Item 86

git_comments:

1. Wait for the domain gets into shutoff state. When it does the dm object will no longer work, so we need to catch it.

git_commits:

1. **summary:** Summary: KVM - remove harmless message about domain not found on VM stop
   message: Summary: KVM - remove harmless message about domain not found on VM stop Detail: When we stop a VM, it's definition is no longer valid. Therefore, we need to catch the exception thrown from libvirt in trying to lookup a non-existent domain by UUID while trying to check if it's shut down. BUG-ID:CLOUDSTACK-600 Signed-off-by: Marcus Sorensen <marcus@betterservers.com> 1363201066 -0600

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** When rebooting KVM local storage VM host, libvirt definitions deleted
   description: This is definitely the case in 3.0.3, and I don't think the relevant code has been touched since. When you reboot a VM host running KVM local storage VMs, the VMs are deleted from libvirt. I presume this is due to CloudStack thinking it's migrating them away from the host, but obviously, given that we're on local storage, it's unable to do that. The result is that the VMs are not able to be restarted when the host comes back online.

jira_issues_comments:

1. Note - I've seen this happen now when the VM host is off the network and I reboot it, and when there's a RAID problem on the VM host and cloud-agent isn't able to communicate with the management server as a result (for some reason).
2. Are you sure this isn't just a misunderstanding in how cloudstack works? Nothing regarding a VM host's libvirt information should be persistent. When a storage pool is needed, cloudstack defines it. When a VM needs to be started, cloudstack creates the XML definition on the fly and feeds it to libvirt. So when a KVM host reboots and the network isn't accessible, or you say no VMs exist, it's no surprise to me, as the management server has to tell the agent which VMs exist and when to start them.
3. The problem is that the management server still thinks the VMs exist (albeit in a stopped state), but there's no definition of the VMs in libvirt on the KVM host. So we end up with VMs in CloudStack that can't be started, since we're on local storage. When the KVM host comes back up, the network is accessible etc, and cloud-agent comes online properly. /var/log/cloud/agent/agent.log ends up with a ton of messages like "libvir: QEMU error : Domain not found: no domain with matching uuid '04517459-47ba-3cb4-b07d-45199e17f0a7'" at this point - cloud-agent seems to think the domains should still exist.
4. The fact that there is no definition for the VMs in libvirt is by design, it won't survive a reboot as they're not persistent. When a rebooted host comes back online, it will attempt to stop any VMs that were previously running on that host, in case they were migrated away. This often fails and results in the 'Domain not found' errors you mention, but it's just a failsafe. So the issue you're having, based on the symptoms you mention, is that: 1) KVM Host is rebooted 2) VMs on that host show up as 'stopped' in the management UI 3) when KVM Host comes back and shows 'connected' in the management UI, you are unable to start the stopped VMs You are using local storage, is there anything else that you can tell us about the environment? Can you put the agent into debug mode (edit /etc/cloud/agent/log4j.xml and replace INFO with DEBUG everywhere, then restart agent), then capture just the section out of /var/log/cloud/agent/agent.log where attempting to start a VM fails?

5. Next time I get to a machine that I need to reboot, I'll put it in DEBUG. And yeah, your description of the issue is right. Migration shouldn't be happening here, since we're in local storage. I'm wondering if the storage for the VMs is getting deleted at reboot time...
6. How do you recover? Or do you just abandon those VMS?
7. No idea how we would recover - we can't start the VMs, so yeah, we abandon them. I'm really wishing I had a box I could replicate this on right now so I could verify if the instances get nuked...I'll try to dig one up tomorrow.
8. One thing I just thought of - the hosts in question weren't in maintenance mode when rebooted (they were either in alert or disconnected status) - I wonder if that might have caused weird behavior.
9. Freed up a host and am doing some experiments. Rebooting the host while not in maintenance mode but on the network etc resulted in the instance being suspended on shutdown, and then on restart, cloud-agent failed to start, with: {quote} Starting Cloud Agent: device virbr0-nic entered promiscuous mode New device virbr0-nic does not support netpoll Disabling netpoll for virbr0 virbr0: starting userspace STP failed, starting kernel STP nf_conntrack version 0.5.0 (16384 buckets, 65536 max) ERROR. Could not send configure hw lro request Ebtables v2.0 registered ip6_tables: (C) 2000-2006 Netfilter Core Team cloudbr0: port 1(eth2) entering forwarding state [FAILED] The host name does not resolve properly to an IP address. Cannot start Cloud Agent. {quote} ...but libvirt resumed the instance just fine, and I was able to start cloud-agent at that point, with no problems. Now trying to see what happens when I reboot with a disconnected network.
10. Rebooting with a disconnected network worked fine - this time, cloud-agent started up on its own properly, just to confuse things more. Fun!
11. And my last experiment before I had to give the machine back to users was to reboot with cloud-agent stopped before I rebooted. That also worked fine. blargh.
12. Ok. I'd say go ahead and turn on the debug, then if the issue comes back up we can perhaps see more about the root cause.
13. Ok, finally got a chance to play with a host with legitimate network problems. On reboot, the VMs were running (in libvirt) but not in CloudStack. Trying to start them in CloudStack would lead to errors and general confusion. Eventually, we restarted cloud-agent to turn on debug logging - when it came back online, it undefined the VMs in libvirt, but left the storage intact. A few minutes later, I tried relaunching an instance, and found it actually worked this time. We need to duplicate this experience to figure out where exactly the issue is that's keeping the instances from syncing up their state/coming back online automatically, but this is definitely progress.
14. So, to fix this issue we'll need a daemon that persists data, local storage (of vms) running on the host, or is there a way to fix it withing libvirt, or possibly fixing in upstream?
15. For CentOS this can be fixed by disabling service libvirt-guests. A host that brings up VMs without cloudstack knowing is a rogue node, even if the storage isn't shared. Disabling this service (probably at package install, in a post script or something) will ensure that the vms don't start up if cloudstack isn't connected. When the agent finally does connect, the VMs will be undefined and then cloudstack can start them. The good news is that the problem can be fixed by an admin, without doing anything to the code, simply by disabling that service. We'll need to look at ubuntu and see what they have that is similar. Actually, a better way might be to make sure that the vms aren't set to autostart when they're created. I don't immediately see a way to do that via the libvirt xml, it's usually done by creating a link to the vm's xml in /etc/libvirt/qemu/autostart.
16. For CentOS this can be fixed by disabling service libvirt-guests. A host that brings up VMs without cloudstack knowing is a rogue node, even if the storage isn't shared. Disabling this service (probably at package install, in a post script or something) will ensure that the vms don't start up if cloudstack isn't connected. When the agent finally does connect, the VMs will be undefined and then cloudstack can start them. The good news is that the problem can be fixed by an admin, without doing anything to the code, simply by disabling that service. We'll need to look at ubuntu and see what they have that is similar. Actually, a better way might be to make sure that the vms aren't set to autostart when they're created. I don't immediately see a way to do that via the libvirt xml, it's usually done by creating a link to the vm's xml in /etc/libvirt/qemu/autostart.
17. Checking the upstart script of libvirt-bin under Ubuntu shows it will destroy all running VMs when going down, no mention of suspending them. We (the agent) should probably not use Connect.domainDefineXML for starting a instance, but use Connect.domainCreateXML. The latter wil launch a instance based on the XML, but won't make it persistent. Otherwise we can also use Domain.setAutostart() to make sure it's not starting on boot, although that would look redundant to me and won't fix the root cause. There should be a way to make XML definitions in libvirt not persistent so it gets lost as soon as you reboot the hypervisor. For Java API, see: http://libvirt.org/sources/java/javadoc/

18. I wasn't going to mention that immediately since I see there are 'transient' properties for disks as well, and that makes them not save the changes, reverting to the original backing file on reboot. I wouldn't want to create a VM in 'transient' state without making sure I understood if that also applied. The domainCreateXML seems to be the transient VM mode according to this: http://wiki.libvirt.org/page/VM_lifecycle#Transient_guest_domains_vs_Persistent_guest_domains Other than checking that, domainCreateXML sounds like it's the way to go if it keeps us from remembering the VM outside of cloudstack. On Thu, Feb 7, 2013 at 8:53 AM, Wido den Hollander (JIRA)
19. I wasn't going to mention that immediately since I see there are 'transient' properties for disks as well, and that makes them not save the changes, reverting to the original backing file on reboot. I wouldn't want to create a VM in 'transient' state without making sure I understood if that also applied. The domainCreateXML seems to be the transient VM mode according to this: http://wiki.libvirt.org/page/VM_lifecycle#Transient_guest_domains_vs_Persistent_guest_domains Other than checking that, domainCreateXML sounds like it's the way to go if it keeps us from remembering the VM outside of cloudstack. On Thu, Feb 7, 2013 at 8:53 AM, Wido den Hollander (JIRA)
20. I pushed a commit for this yesterday evening, see commit 5dfcd309f10e5bd6a918f7fdff3f44a3dff2374a Verified that it works just fine on my hypervisors. Instances run just fine, but their XML doesn't show up in /etc/libvirt/qemu :)
21. Wido, this doesn't seem to work for system VMs, looks like they're sticking around. I haven't looked into it at all. OK, so I've looked into it, and notice that the fix wasn't applied to 4.1. I'm wondering if we should make a case for that, as these bugs are going to continue popping up.
22. Commit 8d7d1cd5623a6744a954fb35eeff8408d6381083 in branch refs/heads/master from Marcus Sorensen <marcus@betterservers.com> [https://git-wip-us.apache.org/repos/asf?p=incubator-cloudstack.git;h=8d7d1cd] Summary: KVM - undefine persistent VMs on stop Detail: A previous patch fixed an issue where we are defining VMs to persist locally on KVM hosts, which can cause issues if the agent isn't running and libvirt decides to start the VM unbeknownst to cloudstack. The previous patch stopped defining VMs as persistent. This patch adds compatibility for existing cloudstack environments, removing the persistent definition on stop if needed. BUG-ID: CLOUDSTACK-600 Signed-off-by: Marcus Sorensen <marcus@betterservers.com> 1363194656 -0600
23. Commit 97d2e3fe7772fa01941295397f9d59d35cf47671 in branch refs/heads/master from Marcus Sorensen <marcus@betterservers.com> [https://git-wip-us.apache.org/repos/asf?p=incubator-cloudstack.git;h=97d2e3f] Summary: KVM - remove harmless message about domain not found on VM stop Detail: When we stop a VM, it's definition is no longer valid. Therefore, we need to catch the exception thrown from libvirt in trying to lookup a non-existent domain by UUID while trying to check if it's shut down. BUG-ID:CLOUDSTACK-600 Signed-off-by: Marcus Sorensen <marcus@betterservers.com> 1363201066 -0600
24. Thanks, all! I appreciate the work.
25. the three mentioned patches, when combined, ensure that VM's aren't locally defined, and that existing persistent definitions will be cleaned up on vm stop.
26. Commit 186fb9ff51c2084a5ca514eeac387158f1cb5cf6 in branch refs/heads/4.1 from Chip Childers <chip.childers@gmail.com> [https://git-wip-us.apache.org/repos/asf?p=incubator-cloudstack.git;h=186fb9f] Summary: KVM - remove harmless message about domain not found on VM stop Detail: When we stop a VM, it's definition is no longer valid. Therefore, we need to catch the exception thrown from libvirt in trying to lookup a non-existent domain by UUID while trying to check if it's shut down. BUG-ID:CLOUDSTACK-600 Signed-off-by: Marcus Sorensen <marcus@betterservers.com> 1363201066 -0600
27. Commit 6df107bcf6439c1de8346bc8b5c6be93694d9ee4 in branch refs/heads/4.1 from Chip Childers <chip.childers@gmail.com> [https://git-wip-us.apache.org/repos/asf?p=incubator-cloudstack.git;h=6df107b] Summary: KVM - undefine persistent VMs on stop Detail: A previous patch fixed an issue where we are defining VMs to persist locally on KVM hosts, which can cause issues if the agent isn't running and libvirt decides to start the VM unbeknownst to cloudstack. The previous patch stopped defining VMs as persistent. This patch adds compatibility for existing cloudstack environments, removing the persistent definition on stop if needed. BUG-ID: CLOUDSTACK-600 Signed-off-by: Marcus Sorensen <marcus@betterservers.com> 1363194656 -0600
28. closing since 4.1.0 is now released