rbd - Bug #7282

Unresponsive rbd-backed Qemu domain causes libvirtd to stall on all connections

01/31/2014 09:47 AM - Florian Haas

Status:	Resolved	**Start date:**	01/31/2014
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	17.00 hours
Source:	other	**Reviewed:**	
Tags:		**Affected Versions:**	
Backport:		**ceph-qa-suite:**	
Regression:	No	**Pull request ID:**	
Severity:	1 - critical		

Description

This isn't fully confirmed yet, because we haven't found a reliable way to reproduce. In short, it seems that if you have either an rbd-backed libvirt storage pool or an rbd-backed Qemu/KVM domain, and some borked objects in your RBD, then something seems to spin inside libvirtd, blocking **all** libvirtd connections on that host.

Which would mean that a handful of bad RADOS objects could take down an entire virtualization host potentially running scores or hundreds of guests, which would be sufficiently painful to merit reporting an issue even if not fully confirmed.

What's extremely painful is that the issue produces zero logs. Even strace is unhelpful.

Platform: Ubuntu 12.04.3
Ceph client tools including librados and librbd: Dumpling (0.67.5-1precise)
Libvirt: 1.0.2 (1.0.2-0ubuntu11.13.04.5~cloud1) from Ubuntu Cloud Archive from Grizzly (can't run 1.1.1, being bitten by https://bugzilla.redhat.com/show_bug.cgi?id=1060248)

Steps to (possibly) reproduce:

- Create a problematic state in your cluster, for example, have some unfound objects in a pool used by your guests' RBD volumes
- Create an rbd storage pool XML definition
- Fire up libvirtd
- Run virsh
    - observe that list quickly responds with a list of currently running VMs
    - run pool-define <path-to-definition.xml>
    - observe that list is still snappy
    - run pool-start <name-of-pool>
    - try list again. If you hit the problem, it will block indefinitely.

The problem is not with the pool though. Remove the pool, try the same with a domain definition using an rbd backed drive. Once you start a guest backed by a problematic RBD, virsh starts blocking.

Of course, that also means once you've hit the problem, a reboot is the only way out (because the only way to kill the pool or domain would be through a libvirtd socket again).

Because it's practically impossible for a user to enumerate all the RADOS objects in a PG, it's almost infeasible to determine ahead of time which RBDs might be affected by something like an unfound object, as the latter are always reported by PG.

All this makes the issue possibly extremely tricky to reproduce, but when it hits, its impact is frightening. I'll leave this at sev 3 since it's not reliably confirmed.

History

#1 - 01/31/2014 09:51 AM - Florian Haas

- Description updated

#2 - 02/01/2014 11:40 AM - Wido den Hollander

I'm aware of this. It's not only with RBD though, the same happens with libvirt when you use NFS and a export is dead.

We should have some timeouts in libvirt / librados for this, but I never got to actually diving into this.

#3 - 02/02/2014 07:53 AM - Florian Haas

Hey Wido,

thanks for responding to this, and doubly for doing so on a weekend. Much appreciated.

> I'm aware of this.

I'm glad that **you** are aware of this, because when asked in the channel, no-one was. :) I'll make a note to take any libvirt/rbd issues to the list straight away, in case I don't find you in the channel.

> It's not only with RBD though, the same happens with libvirt when you use NFS and a export is dead.

OK, I did a comparison with NFS behavior.

- NFS-backed storage pool
- 5 VMs that use images from the NFS mount managed by that pool
- 1 VM that does not

Now, when I kill my NFS server,

- NFS-backed domains obviously freeze, but
- virsh still responds, so the socket is not completely dead
- list, domstate and pool-list still work (and I assume the same is true for other virsh commands)

I can also still run virsh start for my non-NFS-backed domain.

If, while the NFS mount is down, I attempt to **start** an NFS-backed domain, then virsh does indeed freeze, and other virsh sessions also fail to perform domstate and list.

That kinda-sorta adds up. As far as I can tell, Qemu does not treat NFS-backed devices specially in any way, instead treating them just like regular local files. So when the NFS server goes away, all I/O freezes, and nothing that touches the file can complete any operations. So with qemu stuck, the virsh call gets stuck, I am assuming libvirtd grabs some sort of global lock when starting domains, and then of course there is also no way to determine the domain state until the NFS server returns.

All that is awful, but it at least seems to have no impact on non-NFS backed VMs, so while your NFS server is down, your NFS-backed VMs are dead, and virsh list can't complete because the domstate operation is blocked on some VMs.

But it looks to me for RBD it's actually much, much worse:

- The equivalent of an NFS server being down, in RBD terms, would be no MONs being available. We did not see that. Instead, ceph -s and rados calls did complete, meaning the MONs were perfectly responsive.
- Once you start the RBD-backed VMs with your Ceph cluster being somewhat faulty, all of libvirtd comes to a grinding halt. This affects all virtualization components on the host, not just the RBD backed domains and pools.
- And worst of all, this seems to happen in an error state that shouldn't at all affect the entire cluster. The MONs are there, most PGs and objects are fine, there are only a few unfound objects that should really just affect the RBDs that they are part of.

When you said you "were aware of this", did you mean you realized it was this bad, or were you under the assumption that the "dead NFS server" equivalent behavior would occur only in case of all MONs being down?

> We should have some timeouts in libvirt / librados for this, but I never got to actually diving into this.

Yep, particularly since unlike with NFS, Libvirt does treat rbd devices specially, and because the qemu rbd driver is already using async I/O all over the place.

Also, with NFS, I can think of a workaround, which is to mount NFS with the intr option from the fstab, and then relying on Libvirt's network pool shortcut that makes sure not to touch an already existing NFS mount. But for RBD I can think of no such workaround.

Additional thoughts on this?

#4 - 02/03/2014 12:32 AM - Florian Haas

- Severity changed from 3 - minor to 1 - critical

With Wido having confirmed that the issue exists, I'm bumping the severity to critical.

Wido, if you could offer your thoughts on whether the issue is in libvirtd itself, in the Qemu rbd integration, in the libvirt rbd integration, or in librados/librbd, that would be great. Thanks!

#5 - 02/03/2014 12:56 AM - Wido den Hollander

I'm going to try to figure this out, I'm setting up a test env right now.

I think it's libvirt which can't handle stalling Qemu processes. I saw this on a NFS-only setup as well. For some reason the Qemu monitor wouldn't respond to libvirt which then causes libvirt to stall.

#6 - 02/03/2014 01:42 AM - Florian Haas

Right, but why should the qemu process stall if all the MONs are available, and only a few RBDs are shot? (if they are at all, mind you; it's pretty tough to figure that out from the Ceph health state alone)

#7 - 02/03/2014 01:48 AM - Wido den Hollander

Florian Haas wrote:

> Right, but why should the qemu process stall if all the MONs are available, and only a few RBDs are shot? (if they are at all, mind you; it's pretty tough to figure that out from the Ceph health state alone)

The problem is librados here. If a PG is not active a rados_read() or rados_write() will block until the PG becomes active again.

That causes Qemu to lock up. Some time ago I created #6507 about this.

#8 - 02/03/2014 06:00 AM - Wido den Hollander

So I've played around with this today and I can't fully reproduce it.

I have a machine running with libvirt 1.1.1-0ubuntu8~cloud2 (Havana) and one RBD storage pool.

Using ip6tables (cluster runs IPv6) I dropped all traffic to the OSDs:

```
for port in {6800..6808}; do sudo ip6tables -A OUTPUT -p tcp --dport $port -j DROP; done
```

A pool refresh blocks, just like a "sync" in a guest.

However, "virsh list" keeps working for me. It never blocked.

I could still query for VNC ports and such, so that went just fine.

After flushing ip6tables it took about 20 seconds for the "sync" to finish inside the guest and the pool refresh also completed shortly after that.

So while I've seen this blocking behavior in the past, I can't reproduce it with libvirt 1.1.1

Nevertheless, it would be great if librados could be configured to not block for ever and returns with a error code.

#9 - 02/03/2014 10:55 AM - Florian Haas

Thanks Wido. As explained in the original description we went off 1.1.1 here due to a SIGABRT bug.

Two questions:

- Did you only test with an RBD storage **pool**, or do you also have **domains** running off of RBD? If the latter, do you see virsh start blocking when you run it after you've shot your Ceph cluster?
- Any chance you could downgrade your test env to libvirt 1.0.2 (from Ubuntu Cloud Archive for grizzly) and see if you can reproduce the error then? Because if you can't, then you're seeing a different issue than we do.

#10 - 02/03/2014 12:46 PM - Wido den Hollander

Florian Haas wrote:

> Thanks Wido. As explained in the original description we went off 1.1.1 here due to a SIGABRT bug.
>
> Two questions:
>
> - Did you only test with an RBD storage **pool**, or do you also have **domains** running off of RBD? If the latter, do you see virsh start blocking when you run it after you've shot your Ceph cluster?

Yes, I had 5 domains running of RBD. I issued a "sync" inside one of the domains and it blocked for ever.

> - Any chance you could downgrade your test env to libvirt 1.0.2 (from Ubuntu Cloud Archive for grizzly) and see if you can reproduce the error then? Because if you can't, then you're seeing a different issue than we do.

I'll see if I can do that. It's just a test machine of me. Might have time for this tomorrow, otherwise it will be later this week.

#11 - 02/03/2014 12:59 PM - Florian Haas

Wido den Hollander wrote:

> > - Did you only test with an RBD storage **pool**, or do you also have **domains** running off of RBD? If the latter, do you see virsh start blocking when you run it after you've shot your Ceph cluster?
>
> Yes, I had 5 domains running of RBD. I issued a "sync" inside one of the domains and it blocked for ever.

Just for clarification: when you say "it" blocked forever, are you referring to just the sync call inside your domain, or virsh domstate, or virsh altogether?

> > - Any chance you could downgrade your test env to libvirt 1.0.2 (from Ubuntu Cloud Archive for grizzly) and see if you can reproduce the error then? Because if you can't, then you're seeing a different issue than we do.
>
> I'll see if I can do that. It's just a test machine of me. Might have time for this tomorrow, otherwise it will be later this week.

Awesome. I'll try rerunning our own tests as well.

#12 - 02/04/2014 02:34 AM - Wido den Hollander

Florian Haas wrote:

> Wido den Hollander wrote:
>
> > > - Did you only test with an RBD storage **pool**, or do you also have **domains** running off of RBD? If the latter, do you see virsh start blocking when you run it after you've shot your Ceph cluster?
> >
> > Yes, I had 5 domains running of RBD. I issued a "sync" inside one of the domains and it blocked for ever.

> Just for clarification: when you say "it" blocked forever, are you referring to just the sync call inside your domain, or virsh domstate, or virsh altogether?

Ok, let me fully clarify it:

The "sync" command inside the domain blocked while I had the OSD ports blocked, but on the hypervisor "virsh list" and "virsh domstate" never blocked.

The "virsh pool-refresh rbd-pool" blocked during the whole time, but libvirt kept responding.

This system is running Ubuntu 12.04.4 with the Ubuntu Havana Cloud Archive:
- qemu-kvm: 1.5.0+dfsg-3ubuntu5~cloud0
- libvirt: 1.1.1-0ubuntu8~cloud2

So I can't fully reproduce your problem.

I will however submit a patch for libvirt with the timeouts Josh implemented in #6507 to have the pool refresh error out instead of blocking for ever.

> > - Any chance you could downgrade your test env to libvirt 1.0.2 (from Ubuntu Cloud Archive for grizzly) and see if you can reproduce the error then? Because if you can't, then you're seeing a different issue than we do.
>
> I'll see if I can do that. It's just a test machine of me. Might have time for this tomorrow, otherwise it will be later this week.

> Awesome. I'll try rerunning our own tests as well.

Great!

#13 - 02/04/2014 05:02 AM - Florian Haas

Wido den Hollander wrote:

> Florian Haas wrote:
>
>> Just for clarification: when you say "it" blocked forever, are you referring to just the sync call inside your domain, or virsh domstate, or virsh altogether?
>
> Ok, let me fully clarify it:
>
> The "sync" command inside the domain blocked while I had the OSD ports blocked, but on the hypervisor "virsh list" and "virsh domstate" never blocked.
>
> The "virsh pool-refresh rbd-pool" blocked during the whole time, but libvirt kept responding.
>
> This system is running Ubuntu 12.04.4 with the Ubuntu Havana Cloud Archive:
> - qemu-kvm: 1.5.0+dfsg-3ubuntu5~cloud0
> - libvirt: 1.1.1-0ubuntu8~cloud2
>
> So I can't fully reproduce your problem.

Understood, thanks. I reran my own tests in 4 VMs (3 Ceph OSDs/MONs, 1 libvirt host). Exact same results as you got. While the OSDs are unavailable, pool-refresh blocks and so does start for an RBD-backed domain. But virsh never blocks entirely, and any blocking on pool-refresh and start seems to not affect other libvirtd sessions at all.

On the system where I previously saw this issue, virsh calls do block, and indefinitely so. So I'd say there are two possible causes for that:

- The other system's Ceph cluster is in an error condition that both you and I are unable to reproduce in our test environments.
- It's a parallelization issue. The system where I can reproduce the problem has 48 cores; the one where everything is fine has 4. Incidentally, my test environment cannot reproduce https://bugzilla.redhat.com/show_bug.cgi?id=1060248 either, which is super easy to trigger on the 48-core box.

Are you aware of any lock-contention issues in libvirtd running on a large number of cores?

#14 - 02/04/2014 05:22 AM - Wido den Hollander

Florian Haas wrote:

> Wido den Hollander wrote:
>
> > Florian Haas wrote:
> >
> > > Just for clarification: when you say "it" blocked forever, are you referring to just the sync call inside your domain, or virsh domstate, or virsh altogether?
> >
> > Ok, let me fully clarify it:
> >
> > The "sync" command inside the domain blocked while I had the OSD ports blocked, but on the hypervisor "virsh list" and "virsh domstate" never blocked.
> >
> > The "virsh pool-refresh rbd-pool" blocked during the whole time, but libvirt kept responding.
> >
> > This system is running Ubuntu 12.04.4 with the Ubuntu Havana Cloud Archive:
> > - qemu-kvm: 1.5.0+dfsg-3ubuntu5~cloud0
> > - libvirt: 1.1.1-0ubuntu8~cloud2
> >
> > So I can't fully reproduce your problem.
>
> Understood, thanks. I reran my own tests in 4 VMs (3 Ceph OSDs/MONs, 1 libvirt host). Exact same results as you got. While the OSDs are unavailable, pool-refresh blocks and so does start for an RBD-backed domain. But virsh never blocks entirely, and any blocking on pool-refresh and start seems to not affect other libvirtd sessions at all.
>
> On the system where I previously saw this issue, virsh calls do block, and indefinitely so. So I'd say there are two possible causes for that:
>
> - The other system's Ceph cluster is in an error condition that both you and I are unable to reproduce in our test environments.

Could be, but very unlikely imho. If all OSDs are unreachable for the client the rados_read() or rados_write() call will simply block. Same goes if a PG is inactive.

> - It's a parallelization issue. The system where I can reproduce the problem has 48 cores; the one where everything is fine has 4. Incidentally, my test environment cannot reproduce https://bugzilla.redhat.com/show_bug.cgi?id=1060248 either, which is super easy to trigger on the 48-core box.
>
> Are you aware of any lock-contention issues in libvirtd running on a large number of cores?

No, no that I'm aware of. My test system is a 8-core AMD system and the biggest machines I've been running on at 2x 8-core AMD machines. I've never ran libvirt on 48-core machines.

So that's a thing I can't verify for you.

#15 - 02/11/2014 06:27 PM - Josh Durgin

- File 0001-rbd-prevent-blocking-when-cluster-is-non-responsive.patch added

Hi Wido,

Attached is a patch to use the timeouts with a libvirt storage pool. I noticed that libvirt will actually stop a storage pool and undefine it if there is an error refreshing it though. Would that work with CloudStack? Does it already redefine and start the pool if it does not exist?

#16 - 02/11/2014 11:44 PM - Wido den Hollander

Josh Durgin wrote:

> Hi Wido,
>
> Attached is a patch to use the timeouts with a libvirt storage pool. I noticed that libvirt will actually stop a storage pool and undefine it if there is an error refreshing it though. Would that work with CloudStack? Does it already redefine and start the pool if it does not exist?

Thanks! I was already working on a series of patches for libvirt and this is one of them:
https://github.com/wido/libvirt/commit/438ce1e717e58239c0629e681a8a258ec0ca05f0

That works just fine with CloudStack, it indeed re-defines the pool if it does not exist.

I'll try to get all these patches into libvirt asap.

#17 - 02/25/2014 02:39 AM - Wido den Hollander

The patch has been accepted in libvirt upstream: http://libvirt.org/git/?p=libvirt.git;a=commitdiff;h=60f70542f97805af49d656a582be35445046a0c9

I'll see if I can get this patch into Ubuntu 14.04 as well so we don't need a cloud-archive ppa for 14.04.

#18 - 03/11/2014 01:34 PM - Ian Colle

- Project changed from Ceph to rbd

- Category deleted (libvirt)

#19 - 03/14/2014 07:45 AM - Wido den Hollander

So don't forget the patch I got accepted into libvirt a few weeks ago.

If you are running a RBD storage pool it will time out on operations instead of blocking libvirt for ever. So that should probably help.

#20 - 03/25/2014 09:24 AM - Josh Durgin

- Status changed from New to Resolved

Marking resolved since Wido's patches landed upstream. Thanks!

#21 - 03/26/2014 01:32 AM - Florian Haas

Thanks!

Files

0001-rbd-prevent-blocking-when-cluster-is-non-responsive.patch	1.53 KB	02/11/2014	Josh Durgin