# Foreman - Bug #2374

## Libvirt host creation fails with LVM storage pool

04/04/2013 04:20 PM - Chris Barbour

| | | | |
|---|---|---|---|
| **Status:** | Closed | | |
| **Priority:** | Normal | | |
| **Assignee:** | Lukas Zapletal | | |
| **Category:** | Host creation | | |
| **Target version:** | 1.4.0 | | |
| **Difficulty:** | | **Fixed in Releases:** | |
| **Triaged:** | | **Found in Releases:** | |
| **Bugzilla link:** | | **Red Hat JIRA:** | |
| **Pull request:** | | | |

### Description

I'm unable to provision a libvirt host when the storage backend is an LVM pool. Host creation appears to fail when Foreman requests LV parameters that LVM2 is unable to satisfy.

Here's the error from the libvirtd:

```
2013-04-04 00:15:41.898+0000: 3686: error : virCommandWait:2345 : internal error Child process (/sbin/lvcreate --name
test1.example.com-disk1 -L 0K --v
irtualsize 41943040K sec) unexpected exit status 5:   Unable to create new logical volume with no extents
```

It's not clear if this is truly a Foreman bug, a LVM bug, or libvirt issue. However, the Following change in Foreman did work around the problem:

```
--- /usr/share/foreman/lib/foreman/model/libvirt.rb.orig    2013-04-04 12:39:13.000000000 -0700
+++ /usr/share/foreman/lib/foreman/model/libvirt.rb    2013-04-04 12:39:24.000000000 -0700
@@ -136,7 +136,7 @@
        vols = []
        (volumes = args[:volumes]).each do |vol|
          vol.name       = "#{args[:prefix]}-disk#{volumes.index(vol)+1}"
-         vol.allocation = "0K"
+         vol.allocation = "1M"
          vol.save
          vols << vol
        end
```

I can confirm that provisioning using a directory based storage pool works normally in this environment.

Software versions:

libvirt-0.10.2-18.el6.x86_64
lvm2-2.02.87-6.el6.x86_64
foreman-1.1RC5-2.el6.noarch

OS Release is Scientific Linux 6.2

### Associated revisions

**Revision 41fb208d - 12/18/2013 10:23 AM - Lukas Zapletal**

fixes #2374 - added allocation option to libvirt VM

### History

**#1 - 04/04/2013 07:59 PM - Chris Barbour**

Actually... Thin provisioning seems to be pretty broken on this version of LVM. After writing anything past the allocated size of the LV, the entire volume appears to go offline. LVM then starts throwing read errors for the volume

```
# lvdisplay sec/test
```

```
  /dev/sec/test: read failed after 0 of 4096 at 524222464: Input/output error
  /dev/sec/test: read failed after 0 of 4096 at 524279808: Input/output error
  /dev/sec/test: read failed after 0 of 4096 at 0: Input/output error
  /dev/sec/test: read failed after 0 of 4096 at 4096: Input/output error
  --- Logical volume ---
  LV Name                /dev/sec/test
  VG Name                sec
  LV UUID                qvQpd6-7CC6-CVuc-7t50-6kLz-wMz3-1fEmD1
  LV Write Access        read/write
  LV snapshot status     INACTIVE destination for /dev/sec/test_vorigin
  LV Status              available
  # open                 0
  LV Size                500.00 MiB
  Current LE             125
  COW-table size         100.00 MiB
  COW-table LE           25
  Snapshot chunk size    4.00 KiB
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:83
```

(This was a test LV I created on my own. Foreman/FOG/libvirt LVs produce the same results.)

Steps to reproduce this issue:

```
# /sbin/lvcreate --name test -L 100M --virtualsize 500M sec
# mkdir /mnt/test
# mount /dev/mapper/sec-test /mnt/test
# dd if=/dev/zero of=/mnt/test/zero
```

It would be nice to have a way to disable thin provisioning when building the guest, as a temporary workaround for this issue. In general, I'm concerned about this version of LVM's approach to thin provisioning, since it seems to be snapshot based and fairly complex.

#### #2 - 04/04/2013 08:30 PM - Chris Barbour

This seems to be expected behavior for LVM.

What's the use case for --virtualsize?

http://www.globallinuxsecurity.pro/recovering-an-overflowed-lvm-volume-configured-with-virtualsize/

#### #3 - 04/08/2013 10:51 AM - Mark Heily

I can confirm that the LVM volumes provisioned with Foreman 1.1 start generating "Input/output error" in the logs as soon as the OS installer tries to write data to the disk.  This causes the installer to fail.

#### #4 - 04/08/2013 05:29 PM - Chris Barbour

Thanks for confirming, Mark.

I see 2 issues so far:

1. Foreman is unable to provision volumes in a LVM pool, due to hard coded allocation size.
2. Thin provisioned LVM volumes overflow, causing Input/output errors. This situation is difficult to recover from on some platforms.

My personal desire would be to (have an option to) disable LVM thin provisioning on affected platforms.

It appears that more robust LVM thin provisioning is on the way for RHEL7: http://lxadm.wordpress.com/2012/10/17/lvm-thin-provisioning/

#### #5 - 04/24/2013 06:18 PM - Chris Barbour

Alright,

I've done some additional digging. The virtualisize thing is actually a known limitation with the current versions of LVM, and already discussed in the libvirt documentation. The short version is that sparse allocated volumes require some external help to extend the allocated size of the volume.

http://libvirt.org/formatstorage.html#StorageVolFirst

I haven't seen a lot of documentation on how to use dmeventd to manage thin provisioning, but I'm sure I could work through it.

There is a patch to libvirt which resolves the vol.allocation = 0K issue. I think this is a better solution than modifying the default allocation size in Foreman, Modifying the allocation size will impact other pool types as well. It is important to be aware that Foreman is unable to provision volumes in LVM pools with releases of libvirt prior to libvirt-0.10.2.1.

I think a more ideal solution to this problem is to add the option to disable thin provisioning of LVs. Doing so would require a patch to both Foreman and FOG. I can file a separate bug for that.

**#6 - 04/24/2013 06:24 PM - Chris Barbour**

*- File fog-libvirt_full-volume-allocation.patch added*

I've attached a FOG patch to this comment, for those who use LVM and don't want the overhead or complexity of thin provisioning. This patch will disable thin provisioning for ALL new libvirt provisioned guests, not just those using LVM pools.

The patch simply removes the allocation size from the volume XML file. This causes libvirt to thick provision storage for the new VM.

To apply, CD into the root directory of your FOG gem (Example: cd /usr/lib/ruby/gems/1.8/gems/fog-1.9.0/ ) and apply with patch < fog-libvirt_full-volume-allocation.patch

**#7 - 10/21/2013 03:47 PM - Matthias Saou**

I've bumped into this issue too, with foreman 1.3 (the latest, currently). I'm also using libvirt, with LVM and no thin provisionning.

One strange thing is that when using foreman, my resulting libvirt/qemu configuration file's disk section is a bit different.

The original, when manually creating with virt-install :

```
    <disk type='block' device='disk'>
      <driver name='qemu' type='raw' cache='none' io='native'/>
      <source dev='/dev/vg0/test.example.com'/>
      <target dev='vda' bus='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
    </disk>
```

And the one foreman creates :

```
    <disk type='file' device='disk'>
      <driver name='qemu' type='raw'/>
      <source file='/dev/vg0/test.example.com-disk1'/>
      <target dev='vda' bus='virtio'/>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
    </disk>
```

The type/source changes from block/dev to file/file. To me, the block/dev seems more correct for an LV. Making the change in Chris's patch worked around the issue for me, though this type difference is still there.

**#8 - 11/26/2013 04:52 PM - Lukas Zapletal**

*- Description updated*

*- Category set to Host creation*

*- Assignee set to Lukas Zapletal*

*- Target version set to 1.10.0*

Hello,

I am able to reproduce this. I think the simpliest workaround would be to allow change of allocation via Foreman GUI. By default we can leave it on zero, but once this can be set to the same size as its size of the volume, libvirt (at least in RHEL 6.5+) does not thin provisioning and everything works fine.

Example, this host was created when vol.allocation = "0G"

```
  --- Logical volume ---
  LV Path                /dev/vg_data/cs.home.lan-disk1
  LV Name                cs.home.lan-disk1
  VG Name                vg_data
  LV UUID                5tWVBL-etH7-m2x9-Fe4o-l1ek-fQ6N-4jeG2W
  LV Write Access        read/write
  LV Creation host, time ox.home.lan, 2013-11-26 17:23:50 +0100
  LV snapshot status     active destination for cs.home.lan-disk1_vorigin
  LV Status              available
  # open                 1
  LV Size                10.00 GiB
  Current LE             2560
  COW-table size         4.00 MiB
  COW-table LE           1
  Allocated to snapshot  0.00%
  Snapshot chunk size    4.00 KiB
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0
```

And this host was created with vol.allocation = "10G" and size of 10G:

```
  --- Logical volume ---
  LV Path                /dev/vg_data/el.home.lan-disk1
  LV Name                el.home.lan-disk1
  VG Name                vg_data
  LV UUID                OkaJpn-shFB-K3V7-926E-xpid-8ADD-kOdD4y
  LV Write Access        read/write
  LV Creation host, time ox.home.lan, 2013-11-26 17:47:25 +0100
  LV Status              available
  # open                 1
  LV Size                10.00 GiB
  Current LE             2560
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0
```

I will prepare a patch that will add "allocation" to the libvirt VM form. Maybe we can talk also about default value, because from what I have read and seen, preallocation makes HUGE difference for COW2 images as well as for LVM. We should consider to set preallocation to the same size by default.

**#9 - 12/03/2013 02:33 PM - Lukas Zapletal**

*- Status changed from New to Assigned*

I have a patch that adds allocation field in the UI.

**#10 - 12/04/2013 01:11 PM - Dominic Cleal**

*- Target version changed from 1.10.0 to 1.9.3*

**#11 - 12/06/2013 09:20 AM - Lukas Zapletal**

*- Status changed from Assigned to Ready For Testing*

https://github.com/theforeman/foreman/pull/1073

**#12 - 12/18/2013 10:17 AM - Dominic Cleal**

*- translation missing: en.field_release set to 2*

**#13 - 12/18/2013 05:31 PM - Lukas Zapletal**

*- Status changed from Ready For Testing to Closed*

*- % Done changed from 0 to 100*

Applied in changeset 41fb208d55495abc0cc8d0a8471bc33dfc991c78.

## Files

| | | | |
|---|---|---|---|
| libvirt.rb.lvm2.diff | 451 Bytes | 04/04/2013 | Chris Barbour |
| fog-libvirt_full-volume-allocation.patch | 548 Bytes | 04/24/2013 | Chris Barbour |