Whitepaper

Content archive tool

Contents

Overview

The HPE Cloud Service Automation (CSA) Content Archive Tool is a command line interface for exporting artifacts from the CSA system to an archive file and importing artifacts from an archive file to the CSA system. These artifacts provide the basis for cloud automation.

The export operation provides the ability to preserve the selected artifacts so they can be used to replicate the services on another system or to restore the artifacts. The import and update operations provide the ability to install or replace artifacts on the system. The import operation adds artifacts; the update operation overwrites matching artifacts.

The exported archive files are preserved in an industry-standard zip archive file format. An archive contains XML and JSON documents that represent the primary artifact being exported, dependent artifacts, and any customized images or dynamic property JSP files necessary to represent the entire artifact. For example, a service offering archive contains the XML documents representing the service offering, the service design associated with the service offering, resource offerings associate with the service design, and a Manifest XML document. The service offering archive will also contain any dynamic property JSP files and images used for customization of all these artifacts.

Important: Archives created with previous versions of CSA can be imported to the most current version. However, if CSA has been configured to verify the authenticity of import service design, service offering, and catalog content archives (which is done by setting the property `csa.security.enable`), then service design, service offering, and catalog content archives that were successfully imported into CSA 4.50 or earlier will no longer import properly until they are digitally signed. See the *CSA Configuration Guide* for information on how to digitally sign a content archive.

Supported Operations

Export

Use the export operation to export supported artifacts as archives. An archive name includes the primary artifact type, its display name, and its ID. A separate archive is created for each primary/topmost artifact in an exported hierarchy. After an export is complete, a summary of the operation is displayed.

Validate

Use the validate operation to preview actions that will be taken by the import operation. This operation displays a summary of artifacts that can be imported from the specified archive and artifacts that already exist on the system.

Import and Update

The default import behavior adds artifacts, whereas the update behavior not only adds artifacts but also replaces matching artifacts. After an import/update is complete, a summary of the operation is displayed.

Default Import Process

By default, an artifact is not imported if the source and existing target artifacts are considered to be *functionally equivalent*. Artifacts are considered functionally equivalent if the following is true:

Values evaluated to determine functional equivalence

Artifact type	Has the same
Component Palette	Internal name
Resource Offering	Resource category, provider type, properties, and actions
Service Design	Internal name
Service Offering	Internal name
Resource Environment	Internal name
Catalog	Internal name and organization ID

Important: If the import operation fails, changes are rolled back and the target system is left unchanged.

Important: Artifact import rules differ for catalogs and component palettes. If a catalog already exists on the target system, the catalog is updated with any added or removed supported artifacts. Import of a component palette by default is an update operation, which is described below. The import process imports component palettes, including their associated component types, templates, and component type constraints.

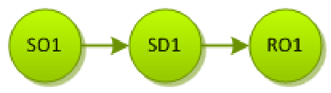
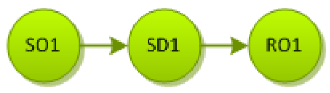
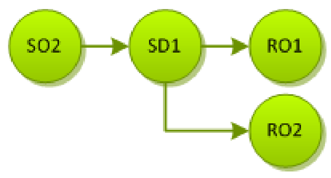
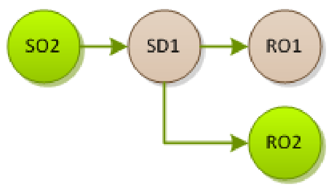
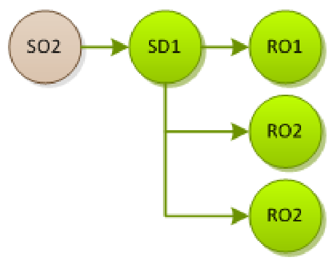
Imported artifacts are associated with existing artifacts in the system. If the existing artifact is the primary artifact in a hierarchy, the import operation exits without importing any of the archive content. For example, when importing a service offering archive, if the service design in the archive already exists in the system and the service offering is not in the system, the service offering is imported and is associated with the existing service design. However, if the service offering (in this case the primary artifact) already exists in the system, the import operation exits without importing any artifacts.

The following figure includes additional examples of how the evaluation process works.

Example import scenarios

Service Offering (SO) = Topmost artifact in hierarchy

Service Design (SD) = Service design artifact associated with the service offering (SO)

Resource Offering (RO) = Resource offering artifact associated with the service design (SD)		
Source	**Target**	**Explanation**
		None of the imported artifacts exists on the target system. The import operation creates the topmost artifact and its associated artifacts on the target.
		SD1 (service design 1) and RO1 (resource offering 1) already exist on the target system. The import operation creates SO2 and RO2, but reuses the existing SD1 and RO1artifacts.
	Operation Fails	The import operation exits without importing any content because S02 already exists on the target system.

Import using validate option

This operation combines validate and import operations. Before importing, this operation validates the contents of the archive and displays a summary of changes that will be made during the import operation. After displaying this summary, the import with validation operation prompts the user before proceeding with the import.

Import using update option

Use this option with caution. Functionally equivalent artifacts that exist on the target system are updated (overwritten) with changes from the archive. New artifacts are created if they do not exist on the target system.

Import using updatePreserveExisting option

Use this option with caution. This operation imports all artifacts present in an archive, whether they exist on the target system or not. Any artifact with the same name on both the target system and the archive is preserved as follows: name, display name, and description of the *existing* artifact are modified internally, and the display name and the description are appended with "Superseded on" and the date. The name, display name, and description of the *imported* artifact remain intact.

This operation is supported for service offerings, service designs, resource offerings, and resource environments, and does not apply to component palettes. In the case of catalogs, this process applies to the associated artifacts, but does not apply to the catalog itself.

Important:
For all imports and updates, if a dynamic property JSP file by the same name already exists on the target system, the existing file will be used and will not be imported from the archive.

See Examples, for examples of all operations.

Note: Content is added to the log file (content-archive-tool.log) for all operations.

Recommended Best Practices

Before importing or updating:

- Use the export operation to create a zip file of your existing artifacts, ensuring a backup of any artifacts you might be affecting.
- Create a backup of your system or data.
- Understand the differences between available import and update options (some of which can destroy existing data) to make sure you choose the one that matches your expectations.

Supported Artifacts

The following CSA artifacts are supported in CSA artifact archives: component palettes, resource offerings, service designs, service offerings, resource environments, and catalogs.

Component Palettes

Content archive for component palettes

Exporting a component palette creates a content archive (.zip) file. The content archive contains XML documents for the component palette and its associated component types, component templates, and component type constraints. If component templates in a component palette have resource bindings on resource offerings, when the component palette is exported, resource offering XML files are included in the component palette archive. The archive also contains icons for customizing the artifacts, and a Manifest XML document, which contains meta information about the archive files.

During an import or update operation, if required dependencies do not exist on the Operations Orchestration system, an error message identifies these missing dependencies (dependencies such as flows). The content pack that contain these flows must have been deployed to the Operations Orchestration system prior to importing these artifacts. The flows must also have identical signatures and identical paths as the flows on the system from which the artifact was exported.

During import, flow signature-related information is verified in or added to the CSA database (flow signatures are used during the creation of an artifact and when adding a resource synchronization action or an external approval type). This information is resolved by **name** which corresponds to the full path to the Operations

Orchestration flow (for example, `/Library/CSA Content Pack/CSA3.2/Providers/Infrastructure/vCenter/vCenter Clone Server/Actions/vCenter Simple Compute - Deploy`). For information about how to deploy Operations Orchestration content packs, see the *Central User Guide*.

Import process for component palettes

- Import of a component palette by default is an update operation. The import process imports component palettes, including their associated component types, templates, and component type constraints.
- If the component palette already exists on the system, it is updated with any added or removed component types, component templates, and component type constraints.
- Component palettes with the same internal name are considered to be functionally equivalent and are not imported.
- Circular dependencies between palettes are not allowed. For example, Palette A cannot have components that are derived from Palette B while, at the same time, Palette B has components that are derived from Palette A.
- When importing multiple, dependent component palettes that already exist on the target system, it is recommended that you import the palettes in the order of their dependencies.
- The import process for component palettes always ensures system integrity for component type derivation and property propagation. For example, if you have modified properties of a component type in a palette called "Palette A," and there are component types in other palettes that derive from this component type, when an update of 'Palette A' occurs during import, changes that have been made to the component types are propagated to component types in other palettes.

Import process for component templates in a component palette with resource bindings on resource offerings

- During a component palette import, resource offerings are either created when a resource offering with same name does not exist on the system, or updated when a resource offering with the same name exists. This resource offering import behavior occurs because the import option for a component palette is always *update*.
- Resource offerings are not automatically associated with resource providers when component palettes are imported. The `–p` command line option does not override this behavior.

Resource Offerings

Import requirements and prerequisites

- Resource categories

When you import a resource offering, resource categories (such as Compute) and provider types (such as VMware vCenter) are resolved first by **name** and secondly by **display name**. Out-of-the box resource categories and provider types have identical **name** values on all installations of CSA, and automatically resolve correctly during import. User-created resource categories and provider types do not have a **name** match on different installations of CSA, and instead are resolved by **display name**. For example, if a user-created resource category with a display name of Auditing is used for a resource offering, when that resource offering is imported to another CSA installation, an attempt will be made to match a resource category with a **display name** of Auditing. This match is successful only if the user has already created this corresponding resource category on the system in which the import occurs. If a resource category or provider type cannot be resolved by either **name** or **display name**, a new resource category or provider type is automatically created during import of the resource offering. There is no need to add user-created resource categories and provider types ahead of time on the import system; however, if you have done so, ensure the **display name** values used match those on the export system.

- Flows

  During an import or update operation, if required dependencies do not exist on the Operations Orchestration system, an error message identifies these missing dependencies. For resource offerings this can include missing flows. The content pack that contain these flows must have been deployed to the Operations Orchestration system prior to importing these artifacts. The flows must also have identical signatures and identical paths as the flows on the system from which the artifact was exported.

  During import, flow signature-related information is verified in or added to the CSA database (flow signatures are used during the creation of an artifact and when adding a resource synchronization action or an external approval type). This information is resolved by **name** which corresponds to the full path to the Operations Orchestration flow (for example, `/Library/CSA Content Pack/CSA3.2/Providers/Infrastructure/vCenter/vCenter Clone Server/Actions/vCenter Simple Compute - Deploy`). For information about how to deploy Operations Orchestration content packs, see the *Central User Guide*.

Content archive for resource offerings

Exporting a resource offering creates a content archive (.zip) file. The content archive contains XML documents for the resource offering you are exporting, icons for customizing the artifacts, and the Manifest XML document, which contains meta information about the archive files.

Default Import process for resource offerings

Resource offerings that have the same resource category, provider type, properties, and actions are considered to be functionally equivalent and are not imported. See Import and Update for a definition of functional equivalence.

Update process for resource offerings

During the update process, resource offerings with the same internal name that exist on the target system are updated (overwritten) with changes from the archive. New resource offerings are created if they do not exist on the target system.

Note: Resource offerings are identified as equivalent by internal name only for the update operation.

UpdatePreserveExisting process for resource offerings

This process imports the resource offering, whether it exists on the target system or not. During this operation, if there is a resource offering with the same name in the system, the name, the display name, and the description of the resource offering are modified internally; the display name and the description are appended with "Superseded on" and the date. The internal name, display name, and description of the artifact that is imported remain intact.

Important: All providers of the same provider type will be automatically associated to the imported resource offerings.

Functionally Equivalent Resource Offerings

When a resource offering is imported, CSA determines if a functionally equivalent resource offering already exists on the system. If a functionally equivalent resource offering exists, then the import is skipped. Functional equivalence is determined by comparing the resource offering to be imported with other resource offerings that exist on the system, as follows:

- Resource offerings are functionally equivalent if they share the same provider type and resource category, the same set of user defined custom properties, and the same set of lifecycle actions.
- Properties are considered equivalent if they share the same type, name, and value (or values for list properties).
- Lifecycle actions are considered equivalent if they share the same lifecycle state and sub-state, execution order, and action input properties and values.
- There are additional attributes on properties and lifecycle actions that must be identical for equivalence.

For specifics about the precise requirements for resource offering equivalence as it relates to entries in the .zip file produced during resource offering export, see the following table.

Functionally Equivalent Resource Offerings

Element	**Necessary for Equivalence**

`property`	`name`
	`valueType`
	`values`
	`confidential` – only for String property types

`action`	`processDefinition name`
	`lifecycleState name`
	`lifecycleSubstate name`
	`lifecycleExecOrder`
	`errorOnTimeout`

	failOnError timeout all properties must be identical, including the consumerVisible and consumerReadOnly elements for each property consumerVisible
resourceCategory	isCriticalSystemObject determines if this is an out -of-box resourceCategory. If true, name determines equivalence, otherwise displayName determines equivalence.
providerType	isCriticalSystemObject determines if this is an out -of-box providerType. If true, name determines equivalence, otherwise displayName determines equivalence.

Service Designs

The import process supports the import of sequence and topology design archives.

Before you import a sequenced service design archive, complete the following prerequisites:

1. Import all flows that are referenced by resource offerings that are part of the service design. During an import or update operation, if required dependencies do not exist on the Operations Orchestration system, an error message identifies these missing dependencies. The content pack that contain these flows must have been deployed to the Operations Orchestration system prior to importing these artifacts. The flows must also have identical signatures and identical paths as the flows on the system from which the artifact was exported.

1. During import, flow signature-related information is verified in or added to the CSA database (flow signatures are used during the creation of an artifact and when adding a resource synchronization action or an external approval type). This information is resolved by **name** which corresponds to the full path to the Operations Orchestration flow (for example, `/Library/CSA Content Pack/CSA3.2/Providers/Infrastructure/vCenter/vCenter Clone Server/Actions/vCenter Simple Compute – Deploy`). For information about how to deploy Operations Orchestration content packs, see the *Central User Guide*.

2. When importing archives with one or more CloudSystem 8.x topology designs, the following process will be followed to determine the provider and resource pool associated with each design:

- If `-q/--forceCloudOSProvider` or `-r/--forceHPHelionOpenStackProvider`, and `-s/--forceResourcePool` options are specified in command line:

   a. Check target system for match of provider name specified.

      - If found, use that provider for import.
      - If not found, fail topology design import with error provider not found.

   b. Check target system for match of resource pool name specified.

- If found, use that resource pool for import.
- If not found, fail topology design import with error resource pool not found for provider.

- If `-q/-r` and `-s` options are not specified in the command line:

  a. Check target system for match of provider name used in archive.

  - If provider name exists on target system, use that provider for import.
  - If provider name does not exist on target system, check for default provider in CSA resource management.
    - If default provider is set, use that provider for import.
    - If no default provider set, fail topology design import with error provider not found.

  b. If a provider is successfully matched by the name used in the archive, check the target system for a match of the resource pool name used in the archive.

    1. If resource pool name exists on target system, use that resource pool for import.
    2. If resource pool name does not exist on target system, check for default resource pool in CSA resource management.
       a. If default resource pool is set and exists on target system, use that resource pool for import.
       b. If no default is set, fail topology design import with error resource pool not found for provider.

  c. If a default provider is used, check target system for default resource pool.

    1. If default resource pool is set and exists on target system, use that resource pool for import.
    2. If no default resource pool is set, Fail topology design import with error resource pool not found for provider.

3. In the case of a sequenced service design, the export operation does not export any custom service component types that the user has created on the source system. If the service design you want to import has a dependency on any custom component types, the component palettes that contain these custom component types must be imported before the service design can be imported.
4. If you are importing a version of a service design that is an upgrade to another version of a service design, the latter service design must be present on the target system and must also be published.

See the CSA online help for more information on upgrading service designs.

Content archive for service designs

Exporting a service design creates a content archive (.zip) file. The content archive contains XML and JSON documents for the service design you are exporting, as well as associated artifacts, dynamic property JSP files, icons for customizing the artifacts, and the Manifest XML document, which contains meta information about the archive files.

Import process for service designs

The import process imports archives of service designs and their supported artifacts. Supported artifacts for service designs include associated resource offerings. Service designs with the same internal name are considered to be functionally equivalent and are not imported.

Note: When you import a service design that is an upgrade to another service design, the import process checks for upgradability rule violations by comparing the imported service design to its corresponding upgradable design. See the CSA online help for more information on upgradability rules and on upgrading service designs.

Update process for service designs

During the update process, identical artifacts that exist on the target system are updated (overwritten) with the changes from the archive. Artifacts are created if they do not exist on the target system.

Note: Importing a service design using an update option fails if the service design being imported is upgradable to other service designs on the target system. See the CSA online help for more information on upgrading service designs.

UpdatePreserveExisting process for service designs

This process imports all the artifacts present in an archive, whether they exist on the target system or not. During this operation, if there is an artifact with the same internal name in the system, the name, the display name, and the description of the artifact are modified internally; the display name and the description are appended with "Superseded on" and the date. The internal name, display name, and description of the artifact being imported remain intact.

During the import operation on a service design archive, if a service design has active service offerings associated with it, the subscriber options for the service design being imported must match that of the service design on the target system; otherwise, the import of the service offerings fails. If any functionally equivalent resource offerings already exist on the target system, these resource offerings are reused for binding with the service design and these resource offerings are not imported.

Note: Neither service design versions that are an upgrade to other service designs, nor service designs that can be upgraded to other service designs, can be imported using the updatePreserveExisting option.

Service Offerings

Prerequisite for service offerings

If you are importing a version of a service offering that is an upgrade to another version of service offering, the latter service offering must be present on the target system. See the CSA online help for more information on upgrading between service designs.

Content archive for service offerings

Exporting a service offering creates a content archive (.zip) file. The content archive contains XML documents for the service offering you are exporting, as well as associated artifacts, dynamic property JSP files, icons for customizing the artifacts, and the Manifest XML document, which contains meta information about the archive files.

Import process for service offerings

The import process imports archives of service offerings and their supported artifacts. Supported artifacts for service offerings include associated service designs and resource offerings. Service offerings with the same internal name are considered to be functionally equivalent and are not imported.

Update process for service offerings

During the update process, identical artifacts that exist on the target system are updated (overwritten) with the changes from the archive. Artifacts are created if they do not exist on the target system.

UpdatePreserveExisting process for service offerings

This process imports all the artifacts present in an archive, whether they exist on the target system or not. During this operation, if there is an artifact with the same name in the system, the name, the display name, and the description of the artifact are modified internally; the display name and the description are appended with "Superseded on" and the date. The name, display name, and description of the artifact being imported remain intact.

Note: Neither service offering versions that are an upgrade to other service offerings, nor service offerings that can be upgraded to other service offerings, can be imported using the updatePreserveExisting option.

Resource Environments

Content archive for resource environments

Exporting a resource environment creates a content archive (.zip) file. The content archive contains an XML document for the resource environment you are exporting, as well as the Manifest XML document, which contains meta information about the archive files.

Import process for resource environments

The import process imports archives of resource environments. Resource environments with the same internal name are considered to be functionally equivalent and are not imported.

Update process for resource environments

During the update process, identical artifacts that exist on the target system are updated (overwritten) with the changes from the archive. Artifacts are created if they do not exist on the target system.

UpdatePreserveExisting process for resource environments

This process imports the resource environment present in the archive, whether it exists on the target system or not. During this operation, if there is an artifact with the same name in the system, the name, the display name, and the description of the artifact are modified internally; the display name and the description are appended with "Superseded on" and the date. The name, display name, and description of the artifact being imported remain intact.

Important: If providers of the same display name and provider type exist on the target system, use the associate to provider option (-p) to automatically associate these providers to imported resource offerings. Otherwise, you must associate providers to resource offerings manually.

Catalogs

Content archive for catalogs

Exporting a service catalog creates a content archive (.zip) file. The content archive contains XML documents for the service catalog you are exporting, as well as associated artifacts, dynamic property JSP files, icons for customizing the artifacts, and the Manifest XML document, which contains meta information about the archive files.

Import process for service catalogs

The import process imports archives of service catalogs and their supported artifacts. Supported artifacts for service catalogs include associated service designs, resource offerings, service offerings, and resource environments. If the service catalog already exists, the catalog is updated with any added or removed supported artifacts. Service catalogs with the same internal name are considered to be functionally equivalent and are not imported.

Update process for service catalogs

During the update process, identical artifacts that exist on the target system are updated (overwritten) with the changes from the archive. Artifacts are created if they do not exist on the target system.

UpdatePreserveExisting process for service catalogs

This process applies to the associated artifacts, but does not apply to the catalog itself. The process imports all the artifacts present in an archive, whether they exist on the target system or not. During this operation, if there is an artifact with the same name in the system, the name, the display name, and the description of the artifact are modified internally; the display name and the description are appended with "Superseded on" and the date. The name, display name, and description of the artifact being imported remain intact.

During the import of a catalog, an organization identifier is required. You can specify this identifier organization using the –o option. If this is not specified in the command line, the organization identifier from the ownedBy field in the catalog’s XML document is used. An exception is generated if you do not supply an organization identifier during import and if the organization that the catalog is associated with on the source system does not exist on target system.

Note: When you import a catalog using the updatePreserveExisting option, import of the following service offerings is skipped: catalog service offerings if they are upgrades of other service offerings, and catalog service offerings if they are upgradable to other service offerings. See the CSA online help for more information on upgrading catalogs.

Important: Importing the global shared catalog is not supported. You cannot associate a catalog with an organization with the "Provider" business role.

Configuration and Usage

The Content Archive Tool is installed during CSA product installation, typically in Tools\ContentArchiveTool\ in the CSA installation folder.

Configuration Details

config.properties file

The config.properties file must be in the same folder as the content-archive-tool.jar file, often under `<CSA install folder>\Tools\ContentArchiveTool`.

The default property filename is `config.properties`, but this file can have any name. Use the `-c` option to specify the name of the file.

Sample Configuration files

The `content-archive-tool.jar` can produce sample configuration files by executing the following at the command prompt: `<csa_jre>\bin\java -jar content-archive-tool.jar -g` where `<csa_jre>` is the directory in which the JRE that is used by CSA is installed.

The following sample configuration files are created:

- `config.properties.oracle`
- `config.properties.mssql`
- `config.properties.postgresql`

Configuration Properties File Parameters

The sample configuration file not needed by the database in use by CSA can be deleted. For example, if you are using a Microsoft SQL Server database, retain the MS SQL configuration file and if desired rename `config.properties` or other name of your choice. The Oracle configuration file can be deleted since it is not needed.

This following table lists the parameters found in the config.properties file.

Property	Description
`jdbc.driverClassName`	The database driver class. Do not change this value.
`jdbc.dialect`	The database dialect. Do not change this value.

`jdbc.databaseUrl`	The JDBC URL. When specifying an IPv6 address, it must be enclosed in square brackets.
	
	Examples:
	
	Oracle (SSL not enabled):
	`jdbc.databaseUrl=jdbc:oracle:thin:@//127.0.0.1:1521/XE`
	
	Oracle (SSL not enabled, using an IPv6 address): `jdbc.databaseUrl=`
	`jdbc.databaseUrl=jdbc:oracle:thin:@ [f000:253c::9c10:b4b4]:1521/XE`
	
	Oracle (SSL enabled, CSA checks the database DN):
	`jdbc.databaseUrl=jdbc:oracle:thin:@(DESCRIPTION =(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCPS)(HOST = <host>)(PORT =1521))) (CONNECT_DATA = (SERVICE_NAME = ORCL)) (SECURITY=(SSL_SERVER_CERT_DN = "CCN=abc,OU=dbserver,O=xyz,L=Sunnyvale,ST=CA,C=US")))`
	
	where `<host>` is the name of the system on which the Oracle database server is installed and the values for SSL_SERVER_CERT_DN are for the DN of the Oracle database server.
	
	MS SQL (SSL not enabled):
	`jdbc.databaseUrl=jdbc:jtds:sqlserver://127.0.0.1:1433/example;ssl=require`
	
	MS SQL (SSL not enabled, using an IPv6 address):
	`jdbc.databaseUrl=jdbc:jtds:sqlserver://[::1]:1433/example;ssl=request`
	
	MS SQL (SSL enabled): `jdbc.databaseUrl=`
	`jdbc:jtds:sqlserver://127.0.0.1:1433/example;ssl=authenticate`
	
	MS SQL (FIPS 140-2 compliant): `jdbc.databaseUrl=`
	`jdbc:jtds:sqlserver://127.0.0.1:1433/example;ssl=authenticate`
	
	PostgreSQL:
	`jdbc.databaseUrl=jdbc:postgresql://127.0.0.1:5432/csadb`

Property	Description
`jdbc.username`	User name for database
`jdbc.password`	Encrypted password for database

`csa_war.loc`	The path of the csa.war location in the CSA installation. You must change this path if CSA is not installed in the default location.
	Example for Windows:
	`csa_war.loc=C:\\Program Files\\HPE\\CSA\\jboss-as\\standalone\\deployments\\csa.war`
	Example for Linux:
	`csa_war.loc=/usr/local/hpe/csa/jboss-as/standalone/deployments/csa.war`
	Note: You must manually change the csa.war file location information in config.properties on Linux installations because it is not generated in the Linux format.

Communicating with the MS SQL or Oracle Database Using SSL

If SSL is enabled between CSA and MS SQL or the Oracle database, the URL property in the database properties file must be configured correctly, and additional command line options might be required when using an Oracle database.

Important: The Content Archive Tool does not support DN verification.

Database	Configuration options	Command line option(s)	jdbc.databaseUrl value
Oracle	CSA does not check the database DN, client authentication is enabled	`-Djavax.net.ssl.keyStore ="<certificate_key_file>" -Djavax.net.ssl.keyStorePassword =<certificate_key_file_password> -Djavax.net.ssl.keyStoreType`	`jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS= (PROTOCOL = TCPS)(HOST = <host>)(PORT = 1521))) (CONNECT_DATA =(SERVICE_NAME = ORCL)))` where `<host>` is the name of the system on which the Oracle database server is installed.

Database	Configuration options	Command line option(s)	jdbc.databaseUrl value
		`=<certificate_key_file_type>` where <`certificate_key_file`> is the same keystore file defined by the path attribute in the ssl element of the `%CSA_HOME%\jboss-as\standalone\configuration\standalone.xml` file (for example, `%CSA_HOME%\jboss-as\standalone\configuration\.keystore`), <`certificate_key_file_password`> is the password to the keystore file (for example, `changeit`), and <`certificate_key_file_type`> is the `keystore` type (for example, JKS or PKCS12)	
	CSA does not check the database DN, client authentication is NOT enabled	<none>	`jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS_LIST=(ADDRESS= (PROTOCOL = TCPS)(HOST = <host>)(PORT = 1521))) (CONNECT_DATA =(SERVICE_NAME = ORCL)))` where <`host`> is the name of the system on which the Oracle database server is installed.
MS SQL	SSL is enabled	<none>	`jdbc:jtds:sqlserver://127.0.0.1:1433/example;ssl=authenticate`

Command Line Options

Invoke the Content Archive Tool from the command line as follows:

```
<csa_jre>\bin\java -jar content-archive-tool.jar <options>
```

where `<csa_jre>` is the directory in which the JRE that is used by CSA is installed.

Usage:

```
C:\Program Files\HPE\CSA\Tools\ContentArchiveTool><csa_jre>\bin\java -jar
content-archive-tool.jar –h
```

Usage:

```
-h |

-e [[-a <Artifact Type>] [-n <Artifact Name>] | [-u <uuid of artifact>]] [-c
<config property file>] [-t <target output folder>] [-j <oracle jars>]|

-i [-v] [-f | -w] -z <individual zip archive| zip archive folder> [-c
<config property file>] [-o <organization name>] [-r <provider name> -s
<resource pool name>] [-p true|false] [-j <oracle jars>] |

-v -z <individual zip archive| zip archive folder> [-c <config property
file>] [-o <organization name>] [-r <provider name> -s <resource pool name>]
[-j <oracle jars>] |

-l |

-g
```

Content Archive Tool command line options and suboptions are shown in the following table.

Option	Option description	Suboptions associated with the option	Suboption description
`-h,` `--help`	Display syntax and usage information.	none	
`-e,` `--export`	Export artifacts.	`-a, --artifact <artifact_type>`	Artifact type to be exported, otherwise all are exported.
		`-n, --name <artifact_name>`	Artifact Display name of artifact to be exported, otherwise all are exported.
		`-u, --uuid <artifact_id>`	UUID of artifact to be exported.
		`-c, --config <config_filename>`	Configuration properties filename.

Option	Option description	Suboptions associated with the option	Suboption description
		`-t, --target <target_folder>`	Target (output) folder for archive.
		`-j, --jars <Oracle_JAR1 Oracle_JAR2 …>`	Load Oracle JDBC JAR files. Note jar filenames are separated by spaces.
`-i, --import`	Import artifact archives.	`-v, --validate`	Validate before the archive is imported.
		`-f, --updatePreserveExisting`	Preserve existing artifacts with the same name by renaming them; then import all artifacts from archive.
		`-w, --update`	Overwrite/update existing artifact(s) without duplication.
		`-z, --zip <archive_folder>|<archive_filename>`	Required; archive folder or filename. If folder, imports all CSA archives in folder. If filename, defaults to current folder unless folder path is specified.
		`-c, --config <config_filename>`	Configuration properties filename.
		`-o, --orgName <organization_id>`	Organization identifier used for catalog import.
		`-q, --forceCloudOSProvider <provider_name>`	Optional with topology design import; provider to use, must already be configured on target CloudOS system. `–s` (`––forceResourcePool`) must also be specified when using this option.
		`-r, --forceHPHelionOpenStackProvider <provider_name>`	Optional with topology design import; provider to use, must already be configured on target Helion OpenStack system. `–s` (`––`

Option	Option description	Suboptions associated with the option	Suboption description
			`forceResourcePool`) must also be specified when using this option.
		`-s, --forceResourcePool <resource_pool_name>`	Optional with topology design import; resource pool to use, must already be defined on target CloudOS or Helion OpenStack system. `–q` (`--forceCloudOSProvider`) or `–r` (`--forceHPHelionOpenStackProvider`) must also be specified when using this option.
		`-p, --rp [true|false]`	Associate imported resource offerings and resource environments to resource providers on the target system. For resource environments - all resource providers with same display name and provider type will be associated. For resource offerings - all resource providers with same provider type will be associated.
		`-j, --jars <Oracle_JAR1 Oracle_JAR2 …>`	Load Oracle JDBC JAR files. Note jar filenames are separated by spaces.
`-v, --validate`	Validate before the archive is imported.	`-c, --config <config_filename>`	Configuration properties filename.
		`-j, --jars <Oracle_JAR1 Oracle_JAR2 …>`	Load Oracle JDBC JAR files. Note jar filenames are separated by spaces.
		`-o, --orgName <organization_id>`	Organization identifier for export of catalog(s).

Option	Option description	Suboptions associated with the option	Suboption description
		`-z, --zip <archive_folder>|<archive_filename>`	Required; archive folder or filename. If folder, validates all CSA archives in folder. If filename, defaults to current folder unless folder path is specified.
`-l, --list`	List the supported valid artifact types.	None	
`-g, --generate`	Generate sample input config.properties file.	None	

Examples

Important: You must use the same JRE that is used by CSA. If, by default, the system is set up to use a different JRE, include the relative or absolute path to the JRE in the command line. For example, if you installed the JRE included with CSA, use "`..\..\openjre\bin\java`". If you installed a different JRE to use with CSA in the directory `<csa_jre>`, use "`<csa_jre>\bin\java`".

Example 1: Display the Content Archive Tool usage/help.	`java -jar content-archive-tool.jar -h`
Example 2: Generate a sample config.properties file to be used by this tool.	`java -jar content-archive-tool.jar -g`
Example 3: Display a list of artifact types supported for export.	`java -jar content-archive-tool.jar --list`
Example 4: Export all supported artifacts to the current folder. An archive will be created for each primary artifact.	`java -jar content-archive-tool.jar -e`
Example 5: Export all service offerings to the current folder. `SERVICE_DESIGN, RESOURCE_OFFERING, CATALOG`, `RESOURCE_ENVIRONMENT`, and	`java -jar content-archive-tool.jar --export -a SERVICE_OFFERING`

`SERVICE_OFFERING`, `COMPONENT_PALETTE` are valid values for the `–a` option.	
Example 6: Export a resource offering with the display name vCenter Resource Offering to the current folder.	`java -jar content-archive-tool.jar –e –a RESOURCE_OFFERING –n “vCenter Resource Offering”`
Example 7: Export an artifact with id `90cec2ff3b7d9a03013b7db4c6ff003f` to the current folder.	`java –jar content-archive-tool.jar --export --uuid 90cec2ff3b7d9a03013b7db4c6ff003f`
Note: In examples 4 to 7, artifacts could be exported to a target folder instead of the current folder by providing the `–t` or `--target` option with the value of the target folder. The following is an example that exports all catalogs to folder C:\temp\catalogs. `java –jar content-archive-tool.jar –export --artifact CATALOG --target “C:\temp\catalogs”`	
Example 8: Import all artifacts present in the specified archive `SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip` from the current folder.	`java -jar content-archive-tool.jar -i -z SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip`
Example 9: Import all archives stored in the folder `C:\temp\archive.`	`java -jar content-archive-tool.jar -i -z "C:\temp\archive"`
Example 10: Import with validation all artifacts present in an archive `SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip` from the current folder.	`java -jar content-archive-tool.jar -import -v --zip SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip` This operation validates whether or not the artifacts in the archive are already present in the system and provides a summary of the validation. You are then prompted to continue with the import operation. When `Y` (yes) is selected, the import operation continues by default. When `N` (no) is selected, the current archive is not imported. When `Q` (quit) is selected, all remaining import activity is terminated.

Example 11: Import all artifacts present in archive `SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip` from the current folder regardless of whether artifacts by the same name already exist on the target system.	`java -jar content-archive-tool.jar -i -f -z SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip` By default, the updatePreserveExisting (-f) operation validates whether or not the artifacts in the archive are already present in the system and provides a summary of validation. You are then prompted to continue with the import operation. When `Y` (yes) is selected, existing artifacts with the same name as artifacts in the archive are renamed, and all artifacts in the archive are then created on the target system. When `N` (no) is selected, the current archive is not imported. When `Q` (quit) is selected, all remaining import activity is terminated.
Example 12: Import a catalog and all the artifacts in the archive named `CATALOG_Consumer_Catalog_90cec2ff3c0763bf013c076fbf7000c0.zip`, associating the catalog with an organization named `CONSUMER_ORG`. **Note:** Before performing this operation on the catalog archive, the organization CONSUMER_ORG should be created on the target system. If this organization is not present, the Content Archive Tool will associate the catalog to the organization it was associated with on the source system. If neither organization exists on the target system, the Content Archive Tool terminates with an exception.	`java -jar content-archive-tool.jar -i -z CATALOG_Consumer_Catalog_90cec2ff3c0763bf013c076fbf7000c0.zip -o CONSUMER_ORG`
Example 13: Import all artifacts and automatically associate imported resource offerings to resource providers with same provider type on the target system.	`java -jar content-archive-tool.jar -i -p true -z SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip`
Example 14: Import all artifacts preserving original artifacts on target system. Automatically associate the resource offerings imported with existing resource providers using the same provider type on the target system.	`java -jar content-archive-tool.jar -i -f -z SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip -p true`
Example 15: Import all artifacts preserving original artifacts from catalog archive `CATALOG_Consumer_Catalog_90cec2ff3c0763bf013c076fbf7000c0.zip`.	`java -jar content-archive-tool.jar -i -f -z CATALOG_Consumer_Catalog_90cec2ff3c0763bf013c076fbf7000c0.zip`

Example 16: Import all artifacts preserving original artifacts from catalog archive and associate resource provider to resource offering and resource environment.	`java -jar content-archive-tool.jar -i -f -z CATALOG_Consumer_Catalog_90cec2ff3c0763bf013c076fbf7000c0.zip --rp true`
Example 17: Import all artifacts present in the specified archive from the current folder. Because the archive includes topology design(s), a configured CloudOS provider and CloudOS system resource pool on the target system can be specified using the `-q` and `-s` suboptions respectively to override the provider and resource pool values found in the archive.	`java -jar content-archive-tool.jar -i -v -z SERVICE_DESIGN_topo1_90b72c4e425ae0db01425b0c1ae40020.zip -c c:\work\temp\exportimport\cfg.properties.oracle -q CloudOS181 -s localResourcePool`
Example 18: Validate the service offering archive: `SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip`	`java -jar content-archive-tool.jar -v -z SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip`
Example 19: Validate catalog archive `CATALOG_Consumer_Catalog_90cec2ff3c0763bf013c076fbf7000c0.zip`, where the catalog in the archive is to be associated with an organization named `CONSUMER_ORG`.	`java -jar content-archive-tool.jar -v -z CATALOG_Consumer_Catalog_90cec2ff3c0763bf013c076fbf7000c0.zip -o CONSUMER_ORG`
Example 20: Update existing artifacts from service offering archive `SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip`.	`java -jar content-archive-tool.jar -i -w -z SERVICE_OFFERING_Sample_Service_Offering_90cec2ff3c0763bf013c076cf4f10083.zip`
Example 21: Update existing artifacts from resource offering archive `RESOURCE_OFFERING_offerings_1_90cef59c3e682fca013e68302a240063.zip`.	`java -jar content-archive-tool.jar -i -w -z RESOURCE_OFFERING_offerings_1_90cef59c3e682fca013e68302a240063.zip`

Note: In examples 4 through 21, it is assumed that the information about the database and `csa.war` location is stored in the `config.properties` file in the same folder as the `content-archive-tool.jar` file. Use the `-c` option to specify the complete path of the configuration file.

Troubleshooting

Issue	Cause	Workaround
Cannot import or export artifacts that use the double quote (“) character in their name. [158310]	An artifact’s display name specified in the command line that uses the double quote (“) character causes an error.	Use a backslash before each double quote used in the artifact’s display name. For example, to export the service design Simple”Compute”Linux: `java -jar content-archive-tool.jar -e -a SERVICE_DESIGN -n Simple\"Compute\"Linux`
During an export, unexpected behavior occurs if artifact names include special characters. [158317]	Issues occur if artifact names include special Windows command line characters, such as < ! ^ ( ) = ; , >|	Use a caret (^) before each special character. For example: ^< or ^!
Unexpected behavior during import and export when providing a folder name with trailing slashes as a value to –z/--zip or –t/--target suboptions in the command line. [158670]	Issues occur when there are trailing slashes in folder names when provided as values to –z/--zip or –t/--target suboptions.	During an export, do not use trailing slashes. For example: Instead of: `"C:\PCA Tool\output folder\"` Use: `"C:\PCA Tool\output folder"`

Document Change Notes

Date	Description
Aug 2016	Original release of document
Oct 2016	Updated for QCCR1D230293

Send documentation feedback

If you have comments about this document, you can send them to docs.feedback@microfocus.com.

Legal notices

Support

Visit the Hewlett Packard Enterprise Software Support Online web site at https://softwaresupport.softwaregrp.com.

This website provides contact information and details about the products, services, and support that Micro Focus offers.

Micro Focus online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support website to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Access the Software Licenses and Downloads portal
- Download software patches
- Access product documentation
- Manage support contracts
- Look up Micro Focus support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require you to register as a Passport user and sign in. Many also require a support contract.

You can register for a Software Passport through a link on the Software Support Online site.

To find more information about access levels, go to https://softwaresupport.softwaregrp.com/web/softwaresupport/access-levels.

To check for recent updates or to verify that you are using the most recent edition of a document, contact your Client Director.