Requirements for Azure Data Lake Storage Gen2

Since v5.6.5/2

Contents

- Location Connection
- Hadoop Client
- Authentication
- Encryption
- Client Configuration Files for Hadoop
- Integrate
  - Customize Integrate

Azure DLS Gen2		
Capture	**Hub**	**Integrate**
✓	✗	✓

This section describes the requirements, access privileges, and other features of HVR when using Azure Data Lake Storage (DLS) Gen2 for replication. For information about compatibility and support for Azure DLS Gen2 with HVR platforms, see Platform Compatibility Matrix.

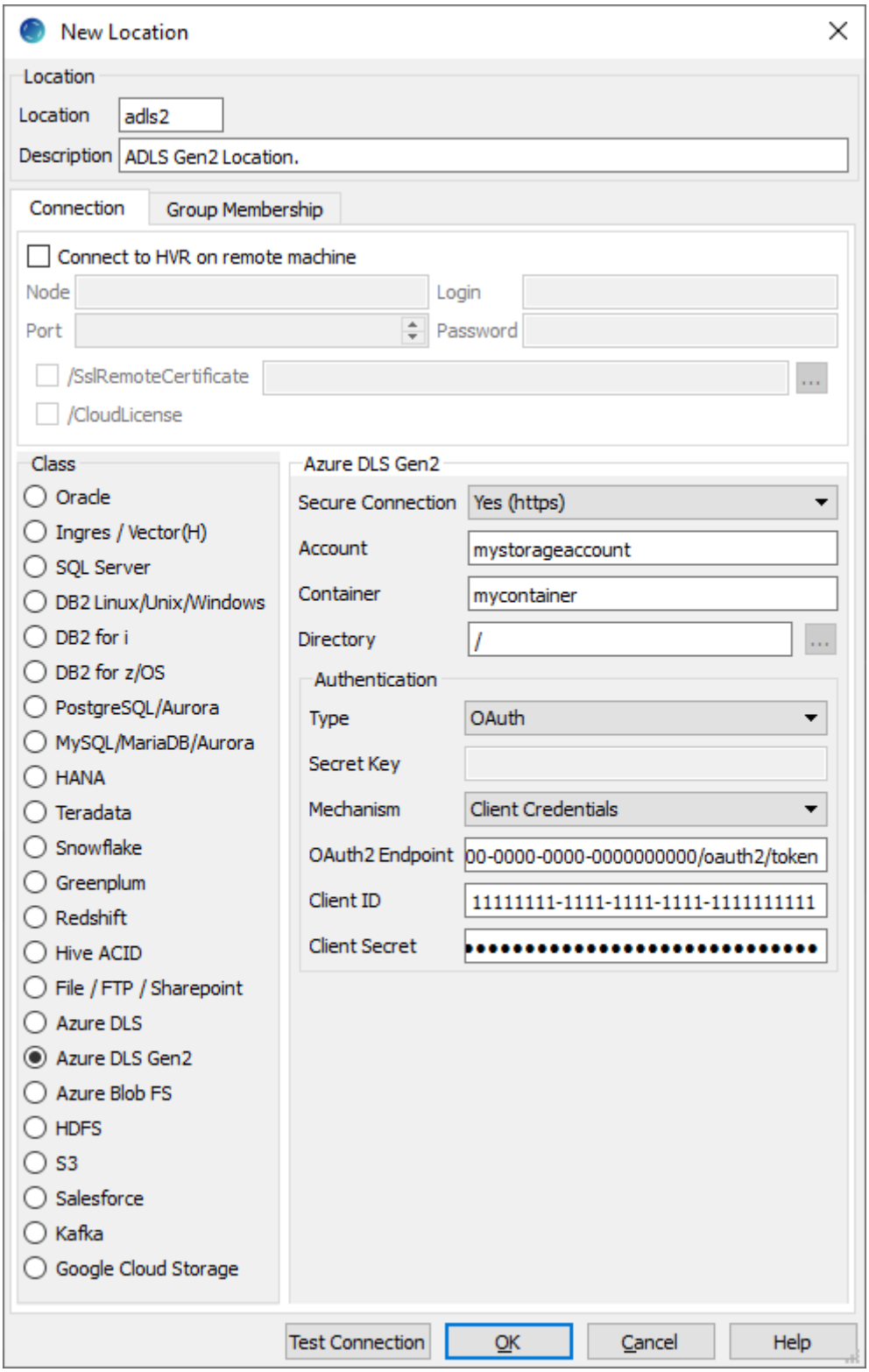
For the capabilities supported by HVR, see Capabilities.

For information about the supported data types and mapping of data types in source DBMS to the corresponding data types in target DBMS or file format, see Data Type Mapping.

For instructions to quickly set up replication using Azure DLS Gen2, see Quick Start for HVR - Azure DLS Gen2.

Location Connection

This section lists and describes the connection details required for creating Azure DLS Gen2 location in HVR.



Field	Description
Azure DLS Gen2	
Secure connection	The type of security to be used for connecting to Azure DLS Gen2. Available options: • **Yes (https)** (**default**): HVR will connect to Azure DLS Gen2 using HTTPS. • **No (http)**: HVR will connect to Azure DLS Gen2 using HTTP.

Account	The Azure DLS Gen2 storage account. **Example:** mystorageaccount
Container	The name of the container available within storage **Account**. **Example:** mycontainer
Directory	The directory path in **Container** to be used for replication. **Example:** /folder
Authentication	
Type	The type of authentication to be used for connecting to Azure DLS Gen2. Available options: • **Shared Key** (**default**): HVR will access Azure DLS Gen2 using **Shared Key** authentication. • **OAuth**: HVR will access Azure DLS Gen2 using **OAuth** authentication. For more information about these authentication types, see section Authentication.
Secret Key	The access key of the storage **Account**. This field is enabled only if authentication **Type** is **Shared Key**.
Mechanism	The authentication mode for connecting HVR to Azure DLS Gen2 server. This field is enabled only if authentication **Type** is **OAuth**. The available option is **Client Credentials**.
OAuth2 Endpoint	The URL used for obtaining bearer token with credential token. **Example:** https://login.microsoftonline.com/00000000-0000-0000-0000-0000000000/oauth2/token
Client ID	A client ID (or application ID) used to obtain Azure AD access token. **Example:** 00000000-0000-0000-0000-0000000000
Client Secret	A secret key used to validate the **Client ID**.

Hadoop Client

For Linux (x64) and Windows (x64), since HVR 5.7.0/8 and 5.7.5/4, it is not required to install and configure the Hadoop client. However, if you want to use the Hadoop client, set the environment variable **HVR_AZURE_USE_HADOOP=1** and follow the steps mentioned below.

It is mandatory to install and configure the Hadoop client for HVR versions prior to 5.7.0/8 or 5.7.5/4.

Hadoop Client Configuration Steps:

The Hadoop client must be installed on the machine from which HVR will access Azure DLS Gen2. HVR uses C API libhdfs to connect, read and write data to the Azure DLS Gen2 during capture, integrate (continuous), refresh (bulk) and compare (direct file compare).

Azure DLS Gen2 locations can only be accessed through HVR running on Linux or Windows. It is not required to run HVR installed on the Hadoop NameNode, although it is possible to do so. For more information about installing Hadoop client, refer to Apache Hadoop Releases.

On Linux, an extra warning is raised: "WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of HADOOP_PREFIX".

To fix this behavior, comment out the following line in the **$HADOOP_PREFIX/libexec/hadoop-condig.sh** file:

```
hadoop_deprecate_envvar HADOOP_PREFIX HADOOP_HOME
```

Hadoop Client Configuration

The following is required on the machine from which HVR connects to Azure DLS Gen2:

- Hadoop client libraries version 3.2.0 and higher. For downloading Hadoop, refer to Apache Hadoop Download page.
- Java Runtime Environment version 8 and higher. For downloading Java, refer to Java Download page.
- Set the environment variable **$JAVA_HOME** to the Java installation directory. Ensure that this is the directory that has a bin folder, e.g. if the Java bin directory is d:\java\bin, **$JAVA_HOME** should point to d:\java.

  If the environment variable **$HVR_JAVA_HOME** is configured, the value of this environment variable should point to the same path defined in **$JAVA_HOME**.
- Set the environment variable **$HADOOP_COMMON_HOME** or **$HADOOP_HOME** or **$HADOOP_PREFIX** to point to the Hadoop installation directory, or the **hadoop** command line client should be available in the path.
- One of the following configurations is recommended:
  - Set **$HADOOP_CLASSPATH=$HADOOP_HOME/share/hadoop/tools/lib/***
  - Create a symbolic link for **$HADOOP_HOME/share/hadoop/tools/lib** in **$HADOOP_HOME/share/hadoop/common** or any other directory present in the classpath.
- On Windows, **winutils.exe** along with **hadoop.dll** is required. These files can be downloaded from the GitHub and should be saved to **$HADOOP_HOME/bin** directory. This is required since the binary distribution of Hadoop lacks this executable.

Verifying Hadoop Client Installation

To verify the Hadoop client installation:

1. The **$HADOOP_HOME/bin** directory in the Hadoop installation location should contain the Hadoop executables in it.
2. Execute the following commands to verify the Hadoop client installation:

   ```
   $JAVA_HOME/bin/java -version
   $HADOOP_HOME/bin/hadoop version
   $HADOOP_HOME/bin/hadoop classpath
   ```

3. If the Hadoop client installation is successfully verified, execute the following command to verify the connectivity between HVR and Azure DLS Gen2:

   ```
   $HADOOP_HOME/bin/hadoop fs -ls abfs://<container>@<account>.dfs.core.windows.net
   ```

In case of any identification errors, certain properties need to be defined in the **core-site.xml** file available in the Hadoop configuration folder (for e.g., **<path>/hadoop-3.2.0/etc/hadoop**). For more information, refer to section Configuring ABFS in the Hadoop Azure Support: ABFS - Azure Data Lake Storage Gen2 documentation.

```
<property>
  <name>fs.azure.account.auth.type.storageaccountname.dfs.core.
windows.net</name>
  <value>SharedKey</value>
  <description>Use Shared Key authentication</description>
</property>

<property>
  <name>fs.azure.account.key.storageaccountname.dfs.core.windows.net<
/name>
  <value>JDlkIHxvySByZWFsbHkgdGabcdfeSSB3LDJgZ34pbm
/skdG8gcGD0IGEga2V5IGluIGhlcmSA</value>
  <description>The secret password.</description>
</property>
```

```
<property>
  <name>fs.azure.account.auth.type</name>
  <value>OAuth</value>
  <description>Use OAuth authentication</description>
</property>


<property>
  <name>fs.azure.account.oauth.provider.type</name>
  <value>org.apache.hadoop.fs.azurebfs.oauth2.
ClientCredsTokenProvider</value>
  <description>Use client credentials</description>
</property>


<property>
  <name>fs.azure.account.oauth2.client.endpoint</name>
  <value></value>
  <description>URL of OAuth endpoint</description>
</property>


<property>
  <name>fs.azure.account.oauth2.client.id</name>
  <value></value>
  <description>Client ID</description>
</property>


<property>
  <name>fs.azure.account.oauth2.client.secret</name>
  <value></value>
  <description>Secret</description>
</property>
```

Verifying Hadoop Client Compatibility with Azure DLS Gen2

To verify the compatibility of the Hadoop client with Azure DLS Gen2, check if the following JAR files are available in the Hadoop client installation directory (**$HADOOP_HOME/share/hadoop/tools/lib**):

```
wildfly-openssl-<version>.jar
hadoop-azure-<version>.jar
```

Authentication

HVR supports the following two authentication modes for connecting to Azure DLS Gen2:

- **Shared Key**
  When this option is selected, *hvruser* gains full access to all operations on all resources, including setting owner and changing Access Control List (ACL). The connection parameter required in this authentication mode is Secret Key - a shared access key that Azure generates for the storage account. For more information on how to manage access keys for Shared Key authorization, refer to Manage storage account access keys. Note that with this authentication mode, no identity is associated with a user and permission-based authorization cannot be implemented.
- **OAuth**
  This option is used to connect to Azure DLS Gen2 storage account directly with OAuth 2.0 using the service principal. The connection parameters required for this authentication mode are **OAuth2 Endpoint**, **Client ID**, and **Client Secret**. For more information, refer to Azure Data Lake Storage Gen2 documentation.

Encryption

HVR does not support client side encryption (customer managed keys) for Azure DLS Gen2. For more information about the encryption of data in Azure DLS Gen2 refer to Data Lake Storage Documentation.

Client Configuration Files for Hadoop

Client configuration files are not required for HVR to perform replication, however, they can be useful for debugging. Client configuration files contain settings for different services like HDFS, and others. If the HVR integrate machine is not part of the cluster, it is recommended to download the configuration files for the cluster so that the Hadoop client knows how to connect to HDFS.

The client configuration files for Cloudera Manager or Ambari for Hortonworks can be downloaded from the respective cluster manager's web interface. For more information about downloading the client configuration files, search for "Client Configuration Files" in the respective documentation for Cloudera and Hortonworks.

Integrate

HVR allows you to perform **HVR Refresh** or **Integrate** changes into an Azure DLS Gen2 location. This section describes the configuration requirements for integrating changes (using **HVR Refresh** or **Integrate**) into the Azure DLS Gen2 location.

Customize Integrate

Defining action **Integrate** is sufficient for integrating changes into an Azure DLS Gen2 location. However, the default file format written into a target file location is HVR's own XML format and the changes captured from multiple tables are integrated as files into one directory. The integrated files are named using the integrate timestamp.

You may define other actions for customizing the default behavior of integration mentioned above. Following are few examples that can be used for customizing integration into the Azure DLS Gen2 location:

Group	Table	Action	Annotation
Azure DLS Gen2	*	**FileFormat**	This action may be defined to: • specify the format (**Xml**, **Csv**, **Avro**, **Json**, or **Parquet**) of the files integrated into the target location. • escape any delimiters (e.g. comma) present in a column using the parameter **/QuoteCharacter**. • escape the quote character (**/QuoteCharacter**) defined, using the parameter **/EscapeCharacter**.
Azure DLS Gen2	*	**Integrate /RenameExpression**	To segregate and name the files integrated into the target location. For example, if **/RenameExpression={hvr_tbl_name}/{hvr_integ_tstamp}.csv** is defined, then for each table in the source, a separate folder (with the same name as the table name) is created in the target location, and the files replicated for each table are saved into these folders. This also enforces unique name for the files by naming them with a timestamp of the moment when the file was integrated into the target location.

Azure DLS Gen2	*	**ColumnProperties**	This action defines properties for a column being replicated. This action may be defined to:
			• integrate the delete operation. By default, for file-based target locations, HVR does not replicate the **delete** operation performed at the source location. So to integrate the delete operation, an extra column for timekey (**/TimeKey**) needs to be added in the target location. For this, action **ColumnProperties** may be defined with the following parameters:
			  • **/Name**: This parameter defines the name for the extra column in the target location.
			  • **/Extra**: This parameter defines that this is an extra column in the target location (a column which is not present in the source location).
			  • **/IntegrateExpression**: This parameter defines the expression to be used for generating the **TimeKey** value. For example, **{hvr_integ_seq}** can be used here. This is a 36 byte string value (hex characters) which is unique and continuously increasing for a specific source location.
			  • **/TimeKey**: This parameter defines that this is a **TimeKey** column.
			  • **/Datatype=varchar**: This parameter defines the data type for the extra column.
			  • **/Length=36**: This parameter defines the data type length for the extra column.
			• add the source operation type (using **hvr_op**) information in the target location. This action definition is required for performing **HVR Compare** if **ColumnProperties /TimeKey** column is defined on a target file location. For this, action **ColumnProperties** may be defined with the following parameters:
			  • **/Name**: This parameter defines the name for the extra column in the target location.
			  • **/Extra**: This parameter defines that this is an extra column in the target location (a column which is not present in the source location).
			  • **/IntegrateExpression={hvr_op}**: This parameter defines the expression to be used for generating the information about source operation type.
			  • **/Datatype=integer**: This parameter defines the data type for this extra column.