Arvados - Bug #10701

[Crunch2] crunch-dispatch-slurm leaks file descriptors

12/09/2016 08:16 PM - Peter Amstutz

Status:	Resolved	**Start date:**	01/31/2017
Priority:	Normal	**Due date:**	
Assigned To:	Tom Clegg	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	2017-02-15 sprint		

Description

Subtasks:

Task # 10741: Review	**Resolved**
Task # 11086: Review 10701-refactor-dispatch	**Resolved**

Related issues:

Related to Arvados - Bug #10700: [Crunch2] crunch-dispatch-slurm pileup	**Resolved**	**01/27/2017**

Associated revisions

Revision c3d4f8a5 - 01/31/2017 09:32 PM - Tom Clegg

Merge branch '10700-dispatch'

closes #10700
refs #10701
closes #10702
closes #10703
closes #10704

Conflicts:
services/crunch-dispatch-slurm/crunch-dispatch-slurm.go

Revision c86cbaa6 - 02/14/2017 04:08 PM - Tom Clegg

Merge branch '10701-refactor-dispatch'

closes #10701

History

#1 - 12/09/2016 08:16 PM - Peter Amstutz

- Subject changed from [Crunch2] crunch-dispatch-slurm possibly leaks file descriptors to [Crunch2] crunch-dispatch-slurm leaks file descriptors

#2 - 12/14/2016 09:00 PM - Peter Amstutz

- Target version set to 2017-01-04 sprint

#3 - 12/14/2016 09:03 PM - Peter Amstutz

- Assigned To set to Peter Amstutz

#4 - 01/04/2017 08:13 PM - Peter Amstutz

- Target version changed from 2017-01-04 sprint to 2017-01-18 sprint

#5 - 01/18/2017 08:10 PM - Peter Amstutz

- Target version changed from 2017-01-18 sprint to 2017-02-01 sprint

#6 - 01/18/2017 08:28 PM - Tom Clegg

- Assigned To changed from Peter Amstutz to Tom Clegg

#7 - 01/25/2017 10:39 PM - Tom Clegg

Just before reaching max fds, server logs show many attempts to run sbatch. Some failed ("Error submitting container [...]: Container submission

failed: [...]") but some seem to leave (or get stuck in?) submit() without saying anything after "sbatch starting":

```
2016-12-08_22:40:45.07549 2016/12/08 22:40:45 sbatch starting: ["sbatch" "--share" "--workdir=/tmp" "--job-nam
e=9tee4-dz642-5s5vdaeeu98qhbr" "--mem-per-cpu=100000" "--cpus-per-task=1"]
2016-12-08_22:40:45.15559 2016/12/08 22:40:45 Monitoring container 9tee4-dz642-5s5vdaeeu98qhbr finished
```

There is a bunch of unnecessary pipe-wrangling in in submit(), which I replaced with this:

```
    cmd.Stdin = strings.NewReader(...)

    var stdout, stderr bytes.Buffer
    cmd.Stdout = &stdout
    cmd.Stderr = &stderr
```

It only takes 100ms for c-d-s to make this transition, which seems fishy given that pollPeriod is 500ms -- squeue should run only 2x per second, and checkSqueue() is always supposed to wait for the *next* squeue result.

```
2016-12-08_22:40:45.07549 2016/12/08 22:40:45 sbatch starting: ["sbatch" "--share" "--workdir=/tmp" "--job-nam
e=9tee4-dz642-5s5vdaeeu98qhbr" "--mem-per-cpu=100000" "--cpus-per-task=1"]
2016-12-08_22:40:45.15559 2016/12/08 22:40:45 Monitoring container 9tee4-dz642-5s5vdaeeu98qhbr finished
```

There might be an opportunity for a race between run() and monitorSubmitOrCancel(), where run() sees the status channel close (and sets monitorDone=true) before monitorSubmitOrCancel() even has a chance to ask for the next squeue after running sbatch. dispatch.go might be shooting itself in the foot here:

```
    if container.State == Queued && dispatcher.checkMine(container, false) {
        // If we previously started the job, something failed, and it
        // was re-queued, this dispatcher might still be monitoring it.
        // Stop the existing monitor, then try to lock and run it
        // again.
        dispatcher.notMine(container.UUID)
    }
```

pollContainers() calls getContainers() three times in quick succession: first with paramsQ ("queued and ready"), then with paramsP ("locked by me"), then a third time with the union of matching container UUIDs from the first two. If the third query runs before handleUpdate() succeeds in locking one of the paramsQ results, that generates another handleUpdate() with state=Queued, which would hit the above "notMine" case.

I've eliminated this race by calling handleUpdate() synchronously from getContainers() instead of using a channel.

#8 - 01/26/2017 03:32 PM - Tom Clegg

Tried running the old version of submit() many times with a command that fails or can't exec, but this doesn't reproduce the file descriptor leak.

#9 - 01/26/2017 08:05 PM - Tom Clegg

- Status changed from New to In Progress

#10 - 02/01/2017 08:06 PM - Tom Clegg

- Status changed from In Progress to Closed

Closing because there's a good chance it got fixed in #10700 by getting rid of a bunch of stdin/stdout/stderr pipes.

#11 - 02/09/2017 05:15 PM - Tom Clegg

- Status changed from Closed to In Progress

- Target version changed from 2017-02-01 sprint to 2017-02-15 sprint

#12 - 02/09/2017 07:30 PM - Tom Clegg

Problems:

- sdk/go/dispatch assumes the dispatcher's RunContainer method is done as soon as the channel closes while in fact, multiple goroutines in crunch-dispatch-slurm might still be doing work.
- after trying dispatcher.Unlock(), monitorSubmitOrCancel waits in sqCheck.HasUUID() for ~PollPeriod. During that interval, the main dispatcher loop is likely to see the container reappear in the queue, lock it, and start another goroutine to monitor it.

Proposed fixes:

- sdk/go/dispatch should wait for RunContainer to return before deleting the container from mineMap.
- crunch-dispatch-slurm should not return from run (RunContainer) until monitorSubmitOrCancel has also finished.
- simplify crunch-dispatch-slurm by combining the two per-container event loops into one goroutine.

#13 - 02/10/2017 07:19 PM - Tom Clegg

10701-refactor-dispatch @ a2fe6f9367de3ee93064fbee3f2df78ce84aa318

#14 - 02/14/2017 12:12 AM - Radhika Chippada

In dispatch.go:

- can we rename Dispatcher.running as “trackers” ?
- In runningUUIDs, this comment is confusing: // API bug: ["uuid", "not in", []] does not match everything > The issue here is that no trackers are found? May be you can instead say since ‘[“uuid", "not in", []]’ does not work, use a dummy or something. Also, instead of “X” can you use “dummy” or something? Also, do we want to log the fact that there were no trackers?
- In Run method for loop, can we get runningUUIDs() once and reuse?
- tracker, running := d.running[c.UUID] > can we please rename “running” as “ok” here? With all the other "running" things, this was quite confusing.

#15 - 02/14/2017 03:46 PM - Tom Clegg

Radhika Chippada wrote:

> - can we rename Dispatcher.running as “trackers” ?

Good call -- done.

> - In runningUUIDs, this comment is confusing: // API bug: ["uuid", "not in", []] does not match everything > The issue here is that no trackers are found? May be you can instead say since ‘[“uuid", "not in", []]’ does not work, use a dummy or something. Also, instead of “X” can you use “dummy” or something? Also, do we want to log the fact that there were no trackers?

Made the comment a bit less enigmatic, and changed "X" to "this-uuid-does-not-exist". len(trackers)0 just means nothing is running right now, which is normal, so I don't think we need to log it.

> - In Run method for loop, can we get runningUUIDs() once and reuse?

Sure thing. I thought it would be easier to follow if we used the most up-to-date list each time, but I don't suppose it matters much.

> - tracker, running := d.running[c.UUID] ==> can we please rename “running” as “ok” here? With all the other "running" things, this was quite confusing.

Renamed to "alreadyTracking".

#16 - 02/14/2017 04:00 PM - Radhika Chippada

LGTM. Thanks.

#17 - 02/14/2017 04:15 PM - Tom Clegg

- Status changed from In Progress to Resolved

- % Done changed from 50 to 100

Applied in changeset arvados|commit:c86cbaa6f286e50900dae3203a42044449e042f7.