General Concepts

1. API Service and HTTPS

The AGM API service is available through HTTPS only. If a web request comes to HTTP port 80, it will be redirected to the HTTPS port 443 automatically.

2. API Payload

2.1. Format

JSON is the only supported format for both request (inbound) and response (outbound) entity bodies, except certain streaming APIs for downloading/uploading raw data. UTF-8 encoding is the only supported encoding for both URLs and entity bodies.

3. API Methods

3.1. Supported HTTP Methods

Method Name	Description
GET	Get an individual resource or a list of resources
POST	Create a resource or perform an operation on a resource
PUT	Incrementally update an individual resource
DELETE	Remove resource(s)
OPTIONS	Get meta information about the resource (not fully implemented)
HEAD	Get a count of resources that meet filter criteria

3.2. PUT semantics

In AGM, HTTP PUT is used to update rather than replace existing resources. Any attribute that is absent in the payload retains its current value.

Due to implementation limitation, special handling is needed for the following cases:

Null valued Attribute

AGM API service treats a null valued attribute and absence of the attribute equivalently. Therefore, there is no way to easily unset an attribute through a PUT call. The workaround is provided case by case in the API specification details.

For example:

```
PUT .../resource
{
    …
```

```
   "attribute1" : null,
   …
}
```

API service will consider attribute1 as absent in the request. Therefore attribute1 will retain its current value.

3.3. HEAD semantics

HEAD is used to get an **estimated** count of resources that meet filter criteria. HTTP header **Actifio-Count** represents the estimated count in the response:

```
Actifio-Count: <number>
```

Here are a couple of examples:

Get estimated number of total backup resources:

```
Request:
HEAD .../backup

Response:
Actifio-Count: 9771
```

Get an estimated number of backup resources that are from an application with an ID of 891061:

```
Request:
HEAD .../backup?limit=10&filter=appid:==891061

Response:
Actifio-Count: 37
```

Note that any pagination control such as limit=10 in the query string is ignored if supplied.

3.4. OPTIONS (Not fully implemented)

OPTIONS is used to get metadata about an endpoint. The most common use is to discover the filter and sort fields that are available.

The following example shows that the *role* endpoint supports filtering, sorting, and pagination. It also lists all fields that can be filtered or sorted:

Request:

```
OPTIONS .../role

Response:
{
   "GET(list)" : {
          "filterable" : true,
          "sortable" : true,
          "pageable" : true,
          "sortablefields" : [ "id", "name",
"description", "createdate" ],
          "filterablefields" : [ {
                     "field" : "organizationid",
                     "type" : "Long"
          }, {
```

```
            "field" : "name",
            "type" : "String"
           }, {
            "field" : "description",
            "type" : "String"
           }, {
            "field" : "createdate",
            "type" : "Long"
           }, {
            "field" : "id",
            "type" : "Long"
           } ]
   }
}
```

3.5. HTTP Status Codes

HTTP Status Code	Description
200	OK with a response entity body.
	Object created with a response entity body. Due to implementation limitations, 200 instead of 201 is used.
204	OK with no response entity body.
400	Bad request. Usually it comes with a response entity body to represent the details of the error. Same applies to the other 4XX and 5XX codes, except 401 and 403.
401	Authentication required (either no session id specified or the id is invalid/expired) WWW-Authenticate header is returned with either “Basic” scheme (for login API) or “Actifio” scheme (for any other APIs)
403	Authorization failure. Operation is not permitted in the user session identified by the session id.
404	Nonexist resource.
405	Particular HTTP method is not allowed for this resource
409	Content conflict (violation of constraint)
415	Content-Type from the request is not acceptable by the server (application/json is the only one supported).
500	Server error.
502	The HTTPS Web server is up but the API service is not available. This could happen during the product upgrade process.

3.6. Error Message

In case of HTTP code 4xx and 5xx (except 401 and 403), the response entity body is as follows:

```
{
  “err_code”: <err_code>,
  “err_message”: <err_message, optional>
}
```

3.7. Access Control

Authentication

Most of the APIs don't allow anonymous access. Access requires Google SSO login that requires a bearer token. Clients need to get a session id from a successful login and pass that session id along with the bearer token with subsequent API calls.

An HTTP response with status code 401 indicates that authentication is required but either the bearer token or the session id is either missing from request or is invalid/expired.

To generate a bearer token follow the below steps

- Grant a service account Cloud BackupDR Admin role.
- Using the same service account, authenticate to gcloud shell. You might need the key file of the service account. Example:

  ```
  gcloud auth activate-service-account $service_account
  --key-file=$full_path_to_key_file
  ```

- Fetch `oauth2ClientId` attribute from the response of below command:

  ```
  curl -H "Authorization: Bearer $(gcloud auth print-
  access-token)"
   -H "Content-Type: application/json"
   https://backupdr.googleapis.com/v1/projects/<consumer_project_id>
  /locations/<consumer_project_region>/managementServers
  ```

  Replace the placeholders `consumer_project_id` and `consumer_project_region` with your project details.
- Use the `oauth2ClientId` attribute from the output of above command to generate bearer token using the below command.

  ```
  gcloud auth print-identity-token --audiences=
  <oauth2ClientId>
  ```

To login, make an HTTP POST call to the *session* resource endpoint with HTTP Authorization header using bearer token (as generated above) that contains RSA256 encoded jwt token:

```
Authorization: Bearer <bearer_token>
```

An example of login call is as follows:

```
POST .../session
Authorization: Bearer eyJam9obl9zbWl0aDpwYXNzd29yZA
```

A session_id attribute will be returned in the response, along with other attributes. The value of session_id is the session id which represents the login context of the specific user. The client supplies it in backupdr-management-session header with Actifio scheme and bearer token in Authorization header for all subsequent API calls:

```
backupdr-management-session: Actifio <session_id>
Authorization: Bearer <bearer_token>
```

session ids expire after 1 hour of inactivity. By default, any API calls carrying the session id will extend the session by another 1 hour. If the following request header with value of true (case insensitive) is specified with an API call, the session lifecycle will not be extended by this particular API call:

```
Actifio-No-Session-Refresh: true
```

An optional explicit logout can also be performed with a HTTP DELETE call to the *session/current* endpoint. An example of logout call is as follows:

```
DELETE .../session/current
backupdr-management-session: Actifio <session_id>
```

Authorization

AGM uses the same access control mechanism as the VDP appliances. When an API call carries a valid bearer token in its authorization header and session id in backupdr-management-session header, the API service will evaluate the user context determined by the session id, to see if sufficient access rights are given to the user for the particular operation. In addition to the rights, resources are only visible to the user if both the resources and user belong to the same organizations.

If a user's rights don't satisfy an API's requirement, a HTTP response with status code 403 will be returned to indicate that request is not permitted.

If some resources are not visible to a user due to the organization partitions, those resources will not be included in the response for a list view API. If the API is to retrieve or operate on those resources explicitly (e.g., if the resource id is leaked) then an HTTP response with status code 404 will be returned.

4. Resources

In AGM API, management objects are organized in resources.

4.1. Top Level Resources

Top level resources are those that have dedicated API endpoints so that they can be queried directly. Non-top level resources reside inside other resource objects. Usually they are not exposed to direct access.

Top resources are listed as follows grouped in various categories:

Client Sessions

Session - for session management, including login and logout

Applications, Hosts and Related Objects

Application - Applications are the fundamental unit managed by AGM.. They include virtual machines, databases, file systems, and cloud instances. Most

applications are discovered by the VDP appliances, although some types are created by users. An application always resides on a host. A host can potentially contain multiple applications.

Consistency Group (CG) - With some restrictions, multiple applications on the same host can be grouped in a CG. A CG is treated like a single application. When protected, it runs the same capture or replicate job and it produces a single backup image for all member applications with identical consistent-date.

Logical Group (LG) - With some restrictions, multiple applications registered in the same VDP appliance can be grouped in a LG. Unlike CG, the members of LG are loosely coupled. Member applications in a LG are sharing the same SLA when protected. However each of the applications runs its own capture or replicate job and each of them produces its own backup image.

Host - Hosts are the physical or virtual machines (servers) that applications run on.

Backups and Job Tracking

Backup - It represents a backup image or active image. An active image represents the disks used in mount, clone, live-clone, and fail-over operations.

Job - Jobs perform operations on applications, such as taking snapshots, copying images to OnVault or Dedup pools, or mounting images.

SLA and Related Objects

SLT - SLA policy template. It is a set of policies that run on a schedule.

- **Policy** defines a job type, schedule, and retention for a backup action. For example, a policy could require taking a snapshot between 10 pm and 6 am and retaining it for 3 days. A policy always belongs to a single SLT. Policies are not a top level resource.
- **Policy option** is an additional key/value pair that defines a specific behavior for a policy. There are many policy options. Most of them have different applicable contexts, based on application's types and other conditions. A policy option always belongs to a single policy. It is not a top level resource.

SLP - SLA profile. It defines remote VDP, disk pools (including OnVault pools) and other resources that a SLA requires.

SLA - Service Level Agreement. It represents the protection of a specific target (application or CG). An SLA is the combination of a protection target, SLT, and SLP.

Diskpools and Related Objects

Diskpool - A diskpool represents storage. There are various types of diskpools, such as snapshot pools, dedup pools, and OnVault pool, etc. Diskpool can be configured to use an external storage array.

Array - Arrays represent external storage arrays used to configure external snapshot pools.

VDP Appliances

Cluster - Clusters represent VDP appliances which could be CDS, Sky, or CDX.

Event - It represents an event that was raised from an VDP appliance.

Users, Roles, and Organizations

User - It represents a user entity. Multiple roles and organizations can be assigned to a single user.

Role - It represents a role. Multiple rights can be assigned to a single role.

Organization - It represents an organization. Organization can be nested.

Auditing Records

Audit - It represents an audit record of AGM itself.

4.2. Resource Consolidation

There are valid practices in which the same resource is registered in multiple VDP appliances. In that case, a unified resource presents in AGM rather than multiple copies coming from the appliances. This is called resource consolidation. The unified resource in API payload contains a “sources” attribute which is a JSON array to reflect the potentially different properties carried by particular appliances.

For example, a VM host can be registered/discovered from two VDP appliances. Therefore two individual host objects are created on the two appliances separately. If AGM manages both of the appliances, then there will be one consolidated host resource on AGM to represent this VM host, with “sources” carrying appliance-specific information such as the resource id on the appliance.

Not all types of resources are consolidated. For example, disk pools are specific to their hosting VDP appliances therefore each disk pool resource in AGM represents a unique disk pool on a single appliance, even if multiple such disk pools are pointing to the same external storage array.

5. Pagination, Filtering, and Sorting

5.1. List View API

Some top level resources have GET APIs that return a list of resources as response. They are referred to as list view APIs. The response entity body has a format described as follows:

```
{
```

```
  "count" : <integer, number of resources returned>,
  "items" : [
     …
  ]
}
```

List view APIs support pagination. Please note that if sorting is not explicitly defined in the request, then the ordering of the resources is undetermined. Pagination still works though.
Query parameters “offset” and “limit” can be used to control the navigation of pages.

Offset

It indicates the starting index number of the potentially sorted resources. The value defaults to 0. If an value is supplied which is bigger than the total amount, a response with empty items will be returned:

```
{
  "count" : 0,
  "items" : [ ]
}
```

Limit

It indicates the upper limit of the maximum items that can be returned in the particular response.
Please note that the actual upper limit is decided by the API service based on the capacity of the server. It may be smaller than what is specified in this parameter. It defaults to 1000.

Examples

The following list view API returns up to 1000 applications with default offset of 0:

```
GET .../application?limit=1000
```

The following list view API returns up to 1000 application with offset of 1000 (i.e. this is the 2nd page to the response of the previous API call):

```
GET .../application?limit=1000&offset=1000
```

5.2. Filtering

Some list view API supports filtering. To determine if a particular top resource supports filtering, use OPTIONS API as described in HTTP Method session. The optional query parameter “filter” can be used to control the filtering of the resources:

Syntax

Filters can be multi-valued. The syntax of filters are defined as follows:

```
filters ::= filter["&" filters]*
```

```
filter ::= "filter=" attr_name ":" URLEncode(filter_operator
attr_value)
filter_operator ::= exact_match | contains | greater_eq |
less_eq | bitwise_and
exact_match ::= "=="
contains ::= "=|"
greater_eq ::= ">="
less_eq ::= "<="
bitwise_and ::= "=b"
```

Restrictions

exact_match applies to numeric, enum, string and timestamp types of attributes.
contains only applies to string.
greater_eq and less_eq apply to numeric and timestamp.
bitwise_and applies to boolean.
Any particular attr_name can only appear once in filters except for the following situations:

1. A greater_eq and less_eq filter with the same attr_name can both exist if the attr_value is a numeric type. Such two filters will define a range of the values.
2. Multiple exact_match filters with the same attr_name define a union of conditions. E.g. filter=attr1:==value1&filter=attr1:==value2 will match the resources that have either value1 or value2 as the value in their attr1.

Characters in attr_value are literal. Wildcard characters are not supported.

Organization Filter

As mentioned in the access control section, only resources that belong to the same organizations of the user (identified by the session id) will be returned in the list view API. In addition, explicit organization filer “organizationid” can be supplied to further narrow down the returning resources. The organization filter is available to all top level resources that allow organization assignments.

Examples

The following API call gets all backup resources that are snapshot images:
```
GET .../backup?filter=jobclasscode:==1
```

The following API call gets all applications that have “app” in their names:
```
GET .../application?filter=appname:=|app
```

Please note that URLEncoding needs to be applied properly as mentioned in the filter syntax section.

5.3. Sorting

Syntax

Some list view APIs support sorting through query parameters defined as follows.

```
sort ::= "sort=" URLEncode(ordered_sort_string)
ordered_sort_string ::= attr_name ":" ("asc" | "desc") ["," 
                        ordered_sort_string]
```

Examples

The following API call lists all hosts (up to the limit that API service determines) sorted by their AGM ids in ascending order:
`GET .../host?sort=id:ASC`

5.4. Free Text Search

Some list view APIs support free text search through query parameter “keyword”.

Syntax

```
keywords :: = keyword_item ["&" keywords]
keyword_item ::= "keyword=" URLEncode(string)
```

When keywords are supplied in the query against a supported resource, they are searched in case-insensitive mode for multiple attributes that are predefined with the product. The qualified search results are those items that have at least one predefined attribute containing at least one of the keywords supplied by the request. Free text search filters can be used in addition to the filtering/sorting/pagination features.

Examples

The following API call lists all hosts that have something to do with “ABC” case insensitively:
`GET .../host?keyword=ABC`

Predefined Searching Attributes

Resources that supports free text search	Predefined free text search attributes
Application	"vmuuid", "hostname", "appname", "description", "networkname", "pathname", "sqlinstance", "sqlserverag"
Backup	"vmuuid", "backupname", "label", "appname", "hostname", "slpname", "sltname", "policyname", "mountedmap"
Event	"clustername", "objecttype", "messagetext", "errorcode", "errormessage", "jobname", "appname"
Host	"uniquename", "hostname", "friendlypath", "svcname", "description", "ipaddress"
Job	"jobname", "hostname", "targethost", "appname", "sltname", "policyname", "label"