Under the Hood with Nova, Libvirt and KVM

Rafi Khardalian, CTO Metacloud

Introduction

About Me

- Who am I and why am I here?
- OpenStack contributions to Nova
- Our unique perspective
  - Broad deployment of production clouds worldwide
  - Centrally managed and supported
  - Large-scale infrastructure operations background
  - Long-running environments with long-running instances
  - Highly diverse set of workloads and use cases

Fundamentals

QEMU (KVM)

metacloud

- KVM is hardware accelerated QEMU; converged project as of QEMU 1.3
- Interactions directly with QEMU should be limited
  - Livbvirt provides most/all of the necessary interfaces
- Do not assume upgrades are seamless (hint: they are not)
- QEMU-monitor interface available, accessible through Libvirt*

QEMU versions provided by Ubuntu for Precise (12.04 LTS):

OpenStack Release	QEMU Version
Grizzly	1.0**
Havana	1.5
Icehouse	2.0

Libvirt

- Handles all management and interaction with QEMU
- Instances (VMs) are defined in Libvirt via XML; referred to a “domain”
- Translates XML to command line options for calling QEMU
- Become comfortable with ‘virsh’
- Libvirt XML reference: http://tinyurl.com/libvirt-xml

Libvirt Versions Provided by Ubuntu for Precise (12.04 LTS):

OpenStack Release	QEMU Version
Grizzly	1.0.2**
Havana	1.1.1
Icehouse	1.2.2

Nova Integration

Nova Compute: Workflow

- Compute Manager:
  - File: nova/compute/api.py
  - File: nova/compute/manager.py

  - Makes calls directly into the driver
  - References to self.driver.<method> are key here
  - Understand what data is being passed in and where

- Nova Libvirt Driver:
  - File: nova/virt/libvirt/driver.py
  - Files: nova/virt/libvirt/*.py

- **Expect to have to read code and become comfortable with doing so**

Spawn

metacloud

- Nova CLI Action: ‘nova boot’

- API -> Scheduler -> Compute (manager) -> Libvirt Driver
  - Compute manager handles network allocation early in the process (commonly confused with scheduler)

- Create disk files (assuming default configuration):
  - Download image from Glance into instance_dir/_base and convert it to RAW (unless it already exists)
  - Create instance_dir/uuid/{disk, disk.local, disk.swap}
    - Create QCOW2 “disk” file, with backing file from the _base image
      - Virtual size set in the QCOW2 image if disk size > 0**
    - Create QCOW2 “disk.local” and “disk.swap” (use of swap makes me sad)
    - Really, don’t use swap in VMs. I’m serious.

Spawn (cont'd)

metacloud

- Generate the libvirt XML and write a copy to the instance_dir
  - instance_dir/libvirt.xml is **never** used by Nova
- Establish volume connections (for boot-from-volume)
  - Operations executed depend on volume driver (examples):
    - iSCSI: Connections made via tgt or iscsiadm
    - RBD: Generates XML for Libvirt; rest handled within QEMU
- Build the supporting network stack for the instance
  - Again, specific operations are driver dependent (assume nova-network here)
  - Bring up any necessary bridges/VLANs
  - Create the security groups (iptables) for the instance

Spawn (cont'd)

metacloud

- Define the domain with Libvirt, using the XML generated earlier in this process (from memory, not disk)
  - Equivalent of 'virsh define instance_dir/<uuid>/libvirt.xml'
- Now, actually start the instance
  - Equivilent of 'virsh start <uuid>' or 'virsh start <domain name>'
- Additional notes
  - Consider a failure to spawn a permanent failure. It should never happen and you should diagnose the issue when it does.
  - The most common failures occur during scheduling; inability to satisfy the user's request (example: resource exhaustion)

Reboot

metacloud

- Two types of reboot available via the API: hard and soft
  - Soft relies completely on the guest OS and ACPI passed through QEMU
  - Hard is at the hypervisor and Nova level and more relevant here
  - Nova CLI: ‘nova reboot’ or ‘nova reboot –hard’
- Hard reboot is the sledge-o-matic of “just fix it” operations
- Hard reboot makes zero assumptions about the state of the hypervisor
  - Notable effort has been placed to make internal operations idempotent, and call them here
- The combination of ‘nova reset-state –active’ and hard reboot is powerful and can fix countless issues
  - Most instance task and power states can actually be handled by hard reboot, even when blocked by the API

Hard Reboot Workflow

How hard reboot resolves most issues:

- Destroy the domain
  - Equivalent of ‘virsh destroy’
  - Does not destroy data, only the QEMU process
  - Effectively a ‘kill -9’ of the QEMU process
- Re-establish any and all volume connections
- Regenerate the Libvirt XML
- Check for and re-download any missing backing files (instance_dir/_base)
- Plug VIFs (re-create bridges, VLAN interfaces, etc.)
- Regenerate and apply iptables rules

Suspend

metacloud

- Nova CLI action: ‘nova suspend’
- Equivalent of ‘virsh managed-save’
- The name is misleading, behavior is that of hibernate
- Questionable value, with several issues to consider
  - Saved memory state consumes disk space equal to instance memory
  - This disk space is not represented in quotas anywhere
  - Neither migration nor live migration deal with this state
  - Can be achieved by the guest OS if needed
  - Installed QEMU version can change between suspend and resume
    - Should work, frequently does not in practice
- Resume simply issues the equivalent of ‘virsh start’
  - Libvirt behaves differently simply due to the existence of the managed save file

Live Migration

metacloud

- Nova CLI Action: ‘nova live-migration [--block-migrate]’
- Two types of live migration with largely different code paths: normal and “block” migrations
- The normal live migration requires the source and target hypervisor both have access to the instance’s data (shared storage, i.e. NAS, SAN)
- Block migration has no special storage requirements. Instance disks are migrated as part of the process.
- Live migration is one of the most sensitive operations in regards to the version of QEMU running on the source and destination
- Heavy lifting is handled by Libvirt

Live Migration Workflow

metacloud

What happens behind the scenes?

- Verify the storage backend is appropriate for the migration type
  - Perform a shared storage check for normal migrations
  - Do the inverse for block migrations
  - Checks are run on both the source and destination, orchestrated via RPC calls from the scheduler

- On the destination
  - Create the necessary volume connections
  - If block migration, create the instance directory, populate missing backing files from Glance and create empty instance disks

- On source, initiate the actual live migration (migrateToURI)

- Upon completion, regenerate the Libvirt XML and define it on the destination

Resize/Migrate

metacloud

- Resize/Migrate are grouped because they actually use the same code
- Migrate differs from live migrate in that it is intended for cold migrations (Libvirt domain is not running)
- Requires SSH key pairs be deployed for the user running nova-compute across all hypervisors
- Resize can and frequently does result in a migrate, since the target flavor might not fit on the current hypervisor
  - By default, the resize will always pick a new target unless “allow_resize_same_host = True”
- Resize will not allow shrinking a disk, since it is unsafe

Resize / Migrate Workflow

- Nova developers know operation needs a significant rework (you will see why)
- Shutdown the instance (ungraceful, ‘virsh destroy’) and disconnect volume connections
- Move the current directory for the instance out of the way (instance_dir -> instance_dir_resize)
  - Resized instance will be built in a temp directory
- If using QCOW2 with backing files (the default), convert the image to be flat
  - Time consuming, resource heavy operation
- For shared storage, move the new instance_dir into place. If not, copy everything via SCP
  - Slow, slow, slow

Snapshots

metacloud

- Two code flows with completely different behavior: “live” snapshot and “cold” snapshot
- Filesystem or data consistency cannot be guaranteed with either form
- Live snapshots were introduced with Grizzly
  - requires Libvirt 1.0.0 and QEMU 1.3
  - No special config required, Nova will handle this automatically
- Cold snapshot results in a disruption to instance availability, here is the workflow:
  - Normalize the instance’s state to be shutdown; executes managed-save if running
  - Once shutdown, executes qemu-img convert to create a copy of the disk in the same format as the instance’s original Glance image
  - Return the instance to its original state
  - Upload the copied/converted image to Glance

Snapshots (Live)

metacloud

Live snapshot workflow:

- Perform checks to determine whether the hypervisor meets the requirements for live snapshot
  - QEMU version check is not always correct**
- The instance needs to be in a “running” state, otherwise we fall back to cold
- Create an empty QCOW2 image in a temp dir
- Via Libvirt (to QEMU), establish a mirror (via block rebase) from our instance disk to the empty disk
- Poll on the status of the block rebase until there are no bytes left to mirror, then break the mirror; we now have a copy of the instance disk
- Using qemu-img, convert the copy to flatten the image and eliminate the backing file
- Upload the image to Glance in a thread

Final Notes / Tips

metacloud

- The most common issues stem from the most basic requirements, such as lack of disk space to copy snapshots around
- Read the code. Read the code. And when you're done, read the code. Never assume anything behaves a particular way.
- Having debug logging enabled, even in production, is important for Nova.
- Configuration of the services and tools which Nova depends on is just as critical as the configuration of Nova itself
  - Example: Libvirt managed save files consume significant space and consumes a fair amount of IO
- More to come in future sessions …

Questions

Thank You

http://jobs.metacloud.com