**Current CLI**

* access-allow Allow access to the share.
* access-deny Deny access to a share.
* access-list Show access list for share.

**Current DB**

**mysql> desc share\_access\_map**

 +--------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+--------------+--------------+------+-----+---------+-------+

| created\_at | datetime | YES | | NULL | |

| updated\_at | datetime | YES | | NULL | |

| deleted\_at | datetime | YES | | NULL | |

| deleted | varchar(36) | YES | | NULL | |

| id | varchar(36) | NO | PRI | NULL | |

| share\_id | varchar(36) | NO | MUL | NULL | |

| access\_type | varchar(255) | YES | | NULL | |

| access\_to | varchar(255) | YES | | NULL | |

| access\_level | varchar(2) | YES | | NULL | |

+--------------+--------------+------+-----+---------+-------+

**mysql> desc share\_instance\_access\_map**

+-------------------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-------------------+-------------+------+-----+---------+-------+

| created\_at | datetime | YES | | NULL | |

| updated\_at | datetime | YES | | NULL | |

| deleted\_at | datetime | YES | | NULL | |

| deleted | varchar(36) | YES | | NULL | |

| id | varchar(36) | NO | PRI | NULL | |

| share\_instance\_id | varchar(36) | YES | MUL | NULL | |

| access\_id | varchar(36) | YES | MUL | NULL | |

+-------------------+-------------+------+-----+---------+-------+

7 rows in set (0.00 sec)

**mysql> desc shares;**

+-----------------------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-----------------------------+--------------+------+-----+---------+-------+

| created\_at | datetime | YES | | NULL | |

| updated\_at | datetime | YES | | NULL | |

| deleted\_at | datetime | YES | | NULL | |

| deleted | varchar(36) | YES | | NULL | |

| id | varchar(36) | NO | PRI | NULL | |

| user\_id | varchar(255) | YES | | NULL | |

| project\_id | varchar(255) | YES | | NULL | |

| size | int(11) | YES | | NULL | |

| display\_name | varchar(255) | YES | | NULL | |

| display\_description | varchar(255) | YES | | NULL | |

| snapshot\_id | varchar(36) | YES | | NULL | |

| share\_proto | varchar(255) | YES | | NULL | |

| share\_type\_id | varchar(36) | YES | | NULL | |

| is\_public | tinyint(1) | YES | | NULL | |

| snapshot\_support | tinyint(1) | YES | | NULL | |

| consistency\_group\_id | varchar(36) | YES | MUL | NULL | |

| source\_cgsnapshot\_member\_id | varchar(36) | YES | | NULL | |

| task\_state | varchar(255) | YES | | NULL | |

| replication\_type | varchar(255) | YES | | NULL | |

+-----------------------------+--------------+------+-----+---------+-------+

19 rows in set (0.01 sec)

**mysql> desc share\_instances;**

+----------------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+----------------------+--------------+------+-----+---------+-------+

| created\_at | datetime | YES | | NULL | |

| updated\_at | datetime | YES | | NULL | |

| deleted\_at | datetime | YES | | NULL | |

| deleted | varchar(36) | YES | | NULL | |

| id | varchar(36) | NO | PRI | NULL | |

| share\_id | varchar(36) | YES | MUL | NULL | |

| host | varchar(255) | YES | | NULL | |

| status | varchar(255) | YES | | NULL | |

| scheduled\_at | datetime | YES | | NULL | |

| launched\_at | datetime | YES | | NULL | |

| terminated\_at | datetime | YES | | NULL | |

| share\_network\_id | varchar(36) | YES | MUL | NULL | |

| share\_server\_id | varchar(36) | YES | MUL | NULL | |

| availability\_zone\_id | varchar(36) | YES | MUL | NULL | |

| access\_rules\_status | varchar(255) | YES | | NULL | |

| replica\_state | varchar(255) | YES | | NULL | |

+----------------------+--------------+------+-----+---------+-------+

16 rows in set (0.01 sec)

**DB Schema Changes**

3 new tables:

**1.access\_groups table**

+---------------------+--------------+------+-----+---------+-------+

| Field               | Type         | Null | Key | Default | Extra |

+---------------------+--------------+------+-----+---------+-------+

| created\_at          | datetime     | YES  |     | NULL    |       |

| updated\_at          | datetime     | YES  |     | NULL    |       |

| deleted\_at          | datetime     | YES  |     | NULL    |       |

| deleted             | varchar(36)  | NO   |     | NULL    |     |

| Id                  | varchar(36)  | NO   | PRI | NULL    |       |

| project\_id          | varchar(36) | YES  |     | NULL    |       |

| name                | varchar(255) | YES  |     | NULL    |       |

| description         | varchar(255) | YES  |     | NULL    |       |

| access\_type         | varchar(255) | YES  |     | NULL    |       |

exists in access\_group table. Homogeneous access\_type should be for

all entries in an access\_group right ?

| access\_level | varchar(2) | YES | | NULL | |<-

Thinking of putting it in access\_group table, what say ?

---------------------------------------------------------------------

**2.access\_group\_entries table**

+---------------------+--------------+------+-----+---------+-------+

| Field               | Type         | Null | Key | Default | Extra |

+---------------------+--------------+------+-----+---------+-------+

| created\_at          | datetime     | YES  |     | NULL    |       |

| updated\_at          | datetime     | YES  |     | NULL    |       |

| deleted\_at          | datetime     | YES  |     | NULL    |       |

| deleted             | varchar(36)  | NO   |     | NULL    |     |

| id                  | varchar(36)  | NO   | PRI | NULL    |       |<- **Primary key**

| access\_to           | varchar(255) | YES  |     | NULL    |       |

| access\_group\_id     | varchar(36)  | NO   | MUL | NULL    |       |<- **foreign key**

**3.share\_access\_group\_mapping table**

+---------------------+--------------+------+-----+---------+-------+

| Field               | Type         | Null | Key | Default | Extra |

+---------------------+--------------+------+-----+---------+-------+

| created\_at          | datetime     | YES  |     | NULL    |       |

| updated\_at          | datetime     | YES  |     | NULL    |       |

| deleted\_at          | datetime     | YES  |     | NULL    |       |

| deleted             | varchar(36)  | YES  |     | NULL    |       |

| id                  | varchar(36)  | NO   | PRI | NULL    |       |

| share\_id            | varchar(36)  | NO   | MUL | NULL    |       | 🡨 **foreign key**

| access\_group\_id     | varchar(36)  | NO   | MUL | NULL    |       | 🡨 **foreign key**

**Modify the share\_instances table**

**mysql> desc share\_instances;**

+----------------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+----------------------+--------------+------+-----+---------+-------+

| created\_at | datetime | YES | | NULL | |

| updated\_at | datetime | YES | | NULL | |

| deleted\_at | datetime | YES | | NULL | |

| deleted | varchar(36) | NO | | NULL | |

| id | varchar(36) | NO | PRI | NULL | |

| share\_id | varchar(36) | YES | MUL | NULL | |

| host | varchar(255) | YES | | NULL | |

| status | varchar(255) | YES | | NULL | |

| scheduled\_at | datetime | YES | | NULL | |

| launched\_at | datetime | YES | | NULL | |

| terminated\_at | datetime | YES | | NULL | |

| share\_network\_id | varchar(36) | YES | MUL | NULL | |

| share\_server\_id | varchar(36) | YES | MUL | NULL | |

| availability\_zone\_id | varchar(36) | YES | MUL | NULL | |

**| access\_rules\_status | varchar(255) | YES | | NULL | |>>>>>>**

**| access\_status\_message| varchar(255) | YES  |     | NULL    |       |<--new**

| replica\_state | varchar(255) | YES | | NULL | |

**Alternative**

2 new tables – but it will affect already existing “share\_access\_map” table.

Its like, merging “share\_access\_group\_mapping” and “access\_group\_entries” tables,

on key “access\_group\_id”.

**1.access\_groups table**

+---------------------+--------------+------+-----+---------+-------+

| Field               | Type         | Null | Key | Default | Extra |

+---------------------+--------------+------+-----+---------+-------+

| created\_at          | datetime     | YES  |     | NULL    |       |

| updated\_at          | datetime     | YES  |     | NULL    |       |

| deleted\_at          | datetime     | YES  |     | NULL    |       |

| deleted             | varchar(36)  | NO   |     | NULL    |     |

| Id                  | varchar(36)  | NO   | PRI | NULL    |       |>>

| project\_id          | varchar(255) | YES  |     | NULL    |       |

| access\_type         | varchar(255) | YES  |     | NULL    |       |

| name                | varchar(255) | YES  |     | NULL    |       |

| description         | varchar(255) | YES  |     | NULL    |       |

---------------------------------------------------------------------

**mysql> desc share\_access\_map**

 +----------------+--------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+-----------------+--------------+------+-----+---------+-------+

| created\_at | datetime | YES | | NULL | |

| updated\_at | datetime | YES | | NULL | |

| deleted\_at | datetime | YES | | NULL | |

| deleted | varchar(36) | YES | | NULL | |

| id | varchar(36) | NO | PRI | NULL | |

| share\_id | varchar(36) | NO | MUL | NULL | |>Foreign Key

| access\_type | varchar(255) | YES | | NULL | |>> Need to decide its place

| access\_to | varchar(255) | YES | | NULL | |

| access\_level | varchar(2) | YES | | NULL | |>> needs to decide its place

 access\_group id | varchar(36)  | NO   | | NULL    |     |>Foreign key

+--------------+--------------+------+-----+---------+-------+---

**Problem of redundancy here:-**

But for an access group – let’s say 5 entries in an access\_group. Now let’s say 2 shares mapped to this AG, whole 5 entries will be repeated for two shares.

This was optimized by refactor in first db schema proposed.

Considering first version of proposed db schema

**CLI & REST API Request & Response**

**For Access Group**

* **Create an access group:-**

**manila access-group-create "AG1" --description "First Access Group" –access\_type “user”**

+-------------+--------------------------------------+

|   Property  |                Value                 |

+-------------+--------------------------------------+

|      id     | f3b4faaa-8617-4d71-b50d-5c92e267bfe7 |

|  created\_at |      2014-08-06T15:23:05.139305      |

|  updated\_at |                 None                 |

|  project\_id |   6807e01e236c4dd1bb499687317fc3ee   |

| description |            First Access Group        |

|     name    |              AG1                |

| access\_type |              user                  |

| entry\_count |                0                     |

+-------------+--------------------------------------+

**POST /v2/<tenant\_id>/access-groups**

Content-Type: application/json

Accept: application/json

JSON Request Body:

{

    "access-group": {

       "description": "First Access Group", (optional)

       "name": "AG1", (required)

       "access\_type": "user",

   }

}

JSON Response:

{

    "access-group": {

       "id": "f3b4faaa-8617-4d71-b50d-5c92e267bfe7"

       "created\_at": "2013-10-02T09:58:12.000000",

       "updated\_at": "2013-10-02T09:58:12.000000",

       "project\_id": "myproject",

       "description": "First Access Group",

       "name": "AG1",

       "access\_type": "user",

 “entry\_count”: “0”,

    }

}

**List access groups with detail**

**manila access-group-list**

+--------------------------------------+---------+------+-------------------+

|                  ID                  |   Name  | Type |  Description  |

+--------------------------------------+---------+------+-------------------+

| f3b4faaa-8617-4d71-b50d-5c92e267bfe7 | AG1 |  user| First Access Group|

| f3b4faaa-8617-4d71-b50d-5c92e267bfe8 | AG2 |  ip | Second Access Group +--------------------------------------+---------+------+------------------

**GET /v2/<tenant\_id>/access-groups**

Accept: application/json

JSON Response:

{

    "access-groups": [

    {

      "id": "f3b4faaa-8617-4d71-b50d-5c92e267bfe7"

       "name": "AG1",

       "access\_type": "user",

 "description": "First Access Group",

   },

 {

      "id": "f3b4faaa-8617-4d71-b50d-5c92e267bfe8"

       "name": "AG2",

       "access\_type": "ip",

 "description": "Second Access Group",

   },

    ]

}

**Show a single access groups**

manila access-group-show "AG1"

+-------------+---------------------------------------+

|   Property  |                Value                 |

+-------------+---------------------------------------+

|      id     | f3b4faaa-8617-4d71-b50d-5c92e267bfe7 |

|  created\_at |      2014-08-06T15:23:05.139305      |

|  updated\_at |                 None                 |

|  project\_id |   6807e01e236c4dd1bb499687317fc3ee   |

| description |            First Access Group        |

|     name    |              AG1                |

| access\_type  |              user                  |

| entry\_count |                2                     |

+---------------+-------------------------------------------+

**GET /v2/<tenant\_id>/access-groups/<group-uuid>**

Accept: application/json

JSON Response:

{

    "access-group": {

       "id": "f3b4faaa-8617-4d71-b50d-5c92e267bfe7"

       "created\_at": "2013-10-02T09:58:12.000000",

       "updated\_at": "2013-10-02T09:58:12.000000",

       "project\_id": "myproject",

       "description": "First Access Group ",

       "name": "AG1",

       "access\_type": "user",

       "entries": [

     {

 "id": "84bc33b5-590c-4476-8295-02e0a4687715"

 "created\_at": "2013-10-02T09:58:12.000000",

 "updated\_at": "2013-10-02T09:58:12.000000",

 "access\_to": "admin"

 “access\_level: “rw”

},

{

 "id": "84bc33b5-590c-4476-8295-02e0a4687716"

 "created\_at": "2013-10-02T09:58:12.000000",

 "updated\_at": "2013-10-02T09:58:12.000000",

 "access\_to": "demo"

 “access\_level: “ro”},

       ]

   }

}

**Doubt**

1) Should access\_level be associated with access\_group or with access\_group entry? –

For this group of users – give rw to all of the entries

OR

access\_level should be a field associated with access\_entry only ?

Which says for user “A” give “rw” access to the share1 and for user “B” give “ro” access to share1.

But then there is no meaning of creating access-group in such case.

**Update an access group**

manila access-group-update "AG1" --description "Web servers"

+-------------+--------------------------------------+

|   Property  |                Value                 |

+-------------+--------------------------------------+

|  created\_at |      2014-08-06T15:25:45.000000      |

| description |             Web servers              |

| entry\_count |                  0                   |

|      id     | f3b4faaa-8617-4d71-b50d-5c92e267bfe7 |

|     name    |               AG1                |

|  project\_id |   6807e01e236c4dd1bb499687317fc3ee   |

|     type    |                user                  |

|  updated\_at |      2014-08-06T15:25:53.492672      |

+-------------+--------------------------------------+

**PUT /v2/<tenant\_id>/access-groups/<group-uuid>**

JSON Body

{

    "access-group": {

       "description": "New description",

       "name": "newname",

   }

}

JSON Response:

{

    "access-group": {

       "id": "f3b4faaa-8617-4d71-b50d-5c92e267bfe7"

       "created\_at": "2013-10-02T09:58:12.000000",

       "updated\_at": "2013-10-02T09:58:12.000000",

 "project\_id": "myproject",

       "description": "New description",

       "name": "newname",

       "type": "user",

   }

}

**Delete an access group**

manila access-group-delete "AG1"

**DELETE /v2/<tenant\_id>/access-groups/<group-uuid>**

**For Access Group Entries**

**Add an entry to an access group**

**manila access-group-entry-create "AG1" “user” “demo” “rw”**

+------------------------------------------+-----------+------------+--------------+

|                  ID                  |Access\_type| Access\_to |  Access\_level|

+----------------------------------------------------------------------------------|

| 84bc33b5-590c-4476-8295-02e0a4687715 | user | demo | rw |

+----------------------------------------------------------------------------------+

**POST /v2/<tenant\_id>/access-groups/<group-uuid>/entries**

JSON Request body:

{

    "entry": {

"access\_type": "user"

"access\_to": "demo"

"access\_level": "rw"

}

}

JSON Response:

{

    "entry": {

"id": "84bc33b5-590c-4476-8295-02e0a4687715"

"created\_at": "2013-10-02T09:58:12.000000",

"updated\_at": "2013-10-02T09:58:12.000000",

"access\_type": "user"

"access\_to": "demo"

"access\_level": "rw"

}

}

*\*\*Creating an entry means the effect of changed mapping will be on all associated shares. Creating an access rule entry will trigger update\_access on all the shares associated with the access\_group. With this new access\_rule in “add\_rule” field of update\_access function.*

**Remove an entry from an access group**

**manila access-group-entry-delete 84bc33b5-590c-4476-8295-02e0a4687715**

**DELETE /v2/<tenant\_id>/access-groups/<group-uuid>/entries/<entry-uuid>**

*\*\*Deleting an entry means the effect of changed mapping will be on all associated shares. Deleting an entry will trigger update\_access on all the shares associated with the access\_group. With this new access\_rule in “delete\_rule” field of update\_access.*

**List entries from an access group**

**manila access-group-entry-list "AG1"**

+------------------------------------------------------------------+

| ID  | access\_group\_id | access\_type  | access\_to | access\_level |

+--------------------------------------+-----------+---------------+

| id1 | AG1\_id |  user | admin | rw |

| id2 | AG1\_id |  user | demo | rw |

|-----+------------------------------------------------------------+

**GET /v2/<tenant\_id>/access-groups/<group-uuid>/entries/detail**

Accept: application/json

JSON Response:

{

    "entries": [

    {

      "id": id1

 “access\_group\_id”: AG1\_id

       “access\_type": "user",

 “access\_to": "admin",

 “access\_level": "rw",

   },

 {

      "id": "id2”

 “access\_group\_id”: AG1\_id

       "access\_type": "user",

 “access\_to": "demo",

 “access\_level": "ro",

   },

    ]

}

**For Share Access Group Mapping**

**Associating an access\_group with a share - Allow access(cli)**

**manila access-allow "Share 1" access-group "AG1"**

**POST /v2/<tenant\_id>/shares/<share\_id>/action**

Accept: application/json

**Action = allow\_access**

JSON Request

{

  "allow\_access": {

     "access\_to": AG1\_id, <--access\_group\_id

   }

}

JSON Response

{

    "access": {

       "created\_at": "2013-10-02T10:29:36.539532",

       "updated\_at": null,

       "deleted\_at": null,

       "deleted": false,

       "id": "%access\_mapping\_id%",

       "share\_id": "%share\_id%",

       "access\_to": AG1\_id,

   }

}

An entry will be added to **“share\_access\_group\_mapping”** table for given share\_id and access\_group\_id.

Lets say, there are few entries in share\_access\_map for this share\_id.

Then, result\_rules = [entries in access\_group\_id AG1] union [set of entries in share\_access\_map table for this share\_id] will be sent to driver like this.

self.driver.update\_access(

 context,

 share\_instance,

 **result\_rules,>>>>>>>>>>>>>>>>>> all rules**

 add\_rules=add\_rules,>>>> rules present only in AG1

 delete\_rules=delete\_rules,

 share\_server=share\_server

 )

If there is an error in applying rules, due to any reason, “access\_rule\_status” for share will reflect that with a descriptive error message.

**Removing group access to a share – Deny Access**

**manila access-deny "Share 1" “AG1”**

This is no different than current behavior, however, manila will now need to look for the access rule in both the “share\_access\_mapping” and “share\_access\_group\_mapping” tables

**POST /v1/<tenant\_id>/shares/<share\_id>/action**

Accept: application/json

JSON Request

{

     "os-deny\_access": {

       "access\_id": "%access\_mapping\_id%"

   }

}

If it’s found in share\_access\_mapping, it will be just what it currently does.

Else

if it’s found in share\_access\_group\_mapping table, mapping will be deleted for this share from table and for this share

Result\_rules = [set of access\_rule entries, for rest of access group mappings, for this share\_id, in share\_access\_group\_mapping table] union [set of rules in share\_access\_map table for this share\_id]

will be sent to driver like this.

self.driver.update\_access(

 context,

 share\_instance,

 **result\_rules,>>>>>>>>>>>>>>>>>> all rules**

 add\_rules=add\_rules,

 delete\_rules=delete\_rules, >>> will contain specific rules to be deleted

 share\_server=share\_server

 )

If there is an error in applying rules, due to any reason, “access\_rule\_status” for share will reflect that with a descriptive error message.

**List Access Rules mapping for a share**

**manila access-list "Share 1"**

**POST /v1/<tenant\_id>/shares/<share\_id>/action**

Accept: application/json

JSON Request

{

 "access\_list": null

}

JSON Response:

{

 “access\_list”: [

    {

"id": "84bc33b5-590c-4476-8295-02e0a4687715"

"created\_at": "2013-10-02T09:58:12.000000",

"updated\_at": "2013-10-02T09:58:12.000000",

"access\_to": "127.0.0.1"

},

{

"id": "84bc33b5-590c-4476-8295-02e0a4687716"

"created\_at": "2013-10-02T09:58:12.000000",

"updated\_at": "2013-10-02T09:58:12.000000",

"access\_to": "1.1.1.1"

},

]

}

**Access list will be obtained consulting both mapping tables.**

**Synchronizing access rules with a shares**

\*\* Tells Manila to ensure the current access rules are being enforced in the backend

POST /v1/<tenant\_id>/shares/<share\_id>/action

Accept: application/json

JSON Request

{

    "access\_sync": {}

}

Whiteboard

(vponomaryov):

    1) Need add next values to the "API Resources":

        - list of shares that uses particular access group <-- added to future plans

        - list of groups for share <-- added to future plans>>>>>>>>>>>>

        - associate SG to share <-- added to future plans

        - dissociate SG from share<-- added to future plans

    2) How many groups can be assigned to share?

    -No explicit limit

    3) If more than 1 SG can be assigned to the share, then what do we expect if several groups assigned with crossing rules?

    -TBD with SG work

    4) How to handle it if granting access is successful for some in the access group, but fails for others?

    - If this occurs, the shares 'access\_status' and 'access\_status\_message' fields will be updated with reasons for the error. The user can then fix the issue and use the 'os-access\_sync' API to have manila try again.

**List access groups mapped to a share**

**POST /v1/<tenant\_id>/shares/<share\_id>/action**

Accept: application/json

JSON Request

{

    "list\_access\_groups": {}

}

JSON Response:

{

    "access-groups": [

    {

      "id": "f3b4faaa-8617-4d71-b50d-5c92e267bfe7"

       "name": "AG1",

       "access\_type": "user",

 "description": "First Access Group",

 “share-id”: “share\_id”

   },

 {

      "id": "f3b4faaa-8617-4d71-b50d-5c92e267bfe8"

       "name": "AG2",

       "access\_type": "ip",

 "description": "Second Access Group",

 “share-id”: “share\_id”

   },

    ]

}

**Alternative:**

**manila access-group-list "Share 1"**

* -EX:
* GET /access-groups?share\_id=f3b4faaa-8617-4d71-b50d-5c92e267bfe7

**List of shares that uses particular access group:**

**manila list –access\_group AG1**

GET /v2/<tenant\_id>/access-groups/<group-uuid>/action

Accept: application/json

JSON Request

{

    "list\_shares": {}

}

JSON Response:

{

    "shares": [

{

       "access\_group\_id": <group-uuid>

       "share\_id": "share\_id1",

       "share\_name": "share1",

       "share\_decsription": "First Share",

       },

{

       "access\_group\_id": <group-uuid>

       "share\_id": "share\_id2",

       "share\_name": "share2",

       "share\_decsription": "2nd Share",

       },

 ]

}

**Alternative**

**manila list –access\_group AG1**

* - EX:
* GET /shares?access\_group=f3b4faaa-8617-4d71-b50d-5c92e267bfe7

**Doubts As per use case**

**Case1 – Contradictory rule is requested to already present rule.**

Ques1 :-

lets say share1 has a access rule already applied to it as

Access entry for Share1 ->( user ,admin, rw) (In share\_access\_map)

Now an AG1 with an (user, admin, ro) is created.

API request comes as allow\_access(share1, AG1).

Share1 is asked to map with AG1, what should happen?

I think its same as case if we apply two contradictory rules even without access

Groups- we will get an error access rule for user admin is already applied.

We need to deny it first to change it.

**Case2:- Two compatible access-groups mapping request got out of order before**

**reaching backend driver.**

Two api requests for two access\_groups where two access rules are compatible,

came such as

AG1 has (user, admin, rw)

AG2 has (ip, 10.1.1.1, None)

2 Api requests come in this order

allow\_access(share1, AG1) – Req1

allow\_access(share1, AG2) – Req2

let’s say these request reach driver, out of order, that is backend driver receives, first Req2 then Req1.

[**https://etherpad.openstack.org/p/manila-access-groups-api-proposal**](https://etherpad.openstack.org/p/manila-access-groups-api-proposal) **line 324 says**

\*\*NOTE\*\* the backend driver will always have to update the export rules to whatever the current state reflected in the database may be, to avoid race conditions with changes to the access group. For example, if someone makes two requests to the api and the messages arrive out of order to the driver, the end state of the first message will incorperate both changes and the second message will be a noop

That is It says

Req2 -> tells the driver to update export rules as per current state in DB.

Current state in DB is -> for share1 -> add two rules for AG1 and AG2 both.

Req1 -> for driver is noop

**Doubt :-**

What I see is Req1 creates a access\_mapping in db for share1, AG1 and gets access\_rules from DB and passes that “only new” set of rules to driver layer.

**Currently driver doesn’t work on current state of DB**

 @add\_hooks

 @utils.require\_driver\_initialized

 def **allow\_access**(self, context, share\_instance\_id, access\_rules):

 """Allow access to some share instance."""

 add\_rules = [self.db.share\_access\_get(context, rule\_id)

 for rule\_id in access\_rules] >>

2. Adding/Deleting of several access rules - 'access\_rules' contains

 all access\_rules, 'add\_rules' and 'delete\_rules' contain rules which

 should be added/deleted. Driver can ignore rules in 'access\_rules' and

 apply only rules from 'add\_rules' and 'delete\_rules'.

Hence driver will only apply rules from req1 only.

My understanding is Req1 driver stage won’t apply req2’s rules on share.

**Case3:- Access group contains multiple entries, where backend could apply only a**

**few successfully, a few could not be applied.**

Ques3: A share allow\_access(share1, AG1)

AG1 has 3 access rule entries

[{ip, 1.1.1.1, none}, {ip, 2.2.2.2, None}, {ip, 3.3.3.3, None}]

Changes made to share\_access\_group\_mapping table.

Backend could not apply 3rd rule successfully{ip, 3.3.3.3, None}

Result status will be marked in access\_rules\_status field, access\_status\_message will show the reason.

