Behaviour based anomaly detection system for smartphones using machine learning algorithm

Khurram Majeed

Faculty of Life Sciences and Computing

London Metropolitan University

This dissertation is submitted for the degree of

Doctor of Philosophy

January 2015

I would like to dedicate this thesis to my loving parents . . .

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other University. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration, except where specifically indicated in the text. This dissertation contains around 30,000 words including appendices, bibliography, footnotes, tables and equations and has 115 references.

Khurram Majeed

January 2015

Acknowledgements

Words can't explain my gratitude for Almighty Allah without whose mercy I would never be able to achieve what I have. I would like to express my special appreciation and thanks to Dr. Dusica Novakovic, Prof. Karim Ouazzane and Dr. Yanguo Jing for being great pillars of this research process, with their kind supervision, valuable suggestions and intellectual activities, inexhaustible energy to steer forth the research and keeping my spirits up during this journey, without which it would have been incomplete. I would also like to thank London Metropolitan University for providing me Scholarship and staff at the School of Computing and research office for their support during my studies. I would also like to extend my heartfelt gratitude to my fellow research students, especially, Majid Djennad, Samson Habte, Deepthi Ratnayake, Teresa Formisano, Victor Costonov and Abu Mansoor for making our research time together, truly memorable and supporting me in the moments when there was no one to answer my queries. Last but not least, I would like to extend special thanks to my family. Words cannot express how grateful I am to my parents, my wife and her parents for all of the sacrifices that they have made on my behalf. Their prayer for me is what sustained me thus far.

Abstract

In this research, we propose a novel, platform independent behaviour-based anomaly detection system for smartphones. The fundamental premise of this system is that every smartphone user has unique usage patterns. By modelling these patterns into a profile we can uniquely identify users. To evaluate this hypothesis, we conducted an experiment in which a data collection application was developed to accumulate real-life dataset consisting of application usage statistics, various system metrics and contextual information from smartphones. Descriptive statistical analysis was performed on our dataset to identify patterns of dissimilarity in smartphone usage of the participants of our experiment. Following this analysis, a Machine Learning algorithm was applied on the dataset to create a baseline usage profile for each participant. These profiles were compared to monitor deviations from baseline in a series of tests that we conducted, to determine the profiling accuracy. In the first test, seven day smartphone usage data consisting of eight features and an observation interval of one hour was used and an accuracy range of 73.41% to 100% was achieved. In this test, 8 out 10 user profiles were more than 95% accurate. The second test, utilised the entire dataset and achieved average accuracy of 44.50% to 95.48%. Not only these results are very promising in differentiating participants based on their usage, the implications of this research are far reaching as our system can also be extended to provide transparent, continuous user authentication on smartphones or work as a risk scoring engine for other Intrusion Detection System.

Table of contents

List of figures

List of tables

Nomenclature

Acronyms / Abbreviations

1G	First Generation Wireless Network
2G	Digital Cellular Network
3G	Mobile Broadband Network
4G	Fourth Generation Wireless Network
ANN	Artificial Neural Network
ASPECT	Advanced Security for Personal Communication Technologies
CFCA	Communications Fraud Control Association
CSV	Comma Separated Values
DOS	Denial-of-Service
EER	Expected Error Rate
EWMA	Exponentially Weighted Moving Model
FAR	False Accept Rate
FDMA	Frequency Division Multiple Access
GPRS	General Packet Radio Service

GPS	Global Positioning System
GSM	Global System for Mobile
HIDS	Host-Based Intrusion Detection System
HSDPA	High-Speed Downlink Packet Access
IDAMN	Intrusion Detection Architecture for Mobile Networks
IDS	Intrusion Detection System
IMSI	International Mobile Subscriber Identity
ISP	Internet Service Provider
KBTA	Knowledge-based Temporal Abstraction
NFC	Near Field Communication
NICA	Non-Intrusive and Continuous Authentication
NMT	Nordic Mobile Telephones
PIN	Personal Identification Number
RBF	Radial Basis Function
SIM	Subscriber Identity Module
SMS	Short Messaging Service
SOM	Self Organizing Map
SVM	Support Vector Machine
TACS	Total Access Communication Systems
TAS	Transparent Authentication System

Chapter 1

Introduction

Smartphones have evolved from simple mobile devices into complex yet compact minicomputers which can connect to a wide assortment of communication networks to service its users, such as: voice calling and messaging through cellular network, video conferencing through 3G, 4G and Wi-Fi, Door-to-Door navigation by Global Positioning System (GPS), multimedia sharing through Bluetooth, mobile payments by using Near Field Communication (NFC), data synchronisation with personal computer, high end gaming.

While this plethora of features intends to provide convenience to users of mobile devices, there are also threats which can make their life less comfortable. Some of these inconveniences include, but are not limited to, loss or theft of the device, service fraud, mobile malware, information disclosure, Denial-of-Service (DOS) attacks, Smishing and Vishing. In 2004, the first articles about smartphone malware were published stating that the mobile devices were the next generation of targets (Dagon et al., 2004; Piercy, 2004). Statistics show that more than 1000 smartphone malware variants, such as worms, Trojan horses, other viruses and spyware have been unleashed against the mobile devices in the world since 2004. In the last year alone 143,211 new modifications of mobile malware were detected (Chebyshev and Unuchek, 2014).

A number of security techniques have been developed to combat these threats, both on the mobile device and the service provider's network. Most commonly used host-based mobile security solutions include personal identification number (PIN) number/pattern based authentication, mobile anti virus for malware detection and firewalls to block unwanted network traffic. Although PIN number and pattern based authentication is widely used on today's smartphones, many users don't employ them properly which limits their usefulness (Clarke and Furnell, 2005; Kurkovsky and Syta, 2010). Unwanted network traffic is generally blocked by firewalls while signature-based antivirus software solutions use a signature of known malware for their detection. Obtaining latest virus signatures and network traffic rules are not easy tasks. As a consequence, mobile antivirus software leave smartphone users exposed to new malware until the signature is available and the security solution provider releases a patch (zero day attacks).

Bulygin researched propagation of MMS and Bluetooth worms and demonstrated that in the worst case a MMS worm targeting phone book numbers can infect more than 0.7 million devices in about three hours (Bulygin, 2007). Furthermore, Oberheide et al. demonstrated that the average time required for a signature-based anti-virus engine to become capable of detecting new threats is 48 days. In some cases, malware instances target a specific and relatively small number of mobile devices (e.g. for extracting confidential corporate information or track owners location) and will therefore take longer to be discovered (Oberheide et al., 2008). The growth rate for the virus signature list for mobile viruses in last two years is equivalent to the growth rate of the virus signature list for personal computers in twenty years. Additionally, current mobile devices are unable to support the existing anti-virus technologies available for personal computers because of limited processing power, storage space, battery life and memory (Alexander Gostev, 2006b; Denis Maslennikov, 2011)

With the rising threat of smartphone malware, both academic community and commercial anti-virus companies proposed many methodologies and products to defend against smartphone malware. Since the modern malware also rely on mutation and very sophisticated cryptography techniques to evade detection by anti-virus or malware detection systems. Research work carried out to evaluate the effectiveness of these defence mechanisms against existing and unknown malware and their findings show that in some instances Android anti-malware systems had average accuracy of 50.95%. They also showed that almost all anti-malware products are suspectable to common evasion techniques (Rastogi et al., 2013; Zheng et al., 2013)

So as it stands, these techniques have serious shortcomings that make them inefficient for mobile devices. This leads us to explore more sophisticated systems such as Intrusion Detected System (IDS). Historically, service providers have implemented such systems on the network-side to monitor mobile user's calling and migration activities to detect telephony service fraud. Hilas et al. used Artificial Neural Network (ANN) in order to detect anomalous behaviour indicating a fraudulent use of the operator services (Hilas and Mastorocostas, 2008; Shabtai et al., 2012).

More recent research has focused on using anomaly detection techniques on mobile devices for malware detection. These studies have utilised classification of normal and malicious applications samples based on their signatures, permissions or system calls at the kernel level (requiring customised version of OS to be installed on the devices) (Dini et al., 2012). Other researchers in the field of anomaly detection have also utilised Machine Learning e.g. Clustering or Support Vector Machine (SVM) or Artificial Neural Networks e.g. Self Organising Map (SOM) to build behaviour based mobile malware detection systems. Mostly, these studies only relied on generating malware behaviour signature based on CPU usage, memory load and system calls and contextual information to high accuracy results (Burguera et al., 2011; Schmidt et al., 2007; Zhao et al., 2011).

Since today's smartphones have the ability to access multiple communication channels and accommodate a wide variety of services, which makes mobile devices hosts of sensitive information. Now user identity verification on mobile device has also gained paramount importance. Existing network-based security mechanisms are unable to offer a comprehensive protection for mobile devices. Host based security solutions for mobile devices can only detect anomalies arising from mobile malware through more application of sophisticated behaviour profiling techniques with high accuracy. Nevertheless, these solutions don't have the ability to detect anomaly arising from device misuse by unauthorised users.

We explored different methods of Artificial Intelligence for generating profiles of smartphone users. Based on our exploration we concluded that, a *Rule Based* expert system, can't be used in our experiment because they require both declarative and procedural knowledge to emulate reasoning process of human experts in a particular domain (McGraw and Harbison-Briggs, 1989). The expert systems are composed three basis entities: the knowledge base, an inference engine and a user interface. The knowledge base contains rules expressing the heuristics for the domain. The inference engine is made up of the rules that are used to control how the rules in the knowledge base are used or processed and the user interface allows communication between the expert system and the end user. So, a rule based system would require eliciting the relevant domain knowledge from experts, a process that is often hard and time consuming (Amershi and Conati, 2007). The knowledge acquired from experts can only recognise and interpret generalised expected behaviour from smartphone users and lacks the ability to handle unanticipated behaviour.

A *Neural Network* can be used for classification problems given that the correct class of every observation in the input data (training set) is known. *Cluster analysis* is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other than to those in other groups (clusters).

It is a main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition and information retrieval when the class/group of data is not known. Based on these facts, we decided to use clustering to determine patterns of smartphone usage dataset. K-Means clustering, which is a very simple machine learning algorithm was used to validate our framework in the tests we conducted in a simulated environment. Both, machine learning and behaviour profiling are a not new concepts, however their application for profiling mobile device users' behaviour is novel.

1.1 Research aim and objectives

The aim of this research is to investigate the application of machine learning to model smartphone users' behaviour for anomaly detection that stems from misuse. The main idea behind user profiling is that past behaviour of a user can be accumulated in order to construct a profile of what might be the expected values of the user's behaviour. Future behaviour of the user can then be compared with his profile in order to examine the consistency with it (normal behaviour) or any deviation from his profile, which may imply fraudulent activity. It is envisaged that this system not only has the ability to detect mobile device misuse but it can also be extended to provide transparent and continuous user authentication.

The overall aim of this research is reached through the following objectives:

1. To investigate the characteristics and requirements of a behavioural profiling approach for smartphone security.

2. To compose a comprehensive review of anomaly detection and user authentication methods and examine the pertinency of a behaviour profiling techniques on smartphones.

3. To conduct an experiment for exploring the feasibility of using behaviour profiling approaches on smartphones.

4. To implement a behavioural profiling based security system on smartphones.

5. To evaluate the effectiveness of the behaviour profiling system through a series of tests.

1.2 Methodology and working assumptions

In 2004, researchers from MIT created a mobile device usage dataset from 100 users of Nokia 6600 mobile phone (Nathan Eagle, 2006). This dataset only includes call logs, Bluetooth devices in proximity, cell tower IDs, application usage, and phone status (such as charging and idle). Since this dataset doesn't include features like internet usage, user mobility, user installed applications etc, it doesn't complectly cover all aspect of today's smartphone usage. Meanwhile, researchers at Cambridge University developed a more modern smartphone usage dataset but it was not publicly available until the end of our research (Wagner et al., 2014). So we developed our own data collection application for Google Android platform, which our participants installed on their smartphones. Due to security concerns from the participants our dataset was limited to 16 features only. These included telephony, internet, application usage and other contextual information.

The data collection happened over a period of two weeks and automatically transferred to a server, where we used K-Means clustering algorithm to create a usage profile for each participant. To test profiling accuracy i.e dissimilarity of smartphone usage among the participants we used simulation using Matlab.

1.3 Thesis structure

The rest of the document is organised as follows. Chapter 2 presents a discussion of previous research in the field of smartphone security solutions. Chapter 3, extends this discussion to the use of biometric and behaviour profiling technique for user authentication on smartphones. The rationale for choosing our behaviour profiling system and its methodology is presented in Chapter 4, followed by an architecture of our behaviour based user profiling system in Chapter 5. The evaluation procedure and the discussion of results are presented in Chapter 6. Subsequent chapter concludes the thesis with discussion on the fulfilment of the aims and objectives of the research, limitations and potential directions for future work.

Chapter 2

Literature review

2.1 Introduction

Smartphones have evolved from simple mobile devices designed to provide telephony services via a cellular network into complex yet compact minicomputers which can connect to a wide assortment of communication networks to service its users, such as: voice calling and messaging through cellular network, video inferencing through mobile cellular networks and Wi-Fi, Door-to-Door navigation by GPS, multimedia sharing through Bluetooth, mobile payments by using NFC, data synchronisation with personal computer, high end gaming.

On the flip side of this evolution, since the mobile devices can host various services and store sensitive information at the same time, this brings a number of security threats to the mobile environment, such as service fraud, DOS attacks, malware and information disclosure (Muir, 2003; Stajano and Anderson, 1999) This chapter will present a comprehensive overview to understand how mobile has evolved based on evolution of all three streams of mobile telecommunication technologies i.e. networks, devices and applications. Then mobile security threats will be investigated followed by mobile security strategies that have been put in place to combat these threats.

2.2 Evolution of the mobile communication

The last few years have witnessed a phenomenal growth in the wireless industry, both in terms of mobile technology and its subscribers. There has been a clear shift from fixed to mobile cellular telephony, especially since the turn of the century. By the end of 2010, there were over four times more mobile cellular subscriptions than fixed telephone lines. Both the mobile network operators and vendors have felt the importance of efficient networks with equally efficient design. This resulted in Network Planning and optimisation related services coming in to sharp focus (Sanou, 2012). The evolution of mobile cellular services over the last 30 years is illustrated in Table 2.1. In

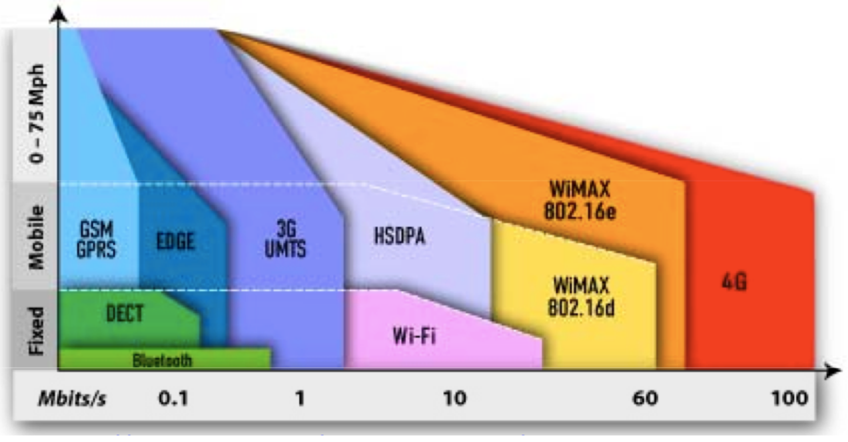


Fig. 2.1 Data rates associated with wireless technologies (Neel, 2010)

addition to these cellular communication technologies, mobile devices are also capable of communicating with a number of other technologies, namely Wi-fi, Bluetooth, NFC, ZigBee and USB. The cellular communication technologies predominantly used nowadays provide a safe and untethered operating environment, nevertheless their effective data throughput is much limited. Conversely, other communication technologies provide more reliable and higher data rate but it happens at the expense of reduced mobility. Figure 2.1 illustrates the relationship between the data rates and mobility of various mobile communication technologies that exist to date.

Table 2.1 Evolution of mobile communication networks

Technology	**1G**	**2G**	**2.5G**	**3G**	**3.5G**	**4G**
Time Period	1970 to 1980	1990 to 2000	2001 to 2004	2004 to 2005	2006 to 2010	since 2010
Core Network	PSTN	PSTN	PSTN, Packet network	Packet network	Packet network	Internet
Bandwidth	14.4 kbps (peak)	9.6/14.4 kbps	384 kbps	3.1 Mbps (peak)	14.4 Mbps (peak)	100-300 Mbps (peak)
Standards	AMPS, NMT & TACS	TDMA & CDMA	GPRS	CDMA 2000, UMTS & EDGE	HSPA	LTE, WiMAX & Wi-Fi
Multiplexing	FDMA	TDMA & CDMA	TDMA, CDMA	CDMA, WCDMA	CDMA	OFDM
Service	Analogue data	Digital narrow-band circuit data	Packet data	Digital broadband packet data	Packet data	ALL IP
Features	Voice only	Voice & data	Internet & Multimedia streaming	Multimedia services with streaming. Universal access & portability	Higher throughput	Very high data rates, HD streaming & increased portability

2.3 Mobile devices

Motorola DynaTAC 8000X was the first commercially available handheld cellphone that was released three decades ago. It took many years to build a network and keep production costs under control in order to make it a viable product. With a price tag of $3,995 it was considered a gimmick and a rich man's toy. Measuring 13 x 1.75 x 3.5 inches and weighing 28 ounces, the 8000X was so big and heavy, even its creators had nicknamed it *"The Brick"* that you could only use for a half an hour before the battery gave out. At the time many people didn't know this technology would change the world. Three decades later there is no doubt of mobile's impact on society. (Wolpin, 2014)

Along with the rapid development of mobile communication technology, the mobile device has also experienced a dramatic evolution. Traditionally, people could only use the handset to make voice calls. Currently, the mobile has become a multimedia and multi-network computing device. The primary driver of device technology evolution has been the mobile operating system starting from the Blackberry OS to Android, Windows or iOS. Secondly, it was input device technology which revolutionised the user interface and made it extremely user friendly. It has moved from large keypads to touch screens. Even the touch screen has evolved significantly as it provides more features like multi-touch gestures. In the future, we may see haptic gestures even captured by sensors on mobile devices.

Below is the summary of some important facts regarding the mobile phones usage (Calloway, 2012; Cisco, 2014; PewResearch, 2013):

- Mobile phones have become such an integral part of our daily life that the mobile adoption is growing 8 times faster than web adoption did in the 1990s and early 2000s. Today's mobile devices rank similarly to a computer in terms of networking, processing power and data capacity. According to latest research, there are more than 6 billion mobile subscribers that equates to more than 86% of the world's population. This is just the beginning of the evolution of mobile devices. It is estimated that there will be 8.6 billion mobile devices by 2017. And by 2018, there will be more than 10 billion mobile-connected devices, including M2M modules exceeding the world's population which is expected to be around 7.6 billion by that time.

- There are total 1.2 mobile internet users which account for 8.49% of the global website hits. In US, 25% of internet users are mobile only.
- 8 trillion text messages were sent in 2011.
- Mobile users spent $3.3 billion through mobile advertisement in 2011 and is predicted to be $20.6 billion in 2015. Google alone accumulates $2.5 billion in annual revenue for mobile advertising.
- More than 300,000 mobile applications have been released in past 3 years with 10.9 billion downloads.

Table 2.2 Applications with their risk scores

Application category	**Application value**	**Threat level**	**Risk temp**	**Vulnerability level**	**Risk level**
Corporate email	8	4	8	2	8
E-banking	7	5	8	1	7
Remote Access	7	5	8	1	7
Voice communication	6	3	6	1	5
Business documents	6	3	6	3	7
Social networking	4	3	4	1	3
Messaging	3	3	3	2	4
Maps & navigation	2	1	1	3	2
Web browser	2	4	3	3	4

There are several criteria which effect the mobile application's risk impact on the mobile system security. These include their connection types, the amount and nature of information they associate with, their threat level.

Table 2.2 shows the risk level of associated with their application categories based on the work by researchers from Portsmouth University (Clarke et al., 2011). The impact on mobile device's security increase with applications risk level. Based on the risk level of the individual application different security controls can be applied. In general, an application with higher risk score requires more stringent security controls.

2.4 Taxonomy of mobile security risks

Mobile devices are becoming malware targets mainly because of vulnerabilities identified by various researchers (Dagon et al., 2004; Leavitt, 2005; Mikko Hypponen, 2006; Nixon, 2011; Piercy, 2004). Commercial Smartphone anti-virus software is generally adopted from traditional personal computing (PC) environment and mostly relies on signature-based techniques. According to (Bulygin, 2007) and (Oberheide et al., 2008) these are not suitable and a more comprehensive security solution tailored for Smartphones is needed. In addition, mobile devices are much more connected to the outside world than PC's. Security researchers' attack simulations have shown

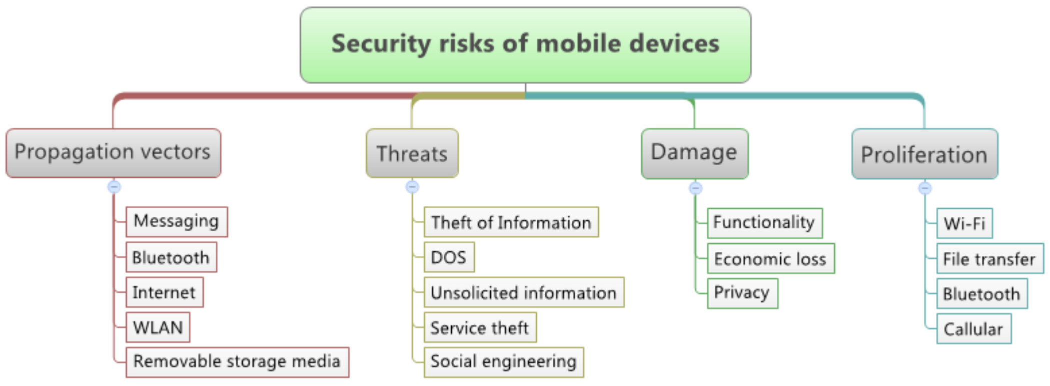


Fig. 2.2 List of security risks for smartphones

that hackers could infect mobile phones with malicious software that deletes personal data or runs up a victim's phone bill by making toll calls or steal financial data. The attacks could also degrade or overload mobile networks. As a glimpse of what future holds for modern Smartphone malware arena, a teenage Dutch hacker used a SSH vulnerability to upload ransom ware application to a number of unsuspecting iPhone users demanding $5 ransom payment (Dancho Danchev, 2009). This was followed by iKee.A worm and iKee.B botnet for Apple iPhone platform within weeks of first ransom ware (Porras et al., 2010). Mobile phones are prone to attacks because they offer internet connectivity, operate like minicomputers, and can download applications or files, some of which could carry malicious code. Therefore, it is vital to distinguish security risks of mobile phones. Figure 2.2 represents a list of risks to mobile phone security.

2.5 Propagation vectors

In order to identify and prevent mobile phone malware, propagation vectors need to be identified. Some of the possible infection routes utilised by the mobile malware are discussed below.

2.5.1 Social engineering

Cyber criminals are now taking over mobile devices by using many of the psychological tricks used to con people online. For several years, social engineers have been trying to fool unsuspecting users into clicking on malicious links and giving up sensitive information by pretending to be old friends or trusted authorities on email and social networks.

And now that mobile devices have taken over our lives, social engineering is an attack method of choice to gain access to a person's smartphone or tablet. Current social engineering tricks used by criminals to get inside your mobile device include the following:

Malicious apps that look like legitimate apps are very hard to distinguish from legitimate apps, when user downloads from non reputable sources. What users may end up with is an application with unwanted things behind it (Dunn, 2011). In 2011, Google removed over 50 malicious apps from Android Market that seemed turned out to be variants of the DroidDream trojan, but looked like legitimate applications and had names like Super Guitar Solo (Brener, 2011). Security experts advise not to rely on application's ratings because many users might be enjoying the app's features without realising that it contains a malicious functionality.

Malicious mobile apps that come from advertisements are also gaining popularity among malware authors. A legitimate application on a smartphone may run a bad advertisement and if the user clicked on the advertisement, they are taken to a web site that tricks the victim into thinking their battery is inefficient, The person is then asked to install an application to optimise the battery consumption, which is instead a malicious application.

Fake security application is another new mobile attack vector used by malware that actually originates with an infected PC. When a user visits a banking site from an infected computer, they are prompted to download an authentication or security component onto their mobile device in order to complete the login process. Most the banks now use two-factor authentication to secure online transactions. In many cases that second factor is implemented as a one-time password sent to the user's phone by the banking provider.

To get access to the phone, once the PC is infected, the person logs onto their bank account and is told to download an application onto their phone in order to receive security messages, such as login credentials. But it is actually a malicious application from the same entity that is controlling the user's PC. Now they have access to not only the user's regular banking logon credentials, but also the second authentication factor sent to the victim via SMS. (Dunn, 2011; Zester, 2011)

2.5.2 Bluetooth

The viruses written so far haven't used any actual vulnerability in Bluetooth but it allows malware to spread among vulnerable phones by mere proximity, almost like the influenza virus in humans. A Bluetooth-equipped Smartphone can identify and exchange files with other Bluetooth devices from a distance of up to 30 meters. As victim travels, their phones can leave a trail of infected mobile devices. And any event that attracts a large crowd presents a perfect breeding ground for Bluetooth viruses. Cabir was the first proof-of-concept network worm which transfers itself as a SIS file disguised as a "Caribe Security Manager" utility which propagates via Bluetooth. It continuously scans for Bluetooth-enabled devices and hence critically reduces the battery life and Bluetooth performance (Alexander Gostev, 2006a; Dagon et al., 2004).

It is a general perception that Wi-Fi and Bluetooth are short-range communication standards and considered as a minimal threat because users would have to be physically near a malicious party to be attacked. However, a team at Flexilis was able to establish a Bluetooth connection with a standard mobile phone more than one mile away with a 19dbi panel antenna (Ho and Heng, 2009).

2.5.3 Messaging

Since the early days, mobile malware have employed Short Message Service (SMS) and Multimedia Messaging Service (MMS) as widely used propagation vector. Commwarrior.A worm spreads via MMS by attaching an infected SIS (Symbian OS distribution) file to MMS message sent to all contacts in the phone's address book. Another worm called Mabir spreads via SMS and MMS, reads the phone address book, monitors received messages and sends fake replies which include a copy of the malware (Leavitt, 2005; Reinhardt A. Botha, 2009). To mitigate such malware, smartphone users should not open URL links or attachments like images from unknown senders to reduce the chances of being a victim.

2.5.4 Internet

Smartphones have now transformed into always connected devices. They offer services like email, instant messaging and social networking. These have become some of the essential features of Smartphones, only made possible with adoption of technologies like 3.5G, High-Speed Downlink Packet Access (HSDPA) and Wi-Fi hotspots etc. Mobile internet has been used successfully by recent malware for iPhone as propagation medium making it vulnerable to more sinister attacks in future (Dancho Danchev, 2009; Porras et al., 2010). When an infected web page is browsed by using mobile phone's browser, the malicious code hidden in web page may execute causing damage to the mobile phone security. Other risks involved with browsing internet include downloaded illegal applications on the mobile because the community that develops these applications can make infected mobile applications available online to unsuspecting users. Smartphone users should refrain from installing applications from un-trusted sources, keep the system software up to date and take extra care while open links from less reputable websites.

2.5.5 Removable media

Removable media like memory cards can also be used by malware to propagate and cross-platform mobile malware further complicates the issue. The Cardtrp worm infects mobile devices running the Symbian 60 operating system and spreads via Bluetooth and Multimedia Messaging Service (MMS) messages. If the phone has a memory card, Cardtrp drops the Win32 PC virus known as Wukill onto the card. Two proof-of-concept Trojans, Crossover and Redbrowser, further show how widespread attacks could simultaneously hit desktops and mobile devices. Both Trojans can infect certain mobile devices when connected to PC as a removable media (Buennemeyer et al., 2008; Peikari, 2006). SMS, MMS, Bluetooth, and the synchronisation between computers and mobile devices are all examples of potential attack vectors that extend the capabilities of malicious actors. Inherent vulnerabilities exist in modern mobile device operating systems that are similar to those of PC's and may provide additional exploitation opportunities. For example, Apple recently fixed vulnerabilities in a security update for iPhone OS for scenarios where playing a maliciously crafted mp4 audio file, viewing a malicious Tagged Image File Format (TIFF) image, or accessing a malicious File Transfer Protocol (FTP) server could result in arbitrary code execution (Apple, 2012; Ho and Heng, 2009).

2.6 Security threats

To anticipate the type of future threats, it is vital to classify existing threats. The mobile threat detection and prevention systems are still immature. The early mobile network security research was focused on routing issues, and more recently on protocol security. With the emergence of feature packed powerful Smartphones, attackers have shifted their attention to exploiting applications to breach mobile platform security. Hence mobile malware threat analysis has become very significant towards achieving the security goal of providing confidentiality, integrity and availability (Dagon et al., 2004). Some of the prominent threats against the phones themselves as described hereafter.

2.6.1 Theft of information

Hackers often target information residing on the devices and recently the focus has been shifted towards information theft for financial motives. Two categories of information theft attacks exist: transient and static information theft. Transient information includes the phone's location, its power usage, and other data the device does not normally record (Koong et al., 2008). Even without advanced location services, attacks can still locate mobile devices with mobile regions, also when the phones are not in active use. Static information attacks target information stored and sent by the mobile devices over mobile network. These attacks compromise user data such as contact information, phone numbers, and programs stored on smart phones. Bluesnarfing and Bluebugging attacks are examples of such attacks. These attacks were made possible primarily because of wrongly configured Bluetooth settings on mobile devices and not because of vulnerabilities in the Bluetooth stack.

2.6.2 Denial-of-service (DOS)

The DOS attack is a hacking method used to force a mobile device or network to become inaccessible to its users. One common DOS technique involves oversupplying the targeted mobile device or network with outside communications requests so that it cannot process real information traffic, or it is accessed so sluggishly that it might as well have been unavailable. Other types of DOS attacks involve utilising other communication protocols, for example Bluetooth, to drain the battery constantly so that mobile device is rendered useless. According to Krishan Sabnan, the monitoring of mobile networks with terminal battery life and limited bandwidth against DOS attacks is of the utmost importance right now (Dominic Laurie, 2009). Possible attack combinations against mobile data networks includes the following scenarios:

1. Prevent the mobile device from going into sleep mode through the delivery of packets. The attack can involve as little as sending 40 bytes every 10 seconds. This attack wastes radio resources and drains mobile batteries.

2. Create congestion at radio network controllers via the re-establishment of connections after they've been released, which results in problems for actual subscribers to the ISP.

3. Infect devices with malware, which causes connected devices to produce excessive port scanning.

2.6.3 Unsolicited information

This type of attack works in opposite direction to the one described earlier. Instead of stealing information attackers and in some cases legitimate companies target mobile device users with cold calling, advertising, messaging, and other unsolicited information. Spam SMS messages are one of the most common attack scenarios. This attack has become even easier recently, as is the case in UK where companies (e.g. 118118) offers mobile number directory service. This company claimed to have more than 15m phones numbers in June 2009 on its database, collected from free to purchase public domain (Dominic Laurie, 2009). This clearly suggests the ease with which a motivated malicious hacker can use publicly available information to launch such an attack against unsuspecting mobile device users.

2.6.4 Theft-of-service

A mobile device provides many services through a telecommunication service provider network connection, such as voice calling, text messaging and web surfing. In order to utilise these services, a charge needs to be paid to the service providers. However, when a person uses such services without paying a charge, a service fraud occurs.

Some malware might attempt to use the victim's phone resources. Possibilities include placing long-distance calls, sending expensive SMS messages, and so on. The Mosquitos virus is perfect example of such attack (McCue, 2004). Pirated copies of games were infected with a virus that sends expensive SMS messages when users played the game on the mobile device (Dagon et al., 2004; Piercy, 2004).

When a mobile device is stolen, an unauthorised person could access the mobile services within a relatively small time frame until the owner of the device reports the incident to their service provider, who then has the power to terminate the connection. In order to maximise the window of opportunity for abusing the services, criminals could plot more sophisticated attacks which have a smaller footprint for detection, such as a Subscriber Identity Module (SIM) card cloning attack. By exploring the coding flaws within a cellular network authentication process, criminals could clone a victim's SIM card and abuse the services at the victim's expense (Rao et al., 2002). In such a case, even at the end of a billing month, a mobile owner may not notice the abuse has occurred unless a thorough checking of their statements is undertaken. Moreover, criminals could launch a far larger scale of attack against the telecommunication service providers by using the SIM cloning trick. According to the Global Fraud Loss Survey 2009, service fraud is estimated to cost telecommunication service providers $72-80 billion every year around the world (Aaronoff, 2009).

2.7 Damage

Damage caused by mobile malware can occur in different ways. It can take up enough memory space on the device or by maliciously utilising other the resources to deny user access to the device hence jeopardising the device performance. At worst it can

Table 2.3 Damage by Mobile Malware

Damage Type	Examples
Functionality	Battery draining Disables antivirus products Prevents access to messaging services Overwrites normal phone utilities Modifies mobile devices display Lower mobile system performance
Inconveniently	Locks the mobile devices multimedia card
Economic loss	Sends messages to expensive toll numbers Continuously send SMS or MMS Deletes private files
Disturb a mobile network	Denial-of-service attacks Loss of network bandwidth
Privacy	Data theft Loss of confidential information Phone hijacking Data modification

modify user data, delete valuable files and cause unwanted billing. It can cause an end-user a lot of time and money to fix up a problem due to a malware. A summary of various examples of damages by mobile malware are outlined in Table 2.3.

The damage caused by mobile malware depends on the attack vector used. Malware can use messaging services to cause unwanted billing, spread spam. Fraudsters use ransomware that encrypts the data held on the mobile device for financial gain. Due to the sensitive nature of data held on mobile devices, state and enterprise level espionage is also becoming common place for smartphones.

2.8 Spreading mechanism

A mobile malware utilises ever evolving propagation mechanisms. There is a danger of increasing trend towards automation as seen in the case of IKee.A worm and iKee.B botnet (Dancho Danchev, 2009; Porras et al., 2010). Other propagation vectors include sending malicious applications via messages, opening TCP/IP connections directly from the applications and therefore offering greater opportunities for the malware to spread. By analysing different families of epidemic malicious codes found until now, mobile phone malware can be defined as:

> "A piece of data or program that spreads among Smartphones by the communication interfaces and can influence the usage of handset or leak out sensitive data" (Yap and Ewe, 2005).

Mobile malware transmitting via MMS and email can transmit data by GPRS and Wi-Fi. For the mobile phone virus that spread by electronic file, it can transmit data by Bluetooth and IrDA. Although there are four wireless transmission ways, some need relay nodes or directional angle. It is suggested that Bluetooth is the best choice for virus writer (Morales et al., 2006). Modern smartphone operating systems like Apple iOS have built in tightened security features to limit the use of Bluetooth for malicious purposed. A possible subset of spreading mechanisms used by mobile malware, as summarised by Shih et al. (2008) are shown in Table 2.4.

Table 2.4 Malware Spreading Mechanisms

Channel	**Distance (m)**	**Direction**	**Neighbour discovery**	**Relay**
Telephony	1,000	Non-directional	Appointed	Yes
Wi-Fi	100	Non-directional	Appointed	Yes
Bluetooth	10	Non-directional	Automatic	No
Infrared	1	Directional	Automatic	No
File transfer	0	Non-directional	Automatic	No

2.9 Mobile malware evolution

The first mobile virus appeared in 2004 but since the emergence of high end, feature rich smartphones the number of mobile malware targeting these devices has seen tremendous growth. The popularity of smartphones and the increase in the number of new services they offer means a parallel increase in the number of malicious programs used by cyber criminals to make money from mobile device users. It is safe to say that today's cyber criminal is no longer a lone hacker but part of a serious business operation.

There are various types of actors involved in the mobile malware industry: virus writers, testers, interface designers of both the malicious apps and the web pages they are distributed from, owners of the partner programs that spread the malware, and mobile botnet owners. This division of labor among the cyber criminals can also be seen in the behaviour of their Trojans. In 2013, there was evidence of cooperation (most probably on a commercial basis) between different groups of virus writers. It is now clear that a distinct industry has developed and is becoming more focused on extracting profits, which is clearly evident from the functionality of the malware.As of January 1st 2014, Kaspersky Lab detected a total of 143,211 new modifications of malicious programs targeting mobile devices in all of 2013. Furthermore, cyber criminals used 3,905,502 installation packages to distribute mobile malware in the yesteryear (Chebyshev and Unuchek, 2014).

It is an alarming trends and calls for a robust security mechanism for mobile devices. Recent distribution of different mobile device operating systems has seen a tremendous change. The Android operating system is consistently winning over new users, leaving other mobile platforms a long way behind. The Apple iOS has also increased its market presence. Android remains a prime target for malicious attacks. 98.05% of all malware detected in 2013 targeted this platform, confirming both the popularity of this mobile OS and the vulnerability of its architecture (Figure 2.3). Malicious programs and attacks have, in general, become more complex and the awe-inspiring majority of the malicious programs detected in the past year are

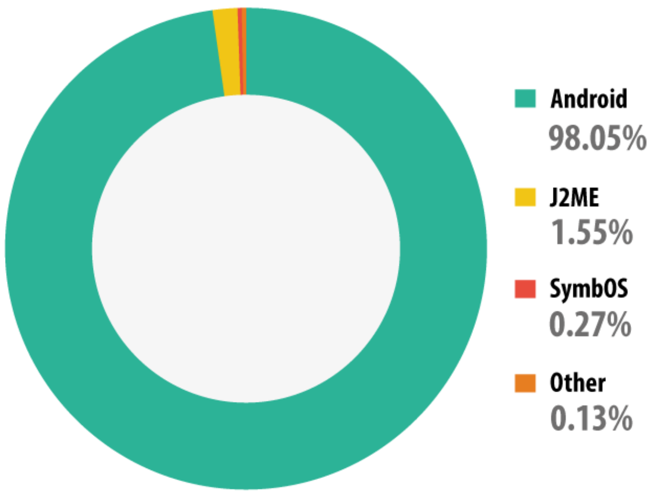


Fig. 2.3 The distribution of mobile malware detected in 2013 by platform (Chebyshev and Unuchek, 2014)

designed to steal money from mobile device users. Table 2.5 shows the distribution of mobile malware released in 2013 based on their categories.

Table 2.5 The distribution of mobile malware by category

Malware category	Distribution
Trojan-SMS	33.5%
Backdoor	20.6%
Trojan	19.4%
Adware	7.1%
Risktool	6.0%
Trojan-Downloader	5.8%
Trojan-Spy	4.0%
Others	3.6%

2.9.1 Future attacks

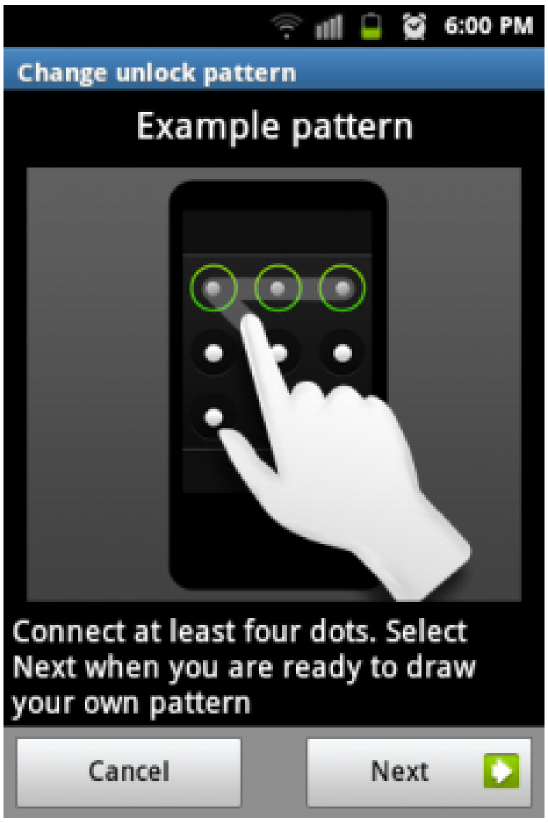
Malicious software that attacks users of mobile banking accounts continues to develop and the number of programs is growing rapidly. It is obvious that this trend will continue, with more mobile banking Trojans and new technologies to avoid detection and removal. Of all the mobile malware samples detected in 2013, bots were the most numerous category. The attackers have clearly seen the benefits of mobile botnet when it comes to making profits. New mechanisms for controlling mobile botnet may appear in the near future. It is expected that in 2014 vulnerabilities of all types will be actively exploited to give malware root access on devices, making removal even more difficult. 2013 saw the first registered malware attack on a PC launched from a mobile device. SMS Trojans are likely to remain among the mobile malware leaders and even conquer new territories. (Chebyshev and Unuchek, 2014)

2.10 Mobile device security controls

In order to counter the highlighted mobile security threats, various security projects have been proposed. For instance, employing an authentication technique to stop unauthorised usage, using mobile antivirus products to detect and remove mobile malware, taking advantage of mobile firewall to filter unwanted traffic, utilising an encryption mechanism to protect the information stored on the devices, and making use of battery based mobile IDS systems to detect malware presence. These security controls will be discussed hereinafter.

2.10.1 Authentication

A PIN is a knowledge based authentication technique. A user is required to enter the correct PIN before accessing a mobile device. Two types of PINs can be deployed: the first is for the mobile device itself and the second is for the SIM card. Normally, a mobile PIN contains between 4 and 8 digits. The PIN is a point-of-entry technique and is therefore only required when a device is initially switched on. Without the correct PIN, the device would not start and the SIM card would not authenticate with the cellular networks. However, most of the time the user will not be required to re-enter the PIN until the next reboot. This provides plenty of opportunities for attackers to abuse a mobile device. Recently, the use of the PIN technique has become more sophisticated; PINs can now be set to be requested again after a certain period of time dependent upon the user's preference. This would significantly reduce the possibility of the device being abused. Nonetheless, in practice, as many mobile users do not employ the technique properly, such as never changing the PIN, sharing it with friends or writing it down on paper, this makes the PIN based authentication technique inadequate as a protection of mobile devices (Clarke and Furnell, 2005; Kurkovsky and Syta, 2010). With increasing hardware availability, a significant portion of mobile



Connect at least four dots. Select Next when you are ready to draw your own pattern

Fig. 2.4 Mobile user authentication based on pattern lock

devices are equipped with new technologies (e.g. touch screens and built-in cameras). This provides opportunities for developing other authentication techniques on the mobile devices, such as the Android password pattern, a type of graphical password. Users are required to draw a shape on the 3 by 3 contact points on a touch screen as their password (as demonstrated in Figure 2.4). As users can only link two adjacent contact points together, this means the points which are not neighbouring with each other (e.g. 1 and 3) can never be used as a combination. As a result, the Android password pattern provides less password combinations than the traditional PIN based password technique can. In consequence, this method is more vulnerable to a brute force attack. As this method is also a knowledge based approach, it suffers several disadvantages as mentioned earlier, such as never being changed or being shared with others. In addition, research showed that this type password can be easily determined when a screen is greasy. (Aviv et al., 2010; Zhang et al., 2012)

2.10.2 Mobile antivirus solutions

Since the discovery of the first mobile phone virus in 2004, the antivirus software industry started to shift its attention from the traditional computing environment towards to the mobile platform. The first mobile antivirus software was developed by F-Secure and became available on the market in August 2005 (BBC, 2005). Since then, other antivirus companies also developed their counterparts. Mobile antivirus software was initially designed to detect and remove malware for the Symbian and Windows CE platforms due to their large market shares in the past. Latest mobile antivirus products also encompass other mobile platforms, such as Android based mobile devices because of its increasing popularity.

Just like traditional antivirus software, mobile antivirus software needs to update its signatures regularly to detect the latest malware. This process requires a dedicated Internet connection which was not available for mobile devices several years ago. However, with the growing availability of 3G and Wi-Fi technologies, it makes the updating process much easier and mobile devices could get the latest signatures promptly.

2.10.3 Mobile firewall products

As mentioned in the last section, the mobile device could be exposed to various network based attacks due to its ability to access multiple wireless networks. In order to protect a mobile device from network based attacks, a mobile firewall software monitors the network traffic continuously and only allows legitimate services (e.g. web browsing) to go through a mobile device. Two types of firewalls can be implemented: on the network or on the mobile device. Nokia Corporation proposed a network based firewall which can be implemented on the mobile service provider's networks. It can be used for blocking malicious data going into a mobile device (Mullins, 2007). As the filtering process is undertaken by the network service providers, there is no overhead for mobile devices. However, it would be impossible for the service provider's firewall to protect mobile devices when they are connected with other networks such as Wi-Fi or Bluetooth. For the host based firewall, several security firms have already released products. As the firewall runs on the mobile device itself, it has the ability to monitor all networks that the mobile handset connects with. Nevertheless, due to the complexity of the multi-network environment, it is difficult to assess how well these host based mobile firewalls perform.

2.10.4 Mobile encryption

Encryption techniques transform information to an unreadable format unless a key for the secured data is provided. As mentioned in earlier sections, mobile devices have the ability to store large amounts of information that could be related to both individuals and corporations. Without encrypting the information, anyone obtaining the device could access the information. In order to protect the information stored on the mobile devices, a number of encryption methods have been proposed. According to the Goode Intelligence mSecurity survey, 40% of organisations are planning to deploy data encryption methods on their mobile devices (Welch, 2012). Nonetheless, encryption methods require a certain level of education for the average mobile device user before they can implement this technique.

2.11 Threats vs security controls

Table 2.6 Mobile security mechanisms vs. Mobile security threats

Threat	**Antivirus**	**Encryption**	**Firewall**	**Authentication**
Mobile service fraud	NO	NO	NO	YES
DOS attack	NO	NO	YES	NO
Malware	YES	NO	NO	NO
Social engineering	NO	NO	NO	NO
Loss/Theft	NO	YES	NO	YES
Inside attack	NO	YES	NO	YES

Although antivirus software can detect the presence of malware, firewall applications can block malicious network traffic, battery based mobile IDS can sense both malware and network related attacks and encryption can translate information to an unreadable format unless a secret key is provided, their abilities of identifying any user related activities are rather limited. For instance, they cannot determine the legitimacy of a user when a voice call is made or a file is accessed. Knowledge based authentication, such as using a PIN, can provide a basic level of protection for the mobile services and information being misused by unauthorised users. For example, without entering a correct PIN, a user will not be able to access any mobile services nor the information on the mobile devices. However, due to a bad practice, such as choosing a simple password, never changing it, writing it on paper or sharing it with co-workers, unauthorised users are able to misuse the mobile device. Moreover, as the PIN is a point-of-entry based authentication method, as long as a correct password is given, a user will be granted access regardless of their true identity. In addition, this raises another issue with the knowledge based authentication approaches. When an authorised user forgets their PIN, they will be denied access despite their legitimacy. Therefore, none of the existing security controls can truly protect the multimedia and multi-networking mobile devices. As a result, a novel security control which can continuously protect both the mobile services and information based upon the users' legitimacy is desperately needed. Table 2.6 illustrates a comparison between existing mobile security mechanisms and mobile security threats

2.12 Malware detection and prevention schemes

The first computer virus infecting mobile devices was identified on June 14, 2004 for Symbian platform while the first malware to infect handhelds running Windows Mobile operating system was released July 17, 2004 (Piercy, 2004). This marked the start of a new era for the virus and antivirus community. The number of anti-malware software available at that time was just a fraction of security solutions available today.

Majority of mobile device users were vulnerable to any possible malware attack. The Security solution providers released antivirus solutions adopted from traditional PC's in an intransigent effort to protect against those threats. Even today, a large proportion of mobile device users don't use antivirus software due to lack of knowledge, limited memory and processing power as compared to PC's (Morales et al., 2006). But even with antivirus software installed not all users can and will keep the signature database up to date which renders the anti-malware system useless for new types of malware that emerge every day.

In order to implement a highly effective security solution tailored to the resource constraint mobile devices, a thorough analysis of existing malware detection and prevention techniques has to be carried out. Once the shortcomings in the existing techniques are identified and rectified, mobile malware will be effectively prevented from causing harm to mobile devices.

Anti-virus technology, a key player in tackling malware these days, is primarily based on two complementary approaches. Signature based methods rely on the identification of unique strings in the binary code. Whenever a new type of malware is unleashed, anti-virus vendors need to catch an instance of the new malware, analyse it, create a new signature and update their clients.

During the period between the appearance of a new (unknown) malware and the update of the signatures of the anti-virus clients, these mobile devices are vulnerable to the new malware. Therefore, while being very precise, this approach is useless against previously unidentified malware (Christodorescu and Jha, 2004; Oberheide et al., 2008). Behaviour based methods involve heuristic based methods, which use rules determined by experts to define a malicious behaviour, or a benign behaviour by scrutinising the execution behaviour of a program in order to enable the detection of unknown malware (Jacob et al., 2008).

A list of different types of malware detection approaches (Shih et al., 2008) is shown in the Figure 2.5 and described hereafter.

2.12.1 Signature based approach

Commercial antivirus scanners look for signatures which are typically a sequence of bytes within the malware code to infer the nature of the scanned program. The signature generation process is either performed by a security expert or done automatically. This technique is the most commonly used in industry.

2.12.1.1 Real-time I/O scanning

In this technique the malware detection engine monitors the input and output streams of the device during the file execution. As with all pattern matching techniques, this

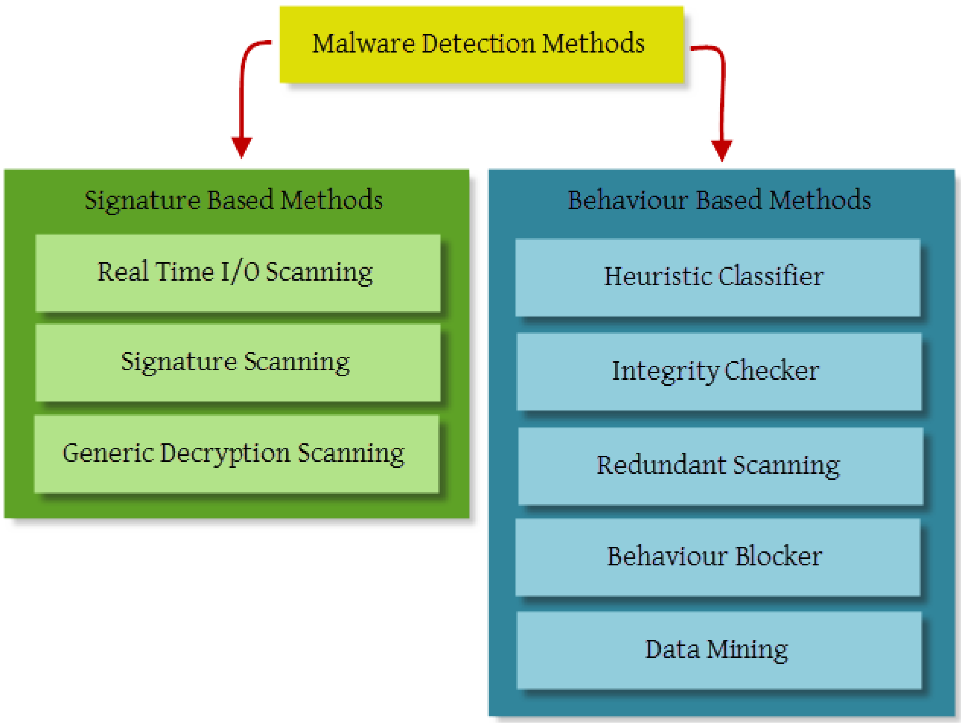


Fig. 2.5 Techniques used for malware detection

approach must also update the virus definition pattern for new malware to be detected, hence the data transfer rate is affected dramatically.

2.12.1.2 Signature scanning

The oldest and most popular technique for malware detection is signature scanning. The main problem with this approach is that it can only identify and remove malware whose signature already exists in the database, hence it is futile against new malware. This is particularly devastating because majority of the users don't keep signatures up-to-date. On the bright side, the major benefit from this technique is that a malicious application can be detected without execution.

2.12.1.3 Generic decryption scanning

A scanner that uses generic decryption relies on this behaviour to detect polymorphic malware. Each time it scans a new program file, it loads this file into a self-contained virtual machine created from RAM. Inside this virtual computer, program files execute as if running on a real computer. A polymorphic virus running inside the virtual machine is isolated and hence causes no damage to the host system. In response to non-virus behaviour, the scanner quickly stops running the file inside the virtual machine, removes the file, and proceeds to scan the next file.

2.12.2 Behaviour based approach

Protection from unknown viruses is the major issue of the day in mobile virology. The anti-virus authors rely heavily on known signatures to detect malicious programs. But all efforts still have not solved the key problems discussed earlier until behaviour-based method appeared. Behaviour based detection differs from the signature based method because it identifies the action performed by the malware rather than the binary pattern in the code. (Rastogi et al., 2013; Zheng et al., 2013) The programs with unrelated

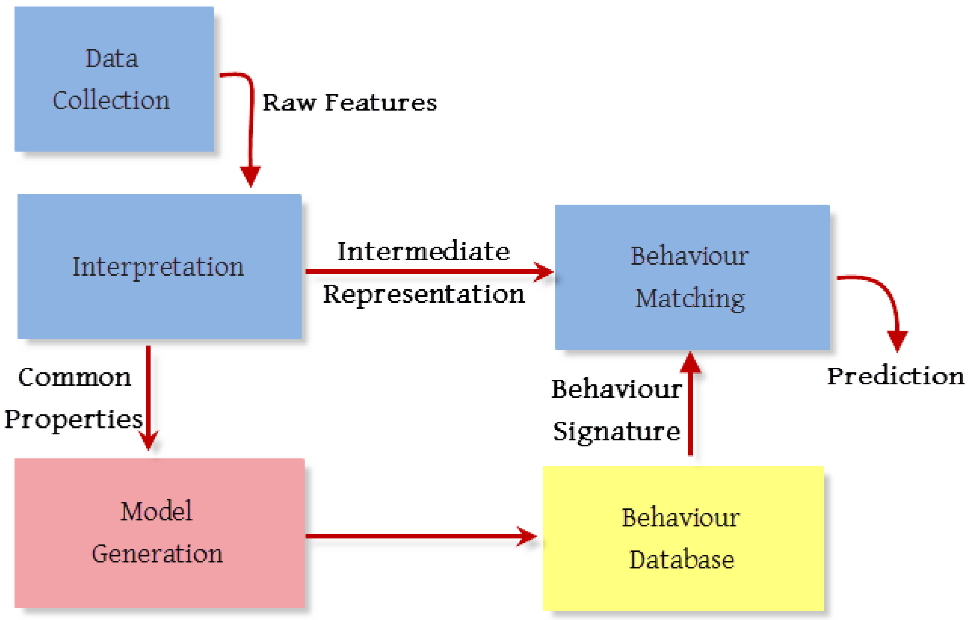


Fig. 2.6 Functional design of generic behaviour detector

syntaxes but having the same behaviour are profiled in a single group, thus this single behaviour pattern can identify various samples of malware. This malware detection is particularly beneficial in detecting the malware which keeps on generating new mutants since they will always use the system resources and services in the similar manner. A generic functional diagram of behaviour based malware detector is shown in the Figure 2.6. The functional components of a behaviour detector are described below:

- **Data Collection:** This component performs raw information collection from different sources to observe the indented actions of the program being scanned. Information is either captured dynamically to capture effectively performed actions or static extraction is utilised to detect all actions. This data can collected from different sources
- **Interpretation:** Behaviour detectors work at a higher interpretation level than simple form-based detection hence, this component is required to analyse and interpret the collected data. This second tasks brings into light the important characteristics of the collected data feeding it to behaviour matching component.
- **Behaviour Matching:** It is used to compare the representation with the behaviour signature based on matching algorithm. According to the result, the program is labelled as malicious or benign.

Some basic types of behaviour based malware detection are discussed below.

2.12.2.1 Heuristic classifier

Heuristic classifiers are generated manually by experts in virology. This type of classifier reconstructs the behaviour of the program under inspection, and uses it as a basis for conclusions about the potential danger from this program. Every type of malware requires a specific heuristic algorithm. This process can be even more costly than generating signatures, so finding an automatic method to generate classifiers has been the subject of research in the anti-virus community. To solve this problem can be solved by utilising high speed time delay neural networks (El-Bakry, 2010).

2.12.2.2 Redundant scanning

Some malware have to use transfer instructions somewhere in the file contents and obtain control when the procedure containing a code transferring control to the malware body is executed rather than started (Shih et al., 2008). Before coding these instructions into a file, the malware author must choose a correct address in this file. Redundant scanning allows for the scanning of not only the system processing entry points, but also the entire contents of any given program.

2.12.2.3 Integrity checker

An integrity checker uses the information of what should be inside a clean file rather than on the knowledge of a malware's appearance for detection of unknown malware (Shih et al., 2008). Integrity checkers can be considered powerful tools not only against known and unknown malware, but also as strong alternative intrusion detection and anti-hacker utilities. However, integrity checkers cannot detect a malware in new files, since there is no information in their databases about these programs.

2.12.2.4 Behaviour blocker

Behaviour blocker monitors any program's activity and prevents harmful actions that could be done by malicious code. In theory, the behaviour blocker may prevent the distribution of any known and/or unknown virus. However, the user must possess sufficient knowledge and experience, otherwise the application will not be able to perform the action required and the malware will penetrate the system. A behaviour blocker can block the function or even terminate the whole application. It is always used in conjunction with other malware scanners to provide a risk score.

2.12.2.5 Data mining

Data mining methods are ideal for this purpose since they detect patterns in large amounts of data and use these patterns to detect future instances in similar data along with detecting known instances. By using data mining, knowledge of known malicious executable can be generalised to detect unknown malicious executable. Matthew et al. use data-mining methods, Ripper and Naive Bayes, to detect malicious binaries. Data mining-based detection is viewed by many as key to the future of virus detection (Shih et al., 2008, 2005).

2.13 Summary

With more than 5 billion users globally, mobile devices have become ubiquitous in our daily life. Mobile devices have the ability to provide various services across multiple communication networks and also on the handsets alone. People utilise them to complete different tasks, that are not only related to basic activities, such as making a phone call or playing games but also, more importantly, may involve sensitive information, such as storing personal data, transferring money over the Internet and accessing corporate emails. As a result, the security requirement should be heightened to ensure the legitimacy of a user throughout the course of usage.

With the increased popularity of mobile devices, the associated threats are also increased from both outside and inside. Several security controls, such as antivirus software, firewall applications and encryption mechanisms, can be used to protect the mobile devices from being harmed by malware, network attacks and information disclosure attacks. However, their ability to detect user related activities (e.g. making phone calls and accessing information) is rather limited. Although the knowledge based authentication technique (e.g. PIN) could provide some level of protection against user misuse, its weaknesses have been well documented by literature.

As demonstrated in this chapter, it is critical to only allow legitimate users to maintain security. Despite the antivirus, firewall and encryption applications that are already commercially available, their impact on monitoring user activity is minimal. The knowledge based authentication technique can provide protection against unauthorised misuse. However, in practice, as the knowledge can be shared and learnt, this renders the method ineffective. As a result, a new security mechanism is needed to continuously authenticate users while they access mobile services and information on smartphones. The feasibility of utilising this method will be discussed in the next chapter.

Chapter 3

Biometrics authentication solutions for smartphones

3.1 Introduction

Research into biometric approaches for authentication is rapidly maturing and commercial biometric development for access control has been growing significantly year on year. With increasing computational power and hardware availability, the chance to adopt biometric based access control approaches on mobile devices is becoming more realistic. Also the modern mobile device provides a variety of network and host based services. It is arguable that people utilise these mobile services differently. For example, when a user accesses their mobile calendar service to find out what their daily schedules are, the features related to this behaviour are the time of access (7:15 AM), the duration of access (1 minute) and the day of access (Monday). However, when an attacker accesses the same service, they are likely to choose a time when the owner does not normally use the device (e.g. 3 AM) and the duration of access might be much longer (e.g. 5 minutes) as they want to explore as much information as possible. As the attacker's activity deviates from the user's normal behaviour profile, a security monitoring system could detect the incident.

3.2 Biometrics based authentication

Today's mobile devices already possess a number of built-in features capable of sensing a variety of user biometric traits, enabling several approaches to be easily deployed upon them.

3.2.1 Face recognition

Most modern mobile devices are equipped with a camera which is designed for taking pictures, shooting videos and making video conference calls. This creates the opportunity to use facial recognition/ear recognition on the mobile device. In addition, by utilising an expensive infra-red camera, iris recognition and retina scanning could also be permitted. Researchers proposed a thin-client based topology for face recognition on mobile devices. A user's facial image was captured using an inbuilt camera and then sent to a network server for further processing (i.e. comparison with the template and reviewing the user's identity). Depending upon individual approaches and the dataset, their system accuracies fell in the range of 79%-95.6%, highlighting the potential of developing facial recognition for mobile devices (Al-Baker et al., 2005; Clarke, 2011; Weinstein et al., 2002).

3.2.2 Fingerprint recognition

Fingerprint recognition is the first biometric approach that has been deployed on mobile devices as an authentication method. In 1998, Siemens and Triodata developed a fingerprint recognition prototype by placing a sensor at the back of a mobile phone (Bromba, 2011). By utilising the prototype, a user can gain instant access to the phone by swiping their finger against the sensor instead of entering the password. Since then, many fingerprint recognition based authentication systems have been developed and implemented on various mobile devices by different manufacturers. Apple iPhone 5s and newer models have a fingerprint recognition system device for locking/unlocking and authenticating users when purchasing item from Apple AppStore (Schneier, 2014). Apple's shiny new iPhone 6 can be spoofed with the same fake fingerprints that tricked its older sibling, the iPhone 5S. That's according to mobile security firm Lookout, which said it discovered that it is possible to create a fake fingerprint that's capable of fooling the TouchID fingerprint sensor of the latest iPhones. Since Apple introduced functionality to make payments directly from the smartphone, the fraudsters have more incentive to abuse access to an iPhone. (Leyden, 2014)

3.2.3 Gait recognition

When a user carries their mobile device in their trouser pocket, their gait information can be collected as they walk. (Derawi et al., 2010) employed a Google G1 mobile phone with an in-built accelerometer to gather a carrier's gait activities. Their experimental result was 20.1% EER when testing 51 volunteers' gait activities. This indicates that gait recognition shows some level of discrimination for mobile device users. However, a substantial improvement is required in this technique before it can be considered for wider deployment.

3.2.4 Handwriting recognition

A significant proportion of mobile devices have been equipped with a touch screen, enabling the handwriting verification technique to be deployed. A user's identity can be verified when they perform their signature (static) or while they write a message by using a stylus (dynamic). (Clarke and Mekala, 2007) proposed a dynamic approach to verify a user when certain words were written. With a 1% EER, their system performance was better compared with other behavioural techniques. Despite their approach not being fully dynamic as the words were pre-chosen, their work demonstrated that it is possible to identify users based upon the way they write on a mobile device.

3.2.5 Voice identification

Traditionally, mobile devices were primarily used for making telephone calls, during which a user's voice sample can be captured for the purpose of voice verification. (Woo et al., 2006) examined the possibility of using static voice verification for the mobile device by using an ASR-dependent speaker verification approach. Despite the comparison process being carried out by a standard computer, their work achieved a 7.8% EER proving that a mobile device user's identity can be verified by their voice, even in a noisy environment (e.g. in a busy office).

3.2.6 Transparent authentication system (TAS)

By utilising various biometric techniques, a TAS authenticates mobile device users in a continuous and transparent manner as shown in Figure 3.1. The TAS chooses individual biometric techniques to verify a mobile user based upon the configuration of their mobile device. For instance, if a mobile device is not equipped with an built-in camera, the TAS will only choose keystroke analysis and voice verification to verify the user. One example for TAS is the Non-Intrusive and Continuous Authentication (NICA) (Furnell et al., 2007). By utilising the combination of various biometric techniques, NICA can achieve an EER less than 0.01%.

Many of the applicable biometric approaches have achieved a good level of performance and some of them have already been utilised by the mobile security industry. In general, based upon the way they provide security, they can be put into two categories:

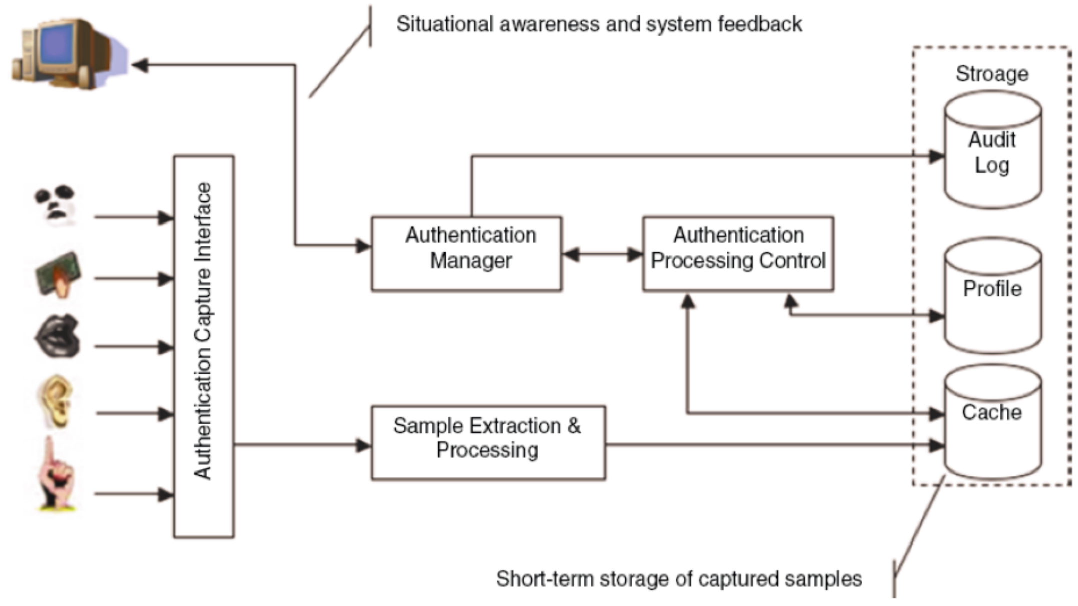

Situational awareness and system feedback
Stroage
Audit Log
Authentication Capture Interface
Authentication Manager
Authentication Processing Control
Profile
Cache
Sample Extraction & Processing
Short-term storage of captured samples

Fig. 3.1 A generic TAS framework (source:(Clarke, 2011))

point-of-entry and transparent based systems (as shown in Table 3.1). For the point-of-entry based techniques, they only verify the users' identity at the beginning of a session. Once the user obtains the initial trust from these techniques, their legitimacy will never be challenged again. In comparison, transparent based approaches work in a similar manner to IDS; they constantly check the users' identity throughout usage, even when a user has successfully passed point-of-entry authentication. The intended

Table 3.1 Biometric authentication approaches on mobile devices

Approach	Point of entry	Transparent	Continuous
Behaviour profiling	NO	YES	YES
Face recognition	YES	YES	NO
Fingerprint recognition	YES	NO	NO
Gait recognition	NO	YES	YES
Keystroke dynamics	YES	YES	YES
Voice recognition	YES	YES	NO

new security system needs to provide continuous protection throughout a usage session. As a result, the point-of-entry based biometric techniques can't be employed by the new security mechanism. For transparent techniques there are behaviour profiling, face recognition, gait recognition, keystroke analysis and voice verification; all of which can be implemented on mobile devices to provide continuous and transparent protection. Face recognition can easily verify users when they make a video call or partake in a video conference. However, on other occasions, the verification process cannot be easily performed because of the difficulty of capturing high quality facial images. Gait recognition authenticates users when they walk from one location to another but it is impossible to perform when they are stationary, such as sitting in front of an office desk or lying in bed. Keystroke analysis and handwriting recognition examine users when a message is composed or a document is modified. However, little protection can be provided if a user views a web page or reads a document. For voice verification, a user can only be authenticated during a conversation. Therefore, with all these activity based techniques a user's continuous protection will become exhausted when a particular activity does not occur.

3.3 Behaviour profiling based authentication

In order to protect unauthorised access, an authentication system is required. Biometric authentication has proven to be a more reliable solution than knowledge based and token based techniques. Indeed, biometric techniques are uniquely individual, impossible to forget or lose, difficult to reproduce or falsify and difficult to change or hide. Despite the well known advantages of biometric authentication approaches, the majority of current state-of-the-art mobile devices embrace point of entry authentication systems including PIN/passwords and one time fingerprint verification that has proven to be inadequate (Aviv et al., 2010; Zhang et al., 2012). Although research conducted into the behaviour profiling technique is limited within the mobile host environment, significant amounts of research suggest that smartphone users can be discriminated between each other via their calling behaviour and migration behaviour through service providers' networks. The research into mobile behaviour profiling started around 1995, focusing mainly upon the area of IDS to detect telephony service fraud. Three mobile user activities have been studied so far: calling activity, migration mobility activity and migration itinerary activity (both the mobility activity and itinerary activity are user's location activities and are defined in the following subsections). To date, all behaviour based mobile IDS systems are network based, as users' behaviours are obtained and monitored by network services providers.

3.4 Telephony service based profiling

The telephony based mobile IDS monitors a user's calling attributes (e.g. IMSI, start date of call, start time of call, duration of call, dialled telephone number and National or International call) to detect service fraud, SIM card cloning and the loss or theft of devices (Moreau et al., 1997). By collecting these features over a period of time, a historical calling usage profile can be acquired. If the deviation between a current calling session and the historical profile exceeds a predefined threshold, an intrusion is identified. There are a number of telephony based mobile IDS systems and they will be introduced in the following section.

The earliest work on telephony service based detection systems was the European Advanced Security for Personal Communication Technologies (ASPECT) project on mobile security (Gosset, 1998). The ASPECT project covered different security aspects within mobile communications such as authentication, encryption and service fraud detection. A number of techniques were used to study a user's calling behaviour, such as supervised neural network and unsupervised neural network (Lerouge et al., 1999). The evaluation process was carried out by several experimental studies on a dataset which was collected from the Vodafone network. The dataset contains a total of 317 fraudulent and 20,212 legitimate users. Their experimental results were a detection rate of 50% and a FAR of 0.02%.

(Samfat and Molva, 1997) proposed an Intrusion Detection Architecture for Mobile Networks (IDAMN) which provided multilevel protections for mobile GSM networks. The multilevel protection is applied to each user across three levels: velocity and clone verification, component wise verification and intrusion detection. The architecture monitors the user's behaviour in terms of both telephony and migration activities. A mobile user's telephony activity is divided into two statistical vectors, call vector and session vector. The call vector is a local parameter that measures all outgoing calls and the session vector is a global parameter that measures the following activities; total number of calls, total duration of calls, network connection duration and total number of handovers within a period of time. By using this information a profile and a threshold can be obtained. If a current vector value is bigger than the threshold an alert is raised to a network administrator. The IDAMN was evaluated by using a simulation method. Four types of users were simulated: domestic, business, corporate and roamer. In general, call vectors had a lower false alarm rate of 1% compared with a 2% false alarm rate for session vectors. However session vectors had a better detection rate of 90% compared with the 70% detection for call vectors. The session based approach had a better detection rate because over a long period of time, an intruder's activity deviates more than a legitimate user's behaviour. Among these four categories, business users had the best performance with an average detection rate of 89% and a false alarm rate of 1.5%.

(Boukerche and Notare, 2002) described a service fraud detection model on the mobile phone system using a Radial Basis Function (RBF) neural network model. By using user calling features, the model can detect possible call service fraud. In addition, the model employed a smart procedure: once a possible fraud is identified, an alert will be sent to both the individual user and network administrators immediately, rather than at the end of a billing month. The experiment dataset was collected by an unnamed telecommunication service provider and contains 4,255,973 telephone calls. A variety of experiment configurations have been tested with combinations of call features and RBF neural network setups. Their lowest system error rate was 4.2% and was achieved using 110 neurones in the hidden layer of the RBF neural network.

3.5 Mobile user location based profiling

A user's migration activity is the way that they travel from one location to another. There are two types of migration activities: mobility (change of location) and itinerary (route taken between two locations). By calculating the possibility of a mobile user travelling from one mobile cell to another, migration mobility based mobile IDS systems could detect telephony service attacks. If the result exceeds a predefined threshold, a possible intrusion is detected. Whilst similar to migration mobility based mobile IDS, migration itinerary based mobile IDS systems also monitor cells to detect telephony service attacks. Instead of monitoring one cell at a time, migration itinerary based mobile IDS systems monitor all cells a user covers during their journey from one location to another. It is believed that people always have the destination in their mind when they travel and so certain routes will be chosen as regular or favourite routes. As a result, the probability of mobile users travelling over these routes is much higher than them travelling through other routes. To extend this, when an attacker carries other people's mobile devices, the route they take is likely to be different to the owners' route.

3.5.1 Migration mobility

The IDAMN (Samfat and Molva, 1997) also monitors the user's migration pattern to detect intrusions for the telephony service. A user's migration mobility activity can be obtained when mobile users traverse from one cell to another. Using this information, their mobility profile can be generated and is maintained when the location updating procedure of the mobile device occurs. By using a mathematical formula, a predefined threshold is derived using historical mobility information. The architecture is evaluated via a simulation method for four categories of users: domestic, business, corporate and roamer. The results show that domestic users achieve the best system performance with a false alarm rate of 2% and a detection rate of 90%. In comparison, due to frequent travel plans, the roamers have the worst system performance with a false alarm rate of 7% and a detection rate of 65%.

(Buschkes et al., 1998) propose an anomaly detection method for GSM networks by using the Bayes decision rule. The Bayes decision rule is a mathematical model which calculates the probability of a current activity occurring based on prior activity. Another method is outlined in their work to aid comparison with the Bayes decision rule; the average algorithm. The average algorithm calculates the Mean Residence Times that a mobile carrier stays in one cell. Using the Mean Residence Times and when the mobile carrier enters the cell, a user's mobility profile can be built. Two scenarios are analysed: town and motorway. The results show that: in the town scenario, the prediction rate for the Bayes rule model was more than 80% after 5 days and it reaches 83% after 15 days monitoring; for the motorway scenario, the prediction rate of the proposed model reached 94% after 5 days and was more than 95% after 15 days.

(Sun et al., 2004) proposed a mobility-based anomaly detection model for cellular mobile networks by utilising the combinations of the following techniques: high order Markov model, Ziv-Lempel data compression algorithm and Exponentially Weighted Moving Model (EWMA). The high order Markov model was used to calculate a mobile user's mobility probability from one cell to another. In order to store this mobility information for each individual mobile user, the Ziv-Lempel data compression algorithm was employed to create a mobility profile. Also, the EWMA technique was utilised to efficiently update the mobility profile. Therefore, a more accurate user profile can be regularly updated. The system constantly compares a mobile user's current mobility action with their mobility profile to detect possible intrusions. Once the comparison value exceeds the threshold, an alert is then generated to notify a network operator. The researchers created a simulated environment for evaluation that consisted of a cellular network containing 40 cells, each having six neighbours on average and the average distance between two cell towers is 1 mile. The simulation results show: the false alarm rates of the system were around 25% and 5% when a user travels at a speed of 20 miles/hour and 60 miles/hour respectively. Also, when a user's speed is 20 miles/hour, the system detection rate is 80% and this reaches almost 95% when a user travels at 60 miles/hour. At a higher speed, the mobile carrier covers more cells than at a lower speed and so the deviation between an attacker and the legitimate user will get accordingly larger.

Two major questions are raised regarding their system performance: what are the detection and false alarm rates the system experiences when the mobile carrier's speed is less than 20 miles/hour? What kinds of people are suitable for the system to monitor? For the first question, their results indicate that both the detection rate and false alarm rate were poor if the speed was less than 20 miles/hour. For the second question, the majority of pedestrians, if not all of them, experience an average speed of less than 2 miles/hour which is much less than 20 miles/hour; furthermore, when people drive in the city, the speed is limited to 30 miles/hour for most areas, with traffic lights and safety islands, the average driving speed would be around 20 miles/hour, which is the worst case in the simulation result.

By aiming to improve the performance of the system, several modifications have been made to the existing model, such as adding Shannon's entropy theory to adjust the system threshold in the followup research (Sun et al., 2006). In the new system not only had a constantly updated profile but also a frequently modified threshold. Furthermore, two profiles were created for each mobile user: weekdays and weekends. The researchers evaluated the system using a simulation method. Their experiment results show that the average false positive rate for the adaptive mechanism is reduced by around 7% compared with the non-adaptive method. In general, the adaptive mechanism had a slightly better average detection rate than the non adaptive mechanism did. However, when a mobile carrier's speed was at 20 miles/hour, the adaptive mechanism had a much higher detection rate with around 7% improvement over the non-adaptive mechanism. The advantages of the modified system are: a better detection rate and a lower false alarm rate due to adapted profile and threshold settings. On the other hand, the improved Intrusion Detection system still has a similar problem as the previous version: both the detection rate and false positive rate are uncertain when a mobile user traverses at a speed less than 20 miles/hour.

3.5.2 Migration route

(Hall et al., 2005) propose a method using public transportation users' mobility profiles to detect intrusions by employing an instance based learning pattern classification technique. By collecting a user's location information from location broadcast of their mobile device, a high level mapping of their location profile can be generated. The comparison process is carried out between a user's current mobile sequence with their profile; if the deviation is larger than the threshold an alert will be raised. The proposed method was evaluated by the researchers using a simulation method with 50 mobile users who took the public transport service in Los Angeles. All 50 mobile users' location information was inserted into a MySQL database and the simulation process was conducted using the Matlab software package. However, the simulation result was not particularly promising. In addition, according to previous studies, around 50% of all mobile users take the public transport system, i.e. buses or trains and so the other

50% of mobile device users who do not use the public transport system cannot be protected by this proposed method. Wu and Dai (2004)

3.6 Application usage based profiling

According to research conducted at Plymouth University smartphone security countermeasures such as anti-virus and firewalls that have been deployed in the past decade are inadequate because of the increasing sophistication of the attacks and require additional measures to be taken (Clarke et al., 2010). The researchers proposed a behaviour-based profiling technique, that is able to build upon the weaknesses of current systems by developing a comprehensive multilevel approach to profiling. In support of this model, they conducted a series of experiments to profile calling, device usage and Bluetooth network scanning. Using neural networks, experimental results for the aforementioned activities' were able to achieve an Equal Error Rate(EER) of: 13.5%, 35.1% and 35.7%. This approach provides truly continuous protection to users unless they do not interact with the mobile device at all.

An investigation was conducted to authenticate users based on their writing vocabulary and style of SMS messages. The findings, based upon 30 participants, revealed the feasibility of the approach. While an overall average EER of 24% is unacceptably high, several users experienced an EER of 0%, suggesting significant potential to apply the technique for a subset of the population (Saevanee et al., 2011).

3.7 Comparison of behaviour based profiling

Table 3.2 illustrates the comparison of the monitored features, classification models and detection rate used in the aforementioned behaviour based mobile IDS. By studying a user's calling or location activities behaviour based IDS systems can achieve a high detection rate and offer the ability to detect unforeseen attacks. In addition, as the classification and identification procedures are processed by network service providers, it does not require any additional computational power from the mobile device. This has traditionally been critical for mobile devices as they have limited processing power and space compared with traditional desktop computers. Nonetheless, if these behaviour-based systems work together (as a hybrid system) to monitor the mobile user's action (e.g. calling a friend) while knowing where the action is occurring (e.g. at home), the overall system performance can be increased.

Table 3.2 Comparison of behaviour profiling systems

Name	Behaviour	Classification Model	Detection rate	FAR
Samfat and Molva (1997)	Itinerary	Mathematical formula	82.5%	4%
	Calls	Mathematical formula	80%	3%
Boukerche and Notare (2002)	Calls	RBF neural network model	97.5%	4.2%
Gosset (1998)	Calls	Neural networks	50%	0.02%
Buschkes et al. (1998)	Mobility	Bayes network	87.5%	NA
Sun et al. (2004)	Mobility	Markov model	87.5%	15%
Hall et al. (2005)	Itinerary	Instance based learning	50%	50%
Sun et al. (2006)	Mobility	Markov model	89%	13%

3.8 Summary

The use of behaviour profiling on mobile devices presents an interesting proposal given that every mobile user has to utilise an application/service to perform tasks on their device. As a result, a user's identity could be continuously verified while they are interacting with their mobile devices. Although little literature was available on behaviour profiling on mobile devices, a significant amount of research work on mobile user's calling and migration behaviour has proved that by using a pattern classification method mobile users can be discriminated by the way they utilise telephony services or the way they carry devices around. However, in practice it can be seen that the mobile network operators can only monitor calling and migration behaviour rather than examining every single mobile service. Therefore, none of the current research in mobile behaviour security approaches provides a comprehensive and continuous protection against device misuse. Hence, a mobile behaviour profiling approach which can offer detection across a wider range of services and connections on the mobile device is needed.

Chapter 4

Machine learning for user behaviour profiling

4.1 Introduction

Machine learning is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of self learning computer programs that can update themselves when exposed to new data. For example, a machine learning system could be trained on email messages to learn to distinguish between spam and non-spam messages. After learning, it can then be used to classify new email messages into spam and non-spam folders.

Cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations within the same cluster are similar according to some predesignated criterion or criteria, while observations drawn from different clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some similarity metric and evaluated for example by internal compactness (similarity between members of the same cluster) and separation between different clusters. Other methods are based on estimated density and graph connectivity. Clustering is a method of unsupervised learning, and a common technique for statistical data analysis.

Clustering technique is different from classification which predicts categorical class labels i.e. classifies data (constructs a model) based on a training set and the values (class labels) in a class label attribute and uses this model in classifying new data. Conversely, clustering tries to group a set of objects and find whether there is some relationship between the objects. In the context of machine learning, classification is supervised learning and clustering is unsupervised learning.

Every person has a number of patterns of usage for his/her smartphone that can't be easily defined by human intuition. The K-Means clustering is one of the very well known and easy to implement unsupervised machine learning techniques that is well suited for identifying patterns by grouping similar usage data into clusters.

4.2 Machine learning for behaviour profiling

Machine Learning is a class of algorithms which is data-driven, i.e. it is the data that *tells* what the *good answer* is. As an example consider a hypothetical non-machine learning algorithm for face recognition in images that would try to define what a face is (round skin-like-colored disk, with dark area where you expect the eyes etc). A machine learning algorithm would not have such coded definition, but will *learn-by-examples*: you'll show several images of faces and not-faces and the algorithm will eventually learn and be able to predict whether or not an unseen image is a face.

Machine learning can be supervised or unsupervised (Bolton and Hand, 2002). This particular example of face recognition is classed as supervised machine learning, which means that your examples must be labeled, or explicitly say which ones are faces and which ones aren't. In an unsupervised machine learning algorithm your examples are not labeled, i.e. you don't say anything. Of course in such a case the algorithm itself cannot *invent* what a face is, but it could be able to cluster the data in different classes, e.g. it could be able to distinguish that faces are very different from panoramas, which are very different from horses.

Many methods have been proposed for the development of the user profiles, which is essentially the usage data (e,g. computer / smartphone usage) grouped into a number of clusters based on their similarity. Most of the work has been done in the area of computer intrusion detection and detection of fraudulent activities in mobile device usage (Lane and Brodley, 1997). The techniques used come from the area of machine learning like rule discovery (Adomavicius and Tuzhilin, 1999), clustering (Oh and Lee, 2003), Bayesian rules (Buschkes et al., 1998), or neural network classification (Manikopoulos and Papavassiliou, 2002).

More recently, these machine learning learning techniques (SOM, SVM and K-means Clustering) have also been applied for behaviour-based malware detection on mobile devices. These studies relied evaluating behaviour of known samples of benign and malicious applications by monitoring kernel level OS calls, system load, battery usage etc. Previous researches e.g Crowdroid utilised K-Means clustering to achieve 85% to 100% accuracy in malware detection (Burguera et al., 2011).

We deliberated on choice of system for implementation of our system based on research mentioned earlier and chose machine learning. It is not feasible to use an expert system because of the difficulty in defining rules to capture all possible variations a user's behaviour. A Neural Network is better suited for classification problems whereby input data is labelled (supervised machine learning) and sometimes requires weight adjustments to achieve desired output from the input. The neural networks cannot be retrained so if you add data later, this is almost impossible to add to an existing network and handling of time series data in neural networks is a very complicated. Due to the nature of our problem, unsupervised machine learning technique is better suited. K-Means clustering, a very simple machine learning algorithm is used to validate our system in simulated environment. Both, machine learning and behaviour profiling are a not new concepts, however their application for profiling mobile device users' behaviour is novel.

4.3 K-means clustering

K-Means clustering, which is one of the most popular techniques for cluster detection. This algorithm was first published by (MacQueen, 1967), but the idea goes back to the 1950s. K-Means relies on geometric interpretation of data as point in space. Each record is considered a point in a scatter plot, which in turn implies that all the input variables are numeric. It's called K-Means because it finds k unique clusters for a given dataset. The number of clusters k is user defined. Each cluster is described by a single point known as the *centroid*. Centroid of a cluster is the mean of all points in that cluster.

4.3.1 Algorithm

In the clustering problem, we are given a training set $x^{(1)},...,x^{m}$, and want to group the data into a few cohesive *clusters*. Here, we are given feature vectors for each data point $x^{(i)} \in \mathbb{R}^n$ as usual; but no labels $y^{(i)}$ (making this an unsupervised learning problem). Our goal is to predict k centroids and a label $c^{(i)}$ for each data point. The K-Means clustering algorithm (Piech and Ng, 2013) is as follows:

```
1     Initialise **cluster centroids** $\mu_1, \mu_2, ..., \mu_k \in \mathbb{R}^n$
2     Repeat until convergence:
2.1       For every $i$, set $c^{(i)} := arg_{min_j} \left\| x^{(i)} - \mu_i \right\|^2$
2.2       For each $j$, set $\mu_j := \frac{\sum_{i-1}^{m} 1\{c^{(i)}=j\}x^{(i)}}{\sum_{i-1}^{m} 1\{c^{(i)}=j\}}$
```

The K-Means algorithm works like this. First, the *k* centroids are randomly assigned to a point. Next, each point in the dataset is assigned to a cluster. The assignment is done by finding the closest centroid and assigning the point to that cluster. After this step, the centroids are all updated by taking the mean value of all the points in that cluster. The term *closest cluster* implies some sort of distance measure. The are various distance measures to choose from and selection of distance measure will effect the performance of K-Means. Hence K-Means clustering has data preparation requirements similar to those for case based reasoning. Some of the popular distance measures used for K-Means clustering are Squared Euclidean, City Block, Cosine, Correlation and Hamming.

For ease of explaining, the technique is illustrated using two-dimensional diagrams. Keep in mind that the data usually has more than two dimensions, as you will see in the implementation of our system. However, limiting the example to two dimensions makes it possible to use simple scatter plots to illustrate the procedure, which works the same way in higher dimensions. Figure 4.1 shows three initial cluster seeds (highlighted blue) that have been chosen randomly (example data adopted from Gordon S. L (2011)).

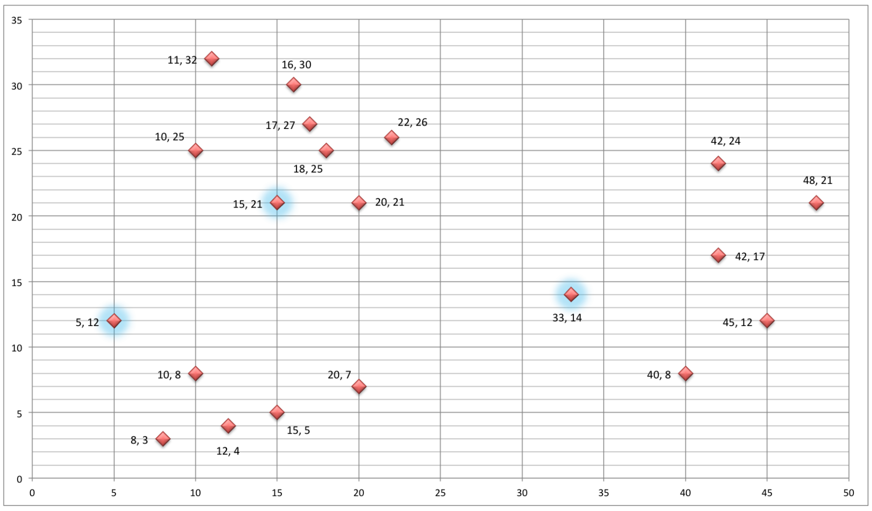


Fig. 4.1 Data points chosen as cluster seeds

After initial clusters are chosen at random from the data, the algorithm alternates between two steps known as the *assignment* step and the *update* step. After choosing cluster seeds, the next step is to assign each record to its closest cluster seed to form initial clusters, as shown in Figure 4.2. This figure also shows the lines that separate the clusters. The cluster boundaries form a Y shape with one cluster on the left, one on the right, and one on the top. Drawing the boundaries between clusters is useful for showing the process geometrically. In practice, though, the algorithm only needs to calculate the distance from each record to each seed, to assign all records to the closest seed. The assignment step can be performed without actually determining the cluster boundaries.

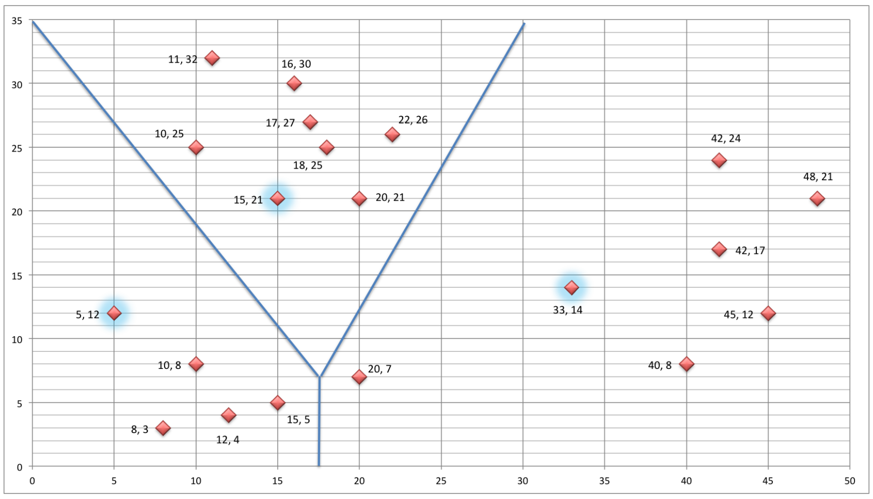


Fig. 4.2 Assigning data points to closest seeds

In the update step, the centroid of each cluster is calculated. The centroid is simply the average position of cluster members in each dimension. Figure 4.3 illustrates the update step. In the assignment step, each data point is assigned to the cluster whose centroid is closest to it. In this example, the highlighted data point is assigned to a new cluster. When the assignment step changes cluster membership, it causes the update step to be executed again.

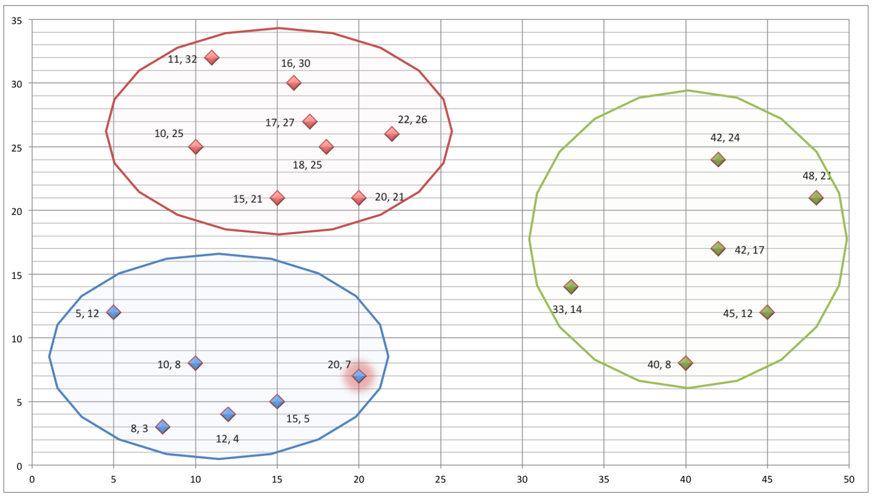


Fig. 4.3 Updating centroids

The algorithm continues alternating between update and assignment until no new assignments are made. Generally, in practice this algorithm converges quite quickly. Figure 4.4 shows the final clusters after the highlighted record has been reassigned and the new centroids have been calculated.

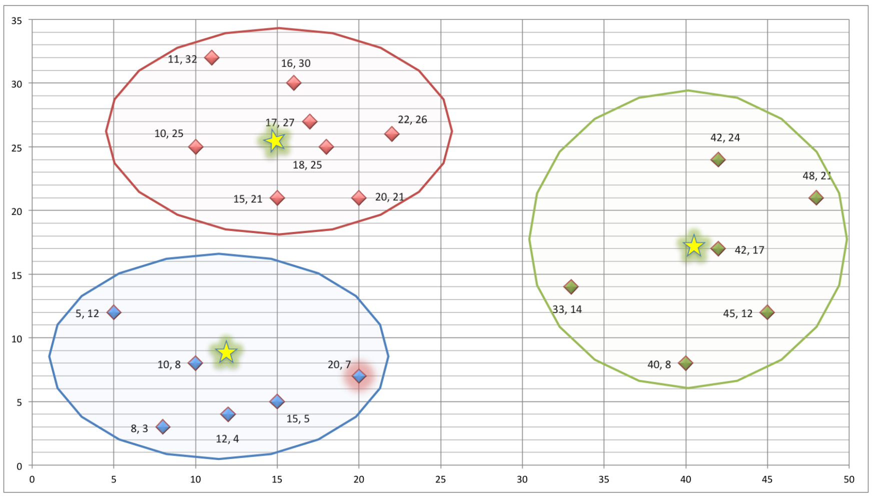


Fig. 4.4 Completion of clustering process

4.3.2 Choosing cluster seeds (initial clusters)

Different choices for the initial cluster seeds can lead to different final clusters. Inappropriate selection of cluster seeds can lead to clearly sub-optimal clusters is easy.

Figure 4.5 is one example. Here, A is closest to C and B is closest to D, but if A and C are chosen as the initial seeds, B is closest to A, and D is closest to C, which leads to the clusters shown. Because of examples like this, researchers have developed a variety of algorithms for choosing good positions for the initial seeds. As a practical matter, however, if different choices of initial seed positions yield radically different clusters, then the K-Means algorithm is not finding stable clusters. This could be because there are no good clusters to be found, or because a different choice for k is needed, or it could be because clusters are present but the K-Means algorithm is unable to detect them.

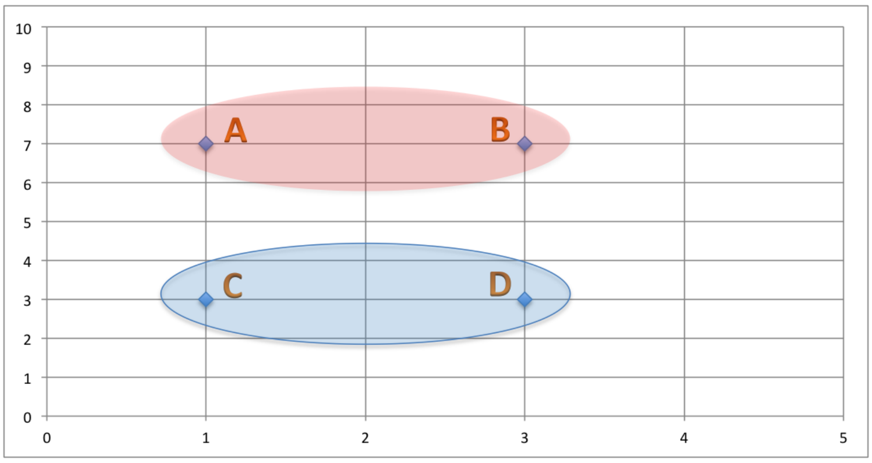


Fig. 4.5 Incorrect clustering

4.3.3 Choosing number of clusters (k)

How many clusters should you look for? In many cases, the answer to that question is supplied by the business purpose. If the point of customer segmentation is to design different offerings for each segment, k is determined by the number of segments the business can reasonably support. Often, however, the needs of the business do not dictate a precise value for k; there is a range of acceptable values. When a range of acceptable values exists, the best way to choose is to build clusters using each value of k and then evaluate the resulting clusters. The evaluation may be subjective, these segments make sense to me, or it may be based on technical criteria such as the ratio of the average intra-cluster distance to the average inter-cluster distance, or the ***cluster silhouette***, a measure of cluster goodness that is explained in section 4.4.2.

4.4 Cluster evaluation

Generally speaking, members of clusters should have a high degree of similarity. In geometric terms, cluster members should be close to each other and the separation between clusters should be wide. When detecting customer segmentation, clusters should have roughly the same number of members. An exception to this rule is when clustering is used for detecting fraud or other anomalies. There may be a small number of compact, but populous clusters representing the most usual cases and a few less-populous clusters of unusual cases that require further investigation.

A general-purpose measure that works with any form of cluster detection is to take whatever similarity measure or distance metric is used to form the clusters and use it to compare the average distance between members of a cluster and its centroid to the average distance between cluster centroids. This can be done for each cluster individually and for the entire collection of clusters.

4.4.1 Cluster measurements

Which is bigger: Glasgow or Edinburgh? If you answered Edinburgh, you're right. Edinburgh covers more than 100 square miles; Glasgow occupy only 68. If you said Glasgow, you are also right because its population of more than more than that of Edinburgh. Similarly, depending on context, a small cluster might be one with few members, or one with low variance, or one with a small diameter. Because of this potential for confusion, using unambiguous adjectives such as *compact* or *populous* in preference to *small* or *large* is a good idea. For describing how much space a cluster takes up and, and how tight or dispersed it is, the technical terms cluster diameter and cluster variance have precise definitions. *A cluster's diameter* is the square root of average mean squared distance between all pairs of points in the cluster and is shown the equation below (Aslam, 2014):

$$D_m = \sqrt{\frac{\sum_{i=1}^{N}\sum_{i=1}^{N}(t_{ip} - t_{iq})^2}{N(N-1)}}$$

Note that, in general, this is not the same as twice the distance from the centroid to the record farthest from the centroid. *A cluster's variance* is the sum of the squared distance from the centroid of cluster members and shown in the equation below (Torsten, 2011):

$$\sigma_i^2 = 1/(|C_i| - 1) * \sum_{xinC_i} (x - \mu_i)' * (x - \mu_i).$$

where

$$\mu_i = 1/|C_i| * \sum_{xinC_i} x$$

A cluster whose members are all close to the centroid has low variance. The square root of the variance also makes a useful measure, because it is in the same units as the distance. Another measure of a cluster's dispersion is its silhouette, described in the next section.

4.4.2 Cluster silhouettes

A cluster's silhouette is a measure of cluster goodness first described by (Rousseeuw, 1987). It has some nice properties, but at the expanse of large number of calculations required for clusters have many members. For the purpose of illustration, Figure 4.6 gives the (x,y) coordinates of the points used as an example. To calculate a cluster's silhouette, first calculate the average distance within the cluster. Each cluster member has its own average distance from all other members of the same cluster. The average

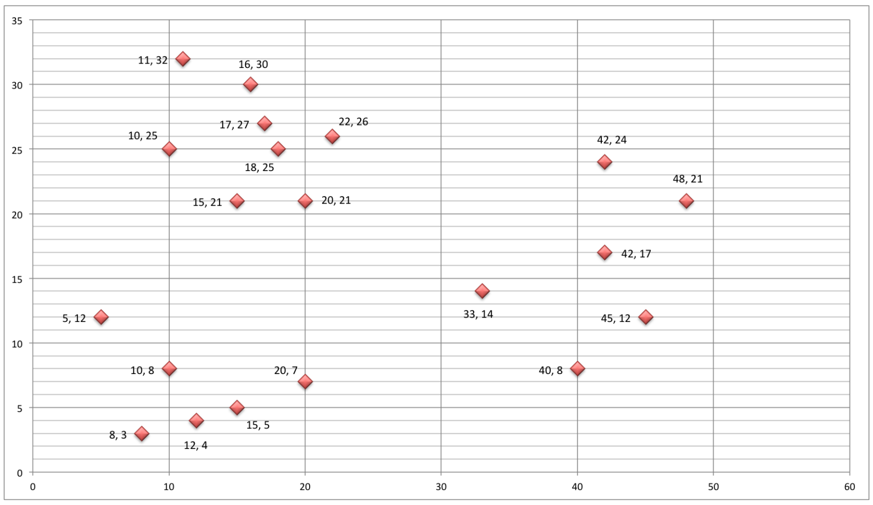


Fig. 4.6 The cluster silhouette measure

of these averages is the dissimilarity score for the cluster. Figure 4.7 shows how the dissimilarity score for one record is calculated. Cluster members with low dissimilarity are comfortably within the cluster to which they have been assigned. The average dissimilarity for the cluster is a measure of how compact it is. Next calculate the

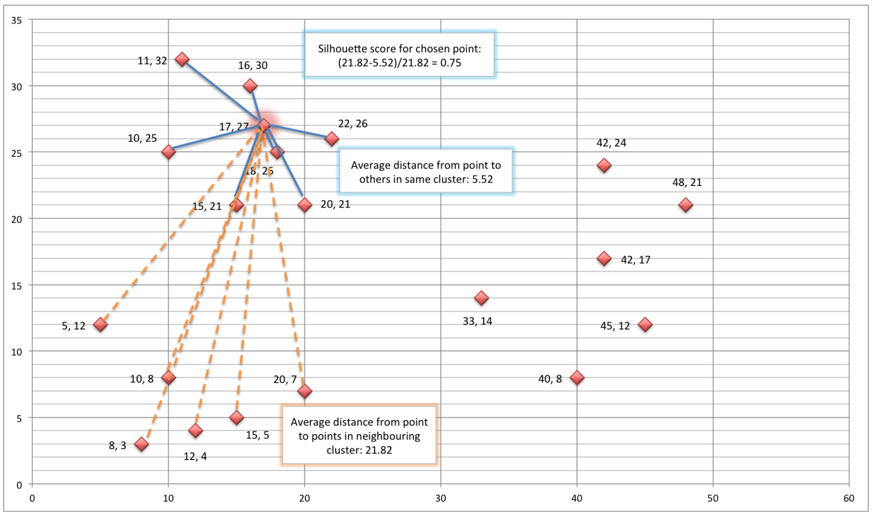


Fig. 4.7 The dissimilarity score

dissimilarity of each cluster member from the members of its next nearest cluster. Note that two members of the same cluster may have different neighbouring clusters. For points that are close to the boundary between two clusters, the two dissimilarity scores may be nearly equal. Points far from a boundary should be much more similar to other members of their own cluster than they are to members of the neighbouring cluster.

Finally, each point is given a score that is the difference between its average dissimilarity with members of its own cluster and its average dissimilarity with members of its neighbouring cluster divided by the latter. This is the *silhouette score*.

4.4.3 Silhouette range

Typically silhouette score ranges from *zero* when a record is right on the boundary of two clusters to *one* when it is identical to the other records in its own cluster. Figure 4.8 shows the silhouette scores for the records and clusters from 4.7. Theoretically

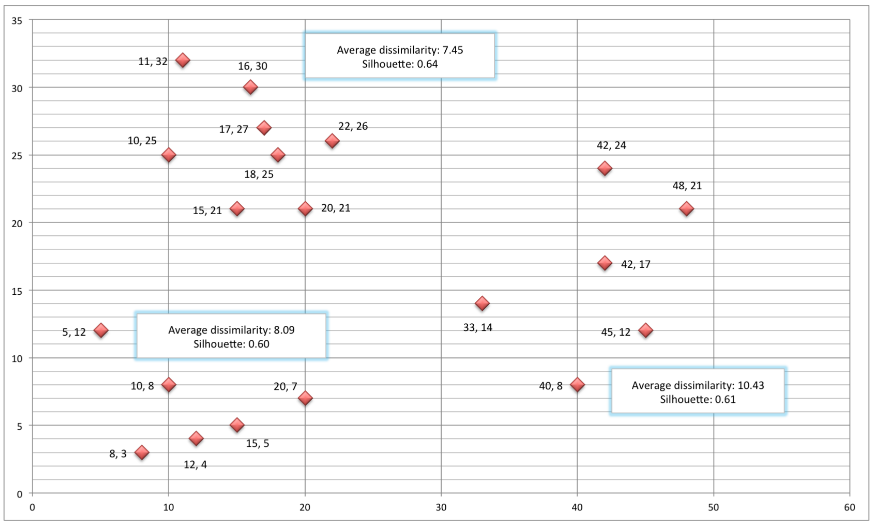


Fig. 4.8 The cluster silhouette

speaking, the silhouette score can go from negative one to one. A negative value means that the record is more similar to the records of its neighbouring cluster than to other members of its own cluster. This could potentially happen because the K-Means algorithm assigns records based on which cluster centroid they are nearest which, in unusual cases, could be different from the most similar cluster according to the silhouette measure. A cluster's silhouette is the average of these scores.

The silhouette score for an entire cluster is calculated as the average of the silhouette scores of its members. This measures the degree of similarity of cluster members. The silhouette measure can be applied at the level of the dataset to determine which clusters are not very good and at the level of a cluster to determine which members do not fit in very well. The silhouette can be used to choose an appropriate value for k in K-Means by trying each value of k in the acceptable range and choosing the one that yields the best silhouette. It can also be used to compare clusters produced by different random seeds.

4.4.4 Local minimisation issue

The starting points (cluster seeds) affect the solution that K-Means reaches. It is possible for K-Means to reach a *local minimum* which may result in less than best possible result. This happens because re-assigning any one point to a new cluster would increase the total sum of point-to-centroid distances, but a better solution does exist. This can be fixed by repeating the process with different starting points, choosing the solution, with lowest value for final sum of distances.

4.5 Summary

This chapter presents the basics of clustering algorithm for profile mobile device users for the purpose of anomaly detection. The clustering process is a widely used technique for classification and customer segmentation in the industry. It is well suited for situations where the dataset in unlabelled. Silhouette score is the most commonly used statistical test for measuring the accuracy of clustering process. Although K-Means clustering is prone to issues like local minimisation, this can be easily fixed by replication. The next chapter will present the design of our proposed behavioural profiling based anomaly detection system which utilises K-Means clustering algorithm as its core.

Chapter 5

User behaviour profiling system

5.1 Introduction

Behaviour profiling has the potential to continuously verify a user in an easy and effective manner: a user is verified based upon which applications they utilise. If the verification of a user was possible while they interact with mobile applications, the verification process could be performed non-intrusively and continuously for the duration of usage. The literature in Chapter 4 has shown that the existing behavioural based mobile IDS can only monitor network-based services (e.g. telephony) through telecommunication service provider's networks. As current mobile devices have the ability to access multiple networks simultaneously, a host based approach must be taken into consideration when investigating a new security mechanism. Little research has been published regarding how behaviour profiling techniques perform within the mobile host environment despite that the mobile application usage represents an overview of how the user interacts with the device (Miettinen and Halonen, 2006). Hence, it is critical to identify the effectiveness of utilising the behaviour profiling technique to verify a user's identity via their application usage within the mobile host environment. Although it is proved that the calling behaviour can be utilised to identify mobile users over telecommunication service provider's networks, the calling service's effectiveness towards verifying a user's identity within the mobile host environment is uncertain as its features have changed slightly, such as the IMSI cannot be utilised anymore. The following section presents the building blocks our proposed system.

5.2 Rationale for implementation of proposed system

The proposed anomaly detection system utilises machine learning method (K-Means Clustering) to perform behaviour analysis of the smartphone users. In our system various metrics including, CPU consumption, Network (3G, Wi-Fi) traffic, number of running processes, battery level etc. are continuously monitored from the smartphone to detect malicious activity. Our proposed Host-based Intrusion Detection System (HIDS) is aimed to be lightweight in terms of processor and memory load and battery consumption. The intrusion detection will work in real-time by collecting, processing and examining of various system features to detect patterns of malicious activity on the smartphone.

We have chosen to implement HIDS prototype on Google Android platform. The rationale for this choice is discussed hereinafter.

5.2.1 Google Android: the user's choice

The number of mobile devices running on the Android operating system has risen considerably during last year and this trend is going to continue in the following years. By the end of this year, Android will be in more devices than the next four competitors combined (Windows, iOS, Mac OS, and BlackBerry). Before the end of this decade, Android will be in nearly as many devices as all other operating systems combined.

According to market research firm Gartner, in 2012, Android pulled ahead of Windows as the world's dominant operating system (counting PCs, tablets, and phones) by about 150 million devices. This year, it will ship on more than double the number of devices running Windows; and it will nearly triple Windows shipments from 2014 to 2017, according to Gartner. Google's Android OS was installed on 497 million

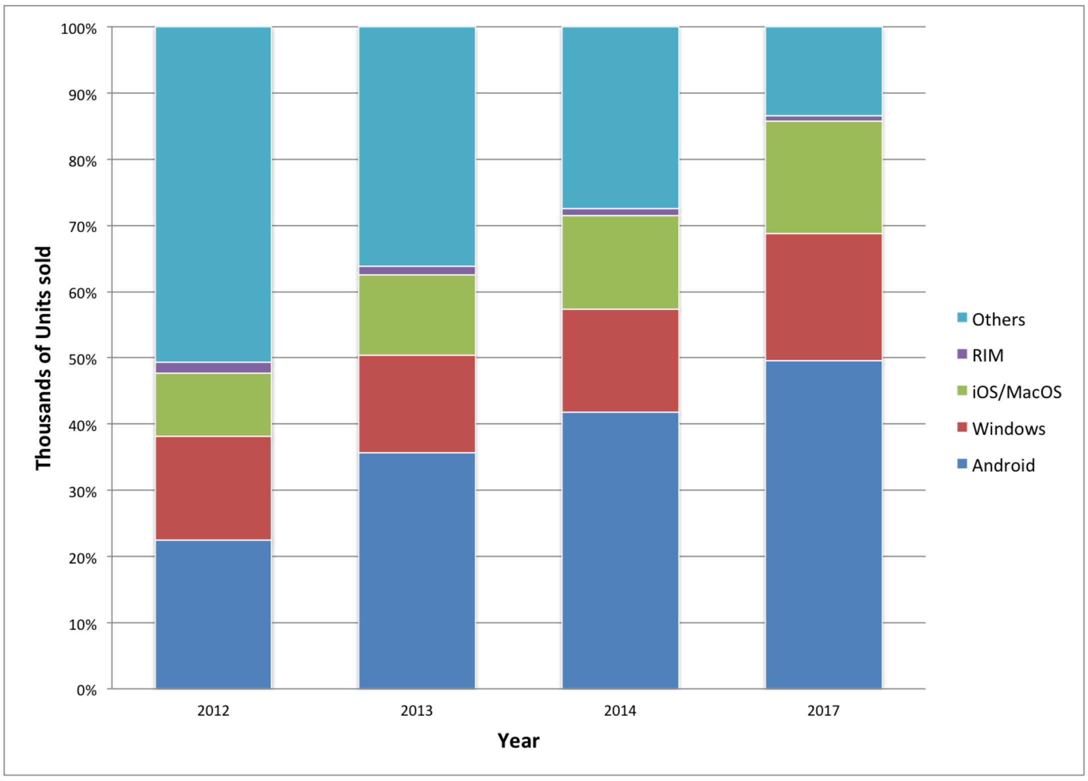


Fig. 5.1 Estimate of worldwide devices shipments by OS (Thousands of Units) Source: (Milanesi et al., 2013)

devices that shipped in 2012 and will grow by 364 million units to about 861 million this year according to Gartner's most recent forecast. In 2014, Android will make the jump into the 10 figures with shipments of 1.07 billion units. By 2017, that figure will climb by another 400 million units to 1.47 billion. (Milanesi et al., 2013). Figure 5.1 shows this trend.

5.2.2 Google Android: hackers haven

The rise in popularity of an OS comes with its price. The same thing happened with Symbian a while back. In 2004-2006, Symbian was indisputably the platform most frequently targeted by virus writers. However, it was pushed out by Android, as threats for this platform were able to reach a greater number of mobile devices. Today, Android is the worlds most widespread operating system for mobile devices, which is why so many new threats have been developed for this platform. Figure 2.3 shows the distribution of mobile malware by platform during the period of 2004 to 2013.

Android has established itself as the cyber criminals target of choice when it comes to mobile operating systems and can be considered the mobile worlds equivalent to Windows. The evolution of mobile malware in 2013 was a predominantly focused on Android. This resulted in both qualitative and quantitative growth in mobile malware for this platform. Over 98% of all the mobile malware detected every month, by Kaspersky labs was designed for Android.

Statistically speaking, a total of 143,211 new modifications of malicious programs targeting mobile devices were detected in all of 2013 (as of January 1, 2014). In 2013, 3,905,502 installation packages were used by cyber criminals to distribute mobile malware. Overall in 2012-2013 we detected approximately 10,000,000 unique malicious installation packages. Virtually all mobile samples that were discovered in the mobile realm during 2013 were targeting Android.

The reason for the huge growth of Android can be explained by three factors: First of all, the Android platform itself has become incredibly popular, becoming the most widespread OS for new phones, with over 70% market share. Secondly, the open nature of the operating system, the ease with which applications can be created and the wide variety of (unofficial) application markets have combined to shine a negative spotlight on the security posture of the Android platform. Last but not least, evidence was found that mobile devices and the data stored on them are targets not only for your run-of-the-mill cyber criminals but also a variety of organisations behind attacks like Red October (Chebyshev and Unuchek, 2014).

Looking forward, there is no doubt this trend will continue, just like it did with Windows malware many years ago. We are therefore expecting 2014 to be filled with targeted attacks against Android users, zero-days and data leaks.

5.3 Use cases for HIDS

A set of high level use cases of our HIDS for both the smartphone user and the remote administrator (the researcher) is shown in figure 5.2 each of which is briefly described here:

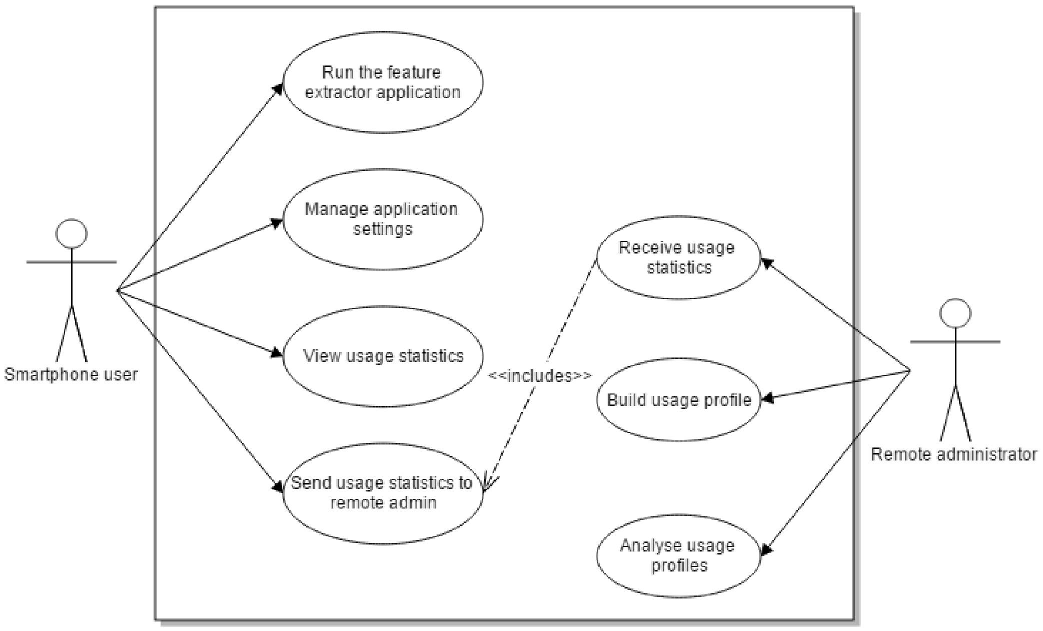


Fig. 5.2 The use cases for HIDS

Run the feature extraction application We developed an Android application for collecting various usage metrics from smartphones including calls, messaging, system load and user mobility etc. Functionality of this application and collected usage metrics are discussed in depth in the section 5.5. The only action required from the participants was to install and run this application on their smartphones as it runs in the background as a service.

Manage feature extraction application settings The smartphone user has the ability to interact with feature extractor application to manage application preferences.

View usage statistics The smartphone user can view their device usage statistics via the feature extractor application in descriptive numerical and graphical form.

Send usage statistics to remote administrator The smartphone user sends the device usage statistics to the remote administrator for profile generation. The feature extractor application (discussed in section 5.5) provides the smartphone user, the ability to send this data.

Receive usage statistics The remote administrator received the smartphone usage statistics from the user via email. The data is received as an SQLite database file.

Build usage profile The remote administrator pre-processes the data received from each user to build usage profiles. This process is conducted offline on a PC using Matlab software. The process of profile generation is discussed in section

Analyse usage profile The remote administrator analyses the usage profiles received from all the smartphone users of our study to determine the accuracy of profiling.

5.4 The components of HIDS

In this section, we present the main components of our proposed HIDS. This system can detect malicious activity on the mobile device in real-time by employing *unsupervised machine learning* technique called *k-means clustering* that has already been explained in section 4.3. The main architecture of our HIDS is presented in 5.3.

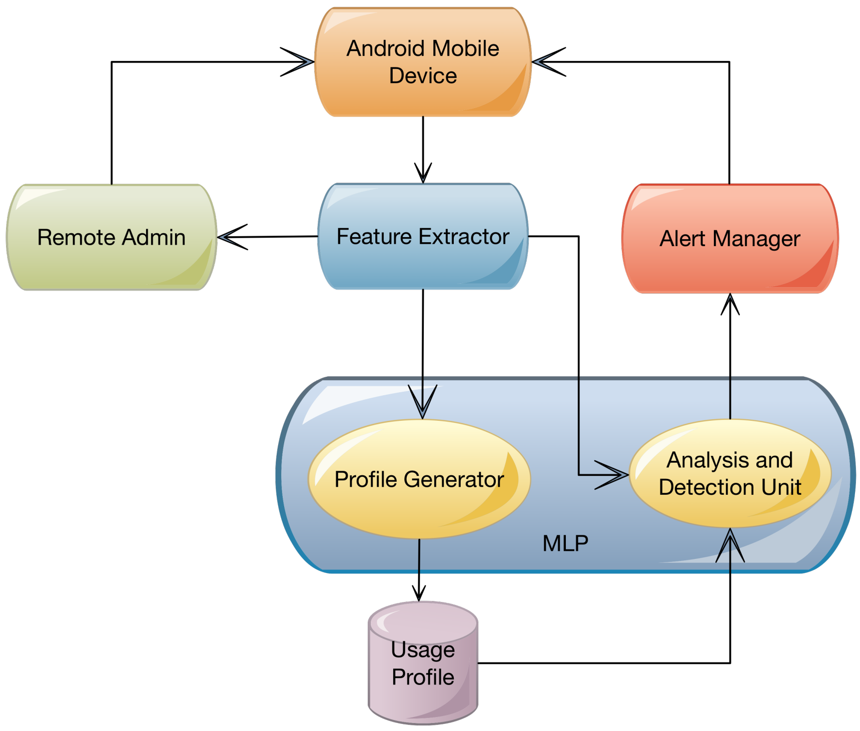


Fig. 5.3 The machine learning based IDS

5.5 Feature extractor

The Feature Extractors is the most crucial component of the HIDS as it communicates with various components of the Mobile Device Operating System, including kernel and the Application Framework Layer in order to collect raw features (system metrics). The collected features are fed to the Machine Learning Processor as input during its two operational phases.

It is implemented as an Android application that uses a background service to collect system features from mobile device. The epoch for feature collection is set at every 5 minutes. Since the smartphone owners may use their phones differently under different conditions for example during week days number of calls/messages may be less than weekends. Smartphone owners may prefer to play games or only read emails while commuting on the train. So in order to ensure that all possible usage scenarios are included in the usage profile optimum number of observations has to be

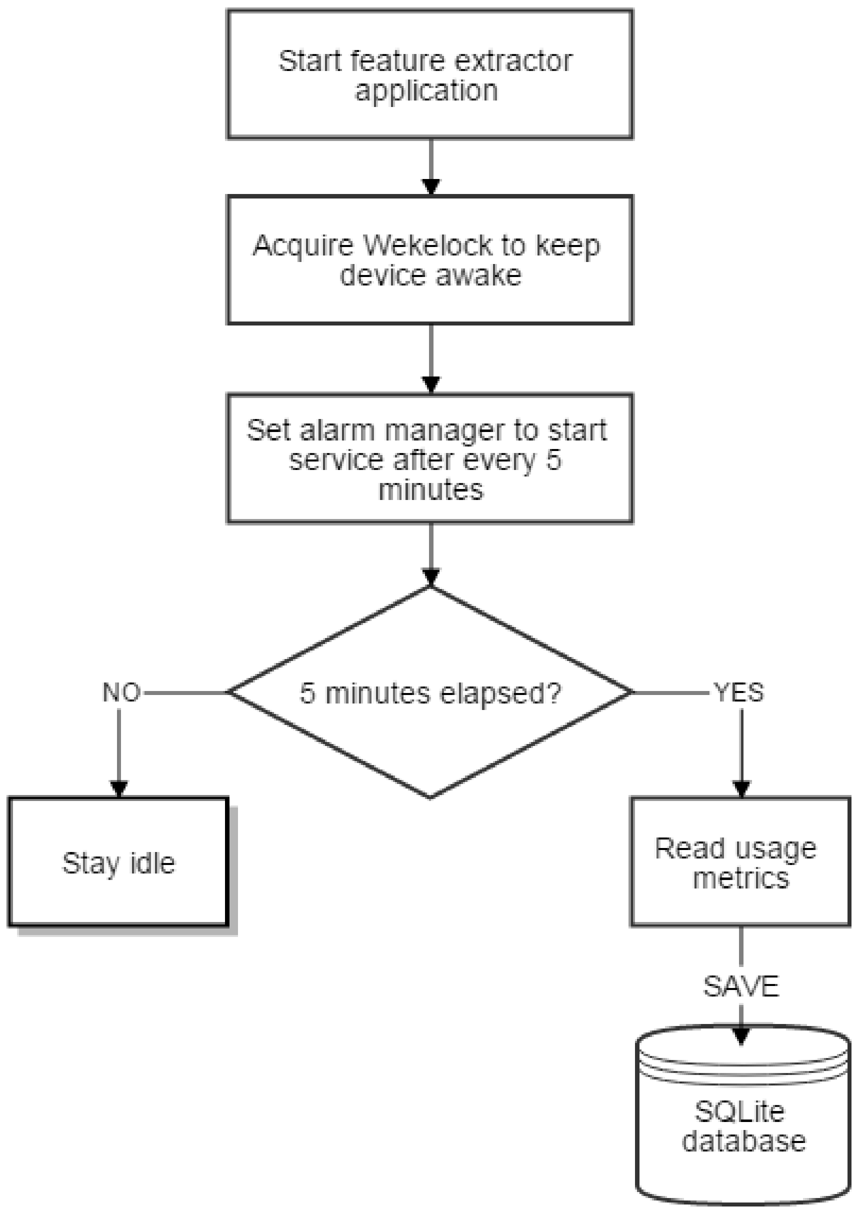

Start feature extractor application
Acquire Wekelock to keep device awake
Set alarm manager to start service after every 5 minutes
5 minutes elapsed?
NO
YES
Stay idle
Read usage metrics
SAVE
SQLite database

Fig. 5.4 The flow chart of feature extractor

made. According to our tests, the feature extraction should continue for two weeks to ensure comprehensive usage profile. The flow chart of the feature extractor application is shown the figure 5.4

The background service in our feature extractor application is very light-weight on processor, memory and battery resources. If we used an always running service, it would be a waste of resources for the mobile device. This type of service is also prone to getting *force closed* by the device user with the use of any Task Killer application. Both these factors will jeopardise the effectiveness of HIDS. To overcome this issue, this service employs the Android's equivalent to *Cron Jobs* for Linux or *Scheduled Tasks* for Windows. That is, using **Alarm Manager** with an **IntentService**. The idea is, the main Activity sets the time interval (5 minutes) for feature extraction. After every 5 minute interval, the AlarmManager starts the background service (feature extraction). After data collection is finished the AlarmManager is set for next interval and the service terminates.

Another major issue which our specially designed service overcomes is those periods when the devices go to sleep. An Android device is in active state when user is interacting with the device. When the screen goes off after the time interval (set by the user usually from 15 seconds to 10 minutes) android system powers down the CPU and GPU, Wi-Fi and other sensors in order to reduce the power consumption.

An android mobile device will spend more time in sleep mode than being actively used by the user. Hence, if the device is asleep during the time when next feature extraction is scheduled, no data can be extracted. To counter this issue we need to bring the device to Awake state. This is achieved by creating a WakeLock which keeps the device awake for as long as all the feature extractors have finished their task.

So our feature extractor application uses Android features like AlarmManager, IntentService and WakeLock to effectively perform the task of feature extraction. The screen-shots of our feature extractor application developed for Google Android platform is shown in Appendix A. After every 5 minute interval, following system metrics are recorded by feature extractor service and saved into SQLite database on the device.

CONTACT_CALLS The number of calls made to phone-numbers stored in the address book.

CONTACT_CALLS_SEC The duration (seconds) of calls made to phone-numbers stored in the address book.

UNKNOWN_CALLS The number of calls made to phone-numbers not present in the address book.

UNKNOWN_CALLS_SEC The duration (seconds) of calls made to phone-numbers not present in the address book.

SMS_OUT_CONTACTS The number of SMS messages sent to phone-numbers stored in the address book.

SMS_OUT_UNKNOWN The number of SMS messages sent to phone-numbers not present in the address book.

CPU_USAGE The Average CPU usage since last epoch.

RAM_USAGE The Average RAM usage since last epoch.

SYS_APPS The number of applications running on the device that are pre-installed by Google.

USER_APPS The number of applications running on the device that are pre-installed by user.

SYS_SERVICES The number of services running on the device that are pre-installed by Google.

USER_SERVICES The number of services running on the device that are installed by the user.

BYTES_TX The number of bytes of data transmitted from the device since last epoch.

BYTES_RX The number of bytes of data received by the device since last epoch.

LOC_LAT The latitude of the geographic location of the device based on cell towers.

LOC_LON The longitude of the geographic location of the device based on cell towers.

TIME_SLICE This represents the portion of a week. In order to get more accurate mobile device usage we have divided a week into 28 slices. The full mapping of time of observation to *TIME_SLICE* is shown in the Table 5.1.

SCREEN_ON The status of the mobile device screen at the time of feature collection. If the user is physically interacting with the device then we record 1 else 0.

READ_AT The time of the feature collection.

Table 5.1 Observation time to TIME_SLICE mapping

	06:00 to 11:59	**12:00 to 17:59**	**18:00 to 23.59**	**24:00 to 05:59**
Monday	11	12	13	14
Tuesday	21	22	23	24
Wednesday	31	32	33	34
Thursday	41	42	43	44
Friday	51	52	53	54
Saturday	61	62	63	64
Sunday	71	72	73	74

The users were asked to run the feature extractor application for a minimum of 15 days and then send the usage statistics (the SQLite database file) via email option provided in the application.

5.6 Machine learning processor

The Machine Learning Processor (MLP) relies on *k-means clustering* technique which has been discussed in detail in Section 4.3 to create usage profile from the input data. We have implemented our MLP using Mathworks Matlab 2013a, which provides feature rich Machine Learning library and tools for data analysis and visualisation. The MLP has has two operational modes (training and testing) that are controlled by separate component of MLP (profile generator and anomaly detector). The processes of usage profile generation and anomaly detection is shown in the figure 5.5 and discussed in more detail in the followings sections.

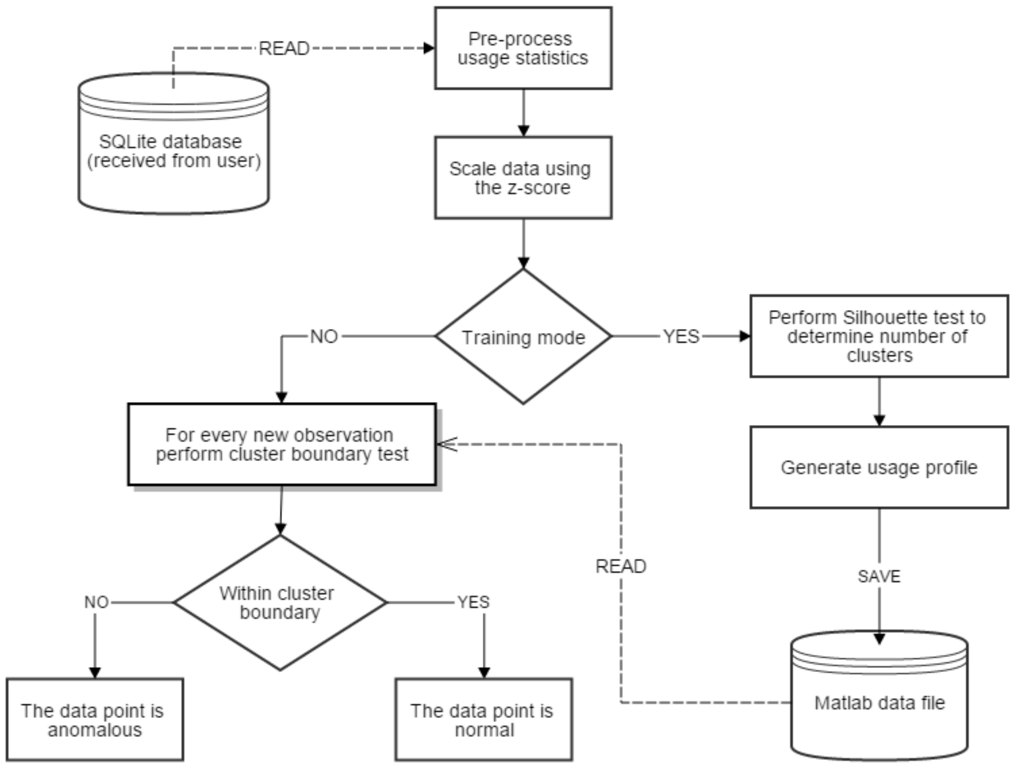

READ
Pre-process usage statistics
SQLite database (received from user)
Scale data using the z-score
Training mode
NO
YES
Perform Silhouette test to determine number of clusters
For every new observation perform cluster boundary test
Generate usage profile
READ
SAVE
Within cluster boundary
NO
YES
Matlab data file
The data point is anomalous
The data point is normal

Fig. 5.5 The flow chart of MLP

5.6.1 Training mode - profile generation

In *training mode* the data (extracted features) from the *Feature Extractors* is processed and scaled for consistency, a statistical test is performed to determine the optimum number of clusters *k* and then a usage profile is generated. The pseudo code for profile generator is presented in listing 5.1.

Listing 5.1 Profile generator pseudo code

```
Read training data from csv file.

for every data point in our dataset
       Call pre-processing routine.
       Scale using z scores.

for a range of values of k
       Perform silhouette test.
       Choose k with highest silhouette score

Call K-Means Clustering routine using k

for k clusters
       calculate the cluster boundary
```

Each stage of profile generation is explained in more detail in the following sections.

5.6.1.1 Input preprocessing

Once the data (extracted features) from the *Feature Extractors* is received in the form of a database file from smartphone user, its data is exported to a CSV file. The mapping of database table to CSV file is shown in the Table 5.2. Once the CSV file is created, its data is imported into Matlab for further processing. In order to facilitate the scaling process the user's Geolocation (longitude and latitude) is converted into distance from

Table 5.2 Database to CSV file mapping

CSV	DATABASE
CALLS	CONTACT_CALLS+UNKNOWN_CALLS
CALLS_DURATION	CONTACT_CALLS_SEC+UNKNOWN_CALLS_SEC
SMS_OUT	SMS_OUT_CONTACTS+SMS_OUT_UNKNOWN
CPU_USAGE	CPU_USAGE
RAM_USAGE	RAM_USAGE
SYS_APPS	SYS_APPS
USER_APPS	USER_APPS
SYS_SERVICES	SYS_SERVICES
USER_SERVICES	USER_SERVICES
BYTES_TX	BYTES_TX
BYTES_RX	BYTES_RX
LOC_LAT	LOC_LAT
LOC_LON	LOC_LON
SCREEN_ON	SCREEN_ON
TIME_SLICE	TIME_SLICE

a reference location which in the case our MLP was chosen to be Charring Cross, London. The conversion is done using the *Haversine formula* that is is used to calculate the great-circle distance between two points, which is the shortest distance over the earth's surface. (Wikipedia, 2014). Given two geolocations (latitude and longitude pairs) i.e location1(ϕ_1, λ_1), location 2 (ϕ_2, λ_2) and r as the radius of the sphere, the distance between these two geolocations can be found by using the equation 5.1

$$d = 2r\arcsin(\sqrt{\sin^2(\frac{\phi_2-\phi_1}{2}) + \cos(\phi_1)\cos(\phi_2)\sin^2(\frac{\lambda_2-\lambda_1}{2})} \qquad (5.1)$$

5.6.1.2 Scaling for consistency

In geometry, all dimensions are equally important. A distance of 5 on X dimension is the same as the distance of 5 on the Y dimension. It doesn't matter what units X and Y are measured in, as long as they are the same. Unfortunately, in commercial data mining problems including ours, usually no common scale is available because of the different units being used to measure different things. For example in our case CPU_USAGE, CALLS, USER_APPS, BYTES_TX can not be converted to a common unit. On the other hand it will be misleading that a CPU_USAGE of 50% is indistinguishable from a value of 50 Bytes for BYTES_TX. The best solution to overcome this problem is to use z_score also known as standardisation. It removes the bias present due to different units so that each feature has a mean of zero and standard deviation of one, they contribute equally when the distance between two records is computed. Equation 5.2 shows the calculation of a z_score.

$$z\_score = \frac{x - \mu}{\sigma} \tag{5.2}$$

where x is the value of a single feature (e.g. CPU_USAGE) from list of observations, μ is the *mean* and σ is the *standard deviation* for all values of that feature.

5.6.1.3 Determining number of clusters

Due to random selection of cluster seeds at the start of K-Means clustering it doesn't always yield the same result and may converge to a local minima. The accuracy of the clustering process (i.e., the correctness of the usage profile) depends on number of clusters(denoted by k). The selection of this number depends on the prior domain knowledge. It is also important to note that finding k in two, three, or four dimensions is much easier than in twenty. For the clustering to be accurate, records should have similar values in all dimensions. To assist in the process of selecting value of k for high dimensional data, we have used both visual and statistical techniques.

Scaling Multidimensional scaling (hereinafter abbreviated to MDS) refers to a set of models by means of which the information contained in a dataset can be represented by a set of points in space. These points are arranged in such a way that the distance between the points reflect the empirical relationships in the data. Thus producing a picture of the data which is much easier to visually assimilate than a large matrix of numbers. In doing so, MDS may also bring out features of the data which were obscured in the original data.

We used MDS to provide a visual representation of the pattern of proximities (i.e., similarities or distances) among a set of observations made on the smartphone. Figure 5.6 shows an example of MDS performed on 4719 rows of 14 dimensional data acquired from a Google Android based smartphone.

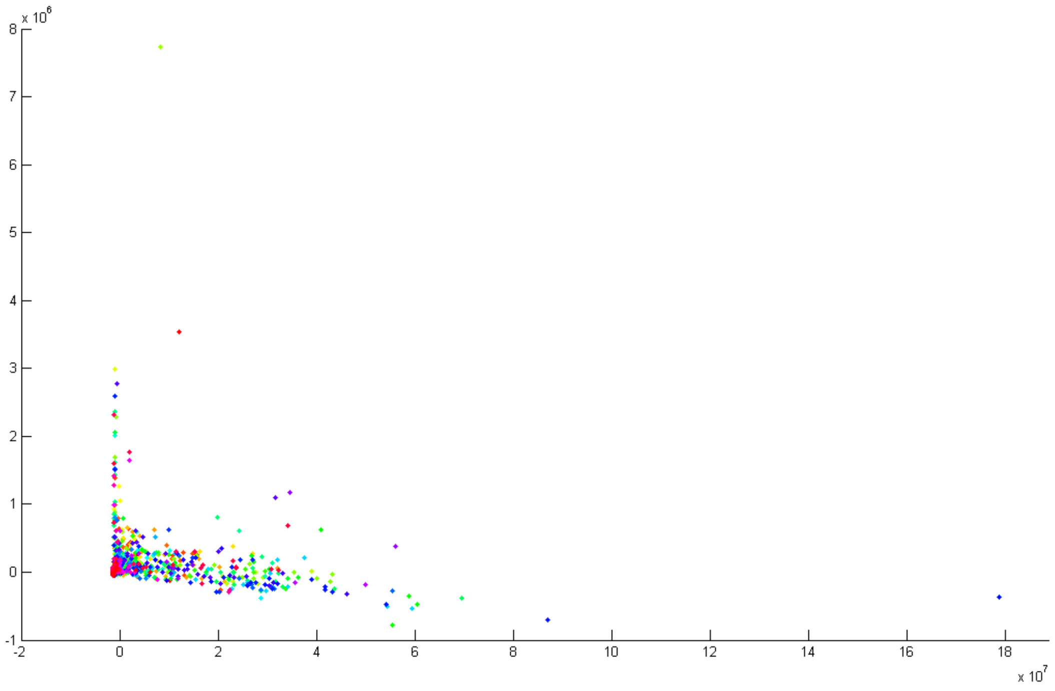


Fig. 5.6 Scatterplot of 14D smartphone usage data scaled to 2D

Silhouette Test Although MDS makes it possible to visually analyse the data to estimate the number of clusters, it is not definitive and should be complemented by some form of statistical measure. Hence, we have used cluster silhouette test which provides a statistical measure of cluster goodness. This process has been expressed in more detail in section 4.4.2. K-Means Clustering process is performed on usage data, for number of clusters ranging from 2 to 10. The value of *k* which results in highest silhouette score is used for profiling. The Table 5.3

shows the silhouette score for K-Means Clustering process where with *k* in the range of 2 to 10.

Table 5.3 Silhouette Score for different values of k

CLUSTERS (k)	SILHOUETTE SCORE
02	0.4140
03	0.3893
04	0.4044
05	0.4377
06	0.2675
07	0.2744
08	0.2824
09	0.2513
10	0.2608

5.6.1.4 Profile generator

Once best value of *k* is determined using the silhouette test, the usage profile of the user is generated by further processing the results of K-Means Clustering. As discussed earlier in section 4.4.4, K-Means suffers from local minimization issue. To overcome this, The clustering process is replicated until a global minimum is achieved. The figure 5.7 shows the flow chart of profile generation process.

The usage profile is saved as a **mat** file which is Matlab's native file format. The usage profile consists of following parameters:

TRAINING DATA The usage data from user's smartphone which is scaled using z_scores and represented as *n-by-14* matrix. Here *n* is the number of observations.

MU The mean calculated during z_score calculation. It is represented as *1-by-14* vector where each column represents the mean of the corresponding variable in the training data.

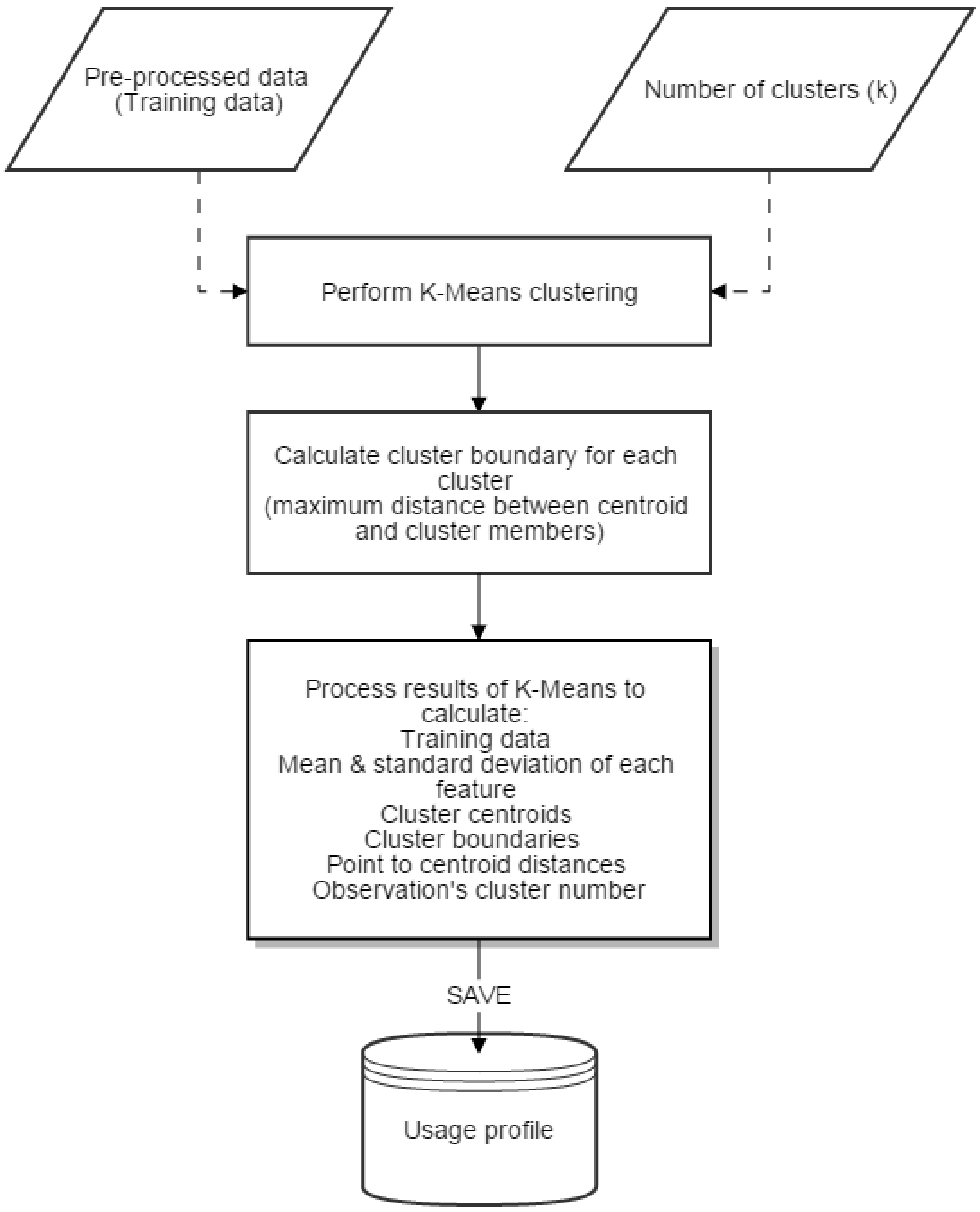

Pre-processed data
(Training data)
Number of clusters (k)
Perform K-Means clustering
Calculate cluster boundary for each cluster
(maximum distance between centroid and cluster members)
Process results of K-Means to calculate:
Training data
Mean & standard deviation of each feature
Cluster centroids
Cluster boundaries
Point to centroid distances
Observation's cluster number
SAVE
Usage profile

Fig. 5.7 The flow chart of profile generation

SIGMA The standard deviation calculated during z_score calculation. It is represented as *1-by-14* vector where each column represents the standard deviation of the corresponding variable in the training data.

IDX The cluster indices of observations (data points) represented by *n-by-1* vector.

C The cluster centroids represented as *k-by-14* matrix, where k is number of clusters. Each row of this matrix represents a cluster centroid.

CB The cluster boundaries, i.e., the distance from centroid and the farthest point with the cluster. It is represented by a *1-by-k* vector.

D The distance from each data point (observation) to every centroid represented by *n-by-k* matrix.

The distance between an data point (observation) in the cluster and its centroid is measured using Squared Euclidean Distance (SED). Euclidean distance between an observation O and centroid C is the length of the line segment connecting them ($\overline{\mathbf{OC}}$). In Cartesian coordinates, if $O = (o_1, o_2, ..., o_n)$ and $C = (c_1, c_2, ..., c_n)$ are two points in Euclidean n-space (in our case $n = 14$), then the euclidian distance (d) from O to C, or from C to O is given by the Pythagorean formula.

$$d(O,C) = d(C,O) = \sqrt{(o_1 - c_1)^2 + (o_2 - c_2)^2 + ... + (o_n - c_n)^2} \tag{5.3}$$

$$d(O,C) = \sqrt{\sum_{i=1}^{n} (c_i - o_i)^2} \tag{5.4}$$

$$SED(O,C) = d(O,C)^2 \tag{5.5}$$

A cluster's boundary **CB** is defined by the maximum distance between data points and its centroid. For a cluster with *n* data points, the cluster boundary can be calculated by using equation 5.6.

$$CB = max(|CO_1|, |CO_2|, ..., |CO_n|) \tag{5.6}$$

5.6.2 Testing mode - anomaly detection

The second operational phase of the MLP is *Testing mode*. In this mode, the data from the *Feature Extractors* is pre-processed, scaled using the same operations as in the training mode. every observation in the test data is compared with the usage profile generated during the training phase to determine anomalous observations.

A test was developed (cluster boundary test) to classify new observations and anomalous (from different user) or benign (same). The figure 5.8 shows the flow chart of this test. In this test we perform following actions:

- Calculate SED between the observation and all the cluster centroids in the usage profile. This is done by using equation 5.5.
- Based on these values, the cluster with shortest SED is selected as possible cluster for this new observation.
- Lastly we test if the observation lies within the cluster boundary or not. If it does, it belongs to this cluster and we will consider this observation as genuine (from the same user) other wise it is anomalous. the equation 5.6.2.1 show this step.

$$CB - SED(O,C) \begin{cases} > 0, & \text{Genuine observation;} \\ <= 0, & \text{Anomalous observation.} \end{cases} \tag{5.7}$$

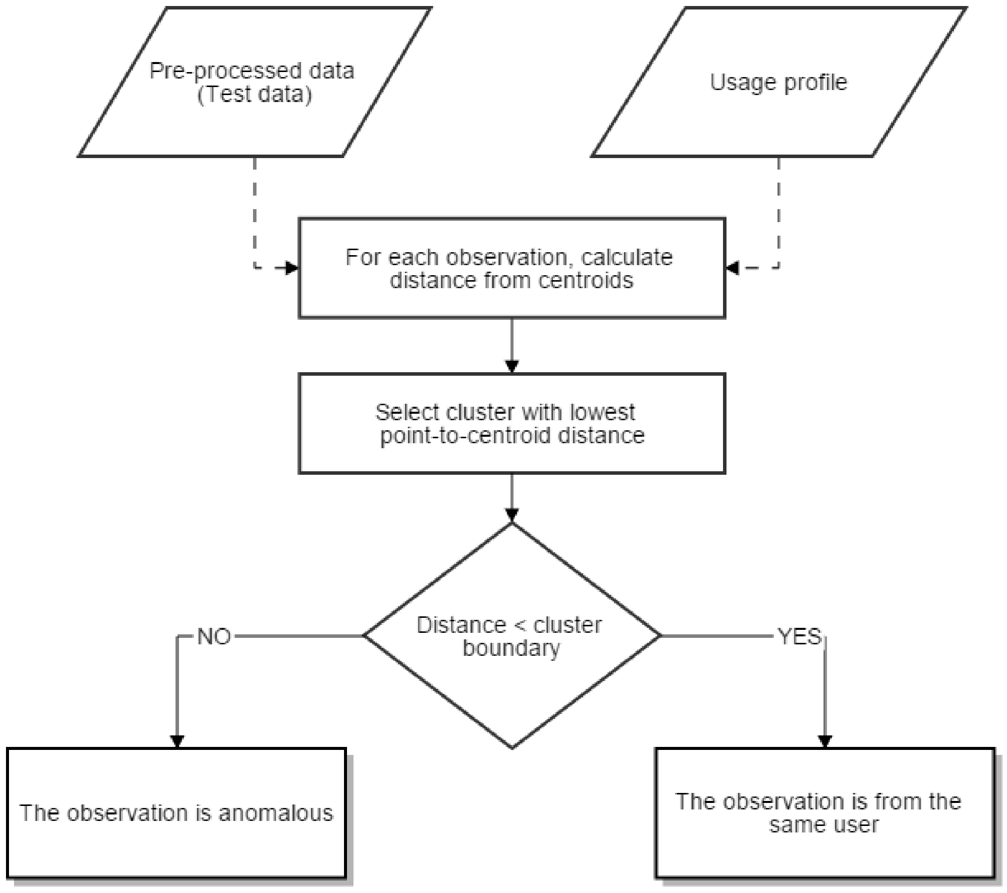

Pre-processed data (Test data)
Usage profile
For each observation, calculate distance from centroids
Select cluster with lowest point-to-centroid distance
Distance < cluster boundary
NO
YES
The observation is anomalous
The observation is from the same user

Fig. 5.8 The flow chart of anomaly detection

The pseudo code for the anomaly detection process is presented in the Listing 5.2.

Listing 5.2 Anomaly detector pseudo code

```
Read usage profile data from mat file.
Read test data from csv file.
Pre-process the test data.

for every data point in our test data
    calculate Squared Euclidean Distances from centroids.
    select the cluster with least point-to-centroid distance.

    if the point is inside the cluster boundary
        the data point is benign
    else
        the data point is anomalous
```

When the anomaly detector encounters a persistent deviation from the usage profile, it notifies the *Alter Manager* to take an appropriate action.

5.6.2.1 Profiling accuracy

To calculate accuracy of usage profile built during training phase we use compare it with the data collected in testing phase by applying the cluster boundary test shown in equation . This accuracy figure can also be called as a dissimilarity score because the more a dissimilar a usage profile is from the test data the better it is. A usage profile's dissimilarity score is percentage of how many observations from the test data as classed as anomalous according to the cluster boundary test.

$$Profiling\,accuracy = \frac{anomalous\,observations * 100}{total\,observations} \tag{5.8}$$

5.7 Alert manager

The role of alert manager is to notify the smartphone user about any anomalous activity on the mobile device. Depending upon the user's configuration various actions can be taken

1. A warning message shown in the smartphone.
2. Prompting the user to enter preconfigured pin-code to unlock the device.
3. Allowing user to update their profile.

Since the MLP has been implemented and tested in Matlab, this component has not been implemented on smartphone and left as future work.

5.8 Remote administrator

For the prototype implementation, it performs the following tasks:

1. Receive data (extracted features) from smartphones.
2. Perform MDS on data for visualisation.
3. Identify optimum values for parameters of K-Means clustering.
4. Generating and testing the usage profile.

Since usage profile generation is computationally expensive task, many low end device may not able to generate usage profile hence the remote administrator can be used as MLP.

5.9 Inter operability with other IDS

The proposed system is designed for two security purposes, either authentication (standalone operation discussed earlier) or as processing component in another IDS or authentication system e.g. TAS. In this mode, the Behaviour Profiling system only provides a verification result or profile matching score respectively and the final decision will be made by the other security mechanism.

As mentioned in section 3.2.6, TAS is an authentication system utilising a number of biometrics to provide transparent and continuous authentication for mobile users. This has been achieved by employing a two-tier approach: Tier 1 selects various biometric techniques and Tier 2 combines a number of multi-biometric methods together.

The Behaviour Profiling system can be used as one of the biometric techniques in Tier 1 by TAS as it employs the way users utilise mobile applications to verify them; also, the Behaviour Profiling system can provide unique contributions in operations of the Tier 2 to the TAS system. As the Behaviour Profiling system verifies mobile users based upon their application usage, TAS can utilise its verification output alone in pass-through mode to provide transparent and continuous authentication. Moreover, the Behaviour Profiling system can work with other biometric techniques to form a fusion mode in the TAS system. For instance, when a user sends a text message, their keystroke activities (i.e. how each character is typed into the message) and their behaviour profiling activities (i.e. where and who the message is sent) can be used in one fusion function. In this way, TAS can provide the authentication decision more confidently.

There are a number of IDS systems proposed to detect malware presence within the mobile device environment, such as the one Knowledge-based Temporal Abstraction based IDS proposed by (Shabtai et al., 2010). This host based IDS can accommodate

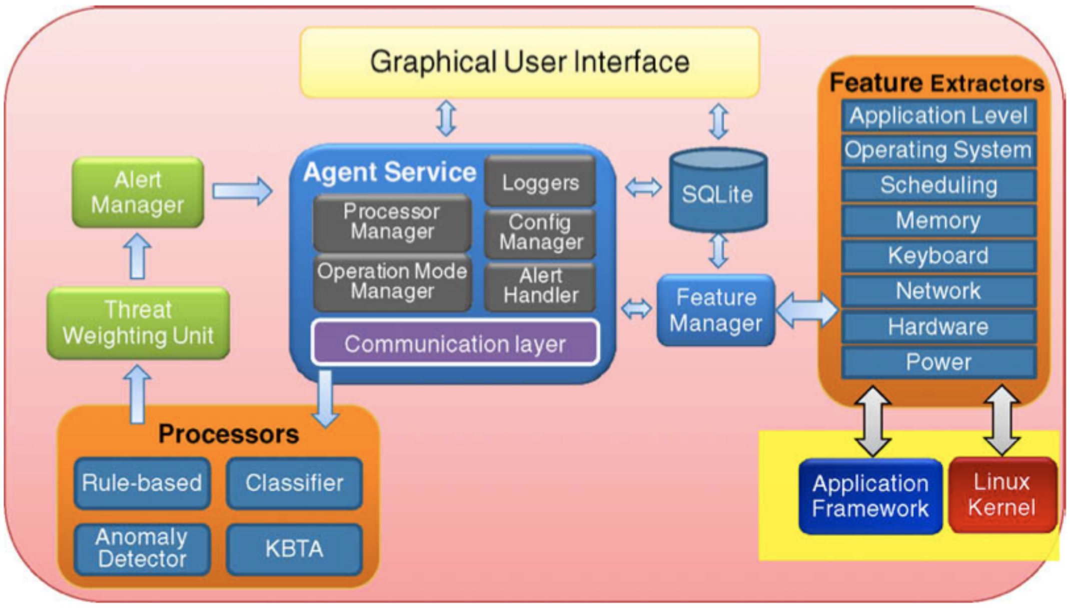


Fig. 5.9 Hybrid IDS for mobiles (Shabtai et al., 2010)

the aforementioned MLP of our HIDS (as depicted in Figure 5.9). Their evaluation indicates that their architecture works well for detecting mobile malware. Nonetheless, their system cannot detect any user related misuse. Therefore, the Behaviour Profiling system and the KBTA based IDS could work together to form a new host based IDS for mobile devices which can provide comprehensive detection of user misuse and also mobile malware.

As a result, an alert will be raised not only when a device is infected by malware but also when it is misused by a user. Also, more accurate alerts would be generated when an application is infected by malware. For instance, when the text messaging service is infected by malware, it may send messages to a premium number without the owner's knowledge. If any messages were sent to the premium number, the Behaviour Profiling system should detect the abnormal activity as it deviated from the user's normal behaviour. In addition, as the messages were generated by malware, the malware detection part of the IDS can also pick up the abnormal activity. Therefore, two alerts would be raised by the IDS system after an unauthorised text message was sent. This will improve the performance of the IDS system significantly.

5.10 Summary

In this chapter, a novel Behaviour Profiling based anomaly detection system which provides robust, transparent and continuous protection for mobile devices by verifying mobile usage activities has been designed and the components and functionalities of the system described in detail. By employing the dynamic profiling technique, the system can generate a fresh user's profile allowing more accurate verification results to be obtained. By utilising the scaling function, the system reduces the impact of the high false rejection problem which every single behavioural biometric technique experiences; hence, the performance of the system can be improved despite decision making taking longer to process. Also, a user's identity is not verified based upon a single pass or fail but a number of consecutive verification results. The HIDS can operate in different modes to serve a variety of purposes. When the system operates in stand alone mode, it verifies a user's identity and responds accordingly in isolation. When the system works in dependent mode, it provides verification results based upon user's activities, with the final decision being made by another security mechanism (either an authentication system or an IDS system). In the next chapter, the Behaviour Profiling system will be evaluated using data collected from 10 subjects using Matlab to simulate the tests.

Chapter 6

HIDS Evaluation

6.1 Introduction

This purpose of this chapter is to outline the evaluation process of our HIDS that proposed in the chapter 5 and to discuss the results of tests that were performed. To begin with this process, we developed an Android application for data collection to work as the Feature Extractor component of HIDS. The working details of this component were discussed in details in the section 5.5. This application was installed on 10 smartphones and the data collection duration was 15 days. After this time the users sent the usage data (SQlite database file) via email option in the application.

A simulation of MLP component (see section 5.6) of the HIDS was implemented as a Matlab program hosted on a PC. Once the data was received from all users, MLP was applied on it to train and test the accuracy of the system based on the criteria set out in section 5.6.2. This enabled a evaluation of the processes and mechanisms over a far wider range of variables than would have been possible with an smartphone application.

Due to privacy concerns from the users of our Android application, the dataset didn't include fine grained details like telephone numbers, names of applications, proximity devices etc. This was a partial limitation in obtaining a more comprehensive dataset. However, it offered the opportunity to directly compare the performance of the system with inconveniencing users. The HIDS was evaluated using a three tests. First, we performed a descriptive statistical approach to analyse the dataset to identify features that can distinguish users and then MLP was applied in two separate tests. The results of these tests confirmed the findings of the first test. The figure 6.1 shows the flow chart of the evaluation process of HIDS while more details about these tests and the discussion of results are provided in the following sections.

6.2 Dataset for evaluation

Since no existing dataset for mobile device usage statistics was available, we employed our own dataset to test the proposed framework which was collected from a prototype mobile application deployed on mobile devices. This application performs the actions of Feature Extractor component of the framework (see section 5.5). The time interval for each observation was set to 5 minutes, so that users' devices are not burdened with more frequent processing. The testers were asked to run the data collection application for minimum two weeks to allow enough observations to be collected. The testers were then asked to provide us their mobile device usage data after this time using the email function in our prototype application.

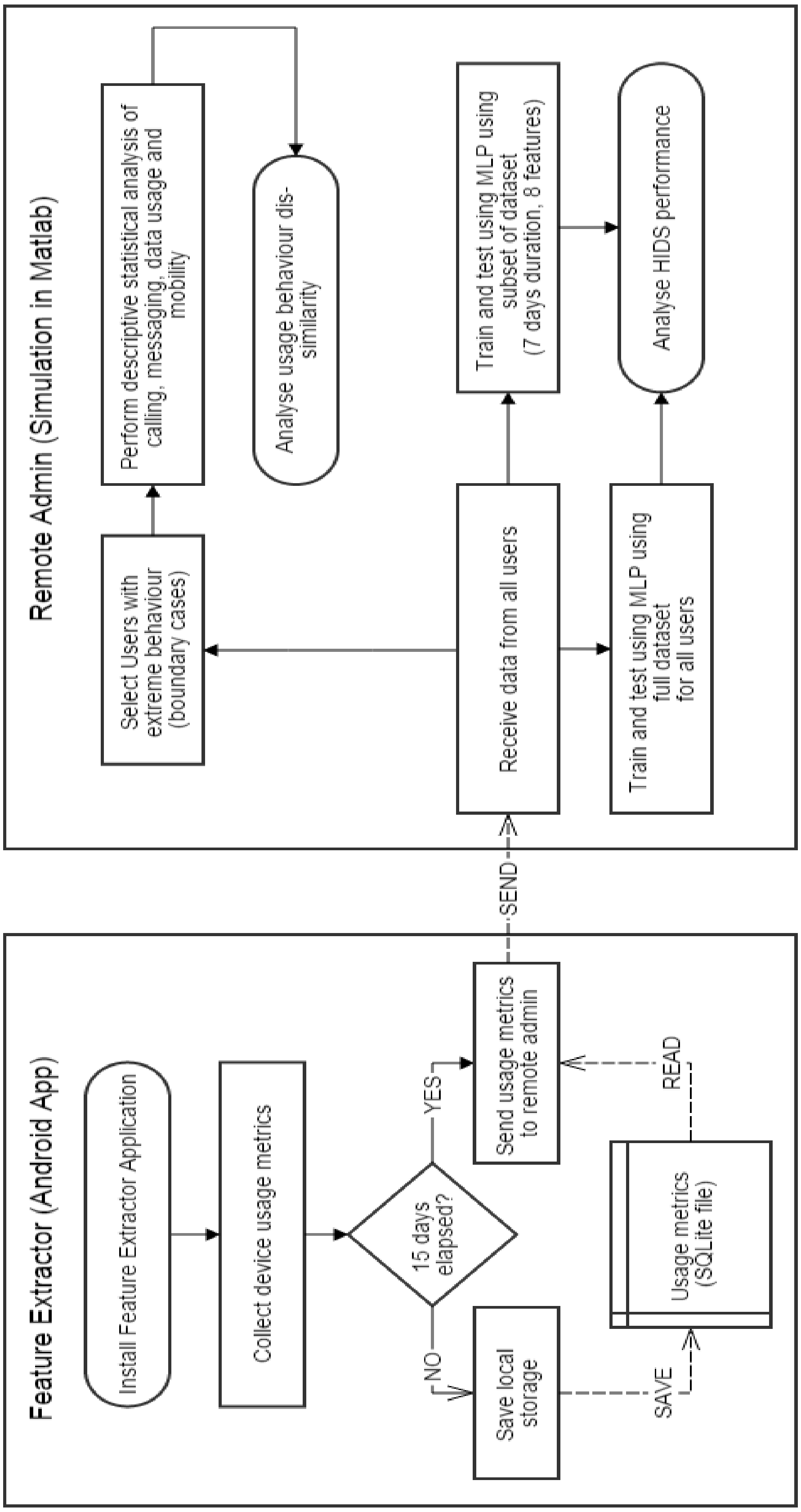

Feature Extractor (Android App)
Install Feature Extractor Application
Collect device usage metrics
15 days elapsed?
NO
YES
Save local storage
Send usage metrics to remote admin
SAVE
READ
Usage metrics (SQLite file)
SEND
Remote Admin (Simulation in Matlab)
Select Users with extreme behaviour (boundary cases)
Perform descriptive statistical analysis of calling, messaging, data usage and mobility
Analyse usage behaviour dis-similarity
Receive data from all users
Train and test using MLP using subset of dataset (7 days duration, 8 features)
Train and test using MLP using full dataset for all users
Analyse HIDS performance

Fig. 6.1 The flow chart of HIDS evaluation

The feature extraction application was deployed to more than 20 mobile device but only 14 testers submitted data to us. However, the data from 4 users was incomplete because either device was not used or there was no user activity. As a consequence we omitted those users from testing. So our dataset consists of 48,536 observations, containing 14 features from mobile devices of 10 different users who actively used their mobile device during the data collection phase. Statistics showed us that each users ran the prototype application for a different amount of time and Table 6.1 presents the list of different smartphones and number of observations made on each of these devices.

Table 6.1 Testers for data collection

User	**Device**	**Number of Observations**
1	Samsumg Galaxy S3	4872
2	Samsung Galaxy S4	5913
3	HTC One	5061
4	HTC One	4807
5	Google Nexus 4	4081
6	Sony Xperia E	4719
7	Motorola Moto G	6080
8	Samsumg Galaxy S3	4032
9	Samsumg Galaxy S2	4941
10	HTC Desire C	4030

The data is received in the form of a portable SQLite 3 database file which enabled us to query data more efficiently for analysis. This data is exported into a CSV file which can be read by our simulated anomaly detection processor in Matlab. The basic premise of behaviour based profiling is that different users utilise their mobile devices differently and this behaviour can be modelled to detect anomalies arising from misuse. Table 6.2 shows the summary of usage statistics for 5 features for each user contributing to the dataset that gives us some identification of different usage behaviour exhibited by different users.

Table 6.2 Summary of mobile device usage

User	Calls	Duration	SMS	Data Tx (MB)	Data Rx (MB)	Avg Distance
1	243	1279.2	15	40.07	285.89	7.067
2	117	308.37	121	435.13	5800.83	11.15
3	160	553.9	54	57.38	717.8	6.85
4	30	87.67	74	144.3	529.49	3.89
5	17	12.42	2	84.37	1239.58	4.15
6	123	179.08	18	365.07	6365.77	10.13
7	298	9290.53	56	39.13	741.84	8.11
8	31	27.7	11	59.99	980.42	6.19
9	97	359.52	87	609.18	9929.21	9.49
10	243	1279.2	15	40.07	285.89	3.95
Total	1359	13377.59	453	1874.69	26876.72	7.0977

In the following sections we present the results of our tests to determine the accuracy of distinguishing mobile device users based on their usage patterns.

6.3 Descriptive statistical analysis

We performed a preliminary study by utilising a descriptive statistical approach to analyse the raw information presented by the dataset. It is well-known that the statistical method has the ability to determine potential positive application features for forming unique patterns to discriminate individual users (Jain et al., 2000). Also, the selection of effective features is a critical process in pattern classification as the system performance is closely related to the features and large number of features will increase the complexity and the size of the classifier. For this study, we selected one week of usage data (Monday to Sunday) from the dataset for each user. To keep the analysis manageable we e selected 5 users from the dataset and grouped the data per hour resulting into 168 observations per user. We also selected only a subset of features for comparison. These features include the following.

- Number of calls

- Duration of calls
- Number of SMS sent
- Data traffic (sent + received)
- Average CPU load
- Average RAM load
- Number of opened applications
- Distance from Charing Cross

6.3.1 Calling behaviour

Previous research has shown that mobile users have different patterns of making calls. To test this hypothesis, we compared call records of users 1,3,4,8 and 9. The results of this comparison suggest that the users in our dataset exhibit different calling behaviour.

6.3.1.1 Comparison of user 1 and 8

User 8 made a total of 12 calls during one week period which amounts to a mere 13.4% of the 90 calls made by user 1 during the same period of time. The average number of calls made by User 8 was around 1.7 whereas, User 1 made a much higher, 12.85 calls per day on average. In terms of favourable time for calls, User 1 made more than 57% of calls during the weekend compared to that of 25% for user 8.

Table 6.3 Calling behaviour of user 1 and 8

Day	User 1 Calls	User 1 Talk Time	User 8 Calls	User 8 Talk Time
MON	11	6.72%	1	6.93%
TUE	6	0.75%	1	15.55%
WED	6	1.10%	2	32.56%
THU	7	1.98%	4	22.90%
FRI	18	2.47%	1	0.84%
SAT	24	46.07%	3	21.22%
SUN	18	40.91%	0	0%

In regards to duration of calls, User 1 was again more active with nearly 17 hours of talk time compared to that of around 8 minutes during the same period of time. In addition to greater number of calls during the weekend, User 1 also spend 87% of the talk time during the weekend. In comparison user 8 only spent 21.22% of the talk time during the weekend.

Another interesting distinction between the two users is that, User 1 made a maximum of 10 calls in one hour compared to a maximum of 2 calls in an hour for

User 8. These statistics clearly indicate that in terms of calling behaviour these users are distinct from each other and hence their profile should also show this dissimilarity.

6.3.1.2 Comparison of user 3 and 4

User 4 only made 14 calls during the one week period which amounts to only 20% of the total 69 calls made by User 3 during the same period. The average number of calls made by User 3 was 9.8 calls per day compared to that of just 2 calls for User 4 who never made more than 2 calls in one hour whereas User 3 made 6 calls in one hour during the same day. In terms of duration of calls, user 3 spent 4.6 hours talking

Table 6.4 Calling behaviour of user 3 and 4

Day	User 3 Calls	User 3 Talk Time	User 4 Calls	User 4 Talk Time
MON	6	8.30%	0	0%
TUE	12	14.05	0	0%
WED	15	30.16%	4	65.27%
THU	12	27.67%	3	15.01%
FRI	9	5.18%	3	6.70%
SAT	11	3.41%	3	12.39%
SUN	4	11.23%	1	0.64%

compared to that of 44 minutes for user 4. An interesting fact for these two users is that both spend their maximum calling time during Wednesday that amounts to 30.16% for user 3 compared to that of 65.27% for user 4. The Summary of their calling behaviour over one week period is shown in Table 6.4.

6.3.1.3 Comparison of user 8 and 9

User 8 only made 12 calls during the one week period which amounts to only 37% of the total 32 calls made by User 9 during the same period. The average number of calls made by User 8 was just 1.7 calls per day compared to that of 4.5 calls for User 9 who never made more than 5 calls in one hour whereas User 8 made 1 call in one hour

Table 6.5 Calling behaviour of user 8 and 9

Day	User 8 Calls	User 8 Talk Time	User 9 Calls	User 9 Talk Time
MON	1	6.93%	2	0.02%
TUE	1	15.55%	10	31.48%
WED	2	32.56%	4	14.89%
THU	4	22.90%	10	45.17%
FRI	1	0.84%	4	5.39%
SAT	3	21.22%	0	0%
SUN	0	0.00%	2	3.06%

during the same day. In terms of calling time, User 8 only made 7.94 minutes of calls that amounts to only 5% of 2.5 hours talk time consumed by user 9 who used 45% of this time on Thursday compared to that 22.9% for user 8 on the same day. User 8 spent 21% of call time during the week end compared to just 3.06% for user 9. It should also be noted that user 9 always made more calls each day compared to user 9 except Saturday.The Summary of their calling behaviour over one week period is shown in Table 6.5.

The calling behaviour comparison of these users is provided in graphical form in the Figures 6.2 to 6.6.

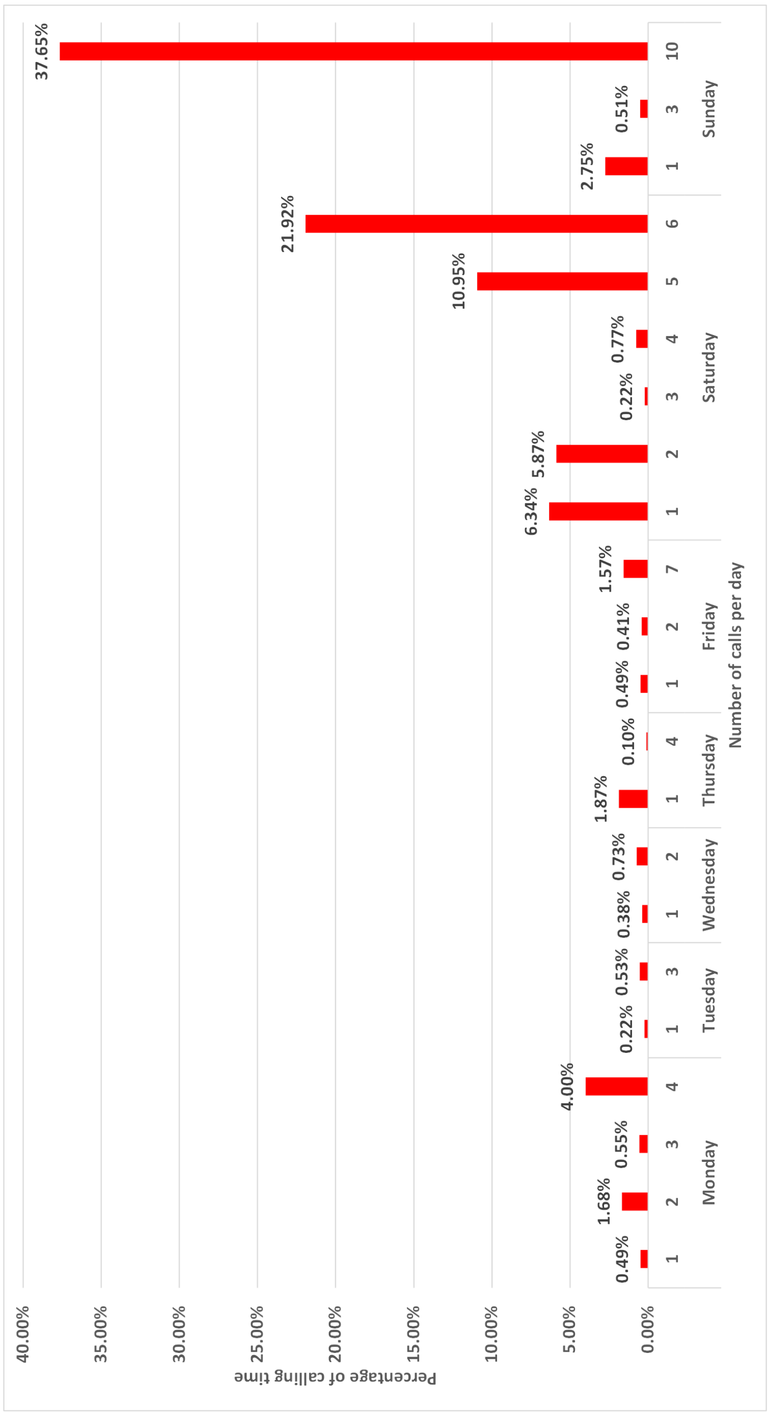


Fig. 6.2 Calling behaviour of user 1

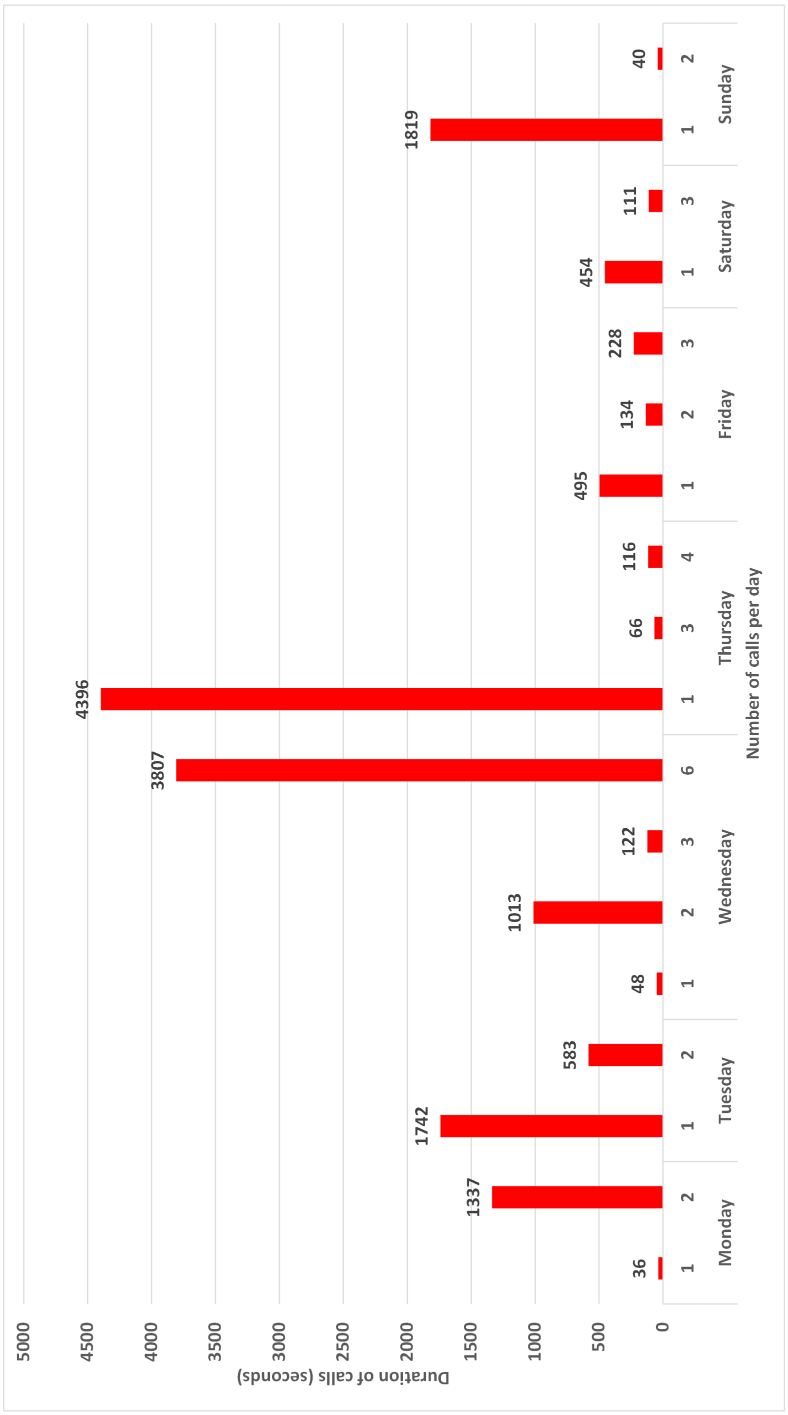


Fig. 6.3 Calling behaviour of user 3

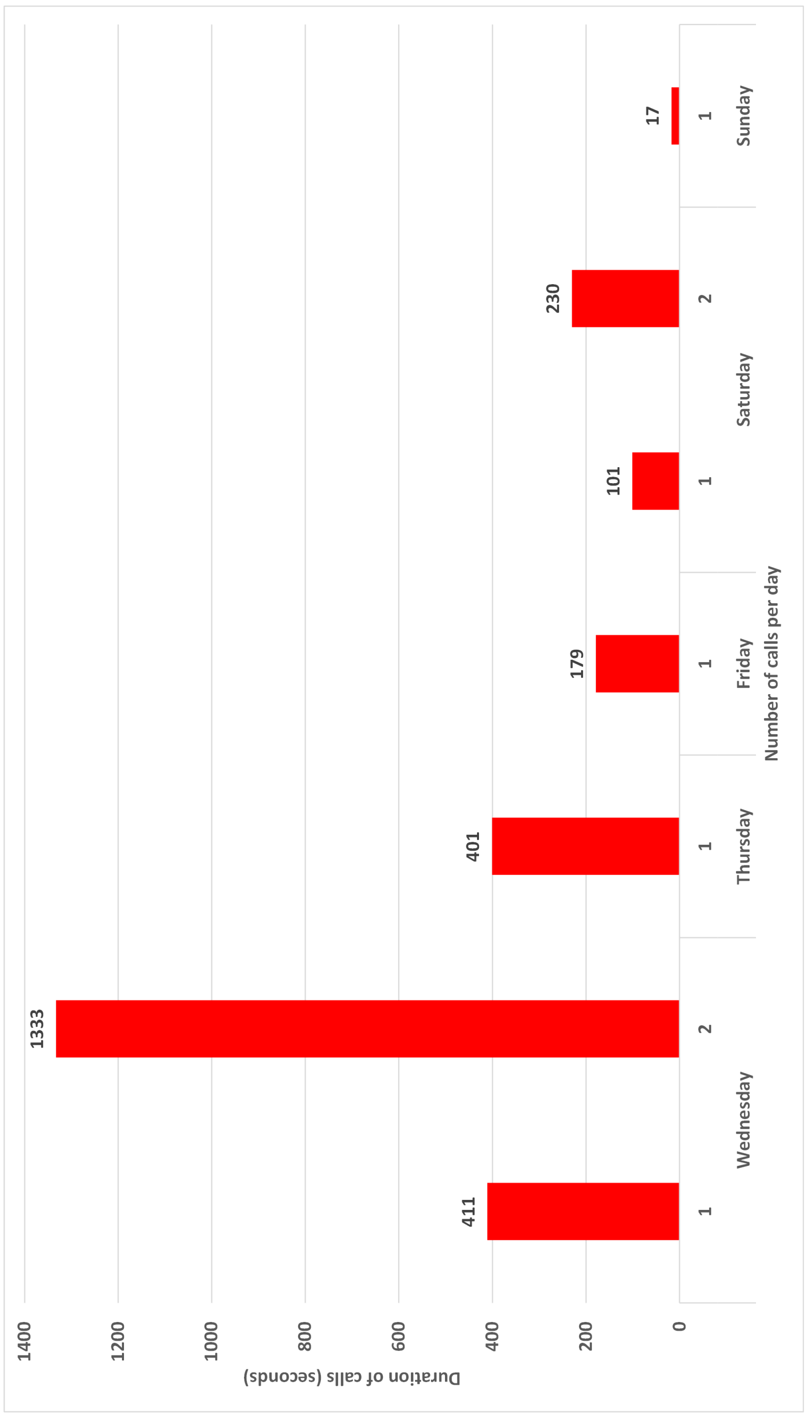


Fig. 6.4 Calling behaviour of user 4

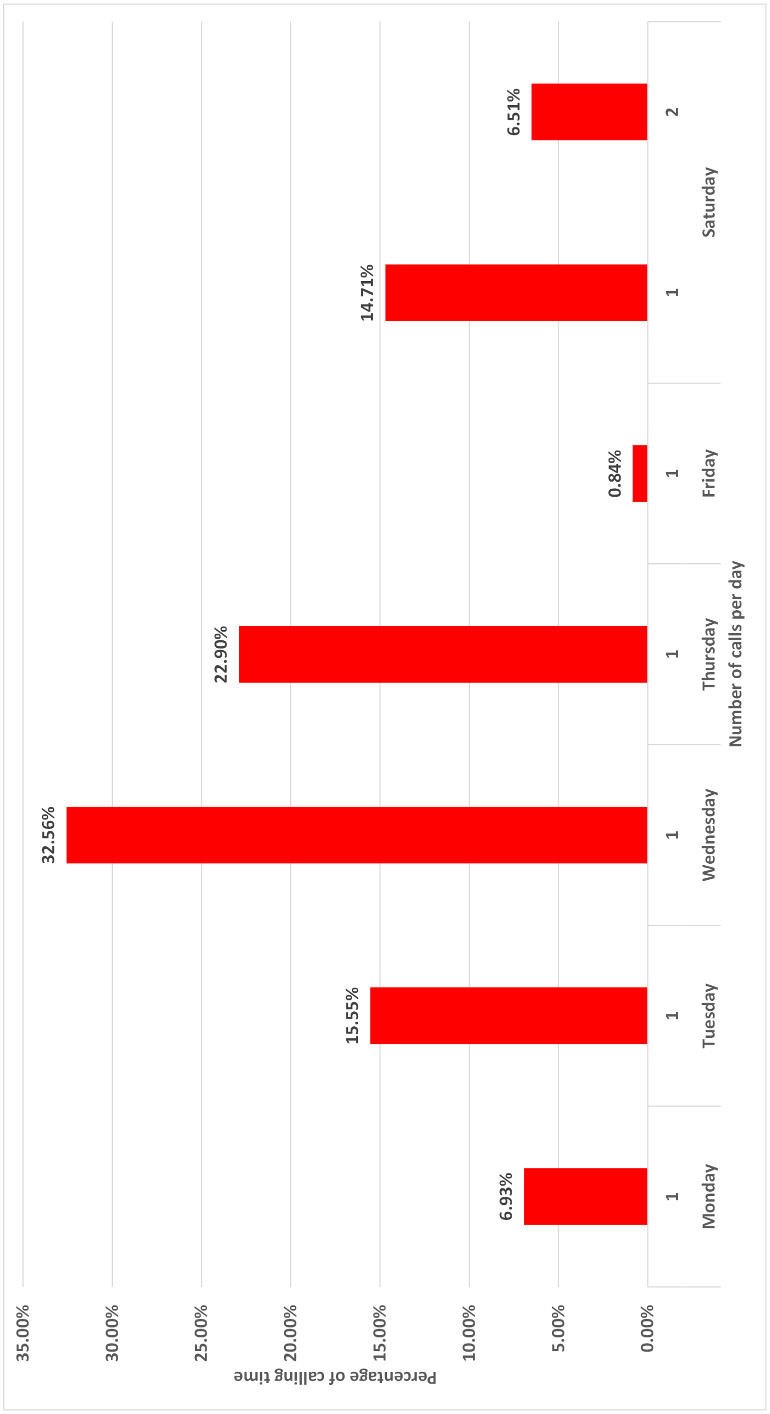


Fig. 6.5 Calling behaviour of user 8

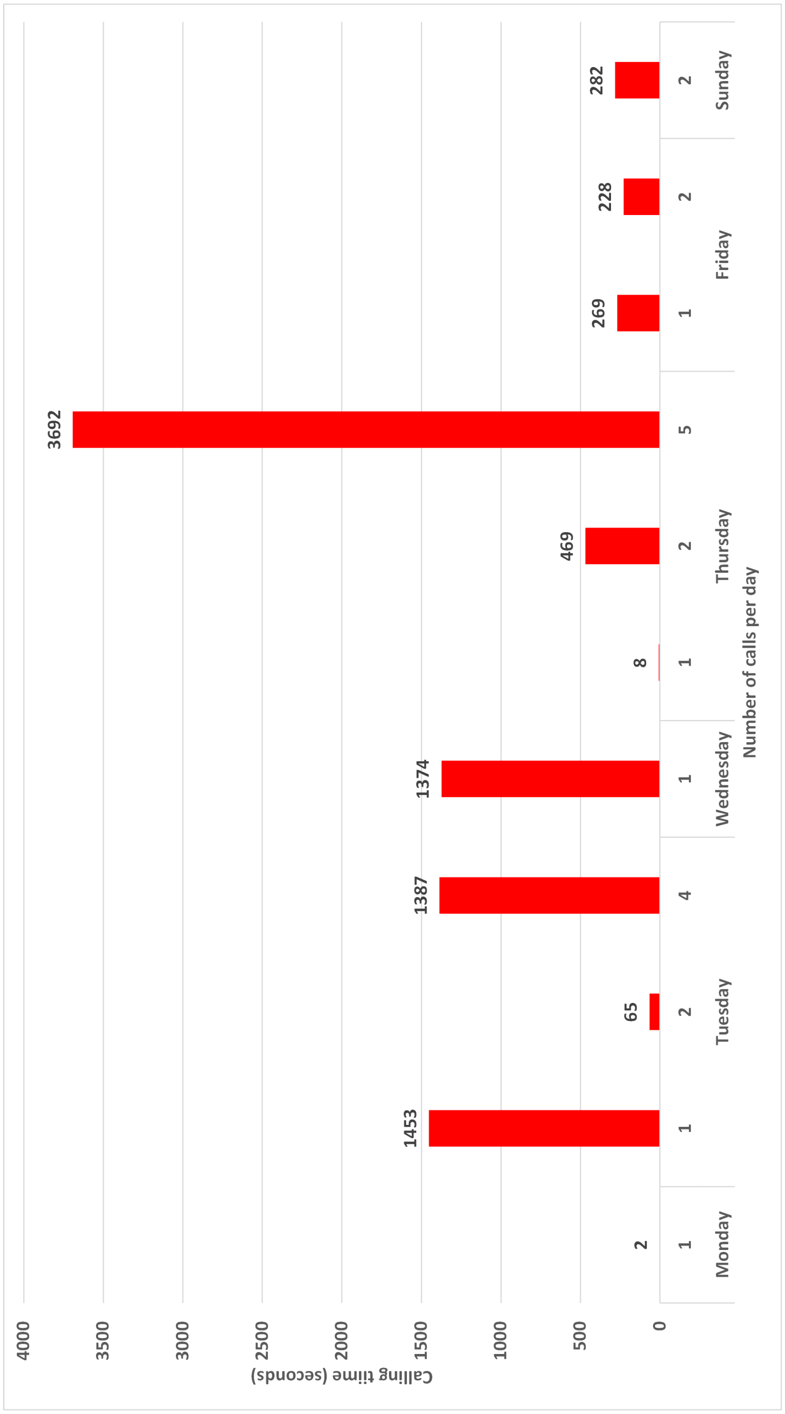


Fig. 6.6 Calling behaviour of user 9

6.3.2 Messaging behaviour

6.3.2.1 Comparison of user 1 and 8

Both users sent almost same number of SMS messages during one week, 4 for user 1 compared to that of 6 for user 8. User 1 didn't sue messaging on Monday, Wednesday, Thursday and Sunday whereas User didn't send any SMS messages during Tuesday, Wednesday, Friday and Sunday. The maximum number of messages sent by user 1 in a day was 4 messages compared to that of 2 messages for user 8.

Since both users didn't make much use of SMS messaging service it can't be considered as discriminating feature for them. One possible reason for this behaviour might be that these users utilised instant messaging application for communication which were not covered by our prototype application and hence SMS messaging can't conclusively distinguish these two users.

Table 6.6 SMS usage behaviour for user 1 and 8

User	MON	TUE	WED	THU	FRI	SAT	SUN
1	0	1	0	0	2	1	0
8	1	0	0	4	0	1	0

6.3.2.2 Comparison of user 3 and 4

User 3 sent 22 SMS messages during one week period which amount to nearly 50% of 38 messages sent by user 4. On average, user 3 sent three messages per day compared to that of 5.4 messages for user 4. Not only user 4 used SMS messaging service more frequently, he/she sent a maximum of 19 messages on Tuesday compared to that of 3 messages for user 3. As shown in the Table 6.7, it is evident that both user have different messaging behaviour, with user 4 being the more active of the two. So it can be assumed that their SMS messaging behaviour can be used as feature for profiling their mobile usage.

Table 6.7 SMS usage behaviour for user 3 and 4

User	MON	TUE	WED	THU	FRI	SAT	SUN
3	2	3	2	3	5	4	3
4	4	19	4	2	6	2	1

6.3.2.3 Comparison of user 8 and 9

User 8 is clearly less active in using SMS messaging service as opposed to user 9. The former only sent 6 messages during the whole week which amount to only 13% of 44 SMS messages sent by the later user in the same time period. User 9 preferred weekdays for messaging during weekdays as it sent more than 95% messages during weekdays. On average, user 8 sent less than a message a day compared to 6 messages per day for user 8. One of the possible reasons behind very low number of SMS

Table 6.8 SMS usage behaviour for user 8 and 9

User	MON	TUE	WED	THU	FRI	SAT	SUN
8	1	0	0	4	0	1	0
9	3	3	22	12	2	1	1

messages by user 8 can be that he is utilising the modern internet based instant messaging service to communicate with his contacts whereas User 9 relies more on the traditional SMS messages. This in itself is a discriminatory feature for these two users hence the results of the comparing their profiles should indicate this behaviour.

The SMS sending behaviour comparison of these users is provided in graphical form in the Figures 6.7 to 6.9.

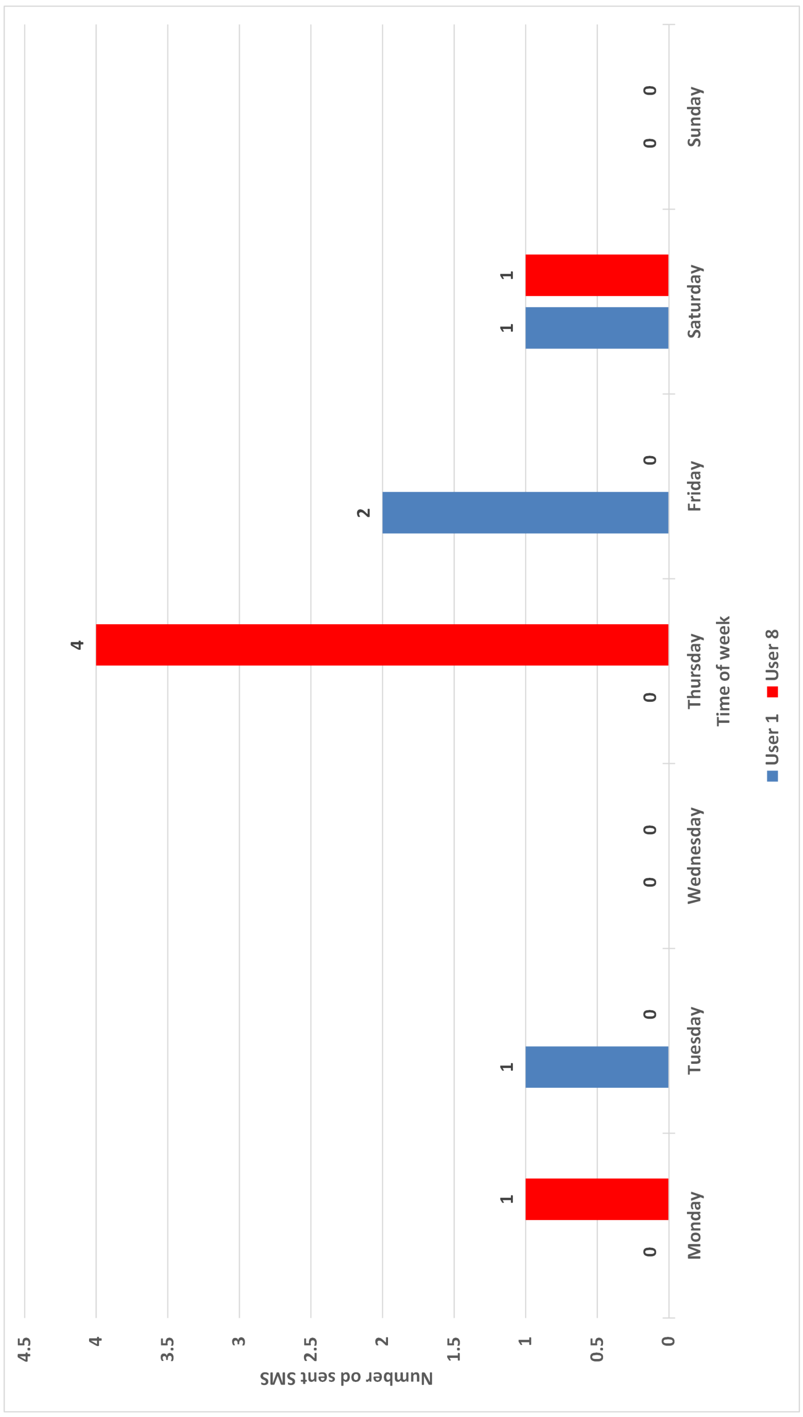


Fig. 6.7 SMS sending behaviour of user 1 and 8

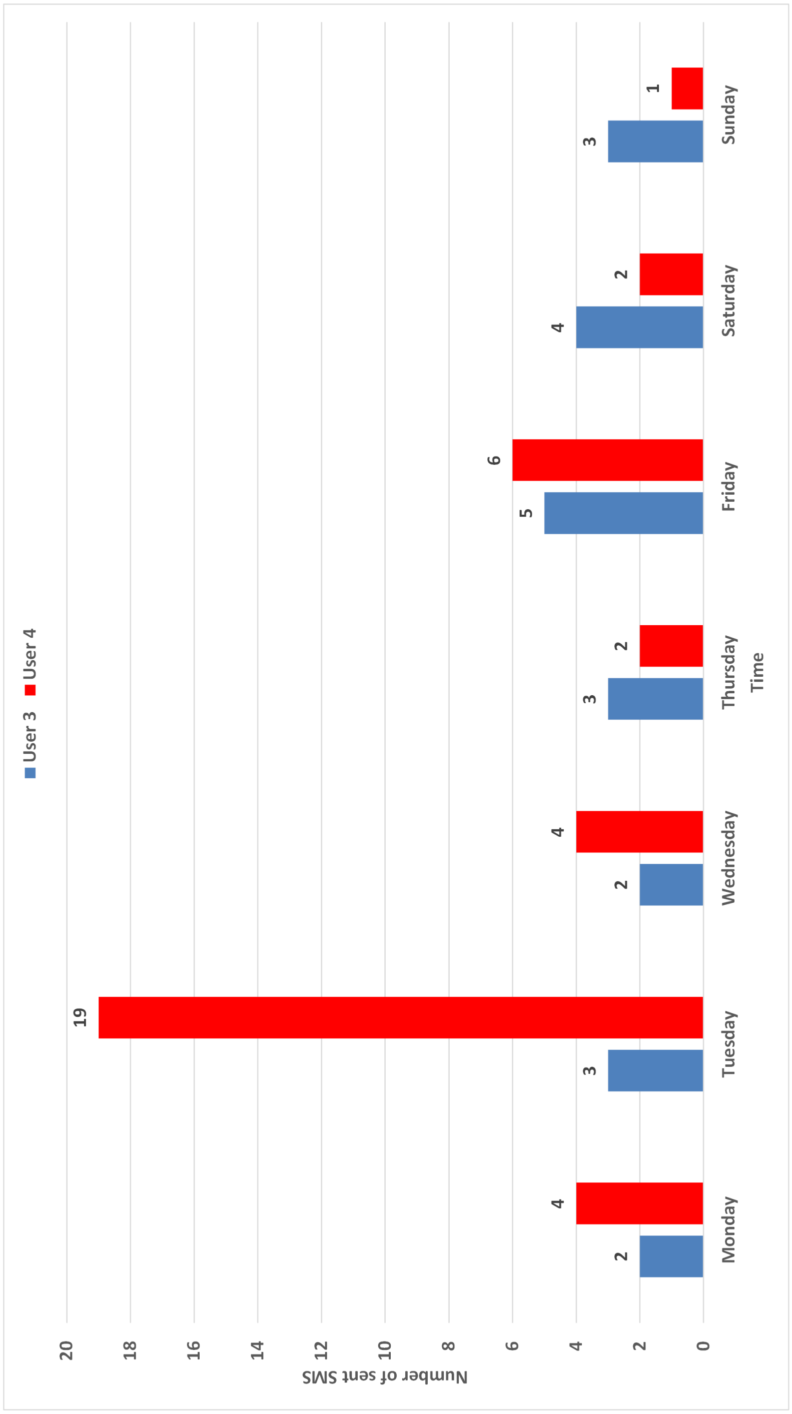


Fig. 6.8 SMS sending behaviour of user 3 and 4

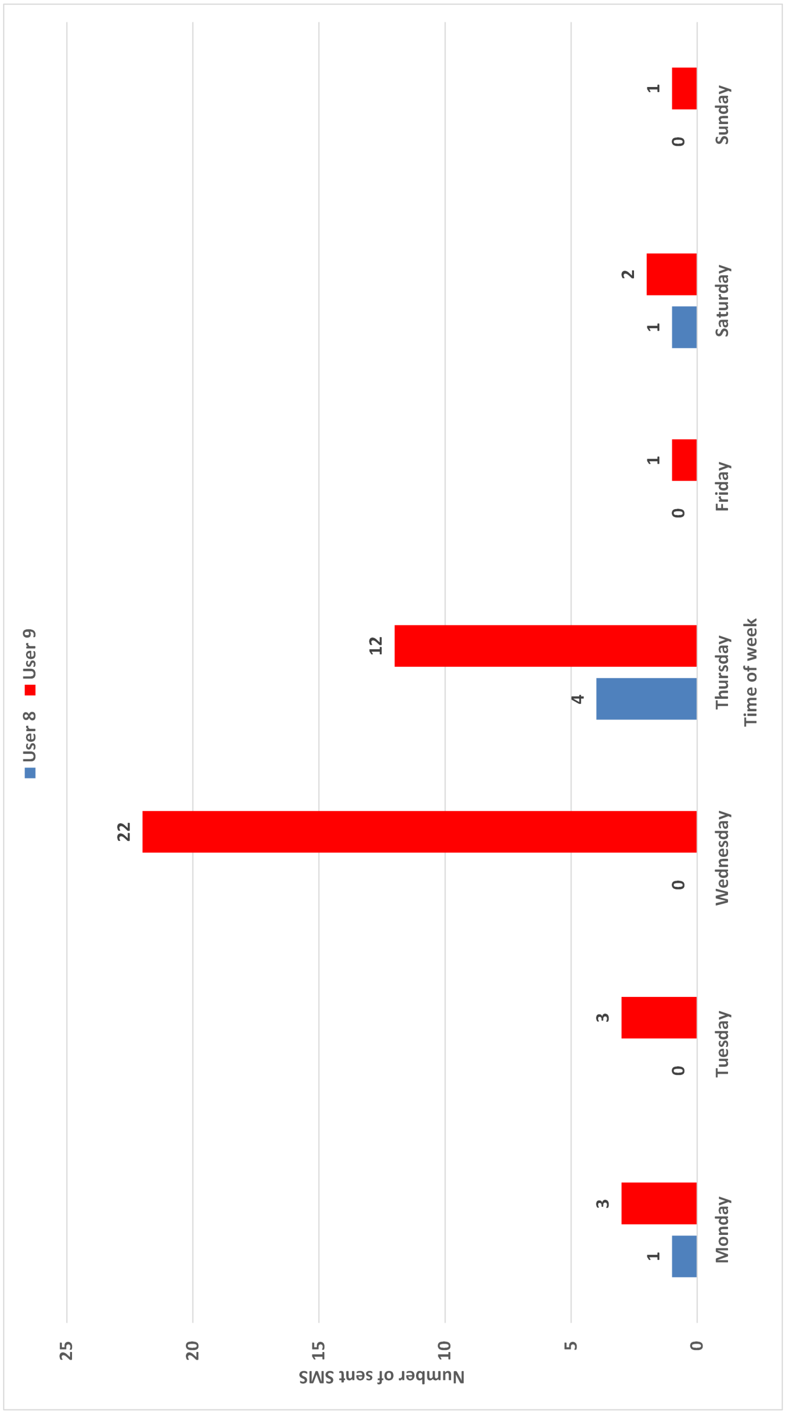


Fig. 6.9 SMS sending behaviour of user 8 and 9

6.3.3 Data traffic usage behaviour

This sections analysis the usage statistics for data services for user 1,3,4,8,9 and describes the similarities and differences to access the suitability of this feature to represent usage profile.

6.3.3.1 Comparison of user 1 and 8

User 1 consumed only 52 MB of data throughout the week which amounts to only 12.55% of 414 MB consumed by user 8 whose average was nearly 60 MB per day. User consumed 80% of his/her data in just one day (Sunday) which may be a one off event as it was not preceded by any similar usage. Rest of the week, user 1 never exceeded 5 MB data usage which is only possible with occasional web surfing or through background system applications.

Conversely, user 8 can be classed as high data user which probably stems from watching movies or videos online as he consumed 232.25 and 123.67 on Tuesday and Sunday respectively. The data usage figures indicate that this user consumes more than occasional web surfing and data usage is a consistent part of his/her daily usage.

Another interesting distinction between the two users is that user 1 only used a maximum of 23 MB per hour compared to that 90MB for user 8. Based on these statistics, It can be inferred that usage profiling based on data usage will yield significant dissimilarity between these users.

6.3.3.2 Comparison of user 3 and 4

User 4 consumed 223 MB of data during one week which amounts to 79% of data consumed by user 3 during the similar period. With a daily average 31.85 and 40.42 respectively data consumption is not much different from each other and significantly higher than an occasional web surfer. This indicates that data consumption is a consistent feature which both uses exhibit.

Table 6.9 Data usage (MB) behaviour for user 1 and 8

Day	User 1	User 8
MON	2.42	20.3
TUE	0.98	232.25
WED	2.09	8.72
THU	0	13.97
FRI	4.96	7.2
SAT	0	8.57
SUN	41.72	123.67
Total	52	414
Average	7.42	59.14

Interestingly, by comparing mobility behaviour for user 3 and the time of data consumption it can be clearly seen that user he/she only consumes significant amount of data while on the move which may indicate usage of application like navigation, watching videos or surfing the internet etc. Conversely user 4 utilises data services throughout the day. Based on the timing and mobility aspects it is envisaged that both users have a distinct data usage patterns hence their profiles should be dissimilar to each other.

Table 6.10 Data usage (MB) behaviour for user 3 and 4

User	3	4
MON	35.63	9.6
TUE	25.5	28.27
WED	16.46	25.3
THU	70.98	42.58
FRI	49.61	28.77
SAT	49.73	44.58
SUN	35.62	44.24
Total	283	223
Average	40.42	31.85

6.3.3.3 Comparison of user 8 and 9

In terms of volume of data usage, user 8 only 414 MB of data which accounts for a mere 11% of 3.6 GB consumed by user 9. This level of data usage indicates that user 9 is extremely high data user, watches on-line videos or engages in long videos calls on daily basis which means he/she is extremely active on the phone throughout the week. Although user 8 has an average data consumption of 59 MB per day, but this is only because of excessive data usage on Tuesday and Sunday.

Table 6.11 Data usage (MB) behaviour for user 8 and 9

User	8	9
MON	20.3	1471.4
TUE	232.3	73.15
WED	8.72	1157.2
THU	13.97	489.66
FRI	7.2	60.32
SAT	8.57	7.74
SUN	123.7	447.16
Total	414	3706
Average	59.14	529.42

The volume and usage time of data traffic indicates that these users distinct usage patterns, so it is envisaged that this feature will positively distinguish the tow users and they will have distinct profiles.

The data traffic consumption behaviour comparison of these users is provided in graphical form in the Figures 6.10 to 6.12.

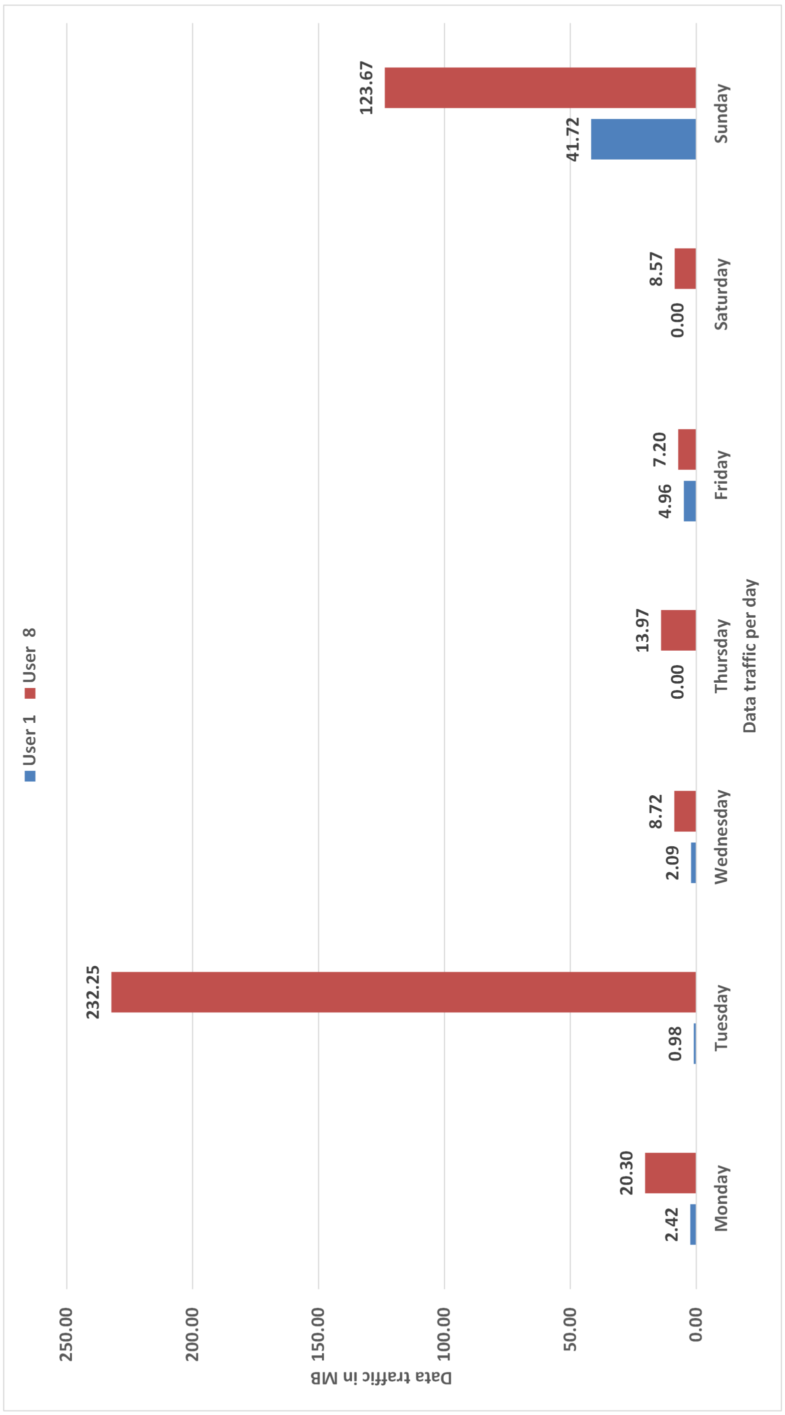


Fig. 6.10 Data traffic consumption for user 1 and 8

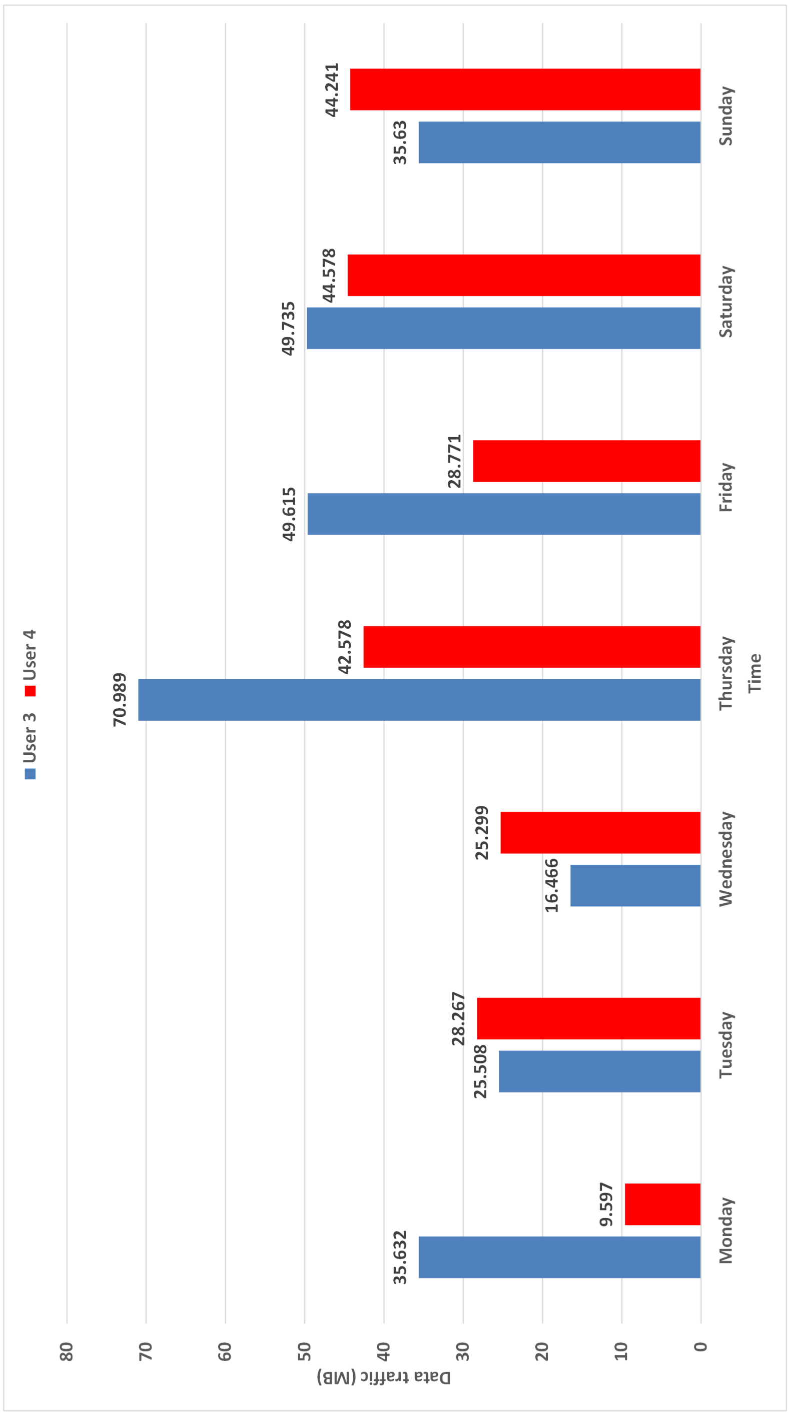


Fig. 6.11 Data traffic consumption for user 3 and 4

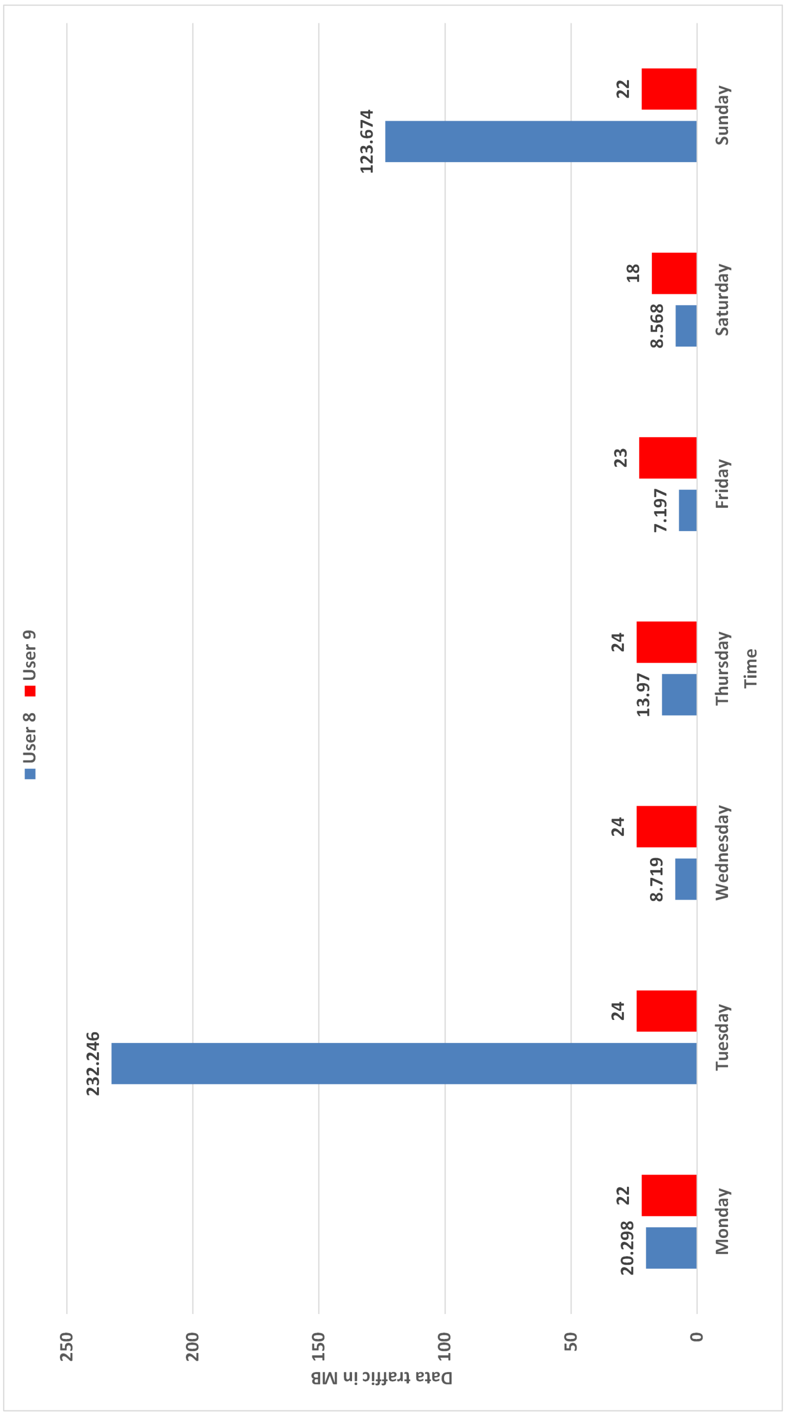


Fig. 6.12 Data traffic consumption for user 8 and 9

6.3.4 Application usage and system load

6.3.4.1 Comparison of user 1 and 8

On average user 1 only utilised 20 applications per hour which amounts to only 62% of that for user 8 who had more memory intensive applications compared. Conversely user 1 choose CPU hungry applications on regular basis. Daily statistics indicate that user 8 had a consistent usage throughout the week whereby user 1 became more active during the weekend. Observing these figures in the context of data consumption indicates that since user 8 consistently used data services like internet surfing and watching videos hence the memory usage is much higher compared to user 1.

Table 6.12 Application usage and system load for user 1 and 8

User	Feature	MON	TUE	WED	THU	FRI	SAT	SUN	Average
1	Apps	20	20	20	19	19	19	25	20
1	CPU	41.34	43.59	42.74	43.46	37.62	38.69	41.49	41.28
1	RAM	75.19	80.34	80.26	79.44	77.54	79.21	76.99	78.42
8	Apps	33	33	32	33	31	32	32	32
8	CPU	36.8	34.39	36.16	35.85	36.9	37.23	36.97	36.33
8	RAM	85.79	87.17	87.4	88.64	88.19	88.96	89.13	87.9

These statistics only indicate that user 1 utilised CPU intensive application compared to that memory intensive applications and both had their own uniquely consistent patterns which may make their profiles unique.

6.3.4.2 Comparison of user 3 and 4

Clearly, user 3 only utilised a small set of application during the week that consisted of 10 applications per day compared to 35 for user 4 who also had higher CPU and memory usage. On closer analysis User 4 had higher CPU and Memory usage during early mornings and evening hours which was this user's mobility time. This may indicate activities like playing games, watching videos or reading eBooks as most people do such actions while commuting on train. Conversely user 3 had a very consistent pattern of lower system load which was unaffected by mobility indicating that this user is not an active mobile device user. This differentiation indicates system load and application usage can positively discriminate the usage profile for these users.

Table 6.13 Application usage and system load for user 3 and 4

User	Feature	MON	TUE	WED	THU	FRI	SAT	SUN	Average
3	Apps	10	10	10	10	10	9	10	10
3	CPU	4.22	7.04	8.29	11.39	6.04	4.35	4.68	6.57
3	RAM	35.14	39.87	39.96	39.82	39.63	40.82	39.99	39.32
4	Apps	35	34	35	36	35	36	36	35
4	CPU	12.63	13.76	17.88	13.62	19.95	16.6	29.14	17.65
4	RAM	50.3	52.18	54.35	55.48	60.4	60.49	62.23	56.49

Table 6.14 Application usage and system load for user 8 and 9

User	Feature	MON	TUE	WED	THU	FRI	SAT	SUN	Average
8	Apps	33	33	32	33	31	32	32	32
8	CPU	36.8	34.39	36.16	35.85	36.9	37.23	36.97	36.33
8	RAM	85.79	87.17	87.4	88.64	88.19	88.96	89.13	87.9
9	Apps	41	41	36	31	32	29	42	36
9	CPU	44.63	34.42	37.6	37.31	33.55	30.92	34.73	36.29
9	RAM	86.86	79.7	81.06	75.74	73.19	53.79	84	76.98

6.3.4.3 Comparison of user 8 and 9

Both users had similar application usage pattern over the period of one week. The main differentiation was shown in terms of memory consumption by the applications they chose to utilise during this time. User 9 had slightly more memory intensive applications in use, which when analysed with his/her data usage pattern indicates that this high memory usage stems from the data intensive applications such as online videos players etc. Also user 9 showed a significant drop of application usage and consequently lower system load on Saturday, this information on its own doesn't reveal the cause but mobility profile indicates that this user was travelling during the same time hence we can infer that this user didn't have access to Wi-Fi which contributed towards lack of activity. Since both users have consistently similar usage pattern, these features may not be well suited to contribute towards building a usage profile hence the results of profiling should indicate very similar profiles for these users.

The Application usage and system load comparison of these users is provided in graphical form in the Figures 6.13 to 6.17.

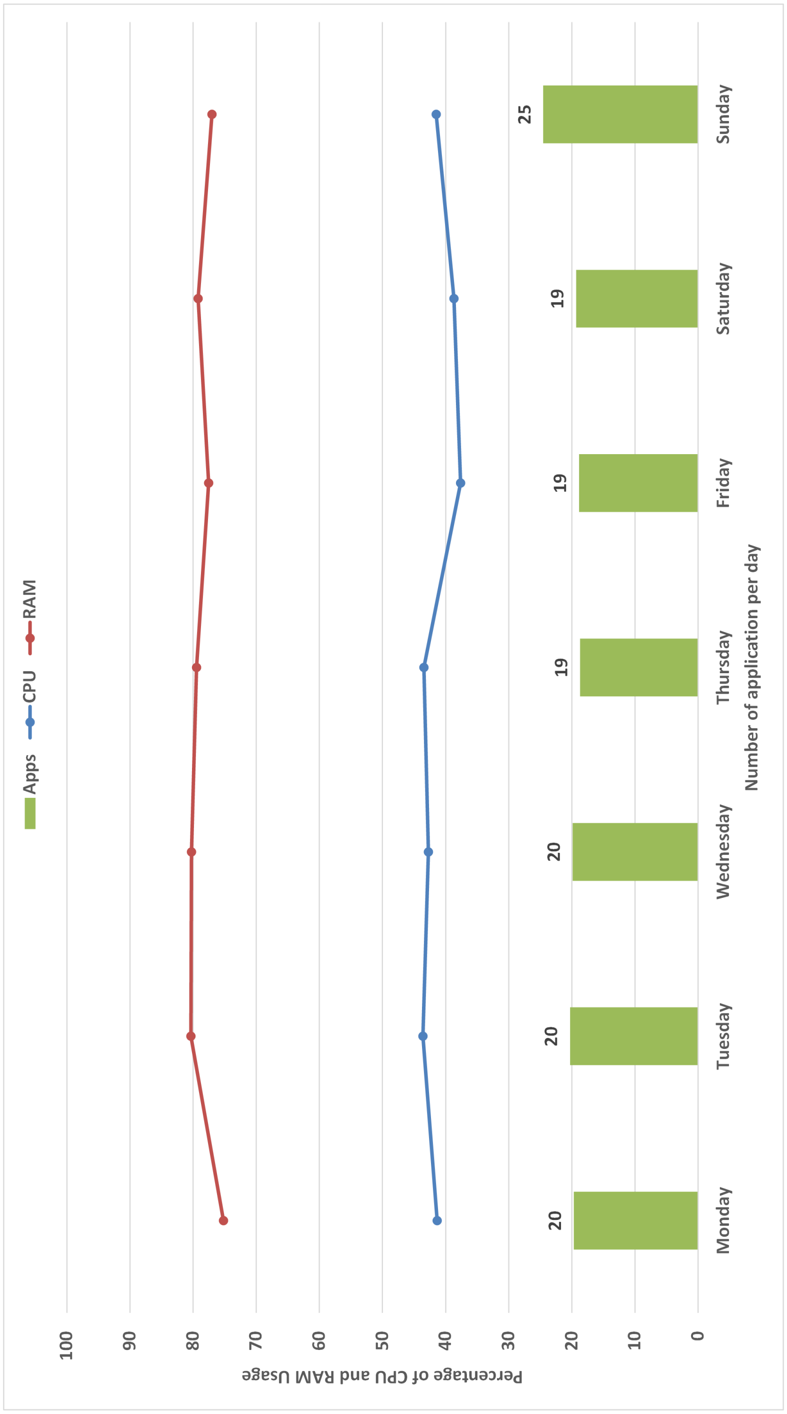


Fig. 6.13 System load for user 1

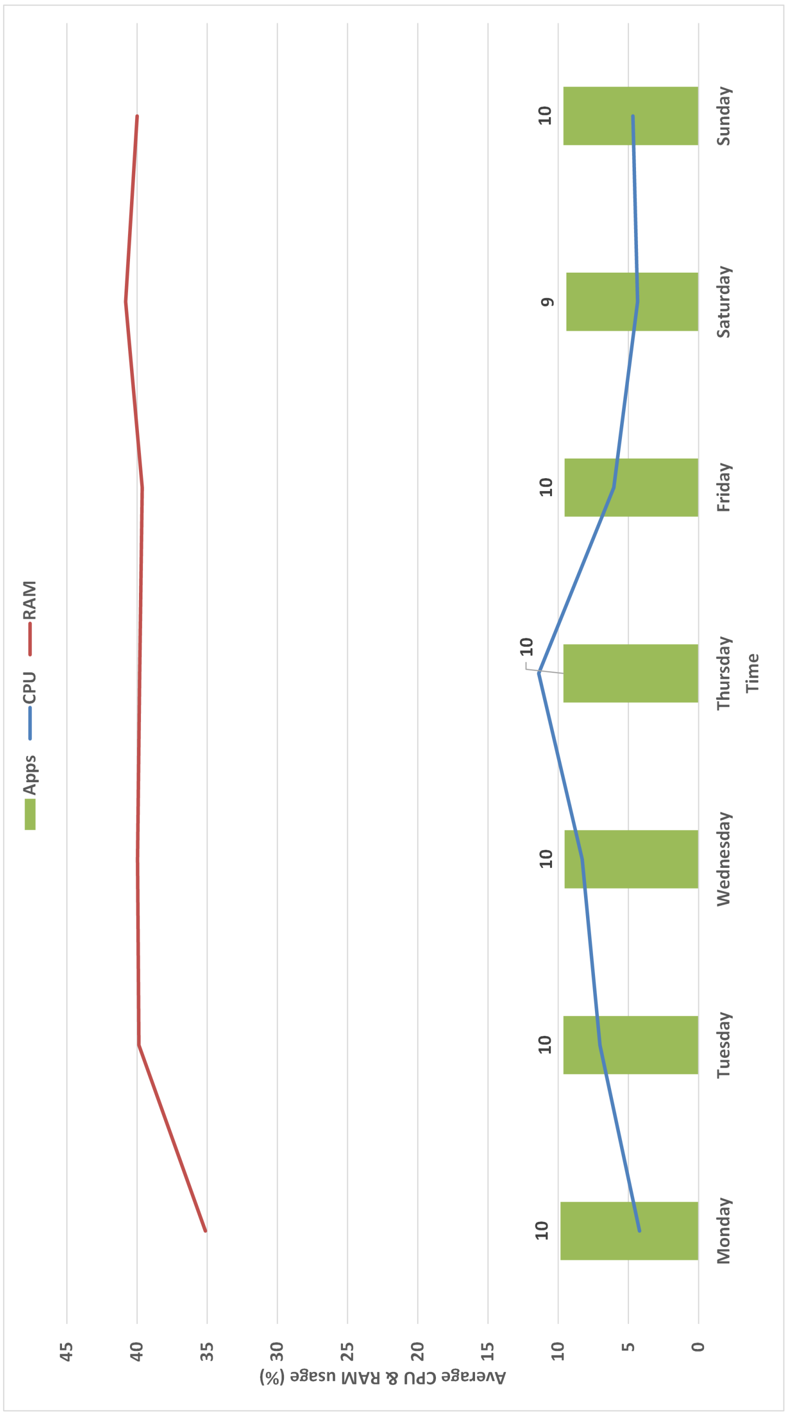


Fig. 6.14 System load for user 3

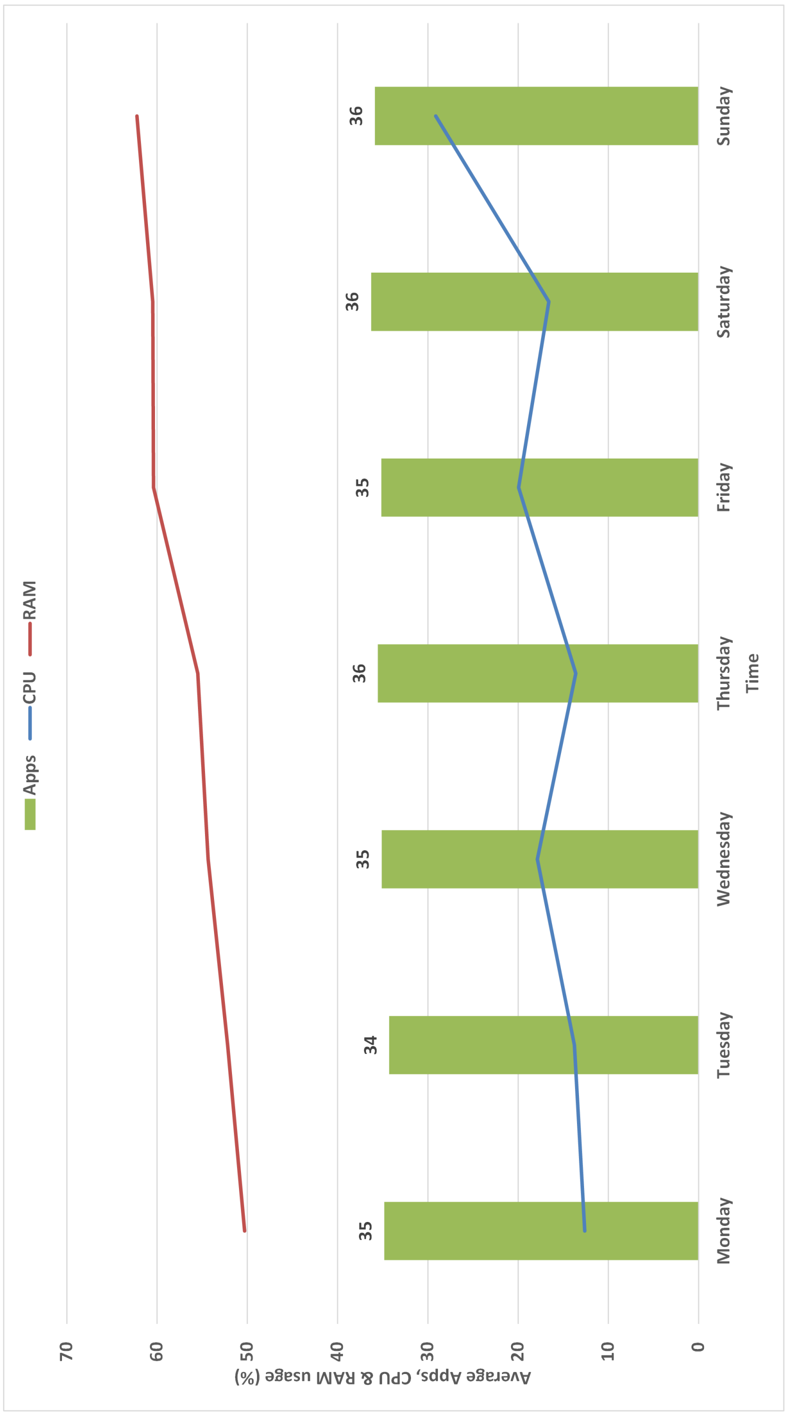


Fig. 6.15 System load for user 4

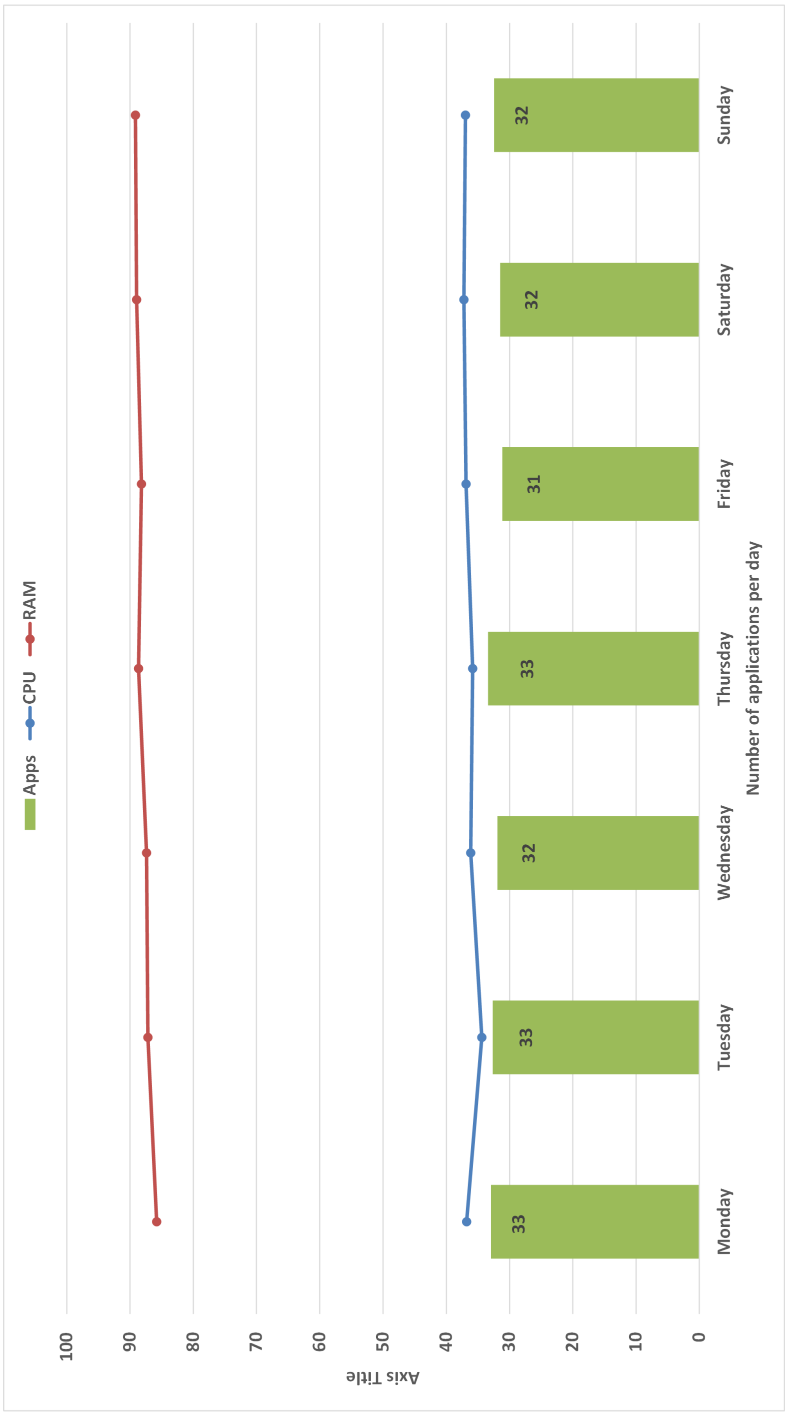


Fig. 6.16 System load for user 8

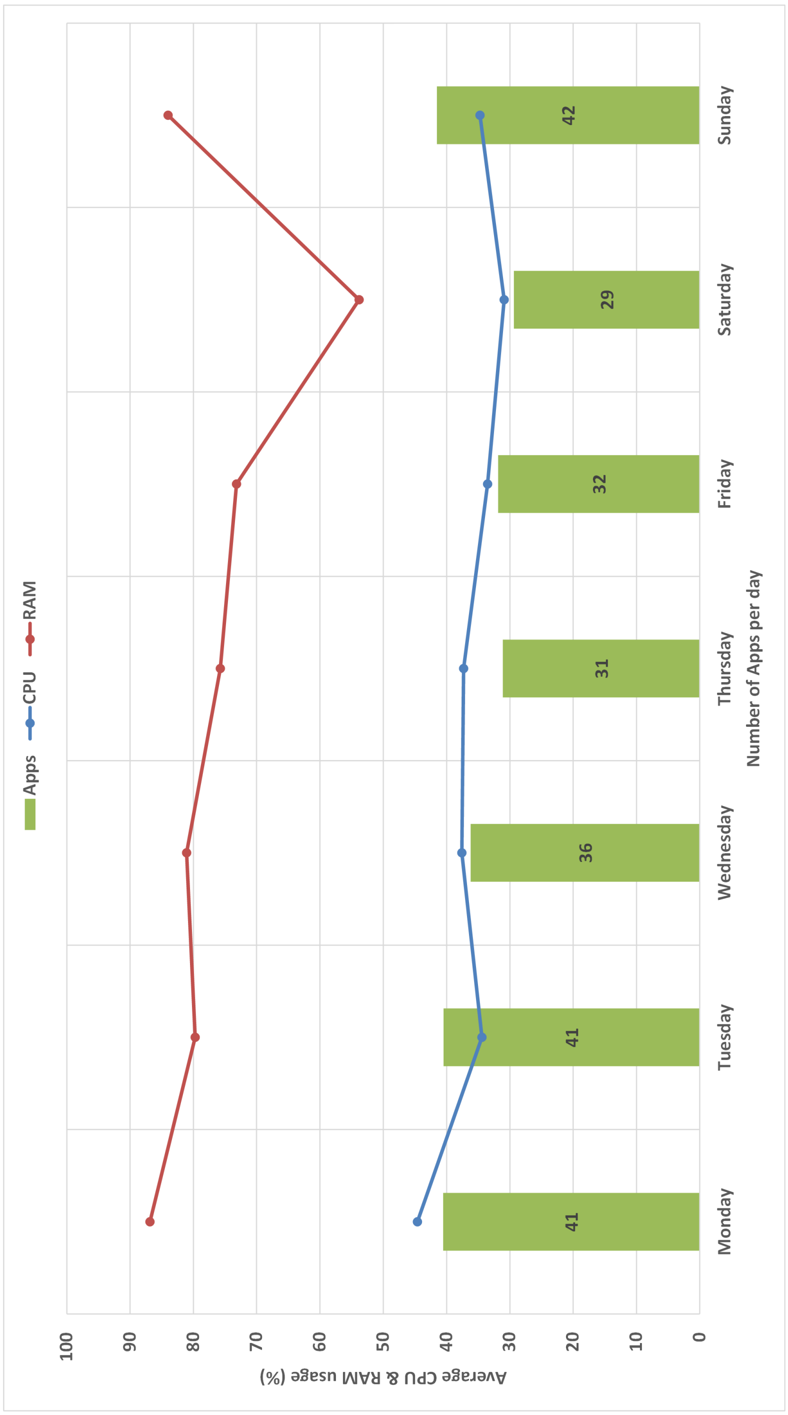


Fig. 6.17 System load for user 9

6.3.5 User mobility behaviour

6.3.5.1 Comparison of user 1 and 8

Table 6.15 Mobility range for user 1 and 8

User	Averge	Maximum	Minimum
1	7.06	7.06	7.06
8	6.19	6.93	2.99

One average the two users were only 1.48 miles apart in terms of their distance from Charing Cross. User 1 was farther away with an average distance of 7.06 miles compared to that of 6.19 miles for user 8. User 1 was completely immobile during the whole week whereas User 8 stayed between 2.99 and 6.94 miles from Charing Cross. On Sunday, User 8 didn't exhibit any movement this may be because he/she spent the weekend at home. Although User 8 was mobile, throughout the week with the exception of Sunday, but the negligible distance between the two users should result in mobility as a weak feature to discriminate them when comparing their profiles.

Table 6.16 Average distance per day for user 1 and 8

User	MON	TUE	WED	THU	FRI	SAT	SUN
1	7.06	7.06	7.06	7.06	7.06	7.06	7.06
8	6.198	6.301	6.249	6.289	5.723	6.283	6.289

6.3.5.2 Comparison of user 3 and 4

Table 6.17 Mobility range for user 3 and 4

User	Average	Maximum	Minimum
3	6.85	10.76	0.19
4	3.89	5.337	0.45

In terms of distance from Charing Cross, user 3 was always farther away with an average of 6.85 miles compared to 3.89 miles for user 4. In terms of their mobility range, user 3 travelled doubled the distance as compared to user 3. Although both uses were 3 miles apart on average, they had clearly distinct mobility patterns. User 4 always travelled towards central London in the morning and returned back in the evening with the exception of Sunday when the user stayed home. In comparison, user 3 travelled in the evenings and returned home in early hours of the next morning.

This clear distinction in distance and time of travel makes mobility a very good discriminatory feature for the two users. As a result these two users should have distinct profiles.

Table 6.18 Average distance per day for user 3 and 4

User	MON	TUE	WED	THU	FRI	SAT	SUN
3	8.009	6.864	7.811	7.176	7.176	4.765	6.178
4	3.511	3.41	4.061	3.526	3.711	3.727	5.308

6.3.5.3 Comparison of user 8 and 9

Table 6.19 Mobility range for user 8 and 9

User	Average	Maximum	Minimum
8	6.19 miles	6.93	2.99
9	9.492	34.1	1.06

On average the delta of distance from Charing Cross was within 3.3 miles of each other for these users. But with the exception of Friday and Saturday both users were less than 1 mile apart. User 9 travelled as far as 35 miles which may a one time event as there was to previous and subsequent mobility event of such magnitude. User 9 travelled four days in a week compared to a single day travel pattern for user 8. Other than these distinctions, both users had almost same mobility patterns which may indicate that only frequency element of the mobility feature will affect the profiles and distance may not play a huge part in distinguishing both users.

Table 6.20 Average distance per day for user 8 and 9

User	MON	TUE	WED	THU	FRI	SAT	SUN
8	6.198 m	6.301 m	6.249 m	6.289 m	5.723 m	6.283 m	6.289 m
9	6.979	6.822	7.037	7.639	7.302	34.1	7.011

The mobility behaviour comparison of these users is provided in graphical form in the Figures 6.18 to 6.20.

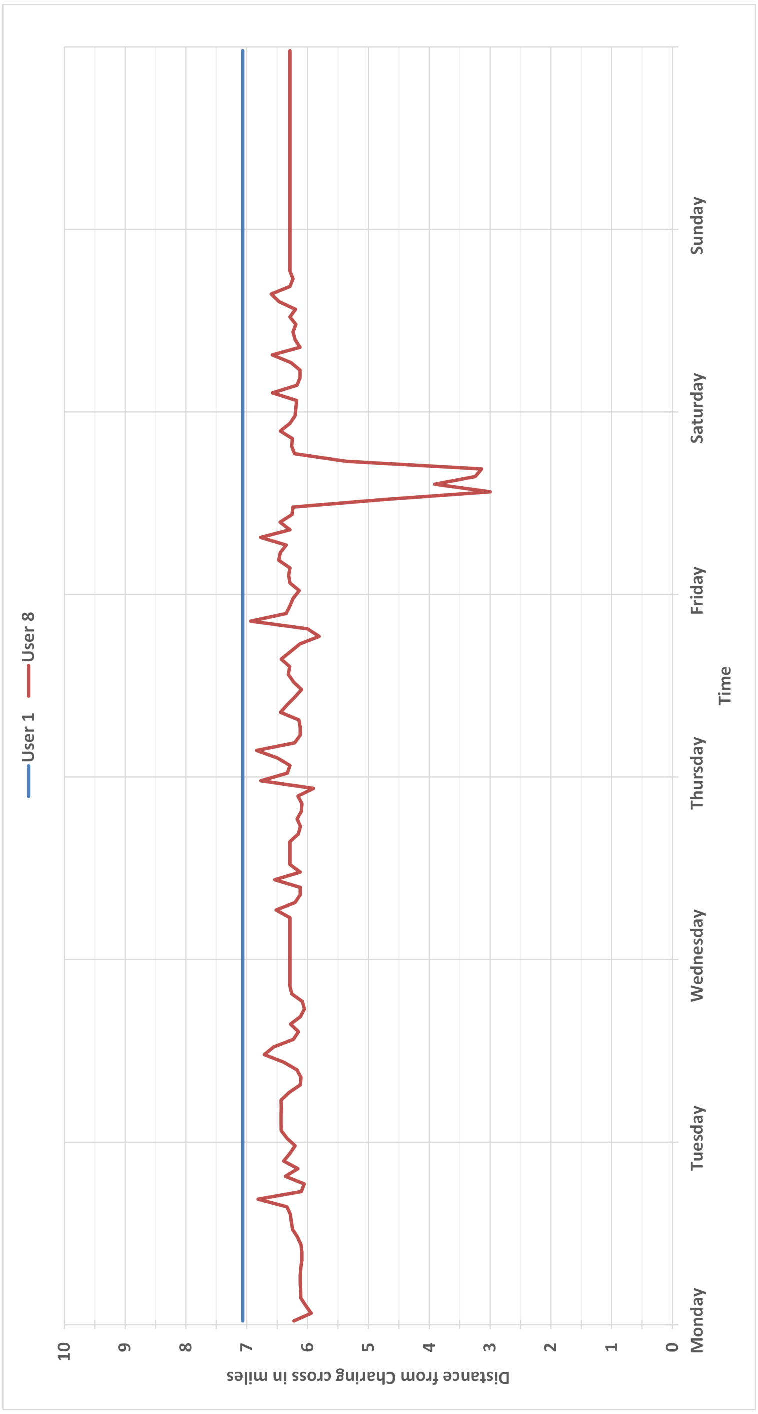


Fig. 6.18 Mobility behaviour of user 1 and 8

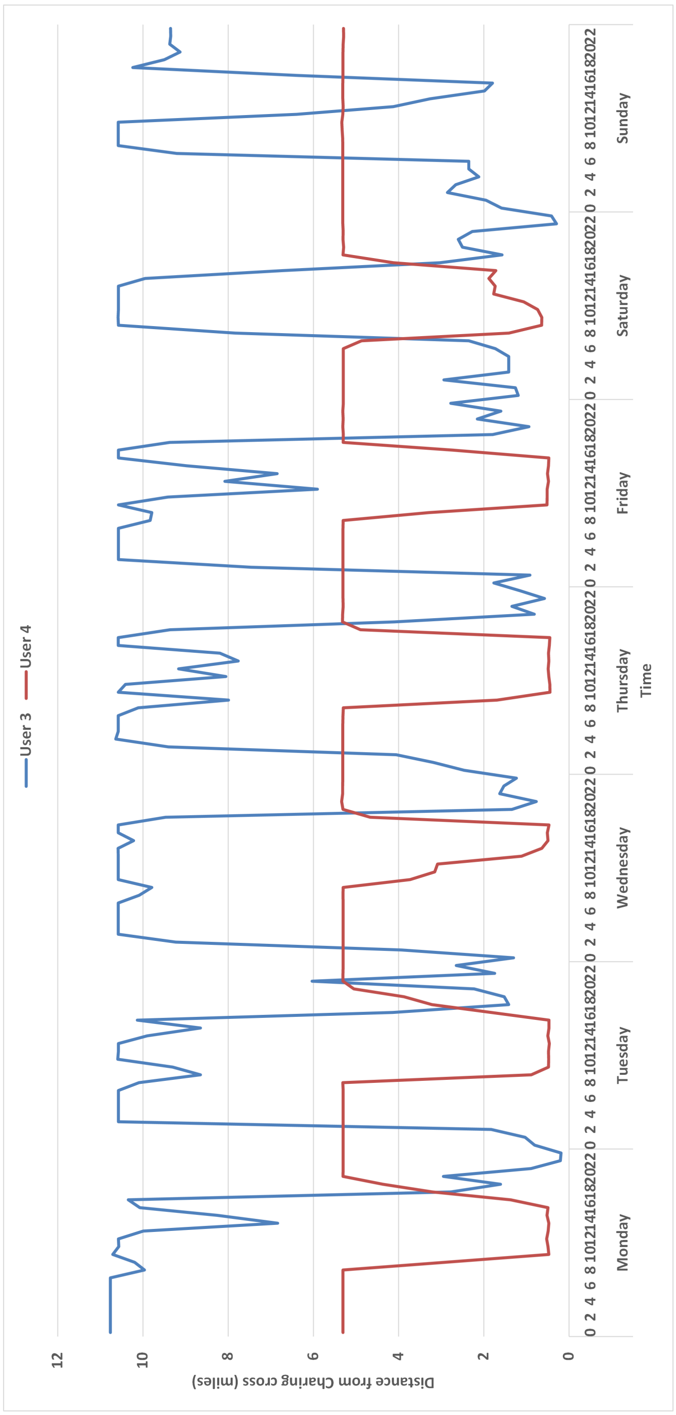


Fig. 6.19 Mobility behaviour of user 3 and 4

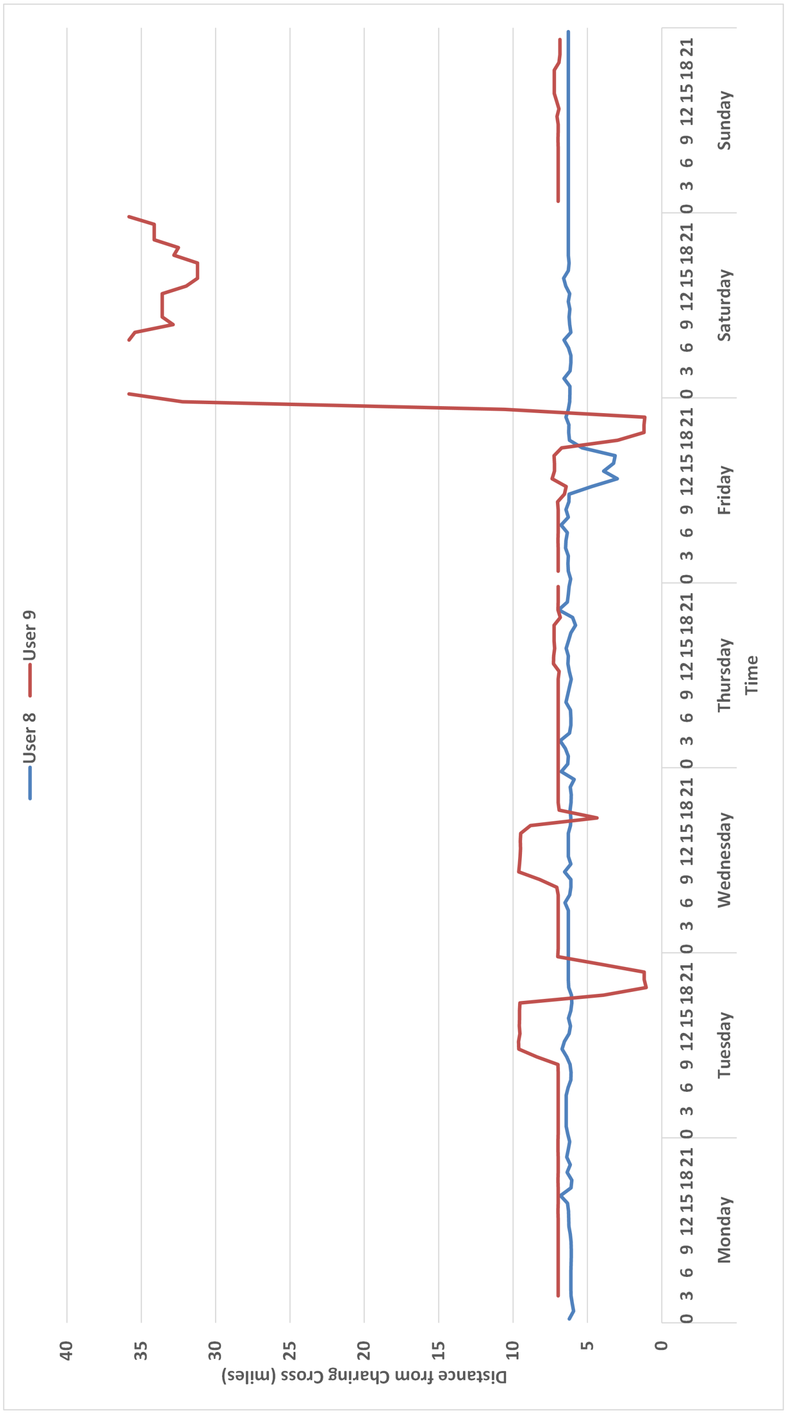


Fig. 6.20 Mobility behaviour of user 8 and 9

6.4 MLP test 1

The descriptive statistical analysis of one week of mobile usage data showed that users' calling patterns, SMS, data traffic usage, mobility and application usage can be positively used to profile different users. This analysis also showed that most users in our dataset had consistently different usage patterns that define how they utilise their mobile device. Features for system load were somewhat different but didn't allow very obvious discrimination of users.

We set out this test to determine the accuracy of machine learning processor by implementing our Machine Learning Processor in a simulated Matlab environment. The results of this test were then compared with the analysis of descriptive statistical test to validate our hypothesis. The following diagram shows the processor of this test. The test was divided into two stages profile generation (training) and testing.

6.4.1 Training phase

This stage takes mobile usage data from the database as an input, after pre-processing for normalisation applies machine learning algorithm to convert it into a usage profile for each user. The resultant profile is stored as Matlab native data file format. The sequence of events for this stage are presented below.

6.4.2 Input selection

Similar to our previous test 6.3, we utilised one week of usage data from our dataset. Instead of using all of the collected features we utilised the following feature set:

- Number of calls made
- Duration of calls
- Number of SMS sent

- The number of bytes of data consumed
- Average CPU load
- Average RAM load
- Number of application used
- Distance from Charing Cross
- Hour of week

As discussed in Section 5.5, our Feature Extractor mobile application used an epoch of 5 minutes to collect observations which amounts to over 2000 readings for one week data. Similar to previous test (see section 6.3) , we averaged our data per hour and ended up with 168 observations for every user.

$$7\,days * 24\,hours * 1\,observation/hour = 168\,observations$$

We used Charing Cross as a reference point for all users and converted their geographic location into distance from it by using Haversine formula. The time stamp of the observation was also converted into a number range 1 to168 to uniquely represent each observation.

6.4.3 Data standardisation

In distance-based classification, we need to normalize each feature value of a feature set in order to not get conditioned by features with wider range of possible values when computing distances. In case of our test, the features have different ranges, for example 0 to 100% for CPU and RAM, but for data usage this range may vary from 0 to several thousands of bytes. In this case, a small variation in data usage is probably more influencing than a big variation for CPU or memory usage when computing the distance of two feature vectors. As discussed in in Section 5.6.1.2, we utilised z-score to standardise our input dataset.

6.4.4 Silhouette Test

In our simulation of Machine Learning Processor, we have utilised the K-Means clustering algorithm. This algorithm, requires prior knowledge of number of clusters. Silhouette test provides a statistical measure of optimum number of clusters for a given dataset. We tested performed this test for 10, 25, 50 and 100 iterations for number of clusters ranging from 2 to 20. Subsequent tests showed that above 50 iteration there was no change in the results hence we selected 50 number of iteration for the tests, means and standard deviation of silhouette score for each value of k (number of clusters) was calculated and value of k with highest silhouette score was selected as an input for the -Means clustering algorithm.

The Table below shows the optimum number of clusters for each user while detailed results for silhouette tests are presented in Figures 6.21 – 6.30. It is observed that users 1,2,5 and 8 have far less number of clusters for this usage profile as compared to users 3,4,6,7,9 and 10 who are presumable more sophisticated users resulting more variability in their usage patterns.

Table 6.21 Optimum number of cluster based on silhouette scores

User	Clusters (k)
1	3
2	8
3	20
4	17
5	6
6	19
7	16
8	2
9	10
10	16

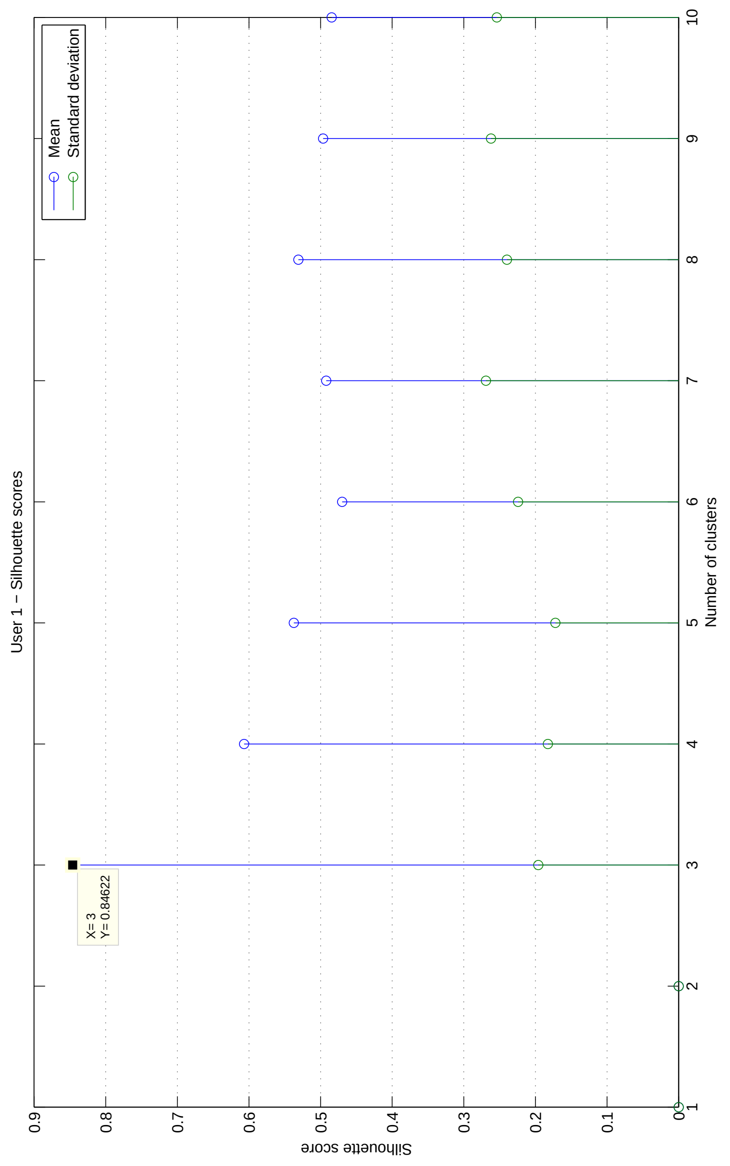


Fig. 6.21 Silhouette score for user 1

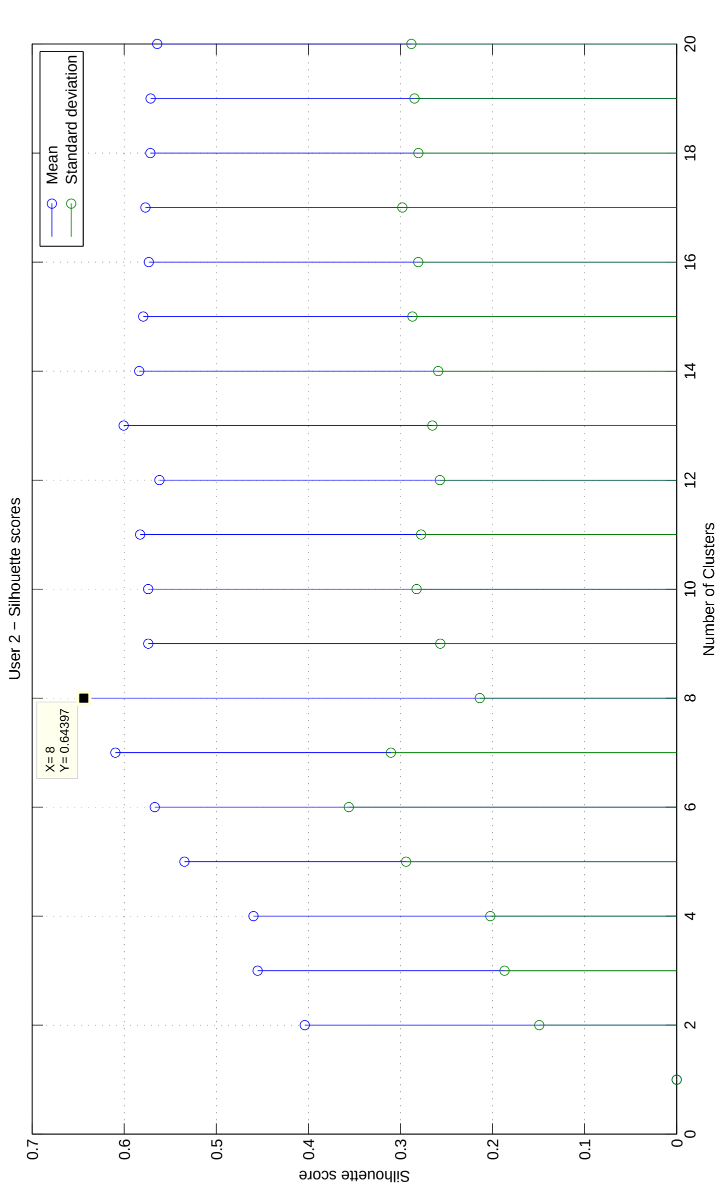


Fig. 6.22 Silhouette score for user 2

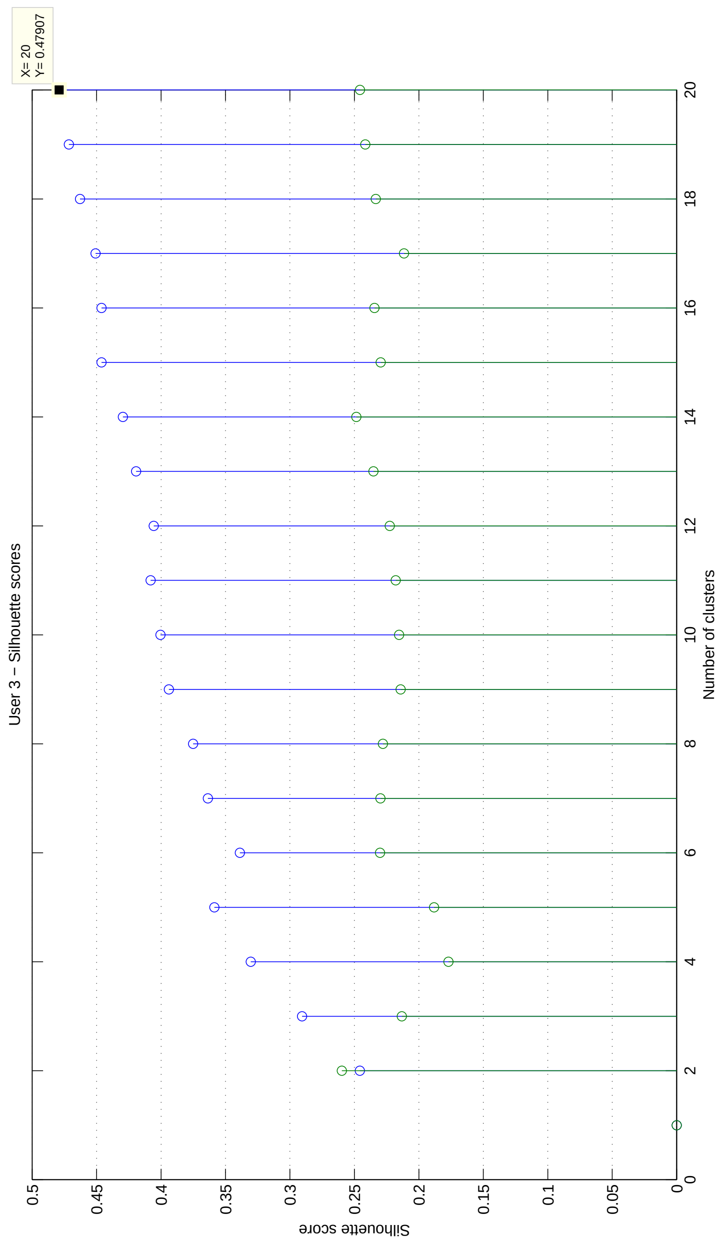


Fig. 6.23 Silhouette score for user 3

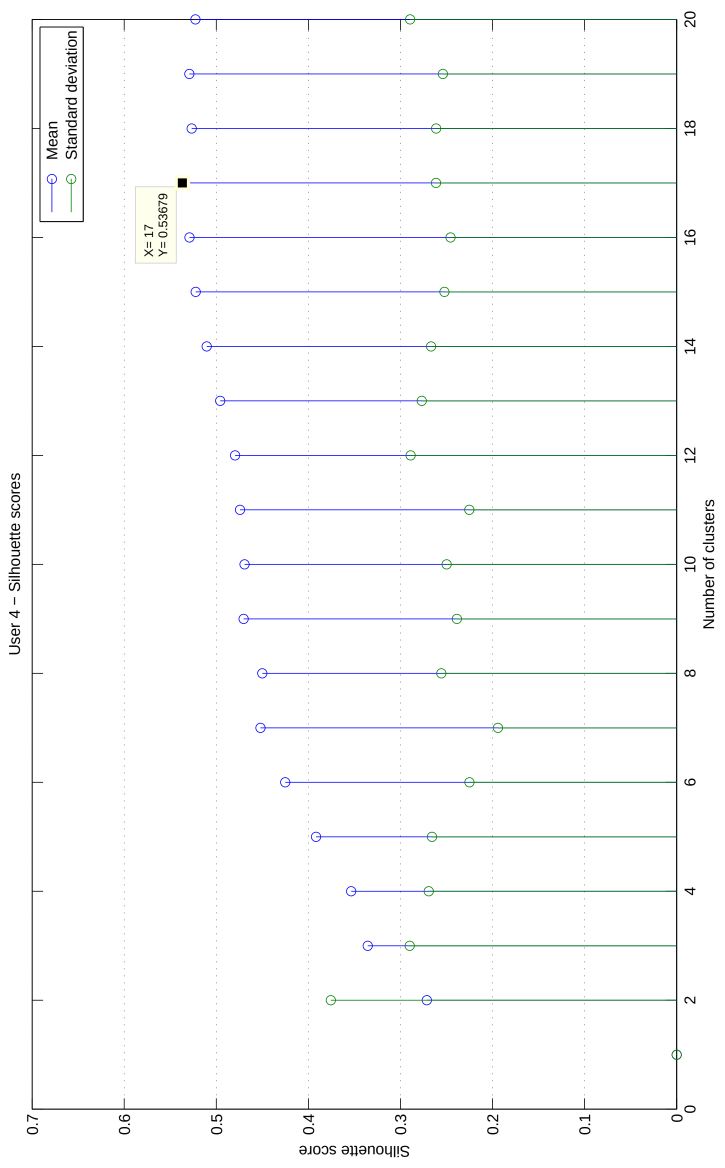


Fig. 6.24 Silhouette score for user 4

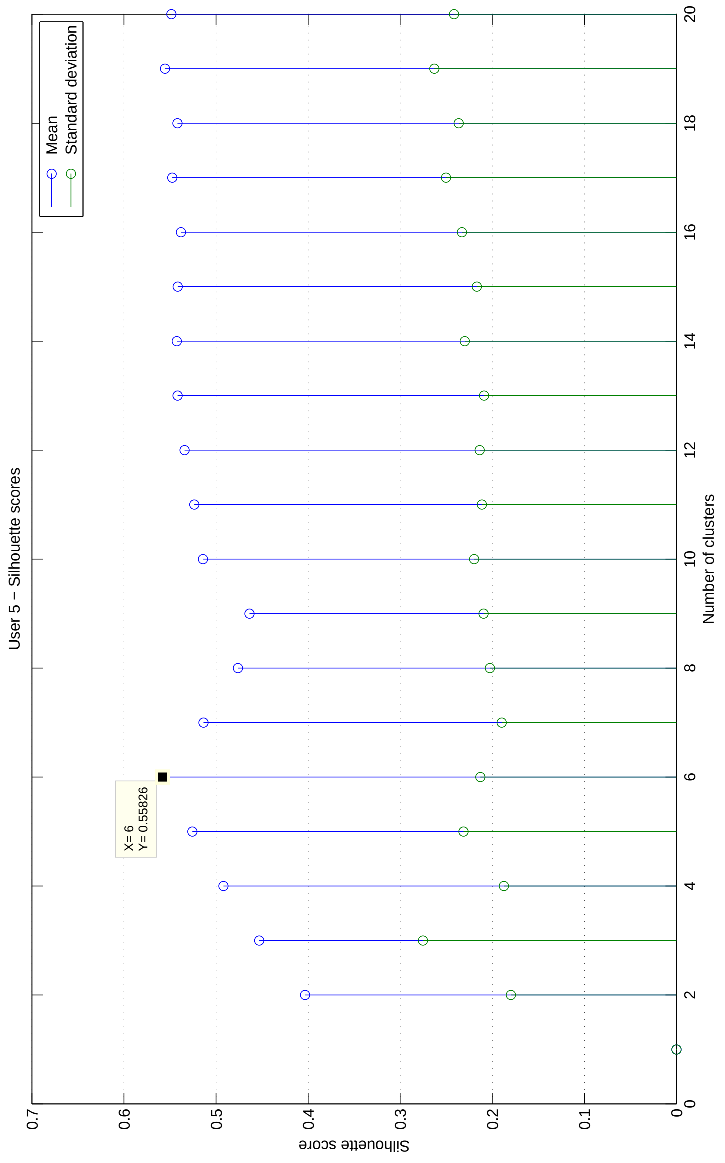


Fig. 6.25 Silhouette score for user 5

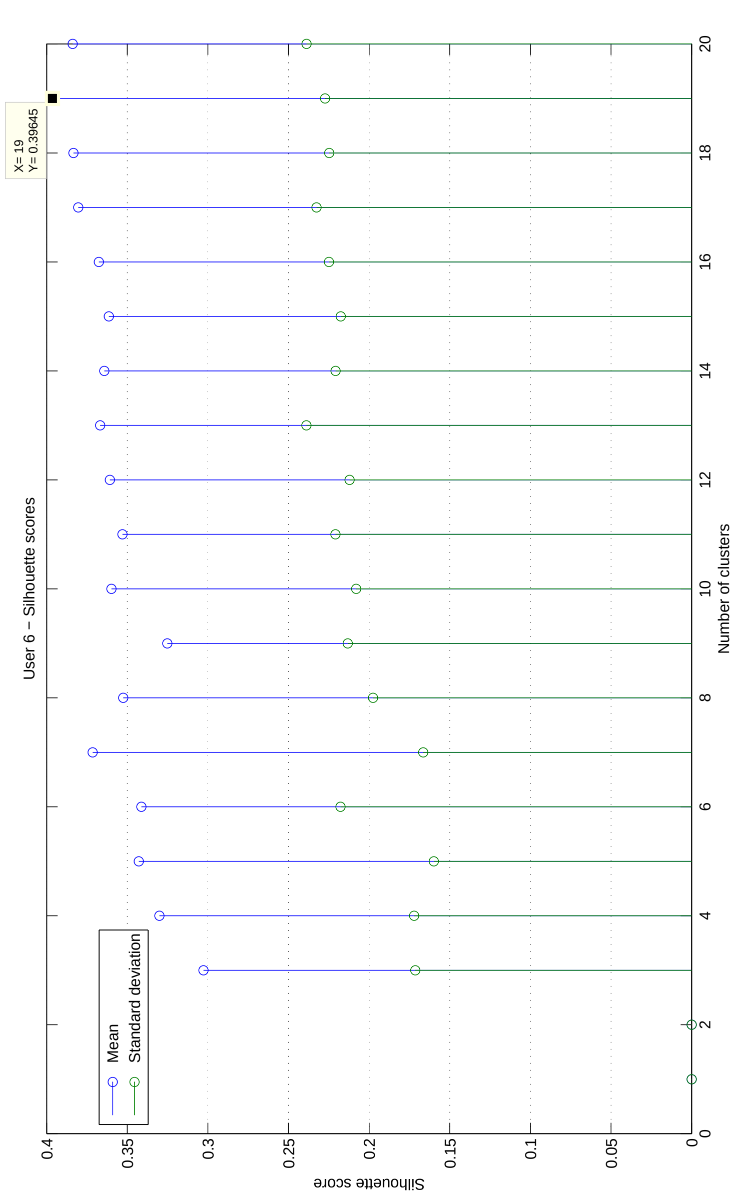


Fig. 6.26 Silhouette score for user 6

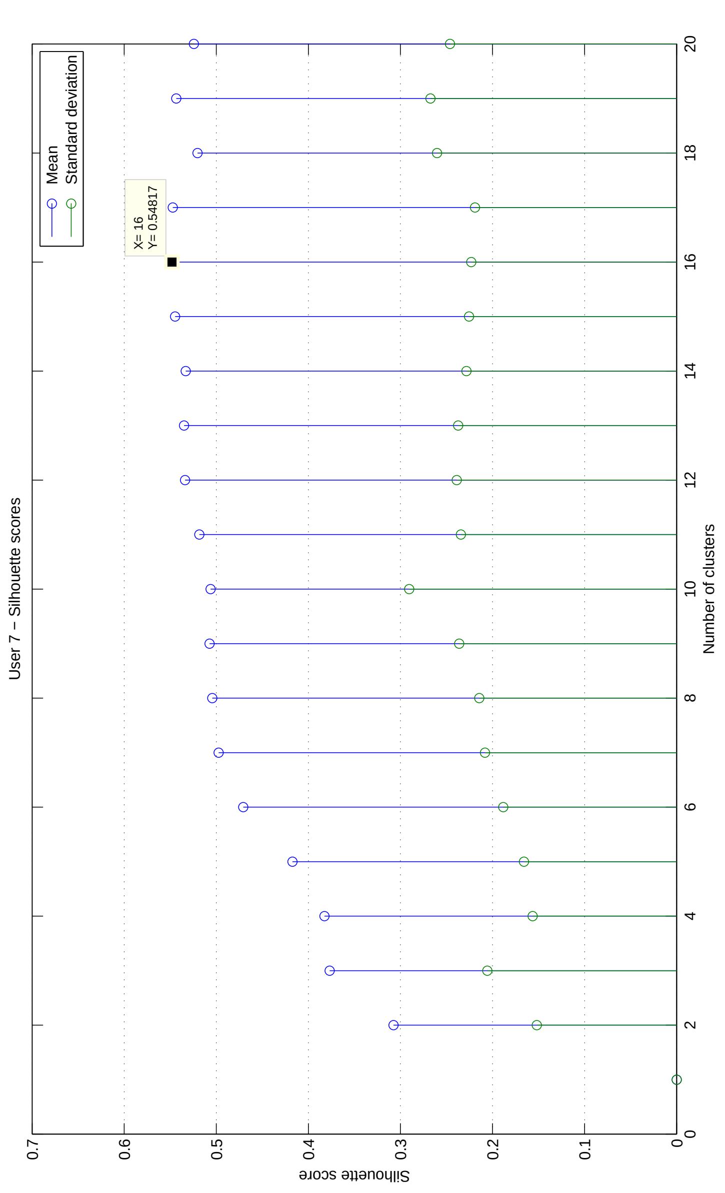


Fig. 6.27 Silhouette score for user 7

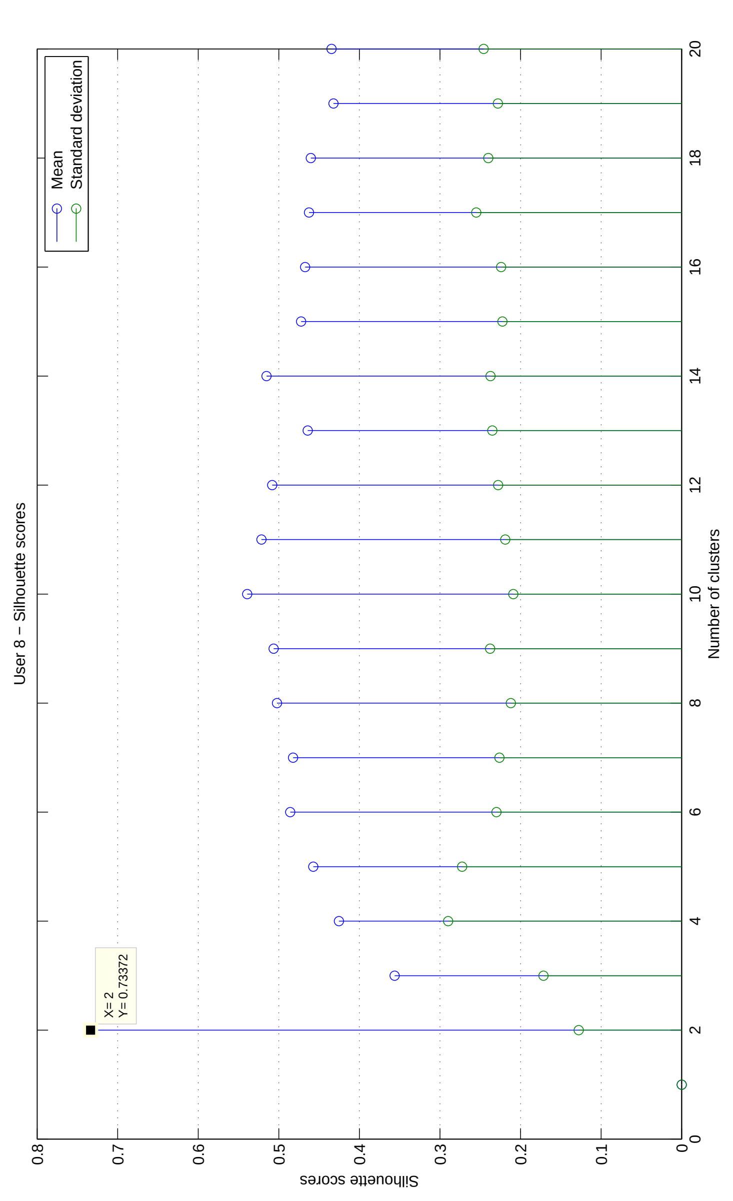


Fig. 6.28 Silhouette score for user 8

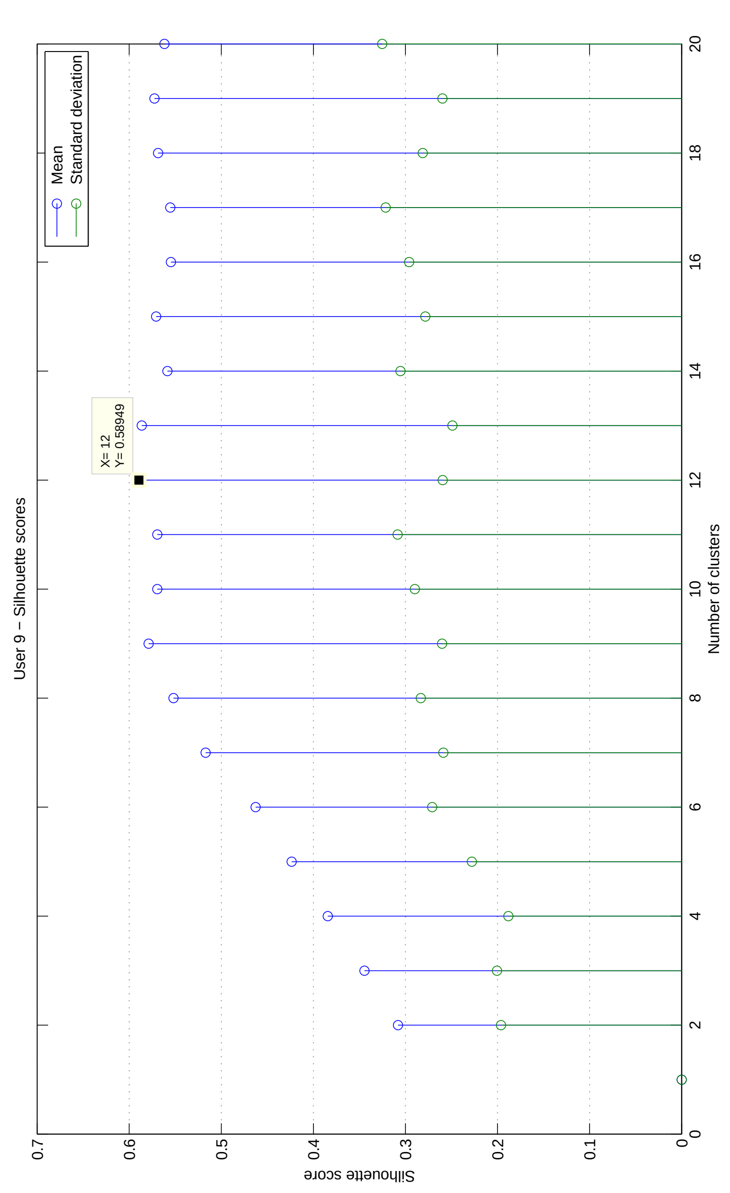


Fig. 6.29 Silhouette score for user 9

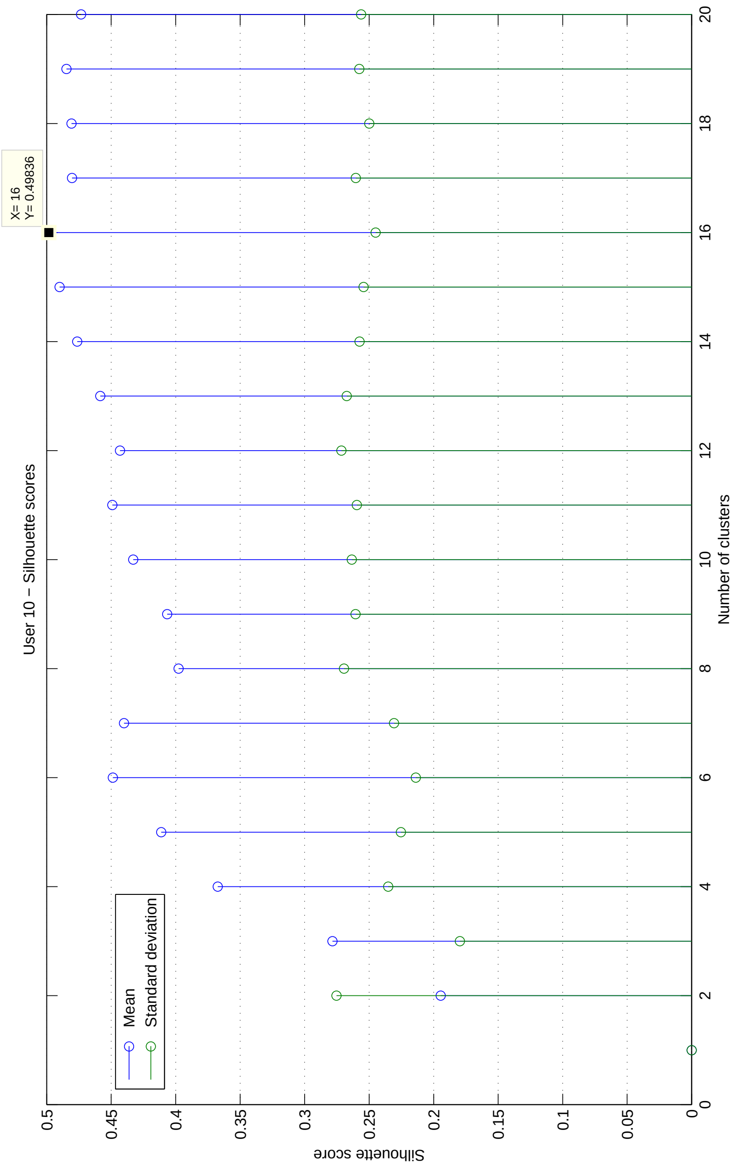


Fig. 6.30 Silhouette score for user 10

6.4.5 Training phase - usage profile generation

The following inputs were used for the K-Means clustering algorithm of our Machine Learning algorithm:

- Number of clusters (k) obtained from silhouette test.
- Standardised dataset using z-scores
- Replication was set to 20 to avoid local minimisation issue.

The generated profiles consists of the following data elements

Centroids A k-by-d dimensional matrix containing cluster means.

Cluster Boundaries A 1-by-k vector, containing distance from centroid to farthest data point in each cluster.

Cluster Indices A N-by-1 Vector containing number of associated cluster for each data point.

Feature Means A 1-by-d vector containing mean for each feature of the dataset.

Feature Standard Deviations A 1-by-d vector containing standard deviation for each feature of the dataset.

6.4.6 Testing phase - anomaly detection

We didn't have access to imposter dataset, nor was it feasible for the testers to hand their devices other people for long period of time. Hence for testing the accuracy of profiles in this test, we utilised data from actual user as training data, and mobile data from all other users as test data. As an upshot we ended up with 9 test samples/cases. In order to detect the deviations from the profile we devised a statistical test (Cluster Boundary Test). Please refer to section 5.6.2 for details of how this was carried out.

6.4.7 Discussion of results

In the previous test (section 6.3) we compared mobile device usage data of our users through descriptive statistical analysis technique. Although we operated on a subset of our data, the analysis of the comparison gave a clear indication that users' had consistent and unique application usage patterns and the contextual information (time and location). We applied this test on three pairs of users and identified the unique features that should make their profiles distinct.

Based on the results of the descriptive statistical analysis, we carried out this test with two objectives in mind:

- Acquire statistical accuracy of our simulated implementation of Machine Learning Processor.
- Evaluate the hypothesis made in previous test (section 6.3) based on profile accuracy scores is this statistical test.

The profile accuracy scores for each user under 9 tests are shown in Table 6.22

Table 6.22 Profile accuracy results

User	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
1		100	100	100	100	100	100	100	100	100
2	100		100	100	100	100	100	100	100	100
3	100	100		100	100	100	100	100	100	100
4	100	100	100		75.59	100	100	100	98.72	100
5	100	100	100	98.21		100	100	100	99.36	100
6	100	100	100	100	100		100	100	100	100
7	100	100	100	100	100	100		100	100	100
8	25.59	100	100	100	100	100	27.38		42.67	87.8
9	92.85	100	100	69.64	39.88	100	53.57	4.76		100
10	100	100	100	100	100	100	100	100	100	

The profiling accuracy is the percentage of observations from test data that don't belong to any cluster in the usage profile as discussed in section 5.6.2.1. Based on the average of nine tests, the profile accuracy range was between 73.41% and 100% and 8 out of 10 users profiles were more than 95% accurate. These results are very promising and fully support the idea that usage statistics can be modelled into profiles that can efficiently decimate genuine users from imposters. In the forthcoming sections we

Table 6.23 Average Accuracy

User	Average Accuracy
1	100
2	100
3	100
4	97.15
5	99.73
6	100
7	100
8	75.94
9	73.41
10	100

will analyse the results of this test against the previous tests. Since the previous test randomly used six users, we will compares the results of those user pairs.

6.4.7.1 Comparison of user 1 and user 8

The profile accuracy figures for users 1 and 8 are shown in Table 6.24. Both users had very low number of clusters compared to other users, which indicate a small number distinct usage patterns without any sudden variations and one off extreme events.

Table 6.24 User 1 and 8 profile comparison

User	Number of Clusters	Accuracy
1	3	100
8	2	25.59

As shown during descriptive statistical analysis (section 6.3), both of these users had very distinct calling, data and application usage patterns which should contribute towards having different profiles. Since SMS sending pattern was almost similar and not a well utilised feature it should contribute towards discriminating these users' profiles. Lastly the only difference in their mobility profile was that user 1 was completely static with average distance of 7.067 miles and compared to a mobility range of 2.99 to 6.94 miles for user 8. Almost 90% of the time the delta of distance was less than 0.5 miles which may also limit the effectiveness of using this feature.

Table 6.25 Cluster boundary comparison

k	User 1	User 8
1	2.9789	39.32
2	25.8	149.83
3	101.16	

Interestingly, in comparison to each other, profile for user 1 was 100% accurate while the second user had one of the least accurate profile in the whole test with accuracy of only 25.59%. User 1 had more usage patterns (clusters) compared to user 8 whose profile contained clusters with larger boundaries. The effect of such a situation is that clusters with large boundaries will accommodate observations inaccurately. In this case User 8 had very poor accuracy because observations from cluster 1, 2 and 3 of user 1 would incorrectly fit in clusters 1, 1 and 2 of user 8.

6.4.7.2 Comparison of user 3 and user 4

The Table 6.26 shows the profile accuracy figures for users 3 and 4 from our dataset. Both users had a high number of clusters compared to other users indicating large number of distinct patterns without any sudden variations and one off extreme events.

Table 6.26 User 3 and 4 profile comparison

User	Number of Clusters	Accuracy
3	20	100
4	17	100

As shown during descriptive statistical analysis in section 6.3, both of these users had very distinct calling, SMS, data consumption, application usage, and system load and mobility patterns. An interesting intra feature difference between the two was that user 3 only utilised signification amount of mobile data services while on the move where user 4's data consumption was consistent throughout the day. As a consequence of all these differences their profiles should be unique with high probability.

This hypothesis is clearly validated by the 100% accuracy figures for both users. Also it should be noted that both users have high number of clusters, indicating more distinct usage patterns compared to user 1 and 8 in the previous test. As a result both users 3 and 4 have more compact clusters which allow for such a high accuracy figure.

6.4.7.3 Comparison of user 8 and user 9

User 8 has very small number of clusters compared to user 9, which indicate a small number distinct usage patterns for user 8. As shown during descriptive statistical analysis in section 6.3, both of these users had very distinct calling behaviour whereby user 8's talk time amounts to only 5% of that of user 9. User 9 had a one off high mobility pattern which also included missing observations. Conversely, all other features including SMS, data consumption, application usage, and system load were almost similar between these users. Based on this data it was predicted that their profile will be similar.

Table 6.27 User 8 and 9 profile comparison

User	Number of Clusters	Accuracy
8	2	42.67
9	10	4.76

The Table 6.27 shows the profile accuracy figures for users 1 and 8 from our dataset. Statistical analysis of cluster boundaries (see Table 6.28) also reveals that due to large number of clusters, their boundaries are also very compact for user 9. Whereas user 8 only has two clusters with much greater boundaries. As a consequence user 9 has 11% of the accuracy achieved by user 8.

Table 6.28 Cluster boundary comparison

k	User 8	User 9
1	39.32	4.17
2	149.83	3.37
3		4.8
4		5.51
5		0.77
6		0.65
7		4.61
8		6.52
9		5.57
10		1.41

6.5 MLP test 2

The descriptive statistical analysis of one week of mobile usage data showed that users' calling patterns, SMS, data traffic usage, mobility and application usage can be positively used to profile different users. This analysis also showed that most users in our dataset had consistently different usage patterns that define how they utilise their mobile device. Features for system load were somewhat different but didn't allow very obvious discrimination of users. Based on these results we implemented our Machine Learning Processor in a simulated Matlab environment. We employed a reduced feature set and one week of usage data grouped per hour resulting into maximum 168 reading for each user. The results were very impressive and the average profile accuracy range was between 73.41% and 100%. Additionally 80% of our user's profiles were more than 95% accurate.

In this last test, we tried to use complete dataset with all features extracted from mobile devices to test the profile accuracy. This test was conducted in there phases, visual input analysis, profile generation (training) and testing.

6.5.1 Visual analysis

We employed Multi-dimensional scaling to plot our dataset for each user as two dimensional graphs to visualise the patterns of proximities (i.e., similarities or distances) in our dataset. The Figures 6.31 to 6.40 presents the results of performing MDS on our dataset. It is observed from these that data distribution in $2Dspace$ is dispersed for our users. It is also noted that just by visual analysis of the scatter plot of the usage data it is very hard to guess the dissimilarity in usage behaviour among different users.

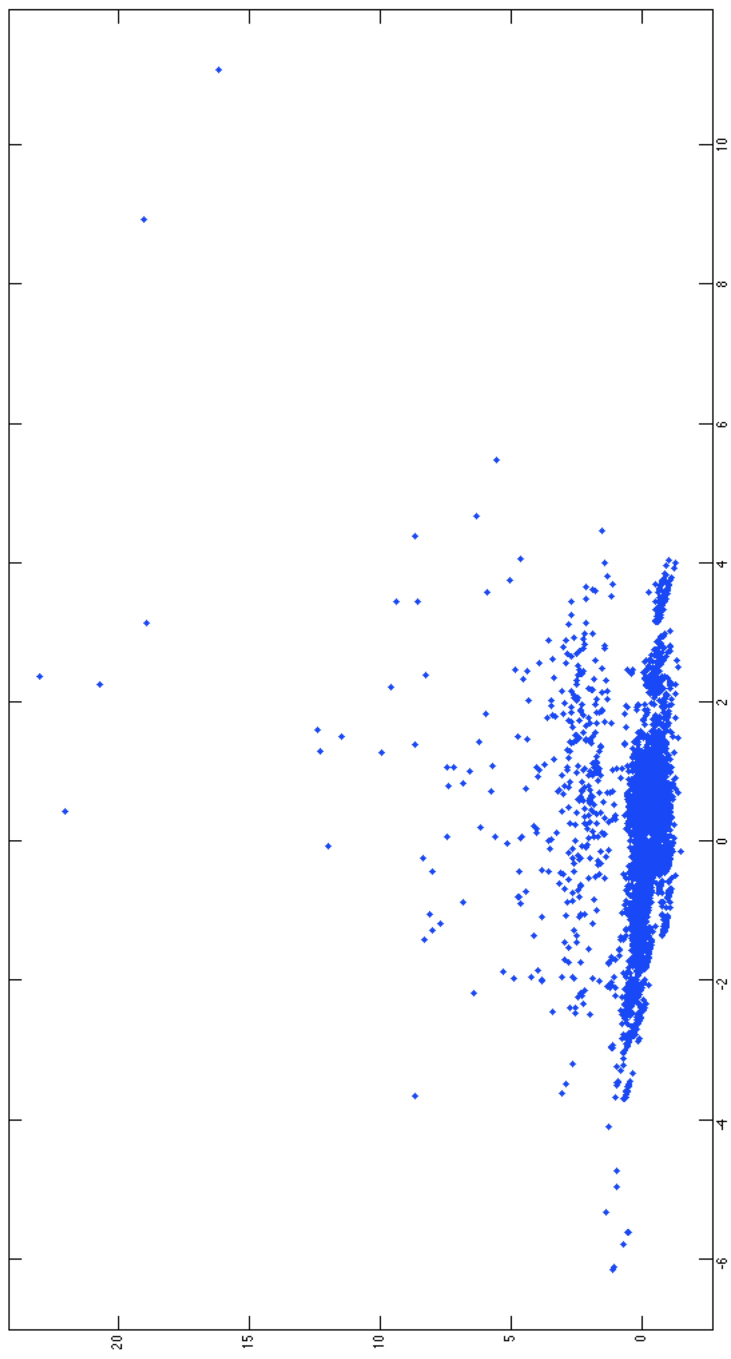


Fig. 6.31 MDS plot of normalized input data from user 1

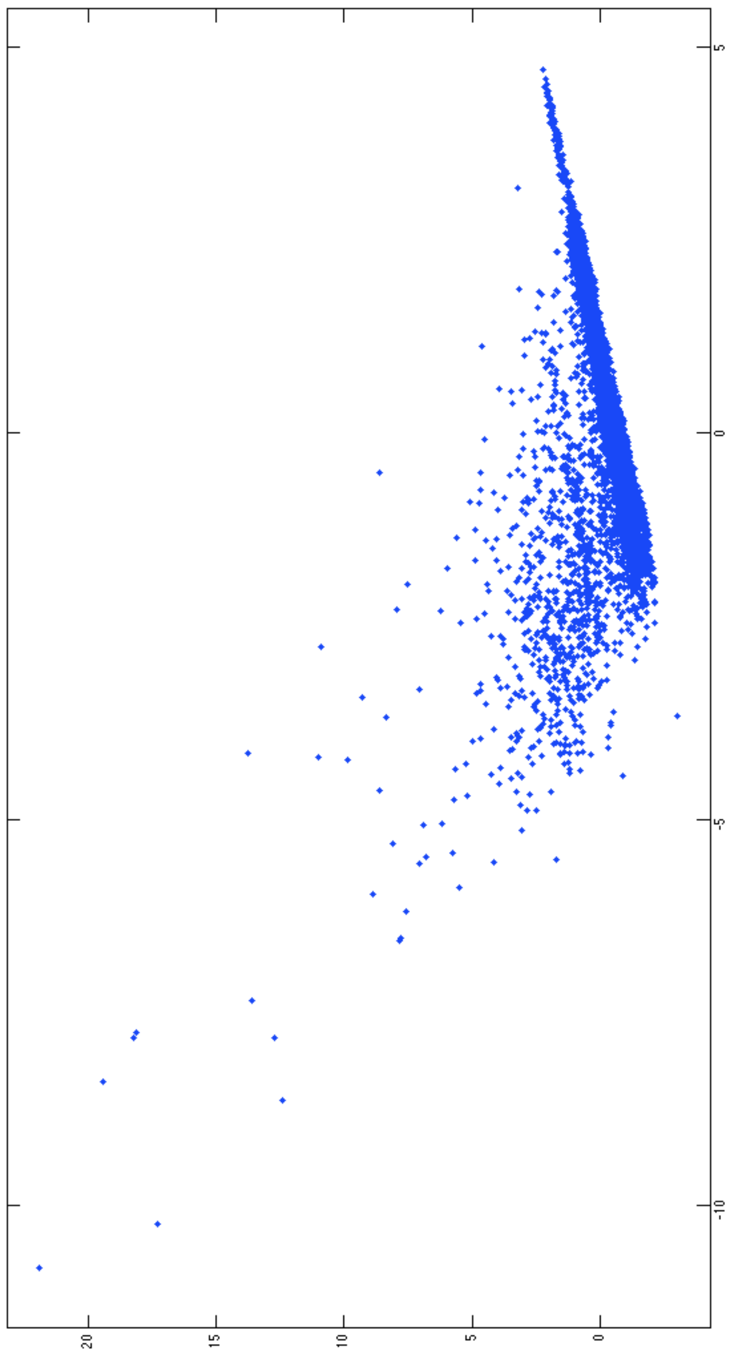


Fig. 6.32 MDS plot of normalized input data from user 2

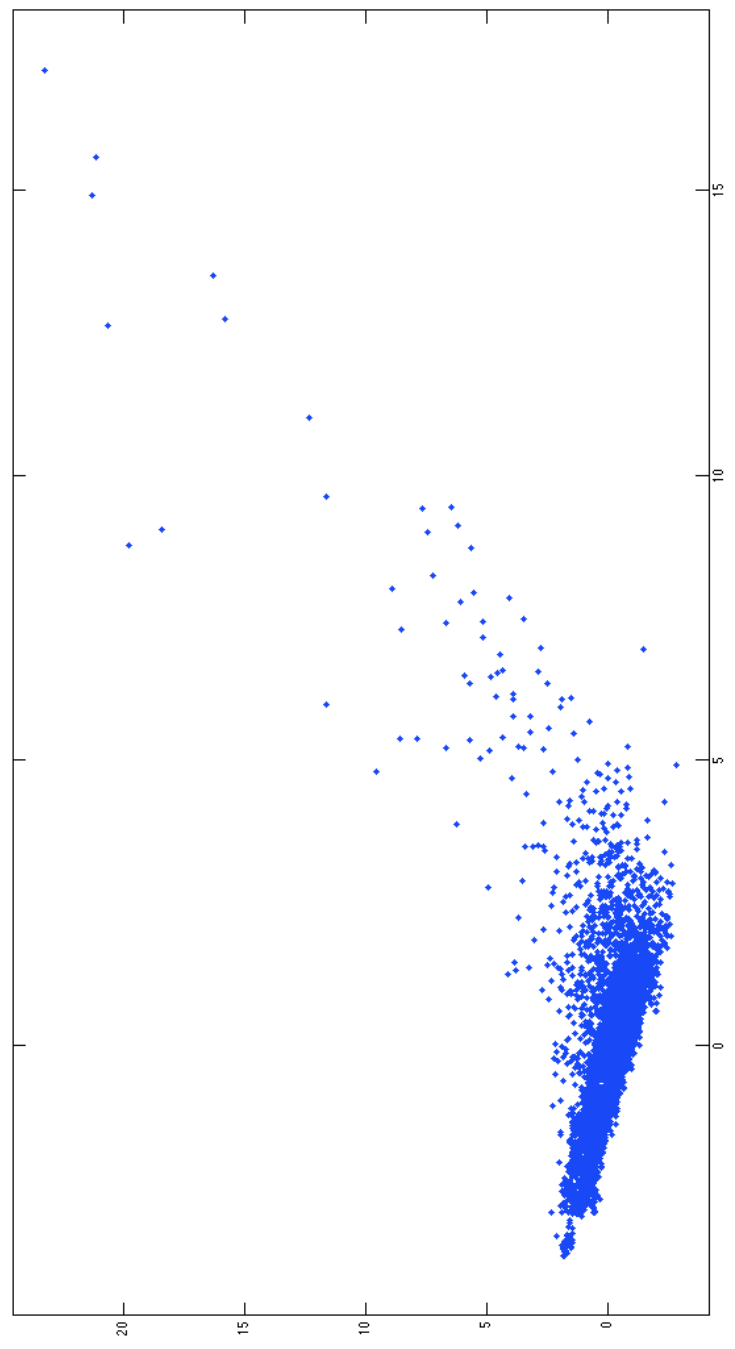


Fig. 6.33 MDS plot of normalized input data from user 3

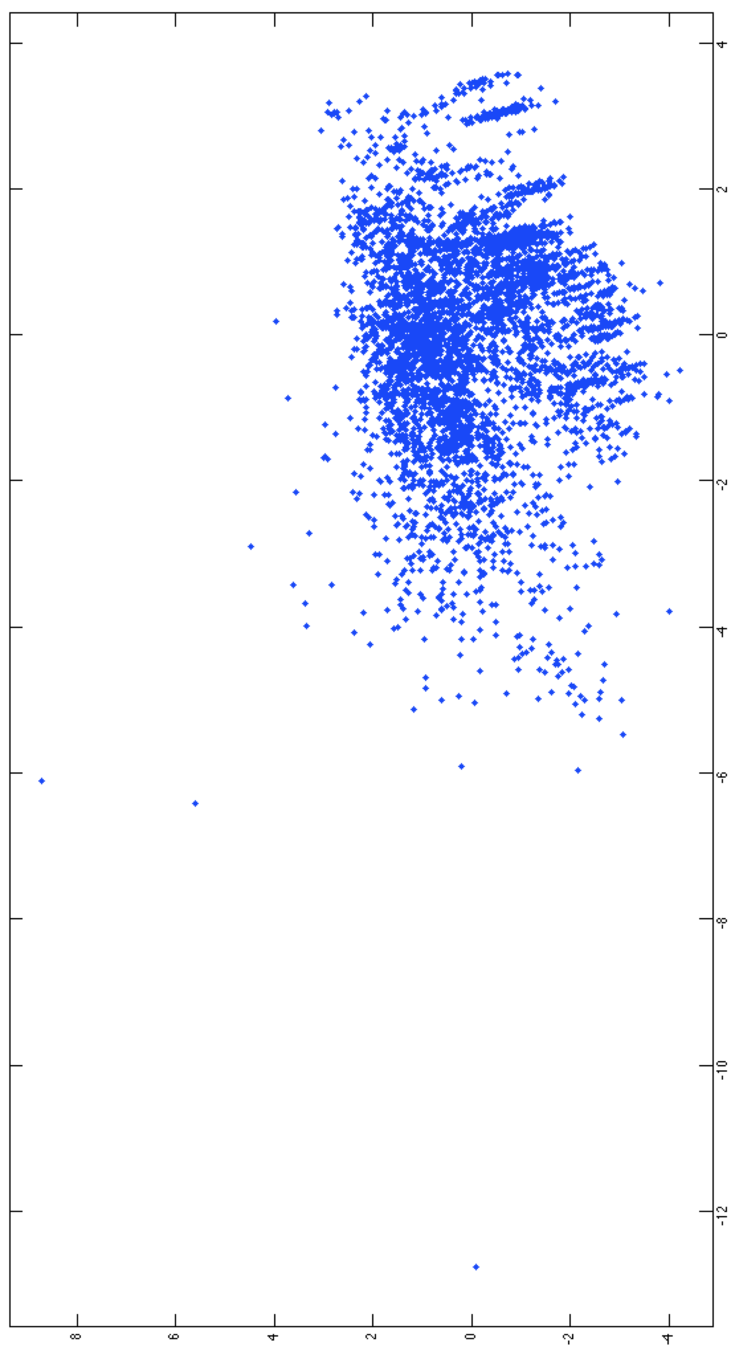


Fig. 6.34 MDS plot of normalized input data from user 4

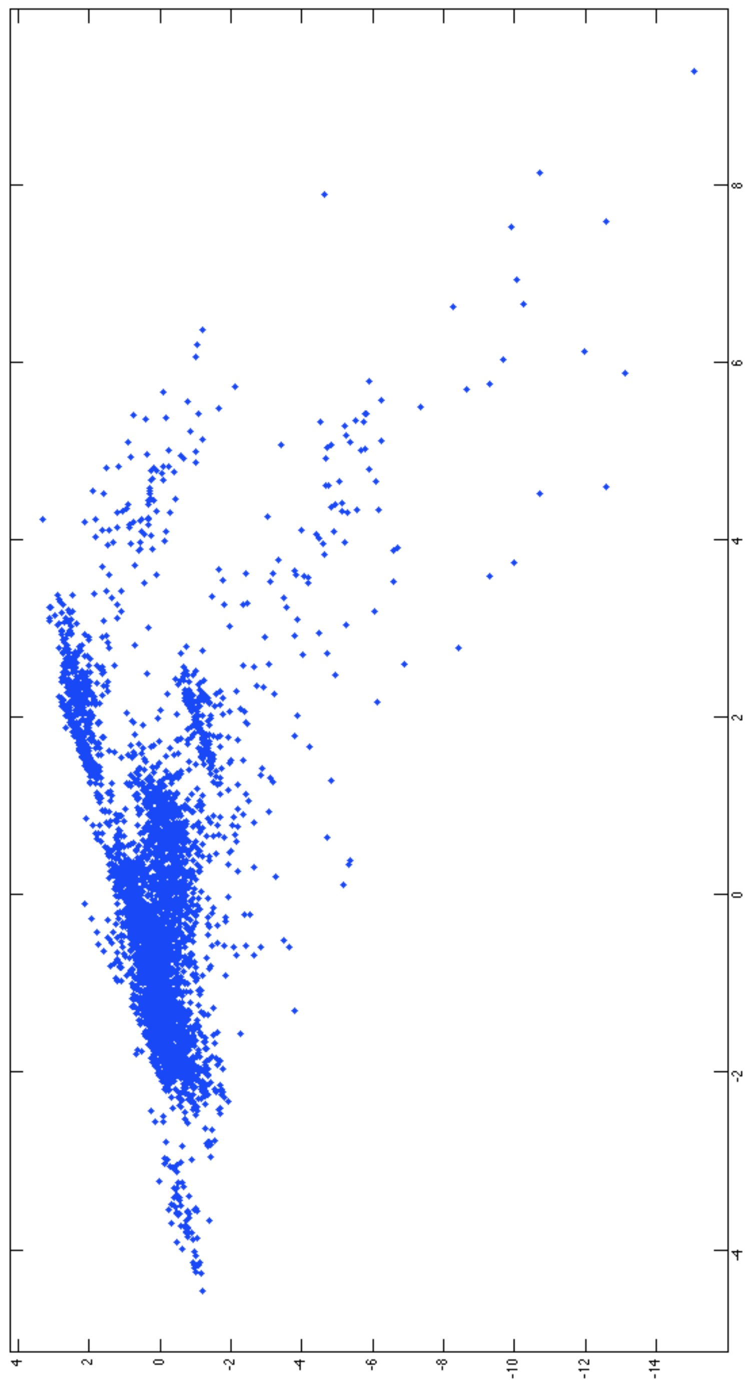


Fig. 6.35 MDS plot of normalized input data from user 5

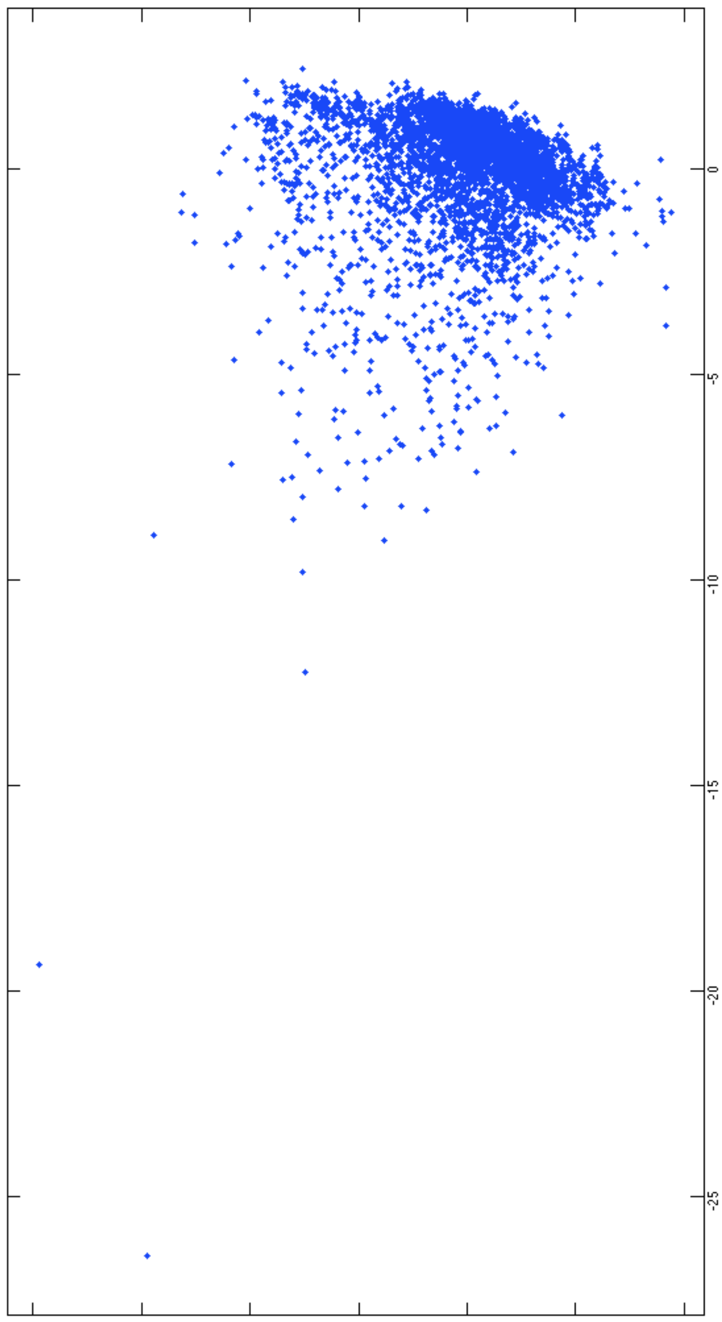


Fig. 6.36 MDS plot of normalized input data from user 6

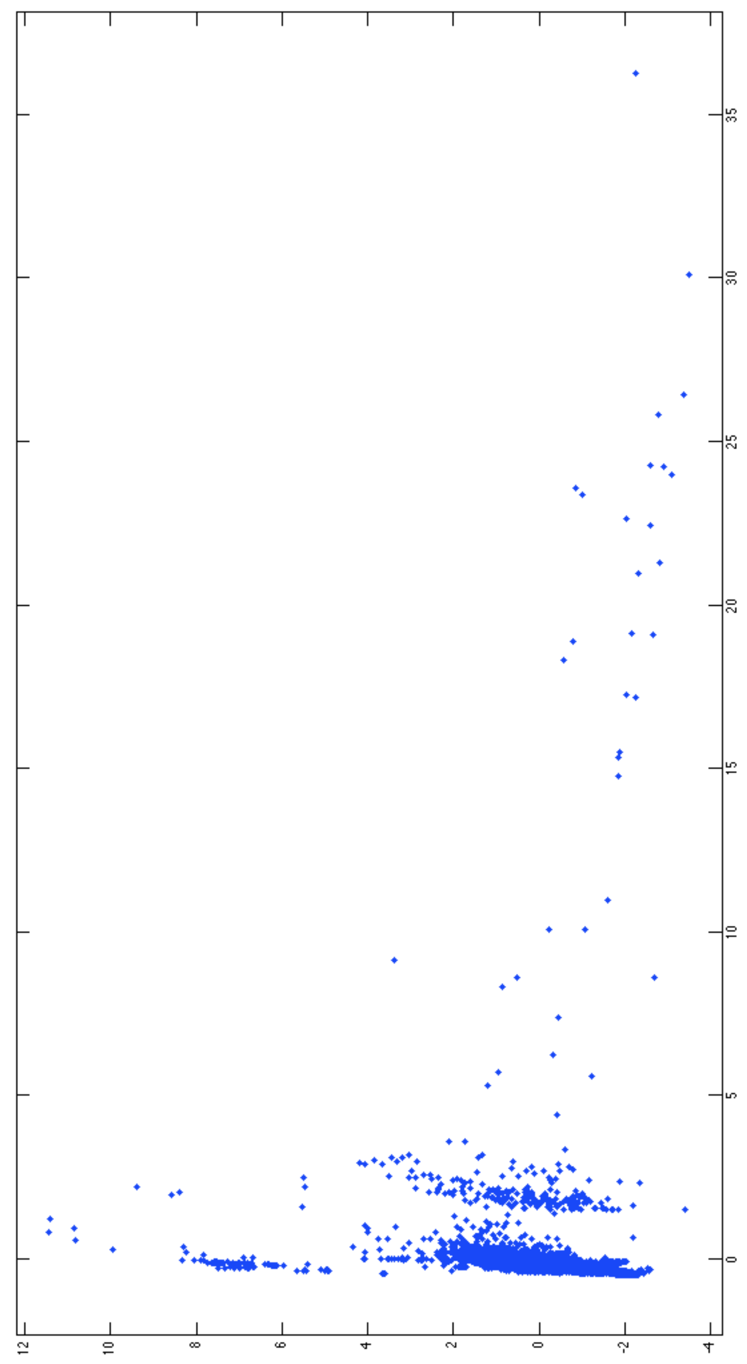


Fig. 6.37 MDS plot of normalized input data from user 7

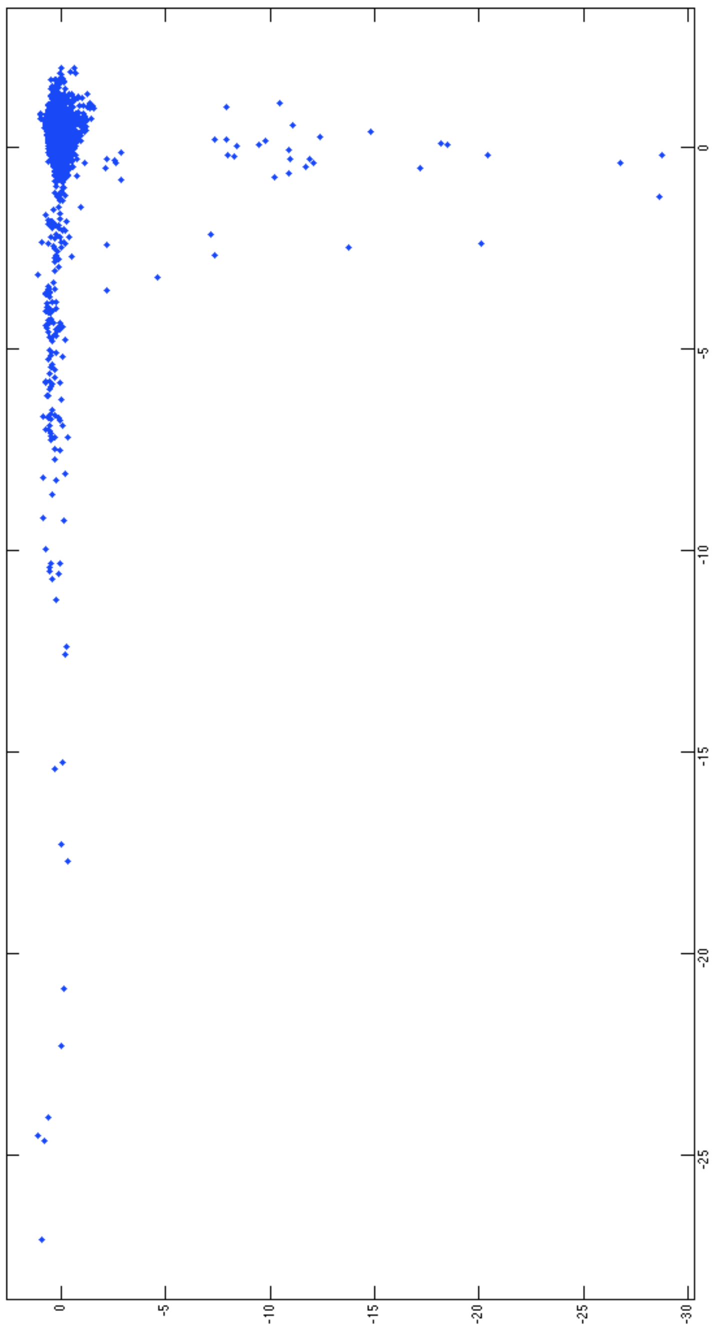


Fig. 6.38 MDS plot of normalized input data from user 8

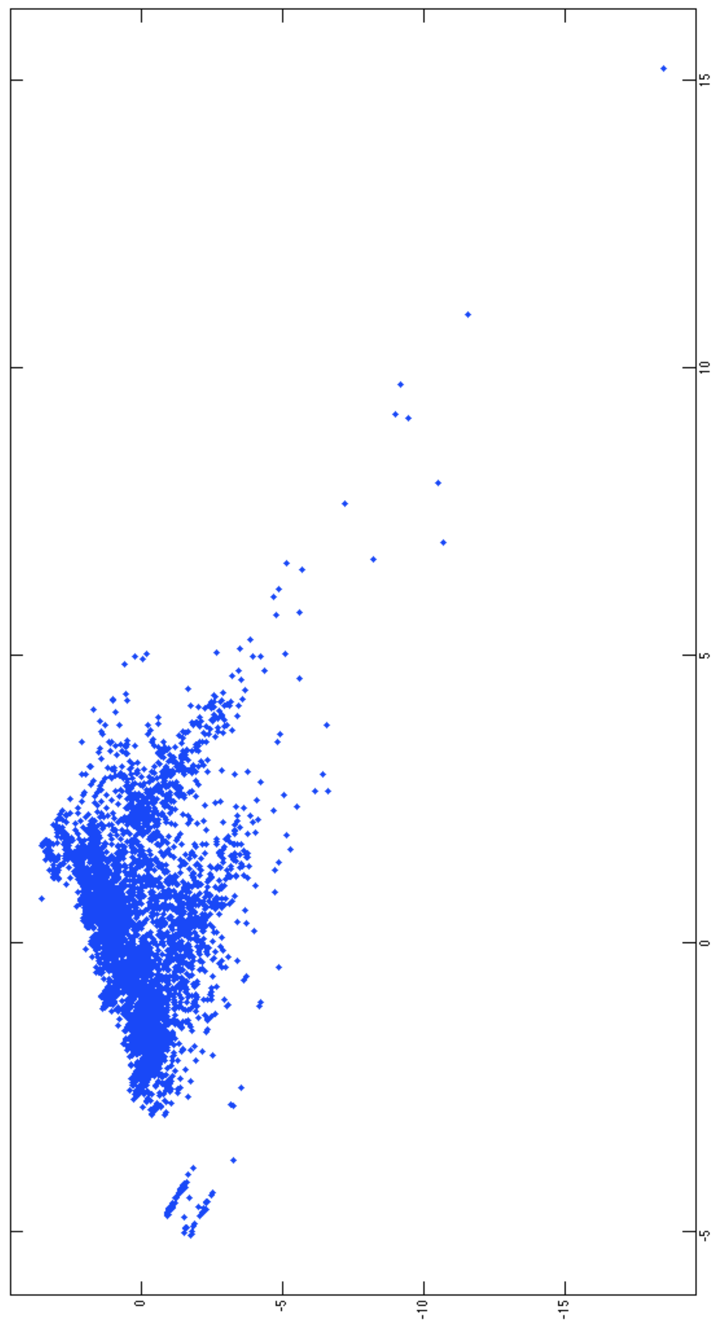


Fig. 6.39 MDS plot of normalized input data from user 9

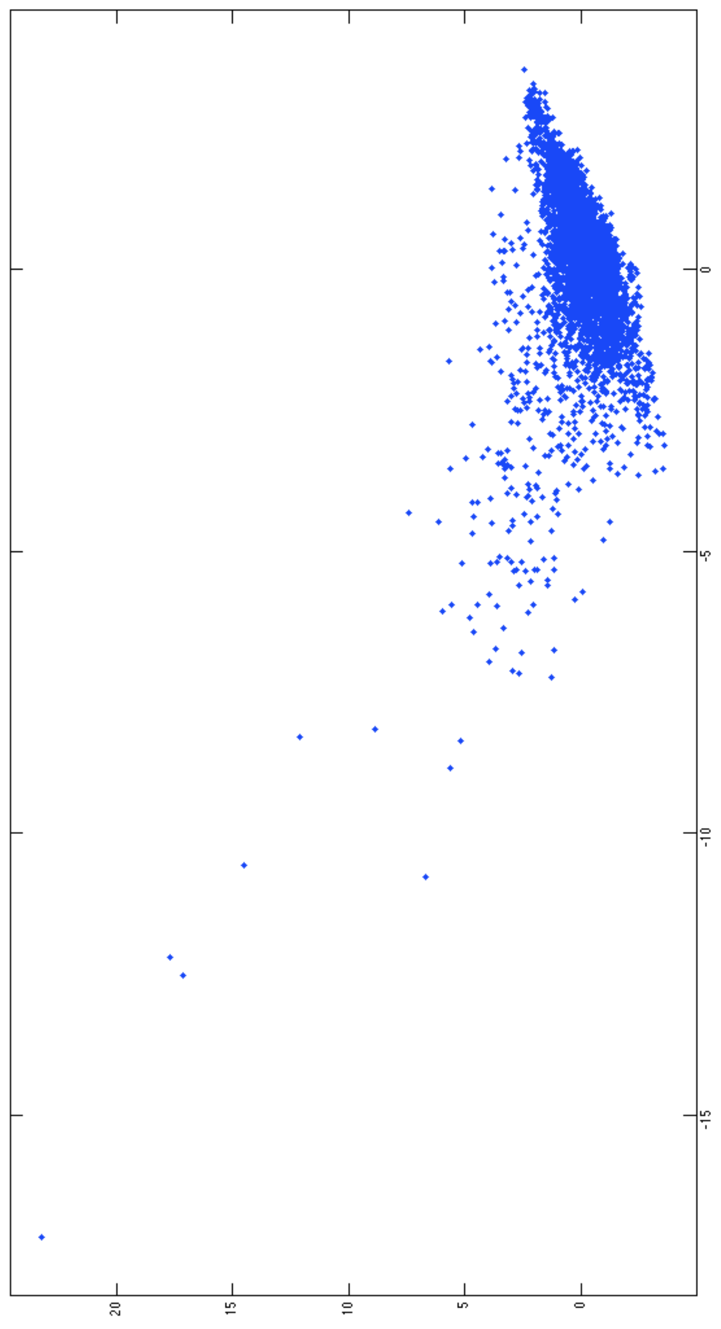


Fig. 6.40 MDS plot of normalized input data from user 10

6.5.2 Training phase

In this stage mobile usage data from the database was used as an input, after pre-processing for normalisation machine learning algorithm was applied to convert it into a usage profile for each user. The resultant profile was stored as Matlab native data file format. The sequence of events for this stage are presented below.

6.5.3 Input selection

Contrary to our previous tests discussed in sections 6.3 and 6.4, we utilised all of the usage data from our dataset. Table 6.1 shows the number of observations collected from each user of our dataset. Additionally, in this test we used following features from our dataset:

1. Number of calls made
2. Duration of calls
3. Number of SMS sent
4. Average CPU load
5. Average RAM load
6. Number of system application
7. Number of system services
8. Number of user applications
9. Number of user background services
10. The number of bytes of data sent
11. The number of bytes of data received

12. Distance from Charing Cross

13. Screen status

14. Time slice

Our Feature Extractor mobile application used an time interval of 5 minutes to collect observations which amounts to over 2000 readings for one week data. We used Charing Cross as a reference point for all users and converted their geo-graphic location into distance from it by using Haversine formula. The timestamp of the observation was also converted a range of values which defined as time slice. See Table 5.1, for more details. Each user utilised the selected features differently and Table 6.2 presents an overall view of utilisation of six features for all users of our dataset.

6.5.4 Data standardisation

As this test relies on distance-based classification, we had to normalize each feature value of a feature set in order to not get conditioned by features with wider range of possible values when computing distances. In case of our test, most of the features had different ranges, for example 0 to 100% for CPU and RAM, but for data consumption this range varies from 0 to several thousands of bytes. Similarly duration of calls varied from zero to several thousand seconds. In this case, a small variation in data usage or duration of calls is probably more influencing than a big variation for CPU or memory usage when computing the distance of two feature vectors. As discussed in in Section 5.6.1.2, we utilised z-score to normalise our input dataset.

6.5.5 Silhouette Test

In our simulation of Machine Learning Processor, we utilised the K-Means clustering algorithm. This algorithm, requires prior knowledge of number of clusters. We employed the Silhouette test as a statistical measure to determine the optimum number of clusters for each user profile in our given dataset. We performed this test for 10, 25, 50 and 100 iterations with number of clusters ranging from 2 to 20. Subsequent tests showed that above 50 iteration there was no significant change in the results hence we selected 50 number of iteration for to use for the tests.

Table 6.29 Optimum number of cluster based on silhouette scores

User	Clusters (k)
1	9
2	7
3	6
4	8
5	10
6	5
7	2
8	12
9	7
10	2

The means and standard deviation of silhouette score for each value of k (number of clusters) was calculated and value of k resulting into highest silhouette score was selected as an input for the K-Means clustering algorithm. The Table 6.29 shows the optimum number of clusters for each user profile determined by this test. It is observed for this table that number of clusters for all user has changed from MLP test 1 (see section 6.4). The average cluster size for all users in now 7 which may indicate a distinct usage pattern for each day in a week. However users 7 and 10 are an exception as they only have 2 clusters in their usage profile. Although we can say that this low number of clusters indicates less variability in their usage behaviour, however, it does not effect the profiling accuracy as it is indicated by the results shown in table 6.30.

6.5.6 Training phase - usage profile generation

The following inputs were used for the K-Means clustering algorithm:

- Number of clusters (k) obtained from silhouette test.
- Standardised dataset using z-scores.
- Replication was set to 20 to avoid local minimisation issue.

The generated profiles consists of the following data elements

Centroids A k-by-d dimensional matrix containing cluster means.

Cluster Boundaries A 1-by-k vector, containing distance from centroid to farthest data point in each cluster.

Cluster Indices A N-by-1 Vector containing number of associated cluster for each data point.

Feature Means A 1-by-d vector containing mean for each feature of the dataset.

Feature Standard Deviations A 1-by-d vector containing standard deviation for each feature of the dataset.

6.5.7 Testing phase - anomaly detection

We didn't have access to imposter dataset, nor was it feasible for the testers to hand their devices other people for long period of time. Hence for testing the accuracy of profiles in this test, we utilised data from actual user as training data, and mobile data from all other users as test data. As an upshot we ended up with 9 test samples/cases. In order to detect the deviations from the profile we devised a statistical test (Cluster Boundary Test) that consists of following operations:

- For each observation identify a cluster (c) in the profile whose centroid is closest to this data point.

- Check if observation to closest centroid distance is less than cluster boundary for closest cluster indicating confusion/error.
- If the observation is outside cluster boundary it is considered different from profile indicating the profile is accurate.

6.5.8 Discussion of results

During descriptive statistical analysis (see section 6.3) we compared mobile device usage data of our users through descriptive statistical analysis technique. Although we operated on a subset of our data, the analysis of the comparison gave a clear indication that users' had consistent and unique application usage patterns and the contextual information (time and location). We applied this test on three pairs of users and identified the unique features that should make their profiles distinct.

Based on the results of the descriptive statistical analysis we carried out another test to acquire statistical accuracy of our simulated implementation of Machine Learning Processor using the same set of feature vectors as descriptive statistical analysis and then evaluate the hypothesis made in previous test (see section 6.4) based on profile accuracy scores is this statistical test. In this last test we utilised all feature vectors extracted from mobile devices that resulted into much bigger data set. Nine tests were performed to determine the accuracy of each profile. And the Table 6.30 shows the results of these test.

Table 6.30 Profile accuracy results

Profile	Test 1	Test 2	Test 3	Test 4	Test 5	Test 6	Test 7	Test 8	Test 9	Test 10
1		99.48	91.96	96.3	99.8	17.16	56.78	86.73	71.52	46.85
2	98.83		99.21	94.94	93.55	99.96	66.38	0.2	13.4	99.31
3	84.83	100		99.92	100	90.44	3.32	94.05	91.99	94.64
4	95.89	1.91	96.96		28.01	99.79	1.86	3.05	15.08	97.99
5	98.32	99.41	98.87	82.09		99.96	97.62	97.54	86.68	98.86
6	1.25	99.64	98.42	85.75	99.83		92.04	98.74	70.65	8.59
7	0.16	100	91.37	42.94	100	5.02		0.22	16.51	44.32
8	97.02	99.59	99.25	99.04	99.71	99.89	93.87		67.42	98.73
9	55.17	97.5	98.38	30.98	99.61	98.83	1.69	0.32		95.66
10	85.57	90.65	95.16	92.34	92.16	96.8	96.13	95.93	78.89	

Table 6.31 Average Accuracy

Profile	Average Accuracy
1	74.06
2	73.98
3	84.35
4	48.95
5	95.48
6	72.77
7	44.5
8	94.95
9	64.24
10	91.52

Based on the average of nine tests, the profile accuracy range was between 44.50% and 95.48% and 7 out of 10 users profiles were more than 70% accurate. These results are although very promising but slightly lower in accuracy as compared to previous test (see section 6.4) . Nevertheless, these results support the basic premise of our idea that usage statistics can be modelled into profiles that can efficiently decimate genuine users from imposters. In the forthcoming sections we will analyse the results of this test against the previous test. Since the previous test randomly used six users, we will compare the results of those user pairs.

6.5.9 Comparison of user 1 and user 8

The Table 6.32 shows the profile accuracy figures for users 1 and 8 from our dataset. Both users had very low number of clusters compared to other users, which indicate a small number distinct usage patterns without any sudden variations and one off extreme events. As shown during previous tests discussed in sections 6.3 and 6.4 both of these users had very distinct calling, data and application usage patterns which should contribute towards having different profiles. Since SMS sending pattern was almost similar and not a well utilised feature it should contribute towards discriminating these users' profiles. Lastly the only difference in their mobility profile was that user 1 was

Table 6.32 User 1 and 8 profile comparison

User	Number of Clusters	Accuracy
1	9	86.73
8	12	97.02

completely static with average distance of 7.067 miles and compared to a mobility range of 2.99 to 6.94 miles for user 8.

Table 6.33 Cluster boundary comparison

Clusters (k)	User 1	User 8
1	35.16	41.42
2	171.46	11.43
3	51.62	47.64
4	50.27	110.91
5	55.42	18.68
6	63.35	37.9
7	78.17	15.41
8	3038	26.47
9	1663	263.23
10		26.32
11		664.35
12		108.61

In this test both users have higher number of cluster i.e., 9 and 12 compared to 3 and 2 in the previous test. The cluster boundaries has also expanded which is result of bigger dataset (see Table 6.33). Interestingly, in comparison to each other, user 1's profile is 86.73% accurate down from 100% in MLP test 1 (section 6.4). While the second user who had one of the least accurate profiles in second test now has much increased accuracy of 97.02%. This seems to be not only because of bigger dataset but also because of 10 more clusters.

6.5.10 Comparison of user 3 and user 4

As shown during previous tests discussed in sections 6.3 and 6.4, both of these users had very distinct calling, SMS, data consumption, application usage, and system load and mobility patterns. An interesting intra feature difference between the two was that user 3 only utilised signification amount of mobile data services while on the move where user 4's data consumption was consistent throughout the day. As a consequence of all these differences their profiles should be unique with high probability.

Table 6.34 User 3 and 4 profile comparison

User	Number of Clusters	Accuracy
3	6	99.92
4	8	96.96

Observing the MDS plots of full feature set for these users supports our hypothesis that their profiles are highly distinct. This hypothesis is further validated by nearly 100% accuracy figures for both users. Also it should be noted that both users have similar number of clusters, indicating more compact usage patterns compared to previous test. The Table 6.34 shows the profile accuracy figures for users 3 and 4.

6.5.11 Comparison of user 8 and user 9

User 8 has very small number of clusters compared to user 9, which indicate a small number distinct usage patterns for user 8. As shown during previous tests discussed in sections 6.3 and 6.4, both of these users had very distinct calling behaviour whereby user 8 talk time amounts to only 5% of that of user 9. User 9 had a one off In

Table 6.35 User 8 and 9 profile comparison

User	Number of Clusters	Accuracy
8	12	67.42
9	7	0.32

comparison to MLP test 1 (see section 6.4), user 8 has 10 more cluster whereas the profile for user 9 now have 5 less clusters. The number of clusters and accuracy figures shown in the Table 6.35 prove our hypothesis that the main reason behind poor accuracy figure for user 9 lies in less number of clusters. The MDS plot for user 8 and 9 also supports our hypothesis as the data points for user 9 appear to be more spread out as compared to user 8, yet the profile has 5 less clusters.

6.6 Summary

A series of three tests were performed to validate the accuracy of our behaviour profiling framework. In the first test, descriptive statistical analysis was performed on the one week usage data from our dataset. The feature vector was limited to calls, sms, data consumption, application usage, system load and mobility patterns. Six users were randomly compared based on individual features and the analysis confirmed that these user had different usage patterns over one week period. Next, we tested the profiling accuracy machine learning processor using simulation in Matlab. Based on the average of nine tests per user, the profile accuracy range was between 73.41% and 100% and 8 out of 10 users profiles were more than 95% accurate. Building on the footsteps of these convincing results we performed the last test on the entire dataset. In this test, MDS was performed to visually analyse normalised usage data, silhouette tests was utilised to determine optimum number of clusters and then machine learning processor was applied for profile generation and testing. Based on the average of nine tests, the profile accuracy range was between 44.50% and 95.48% and 7 out of 10 users profiles were more than 70% accurate. These results were slightly lower in accuracy as compared to MLP test 1. Lower number of clusters, outliers in the dataset and extra features may be accounted for lower accuracy in the last test. Nevertheless, these results support the basic premise of our research idea, that usage statistics can be modelled into profiles that can efficiently decimate genuine users from imposters. This research didn't strive to identify best classification/profiling algorithm but instead its aim was to investigate the application of machine learning to profile users based on their behaviour which has been successfully proven by these tests.

Chapter 7

Conclusion

This chapter summaries the thesis by reviewing the project's main contributions and originality and subsequently outline the limitations. The chapter then highlights future research directions and recommendations pertinent to the mobile device security field.

7.1 Contributions of the research

Largely, the project has accomplished all the goals initially set out in Chapter 1, with a succession of investigational studies and simulations undertaken for the development of a machine learning based behaviour profiling technique. The full achievements are:

- A comprehensive exploration of the current security challenges that mobile devices experience (Chapter 2). By reviewing the complex environment that mobile devices operate in, the sensitive information that they retain, the numerous security threats that lead to misuse and the lack of security that is presented, the need for a robust security mechanism which offer continuous protection is identified.
- A detailed review of user authentication techniques for mobile devices (Chapter 3). By presenting a background for biometric authentication mechanisms with an

emphasis given to those which have the potential of providing security on mobile devices. Contributions of previous behavioural based mobile security projects for both host based authentication and network based IDS have been considered to determine the scope and requirement for reviewing the practicability of utilising a behaviour profiling technique within the mobile host environment.

- The proposal and achievement of an innovative mobile security system by utilising the behaviour profiling technique based on Machine Learning (Chapter 4 and 5). The proposed architecture works on the basic premise that mobile users have unique usage patterns, which can be modelled into profiles. By comparing subsequent usage data with the profile, anomalies arising from physical misuse can be detected. The usage profile is created by utilising the various features extracted from mobile device in real time. These features include both application usage data and contextual information like user mobility patterns and timing of operations. The system can work in three modes: as a stand-alone anomaly detection system or its Machine Learning Processor can be used as risk scoring engine for other IDS or TAS for providing enhanced mobile security.

- An analysis and evaluation of the behaviour profiling technique through mobile user application usage events and contextual information (Chapter 6). A series of test were conducted on real life mobile device usage from the dataset accumulated from our testers. The first test, utilised the descriptive statistical analysis method, the feature vector comprising of calling and messaging behaviour, data traffic consumption, mobility and system load demonstrated the potential for a successful behaviour profiling classification. This was then evaluated by employing more complex test implemented in a simulated environment using Matlab. In these test, a number of configurations of the proposed system were evaluated by utilising our dataset. System performance was calculated for both legitimate users and imposters based up their device usage features. By utilising Observation interval of an hour for one week usage data with eight features average accuracy range between 73.41% and 100% whereby 8 out 10 user profiles were more than 95% accurate. The second simulation utilised complete dataset and achieved average accuracy of 44.50% to 95.48%.

- In comparison with previous behaviour profiling systems, the result of our research is within a level of acceptance and in some case far better. This successfully demonstrates the Machine Learning based behaviour profiling technique has potential to verify mobile user via their application usages with associated contextual information.

A number of posters and presentations related to the research project have been made and published in a conference (see Appendix). As a result it is deemed that the research has made positive contributions to the mobile security domain and especially in the field behaviour profiling based anomaly detection.

7.2 Limitations

Despite the research objectives having been met, a number of limitations related with the project can be identified. The key limitations of the research are summarised below.

- The dataset was collected from students and staff at London Metropolitan University during the period of 2012 and 2013. Due to privacy concerns from the participants, the feature set didn't include application usage data at a granular level. Although, we only utilised cell tower based geo-location but many users didn't enable location access feature. Similarly name or type of applications were not collected which limited the ability to enhance our feature set.

- Our feature extractor application gave more control to the users to manually send the usage statistics as a results which resulted in many user failing to contribute to the collection of dataset. Some users installed the feature extractor application on their secondary devices like tablets which were left unused for long periods of time hence they didn't qualify for the testing because of lack of usage data.

- The chosen dataset contains large amount of information (with up to 14 features and around 5000 observations per device) and it requires significant amount of computing resources to be processed within the Matlab environment. This limitation restrains computational greedy machine learning algorithms (e.g. K-Means clustering) to be employed on mobile devices. Therefore, the system could not be evaluated on the mobile devices in real time. Nevertheless, performance for a behaviour profile technique was obtained through implementation in a simulated environment in Matlab.

- Due to limited resources that were available, only the core functions of the stand-alone mode of the HIDS were evaluated through the our simulated tests, but the performance for the collaborative mode of the Behaviour Profiling system (i.e. working as a plugin with an TAS or IDS system) was un-examined. Given the successful investigation of the stand-alone mode of the Behaviour Profiling system, it was reasoned that when the Machine Learning processor operates as a plugin, it would certainly make a positive contribution towards the overall performance of an authentication or IDS system.

- This aim of this research was to investigate the application of machine learning for the purpose of mobile user behaviour profiling to detect anomalies. It doesn't strive to detect best machine learning technique for this problem. The selection of K-Means clustering algorithm was based on the fact that it has already been used for malware detection using application behaviour profiling, furthermore it is one of the simplest unsupervised machine learning algorithm. Since K-Means clustering algorithm has it own limitations including difficulty in determining the number of clusters, comparing quality of the clusters produced. Additionally, because K-Means assumes spherical clusters but in reality they are not spherical. In essence, it is a heuristic to minimize the variances. As a consequence there may be other machine learning or neural network techniques that can outperform K-Means Clustering.

7.3 Suggestions for future work

This research project has improved the domain of anomaly detection based on behaviour profiling of end users on mobile devices. Nonetheless, there are a number of areas in which future work could be carried out to advance upon what has been achieved in this research. The details of future work are listed below:

- Design a platform independent feature extractor application package similar to our Google Android application. This allow the first requirement to be met to allow the Behaviour Profiling system to be deployed on real mobile devices. It will be necessary that this software has a small footprint without any substantial degradation of mobile.
- Development of Behaviour based profiling prototype on real mobile devices. This will allow for a comprehensive evaluation of behaviour profiling technique on live user application interactions in real time.
- Research in identifying features that contribute positively towards classifying users shall be carried out. This will also help in remove redundant feature that downgraded the user profiling and consume precious processing resources. Also, an evaluation of different Machine Learning and Neural Networks to

identify techniques that can perform better in the resource restraint environment of mobile device.

- Further research and development of the dependent working mode of the Behaviour Profiling system. In particular, cooperating with the TAS authentication system or the KBTA IDS architecture within the mobile host environment. It is predicted that the performance of the TAS system will be improved in a number of scenarios, such as when a user composes a text message, the user's identity is verified based upon their keystroke activities (i.e. original TAS setting) and also to whom and where the message is compiled (i.e. the behaviour profiling technique). The KBTA architecture will also benefit from the behaviour profiling technique, as it will not only be able to detect mobile malware but also user related misuse.

- Further investigation of the data storage for the Behaviour Profiling system. As the simulation system was implemented on a normal PC, data storage was never an issue. With growing number of available storage on mobile devices, storage space itself may not be a problem but issues relating to data retention and privacy concerns need to be accounted for.

7.4 The Future of security for mobile devices

Mobile devices are gaining immense popularity and the developments in technology increase significantly year after year. Every day, more and more people rely on their mobile devices to complete personal and business tasks, from storing private information to accessing corporate emails, from online shopping to using online banking. Most of these activities encompass a certain degree of sensitive and confidential information. With emergence of new mobile services and mind boggling hardware capabilities, such trend will only increase. On the flip side of all this innovation, the impending damage associated to the device is also expanded in case of device misuse. It is envisaged that now user identity verification on mobile device is of paramount importance. Despite many controls that are currently being utilised for verifying users' identity and detecting device misuse, this project has emphasised the need for a robust and reliable security mechanism which has the potential to operate in a continuous and transparent manner to offer both the security and user convenience. This research project has designed and developed a host based novel Behaviour Profiling system capable of providing user friendly anomaly detection of the user. It can also work collaboratively with other security controls (e.g. TAS or KBTA) to offer enhanced security. To conclude, verifying a mobile user's identity will be crucial in the near future due to the financial services the mobile device provides and the sensitive information it carries. These services and information could become the main motivation towards device misuse. In order to provide sufficient security on mobile devices, mechanisms will have to utilise multiple security techniques and operate in a continuous and user friendly fashion.

References

Aaronoff, R. (2009). Communications fraud control association (cfca) announces results of worldwide telecom fraud survey. Accessed from http://www.cfca.org/pdf/survey/2009%20Global%20Fraud%20Loss%20Survey-Press%20Release.pdf on June 13, 2013.

Adomavicius, G. and Tuzhilin, A. (1999). User profiling in personalization applications through rule discovery and validation. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '99, pages 377–381, New York, NY, USA. ACM.

Al-Baker, O., Benlamri, R., and Al-Qayedi, A. (2005). A GPRS-based remote human face identification system for handheld devices. *Audio, Transactions of the IRE Professional Group on*, pages 367–371.

Alexander Gostev (2006a). Mobile malware evolution: An overview, part 1. Accessed from http://securelist.com/analysis/malware-evolution-monthly/36109/mobile-malware-evolution-an-overview-part-1 on Apr 28, 2011.

Alexander Gostev (2006b). Mobile malware evolution: An overview, part 2. Accessed from http://securelist.com/analysis/malware-evolution-monthly/36111/mobile-malware-evolution-an-overview-part-2 on Apr 28, 2011.

Amershi, S. and Conati, C. (2007). Unsupervised and supervised machine learning in user modeling for intelligent learning environments. In *Proceedings of the 12th International Conference on Intelligent User Interfaces*, IUI '07, pages 72–81, New York, NY, USA. ACM.

Apple (2012). About the security content of iOS 3.1.3 and iOS 3.1.3 for iPod touch. Accessed from http://support.apple.com/kb/HT4013 on May 4, 2011.

Aslam, S. (2014). Centroid, radius and diameter of a cluster (for numerical data sets). Accessed from http://slidewiki.org/slide/24030 on December 13, 2014.

Aviv, A. J., Gibson, K., Mossop, E., Blaze, M., and Smith, J. M. (2010). Smudge attacks on smartphone touch screens. In *Proceedings of the 4th USENIX Conference on Offensive Technologies*, WOOT'10, pages 1–7, Berkeley, CA, USA. USENIX Association.

BBC (2005). Mobiles get anti-virus protection. Accessed from http://news.bbc.co.uk/1/hi/technology/4207476.stm on June 13, 2013.

Bolton, R. J. and Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Science*, 17(3):235–255.

Boukerche, A. and Notare, M. S. M. A. (2002). Behavior-based intrusion detection in mobile phone systems. *J. Parallel Distrib. Comput.*, 62(9):1476–1490.

Brener, B. (2011). Mobile security vendor: Droiddream pulling android into botnet army. Accessed from http://www.csoonline.com/article/2134637/data-protection/mobile-security-vendor--droiddream-pulling-android-into-botnet-army.html on Oct 29, 2012.

Bromba (2011). Fingerprint cellphone. Accessed from http://www.bromba.com/protoe.htm#Handy on July 23, 2013.

Buennemeyer, T., Nelson, T., Clagett, L., Dunning, J., Marchany, R., and Tront, J. (2008). Mobile device profiling and intrusion detection using smart batteries. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, page 296.

Bulygin, Y. (2007). Epidemics of mobile worms. *2007 IEEE International Performance Computing and Communications Conference*, (2):475–478.

Burguera, I., Zurutuza, U., and Nadjm-Tehrani, S. (2011). Crowdroid: Behavior-based malware detection system for android. In *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, SPSM '11, pages 15–26, New York, NY, USA. ACM.

Buschkes, R., Kesdogan, D., and Reichl, P. (1998). How to increase security in mobile networks by anomaly detection. In *Computer Security Applications Conference, 1998. Proceedings. 14th Annual*, pages 3–12. IEEE Comput. Society.

Calloway, M. (2012). The rise of mobile. Technical report, Trinity Digital Marketing. Accessed from http://www.trinitydigitalmarketing.com/mobile-on-the-rise-infographic on Sep 2, 2013.

Chebyshev, V. and Unuchek, R. (2014). Mobile Malware Evolution: 2013. Technical report. Accessed from https://www.securelist.com/en/analysis/204792326/Mobile_Malware_Evolution_2013 on Apr 25, 2014.

Christodorescu, M. and Jha, S. (2004). Testing malware detectors. *SIGSOFT Softw. Eng. Notes*, 29(4):34–44.

Cisco (2014). Cisco visual networking index: Global mobile data traffic forecast update, 2013-2018. Technical report, Visual Networking Index (VNI). Accessed from http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html on March 5, 2014.

Clarke, N. (2011). Transparent Techniques. In *Transparent User Authentication*, pages 111–149. Springer London, London.

Clarke, N. and Furnell, S. (2005). Authentication of users on mobile telephones - a survey of attitudes and practices. *Computers & Security*, 24(7):519 – 527.

Clarke, N., Li, F., Papadaki, M., and Dowland, P. (2010). Behaviour profiling on mobile devices. pages 77–82.

Clarke, N., Li, F., Papadaki, M., and Dowland, P. (2011). Misuse detection for mobile devices using behaviour profiling. *International Journal of Cyber Warfare and Terrorism (IJCWT)*, 1(1):41–53.

Clarke, N. L. and Mekala, A. R. (2007). The application of signature recognition to transparent handwriting verification for mobile devices. *Information Management & Computer Security*, 15(3):214–225.

Dagon, D., Martin, T., and Starner, T. (2004). Mobile phones as computing devices: The viruses are coming! *IEEE Pervasive Computing*, 3(4):11–15.

Dancho Danchev (2009). iHacked: jailbroken iPhones compromised, $5 ransom demanded | ZDNet. [Accessed April 28, 2011].

Denis Maslennikov (2011). Mobile malware evolution: An overview, part 4. Accessed from http://securelist.com/large-slider/36350/mobile-malware-evolution-an-overview-part-4 on Apr 28, 2011.

Derawi, M. O., Nickel, C., Bours, P., and Busch, C. (2010). Unobtrusive User-Authentication on Mobile Phones Using Biometric Gait Recognition. *Audio, Transactions of the IRE Professional Group on*, pages 306–311.

Dini, G., Martinelli, F., Saracino, A., and Sgandurra, D. (2012). Madam: A multi-level anomaly detector for android malware. In *Proceedings of the 6th International Conference on Mathematical Methods, Models and Architectures for Computer Network Security: Computer Network Security*, MMM-ACNS'12, pages 240–253, Berlin, Heidelberg. Springer-Verlag.

Dominic Laurie (2009). Mobile phone directory to launch. *BBC*. Accessed from http://news.bbc.co.uk/1/hi/programmes/working_lunch/8091621.stm on April 28, 2011.

Dunn, J. E. (2011). Ca discovers fake antivirus smartphone app. Technical report, CSO Online. Accessed from http://news.techworld.com/mobile-wireless/3276755/ca-discovers-fake-antivirus-smartphone-app on May 15, 2012.

El-Bakry, H. (2010). Fast virus detection by using high speed time delay neural networks. *Journal in Computer Virology*, 6(2):115–122.

Furnell, S. M., Clarke, N. L., and Karatzouni, S. (2007). Nica design specifications. Accessed from http://www.cscan.org/nica on June 6, 2013.

Gordon S. L, M. J. B. (2011). *Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management*. Wiley.

Gosset, P. (1998). Aspect: Fraud detection concepts: Final report. *Researchgate*. Doc Ref. AC095/VOD/W22/ DS/P/18/1.

Hall, J., Barbeau, M., and Kranakis, E. (2005). Anomaly-based intrusion detection using mobility profiles of public transportation users. *Audio, Transactions of the IRE Professional Group on*, 2:17–12.

Hilas, C. S. and Mastorocostas, P. A. (2008). An application of supervised and unsupervised learning approaches to telecommunications fraud detection. *Knowledge-Based Systems*, 21(7):721 – 726.

Ho, Y. L. and Heng, S.-H. (2009). Mobile and ubiquitous malware. In *Proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia*, MoMM 09, pages 559–563, New York, NY, USA. ACM.

Jacob, G., Debar, H., and Filiol, E. (2008). Behavioral detection of malware: from a survey towards an established taxonomy. *Journal in Computer Virology*, 4(3):251–266.

Jain, A. K., Duin, R. P. W., and Mao, J. (2000). Statistical pattern recognition: a review. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(1):4–37.

Koong, K. S., Liu, L. C., Bai, S., and Lin, B. (2008). Identity theft in the USA: evidence from 2002 to 2006. *Int. J. Mob. Commun.*, 6(2):199–216.

Kurkovsky, S. and Syta, E. (2010). Digital natives and mobile phones: A survey of practices and attitudes about privacy and security. In *Technology and Society (ISTAS), 2010 IEEE International Symposium on*, pages 441–449.

Lane, T. and Brodley, C. E. (1997). An application of machine learning to anomaly detection. In *In Proceedings of the 20th National Information Systems Security Conference*, pages 366–380.

Leavitt, N. (2005). Mobile phones: The next frontier for hackers. *IEEE Computer Society*, 38(4):20–23.

Lerouge, E., Moreau, Y., Verrelst, H., Vandewalle, J., Stoermann, C., Gosset, P., and Burge, P. (1999). Detection and management of fraud in umts networks. *Third International Conference on The Practical Application of Knowledge Discovery and Data Mining (PADD99)*, pages 127–148.

Leyden, J. (2014). Apple's new iphone 6 vulnerable to last year's touchid fingerprint hack. Accessed from http://www.theregister.co.uk/2014/09/23/iphone_6_still_vulnerable_to_touchid_fingerprint_hack on Oct 2, 2014.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In Le Cam, L. M. and Neyman, J., editors, *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability - Vol. 1*, pages 281–297. University of California Press, Berkeley, CA, USA.

Majeed, K. (2012). Investigation leading to behaviour based hybrid intrusion detection system for mobile devices. *Psychology of Programming Interest Group - Doctorial Consortium*, page 6.

Majeed, K., Jing, Y., Novakovic, D., and Ouazzane, K. (2014). Behaviour based anomaly detection for smartphones using machine learning algorithm. *International conference on Computer Science and Information Systems (ICSIS'2014) Dubai (UAE)*, pages 67–73.

Manikopoulos, C. and Papavassiliou, S. (2002). Network intrusion and fault detection: a statistical anomaly approach. *Communications Magazine, IEEE*, 40(10):76–82.

McCue, A. (2004). Mobile mosquito premium rate sms trojan not a virus. Accessed from http://www.zdnet.com/mobile-mosquito-premium-rate-sms-trojan-not-a-virus-3040146005/ on May 4, 2011.

McGraw, K. L. and Harbison-Briggs, K. (1989). *Knowledge Acquisition: Principles and Guidelines*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Miettinen, M. and Halonen, P. (2006). Host based intrusion detection for advanced mobile devices. In *Proceedings of the 20th International Conference on Advanced Information Networking and Applications*, volume 2.

Mikko Hypponen (2006). Malware goes mobile: Scientific american. *Scientific American*, (11):70–77. [Accessed April 28, 2011].

Milanesi, C., Tay, L., Cozza, R., Atwal, R., Nguyen, T. H., Tsai, T., Zimmermann, A., and Lu, C. (2013). *Dataquest Market Insights: Computing Hardware Worldwide*, (5):1–6.

Morales, J., Clarke, P., Deng, Y., and Golam Kibria, B. (2006). Testing and evaluating virus detectors for handheld devices. *Journal in Computer Virology*, 2(2):135–147.

Moreau, Y., Verrelst, H., and Vandewalle, J. (1997). Detection of mobile phone fraud using supervised neural networks: A first prototype. *Artificial Neural Networks — ICANN'97*, 1327(Chapter 170):1065–1070.

Muir, J. (2003). Decoding Mobile Device Security. *ComputerWorld*.

Mullins, J. (2007). Cellphone firewall. Accessed from http://www.newscientist.com/blog/invention/2007/04/cellphone-firewall.html on June 13, 2013.

Nathan Eagle, A. S. P. (2006). Reality mining: sensing complex social systems. In *Personal and Ubiquitous Computing Journal*, volume 10, pages 255 – 268.

Neel, J. (2010). Emerging wireless standards 2010. Accessed from http://www.crtwireless.com/EWS_10.html on July 22,2014.

Nixon, J. (2011). AVG ireland - AVG technologies presents global q1-2011 security threat report. [Accessed April 26, 2011].

Oberheide, J., Cooke, E., and Jahanian, F. (2008). Cloudav: N version antivirus in the network cloud. In *Proceedings of the 17th conference on Security symposium*, Security Symposium 08, pages 91–106, Berkeley, CA, USA. USENIX Association.

Oh, S. H. and Lee, W. S. (2003). An anomaly intrusion detection method by clustering normal user behavior. *Computers & Security*, 22(7):596 – 612.

Peikari, C. (2006). Analyzing the crossover virus: The first pc to windows handheld crossinfector. In *Maximum Wireless Security*. Sams. Accessed from http://www.informit.com/articles/article.aspx?p=458169 on May 4, 2011.

PewResearch (2013). Mobile technology fact sheet. Accessed from http://www.pewinternet.org/fact-sheets/mobile-technology-fact-sheet on Jan 11, 2014.

Piech, C. and Ng, A. (2013). K-means algorithm. Accessed from http://stanford.edu/~cpiech/cs221/handouts/kmeans.html on December 13, 2014.

Piercy, M. (2004). Embedded devices next on the virus target list. *Electronic Systems and Software*, 2(6):42–43.

Porras, P., Saïdi, H., and Yegneswaran, V. (2010). An analysis of the ikee.b iphone botnet. In *Security and Privacy in Mobile Information and Communication Systems*, volume 47 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 141–152. Springer Berlin Heidelberg.

Rao, J. R., Rohatgi, P., Scherzer, H., and Tinguely, S. (2002). Partitioning attacks: or how to rapidly clone some GSM cards. In *Security and Privacy, 2002. Proceedings. 2002 IEEE Symposium on*, pages 31–41. IEEE Comput. Soc.

Rastogi, V., Chen, Y., and Jiang, X. (2013). Droidchameleon: Evaluating android anti-malware against transformation attacks. In *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ASIA CCS '13, pages 329–334, New York, NY, USA. ACM.

Reinhardt A. Botha, Steven M. Furnell, N. L. C. (2009). From desktop to mobile: Examining the security experience. *Computers & Security*, 28(3–4):130 – 137.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20(0):53–65.

Saevanee, H., Clarke, N., and Furnell, S. (2011). Sms linguistic profiling authentication on mobile device. pages 224–228.

Samfat, D. and Molva, R. (1997). Idamn: An intrusion detection architecture for mobile networks. *IEEE Journal on Selected Areas in Communications*, 15(7):1373–1380.

Sanou, B. (2012). Itu measuring the information society. Technical report, International Telecommunication Union. Accessed from http://www.itu.int/en/ITU-D/Statistics/Documents/publications/mis2012/MIS2012_without_Annex_4.pdf on July 07, 2013.

Schmidt, A.-D., Peters, F., Lamour, F., and Albayrak, S. (2007). Monitoring smartphones for anomaly detection. In *Proceedings of the 1st International Conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*, MOBILWARE '08, pages 40:1–6.

Schneier, B. (2014). Details of Apple's Fingerprint Recognition. Accessed from https://www.schneier.com/blog/archives/2014/04/details_of_appl.html on July 23, 2013.

Shabtai, A., Kanonov, U., and Elovici, Y. (2010). Intrusion detection for mobile devices using the knowledge-based, temporal abstraction method. *Journal of Systems and Software*, 83(8):1524–1537.

Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C., and Weiss, Y. (2012). "andromaly": A behavioral malware detection framework for android devices. *J. Intell. Inf. Syst.*, 38(1):161–190.

Shih, D., Lin, B., Chiang, H., and Shih, M. (2008). Security aspects of mobile phone virus: a critical survey. *Industrial Management & Data Systems*, 108(4):478–494.

Shih, D.-H., Chiang, H.-S., and Yen, C. D. (2005). Classification methods in the detection of new malicious emails. *Information Sciences*, 172(1-2):241–261.

Stajano, F. and Anderson, R. J. (1999). *The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks*. Springer-Verlag.

Sun, B., Chen, Z., Wang, R., and Yu, F. (2006). Towards adaptive anomaly detection in cellular mobile networks. *... and Networking ...*.

Sun, B., Yu, F., Wu, K., and Leung, V. C. M. (2004). *Mobility-based anomaly detection in cellular mobile networks*. ACM, New York, New York, USA.

Torsten (2011). Matlab newsgroup - variance of a cluster. Accessed from http://uk.mathworks.com/matlabcentral/newsreader/view_thread/304464 on December 13, 2014.

Wagner, D. T., Rice, A., and Beresford, A. R. (2014). Device analyzer: Large-scale mobile data collection. *SIGMETRICS Perform. Eval. Rev.*, 41(4):53–56.

Weinstein, E., Ho, P., Heisele, B., Poggio, T., Steele, K., and Agarwal, A. (2002). Handheld Face Identification Technology in a Pervasive Computing Environment. In *Proceedings of the First International Conference on Pervasive Computing (Short paper)*, pages 48–54, Zurich.

Welch, M. (2012). Goode intellegence msecurity survey report. Accessed from http://www.goodeintelligence.com/media/media_centre/1334824915gi_msecurity_survey_news_release-final_190412.pdf on June 3, 2013.

Wikipedia (2014). Haversine formula wikipedia, the free encyclopedia. Accessed from http://en.wikipedia.org/w/index.php?title=Haversine_formula&oldid=612829004 on June 28, 2014.

Wolpin, S. (2014). The First Cellphone Went on Sale 30 Years Ago for $4,000. *American Heritage of Invention Technology*.

Woo, R. H., Park, A., and Hazen, T. J. (2006). The MIT Mobile Device Speaker Verification Corpus: Data Collection and Preliminary Experiments. *Audio, Transactions of the IRE Professional Group on*, pages 1–6.

Wu, J. W. J. and Dai, F. D. F. (2004). Mobility management and its applications in efficient broadcasting in mobile ad hoc networks. *IEEE INFOCOM. Proceedings*, 1:3–50.

Yap, T. S. and Ewe, H. T. (2005). A mobile phone malicious software detection model with behavior checker. HSI 05, pages 57–65, Berlin, Heidelberg. Springer-Verlag.

Zester, L. (2011). The use of social engineering by mobile device malware. Accessed from http://blog.zeltser.com/post/7641621785/social-engineering-by-mobile-device-malware on Aug 2, 2013.

Zhang, Y., Xia, P., Luo, J., Ling, Z., Liu, B., and Fu, X. (2012). Fingerprint attack against touch-enabled devices. In *SPSM '12: Proceedings of the second ACM workshop on Security and privacy in smartphones and mobile devices*, pages 57–68, New York, New York, USA. ACM Request Permissions.

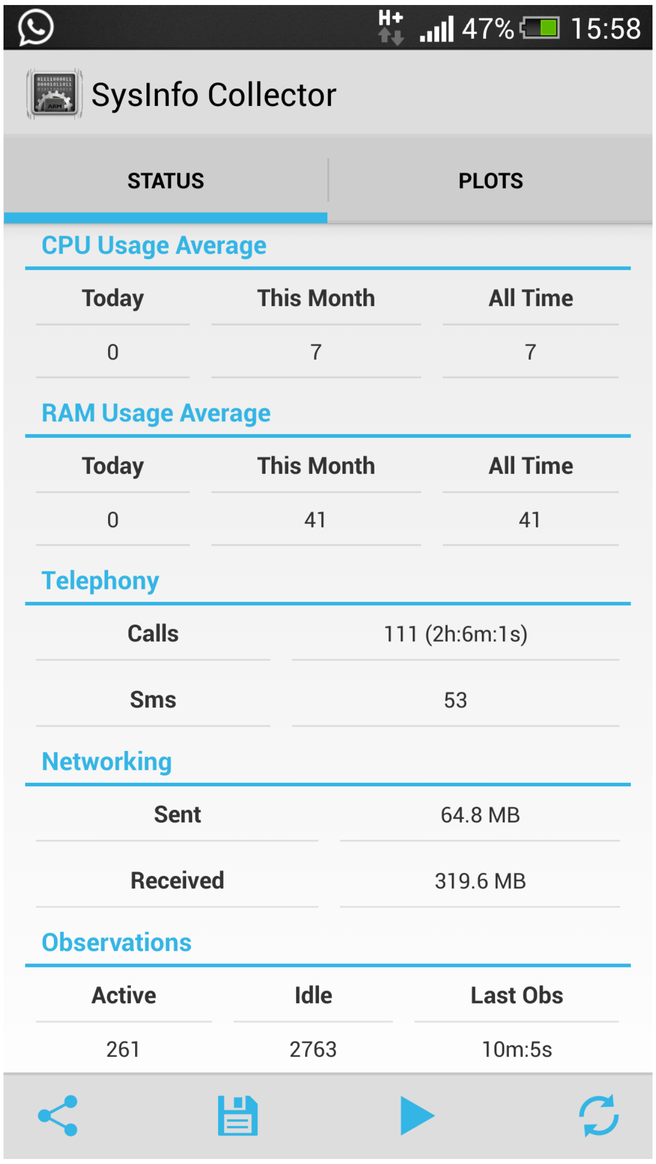
Zhao, M., Ge, F., Zhang, T., and Yuan, Z. (2011). Antimaldroid: An efficient svm-based malware detection framework for android. In Liu, C., Chang, J., and Yang, A., editors, *Information Computing and Applications*, volume 243 of *Communications in Computer and Information Science*, pages 158–166. Springer Berlin Heidelberg.

Zheng, M., Lee, P. P. C., and Lui, J. C. S. (2013). Adam: An automatic and extensible platform to stress test android anti-virus systems. In *Proceedings of the 9th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, DIMVA'12, pages 82–101, Berlin, Heidelberg. Springer-Verlag.

Appendix A

A.1 Feature extractor application

The GUI feature extractor application we developed for Google Android based smartphones only consists of two screens. It doesn't require any user interaction other than to start the feature extraction process by starting the background service and sending the collected data to remote administration via email option. The figure A.1 shows the first screen where the application user can perform these actions. in addition the suer can also view their usage statistics in numerical form. While the figure A.2 shows the second screen that shows graphical view of the same usage statistics.



CPU Usage Average

Today	This Month	All Time
0	7	7

RAM Usage Average

Today	This Month	All Time
0	41	41

Telephony

Calls	111 (2h:6m:1s)
Sms	53

Networking

Sent	64.8 MB
Received	319.6 MB

Observations

Active	Idle	Last Obs
261	2763	10m:5s

Fig. A.1 Application main screen to view statistics and access settings

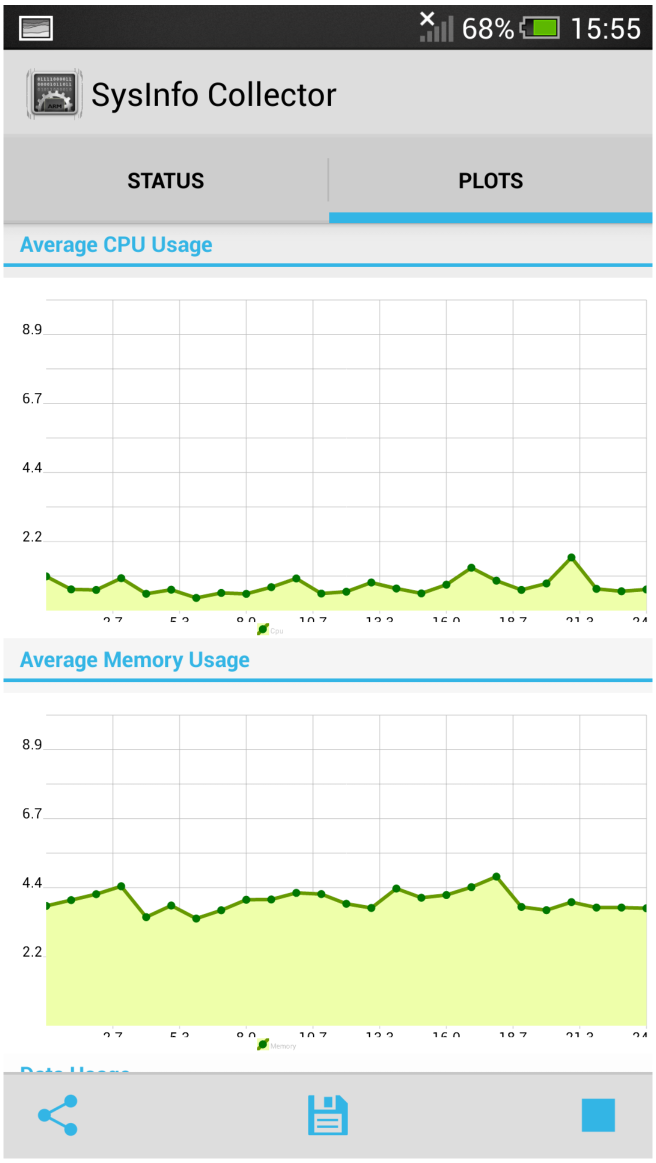


Fig. A.2 Viewing device usage data as graphs

Appendix B

B.1 Publications

- Several poster presentations were made at London Metropolitan University and other events.
- An oral presentation was made and a conference paper was submitted at Psychology of Programming Interest Group - Doctorial Consortium (Majeed, 2012).
- Another conference paper was published at the International conference on Computer Science and Information Systems (ICSIS'2014) (Majeed et al., 2014).