Arvados - Bug #11168

[API] Use JSON instead of YAML for serialized fields in database

02/24/2017 07:41 PM - Tom Clegg

Status:	Resolved	**Start date:**	02/24/2017
Priority:	Normal	**Due date:**	
Assigned To:	Tom Clegg	**% Done:**	100%
Category:	API	**Estimated time:**	0.00 hour
Target version:	2017-03-15 sprint		

Description

Currently hashes (like log properties) and arrays (like api_client_authorization scopes) are encoded in YAML, which is much slower than Oj.

YAML has some features that JSON is missing, but we don't want them; in fact, they get in our way (like in #6347).

If we store JSON, and tell PostgreSQL ≥9.3 that we are doing so, we can do queries on serialized fields.
https://www.postgresql.org/docs/9.6/static/datatype-json.html

Implementation notes

Migration:

- It's possible to do the up-migration in the background while the new server is running. We can detect format when loading, and deserialize accordingly: JSON starts with "{" or "[", YAML starts with "---".
- However, for a downgrade, a full down-migration would need to finish before the old version could work.
- In this version we won't bother migrating existing records -- we'll just use JSON in new/updated rows.

Changing column types:

- PostgreSQL can help us more if we use a json or jsonb column type for serialized fields -- but this ~~can~~ will be deferred to a separate story.

Evaluation

After this change is deployed, we should collect some statistics/graphs about real-world performance impact. The biggest impact will probably be on API response times for "list" actions on container_requests, containers, jobs, pipeline instances, and logs.

Subtasks:

Task # 11195: Review 11168-serialize-json	**Resolved**

Related issues:

Related to Arvados - Story #4019: [API] Support query of "properties" field o...	**Resolved**	**12/12/2017**
Related to Arvados - Story #11807: [API] Migrate old serialized database cont...	**Resolved**	**06/05/2017**

Associated revisions

Revision 660a6143 - 03/03/2017 09:46 PM - Tom Clegg

Merge branch '11168-serialize-json'

closes #11168

History

#1 - 02/28/2017 07:30 PM - Tom Morris

- Target version set to 2017-03-15 sprint

- Story points set to 2.0

#2 - 02/28/2017 07:58 PM - Javier Bértoli

- Subject changed from [API] Use JSON instead of YAML for serialized fields in database to [API] Use JSON instead of YAML for serialized fields in databasethis

This sounds like a nice improvement. That PG page has some warnings to which we should need to probably pay careful attention:

- difference in usage of **json**/**jsonb**, and which one is better depending on the JSON content we want to store.
- "...it is best to avoid mixing Unicode escapes in JSON with a non-UTF8 database encoding, if possible.". China's default encoding is **UTF-16**, not UTF-8, so I don't know if/how that will have any impact in this change, but I think is something to take into consideration.

#3 - 03/01/2017 08:25 PM - Tom Clegg

- Subject changed from [API] Use JSON instead of YAML for serialized fields in databasethis to [API] Use JSON instead of YAML for serialized fields in database

#4 - 03/01/2017 08:47 PM - Tom Clegg

- Assigned To set to Tom Clegg

#5 - 03/01/2017 09:06 PM - Tom Clegg

- Status changed from New to In Progress

#6 - 03/01/2017 09:24 PM - Tom Clegg

11168-serialize-json @ d7c84d69bb62d61bc671b2d5e0ad4ed42dbeb7c0

#7 - 03/01/2017 11:00 PM - Tom Clegg

- Description updated

#8 - 03/01/2017 11:04 PM - Tom Clegg

- Description updated

#9 - 03/02/2017 03:42 PM - Peter Amstutz

11168-serialize-json @ 766ddd6

I notice that a couple of places have been switched to use SafeJSON, but they still have require 'oj'. The specific instances are eventbus.rb and websocket_test.rb. eventbus.rb catches Oj::Error, websocket_test.rb doesn't appear to have any remaining instances of Oj.

I believe the change to Job.sorted_hash_digest may prevent job reuse unless we check for the hashes of both the YAML and JSON serializations.

Should deep_sort_hash happen in where_serialized ? It doesn't particularly make sense to query the string value of a serialized hashed column without sorting it first.

In create_superuser_token_test, the test "existing token has limited scope":

```
-       update_all(scopes: ["GET /"])
+       update_all(scopes: SafeJSON.dump(["GET /"]))
```

Why/how did this work before, and why does it need to be manually serialized now?

I did some manual verification:

1. Looked at "properties" column of "logs" table in pqsl
2. The earliest log item was previously serialized to postgres as YAML
3. The latest log item is serialized to postgres as JSON
4. Both earliest and latest log records can be accessed via API and are reported as JSON.

#10 - 03/03/2017 05:43 AM - Tom Clegg

> I notice that a couple of places have been switched to use SafeJSON, but they still have require 'oj'. The specific instances are eventbus.rb and websocket_test.rb. eventbus.rb catches Oj::Error, websocket_test.rb doesn't appear to have any remaining instances of Oj.

Indeed. Removed import from websocket_test.rb, thanks.

> I believe the change to Job.sorted_hash_digest may prevent job reuse unless we check for the hashes of both the YAML and JSON serializations.

Ah, good catch. Both old and new are JSON, but Oj.dump(h) serializes {"foo":"bar"} as {":foo":"bar"} so changing to compat mode without a migration would break reuse of jobs saved by old versions. I suppose "symbol" mode is fine for this as long as we keep doing it. Reverted.

> Should deep_sort_hash happen in where_serialized ? It doesn't particularly make sense to query the string value of a serialized hashed column without sorting it first.

Yes, good point. Moved into where_serialized.

> In create_superuser_token_test, the test "existing token has limited scope":
>
> update_all(scopes: ["GET /"])
>
> Why/how did this work before, and why does it need to be manually serialized now?

Well, it turns out it didn't work all that well before:

```
======================================================================
CreateSuperUserTokenTest#test_existing_token_has_limited_scope
----------------------------------------------------------------------

  ApiClientAuthorization Load (0.3ms)  SELECT "api_client_authorizations".* FROM "api_client_authorizations" W
HERE "api_client_authorizations"."id" = $1 LIMIT 1  [["id", 279786541]]
  SQL (0.4ms)  UPDATE "api_client_authorizations" SET "scopes" = 'GET /' WHERE "api_client_authorizations"."us
er_id" = 476014017
```

But the purpose of this statement was just to sabotage the test fixture so actual≠desired scopes, and invalid≠desired, so the test passed.

Now it does what it looks like it does:

```
  SQL (0.4ms)  UPDATE "api_client_authorizations" SET "scopes" = '["GET /"]' WHERE "api_client_authorizations"
."user_id" = 476014017
```

(update_all() is just a DB query, it bypasses model logic.)

now at 594e00f9311da95f73843f55b6e1c7c3ad55d8df

#11 - 03/03/2017 04:30 PM - Peter Amstutz

LGTM @ 594e00f

#12 - 03/03/2017 04:45 PM - Peter Amstutz

Hold on, now I'm having trouble starting the API server in arvbox. I don't know if something is just corrupted/confused or there's a real problem here:

```
2017-03-03_16:43:49.18377   Job Load (0.6ms)  SELECT "jobs".* FROM "jobs" WHERE (state = 'Queued') ORDER BY pr
iority desc, created_at
2017-03-03_16:43:49.18440    (0.5ms)  SELECT COUNT(*) FROM "pipeline_instances" WHERE (state = 'RunningOnServe
r')
2017-03-03_16:43:49.57545   ApiClientAuthorization Load (0.7ms)  SELECT "api_client_authorizations".* FROM "ap
i_client_authorizations" WHERE (api_token='4ao313k81hkmc1812j4eo25uf5p
3wlmc78edkkkpqnrhf8bwvb' and (expires_at is null or expires_at > CURRENT_TIMESTAMP)) LIMIT 1
2017-03-03_16:43:49.57617 App 4431 stderr: [ 2017-03-03 16:43:49.5759 4545/0x0055cfdc0e9b40(Worker 1) utils.rb
:87 ]: *** Exception RuntimeError in Rack application object (invalid
serialized data "\"[\\\"al") (process 4545, thread 0x0055cfdc0e9b40(Worker 1)):
2017-03-03_16:43:49.57619 App 4431 stderr:      from /usr/src/arvados/services/api/lib/serializers.rb:32:in `l
oad'
2017-03-03_16:43:49.57619 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/attribute_methods/serialization.rb:24:in `unserialize'
2017-03-03_16:43:49.57619 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/attribute_methods/serialization.rb:15:in `unserialized_value'
2017-03-03_16:43:49.57620 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/attribute_methods/read.rb:84:in `__temp__'
2017-03-03_16:43:49.57620 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/attribute_methods/read.rb:46:in `type_cast_attribute'
2017-03-03_16:43:49.57620 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/attribute_methods/read.rb:127:in `read_attribute'
2017-03-03_16:43:49.57621 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/attribute_methods.rb:185:in `block in attributes'
2017-03-03_16:43:49.57621 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/attribute_methods.rb:185:in `each'
2017-03-03_16:43:49.57621 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/attribute_methods.rb:185:in `attributes'
2017-03-03_16:43:49.57621 App 4431 stderr:      from /usr/src/arvados/services/api/app/models/arvados_model.rb
:498:in `block in convert_serialized_symbols_to_strings'
2017-03-03_16:43:49.57622 App 4431 stderr:      from /usr/src/arvados/services/api/app/models/arvados_model.rb
:497:in `each'
2017-03-03_16:43:49.57622 App 4431 stderr:      from /usr/src/arvados/services/api/app/models/arvados_model.rb
:497:in `convert_serialized_symbols_to_strings'
2017-03-03_16:43:49.57622 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activesupport-3.2.22.5/lib/
```

```
active_support/callbacks.rb:405:in `_run__1338818612019012701__find__4105550916150344441__callbacks'
2017-03-03_16:43:49.57623 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activesupport-3.2.22.5/lib/
active_support/callbacks.rb:405:in `__run_callback'
2017-03-03_16:43:49.57623 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activesupport-3.2.22.5/lib/
active_support/callbacks.rb:385:in `_run_find_callbacks'
2017-03-03_16:43:49.57623 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activesupport-3.2.22.5/lib/
active_support/callbacks.rb:81:in `run_callbacks'
2017-03-03_16:43:49.57623 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/base.rb:523:in `init_with'
2017-03-03_16:43:49.57624 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/inheritance.rb:68:in `instantiate'
2017-03-03_16:43:49.57624 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/querying.rb:38:in `block (2 levels) in find_by_sql'
2017-03-03_16:43:49.57624 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/querying.rb:38:in `collect!'
2017-03-03_16:43:49.57625 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/querying.rb:38:in `block in find_by_sql'
2017-03-03_16:43:49.57626 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/explain.rb:41:in `logging_query_plan'
2017-03-03_16:43:49.57626 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/querying.rb:37:in `find_by_sql'
2017-03-03_16:43:49.57626 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/relation.rb:171:in `exec_queries'
2017-03-03_16:43:49.57626 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/relation.rb:160:in `block in to_a'
2017-03-03_16:43:49.57626 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/explain.rb:41:in `logging_query_plan'
2017-03-03_16:43:49.57627 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/relation.rb:159:in `to_a'
2017-03-03_16:43:49.57627 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/relation/finder_methods.rb:381:in `find_first'
2017-03-03_16:43:49.57627 App 4431 stderr:      from /var/lib/gems/ruby/2.1.0/gems/activerecord-3.2.22.5/lib/a
ctive_record/relation/finder_methods.rb:122:in `first'
2017-03-03_16:43:49.57628 App 4431 stderr:      from /usr/src/arvados/services/api/app/middlewares/arvados_api
_token.rb:39:in `call'
```

#13 - 03/03/2017 04:59 PM - Peter Amstutz

```
arvados_development=# SELECT "api_client_authorizations".scopes FROM "api_client_authorizations" WHERE (api_to
ken='4ao313k81hkmc1812j4eo25uf5p3wlmc78edkkkpqnrhf8bwvb' and (expires_at is null or expires_at > CURRENT_TIMES
TAMP)) LIMIT 1;
   scopes
-------------
 "[\"all\"]"
(1 row)
```

Hmm?

#14 - 03/03/2017 05:02 PM - Peter Amstutz

```
arvados_development=# SELECT id, scopes FROM "api_client_authorizations";
 id |   scopes
----+-------------
  1 | "[\"all\"]"
  2 | ["all"]
```

#15 - 03/03/2017 06:22 PM - Peter Amstutz

```
 id |                       api_token                        | api_client_id | user_id | created_by_ip_address | l
ast_used_by_ip_address |        last_used_at        | expires_at |         created_at         |         update
d_at          | default_owner_uuid |   scopes    |            uuid
----+--------------------------------------------------------+---------------+---------+-----------------------+--
-----------------------+----------------------------+------------+----------------------------+----------------
--------------+--------------------+-------------+----------------------------
  1 | 4ao313k81hkmc1812j4eo25uf5p3wlmc78edkkkpqnrhf8bwvb |             1 |       1 | ::1                   | 1
92.168.5.3             | 2017-03-03 16:43:31.951122 |            | 2017-02-24 16:08:41.941389 | 2017-03-03 16:
43:32.971805 |                    | "[\"all\"]" | 34t0i-gj3su-28nah1lzkhpalqb
  2 | mdvgi7g61ec6gshnwlhi92icny27map7k5o3nghnz18j82192  |             2 |       3 | 192.168.5.1           | 1
92.168.5.2             | 2017-03-02 15:27:32.650515 |            | 2017-03-02 15:21:51.147115 | 2017-03-02 15:
27:32.651099 |                    | ["all"]     | 34t0i-gj3su-u4yxlmib581etwd
(2 rows)
```

#16 - 03/03/2017 09:23 PM - Tom Clegg

07e4083ea451913b988d77e8e4c926da8ad844a4

"Double-decode serialized fields if database was mangled by downgraded API server."

#17 - 03/03/2017 09:50 PM - Tom Clegg

- Status changed from In Progress to Resolved

Applied in changeset arvados|commit:660a6143ecf1e777f33bd84183ba9e821e1d7a8e.