**Arvados - Feature #10181**

**Crunch job output logging improvement stories**

10/03/2016 11:16 PM - Joshua Randall

<table>
<thead>
<tr>
<th>Status:</th>
<th>Resolved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assigned To:</td>
<td>Tom Clegg</td>
</tr>
<tr>
<td>Category:</td>
<td>API</td>
</tr>
<tr>
<td>Target version:</td>
<td>2018-10-03 Sprint</td>
</tr>
<tr>
<td>Start date:</td>
<td>02/16/2017</td>
</tr>
<tr>
<td>Due date:</td>
<td>02/16/2017</td>
</tr>
<tr>
<td>% Done:</td>
<td>100%</td>
</tr>
<tr>
<td>Estimated time:</td>
<td>0.00 hour</td>
</tr>
</tbody>
</table>

**Description**

**Story: job output logged to keep while job is running**

As a user, I would like to be able to retrieve complete (i.e. starting at the beginning and without being silenced by rate limiting) output of any running jobs. My strong preference would be to use keep for this, since (a) the full job log will be in keep anyway, so I'm already used to using that as an interface and (b) the bulk storage available to keep is generally going to be much greater than other places the logs could be kept (such as in the database). I would be ok with not necessarily always having up-to-the minute job logs available in keep, as long as the logs that are there are complete up to the point where they are truncated. Perhaps the final line of a truncated log entry could note that the job is still running and more output will arrive soon, along with stating the timestamp of the point at which the logs were flushed to keep (i.e. I would then know to expect that the next line would be timestamped after that time).

As a sysadmin, I would like to be able to adjust settings for flushing job logs to keep. I assume that any time a crunch job has a full block (i.e. 64MB) of output that it would be immediately written to keep and that the job's log collection would be updated to point to a new portable data hash which includes the new block. However, it would also be good to have a setting for flushing smaller amounts of log data to keep, so that logs from jobs that haven't output very much in some time can nonetheless be available. For example, I might configure a setting such that job output would be written to keep and the collection portable data hash updated every 15m regardless of how much output has been produced. That configuration option would be a tradeoff between creating a potentially large number of partially used keep blocks (although they would end up being cleaned up by keep-balance once a collection no longer points to them) and having a wait a long time for job output to appear in keep.

(remainder moved to #14284)

**Story: job output does not belong in the database logs table and should be able to be directed to non-Arvados logging systems**

As a sysadmin, I'd rather my postgres database not fill up with hundreds of GB of job output logs. In addition to requiring a large amount of storage on the volume where the postgres database lives, this also tends to make queries to the logs table that have nothing to do with job output logging (i.e. fulfilling its role as more of an audit log, such as checking for recent changes to collections) take ridiculously long. I think it would be best if no job output at all was stored in the central postgres database. In conjunction with the above story regarding storing in-progress job logs to keep, it would be great if some other system which is better suited to the task of buffering and distributing recent job output in order to make real-time job output available. It would be great if it could be sent via an existing log broker system such as logstash or fluentd such that it would be possible to not only direct the logs to whatever component Arvados uses to buffer and deliver the logs to consumers (such as via the existing websockets interface) but also to other non-Arvados logging systems (where we may be running the rest of the ELK/EFK stack for search and visualisation).

**Subtasks:**

- Task # 11122: log partial job output to keep while job is running
- Task # 14241: Review 10181-incremental-log

**Related issues:**

- Related to Arvados - Feature #12996: [SDKs] CollectionFS should repack highly...
- Related to Arvados - Story #13048: Refactor crunch2 logging
- Related to Arvados - Story #14284: Send real time container logs to a suitable...

**Associated revisions**

Revision c25f36a0 - 10/01/2018 07:43 PM - Tom Clegg

Merge branch '10181-incremental-log'

refs #10181

Arvados-DCO-1.1-Signed-off-by: Tom Clegg <tclegg@veritasgenetics.com>

**History**

05/09/2021
# Implementation notes

- Containers API needs to permit updating to a non-null "log" field when state=Running.
- This is an opportune time for crunch-run to drop its custom collection-writing code and use "collection filesystem" instead (if that hasn't happened already). CollectionFS makes it easy to get a goroutine-safe manifest snapshot.
- The "flushing logs" and "flushed logs, pdh is X" log entries can go in crunch-run.txt.

Good times to flush:

- Just before starting the container (i.e., immediately after updating state to Running) -- this ensures node-info etc. are stored.
- Every hour (configurable)
- SIGUSR1
- 32 MiB of logs have been written since last flush (or some other threshold that's large but less than 64MiB)

---

# Story points set to 2.0

---

# Related to Feature #12996: [SDKs] CollectionFS should repack highly fragmented files added

---

# from discussion offline we can do this in three phases.

**phase 1: save log snapshots periodically**

(as described in note-2)

**phase 2: improve block packing in CollectionFS**

This is [#12996](#12996).

**phase 3: save log snapshots frequently**

Once [#12996](#12996) is done, it should be feasible to save a snapshot much more frequently -- e.g., every ~10 seconds or ~1KB. This would eliminate the motivation for a user-accessible "tell crunch-run to flush logs" API. Users could usefully run "tail -f" on log collections.

This will also make progress toward obsoleting the "send logs to API server, too" stuff: workbench/users can use "tail -f" instead.

---

# Related to Story #13048: Refactor crunch2 logging added

---

# Target version changed from Arvados Future Sprints to 2018-10-03 Sprint

---

# Status changed from New to In Progress

---

# Capturing from chat:

Anyone have thoughts on how we should give the container-requesting users permission to read the log collection for a running container? extend the permission system to give access through the CR... or create a copy of the (container's own) log collection for each CR before it's finalized and keep their contents synchronized as the main one gets updated... or implement a special symlink/alias kind of collection... or...?

Since the container request has a specific log_uuid, I think we want to create a log output collection per container request, and when the container's log is updated, find all the linked container requests and updates their log collections in the same transaction. So your 2nd idea. If/when we create a separate table to deduplicate manifest text, then it becomes a trivial update.
panic: can't marshal segment type *arvados.memSegment

goroutine 21 [running]:
git.curoverse.com/arvados.git/sdk/go/arvados.(*dirnode).marshalManifest(0xc4203e6090, 0xb55055, 0x1, 0x0, 0x0, 0x0, 0x0)
/home/tom/.cache/arvados-build/GOPATH/src/git.curoverse.com/arvados.git/sdk/go/arvados/fs_collection.go:689 +0x13e2

git.curoverse.com/arvados.git/sdk/go/arvados.(*dirnode).MarshalManifest(0xc42034a0c0, 0xb55055, 0x1, 0x0, 0x0, 0x0)
/home/tom/.cache/arvados-build/GOPATH/src/git.curoverse.com/arvados.git/sdk/go/arvados/fs_collection.go:141 +0x141

created by git.curoverse.com/arvados.git/services/crunch-run.NewContainerRunner
/home/tom/.cache/arvados-build/GOPATH/src/git.curoverse.com/arvados.git/services/crunch-run/crunchrun.go:1693 +0x419

exit status 2

#10 - 09/21/2018 08:51 PM - Tom Clegg
Testing reveals a bug in collectionfs at e7b3a06c0:

panic: can't marshal segment type *arvados.memSegment

goroutine 21 [running]:
git.curoverse.com/arvados.git/sdk/go/arvados.(*dirnode).marshalManifest(0xc4203e6090, 0xb55055, 0x1, 0x0, 0x0, 0x0, 0x0)
/home/tom/.cache/arvados-build/GOPATH/src/git.curoverse.com/arvados.git/sdk/go/arvados/fs_collection.go:689 +0x13e2

git.curoverse.com/arvados.git/sdk/go/arvados.(*dirnode).MarshalManifest(0xc42034a0c0, 0xb55055, 0x1, 0x0, 0x0, 0x0)
/home/tom/.cache/arvados-build/GOPATH/src/git.curoverse.com/arvados.git/sdk/go/arvados/fs_collection.go:141 +0x141

created by git.curoverse.com/arvados.git/services/crunch-run.NewContainerRunner
/home/tom/.cache/arvados-build/GOPATH/src/git.curoverse.com/arvados.git/services/crunch-run/crunchrun.go:1693 +0x419

exit status 2

#11 - 09/25/2018 06:52 PM - Tom Clegg
10181-incremental-log @ 3596af0954df05b06799814585d834502d0d76a

#12 - 09/25/2018 09:01 PM - Tom Clegg
10181-incremental-log @ d5aebe0c9a768004dfb80667853f11b5048ae26f0https://ci.curoverse.com/view/Developer/job/developer-run-tests/905/

#13 - 09/26/2018 08:21 PM - Tom Clegg
10181-incremental-log @ 5d6a2c2e4b85434e9bae1dd0adc27c284cb9ea85
- update log on SIGUSR1
- rename configs to crunchLogUpdateSize/Period
- add configs to apiserver config and discovery doc so they're configurable in practice

#14 - 09/27/2018 08:00 PM - Lucas Di Pentima
I did a manual test using arvbox. Set up the update to be every 5 secs and made a workflow with a step that counted from 1 to 300 doing a 1s sleep in between.

The initial collection got created with the first 5 seconds of data, but that was it, below is a piece of the log streamed to workbench:

 [...]
... then every 5 seconds it produced the same error message. When the count finished, the container was cancelled:

[...]

That collection is trashed so, that could be the reason it returns 404?

#16 - 09/28/2018 06:42 PM - Tom Clegg

Yes, that makes sense. Updated: 10181-incremental-log @ 5e41146b32e4afd9de4eefc22b3c9fef05c471dc

#17 - 09/28/2018 07:32 PM - Lucas Di Pentima

For some reason runtime_status is not being able to be updated when status==Running. The following appeared on stderr.log on the runner container request:

2018-09-28T19:01:05.115298403Z arvados.cwl-runner INFO: Couldn't update runtime_status: <HttpError 422 when requesting https://172.17.0.2:8000/arvados/v1/containers/o967z-d2642-4hyignu6z3r43w8?alt=json returned "Runtime status cannot be modified in state 'Running' ({}), "error"=>"arvados.cwl-runner: [container s1] (o967z-xvhdp-e8j0rgpir5sgegr) error log:","errorDetail=>"... than every 5 seconds it produced the same error message. When the count finished, the container was cancelled:

[...]

That collection is trashed so, that could be the reason it returns 404?

#16 - 09/28/2018 06:42 PM - Tom Clegg

Yes, that makes sense. Updated: 10181-incremental-log @ 5e41146b32e4afd9de4eefc22b3c9fef05c471dc

#17 - 09/28/2018 07:32 PM - Lucas Di Pentima

For some reason runtime_status is not being able to be updated when status==Running. The following appeared on stderr.log on the runner container request:

2018-09-28T19:01:05.115298403Z arvados.cwl-runner INFO: Couldn't update runtime_status: <HttpError 422 when requesting https://172.17.0.2:8000/arvados/v1/containers/o967z-d2642-4hyignu6z3r43w8?alt=json returned "Runtime status cannot be modified in state 'Running' ({}), "error"=>"arvados.cwl-runner: [container s1] (o967z-xvhdp-e8j0rgpir5sgegr) error log:","errorDetail=>"...
In this case step s1 is just an "sleep 5; exit 1" script, I was using the same example wf to demo #13373.

#18 - 09/28/2018 08:24 PM - Tom Clegg
Ah, we were only testing updating runtime_status using the dispatcher token, not using the container's own token. Fixed.

#19 - 10/01/2018 01:33 PM - Lucas Di Pentima
This LGTM, thanks.

#20 - 10/03/2018 03:18 PM - Tom Clegg
- Status changed from In Progress to Resolved

#21 - 10/03/2018 07:24 PM - Tom Clegg
- Related to Story #14284: Send real time container logs to a suitable log distribution system (instead of adding rows to the postgres logs table) added

#22 - 10/03/2018 07:24 PM - Tom Clegg
- Description updated

#23 - 11/13/2018 08:49 PM - Tom Morris
- Release set to 14