Create a KVM backup in IBM PureFlex System

KVM virtual machines disaster recovery

Pradeep K. Surisetty (pradeepkumars@in.ibm.com)  
KVM Developer  
IBM

Pradipta Kumar (bpradipt@in.ibm.com)  
KVM Virtualization and Cloud Management Architect  
IBM

Tony Gargya (gargya@de.ibm.com)  
KVM Systems Management Architect  
IBM

IBM® PureFlex™ System with integrated network and storage virtualization and an open hypervisor provide an open and cost-effective solution to customers. This article shows how to create a backup of KVM (Kernel-based Virtual Machine) virtual machines in a PureFlex System environment.

As virtual environments grow larger and the applications running in them become more important to business success, demand for quick, efficient backup of virtual machines (VMs) residing in Kernel-based Virtual Machine (KVM) data stores is becoming critical. This article addresses the problem of creating consistent backups using IBM Storewize® V7000 storage and IBM FlashCopy® functionality from KVM VMs in an IBM PureFlex System chassis and recovering them on another (possibly remote) PureFlex System chassis.

Configuration and setup

The environment used as part of this setup consists of the following components:

- Red Hat Enterprise Linux® (RHEL) 6.3
- A PureFlex System chassis with IBM System x® IT elements (ITEs) managed by IBM Flex System™ Manager (FSM)
- Storage area network Storewize V7000 storage logical unit numbers connected to servers by Fibre Channel
This article assumes that PureFlex System and FSM with Storewize V7000 storage have been configured. RHEL version 6.2 or later must already be installed and configured properly on the System x ITE. See [Resources](#) for links to more information.

### High-level configuration steps

The steps that follow represent the logical configuration phases for the failover cluster:

1. Install the latest PureFlex System-supported RHEL 6.3 operating system on a System x ITE node in PureFlex System.
2. Create VMs using the management layer (FSM in PureFlex System).
3. Suspend the critical VM.
4. Take a snapshot of the respective volume of VM Storewize V7000 storage using FlashCopy.
5. Create a backup of the `libvirt` XML file of the VM.
6. Resume the VM.
7. Map a volume to the backup server in Storewize V7000 storage.
8. Copy the backup volume and the corresponding XML files to a media tape device.
9. Restore the system using the backed-up volume and XML files.
10. Run an inventory from FSM on the new system.

### Backup and recovery of VMs

Backing up from KVM VMs in PureFlex System using Storewize V7000 storage and FlashCopy provides a complete data-protection solution for the KVM environment.

### Suspend the critical VM

The `virsh` tool is built on top of the `libvirt` management application programming interface to manage guest domains. Listing 1 shows how to suspend a VM before taking snapshot of the VM volume.
Listing 1. Suspending a VM before taking a snapshot of it

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>win2008-r2-64</td>
<td>running</td>
</tr>
<tr>
<td>31</td>
<td>vm1</td>
<td>running</td>
</tr>
</tbody>
</table>

virsh # list

Id    Name                           State
----------------------------------------------------
29    win2008-r2-64               running
31    vm1                               running

virsh # suspend vm1

virsh # list

Id    Name                           State
----------------------------------------------------
29    win2008-r2-64               running
31    vm1                               paused

Take a snapshot of the volume with FlashCopy

FlashCopy is the point-in-time copy capability of the Storewize V7000 disk. It is used to create instant, complete, and consistent copy from a source volume to a target volume. Often, this functionality is called *time-zero copy*, *point-in-time copy*, or *snapshot copy*. FlashCopy is part of Copy Services; Figure 1 shows how to take a snapshot of a volume from FlashCopy.

Figure 1. Creating a snapshot of a volume from FlashCopy

Back up the VM and XML files

Back up the corresponding *libvirt* XML file, and then resume the VM, as shown in Listing 2.
Listing 2. Resuming the VM

# virsh dumpxml vm1 > /tmp/vm1-snap1-xml
virsh # list

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>win2008-r2-64</td>
<td>running</td>
</tr>
<tr>
<td>31</td>
<td>vm1</td>
<td>paused</td>
</tr>
</tbody>
</table>

virsh # resume vm1

virsh # list

<table>
<thead>
<tr>
<th>Id</th>
<th>Name</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>win2008-r2-64</td>
<td>running</td>
</tr>
<tr>
<td>31</td>
<td>vm1</td>
<td>running</td>
</tr>
</tbody>
</table>

Map the new volume to the backup server

If a backup server is available, map the backed-up volume to that server (see Figure 2).

Figure 2. Mapping the volume

When the volume has been mapped, the new volume will be listed on the backup server, as shown in Listing 3.
Listing 3. Mapping the volume to the backup server

```
[root@llvm11 ~]# multipath -ll
mpathe (3600507680280865df80000000000042) dm-7 IBM,2145
  size=5.9G features='1 queue_if_no_path' hwhandler='0' wp=rw
  |+- policy='round-robin 0' prio=50 status=active
  | `- 0:0:1:5  sdq 65:0  active ready running
  `-+- policy='round-robin 0' prio=10 status=enabled
     `- 0:0:0:5  sdf 8:80  active ready running
mpathd (3600507680280865df800000000000017) dm-3 IBM,2145
  size=10G features='1 queue_if_no_path' hwhandler='0' wp=rw
  |+- policy='round-robin 0' prio=50 status=active
  | `- 0:0:0:3  sdd 8:48  active ready running
  `-+- policy='round-robin 0' prio=10 status=enabled
     `- 0:0:1:3  sdo 8:224 active ready running
mpathc (3600507680280865df80000000000004e) dm-0 IBM,2145
  size=12G features='1 queue_if_no_path' hwhandler='0' wp=rw
  |+- policy='round-robin 0' prio=50 status=active
  | `- 0:0:0:0  sda 8:0   active ready running
  `-+- policy='round-robin 0' prio=10 status=enabled
     `- 0:0:1:0  sdl 8:176 active ready running
```

Backed-up volumes of VMs and respective XML files can be stored on backup servers or tape media.

Recovery during a disaster

Map backed-up volumes to the new System x ITE on the PureFlex System chassis. Edit the XML files and define them with `libvirt`, as shown in Listing 4.

Listing 4. Edit VM1 with a new volume path

```
<disk type='block' device='disk'>
  <driver name='qemu' type='raw' cache='none' io='native'/>
  <source dev='/dev/disk/by-id/wwn-0x600507680280865df800000000000017'/>
  <target dev='vda' bus='virtio'/>
</disk>
```

```
virsh # virsh define /etc/libvirt/vm1-snap1.xml
virsh # start vm1
```

Run the inventory

When the VM image has been recovered on a different host, run an inventory on the new host from FSM (see Figure 3).
Future work

Suspending a VM with **libvirt** is crash consistent because the VM is suspended immediately and does not get a chance to flush the contents of its memory to the disks. When this image is resumed on another host, file systems will be checked for errors (**fsck**) and repaired. Because the content of memory is not present in the copy, there is a chance of data loss.

**Freeze and thaw a VM with guest agents**

An upcoming feature called QEMU GuestAgent makes it possible to put the file systems of VMs in a consistent state before taking the snapshot — in other words, you can implement a freeze and thaw a VM with guest agents.

A guest agent is a daemon program running inside the domain that is designed to help management applications execute functions that need the assistance of the guest OS. For example, freezing and thawing filesystems or entering a suspend mode.

Using QEMU GuestAgent avoids inconsistent file systems in the snapshot, which means that you don't need to check file systems while resuming the VM, and there is no chance of data loss. When this QEMU GuestAgent feature is available in enterprise hypervisors, you can use it with the above snapshot procedure to obtain consistent, loss-free backup and restore capability.

To configure guest agents:

1. Install a QEMU GuestAgent on the VM.
2. Start the VM with virtio-serial or an isa-serial channel.
3. Start the QEMU GuestAgent:
   
   ```bash
   "qemu-ga -m virtio-serial -p /dev/virtio-ports/org.qemu.guest_agent.0"
   ```
4. Start the QEMU Management Protocol from a different terminal:
   
   ```bash
   socat unix-connect:/tmp/qga.sock readline
   ```
5. Freeze and thaw the VM by running the following commands:

```json
{ "execute": "guest-info" }
{ "execute": "guest-fsfreeze-freeze" }
{ "execute": "guest-fsfreeze-thaw" }
```
Resources

- For more information about the technology discussed in this article, see the IBM Redbook, IBM PureFlex System and IBM Flex System Products and Technology.
- Read the IBM Redbooks product guide, IBM Flex System Manager, for more information about FSM.
- Learn more about the QEMU GuestAgent.
- In the cloud developer resources on developerWorks, discover and share knowledge and experience of application and services developers building their projects for cloud deployment.
- Follow developerWorks on Twitter.
- Watch developerWorks demos ranging from product installation and setup for beginners to advanced functionality for experienced developers.
- Get involved in the developerWorks community. Connect with other developerWorks users while exploring the developer-driven blogs, forums, groups, and wikis.
- Join a developerWorks community cloud computing group.
- Read all the great cloud blogs on developerWorks.
About the authors

Pradeep K. Surisetty

Pradeep K Surisetty is a KVM developer in the Linux Technology Center, IBM Systems and Technology Group Labs, India. He has 6 years of experience in virtualization and cloud development. He helps IBM client-facing teams with designing of KVM-based solutions.

Pradipta Kumar

Pradipta Kumar Banerjee is a KVM virtualization and cloud management architect in the IBM Linux Technology Center, IBM Systems and Technology Group Labs, India. He is an IBM Master Inventor and works closely with clients to design virtualized data centers and cloud infrastructure for their work loads.

Tony Gargya

Tony Gargya is a KVM systems management architect in the Linux Technology Centre, Germany. His focus is primarily on open source-based projects and their integration with commercial products. He works closely with numerous clients to help them to manage their data centres and has spoken often both at IBM and Linux foundation-sponsored conferences.

© Copyright IBM Corporation 2013

Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)