INTERNATIONAL JOURNAL OF TECHNOLOGY AND COMPUTING (IJTC)
ISSN-2455-099X,
Volume 2, Issue 5 May 2016

SELF-OFFLOADING OF ANDRIOD APPLICATIONS EXPLOITING THE RESOURCES OF CLOUD

Randeep Kumar[a], Sushil Kamboj[b], Sukhpreet Kaur[c]

[a]dhiman.randeep@gmail.com, M-Tech. Student, Computer Science and Engineering ,SUSCET Tangori, Mohali .

[b]principalsuspc@sus.edu.in, Principal SUS Polytechnic College, Tangori, Mohali.

[c]er.sukhpreetkaur@gmail.com, Head Of Computer Science and Engineering Department SUSCET, Tangori, Mohali

ABSTRACT

With the proliferation in the field of science and technology, the usage of smart phones is surging very rapidly over the last few years. There are a number of factors that are propelling their usage like mobility and faster network connectivity and as a result smart phones are increasing three times as compared to PCs. However, they are still constrained by limited storage capacity, restricted battery backup and processing power and as a consequence applications cannot be made very rich and efficient. In this paper a framework is being proposed to make android applications smart enough so that they can take self decision to offload their resource exhaustive and compute intensive part from the Smartphone to the virtual image of Smartphone on the cloud and thus exploiting the unlimited resources of cloud both hardware and software. With the help of this framework capabilities of smart phones can be enhanced and developers will be empowered to build feature rich applications.

I. INTRODUCTION

Cloud Computing is the use of computational resources (hardware and software) that are delivered as a service over a network. It makes a virtual pool of resources such as storage, CPU, networks and memory to fulfill the user's resource requirement and provides on demand hardware and software. Cloud Computing enables seamless access to the user applications and data from anywhere anytime in the world, unleashing the user from the confines of the desktop and making it easier for group members in different locations to collaborate. Cloud Computing portends a major change in how we store information and run applications. Cloud Computing has emerged as the great technology in term of scalability and portability. It has changed our view of carrying data and communication. Smartphones are mobile phones with advanced computing

capability, connectivity and rich set of functionality. In a nutshell, a smartphone combines the functionalities of a phone, personal digital assistant(PDA) and a small computer. With the increasing popularity and a large number of developers developing applications for smartphones, the users of these phones have started using them for high end 3D gaming, to handle their finances i.e. internet banking and as their health and wellness managers (e.g. Eat This, Not That app for Android [1]). These new applications could be very resource exhaustive and the phones have a limited memory, computational power and battery life. That's why it makes good sense to offload the heavy applications to the Virtual Smartphone running on the cloud, thus saving the actual phone's precious resources.

A number of techniques have been proposed to offload the applications of smartphones to the cloud [2-8], including complete offloading of the applications as well as partial offloading of the applications. In these techniques used for offloading, the application is partitioned at the binary level and thus making this par-titioning transparent for the application developer. But this has its drawbacks i.e. Firstly, this process is compute intensive. Secondly, to make changes at the binary level of an application needs changes in the application loader, which is difficult as well as leads to security vulnerabilities. Furthermore, in the proposed techniques an application called the application pratitionar or off loader needs to be in-stalled on the smart phone which makes the partitions and offloads the appropriate partition of other applications to the cloud. The application offloader makes the offloading decision for all the applications in the phone weather small or big in terms of computation required and thus become an overhead on the phone's re-sources. In this paper we propose a framework for offloading an application partially i.e. only the compute intensive, non interactive part of an application is offloaded. The partitioning is done by the application developer at the time of development of the application and the offloading decision is taken by the application itself thus eradicating the need of making changes at the binary level and the need of application partitioner or off loader. These applications will offload their compute intensive part to the cloud autonomously.

The remainder of this paper is organized as follows. In Section 2, we describe the motivation behind the proposed architecture. Section 3 and 4 outline the re-view of the proposed architectures in the related research and the challenges faced by the offloading techniques proposed. In Section 5 we describe the new offloading architecture to cater the challenges discussed in the above sections. Towards the end, we give the details about the working of our framework. In Section 6 the benefits of the proposed architecture are discussed and the paper is concluded in Section 7.

II. MOTIVATION

The applications and features of smartphones are increasing day by day because the usage of these feature rich

phones is increasing. People are replacing their lap-tops and personal computers with these smartphones, thus the demand for processing and memory is increasing. These phones use a battery as their power source which has a limited capacity as compared to plug in devices like personal computers.

Some of the major problems faced by the smartphone users nowadays are listed as follows. Firstly, the applications using heavy graphics, memory or CPU result in a lot of battery drainage. Secondly, due to the small size of the phones the processing power, memory and battery are limited and these phones are not able to perform compute intensive tasks which our laptops or desktops could perform. The solution to these problems is either to increase the size of battery, processing power of the CPU and the size of memory which in turn results in increased size and cost of the phone or to use the resources of the cloud to execute the heavy applications thus saving the phone's scarce resources.

Cloud computing on the other hand provides computing resources (hardware and software) as a service through internet. We can use the resources such as memory, processing power in a pay per use environment. The major motivation for this paper is to use the computing resources provided as a service by the cloud to run the resource exhaustive applications of the smart-phones connected to the Cloud through internet. In the past couple of years some techniques have been proposed to partially or completely offload the applications on to the cloud. We will be discussing those techniques in the next section and the challenges faced by the offloading techniques.

III. REVIEW OF PROPOSED ARCHITECTURES IN THE RELATED RESEARCH

Quite a few approaches have been proposed for offloading applications from a smartphone to the cloud, which includes offloading the complete application, offloading an application partially. Related work in the field of offloading applications from android phones to the cloud have been discussed below [2-8].

Year 2009 witnessed the proposal of Augmented Smartphone Applications Through Clone Cloud Execution by Byung-Gon Chun and Petros Maniatis[3]. This research proposed to augment the smartphone's capabilities by off-loading an application partially or completely to a clone Smartphone. A clone is a virtual system on the cloud running the same operating system as that of the phone using hardware from the cloud's pool of hardware. The application is offloaded partially because only the part of the application which is compute intensive is to be offloaded and thus reducing the load on the Smartphone. While the compute intensive part is being executed by the clone the actual phone executes the remaining application. After the clone is finished with the execution of the compute intensive part of the application it returns the results to the actual phone. The phone processes the results as required and provides the user with the results. The pro-posed architecture includes a Controller and a Replicator installed in the actual phone and an Augmenter and Replicator installed on the clone. The Replicator synchronizes the changes in the phone software and state

to the clone. The controller offloads the application from the Smartphone to the clone and merges back the results from the clone to the phone. The Augmenter running in the clone man-ages the local execution, and returns a result to the actual phone.

Following the vision provided by the above research, Byung-Gon Chun et al. [4] in the year 2011 implemented an architecture named CloneCloud for offloading an application partially to its clone in the cloud. This scheme uses a partition analyzer which partitions the application to be offloaded for remote execution. The partition analyzer has a static analyzer which discovers the possible migration points and the constraints for migration and a dynamic profiler to build a cost model for execution and migration. The partition analyzer helps the migration unit to migrate and re-integrate the application at the chosen points. The migration unit comprises of a migrator, node manager and a partition database. The migrator provides the part of application to be migrated to the node manager which migrates the part to the clone and an entry is made to the partition database which helps in re-integrating the partitioned application.

In the year 2011, a new approach of offloading the applications from android smartphone to the cloud was introduced by Eric Y. Chen and Mistutaka Itoh named Virtual Smartphone over IP [5]. In this approach the complete application was offloaded from the android smartphone to the cloud. In this approach it was proposed to provide cloud computing environment specifically tailored for smart-phone users. This architecture allows users of smartphones to create virtual smart-phone images in the cloud and install and run their applications in these images remotely. The user can create a number of smart phone images using a dedicated server for each user.

In 2012, Eric Y. Chen et al. [6] introduced a framework for offloading heavy back-end tasks of a standalone android application to an android virtual machine in the cloud, which is an extension of Virtual Smartphone over IP. This architecture uses android interface definition language in order to offload without modifying the source code. This framework incorporates a dedicated server for each client to offload their application [5].This architecture divides an application into two parts i.e. GUI and a compute intensive component and it offloads only the compute intensive component to the android virtual machine. This framework comprises of basically three components a helper tool, a service offloader and a virtual machine. The helper tool has to be integrated to the application code at the time of development by the developer. The service offloader has to be installed to the android phone by downloading it from the application market and each user needs to have at least one dedicated instance of the android virtual machine which is the virtual image of the phone hosted on the cloud. The helper tool generates two copies of the application i.e. one for local execution and one for remote execution and calls the service offloader which then analyzes the cost of

remote execution and local execution. If the cost of remote execution is less than the cost of local execution then the service is offloaded otherwise it is not offloaded to the virtual android phone.

In 2010, Georgios Portokalidis et al. [7] proposed a new scheme named Paranoid Android to provide security to android phones by applying security checks on remote security servers that host exact replicas of the smartphones in virtual environments. As the remote servers are not constrained with battery or processing power, multiple detection schemes could be applied simultaneously. On the phone, a tracer records all information needed to accurately replay its execution. The recorded execution trace is transmitted to the cloud over an encrypted channel, where a replica of the phone is running on an emulator. On the cloud, a rep-layer receives the trace and faithfully replays the execution within the emulator.

In 2015 , Hao Qian, Daniel Andresen et al.proposed a scheme named Jade , an energy- aware computation offloading system for mobile devices. Jade, built for mobile devices running Andriod operating system , minimizes energy consumption of mobile applications through fine grained consumption offloading . Jade monitors device and application status and automatically decides where code should be executed. Jade dynamically adjusts offloading strategy by adapting to workload variation, communication costs, and energy status in a distributed network of Android devices. Jade minimizes the burden on developers to build applications with computation offloading ability by providing easy-to-use Jade API. Evaluation shows that Jade can effectively reduce up to 39% of average power consumption for mobile application while improving application performance.[19]

IV. CHALLENGES

The main problem which is identified in the previous architectures is that offloading a compute intensive application partially can improve the battery life of a smartphone, but the offloading system will incur some overhead on the phone especially if the offloading decision is taken for a large number of applications when only a few number of applications are actually required to be offloaded.

The architectures discussed in the above section need to make changes in the binary of the application at the time of execution, to make it offloadable. To make changes at the binary level, the program loader has to be changed which may result in security vulnerabilities and to analyze the binary of an application could be compute intensive thus imposing overhead on the phone.

V. PROPOSED ARCHITECTURE

In this paper we present a framework for automated offloading of compute intensive applications of android smartphones to the virtual image of the smartphone on the cloud. An offloading framework is proposed which

if used by the developers of the application, will empower the application to offload its compute intensive,non interactive parts based on static analysis to the smartphone image on the cloud. The static analysis is done to make the decision making more fast and light than the previous techniques.

This framework proposes to make an application autonomous for offloading it-self from the smartphone to the smartphone image on the cloud. An application will be divided into two major parts i.e. a user interactive part which takes the in-puts from the user and provides the output to the user and a compute intensive non interactive part which does the computations as shown in the fig. 1. This frame-work will empower the application to offload its compute intensive part to the cloud via internet after analyzing the cost of offloading over the cost of running the application on the phone itself. The analysis will be done using parameters like input size and internet connectivity. By using this framework the developers will empower the applications to offload themselves without the need of some other application to analyze and offload parts of the application.

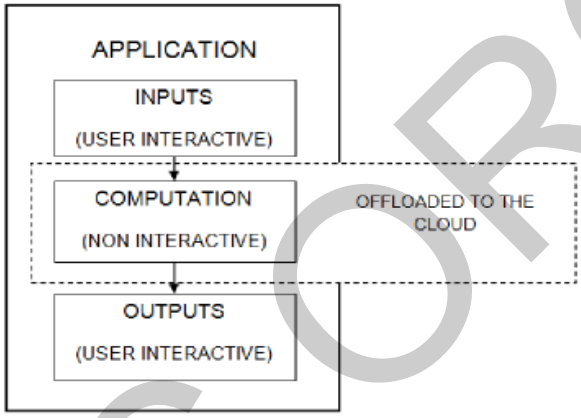

Fig. 1: Partitioning of Applications into Interactive and Non-Interactive Parts

The fig. 2 shows the working of the proposed architecture where the offloading decision is taken by the application itself after analyzing the parameters discussed above. According to the offloading decision the application is partially offloaded to the virtual phone on the cloud. The cloud performs the compute intensive tasks and returns the results to the phone. Thus saving the resources of the phone and increasing the capabilities of the phone.

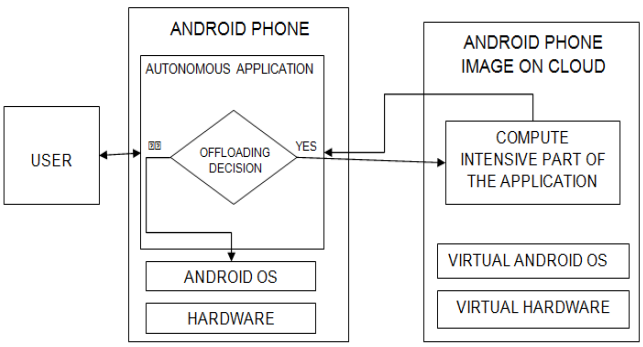

Fig. 2: Framework of Autonomous Computation Offloading Application for Android Phones

VI. Advantages of the Proposed Architecture

The major advantage of using this framework is that the offloading decision will be taken by the application itself. This decision making will have to be inserted in the source code while writing the applications which is advantageous over the previous approaches in the following ways. Firstly, the approaches discussed earlier modified the applications at binary level using modifications in the loader and thus increasing the security vulnerabilities. Secondly, they analyzed each application's binary to offload it, but only a few compute intensive applications need to be offloaded thus causing an overhead on the phone's resources. By using this technique the offloading decision will be taken only for the compute intensive applications which need to be offloaded and the application developers will not use this offloading framework for small applications. This technique will need to modify the applications at the development stage and will not modify the applications binary thus eradicating the need of changing the application loader.

VII. CONCLUSION AND FUTURE WORK

In this paper, we explored the design of a framework which makes an application autonomous to offload its compute intensive part to the cloud thus saving the re-sources of the android phone. This framework makes changes in the application at the development time thus eradicating the need to make changes in the application's binary. The application will make the offloading decision using static analysis.

Our future work includes implementation and evaluation of this design, adding dynamic analysis to support the offloading decision and comparing the performance of both frameworks using static analysis and the one using dynamic analysis. We are also planning to incorporate security, privacy, and trust related models [15-18] in the proposed framework.

REFERENCES

[1] Health and Wellness– Phone Apps. University ofCalifornia. http://wellness.ucr.edu/Wellness%20Apps%20Resources.pdf

[2] A. Rudenko, P. Reiher, G. J. Popek, and G. H. Kuenning.: Saving portable computer battery power through remote process execution. In MCCR'98---ACM SIGMOBILE Mobile Computing and Communications Review Newsletter. Vol. 2,no. 1, 19-26(1998).

[3] B.G. Chun, P. Maniatis.: Augmented Smartphone Applications Through Clone Cloud Execution. In Proceedings of 12th conference on Hot topics in operating systems. 8-8(2009).

[4] B.G. Chun, S. Ihm, P. Manitis, M. Naik and A. Patti.: CloneCloud: Elastic Execution between Mobile Device and Cloud. In Proceedings of 6th Conference on Computer Systems. 301-314(2011).

[5] E.Y. Chen and M. Itoh.: Virtual Smartphone over IP. In Proceedings of IEEE international conference on World of Wireless Mobile and Multimedia Networks. 1-6(20 10).

[6] E. Chen, S. Ogata and K. Horikava.: Offloading Android Applications to the Cloud. In Proceedings of

IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops). 788-793(20 12).

[7] G. Portokalidis, P. Homburg, K. Anagnostakis and H. Bos.: Paranoid Android: Versatile Protection For Smartphones. In Proceedings of the 26th Annual Computer Security Applications Conference. 347-356(20 10).

[8] E. Cuervo, A. Balasubramanian, D.K. Cho, A. Wolman, S. Saroiu, R. Chandra and P. Bahl.: MAUI: Making Smartphones Last Longer with Code Offload. In Proceedings of the 8th international conference on Mobile systems, applications, and services. 49-62(20 10).

[9] A. Saarinen, M. Siekkinen, Y. Xiao, J. K. Nurminen, M. Kemppainen and P. Hui. Smart-Diet: Offloading Popular Apps to Save Energy. ACM SIGCOMM Conference. 297-298(2012).

[10] W. SONG and X. SU.: Review of Mobile cloud computing. In IEEE 3'rd international conference on Communication Software and Networks. 1-4(2011).

[11] J. Peng, X. Zhang, Z. Lei, B. Zhang, W. Zhang and Q. Li.: Comparison of Several Cloud Computing Platforms. In Proceedings of IEEE international conference on Information Science and Engineering. 23-27(2009).

[12] National Institute of Science and Technology.: The NIST Definition of Cloud Computing. http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf (2010).

[13] R. Buyya, J. Broberg and A.Goscinski.: Introduction to Cloud Computing. In Cloud Computing: Principles and Paradigms. Wiley Press [Online]. 1-44. http://media.johnwiley.com.au/product_data/excerpt/90/04708879/0470887990-180.pdf (2009).

[14] Android Open Source Project.Philosophy and Goals. Google. http://source.android.com/about/philosophy.html (2012).

[15] S. Singh and S. Bawa.: A Privacy, Trust and Policy based Authorization Framework for Services in Distributed Environments. International Journal of Computer Science. Vol. 2, no. 1, 85-92 (2007).

[16] S. Singh and S. Bawa.: A Framework for Handling Security Issues in Grid Environment using Web Services Security Specifications. In Second International Conference on Semantics, Knowledge and Grid, 2006, SKG'06, Guilin, China. 68 (2008).

[17] G. Singh and S. Singh.: A Comparative Study of Privacy Mechanisms and a Novel Privacy Mechanism [Short Paper]. In ICICS'09 Proceedings of the 11th International Conference on Information and Communication Security, Beijing, China. 346-358 (2009).

[18] S. Singh.: Trust Based Authorization Framework for Grid Services. Journal of Emerging Trends in Computing and Information Sciences. (2011). Vol. 2, No. 3, 136-144.

[19] Hao Qian, Daniel Andresen , "Jade: Reducing Energy Consumption of Android App," in ternational Journal of Networked and Distributed Computing, August 2015.