redhat.

Failing migrations – how & why

KVM Forum 2017 - Prague

Dr. David Alan Gilbert / dgilbert@redhat.com
Principal Software Engineer
2017-10-27

They don’t fail often!

But if they do, we have some troubleshooting tips:

http://wiki.qemu.org/Features/Migration/Troubleshooting

Expected to work every time

- Migrations used to be manual, carefully planned
- Now often fully automated
  - Automated load balancing
  - host evacuation now common
- Admins don't know what their guests are doing during migration
  - Rebooting?
  - Installation?
  - Already crashed?
  - Still need the migration to work

Basic expectations

- A good socket connection
- Working hosts
- Accurate error reports

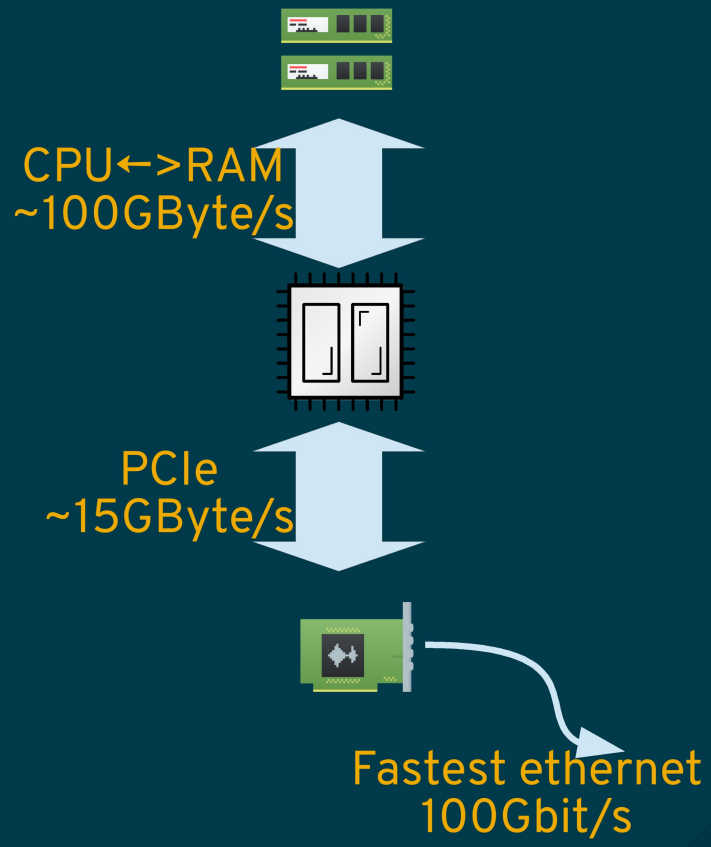
Basic expectations

- A good socket connection
  - Bad NIC drivers corrupting streams
- Working hosts
  - MCEs, bad storage on destination, broken shared storage
- Accurate error reports
  - Mention if the hosts are odd (e.g. nests)

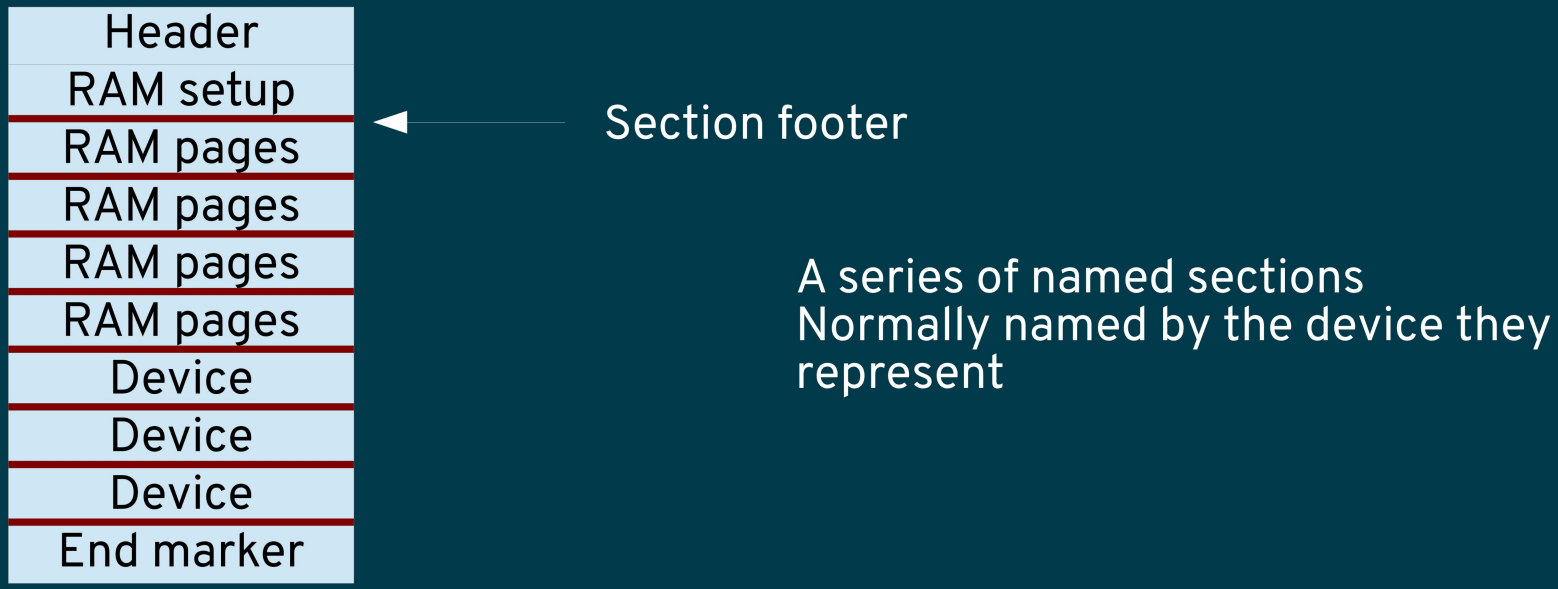
Timeouts

- Busy VMs
  - Dirty memory too fast (10-1000x faster than available networking)
- Poor network bandwidth
  - And shared with other migrations, SAN, client
- Rapid shared disk writes

- Postcopy &/| auto-converge
  - Should get a migrate to complete
- Make sure to cleanup after a timed-out migrate



Structure of a migration stream



A series of named sections
Normally named by the device they represent

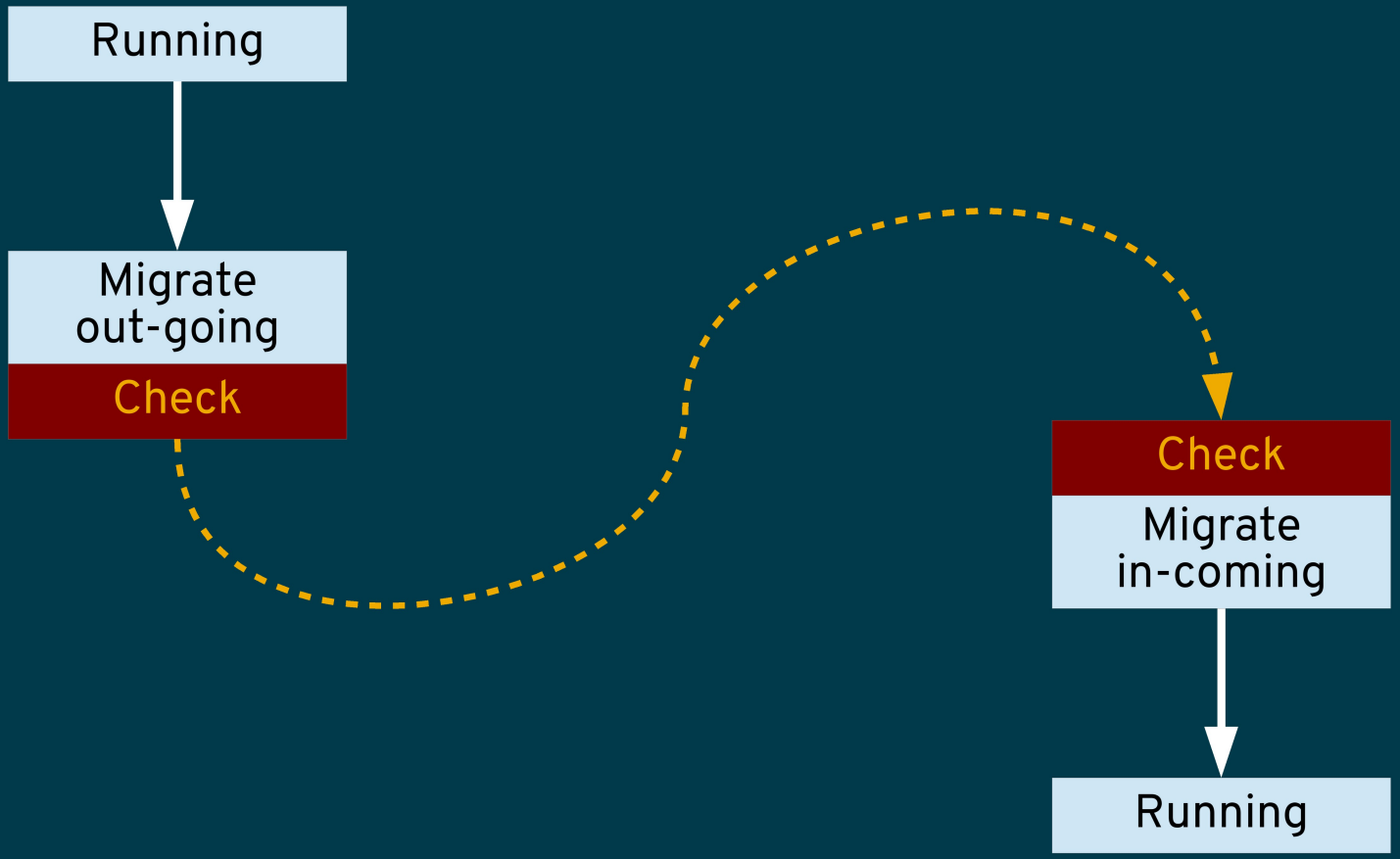
Matching source & destination config

- Matching machine type
  - Machine type name stored in stream (Juan's Configuration section 61964c23)
  - Versioned machine types
    - Try to keep compatible (Amit Shah's scripts/vmstate-static-checker.py)
    - 'interesting' to recover after discovering a mismatch live in the field
- Device addressing/ordering
  - Command line ordering not guaranteed
  - Use explicit addresses (PCI, USB etc)
    - Especially with hot plug
    - Especially with storage (e.g. swap two drives)
- Padding of ROMs

Matching source & destination CPU

- CPU flags – must match
  - Not always an exact match
    - e.g. 100s of CPU variants, no single name can cover them
  - Some less obvious; e.g. number of performance counters vs hyperthread
  - Disappearing features on newer versions & BIOS updates
  - Address space
  - What happens when run on a brand-new host?
    - Pick an old CPU and add flags? Ignore new flags?
- Choose a CPU type for your cloud and stick to it

Devices: Checking state

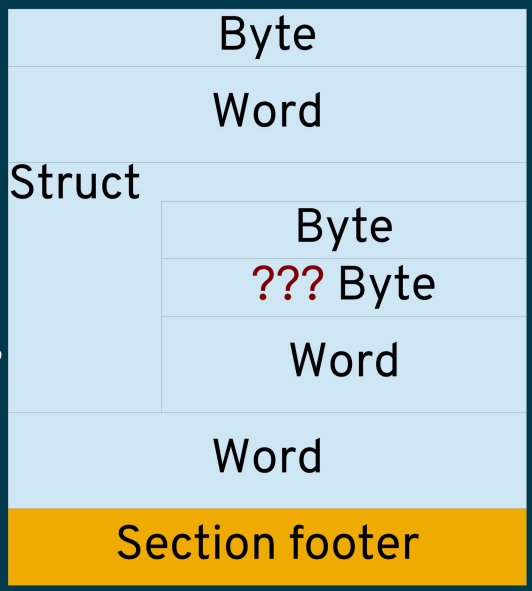


Devices: Checking state

- Devices tend to sanity check during migration
- Don't abort the source
  - If it was running apparently OK prior to migration let it carry on
  - Fail migrate if you must! But try not to
- Sanity check stream on destination
  - Aborting on the destination isn't as bad
- Failing migration due to guest driver behaviour is bad
  - It comes back as a 'migration failure' long before anyone digs into the guest
  - Problem in the cloud where admin doesn't know state of guest
  - Can't wait for the guest to sort itself out before allowing migration
- Watch out for new drivers in the wild
  - Seen cases where we get lots of reports of migration failures when there's a new guest driver version
- error_report !
  - Want something in log to be able to understand migration failure

Devices: Conditional entries

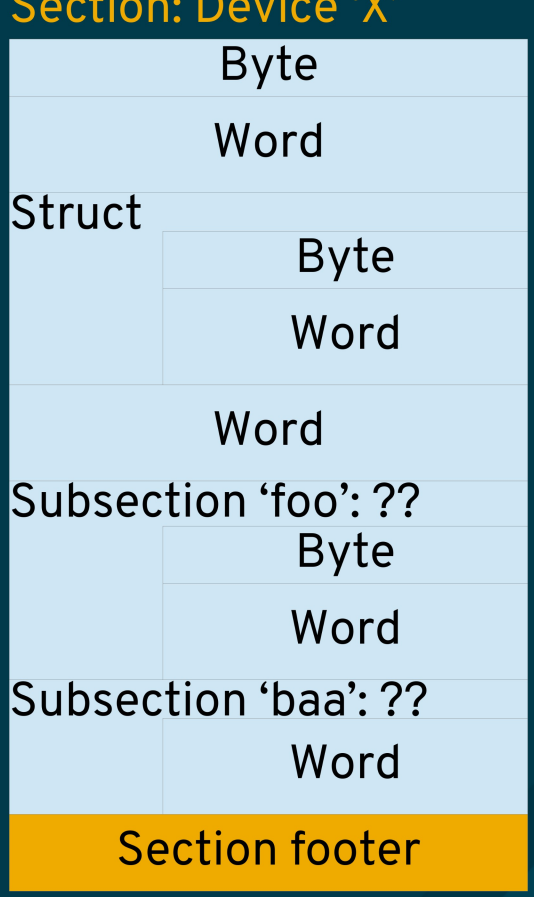
```
VMSTATE_UINT16_TEST(field, myStruct,
bool_func),
```



- Delicate – destination must agree (else odd failures, mostly trapped now by 'section footer')
  - Disagreement leads to arbitrary more/less bytes in stream
  - Easy to break (e.g. different defaults, guest drivers etc) all can cause the bool_func to flip.
- Can tie to:
  - Section version (Bad for reverse compatibility)
  - machine type
  - preferably device property
  - State of device (e.g. only send fifo when enabled)

Devices: Subsections

- Named rather than ordered
  - Error often names unexpected subsection
  - Although still slightly heuristic, so not totally clear to the destination on an unexpected subsection.
- More robust than conditional elements
  - '.needed' tied to boolean function, as for conditionals
- For compatibility:
  - Make device load cope without receiving subsection (forward compatibility)
    - Set up defaults in pre-load
  - Don't send subsection to old machine (reverse compatibility)



Devices: Testing

- Conditionals and subsections can increase test load
  - Test in ‘odd’ states
    - e.g. ejected CDROM
    - Keyboard with full input fifo
    - Pre-initialisation (e.g. migrating during firmware)
    - Re-initialisation (Reboot, screen mode change, reloading driver)
    - OS installers tend to have different/more basic drivers
- Firmware versions
  - Destination uses ROM images migrated from source
    - Even during reboot
    - Can’t rely on adding feature to QEMU and matching BIOS
- Can never assume you’ve tested all OS versions

Devices: PCI

- PCI config space has some fixed bits and some variable, checked on load:

```
get_pci_config_device: Bad config data: i=0xxx read: yy
device: zz cmask: ss wmask: pp w1cmask
```

- 'i' is index in config space
- New features/flags can cause failed migration
- Capabilities link list:
  - Must be in the same order
  - Must contain the same entries (e.g. breaks adding MSI or converting to PCIe)
  - Check with lspci -v in guest

Who failed first?

- Common error messages:

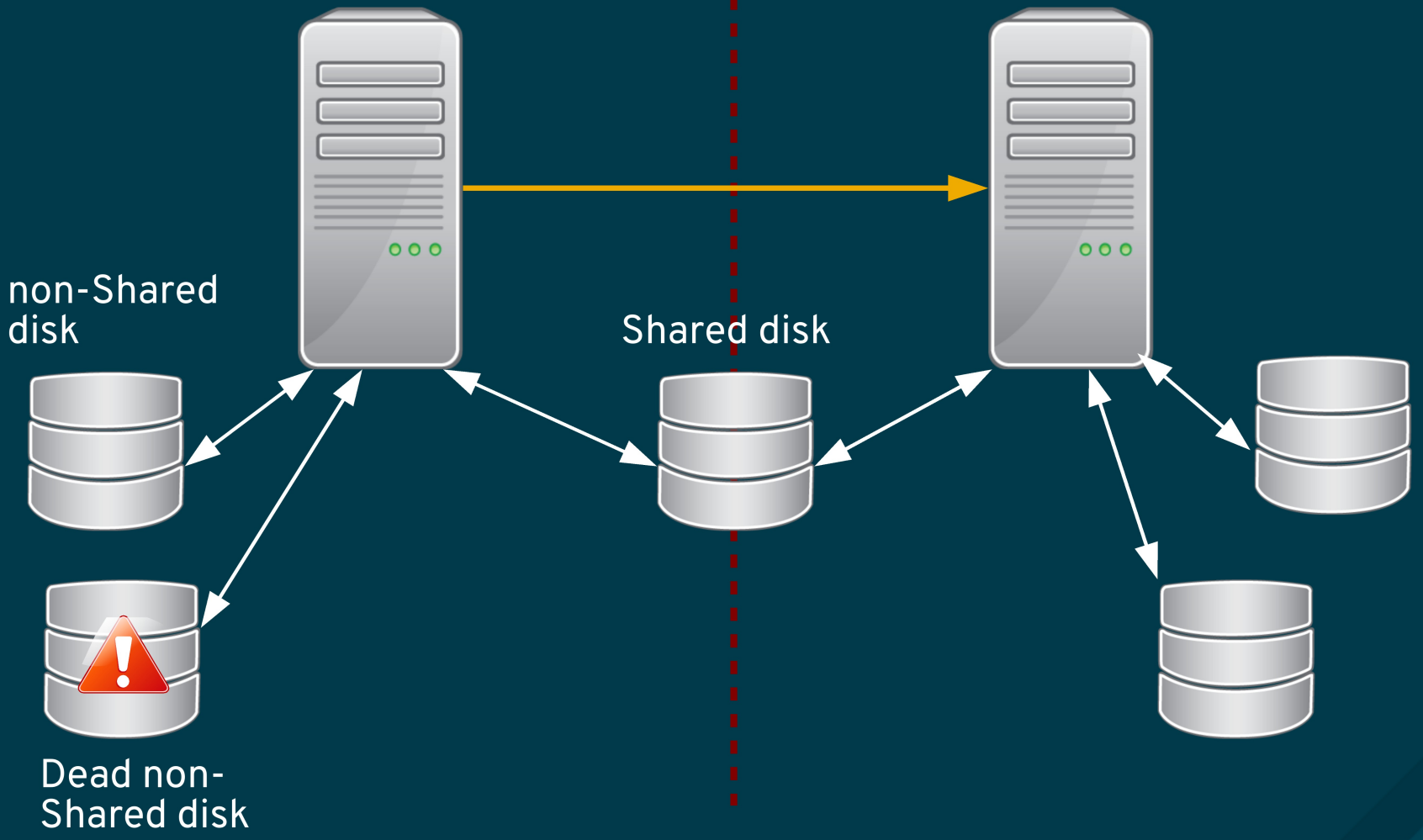
```
2017-10-26T12:MM:SS.ffffffZ qemu-kvm: load of migration failed:
Input/output error
2017-10-26T12:MM:SS.ffffffZ qemu-kvm: socket_writev_buffer: Got err=32
for (131328/18446744073709551615)
```

- The destination complains that it got an I/O error during loading
  - Maybe the source failed and broke the socket
  - Maybe the I/O error was from a disk on the destination
- The source complains it got an I/O error writing to the socket
  - Maybe the destination crashed and broke the socket
- Maybe an actual network problem

Ordering & timing

- MigrationPriority can force ordering
  - e.g. load interrupt controller before PCI devices
    - Only on ARM so far – most use implicit ordering
- Post-load activity:
  - Post-load called after devices state loaded
    - But guest isn't running yet and other devices haven't been loaded
  - Can start a timer in post-load
    - Racy using physical time
    - Can use virtual time
  - Runstate change handler
- Take care about other timers
  - e.g. normal timer loaded by state might fire during migration process
- -S loads to pause
  - Used by libvirt so it reconnects networking/storage before unpausing
- Snapshots

Storage



Storage

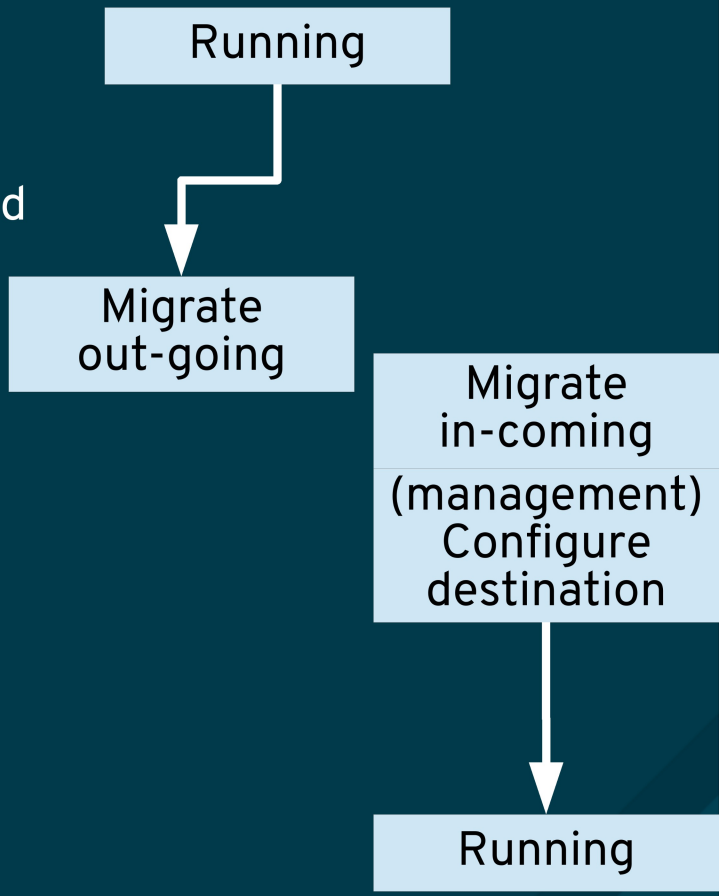
- Take care with shared vs non-shared
  - eg migrating the contents of a shared disk back to itself
  - Permissions/locking on shared disks
- What happens when you migrate with one dead disk?

Storage

- Storage inactivation/draining at end of migration
  - So source has flushed all data before destination starts using it
    - A good point for failures/hangs
      - e.g. one dead but unused device
- Interactions with NBD block-mirroring
  - Mostly separate from main migration process
- Old block-migrate
  - Please use NBD block-mirroring instead
  - Much more entangled with main migration
- Cache mode
  - For POSIX storage cache=none required
    - Otherwise odd corruptions
  - For other storage (e.g. CEPH) – it depends…
    - Depends on the semantics of the library QEMU is using

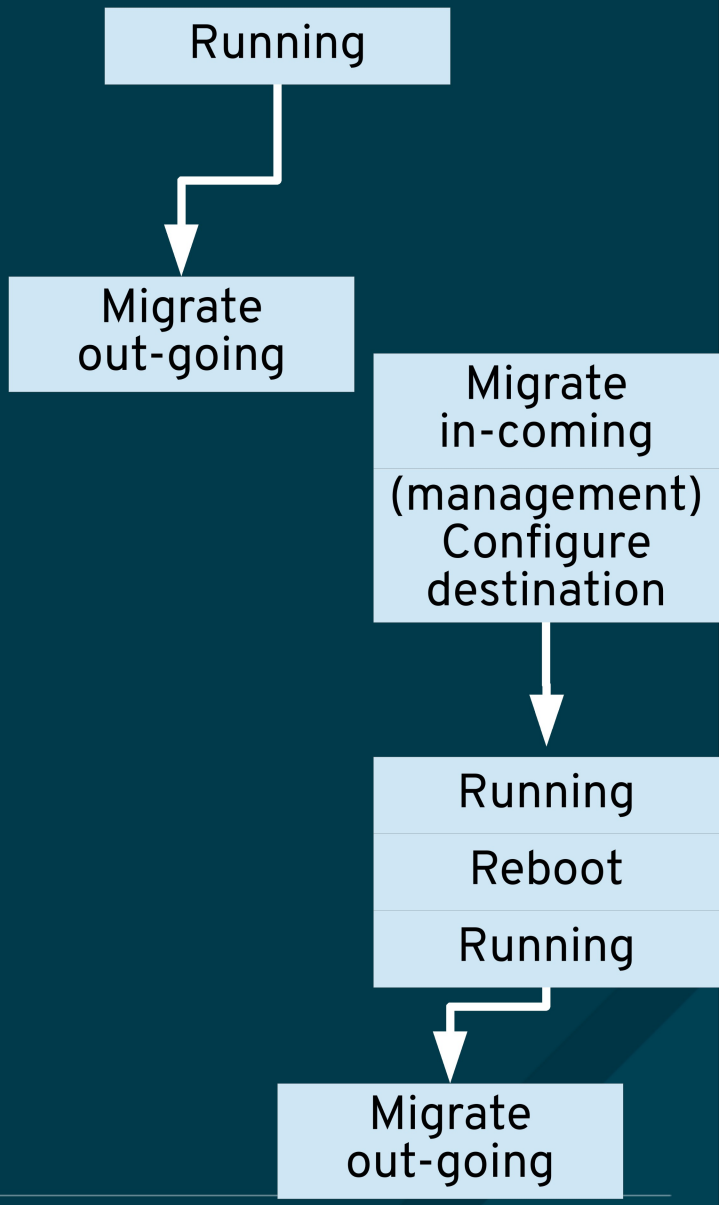
Did it work???

- Source shows success unless there was an IO error on the socket
  - Unreliable for failures in device load at the end
  - Peter Xu's patch allows enabling return-path even for precopy to confirm success
- Can get failures of networking or storage before start
  - Especially where management reconfigs storage/networking prior to start after -S
- Hung guests
  - Debugging a hung guest after migrate is the hardest case
    - Missing interrupts
    - Offset timers (forward or backwards)
      - Some restart when time catches up
    - Corrupt RAM / storage



Are you sure...?

- Can the guest reboot after migrate?
  - Exercises different paths
  - Using source firmware on destination
    - Different versions
- Odd down-times
  - e.g. bad network reconfig loses packets for a few mins
  - Postcopy not really on-demand forwarding packets
- Performance same as pre-migration?
  - Transparent hugepage rebuild
  - Numa-placement
  - Disk cache flags
- Will it migrate again?



redhat.

THANK YOU

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos