# MobiCloud: a Geo-distributed Mobile Cloud Computing Platform

Tianyi Xing[1], Dijiang Huang[1], Shingo Ata[2], Deep Medhi[3]
[1]Arizona State University, U.S.A,{tianyi.xing,dijiang}@asu.edu
[2]Osaka City University, Japan, ata@info.eng.osaka-cu.ac.jp
[3]University of Missouri–Kansas City, U.S.A, dmedhi@umkc.edu

***Abstract*—In a cloud computing environment, users prefer to migrate their locally processing workloads onto the cloud where more resources with better performance can be expected. ProtoGENI [1] and PlanetLab [17] have further improved the current Internet-based resource outsourcing by allowing end users to construct a virtual network system through virtualization and programmable networking technologies. However, to the best of our knowledge, there is no such general service or resource provisioning platform designated for mobile devices. In this paper, we present a new design and implementation of MobiCloud that is a geo-distributed mobile cloud computing platform. The discussions of the system components, infrastructure, management, implementation flow, and service scenarios are followed by an example on how to experience the MobiCloud system.**

## I. Introduction

Cloud computing has grown rapidly in the past few years due to the increasing network bandwidth, mature virtualization techniques, and emerging cloud based business demands. What is more, by 2013, mobile devices will overtake PCs as the most common web access entities worldwide as predicted by Gartner [15]. Thus, a mix of cloud computing with mobile technologies is highly expected. Mobile Cloud Computing (MCC) is a term that refers to an infrastructure where both data storage and data processing are done, outside of mobile devices from which an application is launched. Besides that, a mobile entity is not limited to only a mobile device; more importantly, it could also be cloud resources, infrastructure, services, and human beings. Hence, with this understanding, MCC further refers to a cloud system where mobility happens in infrastructure, resources, services, user devices and even human beings.

The trend of the MCC system is not just aimed at providing fixed services for users in certain areas, but is especially to look forward to establishing connections among mobile users all over the world. Due to the mobility of MCC users, a geographically distributed cloud system is a natural choice that allows users to connect to cloud resources that are geographically "close" to their mobile devices, which usually means less communication delay compared to the centralized approach.

In this research work, our goal is to establish a new MCC service for mobile users with reduced service access latency and increased cloud infrastructure utilization. We present a new geo-distributed MCC infrastructure, called MobiCloud, to address the following research challenges: 1) Resource diversity – Virtual machines are the main resources due to its efficient utilization of hardware resources. However, current service providers pay more attention to system architecture design but not enough to resources provisioning diversity, which means users have little choice on resources. The resource diversity includes not only the type of guest OS, but also different virtual hardware configurations, 2) Network programmability and sensibility – A static and simple network cannot meet the increasing demand from users nowadays. Network programmability usually separates the control and data path of network devices, and provides an interface of the control path so that the control function can be easily programmed. Sensibility provides a bridge connecting cloud resources and the physical world. Sensibility greatly expands the range of experiments supported in the system.

It is essential to propose a cloud provisioning platform by considering mobile users and experimenters to solve the problems raised by the emergence of the popular geo-distributed cloud systems. In this paper, we propose MobiCloud, a geo-distributed MCC resource provisioning system. The major contributions of this paper are summarized as below. We design the MobiCloud framework, define its service model, and propose a novel extension model CaaS. Then, components are implemented and a geo-distributed infrastructure is established. A concrete example on how to experience the MobiCloud system is given to better understand what MobiCloud can support. MobiCloud is a geo-distributed MCC service provisioning platform including elastic computing, secure storage, and layer-2 and above networking capabilities.

This platform is currently implemented connecting three sites located at Arizona State University (ASU), University of Missouri–Kansas City (UMKC) in the USA and Osaka City University (OCU) in Japan. Additional sites in Paris and Beijing are under plan to be part of the MobiCloud system.

The rest of this paper is organized as follows. Section II addresses related work. Section III discusses the design and implementation of the proposed MobiCloud system. An example of how to experience the MobiCloud is illustrated in section IV. Section V concludes the paper and discusses the future work.

## II. Related Work

GENI [5], A Global Environment for Network Innovations, is a project exploring the future global networking infrastructure where different types of resource provisioning platforms are residing. GENI's projects can be divided into backbone networks, programmable hosts, wireless testbeds,

978-3-901882-48-7 

TABLE I
COMPARISON TABLE OF GENI PROJECTS

| Project | Major Resource | Sensing Capability | Programmable Networks | Extension Simplicity |
|---|---|---|---|---|
| PlanetLab | Fedora VM | No | No | Difficult |
| ProtoGENI (Emulab) | PC and VM | **USRP** | **Yes** | Difficult |
| OpenFlow Networks | OF Switch | No | **Yes** | DD |
| GENICloud (opencirrus) | Physical node | No | No | NA |
| Seattle | Experimenter Software | No | No | **Easy** |
| ORBIT | Dedicated node | No | **Yes** | NA |
| DOME | Linux VM | No | No | NA |
| DETER (Emulab) | PC | No | **Yes** | NA |
| Kansei | Sensing node | **Yes** | No | NA |
| ViSE | Debian VM | **Yes** | **Yes** | NA |
| GpENI | Fedora VM | No | **Yes** | DD |
| MobiCloud | Windows, Linux VMs | **Yes** | **Yes** | **CaaS** |

DD: Dedicated Device, NA: Not Allowed

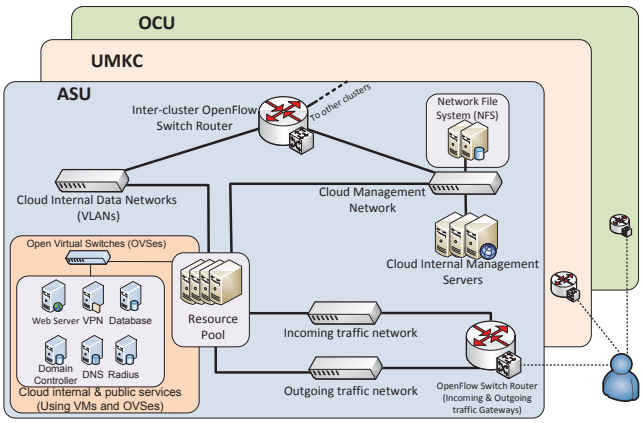


Fig. 1. MobiCloud Architecture Design

and specialized aggregates. Different GENI platforms, e.g., PlanetLab [17], ProtoGENI [1], and OpenFlow Networks [16] have different concentration in terms of provisioning resources, network architecture, programmable networks, etc. For example, ProtoGENI has integrated a large group of resources available from the world to provide resources with network programmability and sensing features. Seattle [10] has an efficient design that can easily make spare nodes join their available resource pool to be further utilized to provide python based experiments. All related GENI projects [10], [9], [18], [19], [8], [7], [14], [20] and our proposed MobiCloud are summarized in Table I.

The proposed and implemented MobiCloud system is originally based on the work discussed in [11]. The major differences between this work and [11] are in the following four perspectives: 1) This paper improves the original design significantly in terms of network connectivity, data storage, and so on; For example, it now allows the network architecture to span in a geo-distributed fashion and introduces the network-based remote storage; 2) This paper presents a fully implemented cloud computing management system but not just a design; 3) This paper establishes a monitoring system for monitoring resources and presents a performance benchmark; 4) This paper identifys practical issues, which need to be implemented in the near future. [12] concentrates on secure data processing based on the overall MobiCloud architecture. To the best of our knowledge, MobiCloud is the only MCC system that is able to address all challenges listed above.

## III. MOBICLOUD SYSTEM DESIGN AND IMPLEMENTATION

This section discusses the MobiCloud platform mainly in terms of components and implementation flow.

### A. System Components

The prototype of the proposed MobiCloud system is now available is presented in Fig. 1. We partition the MobiCloud system into different types of components including computing, storage, administrative, and networking.

**Computing Component**. The computing component is the entity that provides computing resources, i.e., cloud hosts. The major difference among different computing components depends on the virtualization technology being adopted. Virtualization in MobiCloud is based on XEN [**?**] that has impressive scalability and efficiency. A cloud system can provide logically separate resources upon the virtualization layer. Usually, Cloud resources in one domain are grouped into a resource pool that always has at least one physical node known as the *master node*. Other physical nodes that join existing pools are described as in *slave nodes*. Only the master node exposes an administration interface and forwards commands to individual slaves as necessary.

**Storage Component**. Storage cumulates all resource images and users' data. Resource is prepared by cloning resource templates that are stored in the storage repository. We choose to establish a remote storage repository, Network File System (NFS), to manage the storage of resources in our cloud system. An NFS storage server is connected to the computing server via a switch that greatly increases the scalability of storage.

**Administrative Component**. Dedicated physical servers are for administrating resources and monitoring network traffic within and across domains. There is also a set of internal functional servers serving various administrative purposes such as web service, DHCP, DNS, authentication service, DB service, and VPN.

**Networking Component**. The control plane and the data plane are isolated based on the multi-network design of MobiCloud. In Fig. 1, there are four networks in each cluster. Incoming and outgoing traffic switches isolate control traffic (i.e., resource access, OS update, and package download) coming into or going out of the MobiCloud Gateway. The data network switch is a managed switch with supper for VLAN that enables VMs from different physical servers to reside in the same virtual domain. Lastly, the cloud management network connects the internal management and monitoring server and NFS. Each cloud server is installed with an Open vSwitch with which the data traffic between two VMs in the same physical server does not need to go through the physical data network switch out of the cloud host.

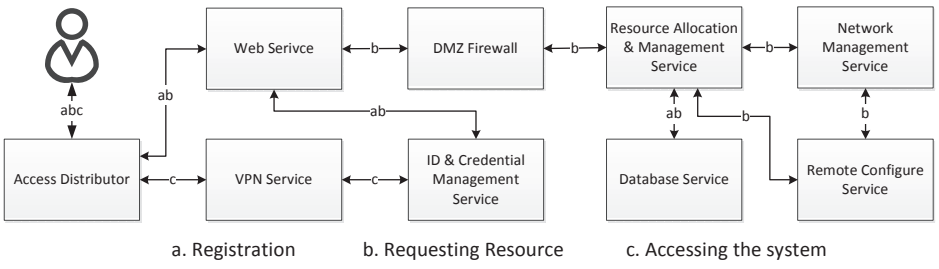


Fig. 2. Processing flows in MobiCloud

### B. Implementation Flow

After designing the framework and configuring the infrastructure, we have implemented the system to provide resource provisioning services to users. The implementation flow is defined in terms of three major aspects: registration, requesting resources, and accessing the system. These three procedures are described below and are shown in Fig. 2.

- **Registration**. Anyone who wants to use the MobiCloud system must have a valid account. Users have to register an account by visiting the MobiCloud portal. After users create an account, the account credential will be automatically stored in the ID and Credential Management Service, which is implemented by the Kerberos based Active Directory, and dedicated Databased Service.
- **Requesting Resource**. After users' credentials are authenticated, users are authorized to request resources from *Resource Allocation & Management Service*. Although all services are hosted in the MobiCloud private domain, we still introduce a DMZ to enhance security of cloud resources. *Database Service* stores all related information of newly created resources, and *Network Management Service* prepares and configures network attributes of resources. So far, *Network Management Service* consists of DHCP and Dynamic DNS. Users are also able to configure network attributes of resources by themselves, which is the reason why *Remote Configure Service* was developed.
- **Accessing Resource**. When users are about to access their created resources, they have to connect to the VPN server through the gateway. The VPN server authenticates the users' accounts at *ID&Credential Management Service* where users' account credentials are stored. After connecting to the VPN, users' mobile devices are assigned with the MobiCloud private IP address and are free to access their resources by using the corresponding domain name assigned for each VM.

## IV. Experiencing MobiCloud

In this section, we present the implementation of MobiCloud through a simple example of its use. Through our developed web interface, users are free to realize all operations (register, request resource, and access resource) we mentioned in the previous section. The web portal can be accessed at *http://mobicloud.asu.edu*.

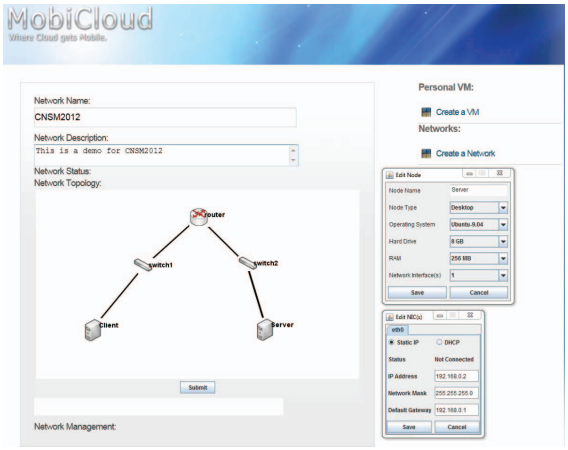


Fig. 3. Virtual network creation

### A. Single VM Creation

To create a single VM, there are several fields that need to be configured by users, which can be found in Table II. After a user specifies all the parameters in the corresponding fields, a VM will be created at the backend side. One of the advantages of MobiCloud over current existing platforms is that the response time of the resource preparation is swift. Because the VM template is already prepared, what the system only needs to do involves two steps: 1) fast copy the VM template and 2) configure the VM. When the VM is done being prepared, users can use the management interface, shown in Fig. 5(b), to manage their VMs. There are several options available such as restart, resume, suspend, stop, and remove. Each VM will be assigned with a unique domain name, in the format of *username.mobicloud.asu.edu* that is registered in the DNS system. A Single VM can also be set as a proxy of a mobile device to enhance its capability. For example, users can set their VMs as proxies with an anti-virus function enabled, which greatly enhances security of mobile devices.

### B. Virtual Network Creation

Besides creating a single VM, users are also allowed to claim multiple VMs that can be connected as a virtual network. The interface can be seen in Fig. 3. Users can simply click, drag, and drop on the canvas to create their desired network components and connect them. There are three major components in the network creation canvas: desktop, switch,

TABLE II
SINGLE VM CREATION SPECIFICATION

| VM Host Configuration | |
|---|---|
| VM name | Define the name of the VM. |
| OS Type | Specify the Operation System of VM. We use Xen Hypervisor, which supports a wide range of guest OS including Windows, Linux, Solaris, and various versions of the BSD operating systems. |
| Hard Disk Size | Specify the size of the virtual hard disk on each VM. There are several options ranging from 8GB to 128GB. |
| Ram Size | Specify the size of the memory. The range of memory can be selected from 128MB to 4GB. |
| VM Network Configuration | |
| NIC configure | Specify IP address, netmask, and default GW. |

and router. Users can configure the components in a pop-up window. Beside the virtual hardware configuration (i.e., CPU, memory, harddisk), network information can also be configured, including creating virtual interfaces, IP addresses, netmasks, default gateways, and so on. We use a switch component to represent a VLAN to partition the network. Users can draw lines to connect two or more components (i.e., desktop or router) through a switch, which means those interfaces are in the same VLAN and are isolated from others. The router is the last component with routing functions enabled by a software-based pre-installed router (Vyatta [6]). After the user confirms the topology by clicking the submit button, a summary page indicating all hardware and network information of VMs is presented. When the preparation is done, a resource page is returned with the details of the resources created. Each VM is assigned a unique domain name that is similar to the single VM creation case. However, each user can have multiple VMs; the domain naming scheme is the format of *username-networkname-number.mobicloud.asu.edu* to identify different VMs for the same user. For example, if user *cnsm2012* creates a network named *mynetwork* with three VMs, the domain name assigned to the first VM would be *cnsm2012-mynetwork-1.mobicloud.asu.edu*.

### C. System Resource Monitoring

For system administrators to better monitor performance of both network and cloud hosts in MobiCloud and track ongoing and potential issues in the system, we introduce a multi-layer monitoring mechanism that is enabled by *sFlow* and *NetFlow* [13]. Note that *NetFlow* provides limited visibility focused on layer-3 network connections, while *sFlow* provides comprehensive visibility into network and system resources needed to manage performance in a virtualized and cloud environment. In each Open vSwitch in Dom 0 of each Cloud server, we enable not only *sFlow* at the switch level but also at the host level, which means not only can the traffic be monitored, but also the host performance (i.e., CPU utilization, memory usage, virtual disk I/O, and so on) can be monitored.

From the analyzer of both NetFlow ManageEngine and sFlow Flowtrend [3], [2], network parameters, e.g., top conversation, top connections, most popular protocols and so on can be monitored. Besides layer-3 network monitoring, sFlow is able to inspect connection relationships from the following three aspects shown in Fig. 4: internal connections, external connections, and non-IP connections. The left-top part of Fig. 4 shows the internal connections that represent the internal

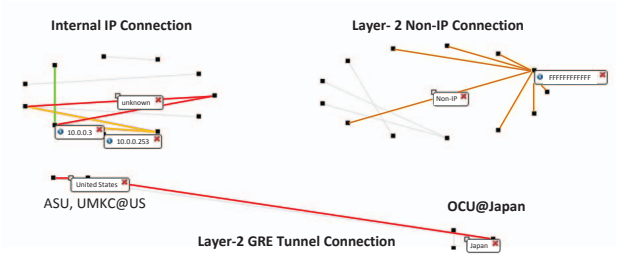


Fig. 4. sFlow Connection Circle

topology at the ASU site where the connections between the cloud server master node(10.0.0.3) and the storage server (10.0.0.253) can be seen. The top-right part shows the layer-2 Non-IP connections including unicast and broadcast. The bottom part shows the external connections that indicate the inter-cluster connection among all three sites.

## V. CONCLUSION AND FUTURE WORK

This paper takes presents an expected cloud environment for mobile devices. Our proposed MobiCloud system is able to provide resources in terms of computing, storage, and networking that greatly enhances the capability of mobile devices. We combine network-based storage, Xen virtualization, and OpenFlow based network management solutions into a single smart system, which has not been done previously to our best knowledge. Also, an example of system experience is given to better state the capabilities of MobiCloud.

In the near future, we are planing to implement OAuth 2.0 to improve the MobiCloud system. The OAuth 2.0 [4] authorization protocol enables a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service, or by allowing the third-party application to obtain access on its behalf. It helps manage MobiCloud users in accessing both inside and outside services. An integrate OAuth 2.0 solution will be an inevitable trend for an ID management system of MobiCloud.

GENI provides collaborative and exploratory environments for academia, industry and public, which is ideal for MobiCloud to extend on. The federation with GENI mainly would involve the following two steps: (a) following the Slice-based Facility Architecture (SFA) specification, (b) providing aggregate API and interfacing with other GENI major platforms. We plan to explore this in the near future.

## ACKNOWLEDGEMENT

This work is supported by US NSF grants CNS-1029562 and CNS-1029546, Office of Naval Researchs (ONR) Young Investigator Program (YIP), an HP IRP grant, and Japan NICT International Collaborative Research Grant.

## REFERENCES

[1] http://www.protogeni.net/trac/protogeni.
[2] inMon Flowtrend, http://www.inmon.com/products/sFlowTrend.php.
[3] ManageEngine, http://www.manageengine.com/.
[4] O-Auth Community, https://oauth.net/2.
[5] The Global Environment for Network Innovations (GENI), http://groups.geni.net.
[6] Vyatta, http://www.vyatta.com/.
[7] A. Arora, E. Ertin, R. Ramnath, M. Nesterenko, and W. Leal, "Kansei: A high-fidelity sensing testbed," *In Proceedings of the DETER Community Workshop on Cyber-Security and Test*, vol. 10, pp. 35–47, 2006.
[8] T. Benzel, R. Braden, D. Kim, and C. Neuman, "Design, deployment, and use of the deter testbed," in *In Proceedings of the DETER Community Workshop on Cyber-Security and Test*, August 2007.
[9] R. Campbell, I. Gupta, M. Heath, S. Ko, M. Kozuch, M. Kunze, T. Kwan, K. Lai, H. Y. Lee, M. Lyons, D. Milojicic, D. OHallaron, and Y. C. Soh, "Open cirrus: Cloud computing testbed: Federated data centers for open source systems and services research," in *Proceedings of the USENIX Hotcloud*, June 2009.
[10] J. Cappos, I. Beschastnikh, A. Krishnamurthy, and T. Anderson, "Seattle: A platform for educational cloud computing." in *The 40th Technical Symposium of the ACM Special Interest Group for Computer Science Education (SIGCSE)*, 2009.
[11] D. Huang, X. Zhang, M. Kang, and J. Luo, "Mobicloud: Building secure cloud framework for mobile computing and communication," in *Proceedings of 5th IEEE International Symposium on Service-Oriented System Engineering*, 2010.
[12] D. Huang, Z. Zhou, L. Xu, T. Xing, and Y. Zhong, "Secure data processing framework for mobile cloud computing," in *IEEE INFOCOM's Workshop on Cloud Computing*, 2011.
[13] N. Instruments, "Extending network visibility by leveraging netflow and sflow technologies," in *White Pape*, February 2011.
[14] D. Irwin, N. Sharma, P. Shenoy, and M. Zink, "Towards a virtualized sensing environment," in *Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom)*, 2010.
[15] Mark Walshy, "Gartner: Mobile to outpace desktop web by 2013," Online Media Daily, January 2010.
[16] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," in *ACM SIGCOMM Computer Communication Review*, April 2008.
[17] L. Peterson, A. Bavier, M. Fiuczynski, and S. Muir., "Experiences building planetlab," in *Proceedings of the Seventh Symposium on Operating System Design and Implementation (OSDI)*, November 2006.
[18] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh, "Overview of the orbit radio grid testbed for evaluation of next-generation wireless network protocols," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2005.
[19] H. Soroush, N. Banerjee, A. Balasubramanian, M. D. Corner, B. N. Levine, and B. Lynn, "Dome: A diverse outdoor mobile testbed," in *Workshop on Hot Topics of Planet-Scale Mobility Measurements (HotPlanet)*, June 2009.
[20] J. Sterbenz, D. Medhi, B. Ramamurthy, C. Scoglio, D. Hutchison, B. Plattner, T. Anjali, A. Scott, C. Buffington, G. Monaco, D. Gruenbacher, R. McMullen, J. Rohrer, J. Sherrell, P. Angu, R. Cherukuri, H. Qian, and N. Tare, "The Great Plains Environment for Network Innovation (GpENI): A programmable testbed for future Internet architecture research," in *Proc. of 6th International Conference on Testbeds and Research Infrastructures for the Development of Networks & Communities (TridentCom)*, Berlin, Germany, May 2010, pp. 428–441.