Go Search

Navigation

- Main Page
- Status
- Guest Support Status
- KVM-Autotest
- HOWTO
- Migration
- Lists, IRC
- Documents
- Downloads
- Bugs
- Code
- TODO
- FAQ

banners

From KVM

Contents

Migration

Introduction

KVM currently supports savevm/loadvm and offline or live migration Migration commands are given when in qemu-monitor (Alt-Ctrl-2). Upon successful completion, the migrated VM continues to run on the destination host.

Note

You can migrate a guest between an AMD host to an Intel host and back. Naturally, a 64-bit guest can only be migrated to a 64-bit host, but a 32-bit guest can be migrated at will.

There are some older Intel processors which don't support NX (or XD), which may cause problems in a cluster which includes NX-supporting hosts. To disable NX for a given guest, start it with such a parameter: -cpu qemu64,-nx

Requirements

- The VM image is accessible on both source and destination hosts (located on a shared storage, e.g. using nfs).
- It is recommended an images-directory would be found on the same path on both hosts (for migrations of a copy-on-write image -- an image created on top of a base-image using "qemu-image create -b ...")
- The src and dst hosts must be on the same subnet (keeping guest's network when tap is used).
- Do not use -snapshot qemu command line option.
- For tcp: migration protocol
- the guest on the destination must be started the same way it was started on the source.

highlights / merits

- Almost unnoticeable guest down time
- Guest is not involved (unique to KVM Live Migration [#1 1])
- Capability to tunnel VM state through an external program (unique to KVM Live Migration [#1 1])
- ssh/gzip/bzip2/gpg/your own
- Upon success guest continues to run on destination host, upon failure guest continues to run on source host (with one exception)
- Short and Simple
- Easy to enhance
- Hardware independence (almost).
- Support for migration of stopped (paused) VMs.
- Open

Anchor(1) 1 These features are unique to KVM Live Migration as far as I know. If you know of other hypervisor that support any of them please update this page or let me (Uri) know.

User Interface

The user interface is through the qemu monitor (alt-ctrl-2 on the SDL window)

Management

```
migrate [-d] <URI>
migrate_cancel
```

The '-d' will let you query the status of the migration.

With no '-d' the monitor prompt returns when the migration completes. URI can be one of 'exec:<command>' or tcp:<ip:port>

Status

```
info migrate
```

Migration Parameters

```
migrate_set_speed <speed>   set bandwidth control parameters (max transfer rate per second)
```

Example / HOWTO

A is the source host, B is the destination host:

TCP example:

1. Start the VM on B with the exact same parameters as the VM on A, in migration-listen mode:

```
B: <qemu-command-line> -incoming tcp:0:4444 (or other PORT))
```

2. Start the migration (always on the source host):

```
A: migrate -d tcp:B:4444 (or other PORT)
```

3. Check the status (on A only):

```
A: (qemu) info migrate
```

Offline example:

1. unlimit bandwidth used for migration:

```
A: migrate_set_speed 1g
```

2. stop the guest:

```
A: stop
```

3. continue with either TCP or exec migration as described above.

Problems / Todo

- TSC offset on the new host must be set in such a way that the guest sees a monotonically increasing TSC, otherwise the guest may hang indefinitely after migration.
- usbdevice tablet complains after migration.

- handle migration completion better (especially when network problems occur).
- More informative status.
- Migration does not work while CPU real-mode/protected mode are still changing.

savevm/loadvm to an external state file (using pseudo-migration)

- To be supported directly by Migration Protocols, but until then...
- Save VM state into a compressed file
    - Save

```
stop
migrate_set_speed 4095m
migrate "exec:gzip -c > STATEFILE.gz"
```

-
    - Load

```
gzip -c -d STATEFILE.gz | <qemu-command-line> -incoming "exec: cat"   or
<qemu-command-line> -incoming "exec: gzip -c -d STATEFILE.gz"
```

- Save VM State into an encrypted file (assumes KEY has already been generated)
    - Save

```
stop
migrate_set_speed 4095m
migrate "exec:gpg -q -e -r KEY -o STATFILE.gpg"
```

- Load VM state from an encrypted file

```
gpg -q -d -r KEY STATEFILE.gpg | <qemu-command-line> -incoming "exec:cat"
```

more exec: options

- Send encrypted VM state from host A to host B (Note: ssh is probably better, this is just for show)

-
    - on host A

```
migrate "exec:gpg -q -e -r KEY | nc B 4444"
```

-
    - on host B

```
nc -l 4444 | gpg -q -d -r KEY | <qemu-command-line> -incoming "exec:cat"
```

Algorithm (the short version)

1. Setup

- Start guest on destination, connect, enable dirty page logging and more

2. Transfer Memory

- Guest continues to run
- Bandwidth limitation (controlled by the user)

- First transfer the whole memory
- Iteratively transfer all dirty pages (pages that were written to by the guest).

3. Stop the guest

   - And sync VM image(s) (guest's hard drives).

4. Transfer State

   - As fast as possible (no bandwidth limitation)
   - All VM devices' state and dirty pages yet to be transferred

5. Continue the guest

   - On destination upon success
     - Broadcast "I'm over here" Ethernet packet to announce new location of NIC(s).
   - On source upon failure (with one exception).

Instructions for kvm-13 and below: MigrationQemu0.8.2.