Jan 19, 2022 | Airflow Multi-Tenancy meeting #3

Proposed agenda:

- Intro: Broad goals of the two upcoming AIPs - Jarek Potiuk
- AIP-43 Walkthrough - Mateusz Henc
- AIP-44 Walkthrough - Jarek Potiuk
- Initial conversations:
    - Ping Zhang/Kevin Yang:
        - Parsing service, Dag serialization Docker env
        - Remove double parsing

Meeting recording:
https://drive.google.com/file/d/1SMFzazuY1kg4B4r11wNt8EQ_PmTDRKq6/view

Attendees:

- Jarek Potiuk
- Mateusz Henc
- Ash Berlin-Taylor
- Kevin Yang
- Ping Zhang
- XD
- Sam Wheating
- Ian Buss
- Elad Kalif
- Rafał Biegacz
- John Jackson
- Winnie Xiong
- Nikolas Oliveira
- … and others

Notes

- Jarek Introduced the concept/scope of the two AIPs being discussed:
    - Security
    - Minimum changes
    - This is just a beginning - more changes will follow up
- Mateusz explained the details of AIP-43
    - Questions were asked/answered, global consensus seems to be reached that we could start voting on it very soon
    - Detailed list of questions below (answers in the recording)
- Jarek explained the details of AIP-44
    - Questions were asked/answered, there is a need to look into details and get more comments from the participants but seems that broadly the idea is accepted

    - Detailed list of questions below (answers in the recording)
- General discussion on the follow up changes that might come next:
    - Kevin/Ping - impersonation of processes running user code (generally yes and we should discuss it as part of finalization of AIP-44 - long term likely this will be addressed as a feature
    - Sam - question about “resource level isolation” - this will be follow-up AIP something that Ian/Cloudera is interested in designing
    - Ash - there is a proposal from Bolke de Bruin about reversing the access to necessary resources (connections, Xcom etc.) will be provided as payload to the Tasks - which will make tasks “standalone” so they will not need to access any external resources besides communicating with the services needed. Consensus is that anything we do now, does not preclude it in the future. It is more complex because it would require to change the way how DAGs are written and the AIP-43/44 approach is to minimize changes needed. We might consider implementing this approach as the follow-up

Action items

- ☐ AIP-43 - final comments and we send it up for voting (Mateusz)
- ☐ AIP-44 - some more comments and clarification and we will try to make sure we reach consensus before we send it up for voting (Jarek)
- ☐ Further discussion on new proposals to be addressed offline/next meeting.

```
00:00:32.170,00:00:35.170
Rafal Biegacz: I like the way Jarek sets expectations :)

00:09:02.565,00:09:05.565
Ash Berlin-Taylor: Nit: s/Agent/Manager/ in that -- the Agent is the bit
of code that runs in the scheduler and watches the Manager. The Manager is
the thing that maintains the pool of parsers and writes to the DB etc.

00:09:20.580,00:09:23.580
Ash Berlin-Taylor: (Just name is wrong is all)

00:14:09.297,00:14:12.297
Alexander Chen: How would the DAGs storage isolation be enforced on
worker?

00:14:55.549,00:14:58.549
Roberto Santamaria: Regarding AIP-43 - Is it neccessary to have 2
strategies "trusted" and "untrusted"?  Why not treat all as "untrusted"?
Possibly simplifies implementation having a single strategy?

00:15:27.932,00:15:30.932
Alexander Chen: Thank you!
```

00:15:29.205,00:15:32.205
Ash Berlin-Taylor: All user code is unstructed

00:15:35.677,00:15:38.677
Mocheng Guo: for multiple dag directories, how does it handle dag id uniqueness, will directoy be part of dag id?

00:16:36.360,00:16:39.360
Ping Zhang: what's the conclusion where the callbacks will be run?

00:16:49.321,00:16:52.321
John Jackson: Today, DAG/plugin code can patch internal Airflow functionality. Will this division of code prevent that from happening in the future?

00:17:21.637,00:17:24.637
Ping Zhang: Will the callbacks in the db extended to callbacks from the airflow worker side?

00:17:23.226,00:17:26.226
Ash Berlin-Taylor: Yes John.

00:17:33.227,00:17:36.227
John Jackson: Thanks Ash!

00:17:53.611,00:17:56.611
John Jackson: Very much so :)

00:18:24.300,00:18:27.300
Ping Zhang: 👍

00:20:15.497,00:20:18.497
Sam Wheating: If we're namespacing DAGs for a multi-tenant setting (for example, dag_id = <namespace>.<dag_id>), should we also be namespacing connections, and only allowing DAGs to use connections within the same namespace?

Apologies if that is out of scope for this AIP, but I think its an important consideration for multi-tenancy.

00:20:30.200,00:20:33.200
Ash Berlin-Taylor: Connections acl etc happens later Sam

00:20:34.584,00:20:37.584
Ash Berlin-Taylor: It builds on top of this

00:20:38.348,00:20:41.348
Sam Wheating: Gotcha, thanks.

00:20:49.446,00:20:52.446
Ash Berlin-Taylor: But sepaarting/removing direct DB access is the first step

00:21:11.025,00:21:14.025
Ping Zhang: 👍 cool, thanks

00:36:57.586,00:37:00.586
Winnie Xiong: How does triggerer come into play and work with other components?

00:39:35.279,00:39:38.279
Winnie Xiong: cool, thank you!

00:39:54.453,00:39:57.453
Ash Berlin-Taylor: Have you thought about API format? OpenAPI? JSON? ProtoBuff?

00:40:08.530,00:40:11.530
Ash Berlin-Taylor: etc. etc..

00:42:18.210,00:42:21.210
Rafal Biegacz: I need to run into another mtg. Jarek & Mateusz - thank you for organizing this mtg.

00:42:32.949,00:42:35.949
Sam Wheating: Is this going to significantly impact the performance of the DagProcessor, especially in environments with many DAGs? If so, would it make sense to refactor some of the API methods used in DAG processing into batch methods, to reduce the volume of API calls?

00:43:09.909,00:43:12.909
Sam Wheating: Since DagProcessing incurs a lot of round trips to the DB, which becomes a lot of HTTP calls

00:44:00.041,00:44:03.041
Winnie Xiong: sorry I might have missed that, but how do you define the boundary between trusted components and untrusted components?

00:46:32.699,00:46:35.699
Winnie Xiong: super helpful, thanks!

00:46:37.373,00:46:40.373
XD: Is pod_mutation_hook considered as trusted or untrusted?

00:47:20.696,00:47:23.696
Ash Berlin-Taylor: Yeah,it's System Admin provided, not dag author provided

00:49:14.410,00:49:17.410
Nikolas Oliveira: "System Admins are trusted and dag authors are not trusted". Does this mean plugin code is trusted or no?

00:50:43.256,00:50:46.256
Ash Berlin-Taylor: Yes it would be

00:56:23.628,00:56:26.628
Ping Zhang: totally, should be pluggable

00:57:13.385,00:57:16.385
John Jackson: Thanks Jarek for leading the call!