AN INTEGRATED APPROACH FOR AUTHENTICATION AND ACCESS CONTROL WITH SMARTPHONES

by

Nitish Bhatia

Submitted in partial fulfilment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
November 2016

This work is dedicated to

My supervisor and well-wisher Dr. Srinivas Sampalli

and

My beloved parents Mr. Rakesh Bhatia and Mrs. Sitara Bhatia for their unconditional support

Table of Contents

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

With the enhancement in technology, people are relying more on mobile devices for accessing social media sites, online shopping or financial transactions. These devices are smart but the methods of authentication are password- or biometric- based, which make the authentication process either frustrating or expensive in terms of cost and computation. Weak passwords, and the absence of authentication mechanisms in mobile devices cause breach of security and privacy of the user data in case of physical theft. In order to mitigate this problem, we study an integrated approach that utilizes smartphone sensor data, network data, and usage data, in authentication schemes and access control mechanisms with smartphones. The proposed scheme switches between the implicit and explicit authentication depending on the usage patterns of the smartphone, at different time intervals of the day. Based on that, an access control mechanism has been proposed. We have also investigated how different weighting techniques play a role in determining the priority towards the sensor selection for user authentication. The proposed approach was tested on a publicly available dataset. The results show that our approach is viable and effective for authentication along with access control using smartphones.

LIST OF ABBREVIATIONS USED

API	Application Programming Interface
OTP	One Time Password
PIN	Personal Identification Number
IDE	Integrated Development Environment
M2M	Machine to Machine
BTS	Base Transceiver Station
BSC	Base Station Control
MSC	Mobile Switching Center
GSN	Gateway Support Node
HLR	Home Location Register
VLR	Visitor Location Register
AUC	Authentication Center
AP	Access Point
NIC	Network Interface Cards
WLAN	Wireless Local Area Network
BSS	Basic Service Set
ESS	Extended Service Set

ACKNOWLEDGEMENTS

I would first like to thank my supervisor Dr. Srinivas Sampalli. Without his support and guidance this thesis would have not been completed. He always steered me in the right direction and was available all the times whenever I had any doubt regarding my thesis.

I would also like to thank my colleagues from MyTech lab who helped me whenever I stuck somewhere and my friends who stood up by side whenever I needed them.

Finally, I must appreciate my parents and brother for providing me constant support and continuous encouragement throughout my years of study.

CHAPTER 1 INTRODUCTION

Smartphones are becoming an essential part in everyone's life. Be it for accessing social media sites, online shopping or financial transactions, smartphones are capable of handling these tasks seamlessly. With the increase in the number of smartphone users and their usage in almost all aspects of our daily lives, the chances of hackers attacking these devices have also increased. In addition, the mobile devices are more vulnerable to physical theft. If the owner of the device does not have an effective access control and protective measure, the information stored in the device can easily be compromised. Hence, it is very essential to have an authentication scheme which will keep the device secure at all times.

The current authentication methods used in smartphones are standard i.e. they are based on "What you know", "What you have" or "What you are". Each of these categories have some drawbacks and need user interaction with the device in order to prove his identity. Passwords, Personal Identification Number (PIN), or Unlock patterns in mobile devices are the best examples of "What you know" category. But people generally do not use this category or use easy to guess passwords or patterns since they prefer convenience over security [1]. In the category of "What you have", only One Time Password (OTP) can be used in smartphones. But it is cumbersome to use OTP for each and every application as well as for screen locks since OTP is mostly preferred as a second factor of authentication. Biometrics such as fingerprint, retina scan, voice recognition etc. belongs to the third category i.e. "What you are". These methods are highly secure as compared to others but the major drawback is, their high cost and complex computation which are not very feasible for devices with limited resources.

Many of the smartphone applications are programmed in such a way that they keep the user logged in for all the time until the user manually logs out of that application. These applications include email, social media or any other banking or payment related applications which are linked to our credit cards. Even after knowing that we have sensitive information stored in our mobile phones 36% of the users do not protect their phones with passwords or PIN [2] due to which the information can easily be compromised if an attacker gets hold of the phone.

Also with the increase in the processing power, RAM, and the integration of numerous sensors, the smartphones are capable of processing all the integrated sensor data but these sensors are mostly used for navigation and/or for exercise related applications. In this thesis we propose an authentication scheme which is based on the user profiles made from various sensor data. The motivation behind this research is based on the assumption that there is some stability in the user daily routine. For example, a user may follow the same route to the office every day, go for groceries at the same store, go to the same place for lunch, call the same person every day at the same time, use the same type of applications or perform the same activities like walking, running, driving a vehicle etc. at the same time. All this information can be collected using the sensors and can be used to make user profiles which can further be used to authenticate the user.

There is a lot of research that has been done in the field of implicit authentication. Kayacik et al. [3] in their study proposed a sensor based authentication scheme which has the capability of deciding when to shift from training to deployment mode. Also their scheme is capable of determining the detection threshold below which an explicit authentication will be triggered. Elaine et al. [4] in their study proposed an implicit authentication scheme based on behavior patterns. They calculate an authentication score based on the recent user activities. The score will increase in case of a good event and will decrease in case of bad event. By good event they mean the habitual events and negative event means the not usually observed behavior of that particular user.

In previous works, the authors have three major setbacks. The first being a fixed threshold for the entire day. There can be some scenarios in the entire day where a user might not do the same set of activities in the particular timestamp. The second setback is that the authors do not assign weights to the sensors in their works. There might be a possibility that some sensors would yield more information about a particular user but might not yield the same amount of information for another. The third setback is; the authors have not utilized the thresholds they obtained for each user to provide access control mechanism.

1.1 BRIEF INTRODUCTION OF THE PROPOSED APPROACH

In this thesis, we propose an approach of calculating different thresholds for different time bins and sensors. During the training phase the 24-hour data of the user will be divided into intervals of particular duration for different sensors and the data for two consecutive days will be matched in those time spans so that it will learn about the match percentage for every sensor. In the testing phase these percentages will be the threshold for that time window. If in any time bin the match percentage will be less than the threshold an explicit authentication will be triggered. Also in the training phase it will calculate the aggregate match percentage for each sensor in order to check the priority of sensor for a user. Depending upon the priority the weights will be assigned to the sensors. After this the aggregate match percentage will be calculated based on weighting scheme and those percentages will decide about the access control in different time bins.

Since tracking the user activity and accumulation of the sensor data is crucial for our proposed scheme, we have developed an android application which collects data from all the available sensors and detects the activities of the user like walking, running, in vehicle, on bicycle, still or tilting. All this information can be combined to make the user profiles. We are testing our approach on the GCU dataset [3]. It is a publicly available dataset and contains the data collected from cell towers, Wi-Fi networks and applications.

1.2 OUTLINE OF THE THESIS

In the rest of the thesis, Chapter 2 gives an overview of authentication, factors of authentication, sensors available in today's smartphones, architecture of cellular & Wi-Fi networks and access control. Chapter 3 involves detail about existing works in which sensor data has been used for different purposes. Chapter 4 explains the proposed methodology in detail. Chapter 5 discusses about the technical specifications of the implementation. This chapter also includes code snippets for each module with the detailed explanation. The in-depth description of the experiments conducted and the evaluated results supporting our proposed approach are contained in Chapter 6. We concluded this thesis with the limitations and future work in Chapter 7.

CHAPTER 2 BACKGROUND

In this chapter, we explained the terms and concepts used in this thesis. We begin with the discussion of authentication and its factors. Then we discuss about various sensors available in the smartphones. We also give brief introduction about the architecture of cellular and Wi-Fi networks followed by a brief discussion about the access control mechanisms used in current systems.

2.1 AUTHENTICATION

In daily life we come across so many situations where we need to prove our identity in order to gain access to any information. These situations can be digital or traditional. By digital situation, we refer to a scenario where one needs to enter his/her username & password or use a biometric in order to gain access to his/her computer, tablet, mobile, social media account, online banking etc. On the other hand, by traditional situation we refer to a scenario where one needs to show his/her Physical Identity Card in order to prove that he/she is the person he/she is claiming to be. For example, let us consider that Person A Sent Person B some money through a money transfer facility such as Western Union or Moneygram. The agent at the facility would ask Person B for an identity proof to ensure that he is the same person on whose name the reception of the money is authorized. So, in both the situations one needs to prove his/her identity to gain access to sensitive information. The process of determining whether a person is the one who he claims to be or not is termed as Authentication.

Before the authentication process, the credentials of the user are stored either in a database or locally in operating system or within an authentication server. Than as an authentication process these credentials are matched with the ones the user enters. If a match is found he will be considered as an authorized user and get access to the authorized information. [5] The basic type of authentication process is shown in Figure 1.

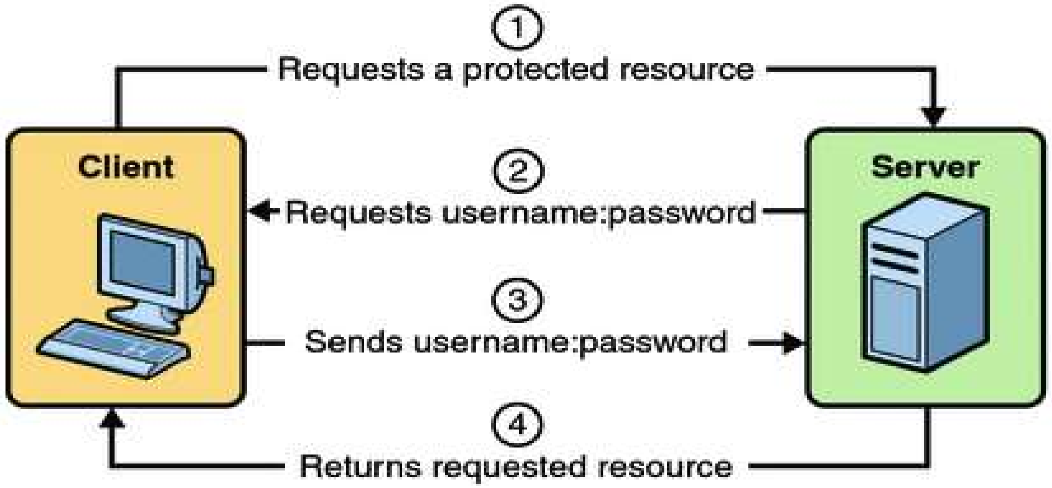

1
Requests a protected resource
Client
Server
2
Requests username:password
3
Sends username:password
4
Returns requested resource

Figure 1 Basic Authentication [6]

2.2 FACTORS OF AUTHENTICATION

2.2.1 Knowledge factor

This factor of authentication relies on something you know. The user needs to prove his knowledge to authenticate. The most common example is the use of password for getting access of any account you created. The other examples could be personal identification number (PIN) for ATM access, or answer to security question etc. [7]

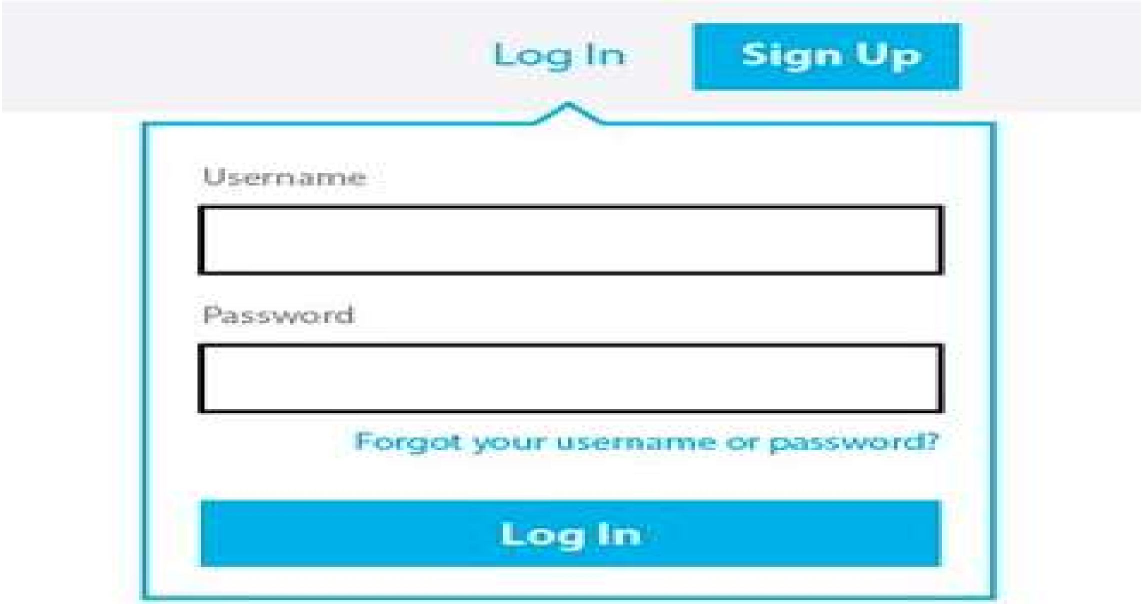


Figure 2 Knowledge Factor [8]

2.2.2 Possession factor

This factor of authentication relies on something you have. This factor of authentication can be understood as a lock and key mechanism. In order to open the lock, you must possess the key. One of the example is access cards used in offices to mark attendance or to open the doors. [7]

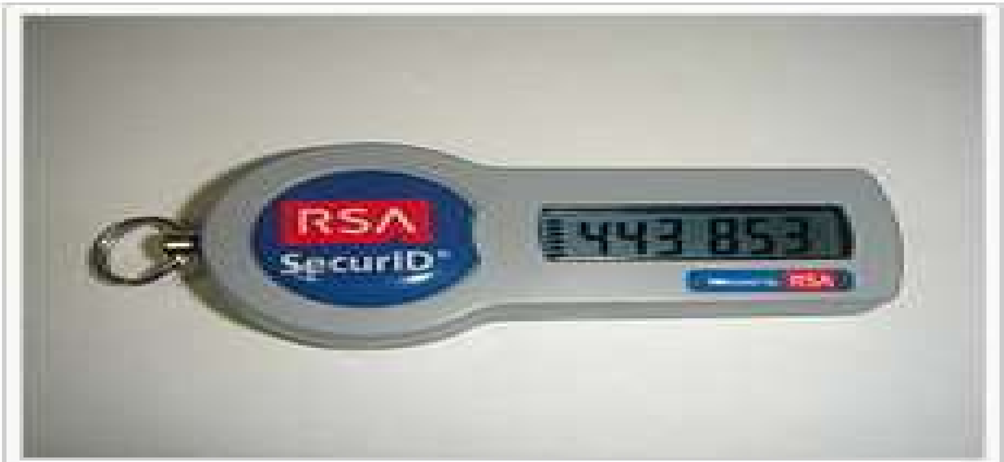


Figure 3 Token Generator [7]

2.2.3 Inherence factor

This factor of authentication relies on something you are. These are usually biometric methods which includes fingerprint, voice recognition, retina scan, etc. [7]

Figure 4 Fingerprint [9]

2.2.4 2-Step Verification

2-step verification is a process of adding an additional layer of security to your account. In this process you will first enter your password and than it requires you to enter an additional information. That information will not be something you already know but will be some unique string of numbers generated dynamically. That information will be given to you via email, text, call or an authenticator app. You will be given access to your account only after you enter that information correctly. Also there is a timestamp associated with that information after which that number will expire and you need to request a new number. So, with this process no one will be able to access your account even if they know your password. [10]

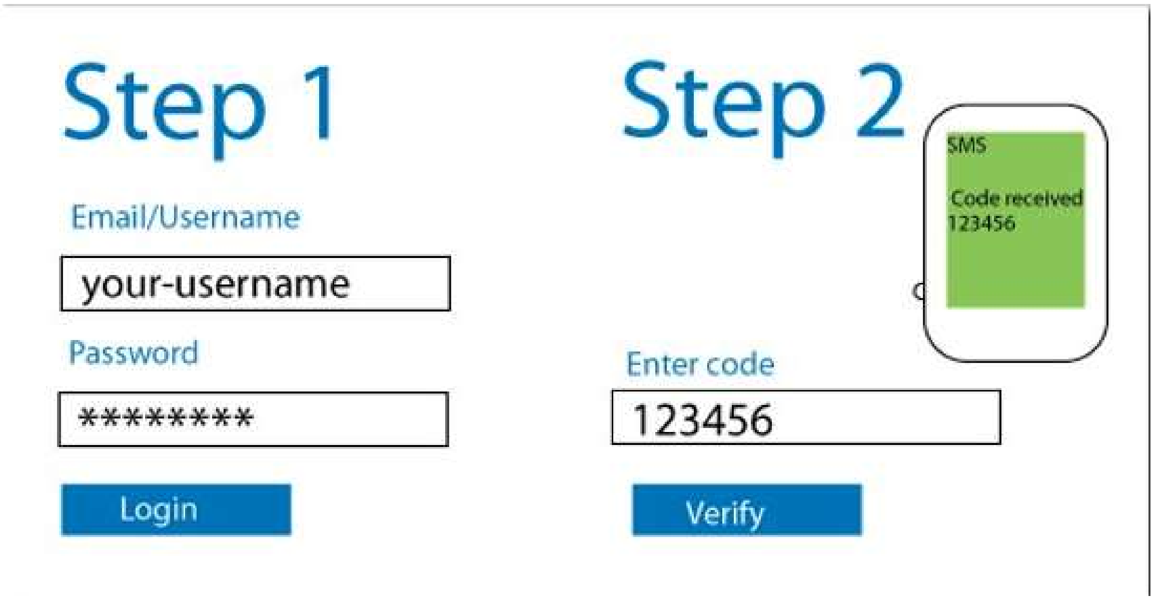


Figure 5 Two-Step Verification [8]

2.3 SENSORS

The sensor is a device that detects and responds to some type of input from the physical environment. The input could be light, heat, motion, moisture, pressure or any other environmental phenomenon. The output of the sensor is generally a signal that is converted to human-readable display or can be transmitted to a server for further processing. [11]

Today most of the smartphones have many built-in sensors capable of detecting motion, orientation and various environmental conditions. These sensors provide precise and accurate raw data which can further be processed to know about movement and positioning of the device or to get knowledge about the changing ambient environment. There are many applications which use the sensors to solve one or the other purpose. Let us take an example of a game. We always wonder that how the device movement will make the character of the game to move. This is achieved using the gravity sensors to infer the complex user gestures and motions, such as tilt, shake, rotation or swing. The other example is the weather application which uses the temperature and humidity sensor to calculate and report the dew point. There are 3 main categories of sensors supported by Android platform:

Motion Sensors

These sensors measure acceleration forces and rotational forces along three axes. This category includes accelerometers, gravity sensors, gyroscopes, and rotational vector sensors.

Environmental sensors

These sensors measure various environmental parameters, such as ambient air temperature and pressure, illumination, and humidity. This category includes barometers, photometers, and thermometers.

Position sensors

These sensors measure the physical position of a device. This category includes orientation sensors and magnetometers. [12]

Figure 6 shows the growth of sensors built in the smartphones in the past few years.

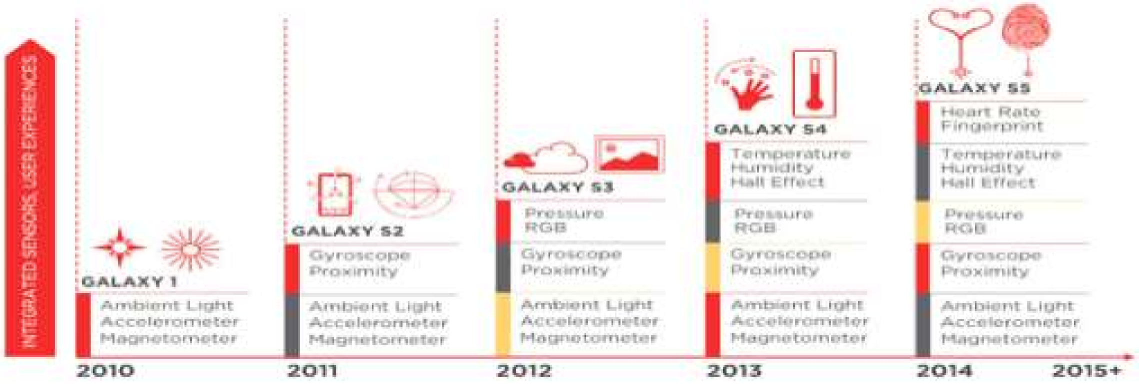


Figure 6 Sensor Growth in Smartphones [13]

2.4 ARCHITECTURE OF CELLULAR NETWORK

The cellular network is distributed over a geographical area divided into regions called cells. The wireless coverage within the cell is provided by a cell site which has a BSC (Base Station Control). BSC incorporates power sources, transmitter, receiver, antennas and interface equipment. One BSC serves many BTS (Base Transceiver Station) which is actually the cell tower and allows mobile phones to access the cellular network. Many cell sites within an area are served by one MSC (Mobile Switching Center). MSC is connected to the Class 5 switch of a telephone network. MSC is also connected to GSN (Gateway Support Node) which is connected to a router for accessing internet in the mobile devices. In the whole cellular network, the connection between the mobile equipment & BTS and BSC & BTS are wireless. The rest of the connections are wired.

Whenever a person uses the cell phone to make a call or receive a call, MSC is responsible for routing it from and to mobile stations. The components of MSC are:

- HLR (Home Location Register): In HLR the data related to each subscribed user is stored.
- VLR (Visitor Location Register): The exact location of the active users in an area served by MSC is stored in VLR database. The information about that user is deleted as soon as he/she leaves that area.
- AUC (Authentication Center): It is responsible for authentication and security services. [14]

The basic cellular network architecture is shown in Figure 7.

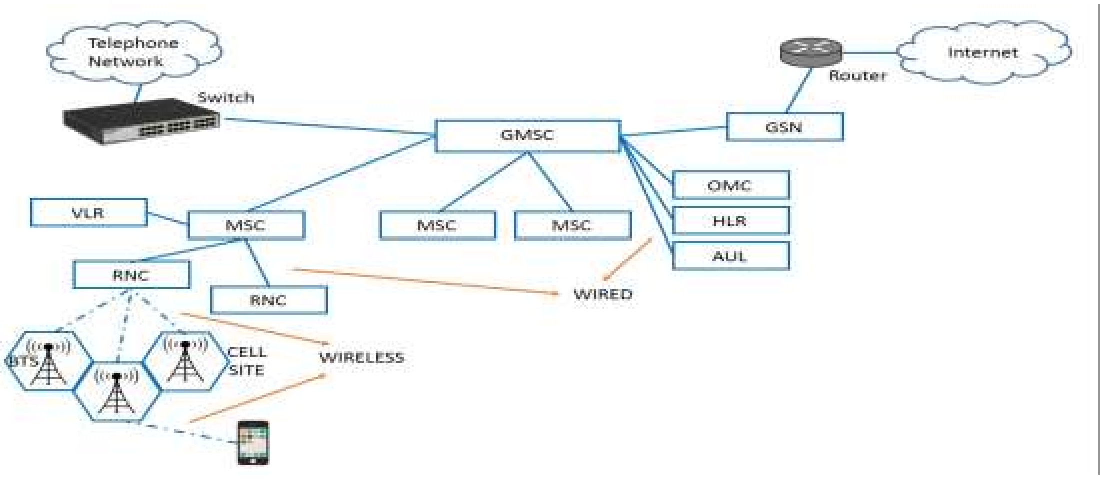


Figure 7 Cellular Network Architecture [14]

2.5 ARCHITECTURE OF WI-FI NETWORK

Wireless network most commonly referred as Wi-Fi is a technology in which radio signals of high frequencies are used to transmit data from one device to the other. A wireless network consists of several components that makes the communication possible through air medium. These components are called stations and are categorized as follows:

- Wireless Clients: Clients can be either mobile nodes such as smartphones, laptops, etc. or fixed nodes such as desktops, workstations etc. having Wi-Fi facility.
- AP (Access Points): This is a hardware device which facilitates the communication in wireless network by allowing a Wi-Fi compliant device to connect to the wired network.
- NIC (Network Interface Cards): The interface between the device and the wireless network infrastructure is provided by network interface cards. These cards can be fit inside the device or can be plugged in to the device like an external adapter.

In the whole WLAN (Wireless Local Area Network) architecture there can be many access points and clients connected to internet through those access points. The area covered by one access point is called BSS (Basic Service Set). In WLAN architecture there can be many BSS's connected together and are known as ESS (Extended Service Set). In ESS, the AP's connected by a distributed system to which facilitates communication among them [14]. The basic Wi-Fi network architecture is shown in Figure 8.

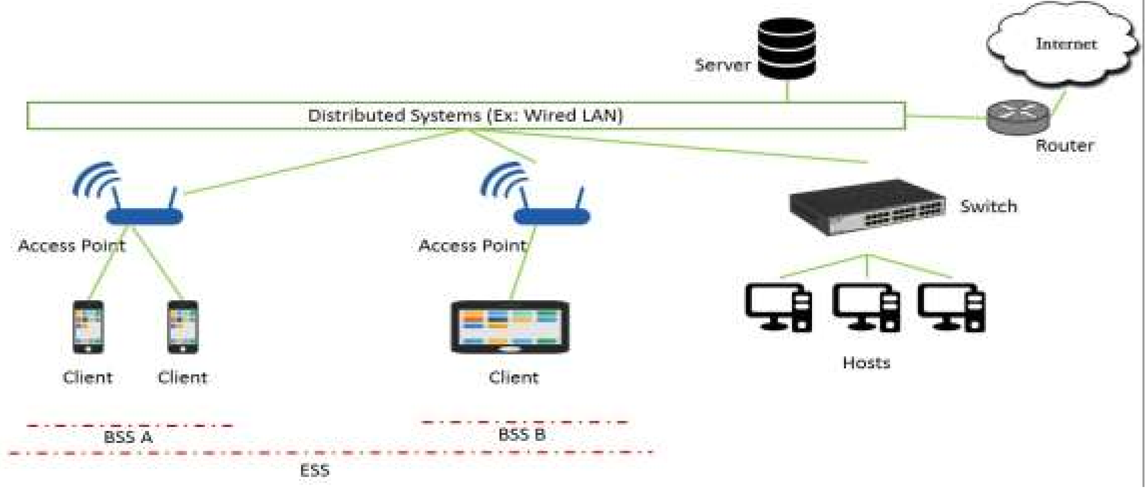


Figure 8 Wi-Fi Network Architecture [14]

2.6 Access Control

Access control is a security technique which can be used to set some rules for checking the operations an authorized user can perform on a system. In places like library, university, office, etc. the systems are shared by many people. In such scenarios, the access control is needed to put constrains onto which resources a legitimate user is allowed to access. Also the purpose of access control is to put restrictions on what action an application running by the user can perform [15]. The access control model can be built on the basis of many factors. One of the factors is, roles of the employees in an organization. So, based on the responsibilities of an employee, he/she will be given access to the respective resources. This is known as Role Based Access Control and is shown in Figure 9. In addition, Rule Based Access Control are used in various sectors, such as financial institutions.

Figure 9 Basic Access Control [16]

CHAPTER 3 RELATED WORK

In this chapter we present the work of other researchers in the field of smartphones. They used the data collected from smartphone sensors in variety of ways. Figure 10 gives an outline of the work which we are going to present in this chapter.

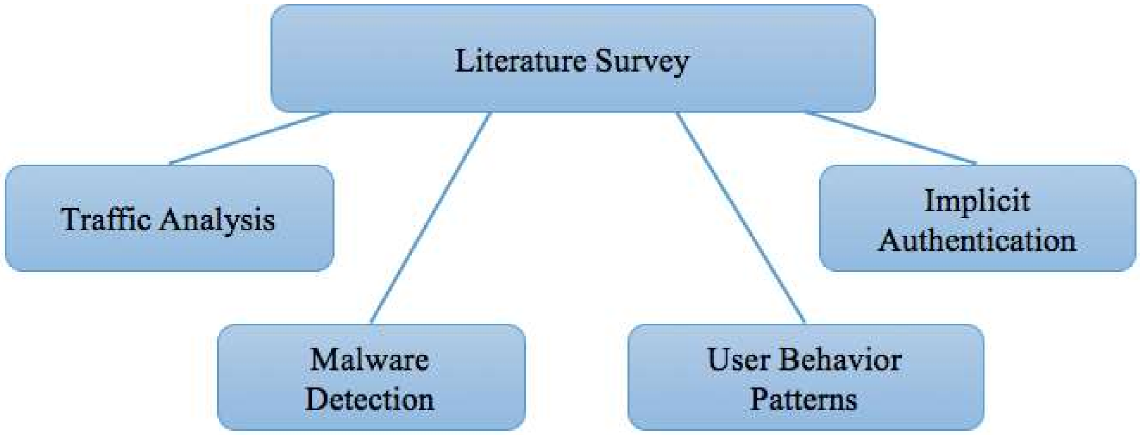


Figure 10 Outline of the Literature Survey

3.1 TRAFFIC ANALYSIS

Shafiq et al. [17] in their study compared cellular Machine-to-Machine (M2M) traffic such as telematics, smart metering, point-of-sale terminals, and home security and automation systems with traditional smartphone traffic. The purpose of their study is to know whether or not the traffic from M2M devices demands new cellular network design and management. They have collected traffic data from tier-1 cellular network in the United states and characterized M2M traffic from a wide range of perspectives, including temporal dynamics, device mobility, application usage, and network performance. They found from their results that the patterns of traffic coming from M2M devices are significantly different from smartphones.

Maier et al. [18] in their study focused on the usage of mobile hand-held devices (MHDs) from a network perspective. They conducted their research on anonymized packet level data collected from more than 20,000 residential DSL customers. Their results show MHDs are active on up to 3% of the monitored DSL lines. Most of the devices are used for multimedia content and for downloading mobile applications. Also they found from the traffic analysis that Apple devices such as IPhones and IPods are the commonly used devices.

Li et al. [19] used the data collected from a big cellular company to identify smartphone OS platform. They used TAC (Type Allocation Code) number which can be retrieved from the first eight digits of IMEI, for identifying the manufacturer and model of the device which can further be used to retrieve the OS information. The user browsing behavior is compared based on traffic dynamics and user application. From the results they found that iOS users generated maximum traffic volume than android and windows.

3.2 MALWARE DETECTION

Liu et al. [20] in their study focused on the detection of malwares in android based smartphones by analyzing the source code and behavior of the application. They combined the two techniques for this purpose, static and dynamic analysis. Static analysis is done without actually running the application. They analyzed the sensitive API of the source code by using data flow analysis and regular expression matching technology. Dynamic analysis is achieved by dynamic running the application in a software simulator and using the log analysis approach, the behavior of the application is monitored through the log output.

Eric et al. [21] presented a methodology for the analysis of application behavior based on the logs from android logging system. Log entries are mapped to bit vectors where each dimension is requested permission/action of the applications and services running on the device. They used Self-Organizing maps for analyzing the logs and generated the pattern of requested permission and performing actions of the applications which allows them to better understand the applications and analyze malware applications easily.

Burguera et al. [22] proposed a framework for malware detection by collecting traces based on crowdsourcing. For demonstrating their framework, they used two types of data sets: artificial malware and crowd collected malware. They designed an application Crowdroid which is a lightweight client for collecting data traces from real time users. This application monitors Linux kernel system calls and send it to the centralized server for processing.

3.3 User Behavior Patterns

Hirabe et al. [23] in their study demonstrates the logging of all the touch operations in an Android system. The motivation behind this study is the claim of the authors that with the user's touch operations his emotion, skill, etc. can be estimated. They used the statistical approach to analyze the frequency and speed of the 7 recognized touch operations.

Jie et al [24] implemented an automatic navigation system which can predict the future destination and route of the user based on the historical data. This system is developed by keeping in mind the problems of the existing navigation applications. One of these problems is that the existing navigation applications needs lot of operations by the user such as destination setting, zooming etc. which might be risky while walking or driving.

Bedogni et al. [25] in their study investigated that how the sensor data collected from a smartphone can be used to detect the user motion type. They collected the sample sensor data and labeled the patterns to each motion type. Based on the labeled data they compared the performance of different supervised algorithms for classifying the motion type. As a part of this study the motion recognition algorithm is integrated into an Android application and the information coming out of this application is being forwarded to other context aware applications. The authors also analyzed the factors which helps in improving the classification accuracy and at the same time maintaining a balance between the system performance and energy consumption.

Ma et al. [26] proposed a framework called MoodMiner which is capable of determining the mood of the user. The framework uses the data (Sensor and communication data) collected from the mobile phone to obtain pattern of the user behavior. The motivation behind this study is that the mood related mental health problems have huge impact on the quality of life and there is no convenient and easy technique for assessing & analyzing the mood of the person.

3.4 Implicit Authentication

Cheng et al [27] in their study presents an authentication framework SilentSense, which uses the features obtained from the touch behavior and the micro-movement of the device while screen-touch operations in order to differentiate between the owner and the attacker.

The authors assert that different users have unique way of using the smartphone and these unique ways can be learnt and detected from sensor data and touch events.

Yanzhi et al [28] in their study proposed a user verification system to detect possible user spoofing in mobile healthcare systems. The user verification is done based on the unique gait patterns which is obtained from the smartphone's accelerometer sensor data.

Weidong et al. [29] in their study proposed a framework 'SenGuard' for implicit authentication in smartphones. It captures data from multiple sensors available in the smartphones and use that as a source of authentication. In their initial prototype they use data from four sensors: voice, location, multi-touch and locomotion.

Kayacik et al. [3] in their study proposed a user behavior modelling technique for sensor based authentication. They are using temporal and spatial models for assigning a comfort score for sensor events at particular location and time. The comfort score depends on the frequency of occurrence of an event. Once the user profile is created they are comparing the current user behavior with the profile. If the behavior of the user diverges sufficiently from the profile, an explicit authentication can be triggered. In their approach the device will automatically switch from the training mode to the deployment mode when it learns the user behavior. The other key thing in their approach is that the device will decide on its own the detection threshold.

Elaine et al. [4] in their study proposed an implicit authentication scheme based on behavior patterns. They calculate an authentication score based on the recent user activities. The score will increase in case of a good event and will decrease in case of bad event. By good event they mean the habitual events and negative event means the not usually observed behavior of that particular user.

Yao et al. [30] in their study proposed an event-driven implicit authentication scheme. They investigated standard deviation and EWMA (exponentially weighted moving average) based algorithms for computing the threshold. With this method of threshold computation, their designed scheme is adaptive to shifts in user behavior. Their approach needs minimal training and since its event-driven there is no requirement of continuously running in the background.

Buriro et al. [31] in their study proposed a multi-modal behavioral biometric scheme for user authentication in smartphones. For their approach they are using the features collected when the user answers a call. These features include slide swipe, arm movement in bringing the phone close to the ear and voice recognition. They are comparing different classification algorithms, they used for their approach.

3.5 Motivation and Research problem

From the literature survey we explored, we found that existing approaches used the smartphone sensors or its usage data in a variety of ways. The cellular network data was used to know differences in M2M traffic, to get information about operating system of smartphones, and mostly for what purposes user use smartphones [17 - 19]. The application behavior data and log entries in smartphones are utilized to detect malwares in mobile devices [20 - 22]. By using touch operations data, emotion of the user is predicted and navigation history is used to predict the future route or destination [23, 24]. Smartphone sensor data is also used by some researchers to determine user's movement type or mood [25, 26]. There is also a lot of research in the field of using smartphone sensor or usage data for user verification and authentication [3, 4] [27 - 31]. Some researchers used touch operations, micro movements and gait patterns [27] [28] for this purpose.

As far as we have surveyed, none of the researchers' have achieved an accuracy with which they can assure that their technique is showing high degree of reliability. Also some of their techniques have a drawback. The authors consider the score obtained after analyzing the data for the entire day. But there is a high possibility that there might be time slots which achieve a score of 100% alongside with other time slots which are achieving a score as low as 40% but the score for the entire day might yield close to 55%. Now, impersonating a user with a score of 55% is relatively easier than impersonating a user with score 95%. Thus we believe that, calculating the score for the entire day by aggregating scores of all time slots, is more susceptible to impersonation when compared to that of the score of individual time slots.

Observing all the limitations in earlier studies, a new approach has been proposed for creating the user profiles based on the smartphone usage.

- Instead of creating user profile based on the data collected from entire day we are considering small time intervals. This will give different threshold for each time interval rather than giving one threshold for complete day.

- We are assigning weights to the sensor according to their importance in terms of the information they give for a particular user. The importance will be grounded on the steadiness of the user in using that sensor.
- Based on the threshold for any time bin the user will be given access to different applications in the device.

CHAPTER 4 AN INTEGRATED APPROACH FOR AUTHENTICATION AND ACCESS CONTROL

4.1 PROPOSED APPROACH

The authentication scheme proposed in this thesis is based on the user behavior. There are millions of smartphone users and the way everyone uses their smartphones is unique. People tend to follow a certain routine like going to office following the same route, going to same restaurant for lunch, staying in office for a particular time, connecting to same Wi-Fi signal everyday, using certain type of applications, etc. All this information can be gathered from the smartphone of a user and can be used to build the user profile. There are different methods of creating user profiles as discussed in Chapter 3. But all those methods are creating profile of the users for the entire day. But in our approach we are also considering the fact that there could be certain time bins in the entire day in which there will be no stability in the user behavior. So, we are creating user profiles for short time intervals instead of the entire day.

We are also considering the fact that not all users use all the features or sensors of the smartphones. Some users might show stability in using applications or connecting to same Wi-Fi signals whereas others might show stability for location or performing some activity like running or walking at fixed time. So, the priority of sensors in terms of the information it can provide for a user varies. Considering this, in our approach we are investigating weighting techniques to assign weights to different sensors. We suggested two weighting schemes for calculating the weighted average of match percentage for all sensors and compared them with the simple average.

Furthermore, based on the match percentage in different time bins, the access control mechanism can be implemented. For example, if in certain time interval the match percentage is approaching 100% the user will have full access to the device, if its 50% the user will not be able to access secure applications like banking or corporate mails but he will be able to access less secure applications like games etc. And if its near to 0% than it will always be explicit authentication. That would be the benefit of considering different threshold for different time bins.

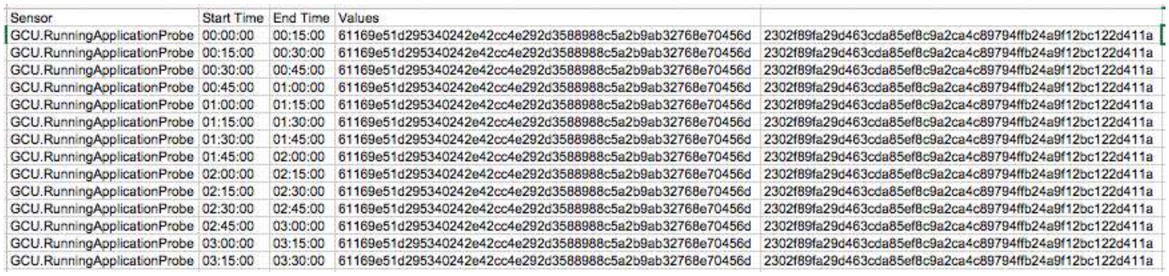
4.2 Proposed Scheme For Authentication

4.2.1 Sensor Data Collection

As a part of this thesis we developed an android application which collects data from various sensors available in the smartphones. This data includes values from accelerometer, light, pressure, GPS, Wi-Fi signals, etc. Also this application is capable of recognizing the user activities like walking, running, on foot, in vehicle, still, tilting or on bicycle. But we have not used this application to actually collect the real time data. Instead we test our approach on GCU dataset Version 1 [3]. This dataset contains anonymized data from Wi-Fi networks, cell towers and application use. It was collected in 2013 from 7 users consisting of staff and students of Glasgow Caledonian University. Android devices are used to collect this data. We are using 2 weeks' data from all the 7 users to test our scheme.

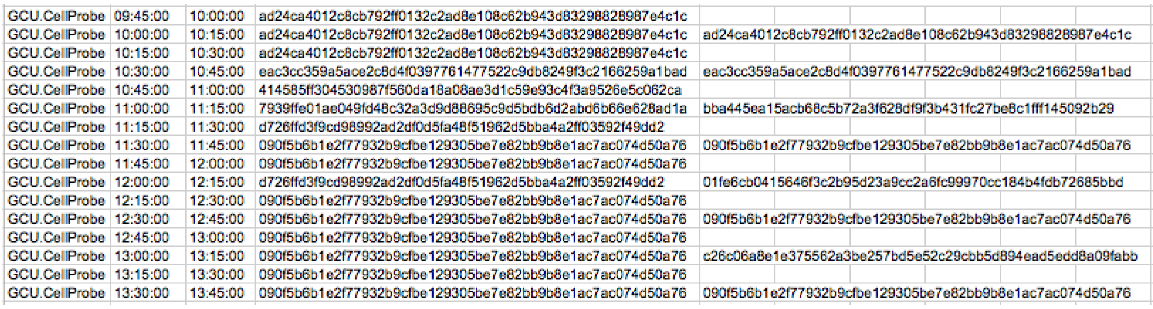
4.2.2 Dividing data into time bins

The original data was collected at a sampling rate of approximately 10 seconds for each sensor. We group the data from each sensor into time bins of 15 minutes. We choose 15 minutes' time bin for testing but it can be made flexible. After this the data for 2 consecutive days was matched in the respective time bins and the match percentage was calculated.



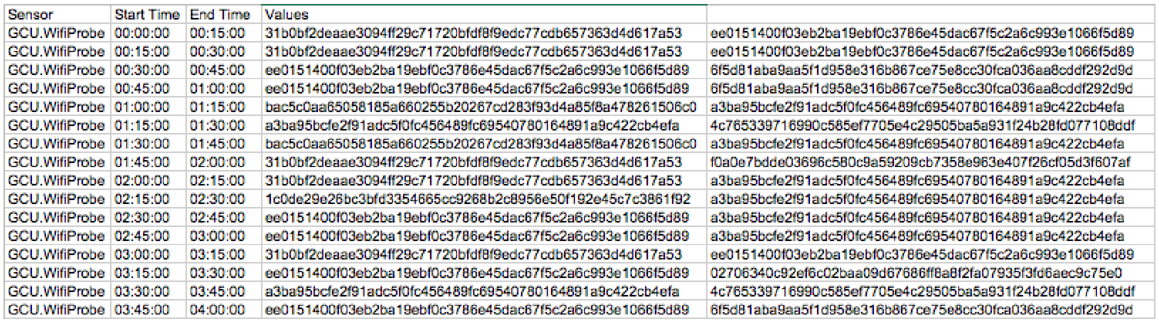
Sensor	Start Time	End Time	Values	
GCU.RunningApplicationProbe	00:00:00	00:15:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	00:15:00	00:30:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	00:30:00	00:45:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	00:45:00	01:00:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	01:00:00	01:15:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	01:15:00	01:30:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	01:30:00	01:45:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	01:45:00	02:00:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	02:00:00	02:15:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	02:15:00	02:30:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	02:30:00	02:45:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	02:45:00	03:00:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	03:00:00	03:15:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a
GCU.RunningApplicationProbe	03:15:00	03:30:00	61169e51d295340242e42cc4e292d3588988c5a2b9ab32768e70456d	2302f89fa29d463cda85ef8c9a2ca4c89794ffb24a9f12bc122d411a

Figure 11 Data from Application Usage



GCU.CellProbe	09:45:00	10:00:00	ad24ca4012c8cb792ff0132c2ad8e108c62b943d83298828987e4c1c	
GCU.CellProbe	10:00:00	10:15:00	ad24ca4012c8cb792ff0132c2ad8e108c62b943d83298828987e4c1c	ad24ca4012c8cb792ff0132c2ad8e108c62b943d83298828987e4c1c
GCU.CellProbe	10:15:00	10:30:00	ad24ca4012c8cb792ff0132c2ad8e108c62b943d83298828987e4c1c	
GCU.CellProbe	10:30:00	10:45:00	eac3cc359a5ace2c8d4f0397761477522c9db8249f3c2166259a1bad	eac3cc359a5ace2c8d4f0397761477522c9db8249f3c2166259a1bad
GCU.CellProbe	10:45:00	11:00:00	414585ff304530987f560da18a08ae3d1c59e93c4f3a9526e5c062ca	
GCU.CellProbe	11:00:00	11:15:00	7939ffe01ae049fd48c32a3d9d88695c9d5bdb6d2abd6b66e628ad1a	bba445ea15acb68c5b72a3f628df9f3b431fc27be8c1fff145092b29
GCU.CellProbe	11:15:00	11:30:00	d726ffd3f9cd98992ad2df0d5fa48f51962d5bba4a2ff03592f49dd2	
GCU.CellProbe	11:30:00	11:45:00	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76
GCU.CellProbe	11:45:00	12:00:00	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76	
GCU.CellProbe	12:00:00	12:15:00	d726ffd3f9cd98992ad2df0d5fa48f51962d5bba4a2ff03592f49dd2	01fe6cb0415646f3c2b95d23a9cc2a6fc99970cc184b4fdb72685bbd
GCU.CellProbe	12:15:00	12:30:00	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76	
GCU.CellProbe	12:30:00	12:45:00	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76
GCU.CellProbe	12:45:00	13:00:00	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76	
GCU.CellProbe	13:00:00	13:15:00	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76	c26c06a8e1e375562a3be257bd5e52c29cbb5d894ead5edd8a09fabb
GCU.CellProbe	13:15:00	13:30:00	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76	
GCU.CellProbe	13:30:00	13:45:00	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76	090f5b6b1e2f77932b9cfbe129305be7e82bb9b8e1ac7ac074d50a76

Figure 12 Data from Cell Tower



Sensor	Start Time	End Time	Values	
GCU.WifiProbe	00:00:00	00:15:00	31b0bf2deaae3094ff29c71720bfdf8f9edc77cdb657363d4d617a53	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89
GCU.WifiProbe	00:15:00	00:30:00	31b0bf2deaae3094ff29c71720bfdf8f9edc77cdb657363d4d617a53	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89
GCU.WifiProbe	00:30:00	00:45:00	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89	6f5d81aba9aa5f1d958e316b867ce75e8cc30fca036aa8cddf292d9d
GCU.WifiProbe	00:45:00	01:00:00	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89	6f5d81aba9aa5f1d958e316b867ce75e8cc30fca036aa8cddf292d9d
GCU.WifiProbe	01:00:00	01:15:00	bac5c0aa65058185a660255b20267cd283f93d4a85f8a478261506c0	a3ba95bcfe2f91adc5f0fc456489fc69540780164891a9c422cb4efa
GCU.WifiProbe	01:15:00	01:30:00	a3ba95bcfe2f91adc5f0fc456489fc69540780164891a9c422cb4efa	4c765339716990c585ef7705e4c29505ba5a931f24b28fd077108ddf
GCU.WifiProbe	01:30:00	01:45:00	bac5c0aa65058185a660255b20267cd283f93d4a85f8a478261506c0	a3ba95bcfe2f91adc5f0fc456489fc69540780164891a9c422cb4efa
GCU.WifiProbe	01:45:00	02:00:00	31b0bf2deaae3094ff29c71720bfdf8f9edc77cdb657363d4d617a53	f0a0e7bdde03696c580c9a59209cb7358e963e407f26cf05d3f607af
GCU.WifiProbe	02:00:00	02:15:00	31b0bf2deaae3094ff29c71720bfdf8f9edc77cdb657363d4d617a53	a3ba95bcfe2f91adc5f0fc456489fc69540780164891a9c422cb4efa
GCU.WifiProbe	02:15:00	02:30:00	1c0de29e26bc3bfd3354665cc9268b2c8956e50f192e45c7c3861f92	a3ba95bcfe2f91adc5f0fc456489fc69540780164891a9c422cb4efa
GCU.WifiProbe	02:30:00	02:45:00	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89	a3ba95bcfe2f91adc5f0fc456489fc69540780164891a9c422cb4efa
GCU.WifiProbe	02:45:00	03:00:00	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89	a3ba95bcfe2f91adc5f0fc456489fc69540780164891a9c422cb4efa
GCU.WifiProbe	03:00:00	03:15:00	31b0bf2deaae3094ff29c71720bfdf8f9edc77cdb657363d4d617a53	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89
GCU.WifiProbe	03:15:00	03:30:00	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89	02706340c92ef6c02baa09d67686ff8a8f2fa07935f3fd6aec9c75e0
GCU.WifiProbe	03:30:00	03:45:00	a3ba95bcfe2f91adc5f0fc456489fc69540780164891a9c422cb4efa	4c765339716990c585ef7705e4c29505ba5a931f24b28fd077108ddf
GCU.WifiProbe	03:45:00	04:00:00	ee0151400f03eb2ba19ebf0c3786e45dac67f5c2a6c993e1066f5d89	6f5d81aba9aa5f1d958e316b867ce75e8cc30fca036aa8cddf292d9d

Figure 13 Data from Wi-Fi Networks

4.2.3 Comparing Data

To compare data between two consecutive days in every time bin we used Union and Intersection. Union gives us all the distinct values in the particular bin and intersection gives us the common values in that bin. We are then calculating the match percentage as follows:

Let us consider, in Day D_1, time bin T_1 applications running in the phone are A, B & C and in Day D_2, time bin T_1 applications running are B, C & D. So,

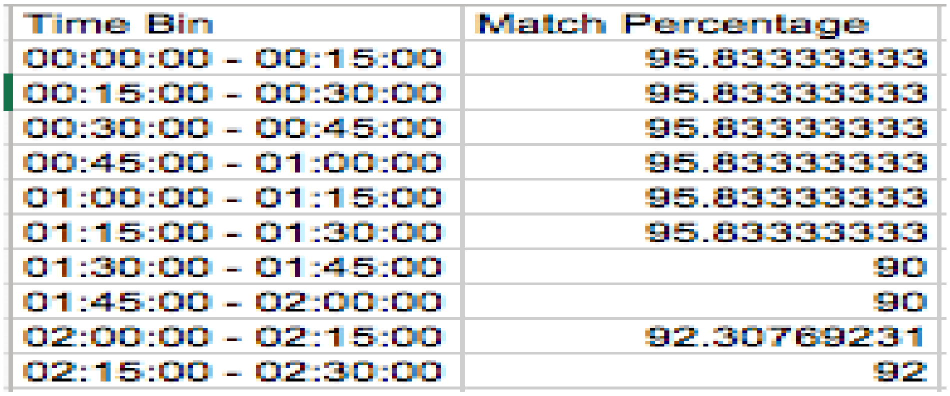
$$D_1 \cup D_2 = \{A, B, C, D\} = 4$$

$$D_1 \cap D_2 = \{B, C\} = 2$$

$$Match\ Percentage = (D_1 \cap D_2) \div (D_1 \cup D_2)$$

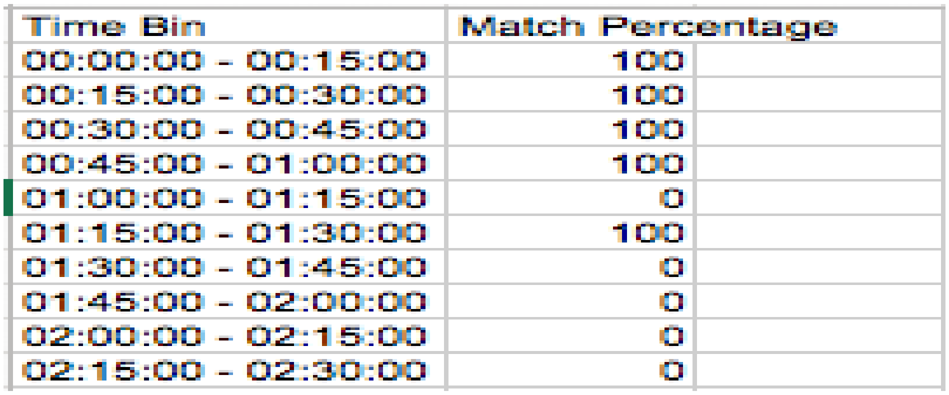
$$= (2 \div 4) \times 100$$

$$= 50\ \%$$



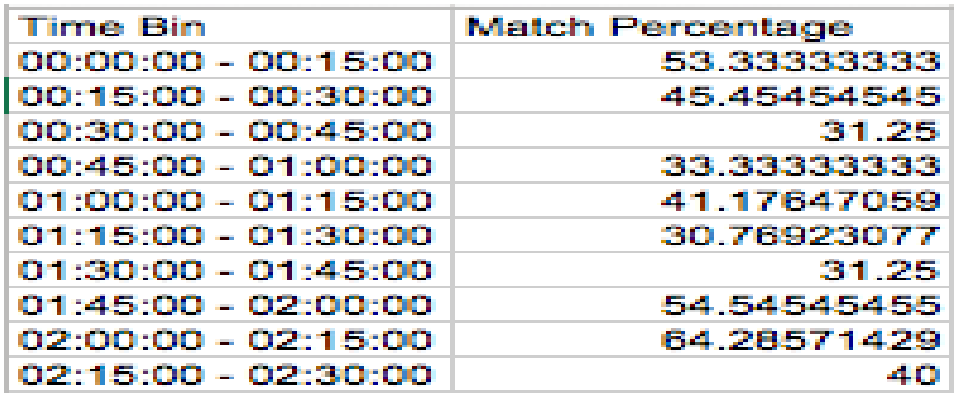
Time Bin	Match Percentage
00:00:00 - 00:15:00	95.83333333
00:15:00 - 00:30:00	95.83333333
00:30:00 - 00:45:00	95.83333333
00:45:00 - 01:00:00	95.83333333
01:00:00 - 01:15:00	95.83333333
01:15:00 - 01:30:00	95.83333333
01:30:00 - 01:45:00	90
01:45:00 - 02:00:00	90
02:00:00 - 02:15:00	92.30769231
02:15:00 - 02:30:00	92

Figure 14 Match Percentage for Application Usage



Time Bin	Match Percentage	
00:00:00 - 00:15:00	100	
00:15:00 - 00:30:00	100	
00:30:00 - 00:45:00	100	
00:45:00 - 01:00:00	100	
01:00:00 - 01:15:00	0	
01:15:00 - 01:30:00	100	
01:30:00 - 01:45:00	0	
01:45:00 - 02:00:00	0	
02:00:00 - 02:15:00	0	
02:15:00 - 02:30:00	0	

Figure 15 Match Percentage for Cell Tower



Time Bin	Match Percentage
00:00:00 - 00:15:00	53.33333333
00:15:00 - 00:30:00	45.45454545
00:30:00 - 00:45:00	31.25
00:45:00 - 01:00:00	33.33333333
01:00:00 - 01:15:00	41.17647059
01:15:00 - 01:30:00	30.76923077
01:30:00 - 01:45:00	31.25
01:45:00 - 02:00:00	54.54545455
02:00:00 - 02:15:00	64.28571429
02:15:00 - 02:30:00	40

Figure 16 Match Percentage for Wi-Fi Networks

4.2.4 Weighting Schemes

In the training phase based on the match percentage, the device will learn that for which sensor there is more stability in the usage for any user. The higher the match percentage, the more is the stability. Herein, stability refers to the consistent usage of a sensor by a user. For example, if the match percentage for application usage in any time bin is 100% it means that user is using the same applications daily in that particular time bin. So, based on these percentages the device will give priorities to the respective sensors in the testing phase. Based on those priorities, the weights will be assigned. We are proposing two weighting schemes for this purpose but this is flexible and any weighting scheme can be applied on our approach.

Weighting Scheme 1:

This weighting scheme is based on the number of values collected from any sensor. When the data between two consecutive days will be matched the number of values in each time bin for every sensor in the previous day will be considered for this weighting scheme. The Maximum, Average and Minimum of those values will be calculated and will be used as weights for the sensors. If a sensor has highest priority the weight for that sensor will be its Maximum, the weight for second sensor will be its average, and so on. Let us consider the following matrix represents the sensors and its maximum, average and minimum values:

$$\begin{pmatrix} \text{Sensor 1} & \text{Max 1} & \text{Average 1} & \text{Min 1} \\ \text{Sensor 2} & \text{Max 2} & \text{Average 2} & \text{Min 2} \\ \text{Sensor 3} & \text{Max 3} & \text{Average 3} & \text{Min 3} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \text{Sensor n} & \text{Max n} & \text{Average n} & \text{Min n} \end{pmatrix}$$

Suppose, the match percentage for Sensor 1 = S_1MP

Sensor 2 = S_2MP

Sensor 3 = S_3MP

And, $S_1MP > S_2MP > S_3MP$

Then,

$$Weighted\ Average = \frac{\{(S_1\mathrm{MP} * \ \mathrm{Max}\ 1) + (S_2\mathrm{MP} * Average\ 2) + (S_3\mathrm{MP} * Min\ 3)\}}{\{(Max\ 1) + (Average\ 2) + (Min\ 3)\}}$$

Since in this dataset we had only 3 sensors so we provided the calculation by considering only those sensors. But this weighting scheme can be extended for n sensors in following way:

Suppose, the match percentage for Sensor 1 = S_1MP

Sensor 2 = S_2MP

Sensor 3 = S_3MP

…

…

Sensor n = S_nMP

And, $S_1MP > S_2MP > S_3MP > \ldots\ldots\ldots > S_nMP$

Then,

$$Weighted\ Average = \frac{\left\{\begin{matrix}(S_1MP * \text{ Max } 1) + (S_2MP * Average\ 2) + (S_3MP * Average\ 3) + \cdots \ldots \ldots + \\ \left(S_{n-1}MP * Average\ (n-1)\right) + (S_nMP * Min\ n)\end{matrix}\right\}}{\{(Max\ 1) + (Average\ 2) + (Average\ 3) + \cdots \ldots \ldots + Average(n-1) + (Min\ n)\}}$$

Weighting Scheme 2:

This weighting scheme is based on the number of sensors. Let us consider there are n sensors. The sensor with highest match percentage will get the weight n, second highest will get (n-1), etc. down to 1.

$$Weighted\ Average = \frac{\{(S_1MP * \ n) + (S_2MP * (n-1)) + \cdots \ldots \ldots \ldots \ldots + (S_nMP * 1)\}}{\{n + (n-1) + \cdots \ldots \ldots \ldots + 1\}}$$

where, $S_1MP > S_2MP > \cdots \ldots \ldots \ldots > S_nMP$

Suppose, there are three sensors in the device and match percentages of these sensors for User X in time bin T are denoted as S_1MP, S_2MP, and S_3MP where,

$S_1MP > S_2MP > S_3MP$

then the weighted average for this user in time bin T will be calculated as:

$$Weighted\ Average = \frac{\{(S_1MP * \ 3) + (S_2MP * 2) + (S_3MP * 1)\}}{\{3 + 2 + 1\}}$$

4.3 Proposed Scheme For Access Control

Once the weighted average is calculated for every time bin, those values will act as the deciding factor for access control. If the value is reasonably high, the user will be having complete access to the device. If the value is very low, for instance, close to zero, then there will always be an explicit authentication for that particular time interval. And if the the match is in between, for instance, near to 50% then the user will have limited access to the device based on the pre-set security settings.

Figure 17 is the diagrammatic representation of the proposed scheme for access control. In this figure S_1, S_2, ……, S_n represents different sensors in smartphones. D_1, D_2, ……, D_n represents different days. T_1, T_2, ……, T_n represents different time bins in a particular day and WA represents the weighted average as discussed in section 4.2.

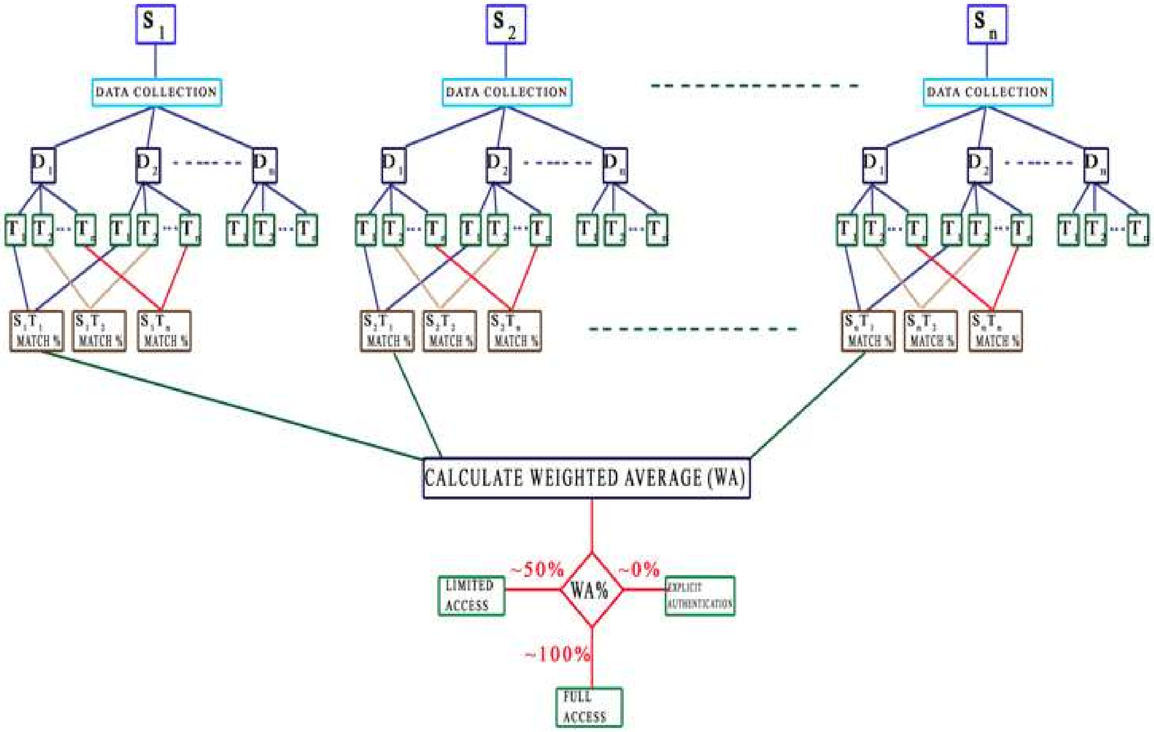


Figure 17 Proposed Scheme for Access Control

CHAPTER 5 IMPLEMENTATION

5.1 DEVELOPMENT ENVIRONMENT AND LIBRARIES USED

The proposed methodology is implemented in Java and run under MAC operating system. The sensor data collection application is developed using Android in Android Studio and run under Android platform in Samsung Galaxy S6 device. Table 1 shows the development environment used for implementing the proposed approach.

Programming Language	Java 1.8.0_101
Android Application Minimum Version	Android 4.0.3
Android Application Target Version	Android 6.0
Operating System	Mac OS 10.12
Device Android OS Version	Android 5.0.1

Table 1 Development Environment for Proposed Approach

The data collected from the users are saved in an excel sheet. So, open source java library is used to read data from those sheets. Also to recognize the activities (running, walking, etc.) of the user an open source google API is used. All those libraries are listed in Table 2.

Recognizing Activities of User	GoogleActivityRecognition API
Reading Data from Excel Sheets	JXL.jar

Table 2 Libraries Used

The IDE's used for writing, editing, building and debugging the code are listed in Table 3.

Java Programming	Netbeans 8.0.2
Android Application	Android Studio 1.4

Table 3 IDE used

5.2 IMPLEMENTATION DETAILS OF EACH PROCESS IN THE PROPOSED APPROACH

5.2.1 Sensor Data Collection Using an Android Application

As discussed in section 4.2 the first step in this approach is to collect data from the sensors in the user's device. This is achieved using an android application, the code snippet for which is as follows:

In order to activate any sensor available in the Android operated smartphone, SensorManager class is used. Following is the code which will first create the object of SensorManager class and then it connects to the sensor service by calling getSystemService function and passing sensor service as a parameter to that function.

```
private SensorManager sensor_manager;
sensor_manager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
```

After getting the sensor service, the sensor will be instantiated by using getDefaultSensor function and passing the type of sensor we want to instantiate. Sensor is the class used to create an instance of a specific sensor.

```
public Sensor pressure_sensor;
public Sensor light_sensor;
public Sensor accelerometer_sensor;
public Sensor proximity_sensor;
public Sensor gravity_sensor;
public Sensor linearAcceleration_sensor;
public Sensor gyroscope_sensor;
public Sensor orientation_sensor;
public Sensor rotationVector_sensor;
public Sensor ambientTemperature_sensor;
public Sensor gps_sensor;
public Sensor magnetometer_orient_sensor;
public Sensor accelerometer_orient_sensor;
```

```
pressure_sensor = sensor_manager.getDefaultSensor(Sensor.TYPE_PRESSURE);
light_sensor = sensor_manager.getDefaultSensor(Sensor.TYPE_LIGHT);
accelerometer_sensor =
sensor_manager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
proximity_sensor = sensor_manager.getDefaultSensor(Sensor.TYPE_PROXIMITY);
gravity_sensor = sensor_manager.getDefaultSensor(Sensor.TYPE_GRAVITY);
linearAcceleration_sensor =
sensor_manager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION);
gyroscope_sensor = sensor_manager.getDefaultSensor(Sensor.TYPE_GYROSCOPE);
ambientTemperature_sensor =
sensor_manager.getDefaultSensor(Sensor.TYPE_AMBIENT_TEMPERATURE);
rotationVector_sensor =
sensor_manager.getDefaultSensor(Sensor.TYPE_ROTATION_VECTOR);
accelerometer_orient_sensor =
sensor_manager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
magnetometer_orient_sensor =
sensor_manager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD);
```

After getting the sensor we can register a SensorEventListener for a given sensor by providing the name of the sensor as a parameter to registerListener() function. The sampling frequency at which the sensor values will be collected is also passed as a parameter to this function. When we call this function, that particular sensor will be activated and start listening to the changes within its hardware. The following snippet will register the light sensor.

```
sensor_manager.registerListener(MainActivity.this, light_sensor,
SensorManager.SENSOR_DELAY_NORMAL);
```

When we want to stop the sensor from listening to the changes, we need to unregister it by calling unregisterListener() function and passing the name of the sensor as one of the parameters as shown below:

```
sensor_manager.unregisterListener(MainActivity.this, light_sensor);
```

After the sensor is registered, if there is any change encountered in the sensor reading or there is any new reading, onSensorChanged() function is called. The parameter passed to this function is the object of SensorEvent class. The type of sensor and the value of it can be extracted from this class as shown in the following snippet:

```
@Override
public void onSensorChanged(SensorEvent event) {
if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER)
{
accelerometer_values_x = event.values[0];
accelerometer_values_y = event.values[1];
accelerometer_values_z = event.values[2];
 }
}
```

For using the GPS and get information about the location of the user, LocationManager class is used. The instance of this class is created to access the location services by calling getSystemService() function as shown below:

```
LocationManager locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
```

After this we checked whether there is a GPS provider available by calling isProviderEnabled() function as shown in the following code snippet:

```
public boolean provider = false;
provider = locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER);
```

Now if the GPS provider is available the location updates are requested by calling requestLocationUpdates() function and passing name of the GPS provider, minimum time between those updates, minimum distance after which the updates will be checked and current context as parameters for that function. After this the latitude and longitude values of the location is obtained by calling getLastKnownLocation() function. Following is the code snippet for this process:

```
public Location location;

if (provider)
{
    if (location == null) {

        locationManager.requestLocationUpdates(
                LocationManager.GPS_PROVIDER,
                MIN_TIME_BW_UPDATES,
                MIN_DISTANCE_CHANGE_FOR_UPDATES, MainActivity.this);

        if (locationManager != null) {
            location = locationManager
                    .getLastKnownLocation(LocationManager.GPS_PROVIDER);
            if (location != null) {
                latitude = (float) location.getLatitude();
                longitude = (float) location.getLongitude();
                }
        }
    }
}
```

In addition to the sensor data, user daily activities are collected by using Google ActivityRecognition API. We first created an object of GoogleApiClient class and use that to get connected to Google Play Services. The code snippet for connection process is shown below:

```
GoogleApiClient mApiClient = new GoogleApiClient.Builder(this)
        .addApi(ActivityRecognition.API)
        .addConnectionCallbacks(this)
        .addOnConnectionFailedListener(this)
.build();
mApiClient.connect();
```

After getting connected, we started a new service ActivityRecognizedService from which we are requesting activity updates by calling requestActivityUpdates() function and passing Goolge API client and time interval at which we need updates, as parameters. Following is the code snippet for this process:

```
@Override
public void onConnected(Bundle bundle) {
    Intent intent = new Intent(this, ActivityRecognizedService.class);
    PendingIntent pendingIntent = PendingIntent.getService(this, 0, intent,
PendingIntent.FLAG_UPDATE_CURRENT);
    ActivityRecognition.ActivityRecognitionApi.requestActivityUpdates(mApiClient,
10000, pendingIntent);
}
```

In the service we defined a function onHandleIntent() and handled the intent which is passed to this service from the main activity. That intent holds all the activity data like name of the activity and its confidence. We send this data back to the main activity by using sendBroadcast() function and passing the intent containing all the information as a parameter to this function. This process is achieved by writing the following code:

```
@Override
protected void onHandleIntent(Intent intent) {
 if (ActivityRecognitionResult.hasResult(intent)) {
      ActivityRecognitionResult result = ActivityRecognitionResult.extractResult(intent);
      if (result.getMostProbableActivity().getConfidence() >= 75) {
         Log.i(TAG, getType(result.getMostProbableActivity().getType()) + "t" +
result.getMostProbableActivity().getConfidence());
         i = new Intent("ACTIVITY_RECOGNITION_DATA");
         i.putExtra("Activity", getType(result.getMostProbableActivity().getType()));
         i.putExtra("Confidence", result.getMostProbableActivity().getConfidence());
         i.putExtra("Time", epochTime());
         i.putExtra("Wi-Fi",getWi-FiSignals());
         sendBroadcast(i);
      }}}
private String getType(int type) {
   if (type == DetectedActivity.UNKNOWN)
      return "Unknown";
   else if (type == DetectedActivity.IN_VEHICLE)
      return "In Vehicle";
   else if (type == DetectedActivity.ON_BICYCLE)
      return "On Bicycle";
   else if (type == DetectedActivity.ON_FOOT)
      return "On Foot";
   else if (type == DetectedActivity.STILL)
      return "Still";
   else if (type == DetectedActivity.TILTING)
      return "Tilting";
   else
      return "";
}
```

For getting information about the Wi-Fi signals we used Wi-FiManager class. We created the instance of this class by calling getSystemService() function. The parameter passed to this function is WI-FI_SERVICE of the current context. WI-FI_SERVICE is the Wi-Fi manager for management of Wi-Fi connectivity. After getting service we get the name of the Wi-Fi signal to which the user is connected by calling getSSID() function. The code snippet for the whole process is shown below:

```
public String getWi-FiSignals() {
   String ssid = null;
   final Wi-FiManager Wi-FiManager;
   final Wi-FiInfo info;
   List<ScanResult> results = null;
   String etWi-FiList = "";
   String textStatus = "";

   ConnectivityManager connManager = (ConnectivityManager)
this.getSystemService(Context.CONNECTIVITY_SERVICE);
  NetworkInfo networkInfo = connManager.getActiveNetworkInfo();
  if (networkInfo!=null) {
     if(networkInfo.getType() == ConnectivityManager.TYPE_WI-FI) {
        Wi-FiManager = (Wi-FiManager) this.getSystemService(Context.WI-
FI_SERVICE);
        info = Wi-FiManager.getConnectionInfo();
        ssid = info.getSSID();
        Log.v(TAG, "SSID: " + ssid);
       }
}
return ssid;
}
```

We received the information, sent by the Service, in the main activity by creating an object of BroadcastReceiver class as shown in the code snippet below:

```
BroadcastReceiver receiver = new BroadcastReceiver() {
   @Override
   public void onReceive(Context context, Intent intent) {
      activity = intent.getStringExtra("Activity");
      String confidence = " " + intent.getExtras().getInt("Confidence");
      time = " " + intent.getStringExtra("Time");
      Wi-Fi = " " + intent.getStringExtra("Wi-Fi");
}
```

5.2.2 Android Permissions

For an Android application to access data from sensors, Wi-Fi signals or GPS we need to set some permissions in AndroidManifest.xml file. For getting information about the Wi-Fi signals and for recognizing activities of the user we set following two permissions:

```
<uses-permission android:name="android.permission.ACCESS_WI-FI_STATE" />
<uses-permission
android:name="com.google.android.gms.permission.ACTIVITY_RECOGNITION" />
```

For collecting data from available hardware sensors and GPS we set the following permissions:

```
<uses-permission android:name="android.permission.BODY_SENSORS" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"
/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

5.2.3 Java Application for Creating Profiles and Assigning Weights

As discussed in section 4.2, for this thesis we created an android application which can collect the user data but we haven’t used that application to actually collect the data. For testing our approach, we used publicly available data set named GCU dataset. This section explains the implementation detail of each step discussed in Chapter 4.

For matching the data between two consecutive days we are using Union and Intersection. For this we created functions *union()* and *intersection()*. The parameters passed to these functions are the lists containing data for a particular time bin for two consecutive days. The *union()* function will return the list of all the distinct values and *intersection()* function will return the list of common values. Than the size of both the lists will be calculated by using *list.size()* function and match percentage will be calculated based on the sizes. This process is achieved by writing the following code snippet:

```
List union = union(day1,day2);
                List intersection = intersection(day1,day2);
                 double unionsize = union.size();
                 double intersectionsize = intersection.size();

match_percent = (intersectionsize/unionsize) * 100;

public static <T> List<T> union(List<T> list1, List<T> list2) {
       Set<T> set = new HashSet<T>();

       set.addAll(list1);
       set.addAll(list2);

       return new ArrayList<T>(set);
   }

   public static <T> List<T> intersection(List<T> list1, List<T> list2) {
       Set<T> list = new HashSet<T>();

       for (T t : list1) {
           if(list2.contains(t)) {
               list.add(t);
           }
       }

       return new ArrayList<T>(list);
   }
```

After we got the match percentages for every time bin of each sensor, we checked the stability in usage for that sensor based on those percentages and assign weights accordingly. For checking the priority, we are first storing the percentages as values and sensors as keys in a HashMap. After that we are applying sorting function on it which gives us the sorted HashMap of sensors and percentages with the highest value in the top. The code snippet for this process is shown below:

```
private static HashMap sortByValues(HashMap map) {
     List list = new LinkedList(map.entrySet());
     Collections.sort(list, new Comparator() {
         public int compare(Object o1, Object o2) {
           return ((Comparable) ((Map.Entry) (o1)).getValue())
             .compareTo(((Map.Entry) (o2)).getValue());
         }
     });

     HashMap sortedHashMap = new LinkedHashMap();
     for (Iterator it = list.iterator(); it.hasNext();) {
          Map.Entry entry = (Map.Entry) it.next();
          sortedHashMap.put(entry.getKey(), entry.getValue());
     }
     return sortedHashMap;
 }
```

CHAPTER 6 EVALUATION RESULTS AND ANALYSIS

For testing our approach, we used subset of GCU dataset Version 1 [3]. This is a publicly available dataset and comprises of the data collected from 7 users. All the users are either staff or students of the Glasgow Caledonian University. The data was collected in the year 2013 by using Android devices and comprises of data from cell towers, Wi-Fi networks and application usage. The length of the original dataset varies from 2 weeks to 14 weeks for different users. But we are using only 2 weeks' data for each user. The different fields in this dataset are:

- Timestamp
- Probe name (GCU.CellProbe, GCU.RunningApplicationsProbe,
  GCU.Wi-FiProbe)
- Username
- Date
- Time
- List of values for each observed probe

All the information available in this dataset is anonymized. Table 4 below summarizes the data we are using.

1	Number of Users	7
2	Number of Days for Each User	15
3	Number of Sensors	3

Table 4 Dataset Details

6.1 EVALUATION ANALYSIS FOR ALL USERS

The evaluations discussed in chapter 4 for comparing the data and assigning weights are applied on data for each user. The data from each sensor is divided into time bins of 15 minutes for each day. For two consecutive days that data is compared to get the match percentage. The results of every user for few random days and time bins is discussed in the following sub sections.

6.1.1 Evaluation analysis for User 1

Figure 18 shows the match percentage of application usage of user 1 in every time bin for two consecutive random days. We have expanded the graph for 2 hours' duration in order to show clear picture of the results. It is clear from the Figure 18 that for this user there is not much difference in the threshold of different time bins for application usage.

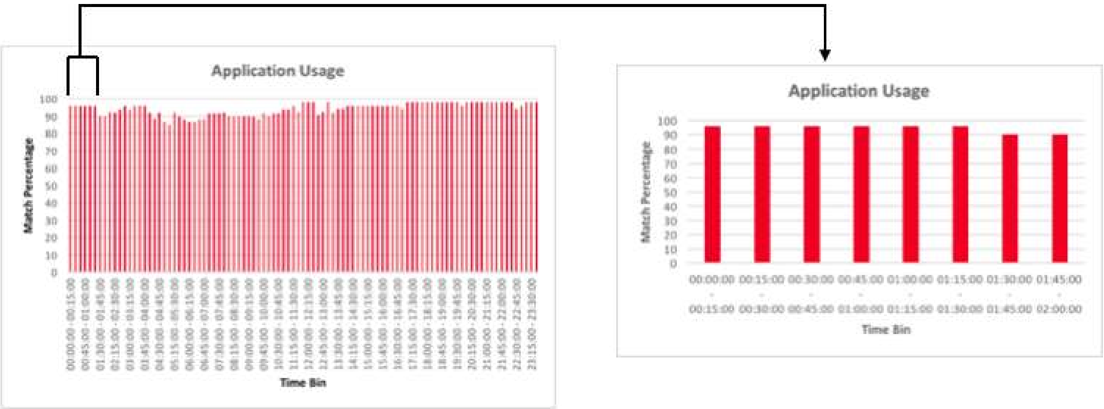


Figure 18 Match Percentage in Different Time Bins for Application Usage (User 1)

Figure 19 shows the results of match percentages of cell tower in each time interval for randomly selected two consecutive days. The graphs show us that there is high variation in the percentages for different time bins.

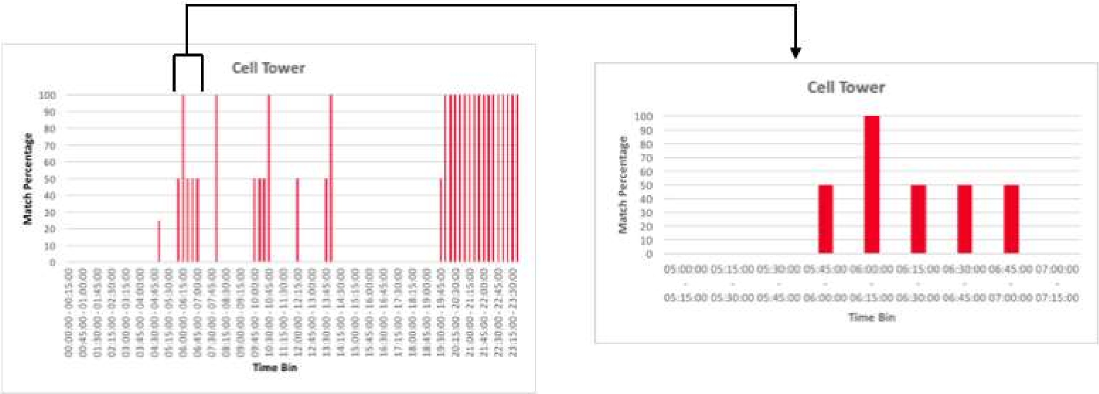


Figure 19 Match Percentage in Different Time Bins for Cell Tower Location (User 1)

Figure 20 shows the same information for Wi-Fi Networks. This sensor also shows disparity in the match percentages for different time bins.

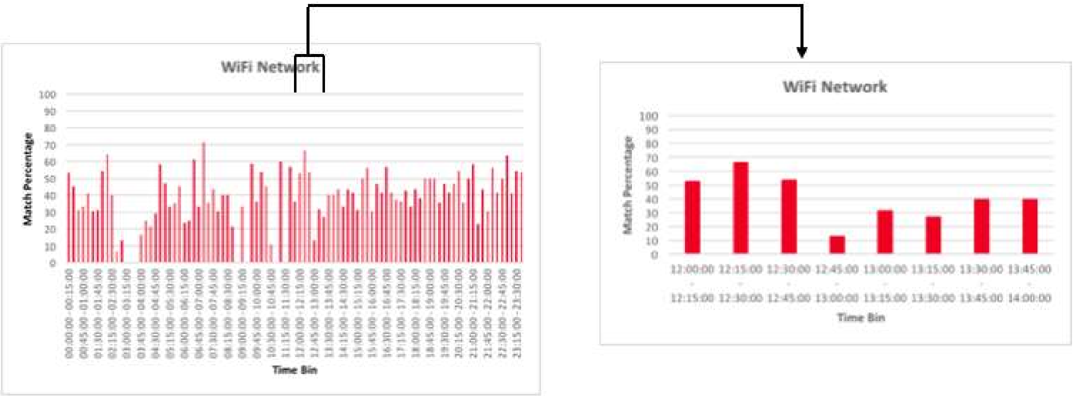


Figure 20 Match Percentage in Different Time Bins for Wi-Fi Networks (User 1)

Hence, it is clear from the above observations that user 1 shows stability in usage of the smartphone during certain time intervals but not in the entire day, except for the application usage.

The next thing we investigated is the need of assigning weights to the sensors. As a proof of our argument we generated graphs for four random time bins of each day for every sensor. These graphs are shown in Figure 21 and Figure 22. From these graphs we can clearly see that user is showing more stability in application usage followed by the Wi-Fi and then cell tower for most of the days.

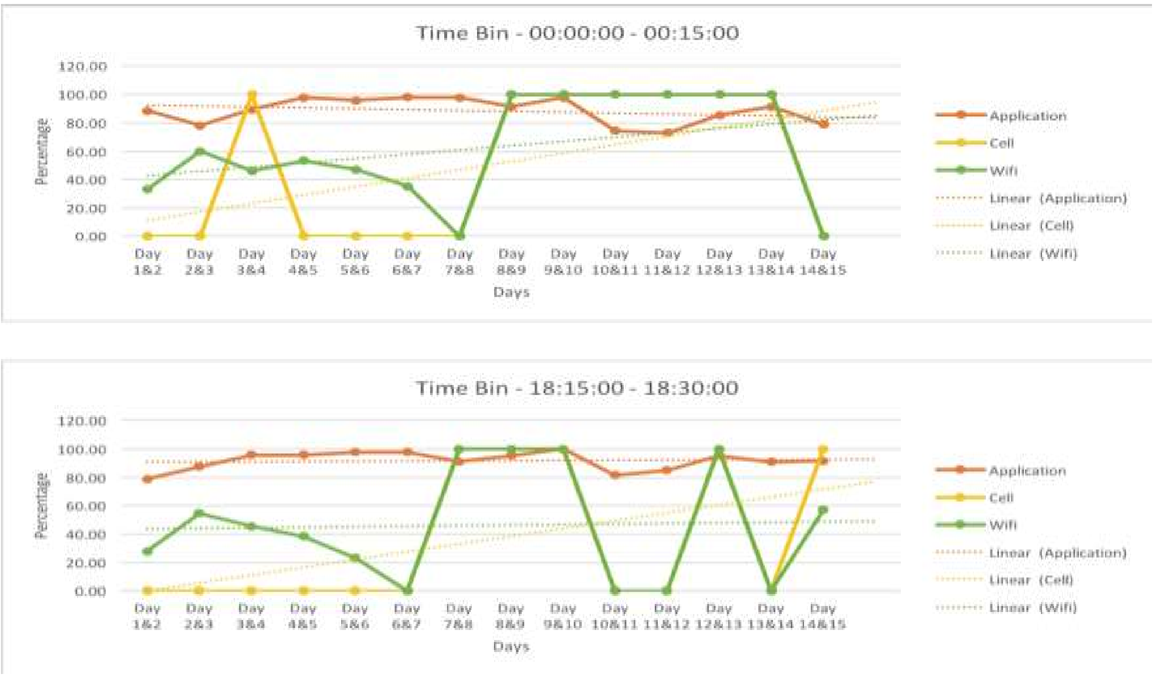


Figure 21 Stability of Different Sensors in Random Time Bins (User 1)

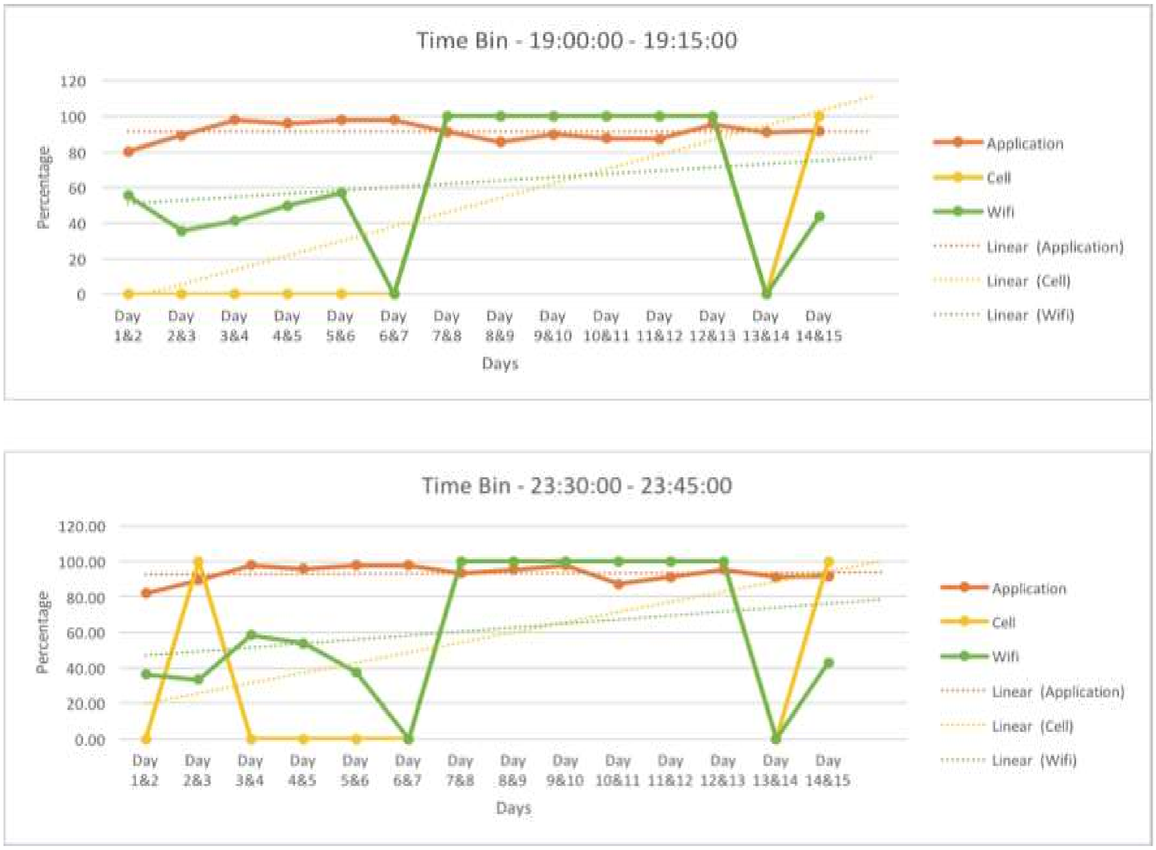


Figure 22 Stability of Different Sensors in Random Time Bins (User 1)

So, based on the information we got from the above graphs we assigned weights to the sensors according to the stability they are showing for this user. We did this by using two weighting techniques discussed in Chapter 4 and calculated the weighted average according to both the techniques. We have also calculated the simple average and compared it with the other two techniques. The results of this comparison are shown in Figure 23 and Figure 24.

We observed from the graphs in Figure 23 and Figure 24 that the weighting scheme 1 is giving the high aggregate percentage followed by weighting scheme 2 and average for most of the time bins. So, weighting scheme 1 is showing better results for user 1 in this dataset.

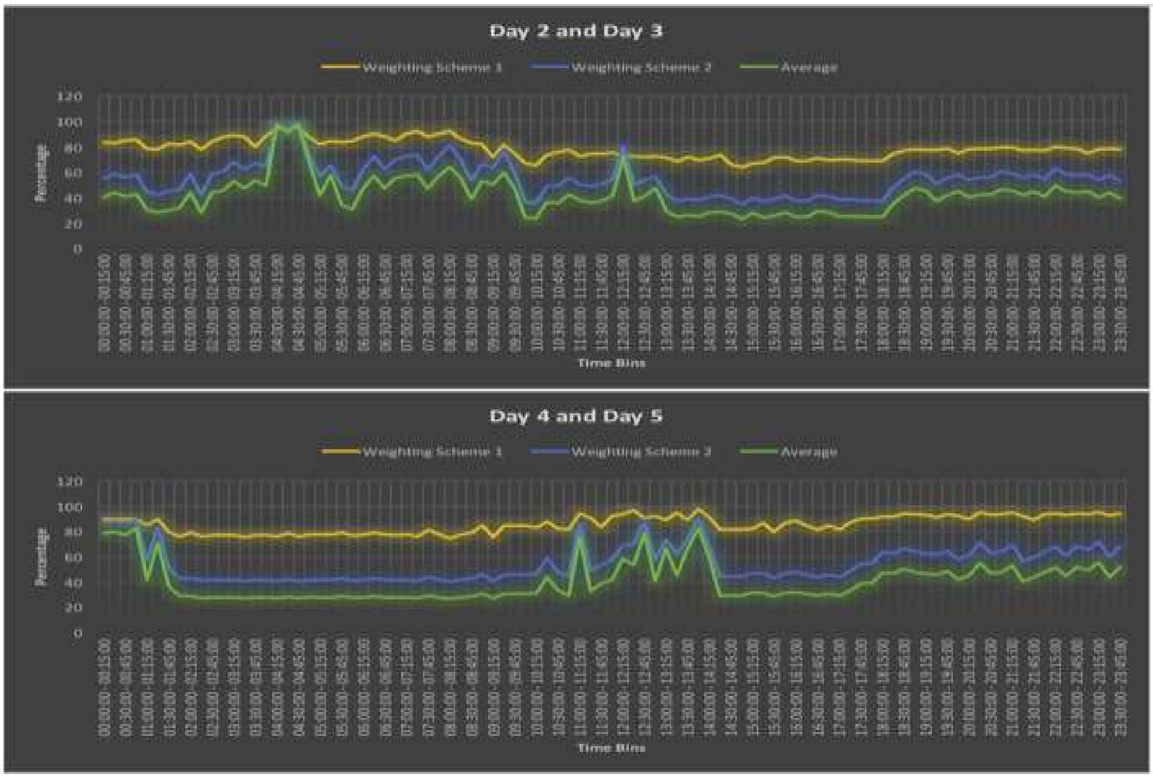


Figure 23 Comparison of Different Weighting Schemes (User 1)

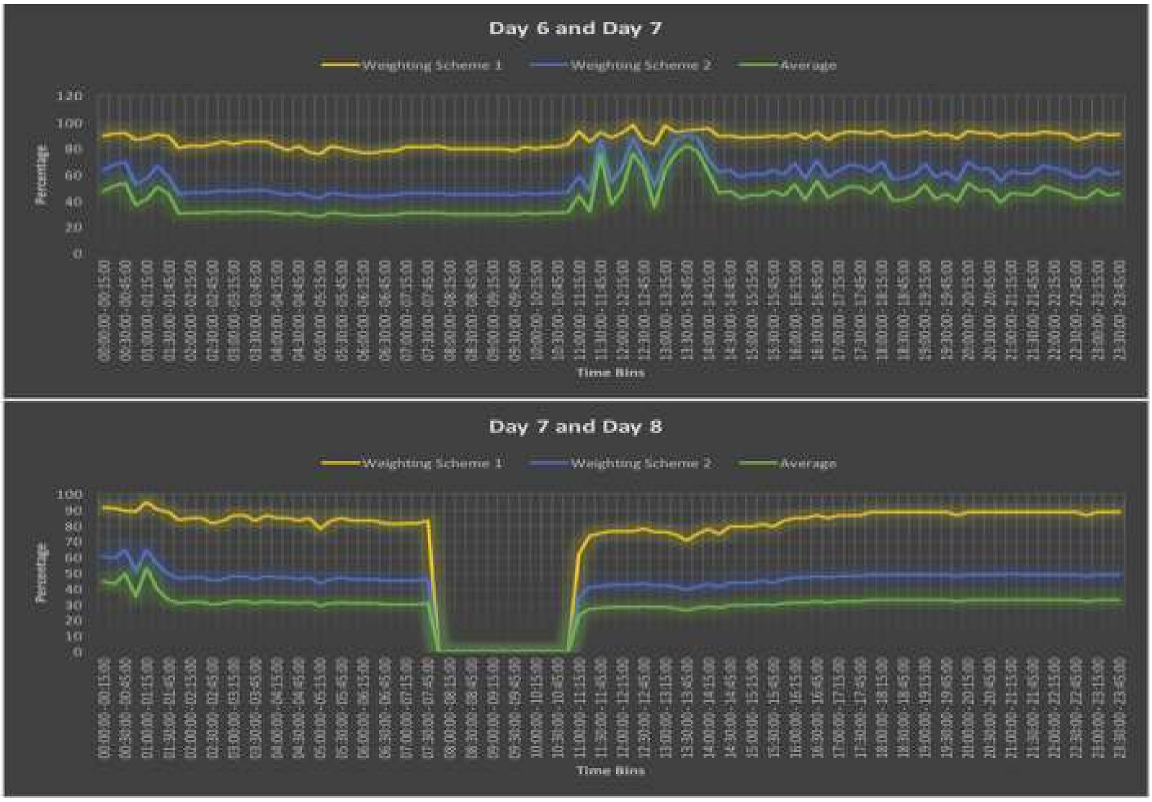


Figure 24 Comparison of Different Weighting Schemes (User 1)

6.1.2 Evaluation analysis for User 2

Figure 25 shows the match percentage of application usage of user 2 in every time bin for two consecutive random days. We have expanded the graph for 2 hours' duration in order to show clear picture of the results. It is clear from the Figure 25 that for this user the match percentages are approximately same for all time bins.

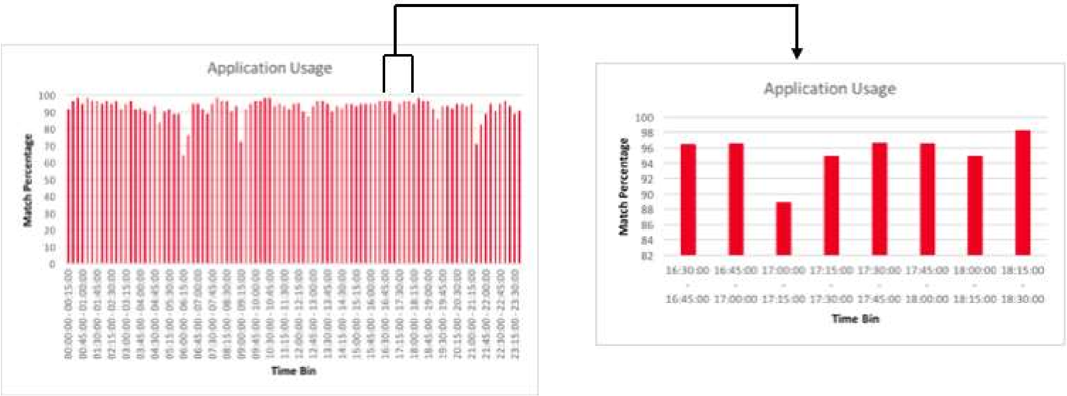


Figure 25 Match Percentage in Different Time Bins for Application Usage (User 2)

Figure 26 shows the results of match percentages for user 2 of cell tower in each time interval for randomly selected two consecutive days. The graphs show us that for most of the time bins it is approximately 50% but still there is high variation in the percentages for different time bins.

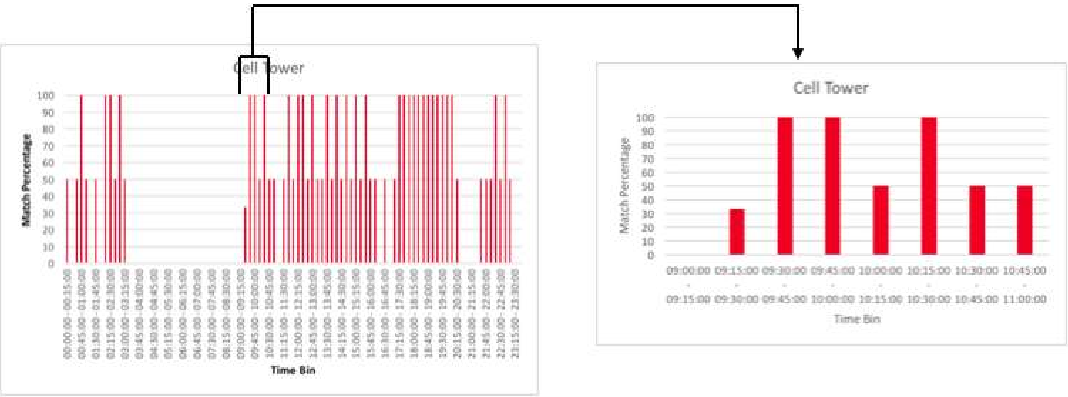


Figure 26 Match Percentage in Different Time Bins for Cell Tower (User 2)

Figure 27 shows that this user shows very less stability in terms of connecting to Wi-Fi signals for most of the time. This sensor also shows disparity in the match percentages for different time bins.

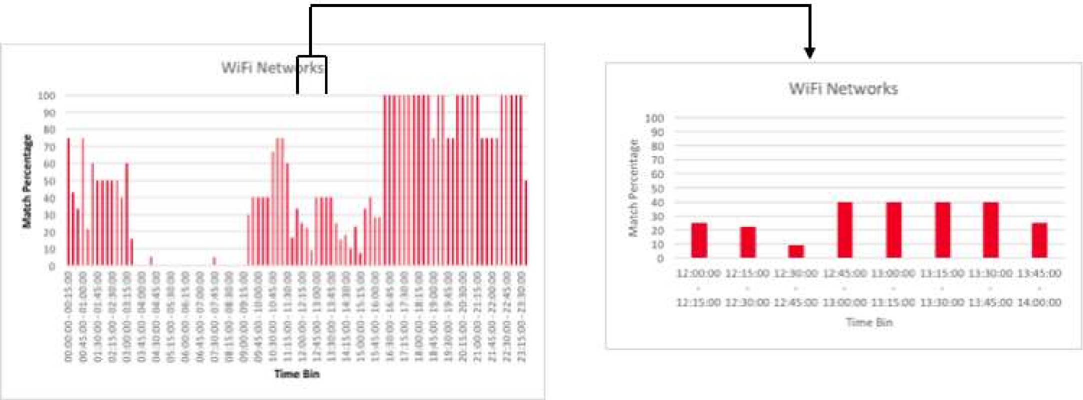


Figure 27 Match Percentage in Different Time Bins for Wi-Fi Networks (User 2)

Hence, it is clear from the above observations that user 2 also shows stability in usage of the smartphone during certain time intervals but not in the entire day except for the application usage.

The next thing we investigated is the need of assigning weights to the sensors. As a proof of our argument we generated graphs for four random time bins of each day for every sensor. These graphs are shown in Figure 28 and Figure 29.

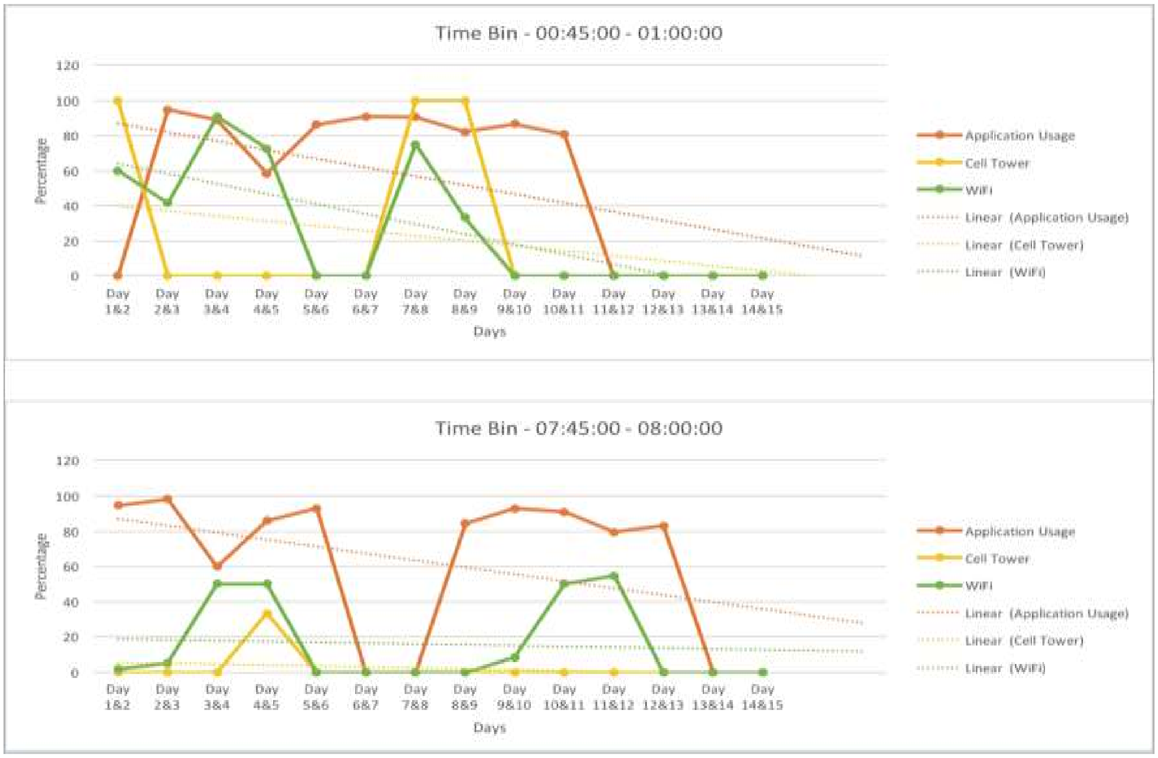


Figure 28 Stability of Different Sensors in Random Time Bins (User 2)

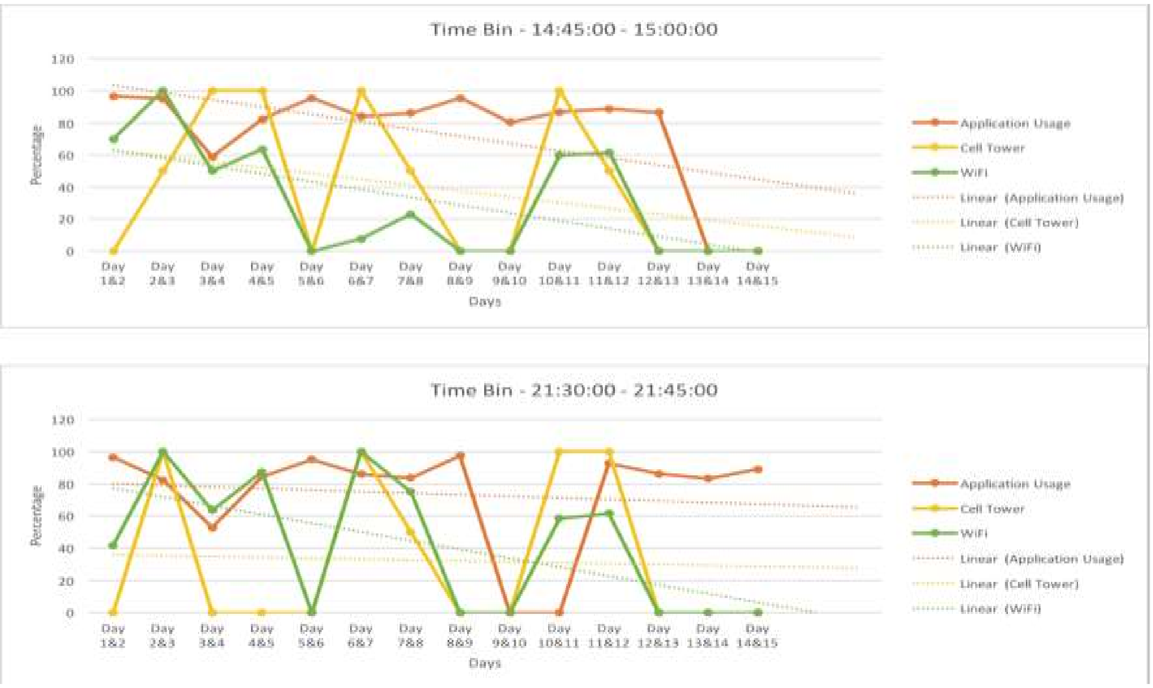


Figure 29 Stability of Different Sensors in Random Time Bins (User 2)

From these graphs we can clearly see that user is showing more stability in application usage followed by the Wi-Fi and then cell tower for most of the days except for time bin 14:45:00 – 15:00:00 where stability in cell tower is greater than Wi-Fi.

We followed the same procedure for calculating weighted average using two different techniques and comparing it with simple average for user 2 as well. The results of this comparison are shown in Figure 30 and Figure 31.

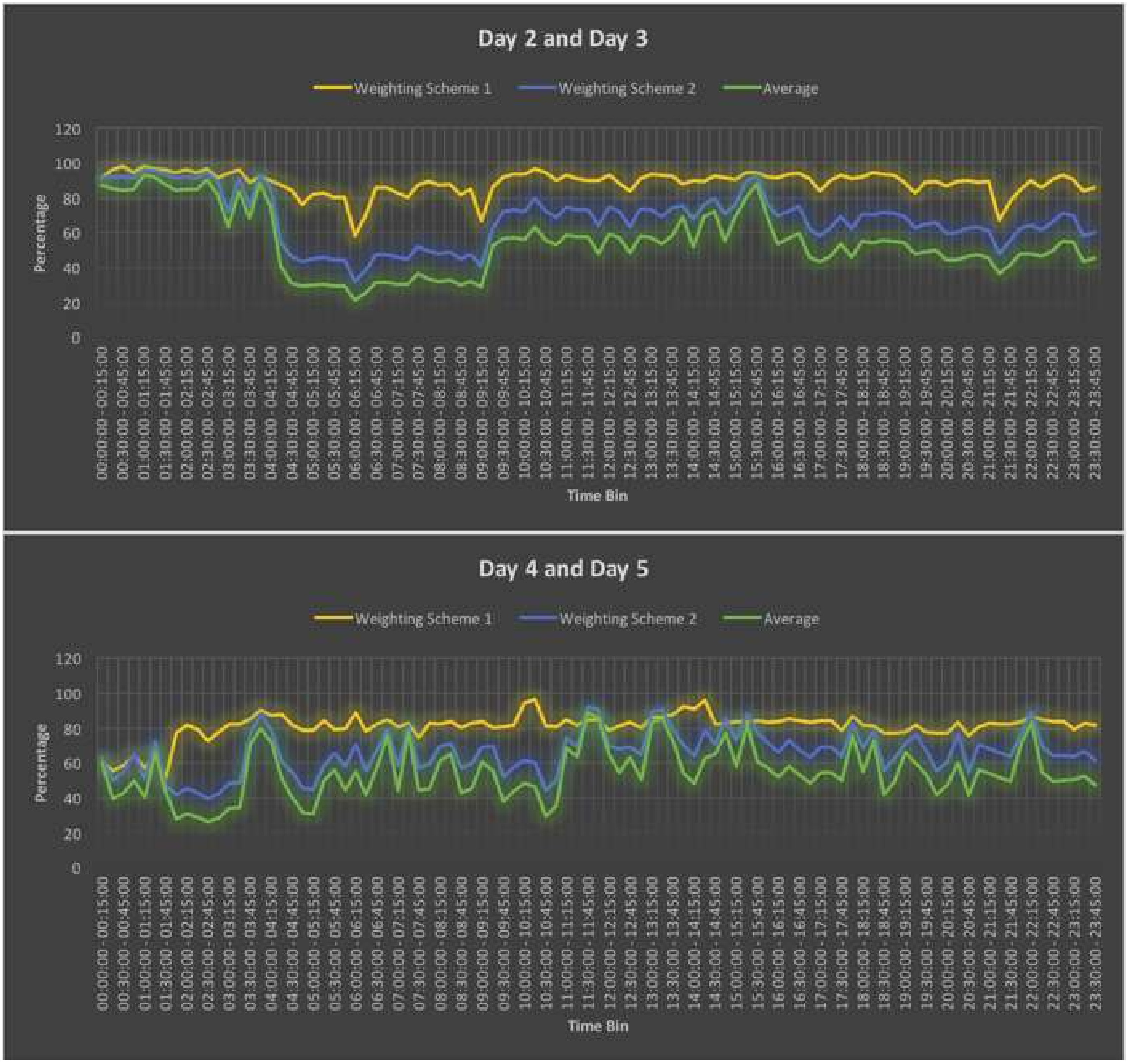


Figure 30 Comparison of Different Weighting Schemes (User 2)

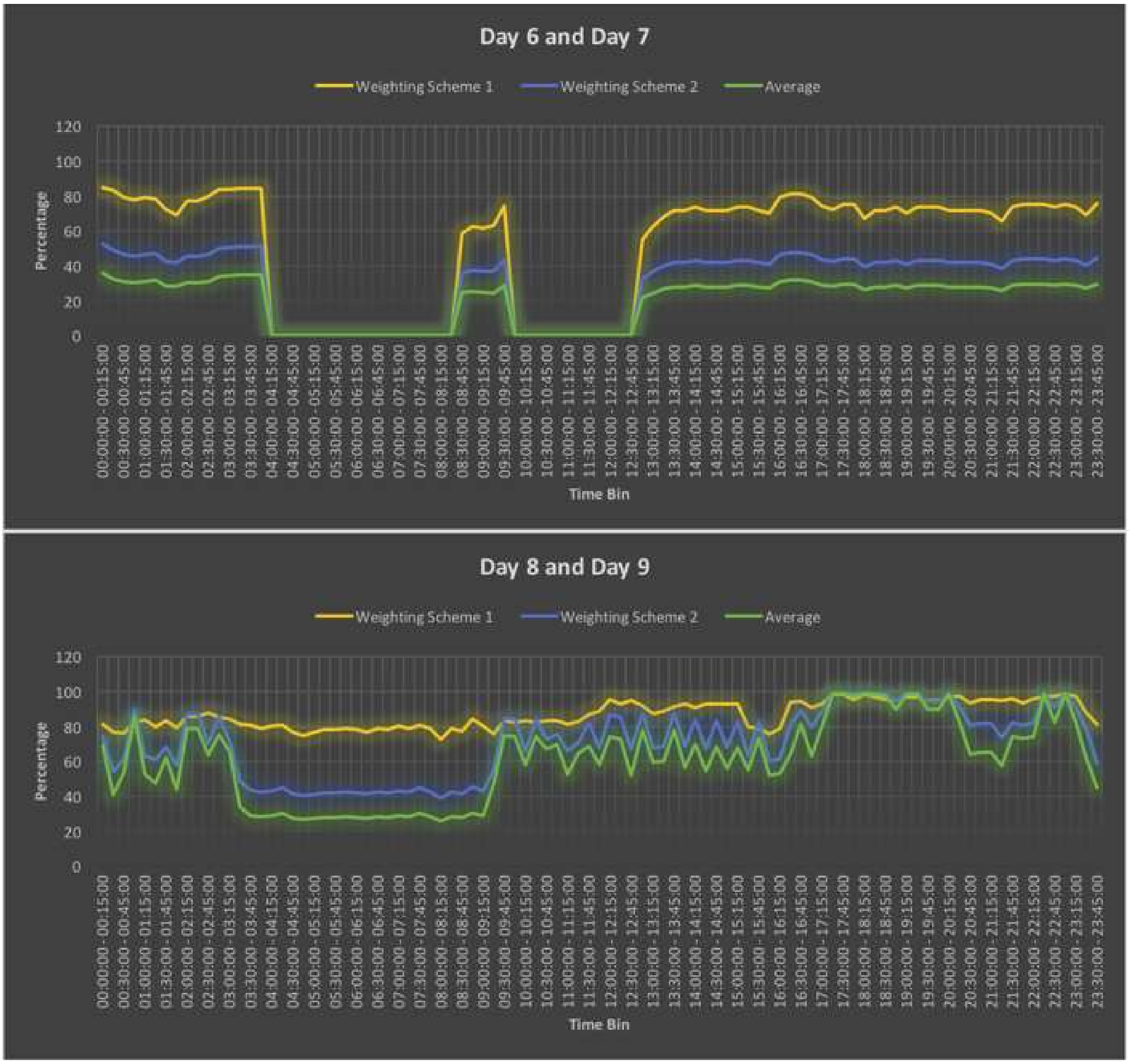


Figure 31 Comparison of Different Weighting Schemes (User 2)

We observed from the graphs that the weighting scheme 1 is giving the high aggregate percentage followed by weighting scheme 2 and average for most of the time bins. For certain time bins we observed that there is no difference or very less difference in the weighted average and the simple average. By analyzing the data, we found that this happens when the values of match percentages for every sensor is approximately same. But overall for this user also, weighting scheme 1 is showing better results for dataset we used.

6.1.3 Evaluation analysis for User 3

Figure 32 shows the match percentage of application usage of user 3 in every time bin for two consecutive random days. We have expanded the graph for 2 hours' duration in order to show clear picture of the results. It is clear from the Figure 32 that for this user there is not much difference in the threshold of different time bins for application usage except for the fact that in certain time intervals the match percentage is 0 which means no stability at all.

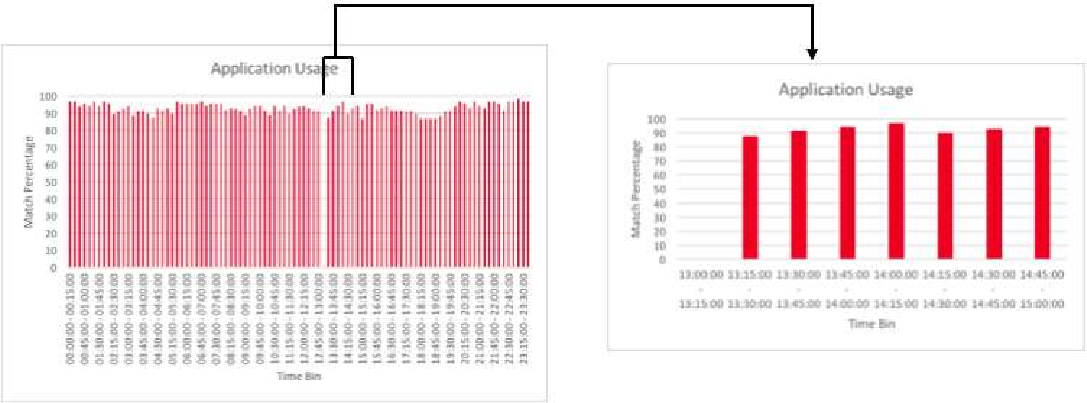


Figure 32 Match Percentage in Different Time Bins for Application Usage (User 3)

Figure 33 shows the results of match percentages of user 3 for cell tower in each time interval for randomly selected two consecutive days. The graphs show us that for most of the time bins it is 100% and user 3 is showing more stability in terms of cell tower as compared to the previous two users.

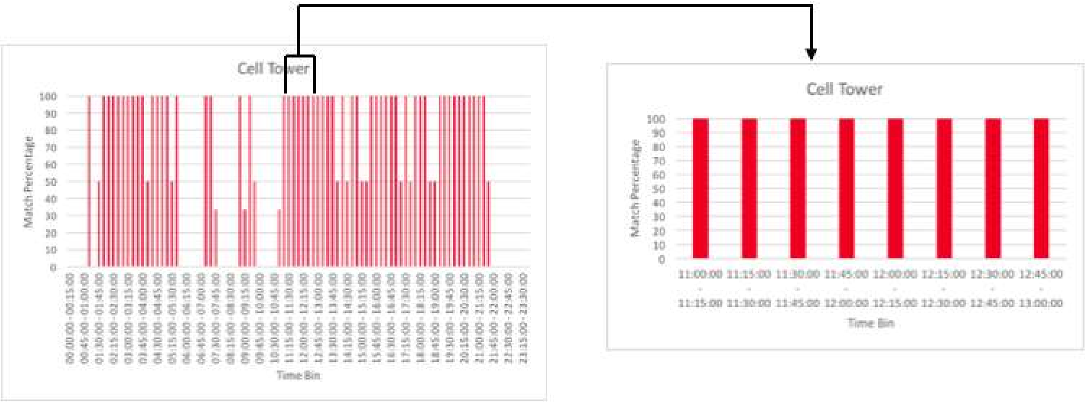


Figure 33 Match Percentage in Different Time Bins for Cell Tower (User 3)

Figure 34 shows high variation in match percentages of Wi-Fi Networks for this user.

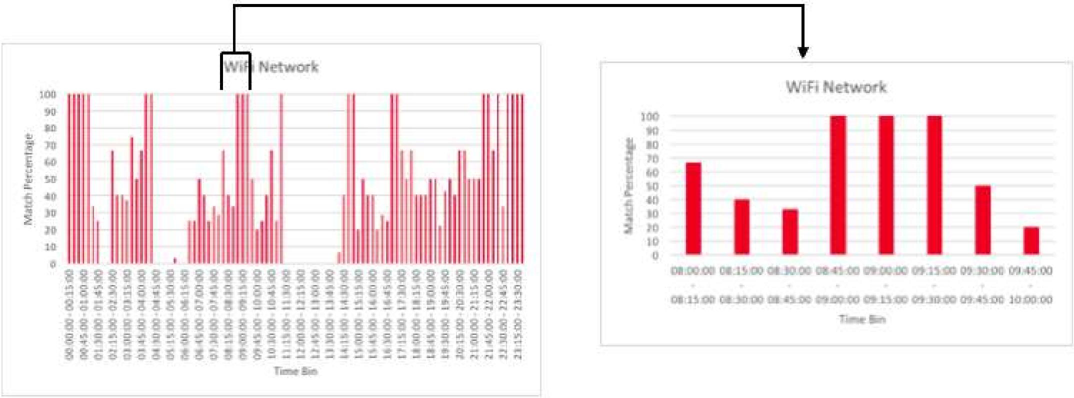
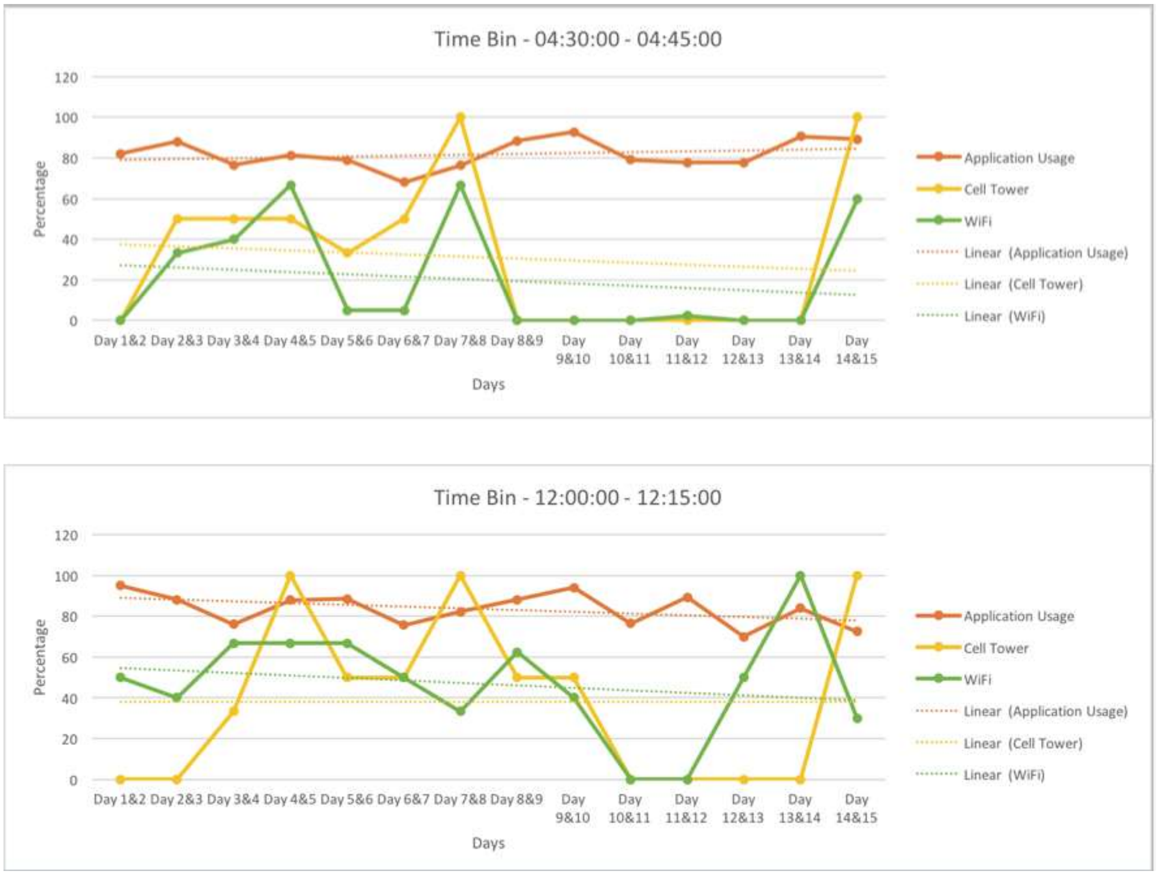


Figure 34 Match Percentage in Different Time Bins for Wi-Fi Networks (User 3)

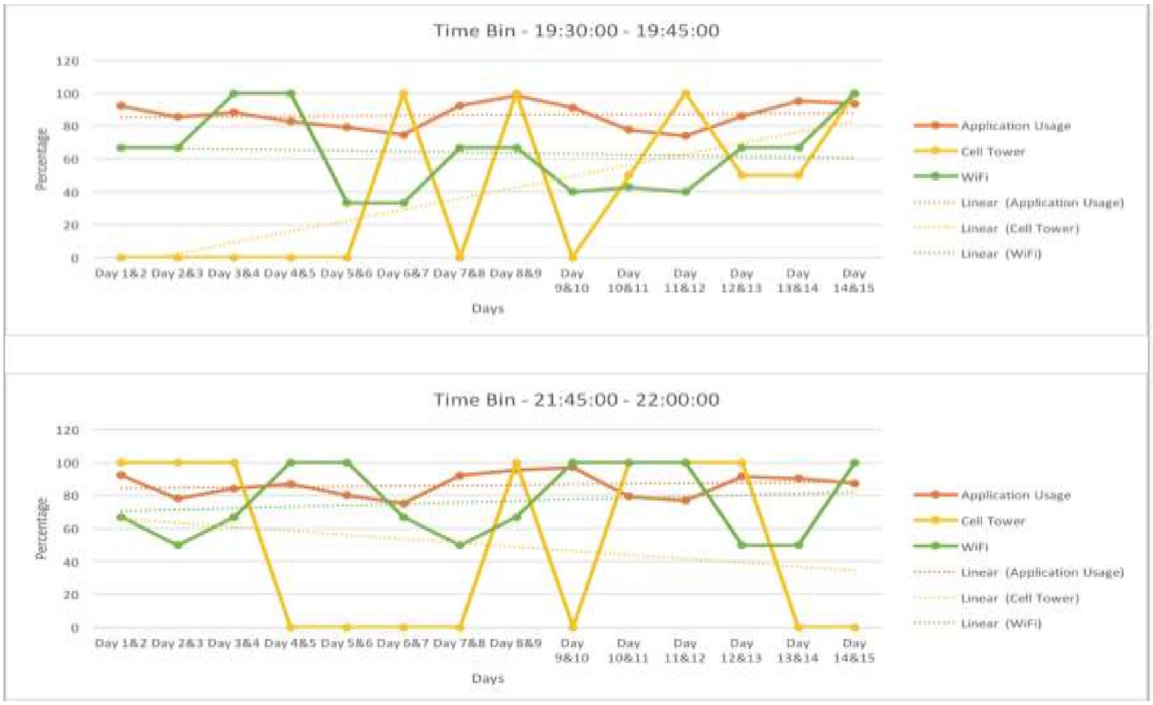
Hence, it is clear from the above observations that user 3 shows stability in usage of the smartphone during certain time intervals but not in the entire day except for the application usage and cell tower.

The next thing we investigated for this user also is the need of assigning weights to the sensors. As a proof of our argument we generated graphs for four random time bins of each day for every sensor. These graphs are shown in Figure 35 and Figure 36.


Time Bin - 04:30:00 - 04:45:00

Time Bin - 12:00:00 - 12:15:00

Figure 35 Stability of Different Sensors in Random Time Bins (User 3)


Time Bin - 19:30:00 - 19:45:00

Time Bin - 21:45:00 - 22:00:00

Figure 36 Stability of Different Sensors in Random Time Bins (User 3)

From the above graphs we can clearly see that in a span of 15 days, user 3 is showing more stability in application usage followed by the Wi-Fi and then cell tower for most of the days and time bins.

The results of comparison of different weighting techniques for user 3 are shown in Figure 37 and Figure 38.

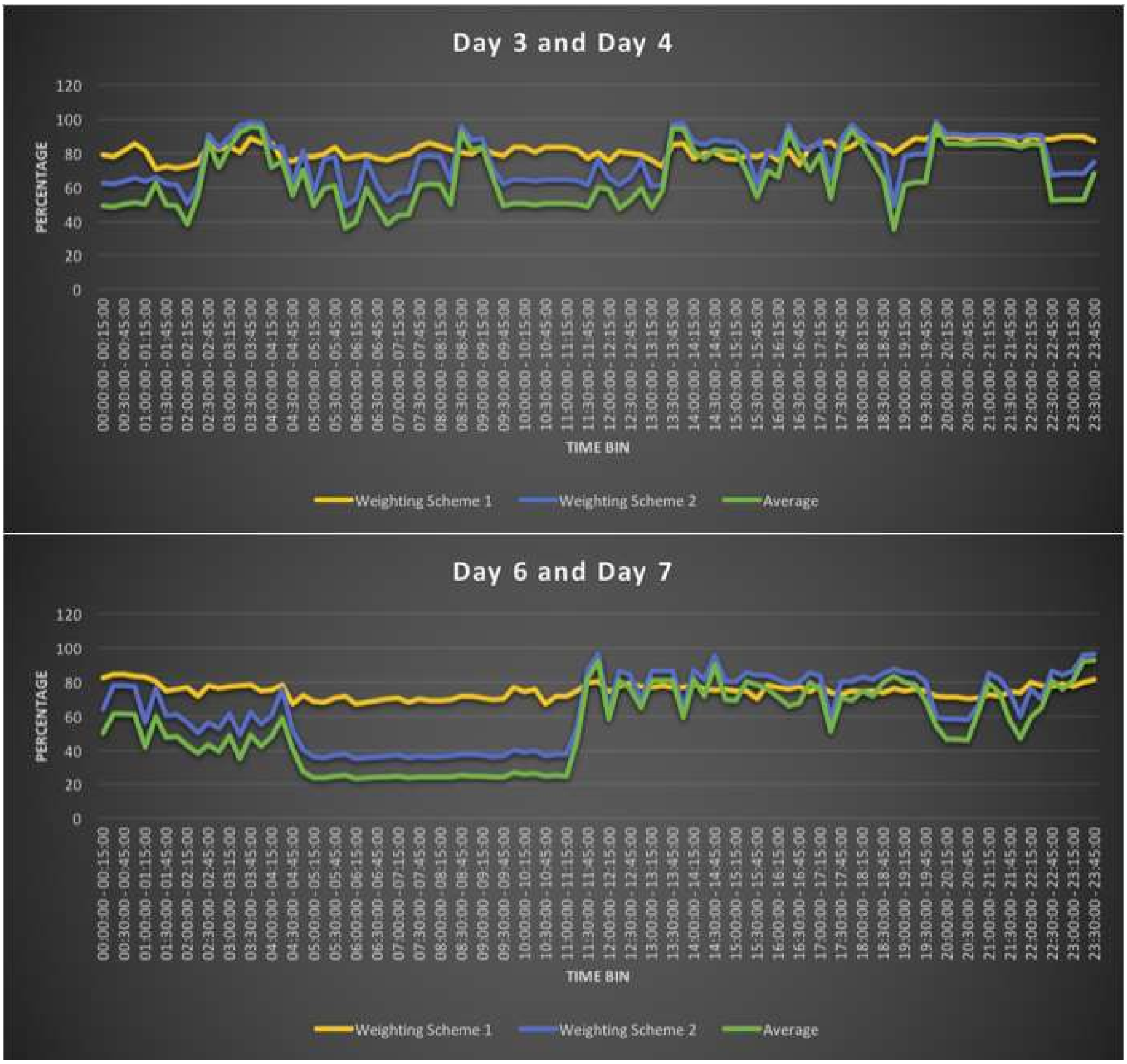


Figure 37 Comparison of Different Weighting Schemes (User 3)

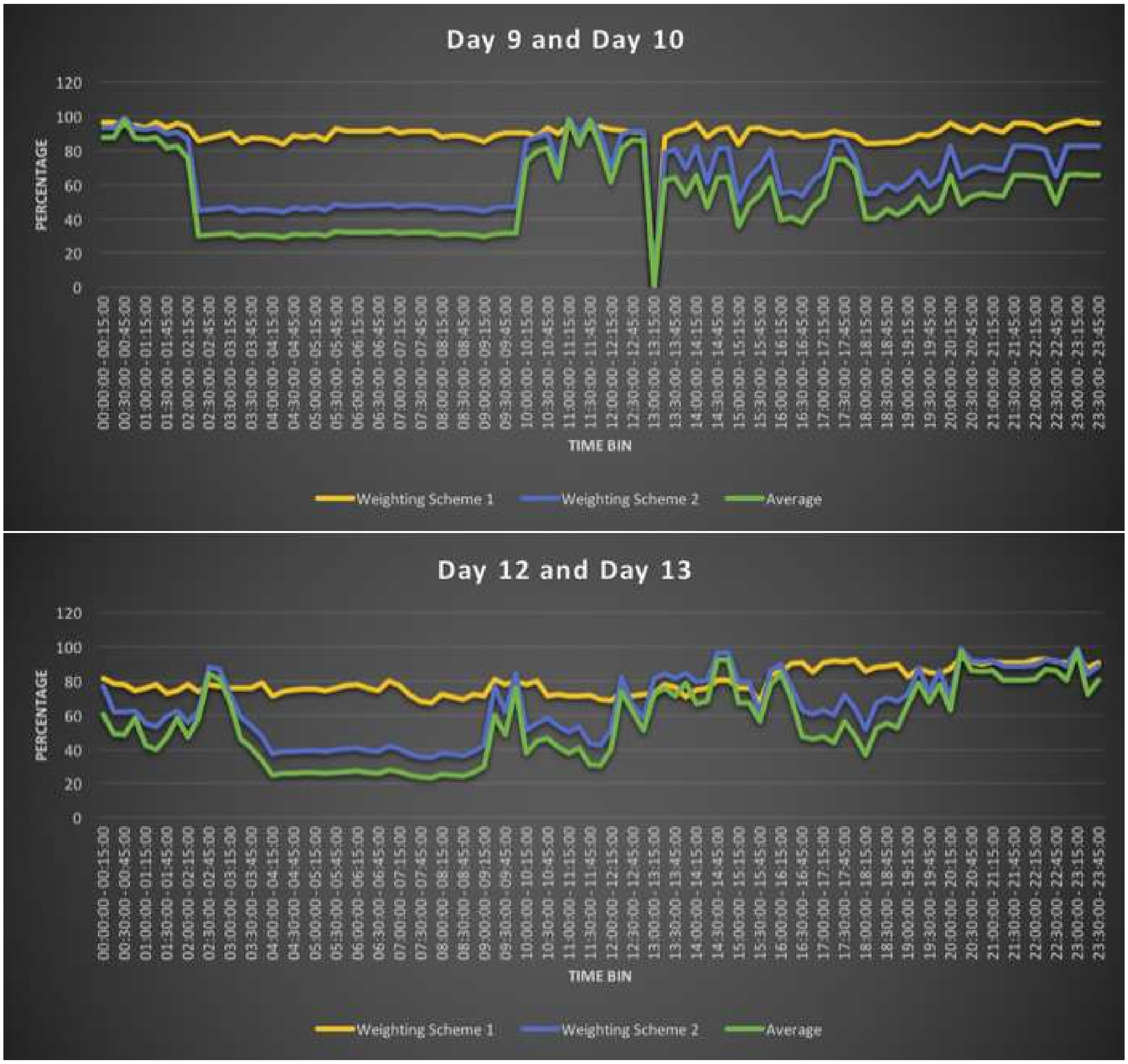


Figure 38 Comparison of Different Weighting Schemes (User 3)

We observed from the graphs that the weighting scheme 1 is giving the high aggregate percentage followed by weighting scheme 2 and average for most of the time bins. So, weighting scheme 1 is showing better results for user 3.

6.1.4 Evaluation analysis for User 4

Figure 39 shows the match percentage of application usage of user 4 in every time bin for two consecutive random days. We have expanded the graph for 2 hours' duration in order to show clear picture of the results. It is clear from the Figure 39 that for this user there is not much difference in the threshold of different time bins for application usage.

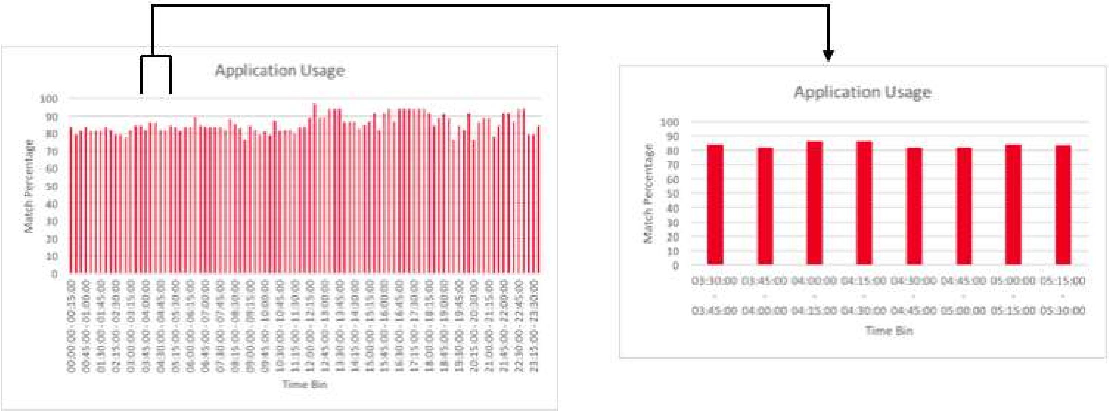


Figure 39 Match Percentage in Different Time Bins for Application Usage (User 4)

Figure 40 shows the results of match percentages of user 4 for cell tower in each time interval for randomly selected two consecutive days. The graphs show us that there is high variation in the percentages for different time bins. Also for most of the time intervals its 0.

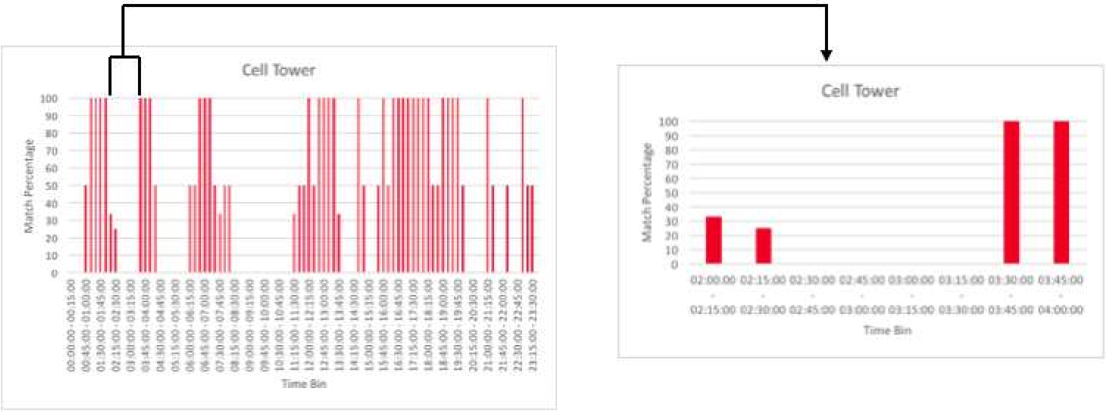


Figure 40 Match Percentage in Different Time Bins for Cell Tower (User 4)

Figure 41 shows the same information for Wi-Fi Networks. This sensor also shows disparity in the match percentages for different time bins.

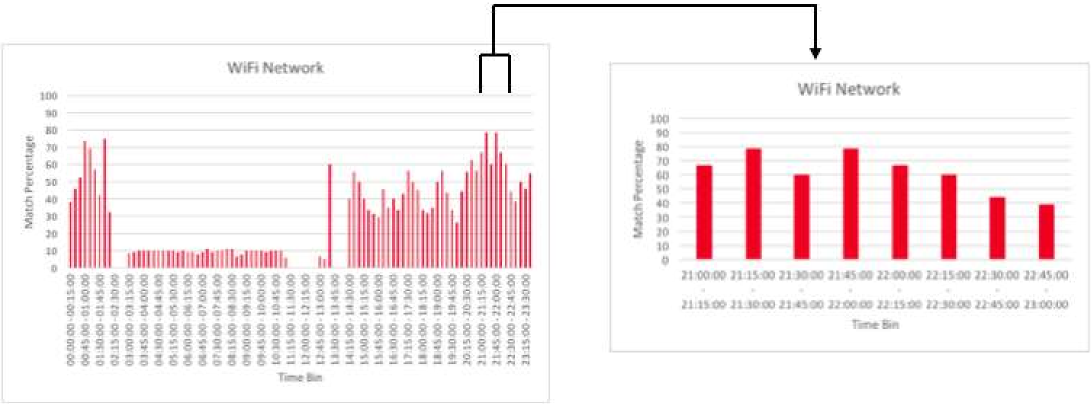


Figure 41 Match Percentage in Different Time Bins for Wi-Fi Networks (User 4)

Hence, it is clear from the above observations that this particular user shows stability in usage of the smartphone during certain time intervals but not in the entire day except for the application usage.

The next thing we investigated is the need of assigning weights to the sensors. As a proof of our argument we generated graphs for four random time bins of each day for every sensor. These graphs are shown in Figure 42 and Figure 43.

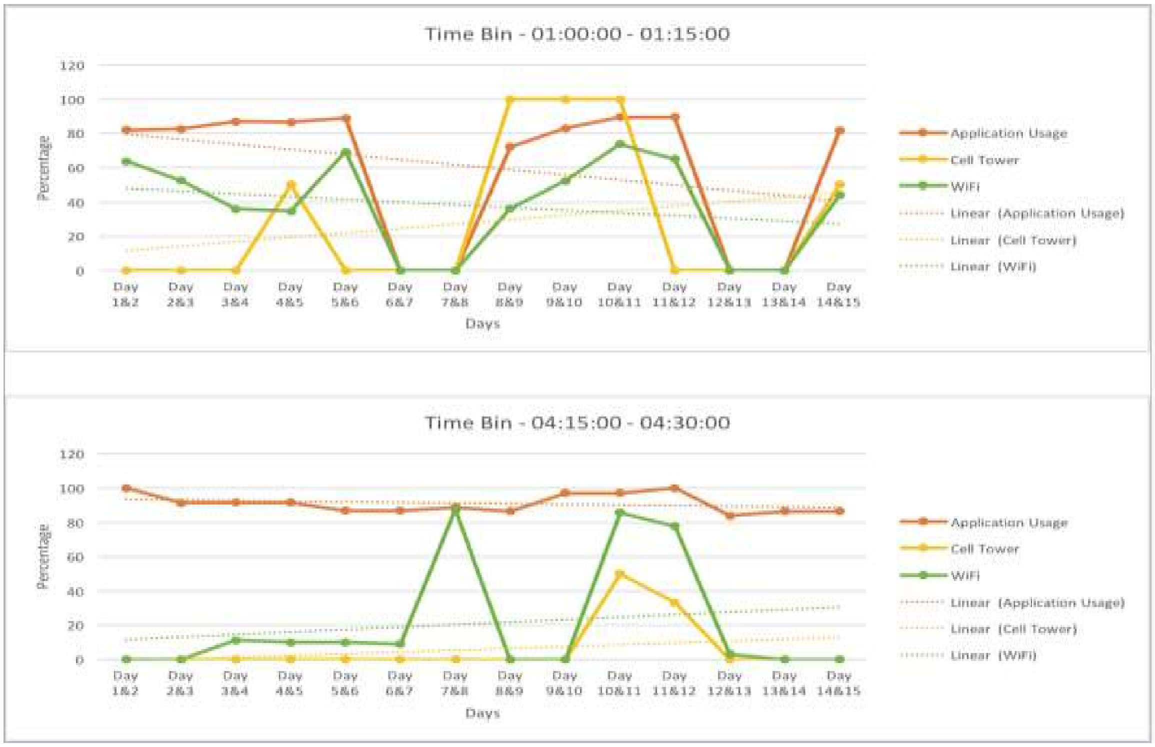


Figure 42 Stability of Different Sensors in Random Time Bins (User 4)

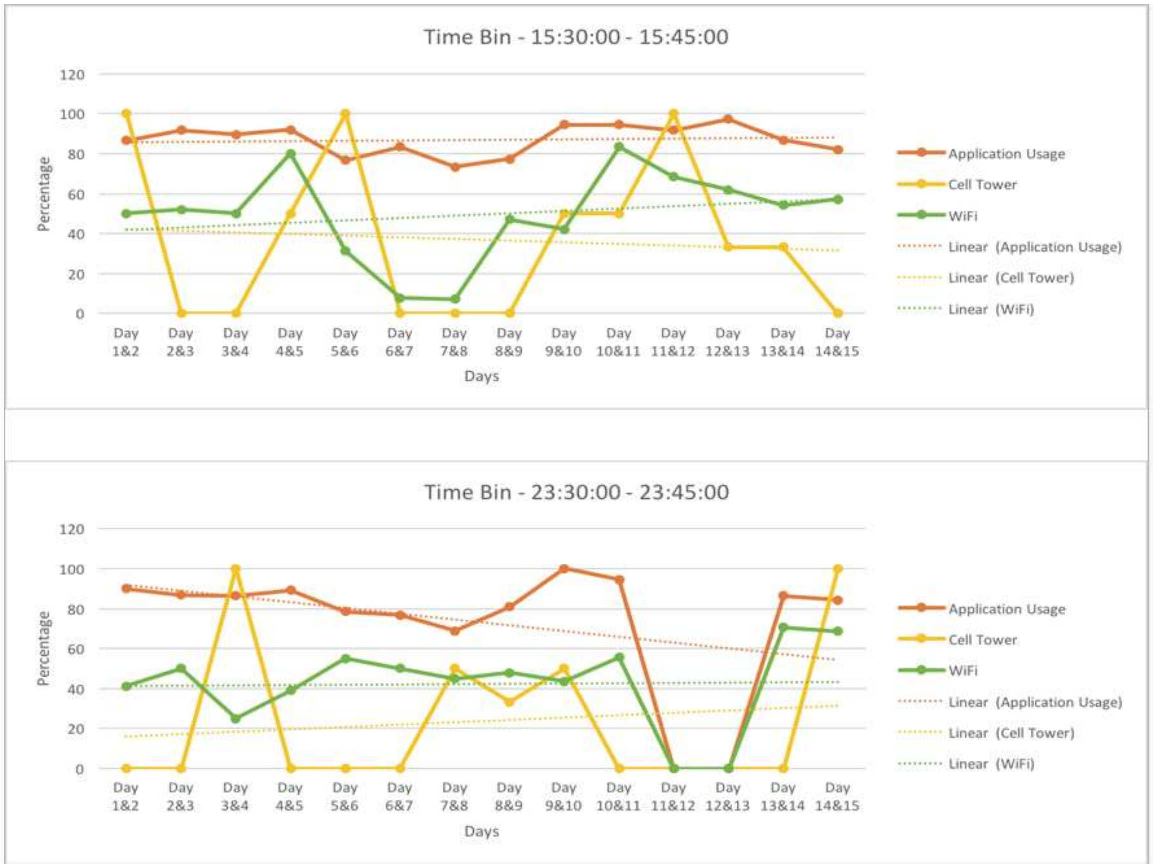


Figure 43 Stability of Different Sensors in Random Time Bins (User 4)

From these graphs we can clearly see that in a span of 15 days, user 4 is showing more stability in application usage followed by the Wi-Fi and then cell tower for most of the days.

So, based on the information we got from the above graphs we assigned weights to the sensors according to the stability they are showing for this user as well and compared the results of these techniques. The results of this comparison are shown in Figure 40 and Figure 41.

We observed from the graphs that the weighting scheme 1 is giving the high aggregate percentage followed by weighting scheme 2 and average for most of the time bins. So, weighting scheme 1 is showing better results for user 4.

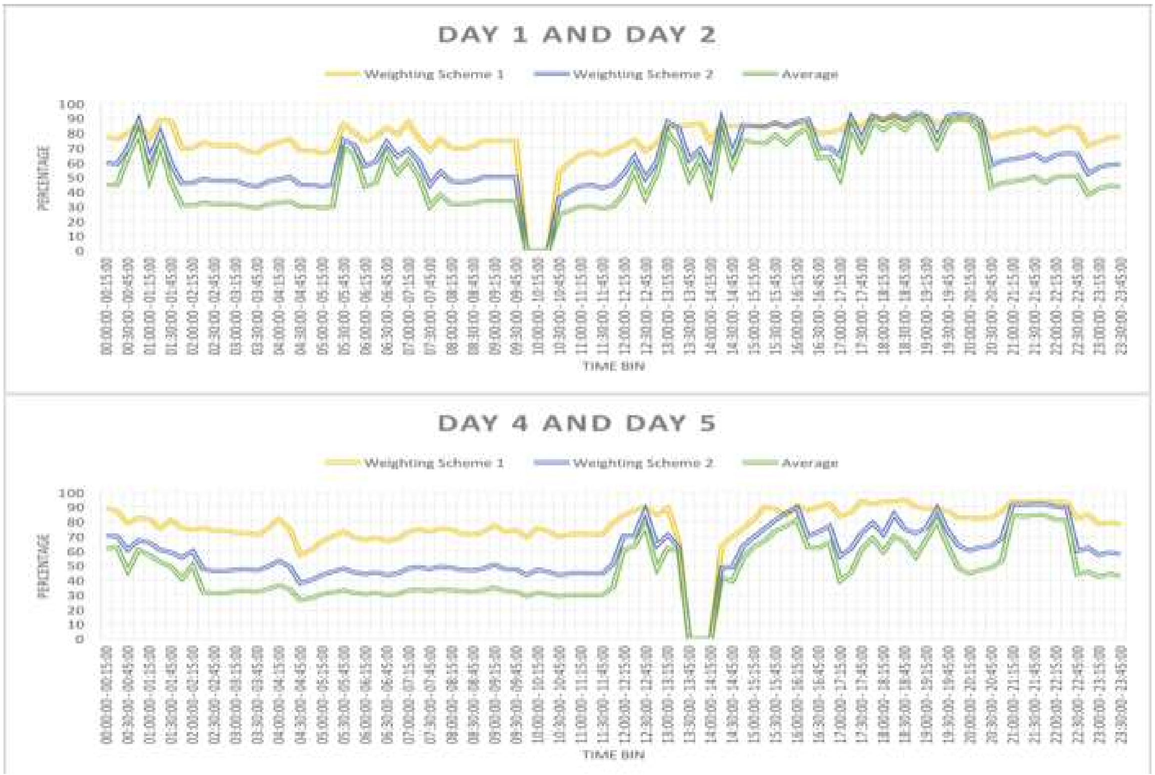


Figure 44 Comparison of Different Weighting Schemes (User 4)

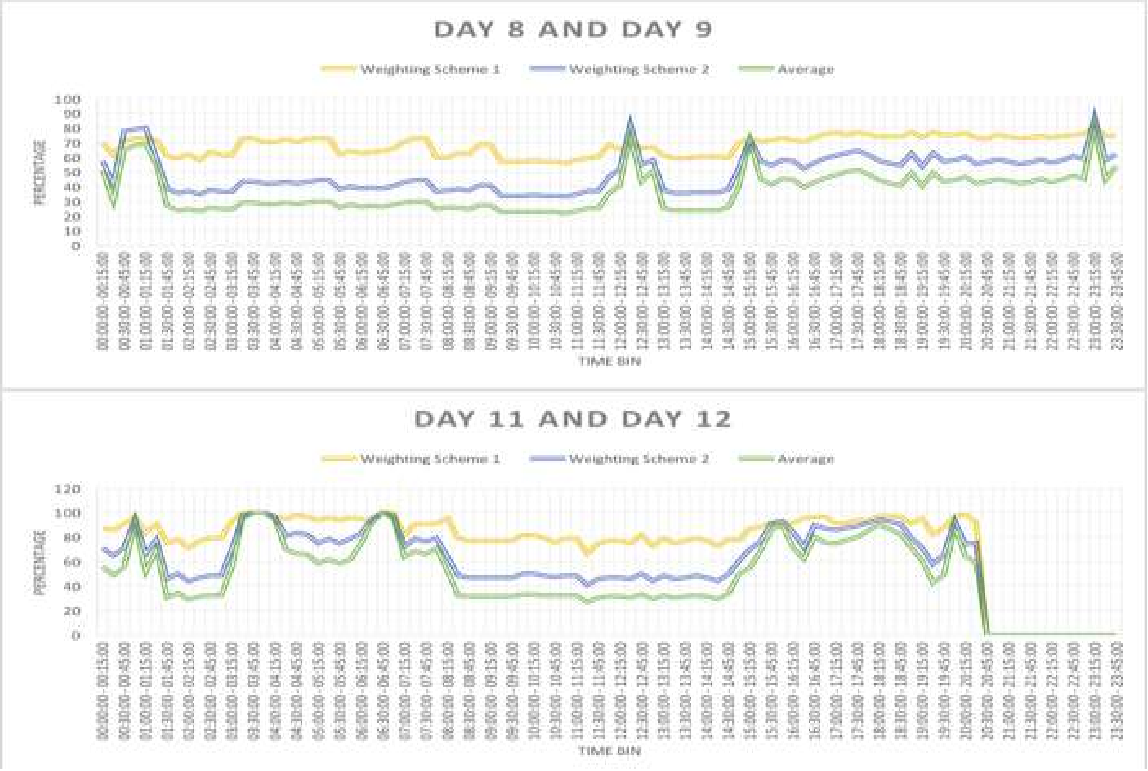


Figure 45 Comparison of Different Weighting Schemes (User 4)

6.1.5 Evaluation analysis for User 5

Figure 46 shows the match percentage of application usage for user 5 in every time bin for two consecutive random days. We have expanded the graph for 2 hours' duration in order to show clear picture of the results. It is clear from the Figure 46 that for this user there is not much difference in the threshold of different time bins for application usage except for the fact that in certain time bins the match percentage is 0 that mean no stability at all.

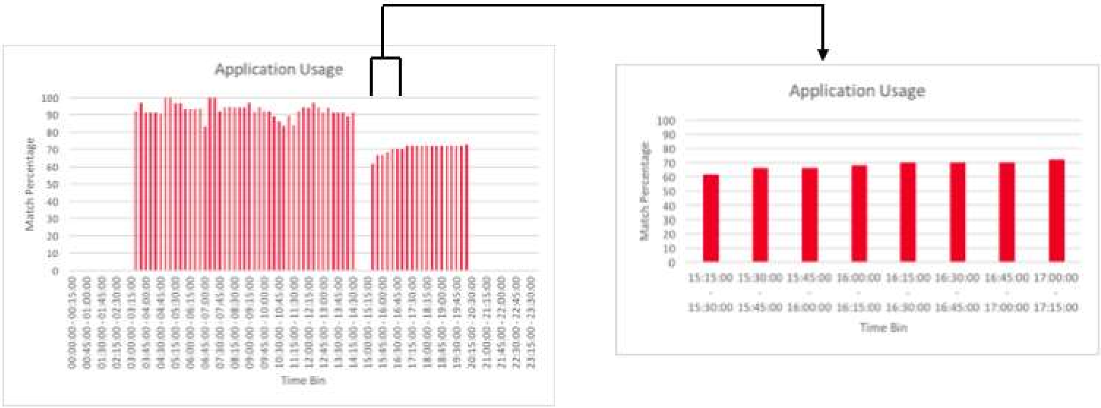


Figure 46 Match Percentage in Different Time Bins for Application Usage (User 5)

Figure 47 shows the results of match percentages of user 5 for cell tower in each time interval for randomly selected two consecutive days. The graphs show us that user 5 is not showing any steadiness in usage of smartphone in terms of cell tower for most of the time.

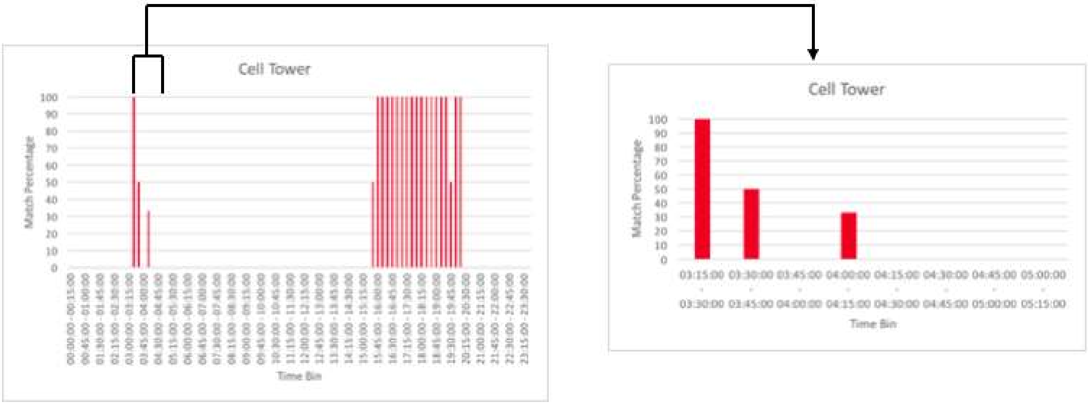


Figure 47 Match Percentage in Different Time Bins for Cell Tower (User 5)

Figure 48 shows the same type of information as of cell towers for Wi-Fi Networks. Either there is high variation or it is 0.

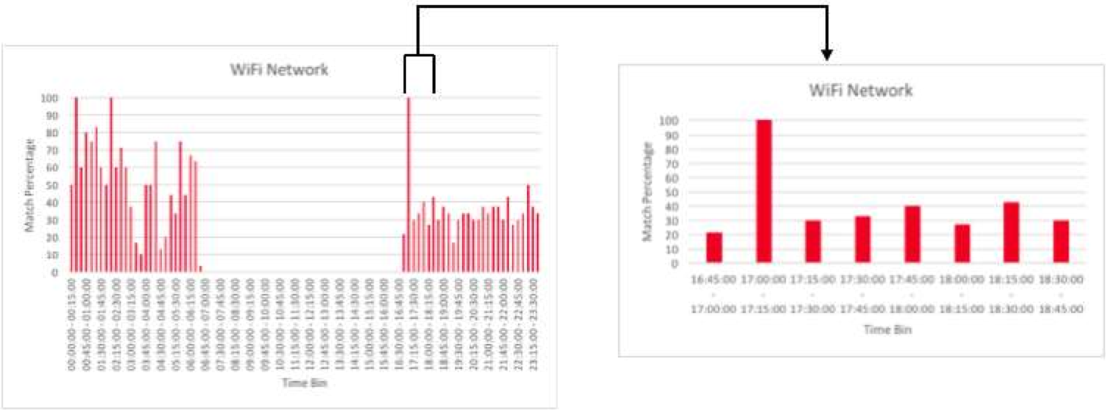


Figure 48 Match Percentage in Different Time Bins for Wi-Fi Networks (User 5)

Hence, it is clear from the above observations that user 5 also shows stability in usage of the smartphone during certain time intervals but not in the entire day.

The next thing we investigated is the need of assigning weights to the sensors. As a proof of our argument we generated graphs for four random time bins of each day for every sensor. These graphs are shown in Figure 49 and Figure 50.

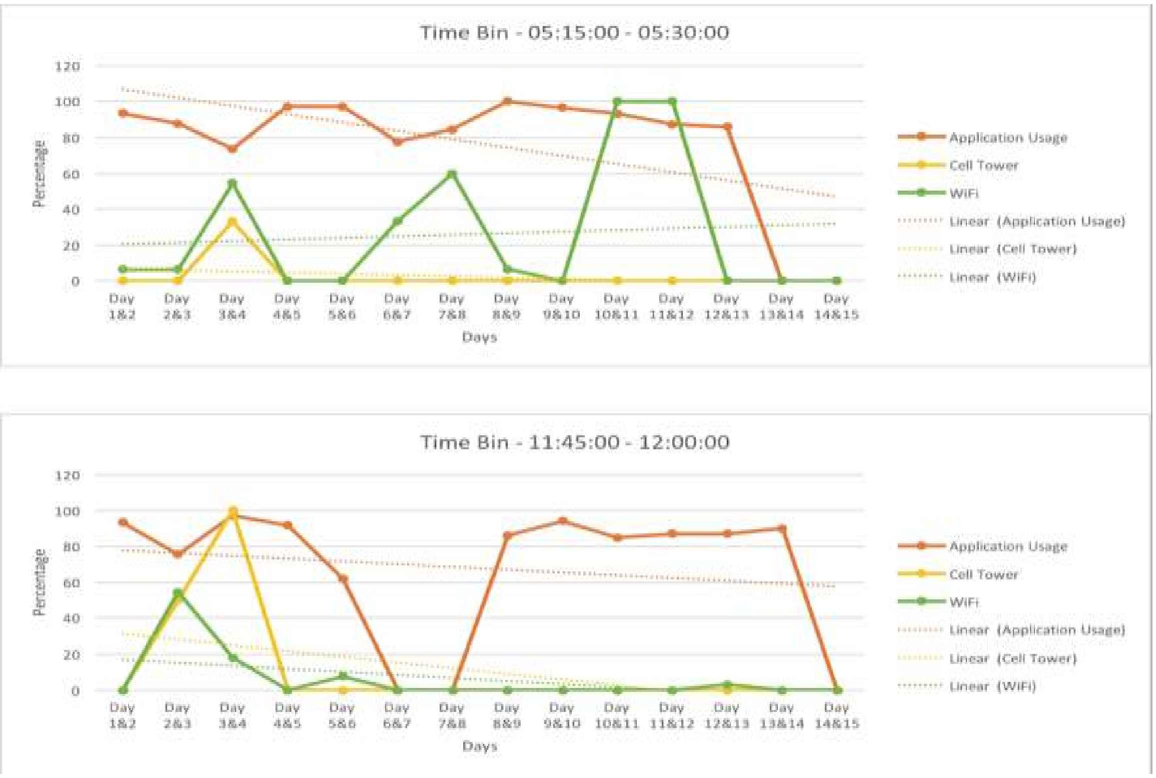


Figure 49 Stability of Different Sensors in Random Time Bins (User 5)

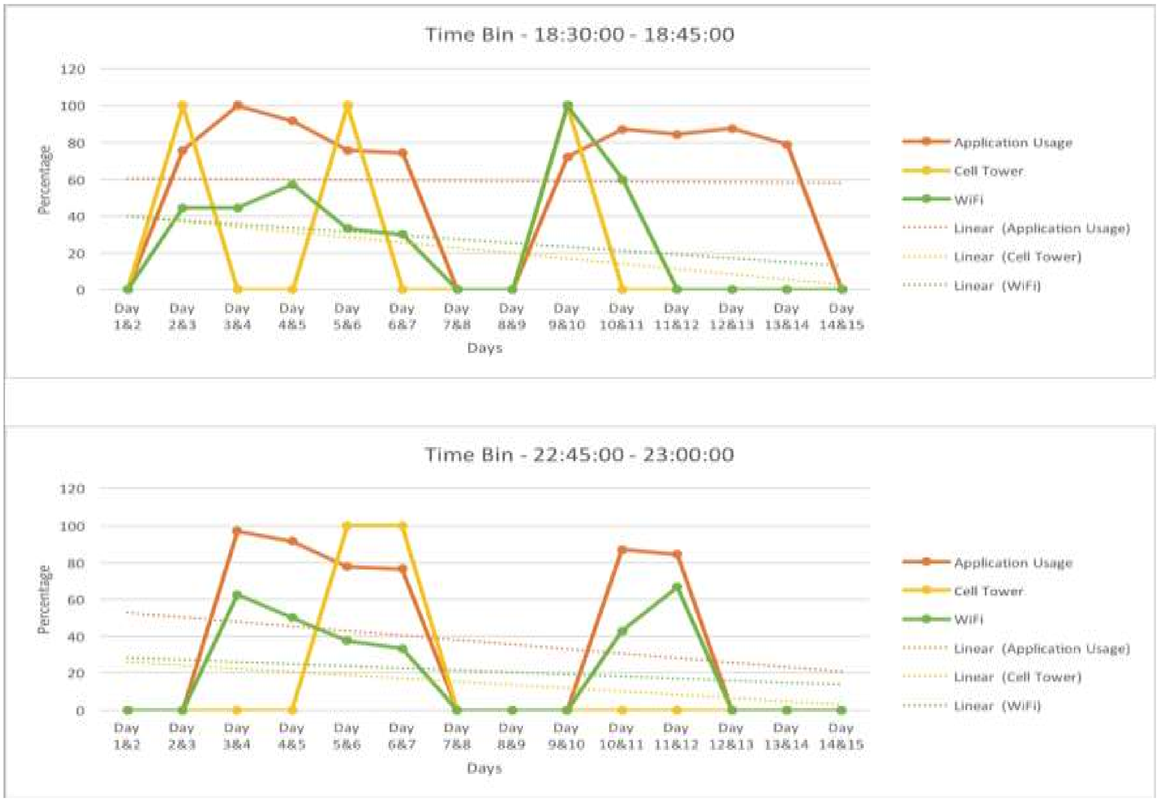


Figure 50 Stability of Different Sensors in Random Time Bins (User 5)

From these graphs we can clearly see that over a period of 15 days, user 5 is not showing much stability in smartphone usage for any of the sensors for most of the days and time intervals.

So, when we applied different weighting techniques for this user, we observed that even weighting scheme 1 is not giving as good results as it is giving for other users. This is because of the fact that this user is not showing much stability for any of the given sensors in this dataset. The results of this comparison are shown in Figure 51 and Figure 52.

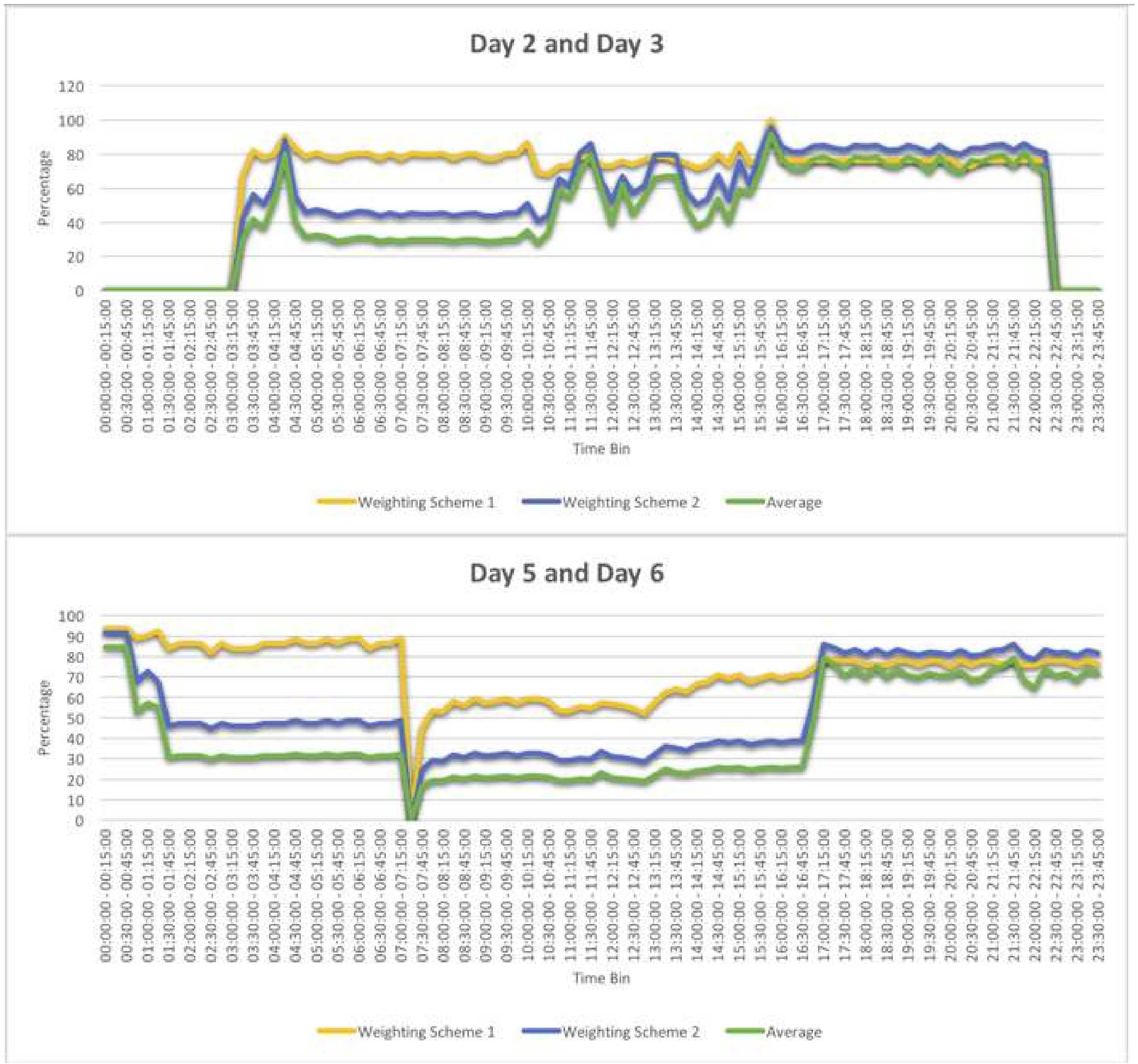


Figure 51 Comparison of Different Weighting Schemes (User 5)

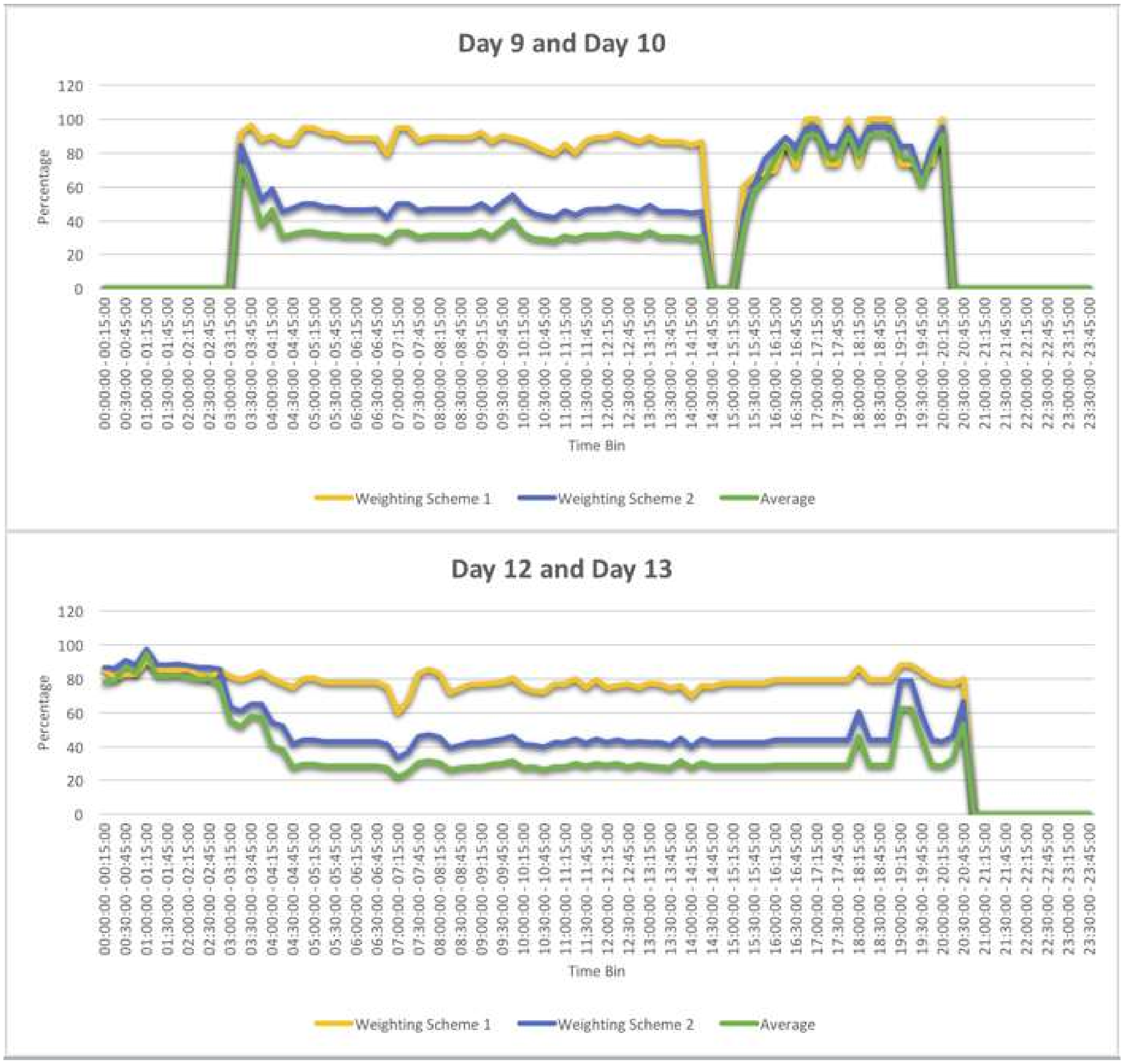


Figure 52 Comparison of Different Weighting Schemes (User 5)

We observed from the graphs that despite of the fact that even weighting scheme 1 is not good for this user for most of the days, it is still performing better than weighting scheme 2 and simple average in few of the time intervals.

6.1.6 Evaluation analysis for User 6

Figure 53 shows the match percentage of user 6 for application usage in every time bin for two consecutive random days. We have expanded the graph for 2 hours' duration in order to show clear picture of the results. It is clear from the Figure 53 that for this user there is slight difference in the threshold of different time bins for application usage.

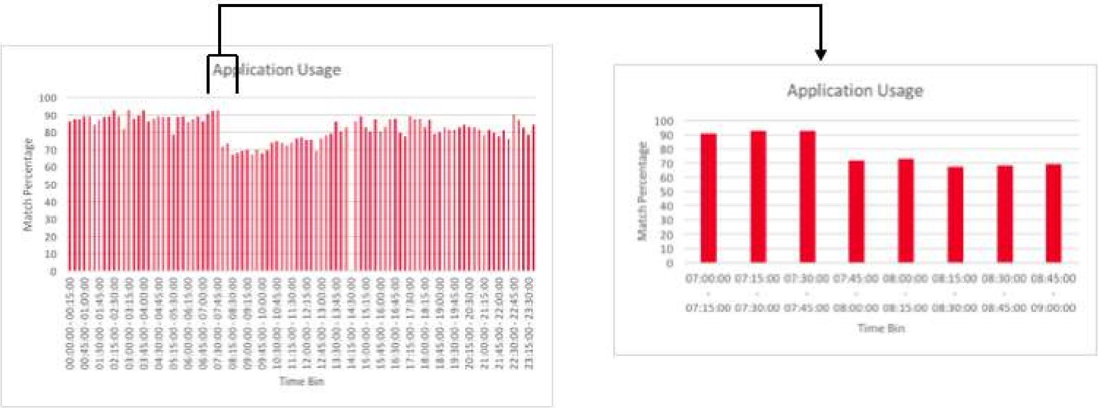


Figure 53 Match Percentage in Different Time Bins for Application Usage (User 6)

Figure 54 shows the results of match percentages of user 6 for cell tower in each time interval for randomly selected two consecutive days. The graphs show us that there is high variation in the percentages for different time bins.

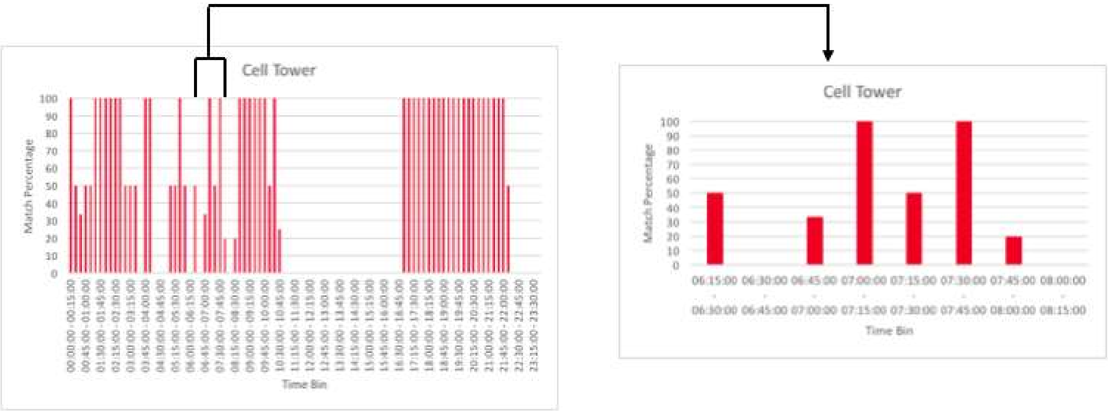


Figure 54 Match Percentage in Different Time Bins for Cell Tower (User 6)

Figure 55 shows the same information for Wi-Fi Networks. This sensor also shows disparity in the match percentages for different time bins.

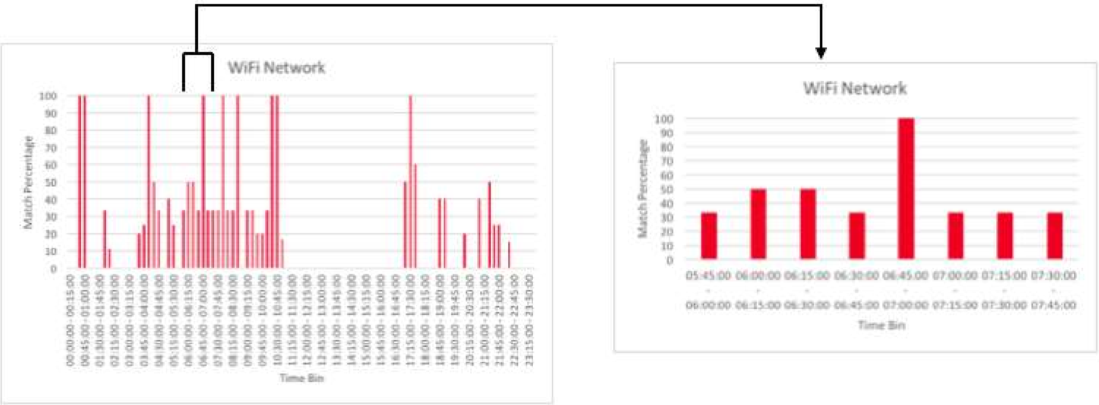


Figure 55 Match Percentage in Different Time Bins for Wi-Fi Networks (User 6)

Hence, it is clear from the above observations that user 6 shows stability in usage of the smartphone during certain time intervals but not in the entire day except for the application usage.

The next thing we investigated is the need of assigning weights to the sensors. As a proof of our argument we generated graphs for four random time bins of each day for every sensor. These graphs are shown in Figure 56 and Figure 57.

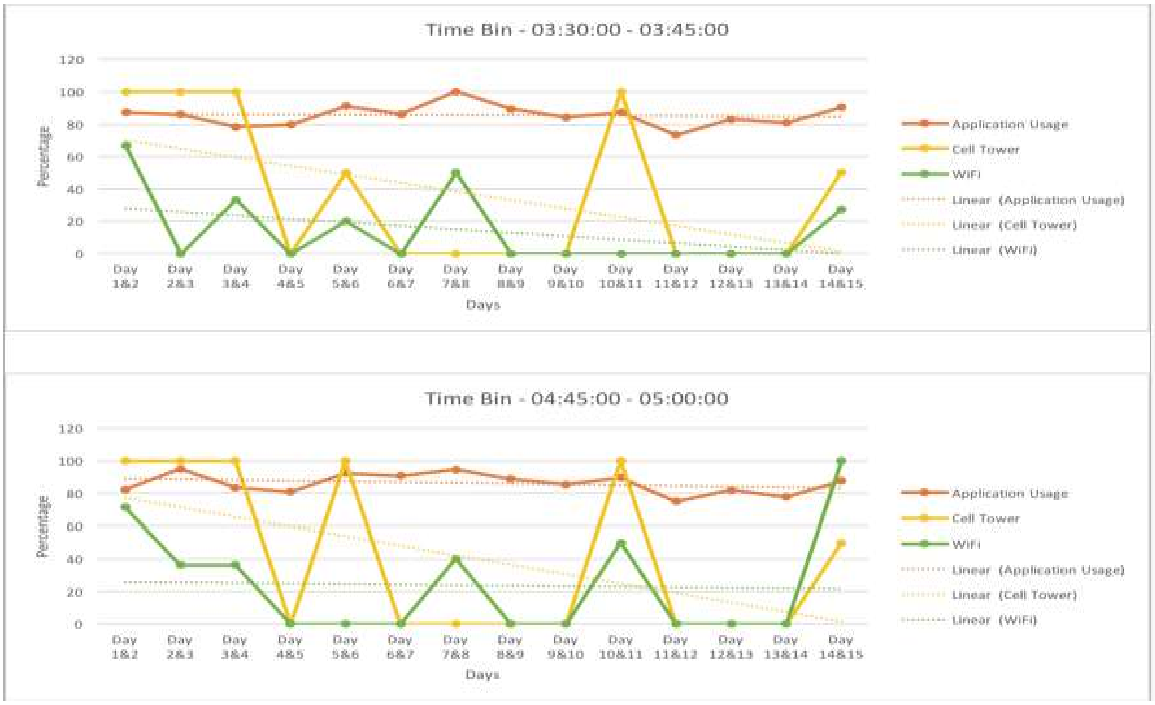


Figure 56 Stability of Different Sensors in Random Time Bins (User 6)

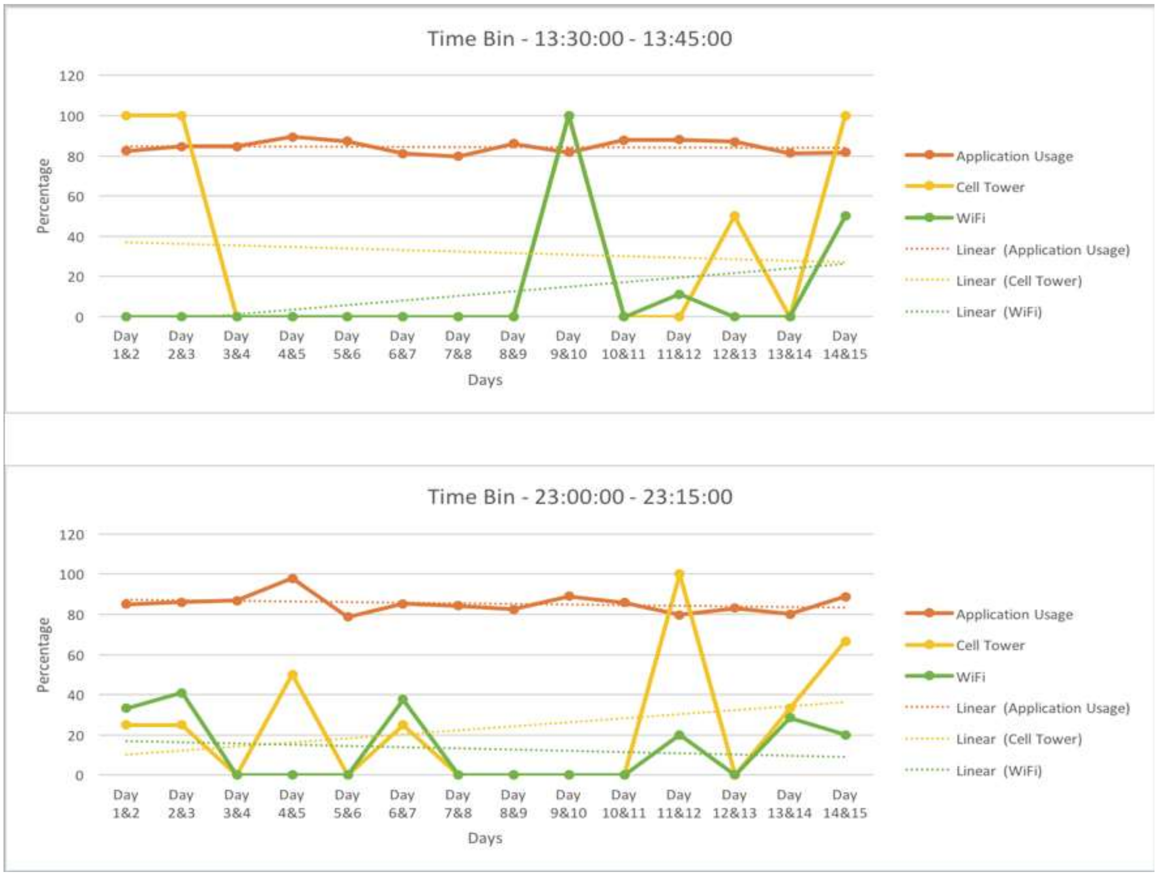


Figure 57 Stability of Different Sensors in Random Time Bins (User 6)

From these graphs we can clearly see that user is showing more stability in application usage followed by the cell tower and then Wi-Fi for most of the days.

We assigned weights to different sensors for this user. The results of this comparison are shown in Figure 58 and Figure 59.

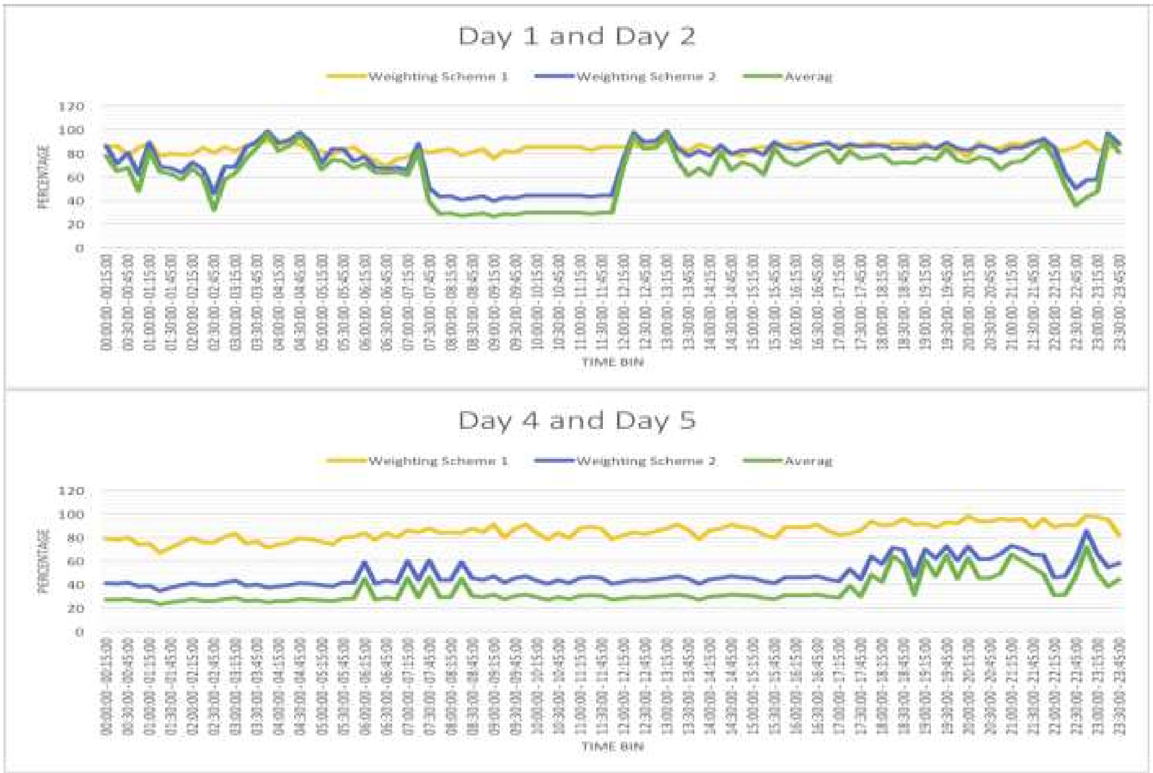


Figure 58 Comparison of Different Weighting Schemes (User 6)

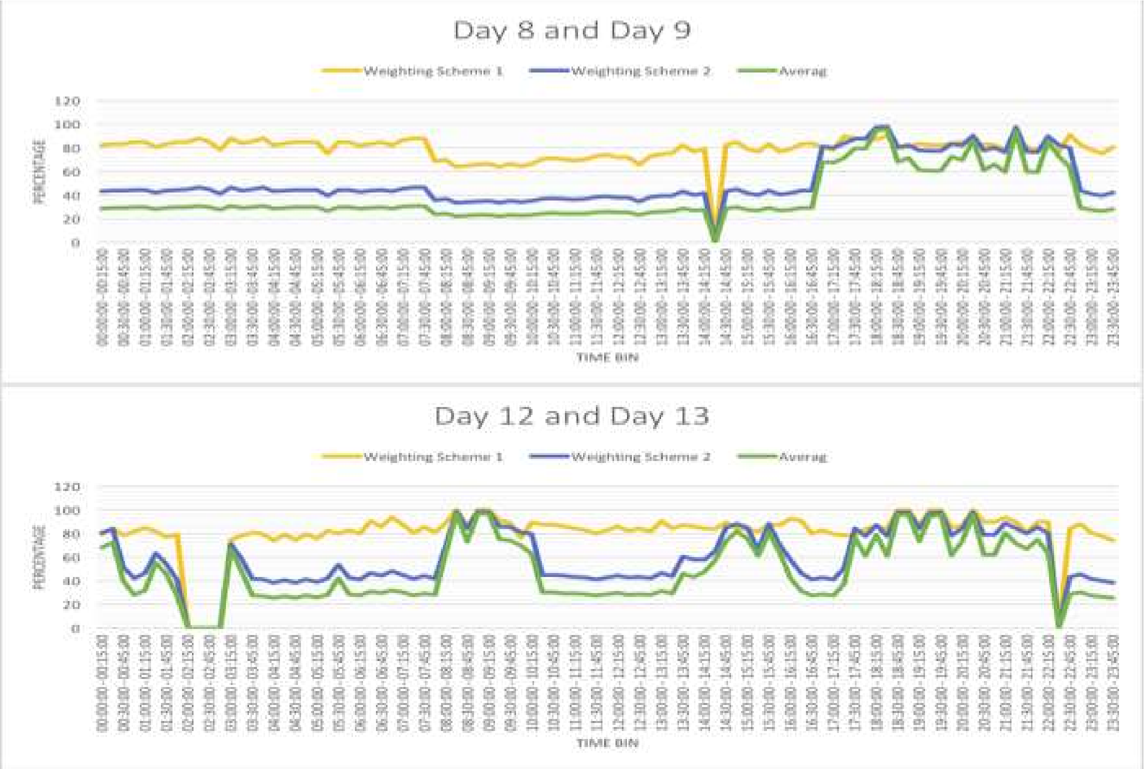


Figure 59 Comparison of Different Weighting Schemes (User 6)

We observed from the graphs that the weighting scheme 1 is giving the high aggregate percentage followed by weighting scheme 2 and average for most of the time bins. So, weighting scheme 1 is showing better results for user 6.

6.1.7 Evaluation analysis for User 7

Figure 60 shows the match percentage of application usage for user 7 in every time bin for two consecutive random days. We have expanded the graph for 2 hours' duration in order to show clear picture of the results. It is clear from the Figure 60 that for this user there is not much difference in the threshold of different time bins for application usage.

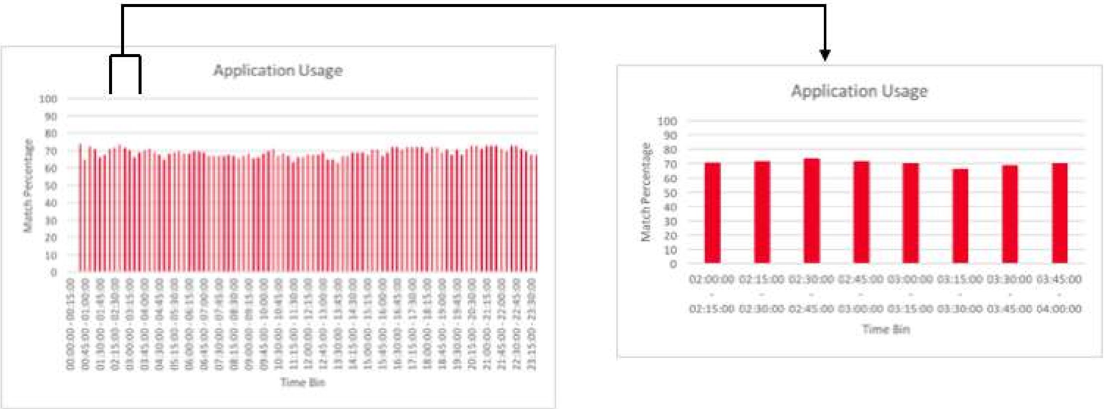


Figure 60 Match Percentage in Different Time Bins for Application Usage (User 7)

Figure 61 shows the results of match percentages of cell tower for user 7 in each time interval for randomly selected two consecutive days. The graphs show us that there is not much variation in the percentages for different time bins except that it is 0 for most of the time bins.

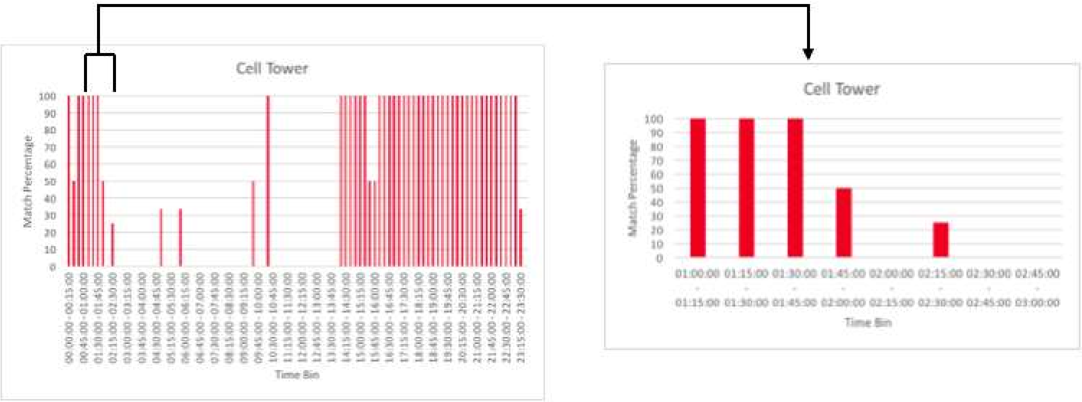


Figure 61 Match Percentage in Different Time Bins for Cell Tower (User 7)

Figure 62 shows disparity in the match percentages of user 7 for different time bins for Wi-Fi networks.

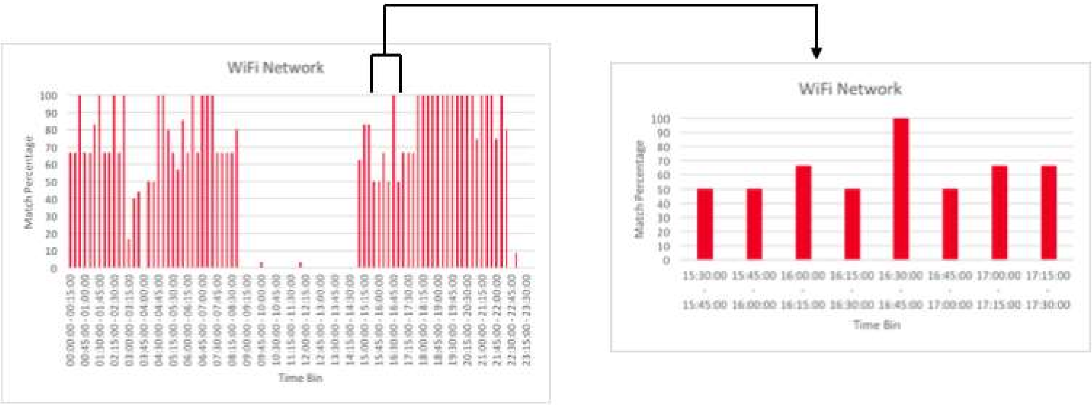


Figure 62 Match Percentage in Different Time Bins for Wi-Fi Networks (User 7)

Hence, it is clear from the above observations that this particular user shows stability in usage of the smartphone during certain time intervals but not in the entire day.

The next thing we investigated is the need of assigning weights to the sensors. As a proof of our argument we generated graphs for four random time bins of each day for every sensor. These graphs are shown in Figure 63 and Figure 64.

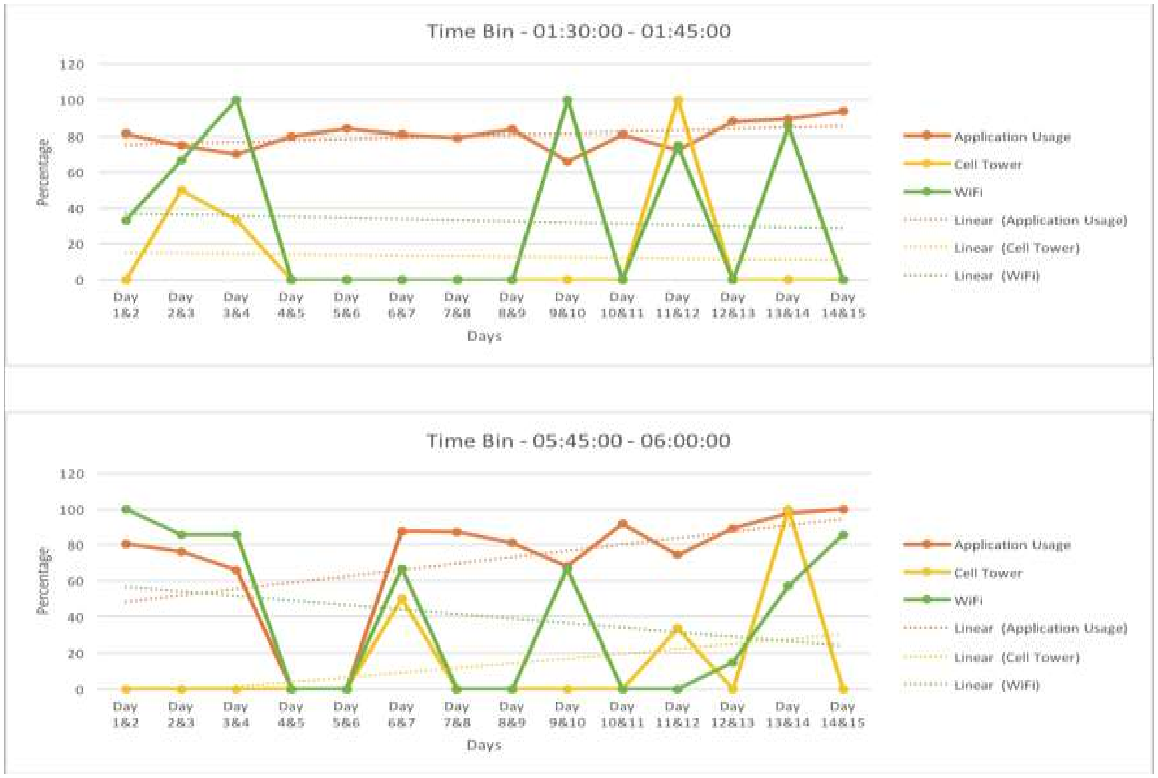


Figure 63 Stability of Different Sensors in Random Time Bins (User 7)

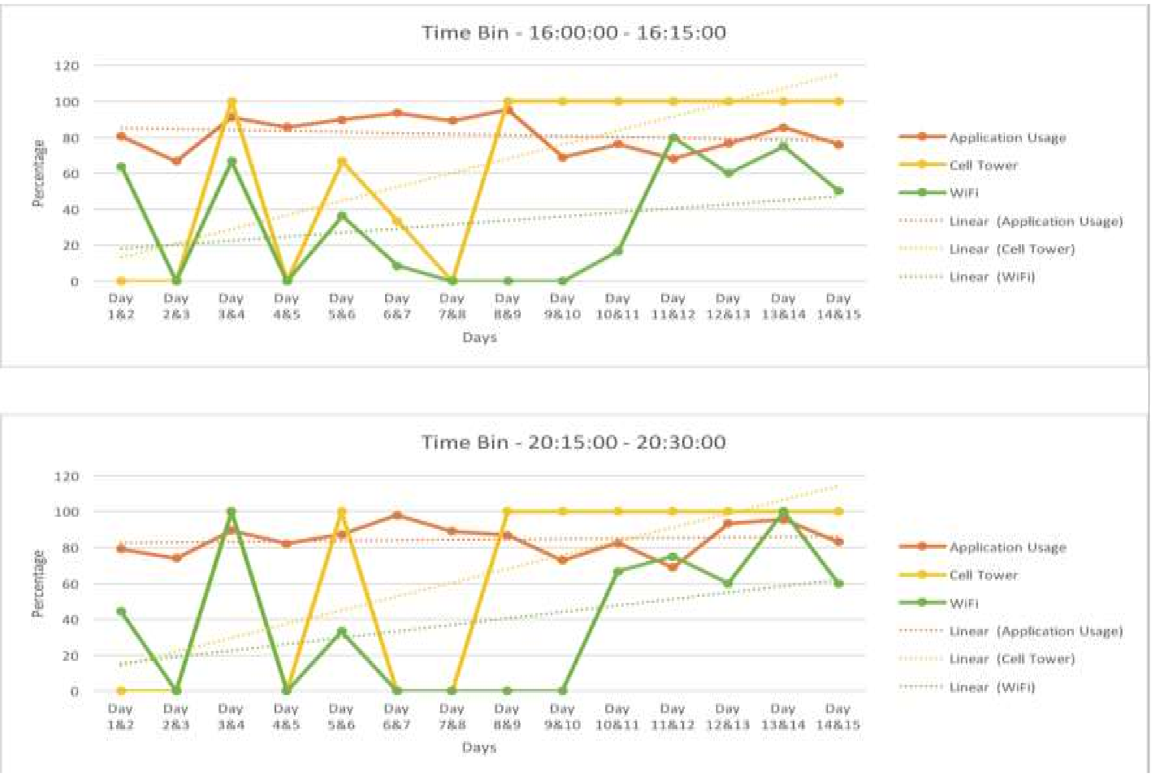


Figure 64 Stability of Different Sensors in Random Time Bins (User 7)

From these graphs we can clearly see that user 7 is showing more stability in application usage for sure. But as far as Wi-Fi and cell towers are concerned, for certain time bins this user is showing more stability for Wi-Fi than cell tower and for some time intervals it is vice versa.

Weighted average is calculated for this user as well and the comparison of results are shown in Figure 65 and Figure 66.

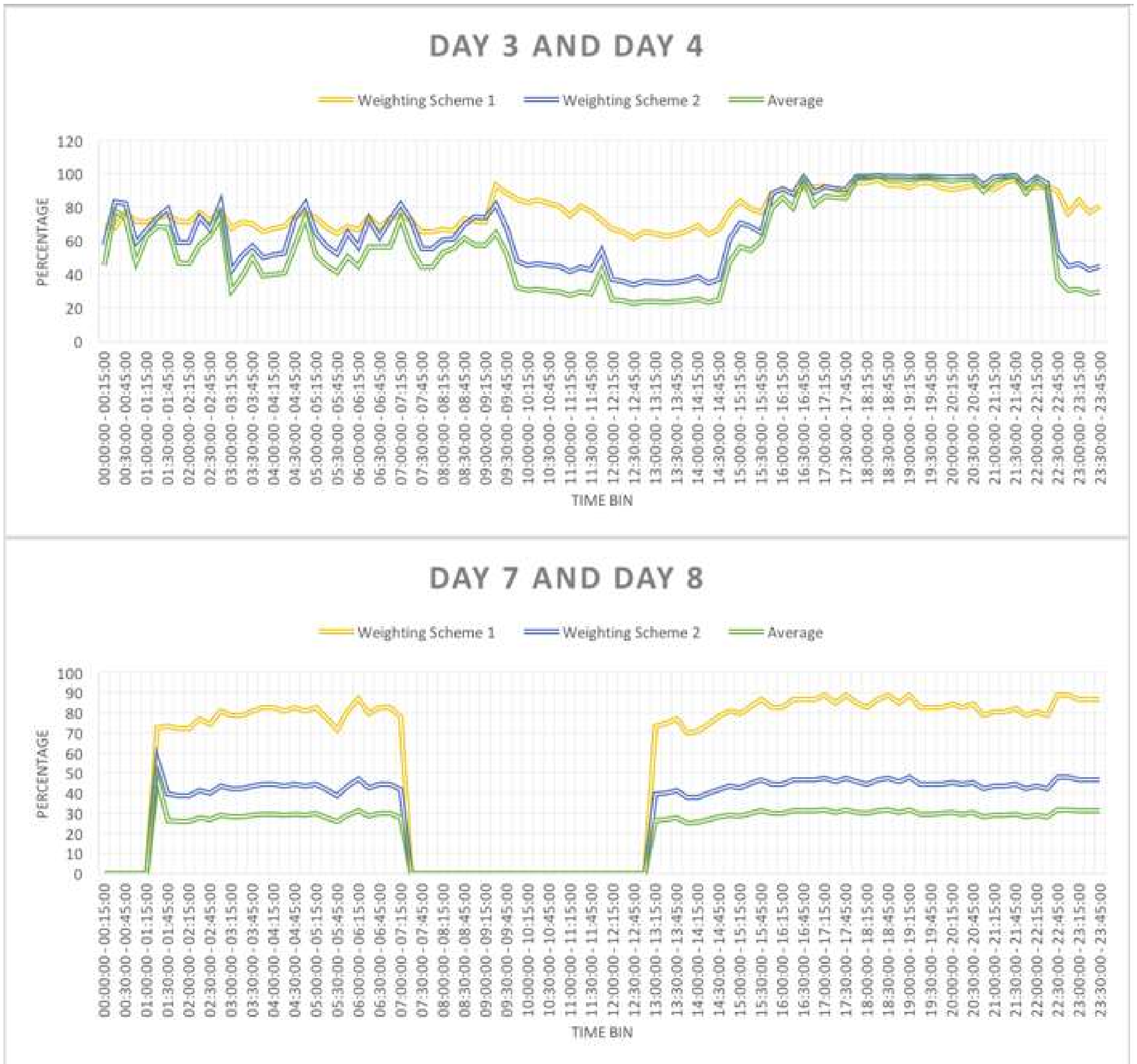


Figure 65 Comparison of Different Weighting Schemes (User 7)

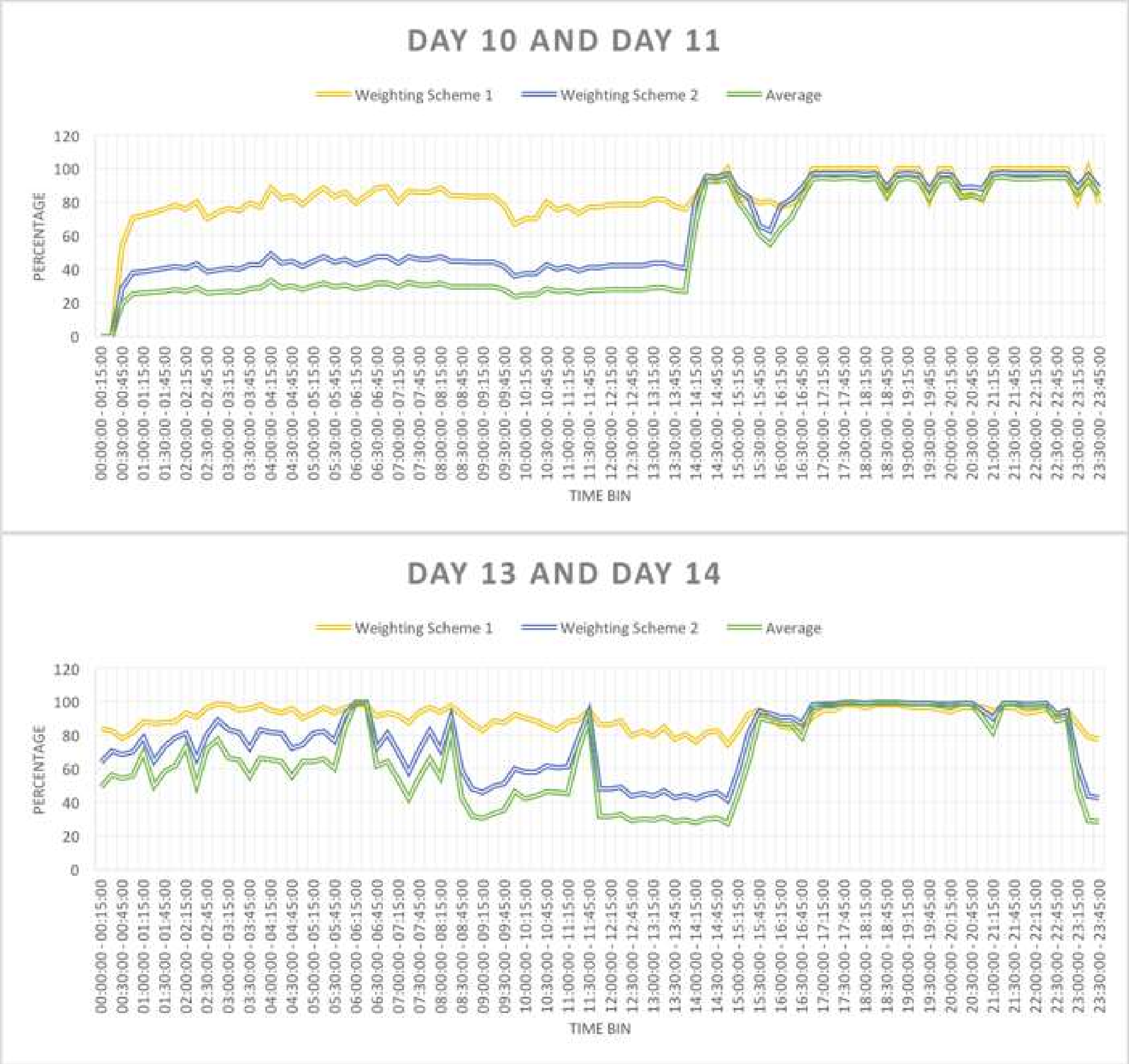


Figure 66 Comparison of Different Weighting Schemes (User 7)

We observed from the graphs that for this user either the weighting scheme 1 is giving the high aggregate percentage followed by weighting scheme 2 and average or they are approximately equal.

6.2 SUMMARY OF THE EVALUATION RESULTS

From the evaluation analysis we observed that most of the users are showing stability in application usage for the entire day but that is not the case for cell tower and Wi-Fi network. So, it is beneficial to create user profiles for short time intervals instead of creating them for the entire day. In that way, we will be having different thresholds, based on which the device can determine whether to opt for implicit or for explicit authentication. Herein, the user need not use any password or any type of explicit authentication in those time intervals where he is steady in using his device or in short, where the match percentage is significantly high. Also if in any time interval the user is showing medium stability he/she will be having limited access to the device features in that time interval. That level of limitation can be set by the user in the security and privacy settings. Finally, if the user is not showing any stability in a time interval, he/she always need to use explicit authentication in that particular time bin. So, the match percentage in different time intervals will decide that whether or not, the data can be used for authentication. Also these percentages will decide how much access of resources should be given to the user. This is generalized in the following equation:

Let X be value of match percentage, then the decision can be made as follows:

$$X\% \sim 0 -\!\!\rightarrow The\ data\ is\ not\ significant\ to\ be\ used\ for\ authentication$$

$$X\% \sim 50 -\!\!\rightarrow The\ data\ is\ not\ entirely\ significant\ to\ be\ used\ for\ authentication,$$
$$so\ limited\ access$$

$$X\% \sim 100 -\!\!\rightarrow The\ data\ is\ significant\ to\ be\ used\ for\ authentication$$

The benefit of using weights is, the sensor yielding more information about the user will be given more importance and hence will increase the threshold. The higher the value of threshold for a user, the more difficult it would be for someone to impersonate his/her activities. If we do not apply any weighting technique and calculate the simple average than all the sensors will be given equal importance. So, the sensor for which the user is not showing any stability at all, will decrease the overall threshold and it will be easier to impersonate the usage patterns of that particular user.

CHAPTER 7 CONCLUSION

In this thesis, we studied how smartphone usage & network data can be used for authentication & access control. For this purpose, we used a publicly available dataset containing data from cell tower, Wi-Fi network and application usage of 7 users. The data was divided into time bins in order to investigate variations in thresholds for a user in different times of a day. We suggested that based on the match percentages, the significance of data to be used for authentication and access control can be decided. We have also evaluated two weighting techniques to assign weights to the sensors. The results of our evaluation showed that by applying weighting schemes, the match percentage can be increased which might be helpful in mitigating impersonation attacks in implicit authentication scheme.

For this research the dataset which we used was small containing data for only two weeks. Hence we have not considered whether the day is a weekday or weekend. But it can be implemented by using our approach. This can be achieved with an additional tag which represents whether it is a weekday. The tag can be added to user profile to differentiate between the pattern of weekday and weekend. While comparing the data, user behavior pattern of weekday will be matched to another weekday and pattern of weekend will be matched with weekend. The rest of the approach will work in the similar way.

We have not tested our approach on real time streaming data. So at this point of time we cannot comment on how it will perform on data coming live from mobile devices. But we can apply this approach in addition with some classification algorithm in order to increase the accuracy and reliability of our approach. It will involve the training and the testing phases. When the user buys a phone with this feature of implicit authentication, the device will learn his behavior for few days. In that training phase, the device will generate some patterns of user profile for different time bins of both weekday and weekend based on some machine learning algorithm. It will also decide about the priorities of those sensors in the training phase. In the testing phase, the data coming from the device will be grouped into time bins and classification accuracy will be calculated for every 15 minutes (or the size of the time bin). Based on that accuracy, the user will be given access to the device.

7.1 LIMITATIONS

Even though the results of our evaluations are good for the dataset we used, there are some limitations of our approach. The first limitation is, we tested our approach on a publicly available dataset. We are not sure how it would affect if the approach were to be applied on real time Live data. The second limitation is that, we utilized only one dataset for evaluating our approach and hence we do not know how it will perform when we have other datasets with more sensors values stored in it. The third limitation is that for every user we compared the data between two consecutive days. But we couldn’t observe how our approach would behave when the data is compared between 3 or 4 consecutive days.

7.2 FUTURE WORK

Our future works includes conducting a user study to collect real time data from the application we developed and test our approach on that data. We would also like to test our approach on the other available similar datasets.

In addition, we would also like to make this a hybrid approach, so that it should work fine if the device has to be shared by multiple users. In such scenarios, the device will learn different behaviors during the training phase and when the testing phase starts, it will match the user behavior with all the stored patterns. If any of the pattern is matched, then the user will be considered as an authentic user.

BIBLIOGRAPHY

[1] Confident Technologies, "New Survey Reveals That Smartphone Users Choose Convenience Over Security," [Online]. Available: http://www.marketwired.com/press-release/new-survey-reveals-that-smartphone-users-choose-convenience-over-security-1566290.htm. [Accessed 12 October 2016].

[2] Intel Security, "More Than 30% of People Don't Password Protect Their Mobile Devices," [Online]. Available: https://blogs.mcafee.com/consumer/unprotected-mobile-devices/. [Accessed 19 August 2016].

[3] H. G. Kayacik, M. Just, L. Baillie, D. Aspinall and N. Micallef, "Data Driven Authentication: On the Effectiveness of User Behaviour Modelling with Mobile Device Sensors," in *Proceedings of the 3rd Mobile Security Technologies Workshop , Held as part of the IEEE S&P Symposium, (MoST-2014)*, May 2014.

[4] E. Shi, Y. Niu, M. Jakobsson and R. Chow, "Implicit Authentication through Learning User Behavior," in *Proceedings of the 13th international conference on Information security, ser. ISC'10*, Berlin, Heidelberg, 2011.

[5] M. Rouse, "SearchSecurity," TechTarget, [Online]. Available: http://searchsecurity.techtarget.com/definition/authentication. [Accessed 9 July 2016].

[6] Oracle, "Specifying an Authentication Mechanism," Oracle, [Online]. Available: https://docs.oracle.com/cd/E19226-01/820-7627/bncbn/index.html. [Accessed 3 October 2016].

[7] Wikipedia, "Multi-factor authentication," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Multi-factor_authentication. [Accessed 9 July 2016].

[8] Graham Cluley, "Two-factor authentication (2FA) versus two-step verification (2SV)," Graham Cluley, [Online]. Available: https://www.grahamcluley.com/factor-authentication-2fa-versus-step-verification-2sv/. [Accessed 3 October 2016].

[9] BTE Technology Group, "Two-Factor Authentication," [Online]. Available: http://bte.co.ug/our-solutions/security-advisory/high-assurance-authentication/two-factor-authenticatiion/. [Accessed 3 October 2016].

[10] Gizmodo, "It's Time to Enable Two-Step Authentication on Everything," 16 June 2015. [Online]. Available: http://gizmodo.com/its-time-to-enable-two-step-authentication-on-everythin-1646242605. [Accessed 9 July 2016].

[11] TechTarget, "WhatIs," TechTarget, [Online]. Available: http://whatis.techtarget.com/definition/sensor. [Accessed 9 July 2016].

[12] Android, "Developers," Google Android, [Online]. Available: https://developer.android.com/guide/topics/sensors/sensors_overview.html. [Accessed 9 July 2016].

[13] Buzinga, "Why Mobile HealthTech Is So Much More Than Fitness," 19 August 2015. [Online]. Available: http://www.buzinga.com.au/buzz/mobile-health-tech/. [Accessed 3 October 2016].

[14] S. Sampalli, *Wireless Networks [Class Handout],* Halifax, NS: Faculty of Computer Science, Dalhousie University, 2014.

[15] V. Krishnasamy, "Authentication And Access Control As A Way Of Securing Our System," May 1995. [Online]. Available: http://www.doc.ic.ac.uk/~nd/surprise_95/journal/vol2/vk3/article2.html. [Accessed 27 October 2016].

[16] Code Project, "Custom Role Based Access Control," [Online]. Available: http://www.codeproject.com/Articles/875547/Custom-Roles-Based-Access-Control-RBAC-in-ASP-NET. [Accessed 27 October 2016].

[17] M. Z. Shafiq, L. Ji, A. X. Liu, J. Pang and J. Wang, "A First Look at Cellular Machine-to-Machine Traffic – Large Scale Measurement and Characterization," in *SIGMETRICS '12 Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, New York, 2012.

[18] G. Maier, F. Schneider and A. Feldmann, "A First Look at Mobile Hand-held Device Traffic," in *International Conference on Passive and Active Network Measurement* , Berlin Heidelberg, 2010.

[19] Y. Li, J. Yang and N. Ansari, "Cellular Smartphone Traffic and User Behavior Analysis," in *International Conference on Communications (ICC)*, 2014.

[20] J. Liu, H. Wu and H. Wang, "A detection method for malicious codes in Android apps," in *10th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2014)*, Beijing, 2014.

[21] E. Finickel, A. Lahmadi, F. Beck and O. Festor, "Empirical Analysis of Android Logs Using Self-Organizing Maps," in *2014 IEEE International Conference on Communications (ICC)*, Sydney, NSW, 2014.

[22] I. Burguera, U. Zurutuza and S. Nadjm-Tehrani, "Crowdroid: Behavior-Based Malware Detection System for Android," in *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, New York, 2011.

[23] Y. Hirabe, Y. Arakawa and K. Yasumoto, "Logging all the touch operations on Android," in *2014 Seventh International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, Singapore, 2014.

[24] J. Tian, G. Wang, X. Gao and K. Shi, "User Behavior based Automatical Navigation System on Android Platform," in *2014 23rd Wireless and Optical Communication Conference (WOCC)*, Newark, NJ, 2014.

[25] L. Bedogni, M. D. Felice and L. Bononi, "By train or by car? Detecting the user's motion type through smartphone sensors data," in *Wireless Days (WD), 2012 IFIP* , Dublin, 2012.

[26] Y. Ma, B. Xu, Y. Bai, G. Sun and R. Zhu, "Daily Mood Assessment based on Mobile Phone Sensing," in *2012 Ninth International Conference on Wearable and Implantable Body Sensor Networks* , London, 2012.

[27] C. Bo, L. Zhang, T. Jung, J. Han, X.-Y. Li and Y. Wang, "Continuous user identification via touch and movement behavioral biometrics," in *2014 IEEE 33rd*

International Performance Computing and Communications Conference (IPCCC), Austin, TX, 2014.

[28] Y. Ren, Y. Chen, M. C. Chuah and J. Yang, "User Verification Leveraging Gait Recognition for Smartphone Enabled Mobile Healthcare Systems," *IEEE Transactions on Mobile Computing* , vol. 14, no. 9, pp. 1961 - 1974, 2015.

[29] W. Shi, J. Yang, Y. Jiang, F. Yang and Y. Xiong, "SenGuard: Passive user identification on smartphones using multiple sensors," in *2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Wuhan, 2011.

[30] F. Yao, S. Y. Yerima, B. Kang and S. Sezer, "Event-Driven Implicit Authentication for Mobile Access Control," in *9th International Conference on Next Generation Mobile Applications, Services and Technologies*, 2015.

[31] A. Buriro, B. Crispo, F. D. Frari, J. Klardie and K. Wrona, "ITSME: Multi-modal and Unobtrusive Behavioural User Authentication for Smartphones," in *International Conference on Passwords*, 2015.