

Dremio ACL Organizer

User Guide

March 22, 2021

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 0.1.0 | 07/12/2020 | Deane Harding | Initial release |
| 0.2.0 | 07/18/2020 | Deane Harding | Introduced option to specify a user that can be used to generate a default ACL in cases were PDSs have no ACLs defined. |
| 0.3.0 | 07/22/2020 | Deane Harding | Added ability to dump all current PDS ACLs to a directory. Added ability to write the report of all incorrect ACLs to a file. |
| 0.4.0 | 07/23/2020 | Deane Harding | Added ability to specify default ACL for a group |
| 0.5.0 | 10/15/2020 | Deane Harding | Added ability to dump all (or select) space, folder and VDS ACLs to a directory |
| 0.5.1 | 10/16/2020 | Deane Harding | Documentation fix to specify –config in examples |
| 0.6.0 | 11/18/2020 | Deane Harding | Added ability to create ACLs at a data source folder level based on a superset of all ACLs of PDSs inside the folder |
| 0.6.1 | 11/18/2020 | Deane Harding | Documentation fix to provide accurate acls-to-folder examples. |
| 0.6.2 | 11/24/2020 | Deane Harding | Introduced ability to set ACLs on database folder from config file. |
| 0.7.0 | 03/22/2021 | Deane Harding | Introduced ability to set ACLs on Spaces, Folders and VDSs |

Related Documents

|  |  |
| --- | --- |
| **Name** | **Version** |
|  |  |

Supported Versions

|  |  |
| --- | --- |
| **Name** | **Version** |
| Dremio | 4.0.0 or later |
| Python | 2.6 or later |

Table of Contents

[Introduction 4](#_Toc67327191)

[Purpose 4](#_Toc67327192)

[Setup and Configuration 5](#_Toc67327193)

[Prerequisites 5](#_Toc67327194)

[Install dremio\_acl 5](#_Toc67327195)

[Install from dremio\_acl.tar.gz 5](#_Toc67327196)

[View dremio\_acl Help Page 6](#_Toc67327197)

[Usage Details 7](#_Toc67327198)

[Options 7](#_Toc67327199)

[config.yaml 7](#_Toc67327200)

[Commands 8](#_Toc67327201)

[dump acl 9](#_Toc67327202)

[dump space-acl 9](#_Toc67327203)

[report acl 11](#_Toc67327204)

[update acl 12](#_Toc67327205)

[update acls-to-folder 15](#_Toc67327206)

[update space-acl 18](#_Toc67327207)

[ACL Definition File 20](#_Toc67327208)

# 

# 

# Introduction

## Purpose

The Dremio ACL Organizer tool, known as dremio\_acl, enables administrators to manage and maintain access control lists (ACLs) for Physical Data Sets (PDSs) in a data source, ensuring that the ACLs set against data sets in Dremio are consistent with the ACL definition data stored external to Dremio and read by dremio\_acl.

The tool provides six endpoints to administrators; the first is to enable reporting on the current state of ACLs in a specified data source in Dremio as compared to the external definition, using this endpoint administrators can receive a report of all Dremio PDSs whose ACLs differ from the configuration, however no changes are made to Dremio.

The second endpoint is similar to the first one, but when ACL differences are detected the tool will update the database folder or PDS object in Dremio with the ACLs in the external definition data for that object. If there is no definition data for an object but in Dremio some ACLs have been set against the object then the tool will revoke the ACLs from the object and will optionally apply a default ACL instead.

The third endpoint will dump all current ACLs for all PDSs in the specified data source to a file.

The fourth endpoint will dump all current ACLs for all spaces, folders and VDSs in a Dremio environment.

The fifth endpoint will generate a superset of ACLs from all PDSs inside a data source folder and will then update the data source folder with the superset of ACLs. Optionally the ACLs set against the PDSs can then be deleted automatically by the tool if required.

The sixth endpoint will apply user\group ACLS to spaces, folders and VDSs based on the ACLs defined in the external definition file for those objects. If there is no definition data for an object but in Dremio some ACLs have been set against the object then the tool will revoke the ACLs from the object and will optionally apply a default ACL instead.

# Setup and Configuration

## Prerequisites

Dremio\_acl is a Python-based tool and therefore one of the prerequisites prior to installing it is that Python 2.6 or greater is installed on the machine where dremio\_acl will run.

The host that dremio\_acl is executed on must have connectivity to send POST, PUT and GET requests to the Dremio Coordinator on port 9047, and it should contain 1GB of free memory for running the tool.

* Install Python 2.6
  + The following site provides an explanation if required:

<https://docs.python-guide.org/starting/installation/>

* Check the Python version to ensure it is at least 2.6:

|  |
| --- |
| python -V |

* Check the pip version to ensure it is installed:

|  |
| --- |
| pip -V |

* + In case pip is not installed:

|  |
| --- |
| python -m pip install pip |

## Install dremio\_acl

dremio\_acl is delivered as a compressed file called dremio\_acl.tar.gz Below are install instructions.

### Install from dremio\_acl.tar.gz

* Copy the tar file to the host VM
* Unpack the file into a directory of your choice, here we choose /opt/dremiotools

|  |
| --- |
| mkdir /opt/dremiotools  tar xvzf dremio\_acl.tar.gz -C /opt/dremiotools |

* Install dremio\_acl.

|  |
| --- |
| cd /opt/dremiotools/dremio\_acl  pip install . |

## View dremio\_acl Help Page

With dremio\_acl successfully installed it should be possible to execute the following command to see the initial help description:

|  |
| --- |
| dremio\_acl --help |

The command should produce the following output:

|  |
| --- |
| Usage: dremio\_acl [OPTIONS] COMMAND [ARGS]...  Use dremio\_acl to interact with Dremio's REST API to report on or update PDS ACLs  Options:  --config DIRECTORY Custom config file.  -h, --hostname TEXT Hostname if different from config file  -p, --port INTEGER Hostname if different from config file  --ssl Use SSL if different from config file  -u, --username TEXT username if different from config file  --password TEXT password if different from config file  --skip-verify skip verification of ssl cert  --help Show this message and exit.  Commands:  dump  report  update |

# Usage Details

The syntax for executing dremio\_acl on the command line is as follows:

dremio\_acl [OPTIONS] COMMAND [ARGS]...

## Options

Whenever dremio\_acl is executed, regardless of which mode of operation you are using it for there are several options that you need to specify before issuing a command. These options can be placed as individual flags on the command line or they can be placed into a configuration file and read in with a single flag.

The individual options are specified in the following table, these can also be viewed using the command dremio\_acl --help:

|  |  |
| --- | --- |
| **Option** | **Description** |
| --config | Directory location of a config.yaml configuration file that contains values for some or all of the options specified in this table. This can be either a relative or absolute path. The structure of this configuration file is documented below this table. |
| -h or --hostname | The hostname of the Dremio Coordinator we want to issue commands against. Default value is localhost |
| -p or --port | The port of the Dremio Coordinator we want to issue commands against. Default value is 9047 |
| --ssl | true/false value to indicate whether communication with the Dremio Coordinator needs to use SSL. Default value is false. |
| -u or –-username | The username of an administrative user in Dremio |
| --password | The password of an administrative user in Dremio |
| --skip-verify | If the Dremio Coordinator requires SSL, this true/false flag indicates that we wish to skip verification of the SSL certificate supplied by Dremio. Default value is true. |

### config.yaml

If the --config flag is set as an option, then dremio\_acl will attempt to read values from a config.yaml file in the directory path specified with the flag. The syntax of the config.yaml file is as follows:

|  |
| --- |
| auth:  type: basic  username: dremio  password: dremio123  hostname: localhost  port: 9047  ssl: false  verify: false |

## Commands

dremio\_acl provides six commands for its various modes of operation. Each of these commands is described in more depth in the following sub-sections.

|  |  |
| --- | --- |
| **Command** | **Description** |
| dump acl | Output a report of all ACLs for all PDSs in a specified data source into the specified directory on local disk. |
| dump space-acl | List all ACLs for spaces, folders and VDSs in a Dremio environment. The list will be written to a specified directory on the disk local to where dump space-acl is executed |
| report acl | Report on the current state of ACLs for PDSs in a specified data source in Dremio as compared to the external definition data. Report will be saved to a specified directory on local disk. |
| update acl | Update PDSs in Dremio to match the ACLs in the external definition data. If there is no definition data for a data set but in Dremio some ACLs have been set against the data set then the tool will revoke the ACLs from the data set and will optionally apply a default ACL instead. |
| update acls-to-folder | Read the ACLs of all PDSs in a specified data source folder\schema and generate a superset of these ACLs. The superset of ACLs will then be used to set the ACLs of the data source folder itself. Optionally, the command can be used to automatically delete the ACLs from all PDSs in the specified folder once the folder has been updated. |
| update space-acl | Update specified spaces, folder and VDSs in Dremio to match the ACLs in the external definition data. If there is no definition data for a space, folder or data set in the path specified but in Dremio some ACLs have been set against the object then the tool will revoke the ACLs from the object and will optionally apply a default ACL instead. |

### dump acl

The dump acl command is used to list all ACLs for all PDSs in a data source (or database/schema within a data source). The list will be written to a specified directory on the disk local to where dump acl is executed.

Issuing the command dremio\_acl dump acl --help gives us the following information about usage of this command:

|  |
| --- |
| Usage: dremio\_acl dump acl [OPTIONS] BASE...  BASE: base directory in Dremio where to start listing ACLs from. Space  separated. e.g. to start at a db within a source: MYSOURCE MYDB  Options:  -d, --dump-path PATH Path where the file containing a list of all ACLs  will be written [required]  --help Show this message and exit. |

These instructions indicate that there is one additional option applicable to the dump acl command and that is the following:

|  |  |
| --- | --- |
| **Option** | **Description** |
| -d | Mandatory directory path where the file containing the list of ACLs will be written to. |

The following command shows an example of how to list all ACLs in a data source called MyDataSource that has a schema called MySchema. The list will be written to a folder called /opt/dremiotools/reports

dremio\_acl --config /opt/dremiotools/conf dump acl -d /opt/dremiotools/reports MyDataSource MySchema

### dump space-acl

The dump space-acl command is used to list all ACLs for all spaces, folders and VDSs in a Dremio environment. The list will be written to a specified directory on the disk local to where dump space-acl is executed.

Issuing the command dremio\_acl dump space-acl --help gives us the following information about usage of this command:

|  |
| --- |
| Usage: dremio\_acl dump space-acl [OPTIONS] BASE...  BASE: optional base directory in Dremio where to start listing ACLs from. Space separated. e.g. to start at a folder within a space: MYSPACE MYFOLDER  Options:  -d, --dump-path PATH Path where the file containing a list of all ACLs  will be written [required]  -v, --include-vds Flag to include VDS ACLs in the dump  --help Show this message and exit. |

These instructions indicate that there are two additional options applicable to the dump space-acl command and that is the following:

|  |  |
| --- | --- |
| **Option** | **Description** |
| -d | Mandatory directory path where the file containing the list of ACLs will be written to. |
| -v | Optional flag indicating whether to also dump the ACLs for all VDSs inside the optional space and folder hierarchy specified by the BASE |

The following command shows an example of how to list all ACLs for a space called MySpace and all its associated sub-folders. This command will not output ACLs for the VDSs in the space. The list will be written to a folder called /opt/dremiotools/reports

dremio\_acl --config /opt/dremiotools/conf dump space-acl -d /opt/dremiotools/reports MySpace

The following command shows an example of how to list all ACLs for a folder called MyFolder (which resides in a space called MySpace) and all its associated sub-folders. This command will also output ACLs for the VDSs in the MyFolder hierarchy. The list will be written to a folder called /opt/dremiotools/reports

dremio\_acl --config /opt/dremiotools/conf dump space-acl -d /opt/dremiotools/reports -v MySpace MyFolder

**Please note:** If no value is provided for BASE, then the tool will capture ACLs for all spaces and folders in Dremio. Including -v will also capture all VDS ACLs in this scenario. For example:

dremio\_acl --config /opt/dremiotools/conf dump space-acl -d /opt/dremiotools/reports -v

### report acl

The report acl command is used to list all PDSs in a data source (or database/schema within a data source) whose ACL values do not match the ACLs specified for that PDS in an external definition file. The report will be written to a specified directory on the disk local to where report acl is executed.

Issuing the command dremio\_acl report acl --help gives us the following information about usage of this command:

|  |
| --- |
| Usage: dremio\_acl report acl [OPTIONS] BASE...  BASE: base directory in Dremio where to start comparing ACLs. Space  separated. e.g. to start at a db\schema within a source: MYSOURCE MYDB  Options:  -a, --acl\_file FILENAME Path to a file containing the ACL  definitions  [required]  -r, --report-path PATH Path where the file containing a list of all  PDSs with incorrect ACLs will be written  [required]  -g, --group-on-acl-empty TEXT Optional group to set ACL for if a PDS ACL  list is empty  -u, --user-on-acl-empty TEXT Optional user to set ACL for if a PDS ACL  List is empty  --help Show this message and exit. |

These instructions indicate that there are three additional option applicable to the report acl command and that is the following:

|  |  |
| --- | --- |
| **Option** | **Description** |
| -a | Mandatory directory path and filename containing ACL definition data for all PDSs that we want to set explicit ACLs against. The file is in JSON format. See the section called ACL Definition File for details on the required JSON structure. |
| -r | Mandatory directory path where the report file will be written to. |
| -g | Optional group name that the tool will use to generate a default ACL for. If this option is specified, then in cases where the PDS has no ACLs defined or if the PDS has ACLs defined but the PDS does not have an entry in the definition data file then the generated default ACL for this group will be suggested instead. Can be used alongside -u option if required. |
| -u | Optional username that the tool will use to generate a default ACL for. If this option is specified, then in cases where the PDS has no ACLs defined or if the PDS has ACLs defined but the PDS does not have an entry in the definition data file then the generated default ACL for this user will be suggested instead. Can be used alongside -g option if required. |

The following command shows an example of how to report on the ACL differences in a data source called MyDataSource that has a schema called MySchema, where the external ACL definition data is stored in a file called /opt/dremiotools/conf/acl\_defs.json. In addition, for any PDSs without any ACLs defined or if there is no reference to the PDS in the definition data file, this command will report on PDSs that would also get set with default ACLs for a user called adminuser. The report will be written to a directory called /opt/dremiotools/reports

dremio\_acl --config /opt/dremiotools/conf report acl -a /opt/dremiotools/conf/acl\_defs.json -u adminuser -r /opt/dremiotools/reports MyDataSource MySchema

### update acl

The update acl command is used to update all PDSs in a specified data source (or database/schema within a data source) whose ACL values do not match the ACLs specified for that PDS in an external definition file.

Issuing the command dremio\_acl update acl --help gives us the following information about usage of this command:

|  |
| --- |
| Usage: dremio\_acl update acl [OPTIONS] BASE…  BASE: base directory in Dremio where to start applying ACLs to. Space  separated. E.g. to start at a db within a source: MYSOURCE MYDB  Options:  -a, --acl\_file FILENAME Path to a file containing the ACL definitions  [required]  -g, --group-on-acl-empty TEXT Optional group to set ACL for if a PDS ACL  list is empty  -u, --user-on-acl-empty TEXT Optional user to set ACL for if a PDS ACL  list is empty  -s, --source-only Flag to only set ACLs at database level,  omit setting PDS ACLs  --help Show this message and exit. |

These instructions indicate that there are three additional options applicable to the update acl command and those are the following:

|  |  |
| --- | --- |
| **Option** | **Description** |
| -a | Mandatory directory path and filename containing ACL definition data for all PDSs that we want to set explicit ACLs against. The file is in JSON format. See the section called ACL Definition File for details on the required JSON structure. |
| -g | Optional group name that the tool will use to generate a default ACL for. If this option is specified, then in cases where the PDS has no ACLs defined or if the PDS has ACLs defined but the PDS does not have an entry in the definition data file then the generated default ACL for this group will be created instead. Dremio recommends using an administrative group as this group, since an administrative group will have access to all PDSs anyway, but this will set the ACL explicitly rather than implicitly.  By setting an administrative group name explicitly we can avoid the situation where if no ACLs are set then all users with access to the parent container will automatically get edit privileges for the PDS, which is the default behavior if this option is not defined. This is often not a desirable outcome. |
| -u | Optional username that the tool will use to generate a default ACL for. If this option is specified, then in cases where the PDS has no ACLs defined or if the PDS has ACLs defined but the PDS does not have an entry in the definition data file then the generated default ACL for this user will be created instead. Dremio recommends using an administrative user as this user, since an administrative user will have access to all PDSs anyway, but this will set the ACL explicitly rather than implicitly.  By setting an administrative user name explicitly we can avoid the situation where if no ACLs are set then all users with access to the parent container will automatically get edit privileges for the PDS, which is the default behavior if this option is not defined. This is often not a desirable outcome. |
| -s | Optional flag that tells the tool to only update ACLs for source objects (i.e. the database folders) and to omit setting ACLs for PDSs within the source. |

The following command shows an example of how to update the schema\database and PDSs in a data source called MyDataSource that has a schema\database called MySchema whose ACL definition does not match that in the external ACL definition data which is stored in a file called /opt/dremiotools/conf/acl\_defs.json. In addition, for any PDSs without any ACLs defined or if there is no reference to the PDS in the definition data file, this command will update those PDSs with default ACLs for a user called adminuser.

dremio\_acl --config /opt/dremiotools/conf update acl -a /opt/dremiotools/conf/acl\_defs.json -u adminuser MyDataSource MySchema

The following command shows a similar example to above, but this time it will only update the ACLs for the schema\database called MySchema inside the data source called MyDataSource. No PDS ACLs inside this schema\database will be updated because of the -s flag:

dremio\_acl --config /opt/dremiotools/conf update acl -a /opt/dremiotools/conf/acl\_defs.json -u adminuser -s MyDataSource MySchema

The following command shows another example of setting ACLs at the schema\database level only. This time it will apply ACLs to all databases under the data source called MyDataSource. Because of the -u flag, for any databases without any ACLs defined or if there is no reference to the database folder in the definition data file, this command will update those databases with default ACLs for a user called adminuser.

dremio\_acl --config /opt/dremiotools/conf update acl -a /opt/dremiotools/conf/acl\_defs.json -u adminuser -s MyDataSource

### update acls-to-folder

The update acls-to-folder command is used to read the ACLs of all PDSs in a specified data source folder\schema and generate a superset of these ACLs. The superset of ACLs will then be used to set the ACLs of the data source folder itself. Optionally, the command can be used to automatically delete the ACLs from all PDSs in the specified folder once the folder has been updated.

Issuing the command dremio\_acl update acls-to-folder --help gives us the following information about usage of this command:

|  |
| --- |
| Usage: dremio\_acl update acls-to-folder [OPTIONS] BASE…  BASE: base directory of data source folder in Dremio for which to generate  superset of ACLs. Space separated. e.g. to start at a folder within a  source: MYSOURCE MYDB  Options:  -g, --group-on-acl-empty TEXT Optional group to set ACL for if a PDS ACL  list is empty  -u, --user-on-acl-empty TEXT Optional user to set ACL for if a PDS ACL  list is empty  -d, --delete-pds-acls Flag to delete PDS ACLs once the data  source folder is updated  --help Show this message and exit. |

These instructions indicate that there are three additional options applicable to the update acl command and those are the following:

|  |  |
| --- | --- |
| **Option** | **Description** |
| -g | Optional group name that the tool will use to generate a default ACL for. If this option is specified, then in cases where the derived superset has no ACLs defined then the generated default ACL for this group will be created instead. Dremio recommends using an administrative group as this group, since an administrative group will have access to all PDSs anyway, but this will set the ACL explicitly rather than implicitly.  By setting an administrative group name explicitly we can avoid the situation where if no ACLs are set then all users with access to the parent data source will automatically get edit privileges for the data source folder and PDSs, which is the default behavior if this option is not defined, which is often not a desirable outcome. |
| -u | Optional username that the tool will use to generate a default ACL for. If this option is specified, then in cases where the derived superset has no ACLs defined then the generated default ACL for this user will be created instead. Dremio recommends using an administrative user as this user, since an administrative user will have access to all PDSs anyway, but this will set the ACL explicitly rather than implicitly.  By setting an administrative user name explicitly we can avoid the situation where if no ACLs are set then all users with access to the parent data source will automatically get edit privileges for the data source folder and PDSs, which is the default behavior if this option is not defined, which is often not a desirable outcome. |
| -d | Optional flag that indicates we want to delete any explicit ACLs assigned to PDSs after the superset of ACLs has been propagated to the folder. |

The following command shows an example of how to update a data source folder\schema called MySchema in a data source called MyDataSource with a superset of ACLs that are derived from the set of all PDSs that have explicit ACLs defined inside the MySchema folder. In addition, if the resulting superset of ACLs is empty, this command will update the ACLs for the MySchema folder with default ACLs for a user called adminuser.

dremio\_acl --config /opt/dremiotools/conf update acls-to-folder -u adminuser MyDataSource MySchema

In order to also delete ACLs associated with PDSs inside the MySchema folder, include the -d flag in the command, as shown below:

dremio\_acl --config /opt/dremiotools/conf update acls-to-folder -u adminuser -d MyDataSource MySchema

### update space-acl

The update space-acl command is used to update specified spaces, folder and VDSs in Dremio to match the ACLs in an external definition file. If there is no definition data for a space, folder or data set in the path specified but in Dremio some ACLs have been set against the object then the tool will revoke the ACLs from the object and will optionally apply a default ACL instead.

Issuing the command dremio\_acl update space-acl --help gives us the following information about usage of this command:

|  |
| --- |
| Usage: cli.py update space-acl [OPTIONS] BASE...  BASE: base directory in the space hierarchy Dremio where to start applying  ACLs to. Space separated. e.g. to start at a folder within a space:  MYSPACE MYFOLDER  Options:  -a, --acl\_file FILENAME Path to a file containing the ACL  definitions [required]  -g, --group-on-acl-empty TEXT Optional group to set ACL for if object  ACLs are not present in the definition file  -u, --user-on-acl-empty TEXT Optional user to set ACL for if object ACLs  are not present in the definition file  --help Show this message and exit. |

These instructions indicate that there are three additional options applicable to the update space-acl command and those are the following:

|  |  |
| --- | --- |
| **Option** | **Description** |
| -a | Mandatory directory path and filename containing ACL definition data for all spaces, folders and VDSs that we want to set explicit ACLs against. The file is in JSON format. See the section called ACL Definition File for details on the required JSON structure. |
| -g | Optional group name that the tool will use to generate a default ACL for. If this option is specified, then in cases where the space\folder\VDS has no ACLs defined or if the space\folder\VDS has ACLs defined but it does not have an entry in the definition data file then the generated default ACL for this group will be created instead. Dremio recommends using an administrative group as this group, since an administrative group will have access to all spaces\folders\VDSs anyway, but this will set the ACL explicitly rather than implicitly.  By setting an administrative group name explicitly we can avoid the situation where if no ACLs are set then all users with access to the parent container will automatically get edit privileges for the child objects, which is the default behavior if this option is not defined. This is often not a desirable outcome. |
| -u | Optional username that the tool will use to generate a default ACL for. If this option is specified, then in cases where the space\folder\VDS has no ACLs defined or if the space\folder\VDS has ACLs defined but it does not have an entry in the definition data file then the generated default ACL for this user will be created instead. Dremio recommends using an administrative user as this user, since an administrative user will have access to all spaces\folders\VDSs anyway, but this will set the ACL explicitly rather than implicitly.  By setting an administrative user name explicitly we can avoid the situation where if no ACLs are set then all users with access to the parent container will automatically get edit privileges for the child objects, which is the default behavior if this option is not defined. This is often not a desirable outcome. |

The following command shows an example of how to update ACLs for a folder called MyFolder which is inside a space called MySpace, which will include any sub-folders and VDSs inside that folder based upon the definitions contained in the external ACL definition file called /opt/dremiotools/conf/acl\_defs.json. In addition, for any folders\VDSsiniside this path without any ACLs defined or if there is no reference to the folders\VDSs in the definition data file, this command will update those objects with default ACLs for a user called adminuser.

dremio\_acl --config /opt/dremiotools/conf update space-acl -a /opt/dremiotools/conf/acl\_defs.json -u adminuser MySpace MyFolder

### ACL Definition File

In the current release of dremio\_acl, the tool relies on an external ACL definition file that contains the source of truth for what ACLs must be set against the various database folders and/or PDSs in a data source, or indeed against the various spaces, folders and VDSs in Dremio. Below is an example of the mandatory structure of one of these files. The example contain a mix of showing how we can set permissions for individual users by using the user’s id as it is defined in Dremio (see blue text), and showing how we can set permissions for groups by using the group’s name as it is defined in LDAP (see red text) :

|  |
| --- |
| {  "entities": [  {  "entityType": "folder",  "entityPath": [  "Hive\_Flights",  "default"  ],  "accessControlList": {  "users": [  {  "id": "ldapuser1",  "permissions": ["READ","WRITE"]  }  ],  "groups": [  {  "id": "ldapgroup1",  "permissions": ["READ","WRITE"]  }  ]  }  },  {  "entityType": "physical\_dataset",  "entityPath": [  "Hive\_Flights",  "default",  "airline\_lookup"  ],  "accessControlList": {  "users": [  {  "id": "ldapuser1",  "permissions": ["READ","WRITE"]  },  {  "id": "ldapuser2",  "permissions": ["READ"]  }  ],  "groups": [  {  "id": "ldapgroup1",  "permissions": ["WRITE"]  }  ]  }  },  {  "entityType": "physical\_dataset",  "entityPath": [  "Hive\_Flights",  "default",  "flight\_dremio\_nodelta"  ],  "accessControlList": {  "users": [  {  "id": "ldapuser1",  "permissions": ["READ","WRITE"]  },  {  "id": "ldapuser2",  "permissions": ["READ"]  }  ],  "groups": [  {  "id": "ldapgroup1",  "permissions": ["WRITE"]  },  {  "id": "ldapgroup2",  "permissions": ["WRITE"]  }  ]  }    ]  } |

As shown above, the definition file makes use of an entityType element, this is so that we can process ACLs for Spaces, Folders, Virtual Data Sets, Physical Data Sets and database folders. Currently entityType values of physical\_dataset, virtual\_dataset, folder and space are used.

The entityPath element must contain an array of strings that represent the location in Dremio to where the Space, Folder, VDS, PDS or database folder resides.

The accessControlList element defines which users or groups we want to give which permissions for accessing the specified object. The names of the users and groups specified in this configuration must match the names of users and groups from the LDAP server that Dremio has been made aware of.