KVM Live Snapshot support

Jes Sorensen <Jes.Sorensen@redhat.com>
LinuxCon Japan, June 1st 2011

Agenda

Snapshot overview

QEMU snapshots

What is next

Definitions

- Snapshot vs “check point - restart”
  - A check point operation saves the entire system state, including guest memory, processor state, etc.
  - A snapshot creates a coherent copy of a number of block devices at a given time.
  - Live snapshot if a snapshot taken while a virtual machine is running.

This presentation is about live snapshot support!

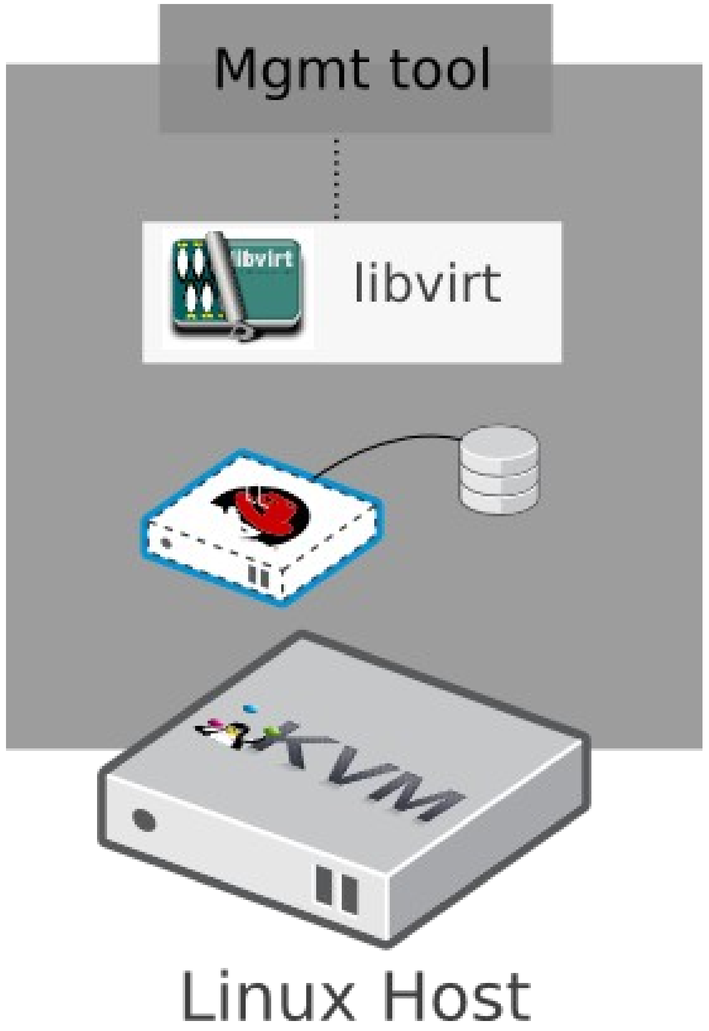
Snapshot 101

- Usage / why snapshots?
    - Ideal for live backup of guests, without guest intervention (kinda sorta)
    - Disk level roll-back, ideal for system upgrade testing etc

Snapshot types

- COW vs full snapshots:
  - Copy-On-Write snapshots creates a new 'root' block device, referencing original device. Original device becomes 'read-only'
    - Variation “referenced” snapshot: binary tree based storage, such as btrfs: data written to new blocks. Snapshot by copy tree structure – once released, unused data blocks are deleted
  - Full snapshot creates a full copy of block device, original device no longer referenced

System example



Snapshots in the I/O stack

- The snapshot operation can be performed at multiple levels of the I/O stack:
  - QEMU snapshots
    - QCOW2, QED
  - LVM
  - File system snapshots
    - btrfs
  - Enterprise storage snapshots
    - NFS, NetApp, EMC etc.

Note: All examples are for storage attached to the host

Snapshot management

- Guest collaboration (agents)
- Coherent API handling all types of media/snapshot mechanisms
- Collapsing/merging snapshots
  - QEMU Live block copy

Guest collaboration

- Agents
  - Guest applications flush data to disk prior to snapshot
    - Optimize 'validity' of backup
    - Valid for traditional backups as well
  - File system freeze
    - Make guest file systems coherent (clean) before snapshot is issued
- Linux guests 'virtagent', work in progress
- Windows: VSS
- Note guest collaboration can **only** ever be best effort! Guests cannot be trusted!

Managing snapshots

- Coherent API for snapshot management:
  - libvirt
- To issue snapshot, management tool needs to know:
  - Storage devices available (QCOW, BTRFS, LVM, enterprise, etc)
    - 'driver' for each device
  - Preferred storage device for snapshot (if multiple layers can do snapshot)
  - Naming conventions
  - Expected output device

Agenda

Snapshot overview

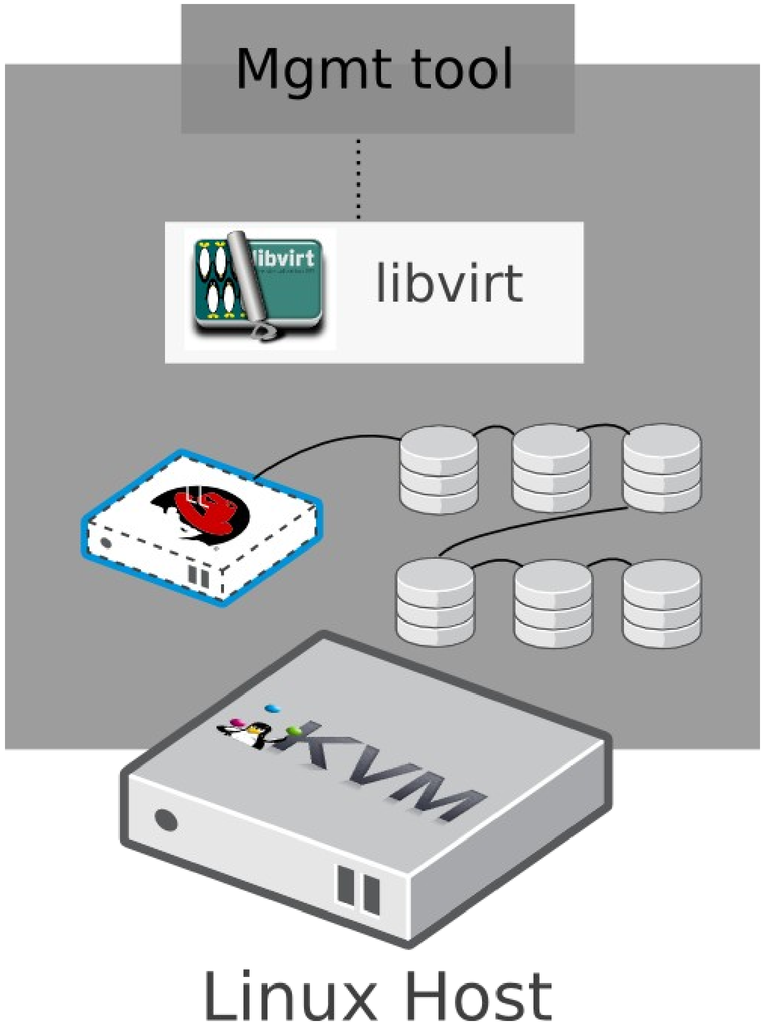
QEMU snapshots

What is next

QEMU snapshots

- Based on COW images
  - References back to previous block image (file, raw device, LVM volume etc.)
  - Supports snapshot of any block format, including raw devices into QCOW2 or QED
  - Results in chain of cow images
    - Snapshot of snapshot of snapshot of snapshot.......
    - Potential performance issue

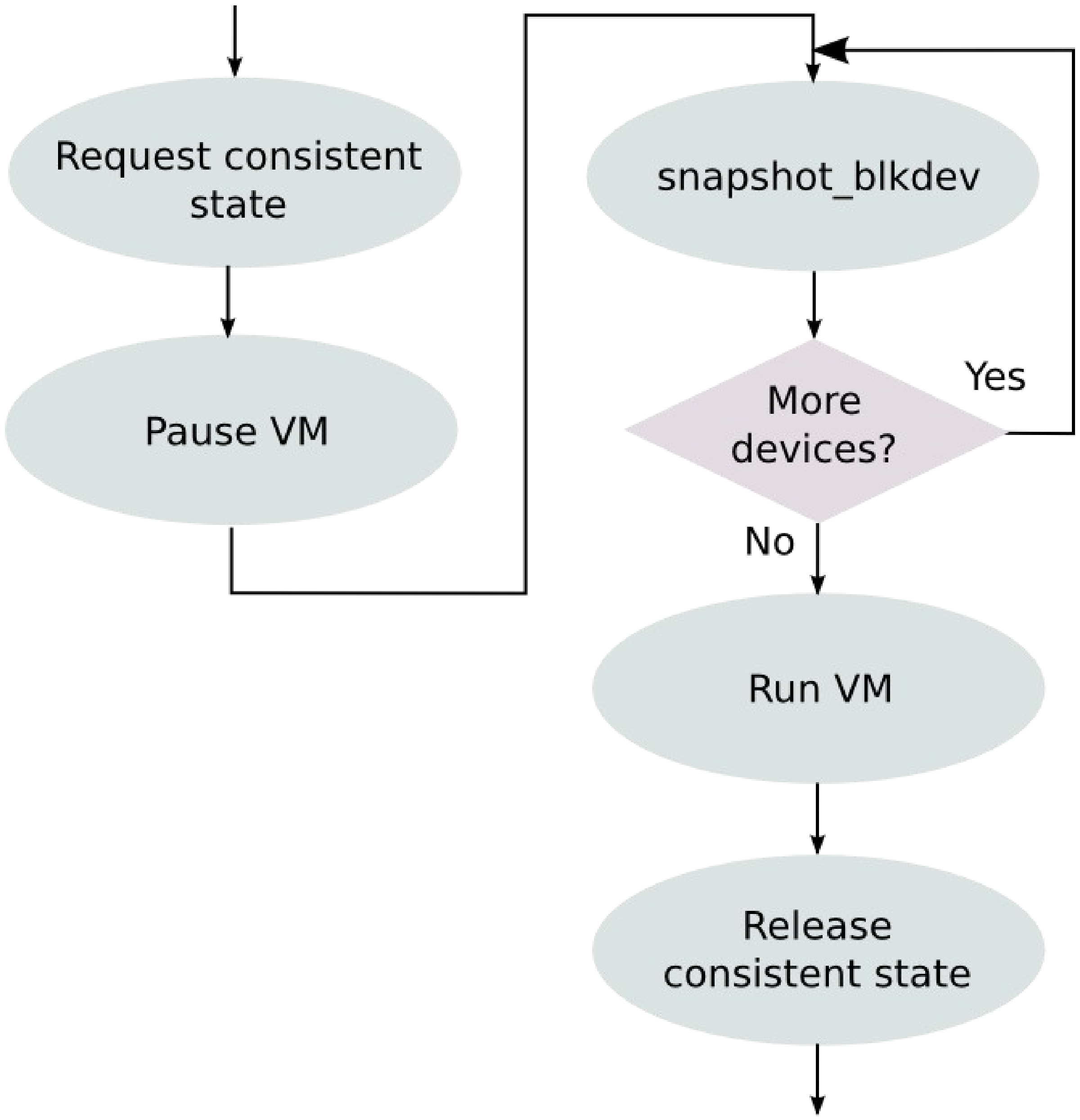
Snapshot chain



Requirements

- Support existing file formats
- Support QCOW2 on raw block devices or LVM
- Simple to use
- Fast
- Support for snapshot of multiple block devices in parallel

Overall snapshot flow



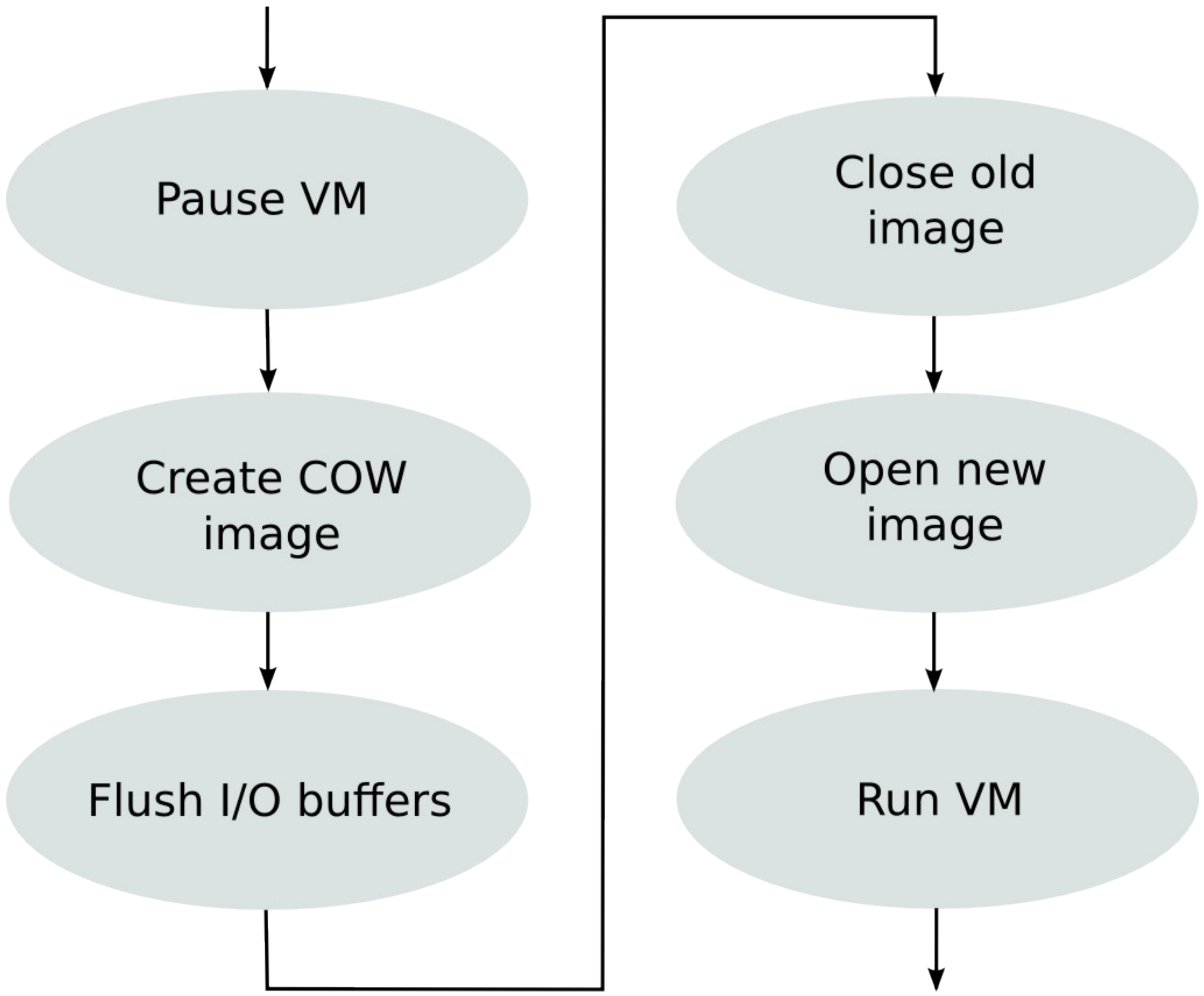
Implementation

- Implemented by refactoring existing external snapshot code used by qemu-img
- One device at a time
  - Multiple devices supported by pausing QEMU during snapshot operation
- Allows for pre-created target image, or QEMU will create it
- Human monitor command:

```
snapshot_blkdev device [new-image-file] [format]
```

- QMP command
  - In progress / under discussion

QEMU flow of live snapshot



virtagent features

- Added support for
  - fsfreeze
    - Freezes all file systems, marking them 'clean'. All writes to a frozen file system will block/sleep
  - fsthaw
    - Thaws all file systems, allowing pending writes to continue
- Requires careful handling as any write to a frozen file system will cause process to sleep!
  - No logging from guest agent during freeze!

Agenda

Snapshot overview

QEMU snapshots

What is next

What is next

- Support asynchronous parallel snapshots
  - Will require pre-created target image
- virtagent guest application call-out API
  - Notification prior to snapshot, allowing application to write out buffers to disk
  - Notification upon snapshot completion, for application to continue normal operation
- libvirt support (in progress)
  - Support for multiple backends (QEMU, btrfs, LVM etc.)
- Higher level management tools to use libvirt API

Conclusions

- It works!
- Performance is very reasonable, despite simple implementation
  - Once file systems frozen, guests quickly stall
- Still more work to be done for asynchronous support
- More work to be done in management layers

Questions ?