# VM Live Migration Deep Dive in OCP-V

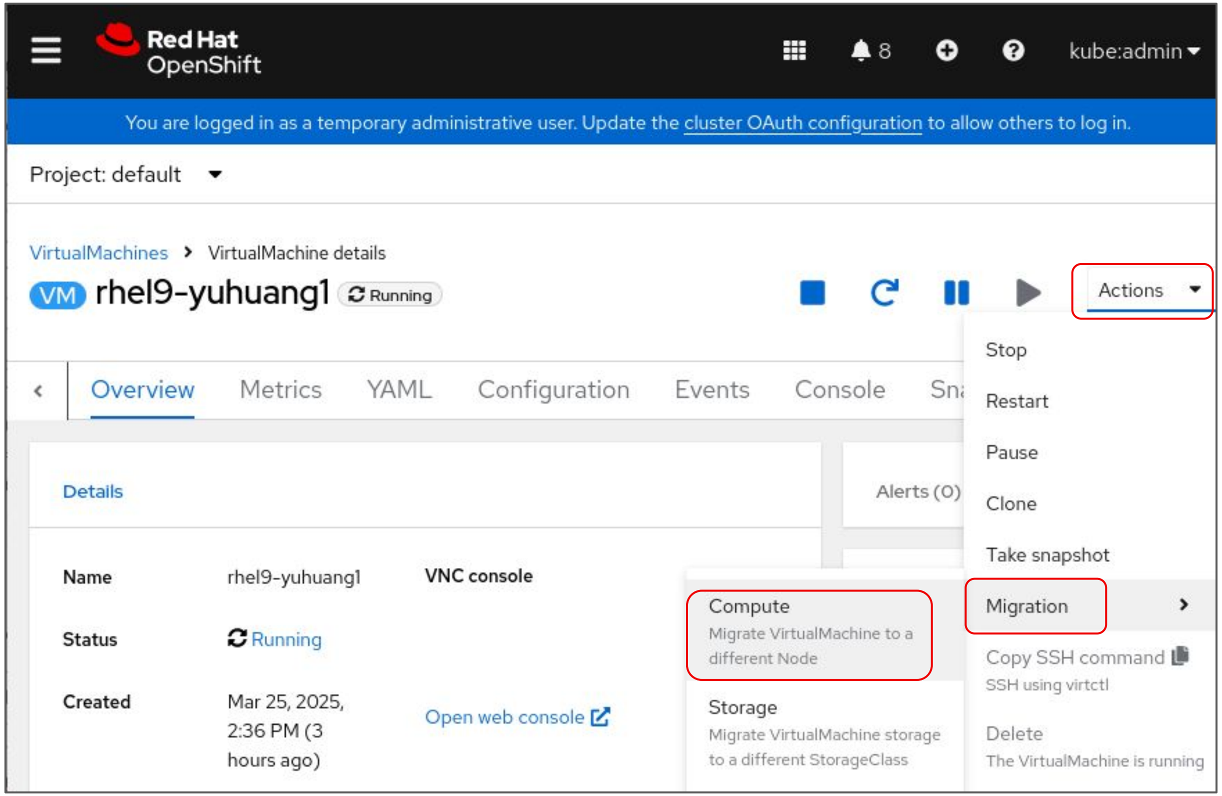
Yumei Huang
yuhuang@redhat.com

Xiaohui Li
xiaohli@redhat.com

## Agenda

- VM live migration in OCP-V
- Implementation in QEMU-KVM
- Live migration features in QEMU-KVM
- Performance comparison between VMWare and KVM

# VM live migration in OCP-V


Red Hat OpenShift

8

kube:admin

You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: default

VirtualMachines > VirtualMachine details

VM rhel9-yuhuang1 Running

Actions

Stop

Restart

Pause

Clone

Take snapshot

Migration

Copy SSH command
SSH using virtctl

Delete
The VirtualMachine is running

Compute
Migrate VirtualMachine to a different Node

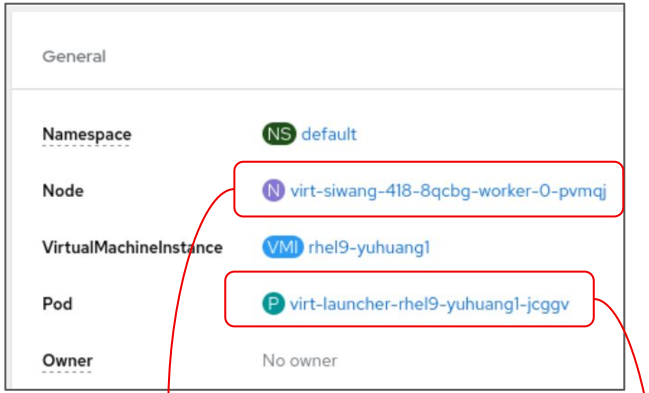
Storage
Migrate VirtualMachine storage to a different StorageClass

Overview | Metrics | YAML | Configuration | Events | Console | Sna

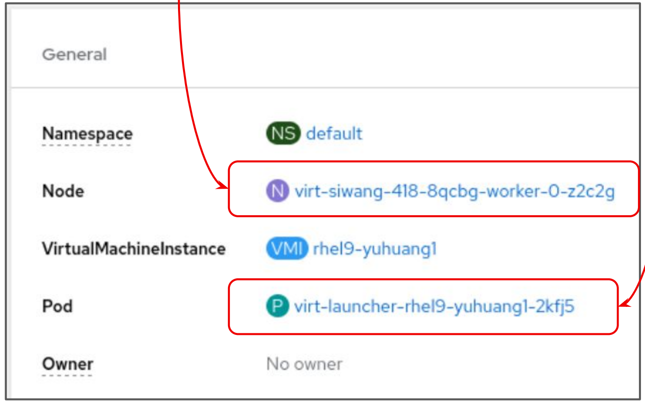
Details

Alerts (0)

| | | |
|---|---|---|
| Name | rhel9-yuhuang1 | VNC console |
| Status | Running | |
| Created | Mar 25, 2025, 2:36 PM (3 hours ago) | Open web console |

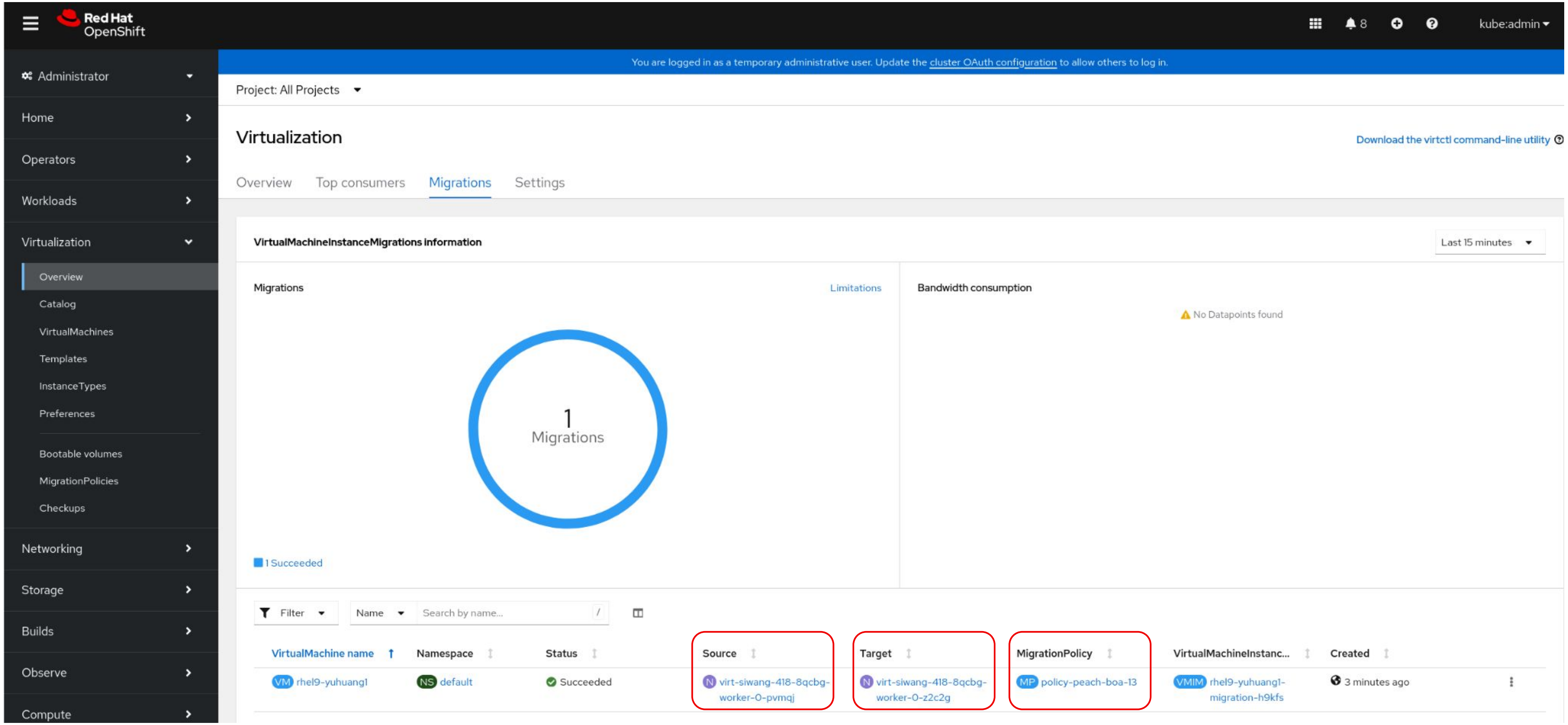

General

| | |
|---|---|
| Namespace | NS default |
| Node | N virt-siwang-418-8qcbg-worker-0-pvmqj |
| VirtualMachineInstance | VMI rhel9-yuhuang1 |
| Pod | P virt-launcher-rhel9-yuhuang1-jcggv |
| Owner | No owner |


General

| | |
|---|---|
| Namespace | NS default |
| Node | N virt-siwang-418-8qcbg-worker-0-z2c2g |
| VirtualMachineInstance | VMI rhel9-yuhuang1 |
| Pod | P virt-launcher-rhel9-yuhuang1-2kfj5 |
| Owner | No owner |

# VM live migration in OCP-V



You are logged in as a temporary administrative user. Update the cluster OAuth configuration to allow others to log in.

Project: All Projects

Virtualization

Download the virtctl command-line utility

Overview | Top consumers | Migrations | Settings

VirtualMachineInstanceMigrations information

Last 15 minutes

Migrations

Limitations

1
Migrations

🟦 1 Succeeded

Bandwidth consumption

No Datapoints found

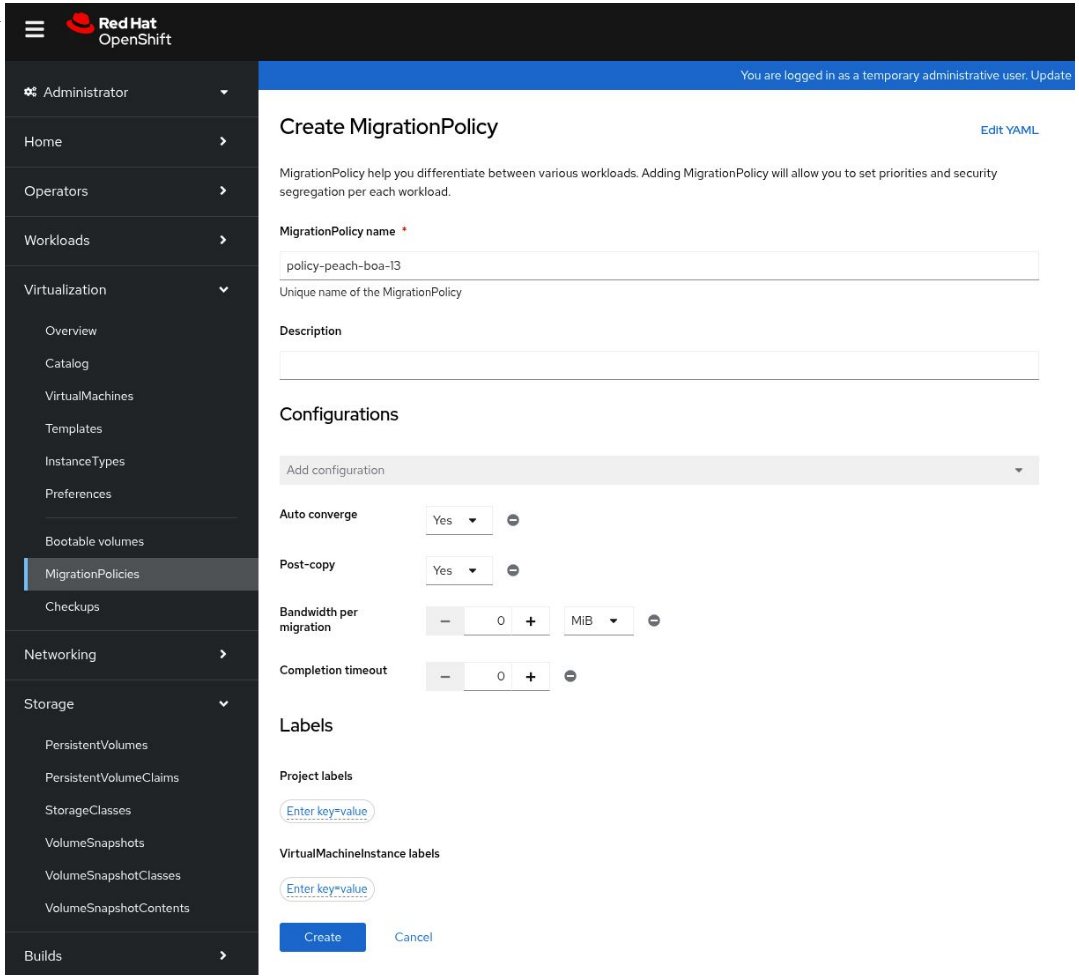
Filter | Name | Search by name...

| VirtualMachine name | Namespace | Status | Source | Target | MigrationPolicy | VirtualMachineInstanc... | Created |
|---|---|---|---|---|---|---|---|
| VM rhel9-yuhuang1 | NS default | ✓ Succeeded | N virt-siwang-418-8qcbg-worker-0-pvmqj | N virt-siwang-418-8qcbg-worker-0-z2c2g | MP policy-peach-boa-13 | VMIM rhel9-yuhuang1-migration-h9kfs | 3 minutes ago |

# VM live migration in OCP-V

Migration Policy

- Configurations
  - **Auto converge**
  - **Post-copy**
  - Bandwidth per migration
  - Completion timeout
- Labels
  - Define scope



You are logged in as a temporary administrative user. Update

Create MigrationPolicy

MigrationPolicy help you differentiate between various workloads. Adding MigrationPolicy will allow you to set priorities and security segregation per each workload.

MigrationPolicy name *

policy-peach-boa-13

Unique name of the MigrationPolicy

Description

Configurations

Auto converge: Yes

Post-copy: Yes

Bandwidth per migration: 0 MiB

Completion timeout: 0

Labels

Project labels

Enter key=value

VirtualMachineInstance labels

Enter key=value

## Implementation in QEMU-KVM

- Precopy migration
- Postcopy migration

## Precopy migration

With shared storage, only migrate VM states, including all the device states

- Iterative device
  - Send states over several iterations
  - E.g. RAM, VFIO device(Only Nvidia for now)
  - Dirty page tracking
- Non-iterative device
  - Send at once
  - E.g. network device, input device, virtio balloon device
  - vmsd (VMStateDescription)

# Precopy migration

- Stage 1 – Precopy phase
  - Source VM keeps running
  - Start dirty page tracking once migration starts
  - Send dirty pages iteratively
- Stage 2
  - Stop source VM once the expected downtime condition is met after last iteration
  - Send remaining dirty pages
  - Send non-iterative device states
- Stage 3
  - Resume VM on destination host

## Precopy migration

- Downtime
  - Default 300ms
  - Can set manually before migration
- Switchover condition
  - Pending_size < threshold_size
  - Threshold_size = bw * downtime_limit

- If dirty page rate >= bw, precopy never ends

```
(qemu) info migrate
globals:
store-global-state: on
only-migratable: off
send-configuration: on
send-section-footer: on
clear-bitmap-shift: 18
Migration status: active
total time: 11356 ms
expected downtime: 300 ms
setup: 7 ms
transferred ram: 1505645 kbytes
throughput: 1082.15 mbps
remaining ram: 5028756 kbytes
total ram: 8409672 kbytes
duplicate: 470584 pages
normal: 374645 pages
normal bytes: 1498580 kbytes
dirty sync count: 1
page size: 4 kbytes
multifd bytes: 0 kbytes
pages-per-second: 33030
precopy ram: 1505643 kbytes
```

```
(qemu) info migrate
globals:
store-global-state: on
only-migratable: off
send-configuration: on
send-section-footer: on
clear-bitmap-shift: 18
Migration status: completed
total time: 16673 ms
downtime: 51 ms
setup: 7 ms
transferred ram: 2211905 kbytes
throughput: 1087.24 mbps
remaining ram: 0 kbytes
total ram: 8409672 kbytes
duplicate: 1591045 pages
normal: 548339 pages
normal bytes: 2193356 kbytes
dirty sync count: 5
page size: 4 kbytes
multifd bytes: 0 kbytes
pages-per-second: 33544
precopy ram: 2181585 kbytes
downtime ram: 30039 kbytes
```

Migration statistics for RAM

## Postcopy migration

- Stage 1
  - Stop VM on source host, transfer device states(except RAM) to destination host
- Stage 2
  - Start VM on destination host
- Stage 3
  - Transfer RAM info from source to destination host

## Live migration features in QEMU-KVM

- generic migration
- postcopy
- postcopy-preempt
- multifd
- auto-converge
- zero-copy-send
- xbzrle
- tls encryption

# Live migration features in QEMU-KVM

## postcopy

postcopy enables VM starts running on the destination host as soon as possible, and the RAM from the source host is transferred into the destination over time

Advantage: 1) minimal downtime; 2) migration always converge with any workloads

## postcopy-preempt

postcopy-preempt is an optimization for postcopy migration, it allows urgent pages (those got page fault requested from destination QEMU explicitly) to be sent in a separate preempt channel, rather than queued in the background migration channel.

Advantage: besides postcopy 1) and 2), 3) reduce the latency of page faults, improve VM performance

Postcopy-preempt are recommended to use when migrate a huge VM on the stable environments

# Live migration features in QEMU-KVM

## multifd

Multiple File Descriptors enables parallel memory page transfer using multiple threads.

Advantage: 1) increase the CPU&bandwidth utilization to accelerate migration convergence

multifd is recommended to use on multi-core CPUs and high-bandwidth networks (>=10Gb/s)

## auto-converge

auto-converge provides a method by dynamically throttling the VM's CPU speed to reduce the rate of dirty page generation, ensuring that migration can eventually complete.

Advantage: migration can converge with high dirty page rate VMs

auto-converge is recommended to use for VMs with high dirty page rates, but no strict performance requirements

## Live migration features in QEMU-KVM

### zero-copy-send

zero-copy-send avoids multiple copies of data between the kernel buffer and the user space buffer.

zero-copy-send is used with multifd.

Advantage: 1) reduce CPU overhead and bandwidth consumption; 2) accelerate migration completion;

### xbzrle

xbzrle is a compression algorithm that reduces the amount of data to be migrated by compressing duplicate data in memory, significantly improving migration efficiency.

Advantage: handle large amounts of duplicate data or similar patterns in memory

## Live migration features in QEMU-KVM

### tls encryption

The migration I/O transport code has been enhanced to allow the use of TLS to provide both data encryption and authentication via x509 certificates

Advantage: protect guest memory and device state against modification or snooping by network based attackers while migrating

# Performance comparison between VMWare and KVM

## Test environment

VMware: ESXi 7.0.3 and RHEL 9.6 VM

KVM: RHEL 9.6 host (kernel-5.14.0-570.4.1.el9_6.x86_64 && qemu-kvm-9.1.0-15.el9.x86_64), RHEL 9.6 VM

Hosts: Milan (AMD), 1.5T memory, 256 cpu, support 200G network

RHEL 9.6 VM: 128 vcpu, 300G memory, and play a youtube video during migration

## Test scenarios

- **Scenario 1**: 500MB/s dirty page rate in VM, set migration bandwidth: 1280 MB/s
- **Scenario 2**:  1000MB/s dirty page rate in VM, set migration bandwidth: 5120 MB/s
- **Scenario 3**: 4000MB/s dirty page rate in VM, set migration bandwidth: 5120 MB/s

**Note**: when test multifd migration on KVM, set multifd threads to 5 in Scenario 2 && Scenario 3

## Test results: VMware VS KVM migration data

# Performance comparison between VMWare and KVM

## Migration total time comparison (s)

| | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| **VMware** | 380 | 117 | 222 |
| **KVM - postcopy-preempt** | 278 | 160 | 341 |
| **KVM - multifd** | 284 | 61 | 84 |
| **KVM - generic migration** | 294 | 153 | - |

Note: '–' represents generic migration is not suitable for Scenario 3, no test on it

# Performance comparison between VMWare and KVM

## Migration bandwidth comparison (MB/s)

| | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| **VMware** | 1140 ~ 1144 | 2634 ~ 3694 | 2252 ~ 2314 |
| **KVM - postcopy-preempt** | 700 ~ 900 | 1000 ~ 1200 | |
| **KVM - multifd** | 1350 ~ 1370 | 5340 ~ 5400 | |
| **KVM - generic migration** | 1340 ~1370 | 2062 ~ 2875 | - |

Note: '–' represents generic migration is not suitable for Scenario 3, no test on it

# Performance comparison between VMWare and KVM

## Migration downtime comparison (ms)

| | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| **VMware** | Unknow<br>“Stopping pre-copy: only xx pages left to send, which can be sent within the switchover time goal of **0.500 seconds**” in vmkernel.log | | |
| **KVM - postcopy-preempt** | 306 | 268 | 266 |
| **KVM - multifd** | 334 | 470 | 517 |
| **KVM - generic migration** | 384 | 631 | - |

Note: ‘–’ represents generic migration is not suitable for Scenario 3, no test on it

# Performance comparison between VMWare and KVM

## Video stuck

| | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| **VMware** | No | No | Yes (2s) |
| **KVM - postcopy-preempt** | Yes (5s) | Yes (5s) | Yes (5s) |
| **KVM - multifd** | No | Yes (3~4s) | Yes (3~4s) |
| **KVM - generic migration** | Yes (3~4s) | Yes (3~4s) | - |

RHEL-83883 – Video stuck after switchover phase when play one video during migration

- RCA: vcpu may have some execution delay, or network recovery process delay

Note: '–' represents generic migration is not suitable for Scenario 3, no test on it

## Performance comparison between VMWare and KVM

### Ping packets loss: transmitted/received

|  | Scenario 1 | Scenario 2 | Scenario 3 |
|---|---|---|---|
| **VMware** | 525/524 | 147/147 | 508/507 |
| **KVM - postcopy-preempt** | 355/355 | 359/359 | 402/402 |
| **KVM - multifd** | 311/310 | 88/87 | 104/103 |
| **KVM - generic migration** | 303/302 | 163/163 | - |

Note: '–' represents generic migration is not suitable for Scenario 3, no test on it

Thanks!