Adding Advanced Storage Controller Functionality via Low-Overhead Virtualization

Muli Ben-Yehuda, Michael Factor, Eran Rom, and Avishay Traeger
Eran Borovik and Ben-Ami Yassour
IBM Research – Haifa
Contact: Avishay Traeger avishay@il.ibm.com

Executive Summary

To meet new requirements, new functions are often added to existing storage systems

Examples: file serving, database, in-line deduplication, in-line compression

Traditional approaches

- Code-level integration
- Run new functions on external gateways

Our approach: Run new functions on virtual machines

- All the pros of the traditional approaches with none of the cons
- Main concern is impact on I/O performance

We demonstrate a set of mechanisms and techniques that achieve near-zero performance overhead

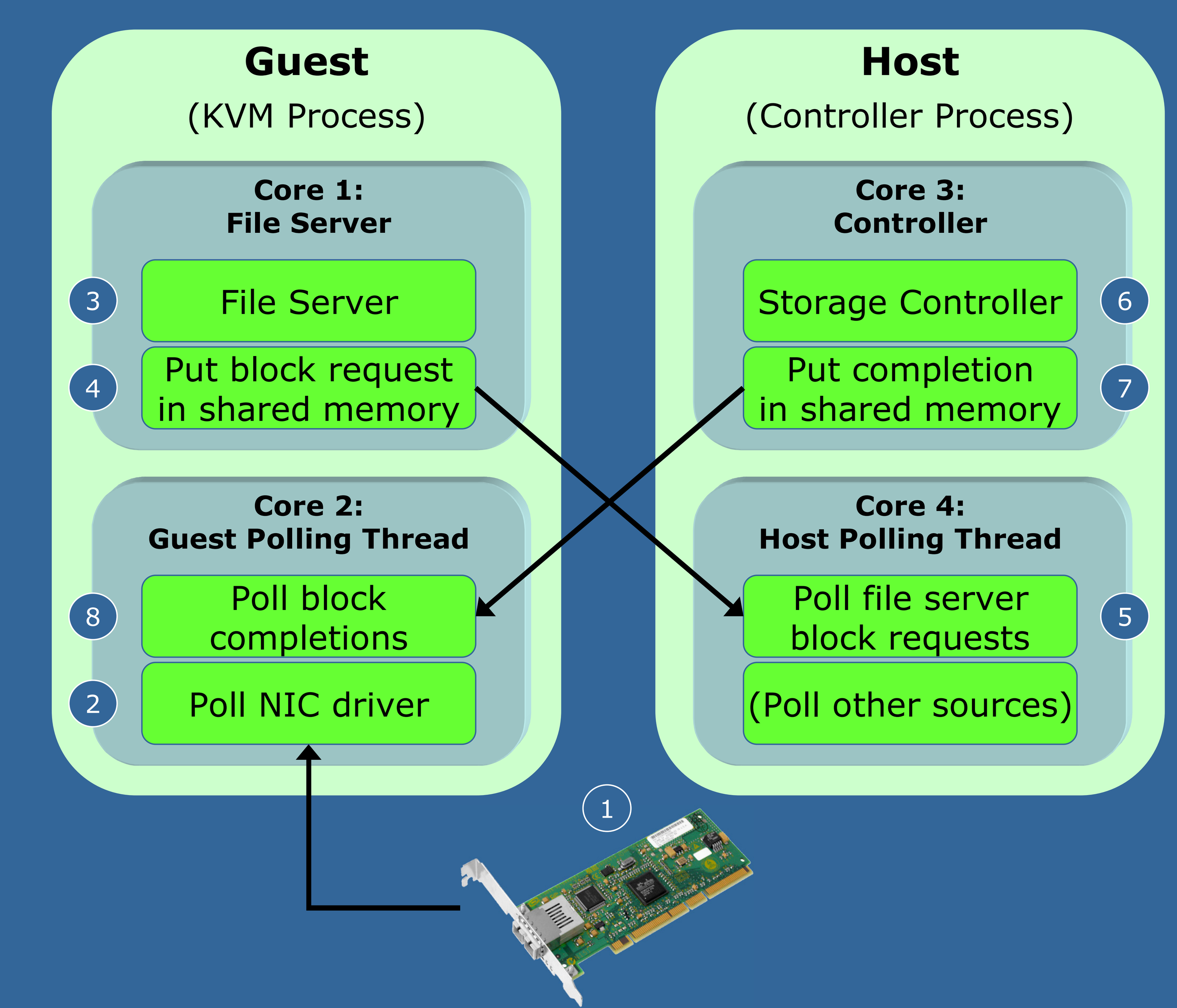
Conclusions

- It is feasible to use a virtual infrastructure to integrate new functions into a storage controller
- Benefit from performance and hardware cost of deep integration
- Benefit from shorter time to market, isolation, and simpler development model of the gateway approach
- Future work: test with multiple VMs, ELI integration

Overview of Our Optimizations

- Allocate a core on the guest for polling the network device, as well as block completions coming from the host
- Use the storage controller's existing polling thread to poll for block requests coming from the guest
- Statically allocate CPU cores and memory
- Use *HugePages* for backing the guest's memory to avoid page faults
- Boot guest kernel with idle=poll to avoid exits from halt/mwait
- Modify thread priorities and affinities

Cache Miss Workflow



Network Micro-benchmarks

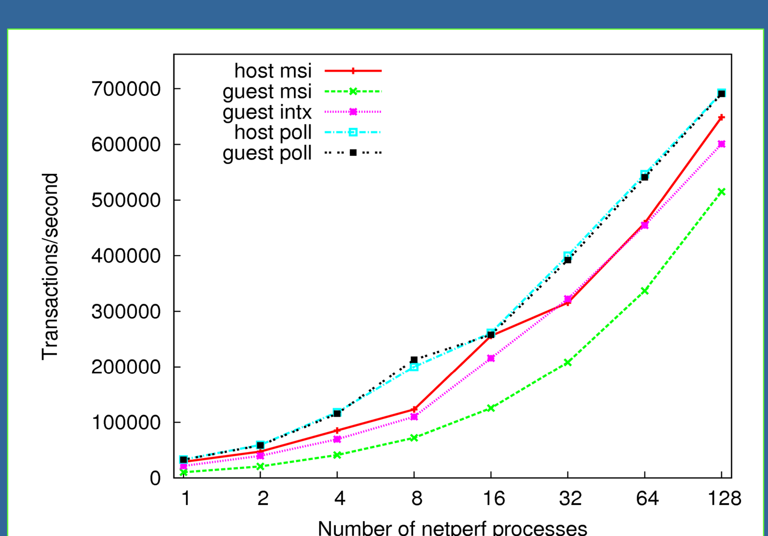
Network latency: ping flood

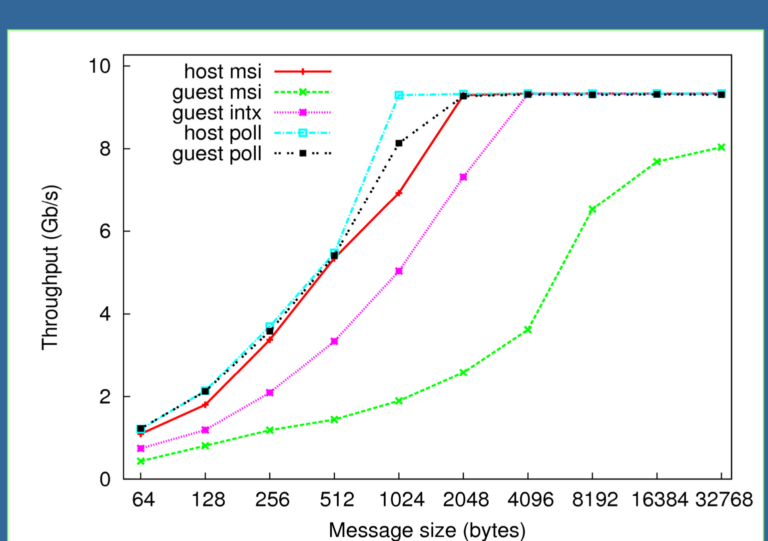
0% latency overhead

	Bare metal	Guest
No polling	24 μs	49 μs
Polling	21 μs	21 μs

Netperf req-resp & send

Negligible throughput overhead





Block Micro-benchmarks

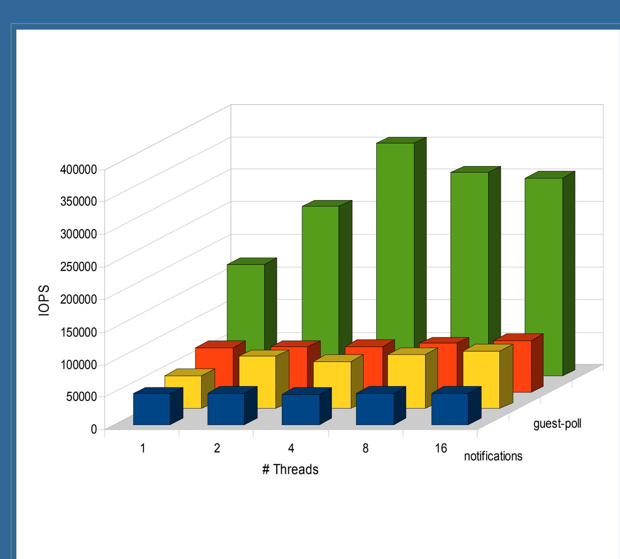
Block latency: 4KB sync writes

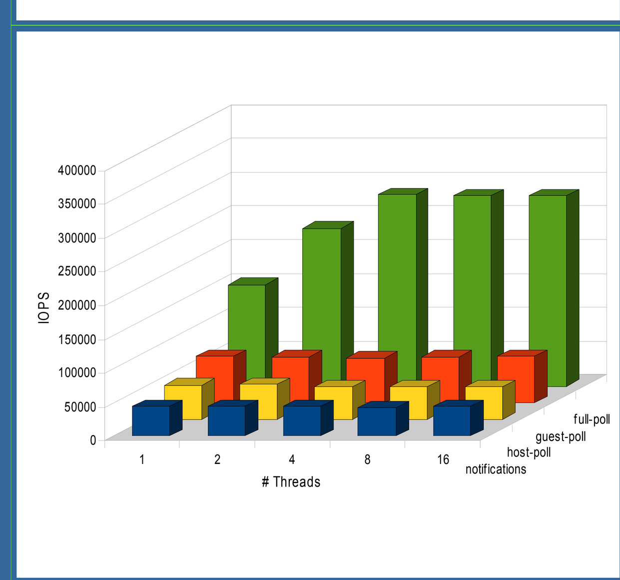
~5% overhead for fastest SPC-1 cache hits (130 μs)

	Initial	Optimized
Total Latency	50 μs	15.9 μs
Added Latency	49 μs	6.6 μs

Block throughput: 4KB sync I/Os

Read IOPS: 350K (↑7.3x)
Write IOPS: 284K (↑6.5x)

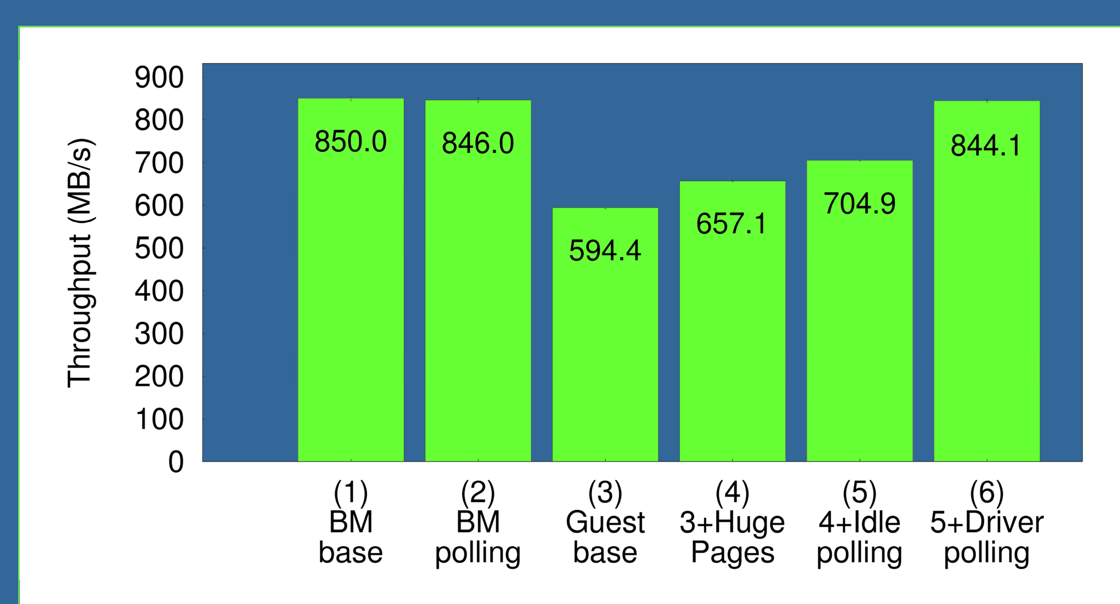




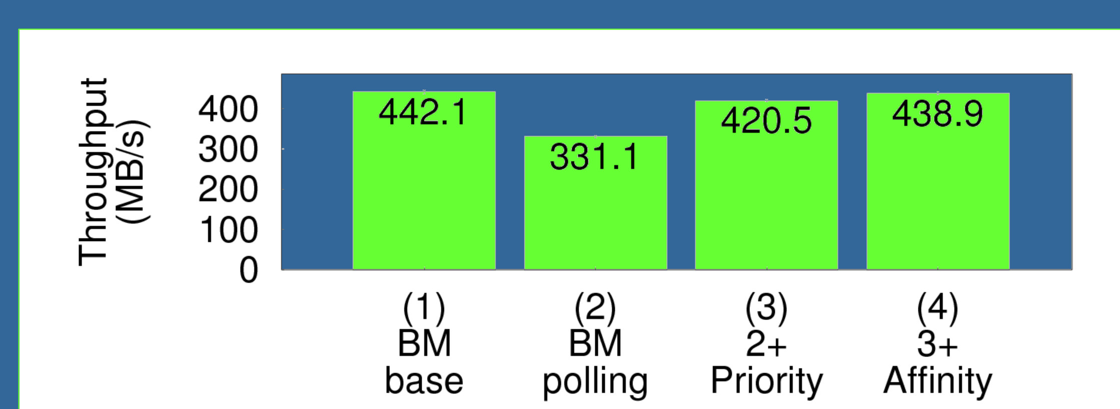
File Server Workload

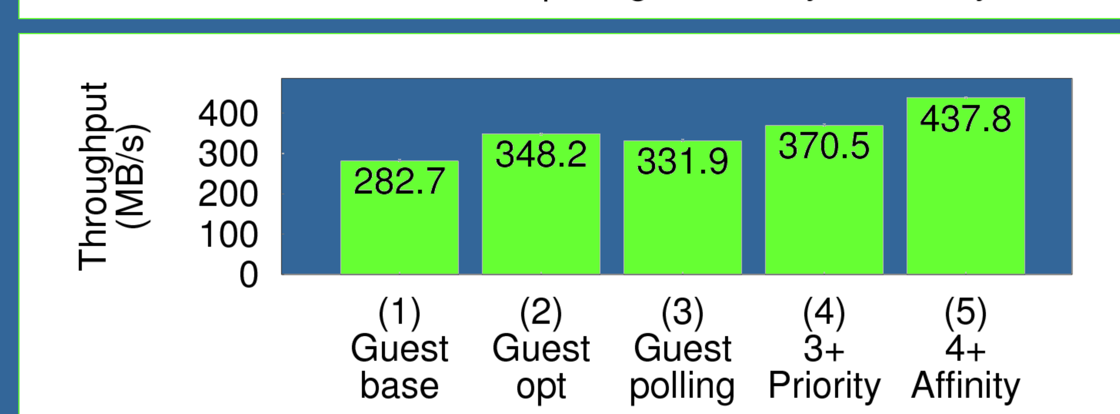
Workload: 4KB read cache miss

6 cores



4 cores





We achieve bare-metal performance running a file server in a VM with 4 and 6 cores