RESEARCH ARTICLE OPEN ACCESS

ANDROID MOBILE AS A SERVER PLATFORM

Ms. Himani Sahu, Ms. Mona Mulchandani
Department of Computer Science & Engineering
Jhulelal Institute of Technology, Nagpur

Abstract:

The number of smart phone users and mobile applications are growing rapidly. Though smart phones are expected to have PC-like functionality, hardware resources such as CPU's, memory and batteries are still limited. To solve this resource problem, many researchers have proposed architectures to use server resources in the cloud for mobile devices.

In this paper, we propose a conceptual architecture of Android as a Server Platform, which enables multiple user Android applications on cloud server via network and backup and restore approach for mobile devices, which helps to reduce the effort in saving and restoring personal data. Android's two other features are useful to construct a server platform, i.e. Android is open source product and runs on an x86 CPU. Another feature of our approach lies in the capability of sharing information in mobile devices among a group of selected persons. This can be useful in many situations e.g., in creating a mobile business network among a group of people.

Keywords- ***Android, multi-tenant; cloud, backup and restore, mobile devices.***

1.INTRODUCTION

Android as a Server Platform is proposed that enables many users to use resources on remote cloud servers. Android is an open source mobile OS initiated by Google. The main reason to use Android as a server platform is that it is able to run not only for smartphones but also for the x86 platform including servers.

Backup is a crucial task, since hardware faults and software or human errors can lead to the loss of important information. In addition to faults, backups are even more important for devices such as laptops and smartphones, since they are more prone to loss or to theft. Currently, smartphones are used more as handheld computers than as mobile phones, and consequently a lot of data is stored in those devices. This makes more critical the need to keep data stored on those devices safe from losses. In addition, the rapid technological evolution in mobile devices makes it more difficult to restore data saved from old devices to new ones. Thus, mobile devices pose new challenges for the backup and restore problem.

Making backups on external memory devices, such as on Secure Digital (SD) cards or on online and to provide backup and restore services based on the cloud computing paradigm, which is considered to be more reliable and less expensive by end users. This approach reduces also the risk of data loss and decouples the data from a specific device.

Once information about backups moves online, it can be used in shared applications.

2. LITERATURE SURVEY

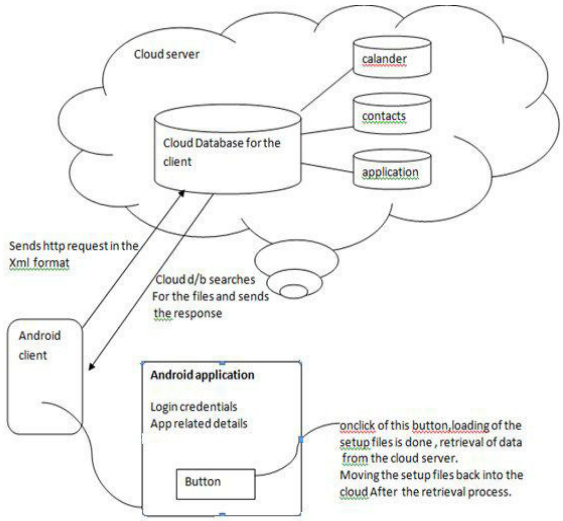
Android is a Linux-based operating system designed primarily for touchscreen mobile devices such as smartphones and tablet computers. Initially developed by Android, Inc., which Google backed financially and later bought in 2005.Android was unveiled in 2007 along with the founding of the Open Handset Alliance: a consortium of hardware,software, and telecommunication companies devoted to advancing open standards for mobile devices.

Android is open source and Google releases the code under the Apache License.This open-source code and permissive licensing allows the software to be freely modified and

distributed by device manufacturers. Additionally, Android has a large community of developers writing applications ("apps") that extend the functionality of devices, written primarily in a customized version of the Java programming language.

Cloud Computing is a synonym for distributed computing over a network and means the ability to run a program on many connected computers at the same time. The phrase is also, more commonly used to refer to network-based services which appear to be provided by real server hardware, which in fact are served up by virtual hardware, simulated by software running on one or more real machines. Such virtual servers do not physically exist and can therefore be moved around and scaled up (or down) on the fly without affecting the end user - arguably, rather like a cloud.

3. SYSTEM ARCHITECTURE



Multitenant architecture for Android

Multi-tenancy, which means that software running on a server provides services to many users, is one of the important features for cloud computing. From the viewpoint of both economy and ecology, it is beneficial to share hardware resources amongusers. Using a mobile OS would be more effective than using a desktop OS because the resource requirements of mobile OS are smaller.

This section discusses to construct multi-tenant architecture for Android. Figure 1 shows an overview of the architecture Android on a server. Android system info is drawn upon Google's "Application Developers" document [7]. There are three types of approach, hypervisor-layer, kernel-layer, and framework-layer, for multi-tenant architecture. The hypervisor-layer approach uses the Virtual Smartphone over IP system. Each user owns his/her Android OS image on a server and freely runs his/her application in a separate VM Multi-tenancy is achieved by running multiple users VMs in a server via a hyper-visor.

This approach has the advantage of application usability and maintenance. From the viewpoint of application usability, every mobile application that can run on Android-x86 is usable because each Android OS runs only one application. Android has

different versions and version up is currently on going.

The second approach implements multi-tenant function in kernel-layer.This approach changes Android OS to run multiple user applications in separate processes. This approach is similar to an ordinary thin client server running multiple user applications in a server. The main challenge is that original Android supports only one display and keypad device since Android is mainly designed to work on a smartphone.

Kernel layer approach

Another approach is to create a multi-tenant function at framework-layer, similar to existing a Java-based multi-tenant framework. This approach remodels Android the framework and APIs to sup-port multiple user applications. The main challenge is how to run existing Android applications in modified framework.

As shown in [8], the hypervisor-layer approach is feasible and good for maintenance. However, it seems to have a scalability limitation caused by a hypervisor. Because each VM try to separately maintain their resources, it is difficult to control unused resources. The other two approaches have an

advantage in scalability but have a disadvantage in maintenance because they change the Android OS. From the viewpoint of running existing application, the kernel-layer approach is better because it does not changes Android runtime environment. Moreover, we assume that the kernel-layer approach is easy to develop because Android is implemented based on the Linux kernel so that can support multiple displays, keypads, and applications.

4. IMPLEMENTATION

Integration of mobile device and cloud

Mobile devices are integrated with cloud computing. Mobile users can seamlessly use nearby computers to obtain cloud-computing resources by instantiating a "cloudlet" that rapidly synthesizes virtual machines on a nearby infrastructure that can be accessed through a Wireless LAN. Cloudlet can be defined as resource rich computer or cluster of computers that provide services to multiple users. The creation of clone VMs enable to run mobile applications as if they were running on mobile devices. Augmented execution is used to speed up mobile applications, namely Primary, Background, Mainline, Hardware, and Multiplicity, and presented a research agenda to bring the vision into reality. Augmented execution is performed in four steps. Initially clone of the Smartphone is created in the single machine into a distributed execution. Second step is that state of the primary and clone is synchronised. In the next step is that application augmentation is executed in the clone on request. Finally results from the clone execution are integrated back into the Smartphone state.

A pair of VNC-based server and client program is implemented. Server program resides in each Android-x86 image that run on top of VMWARE ESXi while the client program is installed in the physical Android device. The client program enables a user to remotely interact and control Anroid-x86 images. The client program transmits various events from the physical device to the virtual Smartphone and receives graphical screen updates from the virtual Smartphone.A virtual sensor driver can be implemented in the Android-x86 image. Most modern Smart phone is equipped with various sensor devices such as GPS, accelerometer and thermometers. While VNC itself supports only keyboard and mouse as the primarily input devices, client program can be extended to transmit sensor readings (accelerometer, orientation, magnetic field and temperature etc) to the virtual sensor driver in the Android-x86 image. The virtual sensor driver can be implemented in such a way that the sensor readings from the physical Android device would appear to come from the Anroid-x86 images itself. This is an important feature as it allows Android applications in an Android-x86 image to obtain sensor readings from the physical Smartphone without any modification.

5. CONCLUSION

In this paper, we proposed Android as a server platform system that enables the use of saving, recovering and sharing personal information into closed groups of smartphones. We also showed the technical difficulty and approaches related to multi-tenant architecture for Android OS. We plan to develop a prototype system about proposed multi-tenant Android architecture. We believe that proposed architecture shows high performance on virtual image-based virtualization for mobile applications.

REFERENCES

[1] Buyya R., Chee Shin Yeo and Venugopal S. (2008) 10th EE International Conference, 5-13.

[2] Wei-Tek Tsai, Xin Sun and Balassooriya J. (2010) Seventh International Conference, 684-689.

[3] ENISA (2007) Giles Hogben.

[4] http://www.dropbox.com/.

[5] http://www.zumodrive.com/.

[6] Chervenak A., Vellanki V. and Kurmas Z. (1998) Joint NASA and IEEE Mass Storage.

[7] AndroidDevelopers. http://developer.android.com/index.html.

[8] Chen E.Y. and Ito M. (2010) IEEE WOWMOM.

[9] Agarwal S. and Starobinski D. and Trachtenberg A. (2002) Department of Electrical and Computer Engineering, Boston University

[10] http://www.openmobilealliance.org/tech/affiliates/syn cml/syncmlindex.html.

[11] Fielding R. (2000) Architectural Styles and the Design of Net-work-based Software Architectures.

[12] Fielding R.T. and Taylor R.N. (2000) Software Engineering, International Conference, 407-416.