Improving disk I/O by paravirtualizing block requests

Matthias Brugger  
(matthias.bgg@gmail.com)

Ramon Nou  
(ramon.nou@bsc.es)

Toni Cortes  
(toni@ac.upc.edu)
Introduction: Problems on disk I/O

- Mechanics have a non negligible overhead
- The way requests are dispatched to the device are important for throughput, latency and fairness between processes
- State of the art I/O scheduler wait some milliseconds for another request of the same process
I/O stack in virtualized environments
Passing information from guest to host

Problem:

- I/O scheduler groups requests on per process basis
- Virtual Machine Monitor runs as user process in the host
- Information about different processes inside the guest are not available for the host I/O scheduler
- Qemu has one global queue per VM

Idea:

- Pass “virtual PID” to the host I/O scheduler
- Host I/O scheduler can work effectively on this abstraction
- Every thread has its own queue
Mapping strategies for block requests

Guest

Host

Native

Threaded

I/O threads

physical disk

queues

mapping
Results - throughput

![Graph showing throughput results for different types of operations (Native-1T, Threaded-1T, Native-2T, Threaded-2T, Native-4T, Threaded-4T) with various throughput values in KB/s.](image-url)
Results – fairness two Qemu instances
Feel free to ask questions...
(matthias.bgg@gmail.com)