Advanced Science and Technology Letters
Vol.63 (NGCIT 2014), pp.30-35
http://dx.doi.org/10.14257/astl.2014.63.07

A Network Resources Scheduling Method for Virtual Machine

Xiaodong Liu,Huating Xu

School of Computer Science and Engineering, Henan Institute of Engineering, Zhengzhou, 451191 China
liuxiaodongxht@qq.com,2279883989@qq.com

Abstract. In order to solve the problem of network resources contention on the virtualization environment. This paper presents a queuing based I/O scheduling method. Our proposed method uses a queuing system to control the I/O bandwidth and provide QoS guarantee for each virtual machine(VM). The proposed techniques are implemented on the xen virtual machine monitor (VMM).

Keywords: Resources Scheduling; queuing system; Virtualization.

1 Introduction

Since multiple VMs share the same network interface card (NIC). The I/O operation of one virtual machine is affected by others. The quality of service (QoS) of the network is hardly guaranteed. How to guarantee the QoS of network I/O has been a key issue. In order to guarantee QoS of network I/O, some techniques have been proposed to provide QoS guarantee for VMs. The virtualization technology, such as xen [1], provides I/O network bandwidth limiting strategy to guarantee I/O bandwidth of each VM. Building separated virtual router for VMs [2, 3] or limiting the network I/O rate or time of VMs [4–6] are both used to guarantee the QoS of network. However, the limiting strategy is static allocate. The I/O network bandwidth cannot be adjusted dynamically. The I/O network bandwidth utilization is low. Firstly, some applications, such as big data processing using parallel computing method, are not always use network I/O. When these applications are not communicates with each other, the network bandwidth will be wasted. Secondly, diverse applications inside VMs may have different network requirements. Unpredictable I/O workloads worsen the semantic gap between bandwidth allocation and I/O requirement of VMs [7], leading to inefficient I/O bandwidth allocation and low bandwidth utilization. Thirdly, the I/O bandwidth cannot be reallocated when we create new VMs or destroy old VMs.

This paper presents a queuing based I/O scheduling method. I/O requests of applications running on VMs are considered as guest and network driver is considered as server system. VMs are allocated a certain number of network credits periodically. VMs will consume the network credits when they execute

ISSN: 2287-1233 ASTL

I/O operation and they cannot execute I/O operation when network credits reduce to zero. Queuing rules are established according to network credits and workloads of VMs. the I/O workloads are predicted through network credits information including allocated credits, consumed credits and remained credits.

The proposed techniques are evaluated for various workloads in terms of guaranteeing QoS of the network. We demonstrate that QoS of VMs can be guaranteed when multiple VMs share the same physical network. In addition, we show that the unused bandwidth of the VM can lend to VMs which need more bandwidth.

The remainder of this paper is organized as follows: The problem is presented in section 2. Section 3 presents our queuing based I/O scheduling strategy. Section 4 shows the experimental results. Finally, section 5 summarizes our conclusions.

2 Problem Description

The queuing based I/O scheduling is the problem of providing QoS guarantee for VMs. queuing based I/O scheduling guarantees that each DomU can use its allocated I/O bandwidth. Meanwhile, when the DomUs network is idle, it can lend its network bandwidth to those VMs which need more network bandwidth.

The I/O bandwidth is converted into network credits in our queuing based I/O scheduling. Network credits are periodically allocated to VMs. when VMs perform I/O operation, they will consume network credits. The VM cannot perform I/O operation when the credits of the VM are no more than zero unless it gets network credits again.

Definition 1. *We define the total credits in each scheduling period are $B \times \triangle T$, and we define the allocated network credits of the VM_i in each scheduling period are $B_i^{apl} \times \triangle T$.*

Definition 2. *Let s be the size of data sent by the VM_i, we define the consumed network credits of the VM_i are s.*

The goal of our queuing based I/O scheduling is to providing QoS guarantee of I/O bandwidth for VMs. Meanwhile, the I/O requirements of VMs are different, and the I/O requirements of the VM may vary. The goal of our queuing based I/O scheduling also should be flexible. It should schedule I/O according to VMs requirements.

Table 1 gives the list of symbols we use in this paper along with their meaning.

3 The queuing based I/O scheduling

We consider I/O requests of VMs as customer. Customers can be single or batches arrive. Network I/O is controlled by queuing rules.

Table 1. Description of notation

Notation	Meaning
N	The number of VMs.
Δt	The time intervals.
$B_i(\Delta t)$	The consumed bandwidth credits in the time interval Δt.
ΔT	The scheduling period.
$C_i^{rem}(t_i)$	The remained network credits of the VM_i in the $i-th$ scheduling period.
$C_i^{alc}(t_i)$	The allocated network credits of the VM_i in the $i-th$ scheduling period.
$C_i^{req}(t_i)$	The needed network credits of the VM_i in the $i-th$ scheduling period.
s_i	The I/O resources status of the VM_i.
$C^b(t_i)$	The total network credits that VMs request borrow in the (i+1)th scheduling period.
$C^l(t_i)$	The total network credits that VMs request lend in the (i+1)th scheduling period.
$B_{total}^{idle}(t_i)$	The total idle network credits in the $i-th$ scheduling period.

Definition 3. *If the network credits of the VM not more than zero, the customer from the VM is not allowed to queue. Or else, if the network credits of the VM more than zero, the customer from the VM is allowed to queue.*

Customers are served by the BE. The serving time is related to the size of the data. The larger the data is. The long the serving time is.

Considering the above situations, the QoS of the network I/O is related to the network credits. The QoS of the network I/O can be controlled by network credits.In general, the queuing based I/O scheduling consists of two phases: (1) predicting future I/O resources requirements, in which the running I/O information of VMs is collected. The future I/O resources requirements of VMs can be predicted based on the collected information. (2) I/O scheduling phase, in which the scheduling of I/O resources is done.

3.1 Predicting future I/O resources requirements

We first give the definition of network credits.

Definition 4. *Let d is the bandwidth or the size of data consumed by the VM_i in a scheduling period. We define the bandwidth or the consumed network credits of the VM_i in the scheduling period are d.*

The queuing base I/O scheduling periodically collects the running I/O information, including the allocated network credits, the remained network credits and the consumed credits. Based on these collected information, the I/O resources requirements of VMs in the next scheduling period can be predicted.

(1) if $C_i^{rem}(t_i) \leq 0$, the allocated network credits of the VM_i are consumed completely. Let t_{con} is the time of these allocated network credits are consumed completely. The needed network credits of the VM_i in the next scheduling period can be calculated as follows..

$$C_i^{req}(t_{i+1}) = \frac{C_i^{alc}(t_i) \times \Delta t}{t_{con}} \quad (1)$$

(2) if $C_i^{rem}(t_i) > 0$, the allocated network credits are not consumed completely, which means that there are surplus network credits in the VM_i. These surplus network credits can be lent to VMs which need more network credits. The needed network credits of the VM_i in the next scheduling period can be calculated as follows.

$$C_i^{req}(t_{i+1}) = C_i^{alc}(t_i) - C_i^{rem}(t_i) \tag{2}$$

3.2 I/O Scheduling Phase

This subsection presents our I/O resouces scheduling strategy.

Definition 5. *We define the I/O resources status s_i of the VM_i is 1, when $C_i^{req}(t_{i+1}) > C_i^{alc}(t_i)$. We define the I/O resources status s_i of the VM_i is 0, when $C_i^{req}(t_{i+1}) \doteq C_i^{alc}(t_i)$. We define the scheduling status s_i of the VM_i is -1, when $C_i^{req}(t_{i+1}) < C_i^{alc}(t_i)$.*

After the I/O resouces statuses of VMs are defined, we can then calculate the following two total value.

$$C^b(t_i) = \sum_{i=1,s_i=1}^{N} (C_i^{req}(t_{i+1}) - C_i^{alc}(t_i)) \tag{3}$$

$$C^l(t_i) = \sum_{i=1,s_i=-1}^{N} (C_i^{alc}(t_i) - C_i^{req}(t_{i+1})) \tag{4}$$

Thus, the I/O resources scheduling can be divided into three conditions:

if $C^b(t_i) \leq C^l(t_i)$,the allocated network credits of VMs which s_i is 1 can be calculated as follows.

$$C_i^{alc}(t_{i+1}) = C_i^{req}(t_{i+1}) \tag{5}$$

the allocated network credits of VMs which s_i is 0 can be calculated as follows.

$$C_i^{alc}(t_{i+1}) = C_i^{alc}(t_i \tag{6}$$

the allocated network credits of VMs which s_i is -1 can be calculated as follows.

$$C_i^{alc}(t_{i+1}) = C_i^{req}(t_{i+1}) + \frac{(C^l(t_{i+1}) - C^b(t_{i+1}))}{\sum_{i=1,s_i=-1}^{N} B_i^{apl}(t_i)} \times B_i^{apl}(t_i) \tag{7}$$

if $C^b(t_i) \doteq C^l(t_i)$,the allocated network credits of all VMs in the next scheduling period can be calculated using Eq.(7).

if $C^b(t_i) > C^l(t_i)$,the allocated network credits of VMs which s_i is 1 can be calculated as follows.

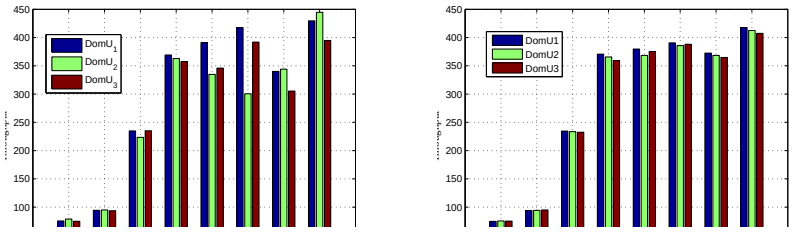
$$C_i^{alc}(t_{i+1}) = C_i^{req}(t_{i+1}) + \frac{(C_i^{req}(t_{i+1} - C_i^{alc}(t_i)}{C^b(t_i)} \times B_{total}^{idle}(t_i) \tag{8}$$

the allocated network credits of VMs which s_i is -1 or 0 can be calculated by equation (6).

4 Performance Evaluation

We have implemented the queuing based scheduling algorithm in the xen 4.1.2 hypervisor. In this section, we compare the performance of the queuing based scheduling algorithm (QBS_xen) with original xen scheduling (Orgi_xen).Two computers were used to evaluate the QBS scheduler. One was a dual-socket and quad-core Inter Xeon CPU running at 2.40GHZ, 32 GB of RAM (PM1). The operation system is running the 64-bit version of Ubuntu 12.04. This one was acted as host machine and was running xen 4.1.2. The system is configured with one to multiple DomUs and one Dom0. The DomUs and Dom0 are both configured with 1G memory and a single virtual CPU. The operating system of Dom0 and DomUs are the same as the host machine. Another is an Intel dual-core CPU running at 2.93GHZ, 2G of RAM (PM2). The operating system is ubuntu 12.04. This one is used as server during the experiment described below.The httperf[8] is used in our experiments. In the httperf benchmarks, all DomUs send http requests to PM2.

We concurrently run three DomUs with the same bandwidth. Fig.1 shows the actual throughput of three DomUs on the machine PM1. Fig.1 (a) shows the throughput of three DomUs in the orginal xen scheduling. When three DomUs sending I/O request in the same time, they will influence each other. The throughput of three DomUs is different, and the difference of DomUs network can even reach 50%. The QoS of the DomU can hardly be guaranteed. Fig.1 (b) shows the throughput of three DomUs in the queuing based I/O scheduling. The throughput of three DomUs is almost the same, which is caused by limiting network credits for each DomU.



(a) (b)

Fig. 1. the performance comparison between Orgi_Xen scheduling and QBS_Xen.

4.1 Conclusions

In this paper, we have proposed the queuing based I/O scheduling for guaranteeing the QoS of network I/O. Network bandwidth is converted into network

credits. The network I/O scheduling is converted into increased or decreased network credits. The queuing based I/O scheduling is a periodical scheduling algorithm. With each periodical round, the I/O resources requirements are diagnosed based on the collected I/O information. Then, VMs are divided into three statuses according to I/O resources requirements and their run information. The queuing based I/O scheduling periodically schedule I/O resources according to VMs status. Our proposed algorithms have been implemented on the xen hypervisor 4.1.2.

Acknowledgments. This work is supported by Doctor Foundation of Henan Institute of Engineering.

References

1. P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, Xen and the art of virtualization, ACM SIGOPS Operating Systems Review, 37:164-177(2003).
2. K. Fraser, S. Hand, R. Neugebauer, I. Pratt, A. Warfield, and M. Williamson, Safe hardware access with the Xen virtual machine monitor, in 1st Workshop on Operating System and Architectural Support for the on demand IT InfraStructure (OASIS), (2004).
3. N. Egi, A. Greenhalgh, M. Handley, M. Hoerdt, F. Huici, and L. Mathy, Fairness issues in software virtual routers, in Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow,pp.33-38(2008).
4. M. B. Anwer, A. Nayak, N. Feamster, and L. Liu, Network I/O fairness in virtual machines, in Proceedings of the second ACM SIGCOMM workshop on Virtualized infrastructure systems and architectures, pp.73-80(2010).
5. A. Gulati, A. Merchant, and P. J. Varman, mClock: handling throughput variability for hypervisor IO scheduling, in Proceedings of the 9th USENIX conference on Operating systems design and implementation,pp.1-7(2010).
6. S. R. Seelam and P. J. Teller, Virtual I/O scheduler: a scheduler of schedulers for performance virtualization, in Proceedings of the 3rd international conference on Virtual execution environments, pp.105-115(2007).
7. H. Kim, H. Lim, J. Jeong, H. Jo, J. Lee, and S. Maeng, Transparently bridging semantic gap in cpu management for virtualized environments,Journal of Parallel and Distributed Computing, 71:758-773(2011).
8. D. Mosberger and T. Jin, httperfa tool for measuring web server performance, ACM SIGMETRICS Performance Evaluation Review,26:31-37(1998).

