Get What You Pay For: Providing Performance Isolation in Virtualized Execution Environments

Hannes Payer, Harald Röck, Christoph Kirsch
Department of Computer Sciences, University of Salzburg, Austria
{hpayer, hroeck, ck}@cs.uni-salzburg.at

Objective

Virtualization allows multiple systems encapsulated in so-called domains to share completely isolated from each other a single physical machine. Several companies are already taking advantage of virtualization technology in order to sell a certain amount of CPU speed and I/O capacity in terms of latency and throughput on demand to their customers.
Independent of the load generated by the domains running on the system each domain has to get what its customer is paying for, not more and not less. We provide performance isolation for domain CPU speed, domain I/O throughput and latency, and time-critical applications running within a domain.

Virtual CPUs

In the Xen hypervisor the scheduling unit is called a virtual CPU (vCPU), which is an abstraction of a physical CPU. The current scheduler in Xen, called credit scheduler, is tuned for high throughput and good fairness among all active vCPUs in the system. Unfortunately, it does not provide low latency or guaranteed CPU shares [**?**].

EDF-vCPUs

For guaranteed CPU shares and low latency execution of vCPUs we propose the concept of an EDF-vCPU, which have a period (π) and slice (λ) as scheduling parameter. A scheduler that supports EDF-vCPUs must guarantee to run them for their respective λ ms every π ms.

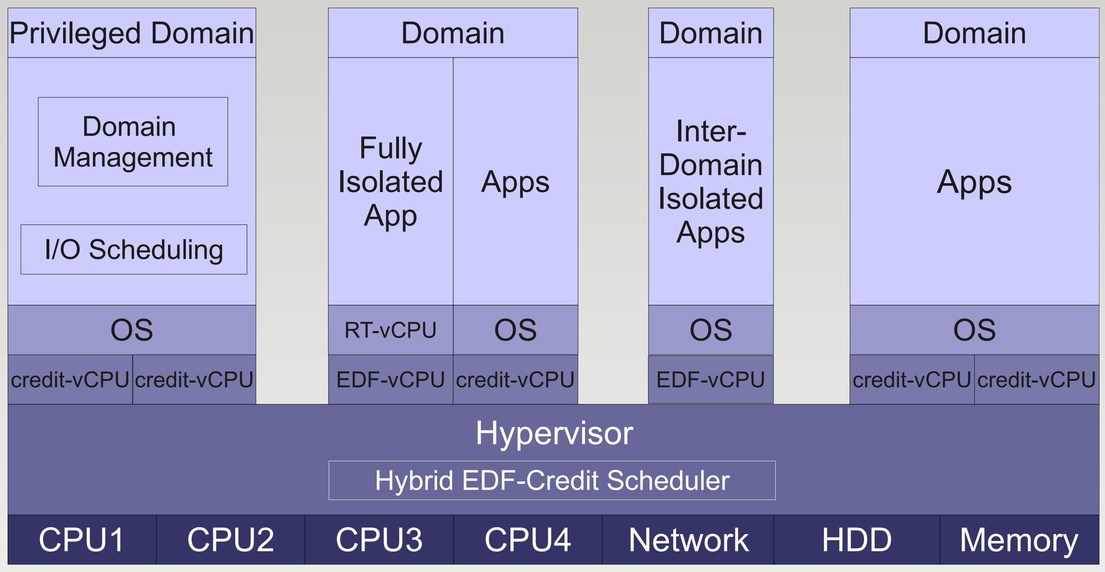
CPU isolation inside a domain

For a virtualized OS running in a domain, a vCPU appears as a regular physical CPU core on which applications are scheduled by the OS scheduler. Therefore, adapting the hypervisor scheduler is not sufficient to provide the requested performance for an end-user application since the OS kernel introduces additional scheduling jitter. Hence, we plan to apply CPU isolation extensions (CPUISOL) to the Linux kernel in order to reduce the jitter and scheduling latencies introduced by the Linux kernel scheduler. CPUISOL isolates a CPU from the Linux scheduler, interrupts, and work-queues, while still be possible to run regular user-applications on the isolated CPU. If the isolated CPU is an EDF-vCPU, we call it an RT-vCPU since the application running on it is temporally isolated from all other applications running inside the domain (intra-domain isolation) as well as from all other domains (inter-domain isolation). Such full temporal isolation weakens to just inter-domain temporal isolation for domains that only use EDF-vCPUs but not the CPU isolation feature.

Funding

This work is supported by the EU ArtistDesign Network of Excellence on Embedded Systems Design, and the Austrian Science Fund No. P18913-N15.

Architecture



Providing performance isolation in virtualized execution environments requires extensions across the entire system, including the hypervisor, guest kernels and driver domains. Our study is based on the open-source virtual machine monitor Xen [**?**]. In the system architecture of Xen, the hypervisor is the lowest layer. On top of the hypervisor run several domains, which encapsulate complete operating system instances. The main tasks of the hypervisor are domain scheduling, I/O handling, and managing memory. Some of the domains running on the hypervisor are special domains which belong to the trusted code base, e.g. a privileged domain for domain management or a driver domain where device drivers are executed.

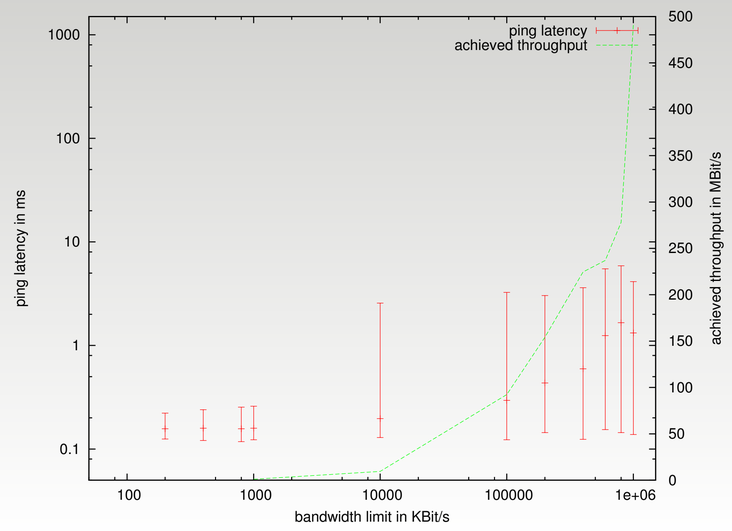
Hybrid EDF-credit scheduler

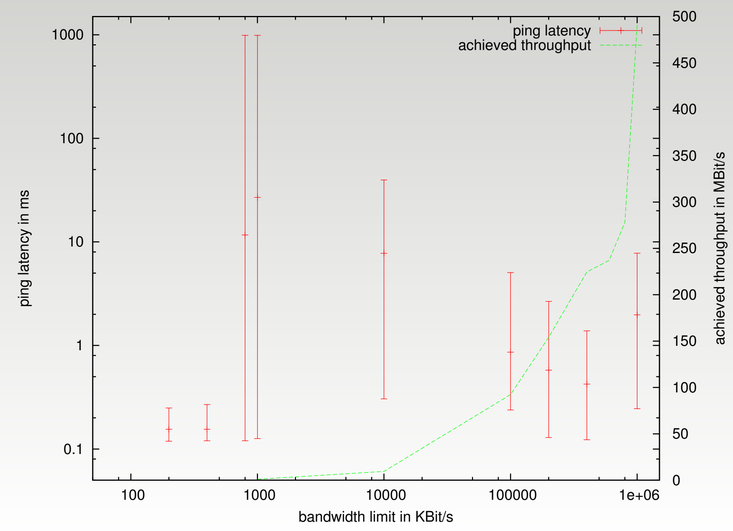
We propose a hybrid EDF-credit scheduler [**?**] that schedules a vCPU either with the standard credit scheduler (credit-vCPU) or alternatively with a CPU-local EDF-based scheduler (EDF-vCPU). The hybrid EDF-credit scheduler provides low latency and allows to specify a guaranteed CPU share for EDF-vCPUs while still achieving high throughput for credit-vCPUs.
We have implemented the hybrid EDF-credit scheduler in the Xen hypervisor. The implementation of the scheduler uses a run-queue for each physical CPU and applies work stealing and dynamic migration of vCPUs in order to balance the workload across all available physical CPU cores. Additionally, we extended the Xen tools to modify the scheduling parameters of a vCPU and to switch the vCPU from a credit-vCPU to an EDF-vCPU and back. Moreover, we also implemented a true global EDF scheduler with a single run queue to evaluate the trade-off between global EDF scheduling accuracy and scheduling overhead.

I/O scheduling

The driver domain is similar to a network router; domains performing I/O send their requests through an event channel to the driver domain, which performs the request.
We experimented with a hierarchical token bucket (HTB) traffic shaping approach in the driver domain, which regulates the packet flow from guest domains to network driver and evaluated its performance. Depending on the HTB parameters and the network backend driver configuration (delayed copy, always copy, and never copy mode) one can trade-off network traffic throughput and driver domain CPU utilization versus network latency jitter.
Moreover we aim at developing a domain bandwith aware IRQ balancer which could distribute I/O interrupts intelligently over the available CPUs. This would optimize overall system performance and guarantee performance.

I/O Benchmark Results





References

[1] Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., and Warfield, A. Xen and the art of virtualization. In *Proc. SOSP* (2003).

[2] Craciunas, S., Kirsch, C., Payer, H., Röck, H., and Sokolova, A. Programmable temporal isolation through variable-bandwidth servers. In *Proc. SIES* (2009).

[3] Ongaro, D., Cox, A., and Rixner, S. Scheduling I/O in virtual machine monitors. In *Proc. VEE* (2008).