oVirt

High performance virtual machines

Sharon Gratch
Senior Software Engineer
oVirt team, RedHat
A Virtual Machine running with highest possible performance and with performance metrics as close to bare metal as possible.
Why Support High Performance VMs?

• Applications requirements (For example: SAP HANA).

• Until now, not a straightforward mission.

• Live migration was not supported for such VMs (due to pinning constraints).

• Implemented additional functionality was essential for improving VM performance (For example: using huge pages, IO and Emulator threads pinning, CPU cache layer 3 support, USB and other devices disabling, headless mode, enable multi queues per virtual Interface etc).
New High Performance VM Type

- Desktop VM type
- Server VM type
- High Performance VM type
By Choosing This New High Performance VM Type

**oVirt UI:**

<table>
<thead>
<tr>
<th>General</th>
<th>Cluster: cluster_42</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
<td>Template: Blank</td>
</tr>
<tr>
<td>Initial Run</td>
<td></td>
</tr>
<tr>
<td>Console</td>
<td>Instance Type: Custom</td>
</tr>
<tr>
<td>Host</td>
<td>Optimized for: High Performance</td>
</tr>
<tr>
<td>High Availability</td>
<td>Name: High Performance</td>
</tr>
<tr>
<td>Resource Allocation</td>
<td>Description:</td>
</tr>
<tr>
<td>Boot Options</td>
<td>Comment:</td>
</tr>
<tr>
<td>Random Generator</td>
<td>VM ID: 5534aceb-bd45-4968-b3c1-562e922c85c3</td>
</tr>
<tr>
<td>Custom Properties</td>
<td>Stateless, Start in Pause Mode, Delete Protection</td>
</tr>
</tbody>
</table>
By Choosing This New High Performance VM Type

1. Automatic settings - the VM will be automatically pre-configured with a set of recommended settings.

2. Manual settings - user will be recommended on a set of additional settings for optimal configuration.

- Non mandatory - the user can apply all, some, or none of these suggested automatic/manual configuration.
- Can be done for both new and existing VMs.
High Performance VM Configuration

flexibility    performance
High Performance -

Automatic Settings...
Configuration - Automatic Settings

**Console:**
- Enable Headless mode (no graphic consoles).
- Enable serial console.

**Disable the following VM devices:**
- qxl (video), spice, vnc (graphics) devices.
- USB devices.
- Sound Card device.
- Smart Card device.
- Memory Balloon device.
- Watchdog device.
- Tablet device.

**Networking:**
- Enable multi queues per virtual interface.

**Entropy:**
- Enable paravirtualized random number generator (virtio-rng device).

**CPU:**
- Set virtual CPU cache layer 3 (L3 cache).
Configuration - Automatic Settings

Storage / IO:
- Disk interface type is SCSI (virtio-scsi device).
- Storage allocation for disks is cloned/preallocated.
- Set mode to non-stateless.
- Enable IO threads (num of IO threads = 1).

Host pinning:
- Enable "Pass-Through Host CPU".
- Set the IO and emulator threads pinning topology.

Live Migration:
- Enable VM migration.

The user can apply all, some, or none of these suggested automatic settings.
VM Live Migration - Migration Mode

- Migration of High Performance VMs may result in VM’s performance decreasing.
- Therefore, default mode is to allow only manual migration (via UI or in case host manually switched to maintenance mode).
- Automatic migration (migration triggered by internal events like Load Balancing or high availability mechanism) is supported, but should be considered with caution.
VM Live Migration - Selecting Destination Host

- Source and destination hosts do not need to be identical.

- Source and destination hosts should be compatible in: number of CPUs (sockets, cores, threads), CPU pinning capacity, huge pages, memory, NUMA nodes topology, CPU model and hardware.

- Only a subset of Cluster’s hosts can be assigned to the VM.

- Not all hosts will supply the same level of performance results for the VM running on it, so the user is recommended to let oVirt choose the most preferable host.
High Performance -
Manual Settings...
Configuration - Manual Settings

- Set CPU pinning topology
- Declare vNUMA nodes and set NUMA pinning topology.
- Set memory backing with huge pages.
- Disable Kernel Same Page Merging (KSM).

The user can apply all, some, or none of these recommended settings.
Enable and define VM’s virtual huge pages:

- Set VM’s virtual huge pages size (KiB in oVirt).
  1G = 1048576, 2M = 2048.

- Verify that virtual huge pages fit physical host(s)’s huge pages:
  - Virtual huge pages size = physical huge pages size supported by the host.
  - Virtual huge pages size is set to the biggest size supported by the host.
  - There are enough free huge pages of that size on the host.

- Huge pages are preallocated when the VM starts to run.
Manual Settings - Huge Pages

Setting VM’s huge pages should consider:

✓ VM’s virtual memory size should fit into free huge pages of the given size on the host.

✓ VM’s NUMA node size should be a multiple of huge pages selected size.

Limitations:

! No memory hotplug/unplug.

! Memory resource is limited for a host.
High Performance - Pinning issues...
The first two pCPUs of one of host's pinned NUMA nodes will be automatically pinned to IO+emulator threads. Usually pCPUs: 0,1.

- If all vCPUs fit into one of the host's NUMA nodes
  - First 2 pCPUs are pinned.

- If VM spans more than one NUMA node
  - First 2 pCPUs of the most pinned NUMA node are pinned (leaves the next pinned NUMA node(s) for vCPU pinning only).
Example Use Case: Host With 2 NUMA Nodes

Host

NUMA Node 0
- 80 GiB
- CPUs 0, 1, 2, 3, 4

NUMA Node 1
- 32 GiB
- CPUs 5, 6
Example Use Case - VM With 2 vNUMA Nodes, IO Threads Enabled

Virtual Machine

- vNUMA Node 0
  - 30 GiB
  - CPUs 0, 1, 2

- vNUMA Node 1
  - 30 GiB
  - CPUs 3, 4

Host

- NUMA Node 0
  - 80 GiB
  - CPUs 0, 1, 2, 3, 4

- NUMA Node 1
  - 32 GiB
  - CPUs 5, 6
Example Use Case - Enable NUMA Pinning

- Enable NUMA Pinning

Host

NUMA Node 0
80 GiB
CPUs 0, 1, 2, 3, 4

NUMA Node 1
32 GiB
CPUs 5, 6

Virtual Machine

vNUMA Node 0
30 GiB
CPUs 0, 1, 2

vNUMA Node 1
30 GiB
CPUs 3, 4

IO+ emulator
Example Use Case - Enable IO + Emulator + CPU Pinning

### Host
- NUMA Node 0: 80 GiB, CPUs 0, 1, 2, 3, 4
- NUMA Node 1: 32 GiB, CPUs 5, 6

### vNUMA Node 0
- 30 GiB, CPUs 0, 1, 2

### vNUMA Node 1
- 30 GiB, CPUs 3, 4

### Virtual Machine
- IO+ emulator
- vNUMA Node 0: 30 GiB, CPUs 0, 1, 2
- vNUMA Node 1: 30 GiB, CPUs 3, 4
Future Improvements

• Set vCPU pinning topology and vNUMA pinning topology automatically.

• Set virtual huge pages automatically.

• Affinity rules management for managing groups of High Performance VMs and physical hosts.

• Continue tuning the High Performance VM solution according to future benchmarks.
THANK YOU!

https://github.com/sgratch
sgratch@redhat.com