Indian Journal of Science and Technology, Vol 8(32), DOI: 10.17485/ijst/2015/v8i32/87506, November 2015
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Enhancing Network I/O Virtualization for Cloud Using Finite Multi Server Queuing Model

L. Lakshmanan[1] and P. Gayathri[2*]

[1]Department of Information Technology, Jerusalem College of Engineering, Chennai - 600100, Tamil Nadu, India; lakshmanan@gmail.com
[2]Department of Information Technology, Bharath University, Chennai – 600073, Tamil Nadu, India; gayathri.cse@bharathuniv.ac.in

Abstract

Virtualization is an important technology that separates computing functions and technology implementation from physical hardware. It reduces costs by increasing energy efficiency and requiring less hardware. The major drawback of network I/O virtualization is its poor networking performance. In this paper the networking performance of virtual machines is calculated on the basis of priority. Also it should allow multiple applications to run simultaneously without affecting the networking performance. For this purpose we put forward a system where we use multi server queuing model where the applications are modeled as queues and the virtual machines are modeled as service providers.

Keywords: Cloud computing, Driver Domain, I/O virtualization, Networking Performance

1. Introduction

Cloud computing is a technology which provides infrastructure, platform and software as service[1]. It is not compulsory that a cloud platform should be virtualized. If the platform is virtualized then it leads to the increase in the resource availability and flexibility of their management. It also reduces cost and requires less hardware. Also the user's job request will be processed in a short period of time. System virtualization refers to the techniques which divide the physical machine into a number of virtual machines that runs concurrently sharing the same physical resources[1]. In the proposed system we are implementing a multi-server queuing model to process the users request effectively and within a very short period of time[7].

2. Proposed System

Our proposed system will be implemented using finite Multi Server Queuing Model to process the users request effectively and within short period of time[6]. First the user's request will be send to the Dispatcher Pool of the Cloud Service Provider[8]. The Dispatcher Pool allocates a queue for the request based on the throughput which is calculated in the virtual machine[9]. The existing system is in a FIFO manner i.e. the first request send by the user will be processed first. The modification done in our system is, we will set a specific priority to process the job. Dispatcher Pool dispatches the request based on the priority, so that job is managed efficiently[5].

The users request will be send to the dispatcher pool of the cloud service provider[11]. The dispatcher pool will

* Author for correspondence

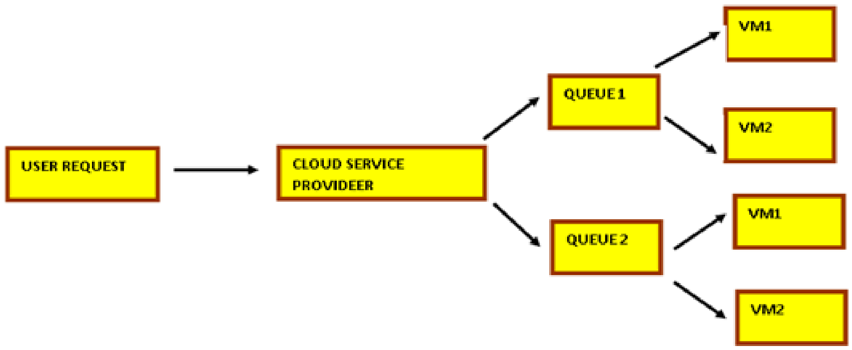

Figure 1. Architecture diagram of users request getting granted using multi-server queuing model[10].

send the request to the queue 1or queue 2 alternatively and throughput is calculated in the Virtual Machines for effective data processing. By calculating the throughput of the virtual machines, the queue can effective allocate the job[12].

3. Methodology used

- Client
- Dispatcher Pool
- Throughput Calculation
- Priority Monitoring

3.1 Client

User is the entry module of this project where by request the data to the server. Server monitors the query used and transfers the data to the user[13]. Cloud server (or) cloud service provider is the main server, where it handles the query and schedules the query according to the user request and forwards to the dispatcher pool[4].

3.2 Dispatcher Pool

Dispatcher pool forwards the data to the queries like queue 1 and queue 2[2]. This will get user's job and divide the work and forwards it to queues. Dispatcher pool schedules the query and transfers to either queue1, queue2 respectively[14].

3.3 Throughput Calculation

Throughput calculation is made on the data received versus data sent. Throughput is calculated in the virtual machine which is deployed after queue storage[15]. The throughput is usually measured in bits per second (bit/s or bps), and sometimes in data packets per second or data packets per time slot[14].The system throughput or aggregate throughput is the sum of the data rates that are delivered to all terminals in a network[16].

3.4 Priority Monitoring

Priority Monitoring is also by setting the data with high/medium/low priority status[17]. High priority request is handled first, then the medium priority finally the low priority request. The video files are given the highest priority followed by the image files and text files. So if the high priority request is handled first then the other low priority requests can be processed within a short period of time rather than handling the low priority requests first[18].

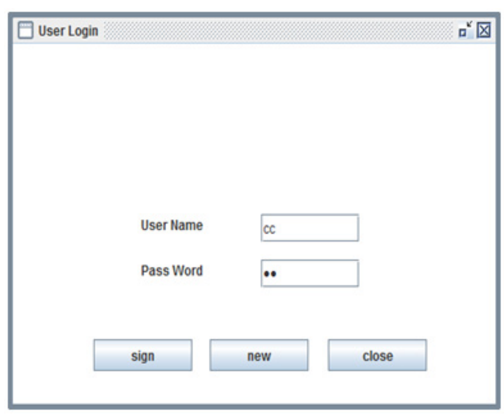

Figure 2. Client side registration.

4. Experimental Result

Thus proposed system helps clients request to be processed in a short time according to a certain priority depending upon the type of file[19]. The system also minimizes the loss of data packets while processing the request and also reduces the waiting time of the request.

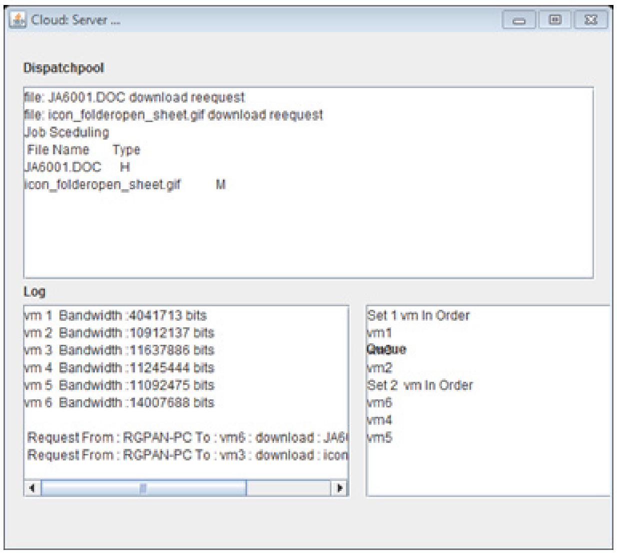

Figure 3. Dispatcher pool assigning jobs based on priority of the request dispatcher pool that gives priority to the file based on the throughput.

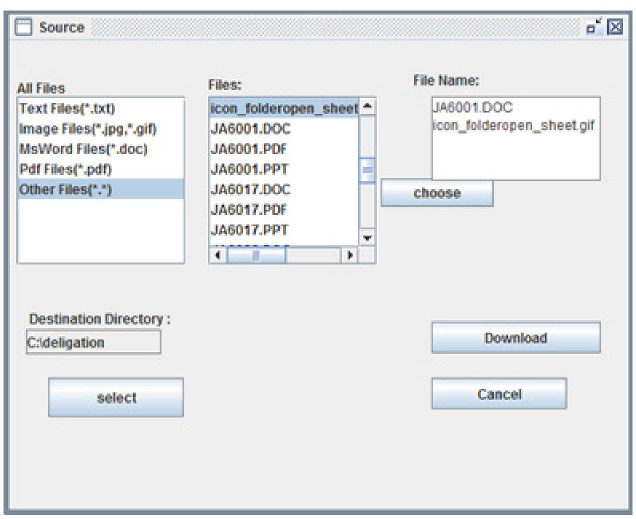

Figure 4. Files that are to be downloaded are selected and the destination folder where the files are to be saved are given.

5. Conclusion

Virtualization is becoming a key technology to enable deploying efficient and cost-effective cloud computing platforms. However, current network I/O virtualization models still suffer from performance and scalability limitations. In this paper, we used showed that VMs exhibit poor reception, transmission and forwarding throughputs. Through profiling the physical resources, we proved that the I/O channels mechanism prevents the guests from achieving line rate throughput. I/O communication between the driver domain and the guests is shown to be the most expensive in memory transactions. Memory transactions are limited by the long memory latency which represents a hardware bottleneck of the system. To overcome this limitation, we proposed a packet aggregation mechanism that allows to transfer containers of packets at once. This results in a more efficient I/O communication. Experimental evaluation shows that the proposed packet aggregation mechanism significantly improves the networking performance.

6. References

1. Bourguiba M, Haddadou K, El Korbi I, Pujolle G. Improving Network I/O Virtualization for Cloud Computing . IEEE Transactions on Parallel and Distributed Systems. 2013.
2. Menon A, Cox AL, Zwaenepoel W. Optimizing network virtualization in Xen. Proceedings of USENIX Annual Technical Conference (USENIX' 06). 2006.
3. Sree Latha R, Vijayaraj R, Azhagiya Singam ER, Chitra K, Subramanian V. 3D-QSAR and Docking Studies on the HEPT Derivatives of HIV-1 Reverse Transcriptase. Chemical Biology and Drug Design. 2011; 78(3):418–26. ISSN: 1747-0285.
4. Santos JR, Janakiraman G, Turner Y. Network optimizations for PV guests. Proc Xen Summit. 2006.
5. Zhang X, McIntosh S, Rohatgi P, Griffin JL. Xen Socket: A High-Throughput Interdomain Transport for Virtual Machines. Proceedings of ACM/IFIP/USENIX Middleware Conference (Middelaware' 07). 2007.
6. Masthan KMK, Aravindha Babu N, Dash KC, Elumalai M. Advanced diagnostic aids in oral cancer. Asian Pacific Journal of Cancer Prevention. 2012; 13(8):3573–6. ISSN: 1513-7368.
7. Wang J, Wright K, Gopalan K. Xen Loop: A Transparent High Performance Inter-vm Network LoopBack. Proc ACM Symp High Performnce Parrallel and Distributed Computing (HPDC'08). 2008.
8. Zhang X, Dong Y. Optimizing Xen VMM based on Intel Virtualization technology. Proceedings of International

Conference on Computer Science and Software Engineering. 2008.
9. Santos JR, Turner Y, Janakiraman G, Pratt I. Bridging the gap between software and hardware techniques for I/O virtualization. Proc. USENIX Annual Technical Conference (USENIX' 08). 2008.
10. Tamilselvi N, Dhamotharan R, Krishnamoorthy P, Shivakumar. Anatomical studies of Indigofera aspalathoides Vahl (Fabaceae). Journal of Chemical and Pharmaceutical Research. 2011; 3(2):738–46. ISSN: 0975–7384.
11. Menon A, Zwaenepoel W. Optimizing TCP Receive Performance. Proc USENIX Annual Technical Conference (USENIX' 08). 2008.
12. Ram KK, Santos JR, Turner Y, Cox AL, Rixner S. Achieving 10Gb/s using safe and transparent Network Interface Virtualization. Proc ACM SIGPLAN/SIGOPS Conf Virtual Execution Environments (VEE' 09). 2009.
13. Dobrescu M, Egi N, Argyraki K, Chun BG, Fall K, Iannaccone G, Knies A, Manesh M, Ratnasamy S. Route bricks: exploiting parallelism to scale software routers. Proceedings of ACM SIGOPS Symp Operating systems principles (SOSP 09). 2009.
14. Devi M, Jeyanthi Rebecca L, Sumathy S. Bactericidal activity of the lactic acid bacteria Lactobacillus delbreukii. Journal of Chemical and Pharmaceutical Research. 2013; 5(2):176–80. ISSN: 0975 – 7384.
15. Ram KK, Santos JR, Turner Y, Cox AL, Rixner S. Achieving 10Gb/s using safe and transparent Network Interface Virtualization. Proceedings of ACM SIGPLAN/SIGOPS Conf Virtual Execution Environments (VEE' 09). 2009.
16. Gamage S, Kangarlou A, Kompella R, Xu D. Opportunistic Flooding to Improve TCP Transmit Performance in Virtualized Clouds. Proceedings of ACM Symp Cloud Computing (SOCC' 11). 2011.
17. Reddy Seshadri V, Suchitra MM, Reddy YM, Reddy Prabhakar E. Beneficial and detrimental actions of free radicals: A review. Journal of Global Pharma Technology. 2010; 2(5):3–11. ISSN: 0975-8542.
18. Bourguiba M, Haddadou K, Pujolle G. Packet Aggregation Based Network I/O Virtualization for Cloud Computing. Elsevier Computer Communications. 2012; 35(3):309–19.
19. Available from: http://en.wikipedia.org/wiki/Throughput.