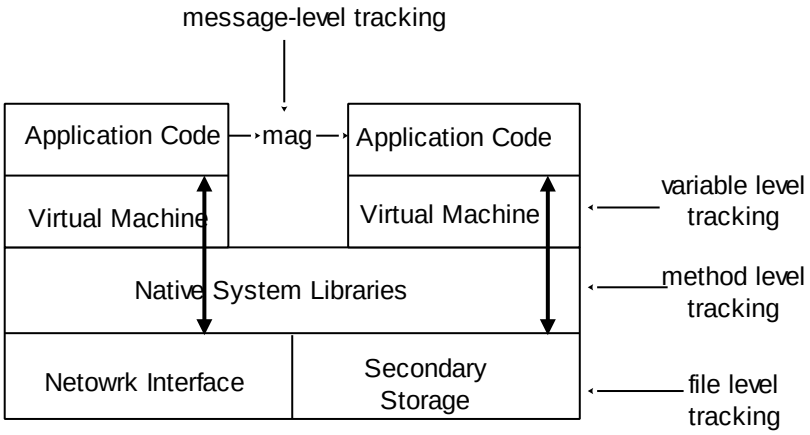



**Figure 4.8** The multilevel approach architecture for taint tracking

3. Next, method level tracking is done to run the native code and flow is tainted using flow semantics.
4. Lastly, file-level tracking is done to check the taint marking is conservatively retained or not on persistent information.

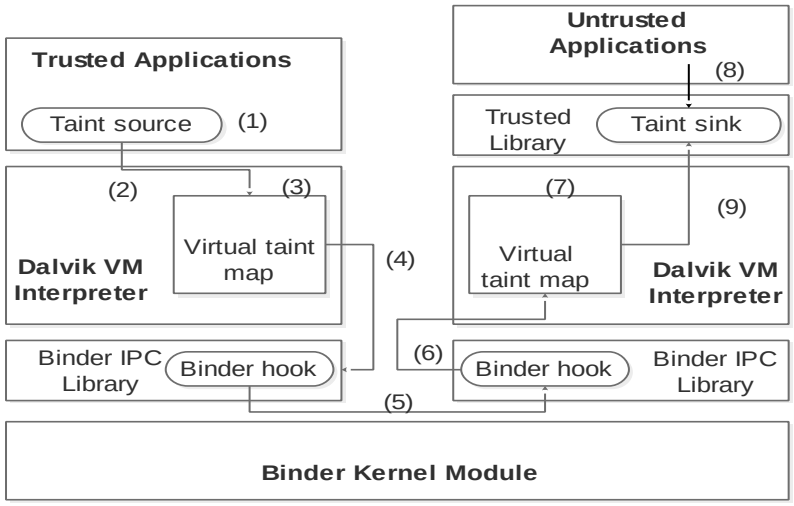



**Figure 4.9** TaintDroid Architecture

To use TaintDroid in our work, a Micromax phone with Android operating system was used. First of all, phone was rooted so that TaintDroid could get installed on that. Secondly, TaintDroid was installed on this phone with access to kernel level processes. Special changes were made in configuration to make it possible to capture the required traffic and communication made by apps.

The working of TaintDroid is shown as follows:

1. Using sufficient context information is tainted in a trusted app.
2. Using native methods taint marking is done in virtual taint map with the help pgf Dalvik VM interpreter.
3. Taint tags are propagated according to rules of data flow using Dalvik VM interpreter.
4. Taint tag is ensured by binder library in which taint tag reflects the combined taint marking of all data.
5. Then packet is transfer to remote untrusted application through kernel.
6. All read values are assigned the taint tag by binder library.
7. Taint tags are propagated to untrusted applications identically by remote Dalvik VM interpreter.
8. When taint sink library is invoked by untrusted application, it retrieves the taint tag and report the event.

**Results:**

a. It has seen that many applications access the various sensitive information of user's phone. Results are shown below in the table.

**TABLE 4.3** Permission requests made by applications

| Application Name | Permissions | | | |
|---|---|---|---|---|
| | Location | Camera | Audio | Phone State |
| App1 | Y | | | |
| App2 | Y | | | Y |
| App3 | Y | Y | | Y |
| App4 | | Y | | |
| App5 | Y | Y | Y | |

# Chapter 5

# CONCLUSION AND FUTURE SCOPE

## 5.1. Conclusion:

One of the severe problem faced by smartphone market is addressed in this work i.e. detection of privacy leakage in mobile applications. Unlike the approaches given earlier, a new idea was presented on basis of third party connections made by an application. In this work, it is argued that the transmission made by an app to a legitimate server or a third party server and data transmission made by an app did not violate the access permissions but it may affect the security and privacy of users severely. As per best of my knowledge no work has been done in this particular domain i.e. third party connection. It was presented with different domains of applications taking various parameters into account. This work also addressed users to educate more about their security and privacy. Before installing any application if user take proper care of what permission the app is requiring and making intelligent decision to allow it or not.

## 5.2. Future Scope:

Here in this work, only the detection mechanism has provided that shows that applications are stealing users' private information without their consent. To make it more effective one can make an application that deals with such issues like permission requests made by application requiring more than what it actually needs. Secondly taking actions to application which is making connections to outside world: third party servers. One can either terminate the connection or completely disabling the application. Also extra features can be added those show all the information transmitted by the application to any outside connection. An alarming functionality can be added which alarms user if any application tries to steal the private information and it asks the users that action should be allowed or not.

# REFERENCES


[1]. https://en.wikipedia.org/?title=Information_security

[2]. https://en.wikipedia.org/wiki/Network_security

[3]. https://www.paloaltonetworks.com/resources/learning-center/what-is-network-security.html

[4]. http://en.wikipedia.org/wiki/Global_Internet_usage

[5]. http://www.internetlivestats.com/internet-users/

[6]. http://en.wikipedia.org/wiki/List_of_countries_by_smartphone_penetration

[7]. http://www.google.com/mobile/maps/

[8]. Y. Zhou and X. Jiang. Dissecting android malware: Characterization and evolution, in *IEEE Symposium on Security and Privacy*, 2012.

[9]. M. Egele, C. Kruegel, E. Kirda, and G. Vigna. Pios: Detecting privacy leaks in iOS applications, In *NDSS*, 2011.

[10]. W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. Taintdroid: an information-flow tracking system for real time privacy monitoring on smartphones, in *OSDI*, pages 1–6, 2010.

[11]. P. Gilbert, B.-G. Chun, L. P. Cox, and J. Jung.Vision: automated security validation of mobile apps at app markets, in *Proc. MCS)*, 2011.

[12]. P. Hornyack, S. Han, J. Jung, S. Schechter, and D. Wetherall: These aren't the droids you're looking for: retrofitting android to protect data from imperious applications, in *CCS*, pages 639–652, 2011.

[13]. O. Tripp, M. Pistoia, S. J. Fink, M. Sridharan, and O. Weisman. Taj: effective taint analysis of web Applications, in *PLDI*, pages 87–97, 2009.

[14]. M. C. Grace, W. Zhou, X. Jiang, and A.-R. Sadeghi: Unsafe exposure analysis of mobile in-app advertisements in *WiSec*, 2012.

[15]. https://en.wikipedia.org/wiki/Mobile_security

[16]. https://en.wikipedia.org/wiki/Android_(operating_system)

[17]. https://en.wikipedia.org/wiki/IOS

[18]. https://en.wikipedia.org/wiki/BlackBerry_OS

[19]. https://en.wikipedia.org/wiki/Symbian

[20]. https://en.wikipedia.org/wiki/Windows_Mobile

[21]. https://en.wikipedia.org/wiki/Bada

[22]. https://en.wikipedia.org/wiki/Application_security

[23]. https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Top_Ten_Mobile_Risks

[24]. http://en.wikipedia.org/wiki/Rooting_(Android_OS)

[25]. C. Mann and A. Starostin: A Framework for Static Detection of Privacy Leaks in Android Applications, in SAC'12, pages 1457-1462, 2012.

[26]. Z. Yang, M. Yang, Y. Yang, G. Gu, P. Ning, X. Sean Wang: AppIntent: analyzing sensitive data transmission in android for privacy leakage detection, in *CCS'13,* pages 1043-1054, 2013.

[27]. S. Rosen, Z. Quin and Z. Morley, AppProfiler: A Flexible Method of Exposing Privacy-Related Behavior in Android Applications to End Users, in CODASPY'13, Pages 221-232, 2013.

[28]. R. Stevens, C.Gibler, J. Erickson and H. Chen: Investigating User Privacy in Android Ad Libraries, In Workshop on Mobile Security Technologies (MoST), 2012.

[29]. L. Batyuk, M. Herpich, S. Ahmet Camtepe,K. Raddatz, A. Derrick Schmidt and S. Albayrak: Using Static Analysis for Automatic Assessment and Mitigation of Unwanted and Malicious ActivitiesWithin Android Applications, in MALWARE ,pages 66-72,2011.

[30]. Igor Bilogrevic, K´evin Huguenin, Berker Agir, Murtuza Jadliwala and Jean-Pierre Hubaux, "Adaptive Information-Sharing for Privacy-Aware Mobile Social Networks", *UbiComp'13*, September 8–12, 2013.

[31]. W. Zhou, Y. Zhou, M. Grace, X. Jiang and S. Zou: Fast, Scalable Detection of "Piggybacked" Mobile Applications, *CODASPY'13,* February 18–20, 2013.

[32]. J. I. Hong and J. A. Landay, "An Architecture for Privacy-Sensitive Ubiquitous Computing", *MobiSys'04*, June 6–9, 2004.

[33]. P. Pearce, A. P. Felt, G. Nunez and D. Wagner: AdDroid: Privilege Separation for Applications and Advertisers in Android, ASIACCS '12, May 2–4, 2012.

[34]. R. Balebako, J. Jung and W. Lu: “Little Brothers Watching You:” Raising Awareness of Data Leaks on Smartphones, Symposium on Usable Privacy and Security (SOUPS) 2013, July 24–26, 2013.

[35]. I. Leontiadis, C. Efstratiou, M. Picon and C. Mascolo: Don’t kill my ads! Balancing Privacy in an Ad-Supported Mobile Application Market, HotMobile’12 February 28–29, 2012.

[36]. E. Chin, A. P. Felt, V. Sekar and D. Wagner: Measuring User Confidence in Smartphone Security and Privacy, Symposium on Usable Privacy and Security (SOUPS) 2012, July 11-13, 2012.

[37]. S. Ickin, K. Wac, M. Fiedler, L. Janowski, J. Hong and Anind K. Dey: Factors Influencing Quality of Experience of Commonly Used Mobile Applications, *Communications Magazine, IEEE* , vol.50, no.4, pp.48,56, April 2012.

[38]. N. li and G. Chen: Sharing Location in Online Social Networks, *Network, IEEE* , vol.24, no.5, pp.20,25, September-October 2010.

# LIST OF PUBLICATIONS

P. Kumar, M. Singh and V.P. Singh, Mobile Applications: "Analyzing Private Data leakage Using Third Party Connections", peer reviewed and accepted for publication in IEEE International Conference Advances in Computing, Communications & Informatics, ICACCI 2015, Aug 10-13,2015, Kochi, India.

# VIDEO LINK

**Link:** https://www.youtube.com/watch?v=Hbz7T_ldljo