Cloudera Runtime 7.2.18

Cloudera Search solrctl Reference

Date published: 2020-06-01
Date modified:



https://docs.cloudera.com/

Legal Notice

© Cloudera Inc. 2026. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

solrctl Reference.....4

Using solrctl with an HTTP proxy..... 12

solrctl Reference

The solrctl utility is a wrapper shell script included with Cloudera Search for managing collections, instance directories, configs, and more.

Note:

- Ensure that the host on which you are running the solrctl utility has either a Gateway or Solr Server role assigned.
- If the SunJCE cryptographic provider is removed to run the solrctl utility, you must set the JDK_JAVA _OPTIONS environment variable. For example,

```
export FIPS_JAVA_ARGS
=
"-Dcom.safelogic.cryptocomply.fips.approved_only=true  --add-modules
=com.safelogic.cryptocomply.fips.core  --add-modules=bctls  --add-ex
ports=java.base/sun.security.provider=com.safelogic.cryptocomply.fip
s.core  --add-exports=java.base/sun.security.provider=bctls  --modul
e-path=/cdep/extra_jars  -Djdk.tls.trustNameService=true  -Dsun.secu
rity.krb5.disableReferrals=true  -Djdk.tls.ephemeralDHKeySize=2048
-Dcom.sun.management.jmxremote.ssl.enabled.protocols=TLSv1.2,TLSv1.3
 --add-opens=java.base/java.lang=ALL-UNNAMED --add-opens=java.base/j
ava.util=ALL-UNNAMED --add-opens=java.base/java.util.zip=ALL-UNNAMED
 --add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.base/jav
a.lang.reflect=ALL-UNNAMED --add-opens=java.base/java.net=ALL-UNNAME
D --add-exports=java.base/sun.net.www.protocol.jar=ALL-UNNAMED --add
-opens=java.base/sun.net.www.protocol.jar=ALL-UNNAMED --add-opens=ja
va.base/sun.net.www.protocol.file=ALL-UNNAMED --add-exports=java.bas
e/sun.net.www=ALL-UNNAMED --add-opens=java.base/sun.net.www=ALL-UNNA
MED"


export JDK_JAVA_OPTIONS
=
"${JDK_JAVA_OPTIONS} ${FIPS_JAVA_ARGS}"
```

In general, if an operation succeeds, solrctl exits silently with a success exit code. If an error occurs, solrctl prints a diagnostics message combined with a failure exit code. solrctl supports specifying a log4j.properties file by setting the LOG4J_PROPS environment variable. By default, the LOG4J_PROPS setting specifies the log4j.properties in the Solr configuration directory (for example, /etc/solr/conf/log4j.properties). Many solrctl commands redirect stderr to /dev/null, so Cloudera recommends that your log4j properties file specify a location other than stderr for log output.

You can run solrctl on any host that is configured as part of the SolrCloud deployment (the Solr service in Cloudera Manager environments) . To run any solrctl command on a host outside of SolrCloud deployment, ensure that SolrCloud hosts are reachable and provide --zk and --solr command line options.

If you are using solrctl to manage your deployment in an environment that requires Kerberos authentication, you must have a valid Kerberos ticket, which you can get using kinit.

For collection configuration, users have the option of interacting directly with ZooKeeper using the instancedir option or using the Solr ConfigSets API using the config option. For more information, see Managing Configuration Using Configs or Instance Directories.

Command syntax

The general solrctl command syntax is:

solrctl [options] command [command-arg] [command [command-arg]] ...

Each element and its possible values are described in the following sections.

solrctl options

If used, the following options must precede commands:

Table 1: solrctl options

Option	Description
--solr <*SOLR_URI*>	Directs solrctl to a SolrCloud web API available at the specified URI. This option is required for hosts running outside of SolrCloud. A sample URI might be: http://search01.example.com:8983/solr.
--zk <*ZK_ENSEMBLE*>	Directs solrctl to a particular ZooKeeper quorum. This option is required for hosts running outside of SolrCloud. For example: zk01.example.com:2181,zk02.example.com:2181,zk03.example.com:2181/solr. Output from solrctl commands that use the --zk option is sent to /dev/null, so no results are displayed.
--jaas */PATH/TO/JAAS.CONF*	Used to identify a JAAS configuration that specifies the principal with permissions to modify Solr metadata. The principal is typically solr@EXAMPLE.COM. In Kerberos-enabled environments where ZooKeeper ACLs protect Solr metadata, you must use this parameter when modifying metadata.
--help	Prints help.
--quiet	Suppresses most solrctl messages.
--debug	Prints errors to stdout.
--debug-dump	Prints the current state of ZooKeeper
--trace	Prints the executed commands to stdout.

Commands and arguments

The solrctl commands init, instancedir, config, collection and cluster affect the entire SolrCloud deployment and only need to be run once per required operation.

The solrctl core command affects a single SolrCloud host.

Table 2: solrctl commmands and arguments

Command	Argument	Description
init		Initializes a SolrCloud deployment. Must be run before starting solr-server daemons for the first time. The command has a built-in security check that prevents it from running on a deployment that has already been initialized.
	[--force]	Allows you to re-initialize an already initialized SolrCloud deployment. Use this command cautiously because it erases all SolrCloud deployment state information from ZooKeeper, including all configuration files. It does not delete collections.

Command	Argument	Description
instancedir		Manipulates instance directories.  **Note:** solrctl instancedir commands talk directly to ZooKeeper. Starting from CDH version 6.0, the ZooKeeper node that stores the instance configuration files (e.g.: /solr/configs) is only writable by the user 'solr'. This means that in a Kerberized environment you will have to create a Java Authentication and Authorization Service (JAAS) configuration file and a keytab for the solr user, and specifying this jaas.conf file when using solrctl. For example: solrctl --jaas jaas.conf instancedir --create myInstanceDir /tmp/myInstanceDir. Cloudera recommends using the solrctl config --upload command instead, as it does not require a keytab of the solr user (a simple kinit is enough) and additionally it can utilize Ranger to control who is authorized to modify instance configurations.
	--generate <*PATH*> [-schemaless] [-localfs]	--generate <*PATH*>: Generates an instance directory template on the local filesystem at <*PATH*>. The configuration files are located in the conf subdirectory under <*PATH*>. • If used with the -schemaless option, it generates a schemaless instance directory template. For more information on schemaless support, see Schemaless Mode Overview and Best Practices. • If used with the -localfs option, the default HdfsDirectoryFactory is overwritten with NRTCachingDirectoryFactory in the generated solrconfig.xml. The index of a collection using this configuration is written to the local file system, not to HDFS.
	--create <*NAME*> <*PATH*>	Uploads a copy of the instance directory from <*PATH*> on the local filesystem to ZooKeeper. If an instance directory with the specified <*NAME*> already exists, this command fails. Use --update to modify existing instance directories.
	--update <*NAME*> <*PATH*>	Overwrites an existing instance directory in ZooKeeper using the specified files on the local filesystem. This command is analogous to first running --delete <*NAME*> followed by --create <*NAME*> <*PATH*>.
	--get <*NAME*> <*PATH*>	Downloads the specified instance directory from ZooKeeper to the specified path on the local filesystem. You can then edit the configuration and then re-upload it using --update.
	--delete <*NAME*>	Deletes the specified instance directory from ZooKeeper.

Command	Argument	Description
	--list	Lists existing instance directories, including configs created by the solrctl config command.
config		Manipulates configs.
	--create name <*BASECONFIG*> [-p <*NAME*>=<*VALUE*>	Creates a new config based on an existing config. The config is created with the specified *NAME*, using <*BASECONFIG*> as the template. For more information about config templates, see Config Templates. The -p name=value option overrides a <*BASECONFIG*> setting. The only config property that you can override is immutable, so the possible options are -p immutable=true and -p immutable=false. If you are copying an immutable config, such as a template, use -p immutable=false to make sure that you can edit the new config.
	--upload *NAME* *PATH*	Uploads a new configset in zip file format. You cannot use this option to update an existing config. The script will ask you to delete the existing version before allowing you to upload the new one. The *PATH* argument of this command needs to point to the local directory containing the instance configuration (meaning it has a conf subdirectory and the config files like conf/solrconfig.xml). This can also be an instance configuration directory generated using solrctl instancedir --generate *NAME* or downloaded using solrctl instancedir --get *NAME* *PATH*. The underlying Solr API requires a .zip archive to be created, this is automatically performed by the command.  **Note:** This function needs the zip binary to be installed and executable by the user running the solrctl config --upload command.
	--delete name	Deletes the specified config. You cannot delete an immutable config without accessing ZooKeeper directly as the solr super user.
collection		Manipulates collections.
	--create <*NAME*> -s <*NUMSHARDS*> [-a] [-c <*CONFIGNAME*>] [-r <*REPLICATIONFACTOR*>] [-m <*MAXSHARDSPERHOST*>] [-n <*HOSTLIST*>]]	Creates a new collection with <*NUMSHARDS*> shards. The -a option enables automatic addition of replicas (autoAddReplicas=true) if machines hosting existing shards become unavailable. The collection uses the specified <*CONFIGNAME*> for its configuration set, and the specified <*REPLICATIONFACTOR*> controls the number of replicas for the collection. Keep in mind that this replication factor is on top of the HDFS replication factor. The maximum shards per host is determined by <*MAXSHARDSPERHOST*>, and you can specify specific hosts for the collection in the <*HOSTLIST*>. The only required parameters are <*NAME*> and -s <*NUMSHARDS*>. If -c <*CONFIGNAME*> is not provided, it is assumed to be the same as the name of the collection.

Command	Argument	Description
	--delete <*NAME*>	Deletes a collection.
	--reload <*NAME*>	Reloads a collection.
	--stat <*NAME*>	Outputs SolrCloud specific run-time information for a collection.
	--deletedocs <*NAME*>	Purges all indexed documents from a collection.
	--list	Lists all collections.
	--create-snapshot <*SNAPSHOTNAME*> -c <*COLLECTIONNAME*>	Creates a named snapshot for the specified collection.
	--delete-snapshot <*SNAPSHOTNAME*> -c <*COLLECTIONNAME*>	Deletes the specified snapshot for the specified collection.
	--describe-snapshot <*SNAPSHOTNAME*> -c <*COLLECTIONNAME*>	Provides detailed information about a snapshot for the specified collection.
	--list-snapshots <*COLLECTIONNAME*>	Lists all snapshots for the specified collection.
	--prepare-snapshot-export <*SNAPSHOTNAME*> -c <*COLLECTIONNAME*> -d <*DESTDIR*>	Prepares the snapshot for export to a remote cluster. If you are exporting the snapshot to the local cluster, you do not need to run this command. This command generates collection metadata as well as information about the Lucene index files corresponding to the snapshot. The destination HDFS directory path (specified by the -d option) must exist on the local cluster before you run this command. Make sure that the Solr superuser (solr by default) has permission to write to this directory. If you are running the snapshot export command on a remote cluster, specify the HDFS protocol (such as WebHDFS or HFTP) to be used for accessing the Lucene index files corresponding to the snapshot on the source cluster. This configuration is driven by the -p option which expects a fully qualified URI for the root filesystem on the source cluster, for example webhdfs://namenode.example.com:20101/.  **Note:** Creating backups and performing restore operations using the solr ctl CLI tool exclusively works on HDFS. If you want to create a backup to, or perform a restore from a different FS, consult the Apache Solr documentation.

Command	Argument	Description
	--export-snapshot <*SNAPSHOTNAME*> -c <*COLLECTIONNAME*>|-s <sourceDir> -d <*DESTDIR*>	Creates a backup copy of the Solr collection metadata as well as the associated Lucene index files at the specified location. If exporting to a local cluster, use the -c <*COLLECTIONNAME*> argument. If exporting to a remote cluster, use the -s <*SOURCEDIR*> argument. The -d configuration option specifies the directory path where this backup copy is be created. This directory must exist before exporting the snapshot, and the Solr superuser (solr, by default) must be able to write to it.  **Note:** Creating backups and performing restore operations using the solr ctl CLI tool exclusively works on HDFS. If you want to create a backup to, or perform a restore from a different FS, consult the Apache Solr documentation.

Command	Argument	Description
	--restore <*RESTORECOLLECTIONNAME*> -l <*BACKUPLOCATION*> -b <*SNAPSHOTNAME*> -i <*REQUESTID*> [-a] [-c <*CONFIGNAME*>] [-r <*REPLICATIONFACTOR*>] [-m <*MAXSHARDSPERHOST*>] [-n <*SELECTEDNODES*>]	Restores the state of an earlier created backup as a new Solr collection. Run this command on the cluster on which you want to restore the backup. The -l configuration option specifies the local HDFS directory where the backup is stored. If the backup is stored on a remote cluster, you must copy it to the local cluster before restoring it. The Solr superuser (solr by default) must have permission to read from this directory. The -b configuration option specifies the name of the backup to be restored. Because the restore operation can take a long time to complete depending on the size of the exported snapshot, it is run asynchronously. The -i configuration parameter specifies a unique identifier for tracking the operation. The optional -a configuration option enables the autoAddReplicas feature for the new Solr collection. The optional -c configuration option specifies the configName for the new Solr collection. If this option is not specified, the configNa me of the original collection at the time of backup is used. If the specified configName does not exist, the restore operation creates a new configuration from the backup. The optional -r configuration option specifies the replication factor for the new Solr collection. If this option is not specified, the replication factor of the original collection at the time of backup is used. The optional -m configuration option specifies the maximum number of replicas (maxShard sPerNode) to create on each Solr Server. If this option is not specified, the maxShardsPerNode configuration of the original collection at the time of backup is used. The optional -n configuration option specifies the nodes to spread the restored collection across. If this option is not specified, the operation will create shard-replica spread across all live Solr nodes. **Note:** Creating backups and performing restore operations using the solr ctl CLI tool exclusively works on HDFS. If you want to create a backup to, or perform a restore from a different FS, consult the Apache Solr documentation.

Command	Argument	Description
	--readonly *NAME*	Puts the collection in read-only mode, in which any index update requests are rejected. Other collection-level actions (e.g., adding / removing / moving replicas) are still available in this mode. The transition from the (default) read-write to read-only mode consists of the following steps: • the readOnly flag is changed in collection state, • any new update requests are rejected with 403 FORBIDDEN error code (ongoing long-running requests are aborted, too), • a forced commit is performed to flush and commit any in-flight updates. **Note:** This may potentially take a long time if there are still major segment merges running in the background. • a collection RELOAD action is executed. Issuing the solrctl collection --readwrite *NAME* command enables the processing of updates and reloads the collection.
	--readwrite *NAME*	puts the collection in the default read-write mode, in which any index update requests are allowed.
	--request-status <*REQUESTID*>	Displays the status of the specified operation. The status can be one of the following: • running: The restore operation is running. • completed: The restore operation is complete. • failed: The restore operation failed. • notfound: The specified requestID is not found.
core		Manipulates cores.
	--create <*NAME*> [-p <*NAME*>=<*VALUE*>]	Creates a new core. The core is configured using <*NAME*>=<*VALUE*> pairs. For more information about configuration options, see Solr documentation
	--reload <*NAME*>	Reloads a core.
	--unload <*NAME*>	Unloads a core.
	--status <*NAME*>	Prints the status of a core.
cluster		Manages cluster configuration.
	--get-solrxml <*FILE*>	Downloads the cluster configuration file solr.xml from ZooKeeper to the local system.
	--put-solrxml <*FILE*>	Uploads the specified file to ZooKeeper as the cluster configuration file solr.xml.
	--set-property <*NAME*> <*VALUE*>	Sets property names and values. For example, to configure a cluster to use TLS/SSL: `solrctl cluster --set-property urlScheme https`

Command	Argument	Description
	--remove-property <*NAME*>	Removes the specified property.
	--get-clusterstate <*FILE*>	Downloads the clusterstate.json file from ZooKeeper to the local system.

Using solrctl with an HTTP proxy

About this task

Using solrctl to manage a deployment in an environment that uses an HTTP proxy fails because solrctl uses curl, which attempts to use the proxy. You can disable the proxy so solrctl succeeds:

Procedure

- Modify the settings for the current shell by exporting the NO_PROXY environment variable. For example:

```
export NO_PROXY='*'
```

- Modify the settings for single commands by prefacing solrctl with NO_PROXY='*'. For example:

```
NO_PROXY='*' solrctl collection --create <COLLECTIONNAME> -s 3
```