Arvados - Story #3036

[API] Use regular uuids instead of content hashes to identify collections

06/17/2014 01:48 PM - Tom Clegg

Status:	Resolved	**Start date:**	08/07/2014
Priority:	Normal	**Due date:**	
Assigned To:	Peter Amstutz	**% Done:**	100%
Category:	API	**Estimated time:**	0.00 hour
Target version:	2014-08-27 Sprint		

Description

Summary

Collections should have system-assigned UUIDs that look like other Arvados UUIDs; should be mutable; and should have a "name" attribute.

Background (current behavior)

Collections are content-addressed: they have uuid = hash(manifest_text) and they are immutable. This makes them behave differently than all other objects in Arvados, which tends to be confusing and awkward.

- Feature: This makes it possible to do a bitwise comparison of collections (e.g., job outputs) without even looking up the collection itself.
- Feature: This (partially) de-duplicates manifest storage: a given manifest is only stored once in the collections table. But this does not survive tiny changes like renaming a file within a collection.
- Feature: Collection metadata (manifests) cannot be deleted by users. Even if the content data has been deleted, a superuser can still see the filenames and sizes for every collection ever made. (Not clear whether this feature is valuable, though.)
- Drawback: Applications must create Link objects (link_class="name") in order to attach names to collections (analogous to file/directory names in regular filesystems). This API is unwieldy.
- Drawback: The default permission model -- i.e., the creator of an object has permission to read/edit/delete it -- cannot be achieved using the owner_uuid attribute. In order to provide a predictable outcome for "create a collection" regardless of whether another user has already created an identical collection, we are forced to give all collections owner_uuid=root and create a "permission" link for each collection creation. We also have to synchronize "name" links with "permission" links in order to achieve reasonable behavior for users.
- Drawback: The timestamps of collections are confusing. If I create a new collection when (unbeknownst to me) another user has already created one with identical content some days ago, the "created" and "updated" timestamps and similar metadata will be surprising and generally useless (except perhaps as an undesirable information leak).

New behavior

- Collections are mutable, and have a name attribute.
- Look up the hash of a collection's manifest when you want to do a bitwise comparison of content.

(Certainly incomplete) list of changes/consequences:

- First step: allow clients to call collections.create without providing a uuid. (merged in 5bbd6abc)
- Update uuid→class regexps to accept collection uuids in the usual arvados uuid format as well as portable_data_hashes.
- Copy current uuid values to portable_data_hash
- If clients provide portable_data_hash to collections.create, verify that as uuid is verified now (i.e., compare it to the portable_data_hash computed from the provided (stripped) manifest, and respond 422 if it doesn't match). Skip this check if no portable_data_hash provided by client.
- Fix clients so they pass the expected portable_data_hash instead of uuid (or pass neither) and use the uuid provided by Arvados, rather than assuming the new collection's uuid will be a content address.
- Add usual mutable fields like "name", "description", and "properties" to the collections table.
- Remove "all collections are owned by root" logic.
- Remove "add a permission link for me after creating a collection" logic.
- Update Workbench to use collections' "name" attributes instead of name links.
- Migrate existing name links in the database to become new collections.

Looking up collections by portable data hash

Existing workbench links with old collection UUIDs should still work. Crunch jobs (new ones and repetitions of old ones) should continue to use portable data hashes.

- Look up by portable_data_hash if collections.get called with old format collection UUID, and redact the mutable fields.
- Jobs' script_parameters should be filled in with portable data hashes rather than collection UUIDs. Pipelines will record both fields (UUID and portable data hash) much like they do now with link_uuid, link_name, and value keys. (See Workbench application_helper.rb)

Subtasks:

Task # 3502: Document expectations for content addressing	**Resolved**
Task # 3503: Document intended behavior and design for projects, collections and permis...	**Resolved**
Task # 3581: Update workbench	**Resolved**
Task # 3509: Links to collections in workbench render the "name" field sensibly	**Resolved**
Task # 3579: Update test fixtures	**Resolved**
Task # 3632: Review 3036-collection-uuids	**Resolved**
Task # 3580: Update tests	**Resolved**
Task # 3578: Write db migration	**Resolved**

Related issues:

Related to Arvados - Bug #3024: [API] Synchronize read permissions and collec...	**Resolved**	
Related to Arvados - Story #3504: [SDKs] Clients are compatible with #3036	**Resolved**	**08/07/2014**
Related to Arvados - Bug #4756: [API] Add migration to change collection uuid...	**Rejected**	**12/09/2014**

Associated revisions

Revision 61cd5749 - 08/26/2014 10:45 AM - Peter Amstutz

Merge branch '3036-collection-uuids' closes #3036

Revision d5835c04 - 08/26/2014 11:14 AM - Tom Clegg

Fix migration and api templates that rely on changes that did not end up happening. refs #3036

Revision 94982cd8 - 09/02/2014 04:00 PM - Peter Amstutz

Delete names and description columns from jobs that shouldn't be there. Delete
jobs_owner_uuid_name_unique and pipeline_instance_owner_uuid_name_unique
indexes added by mistake. refs #3036.

Revision 81ead9fd - 09/02/2014 04:35 PM - Peter Amstutz

Add name and description columns back in for jobs. refs #3036

Revision 5184743d - 09/04/2014 09:10 PM - Peter Amstutz

Remove name and description columns on jobs table introduced accidentally
(refs #3036). Fixed workbench so tests pass.

Revision 8012f6cc - 09/11/2014 03:01 PM - Ward Vandewege

Fallout from #3036: the test for the checkbox value on acceptance of the
user agreement was expecting a keep hash, rather than an Arvados UUID.

This fixes accepting the user agreement (if one or more are present).

refs #3036

Revision 469356d1 - 05/31/2015 12:39 PM - Tom Clegg

Remove non-existent migration from structure.sql. refs #3036

History

#1 - 06/17/2014 01:59 PM - Tom Clegg

- Description updated

#2 - 07/11/2014 04:42 PM - Tom Clegg

- Target version set to 2014-08-06 Sprint

#3 - 07/11/2014 04:45 PM - Tom Clegg

- Subject changed from Use regular uuids instead of content hashes to identify collections to [API] Use regular uuids instead of content hashes to identify collections

#4 - 07/11/2014 05:38 PM - Tom Clegg

- Description updated

- Category set to API

#5 - 07/16/2014 03:22 PM - Tom Clegg

- Target version changed from 2014-08-06 Sprint to Arvados Future Sprints

#6 - 07/17/2014 10:00 AM - Tom Clegg

- Subject changed from [API] Use regular uuids instead of content hashes to identify collections to [API] [Draft] Use regular uuids instead of content hashes to identify collections

- Description updated

#7 - 07/17/2014 12:47 PM - Tom Clegg

- Description updated

#8 - 07/17/2014 12:48 PM - Tom Clegg

- Description updated

#9 - 07/17/2014 12:56 PM - Tom Clegg

- Description updated

#10 - 07/17/2014 12:59 PM - Tom Clegg

- Description updated

#11 - 07/31/2014 05:13 PM - Tom Clegg

- Target version changed from Arvados Future Sprints to 2014-08-27 Sprint

#12 - 08/05/2014 10:29 PM - Peter Amstutz

How does this interact with crunch? Do they continue to use manifest hashes, or switch to using uuids?

#13 - 08/06/2014 03:28 PM - Peter Amstutz

- Assigned To set to Peter Amstutz

#14 - 08/06/2014 03:30 PM - Tom Clegg

- Subject changed from [API] [Draft] Use regular uuids instead of content hashes to identify collections to [API] Use regular uuids instead of content hashes to identify collections

- Description updated

#15 - 08/08/2014 09:23 AM - Tom Clegg

- Description updated

#16 - 08/12/2014 02:30 PM - Peter Amstutz

- Status changed from New to In Progress

#17 - 08/20/2014 02:09 PM - Tom Clegg

At d08c3a5...

sdk/python/arvados/commands/put.py

- I think it would be neater to provide owner_uuid up front in the initial collections().create() call (I think this will work equally well before and after #3036, unlike the name attribute).

sdk/python/tests/test_arv_put.py

- I know this seems trivial but please update indentation when this sort of thing happens (3 occurrences here):

```
-        link = self.run_and_find_link("Test unnamed collection",
+        link = self.run_and_find_collection("Test unnamed collection",
                                            ['--project-uuid', self.PROJECT_UUID])
```

services/api/app/controllers/application_controller.rb

- While we're at it, couldn't this be reduced to

```
-    if (@object and @object.respond_to? :errors and
-        @object.errors and @object.errors.full_messages and
-        not @object.errors.full_messages.empty?)
+    if (@object.respond_to? :errors and
+        @object.errors.andand.full_messages.andand.any?)
```

(ok, now we're getting to the fun part)

services/api/app/controllers/arvados/v1/collections_controller.rb

- Shouldn't this be done with a model validation? That would protect all create/update operations, rather than just ones that come through CollectionsController#create, and reporting would be more consistent. (Bypassing render_error and going directly to send_error seems especially snowflakey.) I suspect the only reason we used to do all this work on resource_attrs, instead of doing this work in the model, is that act_as_system_user makes the model think everyone's an admin. Now that we don't need this, should we move the "check signatures" stuff into a validation on the Collection model?

```
+    if !resource_attrs[:manifest_text]
+      return send_error("'manifest_text' attribute must be specified",
+                        status: :unprocessable_entity)
+    end
```

  - (One way or another, we'd better check signatures during #update too, if manifest_text_changed? -- I don't see that here yet.)
- Ah, so nice to see that act_as_system_user block in #create disappear into history.
- CollectionsController#find_objects_for_index seems like the wrong place to do something that's only needed by #update. There's now a cleaner solution to this general column dependency problem in ApplicationController.apply_where_limit_order_params. Perhaps it's reasonable to put "always select the :id column" in there with a comment about the future headaches that strategy is likely to avoid.

```
+    if @select.nil?
+      @select = model_class.api_accessible_attributes(:user).map { |attr_spec| attr_spec.first.to_s }
+      @select -= ["manifest_text"]
+      # have to make sure 'id' column is included or #update will break.
+      @select += ["id"]
     end
```

services/api/app/helpers/collections_helper.rb

- New stripped_portable_data_hash method seems to be a re-implementation of Locator.parse!(uuid).strip_hints.to_s and isn't used by anything. Remove?

services/api/app/controllers/arvados/v1/groups_controller.rb

- Add a comment near include_linked in _index_requires_parameters scheduling it for deletion, so it's not just a mystery why it's listed but not used.

services/api/app/controllers/arvados/v1/jobs_controller.rb

- I noticed this new code issues a separate find() query for each result. Then I noticed new code in uuids_for_docker_image itself issues a separate find() query for each result. Then I noticed the next move made by each of the (now) two callers of uuids_for_docker_image is to issue one or more find() queries on the results. If that's what the callers want, then I suggest renaming uuids_for_docker_image to find_all_for_docker_image having it return the collections instead of just the uuids. So, instead of doing this in jobs_controller:

```
          Collection.uuids_for_docker_image(image_search, image_tag, @read_users).map do |uuid|
            Collection.find_by_uuid(uuid).portable_data_hash
          end
```

- ...could we do this in Collection.find_all_for_docker_image()?

```
    matches = Collection.where('uuid in (?)', matches)
    matches.sort_by! do |collection|
      ...
    end
```

services/api/app/models/arvados_model.rb

- Why is it necessary to prevent subclasses from overriding resource_class_for_uuid as used here?

```
-       while (owner_class = self.class.resource_class_for_uuid(x)) != User
+       while (owner_class = ArvadosModel::resource_class_for_uuid(x)) != User
```

  - (Another occurrence in ensure_owner_uuid_is_permitted)
- Error message "can only be set to User or Group" → more direct "must be User or Group"?

services/api/app/models/collection.rb

- Collection.new.valid? crashes because manifest_text is nil during set_portable_data_hash. If it weren't for that, I think ensure_hash_matches_manifest_text would report true because neither attribute has changed. If both attributes are nil, I think it would be a bit neater to
  - not crash before_validation, and
  - return false from ensure_hash_matches_manifest_text (or a separate validation) when manifest_text is nil.

services/api/app/models/link.rb

- Update indent to suit new code:

```
     if link_class == 'name'
-      unless name.is_a? String and !name.empty?
-        errors.add('name', 'must be a non-empty string')
-      end
+        errors.add('name', 'Name links are obsolete')
     else
```

services/api/app/views/

- Thanks for cleaning up the unused views.

services/api/db/migrate/20140811184643_collection_use_regular_uuids.rb

- Suggest expires_at instead of expire_time to be consistent with our other timestamp columns.
- Down-migration should admit failure instead of wedging your database into a state where it can't migrate back up again, either. You want this:

```
  def down
    raise ActiveRecord::IrreversibleMigration, "Explain why its irreversible!"
  end
```

services/api/db/migrate/20140815171049_add_name_description_columns.rb

- There's another story on this sprint (#2875) that adds description to PipelineInstance. Hiding this migration in a big commit in a long-lived branch can create unnecessary merging/backporting awkwardness. (I put a note on #2875 which should be enough to avert some extra work.)

services/api/lib/has_uuid.rb

- This produces messages like "uuid Not permitted to specify uuid". Change to something like (:uuid, "assignment not permitted") / "change not permitted" ...?

```
+        if self.new_record?
+          self.errors.add(:uuid, "Not permitted to specify uuid")
+        else
+          self.errors.add(:uuid, "Not permitted to change uuid")
+        end
```

- This message is still a bit wonky too. Perhaps: "has type segment '#{re[1]}', expected [...]" ...?

```
+            self.errors.add(:uuid, "Matched uuid type '#{re[1]}', expected '#{self.class.uuid_prefix}'
")
```

Posting this so I'm not keeping you waiting. Review still todo:

- Review big migration

- Review the tests
- Run migrations
- Run tests

Thanks!

#18 - 08/22/2014 04:59 PM - Peter Amstutz

- Status changed from In Progress to New

Tom Clegg wrote:

> At d08c3a5...
>
> sdk/python/arvados/commands/put.py
>
> - I think it would be neater to provide owner_uuid up front in the initial collections().create() call (I think this will work equally well before and after #3036, unlike the name attribute).

Fixed

> sdk/python/tests/test_arv_put.py
>
> - I know this seems trivial but please update indentation when this sort of thing happens (3 occurrences here):
>   - [...]

Fixed

> services/api/app/controllers/application_controller.rb
>
> - While we're at it, couldn't this be reduced to
>   - [...]

Fixed

> services/api/app/controllers/arvados/v1/collections_controller.rb
>
> - Shouldn't this be done with a model validation? That would protect all create/update operations, rather than just ones that come through CollectionsController#create, and reporting would be more consistent. (Bypassing render_error and going directly to send_error seems especially snowflakey.) I suspect the only reason we used to do all this work on resource_attrs, instead of doing this work in the model, is that act_as_system_user makes the model think everyone's an admin. Now that we don't need this, should we move the "check signatures" stuff into a validation on the Collection model?
>   - [...]
>   - (One way or another, we'd better check signatures during #update too, if manifest_text_changed? -- I don't see that here yet.)
> - Ah, so nice to see that act_as_system_user block in #create disappear into history.

Fixed.

> - CollectionsController#find_objects_for_index seems like the wrong place to do something that's only needed by #update. There's now a cleaner solution to this general column dependency problem in ApplicationController.apply_where_limit_order_params. Perhaps it's reasonable to put "always select the :id column" in there with a comment about the future headaches that strategy is likely to avoid.

Fixed.

> services/api/app/helpers/collections_helper.rb
>
> - New stripped_portable_data_hash method seems to be a re-implementation of Locator.parse!(uuid).strip_hints.to_s and isn't used by anything. Remove?

Removed.

> services/api/app/controllers/arvados/v1/groups_controller.rb
>
> - Add a comment near include_linked in _index_requires_parameters scheduling it for deletion, so it's not just a mystery why it's listed but not used.

Added comment.

services/api/app/controllers/arvados/v1/jobs_controller.rb

- I noticed this new code issues a separate find() query for each result. Then I noticed new code in uuids_for_docker_image itself issues a separate find() query for each result. Then I noticed the next move made by each of the (now) two callers of uuids_for_docker_image is to issue one or more find() queries on the results. If that's what the callers want, then I suggest renaming uuids_for_docker_image to find_all_for_docker_image having it return the collections instead of just the uuids. So, instead of doing this in jobs_controller:

Refactored.

services/api/app/models/arvados_model.rb

- Why is it necessary to prevent subclasses from overriding resource_class_for_uuid as used here?
  - [...]
  - (Another occurrence in ensure_owner_uuid_is_permitted)

Fixed.

- Error message "can only be set to User or Group" → more direct "must be User or Group"?

Fixed.

services/api/app/models/collection.rb

- Collection.new.valid? crashes because manifest_text is nil during set_portable_data_hash. If it weren't for that, I think ensure_hash_matches_manifest_text would report true because neither attribute has changed. If both attributes are nil, I think it would be a bit neater to
  - not crash before_validation, and
  - return false from ensure_hash_matches_manifest_text (or a separate validation) when manifest_text is nil.

Fixed.

services/api/app/models/link.rb

- Update indent to suit new code:
  - [...]

Fixed.

services/api/db/migrate/20140811184643_collection_use_regular_uuids.rb

- Suggest expires_at instead of expire_time to be consistent with our other timestamp columns.
- Down-migration should admit failure instead of wedging your database into a state where it can't migrate back up again, either. You want this:
  - [...]

Fixed

services/api/db/migrate/20140815171049_add_name_description_columns.rb

- There's another story on this sprint (#2875) that adds description to PipelineInstance. Hiding this migration in a big commit in a long-lived branch can create unnecessary merging/backporting awkwardness. (I put a note on #2875 which should be enough to avert some extra work.)

First one to merge wins?

services/api/lib/has_uuid.rb

- This produces messages like "uuid Not permitted to specify uuid". Change to something like (:uuid, "assignment not permitted") / "change not permitted" ...?
  - [...]
- This message is still a bit wonky too. Perhaps: "has type segment '#{re[1]}', expected [...]" ...?
  - [...]

Fixed.

Posting this so I'm not keeping you waiting. Review still todo:

- Review big migration
- Review the tests
- Run migrations
- Run tests

Still waiting on these?

#19 - 08/25/2014 04:37 PM - Peter Amstutz

- Status changed from New to In Progress

#20 - 08/25/2014 04:43 PM - Peter Amstutz

I'm thinking about backing out the owner/name uniqueness for jobs and pipeline instances. It seems typical that the name of a pipeline instance or job will be copied from the template/component/run script and will yield many similarly named jobs or instances; making them unique by adding a timestamp isn't very interesting when there is already a real timestamp field.

#21 - 08/25/2014 04:51 PM - Tom Clegg

Peter Amstutz wrote:

> - There's another story on this sprint (#2875) that adds description to PipelineInstance. Hiding this migration in a big commit in a long-lived branch can create unnecessary merging/backporting awkwardness. (I put a note on #2875 which should be enough to avert some extra work.)
>
> First one to merge wins?

Better communication → everybody wins :P

All of the changes look good, thanks.

> Posting this so I'm not keeping you waiting. Review still todo:
>
> - Review big migration
> - Review the tests
> - Run migrations
> - Run tests
>
> Still waiting on these?

Meh, don't really want to hold up the merge for that. I say go ahead. Thanks!

#22 - 08/26/2014 10:50 AM - Anonymous

- Status changed from In Progress to Resolved

- % Done changed from 85 to 100

Applied in changeset arvados|commit:61cd57499905e8e8cca07c774d1bf8c6bfa069a7.