How we effectively manage virtualization in Mobile Cloud Computing (MCC) ?

Amrapali S. Chavan

Submitted as part of the requirements for the degree
of MSc in Cloud Computing
at the School of Computing,
National College of Ireland
Dublin, Ireland.

January 2016

Supervisor Dr. Anu Sahni.

Submission of Thesis to Norma Smurfit Library, National College of Ireland

Student name: Miss. Amrapali S. Chavan. Student number: X14123762

School: School of Computing Course: MSc in Cloud Computing

Degree to be awarded: MSc in Cloud Computing

Title of Thesis: How We Effectively Manage Virtualization In Mobile Cloud Computing (MCC)?

One hard bound copy of your thesis will be lodged in the Norma Smurfit Library and will be available for consultation. The electronic copy will be accessible in TRAP (http://trap.ncirl.ie/), the National College of Ireland's Institutional Repository. In accordance with normal academic library practice all theses lodged in the National College of Ireland Institutional Repository (TRAP) are made available on open access.

I agree to a hard bound copy of my thesis being available for consultation in the library. I also agree to an electronic copy of my thesis being made publicly available on the National College of Ireland's Institutional Repository TRAP.

Signature of Candidate: Amrapali S. Chavan.

For completion by the School:
The aforementioned thesis was received by_____ Date: 25/01/2016

This signed form must be appended to all hard bound and electronic copies of your thesis submitted to your school

Submission of Thesis and Dissertation

National College of Ireland

Research Students Declaration Form

(Thesis/Author Declaration Form)

Name: Miss. Amrapali S. Chavan Student Number: X14123762

Degree for which thesis is submitted: MSc in Cloud Computing

Material submitted for award

(a) I declare that the work has been composed by myself.

(b) I declare that all verbatim extracts contained in the thesis have been distinguished by quotation marks and the sources of information specifically acknowledged.

(c) My thesis will be included in electronic format in the College Institutional Repository TRAP (thesis reports and projects)

(d) ***Either*** *I declare that no material contained in the thesis has been used in any other submission for an academic award.

Or *I declare that the following material contained in the thesis formed part of a submission for the award of

Master of Science in cloud computing awarded by QQI at level 9 on the National Framework of Qualification

(State the award and the awarding body and list the material below)

Signature of research student: Amrapali S. Chavan.

Date: 25th January 2016.

Contents

List of Figures

Abstract

Virtualization technology is used to manage computing resources in smart phones as well as desktops. This paper analyses the influence of virtualization on Operating Systems (OSs) of smart phone devices. Virtualization could be on hardware level or software level of smart phone device. Since Android Operating System was introduced, the smart phone users were rapidly increasing. In addition, we discuss mobile security problem can solve by encapsulating original smart phone operating system (OS) by the virtual machine. Virtualization can help us to solve issues of storage, bandwidth, battery and computation resources. In our study we measure the performance of Android OS and Tizen OS in virtualize platform and compare it with Native platform. Virtualization technique works as a security control mechanism for smart phones which offers performance efficiency and protection against mobile threats. We create virtual machines for Android and Tizen OS. We used Samsung Galaxy S 5 smart phone, to run the Android LINPACK test. In this paper we discussed the performance analysis between Android and Tizen operating systems. This analysis is helpful to address the problems in Bring Your Own Device (BYOD).

Key Points- Virtualization, Android, Tizen, Samsung Galaxy S5 , Samsung Gear S, LINPAC.

Acknowledgment

Foremost, I would like to express my sincere gratitude to my supervisor, Dr. Anu Sahni for her patience, motivation and sharing immense knowledge. She helped me to successfully completing my research work and this thesis in all possible ways. I am thankful for her efforts, valuable help, discussions and motivation. In each meeting with her, she encouraged me in all ways and increase my confidence level for this study. She always gives me better aspects of my work and implementation.

Besides my supervisor, I would like to thank Dr. Pramod Pathak, Dr. Horacio Gonzalez-Velez, Robert Duncan, IT support department and all NCI staff for their valuable guidance and help in the completion of my course. I would like to thank for NCI online resources to enhance my writing and referencing skills. I am thank full to all people who support me all the time directly and indirectly throughout this year of MSc Cloud Computing.

I would like to thank my classmates of cloud computing and friends in other course also, who always been motivated me in difficult time. Their motivation has given me energy to concentrate on my work.

I express my sincere love and obligation to my parents: Shivajirao Chavan, Sushila Chavan, Sudarshan Chavan and Karan Chavan for their continuous love, motivation and encouragement. This work is especially dedicated to mother and father because without their blessings, I couldn't reach to completion.

Chapter 1

Introduction

In Bring Your Own Device (BYOD) trend, many opportunities are available for mobile virtualization. There is no single dedicated solution for BYOD, it offers a huge business in multiple ways from a mobility device to the network. Enterprise data security is a critical issue in BYOD. All smart phones are pre-installed with their own operating systems like iOS, Android and windows. Even in IT industry multiple mobile users have a lack of awareness about mobile security.

Security softwares for mobiles are specially customized for each mobile operating system version. Extra security tools for mobiles causes computational difficulties and power problems. Mobile virtualization is a long-term solution that gives, right sort of support to current Android smart phones. Virtualization technique is one key solution for BYOD issues, where we can separate storage space and data for e.g VMware Horizon Mobile.

- **Heavy duty VM**

Multiple OSs can be installed on the same phone so these OS instances are affected on a computational capacity, device power and storage space. Vendors by phone may not supports any other OS on the same phone.

Cagalaban et al. (2012) tackled about some issues like real time support, resource allocation, power consumption and security. These issues are critical to mobile phone. So it is needed to detect new approaches to mobile virtualization. New Operating Systems (OS) like Tizen have not yet taken into consideration for virtualization by any researches. Tizen is newest operating system introduced by Linux foundation, Tizen association, Intel and Samsung.

1.1 Hypothesis

Many researchers have been studied, different techniques of mobile virtualization. A distributed computing model, that enables smart mobile devices to access different services of cloud datacenters is called (Mobile Cloud Computing) MCC. To solve BYOD problems, we will use three main key parts- cloud, Mobile Device OS and Virtualization. Today there is no dedicated virtualization solution for the BYOD use case. It is needed to create a virtualization technique for the system software environment and resolve the complexity between hardware and software. This research question:

"How we effectively manage virtualization in MCC ?"

To solve BYOD problems such as security and isolation, we measure the performance of the Android OS and Tizen OS. This experiment is performed on virtual environment and native machine. Our virtual phone solutions depend on hardware backing for devoted execution.

In this paper, we will focus on a specific case of mobile virtualization. New Operating Systems (OS) like Tizen have not yet taken into consideration for virtualization by any researches. For detail experiment, we will use the virtual set up of Tizen OS and Android OS. As well as, we use native Android and Tizen devices to compare their performance with virtual machines. We focus, to measure ability of Tizen and Android devices to in virtual environment.

1.2 Contribution

As discussed, mobile virtualization solution addressed the BYOD related problems. In IT industry multiple mobile devices were used by employees. Tizen is a new OS, were initially released in January 2013. Like Android OS, Tizen also based on Linux kernel and GNU C library. Tizen based mobile virtualization work is not done yet by any researcher. Our performance comparison, will help to point out the behaviour of Android and Tizen OS in a virtual environment.

Hence we design a comparable model, for new generation mobiles and for future mobile phones. It will identify OS comparison with LINPACK benchmark, to find out the behaviour of Android and Tizen based devices. Our project is becoming helpful, to isolate car management domain i.e. business domain and individual domain.

Chapter 2

Literature Review

This chapter gives the literature review regarding BYOD technology. In section 2.1 we give a detailed review about Cloud Computing and Mobile. In section 2.2 we will discuss about, what is need of virtualization technology in Cloud Computing? Section 2.3 gives the previous work of BYOD platform and how previous researchers were addressed and approaches problems of BYOD. In section 2.4, we focused, smart phone device and server virtualization technology. In section 2.5, we discussed how live migration is possible with virtualization. Section 2.6 described, how BYOD works in real time. The total overview of BYOD gives us an idea about how it works in enterprise sector. we also talk about the benefits of BYOD in today's IT world. It also highlights that, today's growing market of smart phones are needed new virtualization techniques. In section 2.6, we described Android architecture in detail. The detailed overview of Tizen architecture is given in section 2.8. Also, in section 2.9 a lot of importance should be given to comparison of Android and Tizen application system. We have discussed about current trends about new mobile display virtualization technology. New Operating Systems (OS) like Tizen have not yet taken into consideration for virtualization by many researches.

2.1 Cloud Computing and Mobile

Cloud computing (CC) has grown rapidly in the past few years. Buyya et al. (2009) states in his research that, over the last half century computing has become the fifth important utility after water, electric power, gas and telecommunication system. According to National Institute of Standards and Technology (NIST) cloud computing reference model Liu et al. (2011) categorized the three basic service models, through

which service providers can offer services. These service models are: Software as a Service (SaaS), Infrastructure as a Service (IaaS and Platform as a Service (PaaS). CC can provide services on demand basis by using on demand self-service and utility business computing models. Buyya et al. (2009) argues that, now it is easy to run the Virtual Machine (VM) in microprocessor and software level. The VM can isolate the applications from the underlying hardware and VM is also used to allocate physical resources on user demand.

Xing et al. (2012) confer that, one key feature of CC is virtualization, which makes it possible to run different operating systems and applications over the same machine or set of machines. Virtualization is more useful when user will understand resource management issues and security problems before migrating into the cloud. With Mobile Cloud Computing (MCC) research Xing et al. (2012) mention that, we can move data storage and data processing mobile devices to the cloud. MCC interconnects, geographically distributed users all over the world using shared resources. Current service providers did not pay enough attention, resources provisioning diversity. It includes the type of guest OS and virtual hardware configuration (Xing et al., 2012). Armbrust et al. (2010) mention the main factor about cloud service providers, that provides different services on demand basis. He also states that, to expand computing potential of resources we can use MCC effectively. (Xing et al., 2012) and Armbrust et al. (2010) agree on MCC, that it allows mobile devices to provide services, applications and resources worldwide. By expanding storage and computing power of data centers, we can achieve virtualization goal in MCC.

Durairaj and Manimaran (2014) research for virtualization puts MCC in the next higher level in cloud computing. Study of Durairaj and Manimaran (2014) proves that, virtualization on a mobile device would mean just one smart phone with virtual partitions, so people could use it for both work and their personal lives. According to Buckley (2012) while adopting virtualization in MCC, the mobile users and enterprise markets for mobile cloud based applications will increase rapidly. The virtualization technique creates the environment for system software and resolve the complexity between hardware and software. Durairaj and Manimaran (2014) perform an experiment to prove that, virtualization can effectively manage using Hypervisor. He states that, the Hypervisor is a software program that can virtualizes system resources. Durairaj and Manimaran (2014) confer about emulation technique also. Durairaj and Manimaran (2014) further explain that, emulation is a virtualization technique which used to translate the hardware program into the software program. Emulation provides flexibility to guest OS, but processing speed is slow as compare to Hypervisor.

Durairaj and Manimaran (2014) explain virtualization technique in his study. Paravirtualization is one of the virtualization technique in which guest OS is recompiled prior to install inside the virtual machine. Full virtualization is a virtualization that guest OS doesn't know about virtualize environment and therefore hardware is virtualized by the host OS (Durairaj and Manimaran, 2014). Liang and Yu (2015) discussed about wireless virtualization in MCC . Liang and Yu (2015) mention that, wireless virtualization in which cost of network deployment and operation is reduced by sharing computing resources. It is a process of abstracting, slicing, isolating and sharing mobile resources.

For large scales Buckley (2012) discussed about virtualization solutions that can eliminate the problem of data loss in corporate sectors. It can happen to storing all data and applications on a server and simply giving users a virtual window access to these resources. Charland and Leroux (2011) claim that, Apple changed our mobile experiences with the iPhone, but its difficult for them to develop different apps for each platform separately. Charland and Leroux (2011) further assert that, for gaming and image processing apps Apple faced some performance penalty with well-developed business applications and developing a new application for each mobile platform is cost effective.

In an important study on the VMware mobile virtualization platform by Barr et al. (2010), he provides a solution for Type 2 Hypervisors. Mobile Virtual Platform (MVP) hypervisor is mainly used for BYOD concept in corporate sectors. Barr et al. (2010) believe that, to design hypervisor for mobile virtualization the main goals are portability, compatibility, security, low complexity, performance and manageability. ARMv7 core technology used for all mobiles, Instruction set architecture (ISA) and memory are the two main approaches used in ARM core virtualization technology (Barr et al., 2010).

Mobile Cloud Computing

According to Shiraz et al. (2013) MCC architecture has three major components; smartphone, Internet and computational cloud. Shiraz et al. (2013) claims that, virtualization is a technique to create virtual instances of a device or resource, such as a compute server, storage disks, network or an operating system. Shiraz et al. (2013) further describe about the services that, compute service is responsible to aggregate the server resources across many discrete servers and assign them to applications. Storage service is of technologies that enable the most efficient and management of storage in virtual environment. Network service simplifies and enhance networking in virtual environments.

Hung et al. (2011) justify that, CC technology helps to solve problems like a resource sharing, storage, computing power, energy consumption in IT industry. But still some

problems are puzzling like: application redesign and deployment, Service availability and privacy of personal data. According to National Institute of Standards and Technology (NIST) model Liu et al. (2011) and Zhang et al. (2010), CC provides On-demand self-service so resources can be dynamically added and manage on the network. In todays world, people widely used mobile devices and the number is increasing day by day because of different applications and techniques being used in mobile device(Kemp et al., 2012). Schüring (2011) point out some research study related to MCC like open issues and possible solutions, services and communication network used by smart phones and addressed different solutions associated with MCC.

2.2 Need to Virtualization

In recent study Xu et al. (2010) claims that, the mobile phones connected to the Internet for downloading files, images, audio and video. Users personal information like emails, contacts, debit or credit card numbers are residing into the mobile device, so mobile phone become need to be more secure. Xu et al. (2010) confer that, IBMs VM/370 was a first commercial Virtual machine (VM) developed in the 1960s. VM is a copy of real time system and Virtual Machine Monitor (VMM) is used to control the resources of VM. Xu et al. (2010) further mention that, Mobile phone virtualization requirements are quite different from system virtualization. Virtualization techniques for high performance system are not applicable for mobile devices because of different hardware resources and power efficiency. So Xu et al. (2010)suggest that, for mobile phones, high performance virtualization solutions are preferred. The main goal of virtualization is to utilize the resources like storage, network and processor in minimum cost of performing multiple tasks simultaneously.

Durairaj and Manimaran (2014) discuss about emulation technique in his research. They state that, emulation is one of the virtualization technique where the guest OS is lying over the hypervisor and converts hardware to the software layer. Durairaj and Manimaran (2014) further categorize three types of virtualization techniques, such as Server virtualization, Client Virtualization and virtualization of storage. In Server Virtualization one server can be virtualized into multiple servers across the multiple environment.

Hypervisor allows the server to manage different applications locally and remotely. Durairaj and Manimaran (2014) point out that, using Server virtualization we can achieve cost saving, high availability and resource sharing. Durairaj and Manimaran (2014) further describe that, in Client Virtualization we can virtually monitor different client machines such as a laptop, desktop system and mobile phones. Durairaj and

Manimaran (2014) gives the idea behind Storage Virtualization, is to create logical storage system from actual physical storage.

Machine to Machine Communication (M2M)

On different computing technologies Internet, wireless technology, personal computer and mobile devices are able to create machine to machine communication (M2M). Cagalaban et al. (2012) states that,by virtualizing software architecture of a mobile phone we can reduce on device CPU and memory resources. Companies like Google are starting mobile virtualization for reuse of software and hardware and for improving host security. Using virtualization in the mobile phone can run multiple OS on the same hardware like legacy OS in televisions and entertainment systems.

In Mobile virtualization study, citetcagalaban2012mobile focuses on migration of security services to the cloud detection service. This architecture contains a mobile device with a virtual machine on it that sends les to the network for security analysis. This architecture could be deployed by the cloud service provider(Cagalaban et al., 2012). M2M study of Cagalaban et al. (2012) tackled about some issues like real time support, resource allocation, power consumption and security. These issues are critical to mobile phone. So it is needed to detect new approaches to virtualization.

2.3 Related Work

In recent studies for Mobile virtualization Oh et al. (2010) suggest full virtualization for Advanced RISC Machine (ARM) mobile systems, where multiple OSs running on single mobile system at a same time. Oh et al. (2010) discuss about two technologies used for virtualization: full-virtualization and paravirtualization. Oh et al. (2010) state that, in full virtualization it is not required to make changes in the source code of a guest OS. The OS can run on VM directly and guest OS does not even realise it (e.g. VMware ESX, Xen).

In case of paravirtualization Oh et al. (2010) state that, it is necessary to modify source code of the guest OS by humans or tools. The main modification is made in system call interfaces, memory and interrupt handling. Oh et al. (2010) confer the main advantage of paravirtualization is a High Performance (e.g. L4Linux). With Virtualization for Mobile (ViMo) architecture Oh et al. (2010) describe the full virtual mobile system based on ARM. This virtualization technique creates, one virtual machine per each operating system and the OS runs on that virtual machine. According to experimental results carried out by Oh et al. (2010) in ViMo has around 37 percent overhead. So

it is essential to improve performance of all components, including CPU and memory using better virtualization algorithms in future.

According to Andrus et al. (2011) virtualization is a relatively new concept for various mobile and embedded devices. In study of Open Kernel Lab 4 (OKL4) Microvisor by Andrus et al. (2011) state that, it is actually a bare metal Hypervisor which is useful only for smaller computing operations. A solution is given in Cell architecture of Andrus et al. (2011), but it has high overhead of virtual phone switching. VMware and MVP are settled virtualization solutions in Android device that runs on recent hardware. Barr et al. (2010) claims that, unlike OKL4 its trusted virtualization base is more compatible with both Android user environment and host Linux OS.

In Virtual Ad hoc Network (VAN) system to virtualize mobile devices paravitualization is used by Xen hypervisor. Mobile Network Layer and Mobile Link Layer are used to support the mobility in nodes (Poylisher et al., 2010). In L4Android study by Lange et al. (2011) they used Nitpicker component for virtualization, but it does not offer any performance metrics. The simple way to build a virtual environment on mobile phone is used, existing techniques but its difficult to native implementation. The reason behind it is software and hardware related issues, effective use of memory subsystem is one of the solution in mobile virtualization (Lee and Hsueh, 2013).

2.4 Memory Management for Virtualization

Memory Management Unit (MMU) plays an important role in mobile virtualization, nested paging in which address translation become possible from different virtual machines (Bhargava et al., 2008). But according to a study of Lee and Hsueh (2013) MMU is not suitable for mobile virtualization because it takes long latency for address translations. Optimized page translation (oPT) is another method developed to solve this problem. oPT can minimize a number of memory accesses to more than 50 percent, sustained due to address translation. Virtual Machine Monitor (VMM) is a layer to manage Virtual Machines (VMs) and every guest OS is running on the separate VM. Guest Physical Address (GPA) is managed by VMM, Block Table bTLB address translation allow 100 percent hit ratio rate in virtualization (Lee and Hsueh, 2013).

2.5 Smart Phone Device and Server Virtualization

A mobile virtualization system is comprised of a mobile device and virtualization server in Figure 2.1. Roh et al. (2014) state that a mobile platform is divided into two parts:

Figure 2.1: Mobile virtualization service (Roh et al., 2014)

Terminal and Server platforms. The Terminal platform is which the operating system, middleware and browser are mounted on a device. It is used to control hardware and user interface (UI). Now a days UI becomes more user friendly, so user can easily access any platform. The Server platform is which authentication, billing, gateway and online marketplace are mounted on the server.

Roh et al. (2014) also mention that, mobile device and virtualization server, possibly-connect by Wi-Fi, 3G and LTE. Newly evolved light-weighted platform technology used to provide high quality web application services to different terminal platforms. It includes content virtualization without any hardware dependency. Light weight devices as mobile have minimum codecs and sensors. They only decode the data from servers. An android-x86 virtual machine present in a virtualization server based cloud system, receives events from mobile devices and send encoded data to mobile devices.

In a Server platform Android software structure works as shown in Figure 2.2 (Roh et al., 2014). The Linux kernel is located above a hardware layer and at the bottom of the Android system. The Android applications and C/C++ libraries are located above the kernel layer in the system. In absence of physical GPS android x86 is unable to identify values transmitted by the device so, we need a virtual path here. Roh et al. (2014) confers that, driver virtualization technology is divided into two types: Virtual device driver and Hardware Acceptance Layer (HAL). Virtual device drivers divided into two types again: Virtual sensor driver and GPS driver. The virtual sensor driver is used to transfer sensor values to android framework and GPS driver transmitted GPS data towards the Hardware Acceptance Layer (HAL).

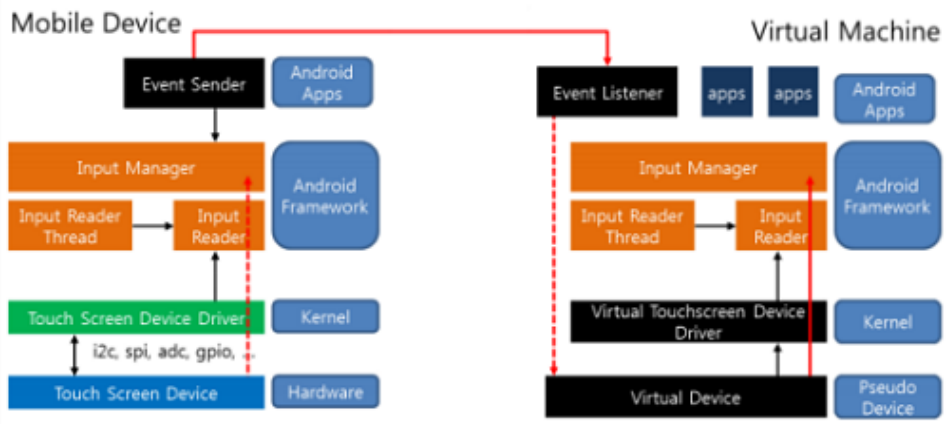

Figure 2.2: Device events management (Roh et al., 2014)

In study about, interface library layer Roh et al. (2014) discussed virtualization system transfers, where the event values are nothing but GPS and sensor data to the android application layer. Open Graphics Library (OpenGL) is used when the system needs hardware acceleration. OpenGL is an interface that is connected to host OS using virtual box as a Type 2 Hypervisor and it will increase performance of virtually execution window.

2.6 Live Migration with virtualization

Hung et al. (2011) stress that, resources for computing such as data storage, network bandwidth and battery capacity are the main issues in MCC. It is essential to develop models for migrating applications and synchronising data between execution environments. Hung et al. (2011) further conclude that, application offloading is beneficial in MCC where application workload is offloaded onto the server machines to save execution time and conserve energy. Application re-design and deployment, network condition and service availability, are the puzzling issues in current mobile cloud services. To solve these issues a virtualized execution framework is used where application re-build is not needed. A user may run existing application

on physical device or virtual environment. But the communication cost of migrating a process of mobile applications is prohibitively expensive (Hung et al., 2011). Collaborative Computing in that mobile devices and cloud servers work collaboratively and being aware of quality-of-service (QoS). Where each of the virtual network devices provides a specific QoS guaranteed communication channel. In QoS guaranteed communication

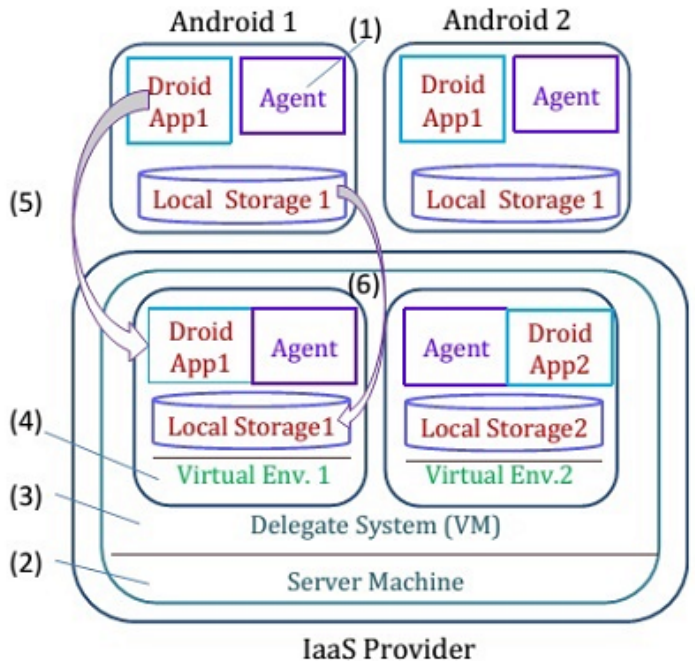

Figure 2.3: Creating a virtual environment (Hung et al., 2011)

framework the communication messages can be sent over a low bandwidth channel with high transmission probability (Hung et al., 2011).

In cloud virtual system Figure 2.3. Hung et al. (2011) states that we can run a set of different applications on the mobile device. The user has the choice to either use mobile device or virtual environment to run applications or he can migrate applications between two virtual environments. This framework is develop for Android users, it automatically creates and migrate applications in virtual environment. For this framework developer need not to rebuild their applications. This framework will placed on the server machine of IaaS provider.

Hung et al. (2011) described six steps, to create virtual environment: First, Install and run user program, it provides an interface between user and applications to cooperate with the virtual environment. Second, the agent allocates delegate system virtual environment to host using the virtual machine from IaaS provider. In third step Hung et al. (2011) state that, Agent sets up virtual environment for virtual phone on the delegate system to affect an android phone.

Hung et al. (2011) further defines that, virtual phone need to be more compatible with android device. Fourth, using image stored in the delegate system agent creates a new virtual environment. The agent then copies data and applications from physical phone to virtual phone. Exact copy of the environment will more compatible applications.

Fifth, when a user send command to the physical machine agent then he established communication with agent in virtual machine (Hung et al., 2011).

Hung et al. (2011) also define that,p hysical agent takes control of all operations in a virtual environment. The user sends a request to agent to migrate applications. Sixth, both agents on physical and virtual phone now synchronizes the applications and keeps user data steady and coherent on both phones. But this framework has some issues like: to copy physical environment and create a virtual one, minimize the time required to migrate applications, minimize the costs for sharing data and secure virtual environment (Hung et al., 2011). To cope with these issues we need to build more secure framework in mobile virtualization.

2.7 What is BYOD?

Bring Your Own Devices (BYOD) is a new clause where, employees in IT industry can use their personal computing devices specially smart phones at their work places. According to Buckley (2012), employees can access enterprise resources using their smart phones. How effective virtualization is used in smart phones will depend on company policy and employees. In opinion of Buckley (2012), network access control is important methodology used in many enterprises for monitoring. In today's smart phone era mobiles are widely used as computing device rather than desktops, PCs and laptops. All new generation android smart phones were fully equipped with touch screen, Graphical processing unit (GPU), Wi-Fi, Bluetooth and Global Positioning System (GPS).

Buckley (2012) confers that, BYOD is very use full in enterprises to test their new applications on employees computing devices before implementing in the real world. Bring Your Own Device(BYOD) is a phrase that has become widely adopted to refer to employees in IT industry who brings their own computing device such as smartphones, laptops and PDAs to the workplace for use. The main idea is to provide mobile access to corporate resources. For supervision activity authentication and Network Access Control (NAC) is used. Network Access Control (NAC) and authentication techniques are used for supervision process in enterprise sectors. IT companies and their employees can effectively work on mobile virtualization technology. Data loss and data security are major issues in BYOD. But although there are clear security issues to consider, Buckley (2012) states that cloud, mobile device management (MDM) and virtualization will effectively use for BYOD.

Buckley (2012) notified problem that, there will be a possibility of data loss if the user access corporate account outside the virtualize environment. It is necessary to provide document level security and it needs to mitigate separately. BYOD concept is also called as Virtual Phones because a user is able to use enterprise domains and personal domain on the same mobile device. So BYOD is depending on security as well as isolation concern and mobile virtualization is the solution for this (Carabas et al., 2014).

Embedded Virtualization Platform design for a BYOD where corporate's important information must be secured from unauthorised employee access. Traditional server virtualization solutions are not suitable for new mobile technology as it is more user friendly and power consuming. Today there is no dedicated virtualization solution for BYOD use case Dong et al. (2015). For example OKL4 (Open Kernel Lab) Microvisor is expensive in context switching, Bromium Microvisor contained only single isolated VM to run all applications. The OS has high priority so it runs in Kernel or Supervisor mode and all user applications run in User mode Lange et al. (2011). In case of Open Source, Kernel Virtual Machine (KVM) is available only for Linux 3.9 onward versions, Xen 4.3 onward versions are available for ARM. In OS modification container technology is used, one good example is Cell. The Cell can implement multiple Android virtual mobile phones on a single mobile device (Andrus et al., 2011).

Benifits of BYOD

- It helps to improve the productivity of employees.

- Developing good relations with clients.

- Increased attention of employees to work.

- Virtualization in BYOD can helps to use enterprise domain and personal domain on the same phone.

2.8 Android Framework

We have been using Android 5.1 i.e. Lollipop for our setup. Figure 2.4 shows the architectural overview of the Android OS. The Android OS is basically invented for mobile devices. The Architecture contains four layers.

1. Linux Kernel: It consists of a kernel with Linux 2.6 version. Kernel offers built in services like camera driver, WI-Fi driver, Display driver, keypad driver, flash, audio driver and power memory (Brahler, 2010).

Figure 2.4: Android Architecture (*Android ArchitectureKernel Description*, 2013)

2. Libraries and Android Runtime: The second green layer is sets of C/C++ libraries for system components. The kernel uses bluez for Bluetooth functioning and the wpa supplicant to encrypt WI-Fi. As mobile devices are with less power and CPUs native code is used for CPU and graphical processes. Kernel libc and libm libraries are specially built for small memory and licensing problems with Android. Surface Manager acts as the window manager and manage successes to screen. Media Framework contains video and audio features. Android Runtime layer is placed in Android kernel and acts as host for Dalvik VM and core libraries of Java. Dalvik VM interprets the Java byte code into byte code of Dalvik VM (Brahler, 2010).

3. Application Framework: As per Android developer guide, this layer provides a development platform for Android development and communicate and manage applications in the top layer. Brahler (2010) notifies that, blue colour contents in this layer are written in Java language. The Dalvik Virtual machine is responsible to run this Java code.

4. Android Applications: Each Dalvik virtual machine is sandboxed and apps are running into it. Applications are consisted of various contents like service, activity, content provider and receiver for broadcast (Brahler, 2010).

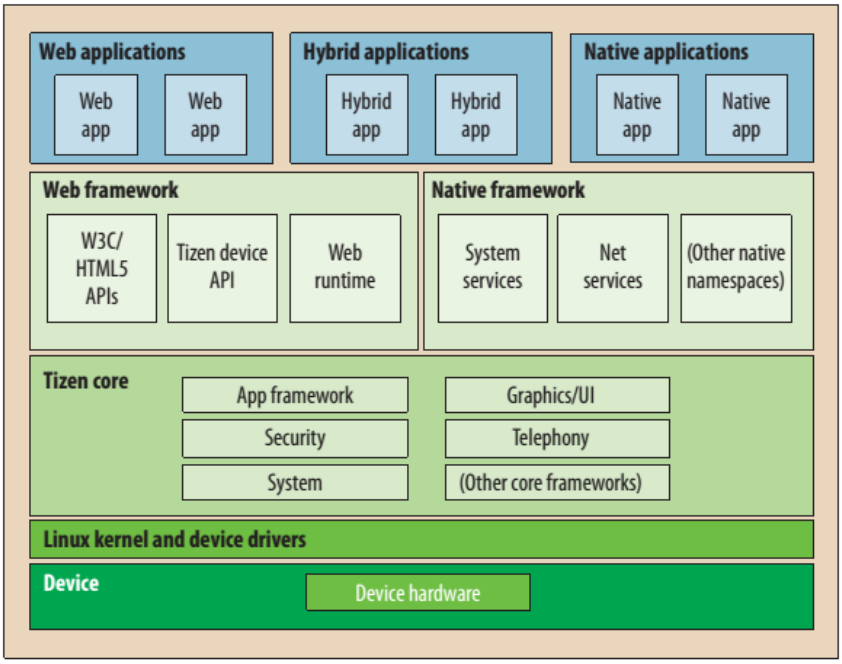

Figure 2.5: Architecture of Tizen OS (Gadyatskaya et al., 2014)

2.9 Emerging Tizen

New mobile operating systems like Tizen or Firefox OS are invented on Android basis and tries to overcome limitations of Android. Like Android Tizen is also Linux-based operating system, invented in collaboration with Fujitsu, Huawei, Samsung and Intel. Initially, it was only used for web apps, but Tizen 2.0 onwards versions are able to run C++ applications. Tizen is now become the competitor of Android and share the market segment of Android phones. In figure 2.5, Gadyatskaya et al. (2014) gives a detail view of architecture of Tizen OS. The Linux kernel is the located with the device drivers, on the underlying hardware of the system. The next layer is Tizen core, offers functionalities like App framework, GUI, Security, System and telephony. The third layer is about Web framework and Native framework. Web framework contains W3C, Tizen API and web runtime. Native framework at present with functionalities like System services, Net services and Namespaces. The top most layer of Tizen OS is nothing but user applications including Web, Hybrid and Native.

Hoy (2011) states that, unlike Android, Tizen framework does not include VM strategy. It gives vast application packaging system, SDK for web framework which offers HTML5. Tizen acquires device profile technology, which offers a platform for different types of devices (Willis, 2014). Tizen also make available for multiple hardware

components and cross device platform technology like mobile devices, In-Vehicle infotainment (IVI) and cameras (*Samsung's NX300M smart camera is its first to run Tizen OS*, 2013). Tizen provides special packages for wearable electronic devices like watches (Prabhakaran, 2014). SQLite have been used in both Android and Tizen platform through fsync() system call. Both OSs used default file system EXT4 by journaling the files in Ordered Mode (Kim and Kim, 2012).

The Tizen documentation describes in detail on its official website *Tizen Web Guides* (2012) the guide for developers. Overall Tizen architecture provides secure framework with various utilities and system tools like secure shell client and rpm package manager. Tizen Hybrid Apps is a combination of the two frameworks. HTML5 are used to develop high level Application Programming Interfaces (APIs). Native Applications are developed for system services using built in programming libraries in C++.

Tizen supports mobile web apps through its web browser. Tizen App store able to publish package applications as well as hosted applications. Packaged Apps are accessing the API of mobile devices as they are in category of standard web app. On the other side Native Apps are used as service Apps or User Interface(UI). The difference between UI and Service App is that, the UI is for graphical interface and Service Apps can run on device background. Sandboxing is the most important feature in Tizen OS that offers security to Tizen Apps. This security is called application sandboxing, implemented in the OS kernel with MAC.

Asrar and irfan (sept 2014) from Intel security group defines Tizen core layer in detail which includes following services such as:

• App Framework : It is also called AppCore, offers services for middle-level and system hardware. The framework is interconnected with the Linux kernel to handle hardware calls, web apps and APIs. App Framework manages the life cycle of Apps and responsible to launch them. It handles the system events, configures applications and able to install or uninstall applications.

• Base: It is foundation of system libraries in Linux.

• Graphic/UI: In includes in Native framework and provide graphic facility and User Interface (UI).

• Security: Provides security to the entire system through certificates,

• Connectivity: This feature used to interconnect within the network.

• PIM : Offers features like contacts, calendar etc.

- Location: Offers GPS location with satellite metadata and geocoding.
- Messaging : Offers all communication styles like SMS, email, chat, MMS, etc.
- Multimedia: Supports feature like audio, video, images etc.

Tizen Sandboxing

Gadyatskaya et al. (2014) described that, to sandbox the application root and app are the two IDs for the user. SMACK (Simplified Mandatory Access Control Kernel) is used for Linux security purposes with MAC and allow the sandboxing at kernel level. The process is marked with unique label along with its resources. SMACK is responsible for resource allocation to processes. Application file in sandbox, is labelled with 10 character unique key SMACK64EXEC attribute in ext file. This label is used as default label and will be used for all resources assigned to Apps.

Gadyatskaya et al. (2014) further state that, when user installs Apps package manager allocates the unique label to each file. In installation process SMACK rules apply to system to offer authorized access over resources. Smalley and Craig (2013) stress about Tizen's built in security model, that contains application sandboxes and resource access control. It will help us to provide better security in BYOD environment, by isolating applications at a kernel ground with smack. Smack security is basically three domain model used to control and manage access resources. It delivers user domain, system domain and floor domain for user, system and public data processing activities.

Package manager sends a request to SMACK to grant permission for accessing resources. Using this grant applications are able to use SMACK objects and transfer them into security checking. SMACK sandboxes are used for all development applications. Smalley and Craig (2013) states that, in web application installation process Tizen make AppID for SMACK rule and build a soft link for client in Web Runtime. The soft link will run with the application and then execution will happen in the sandbox. This sandboxing framework is different from Android, where developer uses Linux DAC (Dictionary Access Control) model and not a MAC.

2.10 Comparision of Android and Tizen Application System

Gadyatskaya et al. (2014) compares Tizen applications with Android applications and make a statement that, like Android Apps Tizen Apps able to communicate directly.

Tizen applications are made, their functions available publicly to use for other application and this feature is similar to Android. AppManager is maintaining records of all App functions through AppControl technique. Gadyatskaya et al. (2014) compare Tizen AppControl technique with Androids Intent Technique. Like Intent Technique of Andoird, AppControl has two types: Explicit control and Implicit Control. In explicit control application uses AppID, which is unique to each App to call other App and in Android App system package name is used for calling. The second is Implicit control, in this type request will be processed using MIME, URI or operation ID whereas, in Android implicit intent resolution is used.

In Tizen there is one default implicit control for calling an application, in Android default implicit intent is set to call the main activity function of an application. Like content providers in Android, Tizen have mechanism of data control and message ports to interconnect applications and established communication among them. Tizen Developers also able to establish personal communication among Tizen applications. Such type of application is with signed privacy certificate and user cannot export its functionality. Android Apps uses signature permission protection with fine grained policy that application developers can moderate access to selective components of applications. (Kim et al., 2015).

According to previous research by Kim and Kim (2012) EXT4 file system in Android uses asynchronous journal to commit to return the fsync() system call. When this call returns before completion of write operation of the commit. The checksum field of journal commits, is use to check records of journaling of the file system. On other side EXT4 in Tizen does not use asynchronous commit for journaling.

Permissions

Tizen privileges and request to use the API are same as Android permission granting process. SMACK rule in Tizen is nothing but granted permission during installation. Tizen have two types of privileges: Public and platform. In the public privilege list of contract is available to all Apps. Platform level is also called as Partner privileges, they are only available for trusted and registered partners with company in Tizen store. This privilege includes package manager and available for authorised Tizen consortium developer. Platform level privilege is defined in the application with authorised signature. Dissimilar to Android Tizen Apps have two signature one is for author and developer, second is for marketing, distributors and manufacturers Gadyatskaya et al. (2014).

• Additional security in Tizen : Tizen uses Content Security Policy (CSP) to control the web content sources. It also restricts the navigation of webapp to various domains

listed in ¡tizen:allownavigat ion¿ tag of the manifest file. Against plagiarism Tizen provides encryption service, developers can on it.

• Platform comparison of Android and Tizen: According to *S.Suzuki* (2013), from Android version Jelly Bean (4.2) it completely supports data execution prevention (DEP) and memory management security feature like address space layout randomization (ASLR). Tizen OS is inherited from Android, and the developer uses native development languages and till 2014 Tizen work is going on DEP and ASLR.

• Sandboxing Comparision: Bugiel et al. (2012) states that, Android has different approach in sandboxing than Tizen. Android utilities Linux DAC (Dictionary Access Control) model with separate UID for every application, whereas Tizen uses unique UID for all applications SMACK rule is uses for security in sandboxing.

Chapter 3

Specification

We developed our experiment on the basis of virtualization theorems of Popek and Goldberg. It states that, For any conventional third-generation computer, an effective Virtual Machine Monitor i.e. hypervisor may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions (Popek and Goldberg, 1974). In mobile virtualization test framework, virtualization constitutes some challenges. The first challenge is virtualization should not be affected on the performance of mobile, otherwise this would results weak quality of the user experience in BYOD environment. We will try to compare the virtualization performance for Android OS and Tizen OS.

There are multiple consistent operations on the mobile device of process switching. Switching the interface from one domain to another domain or switching from one application to another application in same domain requires many logical operations. In this project we only focused on performance comparison of virtualization environment and native environment of the Android OS and Tizen OS. The main reason behind that is because the integral for virtualization needed less memory than guest OS. Guest OS requires large allocation of RAM memory to run efficiently. Finally, the best solution is working with this comparison is, we have used benchmarking tools like LINPACK to measure performance.

In this project we use following sources:

i) We developed the framework to test and evaluate the performance of Android 5.1 (Lollipop) OS and Tizen IVI (In-Vehicle Infotainment) 3.0 on VMware Workstation. We used LINPACK benchmark to measure the performance of the operating system and processor in the form of number of Floating Point Operations Per Second (FLOPS).

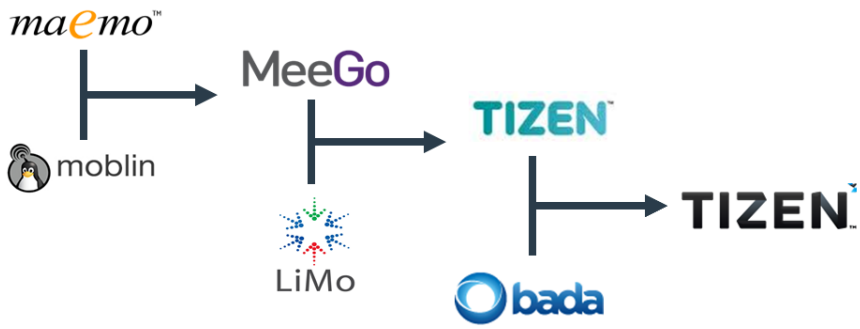

Figure 3.1: Evolution of Tizen OS

ii) We have used Samsung Galaxy S 5 mobile to test Native performance of applications using Benchmarking tool GameBench (Huckle and Cleaver, 2012).

iii) To measure performance of Tizen OS in wearable device we used Sumsung Gear S wearable device. To measure performance of applications in watch we used 3D G-Mark benchmarking tool. This tool conducts tests for wearable devices (*3D Mark Technical Guide*, 2015).

iv) We have used, Wayland rendering tool to measure the rendering and graphics performance of Tizen Wayland kernel. This tool gives output of frames per second status.

3.1 Tizen OS

As shown in Fig 3.1, Tizen is a cross-architecture, open source software platform. Tizen is a Linux-based operating system with Linux kernel and the GNU-C library. The software development kit (SDK) allows developers to use HTML5 and related Web technologies to write applications that run on supported devices. It is an associate project Linux Foundation with Samsung and Intel. It was developed in Linux-Kernel and Webkit run time. The user can access source code and modify the program for Tizen OS. Samsung merge Limo project Morita et al. (2007) and Bada project Woyke (2012) into Tizen. Intel associates with Tizen by putting their MeeGo project work into Tizen OS (Grabham, 2010).

3.2 Samsung Gear S Watch

We have used, Samsung Gear Watch to measure frames per second (FPS) performance for native Tizen operating system. This watch contains inbuilt application like contacts, calender, calculator and email. We have used 3D G Mark benchmarking tool, to calculate FPS ststus of native Tizen OS.

Table 3.1: Samsung Gear S Watch Specification.

Technology	GSM/HSPA
Launched	August 2014
SIM	Nano sIM
size	2.0 inches
Resolution	360 x 480 pixels
Display Type	AMOLED Touch screen, 16 M colors
OS	Tizen wearable
CPU	Dual core 1 GHz
Memory	4 GB and 512 MB RAM

We used Samsung Gear S watch to measure Tizen wearable native performance, above table 3.1 gives its specification.

3.3 Samsung Galaxy S 5 specifications

We used Samsung Galaxy mobile to measure native Android performance on phone with GameBench Benchmarking tool. This mobile phone have Android Lollipop operating system as a native one. In our virtual machine setup, we also used Lollipop OS to create virtual instance. Following table 3.2 gives specification of mobile.

Our experiment setup used to check performance of FLOPS rate for Android Native machine and Android VM and compare it with Tizen Wayland VM.

Table 3.2: Samsung Galaxy S5 Mobile Specification.

Technology	GSM/HSPA/LTE
Launched	Febuary 2014
SIM	Micro sIM
size	5.1 inches
Resolution	1080 x 1920 pixels
OS	Android OS Lollipop 5.1
CPU	Quad-core 2.5 GHz Krait 400
Memory	16/32 GB, 2 GB RAM

3.4 LINPACK

LINPACK is a very popular benchmarking tool, to measure system performance in (Floating Point Operation Per Second) FLOPS. LINPACK contains inbuilt FORTRAN subroutine programs to solve linear equations. LINPACK benchmark tool package built with Basic Linear Algebra Libraries (BLAS) for matrix related operations. The main reason to use this tool is to calculate the time and speed to solve real problems (Jack J. Dongarra, 2012). We choose this tool to measure and compare the performance between Android virtual machine on VMware and Android Native on mobile. Also we compare performance between Android VM and Tizen VM.

According to (Jack J. Dongarra, 2012) the virtual machine or native machine, that perform larger number of floating point operations per second is the best machine. In our study we checked FLOPS rating and computational power of system with Android OS and Tizen Os. Our main aim is to find out how these operating system behaves in virtual environment. The performance result of LINPACK test, will helpful to identify isolation capability of the OS in BYOD.

Jack J. Dongarra (2012) explains that, LINPACK benchmark package is contains extra package of linear algebra for numerical calculations. MFLOPS is concerned with Megaflops, that includes millions of critical floating point operations. LINPACK also includes loop unrolling method, it is related to measuring CPU frequency. Loop unrolling is responsible to reduce CPU overhead and increase the system performance. Jack J. Dongarra (2012) also add the vector operations into this tool. Vector tool is used to manage the reuse of data using special algorithm.

LINPACK is suitable for Android and Tizen OS benchmarking, because it's written in FORTRAN language. Using FORTRAN subroutines, Jack Dongarra invented LINPACK benchmark in 1979. Our main focus is, to calculate how fast the virtual machine can solve critical problems like matrix calculation. We have to take care that, the overall performance of the system should not be affected by this benchmarking tool. Virtual machine performance should not be affected by benchmarking techniques. Jack J. Dongarra (2012) point out that, to calculate computing system performance is critical thing and depends on multiple things. From above arguments, it can be proven that, benchmarking tools are best option to calculate the performance of native as well as a virtual machine.

Chapter 4

Design

As we described in chapter 2.7, Bring Your Own Device (BYOD) is now growing trend in IT industry. Mobile devices are important paradigm in BYOD environment. It helps to reduce company's production cost and increases employee's productivity. To solve this question one answer is to run multiple virtual phones on the same mobile device. But the question is how mobile OSs like Android and Tizen will perform in virtualize environment.

Our aim is to measure isolated performance of virtual instances of Android and Tizen on VMware Workstation. Also, we have measured Native performance of Android OS using a mobile device. The mobile device which we have used is Samsung Galaxy S 5. We installed the GameBench tool on the phone to measure the performance of mobile applications like contacts, calculator, video, play music, YouTube and browser. Every virtual instance contains separate memory management unit and OS kernel. So even if one gets failure other will not be affected. To have a secure and trusted OS is also an important requirement of BYOD. We used LINPACK benchmark tool for Android and Tizen VM and also in Native mobile phone to check FLOPS of the system. Our design model will compare the performance test between Native and virtualize platform for Android OS and Tizen OS.

Figure 4.1 shows the comparison module of our experiment. We created a virtual instance of Android 5.1 (Lollipop) as a guest OS on workstation. As well as we created a virtual instance of Tizen Wayland kernel as a guest OS on a workstation. We install LINPACK tool to measure the performance of computing capabilities over these two guest OSs. Here we just benchmark the FLOPS performance on native Android of mobile device and two VMs.

Our proposed comparison module measures the performance and Android and Tizen in

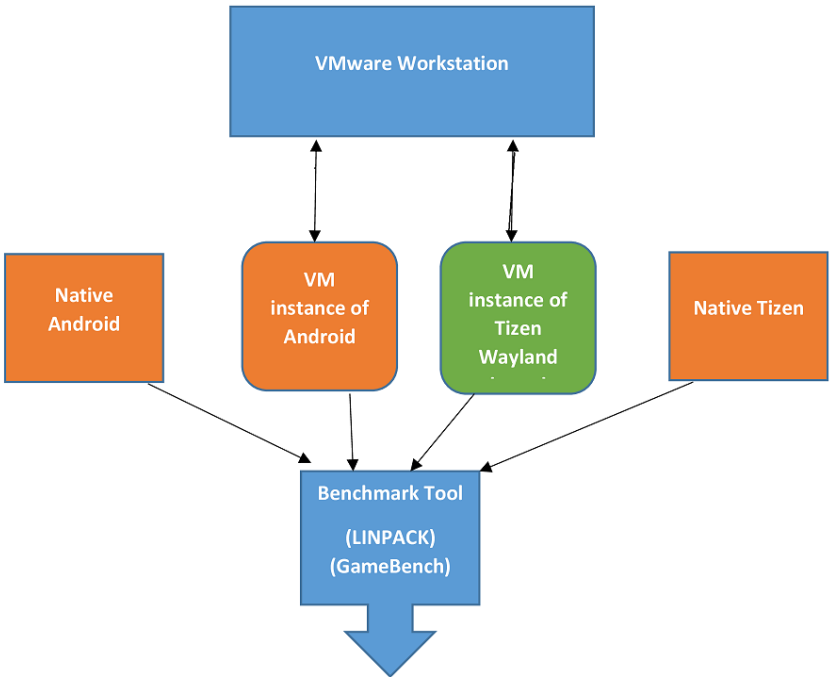

Figure 4.1: Comparison Module

a virtual environment and compare it. This comparison is useful in BYOD framework where mobile virtualization is one of the solution to access personal as well as enterprise domain.

4.1 Performance Criteria

Computing Cost

In cloud computing and virtualization framework, production cost depends on computation cost. Our design will help to identify the operating system and its computational power. We measured this computing power by LINPACK benchmark.

Computing Power

Computing power refers to speed of processing power to solve critical tasks. We can calculate this speed with Floating Point Operations Per Second (FLOPS). Our setup will identify machine that can give higher FLOPS rate.

Graphical Performance In mobile operating systems, graphical performance like video and Graphic Processing Unit (GPU) is measured by Frames Per Second. GPU

used in mobile device have vendor lock-in, so we have limited access over its functioning. To measure 3D performance for mobile games and other video applications FPS tool can be used.

Virtual Machine

Virtual Machine(VM) is software ground implementation of physical computer. The VM can executes and run applications and programs like native machine. VM in a guest operating system and have limited access to underlying hardware. We used Android Lollipop and Tizen Wayland as our guest operating systems.

Hypervisor

Hypervisor is also called as Virtual Machine Monitor (VMM). Hypervisor is responsible to manage all virtual machines. Guest OS can communicate with system hardware through Hypervisor. We have used Type 2 Hypervisor for our set up.

4.2 Benchmark Technique

Benchmarking technique is a method, to measure the performance or compare the computing capabilities of one content. This computing performance contains, standard output values using a special type of indicator. For virtual machines, benchmarking is a special technique that can be implemented easily and used to check problem solving capability. This can be done with a special type of tools like LINPACK, in which critical problem solving tasks can be measured.

4.3 FLOPS comparison

This solution will achieve maximum number of floating point operations and compare them for two VMs: Android and Tizen. Number of FLOPS shows the speed of the system to solve critical problems. It gives test for residual norms for built in mathematical calculations.

4.4 Setup configuration on PC

We used following configuration on physical machine:

- Processor: Intel(R) Core(TM) i5 4288U

- Clock speed: CPU @ 2.60GHz
- Installed Memory RAM: 8.00 GB
- System Type: 64 bit OS, x64-based Processor
- Host OS: Microsoft Windows 8.1
- Virtualization Platform: VMware Workstation 12.0

Chapter 5

Implementation

In this section, we have given the implementation and installation steps of our comparison module. In section 5.2 installation of benchmarking tool LINPACK on Android VM and Tizen Wayland kernel VM. In section 5.3 we calculate and compare the result of FLOPS in native Android, Android VM and Tizen Wayland kernel VM.

5.1 Android virtual instance creation

In this section we will give detail implementation steps of Android VM. We have used Android 5.1 Lollipop version as our first instance of VM. In figure 5.1 we give an overview of Android installation Steps. We have used Android 5.1 Lollipop iso image for virtual machine. This image supports the following file systems:

- ext3
- ext2
- NTFS (New Technology File System)
- FAT 32 (File Allocation Table)

Table 5.1 shows the instance specification of Android VM. Following are the detailed steps to create an Android VM

i) Download an iso image of Android 5.1 from: http://www.android-x86.org/download website

ii) In VMware Workstation, click on "Create New Virtual Machine", upload the downloaded iso image from "Browse" option.

Table 5.1: Instance specification for Android VM

Sr. No	Device	Specification
1	Operating System	Android 5.1
2	Memory	1GB
3	Processors	1
4	Core	1
5	Network Adapter	NAT
6	Cache	512 MB
7	Hard Disk	3.5 GB

iii) Allocate memory 1 GB, CPU 1, HDD 3.5 GB, select Network Adapter NAT . Select the path for disk storage space. In VMware we have to change the virtual disk type to the IDE (Integrated Drive Electronics) from SCSI (Small Computer System Interface) default type. This can be done by changing VM settings.

iv) Power On the VM.

v) We used USB device to boot the Android OS from it. For windows host OS adds following code in Listing 5.1.

vi) An shown in figure 5.2, select option : "Installation− Install Android to hard disk" from selection dialog box. It will boot automatically for few seconds.

```
title Windows 8
    rootnoverify (hd0,0)
    chainloader +1
```

Listing 5.1: Bootable USB for windows

vii) After a few seconds it will show second selection dialog box as shown in fig 5.3 to select a partition to install Android−x86. Select " sda1 Linux VMware Virtual partition".

viii) We coexist Android with another OS. To format sda1 disk, select ext3 file system as shown in figure 5.4. Other file systems like FAT 32 aren't supported Android −x6.

ix) To install Android on hard disk, it is necessary to install boot loader GRUB. The next step will display a question :" Do you want to install boot loader GRUB ?" . Click on "Yes".

x) Android installation will start and it will show progress bar.

xi) After completing the progress bar, it will display a message as in figure 5.5. The Android −x86 will install and we can reboot and run the VM.

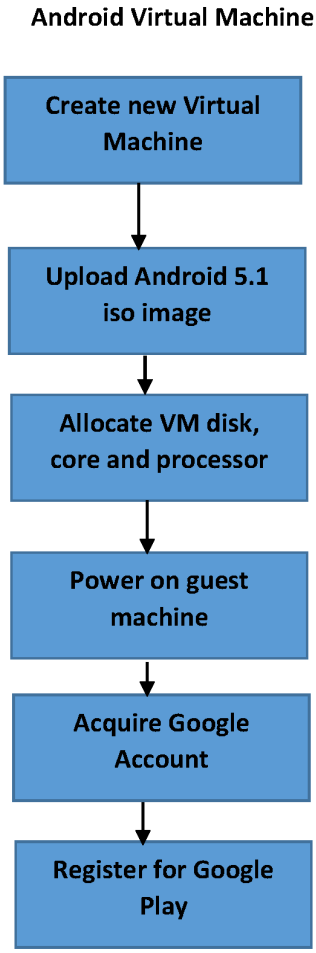

Figure 5.1: Android x86 Virtual Machine Flow

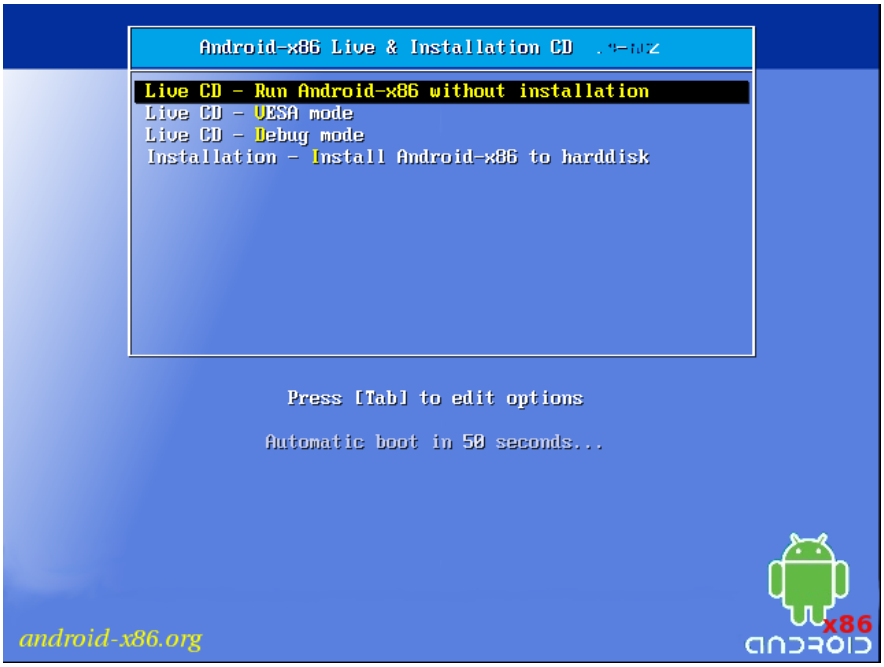

Figure 5.2: Install Android to harddisk

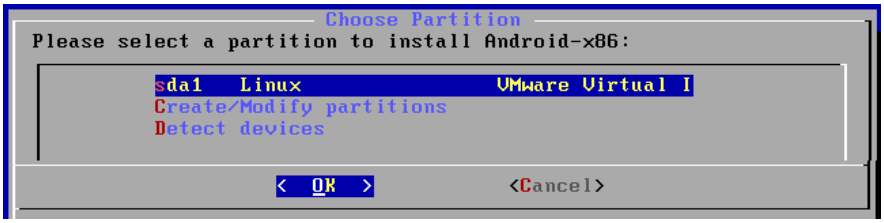

Figure 5.3: Select partition to install Android-x86

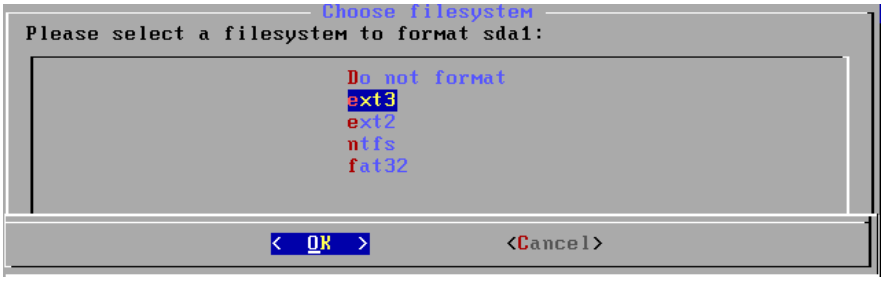

Figure 5.4: Select filesystem to format sda1

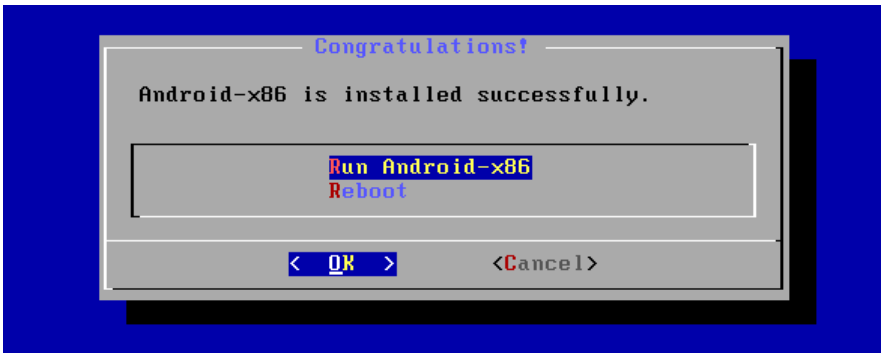

Figure 5.5: Android install successfully

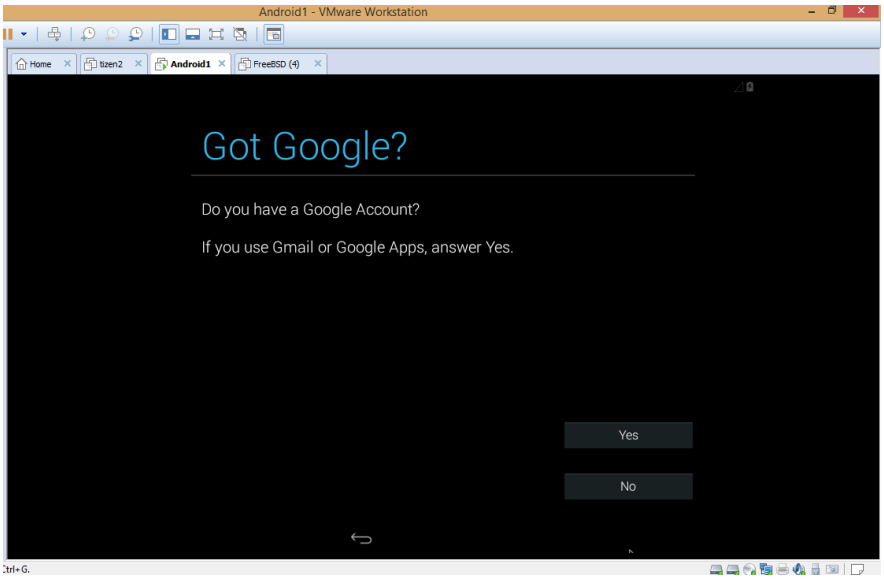

Figure 5.6: Acquire Google account

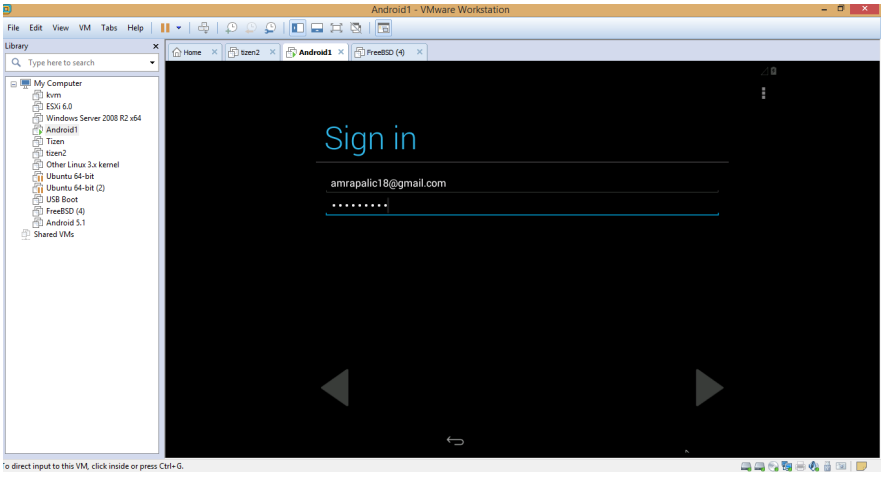

Figure 5.7: Sign in into Google account

Figure 5.8: Android Vitual Machine

xii) Once Android Virtual machine is created, we have to acquire Google account and sign in. It will ask for important credentials to authenticate the Google account, as shown in figure 5.6 and figure 5.7.

Xiii) Android x86 virtual machine is then ready to use, as shown in figure 5.8. It will show all applications on the screen.

5.2 Tizen Wayland kernel virtual instance creation

In this section we will give detail implementation of Tizen VM. We have used Tizen Wayland kernel version as our second instance of VM. In figure 5.9 we give overview of Tizen installation Steps. We have used Tizen IVI image and convert it by using qemu−img. Qemu−img is a disk image utility which is used to convert default raw image format into a VMware compatible image format. Following are the detail steps to create Tizen VM.

i) Download Tizen ivi image from the following link:

http://www.download.tizen.org/release . Choose Fedora 64 bit OS image for virtual machine.

ii) Extract the downloaded image using following command shown in Listing 5.2:

```
$ bunzip2 -k ivi-tizen-3.0_20151119.2_ivi-mbr-i586-sdb.raw.bz2
$ dd if=ivi-3.0-wayland-tizen-3.0_20151119.2-sdb.raw of=/dev/sdd bs=8M
```

Listing 5.2: Extract the ivi file

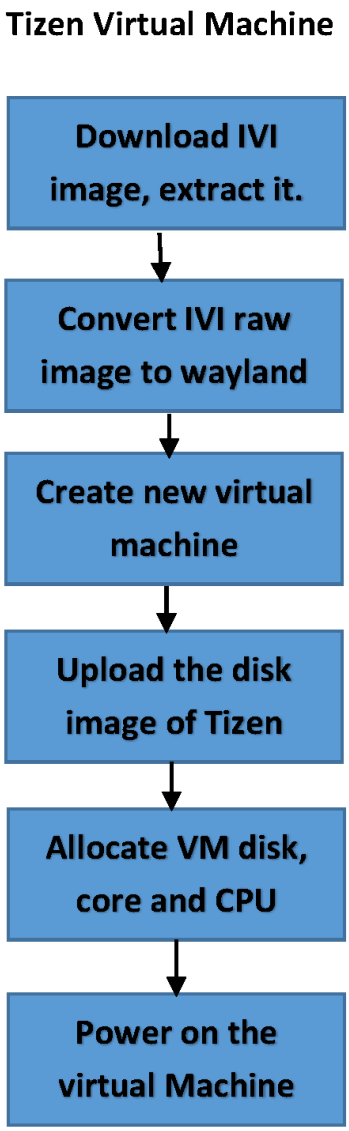

Figure 5.9: Tizen x86 Virtual Machine Flow

iii) To convert the Tizen ivi image into Wayland kernel, we need to add Tizen repositories. This can be using following code in listing 5.3.

iv) We have used the qemu−img to convert the ivi image to Wayland. The code we have used is given in Listing 5.4.

v) Create a new Tizen virtual machine in VMware. Select " I will install the operating System later". Use Linux− > Fedora 64−bit for 64−*bit* virtual vmdk Tizen image.

```
$ sudo zypper addrepo http://download.tizen.org/tools/latest-release/Fedora_17/tools ↩
    .repo

   $ sudo yum makecache
   $ sudo yum install lthor
```

Listing 5.3: Add the Tizen tools repository

```

$ qemu-img convert  f  raw ivi-tizen-3.0_20151119.2-sdb.raw O vmdk ivi-tizen-3.0 ↩
    _20151119.2-sdb.vmdk

Transfer that vmdk ivi-tizen-3.0_20151119.2-sdb.vmdk VMware disk image to your  ↩
    system
```

Listing 5.4: Convert ivi image to vmdk

vi) Allocate new IDE(0:0) type virtual disk as a single file. Go to edit virtual machine and uncheck all the connections for CD/DVD, printer,sound card and display.

vii) As shown in figure 5.10, upload the VMware vmdk image that we have created in step iv.

viii) Add a new hard disk and select disk type IDE as shown in figure 5.11.

ix) Convert the disk format to support VMware workstation as shown in figure 5.12.

viii) Select Tizen Wayland 3.0.vmdk image file. Power on the Tizen Wayland kernel virtual machine as shown in figure 5.13. Tizen is still in developing state, so Tizen kernel shows multiple Tizen launcher like carol, developer, bob, alice and guest as shown in figure 5.14.

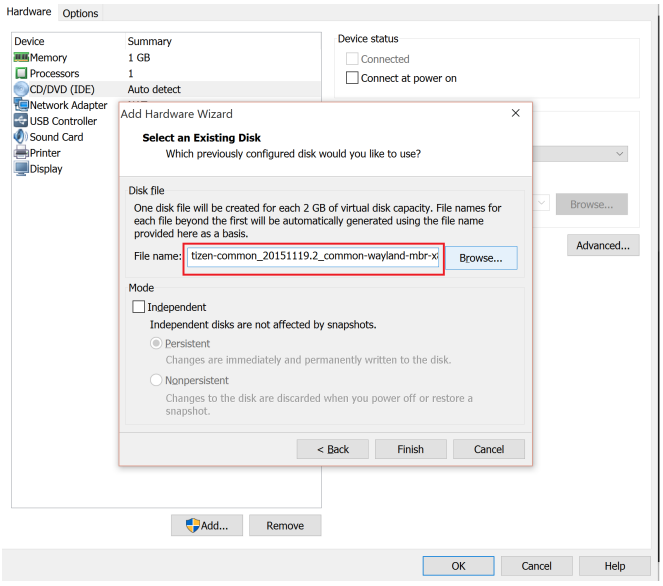

Figure 5.10: Upload the Tizen image

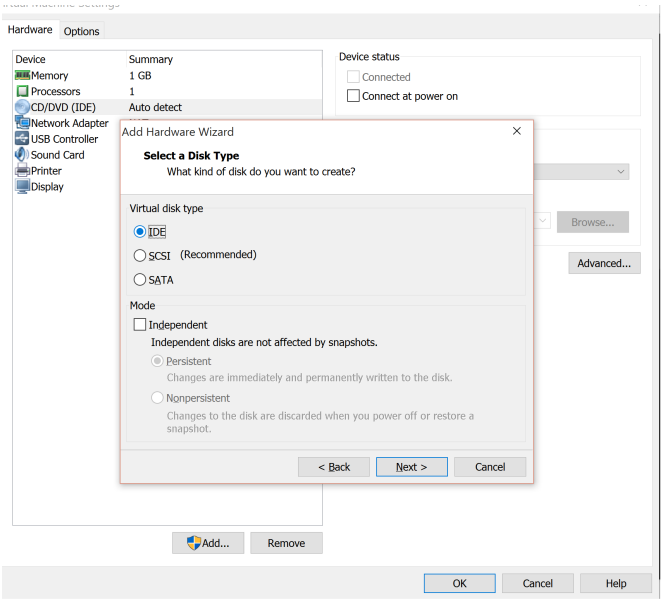

Figure 5.11: Select IDE hard disk

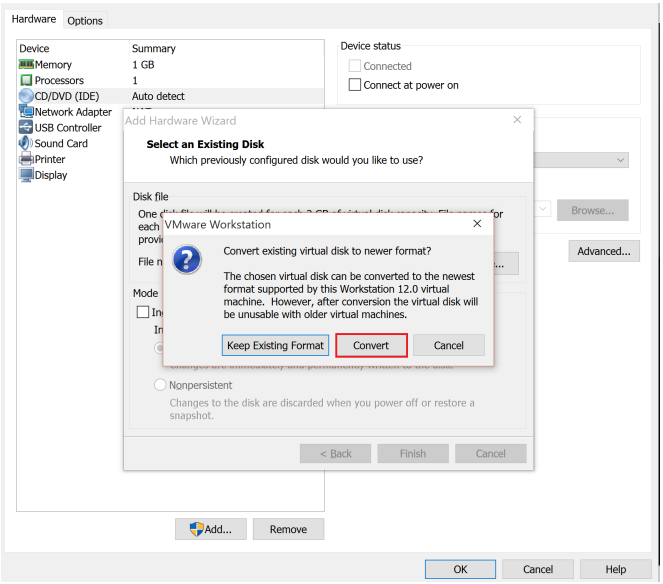

Figure 5.12: Convert the disk format to support VMware

Figure 5.13: Power on Tizen wayland virtual machine

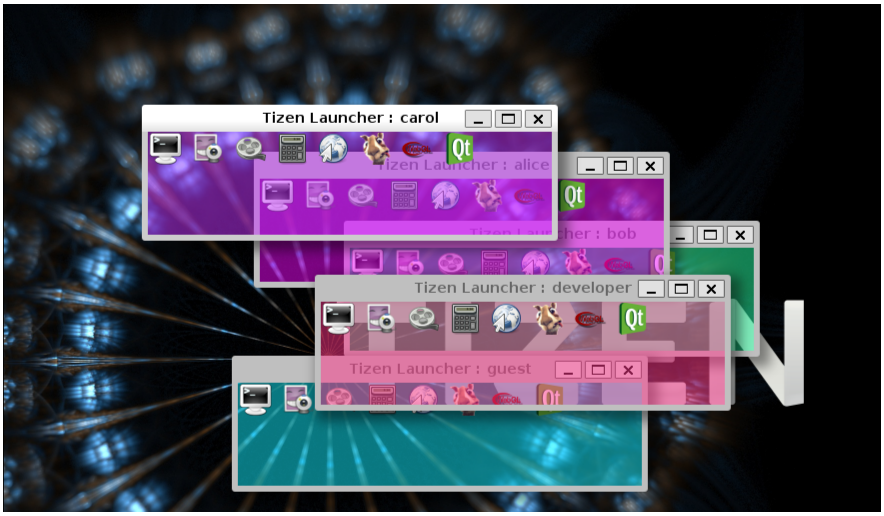

Figure 5.14: Tizen wayland kernel

Chapter 6

Evaluation

This chapter evaluates the performance comparison of virtual instances of Android and Tizen operating systems. In section 6.1, we describe the first test that compares EXT4 Filesystem of Android and Tizen. In section 6.2 we describe the difference file system structure of Android and Tizen.

6.1 EXT4 file system Mounting Test

According to Sobell (2013) practical guide on Fedora, EXT4 filesystem is an extension of EXT3 file system. In this test, we compare mounting methods of EXT4 file system on Android and Tizen platform. For this test we have used Android VM, Tizen VM and Android Native platform. EXT4 file system contains "atime" field in its inode. In table 6.1, we differentiate methods to mount EXT4 file system.

Table 6.1: EXT4 Filesystem mounting comparision between Android and Tizen

EXT4 mounting method- Android
adb -d shell mount /dev/block/mmcblk0p9 /system ext4 ro,relatime,barrier=1,data=ordered 0 0 /dev/block/mmcblk0p7 /cache ext4 rw,nosuid,nodev,noatime,barrier=1, $journal_a sync_c ommit, data = ordered$ /dev/block/mmcblk0p10 /data ext4 rw,nosuid,nodev,noatime,barrier=1,data=ordered,
EXT4 mounting method- Tizen
mke2fs -t ext4 /dev/hda1 mount -t ext4 /dev/hda1 /wherever /opt relatime,user xattr,barrier=1 /var relatime,user xattr,barrier=1 /opt/usr relatime,user xattr,barrier=1

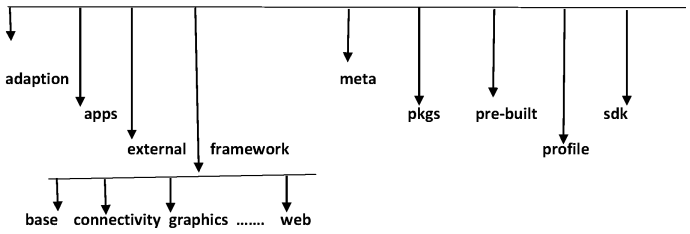

Figure 6.1: Tizen OS File System Tree

The analysis of this comparison includes, Android and Tizen EXT4 file system mouting methods. The Tizen OS structure uses an old Linux directory format like /var and /opt. Whereas, Android uses updated Linux directory format including /data and /system. Android uses asynchronous journal commit, to check records of file system. Asynchronous journal commit is use to track records of read, write and open operations of file. EXT4 file system in Tizen have inode with atime field, it will update for each file accessed. In each file access or read operation inode become dirty every time. Whereas, in Android atime field is already disabled and set to notime, so it will reduce the IO traffic. In Tizen EXT4 mounting method, user can extend and add extra attributes in /opt and /var directory.

6.2 Comparison of File System Structure

In file system comparison we observed the folder structure of Tizen Os and Android Lollipop OS. Although both the operating systems are based on Linux, the file system structure is different. Tizen is Linux based operating system. Unlike Android , Tizen used legacy structure of Linux file system. Figure 6.1 gives detailed overview of Tizen file system hierarchy.

•Tizen File Sytem

• adaption: This folder is used to save external projects. These projects are made by external parties with open source libraries. To integrate such open source projects like openGL, in Tizen OS adaption folder is used. This folder is consider for Hardware Abstraction Layer (HAL).

• Apps : Tizen in built apps like contacts, calculator, calender and email can be make available in this folder. We can access and manage these apps through apps folder.

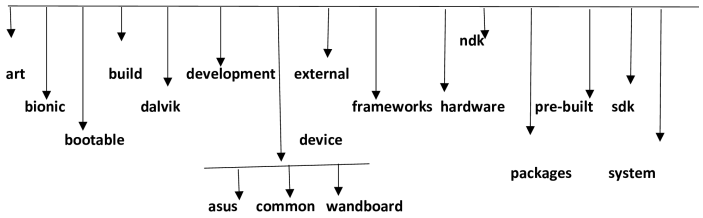

Figure 6.2: Android Lollipop OS File System Tree

• external: The library functions and all other external apps can be found in external folder.

• framework: Tizen framework folders and core level access code is contain in framework folder. Source code of OS is available here.

• meta: Meta folder offers, all the configuration files for related operating system type. This folder contains IVI, mobile or computer OS configurations.

• pkgs: This file stores packages of source code. To find perticular packages in this folder, we can use separate tool called mtool.

• pre-built: This folder contains important packages for ARM and x86 system. These packages are need to be pre-install.

• profile: All system configuration related files are stored here. Configuration files are used for specific device.

• SDK: Software Development Kit folder offers the source code for developers.

Android Lollipop File System

Unlike Tizen, Android does not use old Linux file system structure. Figure 6.2 shows the hierarchy of Android File System.

• art: In this folder virtual machine sources are stored, that the Android operating system supports.

• bionic: This folder contains the operating system version with C library.

• bootable: The failure recovery files and bootable files are stored in bootable folder.

• build: The source code and important script files, needed to configure the system are stored in the build file. Using build folder system configuration can run.

• dalvik: Dalvik is the one virtual machine system, that Android supports. All the files related to this virtual machine are stored in the dalvik file.

• development : Development tools and files stored here.

• device: All device related files and source codes are available here.

• external: All third party or external programs are save here. This file is use to import such external programs in Android operating system.

• frameworks: Application framework layer and system library can be found in framework folder. The sources to access library are stored in framework file.

• hardware: In hardware folder, Android Hardware Abstraction Layer information and device implementation sources are stored.

• ndk: The Native Development Kit is also called as ndk. This folder contains, the development kits to manage and develop applications for Android operating system.

• packages: Android system default applications like contact, calender, calculator and email are available here. The source code of these in built application can easily available in this folder.

• pre-built: The special type of Android tools like emulator or misc tools are stored here.

• sdk: The Software Development Kit is also called as sdk. This folder contains, the development kits to manage and develop applications for Android operating system.

• system: All bootable and network programs are stored here.

6.3 LINPACK Bechmarking

In this section, we performed a comparison of Android VM, Tizen VM and Android Native. We installed the LINPACK benchmarking tool on these three machines. In this test we point out and measure the performance of FLOPS for multiple operations. As LINPACK is a highly efficient tool, to calculate FLOPS and matrix calculations. The main focus is to estimate, how speedily system can resolve real time issues. Our main goal to observe floating point power, residuals and time to calculate operations. LINPACK calculates aggregate time to perform difficult tasks and then convert this aggregation into performance rate. It uses partial pivoting technique where, any high absolute value is set as a pivoting matrix value. LINPACK also uses, SMP parallelism to synchronize number threads and number of cores in system. As per LINPACK

benchmark, the machine that gives high performance in FLOPS rate is the best machine to solve real time problems.

Based on this test we find out, which operating system will perform well in a virtual environment as compared to native. We launch LINPACK tool on Samsung Galaxy S 5 to measure native performance.

Table 6.2 and 6.3 depicted the test result with FLOP rates, time and residuals.

Table 6.2: LINPACK Test table for virtual environment

Test	Tizen VM			Android VM		
	MFLOPS	Time	Residual	MFLOPS	Time	Residual
Test 1	5000	0.015	2.03	1982.223	9.063s	1.6
Test 2	2108.6	0.040	2.03	2195.616	9.144s	1.6
Test 3	8240.2	0.010	2.03	2280.986	7.815	1.6
Test 4	5862.2	0.014	2.03	1862.218	9.04s	1.6
Average	5302.75	0.020	2.03	2080.26075	8.7655s	1.6

Table 6.3: LINPACK Test table for Native environment

Test	Android Native Machine		
	MFLOPS	Time	Residual
Test 1	1394.276	7.655s	1.6
Test 2	1415.277	7.667s	1.6
Test 3	1403.523	7.557s	1.6
Test 4	1431.268	7.810	1.6
Average	1096.086	7.67255s	1.6

This test result is conducted according to Gaussian elimination with partial pivoting. Partial pivoting is used to reduce round off errors. For pivoting, it selects a matrix operation with largest absolute value from column of the matrix and it considers as a pivot element. We use LINPACK to measure speed in terms of FLOPS.

The floating point operation used by LINPACK for partial pivoting is $2/3*N3+2*N2$.

We conduct four tests for Tizen VM and Android VM and calculate the average MFLOPS. Figure 6.3 clearly shows the graphical representation of Table 6.2 and Table 6.3. The average MFLOPS output of Tizen VM is greater than Android VM, as shown in figure 6.4. The analysis of this test is, in average MFLOPS rate comparison Tizen VM give high performance than Android VM and Android Native machine.

Figure 6.4 clearly depicts the comparison of average MFLOPS performance among Tizen VM, Android VM and Android native machine. The average value of MFLOPS is greater than Android VM and Android Native machine.

Tizen OS gives performance in a virtualize environment as compare to Android OS in

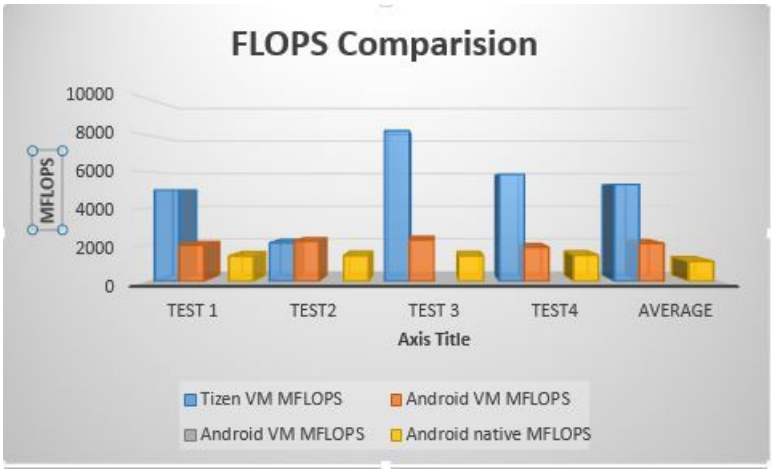

Figure 6.3: FLOPS comparison

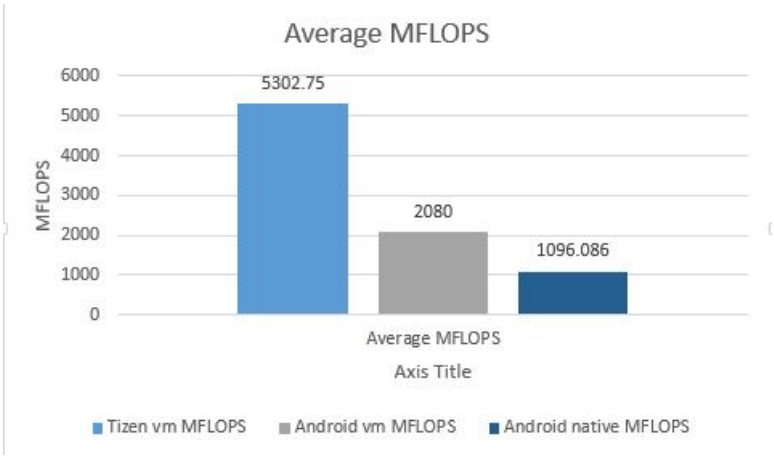

Figure 6.4: Average MFLOPS comparison using LINPACK

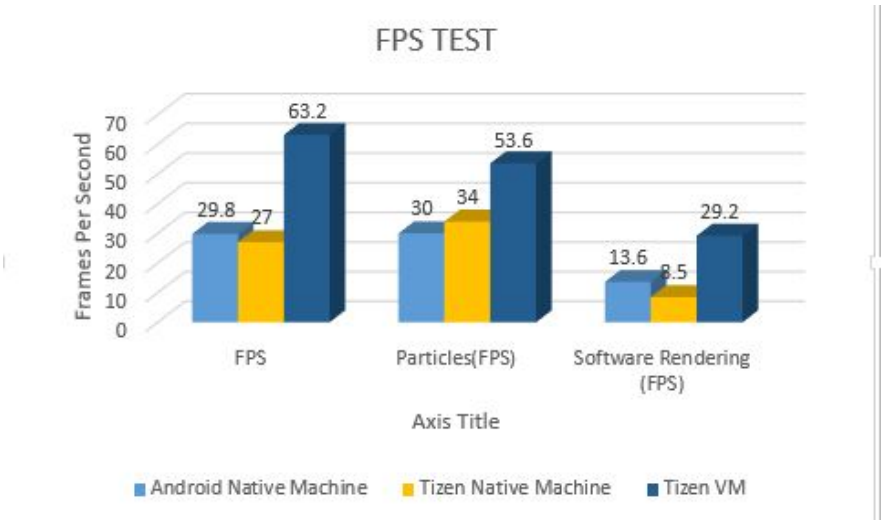

Figure 6.5: FPS Test

a virtual environment. On the other hand Native Android OS gives less performance as compared to Android VM and Tizen VM.

6.4 Frame Per Second (FPS)

In this section we perform Frame Per Second (FPS) test for Tizen native device. We used Samsung Gear S wearable device with Tizen OS. For this test we install Samsung Gear Manager tool on Samsung Galaxy S 5 mobile phone. This tool is used to install and manage applications on the wearable device. We launch 3D G Mark benchmark tool on wearable device trough Samsung Gear Manager.

Table 6.4: FPS test

Test	Android Native	Tizen Native	Tizen VM	Android VM
WebGL	Detected	Detected	Detected	Detected
Graphics FPS	29.8 (25.7− 60.0)	27	63.2	NA
Particles(FPS)	30 (20.6−60.0)	34	53.6	NA
S/W Rendering (FPS)	13.6 (12.2−16.6)	8.5	29.2	NA

3D G− Mark is the first developed benchmarking tool for smart wearable devices. In mobile device, we install 3D Mark tool which performs the same test as 3D G− Mark. To measure the performance in Tizen VM, we have used Wayland rendering tool. This tool gives FPS output for graphics, particles and software rendering tests.The table 6.4 shows the FPS test performance.

- Wayland Tool in Fedora

```
- Download Source Code
git.clone git://github.com/quanxianwang/wr-graph.git
- Install Python packages
zypper install python2.7
Zypper install python-cairo
Zypper install python-wxgtk2.0
Zypper install python-wxglade
Zypper install python-wxmpl python-wxtools python-wxversion
- Download Logs in Weston
git clone https://reviwe.tizen.org/git/platform/upstram/weston
git checkout tizen
- Run tool
cd wr-graph/src
./gui.py --output=./output/motion/ --config=../config/config.xml --log=./weston.log ↩
    --show=false
```

Listing 6.1: Run Wayland Tool (Fedora)

Figure 6.5 represents, the graph of FPS benchmark for the Android Native device, Tizen Native device and Tizen VM. As we mention in the table 6.4, Android VM is not supported this test. As we tried to install 3D Mark tool to conduct FPS test, but it does not run properly. Overall FPS performance is almost double in Tizen Wayland than Android Native and Tizen Native. In another test for particles, Tizen VM achieved a higher FPS rate than Android and Tizen Native. In software rendering test, Tizen VM achieved almost double FPS rate than Android Native device. Tizen smart watch only 8.5 FPS rate, which is very less as compare to Tizen VM. As we have used Samsung Gear smart watch for this test, the FPS performance may increase for smart phone device.

Chapter 7

Conclusion

Virtualization in desktop and network servers are not compatible with new generation mobile devices. Hardware requirements of high performance system is different from a mobile device. So here we conclude need of new virtualization technique for mobile cloud computing. This thesis recommends a virtualization solution for new generation mobile operating systems like Android and Tizen. Even if Android is a well established OS since many years, it's the FLOPS rate is less as compare to Tizen OS. Tizen OS is available for many things from IVI, wearable to mobile devices. Tizen VM achieved more than double FLOPS performance than Android VM and Android Native. As Tizen give a better virtualize performance than Android and it is available for many smart devices, it can be use to effectively manage virtualization in MCC. Moreover, BYOD enables the virtualization environment for mobile devices. As Tizen give a better virtualize performance than Android and it is available for many smart devices, it can be use for BYOD.

In evaluation study, we conduct the EXT4 Filesystem mounting test. The Tizen OS structure uses an old Linux directory format like /var and /opt and Android updates Linux directory format including /data and /system. EXT4 file system in Tizen have inode with atime field, it will update for each file accessed. In each file access or read operation inode become dirty every time. Whereas, in Android atime field is already disabled and set to notime, so it will reduce the IO traffic.The evaluation chapter also compared, the FPS performance of Android and Tizen. We achieved, greater FPS rate in Tizen Wayland VM. On the other hand, we analyzed than Android VM is not supported graphic and FPS test.

The performance measurement taken in this study, is our first step and only focused on FLOPS and FPS tests. This thesis study is helpful in BYOD environment where, the

employee can access their organization domain and personal domain on the same mobile device. Using the virtualization technique, we can provide isolate and secure BYOD environment. In future work, we consider to implement virtualization on mobile device. We will try to create virtual instance on mobile devices and check its performance. We will perform this task, on Android and Tizen based Mobile phones.

Bibliography

3D Mark Technical Guide (2015), http://s3.amazonaws.com/download-aws.futuremark.com/3DMark_Technical_Guide.pdf. [Online] [Accessed: 2015-11-02].

Android ArchitectureKernel Description (2013), http://developer.android.com/about/versions/lollipop.html. [Online] [Accessed: 2015-10-30].

Andrus, J., Dall, C., Hof, A. V., Laadan, O. and Nieh, J. (2011), Cells: a virtual mobile smartphone architecture, *in* 'Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles', ACM, pp. 173–187.

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I. et al. (2010), 'A view of cloud computing', *Communications of the ACM* **53**(4), 50–58.

Asrar and irfan (sept 2014), 'Attack surface analysis of the tizen os'.

Barr, K., Bungale, P., Deasy, S., Gyuris, V., Hung, P., Newell, C., Tuch, H. and Zoppis, B. (2010), 'The vmware mobile virtualization platform: is that a hypervisor in your pocket?', *ACM SIGOPS Operating Systems Review* **44**(4), 124–135.

Bhargava, R., Serebrin, B., Spadini, F. and Manne, S. (2008), 'Accelerating two-dimensional page walks for virtualized systems', *ACM SIGOPS Operating Systems Review* **42**(2), 26–35.

Brahler, S. (2010), 'Analysis of the android architecture', *Karlsruhe institute for technology* .

Buckley, R. (2012), 'How to cope with BYOD?', *SC Magazine UK* pp. 23–26. [Online] [Accessed on 2015-07-15].
URL: *http://www.scmagazineuk.com/how-to-cope-with-byod/article/264855/*

Bugiel, S., Davi, L., Dmitrienko, A., Fischer, T., Sadeghi, A.-R. and Shastry, B. (2012), Towards taming privilege-escalation attacks on android., *in* 'NDSS'.

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J. and Brandic, I. (2009), 'Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility', *Future Generation computer systems* **25**(6), 599–616.

Cagalaban, G., Kim, S. and Kim, M. (2012), A mobile device-based virtualization technique for m2m communication in cloud computing security, *in* 'Computer Applications for Security, Control and System Engineering', Springer, pp. 160–167.

Carabas, M., Mogosanu, L., Deaconescu, R., Gheorghe, L. and Tapus, N. (2014), Lightweight display virtualization for mobile devices, *in* 'Secure Internet of Things (SIoT), 2014 International Workshop on', IEEE, pp. 18–25.

Charland, A. and Leroux, B. (2011), 'Mobile application development: web vs. native', *Communications of the ACM* **54**(5), 49–53.

Dong, Y., Mao, J., Guan, H., Li, J. and Chen, Y. (2015), 'A virtualization solution for byod with dynamic platform context switching', *Micro, IEEE* **35**(1), 34–43.

Durairaj, M. and Manimaran, A. (2014), 'The international journal of science & technoledge'.

Gadyatskaya, O., Massacci, F. and Zhauniarovich, Y. (2014), 'Security in the firefox os and tizen mobile platforms', *Computer* (6), 57–63.

Grabham, D. (2010), 'Intel and nokia merge moblin and maemo to form meego', *t echradar. com. Retrieved* **15**.

Hoy, M. B. (2011), 'Html5: a new standard for the web', *Medical reference services quarterly* **30**(1), 50–55.

Huckle, D. and Cleaver, J. (2012), 'Game bench final project report eecs 395spring 2012'.

Hung, S.-H., Shih, C.-S., Shieh, J.-P., Lee, C.-P. and Huang, Y.-H. (2011), An online migration environment for executing mobile applications on the cloud, *in* 'Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2011 Fifth International Conference on', IEEE, pp. 20–27.

Jack J. Dongarra, Piotr Luszczek, A. P. (2012), 'The LINPACK Benchmark: Past, Present, and Future', http://cluster.earlham.edu/project/curriculum-modules/JOCSE_PetaKit/Resources/LINPACK.pdf. [Online] [Accessed: 2015-11-15].

Kemp, R., Palmer, N., Kielmann, T. and Bal, H. (2012), Cuckoo: a computation offloading framework for smartphones, *in* 'Mobile Computing, Applications, and Services', Springer, pp. 59–79.

Kim, H.-J. and Kim, J.-S. (2012), Tuning the ext4 filesystem performance for android-based smartphones, *in* 'Frontiers in Computer Education', Springer, pp. 745–752.

Kim, M., Lee, H.-U. and Won, Y. (2015), Io characteristics of modern smartphone platform: Android vs. tizen, *in* 'Wireless Communications and Mobile Computing Conference (IWCMC), 2015 International', IEEE, pp. 142–147.

Lange, M., Liebergeld, S., Lackorzynski, A., Warg, A. and Peter, M. (2011), L4android: a generic operating system framework for secure smartphones, *in* 'Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices', ACM, pp. 39–50.

Lee, Y.-C. and Hsueh, C.-W. (2013), An optimized page translation for mobile virtualization, *in* 'Proceedings of the 50th Annual Design Automation Conference', ACM, p. 85.

Liang, C. and Yu, F. R. (2015), 'Wireless virtualization for next generation mobile cellular networks', *Wireless Communications, IEEE* **22**(1), 61–69.

Liu, F., Tong, J., Mao, J., Bohn, R., Messina, J., Badger, L. and Leaf, D. (2011), 'Nist cloud computing reference architecture', *NIST special publication* **500**, 292.

Morita, M., Ichikawa, Y., Terunuma, K. and Seung, H. (2007), 'Limo foundation: toward a common linux-based mobile platform', *NTT DoCoMo Tech. J* **9**(2), 41–46.

Oh, S.-C., Kim, K., Koh, K. and Ahn, C.-W. (2010), 'Vimo (virtualization for mobile): a virtual machine monitor supporting full virtualization for arm mobile systems', *Proc. Advanced Cognitive Technologies and Applications, COGNITIVE* .

Popek, G. J. and Goldberg, R. P. (1974), 'Formal requirements for virtualizable third generation architectures', *Communications of the ACM* **17**(7), 412–421.

Poylisher, A., Serban, C., Lee, J., Lu, T.-C., Chadha, R., Chiang, C.-Y., Orlando, R. and Jakubowski, K. (2010), 'A virtual ad hoc network testbed', *International Journal of Communication Networks and Distributed Systems* **5**(1-2), 5–24.

Prabhakaran, V. (2014), 'Samsung announced tizen-based gear 2 and gear 2 neo'.

Roh, H.-s., Lee, H.-w. and Lee, S.-h. (2014), A study on mobile virtualization, *in* 'Advanced Communication Technology (ICACT), 2014 16th International Conference on', IEEE, pp. 593–596.

Samsung's NX300M smart camera is its first to run Tizen OS (2013), http://www.engadget.com/2013/11/11/samsungs-nx300m-mirrorless-camera-is-its-first-to-run-tizen-os/. [Online] [Accessed: 2015-10-01].

Schüring, M. (2011), Mobile cloud computing–open issues and solutions, *in* '15thTwente Student Conference on IT, Enschede, The Netherlands'.

Shiraz, M., Abolfazli, S., Sanaei, Z. and Gani, A. (2013), 'A study on virtual machine deployment for application outsourcing in mobile cloud computing', *The Journal of Supercomputing* **63**(3), 946–964.

Smalley, S. and Craig, R. (2013), Security enhanced (se) android: Bringing flexible mac to android., *in* 'NDSS', Vol. 310, pp. 20–38.

Sobell, M. G. (2013), *A Practical Guide to Fedora and Red Hat Enterprise Linux*, Pearson Education.

S.Suzuki (2013), http://www.ffri.jp/assets/files/monthly_research/MR201305_Tizen_Security_ENG.pdf. [Online] [Accessed: 2015-11-30].

Tizen Web Guides (2012), https://developer.tizen.org/development/guides/web-application. [Online] [Accessed: 2015-11-01].

Willis, N. (2014), 'Tizen Common and open hardware', *Tizen Developer Summit* .
URL: *https://www.sharelatex.com/project/55b3b2dcd2ca043251bce938*

Woyke, E. (2012), 'Samsung merging its bada os with intel-backed tizen project. 13.01. 2012', *URL http://www. forbes. com/sites/elizabethwoyke/2012/01/13/samsung-merging-itsbada-os-with-intel-backed-tizen-project/, Abruf am* **8**, 2012.

Xing, T., Huang, D., Ata, S. and Medhi, D. (2012), Mobicloud: a geo-distributed mobile cloud computing platform, *in* 'Proceedings of the 8th International Conference on Network and Service Management', International Federation for Information Processing, pp. 164–168.

Xu, Y., Bruns, F., Gonzalez, E., Traboulsi, S., Mott, K. and Bilgic, A. (2010), Performance evaluation of para-virtualization on modern mobile phone platform, *in* 'Proceedings of the International Conference on Computer, Electrical, and Systems Science, and Engineering'.

Zhang, Q., Cheng, L. and Boutaba, R. (2010), 'Cloud computing: state-of-the-art and research challenges', *Journal of internet services and applications* **1**(1), 7–18.

Appendix A

Chapter 8

A.1 Software Requirement for Wayland Tool

```
Software package requirements
* python (>=2.7)
 * python-wxgtk2.8 - wxWidgets Cross-platform C++ GUI toolkit (wxPython binding)
 * python-wxglade - GUI designer written in Python with wxPython
 * python-wxmpl - Painless matplotlib embedding in wxPython
 * python-wxtools - wxWidgets Cross-platform C++ GUI toolkit (wxPython common files)
 * python-wxglade - GUI designer written in Python with wxPython
 * python-wxversion - wxWidgets Cross-platform C++ GUI toolkit (wxPython version ↩
    selector)
 * python-cairo
 * python-gi-cairo
```

A.2 Linpack Snapshots for Android VM and Tizen VM

This section, we include snapshots for test result of Linpack benchmarking for Android OS. Figure A.1 and A.2 shows, the Test results in MFLOPS.

A.3 Linpack Snapshot for Android VM

A.4 Linpack Snapshot for Android VM

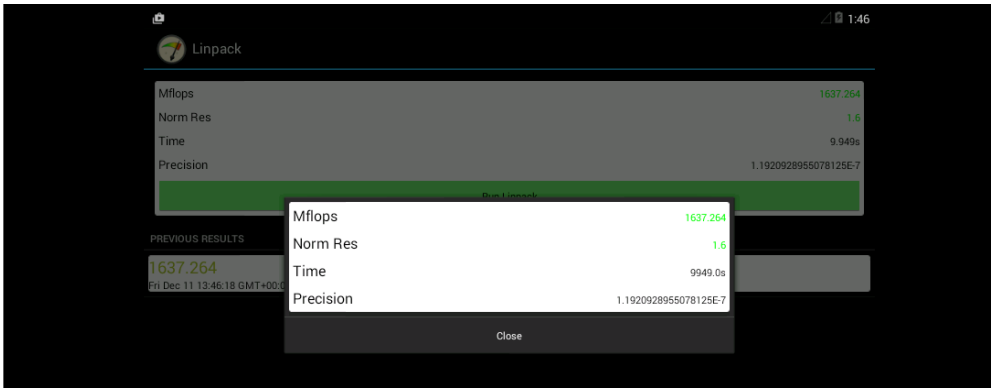

Figure A.1: Android VM Linpack Test 1

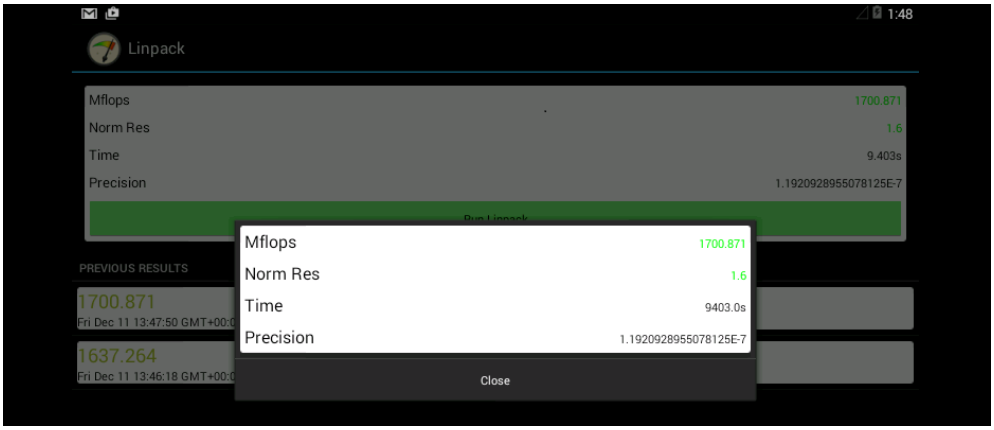

Figure A.2: Android VM Linpack Test 2

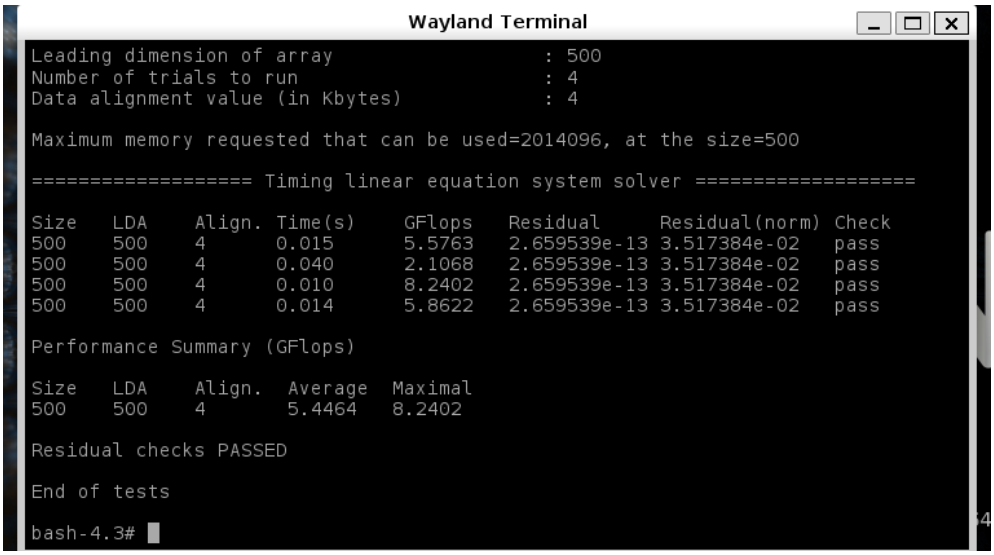

Figure A.3: Tizen VM Linpack Test