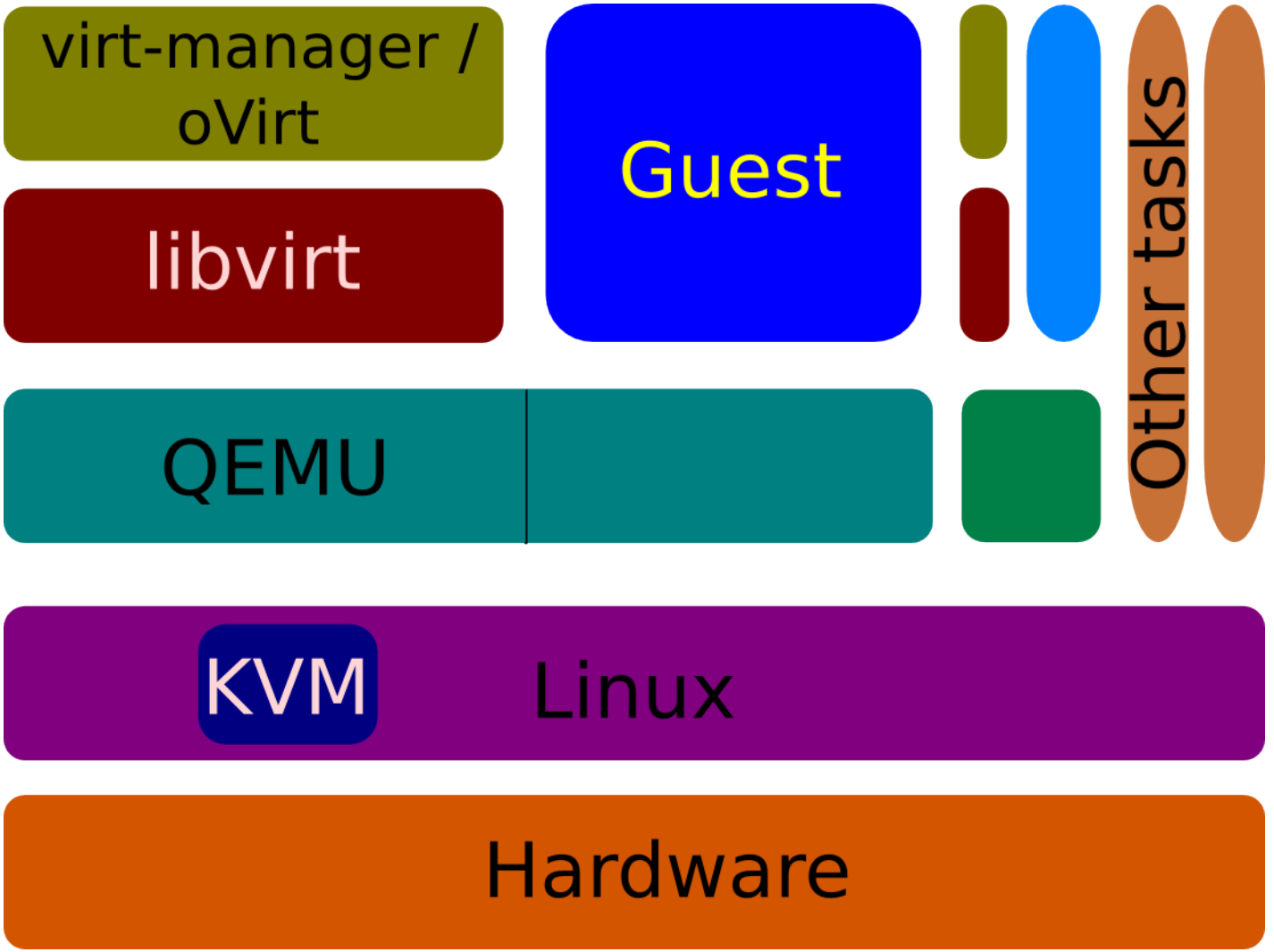
Live Migration of QEMU/KVM Virtual Machines

devconf.cz/2015

Amit Shah | Red Hat | amit.shah@redhat.com

Virtualization



QEMU

- Creates the machine
- Device emulation code
  - some mimic real devices
  - some are special: paravirtualized
- Uses several services from host kernel
  - KVM for guest control
  - networking
  - disk IO
  - etc.

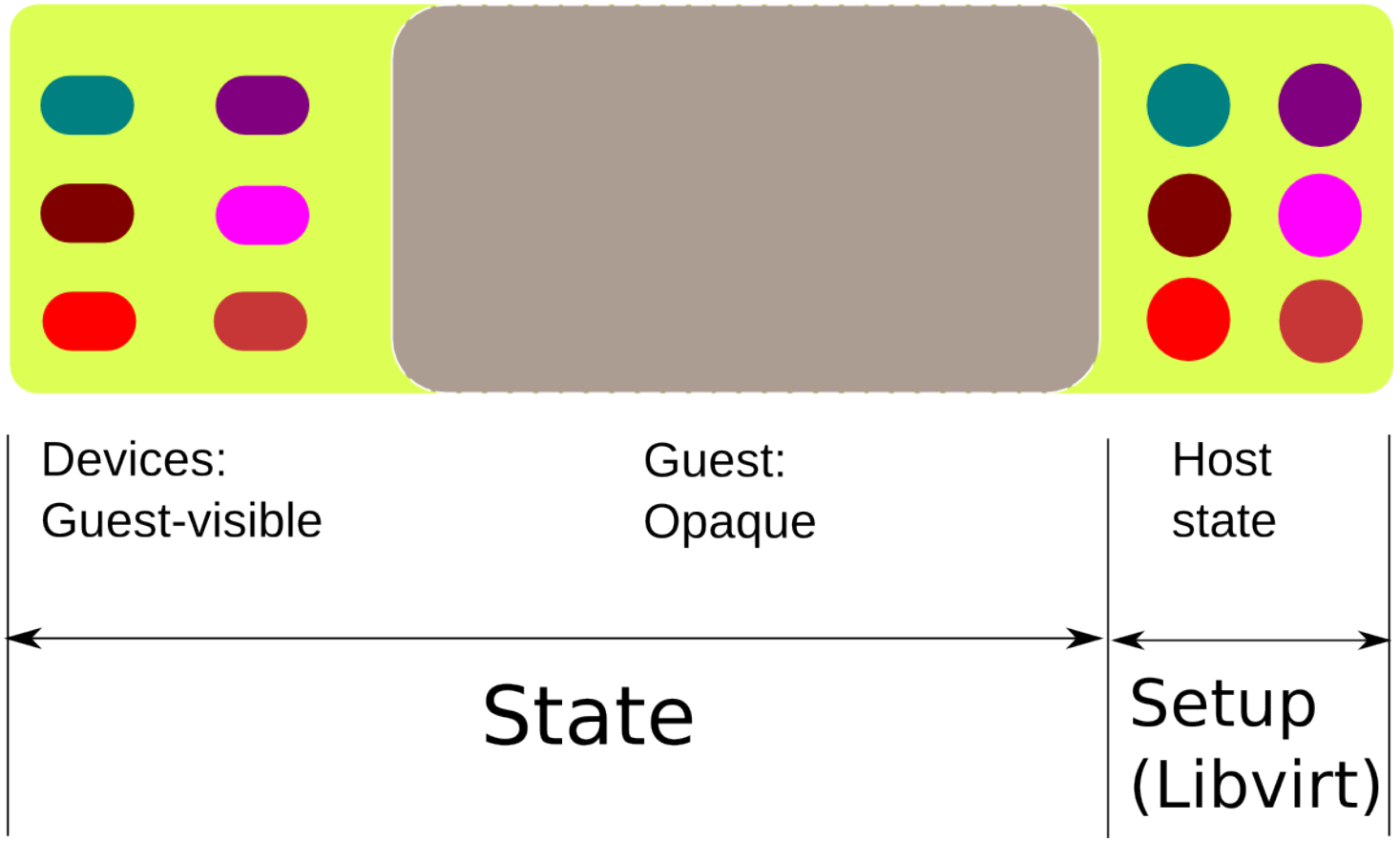
KVM

- Do one thing, do it right
- Linux kernel module
- Exposes hardware features for virtualization to userspace
- Enables several features needed by QEMU
  - like keeping track of pages guest changes

Live Migration

- Pick guest state from one QEMU process and transfer it to another
  - while the guest is running
- The guest shouldn't realize the world is changing beneath its feet
  - in other words, the guest isn't involved in the process
  - might notice degraded performance, though
- Useful for load balancing, hardware / software maintenance, power saving, checkpointing, ...

QEMU Layout


Devices:
Guest-visible
Guest:
Opaque
Host
state
State
Setup
(Libvirt)

Getting Configuration Right

- Shared storage
  - NFS settings
- Host time sync
  - Can't stress enough how important this is!
- Network configuration
- Host CPU types
- Guest machine types
  - esp. if migrating across QEMU versions
  - ROM sizes

Stages in Live Migration

- Live migration happens in 3 stages
- Stage 1: Mark all RAM dirty
- Stage 2: Keep sending dirty RAM pages since last iteration
  - stop when some low watermark or condition reached
- Stage 3: Stop guest, transfer remaining dirty RAM, device state
- Continue execution on destination qemu

Stages

- Live migration happens in 3 stages
- Stage 1: Mark all RAM dirty `<ram_save_setup()>`
- Stage 2: Keep sending dirty RAM pages since last iteration `<ram_save_iterate>`
  - stop when some low watermark or condition reached
- Stage 3: Stop guest, transfer remaining dirty RAM, device state `<migration_thread()>`
- Continue execution on destination qemu

Ending Stage 2 (or Transitioning from Live to Offline State)

- Earlier
  - 50 or fewer dirty pages left to migrate
  - no progress for 2 iterations
  - 30 iterations elapsed
- Now
  - admin-configurable downtime (for guests)
    - involves knowing # of pages left and bandwidth available
  - host policies: like host has to go down in 5 mins, migrate all VMs away within that time

Other Migration Code in QEMU

- General code that transmits data
  - tcp, unix, fd, exec, rdma
- Code that serializes data
  - section start / stop
- Device state

VMState

- Descriptive device state
- Each device does not need boilerplate code
- Each device does not need identical save and load code
  - Which is easy to get wrong

VMState Example

- e1000 device
- `e482dc3ea e1000: port to vmstate`
- `1 file changed,`

  `81 insertions(+),`

  `163 deletions(-)`

VMState Example (before)

```
-static void
-nic_save(QEMUFile *f, void *opaque)
 {
-    E1000State *s = opaque;
-    int i;
-
-    pci_device_save(&s->dev, f);
-    qemu_put_be32(f, 0);
-    qemu_put_be32s(f, &s->rxbuf_size);
-    qemu_put_be32s(f, &s->rxbuf_min_shift);
```

VMState Example (after)

```
+static const VMStateDescription vmstate_e1000 = {
+    .name = "e1000",
+    .version_id = 2,
+    .minimum_version_id = 1,
+    .minimum_version_id_old = 1,
+    .fields      = (VMStateField []) {
+        VMSTATE_PCI_DEVICE(dev, E1000State),
+        VMSTATE_UNUSED_TEST(is_version_1, 4), /* was instance id */
+        VMSTATE_UNUSED(4), /* Was mmio_base.  */
+        VMSTATE_UINT32(rxbuf_size, E1000State),
+        VMSTATE_UINT32(rxbuf_min_shift, E1000State),
```

Updating Devices

- Sometimes devices get migration-breaking changes
- One idea is to bump up version number
  - Adds dependencies from higher versions to lower ones
  - Difficult to cherry-pick fixes to stable / downstreams
- Another is to introduce new subsection

Subsection Example

- commit c2c0014 pic_common: migrate missing fields

```
-          VMSTATE_INT64(timer_expiry,
-                        APICCommonState), /* open-coded timer state */
-          VMSTATE_END_OF_LIST()
+    },
+    .subsections = (VMStateSubsection[]) {
+        {
+            .vmsd = &vmstate_apic_common_sipi,
+            .needed = apic_common_sipi_needed,
+        },
+        VMSTATE_END_OF_LIST()
      }
  };
```

Things Changed Recently

- Guests have grown bigger
  - More RAM
    - Means a lot of time spent transferring pages
  - More vCPUs
    - Means active guests keep dirtying pages

New Features

- autoconverge
- xbzrle
- migration thread
- migration bitmap
- rdma
- block migration

Stuff that's lined up

- postcopy
- debuggability

Future work

- Finish vmstate conversion
- self-describing wire format

Thank You!

Amit Shah | http://log.amitshah.net | amit.shah@redhat.com

Extras

- Feedback: http://devconf.cz/f/62
- FUDCon APAC in Pune, India: Jun 26-28
  - http://fudcon.in