Arvados - Bug #18346

Login federation: request storm overwhelming login cluster rails api server

11/09/2021 03:34 PM - Peter Amstutz

Status:	Resolved	**Start date:**	11/10/2021
Priority:	Normal	**Due date:**	
Assigned To:	Tom Clegg	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	2021-11-24 sprint		

Description

A customer has seen this behavior in 2 different scenarios:

a) when a user used an old token that was issued by a local cluster prior to the migration to a login federation. Local cluster and login cluster on Arvados 2.2.2
b) when a big workflow is run on a 2.3.0 cluster with the login cluster on 2.2.2

The b) case appears to be a 2.3 regression: the workflow that triggered the outage is a re-run that did not cause problems on Arvados 2.2.x (or older, that's not clear).

The requests that end up at the login cluster api server have a specific request parameter pattern (include_trash=true&select=[uuid]). They seem to be user and collection requests.

The collection requests seem to be for log collections (i.e. the workflow steps writing to them, presumably?).

The requests all get a 401 response from the login cluster api server, but this does not appear to impede the running of the big workflow on the local cluster.

The customer implemented a workaround: greatly increasing the number of passenger workers on the login cluster api server made it able to handle many more concurrent requests (and return a 401 for them), which avoids the overload death spiral when clients retry.

Subtasks:

Task # 18351: Review 18346-container-token	**Resolved**
Task # 18365: build 2.3.1~rc1 with bugfix	**Resolved**
Task # 18366: Review 18346-crunchrun-no-events	**Resolved**
Task # 18373: merge fixes into 2.3.1	**Resolved**

Associated revisions

Revision a4886639 - 11/10/2021 08:11 PM - Tom Clegg

Merge branch '18346-container-token'

refs #18346

Arvados-DCO-1.1-Signed-off-by: Tom Clegg <tom@curii.com>

Revision ee0b90f8 - 11/12/2021 02:38 PM - Peter Amstutz

Merge branch '18346-crunchrun-no-events' refs #18346

Arvados-DCO-1.1-Signed-off-by: Peter Amstutz <peter.amstutz@curii.com>

Revision b9b43736 - 11/17/2021 07:47 PM - Peter Amstutz

Merge branch '18346-crunchrun-no-events' refs #18346

Arvados-DCO-1.1-Signed-off-by: Peter Amstutz <peter.amstutz@curii.com>

History

#1 - 11/09/2021 03:34 PM - Peter Amstutz

- Status changed from New to In Progress

#2 - 11/09/2021 03:35 PM - Peter Amstutz

- Description updated

#3 - 11/09/2021 03:37 PM - Ward Vandewege

- Subject changed from Request storm overwhelming federation to Request storm overwhelming login federation

#4 - 11/09/2021 03:37 PM - Ward Vandewege

- Subject changed from Request storm overwhelming login federation to Login federation: request storm overwhelming login cluster rails api server

#5 - 11/09/2021 03:41 PM - Ward Vandewege

- Description updated

#6 - 11/09/2021 03:43 PM - Ward Vandewege

- Description updated

#7 - 11/09/2021 03:44 PM - Peter Amstutz

Workflow is running on local cluster successfully despite apparently producing a bunch of extra token checking traffic.

#8 - 11/09/2021 03:45 PM - Ward Vandewege

- Description updated

#9 - 11/09/2021 04:22 PM - Peter Amstutz

The questions seem to be:

1. Where exactly are all the user GET requests coming from
2. Can we protect the login cluster / Arvados in general against these situations where the API server is already overloaded, and client retries (because the server isn't responding) makes it much worse

#10 - 11/09/2021 05:17 PM - Peter Amstutz

1. In the login cluster case, never forward locally-issued (by the satellite) (e.g. per-container) tokens to the login cluster.
2. are requests going to the OIDC provider? Shouldn't happen if they are v2/ format.
3. Do we need to cache negative results for token lookup?
4. Per-client throttling in nginx?
5. Track outstanding validation requests for tokens, don't send a new check if one is already in flight.
6. In the peer federation case, if cluster gets a token that belongs to another cluster, it will go back to the 1st cluster to validate the token, does this also happen in the login cluster case? Then if the satallite cluster returns a login cluster user uuid, that would be a token failure.

#11 - 11/09/2021 05:43 PM - Ward Vandewege

- Release set to 45

#12 - 11/10/2021 04:06 PM - Peter Amstutz

- Target version changed from 2021-11-10 sprint to 2021-11-24 sprint

#13 - 11/10/2021 04:13 PM - Peter Amstutz

- Assigned To set to Tom Clegg

#14 - 11/10/2021 04:16 PM - Tom Clegg

18346-container-token @ 69a9857a37007723c17007b0c2f960b87e95bc02 -- https://ci.arvados.org/view/Developer/job/developer-run-tests/2783/

#15 - 11/10/2021 08:03 PM - Lucas Di Pentima

Just one cosmetic suggestion:

- If we're going to link this ticket on the test, we should anonymize it and remove its private status.

Other than that, LGTM.

#16 - 11/10/2021 08:08 PM - Tom Clegg

- Description updated

#18 - 11/10/2021 08:11 PM - Tom Clegg

- Project changed from Arvados Private to Arvados

#19 - 11/10/2021 08:13 PM - Tom Clegg

Merged to main, and cherry-picked to 2.3-dev @ 7de380d5e

#20 - 11/11/2021 07:43 PM - Peter Amstutz

Customer discovered that requests are coming from arv-ws.

This fills in the missing piece of the puzzle.

1. Our implementation of Singularity support includes converting Docker images to Singularity on demand, on the compute node, and caching the SIF files in keep
2. Caching involves writing the SIF files uses writable arv-mount
3. Previously, arv-mount only ever mounted collections by portable data hash, but in order to write to a collection, it must be mounted by UUID
4. In addition to "by_id" crunch-run also adds a "by_uuid" mount point.
5. When a collection is mounted by UUID, arv-mount subscribes to the websockets services in order to receive notifications if the collection changes, when it does this, it uses the token that was issued to the container
6. When an event happens, the websocket services needs to know if a given client is permitted to receive the event
7. The Go-based websocket service predates both the Go-based controller and materialized_permission table, so at the time, in order to avoid reimplementing permission system checking that existed only in Ruby, it would call back to the API server with a GET request on behalf of the user
8. Some events related to user records, so it would send a GET request for the user record.
9. Requests for user records are automatically forwarded to the login cluster, with the token that was provided
10. The Login cluster sees the token that was provided, which has a *token uuid* prefix which is the satellite cluster (because the token was issued by the satellite cluster), and *calls back* to the satellite cluster to validate the token
11. The satellite responds that the token belongs to a user owned by the login cluster
12. Since the user isn't owned by the satellite cluster, the token lookup is rejected
13. Everything unwinds from there, except that because this fills two active requests on the satellite cluster and an active request on the login cluster, if either of those request queues start to fill up, the both systems (and anything else that relies on the login cluster) becomes gridlocked

#21 - 11/11/2021 08:43 PM - Peter Amstutz

Takeways / additional things to fix:

- The root cause is compute nodes unintentionally subscribing to the websocket service. Crunch-run's arv-mount should be invoked with --disable-event-listening
- arvados-ws already talks to postgres directly. It should be directly querying materialized_permissions to determine if a user is permitted to receive events for a given record
- I don't think arvados-ws distinguishes between a 403/404 (user isn't allowed to access the resource) and a 401 (invalid token) so it may continue to try to use the invalid token over and over again.
- When a login cluster is configured, making requests to the login cluster with a locally-issued tokens will never work, so it shouldn't make the request (this is the #note-14 fix)
  - In general, federated requests where the token's cluster isn't the same as the user's cluster will never work
- The login cluster receives a token created by a different cluster, the login cluster needs to call back to validate the token.
  - This is correct behavior for the peer federation case.
  - In the login cluster case, it is incorrect if the token is from a satallite cluster, but the login cluster itself currently doesn't know which clusters are configured to delegate to it (?) so it can't decide
- Token validation doesn't cache negative lookups.
  - If the login cluster cached the negative result (that the token was rejected) it could have rejected subsequent attempts much faster

#22 - 11/11/2021 09:35 PM - Peter Amstutz

18346-crunchrun-no-events @ d0a50cd1fafca2a931f35f7997bd40f01a295ee0

- Add --disable-event-listening to crunch-run arv-mount invocation so it doesn't subscribe to websocket events

https://ci.arvados.org/view/Developer/job/developer-run-tests/2792/

#23 - 11/11/2021 10:02 PM - Tom Clegg

- Add --disable-event-listening to crunch-run arv-mount invocation so it doesn't subscribe to websocket events

This seems to change the semantics of mounting collections by UUID, and I'm not seeing why it's needed here given that we've fixed the websocket issue by making the controller permission check return quickly. Suggest not making this change until/unless we decide we want to change/document the mount semantics.

#24 - 11/11/2021 10:35 PM - Peter Amstutz

Tom Clegg wrote:

- Add --disable-event-listening to crunch-run arv-mount invocation so it doesn't subscribe to websocket events

> This seems to change the semantics of mounting collections by UUID, and I'm not seeing why it's needed here given that we've fixed the websocket issue by making the controller permission check return quickly. Suggest not making this change until/unless we decide we want to change/document the mount semantics.

We *really* don't want to have 200 compute nodes subscribing to websockets, and then having the websockets server make 200 requests to the API server every time any object changes on the API server.

18346-crunchrun-no-events @ 1c36c7a9d4cb3829e57aab9ac84a6b85ec35459c

Only add --disable-event-listening when all container collection mounts are by PDH.

https://ci.arvados.org/view/Developer/job/developer-run-tests/2793/

#25 - 11/12/2021 02:35 PM - Tom Clegg

Peter Amstutz wrote:

> We *really* don't want to have 200 compute nodes subscribing to websockets, and then having the websockets server make 200 requests to the API server every time any object changes on the API server.

To be fair, handling 200 clients without doing 200 API requests per change is pretty much the websocket server's job description.

The websocket server could detect this (login-cluster-owned object, local token) situation itself and skip the permission lookup -- but websocket server already caches permission lookups, and controller now answers this without calling the database or rails, so I'm not sure it would make a huge difference.

> 18346-crunchrun-no-events @ 1c36c7a9d4cb3829e57aab9ac84a6b85ec35459c
>
> Only add --disable-event-listening when all container collection mounts are by PDH.
>
> https://ci.arvados.org/view/Developer/job/developer-run-tests/2793/

LGTM, thanks!

#26 - 11/12/2021 02:53 PM - Peter Amstutz

Tom Clegg wrote:

> Peter Amstutz wrote:
>
> > We *really* don't want to have 200 compute nodes subscribing to websockets, and then having the websockets server make 200 requests to the API server every time any object changes on the API server.
>
> To be fair, handling 200 clients without doing 200 API requests per change is pretty much the websocket server's job description.

My reading is that it caches the result per-uuid, per-token. So if you have 200 different tokens and 10 changes on 10 different objects you would still end up with 2000 GET requests. There's definitely ways we could make ws permission lookups much more lightweight (and the ws server more useful in general), but that's out of scope for this bugfix.

> LGTM, thanks!

thanks, merged.

#27 - 11/17/2021 07:53 PM - Peter Amstutz

- Status changed from In Progress to Resolved