HCLSoftware

Unica Link V12.1.8 Installation Guide

Contents

Chapter 1. Introduction to Unica Link

Unica Link is a component that can be installed separately from the Unica application.

Unica Link provides both design-time and runtime services. It is invoked at design-time when an administrator configures a Link connection, or when you configure a process box in the Unica Campaign application, or a touchpoint in the Unica Journey application. These configurations are stored in the Unica Link MongoDB repository.

The Campaign application communicates with Unica Link in these ways:

- Passes input and output data by use of CSV files written to a shared location.
- Invokes the Link APIs to send data to a delivery engine or other resource.

Apache Kafka-link

The Journey application communicates with Link by using Apache Kafka messages.

The Kafka component of Unica Link is named 'Kafka-link'. Kafka-link is installed and configured only if you are using Unica Link with Journey.

Unica Link Docker, Native, and Microsoft Windows installation

Unica Link can be installed and deployed by using either the Native installer, the Docker installer, or the Microsoft Windows installer. The same components are installed in either case.

If you are using Docker for other applications or components, use the Docker installer to create and run the Unica Link containers. If Docker is not available, or if it is not currently used, then install Unica Link by using the native installer. Later sections in this documentation describes the installation in Native mode, Docker mode, and Windows mode.

Unica Link must be configured to point to the Unica installation, to set the security methods to use for API calls (for example, HTTP and HTTPS), and the Kafka server details (if you are using Unica Journey). Some of these parameters need to be specified before installation, but they can also be modified after the installation is complete. The configuration of Link differs between Docker, Native, and Windows installation and details are provided in this documentation, under separate sections for each mode.

Before you install Unica Link

These conditions must be met before you install Unica Link:

- Determine whether you want to install on Docker or by using the native installer.
- Check that pre-requisites are satisfied.
- Perform pre-installation configuration to set parameters for the installation.
- Run the Docker or native installer.
- Optionally modify configuration parameters once installation is complete.
- Configure Unica applications to reference the Link installation.

Chapter 2. Unica Link prerequisites

Ensure that these prerequisites are met before installing Unica Link.

Unica Link upgrade

To upgrade Link, the user must uninstall the previous version of Link using the following commands or steps:

Platform	**Commands/Steps**
Native	./Link uninstall
Docker	./Link uninstall
Microsoft Windows	To uninstall Link, complete the following steps: 1. Open the command line interface. 2. Change the current folder to `C:\HCL\Link_<version_number>\DesignServer`. 3. To stop and uninstall Link, run the following command `clean.bat`. 4. Manually delete the folder `DesignServer` from the location `C:\HCL\Link_<version number>\`. 5. Uninstall the Windows Link installation completely (Link comes with Link runtime installation, uninstall runtime component also) by performing the following steps: • Click Windows Start button. • In the **Search box**, type `Add or Remove Programs`. • Select **HCL Link <*Version_Number*> uninstall**.

After successful uninstallation, install the new version of Link. For more details, see Installation of Unica Link in Native Mode on page 15 and Installation of Unica Link in Docker Mode on page 19.

Note: Unica Link is currently supported only on the Chrome browser.

Installation prerequisites for Linux

You need Linux for Native and Docker Installation. The system requirements for Linux are as follows:

Disk space and memory:

- **Disk space** – The minimum amount of disk space required is 8 GB. The amount of disk space required in a production environment varies based upon the solution implementation.
- **Memory** – The minimum amount of memory required is 8 GB. The amount of memory required in a production environment varies based upon the solution implementation.

Recommended development/production environment:

- 4 CPUs,
- 8 GB RAM per CPU so 4 CPUs x 8 GB/CPU = 32 GB RAM
- local disk of 40 to 60 GB

The Design server and the Runtime server can be on the same box.

For production: The amount of disk space or memory required in a production environment varies based upon the solution implementation.

For MongoDB: Minimum version requirement for MongoDB is 4.0.5 and Link only stores design artifacts there, so the required storage is minimal.

Docker installation prerequisites

Ensure that the following system requirements are met prior to performing the Docker installation of Unica Link.

- Linux dockerized container - "Docker Engine" is required in order to run Docker containers.
- Tomcat Version 9.0.14 - Automatically downloads.
- MongoDB Version 4.0.5 - Automatically downloads.
- Redis cache - Automatically downloads.

These prerequisites must be satisfied before you perform a Docker installation of Unica Link:

- Kafka must be installed and request and reply topics must be created with the appropriate topic names specified in the **kafalink.properties** file.
- Docker must be installed and running.
- Any previous installation of Unica must be fully removed by using the **uninstall** command of the **./Link uninstall script**.

Native installation prerequisites

Ensure that these system requirements are met before you perform the Native installation of Unica Link:

- Linux Red Hat 7 - RHEL 7+
- Tomcat version 9.0.26 - Prepackaged in the installer. You do not need to download or acquire.
- MongoDB version 4.0.5 - You must perform a manual installation and configuration of MongoDB. See the instructions on the MongoDB web site.

In addition, the following are required:

- Configuration of `hip-server`
- Configuration of `hip-rest`
- Kafkalink (Native)

Microsoft Windows installation prerequisites

Before you install Link on Microsoft Windows, install Mongo DB Server version 4.0.5 (or higher) on the same computer. By default, Link and MongoDB server is expected to run on the same computer.

If you want Link and MongoDB Server to run on different computers, you must configure the Link installation configuration settings. To change the MongoDB server host and port configuration settings, edit the `config.yaml` file, in the `C:\HCL\Link_<version_number>\config\config.yaml` location, to match your environment.

Link installation prerequisites

To use Unica Link, you must install Link Runtime server.

If you want to use the non-default installation, customize the installation using the instructions described in the Customizing installation section of this documentation.

Enabling authentication with REST API

Before installing Link or REST API, first enable authentication for the REST API execution time.

This enables authentication and enforces security checking before the flow is executed. The REST API call is authorized by accessing the Link user repository using the REST API supplied user and password, or security token. To enable the authentication mechanism, edit the file:

- For native installation: `<Link_install>/config.yaml`
- For docker installation: `<Link_install>/config.yaml`
- For Windows installation: `C:\HCL\Link_<`*`version number`*`>\config\config.yaml`

and then change the settings in `/rest/authentication/enabled` as shown below:

```
enabled=true
```

Link to Proxy Server Documentation

Click on the following link to access Proxy Server documentation in Link 1.1.6:

https://help.hcltechsw.com/hcllink/1.1.6/proxy_server/concept/c_proxy_server_top_lvl.html

Chapter 3. Microsoft Windows installation

This section describes how to install Link using the Windows Installer.

1. Install Unica Link runtime using the Windows Link installer.
2. Open command prompt with administrator privileges, access the Link installation location, and execute the following command:

   ```
   Copy C:\<Link installation location>\.passport.platform to C:\<Link installation location>\.passport
   ```
3. Enable the Unica application to embed the Link UI. Customize the Link configuration by changing the value of the **/client/headers/contentSecurityPolicy** parameter in `C:\HCL\Link_<version-number>\config\config.yaml`. The value must be set to a coma separated list of URLs that point to Unica Journey and Unica Campaign WEB application servers. For example, the URLs can be `http://ipaddress:port` or `http://hostname:port` which point to Unica Journey and Unica Campaign WEB application servers. To apply the changes, restart Link after customization. For more information, see Customizing the installation on page 6.
4. If the URLS are not set correctly Unica cannot access Link in iframe and you will see error while loading Link in iframe.

Licensing

1. From the `<Link installer>\hipmodules`, edit the `platform.properties` file.
2. Set to the base URL of the Unica Platform server. For example, `http://myserver:7001`.
3. Provide the credentials.

Installation prerequisites

Unica Link requires the Link Runtime server to be installed.

If you want to want to use the non-default installation, customize the installation using the instructions described in the Customizing installation section of this documentation.

Starting and stopping Unica Link

After installation, to start and stop Link, complete the following steps:

1. In the Command window, change the current folder to `C:\HCL\Link_<version-number>\DesignServer`.
2. To start the server run the start.bat command. The server may require several minutes, depending on the performance of your computer.
3. To stop Link, run the stop.bat command. The server should stop in a few seconds.

Accessing Unica Link

After the Link is installed, user can access the server by pointing the Web browser to local URL `https://localhost:<port>/login`.

To login for the first time, use username `admin` and the initial administrator password which is stored in `c:\HCL\hipfiles\adminpassword.txt`.

After you login for the first time, change the default password of the admin account for security reasons.

Customizing the installation

You can customize Link by editing settings in the file `C:\HCL\Link_<version-number>\config\config.yaml`.

Review the file content for detailed description of all customization options. Post installation, if you change any of the configuration settings to activate the change, you must reinstall Link by running `clean.bat` followed by `install.bat`.

Customizing the Redis and Link Database

Link installation includes Redis server. If you want to use a preinstalled Redis server, you must customize your Link installation.

Redis customization instructions are written in the `config.yaml` file. Locate the `redis:` section, and modify Redis host and port settings to match your environment. Then restart the Link application.

If you want to customize Redis or MongoDB database, change REDIS or MONGO settings.

Customizing Link URL and Selecting HTTPS protocol

The Link default URL: `https://localhost:4443/login` uses HTTPS protocol on port 4443.

If you want to use a non-default port and HTTPS protocol, you must update the client inbound settings in the configuration file `config.yaml`.

- `port`
- `protocol`
- `host`

For detailed instruction see the `config.yaml` file.

You can modify the Link UI Client configuration after installing Link without the need to reinstall the product by editing settings in this file: `C:\HCL\Link_<version-number>\config\config.yaml.`

To change any of the Link UI Client specific parameters, modify the related settings in the `config` file and save the changes. Then restart Link to activate the changes.

For example, to enable HTTP transport for the Link UI:

1. Change the Link UI specific parameters as defined in the file `config\config.yaml`.
2. Change the **client/inbound/protocol** parameter value to `http`:

```
protocol: "http"
port: 80
```

3. Open an Administrative terminal window and change current folder to DesignServer:

```
cd C:\HCL\Link_<version>\DesignServer\
```

4. Run the Link UI client update utility.

```
stop.bat
start.bat
```

5. Access the Link UI running on `<my Link Server hostname>` server using the following http URL:

```
http://<my Link Server hostname>:<port>/login
```

Installing connectors

Use these steps to install connectors:

1. Copy the connector `ZIP` files and the content of the `apps` folder, from the location `C:\HCL\<Link Version>\` to the location mentioned for the modules parameter in the `%DTXHOME%\config\config.yaml` script.
2. Restart Link.
3. Refer the file `config.yaml`, present in the `<link installation location>\config`, to see the folder path that is set for the **persistence:files** parameter. The default path is `C:\data`. If the default path is `C:\data`, the tmp folder file path is `C:\data\tmp`.

Setting up Campaign

Steps to set up Campaign.

1. Edit the **campaign.properties** file. For more information, see Setting up Campaign on page 23.

Setting up Journey

Steps to set up Journey.

1. Edit the **journey.properties** file. For more information, see Setting up Journey from HCL Link on page 23.

Uninstalling Link

To uninstall Link, complete the following steps:

1. Open the Command window, and change the current folder to `C:\HCL\Link_<version-number>\DesignServer`.
2. Run the **clean.bat** command to stop and uninstall Link.
3. Uninstall Link runtime by performing the following steps:
   - Click Windows Start button.
   - In the **Search box**, type `Add or Remove Programs`.
   - Select **HCL Link <*Version_Number*> uninstall**.
4. Manually delete the `C:\HCL\Link_<version number>\DesignServer` folder.

Verify that the following ports are available on your machine <>

The following table lists the required ports and the corresponding Link configuration properties:

Default port	Used by Link features	Configuration file location (relative to the Link installation folder)	Configuration property name

`4443`	Link server	`config\config.yaml`	`/client/inbound/port`
			used with
			`/client/inbound/protocol`
			Formerly:
			`HIP_HTTPS_PORT`, `HIP_HTTP_PORT`

`80`	Link server	`config\config.yaml`	`/client/inbound/port`
			used with
			`/client/inbound/protocol`
			Formerly:
			`HIP_HTTPS_PORT`, `HIP_HTTP_PORT`

`8080`	Runtime and Link server	`config\config.yaml`	`/tomcat/inbound/ports/http`
			`/rest/inbound/ports/http`
			`/keycloak/serverUrl` (used only if the Keycloak feature is enabled, `/keycloak/enabled`)
			Formerly:
			`authentication.keycloak.serverUrl`
			`HIP_KEYCLOAK_SERVER_URL`

`8443`	Runtime and Link server	`config\config.yaml`	`/tomcat/inbound/ports/https`
			`/server/inbound/ports/https`
			`/client/outbound/server/port`
			`/rest/inbound/authentication/server`
			Formerly:
			`authentication.server`
			`HIP_SERVER_HOST`

Default port	Used by Link features	Configuration file location (relative to the Link installation folder)	Configuration property name
`8005`	Runtime and Link server	`config\config.yaml`	`/tomcat/inbound/ports/shutdown`

`15901`	Event Reporter	`config\config.yaml`	`/rest/eventReporter/port`
			Formerly:
			`event.logger.port`

		`eventreporter.properties`	`LISTENING_PORT`

The Link Windows installation opens a dialog that allows you to configure the Link design time (Link UI) ports. Typically, the Link UI ports are ports `4443` and `80`. The Windows installation verifies that the ports are available on the machine. The installation does not continue if these ports are not available.

Also, the Link Windows installation opens a dialog that allows you to configure Link application server ports (typically ports `8080`, `8443` and `8005`). The Windows installation verifies that the selected ports are available on the machine. The installation does not continue if those ports are not available.

The Event Reporter port (port `15901`) must be manually verified to ensure that it is available.

Installing Apache Kafka-link

These instructions describe how to install Apache Kafka-link on Windows.

Post-installation steps

Steps for editing the Kafka-link properties file

Follow these steps to edit the Kafka-link properties file:

1. Go to this locaiton to edit the files:`C:\<`*`Link installation location`*`>\kafka-link`
2. Provide broker IP with port. For example:**brokers=localhost:9092**
3. Provide the Hip URL of the Link for your Windows local computer Link installation:`hip-url=https://localhost:8443/hip-rest`
4. The name of the request topic: `request-topic=<`*`Topic Name`*`>`For example - `request-topic=OUTGOING_MESSAGES`
5. The name of the response topic: `response-topic=<`*`Topic Name`*`>`For Example - `response-topic=INCOMING_RESPONSES`
6. Save the file.

Starting Kafka-link

Steps for starting Kafka-link.

Follow these steps to start Kafka-link:

1. Open a command prompt, and cd to the following location:`C:\<Link installation location>\kafka-link\`
2. Press **Enter**.
3. Enter the following command:"`kafkalink.bat -P kafkalink.properties`"
4. Press **Enter** to start Kafka-link.

Kafka-link logs

Location information about Kafka-link logs.

Installing Link Offline

For installing Link offline, Internet access is not required. Link offline installation can be done, only with MongoDB preinstalled.

Troubleshooting the installation

In the case of Link Install, Start, or Stop errors, review the logs files from the following folders:

Folder type	File / Folder Path	Description

Main Folder	`<Link-install>\DesignServer\install.log`
	`<Link-install>\DesignServer\start.log`
	`<Link-install>\DesignServer\stop.log`
		Contains information required for resolving Link Design Server installation issues.

Additional Folders	`C:\HCL\Link_<version-number>\restapi\tomcat\server\logs`	Link, Tomcat Application server files
	`c:\HCL\Link_<version-number>\DesignServer\client\daemon\*.log`	Link, WEB NodeJS Application server files
	`C:\HCL\Link_<version-number>\restapi\redis\logs`	Link, Redis server files

Manually updating Apache Tomcat 9 for the Link application

This procedure defines the steps to manually update the installed Apache Tomcat 9 software for the Link application. If the currently installed version of Apache Tomcat 9 has a security issue that is resolved in a later version, you can follow these steps to update it.

Before you begin

Important: Ensure that you back up the `<Link installation folder>/restapi/` folder.

About this task

To manually update Apache Tomcat 9 for Link application on windows platform, complete the following steps:

1. Download the latest 64-bit Windows binaries of Apache Tomcat 9 in ZIP format from the official Apache Tomcat website.
2. Copy the downloaded ZIP archive into the `<Link installation folder>/restapi/tomcat` folder.
3. Complete the following steps:
    a. Type WINDOWS+R.
    b. Right-click the Command Prompt window and select **Run as Administrator**.
    c. In the Command Prompt window, change the current directory to `<Link installation folder>\DesignServer`.
4. Run the following command to clean any existing Apache Tomcat installation:

    ```
    clean.bat
    ```
5. Use a Text Editor to modify the `dtxtomcat.ini` configuration file located at the `<Link installation folder>\restapi\tomcat\` location. Update the string `TomcatVersion=9.0.XX` to match the version of the downloaded Apache Tomcat 9 from *Step 1*. Save the changes.
6. From the same command prompt window folder, as mentioned in *Step 3*, run the following command to install Apache Tomcat:

    ```
    install.bat
    ```

Chapter 4. Linux Installation

You can install Unica Link in two modes:

- Native Installation on page 15
- Docker Installation on page 19

There is a consistent command line interface for the installation and management in either modes. The Link script is the entry point for the CLI.

Link Utility

After extracting the HCL Link package, you can execute the Link script to begin installation.

For more information on installation, see Installation of Unica Link in Native Mode on page 15 and Installation of Unica Link in Docker Mode on page 19.

The Link script supports the following commands:

- configure
- install
- start
- status
- stop
- logs
- uninstall
- <add import certificates command>

configure Command

The configure command is required before using any other command. This command is used to specify the installation type, the Unica integration, and the user or group that the installation is completed as. You must specify the correct option for `--integration` at a minimum.

Options

- `--type <native|docker>` - Specify the installation type. Default: `native`.
- `--integration <unica>` - Specify your integration.
- `--license-file` - This option is intended for standalone customer use only. Do not use if you have received Link as an embedded solution.
- `--user` - The user to install Unica Link as. This will determine which user has ownership of the processes and directories. This must be an already existing user. Default: `current user`.
- `--group` - The group to install Unica Link as. This provides you with flexibility to share Unica Link directories with a group of admins. Default: `current group`.

Notes

You cannot change the integration type after you run install. If you notice an error after installation, you must extract the Unica Link package into a new directory and start over.

install Command

The behavior of the install and uninstall commands is extremely similar. Both commands will act upon the default components if none are specified as options. You can finetune the scope of each action by listing individual components.

CAUTION: Embedded installations generally require all default components running for proper functionality. Do not use these options if you do not understand them fully.

Options

- `--runtime` - Runtime REST API.
- `--server` - Link Server.
- `--kafka-link` - Kafka Link.
- `--apps` - Apps in `integration-context/apps`.
- `--connectors` - Connectors in `integration-context/connectors`.

Notes

Connectors are not included in the Link package by default. You must create the `<link_package>/integration-context/connectors` directory and place your connector files there before connector installation. You can update app property files by re-running the installation (with the `--apps` option).

start Command

The behavior of the start and stop commands is extremely similar. Both commands will act upon the default components if none are specified as options. You can fine tune the scope of each action by listing individual components.

CAUTION: Warning: Embedded installations generally require all default components running for proper functionality. Do not use these options if you do not understand them fully.

Options

- `--runtime` - Runtime REST API.
- `--server` - Link Server.
- `--kafka-link` - Kafka Link.

status Command

The status command retrieves the status of each Unica Link component. Though the usage is similar between the native and docker installation types, the possible states are different.

Options

- `--runtime` - Runtime REST API.
- `--server` - Link Server.
- `--kafka-link` - Kafka Link.

Possible States - Native

- Not Installed
- Stopped
- Running
- Unknown

Possible States - Docker

- Not Installed
- Created
- Dead
- Exited
- Paused
- Running
- Restarting

stop Command

The behavior of the start and stop commands is extremely similar. Both commands will act upon the default components if none are specified as options. You can fine tune the scope of each action by listing individual components.

CAUTION: Warning: Embedded installations generally require all default components running for proper functionality. Do not use these options if you do not understand them fully.

Options

- `--runtime` - Runtime REST API.
- `--server` - Link Server.
- `--kafka-link` - Kafka Link.

logs Command

The logs command prints the primary log of the chosen component. Because the log files may increase in size, you should pipe the output of this command into a tool so you can navigate better.

Options

- `--runtime-app` - Runtime REST API.
- `--server-app` - Link Server - Backend.
- `--client` - Link Server - User Interface.
- `--kafka-link` - Kafka Link.

uninstall Command

The behavior of the install and uninstall commands is extremely similar. Both commands will act upon the default components if none are specified as options. You can finetune the scope of each action by listing individual components.

CAUTION: Embedded installations generally require all default components running for proper functionality. Do not use these options if you do not understand them fully.

Options

- `--runtime` - Runtime REST API.
- `--server` - Link Server.
- `--kafka-link` - Kafka Link.
- `--apps` - Apps in `integration-context/apps`.
- `--connectors` - Connectors in `integration-context/connectors`.

Notes

Connectors are not included in the Link package by default. You must create the `<link_package>/integration-context/connectors` directory and place your connector files there before connector installation. You can update app property files by re-running the installation (with the `--apps` option).

Installation of Unica Link in Native Mode

Dependencies

- `bash` (Bourne Again SHell)
- `ksh` (KornShell)
- `libnsl`
- GNU `sed`
- MongoDB server

Accessing Unica Link on HTTP through load balancer HTTPS URL

This section describes the accessing of Unica Link on HTTP through load balancer HTTPS URL and internally routed to link by using nginx or any other load balancer.

For accessing Unica Link through load balancer URL, you need to set three context rules as follows:

- `hip-rest` - It must be redirected to the rest port.
- `tx-server` - It must be redirected to the client/node port.
- `link client` - It must use the client port.

See note for the `tx-server` context rules:

- The nginx `tx-server` context communicate to link node via the Link node/client port, not directly to link tomcat through the tomcat port (Link server).
- The connection from the Link node/client to the backend server will run through the settings which we have under client → outbound in `config.yaml` (Link node/client redirection to tomcat/link server).

Native System Configuration

Native system configuration includes:

- changing the temporary directory, or
- removing `noexec` from the `/tmp` directory.

Changing the Link Temporary Directory

To change the HCL Link temporary directory, complete the following steps:

1. Create a new directory and grant all permissions to the user, group, and other. Run the following command:

```
mkdir /opt/hiptmp
chmod 777 /opt/hiptmp
```

2. Add a new line to `tomcat-context/install/restapi/tomcat/setenv.sh` to specify the new directory. For example:

```
export JAVA_OPTS="$JAVA_OPTS -Djava.io.tmpdir=/opt/hiptmp"
```

3. Modify the existing lines in `config.yaml` for the long and short workers as shown here:

```
longTaskProcessJvmOptions="-Xmx2g -Djava.io.tmpdir=/opt/hiptmp"
shortTaskProcessJvmOptions="-Djava.io.tmpdir=/opt/hiptmp"
```

4. Configure link runtime to use the same directory for JVM options. Go to "JVM Options" under "runtime" section of `config.yaml` and add a new option for JVM:

```
option4: "-Djava.io.tmpdir=/opt/hiptmp"
```

Note: If the connectors do not use the access token expiry action to re-generate the access token after its expiry, then step 4 can be ignored.

5. Unica Journey application related only - configure Kafka-Link to use the same Temporary Directory, open the Kafka-Link script file located at:

```
<link_installation_dir>/integration-context/install/kafkalink.sh.
```

6. Unica journey related only - replace below line in kafkalink.sh and save:

```
java -classpath "./" com.hcl.hch.service.KafkaLink $
```

with the modified line to set the temporary directory:

```
java -Djava.io.tmpdir=/opt/hiptmp -classpath "./" com.hcl.hch.service.KafkaLink $
```

7. Unica journey related only - open the KafkaLink properties file at `<link_installation_dir>/integration-context/install/kafkalink.properties`. Change data-directory as per your choice (default value is /tmp) and save. For example:

   ```
   data-directory=/tmp
   ```

   As per above example – it should be as below (assuming /opt/hiptmp is a new temporary location)

   ```
   data-directory=/opt/hiptmp
   ```

   Note: Steps 5, 6 and 7 are not required if you are using Unica campaign only.

8. Restart Link.

Changing the Mount Option for `/tmp`

The `/tmp` directory must be mounted with `noexec`. To verify, run the following commands:

```
mount | grep /tmp
```

```
tmpfs on /tmp type tmpfs (rw,nosuid,nodev,noexec,relatime)
```

If `/tmp` directory is already mounted with `noexec`, you can change the mount option for `/tmp` directory. To change the the mount option for `/tmp` directory, complete the following steps:

1. Ensure that you are using root user privileges.
2. Edit `/etc/fstab` and remove noexec from the mount options of `/tmp`.
3. Either:
   a. Reboot the system, or
   b. Run the following command:

      ```
      mount -o remount /tmp
      ```

Configuration

To configure most properties of Link, use the file `config.yaml`.

Review the `config.yaml` file before you perform the installation to confirm that all default options are as expected.

To reconfigure after installation, update those files, and restart Link.

You will notice several directories specified in the configuration files. If you, or the group you install as, lack Read/Write access to these directories, contact a system administrator to create the directories and transfer ownership.

If you are installing as a non-root user, use ports above 1024 to avoid permission issues. The default ports are:

```
Tomcat (REST API + Server): HTTPS 8443, HTTP 8080
UI Client: HTTPS 4443
```

Installation

Read the Link Utility documentation before following the installation steps. You may need to adjust commands according to your use-case.

1. The following commands are intended for a quick-start scenario:
    a. Use the following command to generate configuration:

    ```
    ./Link configure --generate
    ```

    b. Change the Unica Journey/Unica Campaign/Unica Platform properties under `<Link installer>/ integration-context/apps/` and place the connectors under `<Link installer>/ integration-context/connectors`.
    c. Enable the Unica application to embed the Link UI. Customize the Link configuration by changing the value of the **contentSecurityPolicy** parameter in `<Link installer>/Config.yaml` file. The value is list of URLs that point to Unica Journey and Unica Campaign web application servers. For example, the URLs can be

    `http://ipaddress:port` or `http://hostname:port` which point to Unica journey and Unica Campaign web application servers.

    If the URLs are not set correctly, Unica cannot access link in iframe and you will see error while loading link in iframe.
    d.
    ```
    ./Link configure --type native --integration unica
    ```
    e.
    ```
    ./Link install
    ```
    f. Change the kafkalink properties under `<Link-install>/integration-context/install/`.
    g. Either start all components or continue to the troubleshooting section for any errors.

    ```
    ./Link start
    ```

2. When Link is up and running, navigate to the location that is set for the environment variable files and create a new folder `tmp`.

    Refer to the file `config.yaml`, present in the installation location, to see the folder path that is set for the files parameter. The default path is `/opt/hip-rest`. If the default path is `/opt/hip-rest`, the `tmp` folder file path is `/ opt/hip-rest/tmp`.

Installing Link Offline

For installing Link offline, Internet access is not required. Link offline installation can be done, only with MongoDB preinstalled.

Manually updating Apache Tomcat 9 for the Link application

This procedure defines the steps to manually update the installed Apache Tomcat 9 software for the Link application. If the currently installed version of Apache Tomcat 9 has a security issue that is resolved in a later version, you can follow these steps to update it.

❗ **Important:** Ensure that you back up the `<`*`Link installation folder`*`>`*`tomcat-context`*`/install/restapi/` folder.

To manually update Apache Tomcat 9 for Link application on Linux native platform, complete the following steps:

1. Download the latest Linux binaries of Apache Tomcat 9 in `tar.gz` format from the official Apache Tomcat website.
2. Open a Linux terminal window and change the current directory to the Link installation directory `<Link installation folder>`.

   Important: Ensure that you are logged in as the same user who initially installed the Link application.

3. Execute the following commands to stop and uninstall the current Link installation:

   ```
   ./Link stop
   ./Link uninstall
   ```

4. Change the current directory to the installation directory `<Link installation folder>/tomcat-context`, run the following command:

   ```
   cd <Link installation folder>/tomcat-context
   ```

5. Create a temporary folder and extract the contents of Link `hip-core.tar.gz` to it, run the following command:

   ```
   mkdir tmp
   tar -xf hip-core.tar.gz -C tmp
   ```

6. Copy the downloaded `tar.gz` binaries into `<Link installation folder>/tomcat-context/tmp/restapi/tomcat` folder.
7. Open the `dtxtomcat.ini` configuration file located at `<Link installation folder>/tomcat-context/tmp/restapi/tomcat/` location using a Text Editor. Update the string `TomcatVersion=9.0.XX` to match the version of the downloaded Apache Tomcat 9 from *Step 1*. Save the changes.
8. Update the Link `hip-core.tar.gz` file by running the following command from the `<Link installation folder>/tomcat-context/tmp` folder:

   ```
   tar -czf ../hip-core.tar.gz .
   ```

9. Change the current directory to the Link installation directory `<Link installation folder>`, by running the following command:

   ```
   cd <Link installation folder>
   ```

10. Execute the following commands to install and start Link:

    ```
    ./Link install
    ./Link start
    ```

Installation of Unica Link in Docker Mode

Dependencies

- Docker

Important: Unica Link does not offer Docker installation compatibility on Red Hat platforms version 8 and beyond.

Configuration

To configure most properties of Link, use the file `config.yaml`.

Review the `config.yaml` file before performing the installation to confirm all default options are as expected.

To reconfigure after installation, update those files and reinstall Link.

You will notice several directories specified in those configuration files. If you, or the group that you install as, lacks Read/Write access to these directories, contact a system administrator to create the directories and transfer ownership.

If you are installing as a non-root user, use ports above 1024 to avoid permission issues. The default ports are:

```
REST API: HTTPS 9443, HTTP 8080
```

Installation

Read the Link Utility documentation before following the installation steps. You may need to adjust commands according to your use-case.

1. The following commands are intended for a quick-start scenario:
    a. Use the following command to generate configuration:

    ```
    ./Link configure --generate
    ```

    b. Change the Unica Journey/Unica Campaign/Unica Platform properties under `<Link installer>/ integration-context/apps/` and place the connectors under `<Link installer>/ integration-context/connectors`.
    c. Enable the Unica application to embed the Link UI. Customize the Link configuration by changing the value of the contentSecurityPolicy parameter in `<Link installer>/Config.yaml` file. The value is list of URLs that point to Unica journey and Unica Campaign web application servers. For example, the URLs can be `http://ipaddress:port` or `http://hostname:port` which point to Unica journey and Unica Campaign web application servers.

    If the URLs are not set correctly, Unica cannot access link in iframe and you will see and error while loading the link in iframe.

    d.
    ```
    ./Link configure --type docker --integration unica
    ```
    e.
    ```
    ./Link install
    ```
    f. Change the kafkalink properties under `/opt/hip-rest/config/`.
    g. Either start all components or continue to the troubleshooting section for any errors.

    ```
    ./Link start
    ```

Installing Link Offline

Internet access is required for the installation of Link.

Chapter 5. Upgrading the Unica Link Multi Partition Utility

Admin tool guide to update the multi partition upgrade utility for Linux Docker, Windows and Linux Native.

Upgrading the Multi Partition Utility for Linux Docker

To update the multi partition for Linux follow these steps:

1. Copy the `hl8455.csv` file to this location: `hip-server:/usr/local/hl8455.csv`.
2. Execute the admin tool command:

```
docker exec -it hip-server java -classpath /usr/local/tomcat/webapps/tx-server/WEBINF/
 classes:/usr/local/tomcat/webapps/tx-server/WEB-INF/lib/*com.hcl.tx.server.tools.AdminTool
 updateconnectionpartition -accountname Default -name _app_testunica -username admin -email
 admin@example.com -path /usr/local/hl8455.csv
```

3. Use the `updateconnectionpartition` operation to update the connection partition for the first time.

Important: Make sure that you have the connection and partition data prepared in a csv format. For example: `csvheader: connectionName` and `partitionName`.

Upgrading the Multi Partition Utility for Windows

To update the multi partition for Windows follow these steps:

1. Copy the CSV file to the link installation directory. For example, `C:\HCL\Link_10.9.9`.
2. Set the `DTXHOME` to the link installation home directory. For example, `C:\HCL\Link_10.9.9`.
3. Set the `JAVA_HOME` to the `JRE` home directory. For example: `%DTXHOME%\java`.
4. Make sure that the PATH includes `%DTXHOME%` and `%JAVA_HOME\bin%`.
5. Go to the Tomcat installation directory and navigate to `webapps\tx-server\WEB-INF\classes` subdirectory.
6. Set the commands:

```
set CLASSPATH=.;..\lib*
java com.hcl.tx.server.tools.AdminTool updateconnectionpartition -accountname Default -name
 _app_testunica -username admin -email admin@example.com -path C:\HCLLink_10.9.9\filename.csv
```

Upgrading the Multi Partition Utility for Linux Native

To update the multi partition for Linux Native follow these steps:

1. Copy the csv to the link installation directory. For example: `/opt/hcl/hip`.
2. Set the `DTXHOME` to the link installation home directory. For example: `/opt/hcl/hip`.
3. Set the `JAVA_HOME` to the `JRE` home directory. For example: `$DTXHOME/java`.
4. Make sure that the `LD_LIBRARY_PATH` includes `$DTXHOME/libs`.
5. Make sure that the `PATH` includes `$JAVA_HOME/bin`.
6. Go to the Tomcat installation directory and navigate to `webapps/tx-server/WEB-INF/classes` subdirectory.
7. Set the commands:

```
export CLASSPATH=.;..\lib*
java com.hcl.tx.server.tools.AdminTool updateconnectionpartition -accountname Default -name
 _app_testunica -username admin -email admin@example.com -path C:\HCL\Link_10.9.9\filename.csv
```

To list the operations supported by admin utility, run the following command:

```
java com.hcl.tx.server.tools.AdminTool help
```

Chapter 6. Setting up Journey and Campaign for Unica Link

The Unica system IPADDRESS contains the following setup:

- Kafka broker
- Unica Journey where Journey engine is running
- Unica Campaign

Setting up Journey from HCL Link

About this task

To set up Journey follow these steps:

1. Go to the apps folder: `<`*`link_install`*`>/link-context/apps`.
2. Update the `journey.properties` file as follows:
   a. Set the file to point to the Journey server. Ensure that a slash does not exist at the end of the URL: login_url=`http:// IPADDRESS:PORT/journey/api/login` data_definition_url=`http:// IPADDRESS:PORT/journey/api/datadefinitions/point`
   b. Specify Journey credentials:username=<*username*>password=<*password*>

Setting up Campaign

About this task

To set up Campaign follow these steps:

1. Go to the apps folder: **UnicaLink_**<*n.n.n.n*>.
2. Update the **campaign.properties** file as follows:
   a. Set the file to point to the Campaign server. Ensure there is not a slash at the end of the URL: login_url=`http:// IPADDRESS:PORT/unica/api/manager/authentication/login` data_definition_url=`http:// IPADDRESS:PORT/Campaign/api/campaign/rest/v3/link/field-information?actionId`
   b. Specify campaign headers:m_user_name=<*username*>m_user_password=*password*api_auth_mode=<*manager*>

Setting up Journey to send messages to Kafka topics

In order for the Unica Journey application to send messages to Kafka topics, this setup is required.

About this task

For Kafkalink to work from Unica Link, perform these steps:

1. Update the **kafkalink.properties** file located in this location: <*link_install*>**/integration-context/install**.
2. Update the following:

   - List of one or more Kafka brokers= **IPADDRESS:9092**
   - The base URL for the HIP runtime hip-url=`https://IPADDRESS:<PORT>/hip-rest/`
   - The base URL for the HIP designTime, hip-design-url= `https://IPADDRESS:<PORT>/tx-server/rest/`
   - App project name project-name=`_app_journey`
   - The name of the request topic request-topic=**OUTGOING_MESSAGES**
   - The name of the response topic response-topic=**INCOMING_RESPONSES**
   - The directory where files are created before sending to HIP data-directory=**/tmp**
   - For SSL enabled authentication, set the following properties:

     ```
     security.ssl=true
     # For kerberos, use protocol as SASL_PLAINTEXT
     security.protocol=SSL
     ssl.truststore.location=/data/config/kafka.client.truststore.jks
     ssl.truststore.password=hcl2020

     security.authentication =username
     ssl.keystore.location=/data/config/kafka.client.keystore.jks
     ssl.keystore.password=hcl2020
     ssl.key.password=hcl2020
     ssl.endpoint.identification.algorithm=
     ```
   - For SASL plaintext related enabled authentication, set the following properties:

     ```
     security.ssl=false
     security.sasl=true
     security.protocol=SASL_PLAINTEXT
     sasl.mechanism=PLAIN
     security.sasl.auth.login.config =<Path for kafka_client_jaas.conf>
     ```

     Example: If folder path for `kafka_client_jaas.conf` is `/opt` then `security.sasl.auth.login.config=opt/kafka_client_jaas.conf`

     Please note that the contents of `kafka_client_jaas.conf` should be like this for example:

     ```
     KafkaClient
     { org.apache.kafka.common.security.plain.PlainLoginModule required username="xxxxx"
      password="xxxxx"; };
     ```
   - For Kerberos enabled authentication, set the following properties:

     ```
     security.sasl=true
     sasl.mechanism=GSSAPI
     sasl.kerberos.service.name=kafka
     security.sasl.auth.login.config=
     # Either specify the location JAAS configuration file in the above property or
     # provide the configurations directly in the below property.
     sasl.jaas.config=
     # Specify the location of krb5 configuration file
     java.security.krb5.conf=
     ```

3. Update the **platform.properties** file located under *<link_install>***/integration-context/apps**. Set the following properties:
   - Set the base URL of the Unica Platform server. For example: `http://myserver:7001`. Do not specify a slash (/) after the URL.
   - base_url=`http://IPADDRESS:<PORT>`
   - Credentials: username=**asm_admin** password=**password**

Chapter 7. Link Cosmos DB integration

The purpose of this document is to guide on integrating your web application with Azure Cosmos DB for MongoDB. By following the steps outlined herein, you will seamlessly incorporate Cosmos DB into your application's data storage and retrieval processes.

This document aims to:

- Offer a clear and systematic approach to integrating your web application with Cosmos DB alongside MongoDB as the primary database.
- Provide troubleshooting tips and solutions for common integration challenges.
- Allow switching between MongoDB and Cosmos DB by making changes in the `config.yaml` configuration file.

Audience and prerequisites

This document is designed to serve various audiences of professionals involved in the integration and management of a web application utilizing Azure Cosmos DB for MongoDB along with existing MongoDB in Link.

Ensure that these prerequisites are met before integrating your web application with Azure Cosmos DB for MongoDB:

- **Access Credentials**: Obtain valid access credentials for Azure Cosmos DB and MongoDB instances.
- **Link Framework**: Ensure your web application is built on the Link framework.
- **Configuration Detail**s: Have the necessary configuration details, including Cosmos DB URI and relevant settings.

Integration overview

The Link Cosmos DB Integration is a strategic enhancement for web applications utilizing Azure Cosmos DB for MongoDB in conjunction with existing MongoDB setups. This integration seamlessly integrates Cosmos DB, offering a systematic approach to data storage and retrieval processes within your web application.

By enabling the coexistence of MongoDB and Cosmos DB, this integration ensures flexibility and scalability in managing diverse data requirements. The following are the benefits:

Flexibility in Database Selection	The integration provides the flexibility to choose between MongoDB and Cosmos DB, allowing developers to leverage the strengths of each database for specific use cases without major code modifications.
Seamless Transition	With changes made in the config.yaml file, transitioning between MongoDB and Cosmos DB is a straightforward process. This ensures minimal disruption to existing workflows during database configuration changes.
Enhanced Query Performance	Optimizations in index configurations and parameter adjustments contribute to improved query performance,

	ensuring that data retrieval processes are efficient, regardless of the underlying database technology.
Compatibility Assurance	The integration undergoes meticulous unit and integration testing, ensuring that existing functionalities within Link remain consistent across MongoDB and Cosmos DB. This compatibility assurance minimizes the risk of unforeseen issues during and after the integration process.
Scalability and Future-Readiness	Link CosmosDB Integration is designed to scale with the evolving needs of your web application. Whether accommodating growing data volumes or adapting to changes in database technology, this integration provides a robust foundation for future growth.

Key features

The following are the key features:

Unified Database Support	Link CosmosDB Integration facilitates the simultaneous use of MongoDB and Cosmos DB within the Link framework. This means your application can seamlessly switch between these databases based on specific needs, providing a unified approach to data management.
Metadata Verification	Verifies the install log. This allows applications to dynamically identify and adapt to Cosmos DB, ensuring proper handling of database-specific features and requirements.
Optimized Indexing	Adjustments in index configurations.
Schema and Chunk Size Customization	This allows schema and chunk size customization.

Compatibility

The Link Cosmos DB Integration is designed to work seamlessly within various system and platform environments, ensuring broad compatibility and flexibility in deployment.

Here is an overview of the compatibility aspects:

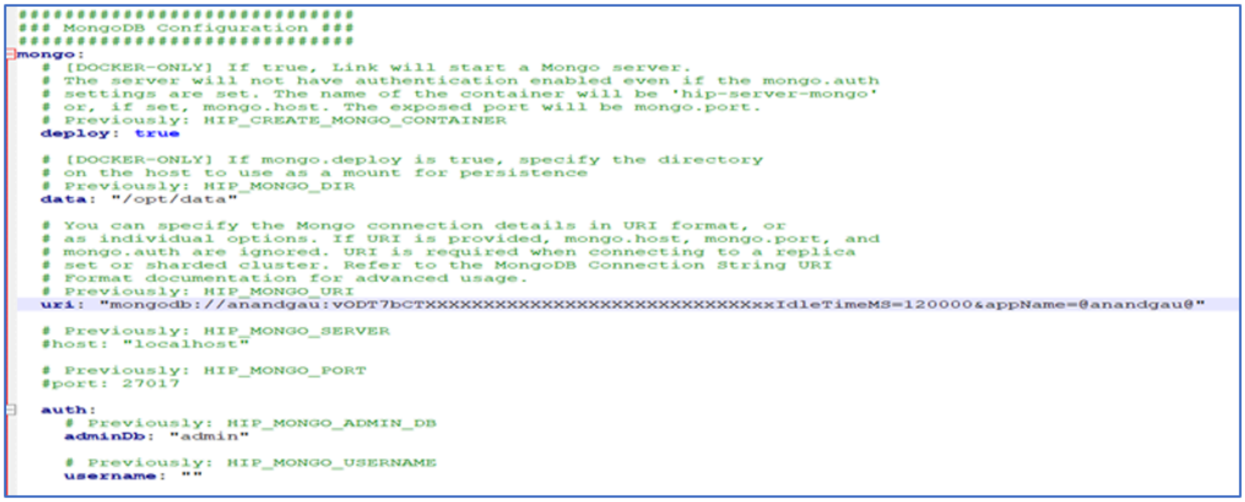
Operating Systems	The integration is compatible with popular operating systems, such as: • Microsoft Windows • Linux

Web Browsers	The web applications utilizing Link Cosmos DB Integration are compatible with standard web browsers, such as: • Google Chrome • Mozilla Firefox • Microsoft Edge

Configuration details

The following are the configuration details for successfully configuring the Link Cosmos DB Integration:

- Update the value of cosmosDBUri to the URI of your Azure Cosmos DB for MongoDB in the config.yaml file if the customer is using cosmosDB, otherwise keep the existing MongoDB setting.



```
###############################
### MongoDB Configuration ###
###############################
mongo:
  # [DOCKER-ONLY] If true, Link will start a Mongo server.
  # The server will not have authentication enabled even if the mongo.auth
  # settings are set. The name of the container will be 'hip-server-mongo'
  # or, if set, mongo.host. The exposed port will be mongo.port.
  # Previously: HIP_CREATE_MONGO_CONTAINER
  deploy: true

  # [DOCKER-ONLY] If mongo.deploy is true, specify the directory
  # on the host to use as a mount for persistence
  # Previously: HIP_MONGO_DIR
  data: "/opt/data"

  # You can specify the Mongo connection details in URI format, or
  # as individual options. If URI is provided, mongo.host, mongo.port, and
  # mongo.auth are ignored. URI is required when connecting to a replica
  # set or sharded cluster. Refer to the MongoDB Connection String URI
  # Format documentation for advanced usage.
  # Previously: HIP_MONGO_URI
  uri: "mongodb://anandgau:vODT7bCTXXXXXXXXXXXXXXXXXXXXXXXXXXXXXxxIdleTimeMS=120000&appName=@anandgau@"

  # Previously: HIP_MONGO_SERVER
  #host: "localhost"

  # Previously: HIP_MONGO_PORT
  #port: 27017

  auth:
    # Previously: HIP_MONGO_ADMIN_DB
    adminDb: "admin"

    # Previously: HIP_MONGO_USERNAME
    username: ""
```

Note: Link is already using MongoDB and mongoClient supports CosmosDB, so no change is required in existing jars.

- Updated Link Hip-Server build to integrate MongoDB and cosmosDB.
  - `_isCosmosDB` flag added to verify Cosmos DB from metadata.
  - For cosmos indexes on the collections changed from `index.hashed` to `index.ascending` as Cosmos DB does not support hashed indexes.
  - The `schema.getCurrentTypes` is set to `2,000` for Cosmos DB while it is `10,000` for MongoDB.
  - The chunk size is set to `2,000,000` (2 Mb) for Cosmos DB as it is `10,000,000` (10 Mb) for MongoDB.

Integrating Cosmos DB with MongoDB

By following these steps, you can successfully integrate Cosmos DB with MongoDB:

1. **Setting Up Cosmos DB**

   To initiate the integration of Cosmos DB with MongoDB, start by creating a Cosmos DB Account specifically configured for MongoDB.

   - Creating a Cosmos DB Account (Cosmos for MongoDB).
   - Select Cosmos for MongoDB option while creating Cosmos account in Azure.

   Note: Ensure that the total throughput limit is sufficiently high, with a minimum of 40,000 Request Units per second (RU/s).

2. **Download and Install Link**

   For installing Link, see the earlier chapters related to Link installation.

   Note: To utilize Cosmos DB with Link, if Link is already installed, it is recommended to uninstall the existing installation and then perform a fresh installation. This ensures that the installation process is aligned with the Cosmos DB configuration requirements and helps prevent any potential conflicts or issues that may arise from an existing installation. Follow the uninstallation guidelines provided in this guide before initiating the reinstallation process to ensure a clean and seamless integration with Cosmos DB.

3. **Link Configuration**

   Configure the database connection settings to establish the link between Cosmos DB and MongoDB.

   - *Database Connection Settings*: Update the `config.yaml` file and ensure that the Cosmos DB URI is correctly specified for linking with Cosmos DB.
   - *Authentication and Authorization*: No changes are required in the authentication and authorization settings.

4. **Data Migration**

   During the installation of the link, the `mdb` and `rdb_XXXXX` databases will be created under Cosmos DB. All predefined collections and documents will be generated, ensuring a seamless migration of existing data.

5. **Restart Link**
6. **Test and Deployment**

   - Conduct thorough testing of the integration.
   - Deploy the integration to the production environment after successful testing.

Known Issues while using Cosmos DB

Refer to the following known issues while using Cosmos DB:

Performance Degradation Compared to MongoDB

Users may experience performance degradation when using Cosmos DB compared to MongoDB in certain scenarios. This could be attributed to differences in underlying architecture, indexing strategies, or query optimization techniques between the two databases.

Credential Entry Discrepancy on Windows Installation

During the configuration of Cosmos DB on a Windows installation, users may encounter a discrepancy in the credential entry process. Unlike MongoDB, where users typically enter MongoDB credentials, Cosmos DB may require users to input specific

Cosmos DB credentials. This difference in the credential management process could potentially lead to confusion or errors during setup.

Chapter 8. Logging

Logs are produced for your use by Link, the Runtime REST API, and Kafkalink.

Link Logs

These are the logs produced by Link:

- For Native Installer:
  - **./Link logs --runtime-app**
  - **./Link logs --client**
- For Docker Installer:
  - **docker logs hip-server**
  - **docker logs hip-rest**

Runtime REST API logs

These logs are produced by the Runtime REST API:

Native Install:

- **./Link logs --runtime-app**

Docker Install:

- **docker logs hip-rest**

Kafkalink logs

These logs are produced by Kafkalink:

Native Install:

- **./Link logs -kafka-link**

Docker Install:

- **docker logs kafka-link**

Making logs verbose

To make logs more verbose you can modify the **config.yaml** file. Modify the file as follows:

1. For Native or Docker:
   - Native Install: Go to: <*install_dir*>/**config.yaml**. By default, <*install_dir*> is in <*installation_tar*>**/tomcat-context/install**
   - Docker install: Edit **config.yaml** in the **hip-rest** Docker container by using the command:

     docker exec -it hip-rest vi /opt/runtime/config.yaml
2. Change the value of **logging:level: TRACE**.
3. For Native or Docker:
   - Native Install:`./Link stop` and `./Link start`
   - Docker Install: **docker restart hip-rest**

Chapter 9. Enabling Unica Link to run on HTTP protocol for Windows and Linux Installation

Enabling Unica Link for Microsoft Windows

About this task

This documentation describes how to enable the Unica Link to run on `HTTP` protocol before running the link install command for the Microsoft windows installation.

To enable the Unica Link to run link server and rest on http port 8080 and client on http port 4443. Modify below files and complete the following steps:

1. Modify `config.yaml`.

    ```
    set HIP_HTTP=true
    set HIP_HTTPS_PORT=443
    set HIP_HTTP_PORT=4443

    ::HIP server host location

    set HIP_SERVER_HOST=localhost:8080

    Modify config.yaml -

    authentication.server=http://localhost:8080tomcat:
      inbound:
    ```

2. Run install.bat command
3. Set the environment variable `HIP_SERVER_USE_HTTP= True`, and complete the following steps:
    1. Click **This PC**.
    2. Select properties and click **Advanced System Settings**.
    3. Under the **Advance** tab, click **Environment Variables** and add a system variable as mentioned in *Step 3*.
4. Run Start.bat command.
5. Check client log for the following message:

    ```
    <Link installation folder>\DesignServer\client\daemon\hcllinkwebui.out.log

    Server listening at http://0.0.0.0:4443
    Using server backend at url: http://localhost:8080
    ```

6. Launch the Link UI and log in.

Enabling Unica Link for Linux Native and Docker for Port 8080

About this task

This documentation describes how to enable the Unica Link to run on HTTP protocol before running the link install command for the Native and Docker installation.

To enable the Unica Link to run on `HTTP` 8080 protocol for the Native and Docker installation, complete the following steps:

1. Open the `config.yaml` file.
2. In client configuration section, for inbound and outbound connections, change the protocol properties from `https` to `http` as shown below in the following example:

```
client:
  # Settings for inbound connections to the client such as when accessing
  # via a web browser.
  inbound:
    # Protocol for inbound connections. Valid options are [http, https].
    # Previously: HIP_HTTP
    protocol: "http"
    # Port to listen on for inbound connections. This value must be >1000
    # to run the client process as a non-root user.
    # Previously: HIP_HTTPS_PORT, HIP_HTTP_PORT
    port: 4443
  # Settings for outbound connections from client
  outbound:
    # Connection from client to backend server
    server:
      # Protocol for connection. Valid options are [http, https].
      # Previously: HIP_SERVER_USE_HTTP
      protocol: "http"
```

3. In the Rest configuration section perform these steps:

   a. Change the settings in `authentication.enabled` as follows:

   ```
   authentication.enabled=true
   ```

   b. Change the server property from `https://localhost:8443` to `http://localhost:8080` as shown in the following example:

   ```
   rest:
       authentication:
         # Specify if inbound API calls should be authenticated.
         # Previously: authentication.enabled
         enabled: true
         # The authentication server MUST point to the Server instance configured
         # for this installation. The format is {protocol}://{host}:{port}
         # [NATIVE] Default: 'https://localhost:{.server.inbound.ports.https}'
         # [CONTAINERIZED] Default: 'https://hip-server:8443'
         # Previously: authentication.server
         server: "http://localhost:8080"
   ```

4. In the tomcat configuration section, the default configuration for `http` port is `8080` and `https` port is `8443`. Any mismatch found in default configuration will require changing ports as shown in the following example:

```
tomcat:
  inbound:
    # [NATIVE-ONLY] Ports to listen on for connections. Neither protocol can be
    # disabled directly from this configuration file. To do so, complete
    # installation first then go to the Tomcat configuration files for advanced
    # options.
    # To set ports for docker installations use the .rest.inbound.ports and
    # .server.inbound.ports to configure on a per-container basis. Native
```

```
    # installations share the same Tomcat instance between rest and server.
    ports:
      http: 8080
      https: 8443
      shutdown: 8005"
```

5. Save the `config.yaml` file.
6. Execute ./Link install and ./Link start.

Note: The Link Runtime server should also use `HTTP` URL.

Enabling Unica Link for Linux Native for Port 8443

About this task

To enable the Unica Link to run on HTTP 8443 protocol for the Native installation, complete the following steps:

1. Open the `config.yaml` file.
2. In client configuration section, for inbound and outbound connections, change the protocol properties from `https` to `http` as shown below in the following example:

```

client:
  # Settings for inbound connections to the client such as when accessing
  # via a web browser.
  inbound:
    # Protocol for inbound connections. Valid options are [http, https].
    # Previously: HIP_HTTP
    protocol: "http"
    # Port to listen on for inbound connections. This value must be >1000
    # to run the client process as a non-root user.
    # Previously: HIP_HTTPS_PORT, HIP_HTTP_PORT
    port: 4443
  # Settings for outbound connections from client
  outbound:
    # Connection from client to backend server
    server:
      # Protocol for connection. Valid options are [http, https].
      # Previously: HIP_SERVER_USE_HTTP
      protocol: "http"
```

3. In the Rest configuration section perform these steps:

   a. Change the settings in `authentication.enabled` as follows:

   ```
   authentication.enabled=true
   ```

   b. Change the server property from `https://localhost:8443` to `http://localhost:8443` as shown in the following example:

   ```

   rest:
       authentication:
         # Specify if inbound API calls should be authenticated.
         # Previously: authentication.enabled
         enabled: true
   ```

```
      # The authentication server MUST point to the Server instance configured
      # for this installation. The format is {protocol}://{host}:{port}
      # [NATIVE] Default: 'https://localhost:{.server.inbound.ports.https}'
      # [CONTAINERIZED] Default: 'https://hip-server:8443'
      # Previously: authentication.server
      server: "http://localhost:8443"
```

4. In the tomcat configuration section, change the `http` port to `8443` and `https` port to `8444` as shown in the following example:

```

tomcat:
  inbound:
    # [NATIVE-ONLY] Ports to listen on for connections. Neither protocol can be
    # disabled directly from this configuration file. To do so, complete
    # installation first then go to the Tomcat configuration files for advanced
    # options.
    # To set ports for docker installations use the .rest.inbound.ports and
    # .server.inbound.ports to configure on a per-container basis. Native
    # installations share the same Tomcat instance between rest and server.
    ports:
      http: 8443
      https: 8444
      shutdown: 8005"
```

5. Save the `config.yaml` file.
6. Execute ./Link install and ./Link start.

Note: The Link Runtime server should also use `HTTP` URL.

Chapter 10. Encrypting the password in the properties file

To enhance security and safeguard sensitive information, encrypt the password stored in the properties file adding a layer of protection to the data.

About this task

To encrypt passwords in the properties file, complete the following steps:

1. In the Link installation directory, locate the `encrypt_password.sh` script.
2. Open the properties file and use the following syntax to encrypt the password:

   ```
   <Field Name>=plain:<TEXT/PASSWORD>
   ```

   Note: For the strings to be encrypted, add the prefix `plain:`.

   Example:

   ```
   password=plain:password
   ```
3. Run the encryption script ./encrypt_password.sh <Properties File Path>.

   Note: Replace *<Properties File Path>* with the absolute path to the properties file you want to encrypt the password in.

   For example, if the properties file is available at `/opt/hipmodules/journey.properties`, the command would be as follows:

   ```
   ./encrypt_password.sh /opt/hipmodules/journey.properties
   ```

4. After executing the script, open the properties file. You will see that the encrypted string, or text, has the prefix `encrypted`. For example:

   ```
   password=encrypted<Encrypted Sting>
   ```
5. Restart Link.

Chapter 11. Encrypting the `config.yaml` properties

This section describes how to encrypt the `config.yaml` properties for Linux -A and Windows.

Encrypting the config.yaml properties for Linux-A

Use these commands to encrypt config.yaml properties for Linux -A:

- **`Link encryptproperty`** `<property> <value>[<MKF key id>]`

  If the key `[<MKF key id>]` is not specified, encrypt the `<value>` using the default key from the `<DTXHOME>/config/configvar.mkf`. If the file path for `MKF` is not changed, where `[<MKF key id>]` is an optional parameter. See `masterKeyFile` property for `MKF` file path in the server section of `config.yaml`.

  If you want to encrypt the password for mongo which is under `auth` section, the command:

  ```
  ./Link encryptproperty /mongo/auth/password <any password>
  ```

  It updates the password property with the encrypted value of the password.

- **`Link encryptproperty`** `-inplace <property>[<MKF key id>]`

  If the key `[<MKF key id>]` is not specified, encrypt the `<value>` using the default key from the `<DTXHOME>/config/configvar.mkf`. If the file path for `MKF` is not changed, where `[<MKF key id>]` is an optional parameter. See `masterKeyFile` property for `MKF` file path in the server section of `config.yaml`.

  If the `-inplace` command is specified, the value argument is not required, the current value of the `<property>` will be encrypted.

  ```
  ./Link encryptproperty /mongo/auth/password
  ```

  The current value of the password property will be encrypted.

Encrypting the config.yaml properties for Windows

If you want to encrypt the password for mongo which is under `auth` section, run this command from `<Link Installation location>\DesignServer`.

```
C:\HCL\Link_1.1.5\DesignServer>enctyptproperty.bat - inplace/mongo/auth/password
```

Chapter 12. Verifying the `config.yaml` file

This section describes how to verify the validity of your `config.yaml` file after making changes in it. When editing the `config.yaml` file, ensure that the modifications you make are valid. To verify the validity of your `config.yaml` file after making changes, perform one of the following procedures based on your setup:

Verifying the `config.yaml` file in Microsoft Windows

1. Go to Link installation folder, create a `config` folder and copy your edited `config.yaml` file inside it.
2. Open command line, run the following command:

   ```
   flowcmdserver.exe -v
   ```

   Result

   This command will check the configuration file for any errors and provide feedback on its validity.
3. If the `config.yaml` file is valid, you should see an output similar to the following:

   ```
   HCL Link
   **VERSION**
   (c) Copyright IBM Corp. 2006, 2016 All rights reserved. (c) Copyright HCL Technologies Ltd. 2017, 2023
    All rights reserved
   Configuration loaded from <Program Data Folder>\config\config.yaml file
   ```

   This indicates that your configuration file is valid and successfully loaded.
4. If there are any issues with the `config.yaml` file, you will receive an error message similar to the following:

   ```
   HCL Link
   **VERSION**
   (c) Copyright IBM Corp. 2006, 2016 All rights reserved. (c) Copyright HCL Technologies Ltd. 2017, 2023
    All rights reserved.
   Failed to load yaml file. illegal map value line=360 collumn=8 Configuration loaded from <Program Data
    Folder>\config\config.yaml file
   ```

   This error message will provide details about the specific problem in the configuration file.

   Note: Make sure to carefully review any error messages and correct the issues in your `config.yaml` file accordingly. This step is essential to ensure the proper functioning of your system after making changes.

Verifying the `config.yaml` file in Linux Native

1. Go to Link installation folder and create a `config` folder and copy your edited `config.yaml` file inside it.
2. Open command line, run the following command:

   ```
   flowcmdserver.exe -v
   ```

   Result

   If you receive following error message, then perform Step 3 through Step 6.

   ```
   ./bin/flowcmdserver: error while loading shared libraries: libdstxpi.so: cannot open shared object file:
    No such file or directory.
   ```
3. Export LD_LIBRARY_PATH=`<Link Installation Folder>/tomcat-context/install/libs`.

4. Export DTX_DATA_DIR=`<Link Installation Folder>/tomcat-context/install`.
5. Go to `<Link_installation_dir>/bin` and open command line.
6. Run the following command:

   ```
   ./flowcmdserver -v
   ```

   This command will check the configuration file for any errors and provide feedback on its validity.
7. If the `config.yaml` file is valid, you should see an output similar to the following:

   ```
   HCL Link*****(1301)(c) Copyright IBM Corp. 2006, 2016 All rights reserved. (c) Copyright HCL
    Technologies Ltd. 2017, 2023 All rights reserved.Configuration loaded from <Link installation
    dir>/config/config.yaml file
   ```

   This indicates that your configuration file is valid and successfully loaded.
8. If there are any issues with the `config.yaml` file, you will receive an error message similar to the following:

   ```
   HCL Link******(1301)(c) Copyright IBM Corp. 2006, 2016 All rights reserved. (c) Copyright HCL
    Technologies Ltd. 2017, 2023 All rights reserved.Failed to load yaml file. illegal map value line=360
    collumn=8Configuration file is <Link installation dir//config/config.yaml
   ```

   This error message will provide details about the specific problem in the configuration file.

Note: Make sure to carefully review any error messages and correct the issues in your `config.yaml` file accordingly. This step is essential to ensure the proper functioning of your system after making changes.

Verifying the `config.yaml` file in Linux Docker

1. Go to `hip-server container/opt/hcl/hip/bin` for hip-server and `hip-rest container/opt/runtime/bin` folder for hip-rest.
2. Export DTX_DATA_DIR=`/opt/hcl/hip` and DTX_DATA_DIR=`/opt/runtime/bin`.
3. Open command line, run the following command:

   ```
   flowcmdserver -v
   ```

   Result

   This command will check the configuration file for any errors and provide feedback on its validity.
4. If the config.yaml file is valid, you should see an output similar to the following:
   - For hip-server:

     ```
     root@c60fa85561aa:/opt/hcl/hip/bin# ./flowcmdserver -v
     HCL Link
     10.9.9.9(1206)
     (c) Copyright IBM Corp. 2006, 2016 All rights reserved. (c) Copyright HCL Technologies Ltd. 2017,
      2023 All rights reserved.
     Configuration loaded from /opt/hcl/hip//config/config.yaml file
     ```
   - For hip-rest:

     ```
     root@c60fa85561aa:/opt/runtime/bin# ./flowcmdserver -v
     HCL Link
     10.9.9.9(1206)
     (c) Copyright IBM Corp. 2006, 2016 All rights reserved. (c) Copyright HCL Technologies Ltd. 2017,
      2023 All rights reserved.
     Configuration loaded from /opt/runtime//config/config.yaml file
     ```

This indicates that your configuration file is valid and successfully loaded.

5. If there are any issues with the `config.yaml` file, you will receive an error message similar to the following:
    - For hip-server:

      ```
      root@018b758fe98d:/opt/hcl/hip/bin# ./flowcmdserver -v
      HCL Link
      10.9.9.9(1206)
      (c) Copyright IBM Corp. 2006, 2016 All rights reserved. (c) Copyright HCL Technologies Ltd. 2017,
       2023 All rights reserved.
      Failed to load yaml file. bad file: /config/config.yaml line=0 collumn=0
      Configuration file is /config/config.yaml
      ```
    - For hip-server:

      ```
      root@018b758fe98d:/opt/runtime/bin# ./flowcmdserver -v
      HCL Link
      10.9.9.9(1206)
      (c) Copyright IBM Corp. 2006, 2016 All rights reserved. (c) Copyright HCL Technologies Ltd. 2017,
       2023 All rights reserved.
      Failed to load yaml file. bad file: /config/config.yaml line=0 collumn=0
      Configuration file is /config/config.yaml
      ```

   This error message will provide details about the specific problem in the configuration file.

Note: Make sure to carefully review any error messages and correct the issues in your `config.yaml` file accordingly. This step is essential to ensure the proper functioning of your system after making changes.

Chapter 13. Configuring SSL Certificates for Link

After installing Link, setup the custom SSL certificates for the Link web UI and the Link runtime application server.

Note: For the Link web UI to use the certificate, ensure that /client/inbound/protocol is set to https and it is the default setting.

Create the SSL certificates or use the SSL certificates from a third-party provider (example GoDaddy Inc). PEM is the preferred format for the certificates. If the certificates are not of the PEM format, convert the certificates to PEM format.

Split the SSL certificates into three files. Each file must contain one of the following SSL certificate components:

- Private key
- Server certificate key
- Certificate authority keys

The following table lists the GoDaddy-provided certificates in their corresponding files:

File	Description

Private key file	The file format is PEM and the extension of the file is KEY.
	The content of the file starts and ends with the following lines:
	`------BEGIN PRIVATE KEY------`
	`…`
	`-----END PRIVATE KEY-----`

Server certificate key file	The file format is PEM and the extension of the file is KEY.
	The content of the file starts and ends with the following lines:
	`------BEGIN CERTIFICATE------`
	`…`
	`-----END CERTIFICATE-----`

Certificate authority (CA) keys	You need these keys to sign off the private keys and public keys. Bundle the keys into a single file or multiple files. The files are not required if the certificates are self-signed.
	The CA certificates start and stop with the following lines:
	`------BEGIN CERTIFICATE------`
	`…`
	`-----END CERTIFICATE-----`

Configuring SSL certificates for Link web UI

To configure SSL certificates for Link web UI, complete the following steps:

1. Prepare the SSL certificate files.
2. Within the Link client installation folder, copy the certificate files to the `ssl-certificates/public-certificates/` folder. The relative path of the installation root folder for different platforms are as follows:

Microsoft Windows	The client installation folder is `DesignServer/client`.
Linux (Native)	The client installation folder is `node-context/install`.

3. Setup the config.yaml "/client/inbound/ssl" section to point to the certificate files copied in Step 2.

   Example:

```
key: "ssl-certificates/public-certificates/server.key"
cert: "ssl-certificates/public-certificates/server.crt"
ca: "
 ssl-certificates/public-certificates/gd1.crt,ssl-certificates/public-certificates/gd2.crt,ssl-certifica
tes/public-certificates/gd3.crt "
```

4. Restart the Link application. If your user profile has all the appropriate privileges, open a shell and change the working directory to the installation root.

Steps for configuring SSL Certificates in Linux Docker

About this task

To configure SSL certificates for Linux Docker, complete the following steps:

1. Prepare the SSL certificate files.
2. Within the Link client installation folder, copy the certificate files to the `ssl-certificates/public-certificates/` folder.
3. Stop Link.
4. Change the following properties in the client section of `config.yaml` and save the file.

```
key: "ssl-certificates/public-certificates/server.key"
cert: "ssl-certificates/public-certificates/server.crt"
ca:
 "ssl-certificates/public-certificates/gd1.crt,ssl-certificates/public-certificates/gd2.crt,ssl-certific
ates/public-certificates/gd3.crt"
```

5. Start Link.
6. Ignore Certificate Error.

   Note: The **hip-client** container might fail to start due to a missing certificate in the specified path.

7. Copy the certificates from the local folder to the HIP client container at `/usr/src/app` using docker cp command.

```
docker cp ssl-certificates hip-client:/usr/src/app
```

   Replace **ssl-certificates** with the path to your local folder containing certificates.

8. Restart Hip Client.

Configuring SSL certificates for the Link Apache Tomcat application server

If you implement Link on a setup containing Apache Tomcat sever, the implementation works as a Tomcat application.

About this task

Apache Tomcat uses a Java keystore to store SSL certificates. To import certificates into the Link application server Java keystore, use the `openssl` utility. For more details related to `openssl` installation, see https://www.openssl.org/.

To configure SSL certificates for the Link Apache Tomcat application server, complete the following steps:

1. To export the SSL certificates into pkcs12 format, run the command open ssl pkcs12. The system prompts you to provide the password phrase (for example <*my passphrase*>) to protect the generated certificate file. The `-CAfile` command option supplies the CA bundle.
   If GoDaddy provides multiple CA PEM files, bundle all CA files (concatenated as text files) into single file named cabundle.crt:

```
gd1.crt>> cabundle.crt
gd2.crt>> cabundle.crt
gd3.crt>> cabundle.crt
openssl pkcs12 -export -in sever.crt -inkey server.key -out dtxtomcat.p12 -name dtxtomcat -CAfile
 cabundle.crt -caname root
```

2. Depending on the platform, access the keytool application from the provided location:

Microsoft Windows	`C:\HCL\Link_<version>\java\bin\keytool.exe`
Linux-based OS	`tomcat-context/install/java/bin/keytool`

   To generate java keystore, run the following command:

```
keytool.exe -importkeystore -deststorepass <my passphrase> -destkeypass <my passphrase> -destkeystore
 dtxtomcat.keystore -srckeystore dtxtomcat.p12 -srcstoretype PKCS12 -srcstorepass changeit -alias
 dtxtomcat keytool -v -list -keystore dtxtomcat.keystore
```

   Note: The value of <*my passphrase*> must match the passphrase value set for the configuration property `/tomcat/keystore/password`.

3. For Microsoft Windows and Linux-based operating systems, copy the generated Java keystore to the following location:

```
<Link installation folder>/restapi/tomcat/server/dtxtomcat.keystore
```

   where <Link installation folder> is:

   - For Microsoft Windows - C:\HCL\Link_<version>
   - For Linux-based operating systems - tomcat-context/install

4. In case of Link installation on Docker environment, copy the keystore file from a local folder to the `hip-rest` container at `/usr/local/tomcat` using the docker cp command. The value of <*my pass phrase*> must match the passphrase value in the `/usr/local/tomcat/conf/server.xml` file within the container.
5. Restart the Link application. If your user profile has all the appropriate privileges, open a shell and change the working directory to the installation root.

    a. In case of Microsoft Windows, execute the following commands:

    ```
    DesignServer\stop.bat
    DesignServer\start.bat
    ```

    b. In case of Linux-based systems, execute the following commands:

    ```
    ./Link stop
    ./Link start
    ```

Configuring SSL certificates for Link using command line tool

To configure the SSL certificates for Link using command line tool to import certificate in PEM format, complete the following steps:

1. Prepare the SSL PEM certificate files.
2. Import SSL PEM certificates into Link environment using `importcertificate` tool.

    Note:

    - If the SSL certificates are in PEM format, then certificates can be imported into Link environment using importcertificate tool.
    - PEM is the preferred format for the certificates. If the certificates are not of the PEM format, convert the certificates to PEM format.

3. Split the prepared SSL certificates into three PEM files. Each file must contain one of the following SSL certificate components:
    - Public Server certificate PEM file
    - Private certificate key PEM file
    - Certificate authority PEM file
4. Depending on the platform, after installing Link, open a command line with administrative privileges, then change directory to:

Microsoft Windows	cd to the DesignServer installation folder, for example: `cd C:\HCL\Link_1.1.6\DesignServer`
Linux-based OS	cd to Link installation folder, for example: `cd /home/myuser/Link_1.1.6`

	Note: Log in to the system as the same user, that have Link installed.

5. Copy the SSL PEM files on the computer where Link is installed, for example:

Microsoft Windows	**Private key**:
	`c:\ssl-certificates\self-signed-test\server-key.pem`
	Public certificate:
	`c:\ssl-certificates\self-signed-test\server-crt.pem`
	Certificate authority:
	`c:\ssl-certificates\self-signed-test\self_ca_root.pem`

Linux-based OS	**Private key**:
	`/home/myuser/ssl-certificates/self-signed-test/server-key.pem`
	Public certificate:
	`/home/myuser/ssl-certificates/self-signed-test/server-crt.pem`
	Certificate authority:
	`/home/myuser/ssl-certificates/self-signed-test/self_ca_root.pem`

6. For Microsoft Windows and Linux-based operating systems, run the command from the following folder location:
   - Command for Microsoft Windows from `C:\HCL\Link_1.1.6\DesignServer` folder:

```
.\importcertificates.bat c:\ssl-certificates\self-signed-test\server-key.pem
 c:\ssl-certificates\self-signed-test\server-crt.pem
 c:\ssl-certificates\self-signed-test\self_ca_root.pem
```

   - Command for Linux-based operating systems from `/home/myuser/Link_1.1.6` folder:

```
./Link
 importcertificates /home/myuser/ssl-certificates/self-signed-test/server-key.pem /
home/myuser/ssl-certificates/self-signed-test/server-crt.pem /
home/myuser/ssl-certificates/self-signed-test/self_ca_root.pem
```

7. Restart the Link application. If your user profile has all the appropriate privileges, open a shell, and change the working directory to the installation root.

- In case of Microsoft Windows, execute the following commands:

```
DesignServer\stop.bat
DesignServer\start.bat
```

- In case of Linux-based systems, execute the following commands:

```
./Link stop
./Link start
```

Configuring SSL certificates for Link Java WEB Client

This documentation explains how to configure an SSL certificate for the Link Java WEB client.

About this task

Note:

- Please note that if you encounter the following error while configuring the HCH runtime server, you will need to import the Link server certificate into the JRE keystore or truststore. This problem occurs when java web clients try to access web servers that have server certificates signed by unknown authority.

```
com.hcl.tx.server.impl.ServiceException: An error was encountered when invoking
 an endpoint: javax.net.ssl.SSLHandshakeException: PKIX path building failed:
 sun.security.provider.certpath.SunCertPathBuilderException: unable to find valid certification
 path to requested target.
```

- Please note that if you encounter the following license-related SSL errors when accessing a UNICA platform running on HTTPS, you will need to import the Unica Link platform certificate into the JRE (Java Runtime Environment) keystore or truststore. This problem occurs when java web clients try to access web servers that have server certificates signed by unknown authority.

```
SEVERE [https-jsse-nio-8443-exec-5] com.hcl.hip.ws.resources.AuthFilter.checkLicense Invalid
 license, request GET https://localhost:8443//tx-server/rest/v1/tx/license_check//platform
 failed, this Link instance is not licensed to execute the request.
```

```
SEVERE [https-jsse-nio-8443-exec-5] com.hcl.hip.ws.resources.AuthFilter.checkLicense This Link
 instance is not licensed to execute the REST requests. Error: Internal Server Error
```

To configure SSL certificates for the Link Java WEB client, complete thefollowing steps:

1. Collect the WEB server certificate by executing the following command:

For Link Server	openssl s_client -connect <Link server host-name>:<PORT>
For Unica Platform	openssl s_client -connect <UNICA platform host-name>:<PORT>

2. Copy the following content from the generated output after running the earlier command and save the certificate in either `.cer` file format:

```
------BEGIN CERTIFICATE------

..........

------END CERTIFICATE------
```

3. Import the certificate from the file (created in step 2) into JRE (Java Runtime Environment) keystore using the following command:

   For Windows

   ```
   sudo keytool -import -trustcacerts -keystore <Link installation folder>/java/lib/security/cacerts
    -storepass changeit -noprompt -alias <Alias> -file <Cert File path>
   ```

   For Linux Native

   ```
   sudo keytool -import -trustcacerts -keystore <Link installation
    folder>/tomcat-context/install/java/lib/security/cacerts -storepass changeit -noprompt -alias <Alias>
    -file <Cert File path
   ```

4. Restart Link.

Chapter 14. JWT Configuration for Connectors

To configure the JWT (JSON Web Token) there are two options for a customer regarding certificate.

JWT Configuration Using the Server Certificate and Private Key

Unica Link can run some steps to create pkcs12/p12 bundle to import these to JKS/java keystore. Unica Link can set up the key alias name, password, and keystore password.

Prerequisites

Ensure that these system requirements are met before you configure the JWT:

- Certificate file
- Private key
- Java keytool command is installed
- OpenSSL

To configure JWT for connectors, follow these steps:

1. To generate `pkcs12` file from certificate and private key, run the following command using OpenSSL:

   ```
   openssl pkcs12 -export -in certificate.pem -inkey privatekey.key -name <ALIAS_NAME>
    -outoutpkcs12cert.p12
   ```
2. Enter the password for the `pkcs12` file.
3. To generate a `JKS` (java keystore) file from the generated `pkcs12` file, run the following command using Java keytool.

   ```
   keytool -v -importkeystore -srckeystore outpkcs12cert.p12 -srcstoretype PKCS12 -destkeystore
    sfkeystore.jks -deststoretype JKS
   ```
4. Enter the **Source password** to generate the `pkcs12` file.
5. Enter the **Destination password** to generate the `JKS` file.

Note: `keytool` is provided with any standard JDK/JRE distributions, which can be found under the `%JAVA_HOME%/bin` folder.

JWT Configuration Using P12/pkcs12 File

Customer can create `P12/pkcs12` file and share the p12 file, key alias name, and password. Unica Link support can import it to `JKS`.

To configure JWT for connectors, follow these steps:

1. To generate a `JKS` file from this `pkcs12` file. Run the following command using Java keytool.

   ```
   keytool -v -importkeystore -srckeystore outpkcs12cert.p12 -srcstoretype PKCS12 -destkeystore
    sfkeystore.jks -deststoretype JKS
   ```
2. Enter the **Source password** to generate the `pkcs12` file.
3. Enter the **Destination password** to generate the `JKS` file.

Note: `keytool` is provided with any standard JDK/JRE distributions, which can be found under the `%JAVA_HOME%/bin` folder.

`config.yaml` File Changes for JWT Configuration

Put the generated JKS file (`sfkeystore.jks`) in your local machine where HCL Link is installed. For example: `/opt/hipfiles/sfkeystore.jks` and configure the same in `config.yaml` file.

Add the following fields in the `config.yaml` file under the server section:

1. `configuration` of keystore for JWT oauth2 authentication in salesforce connectors.

   `connectors:`

   `jks:`
2. `filepath` is the path of generated JKS file. For example: `/opt/hipfiles/sfkeystore.jks`.

   `filepath: <JKS_FILE_PATH>`
3. `password` field is the password entered while creating `pkcs12` file.

   `password: <PKCS12_PASSWORD>`
4. `privateKeyAlias` is the alias name for private key entered while creating `pkcs12` file.

   `privateKeyAlias: <ALIAS_NAME>`
5. `privateKeyPassword` field is the password entered while creating `JKS` file.

   `privateKeyPassword: <JKS_PASSWORD>`

Chapter 15. Troubleshooting

Troubleshooting: Installation

Most installation issues will include an error message, either printed to your terminal, or written to `<command>.log`. If you do not see an explanation on the screen, check the log.

For any errors relating to start up of a component, you will likely find more information in the component’s log. Use the logs command to view it.

Error 1

Link Client logs show `ENOSPC: System limit for number of file watchers reached`.

Solution 1

Increase the inotify max_user_watches property.

```
echo fs.inotify.max_user_watches=524288 | sudo tee -a /etc/sysctl.conf
```

```
sudo sysctl -p
```

Error 2

Link shows the following error message while installing Link on Redhat and CentOS:

```
sed: error while loading shared libraries: libsnl.so.1: cannot open hared object file: No such
 file or directory
sed: error while loading shared libraries: libsnl.so.1: cannot open hared object file: No such
 file or directory
sed: error while loading shared libraries: libsnl.so.1: cannot open hared object file: No such
 file or directory
```

Solution 2

To install Link on Redhat and CentOS, complete the following steps:

1. Run the command sudo yum install libnsl.
2. Run the command ./Link install.

Licensing

You can check to see if your Runtime REST API is licensing by executing this request through Swagger.

1. In your browser, navigate to this location: `https://<RUNTIME_REST_API_IP>:<PORT>/hip-rest/api-docs?url=openapi.json#/V2_Map_and_Flow_Deployment_API/listPackages`
2. Click on the 'unlocked /lock icon in the upper right corner of the page to specify a username and password for your request. Use the credentials that work for Link. Click **Authorize** then Close. The lock icon will show locked.
3. Click on **Try it out**.
4. Click on Execute.
5. Check the Response Code:

- **200**=Valid license
- **401 = with Response Body** “This Link instance is not licensed to execute the REST requests.”

  Invalid license. Check **<Link_Install>/tomcat-context/apps/platform.properties**. Check entitlements through the Platform.
- **401 with empty Response Body**.- Invalid credentials. Repeat step

Runtime REST API server missing in Link

The Runtime REST API server is configured during connector installation.

1. Open Link in your browser. The default configuration will be available with `https://<LINK_IP>:443`.
2. Open the Servers page from the link home page.
3. Click the **Create New Server** button in the upper right corner of the page to add a new server.
4. Configure the server details:
   - **Name**: Link Runtime
   - **Type**: Web
   - **Base URL**: `https://localhost:8443/hip-rest` (native) or `https://hip-rest:8443/hiprest` (docker).
5. Test the connection by clicking the **Test** button. Check the server log if the connection fails.