Virtio SCSI

An alternative virtualized
storage stack for KVM

Stefan Hajnoczi
stefanha@linux.vnet.ibm.com

Paolo Bonzini
pbonzini@redhat.com

Overview

- Limitations in QEMU's storage stack
- virtio-scsi: a new storage stack for KVM
- Improving QEMU as a SCSI target
- tcm_vhost: using the in-kernel target with KVM

virtio-blk: features and shortcomings

- **High performance**
  - Paravirtualized device
  - Ring buffers provide a simple and efficient mechanism for guest-host communication
- Limited features

- Limited scalability

- Not a drop-in replacement

virtio-blk: features and shortcomings

- High performance
  - Paravirtualized device
  - Ring buffers provide a simple and efficient mechanism for guest-host communication
- **Limited features**
  - Even trivial new features require a change to the spec
  - Limited SCSI passthrough
  - No access to advanced features
- Limited scalability

- Not a drop-in replacement

virtio-blk: features and shortcomings

- High performance
  - Paravirtualized device
  - Ring buffers provide a simple and efficient mechanism for guest-host communication
- Limited features
  - Even trivial new features require a change to the spec
  - Limited SCSI passthrough
  - No access to advanced features
- **Limited scalability**
  - One PCI device per disk
  - Multifunction devices and PCI bridges help, but they are not really a solution
- Not a drop-in replacement

virtio-blk: features and shortcomings

- High performance
  - Paravirtualized device
  - Ring buffers provide a simple and efficient mechanism for guest-host communication
- Limited features
  - Even trivial new features require a change to the spec
  - Limited SCSI passthrough
  - No access to advanced features
- Limited scalability
  - One PCI device per disk
  - Multifunction devices and PCI bridges help, but they are not really a solution
- **Not a drop-in replacement**
  - /dev/vda instead of /dev/vda complicates p2v/v2v

SCSI passthrough limitations

- No support for advanced features
  - Persistent reservations
  - Multipathing
- HBA device assignment is limited
  - No migration
  - Blades have a limited number of slots
  - Exposes the host fabric to the guest
- Kills pretty much all interesting scenarios
  - To be able to send SCSI commands, guests need exclusive access to a disk
  - No two guests can be initiators for the same device at the same time

virtio-scsi: solving virtio-blk limitations

- High performance
  - Keep the efficient design of virtio-blk
- Rich features
  - Feature set depends on the target, not on virtio-scsi
  - Multipath: one virtio-scsi device = one SCSI host
  - Effective SCSI passthrough
  - Multiple target choices: QEMU, lio
- Almost unlimited scalability
  - Thousands of disks per PCI device
- Drop-in physical disk replacement
  - True SCSI devices, good p2v/v2v migration

What is a SCSI transport protocol?

- SCSI defines a set of services exposed by the target
- A transport protocol provides the communication channel between initiator and target
- Many existing protocols:
  - Parallel SCSI
  - SAS
  - Fibre Channel
  - iSCS
  - SRP (IBM vSCSI)

The virtio-scsi transport protocol

- Three or more virtqueues: controlq, eventq, request queues
- Controlq for everything but SCSI commands
  - Invoke task management functions (Abort, Reset, etc.)
  - Subscribe to asynchronous notifications (media change)
- Eventq receives information from the host
  - Selected unit attention events: reset, hot-plug, hot-unplug
  - MMC asynchronous notifications
  - Events are delivered faster, and handled more easily than sense data
  - Guest can be notified of lost events, and fall back to sense data
- Request queues for SCSI commands
  - Multiqueue possible, but *no ordering guarantees*

The two possible targets

- virtio-scsi (device & driver) acts as the initiator
- Who is the target?
  - Userspace QEMU target
  - In-kernel linux-iscsi.org target

QEMU as a SCSI target

- QEMU provides a very basic target:
  - Disk, CD-ROM, passthrough (scsi-generic)
  - 1 logical unit per target
  - Very small subset of the SCSI spec
- Other limitations:
  - No migration support
  - Designed for parallel SCSI
  - Limited hot-plug support

Improving the userspace target

- Modernization started in 0.15 (Hannes Reinecke + myself):
  - Remove relics of parallel SCSI
  - Actually follow the SCSI specification in more cases
  - Autosense
  - New, more easily extensible API
- More work planned
  - Provide a better abstraction for target functionality
  - Scatter/gather lists for speed
  - Migration support
  - More flexible addressing (more LUNs per target)

Improving the userspace target

- All SCSIDevices reimplement parts of the spec
  - Invalid commands
  - Invalid LUN
  - REQUEST SENSE
  - REPORT LUNS
- Other features are completely missing
  - Unit attention
  - Many more
- Abstract operations common to all devices
  - Avoid code duplication
  - Enables advanced features (hotplug, multiple LUNs per target, migration)

The missing features

- Hot-plug
  - Hot-plugging logical units is simple
  - Hot-plugging targets requires collaboration from the transport protocol
  - Supported by virtio-scsi
- Migration
  - Does not happen often
  - Device triggers an internal reset just before migration
  - In-flight requests are resubmitted by the guest OS
- Scatter/gather support
  - Avoids bounce buffering: I/O goes straight from host device to guest memory
  - Generic implementation, not limited to PV devices
  - Can be used by all device models (MegaSAS)

Using the in-kernel SCSI target

New **in-kernel SCSI target** in Linux 2.6.38

- Fabrics: Fibre Channel, FCoE, iSCSI, SRP
- Backstores: Files, Block devices, SCSI pass-through
- Certified in Netgear, QNAP, Synology appliances
- Much more, see http://linux-iscsi.org/

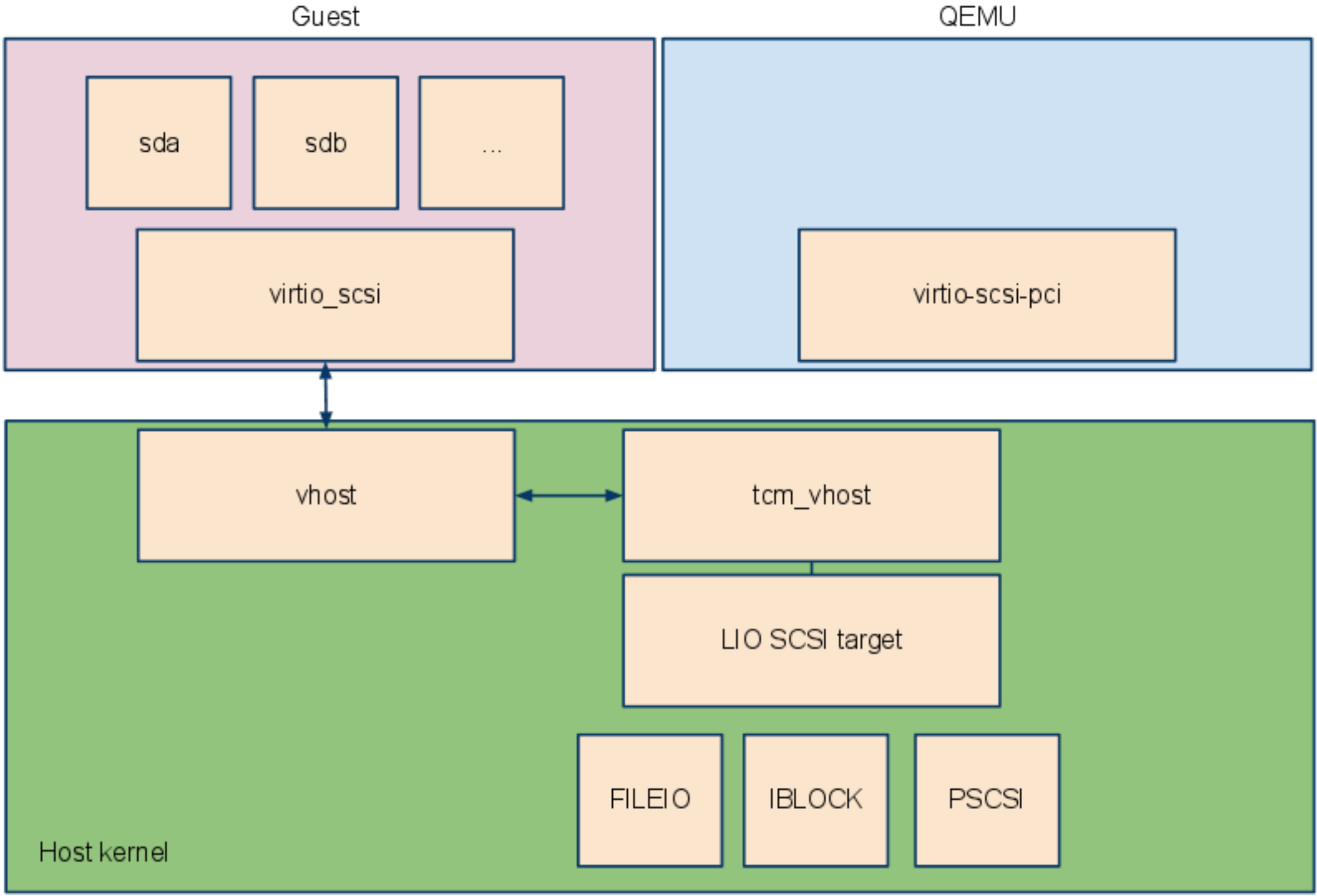
Use LIO Target as KVM's SCSI emulation

- Robust, reliable SCSI target
- Direct guest to host kernel codepath, no userspace
- Powerful LUN and target management tools

How?□□

- Implement virtio-scsi using **vhost**
- **tcm_vhost** fabric module for LIO Target

tcm_vhost architecture: virtio SCSI using LIO Target



Stay tuned for more...

virtio-scsi draft specification is being discussed on qemu-devel@nongnu.org

tcm_vhost kernel code is in Nicholas Bellinger's lio tree: http://git.kernel.org/?p=linux/kernel/git/nab/lio-core-2.6.git;a=shortlog;h=refs/heads/master

virtio-scsi QEMU code is in Stefan Hajnoczi's qemu tree: http://repo.or.cz/w/qemu/stefanha.git/shortlog/refs/heads/virtio-scsi