Arvados - Story #2755

Implement Keep permission signatures in API server and Python SDK

05/07/2014 12:24 PM - Tom Clegg

Status:	Resolved	**Start date:**	05/14/2014
Priority:	Normal	**Due date:**	
Assigned To:	Tim Pierce	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	2014-06-17 Curating and Crunch		

Description

Phase 1

- Collections.create() - verify permission signatures in provided manifest_text. Strip them -- and all other +whatever hints except one size hint -- before verifying uuid==hash(manifest_text) and storing manifest_text in database.
  - Pass signature verification step (until Phase 4) if a blob locator is missing the permission signature entirely.
- Collections.get() - return a manifest_text with a +A... permission signature added to each blob locator.

(Phase 1 can be deployed any time now.)

Phase 2

- Python SDK, when writing a collection,
  - Stop throwing away the +A signatures that (might) emanate from the keep servers during Keep.put().
  - In arv-put, include the +A signatures in the manifest_text when sending to server.
  - In arv-put, compute collection uuid based on a version of manifest_text with the +A signatures (and all other +anything other than +size) stripped off.
  - For good form, when doing collections.create() in arv-put, ensure the UUID returned by API server matches the one you sent.
- Python SDK, when reading a blob,
  - Set "Authorization: OAuth2 $ARVADOS_API_TOKEN" header in http requests to Keep servers.

(Phase 2 package can be published any time now.)

Phase 3

- Deploy Keep server with signature *generation* feature enabled.
- Test old and new Python clients.

Phase 4

Upgrade all python SDKs/clients first. Then:

- Remove "no signature provided" exemption from API server.
- Enable signature *verification* on keep servers.

Subtasks:

Task # 2784: Generate cooked manifests	**Resolved**
Task # 2927: Review 2755-python-sdk-permissions (support for manifests)	**Resolved**
Task # 2825: Review 2755-api-collection-permissions	**Resolved**
Task # 2786: Make Python SDK handle signatures and cooked manifests correctly	**Resolved**
Task # 2787: Accept manifests with signature tokens during collections.create	**Resolved**
Task # 2859: Review 2755-python-sdk-permissions	**Resolved**
Task # 3007: Review 2755-require-keep-permission	**Resolved**

Associated revisions

Revision 278fe704 - 05/15/2014 05:16 PM - Tom Clegg

2755: Defer to CollectionReader to get manifests instead of going directly to Keep (which only works when Keep has no permission checks).

refs #2755

Revision aad9cd74 - 05/19/2014 05:54 PM - Tim Pierce

2755: Verify permission signatures on create.

Phase 1 of #2755: when creating a new collection, verify any permission signatures found in the manifest. Unsigned locators in the manifest are implicitly permitted (to be disabled in Phase 4)

- New "Locator" class to parse, examine and manipulate Keep locators.
- Collections.create checks permission signatures in a manifest.
- Collections.show signs locators in a manifest.
- collections_controller_test.rb: new unit tests to exercise signed manifests and related features:
  - "create collection with signed manifest"
  - "create collection with signed manifest and explicit TTL"
  - "create fails with invalid signature"
  - "create fails with uuid of signed manifest"
  - "multiple locators per line"
  - "multiple signed locators per line"
- application.yml.example: new configuration variables
  - Rails.configuration.blob_signing_key
  - Rails.configuration.blob_signing_ttl

(refs #2755)

Revision 66d5eced - 05/19/2014 06:17 PM - Tim Pierce

2755: fix merge conflicts (refs #2755)

Revision 1f43dd85 - 05/21/2014 03:51 PM - Tim Pierce

Merge branch '2755-api-collection-permissions' of git.curoverse.com:arvados into 2755-api-collection-permissions

Refs #2755

Conflicts:
services/api/app/controllers/arvados/v1/collections_controller.rb
services/api/config/application.default.yml
services/api/config/application.yml.example
services/api/test/functional/arvados/v1/collections_controller_test.rb

Revision 64d339fa - 05/21/2014 03:54 PM - Tim Pierce

2755: fix merge conflicts (refs #2755)

Revision 25bdeb97 - 05/21/2014 03:54 PM - Tim Pierce

2755: fix blob signing bug. (refs #2755)

Revision 1b3750c9 - 05/21/2014 03:54 PM - Tim Pierce

Merge branch 'master' of git.curoverse.com:arvados

Refs #2755.

Revision 52145737 - 05/23/2014 05:25 PM - Tim Pierce

2755: add support for signed locators in the Python SDK.

- arvados.Keep.put() saves the response body (which may contain a signed locator) and returns it to the caller.
- arvados.Keep.get() passes the full signed locator to the remote Keep server. The bare MD5 hash is still used for caching and for shuffled_service_roots
- run_test_server.run_keep() takes arguments 'blob_signing_key' and 'enforce_permissions', for testing permission signatures in unit tests.
- test_keep_client: new unit tests for permissions:
  - with --enforce-permissions=true:
  - GET with a signed locator works

- GET with an unsigned locator fails
- unauthenticated GET fails
- with --enforce-permissions=false:
- GET with a signed locator works
- GET with an unsigned locator works
- unauthenticated GET works

Bug fixes to permission handling in the Keep server:

- Locator hints may appear in any order; be flexible. Parse them in GetBlockHandler rather than in the REST router.
- Returned locators are terminated with newline (consistent with Warehouse, and more friendly for human debugging).
- The locator returned from a PUT request always has a size hint.
- The correct Authorization header keyword is "OAuth2", not "OAuth". D'oh.
- Updated unit tests to accommodate newlines, size hints and OAuth2.

Refs #2755.

Revision d269c98c - 05/24/2014 12:44 AM - Tim Pierce

2755: incorporate code review.

- Unit tests cover all permutations of signature/authorization when --enforce-permissions=false
- Keep is more forgiving about the structure of locators, permits locator hints of unknown type (as long as they begin with an uppercase letter)
- Keep delivers 400 Bad Request for requests that do not match any route, or are lexically invalid. 404 Not Found only for requests with a syntactically valid hash not found on disk.

Refs #2755.

Revision 010a56c1 - 05/27/2014 11:06 AM - Tim Pierce

Merge branch '2755-python-sdk-permissions'

Refs #2755.

Revision 46460c96 - 05/28/2014 06:12 PM - Tim Pierce

2755: add support to arv-put for signed manifests.

When arv-put finishes a stream, the manifest it stores in Keep now has been stripped of signatures and other variable hints.

test_cmdline.py tests arv-put's handling of the manifest to make sure that, when permissions are enabled, the manifest in Keep lacks signatures, and the same manifest returned from the API server includes signatures.

Refs #2755.

Revision 39b2ed30 - 05/30/2014 02:42 PM - Tim Pierce

Merge remote-tracking branch 'refs/remotes/origin/2755-python-sdk-permissions-TC' into 2755-python-sdk-permissions

Refs #2755.

Revision b6ea1fe3 - 05/30/2014 03:27 PM - Tim Pierce

Merge branch '2755-python-sdk-permissions'

Closes #2755.

Revision 4d84c7d2 - 06/12/2014 03:21 PM - Tom Clegg

History

#1 - 05/07/2014 12:24 PM - Tom Clegg

- Target version set to 2014-05-28 Pipeline Factory

#2 - 05/07/2014 01:47 PM - Tom Clegg

- Description updated

#3 - 05/07/2014 04:12 PM - Tom Clegg

- Assigned To set to Tom Clegg

#4 - 05/07/2014 04:17 PM - Tom Clegg

- Assigned To changed from Tom Clegg to Tim Pierce

#5 - 05/12/2014 03:23 PM - Tom Clegg

- Description updated

#6 - 05/14/2014 07:02 PM - Tom Clegg

Comments @ b12f667

Add to test cases:

- More than one blob on one line of a manifest
- Blob locator like hash+size+K@foo+Asig@time
- Blob locator like hash+size+Asig@time+K@foo
- When testing resp['manifest_text'], make sure you found the expected number of locators/signatures
- Test that collections.create fails when given a bad signature

Config

- Add "blob_signing_ttl" config. 2 weeks would be a reasonable default (until clients get smart enough to refresh their sigs, this will essentially put an upper bound on crunch job duration).
- Consider renaming "permission_key" config to something more descriptive, perhaps "blob_signing_key"? ("Secret_token" is vague too, but at least we can blame that on Rails.)
- Put config docs and defaults in application.default.yml. Leave application.yml.example empty except the "at minimum, you need" part as a reminder during initial install.
- Leave default=nil instead of random string in development section. (Random key can cause confusing behavior, which is worse than a one-time interruption "can't start server until you fix config".)
- (Basically, should probably do exactly the same thing secret_token does in config files, but without duplicating the "no explanation given for secret_token in application.default.yml" bug.)

#7 - 05/15/2014 12:22 AM - Tim Pierce

Tom Clegg wrote:

> Comments @ b12f667
>
> Add to test cases:
>
> - More than one blob on one line of a manifest
> - Blob locator like hash+size+K@foo+Asig@time
> - Blob locator like hash+size+Asig@time+K@foo

I didn't realize those were both legal. In the interest of establishing explicit rules for parsing legal blob locators, I hereby define the following BNF grammar. Please correct this if I have it wrong, otherwise I'll add it to the docs and use it as my guide here.

```
locator       ::= hash option*

hash          ::= digest size-hint

digest        ::= <32 hexadecimal digits>

size-hint     ::= "+" [0-9]+

option        ::= "+A" signature "@" timestamp | "+K@" location-hint

signature     ::= <alphanumeric>+
```

```
timestamp     ::= <8 hexadecimal digits>

location-hint ::= <alphanumeric>+
```

#8 - 05/16/2014 06:34 PM - Tom Clegg

"option" (which we've always called "hint" before, and maybe should continue? maybe not?) is an extensible facility analogous to HTTP headers. Blob-signing and signature-verification code needs to understand the structure of a +A hint but other code should not presume anything beyond the general form of a hint.

How about:

```
locator       ::= sized-digest hint*

sized-digest  ::= digest size-hint

digest        ::= <32 hexadecimal digits>

size-hint     ::= "+" [0-9]+

hint          ::= "+" hint-type hint-content

hint-type     ::= [A-Z]+

hint-content  ::= [^+,\s\000A-Z][^+,\s\000]*

sign-hint      ::= "+A" <40 lowercase hex digits> "@" sign-timestamp

sign-timestamp ::= <8 lowercase hex digits>
```

Caveats

- It seems confusing to use the term "hash" to mean "digest+size" but I'm not sure offhand what else to call it. It's surely a special enough case of "locator" to deserve some name. I'm not super excited about "sized-digest" either. Suggestions?
- We should surely restrict hint-content further than this. How far? "Printable" chars, minus + and ,? Reuse some compatible allowed-chars set, like URL chars?

#9 - 05/19/2014 09:37 AM - Tim Pierce

Tom Clegg wrote:

> "option" (which we've always called "hint" before, and maybe should continue? maybe not?) is an extensible facility analogous to HTTP headers. Blob-signing and signature-verification code needs to understand the structure of a +A hint but other code should not presume anything beyond the general form of a hint.

Sure, "hint" is okay as general terminology. I was uncomfortable with "permission hints" because with permissions, it's really not a hint, it's a binding requirement. But somehow I don't mind the idea that a locator hint can include a permission signature. Oh, language, you so kooky.

> Caveats
>
> - It seems confusing to use the term "hash" to mean "digest+size" but I'm not sure offhand what else to call it. It's surely a special enough case of "locator" to deserve some name. I'm not super excited about "sized-digest" either. Suggestions?

"Address"? This section of the locator is literally the address of the block within the space of possible MD5 hash strings.

> - We should surely restrict hint-content further than this. How far? "Printable" chars, minus + and ,? Reuse some compatible allowed-chars set, like URL chars?

Based on our current usage, I suggest that a hint is limited to alphanumerics, "@", "_" and "-":

```
hint-content ::= [A-Za-z0-9@_-]+
```

If we define a hint that needs for some reason to use additional punctuation, it can base64-encode the content. But I think we're better off starting with as conservative a definition as we can, to decrease the complexity of parsing locators from manifests and other text files.

#10 - 05/19/2014 06:18 PM - Tim Pierce

Changes @ 66d5ece:

- Added tests:
  - create collection with signed manifest

    - create collection with signed manifest and explicit TTL
    - create fails with invalid signature
    - create fails with uuid of signed manifest
    - multiple locators per line
    - multiple signed locators per line
- Added Locator class with support for hints in any order
- Configuration settings:
    - blob_signing_ttl (default 2 weeks)
    - blob_signing_key (was permission_key)

#11 - 05/20/2014 06:33 PM - Tom Clegg

at 66d5ece

- If you move locator.rb to app/models it will get loaded automatically and you can lose the require.
- Locator.parse! is too demanding about hints. It should not presume to understand what +A and +K hints look like, or whether kinds other than A and K exist. According to the above definition it should be enough to check:
    - split on "+"
    - first token is 32 hex digits
    - next token if any is just decimal digits
    - additional tokens if any start with A-Z
- Instead of rescuing all unforeseen exceptions in Locator.parse(), just rescue the one you expect.
- Remove blob_signing_ttl stuff from application.yml.example.
- Might want to put a space at the beginning of this regexp, to ensure it doesn't match filenames:

```
+    # Remove any permission signatures from the manifest.
+    resource_attrs[:manifest_text]
+      .gsub!(/[[:xdigit:]]{32}(\+[[:digit:]]+)?(\+\S+)/) { |word|
```

#12 - 05/21/2014 01:38 PM - Tim Pierce

Changes at 2bdd464

- lib/locator.rb renamed => app/models/locator.rb
- Relaxed Locator.parse! handling of hint content.
- Locator.parse() rescues only from ArgumentError.
- Removed blob_signing_ttl from application.yml.example.
- Collections.show only matches locators that are preceded by a space, when parsing manifest_text.

#13 - 05/21/2014 02:12 PM - Tom Clegg

Looks great, thanks!

#14 - 05/23/2014 06:01 PM - Tom Clegg

Comments @ 521457373

This should probably say "without a signature":

```
+        # With Keep permissions enabled, a GET request without a locator will fail.
```

This suggests it's going to test "signature yes, token no" but the rest of the method seems to be a duplicate of the test immediately above ("signature no, token yes"):

```
+    def test_KeepUnauthenticatedTest(self):
+        # Since --enforce-permissions is not in effect, GET requests
+        # need not be authenticated.
```

Server should ignore correctly formatted hints that do not alter its behavior (i.e., everything except +A). If each hint starts with an uppercase letter, parsing worked. (Adding a +Foo hint shouldn't require upgrading all Keep services, or teaching all clients/SDKs to keep track of which services need to have which hints redacted...)

```
+                        } else {
+                                // Not a valid locator: return 404
+                                http.Error(resp, NotFoundError.Error(), NotFoundError.HTTPCode)
+                                return
```

Also: if the locator/hints are truly not parseable, the response should be something like "400 bad request" rather than "404 not found".

This also looks like it's too picky about other hints that it should be impervious to. Perhaps .*\+A([0-9a-f]{40})@([0-9a-f]{8}) and lose the $ anchor:

```
+       if re, err := regexp.Compile(`^([a-f0-9]+(\+[0-9]+)?)\+A(.*)@(.*)$`); err == nil {
```

#15 - 05/24/2014 12:49 AM - Tim Pierce

Updated @ d269c98

Tom Clegg wrote:

> Comments @ 521457373
>
> This should probably say "without a signature":
>
> ```
> +        # With Keep permissions enabled, a GET request without a locator will fail.
> ```

Oops, you're right. Fixed.

> This suggests it's going to test "signature yes, token no" but the rest of the method seems to be a duplicate of the test immediately above ("signature no, token yes"):
>
> ```
> +    def test_KeepUnauthenticatedTest(self):
> +        # Since --enforce-permissions is not in effect, GET requests
> +        # need not be authenticated.
> ```

It's "signature no, token no" -- note that it deletes the ARVADOS_API_TOKEN from arvados.config.settings() before attempting the GET. I've added tests for all four permutations of signature/authentication.

> Server should ignore correctly formatted hints that do not alter its behavior (i.e., everything except +A). If each hint starts with an uppercase letter, parsing worked. (Adding a +Foo hint shouldn't require upgrading all Keep services, or teaching all clients/SDKs to keep track of which services need to have which hints redacted...)
>
> [...]

That makes sense. Modified the parser to accept any unrecognized hint that starts with an uppercase letter.

> Also: if the locator/hints are truly not parseable, the response should be something like "400 bad request" rather than "404 not found".

Here's what Keep does at

- 400 Bad Request for lexical/syntax errors:
  - Request does not match any known routes
  - Hash does not match [hexdigit]{32}
  - Any hint does not match ^[0-9]+$ or ^[A-Z][a-z0-9@_-]+$
- 404 Not Found:
  - Only for a lexically valid hash, with permission, that can't be found on disk

> This also looks like it's too picky about other hints that it should be impervious to. Perhaps .*\+A([0-9a-f]{40})@([0-9a-f]{8}) and lose the $ anchor:
>
> [...]

Updated. It needs to be a *little* picky -- it has to be mindful of the presence of size hints -- but it should certainly allow the signature to appear anywhere subsequently in the locator.

#16 - 05/27/2014 10:15 AM - Tom Clegg

Tim Pierce wrote:

> It's "signature no, token no" -- note that it deletes the ARVADOS_API_TOKEN from arvados.config.settings() before attempting the GET. I've added tests for all four permutations of signature/authentication.

Indeed, I missed that, saw authorize_with -- but that only lasts long enough to do the PUT, as you say.

Everything else seems good now. Thanks.

#17 - 05/29/2014 10:59 AM - Tom Clegg

at 46460c96

- Don't strip all hints, just permission hints.
- The caching mechanism introduced to CollectionWriter.manifest_text() never gets invalidated, so data will be mysteriously lost if a client does "write stuff; get manifest; write more stuff; get manifest". Either reset _manifest_text in all the methods that might invalidate it, or (my preference) stay safe and simple, and don't even bother caching it.
- If you do keep the cache, I'd rather use None than an empty string to indicate "nothing cached".

Changes to services/api/config/application.yml.example seem misplaced on this branch, but plausible, and doesn't seems to break anything. (Did the existing commenting start causing trouble somewhere, or is this a human-readability improvement?)

#18 - 05/29/2014 11:20 AM - Tim Pierce

Tom Clegg wrote:

> at 46460c96
>
> - Don't strip all hints, just permission hints.

FYI, this contradicts the requirement in the story, which is "In arv-put, compute collection uuid based on a version of manifest_text with the +A signatures (and all other +anything other than +size) stripped off." This seems like the right thing to do: hash+size should uniquely identify a block in an Arvados cloud, and any other hints may vary without changing the underlying content, so if we generate a manifest for the same content but with different hints, we will get a different manifest.

Can I quick confirm whether that's what we actually want here?

- The caching mechanism introduced to CollectionWriter.manifest_text() never gets invalidated, so data will be mysteriously lost if a client does "write stuff; get manifest; write more stuff; get manifest". Either reset _manifest_text in all the methods that might invalidate it, or (my preference) stay safe and simple, and don't even bother caching it.
- If you do keep the cache, I'd rather use None than an empty string to indicate "nothing cached".

I'm happy to remove the cache: I wasn't 100% positive that the scenario you describe was actually permitted/desired.

> Changes to services/api/config/application.yml.example seem misplaced on this branch, but plausible, and doesn't seems to break anything. (Did the existing commenting start causing trouble somewhere, or is this a human-readability improvement?)

Sorry for leaving out a comment. The trailing uncommented bits caused PyYAML to barf, and the Python unit tests need to be able to parse application.yml in order to find the blob_signing_key for the test environment.

#19 - 05/29/2014 11:32 AM - Tom Clegg

Tim Pierce wrote:

> - Don't strip all hints, just permission hints.
>
> FYI, this contradicts the requirement in the story, which is "In arv-put, compute collection uuid based on a version of manifest_text with the +A signatures (and all other +anything other than +size) stripped off." This seems like the right thing to do: hash+size should uniquely identify a block in an Arvados cloud, and any other hints may vary without changing the underlying content, so if we generate a manifest for the same content but with different hints, we will get a different manifest.
>
> Can I quick confirm whether that's what we actually want here?

Indeed. But currently API server strips only +A hints before computing/verifying uuid, so I figure we're less likely to have unexpected trouble if the Python client does the same.

(Currently hints like +K@qr1hi (should) actually work if you can get them into the manifest somehow; we can change the API server's behavior when it has some other way to attach those by itself on the fly according to Data Manager's idea of what's where.)

> - The caching mechanism introduced to CollectionWriter.manifest_text() never gets invalidated, so data will be mysteriously lost if a client does "write stuff; get manifest; write more stuff; get manifest". Either reset _manifest_text in all the methods that might invalidate it, or (my preference) stay safe and simple, and don't even bother caching it.
> - If you do keep the cache, I'd rather use None than an empty string to indicate "nothing cached".
>
> I'm happy to remove the cache: I wasn't 100% positive that the scenario you describe was actually permitted/desired.

Well, we have no tests for it, but we should definitely either handle it correctly or prevent it from happening. (It's easy, so I prefer the former.)

> Sorry for leaving out a comment. The trailing uncommented bits caused PyYAML to barf, and the Python unit tests need to be able to parse application.yml in order to find the blob_signing_key for the test environment.

Aha.

#20 - 05/29/2014 12:59 PM - Tim Pierce

Changes @ 30b6c5a

manifest_uuid is computed from a manifest that been stripped only of permission hints, and CollectionWriter does not cache the manifest.

#21 - 05/29/2014 01:37 PM - Tom Clegg

Code looks good now.

But I looked more carefully at the tests after getting this:

```
======================================================================
FAIL: test_ArvPutSignedManifest (test_cmdline.ArvPutTest)
----------------------------------------------------------------------
Traceback (most recent call last):
  File "/home/tom/src/arvados/sdk/python/test_cmdline.py", line 53, in test_ArvPutSignedManifest
    self.assertRegexpMatches(manifest_uuid, r'\+A[0-9a-f]+@[0-9a-f]{8}')
AssertionError: Regexp didn't match: '\\+A[0-9a-f]+@[0-9a-f]{8}' not found in '00b4e9f40ac4dd432ef89749f1c01e7
4+47'

----------------------------------------------------------------------
```

I think the test is wrong -- arv-put is not actually expected/required to output a signed locator. The next assertion is also not quite right:

```
        # The manifest text stored in Keep must contain unsigned locators.
        m = arvados.Keep.get(manifest_uuid)
        self.assertEqual(m, ". 08a008a01d498c404b0c30852b39d3b8+44 0:44:foo\n")
```

This should fetch the manifest from the API server using collections.get, rather than Keep, and it should contain signed locators.

(The desired effect of running arv-put is that the collection can be retrieved using arv-get collection_uuid/..., i.e., ask API server for a manifest with fresh signatures, then use those to get the data blobs from Keep.)

The test could also be a bit better if it confirmed before running arv-put that the collection isn't somehow already arv-getable (or collections.get()able, or whatever).

#22 - 05/29/2014 03:34 PM - Tom Clegg

- Target version changed from 2014-05-28 Pipeline Factory to 2014-06-17 Curating and Crunch

#23 - 05/29/2014 03:34 PM - Tom Clegg

- Story points changed from 2.0 to 0.5

#24 - 05/29/2014 05:13 PM - Tim Pierce

Changes @ 9d4f6f0

- arv-put returns an unsigned manifest UUID.
- The *ArvPutSignedManifest* test first confirms that the collection is not present in the API server. Also asserts that the arv-put command completed successfully.
- Additional tests for *KeepPermissionTestCase* for each potential combination of wrong authorization and signature.

#25 - 05/30/2014 09:50 AM - Tim Pierce

Oops: corrected one comment @ b2812e9

#26 - 05/30/2014 11:13 AM - Tom Clegg

I propose 3107d80 (2755-python-sdk-permissions-TC branch): rather than having more custom locator-munging logic, arv-put trusts CollectionWriter's finish() method to give it a UUID suitable for collections.create, and trusts collections.create to give it a uuid suitable for presenting to a user/caller.

(If collections.create can't pick up what CollectionWriter is laying down, one of those needs to be fixed, not {arv-put and all other clients}. But according to the test suite and my expectations, they do behave properly.)

If that change looks good to you too, I think this is ready to merge.

#27 - 05/30/2014 02:43 PM - Tim Pierce

Tom Clegg wrote:

> I propose 3107d80 (2755-python-sdk-permissions-TC branch): rather than having more custom locator-munging logic, arv-put trusts CollectionWriter's finish() method to give it a UUID suitable for collections.create, and trusts collections.create to give it a uuid suitable for presenting to a user/caller.
>
> (If collections.create can't pick up what CollectionWriter is laying down, one of those needs to be fixed, not {arv-put and all other clients}. But according to the test suite and my expectations, they do behave properly.)

> If that change looks good to you too, I think this is ready to merge.

I see what's going on now. I hadn't walked through all the validations to understand why you were so confident that it didn't matter whether the manifest uuid included extra hints. Yes, this makes sense, thanks.

I'm comfortable with this as long as we have an explicit unit test for normalizing collection UUIDs on Collections.create. Added at 5b15dc03.

#28 - 05/30/2014 03:30 PM - Tim Pierce

- Status changed from New to Resolved

- % Done changed from 95 to 100

Applied in changeset arvados|commit:b6ea1fe3bf38bf28823c80b3aef98239a1c0311b.

#29 - 06/11/2014 11:16 PM - Tom Clegg

- Status changed from Resolved to In Progress

#30 - 06/12/2014 11:33 AM - Brett Smith

Reviewing 3377c83ec

Your comment in application.default.yml is unfinished.

It looks like these changes mean we'll need to revisit crunch-job's handling of output collation (see a4378cd48). Should that be part of this branch?

That's all I've got.

#31 - 06/12/2014 02:57 PM - Tom Clegg

Brett Smith wrote:

> Your comment in application.default.yml is unfinished.

Oops, fixed

> It looks like these changes mean we'll need to revisit crunch-job's handling of output collation (see a4378cd48). Should that be part of this branch?

Yes, that commit (and #2953) is close but not quite correct. Crunch-job should only be stripping the signatures for purposes of predicting the collection uuid. It should still be sending the same manifest_text that it used to send. I'll see if I can squeeze a patch into this branch.

#32 - 06/12/2014 03:01 PM - Tom Clegg

Tom Clegg wrote:

> I'll see if I can squeeze a patch into this branch.

In c290dd5

#33 - 06/12/2014 03:10 PM - Brett Smith

Tom Clegg wrote:

> Tom Clegg wrote:
>
> > I'll see if I can squeeze a patch into this branch.
>
> In c290dd5

Sorry, but this needs another pass. The Collections create call passes in $manifest_text as the text, but that variable no longer exists. You'll need to build it in the loop above, or something. (Remember that Perl strings are mutable—if you try to build two lists of manifest lines in parallel, and then strip signatures from the strings in one, you'll also strip them from the other unless you take additional precautions.)

#34 - 06/12/2014 03:17 PM - Tom Clegg

Brett Smith wrote:

> Sorry, but this needs another pass. The Collections create call passes in $manifest_text as the text, but that variable no longer exists. You'll need to build it in the loop above, or something. (Remember that Perl strings are mutable—if you try to build two lists of manifest lines in parallel, and then strip signatures from the strings in one, you'll also strip them from the other unless you take additional precautions.)

Darn tootin'. Also clarified some variable names in 4161894

#35 - 06/18/2014 12:24 AM - Tom Clegg

- Status changed from In Progress to Resolved