mVDI : A New Paradigm Shift for Mobile Cloud

뉴엔 티엔 융, 후인 콩 틴, 허의남
경희대학교 컴퓨터공학과
e-mail : {ntiendung, thinh, johnhuh}@khu.ac.kr

mVDI : A New Paradigm Shift for Mobile Cloud

Tien-Dung Nguyen, Cong-Thinh Huynh, Eui-Nam Huh
Dept. of Computer Engineering, Kyung Hee University
e-mail : {ntiendung, thinh, johnhuh}@khu.ac.kr

Abstract

Mobile Virtual Desktop Infrastructures (mVDI) are gaining popularity in cloud computing by allowing mobile devices to execute their mobile applications in a cloud server instead of relying on physical mobile devices. Consolidating many users into mVDI environment can significantly lower IT management expenses and enables new features such as "available-anywhere" desktops. However, there are many barriers to broad adoption including the slow performance of virtualized I/O, CPU scheduling interference problems. In this paper, we will discuss about mVDI with the current issues, the corresponding solutions and challenges.

1. Introduction

Cloud computing infrastructure has seen explosive growth in the last few years as a source of on-demand storage and server power. Beyond simply being used to run web applications and large data analytic jobs, the cloud is now being considered as an efficient source of resources for desktop users. According with the development of cloud computing, Virtual Desktop Infrastructure (VDI) is growing overwhelmingly. VDI is an alternative desktop delivery model that allows users to run an OS/application inside a Virtual Machine (VM) running in the data center. By utilizing network connected VMs, VDI can provide desktop services with easier management, greater availability, and lower cost.

Recently, with the rage of mobile devices including wireless Personal Digital Assistant (PDA), mobile phone, smartphone (integrated PDA with mobile phone), tablet, etc, they are opening the post-PC era. By using mobile devices, you can play and work anytime, anywhere. However, with the current limitation of mobile devices such as CPU, memory, storage, battery, etc, the applications (including on-demand movies, 3D games, graphical design tools, etc.) cannot perform efficiently. Because of this limitation, Mobile Virtual Desktop Infrastructure (mVDI) is proposed to address these issues. mVDI is the evolution of server virtualization supported for mobile devices. mVDI has the same look and feel of a traditional mobile device, but all actions are happening remotely in the cloud server. Fig. 1 presents the scenario of mVDI. On the server side, audio/video output from server would not be delivered to locally attached devices (as monitor and speaker), but redirected over the network to client to be displayed.

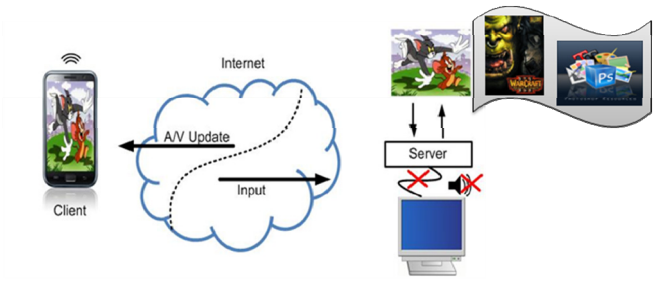

Figure 1. Mobile Virtual Desktop Infrastructure Scenario

In this paper, we discuss about the key technologies of mVDI in Section 2. We then suggest the difficulties, solutions and challenges in Section 3. Finally, we give some conclusions in Section 4.

2. Key Technology in mVDI

In mVDI, the key technology is exactly the communication protocol between mobile devices and cloud servers: remote control technology [1]. A

remote control system decouples a server from the mobile devices used to interact with it. In particular, the monitor, speaker keyboard, and mouse no longer need to be directly attached to the physical ports in the server in order to interact with it. Instead, a communication channel is provided between these mobile devices and the server via network connection. A/V output from the server, which would normally be sent to the local video hardware, is instead intercepted and redirected over the network to a client. Similarly, in response to the user interacting with the desktop at the client, input events are generated and sent back to the server. The client and server use a remote control protocol for this back and forth communication.

The remote control technology can be categorized into three distinctive groups based on three intercepted layers as Fig. 2 shows.

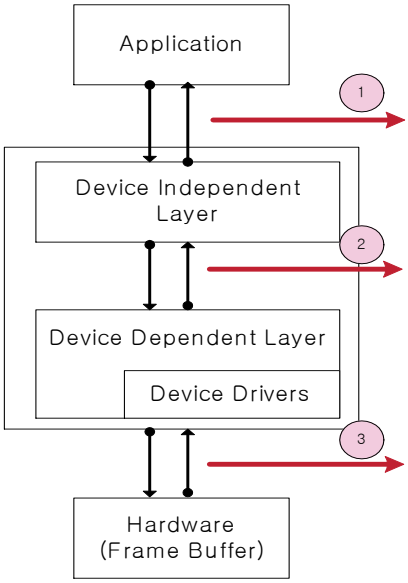

Figure 2. Display Intercept Layer

At application/OS layer (1), Remote Desktop Protocol (RDP) [2] is a typical protocol developed by Microsoft, which concerns providing a user with a graphical interface to another computer. The advantage of interception at layer 1 is no trouble to isolate data for each application that is beneficial way to support multi-users with multi-applications. However, client and server must be tight-coupling model.

At device driver layer (2), THINC uses its virtual device driver to intercept display output in an application and OS agnostic manner [3]. Then server uses that library to convert video/audio to other format and delivery to client. It efficiently translates high-level application display requests to low-level protocol primitives and achieves efficient network usage. However, client's graphic hardware processing capability is mandatory to decode video frame and display on screen.

At hardware frame buffer layer (3), VNC uses the RFB protocol [4] to remotely control another computer. The benefit of layer 3 interception is that server only delivers the graphical screen (raw data) to the client. Accordingly, client only need to open this raw data without supporting hardware else. Therefore, interception at layer 3 is the key technology to leverage mobile devices to communicate with cloud servers.

3. Issues, Solutions and Challenges

a. M-Virtualization

Issue: The first issue comes from the virtualization for mobile devices. As discussed in [1], one mobile device remotes to the server by VNC which only supports single session and does not support audio properly. Therefore, we cannot use VNC to remote directly physical server. It is too expensive to provide the service because each mobile device will manage one physical server. On the other hand, we can virtualize the physical server by some Hypervisor such as Xen, VMWare, VirtualBox, etc. Then mobile devices can remotely control each VM located in the physical machine. However, using the traditional OS consumes more CPU. Fig. 3 shows that one server is able to virtualize maximum 2 VMs.

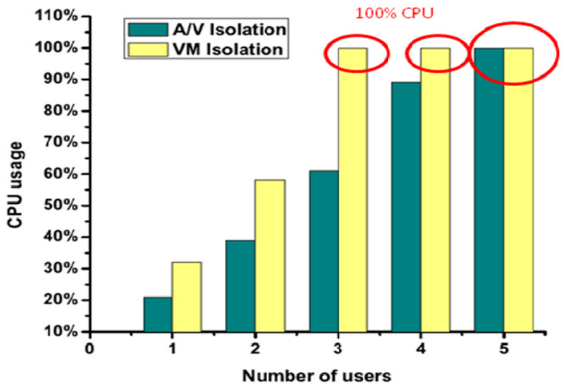

Figure 3. VM isolation vs. A/V isolation

Solution: In our previous work [5], we propose one method to virtualize the physical server, so-called A/V isolation. There are two modules in this solution. The former is video isolation technology. It assigns a non-overlapping rectangle area for each user session. As client controls the remote screen, the coordinates of the

corresponding rectangle will be captured and delivered to the client. The latter is audio isolation technology (as shown in Fig. 4). It includes three parts: 1) using Virtual Audio Driver (VAC) to store data buffer of each session, 2) using Windows Audio Session API (WASAPI) to hook audio buffer from VAC and 3) using RTP protocol to deliver the data buffer to the corresponding client.

Challenge: 1) In term of the profit of cloud provider, how to increase the number of mobile devices which can be provided by one cloud server. 2) How to schedule in cloud server to guarantee quality of real-time service and Service Level Agreement (SLA).

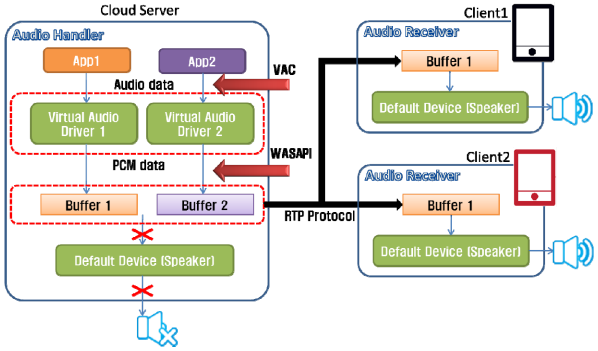

Figure 4. Audio Isolation Technology

b. Network/CPU Consumption

Issue: The performance table 1 shows that when we use those technologies to deal with common document operation or browse the website, the bandwidth consumption and resource consumption can be supported by current mobile network and mobile terminal device [6]. Then the display quality is good. But when we switch to video service, the current environment may provide the worse performance.

Scenario	Metrics	VNC-Tight	VNC-Auto
Website	Bandwidth Client CPU	58KBps 1.3%	370KBps 2.3%
Video (640x480)	Banwidth Client CPU QoE	1.69MBps 7% Bad	5.19MBps 5.7% Good

Table 1. Network Consumption

Solution: In [6], Wei Tang et al. proposed the hybrid remote display protocol, in which it utilizes the MJPEG codec to compress the data. MJPEG codec can reduce size of output, thus we can guarantee the display quality (as shown in Fig. 5).

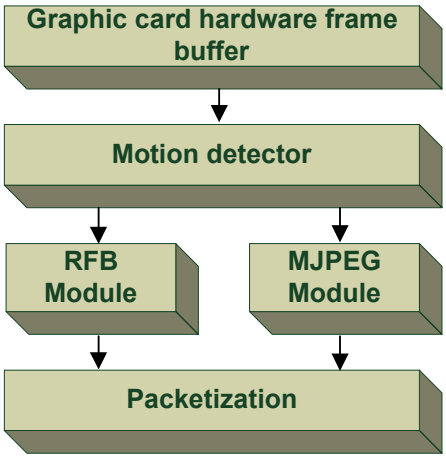

Figure 5. MJPEG Encoding

However, MJPEG encoding costs too much CPU resources. [6] also proposed GPU acceleration module to utilize GPU to reduce computation on CPU.

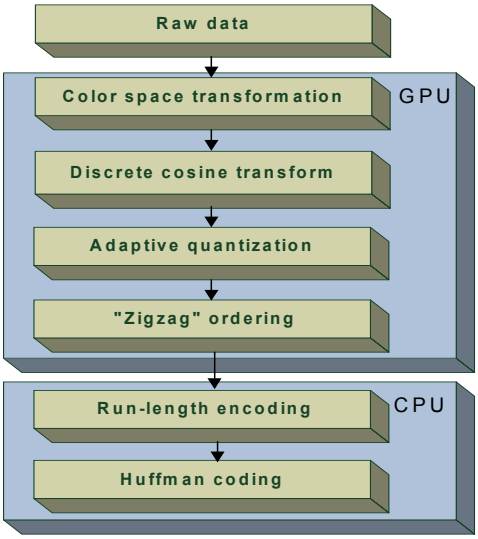

Figure 6. GPU acceleration

Challenge: 1) In Fig. 5, motion detector module is very important to figure out slow motion or high motion. Hence, how to enhance accuracy of motion detector module to improve the performance of cloud server? 2) How to establish the parallel GPUs system to accelerate the computation on GPU?

c. NDK applications Incompatibility

Issue: In mVDI, the mobile OS is often used to install on cloud server (eg. Androidx86). Mobile OS is modified to run on x86-based CPU instead of ARM-based CPU. However, the Androidx86 is incompatible with NDK applications such as Angry Bird.

Solution: The reason why the NDK application cannot run in Androidx86 is the incompatibility between x86-based and ARM-based CPUs. Because native libraries are built only for ARM-based CPU, Dalvik cannot invoke the native functions from shared library objects. We insert the translation binary libraries into system library of Androidx86 (as shown in Fig. 7).

Challenge: Though the NDK applications can be executed properly, the performance is quite slow. How to improve the performance of NDK applications in Androidx86?

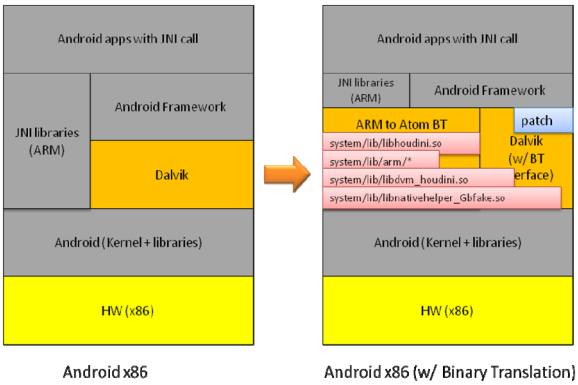

Figure 7. Androidx86 Binary Translation

d. Web-based mVDI

Issue: The current VDI and mVDI are implemented as applications. They limit the seamless and mobility of mobile devices. Hence, web-based mVDI should be implemented so as to any mobile device can access easily without any cross platforms problem.

Solution: We propose a system (so-called iConnect) which is .NET-based web-server. It supports mVDI to control VMs through tunneling/port forwarding. We build the datacenter with multiple Xen servers, thus we use XenAPI to monitor and manage the XenServers. The system also supports social network functions.

Challenge: HTML5 is predicted as the key protocol for mobile cloud computing with a lot of advantages such as supporting multimedia, hardware integration, user interactions, data storage offline, network enhancement. Therefore, 1) how to use HTML5 to implement mVDI is the attracting topic. 2) How do we enhance the social network function in mVDI? 3) How do we utilize mobile device's resource such as sensors, camera, GPS, etc.?

4. Conclusion

In this paper, we present some key issues in mVDI system. Based on each issue, we also discuss about the appropriate solutions. Besides, we give some challenges in mVDI for the future researches. With these challenges, we can build the mVDI efficiently.

Acknowledgement

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology (2012- 0006418). Professor Eui-Nam Huh is corresponding author.

Reference

[1] Tien-Dung Nguyen, Song Biao, Tang Wei, Jun-Hyung Lee, Eui-Nam Huh, Client Technology on a Server for Mobile Cloud, Information and Communications Magazine, 10. 2011, vol.28, no.10, pp.3-10

[2] Remote Desktop Protocol http://en.wikipedia.org/wiki/Remote_Desktop_Protocol

[3] Ricardo Baratto, "THINC: A Virtual and Remote Display Architecture for Desktop Computing and Mobile Devices", Ph.D. Thesis, Department of Computer Science, Columbia University, April 2011.

[4] "The Remote FrameBuffer Protocol" [Online]. Available: http://tools.ietf.org/html/rfc6143

[5] Tien-Dung Nguyen, Seungun Choe, and Eui-Nam Huh, An Efficient Mobile Thin Client Technology supporting multi-sessions remote control over VNC, 2012 IEEE International Conference on Computer Science and Automation Engineering., 06. 2012, vol.3.

[6] Wei Tang, Biao Song, Myeong Seob Kim, Nguyen Tien Dung, Eui Nam Huh, Hybrid Remote Display Protocol for mobile Thin Client Computing, 2012 IEEE International Conference on Computer Science and Automation Engineering, 05. 2012, vol.2, no., pp.435-439