HDFS Binary Writer Introduction

The HDFS binary file writer is a workflow stage that “intercepts” the data stream associated with an Aspire job as it passes through the workflow and writes it to an HDFS file system. The administrator configures a “base directory” and all content is written below this directory on the HDFS file system. The written content is further subdivided by “content source” (which can be extracted from the Aspire job, or hard coded) and then by in to deterministic directories based on an MD5 hash of the document id (taken from the Aspire job). The structure of the directories is configurable, allowing the administrator to control the distribution of files across directories. Information about the file written is added to the Aspire job so that the data may be easily located.

The writer may also be configured with HDFS resource files and have security enabled (Kerberos) if required.

> ⚠ The HDFS binary file writer needs access to the data stream from the Aspire job BEFORE it has been consumed by another stage (such as text extraction). Therefore, the connector used should have text extraction disabled. If you wish to perform text extraction, you can add a separate workflow component after the binary writer to perform this.

On this page

Related pages

- Prerequisites
- How to Configure
- FAQ & Troubleshooting

Features

Some of the features of the HDFS Binary Writer include:

- Captures binary content to HDFS via a tee-stream allowing the data to be used in subsequent stages (such as text extraction)
- Separates captured content by content source
- Provides a dual level directory hierarchy that distributes captured files across multiple directories while placing the files in a deterministic location
- Compatible with the HDFS Archive (HAR) Compactor
- Supports Kerberos security
  - via a key tab file
- Allows addition of Hadoop or HDFS resource files to simplify configuration

Directory Structure and File Names

As mentioned, the binary writer writes data to files under a base directory on HDFS. Under that, a directory is created for the “content source”. This “content source” name can be extracted from the Aspire job (from a user defined location in the Aspire document) or can be set to a static value (allowing data from multiple Aspire connectors to be written to the same “content source” directory.

The id of the document is then used to determine the rest of the path to and the name of the stored file. An MD5 hash of the document id is taken. The file name is constructed from that hash (to ensure uniqueness) and a “name” taken from the document id. This “name” is the file name (if the id was a file path) or the “page name” if the id was a url. If required, this “name” may be shortened by removing a section from the middle if the path to the HDFS file would exceed the HDFS limit of 255 characters. This approach ensures that the filename is unique, but gives a “recognisable” part of the file name to aid the user to locate the file.

A configurable number of the lowest significant bits of the hash are used to generate two levels of directory in which the file will be written. This allows the distribution of the files across the directories. Two configuration parameters are available – hash bits (default 16 bits or 4 characters) and directory bits (default 8 bits or 2 characters). “hash bits” defines the total number of (lowest significant) bits that will be used for the second level of directory name. Directory bits defines the number of (most significant) bits taken from the lower directory name as the higher directory name.

Example

Hash bits:	16
Directory bits:	8
Document ID:	file:///c:/testdata/1/0.txt
HFDS file name:	A3/A3BC/AF0327674491B77556554227D915A3BC-0.txt
Hash bits:	8

Directory bits:	4
Document ID:	file:///c:/testdata/1/0.txt
HFDS file name:	B/BC/AF0327674491B77556554227D915A3BC-0.txt

File distribution across directories

The table below shows the values of the parameters and the distribution of files across the directories.

hash bits	dir bits	total directories	top level name length (chars)	lower level name length (chars)	top level directories	lower level directories	Average files per directory													
							1m	5m	10m	25m	50m	100m	250m	500m	1b	5b	10b	25b	50b	100b
8	4	256	1	2	16	16	3,906.25	19,531.25	39,062.50	97,656.25	195,312.50	390,625.00	976,562.50	1,953,125.00	3,906,250.00	19,531,250.00	39,062,500.00	97,656,250.00	195,312,500.00	390,625,000.00
12	8	4,096	1	3	16	256	244.14	1,220.70	2,441.41	6,103.52	12,207.03	24,414.06	61,035.16	122,070.31	244,140.63	1,220,703.13	2,441,406.25	6,103,515.63	12,207,031.25	24,414,062.50
12	4	4,096	2	3	256	16	244.14	1,220.70	2,441.41	6,103.52	12,207.03	24,414.06	61,035.16	122,070.31	244,140.63	1,220,703.13	2,441,406.25	6,103,515.63	12,207,031.25	24,414,062.50
16	12	65,536	1	4	16	4,096	15.26	76.29	152.59	381.47	762.94	1,525.88	3,814.70	7,629.39	15,258.79	76,293.95	152,587.89	381,469.73	762,939.45	1,525,878.91
16	8	65,536	2	4	256	256	15.26	76.29	152.59	381.47	762.94	1,525.88	3,814.70	7,629.39	15,258.79	76,293.95	152,587.89	381,469.73	762,939.45	1,525,878.91
16	4	65,536	3	4	4,096	16	15.26	76.29	152.59	381.47	762.94	1,525.88	3,814.70	7,629.39	15,258.79	76,293.95	152,587.89	381,469.73	762,939.45	1,525,878.91
20	16	1,048,576	1	5	16	65,536	0.95	4.77	9.54	23.84	47.68	95.37	238.42	476.84	953.67	4,768.37	9,536.74	23,841.86	47,683.72	95,367.43
20	12	1,048,576	2	5	256	4,096	0.95	4.77	9.54	23.84	47.68	95.37	238.42	476.84	953.67	4,768.37	9,536.74	23,841.86	47,683.72	95,367.43
20	**8**	**1,048,576**	**3**	**5**	**4,096**	**256**	**0.95**	**4.77**	**9.54**	**23.84**	**47.68**	**95.37**	**238.42**	**476.84**	**953.67**	**4,768.37**	**9,536.74**	**23,841.86**	**47,683.72**	**95,367.43**
20	4	1,048,576	4	5	65,536	16	0.95	4.77	9.54	23.84	47.68	95.37	238.42	476.84	953.67	4,768.37	9,536.74	23,841.86	47,683.72	95,367.43
24	20	16,777,216	1	6	16	1,048,576	0.06	0.30	0.60	1.49	2.98	5.96	14.90	29.80	59.60	298.02	596.05	1,490.12	2,980.23	5,960.46
24	16	16,777,216	2	6	256	65,536	0.06	0.30	0.60	1.49	2.98	5.96	14.90	29.80	59.60	298.02	596.05	1,490.12	2,980.23	5,960.46
24	12	16,777,216	3	6	4,096	4,096	0.06	0.30	0.60	1.49	2.98	5.96	14.90	29.80	59.60	298.02	596.05	1,490.12	2,980.23	5,960.46
24	8	16,777,216	4	6	65,536	256	0.06	0.30	0.60	1.49	2.98	5.96	14.90	29.80	59.60	298.02	596.05	1,490.12	2,980.23	5,960.46
24	4	16,777,216	5	6	1,048,576	16	0.06	0.30	0.60	1.49	2.98	5.96	14.90	29.80	59.60	298.02	596.05	1,490.12	2,980.23	5,960.46

Limitations

HDFS Binary Writer has been tested again Cloudera v5.10.1

Anything we should add?

Please let us know.