Arvados - Story #8236

[NodeManager] Node Manager stops itself when actors stop responding

01/19/2016 07:50 PM - Peter Amstutz

Status:	Resolved	**Start date:**	05/17/2016
Priority:	Normal	**Due date:**	
Assigned To:	Peter Amstutz	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	2016-05-25 sprint		

Description

- Add code to _check_poll_freshness that triggers the existing on_failure code to cause Node Manager to die if any of the lists are hopelessly stale.
- There should be a new configuration knob to decide when the poll is hopelessly stale.
- It should default to some multiple of the configured freshness time.
- Ops can decide what the default multiplier is, and maybe the name of the configuration value.

Subtasks:

Task # 9204: Review 8236-nodemanager-watchdog	**Resolved**
Task # 9203: Add watchdog	**Resolved**

Associated revisions

Revision f2cb2d2f - 05/18/2016 01:27 PM - Peter Amstutz

Merge branch '8236-nodemanager-watchdog' closes #8236

History

#1 - 01/20/2016 06:13 PM - Brett Smith

- Target version set to Arvados Future Sprints

#2 - 05/10/2016 06:09 PM - Brett Smith

- Tracker changed from Bug to Story

- Subject changed from [NodeManager] Watchdog to restart node manager when actors stop responding to [NodeManager] Node Manager stops itself when actors stop responding

- Description updated

- Story points set to 1.0

We are doing this as our way of addressing #7667.

#3 - 05/11/2016 05:07 PM - Brett Smith

- Target version changed from Arvados Future Sprints to 2016-05-25 sprint

#4 - 05/11/2016 07:16 PM - Peter Amstutz

- Assigned To set to Peter Amstutz

#5 - 05/17/2016 03:17 PM - Peter Amstutz

I choose a different strategy than that in description, this approach addresses the general class of problems where an actor becomes hopelessly stuck rather that just the polling actors specifically:

Added WatchdogActor. This actor goes through the actor list using pykka.ActorRegistry.get_all() and calls ping().get(timeout) on each one. If ping() times out, the actor is stuck, so kill node manager.

#6 - 05/17/2016 05:33 PM - Nico César

review 1fd5716e1714337b6ff96f6725e1f22c7a6ceb65

So I see 2 kill() executions. one inside the WatchdogActor.killself which I have no complains and another in BaseNodeManagerActor. here the relevant part of the diff:

```
diff --git a/services/nodemanager/arvnodeman/baseactor.py b/services/nodemanager/arvnodeman/baseactor.py
index 9591b42..840ba4c 100644
--- a/services/nodemanager/arvnodeman/baseactor.py
+++ b/services/nodemanager/arvnodeman/baseactor.py
@@ -82,4 +84,39 @@ class BaseNodeManagerActor(pykka.ThreadingActor):
         if (exception_type in (threading.ThreadError, MemoryError) or
             exception_type is OSError and exception_value.errno == errno.ENOMEM):
             lg.critical("Unhandled exception is a fatal error, killing Node Manager")
-            os.killpg(os.getpgid(0), 9)
+            os.kill(os.getpid(), signal.SIGQUIT)
```

switching from signal.SIGKILL / 9 to signal.SIGQUIT / 3 and from killpg() to kill() could bring us some **unknown** problems when threading.ThreadError is coming up. Since we still don't know the cause of this. And also we're doing 2 changes at once. killpg -> kill AND SIGKILL -> SIGQUIT

My approach here will leave this line as is (with some minor change for clarity):

```
os.killpg(os.getpgid(0), signal.SIGKILL)
```

and add a comment with

```
# we will try
#   os.killpg(os.getpid(), signal.SIGQUIT)
# and
#   os.kill(os.getpid(), signal.SIGQUIT)
# in the future
```

making this a minimal impact for a situation that we don't know.

if we have a graph for this restarts in https://termite.curoverse.com/app/kibana#/dashboard/Node-Manager?_g=%28refreshInterval:%28display:Off,pause:!f,value:0%29,time:%28from:now-30d,mode:quick,to:now%29%29&_a=%28filters:!%28%29,options:%28darkTheme:!f%29,panels:!%28%28col:1,id:restarted-node-manager,panelIndex:1,row:1,size_x:12,size_y:2,type:visualization%29%29,query:%28query_string:%28analyze_wildcard:!t,query:%27*%27%29%29,title:%27Node%20Manager%27,uiState:%28%29%29

since watchdog will have gracefully suicide death we can monitor both consequences. and see which problem is more common in our clusters.

does it makes sense?

#7 - 05/17/2016 06:15 PM - Brett Smith

Peter Amstutz wrote:

> Added WatchdogActor. This actor goes through the actor list using pykka.ActorRegistry.get_all() and calls ping().get(timeout) on each one. If ping() times out, the actor is stuck, so kill node manager.

Define "stuck." An actor can still be making progress through a large mailbox where each message takes a while to process. If that's the case, this ping will almost certainly timeout, even though the actor is still alive and working.

Right now we know this happens most often today with ComputeNodeUpdateActor. If it loses contact with the cloud API server, it is expected that its backlog will grow long and it will take a long time to respond to any individual request. Given enough time, it still will recover correctly. And restarting won't really improve the situation, since the fundamental problem is that the cloud API server is gone, so Node Manager can't do any work at all.

#8 - 05/17/2016 09:02 PM - Peter Amstutz

Nico Cesar wrote:

> review 1fd5716e1714337b6ff96f6725e1f22c7a6ceb65
>
> So I see 2 kill() executions. one inside the WatchdogActor.killself which I have no complains and another in BaseNodeManagerActor. here the relevant part of the diff:
>
> [...]
>
> switching from signal.SIGKILL / 9 to signal.SIGQUIT / 3 and from killpg() to kill() could bring us some **unknown** problems when threading.ThreadError is coming up. Since we still don't know the cause of this. And also we're doing 2 changes at once. killpg -> kill AND SIGKILL -> SIGQUIT
>
> My approach here will leave this line as is (with some minor change for clarity):
> [...]
>
> and add a comment with

> [...]
>
> making this a minimal impact for a situation that we don't know.
>
> if we have a graph for this restarts in https://termite.curoverse.com/app/kibana#/dashboard/Node-Manager?_g=%28refreshInterval:%28display:Off,pause:!f,value:0%29,time:%28from:now-30d,mode:quick,to:now%29%29&_a=%28filters:!%28%29,options:%28darkTheme:!f%29,panels:!%28%28col:1,id:restarted-node-manager,panelIndex:1,row:1,size_x:12,size_y:2,type:visualization%29%29,query:%28query_string:%28analyze_wildcard:!t,query:%27*%27%29%29,title:%27Node%20Manager%27,uiState:%28%29%29
>
> since watchdog will have gracefully suicide death we can monitor both consequences. and see which problem is more common in our clusters.
>
> does it makes sense?

I restored os.killpg(os.getpgid(0), signal.SIGKILL) but added os.setsid() to main() so that it creates a new process group. That fixes the original issue that was raised (killing the process group could kill the parent, too.)

#9 - 05/17/2016 09:20 PM - Peter Amstutz

Brett Smith wrote:

> Peter Amstutz wrote:
>
> > Added WatchdogActor. This actor goes through the actor list using pykka.ActorRegistry.get_all() and calls ping().get(timeout) on each one. If ping() times out, the actor is stuck, so kill node manager.
>
> Define "stuck." An actor can still be making progress through a large mailbox where each message takes a while to process. If that's the case, this ping will almost certainly timeout, even though the actor is still alive and working.
>
> Right now we know this happens most often today with ComputeNodeUpdateActor. If it loses contact with the cloud API server, it is expected that its backlog will grow long and it will take a long time to respond to any individual request. Given enough time, it still will recover correctly. And restarting won't really improve the situation, since the fundamental problem is that the cloud API server is gone, so Node Manager can't do any work at all.

Noted.

I adjusted it so that instead of pinging all actors, it only checks the four most important ones: the cloud, arvados, and job pollers, and the daemon actor. Based on the behavior and implementation of these classes, I think we can reasonably assume that none of them should take more than 10 minutes to respond during normal operation. How does that sound?

#10 - 05/17/2016 09:25 PM - Peter Amstutz

Now at c193d814c22e2a4227c7f49e76b0d9b589cff4be

#11 - 05/18/2016 01:24 PM - Nico César

test c193d814c22e2a4227c7f49e76b0d9b589cff4be :)

LGTM, I'm happy with the logging we have so I want to deploy this and see when the watchdog is actually invoked.

#12 - 05/18/2016 01:30 PM - Peter Amstutz

- Status changed from New to Resolved

- % Done changed from 50 to 100

Applied in changeset arvados|commit:f2cb2d2f14c8509b7e06126fefead0da282ef2fd.