Feedback Scheduling for Realtime Task on Xen Virtual Machine

Byung Ki Kim*, Kyung Woo Hur, Jae Hyuck Jang, and Young Woong Ko

Dept. of Computer Engineering, Hallym University,
Chuncheon, Korea
{bkkim,firehur,jaehyuck,yuko}@hallym.ac.kr

Abstract. In virtual machine environments, it is difficult to allocate CPU resource to virtual machine efficiently because virtual machine lacks knowledge of each domains workload. Especially, realtime tasks in guest domain have to finish before their deadline, however, virtual machine scheduler is not aware of guest-level tasks and how much resources guest domain requires. In this paper, we present a virtual machine scheduling framework based on feedback mechanism. The proposed mechanism exploits various scheduling information from each domain. Xen scheduler controls the CPU allocation by increasing or decreasing CPU slices. We evaluate our prototype in terms of realtime task performance over diverse workload. Our experiment result shows that feedback mechanism effectively allocates CPU resources for guest domain in varying workloads.

Keywords: Xen, realtime, scheduler, QoS, resource monitor.

1 Introduction

To support realtime tasks on a virtual machine, virtual machine monitor (VMM) should predict guest domains CPU requirement exactly. For example, guest domain for multimedia streaming service must be executed in a time-sensitive manner, or it will fail to support multimedia task. Therefore, to satisfy quality of service (QoS) of multimedia, guest domain must receive appropriate timeliness guarantees from virtual machine monitor. However, multimedia streaming server has diverse workloads and it is difficult to predict exact amount of requirement of workloads. Since this lack of knowledge about future workloads makes VMM difficult to allocate CPU resources efficiently. VMM could not track which domain is busy and need more CPU because VMs are consolidated. It will degrade realtime guest domains performance and responsiveness. This paper present a realtime-aware virtual machine scheduling mechanism based on feedback mechanism. Our goal is to improve the QoS of realtime task under the condition of

* This research was financially supported by the Ministry of Education, Science Technology (MEST) and National Research Foundation of Korea(NRF) through the Human Resource Training Project for Regional Innovation and Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2011-0006040).

varying workloads. We designed and implemented VMM monitoring tools and measured the workload of each domain. By predicting CPU usage of each domain, we can dynamically increase and decrease CPU resource of each domain. The reminder of this paper is organized as follows: Section 2 describes the design and implementation of the proposed scheduling mechanism. Section 3 demonstrates and discuses experimental results and Section 4 explains related works. Finally, in Section 5, we concludes our works and discuss future works.

2 Feedback Scheduling Framework

To allocate CPU efficiently, VMM should provide admission control mechanism in real time. However, it is difficult because VMM lacks knowledge of each domains workloads as we mentioned earlier. Figure 1 shows the overall architecture of the proposed system. The proposed system is composed of three parts; first, QoS monitor module in guest operating system checks whether realtime tasks miss their deadline. If there is deadline miss in a certain level, QoS monitor

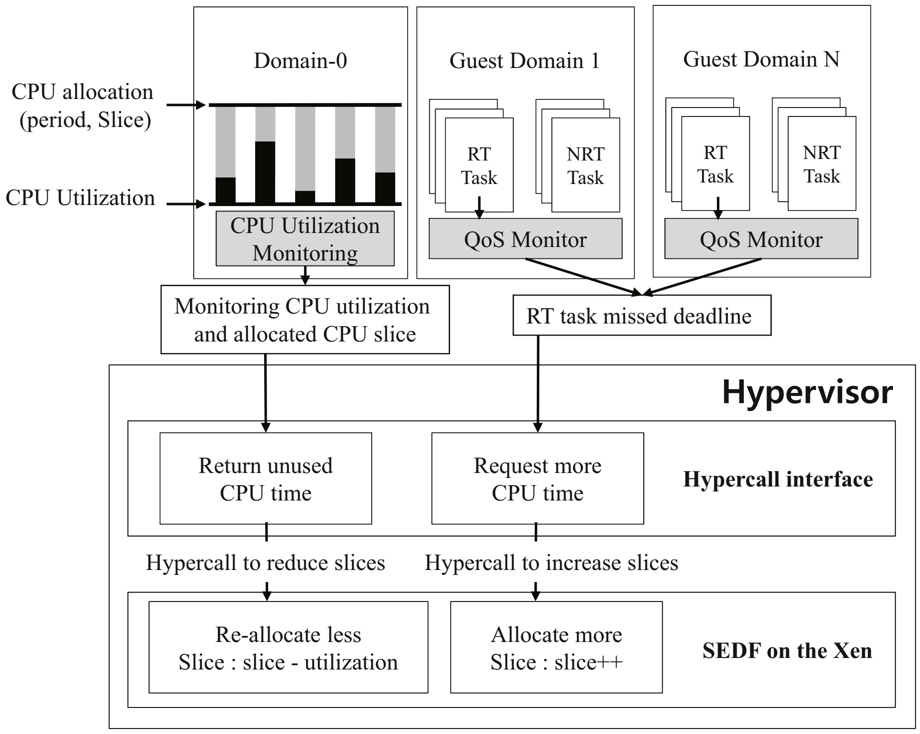
Domain-0
Guest Domain 1
Guest Domain N
CPU allocation (period, Slice)
CPU Utilization
CPU Utilization Monitoring
RT Task
NRT Task
QoS Monitor
Monitoring CPU utilization and allocated CPU slice
RT task missed deadline
Hypervisor
Return unused CPU time
Request more CPU time
Hypercall interface
Hypercall to reduce slices
Hypercall to increase slices
Re-allocate less Slice : slice - utilization
Allocate more Slice : slice++
SEDF on the Xen

Fig. 1. The general architecture of the proposed system

requests more CPU slice to VMM scheduler. In this work, we made additional hypercall interface to adjust slices. Second, Scheduling tool in domain0 manages system resource and controls scheduling policy. User can change scheduling information using scheduling tool by changing period and slice of each domain. This tool periodically checks available CPU resource and reallocates CPU resource dynamically. Third, in hypervisor, we implemented feedback scheduling module which receives hypercall from guest operating system and adjusts slice of each domain. When VMM receives the hypercall from guest operating system then SEDF will increase its slice for 1 millisecond. SEDF stop increasing slice when there is no more slice request. In this work, if there is no enough CPU resource to allocate for a VM requesting slice, we have to prohibit excessive resource allocation.

2.1 Requesting CPU Resource

Credit scheduler is a default scheduler in Xen hypervisor and it automatically allocates the same weight to the domains. Default weight value is specified as 256. In credit scheduler, user can adjust weight value differently, however its difficult to allocate fixed amount of CPU resource. SEDF is a second scheduler supported by Xen. We can use SEDF algorithm if we specify SEDF as a default scheduler during boot time. SEDF can execute domain as a realtime manner, however it has drawback for supporting global scheduling which means SEDF cannot distribute workloads between processors. In our work, we used SEDF as a main scheduler, so, each guest domain was initially allocated period and slice. We implemented QoS monitoring tool with library routine. Therefore, all the realtime tasks running on a guest domain calls this library routine. Library routine requests more CPU allocation to SEDF scheduler by calling hypercall routine. Therefore, whenever a realtime task misses deadline, library routine hypercall to VMM.

2.2 Returning CPU Resource

To prevent excessive CPU request by a domain, we have to isolate each domain in a certain level. In this work, scheduling tools running on a domain0 periodically evaluates CPU usage of each domain. For every second, VMM tracks domains CPU utilization. When CPU utilization of current a domain is lower than assigned amount of CPU time, VMM will reduce its CPU slices by one slice for every second. Its heuristic approach, but our prototype focuses on allocation CPU slice real time in diverse workloads. For example, if a domain is assigned 20% of CPU resource (period: 10 ms, slice: 2 ms) then the domain have to consume 20% of CPU. When CPU utilization of a domain has decreased to 15%, the slice must be decreased to 5%. In VMM scheduling modules, there is a runnable queue where each VCPU is sorted by ascending deadlines. SEDF schedules domains from head of PCPU on run-queue. Once a VCPU has consumed its slice for current period, this VCPU removed from run-queue and moved to wait-queue to receive a fresh slice and period. The wait queue is sorted in ascending order of the start time of next period. Only User (Administrator) can configure these scheduling parameters. Before the administrator adjusts these parameters, VM scheduled as best-effort mode. SEDF has two more queues, utilization and penalty queue called extra queue. VMs scheduled under best-effort mode, every VCPU is on extra queue and scheduled every 500 micro seconds by round robin manner. When VCPUs increase on a PCPU its scheduling delay increases proportionally.[8] proposed to improve the worst case scheduling response time. They remove its short unblocking situation. But there is still scheduling delay exist. To reduce the effect of context-switch overhead, when the domain is blocked before the end of the current period, SEDF removes current VCPU to extra queue.

3 Experiment and Evaluation

In this experiment, we focus on deadline miss ratio for measuring realtime task. We made a various experiments to draw all the aspects of the proposed system capabilities. Our experimental platform consists of the following components. As shown in table 1, our hardware platform has dual core processor. The software platform is based on CentOS Linux kernel that is widely used in Xen virtualization. We installed 2 domains on Xen hypervisor.

Table 1. Experiment Environment

H/W	CPU	Intel Core i7 920 2.66Ghz, Dual
	Cache	L2: 256KB * 4, L3 : 8MB
	Memory	DDR3 PC-10600 2G * 3
	Network	Gigabit Ethernet
	Disk	Seagate 1TB 7200 RPM
S/W	VMM	Xen 3.4.3
	Dom0	CentOS 5 2.6.18.8-xen0
	DomU	CentOS 5 2.6.18-164.el5xen

In this experiment, we made two realtime domains, two non-realtime domains and domain0. Figure 2 shows VCPUs on each PCPU. Domain0 is pined on PCPU0 and other VCPUs are pined on PCPU1. Only VMs can share 100 percent of PCPU1 resources. We set these VCPUs on a PCPU because VMs are not affected by Domain0. To evaluate our proposed mechanism, we implemented time-driven periodic realtime task that periodically produce hash work. These types of workloads are periodically executed on real-time domain. We made a periodic task using MD5 hash function, which executes E time units during P period. We can control execution time E by varying hash block size. The period of realtime task is set 33, so every second the task can produce 30 hash data. If we try to change workloads then we simply increase/decrease hash data size. To eliminate additional overhead incurred by I/O system, we generate data which consumed by MD5 hash function.

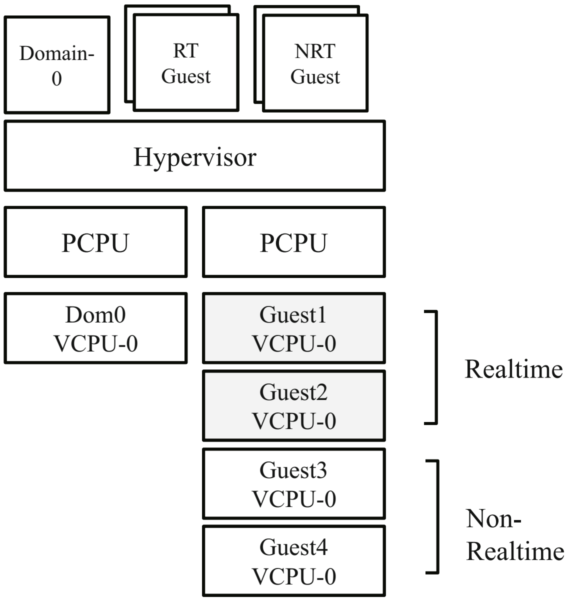

Fig. 2. VCPU configuration for experiment

3.1 Workload Setup

Figure 3 shows how realtime domains work when the workloads of each realtime domain changed. X-axis is processing time and Y-axis is hash data size. To make varying workloads, we changed hash data every 60 seconds. The total hash size is between 0.8 MByte to 1 MByte. In this experiment, two realtime tasks are executed concurrently on each realtime domains. At first, we setup SEDF parameter based on start time workloads. As time goes, the workloads are fluctuating whenever the hash data size is changed. If a domain increases hash data size, it will need more CPU resource. Accordingly, the system starts to increase slice, then the domain may prevent deadline miss for a realtime task. After period of time, the domain may have enough CPU resource and not used for realtime task. At this point, available CPU resource should be returned to help other realtime domain.

3.2 Deadline Miss Result

Figure 4 presents how many times each domain missed deadline. X-axis means hash data size and black line means deadline miss. As we can see in this figure, when hash data size is increased, there are lots of deadline misses. However, if hash data size is decreased then there is no deadline miss. Figure 5 shows the proposed system result. As mentioned earlier, if realtime task misses its deadline, QoS monitor requests more CPU slice with hypercall interface. As shown in figure 5, there exists instant deadline misses when hash data size is increased,

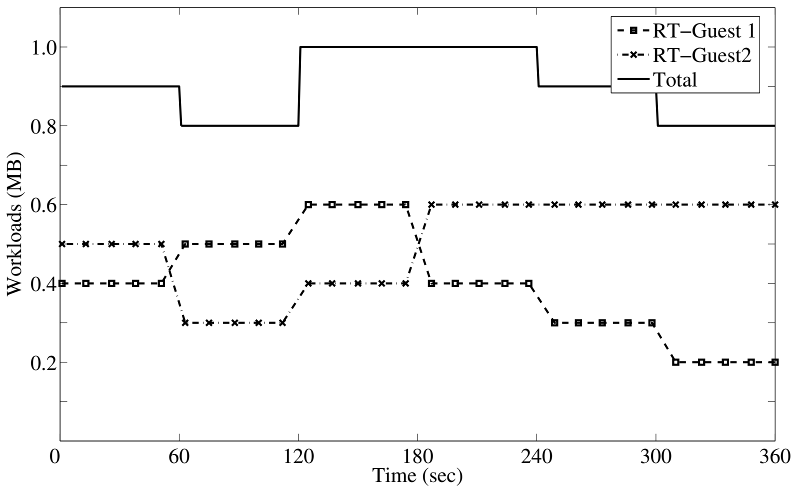

Fig. 3. Workloads for each domain and total workloads

however with slice increasing for the domain, deadline miss is eliminated. Figure 6 shows how much slices were allocated to VMs at each workload. All of guest domain receives 100 ms period as default. Default SEDF schedules each VM by round-robin fashion. Every VM has 500 microseconds. If 4 domains were booted and work with same workload, their utilizations are 25% for each. It means Xen hypervisor guarantees the fairness, but we are focused on performance enhancement of realtime domain. Although our mechanism does not guarantee the fairness, we guarantee performance of realtime domain which has higher priority.

4 Related Work

Performance analysis and Resource allocation have been well conducted on the Xen VMM[1]. Many researches are roughly focused on improving I/O performance, network response, CPU allocation, resource monitoring and real time guarantee. Three scheduling mechanisms are well analyzed in terms of I/O performance and CPU utilization according to different parameters in various workloads [2][3]. Analyzing I/O bound tasks is complicate because isolated driver domain (IDD) processes I/O processing on behalf of VMs[9]. To improve I/O processing like disk I/O and communication via NICs, SEDF-DC introduces a performance monitoring and profiling tool that reports VPU usage of VMs and VM scheduler with feedback [4]. Govindan et al. introduced communication-aware VM scheduling mechanism for high throughput of network intensive workloads.

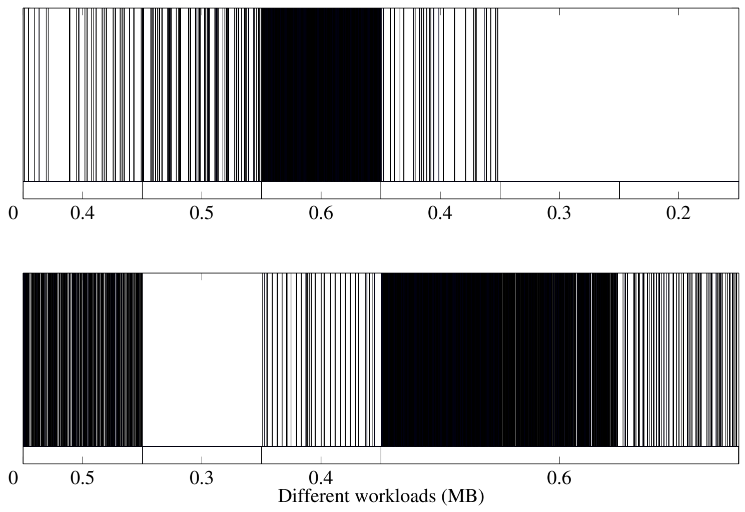

Fig. 4. Deadline miss rate in different workloads with default SEDF

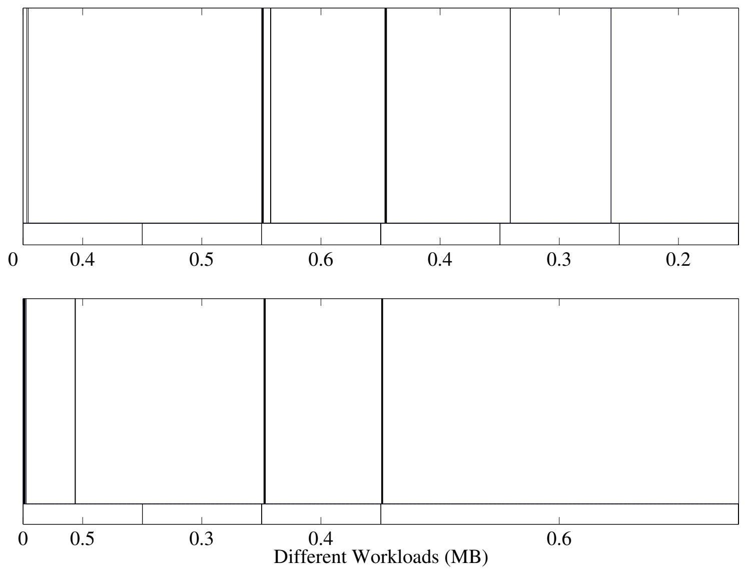

Fig. 5. Deadline miss rate in different workloads on proposed system

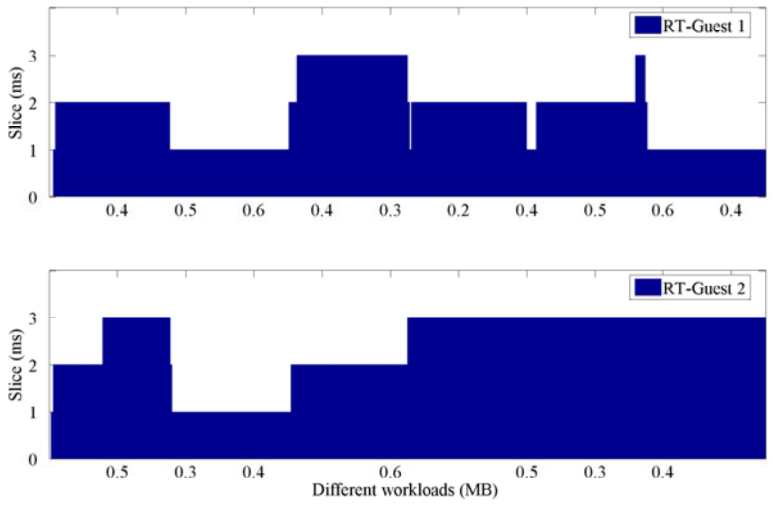

Fig. 6. Slice allocation result on different workloads

Their scheduling mechanism is picking a domain that is likely to experience the most overall reduction in scheduling delay. The domain that has received the most number of packets has the highest priority [5]. Lee et al. suggests soft real-time scheduler for Xen hypervisor by modifying credit scheduler to calculate scheduling priority dynamically. They defined laxity value that provides an estimate of when the task needs to be scheduled next. When a VCPU of a real-time domain is ready, it is inserted where its deadline is expected to be met. This approach deals with low-latency tasks to be executed timely manner. However, it has difficult to guarantee real-time workloads because there is no admission control mechanism. Therefore, if there increase workloads, it cannot meet realtime tasks deadline[6]. RT-Xen introduces a hierarchical real-time scheduling framework in Xen. The key idea is twofold, first, RT-Xen provides a sporadic server root scheduler for Xen that is compatible with eh RM scheduling. Second, they use 1 ms scheduling resolution while incurring moderate overhead. However, RT-Xen cannot support an admission control mechanism for real-time domain, therefore in an excessive workloads, its difficult to guarantee real-time task [7].

5 Conclusion

In this paper, we show a practical scheduling mechanism for realtime guest domain based on task feedback. When a domain needs more CPU then VMM catch this event immediately and adjust CPU resource to guarantee QoS of each domain. In case of varying workloads, VMM has difficulty to predict how much CPU allocation to each domain. On default SEDF scheduler every VM works as best-effort mode, therefore it is difficult to satisfy varying workloads of realtime domain. In this work, we propose a mechanism that VMM scheduler let notify scheduling event to VMM. Experiment result shows that our prototype system excessively minimizes deadline miss ratio compared to default SEDF. For future work, we consider self-adaptable scheduler that supports varying I/O workloads.

References

1. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the art of virtualization. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, pp. 164–177. ACM, USA (2003)
2. Cherkasova, L., Gupta, D., Vahdat, A.: Comparison of the three CPU schedulers in Xen. SIGMETRICS Perform. Eval. Rev. 35, 42–51 (2007)
3. Cherkasova, L., Gupta, D., Vahdat, A.: When virtual is harder than real: Resource allocation challenges in virtual machine based it environments. Tech. Rep. HPL-2007-25 (2007)
4. Gupta, D., Cherkasova, L., Gardner, R., Vahdat, A.: Enforcing performance isolation across virtual machines in Xen. In: Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware, pp. 342–362. Springer, New York (2006)
5. Govindan, S., Nath, A.R., Das, A., Urgaonkar, B., Sivasubramaniam, A.: Xen and co.: communication-aware CPU scheduling for consolidated xen-based hosting platforms. In: Proceedings of the 3rd International Conference on Virtual Execution Environments, pp. 126–136. ACM, USA (2007)
6. Lee, M., Krishnakumar, A.S., Krishnan, P., Singh, N., Yajnik, S.: Supporting soft real-time tasks in the xen hypervisor. In: Proceedings of the 6th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, pp. 97–108. ACM, Pennsylvania (2010)
7. Xi, S., Wilson, J., Lu, C., Gill, C.: RT-Xen: Real-time virtualization based on hierarchical scheduling. Washington University Technical Report WUCSE-2010-38 (2010)
8. Masrur, A., Drossler, S., Pfeuffer, T., Chakraborty, S.: VM-Based Real-Time Services for Automotive Control Applications. In: Proceedings of the 2010 IEEE 16th International Conference on Embedded and Real-Time Computing Systems and Applications, pp. 218–223. IEEE Computer Society (2010)
9. Kim, H., Lim, H., Jeong, J., Jo, H., Lee, J.: Task-aware virtual machine scheduling for I/O performance. In: Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, pp. 101–110. ACM, Washington, DC, USA (2009)