Arvados - Story #13484

[API] Support multiple load-balanced API server nodes

05/14/2018 09:07 PM - Tom Clegg

Status:	New	**Start date:**	03/18/2019
Priority:	Normal	**Due date:**	
Assigned To:		**% Done:**	0%
Category:	API	**Estimated time:**	0.00 hour
Target version:	Arvados Future Sprints		

Description

Background: the Rails API server application becomes a performance bottleneck during heavy load, e.g., when hundreds of containers/nodes are running. There are some ways to respond to this -- use a bigger/faster machine, adjust logging configs, move postgresql to a different machine -- but it would be much better if the operator could add API server nodes to increase capacity. However, there are some parts of the code base that assume there's only one API server.

In this issue, we remove those barriers, so a site admin can safely add and remove additional API servers and route traffic to them with a load balancer.

(However, multi-API-server installations are not expected to support crunch1 jobs.)

Known/suspected issues:

- Job validation code assumes git repositories are stored in the local filesystem (todo: confirm this only affects crunch1)
- Audit log cleanup code uses flock() to avoid wastefully running concurrent cleanup threads (todo: confirm concurrent cleanup threads are harmless, and/or use a database lock instead)
- Sample DNS update scripts (triggered by "node ping") assume the API host is the DNS server (todo: offer a sample DNS update strategy suitable for multiple nodes).

Subtasks:

Task # 14948: Update install docs	**In Progress**
Task # 14949: Ops deploy multi-API server configuration on test cluster	**New**
Task # 14978: Review	**New**

Related issues:

Blocked by Arvados - Story #14873: [API] Update to Rails 5	**Resolved**	**03/20/2019**

History

#1 - 05/16/2018 04:03 PM - Lucas Di Pentima

Some observations/questions:

Potential issues:

- RequestIDs: Is this solved by a smart load balancer?
- Multiple sweep trashed objects processes on every API server
  - Is this configuration thing?
  - Should be split that code into a separate service?
- How about the logs table being read from several api instances?

#2 - 05/16/2018 04:24 PM - Tom Clegg

Simplifying assumptions

- OK to waste some work due to every apiserver running a trash/log sweeping thread
- Load balancer must be configured to route all node ping requests to a single API server which is also the DNS server
- All API servers are shut down while any API server is being upgraded
- API servers are not aware of anything like "my ID" or what other API servers are running

#3 - 05/16/2018 04:26 PM - Tom Clegg

- Story points set to 1.0

#4 - 05/22/2018 08:28 PM - Tom Morris

- Target version changed from To Be Groomed to Arvados Future Sprints

#5 - 03/13/2019 03:37 PM - Tom Morris

- Target version changed from Arvados Future Sprints to 2019-03-27 Sprint

#6 - 03/13/2019 03:39 PM - Tom Morris

- Story points changed from 1.0 to 2.0

#7 - 03/13/2019 03:39 PM - Tom Morris

- Assigned To set to Peter Amstutz

#8 - 03/15/2019 04:07 PM - Peter Amstutz

audit_logs.rb#delete_old creates tmp/audit_logs.lock

update_priority.rb#update_priority creates tmp/update_priority.lock

refresh_permission_view runs in a transaction and takes a table lock

sweep_trashed_objects#sweep_now does not set up explicit transaction but individual statements should be transactional. Seems like delete_project_and_contents should be in an explicit transaction at least.

crunch_dispatch.rb interacts with local slurm and git repo (specific to crunchv1)

commit_ancestor.rb and commit.rb read a local git repo (specific to crunchv1)

job.rb touches the file at crunch_refresh_trigger (specific to crunchv1)

The nodes table updates local DNS configuration. Used by node manager, could technically be used by on-prem configuration but I don't think anyone does.

Websocket events are triggered by database NOTIFY.

Async permission updates use local cache freshness to suppress updates. However if another node performs an update it is not harmful.

The discovery document is stored in local cache. There is a generatedAt field which is db_current_time, which means otherwise identical servers on different hosts are likely to report different generateAt times.

Login process uses a session. I'm not sure if this is necessary, or what the implications are if the user starts a session on one host and the next request goes to a different host.

#9 - 03/15/2019 07:05 PM - Peter Amstutz

Peter Amstutz wrote:

> audit_logs.rb#delete_old creates tmp/audit_logs.lock
>
> update_priority.rb#update_priority creates tmp/update_priority.lock
>
> refresh_permission_view runs in a transaction and takes a table lock

I think these would benefit from being wrapped in explicit transactions, but there doesn't appear to be much benefit from the file lock, except in preventing overlapping threads if the operation takes longer than the quiet period based on Rails.cache expiration. If that is important, we could use a table lock instead.

> sweep_trashed_objects#sweep_now does not set up explicit transaction but individual statements should be transactional. Seems like delete_project_and_contents should be in an explicit transaction at least.
>
> Async permission updates use local cache freshness to suppress updates. However if another node triggers a permission update it is not harmful.

These doesn't have the file lock, but otherwise the same comment as above applies.

> crunch_dispatch.rb interacts with local slurm and git repo (specific to crunchv1)
>
> commit_ancestor.rb and commit.rb read a local git repo (specific to crunchv1)
>
> job.rb touches the file at crunch_refresh_trigger (specific to crunchv1)

Multiple API hosts will not support crunchv1.

> The nodes table updates local DNS configuration. Used by node manager, could technically be used by on-prem configuration but I don't think

anyone does.

Multiple API hosts is incompatible with "classic" node management (node manager) where ping scripts and the nodes table triggers DNS updates.

Should be fine using either crunch-dispatch-slurm with a externally managed / static slurm cluster, or using crunch-dispatch-cloud.

Websocket events are triggered by database NOTIFY.

The discovery document is stored in local cache. There is a generatedAt field which is db_current_time, which means otherwise identical servers on different hosts are likely to report different generateAt times.

I don't think there is anything to do here.

Login process uses a session. I'm not sure if this is necessary, or what the implications are if the user starts a session on one host and the next request goes to a different host.

I need to research this some more.

#10 - 03/15/2019 07:46 PM - Peter Amstutz

There's some code from 2013 that sets values in the session, but I can't find anything that reads the session. So the session is probably irrelevant.

#13 - 03/18/2019 06:47 PM - Peter Amstutz

- Blocked by Story #14987: [API] Upgrade to Rails 5 added

#14 - 03/18/2019 06:47 PM - Peter Amstutz

- Blocked by Story #14873: [API] Update to Rails 5 added

#15 - 03/18/2019 06:47 PM - Peter Amstutz

- Blocked by deleted (Story #14987: [API] Upgrade to Rails 5)

#16 - 03/27/2019 02:24 PM - Peter Amstutz

- Target version changed from 2019-03-27 Sprint to 2019-04-10 Sprint

#17 - 03/27/2019 03:43 PM - Tom Morris

- Target version changed from 2019-04-10 Sprint to Arvados Future Sprints

#18 - 03/27/2019 03:43 PM - Peter Amstutz

- Assigned To deleted (Peter Amstutz)