ISSN 2348 – 9928
IJAICT Volume 2, Issue 9, September 2015 Doi:01.0401/ijaict.2014.03.16 Published Online 05 (09) 2015

COMPUTATION AND STORAGE USING CLOUD-BASED DESKTOP

Mr. Nirmal Zachariah
PG Scholar,
Computer Science and Engineering,
Mangalam College of Engineering,
Ettumanoor, Kerala, India

Prof. T. Syamamol
Assistant Professor,
Computer Science and Engineering,
Mangalam College of Engineering,
Ettumanoor, Kerala, India

Abstract— Cloud desktop application is molded on the SaaS cloud computing architecture which is an Open Source Platform designed to hold a wide variety of Web Applications. A new concept for desktop visualization, where a desktop can be accessed from anywhere within a seamless network, with the internet has been implemented. All that the clients need to do is to login to his/her own system with any internet browser, and then access personal desktop irrespective of location of the user and hardware used. By this system users can store and manipulate data values stored anywhere in the cloud. The user can select the remote machines based on a normalized list. Based on that list the user can select the remote machines for storage and these can do computation function by getting data stored in remote systems. When a computation request is passed then the server machine get the data for computation from different remote systems and then solve the computation. This system can be used in software enterprises.

Keywords — Cloud Desktop, Computation, Storage, Resource, Payment, Auditing.

I. INTRODUCTION

Cloud computing architectures are rapidly spreading over the world of Information Technology (IT), supporting the idea of provisioning various computing capabilities "as-a-service", in a transparent way for users [1]. Among those various "services", DaaS, deriving from SaaS, has drawn considerable attention in recent years due to its formation of a significant interface over the gap between clustered servers and clients. Usually implemented with desktop virtualization technology, it has the potential to offer a new, cost-efficient, scalable and comfortable service on demand. Cloud can have high utilization and thus great power efficiency. People are seeking for a green cloud computing architecture these years. Baliga et al.[2] compare the power consumption of different cloud paradigms and conventional computing. Berl et al.[3] review the usage of methods and technologies currently used for energy-efficient operation of computer hardware and network infrastructure.

It also identifies some of the key challenges in green cloud computing. Liu et al. [4] present a new green cloud architecture to reduce power consumption of data center is presented. All these have shown the increasing concern of green cloud computing.

With the development of technology, DaaS's potential is drawing people's attention from personal computer into a thin client's mobile platform, which has also been heavily impacted by cloud computing as well. Thousands of smart devices could access the internet, which will help drive the mobile cloud computing trend. In mobile systems, many devices have significant constraints imposed on them due to the requirement of smaller sizes, low power consumption and real-time issue.

To address these problems, a HW/SW code design of an FPGA-based system is proposed to implement the DaaS infrastructure, which would produce several advantages: thin cloud ends, low communication bandwidth, high power efficiency and dynamic web enabled interaction. FPGA-based prototype provides virtual desktop web pages written in HTML/ JavaScript with no special client software. The proposed system also merge JOP[6] and a web server to build DaaS server interacting with user's requests to copy with dynamic web pages, with which the user could further implement more complicated applications.

II. SYSTEM ARCHITECTURE

Existing cloud platforms fulfill the hardware requirements for implementing DaaS. However, an emerging category of mobile applications "including augmented reality, rich sensing, and multimedia editing" pose stringent requirements on delays. Current cloud management systems can't meet user expectations for these applications, especially in terms of latency. A clear need exists for novel cloud management algorithms that consider the specific requirements of mobile thin client computing. Proposed system architecture implements such algorithms in the service manager's self-component. The manager can be implemented as part of existing cloud management systems such as OpenNebula, OpenStack, and Eucalyptus.

ISSN 2348 – 9928
IJAICT Volume 2, Issue 9, September 2015 Doi:01.0401/ijaict.2014.03.16 Published Online 05 (09) 2015

Fig 1 shows the architecture of the system. Simplified OS image management (that is, re-using an OS image among users to reduce the storage per user) and application management are essential for the service to scale. The system builds a VD from a shared golden image from the OS database and augments it with personal settings for example, by using a copy-on-write solution with UnionFS (http://unionfs.filesystems. org). Multilayer VDs simplify the complexity of upgrading the golden image without causing broken dependencies or conflicts. To improve DaaS usability, thus combine DaaS with application virtualization technologies such as Softricity and Microsoft App-V. The system would then dynamically deliver applications to the user's VD without having to install, configure, and update them. This approach further reduces the complexity of upgrading golden images because applications aren't installed in the user's VD and thus can't be broken.

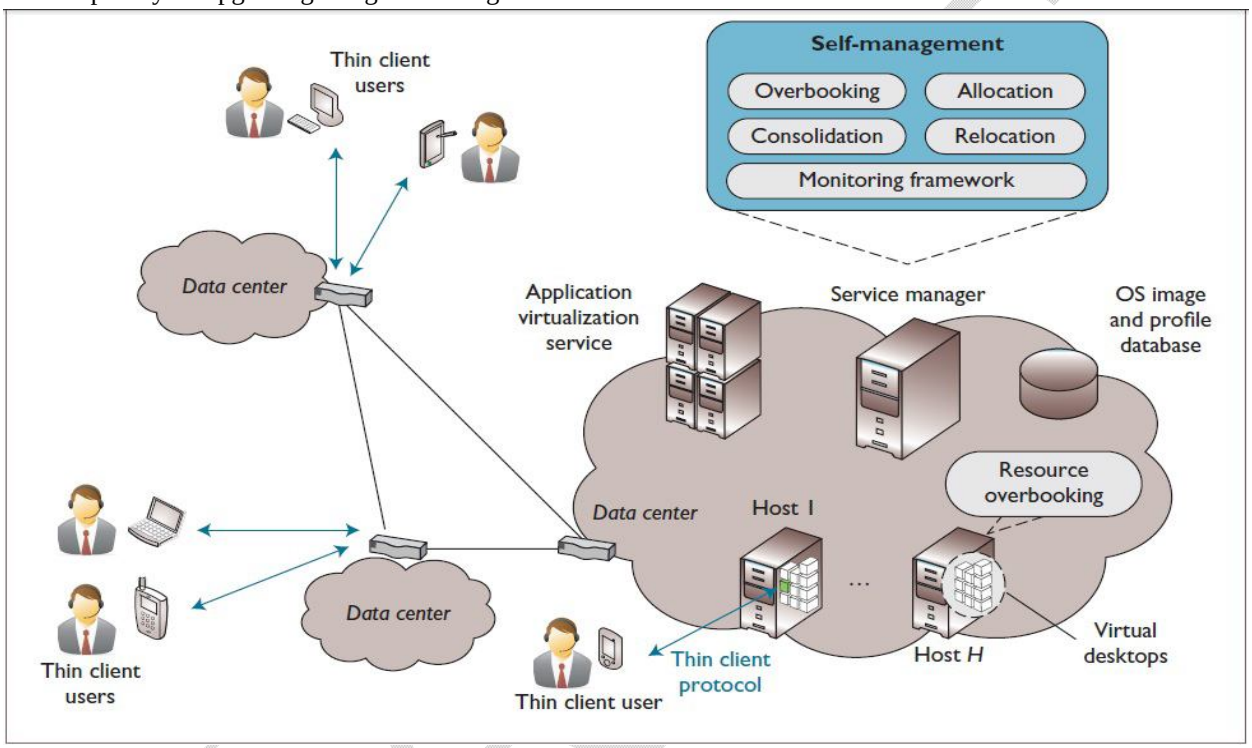

Fig 1: System Architecture for cloud based desktop

III. PROPOSED SYSTEM

Among the most related solutions, PC environment has always been the most popular platform. Though suffering limitations due to platform difference, the solution is implemented on FPGA for better power efficiency and an exploration of FPGA as a generally-purpose device. A prototype web system is developed using FPGA to store web pages, provide service and deal with requests. On architecture level, the system can be divided into two parts:

- The software parsing front-end
- The hardware-based back-end

3.1 Overall Architecture

Fig 2 shows the system architecture of prototyping system from a functional point of view. It consists of:

- The general client hardware
- The client application
- The remote web access engine
- The processing engine
- The data center

Internet, however, separates them into the so-called software parsing front-end in the top and the hardware-based back-end in the bottom. Despite all the actual implementation limits, this robust system view is provided with most common requirements considered, making provision for gradually porting onto FPGAs. People who sign in for service should have devices providing similar functional structure with (1) and (2). Such devices need

ISSN 2348 – 9928
IJAICT Volume 2, Issue 9, September 2015
Doi:01.0401/ijaict.2014.03.16 Published Online 05 (09) 2015

to do self-processing, access network, store some data and accept the users" input while present the output. Above such hardware layer, simple application should be adopted to cache remote libraries and configuration files while parsing them for output display and processing tasks. On the server side, the Remote Web Access engine, takes the role of users requests parsing, passing file content fragments to the data center while transferring special requests to certain commands for processing engine to cope with. Processing engine keeps real-time interactions with clients by transmitting library and configuration files. It also fetches commands to Processing Unit (PU) in which dynamic requests are handled, such as code compiling, scientific computing etc. Data center, on the other side, could be established on a separate cluster dealing with data backup and recovery.

3.2 Hardware Platform Partition

To estimate the feasibility of providing the services on FPGA as well as to pursue for less power consumption and better efficiency, thus combine the TCP/IP stack LwIP and the hardware processing unit JOP together to accelerate the processing speed and form the hardware web server environment.

3.3 Software Application Partition

To make the system platform-independent, the service should be web-accessible. Thus, focus should be on HTML and JavaScript which are composed into visible or audible web pages. HTML elements form the building blocks of all websites. It allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text. It also adopts Cascading Style Sheets (CSS) to define web page appearances and layouts while using JavaScript to affect web page behaviors. Although web pages are stored on server side, they work mostly the same despite platform difference. The web pages along with the front-end part are discussed as a whole

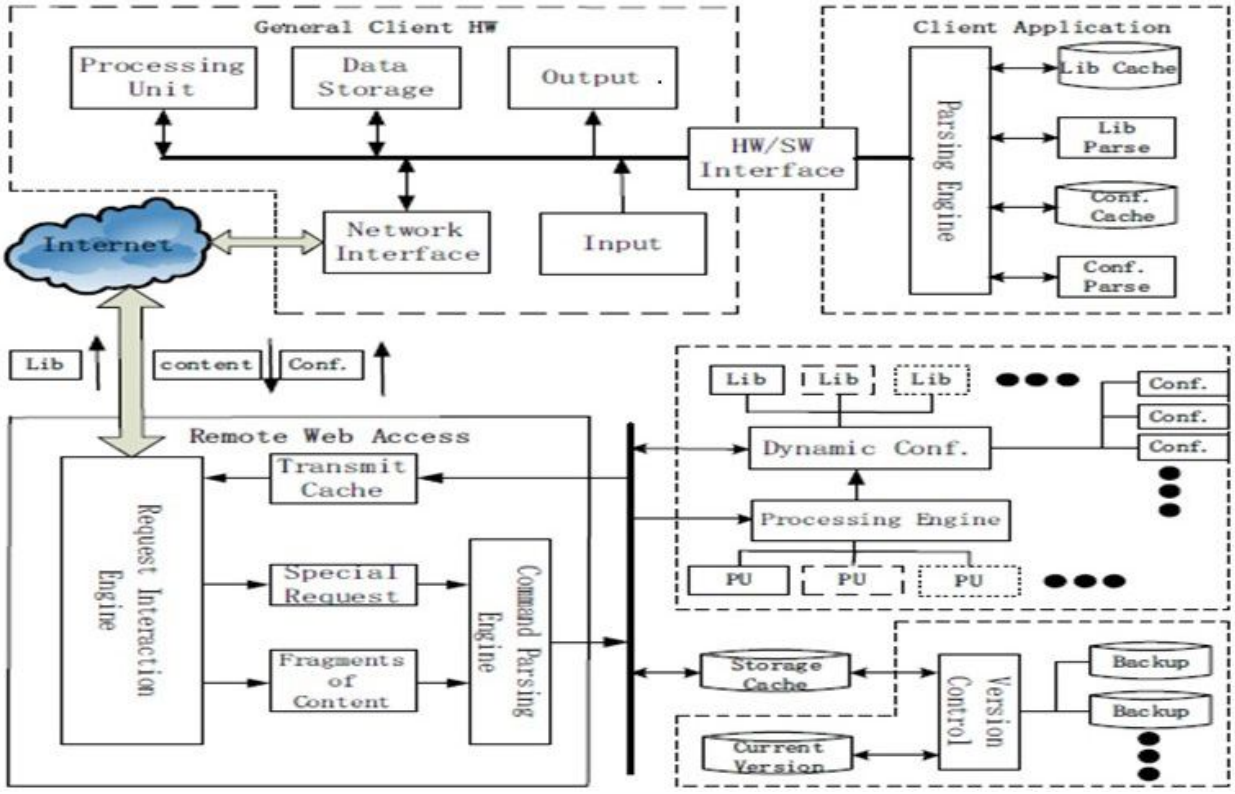

Fig 2: System Architecture Protype

ISSN 2348 – 9928
IJAICT Volume 2, Issue 9, September 2015 Doi:01.0401/ijaict.2014.03.16 Published Online 05 (09) 2015

IV. IMPLEMENTATION

4.1 Creation of an Account

In order to start the services in the system when user first need to create an account based on that account he/she will be processed in the system. The account generation is same as that of creating an account in Gmail or Facebook. User has to enter many details like user id, password, address & a small description about the account holder. When the account is created user will be able to enter the system information. After the process of account creation completed then user is able to enter remote computer's to main system & can use there system for the process of computation. By sending friends request to their profile if they accepted us then user can use there system configurations for the process of computation.

4.2 Cloud Server

To start the processing of the system the first need is to start the cloud server. If the cloud server is started then the operations can be started. The cloud server has to be started in all machines or system that uses this operation. When the cloud server starts function then a window is said to be opened. That window will contain all the details of the remote system with their userid, login time and logout time. Based on that table system can infer that which all remote system are said to active or inactive at a specific time. When the process of computation is said to be completed then the server has to be stopped.

4.3 Login Method

In this method it is made to enter the username & password. After that user enter into the system. Then a new window is opened for the login machine, it will contain the details of all online remote systems in the network. There will be a list of online remote machines & the first machine in the list will be had more suitable to select because it will be online for a long time. The list is said to be prepared by the server by analyzing the login & logout time of different machines, based on this the server machine makes the list of the remote machines. After selecting a machine from the list then a disk manager window is opened and there one can store or perform computation of the data values. There will be a calculation is said to be performed in the login window with Income statement and Bill statement. Income statement consist of the details of the computation and storage performed by different machines for using a specific machine there will be a charge, so that concern concern machine will get income from the different machines that use specific machine. So this will calculate the income for a specific machine. Bill statement is the reverse of the income statement if the specific machine uses other system for doing its computation & storage the specific system had to pay some money. It will calculate how much money a specific system had to pay.

V. PERFORMANCE EVALUATION

5.1 Volumes of Data traffic

Image-transmitting-based sharing software is continuously sending compressed images to the client. As illustrated in Figure.3, for every transmitting interval, servers sent about 96 KB under the resolution of 1024 x 768, while 143 KB under that of 1280 x 800. In this system, however, users may download about 270 KB data for the first time to sign in, but will not suffer continuous data exchanging afterwards. This data includes some data for initialization that is not likely to change frequently, such as frame and basic data. With the data of the basic framework of desktop, when users access the system for the rest of times, they will not have to reload unmodified page resources due to the help of the browser cookie mechanism. If those data changes later, then it just needs to transmit the changed amount of data. From the second time on, sign in to the system will receive no more than 10 KB, just as little as accessing any directory shown in Figure.3. For other particular applications, file editing should download a 350 KB library file at first while image slides show should get a 50 KB preview. Although the system needs to send relatively a bit larger volume of data in the beginning, it does not transmit data continuously. As a result, the system will not occupy much bandwidth.

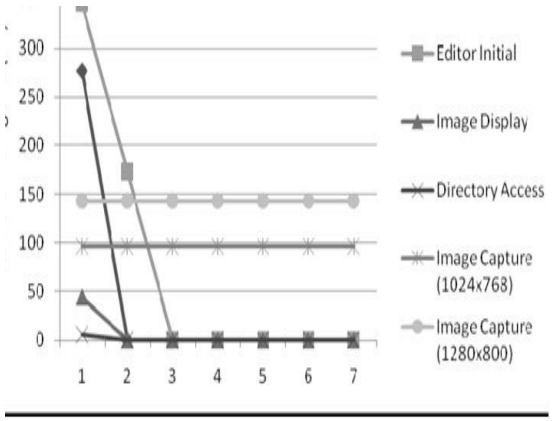

Fig 3: Volumes of transmitted data for different requests (in KBs)

5.2 UI operation latency

Both the latency and the data transferred for each of the four regular operation modes are measured: key stroke, scroll the text,

fill in the form and load bitmap images. These results are shown in Figure.4. The proposed system is compared to two other platforms, the Microsoft Remote Desktop Protocol (RDP) windows and Citrix windows. Compared with other platforms, the system is able to perform those operations with little latency except for loading bitmap images due to the reason that the system is web-based and these operations are performed on the client side. When user needs to interact with server, like downloading images, the system still does better than the other platforms. With relatively small data in transmission, the system does not suffer a long-time latency, which is only one fourth compared to other platforms.

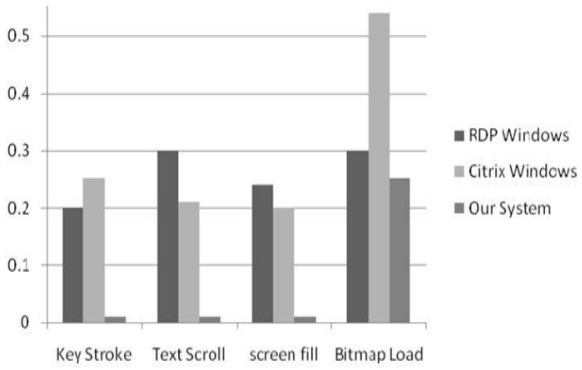

Fig 4: Time latency of UI operations

5.3 System Power Efficiency

The Apache performance is evaluated on a quad-core processor computer: Intel Xeon 5520. Results show that the throughput of Apache is about 600Mbps on transmitting small pages and 800Mbps on transmitting larger pages. For the system, the number varies from 47 to 140Mbps at the average throughput on transmit and receive. On the other hand the power consumption of the FPGA system apache on CPU system is about 20w and 280w respectively. Consider throughput for every Watt as the power efficiency here to show how much performance could be generated with the same energy cost. As illustrated in Figure.3, without OS or peripherals, thus the system does 45.7% better than that of the comparing software system on average. If provided with a better-throughput-hardware to implement TCP/IP stack into hardware on a bigger FPGA, thus the system could gain further improvement in the overall performance and energy efficiency. Also, DaaS can be further implemented in pure hardware as a fixed part of IaaS. Thus with the above factors the performance evaluation can be computed. Thus with the above factors, the project is more efficient than the existing system. Thus the system is more efficient than other systems.

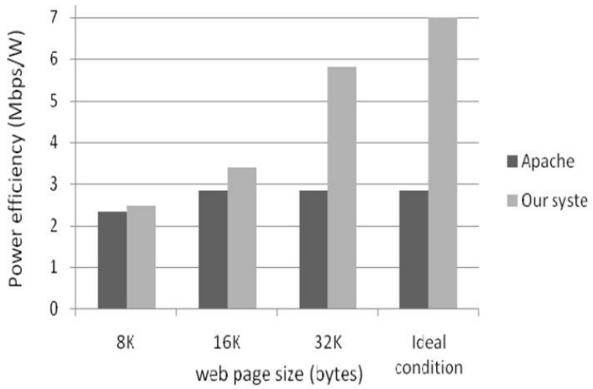

Fig 5: Power efficiency of systems processing dynamic web pages containing matrix multiplications

VI. CONCLUSION

Most of the firms are moving to cloud environment now a days. Moving to cloud is clearly a better alternative as they can add resources based on the traffic according to a pay-per-use model. But for cloud computing to be efficient, the individual servers that make up the datacenter cloud will need to be used optimally. Even an idle server consumes about half its maximum power. In this project the modified the concepts of thin client, now the security of the file transfer between different desktops is said to be more secure with this concept. Data is said to be stored in an encrypted format for more secure transmission. Thus the others can't enter into the system & decrypt the data values. Based on the online time of the remote systems, the priority is said to be assigned. Thus there will be no problem in the case of data sharing like loss of data or data mismatch. Data values are said to be accessed from the remote system only when the system exist for a long time. Valid users are only said to be enter into the system with a valid user id & a password.

The current system is implemented on the basis of storage. Storage based model is now used in software development sector. Thus introduced a new method called computation to the system, thus the system can do computation activities also. Research based on cloud computation and storage can be extended in the way as follows. A list of remote machines can be accessed based on viewing the list provided by the server. To improve the system by implementing a method to select the friends based on name or to track the remote machines based on the name scenario. When a remote machine is enters into the network the server machine treated it as a client and starts the operation. To process the computation and storage it takes the memory if available and start

computation and executes the work given. So implement a method to check whether the details entered by the user are correct or not. Methods like Face recognition password or Graphical password can be used for ensuring the security with actual user identification. The system is said to be implemented by the above concepts then it can be used in high secure data transmission medium.

References

[1] S. Cristofaro, F. Bertini, D. D. Lamanna and R. Baldoni, Virtual Distro Dispatcher: a light-weight Desktop-as-a-Service solution. Proceedings of Cloud Computing Conference pp. 247-260, 2009.

[2] A. Berl, E. Gelenbe, M. D. Girolamo, G. Giuliani, H. D. Meer, M. Q. Dang and K. Pentikousis, Energy-efficient cloud computing, The Computer Journal, 53(7), pp. 1045-1051, 2010.

[3] L. Liu, H. Wang, X. Liu, X. Jin, W. B. He, Q. B. Wang and Y. Chen, GreenCloud: a new architecture for green data center, Proceedings of the 6th international conference industry, ACM, pp. 29-38, 2009.

[4] X. F. Liao, H. Jin, L. T. Hu and X. J. Xiong, LVD: A light weighted virtual desktop management architecture, Proceedings of the 2nd International DMTF Academic Alliance Workshop on Systems and Virtualization Management: Standards and New Technologies (DMTF SVM 2008), Springer- Verlag, pp. 25-36, 2008.

[5] M. Schoeberl, Jop: A java optimized processor for embedded real-time systems, Ph.D. dissertation, Vienna University of Technology, 2005.

[6] M. Stefik, G. Foster, D. G. Bobrow., et al, Beyond the Chalkboard: Computer Supported for Collaboration and Problem Solving in Meetings, Communications of the ACM, 30(1), pp. 32-47, Jan. 1987.

[7] J. R. Lange, P. A. Dinda and S. Rossoff, Experiences with clientbased speculative remote display, Proceedings of USENIX 2008 Annual Technical Conference