

Warning! AAI REST API Not Safe With Concurrent Access (i.e. more than 1 REST client)

Problem Overview

The REST API of the CRUD webservice [A&AI Resources](#) is not consistent under concurrent ~~access~~ modification to the same entity (calls using the same ID).

If multiple requests modify the same entity the result is undetermined. The worst case behaviour observed is when concurrent access produces duplicates of that entity - the result is that **no further REST queries (GET/POST/DELETE ...) to this entity are possible and the entity becomes inaccessible**. Afterwards only a manual intervention in the database can restore proper service behaviour. Access the data concurrently at your own risk.

It is recommended that at all times there is only 1 client calling the A&AI REST services.

Technical aspects of the problem

It is unclear what causes this problem. The most probable cause is that the code calling [JanusGraph](#) library is buggy. In order to guarantee data consistency special precautions have to be made described here [Chapter 31. Eventually-Consistent Storage Backends](#). Unfortunately the JanusGraph features that ensure data consistency are not used in the current A&AI code.

Path forward

There are 4 solutions possible (subjectively sorted from worst to best)

1. Recommend using max. 1 A&AI REST client.
 - PROS: the current state - no effort needed, works "most of the time".
 - CONS: not what would be expected of a carrier-grade system by any rational measure
2. Having batch jobs scan the database and remove duplicates and corrupted data created due to inconsistencies (this is being used now as problem mitigation)
 - PROS: you don't have to deal with the real problem, it is easy because you only target symptoms
 - CONS: solution does not work because if data gets corrupted between batch sweeps then the data is unavailable until the batch job runs again. Also adds accidental complexity with batch jobs and their timing.
3. Not write data in A&AI Resources directly but use the [Champ](#) service (Note: in the future architecturally the Champ project should be the only one accessing the JanusGraph database and A&AI Resources would only forward entity change requests to Champ)
 - PROS: Access to JanusGraph database can be properly implemented in Champ without of braking existing functionality
 - CONS: Champ seems like a dead initiative and is not going to be finished in the next few years. Correct me if this assessment of Champ is wrong, for example with a concrete finish date.
4. Correct the root cause of the inconsistency
 - PROS: Webservice would work as expected
 - CONS: Changes to core A&AI libraries are needed, potential to break functionality or trigger software regressions. Deep A&AI expertize around data handling needed

Recovery action

See also [ONAP Dublin Troubleshooting](#)

For instance, we had many duplicate Logical-links and we could not fetch/delete them.

So, we have to first get their Vertex IDs so that we can manually delete them from Graph database.

Fetching Vertex IDs

```
curl -X GET \  
'https://172.30.0.90:30233/aai/v16/network/logical-links?format=id' \  
-H 'Accept: application/json' \  
-H 'Authorization: Basic QUfJ0kFBSQ==' \  
-H 'Content-Type: application/json' \  
-H 'Postman-Token: 5f14e474-f704-4f48-859e-ac1977e7243c' \  
-H 'X-FromAppId: Postman Application' \  
-H 'X-TransactionId: Postman REST Transaction' \  
-H 'cache-control: no-cache'
```

An example of an output is shown [the official documentation](#)

Upon fetching all the vertex IDs that need to be manually removed, for each one of them we execute the following command (replace the graph-admin POD and VERTEX_ID placeholders as per your environment)

Manually Remove Objects from A&AI

```
kubectl exec -it <graph-admin-pod> -- bash -c 'gosu aaiadmin ./scripts/forceDeleteTool.sh -action DELETE_NODE -userId testId1 -vertexId <VERTEX_ID>'
```

For example, here is one of the executions to delete a duplicate entry of a logical-link

Command Output

```
root@onap-rancher-daily:/home/ubuntu# kubectl exec -it dev-aai-aai-graphadmin-c8b9c58c5-kh7t7 -- bash -c 'gosu aaiadmin ./scripts/forceDeleteTool.sh -action DELETE_NODE -userId testId1 -vertexId 749696'
```

Defaulting container name to aai-graphadmin.

Use 'kubectl describe pod/dev-aai-aai-graphadmin-c8b9c58c5-kh7t7 -n onap' to see all of the containers in this pod.

Wed Jul 3 08:20:16 UTC 2019 Starting ./scripts/forceDeleteTool.sh

NOTE - if you are deleting data, please run the dataSnapshot.sh script first or at least make a note the details of the node that you are deleting.

---- NOTE --- about to open graph (takes a little while)-----

ForceDelete called by: userId [testId1] with these params: [-action DELETE_NODE -userId testId1 -vertexId 749696]

>>> Found Vertex with VertexId = 749696, properties:

[resource-version|1560082563941]

[in-maint|false]

[last-mod-source-of-truth|prh]

[aai-created-ts|1560082563941]

[aai-last-mod-ts|1560082563941]

[source-of-truth|prh]

[aai-uri|/network/logical-links/logical-link/167772160-3-0]

[aai-uuid|5686dadd-d40d-454f-9018-5a2afb8656ba]

[link-name|167772160-3-0]

[link-type|attachment-point]

[aai-node-type|logical-link]

No OUT edges were found for this vertex.

No IN edges were found for this vertex.

Found 0 descendant nodes. Note - forceDelete does not cascade to child nodes, but they may become unreachable after the delete.

Found total of 0 edges incident on this node.

Are you sure you want to do this delete? (y/n): y

User [testId1] has confirmed this delete request.

>>>>>>>>> Removed node with vertexId = 749696

Failed to run the tool ./scripts/forceDeleteTool.sh successfully
command terminated with exit code 1