HCLSoftware

Unica Journey V12.1.6 Installation Guide

Contents

Chapter 1. Installation overview

An installation of HCL Unica products is complete when you install, configure, and deploy the HCL Unica products . The Installation Guide provides detailed information about installing, configuring, and deploying the products.

How the installers work

You must use the suite installer and the product installer when you install or upgrade any Unica product. For example, for installing Unica Journey, you must use the Unica suite installer and the Unica Journey installer.

Make sure that you use the following guidelines before you use the Unica suite installer and the product installer:

- The Unica installer and the product installer must be in the same directory on the computer where you want to install the product. When multiple versions of a product installer are present in the directory with the Unica installer, the Unica installer always shows the latest version of the product on the Unica products screen in the installation wizard.
- If you are planning to install a patch immediately after you install a Unica product, make sure that the patch installer is in the same directory as that of the suite and product installers.
- The default top-level directory for Unica installations is `/HCL/Unica` for UNIX™ or `C:\HCL\Unica` for Windows™. However, you can change the directory during installation.

Modes of installation

The Unica suite installer can run in one of the following modes: GUI mode, Console mode, or Silent mode (also called the unattended mode). Select a mode that suits your requirements when you install Unica Journey.

GUI mode

Use the GUI mode for Windows™ or UNIX™ to install Unica Journey by using the graphical user interface.

Console mode

Use the console mode to install Unica Journey by using the command line window.

> **Note:** To display the Installer screens correctly in console mode, configure your terminal software to support UTF-8 character encoding. Other character encoding, such as ANSI, will not render the text correctly, and some information will not be readable.

Chapter 2. Planning the Unica Journey installation

When you plan your Unica Journey, you must ensure that you have set up your system correctly, and that you have configured your environment to deal with any failures.

Prerequisites

Before you install or upgrade any Unica Journey product, you must ensure that your computer complies with all of the prerequisite software and hardware.

System requirements

For information about system requirements, see the *Recommended Software Environments and Minimum System Requirements guide*

Network domain requirements

The Unica products that are installed as a suite must be installed on the same network domain to comply with the browser restrictions that are designed to limit the security risks that can occur with cross-site scripting.

> **Note:** Unica Journey and Unica Link installation must be done with domain name specified for the application URLs.

JVM requirements

Unica applications within a suite must be deployed on a dedicated Java™ virtual machine (JVM). Unica products customize the JVM that is used by the web application server.

Knowledge requirements

To install Unica products, you must have a thorough knowledge of the environment in which the products are installed. This knowledge includes knowledge about operating systems, databases, Kafka, and web application servers.

Internet browser settings

Make sure that your internet browser complies with the following settings:

- The browser must not cache web pages.
- The browser must not block pop-up windows.

Access permissions

Verify that you have the following network permissions to complete the installation tasks:

- Administration access for all necessary databases

  > **Note:** Administrator must have `CREATE`, `SELECT`, `INSERT`, `UPDATE`, `DELETE`, and `DROP` rights for both tables and views.

- Read and write access to the relevant directory and sub-directories for the operating system account that you use to run the web application server and Unica components.

- Write permission for all files that you must edit.
- Write permission for all directories where you must save a file, such as the installation directory and backup directory if you are upgrading.
- Appropriate read, write, and execute permissions to run the installer.

Verify that you have the administrative password for your web application server.

For UNIX™, all installer files for products must have full permissions, for example, `rwxr-xr-x`.

The following additional permissions are necessary for UNIX™:

- The user account that installs Unica Journey and Unica Platform must be a member of the same group as the Unica Journey users. This user account must have a valid home directory and have write permissions for that directory.
- All installer files for HCL Unica products must have full permissions, for example, `rwxr-xr-x`.

Points to consider before you install Unica Journey

For Unica Journey installation you are required to consider the following points.

JAVA_HOME environment variable

If a JAVA_HOME environment variable is defined on the computer where you install a Unica product, verify that the variable points to a supported version of JRE. For information about system requirements, see the *Recommended Software Environments and Minimum System Requirements* guide.

If the JAVA_HOME environment variable points to an incorrect JRE, you must clear the JAVA_HOME variable before you run the Unica installers.

You can clear the JAVA_HOME environment variable by using one of the following methods:

- Windows™: In a command window, enter set JAVA_HOME= (leave empty) and press **Enter**.
- UNIX™: In the terminal, enter export JAVA_HOME= (leave empty) and press **Enter**.

You can clear the JAVA_HOME environment variable by running the following command in the terminal:

export JAVA_HOME= (leave empty)

The Unica installer installs a JRE in the top-level directory for the Unica installation. Individual Unica application installers do not install a JRE. Instead, they point to the location of the JRE that is installed by the Unica installer. You can reset the environment variable after all installations are complete.

For more information about the supported JRE, see the *Recommended Software Environments and Minimum System Requirements* guide.

Unica Platform requirements

User must install support base Unica Platform Version before installing and upgrading Journey. For each group of products that work together, you must install or upgrade Unica Platform only once. Each product installer checks whether the required products are installed. If your product or version is not registered with Unica Platform, a message prompts you to install or

upgrade Unica Platform before you proceed with your installation. Unica Platform must be deployed and running before you can set any properties on the **Settings > Configuration** page.

Platform and Journey can be installed on different servers. In such case, when Platform is installed on a different server then make sure that Journey application should have access to Platform URL. Journey host should be able to communicate with Platform host through the unica application port.

Table 1. Journey Supported Installation Paths

Base Journey version	Upgrade path	Tasks to be performed
Unica Journey 12.1.0 or 12.1.0.x (with system tables on Oracle, MS SQL Server, OneDB, MariaDB) or 12.1.4 (with system tables on DB2)	In place upgrade to Unica Journey 12.1.6	1. Upgrade Unica Marketing Platform to 12.1.6 2. Run upgrade Unica Journey to 12.1.6 installer 3. Configure Journey application 4. Deploy Journey application 5. Run Journey application
Clean Installing Journey on existing Unica environment when you have system tables as OneDB, MariaDB, DB2 and SQL Server databases	In place upgrade to Unica Journey 12.1.6	1. Upgrade Unica Marketing Platform and required Unica products to 12.1.6 except Journey. 2. Run clean installer of Unica Journey 12.1.6 3. Configure Journey application 4. Deploy Journey application 5. Run Journey application

Note:

1. If users do not have Journey already installed on existing Unica environment with Oracle database, then they need to install Journey version 12.1.0 and then upgrade Unica Journey to 12.1.6
2. If you do not have Journey already installed on existing Unica environment with either MS SQL Server, OneDB or MariaDB and DB2 database then you can install Journey version 12.1.6 directly as clean install.

Database pre-requisites

User need to provide Journey System DB privileges to Reports DB user and vice versa for the performance tab to work efficiently. For Journey installation we recommend to use the separate schema for Journey system tables and Journey report tables.

📝 **Note:** System DB and Report DB should be in different schema DB for all the DBs.

For MariaDB use the following commands:

grant all privileges on {Journey_SystemDB}.* to '{Journey_Reports_User}'@'%' identified by '{Journey_Reports_User_Password}';

GRANT ALL ON {Journey_SystemDB}.* TO '{Journey_Reports_User}'@'%';

grant all privileges on {Journey_ReportsDB}.* to '{Journey_SystemDB_User}'@'%' identified by '{Journey_SystemDB_User_Password}';

GRANT ALL ON {Journey_ReportsDB}.* TO '{Journey_SystemDB_User}'@'%';

For Oracle

For oracle database create a system user account and a report user for creating Report schema. The system user account must have the following rights:

- CREATE TABLES
- CREATE VIEWS (for reporting)
- CREATE SEQUENCE (Oracle only)
- CREATE INDICES
- ALTER TABLE
- INSERT
- UPDATE
- DELETE

📝 **Note:**

- Report user also have the above mentioned rights. In addition, report user have to grant permission to system user for accessing the Report schema table. Run the following command

  ```
  GRANT ALL PRIVILEGES TO (JOURNEY_SYSTEM_SCHEMA_USER_NAME);
  ```
- For improving journey performance with oracle database:

  Redo log size should be increased if DB is running with default redo log size. For production environment, it is advised to size redo log file in a way that not more than 5 log switches happen per hour for optimum performance.

  Please refer oracle documentation to check how to increase redo log size.

For DB2

Perform the following setup before starting the installer:

1. DB2 Server setting

    Run db2start

    Run db2set DB2_COMPATIBILITY_VECTOR=ORA

    Run db2set DB2_DEFERRED_PREPARE_SEMANTICS=YES

    Run db2stop

    Run db2start

2. Create two Schema/user in DB2
    a. journeydb
    b. reportdb

3. Grant below command to Journey system schema user DB and Journey Report DB user by DB2 system user

    GRANT:
    a. GRANT
       DBADM,CREATETAB,BINDADD,CONNECT,CREATE_NOT_FENCED_ROUTINE,IMPLICIT_SCHEMA,LOAD,CREATE_EXTERNAL_RC
       ON DATABASE TO USER <Journey schema user DB>
    b. GRANT
       DBADM,CREATETAB,BINDADD,CONNECT,CREATE_NOT_FENCED_ROUTINE,IMPLICIT_SCHEMA,LOAD,CREATE_EXTERNAL_RC
       ON DATABASE TO USER <Journey Report schema user DB>

    TABLESPACE:

    a. GRANT USE OF TABLESPACE <TABLESPACE_NAME> TO USER <Journey schema user DB> WITH GRANT
       OPTION
    b. GRANT USE OF TABLESPACE <TABLESPACE_NAME> TO USER <Journey Report schema user DB> WITH
       GRANT OPTION

For more details, see

Distributed Environment for Journey:

Journey engine files location needs to be shared on Journey engine and Web machine. If Journey engine is installed on multiple machines, then this file directory needs to be shared/ mounted on the same path across all machines.

Deployment Diagram

Please find below Unica applications deployment diagram including Unica Journey. Unica Journey needs to be installed on top of Unica Platform being used for other products in Unica suite.

Unica Journey Web and Engine components can be deployed on the same or different machines. Unica Journey product has below components:

1. Unica Journey Web
2. Unica Journey Engine
3. Kafka Instance being used for underlying communications. Kafka instance has kafka server and zookeeper

Currently Journey Web and Journey Engine are supported as standalone installations. Please reach out to HCL Support Team if you need to deploy them in cluster.
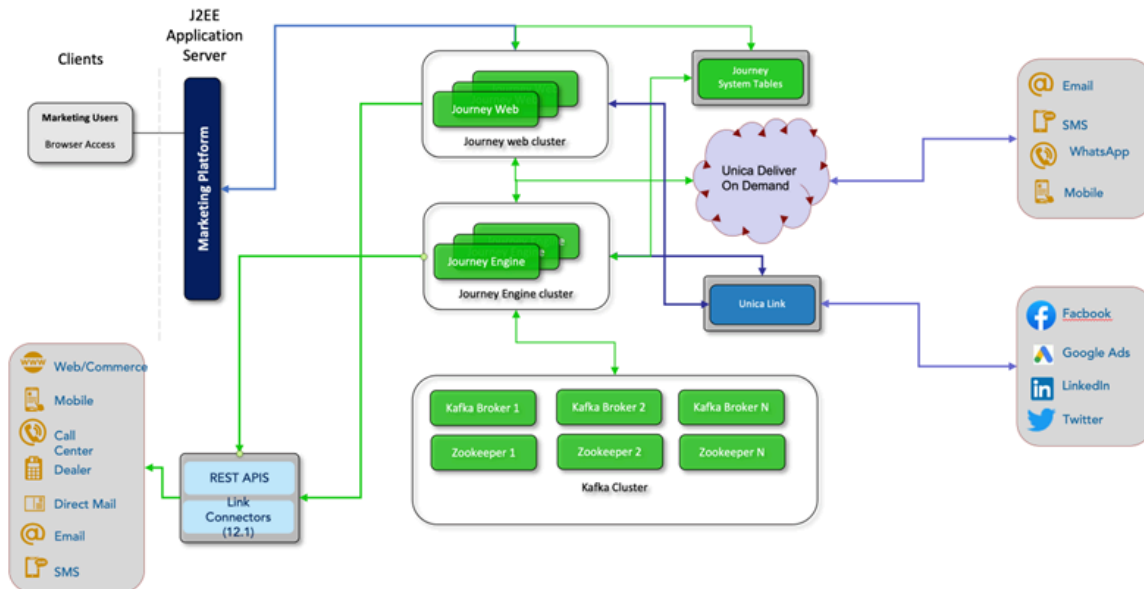
Unica Journey installation worksheet

Use the Unica Journey installation worksheet to gather information about the Unica Journey database and about other Unica products that are required for the installation of Unica Journey.

Use the following table to gather information about the empty database that was created for the Unica Journey system tables. The empty database that you set up for Unica Journey can have any name.

Table 2. Supported Database

Field	Notes®
Database type	
Database name	
Database account user name	
Database account password	
JNDI name	JourneyDS, JourneyReportDS
ODBC name	

Table 3. Information about the Kafka Instance

Fields	Notes
Kafka server host	
Kafka server port	
Kafka server certificate (if Kafka is SSL enabled)	
Kafka server - user id (if Kafka connection is SASL plaintext)	
Kafka server - user password (if Kafka connection is SASL plaintext)	

Oracle

- Database Driver: `oracle.jdbc.OracleDriver`
- Default port: 1521
- Driver class: `oracle.jdbc.OracleDriver`
- Driver URL: `"jdbc:oracle:thin:@<Host>:<Port>:<SID_NAME>"`

```xml
<?xml version="1.0"?>
<Context docBase="<Journeys_Install_Path>/Web/journey.war">
<Environment name="journey.web.home" value="<Journeys_Install_Path>/Web/" type="java.lang.String"/>
<Resource name="JourneyDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>" driverClassName="oracle.jdbc.OracleDriver"
url="jdbc:oracle:thin:@<Host>:<Port>:<SID_NAME>"/>
<Resource name="JourneyReportDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>" driverClassName="oracle.jdbc.OracleDriver"
url="jdbc:oracle:thin:@<Host>:<Port>:<SID_NAME>"/>
</Context>
```

SQL Server

- Database Driver: `com.microsoft.sqlserver.jdbc.SQLServerDriver`
- Default port: 1433
- Driver class: `com.microsoft.sqlserver.jdbc.SQLServerDriver`
- Driver URL: `jdbc:sqlserver://<your_db_host>\`
  `\<named_instance>:<your_db_port>;databaseName=<your_db_name>`
- Properties: Add `user=<your_db_user_name>`

```xml
<?xml version="1.0"?>
<Context docBase="<Journeys_Install_Path>/Web/journey.war">
<Environment name="journey.web.home" value="<Journeys_Install_Path>/Web/" type="java.lang.String"/>
```

```
<Resource name="JourneyDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>"
 driverClassName="com.microsoft.sqlserver.jdbc.SQLServerDriver"
url="jdbc:sqlserver://<your_db_host>\\<named_instance>:<your_db_port>;databaseName=<your_db_name>"/>
<Resource name="JourneyReportDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>"
 driverClassName="com.microsoft.sqlserver.jdbc.SQLServerDriver"
url="jdbc:sqlserver://<your_db_host>\\<named_instance>:<your_db_port>;databaseName=<your_db_name>"/>
</Context>
```

OneDB Database

- Database Driver: `com.informix.jdbc.IfxDriver`
- Default port: 9088 <User Defined Database Port>
- Driver class: `javax.sql.DataSource`
- Driver URL: `jdbc:Informix-sqli://host:port/database_name:informixserver=servername;`
- Properties: Add `user=<your_db_user_name>`

```
<?xml version="1.0"?>
<Context docBase="<Journeys_Install_Path>/Web/journey.war">
<Environment name="journey.web.home" value="<Journeys_Install_Path>/Web/" type="java.lang.String"/>
<Resource name="JourneyDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>" driverClassName="javax.sql.DataSource"
url="jdbc:Informix-sqli://host:port/<database_name>:informixserver=<servername>"/>
<Resource name="JourneyReportDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>" driverClassName="javax.sql.DataSource"
url="jdbc:Informix-sqli://host:port/<database_name>:informixserver=<servername>"/>
</Context>
```

MariaDB Database

- Database Driver: `org.mariadb.jdbc.Driver`
- Default port: 3306
- Driver class: org.mariadb.jdbc.Driver
- Driver URL: `="jdbc:mariadb://host:port/<DB_USER_NAME>"`
- Properties: Add `user=<your_db_user_name>`

```
<?xml version="1.0"?>
<Context docBase="<Journeys_Install_Path>/Web/journey.war">
<Environment name="journey.web.home" value="<Journeys_Install_Path>/Web/" type="java.lang.String"/>
<Resource name="JourneyDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>" driverClassName="org.mariadb.jdbc.Driver"
url="jdbc:mariadb://host:port/<DB_USER_NAME>"/>
```

```
<Resource name="JourneyReportDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>" driverClassName="org.mariadb.jdbc.Driver"
url="jdbc:mariadb://host:port/<DB_USER_NAME>"/>
</Context>
```

DB2

- Database Driver: `db2jcc4.jar`
- Default port: 50000
- Driver class: com.ibm.db2.jcc.DB2Driver
- Driver URL: `jdbc:db2://<your_db_host>:<your_db_port>/<your_db_name>`

Checklist for Unica Platform database

The installation wizards for each Unica product must be able to communicate with the Unica Platform system table database to register the product. Each time that you run the installer, you must enter the following database connection information for the Unica Platform system table database:

- JDBC connection URL
- Database host name
- Database port
- Database name or schema ID
- User name and password for the database account

Checklist for Unica Platform deployment on the web application server

Obtain the following information before deploying Unica Platform:

- **Protocol**: HTTP, or HTTPS if SSL is implemented in the web application server.
- **Host**: The name of the machine on which the Unica Platform will be deployed.
- **Port**: The port on which the web application server listens.
- **Domain name**: The company domain of each machine where HCL products are installed. For example, `example.com`. All HCL products must be installed in the same company domain, and you must enter the domain name in all lower case letters.

  If there is a mismatch in domain name entries, you might encounter problems when you attempt to use Unica Platform features or navigate among products. You can change the domain name after the products are deployed by logging in and changing values of the relevant configuration properties in the product navigation categories on the **Settings > Configuration** page.

Checklist for Unica Journey Installation

Obtain the following information to install Unica Journey's different components.

- hostname - The name of the system on which Journey web application will be installed.
- The port on which the application server listens. If you plan to implement SSL, obtain the SSL port.
- The network domain for your deployment system. For example, `mycompany.com`.

Installation order for Unica Journey

When you install Unica Journey, you must install them in a specific order.

The following table provides information about the order that you must follow when you install Unica Journey.

Table 4. Installation order for Unica Journey

Product	Install in this order
Unica Journey	1. Unica Platform 2. Unica Journey

Note: Unica Journey installs three components:

- Unica Journey Web application -- Unica Journey web application can be deployed in supported application server, like Apache Tomcat, IBM WebSphere, and Red Hat JBoss EAP.
- Unica Journey Engine: It does not require any deployment in application server, Journey engine can get started from command line / terminal as a standalone application.
- Apache Kafka: Kafka server and zookeeper gets installed together and can be started on command line or terminal. Unica Journey all three components can be installed on same or different machines.

Journey Oracle 12C Database Support

Journey Upgrade 12.1 > 12.1.0.3 on Oracle 12C (12.1.0.2)

If you want to upgrade Journey 12.1.0.3 on oracle 12C then first clean database schema for Journey Web and Journey report and run 12.1.0.3 scripts(Provided as a Hotfix) manually.

Journey Upgrade 12.1 > 12.1.0.4 on Oracle 12C (12.1.0.2)

If you want to upgrade Journey 12.1.0.4 on oracle 12C then first need to upgrade to version 12.1.0.3. Clean the database schema for Journey Web and Journey report and run 12.1.0.3 scripts (Provided as a Hotfix) manually and then run the 12.1.0 FP4 installer.

Note: Contact the Support team for the hotfix.

Journey Upgrade 12.1.0.4 > 12.1.1 on Oracle 12C (12.1.0.2)

If you want to upgrade Journey 12.1.1.0 on oracle 12C then first need to upgrade to version 12.1.0.3 from 12.1.0. Clean the database schema for Journey Web and Journey report and run 12.1.0.3. Scripts (Provided as a Hotfix) manually and then run the 12.1.0 FP4 installer. after that run the 12.1.1.0 Installer.

Limitation

Oracle 12C has a limitation that the object name should not be greater than 30 characters.

Note: You can also follow the Upgrade path for Unica Journey from version 12.1.1. For more information, see *Unica Journey Upgrade Guide*.

Chapter 3. Creating the Unica Journey Data Sources

You must create Unica Journey data sources before you can install Unica Journey. Complete the following steps to prepare the data sources for Unica Journey:

1. Create a database or a database schema for the Unica Journey and Journey report system tables.
   The following table provides information about vendor-specific guidelines for creating a database or a database schema for the Journey system tables.

   Table 5. Guidelines for creating data sources

Database vendor	Guideline
Oracle	Enable the auto commit feature for the environment open. See the Oracle documentation for instructions.
MariaDB	Lower_case_table_names is 1 so that table names are considered case insensitive. Set wait_timeout=<`Time in seconds that the server waits for a connection to become active before closing it. The session value is initialized when a thread starts up from either the global value, if the connection is non-interactive, or from the interactive_timeout value, if the connection is interactive.`> e.g. Set this to 25,92,000 (seconds) in case the set up can be inactive for 30days max_connections=<`The maximum number of simultaneous client connections.`>
SQL Server	Use either SQL Server authentication only, or both SQL Server and Windows™ authentication, because the Platform requires SQL Server authentication. If required, change the database configuration so that your database authentication includes SQL Server. Also be sure that TCP/IP is enabled in your SQL Server.

   > **Note:** If you plan to enable locales that use multi-byte characters (for example, Chinese, Korean, and Japanese), ensure that the database is created to support them.

2. Create a system user account.
   The system user account must have the following rights:
   - CREATE TABLES
   - CREATE VIEWS (for reporting)
   - CREATE SEQUENCE (Oracle only)

- CREATE INDICES
- ALTER TABLE
- INSERT
- UPDATE
- DELETE

3. Create ODBC or native connections.
4. Configure the web application server for your JDBC driver.
5. Create JDBC connections in the web application server.

Creating the JDBC connection in the web application server

About this task

The Unica Journey web application must be able to communicate with its system table database using a JDBC connection.

You must create this JDBC connection in the web application server where you plan to deploy Unica Journey.

Follow these guidelines if you decide to create the data source manually.

- In WebSphere®, set the classpath for your database driver during this process.
- When the Unica Journey system tables are created in a different schema from the default schema of the database login user, you must specify that non-default schema name in the JDBC connection used to access the system tables.
- In Tomcat, set the classpath for your database driver during this process.
- In JBOSS, set the classpath for your database driver by adding the module for the JDBC driver and register the SQL JDBC driver.
- You must use `JourneyDS` and `JourneyReportDS` as the JNDI name. This name is required, and is noted in the Unica Journey installation worksheet on page 7.

Configuring the web application server for your JDBC driver

The web application server where you plan to deploy Unica Journey must contain the correct JAR file to support JDBC connections. This enables your web application to connect to the system tables. The location of the JAR file must be included in the class path of the web application server.

WebSphere

About this task

Automatic Datasource Creation by the installer is not supported in the Journey Application. We need to do Manual steps to create the DataSources for the Journey Application.

Perform the following step to create the datasource:

1. Access the WebSphere Admin Console
2. Set Up the Data Source in WebSphere
3. Continue the Wizard: Set Up the JDBC Provider
4. Specify Security Aliases
5. Test the Data Source

For more information, please refer the WebSphere documentation.

JBoss

About this task

If you are using JBoss, you must perform this entire procedure.

1. Obtain the latest vendor-provided Type 4 JDBC driver for your system table database that is supported by Unica, as described in the *Recommended Software Environments and Minimum System Requirements* guide.
   Use the following guidelines after you obtain the JDBC driver.
   - If the driver does not exist on the server where you plan to deploy Unica Journey, obtain it and unpack it on the server. Unpack the drivers in a path that does not include spaces.
   - If you obtain the driver from a server where the data source client is installed, verify that the version is the latest supported by Unica Journey.
2. Add the full path to the driver, including the file name, to the class path of the web application server where you plan to deploy Unica Journey.
   Use the following guidelines.
   - For all supported versions of JBoss, add the JDBC driver as module. Use the following procedure to add the JDBC driver as a module.

     For example, for SQL Server:

     ```
     module add --name=com.microsoft.sqlserver.jdbc --resources=<JDBC_Driver_Location>\mssql-
     jdbc-7.0.0.jre8.jar --dependencies=javax.api,javax.transaction.api
     ```
   - Register this SQL JDBC Driver using the following guidelines: For example:
     - ```
       /subsystem=datasources/jdbc-driver=sql:add(driver-module-
       name=com.microsoft.sqlserver.jdbc,driver-name=sql,driver-xa-datasource-class-
       name=com.microsoft.sqlserver.jdbc.SQLServerXADataSource)
       ```
     - `/subsystem=datasources/jdbc-driver=sql:read-resource`
     - `/subsystem=ee/service=default-bindings:write-attribute(name=datasource, value=undefined)`
3. Make a note of the database driver class path in the Unica Journey installation worksheet, as you must enter the path when you run the installer.
4. Restart the web application server so that your changes take effect.

   During startup, monitor the console log to confirm that the class path contains the path to the database driver.

Apache Tomcat

About this task

If you are using Apache Tomcat, you must perform this entire procedure.

1. Obtain the latest vendor-provided Type 4 JDBC driver for your system table database that is supported by Unica, as described in the *Recommended Software Environments and Minimum System Requirements* guide.
   Use the following guidelines after you obtain the JDBC driver.
   - If the driver does not exist on the server where you plan to deploy Unica Journey, obtain it and unpack it on the server. Unpack the drivers in a path that does not include spaces.
   - If you obtain the driver from a server where the data source client is installed, verify that the version is the latest supported by Unica.
2. Add the full path to the driver, including the file name, to the class path of the (<Tomcat_Installed Location>/lib) web application server where you plan to deploy Unica Journey.
3. Make a note of the database driver class path in the , as you must enter the path when you run the installer.
4. Restart the web application server so that your changes take effect.

   During startup, monitor the console log to confirm that the class path contains the path to the database driver.

Information for creating JDBC connections

Use default values when you create JDBC connections if specific values are not provided. For more information, see the application server documentation.

> **Note:** If you are not using the default port setting for your database, make sure that you change it to the correct value.

WebSphere

Use the following values if your application server is WebSphere:

SQLServer

- Driver: N/A
- Default port: 1433
- Driver class: `com.microsoft.sqlserver.jdbc.SQLServerConnectionPoolDataSource`
- Driver URL: `jdbc:sqlserver:/<DBhostName>:1433;databaseName=<DBName>`

In the Database Type field, select User-defined.

After you create the JDBC Provider and data source, go to the Custom Properties for the data source and add, modify properties as follows:

- `serverName=<your_SQL_server_name>`
- `portNumber =<SQL_Server_Port_Number>`
- `databaseName=<your_database_name>`

Add the following custom property:

- Name: `webSphereDefaultIsolationLevel`
- Value: 1
- Datatype: Integer

Oracle

- Driver: Oracle JDBC Driver
- Default port: 1521
- Driver class: `oracle.jdbc.OracleDriver`
- Driver URL: `jdbc:oracle:thin:@<your_db_host>:<your_db_port>:<your_db_service_name>`

  Enter the driver URL by using the format that is shown. Unica applications do not allow the use of Oracle's RAC (Real Application Cluster) format for JDBC connections.

Add the following custom property:

- Name: `webSphereDefaultIsolationLevel`.
- Value: 2
- Datatype: Integer

MariaDB

- Database Driver: mariadb-java-client-2.5.2.jar
- Default port: 3306
- Driver class: `org.mariadb.jdbc.Driver`
- Driver URL: `jdbc:mariadb://<your_db_host>:<PORT>/<Your_DB_user_name>`
- Properties: Add user =*<your_db_user_name>*
- Properties: Add user = *password=<your_db_password>*
- Driver module xa-datasource-class= org.mariadb.jdbc.MySQLDataSource

OneDB

- Database Driver: onedb-jdbc-complete-8.0.0-SNAPSHOT.jre8.jar
- Database Port :20195
- Driver: Informix JDBC Driver
- Driver class: `com.informix.jdbc.IfxDriver`
- Driver URL: `jdbc:informix-sqli://<your_db_host>/`
  `<your_db_name>:INFORMIXSERVER=<your_db_servername>;`

DB2

- Database Driver: db2jcc4.jar
- Database Port :50000

- Driver class: `com.ibm.db2.jcc.DB2Driver`
- Driver URL: `jdbc:db2://<your_db_host>:<your_db_port>/<your_db_name>`

JBoss

Specify the native library path of the database driver JAR file on your server. For example: `db2jcc4.jar/ojdbc8.jar/mssql-jdbc-7.0.0.jre8.jar`.

Use the following values if your application server is JBoss:

SQLServer

- Database Driver: Microsoft MS SQL Server Driver (Type 4) Versions: 2012, 2012 SP1 and SP3, 2014, 2014 SP1, 2016 SP1
- Default port: 1433
- Driver class: `com.microsoft.sqlserver.jdbc.SQLServerDriver`
- Driver URL: `jdbc:sqlserver://<your_db_host>[\`
  `\<named_instance>]:<your_db_port>;databaseName=<your_db_name>,valid-connection-`
  `checker-class-name`
  `=org.jboss.jca.adapters.jdbc.extensions.mssql.MSSQLValidConnectionChecker`

For example :/subsystem=datasources/data-source=UnicaPlatformDS:add(jndi-name="java:/UnicaPlatformDS",connection-url="jdbc:sqlserver://localhost:1433;databaseName=plat11",driver-name=sql,user-name=sa,password=test1234,valid-connection-checker-class-name="org.jboss.jca.adapters.jdbc.extensions.mssql.MSSQLValidConnectionChecker")

Oracle

- Driver: Oracle JDBC Driver
- Default port: 1521
- Driver class: `oracle.jdbc.OracleDriver`
- Driver URL: `jdbc:oracle:thin:@<your_db_host>:<your_db_port>:<your_db_service_name>`

For example

MariaDB

- Database Driver: mariadb-java-client-2.5.2.jar
- Default port: 3306
- Driver class: `org.mariadb.jdbc.Driver`
- Driver URL: `jdbc:mariadb://<your_db_host>:<PORT>/<Your_DB_user_name>`
- Properties: Add user =*<your_db_user_name>*
- Properties: Add user = *password=<your_db_password>*
- Driver module xa-datasource-class= org.mariadb.jdbc.MySQLDataSource

OneDB

- Database Driver: onedb-jdbc-complete-8.0.0-SNAPSHOT.jre8.jar
- Database Port :20195
- Driver: Informix JDBC Driver
- Driver class: `com.informix.jdbc.IfxDriver`
- Driver URL: `jdbc:informix-sqli://<your_db_host>/`
  `<your_db_name>:INFORMIXSERVER=<your_db_servername>;`

DB2

- Database Driver: db2jcc4.jar
- Database Port :50000
- Driver class: `com.ibm.db2.jcc.DB2Driver`
- Driver URL: `jdbc:db2://<your_db_host>:<your_db_port>/<your_db_name>`

Tomcat

Specify the native library path of the database driver JAR file on your server. For example: mariadb-java-client-2.5.2.jar/
onedb-jdbc-8.0.0.1-complete.jar/ojdbc7.jar/mssql-jdbc-7.0.0.jre8.jar.

Use the following values if your application server is Tomcat:

SQLServer

- Database Driver: Microsoft MS SQL Server Driver (Type 4) Versions: SQL Server (e) 2014, 2016 SP1, 2017, 2019
- Default port: `1433`
- Driver class: `com.microsoft.sqlserver.jdbc.SQLServerDriver`
- Driver type: `javax.sql.DataSource`
- Driver URL: `jdbc:sqlserver://<your_db_host>[\`
  `\<named_instance>]:<your_db_port>;databaseName=<your_db_name>`

Oracle

- Driver: Oracle JDBC Driver
- Default port: 1521
- Driver class: `oracle.jdbc.OracleDriver`
- Driver URL: `jdbc:oracle:thin:@<your_db_host>:<your_db_port>:<your_db_service_name>`

MariaDB

- Driver: mariadb-java-client-2.5.2.jar
- Default port: 3306
- Driver class: `org.mariadb.jdbc.Driver`
- Driver URL: `jdbc:mariadb://<your_db_host>:<PORT>/<Your_DB_user_name>`
- Properties: Add *user=<your_db_user_name>*

OneDB

- Database Driver: onedb-jdbc-complete-8.0.0-SNAPSHOT.jre8.jar
- Database Port :20195
- Driver: Informix JDBC Driver
- Driver class: `com.informix.jdbc.IfxDriver`
- Driver URL: `jdbc:informix-sqli://<your_db_host>/`
  `<your_db_name>:INFORMIXSERVER=<your_db_servername>;`

DB2

- Database Driver: db2jcc4.jar
- Database Port :50000
- Driver class: `com.ibm.db2.jcc.DB2Driver`
- Driver URL: `jdbc:db2://<your_db_host>:<your_db_port>/<your_db_name>`

Chapter 4. Installing Unica Journey

You must run the Unica installer to start the installation of Unica Journey. The Unica installer starts the product installer during the installation process. Make sure that the Unica installer and the product installer are saved at the same location.

Each time that you run the Unica suite installer, you must first enter database connection information for the Unica Platform system tables. When the Unica Journey installer starts, you must enter the required information for Unica Journey

📝 **Note:** Unica Journey web application deployment using EAR file in tomcat is not supported.

Installation files

The installation files are named according to the version of the product and the operating system on which they must be installed, except UNIX™. For UNIX™, different installation files exist for the GUI mode and the console mode.

Example
The following table displays examples of the installation files that are named according to the product version and the operating system:

Table 6. Installation files

Operating system	Installation file
Windows™: GUI and console mode	`Product_N.N.N.N_win64.exe`, where *Product* is the name of your product, *N.N.N.N* is the version number of your product, and Windows™ 64-bit is the operating system on which the file must be installed.
	`Product_N.N.N.N_win.exe`, where *Product* is the name of your product, *N.N.N.N* is the version number of your product, and Windows™ 32 is the operating system on which the file must be installed.
LINUX: GUI mode	`Product_N.N.N.N_linux.bin`, where *Product* is the name of your product, and *N.N.N.N* is the version number of your product.
	`Product_N.N.N.N_linuxrhel64.bin`, where *Product* is the name of your product, and *N.N.N.N* is the version number of your product.
LINUX: Console mode	`Product_N.N.N.N.bin`, where *Product* is the name of your product, and *N.N.N.N* is the version number of your product. This file can be used for installation on all UNIX™ operating systems.

Unica Journey components

To use the Unica Journey utilities on additional computers, you must install the utilities and the web application on the additional computers. This is required because the utilities use the jar files in the web application. However, when you install

the Unica Journey to use the utilities, you do not have to deploy the Unica Journey again, and you do not have to create additional Unica Journey system tables.

The following table describes the components that you can select when you install Unica Journey:

Table 7. Journey components

Component	Description
Journey Web Application	Journey Web component gives user ability to use to design entry source, data definition and journey flows.
Journey Engine	Journey Engine processes the audience data, sends the communications to touch points and listens and capture the response information.
Kafka Standalone	Journey will be installing Kafka and Zookeeper components along with current installation process.
Journey Template	This is created for packaging Journey template in 12.1.2 and later releases.
	Journey templates are available when we install Journey product. During installation you can choose the templates that you want.

Note: In case Journey Web and Engine are installed on different machines, then folder /Journeys/Files need to be shared. It should be accessible for the Journey Engine Server in order to read and write files. If not done, the Journey could not read the csv files present in the /Journeys/ Files folder. If there are multiple journey engine, then this folder needs to be shared on each Journey engine system and read write access needs to be provided.

Installing Unica Journey using the GUI mode

For Windows™, use the GUI mode to install Unica Journey. For LINUX, use the GUI mode to install Unica Journey.

Before you begin

Important: Before you use the GUI mode to install Unica Journey, ensure that the available temporary space on the computer where you install Unica Journey is more than three times the size of the Unica Journey installer.

Make sure that the Unica installer and the Unica Journey installers are in the same directory on the computer where you want to install Unica Journey.

Complete the following actions to install Unica Journey by using the GUI mode:

1. Go to the folder where you have saved the Unica installer and double-click the installer to start it.
2. Click **OK** on the first screen to see the **Introduction** window.
3. Follow the instructions on the installer and click **Next**.

Use the information in the following table to take the appropriate actions on each window in the Unica installer.

Table 8. Unica Installer

Window	Description
Introduction	This is the first window of the Unica suite installer. You can open the Unica Journey installation and upgrade guides from this window. You can also see a link for the installation and upgrade guides for the products whose installers are saved in the installation directory.
	Click **Next**.
Response Files Destination	Click the **Generate Response File** check box if you want to generate response files for your product. Response files store the information that is necessary for the installation of your product. You can use response files for an unattended installation of your product, or to get pre-filled answers if you rerun the installer in GUI mode
	Click **Choose** to browse to a location where you want to store the response files.
	Click **Next**.
Unica Products	In the **Install Set** list, select **Custom** to select the products that you want to install.
	In the **Install Set** area, you see all of the products whose installers are in the same directory on your computer.
	The **Description** field shows the description of the product that you select in the **Install Set** area.
	Click **Next**.
Installation Directory	In the **Specify the installation directory** field, click **Choose** to browse to the directory where you want to install your product.
	If you want to install the product in the folder where the installers are stored, click **Restore Default Folder**.
	Click **Next**.
Select Application Server	Select Application Server type. If you are installing other products along with Journey in same installation

Window	Description
	- then you can select application server type where Platform will be deployed.
	Click **Next**.
Platform Database Type	Select Oracle or OneDB for Unica Platform database type.
	Click **Next**.
Platform Database Connection	Enter the following information about your database:
	• Database host name
	• Database port
	• Database name or System ID (SID)
	• Database user name
	• Database password
	Click **Next**.
Platform Database Connection (continued)	Review and confirm the JDBC connection.
	Click **Next**. The URL can be customized with additional parameters if needed.
Preinstallation Summary	Review and confirm the values that you added during the installation process.
	Click **Install** to start the installation process.
	The Unica Journey installer opens.

4. Follow the instructions on the Unica Platform installer to install or upgrade Unica Platform. See *Unica Platform Installation Guide* for more information.

5. In the **Installation Complete** window, click **Done**.

   Result

   The Unica Platform installation is complete and the Unica Journey installer opens.

6. Use the information in the following table to navigate the Unica Journey installer. In the **Platform Database Connection** window, enter all the required information and click **Next** to start the Unica Journey installer.

   Table 9. Unica Journey Installer GUI

Window	Description
Introduction	This is the first window of the Unica Journey installer. You can open the Unica Journey installation and upgrade guides from this window.
	Click **Next**.

Window	Description
Software License Agreement	Carefully read the agreement. Use **Print** to print the agreement. Click **Next** after you accept the agreement.
Installation Directory	Click **Choose** to browse to the directory where you want to install Unica Journey. Click **Next**.
Components	Select the components that you want to install. When you select the components, the installer displays information about the components. Click **Next**. Note: You can select any of these components for installation. Unica Journey all three components can be installed on same or different machines. • Unica Journey Web application • Unica Journey Engine • Apache Kafka
Unica Journey Database Setup	Unica Journey database setup is automatic. By default, it runs the SQL with Unicode support. If you select Automatic database setup, select Run Unicode SQL Script if your system tables are configured for Unicode. Click **Next**.
Unica Journey Database Type	Select the database type from Oracle, SQL. MariaDB or OneDB. Click **Next**.
Unica Journey Database Connection	Enter the following details for the Journey database: • Database host name • Database port • Database system ID (SID) • Database user name • Password Click **Next**.
JDBC Connection	Review and confirm the JDBC connection. Click **Next**.

Window	Description
Unica Journey Connection Settings	Enter the following connection settings: • Network domain name 📝 **Note:** When you add the network domain name, you might see the following message: ``` Warning-Server name includes domain name, final URL contains several occurrences of domain name ``` Select **Modify** to change the domain name or click **Cancel** to cancel the message. • Host name • Port number Select the **Use secure connection** check box if necessary. Click **Next**.
Unica Platform Connection Settings	Enter the following connection settings: • Network domain name 📝 **Note:** When you add the network domain name, you might see the following message: ``` Warning-Server name includes domain name, final URL contains several occurrences of domain name ``` Select **Modify** to change the domain name or click **Cancel** to cancel the message. • Host name • Port number

Window	Description
	Select the **Use secure connection** check box if necessary.
	Click **Next**.
Kafka Standalone Server Details	If you are installing Kafka standalone instance along with this instance it will update below details in Kafka configuration.
	• Host name: Include the host name of Kafka Standalone server, where Kafka is installed.
	• Port number: Include the port number of Kafka Zookeeper.
Preinstallation Summary	Review and confirm the values that you added during the installation process.
	Click **Install** to start the installation process.
	The Unica Journey installer opens.
Installation Complete	Click **Done** to close the Unica Journey installer and go back to the Unica installer.

7. In the **Installation Complete** window, click **Done** to exit the Unica Journey installer and go back to the Unica installer.
8. Follow the instructions on the Unica installer to finish installing Unica Journey.

Use the information in the following table to take the appropriate actions on each window in the Unica installer.

Table 10. HCL Unica Installer GUI

Window	Description
Deployment EAR file	Specify whether you want to create an enterprise archive (EAR) file to deploy your Unica products.
	Click **Next**.
Installation Complete	This window provides the locations of the log files that are created during installation.
	Click **Previous** if you want to change any installation details.

Window	Description
	Click **Done** to close the Unica installer.

✏️ **Note:** Journey does not support EAR deployment.

Installing Unica Journey using the console mode

The console mode allows you to install Unica Journey using the command-line window. You can select various options in the command-line window to complete tasks such as selecting the products to install or selecting the home directory for the installation.

Before you begin

Before you install Unica Journey, ensure that you have configured the following.

- An application server profile
- A database schema

To display the installer screens correctly in console mode, configure your terminal software to support UTF-8 character encoding. Other character encodings, such as ANSI, do not render the text correctly, and some information is not readable with these encodings.

1. Open a command-line prompt window and navigate to the directory where you have saved the Unica installer and the Unica Journey installer.
2. Complete one of the following actions to run the Unica installer.
   Choose from:
   - For Windows™, enter the following command:

     `HCL_Unica_installer_12.1.6.0_win.exe` -i console

     For example, HCL_Unica_Installer_12.1.6.0_win.exe -i console
   - For UNIX™, invoke the `HCL_Unica_installer_12.1.6.0`.sh file.

     For example: HCL_Unica_installer_12.1.6.0.sh
3. Follow the directions that are displayed in the command-line prompt. Use the following guidelines when you must select an option in the command-line prompt:
   - The default options are defined by the symbol [X].
   - To select or clear an option, type the number that is defined for the option, and then press `Enter`.
   Example
   For example, suppose that the following list displays the components that you can install:
       1 [X] Unica Platform
       2 [X] Unica Journey

> 📝 **Note:** Do not clear the option for Unica Platform unless you have already installed it.

4. The Unica installer launches the Unica Journey installer during the installation process. Follow the instructions in the command-line prompt window of the Unica Journey installer.

5. After you enter `quit` in the Unica Journey installer command-line prompt window, the window shuts down. Follow the instructions in the command-line prompt window of the Unica installer to complete the installation of Unica Journey.

> 📝 **Note:** If any error occurs during the installation, a log file is generated. You must exit the installer to view the log file.

Installing Unica Journey silently

Use the unattended or silent mode to install Unica Journey multiple times.

Before you begin

Before you install Unica Journey, make sure that you have configured the following elements:

- An application server profile
- A database schema

About this task

When you install Unica Journey by using the silent mode, response files are used to obtain the information that is required during installation. You must create response files for a silent installation of your product. You can create response files by using one of the following methods:

- Using the sample response files as a template for creating your response files. The sample response files are included with your product installers in the `ResponseFiles` compressed archive.
- Running the product installers in the GUI mode or the console mode before you install the product in the silent mode. One response file is created for the Unica suite installer, and one or more response files are created for your product installer. The files are created in the directory that you specify.

> ⚠️ **Important:** For security reasons, the installer does not save database passwords in the response files. When you create response files, you must edit each response file to enter database passwords. Open each response file and search for PASSWORD to find where you must edit the response file.

When the installer runs in the silent mode, it looks for the response files in the following directories sequentially:

- In the directory where the Unica installer is saved
- In the home directory of the user who installs the product

Make sure that all response files are in the same directory. You can change the path where response files are read by adding arguments to the command line. For example: -DUNICA_REPLAY_READ_DIR="myDirPath" -f myDirPath/installer.properties

Use the following command for Windows™:

- `HCl_Unica_Installer_12.1.6.0_win.exe` -i silent

    HCL_Unica_installer_12.1.6.0_Operating_system.bin -i silent

    Example

    For example:

    HCL_Unica_installer_12.1.6.0_win.exe -i silent

Use the following command for Linux™:

- *HCL_Unica_installer_12.1.6.0 _operating_system* .bin -i silent

    Example

    For example:

    HCL_Unica_installer_12.1.6.0_linux.bin -i silent

    > **Note:** After upgrade with Oracle as system database is complete, please check journey system tables
    > schema and make sure all the procedures and triggers under this schema are marked successfully complied.
    > If any procedure and trigger is not compiled, please compile the procedure and trigger manually. It does not
    > need any SQL/script changes in procedures and triggers. Name of procedures and triggers are as follows:
    >   - EMAIL_PERF
    >   - PROCESS_JOURNEY_GOALS
    >   - PROCESS_JOURNEY_GOALS_HIST
    >   - PROCESS_JOURNEY_GOALS_META
    >   - PUSH_PERF
    >   - SMS_PERF

Sample response files

You must create response files to set up a silent installation of Unica Journey. You can use sample response files to create
your response files. The sample response files are included with the installers in the `ResponseFiles` compressed archive.

The following table provides information about sample response files:

Table 11. Description of sample response files

Sample response file	Description
`installer.properties`	The sample response file for the Unica suite installer.
`installer_`*`product initials`* *`and product version`* *`number`*`.properties`	The sample response file for the Unica Journey installer. For example, `installer_uj`*`n.n.n.n`*`.properties` is the response file of the Unica Journey installer, where *n.n.n.n* is the version number.

Table 11. Description of sample response files

(continued)

Sample response file	Description
	For example, `installer_ujn.n.n.n.properties` is the response file of the Unica Platform installer, where `n.n.n.n` is the version number.

Chapter 5. Configuring Unica Journey

Before you deploy Unica Journey, you must set up the Unica Journey system user, set Unica Journey configuration properties, and verify the Unica Journey installation.

After the installation please check in the Database if the procedures - process_journey_goals_hist compiled successfully. Recompile the procedure - process_journey_goals_hist if it shown as invalid.

Unica Journey configuration properties

The Unica Journey has additional properties on the Configuration page that perform important functions that you must specify. See the *Unica Journey User Guide* to learn more about what they do and how to set them.

Setting the Link and Deliver configuration properties

User can configure Unica Journey to connect to the Unica Link by using Link Configuration under the path `Affinium|Journey|Link_Configurations` in Unica Platform.

Set the following Unica Link configuration properties under the **Journey > Link_Configurations** category manually:

- Link_URL: Specify Unica Link design server URL Ensure that it does not have the trailing /. Example: http://<FQDN>:<PORT>
- Link_Data_Source_User: Specify the Platform user which stores the credentials to Unica Link design server.
- Link_Data_Source_Name: Specify the datasource name which has the credentials information.

Also, You can configure Unica Journey to connect to theUnica Deliver by using Deliver Configuration under the path `Affinium|Journey|Deliver_Configurations`.

Set the following Unica Link configuration properties under the **Journey > Deliver_Configurations** category manually:

- Deliver_URL: Specify the Unica Deliver TMS server URL. For example: `http://<UNICA_DELIVER_HOST>/delivertms/api/deliver/rest/v1/tms`.
- Deliver_Partition: Specify the partition name in Campaign where Deliver TMS is configured.

> **Note:** Make sure you entered correct partition name in Deliver_Partition.

You can use the following configurations to enable Unica Link and Unica Deliver integrations:

You can navigate under Platform configuration settings:

Settings for 'Journey' (Affinium|Journey)

Link_Configured - This configuration defines whether Unica Link is integrated with Unica Journey (for email/SMS/CRM channels).

Possible values - Yes/ No

Yes - Enables Unica Link integration with Journey

Deliver_Configured - This configuration defines whether Unica Deliver is integrated with Unica Journey for email.

Possible values - Yes/ No

Yes - Enables Unica Deliver integration with Journey

You are required to restart Journey web and engine components.

Platform_Configured - This configuration defines whether Unica Platform is integrated with Unica Journey

Following Platform APIs should be disabled for integrating Journey:

Path = Configuration/Unica Platform/Security/API Management/Unica Platform

- Authentication = Disable all
- Datasource = Disable all
- User details = Disable all
- Get Configuration Property = Disable all
- Configuration = Disable all
- Login = Disable all
- User roles permission = Disable all
- User details = Disable all
- License = Disable all
- Get installed applications = Disable all

Note: The token validity of platform should be increased manually to 1800 seconds (30 minutes) because the default token validity of platform is 15 seconds. You will find the Path for increase token validity for platform: **Platform > Settings > Configuration > General > Miscellaneous > Token lifetime**

Contact_Central_Configured - For enabling the contact central for Journey the value of Contact_Central_Configured value should be set to 'Yes' from platform. By default, the value is set to No. User can select the desired value Yes/No for Contact_Central_Configured from the path Affinium|Journey in Platform.

API settings to access Journey in HTTPS:

- Settings for 'Authentication' (Affinium|suite|security|apiSecurity|manager|managerAuthentication)
    ◦ API URI - /authentication/login
    ◦ Block API access - Disabled
    ◦ Secure API access over HTTPS - Enabled
    ◦ Require authentication for API access - Disabled
- Settings for 'User' (Affinium|suite|security|apiSecurity|manager|managerUser)
    ◦ API URI - /user/partitions/*
    ◦ Block API access - Disabled
    ◦ Secure API access over HTTPS - Disabled
    ◦ Require authentication for API access - Enabled
- Settings for 'Policy' (Affinium|suite|security|apiSecurity|manager|managerPolicy)
    ◦ API URI - /policy/partitions/*
    ◦ Block API access - Disabled

- ◦ Secure API access over HTTPS - Disabled
- ◦ Require authentication for API access - Enabled
- Settings for 'Configuration' (Affinium|suite|security|apiSecurity|manager|Configuration)
    - ◦ API URI - /datasource/config
    - ◦ Block API access - Disabled
    - ◦ Secure API access over HTTPS - Enabled
    - ◦ Require authentication for API access - Enabled
- Settings for 'Datasource' (Affinium|suite|security|apiSecurity|manager|Datasource)
    - ◦ API URI - /datasource
    - ◦ Block API access - Disabled
    - ◦ Secure API access over HTTPS - Enabled
    - ◦ Require authentication for API access - Disabled
- Settings for 'Login' (Affinium|suite|security|apiSecurity|manager|Login)
    - ◦ API URI - /authentication/v1/login
    - ◦ Block API access - Disabled
    - ◦ Secure API access over HTTPS - Enabled
    - ◦ Require authentication for API access - Disabled
- Settings for 'User roles permissions' (Affinium|suite|security|apiSecurity|manager|managerGetRolesPermission)
    - ◦ API URI - /policy/roles-permissions
    - ◦ Block API access - Disabled
    - ◦ Secure API access over HTTPS - Disabled
    - ◦ Require authentication for API access - Enabled
- Settings for 'User details' (Affinium|suite|security|apiSecurity|manager|managerGetUserDetails)
    - ◦ API URI - /user/user-details
    - ◦ Block API access - Disabled
    - ◦ Secure API access over HTTPS - Enabled
    - ◦ Require authentication for API access - Enabled
- Settings for 'Get configuration property' (Affinium|suite|security|apiSecurity|manager|managerGetConfigProperty)
    - ◦ API URI - /configuration/get
    - ◦ Block API access - Disabled
    - ◦ Secure API access over HTTPS - Enabled
    - ◦ Require authentication for API access - Disabled
- Settings for 'License' (Affinium|suite|security|apiSecurity|manager|managerLicense)
    - ◦ API URI - /license/*
    - ◦ Block API access - Disabled
    - ◦ Secure API access over HTTPS - Disabled
    - ◦ Require authentication for API access - Disabled

Note: After applying these configuration changes, restart the platform application.

How to generate encrypted password

You are required to generate encrypted passwords for each plain text password. Run encryption tool multiple times to generate encrypted passwords.

1. Navigate to `<JOURNEY_WEB_HOME>/tools/`.
2. Set `JAVA_HOME` in JourneyEncryptionUtility .

    JAVA_HOME=<UNICA_HOME>/jre

    export JAVA_HOME
3. If you are using Linux operating system, convert JourneyEncryptionUtility to Unix mode using the following command:

    dos2unix JourneyEncryptionUtility
4. Run JourneyEncryptionUtility with the following command:

    JourneyEncryptionUtility <PASSWORD TEXT>
5. JourneyEncryptionUtility prompts the password in encrypted mode on the console output.
6. If due to any reason user changes Journey system tables database password or Journey Reports database user password, then they can use the password encryption utility to update these passwords in Journey Engine application.properties file or in case user is using tomcat then they need to generate password and add in Journey.xml

Steps to generate ClientID and ClientSecret

Run Unica Platform's clientDetailsUtility to generate client details for Journey as below:

On Linux system, use .sh file instead of .bat.

1. Navigate to PLATFORM_HOME\tools\bin directory. If you have Platform installed on different machine, you can execute this command on the machine where Platform is installed.
2. Execute command as - clientDetails -a Journey. This generates the ClientID and ClientSecret. The following is an example.

    C:\Unica\Platform\tools\bin>clientDetails.bat -a Journey

    C:\Unica\Platform\tools\bin>echo off

    WARN com.unica.manager.configuration.ConfigurationManager - Local cache is turned off. Default behavior implies based on Hibernate cache

    Parameter value

    ClientID: 885345

    ClientSecret: IfnKG2eqniVnaT8

    AppName: Journey

    ClientSecret and ClientId generated successfully!!
3. Use generated ClientID and ClientSecret in Journey Web application.properties:

    platform.clientId=ClientID generated in above step.

    platform.clientSecret= encrypted ClientSecret in above step

Update the Journey Web and Journey Engine application properties

Update the Journey Web and Journey Engine application properties. The following procedure provide the following steps to perform updates.

Users are required to complete the following steps to perform the updates in Journey Web `application.properties`.

1. Below mentioned properties are used to start Platform and Journey parallelly. Platform must be started before Journey start; some application server takes more time than usual to start the platform. These properties are used while starting Journey, it will try to connect platform in specified retry number and time interval.
   - `platform.connect.retry.number`: number of retries to connect platform
   - `platform.connect.retry.interval`: retry interval time for connect to platform in milliseconds

   User can change the value of these properties in <Journey_Home>/Web/ Properties/application.properties. Values of these properties are different based on which application server is used. For Tomcat default values will work and for Websphere Application Server (WAS) need to increase time interval between retries.

2. Modify JOURNEYS_HOME/Web/properties/application.properties to update "spring.entity.files.upload.defaultPath" parameter path to include double forward slash (\\) instead of single Forward slash (\).This is in case of installation of Journeys on Windows.

3. Modify JOURNEYS_HOME/Web/properties/application.properties to update "spring.ignite.storage.path" parameter path to include double forward slash (\\) instead of single Forward slash (\).This is in case of Windows.

   Note: By default, properties `spring.entity.files.upload.defaultPath` and `spring.entity.files.upload.defaultFileReadBuffer` appear in single line.

   user needs to remove space available after path end of spring.entity.files.upload.defaultPath=/opt/SD1215/ Journeys/Files/{}

   and user needs to separate out spring.entity.files.upload.defaultpath and spring.banner.location

   Note: If on base version in Journey web and engine application.properties files few manual configuration changes are done, then after upgrading the Journey standalone or as cluster, if manual configurations are lost from application.properties, then user needs to reconfigure application.properties. Journey V12.1.6 installer will take a backup of application.properties at same location by name application.properties.bkp for restoring old version configuration changes.

   Note: From V12.1.6 onwards, a new feature Full Masking is introduced. Earlier, Unica Journey audience fields that are masked are displayed as * for any random characters. Currently, the audience fields that are masked displays the field values as * format but this is done partially and not in entire field value.

With this feature, if the property **isFullmask** is set to true both web and engine `application.properties` file as `journey.isFullMask=true`. Complete masking for entire data that is displayed will be shown as *. Example - email address shown as - rahul@abc.com will be shown after masking as - *************.

In case of isFullmask is set to false masking will be done partially as earlier.

Update Journey Engine - `application.properties` file

You are required to set passwords in encrypted format in Engine application.properties files(Journeys_Install_location/Engine/), This is a manual process.

The following procedure provide the following steps to perform updates.

1. Generate encrypted password for the following properties and mention in the Engine application.properties file using the: `/JourneyEncryptionUtility.sh <JOURNEYS_HOME/tools>`

   - `journey.datasource.password`
   - `journey.report.datasource.password`

   Execute command as JourneyEncryptionUtility.sh (<JOURNEYS_HOME/tools>)<Journey System schema password> or <Journey Report schema password>. This generates encrypted password .

   The following is an example.

   [unica@cobra009 tools]$ ./JourneyEncryptionUtility.sh JourneySysctemschema

   Encryption Shell Script started...

   Entered String is : JourneySysctemschema

   Encrypted String is : 3CKsX5SWYtGl+psHqlYUGkjXF9EVv6+XYP6GTIMa7WQ=

2. Modify JOURNEYS_HOME/Engine/application.properties to update "spring.entity.files.upload.defaultPath" parameter path to include double forward slash (\\) instead of single Forward slash (\).This is in case of installation of Journeys on Windows.

3. Modify JOURNEYS_HOME/Engine/application.properties to update "spring.ignite.storage.path" parameter path to include double forward slash (\\) instead of single Forward slash (\).This is in case of Windows.

4. Use generated ClientID and ClientSecret in Journey Engine application.properties:

   platform.clientId=ClientID generated in above step for Journey Web Application properties file

   platform.clientSecret= encrypted ClientSecret for in above stepJourney Web Application properties file

Note: By default, properties `spring.entity.files.upload.defaultPath` and `spring.entity.files.upload.defaultFileReadBuffer` appear in single line. User needs to bifurcate them into two different properties, like below:

`spring.entity.files.upload.defaultPath`

> ✎ `spring.entity.files.upload.defaultFileReadBuffer`

> ✎ **Note:** The above Web and Engine ignite and temp folder paths after upgrade should be same as the one before upgrade to avoid any inconsistences in data processing for Journeys created before upgrade.

For taking backup of Journey cache set the property `journey.cache.backup` as True. By default the property is set to False.

Archival of audience data

Archival of audience data and duplication will be based on flag dedup.end.audience.flag , which is configured in application.properties of engine.

`(dedup.end.audience.flag=false)` - Journey will check whether audience exists in live audience set, if it finds it there, data refresh will happen. If it does not find audience in live audience set, it will check in ended audience set. If it finds it there, it will discard the record. Milestone will not update in this case.

`(dedup.end.audience.flag=true)` - If customer understands that in their ecosystem, they are not sending duplicate data unnecessary & performance is their priority, Journey will check whether audience exists in live audience set. If it finds it there, data refresh will happen. If it does not find in live audience set, the new audience will be added to journey.

By default the flag is set to false.

List of application.properties of Journey Engine

Property	Description
spring.jpa.properties.hibernate.jdbc.batch_size=900	Maximum size of DMLs batched together
spring.jpa.properties.hibernate.order_inserts=true spring.jpa.properties.hibernate.order_updates=true	Enables batch record insertions & updates respectively (wherever used). Turning these off causes individual record insertion/update & slows down overall operation, functionality remains unchanged.
journey.kafka.producer.linger.ms=500	Time in milliseconds Kafka producer waits for gathering 16kb (default) data before making network dispatch
journey.kafka.consumer.fetch.min.bytes=1048576	Minimum number of bytes fetched by Kafka consumer in each fetch request
journey.kafka.consumer.max.partition.fetch.bytes=1048576	Maximum number of bytes Kafka consumer can fetch from single partition in one fetch request. Limits {fetch.min.bytes} by this upper cap if consumer is assigned to only one partition.
service.backlog.cleanup.standby.mins=10	Number of minutes deprecated services will wait for pre-upgrade backlog to appear in their Kafka topic poll response before terminating themselves. If no record is found within these many minutes, deprecated services are closed, and are never created again while the engine is up.

Property	Description
	If any record is found, corresponding service stands by for these many minutes more. Such services are created again to check for backlog on next application restart though.
decision-split.retry.max-attempts=3	Maximum attempts to be made for running decision split query. Execution will happen for (decision-split.retry.max-attempts + 1) times in case of persistent failure
decision-split.retry.back-off.delay-ms=200	Fixed delay (in milliseconds) between two retrial attempts for decision split query execution
dedup.end.audience.flag=false	This flag is used for de-duplication for audiences who completed their journey. By default dedup.end.audience.flag=false, journey will check whether audience exists in live audience database table, if it finds it there, data refresh will happen. If it does not find in live audience set, it will check in ended audience database table. If it finds it there, it will discard the record. Milestone will not update in this case. If, dedup.end.audience.flag=true => if customer understands that in their ecosystem they are not sending duplicate data unnecessary & performance is their priority,journey will check whether audience exists in live audience database table. If it finds it there, data refresh will happen. If it does not find in live audience database table, the new audience will be added to journey.
audience.archival.cron=0 0/30 * * * ?	This is audience archival job configuration.
journey.ignite.cluster=false	
wait.thread.sleep = 30000	
journey.engagement.split.sleep.time=1000	
journey.cache.fetch.limit=20000	
web.client.max.connections=1000	The maximum number of connections (per connection pool) before start pending
web.client.max.idle=10	The Duration after which the channel will be closed when idle (resolution: second)

Property	Description
web.client.evict.time=120	Specifies the interval to be used for checking the connection pool (resolution: second)
web.client.connect.timeout=10000	Timeout value to setup a channel connection (resolution: ms)
web.client.response.timeout=30000	httpclient response timeout duration (resolution: ms)
web.client.connections.pendingAcquireTimeout=90	The maximum time after which a pending acquire must complete or the TimeoutException will be thrown (resolution: second)
journey.datasource.transactionTimeout=100000	Database transaction timeout in milliseconds. Default is 100 seconds if not specified.

log4j2.xml

Section	Description
<RollingFile name="batchDmlsFileLogger" fileName="${APP_ROOT_LOG}/BatchDMLs.log" filePattern="${APP_ROOT_LOG}/BatchDMLs-%d{yyyyMMdd}-%i.log"> <PatternLayout pattern="${LOG_PATTERN}" /> <Policies> <SizeBasedTriggeringPolicy size="50MB" /> <TimeBasedTriggeringPolicy interval="1" /> </Policies> <DefaultRolloverStrategy max="10" /> </RollingFile>	Rolling file appender for generating new log file containing statistics pertaining to batch DML operations controlled using spring. jpa. properties. hibernate. jdbc. batch_size, spring. jpa. properties. hibernate. order_inserts & spring. jpa. properties. hibernate. order_updates properties in application.properties file. This section is commented by default.
<Logger name="batch-dmls-logger" level="DEBUG"> <appenderRef ref="batchDmlsFileLogger" /> </Logger>	Logger for turning batch DML stats logging on & off. Setting log level to DEBUG turns the logging on. This section is commented by default.

Update KafkaStandalone server.properties and zookeeper.properties file

If you are using Windows, perform the following sub-steps.

1. Modify `JOURNEY_HOME/KafkaStandalone/config/zookeeper.properties` under to update "dataDir" parameter path to include double forward slash (\\) instead of single Forward slash (\).
2. Modify `JOURNEY_HOME/KafkaStandalone/config/server.properties` to update "log.dirs" parameter path to include double forward slash (\\) instead of single Forward slash (\).

Update journey.xml used in Tomcat deployment

Journey deployment XML file needs to have encrypted password for Journey system tables. You can encrypt the password using JourneyEncryptionUtility and specify the journey.xml file for field "password".

In Zookeeper properties files below two parameters needs to be added:

- admin.serverPort=<SPECIFY PORT>
- admin.enableServer=false/true

When admin.serverPort parameters not specified then it takes the default values as mentioned below, if these are conflicting to any other application then zookeeper does not get started.

admin.serverPort=<SPECIFY PORT>

Zookeeper installed by Journey installer is not getting started. There is a known issue with the zookeeper upgrade.

Reference defect on zookeeper: https://issues.apache.org/jira/browse/ZOOKEEPER-3056

Workaround mentioned on this defect for zookeeper is as below:

1. Add below property in zookeeper.properties file, this property will skip the check.
   zookeeper.snapshot.trust.empty=true
2. Start Zookeeper server after adding this property in zookeeper.properties files.
3. Once the zookeeper is started then remove this property as this is not needed from next time.
4. This check is important to ensure that the system is in good shape.

Registering Unica Journey manually

For using New Platform UX, please execute the following commands using the xmls as mentioned below:

Navigation Path	Name of xml	Command to be run

`|`
`Affinium|suite|uiNavigation|main`
`Menu|`
	JOURNEY_HOME/Web/conf/upgrade 1213To1214/MenuCategory_Journey. xml	configTool.bat -vp -p "Affinium|suite|uiNavigation|mainM enu" -f "MenuCategory_Journey.xml"

Installer will laydown the latest journey_configuration.xml to *JOURNEY_HOME/Web/conf/journey_navigation.xml*

Installer will lay down the MenuCategory_Journey.xml to JOURNEY_HOME/Web/conf/upgrade1213To1214/ MenuCategory_Journey.xml{*}{{*}}

Starting and Verifying Unica Journey installation

If you performed all of the steps to install and configure Unica Journey, deploy the Unica Journey web application, and configure Unica Journey after deployment, you are ready to verify your installation.

Before you begin

Prerequisites for starting Journey application

Prerequisites to start Journey Web or Engine application:

- Unica Platform must be started.
- Zookeeper server is up and running.
- Kafka server is up and running.

Starting and Verifying Unica Journey

Unica Journey web application which is deployed in Tomcat application server is required to be started by starting the Tomcat instance.

Starting Kafka server and Zookeeper

You can use the following commands to start Kafka server and Zookeper server.

- Navigate to `JOURNEY_HOME/KafkaStandalone/bin` (for Linux)
- Navigate to `JOURNEY_HOME/KafkaStandalone/bin/windows` (for Windows)

Execute the following command to start Zookeeper first (Zookeeper needs to be up and running while you start stop Kafka server.

```
zookeeper-server-start <PATH TO ZOOKEEPER CONF FILE>
```

For example: `zookeeper-server-start JOURNEY_HOME/KafkaStandalone/config/zookeeper.properties`

```
kafka-server-start <PATH TO SERVER CONF FILE>
```

For example: `kafka-server-start JOURNEY_HOME/KafkaStandalone/config/server.properties`

Starting Unica Journey Engine/Server

- Unica Journey Engine application is a standalone application and that can be started as below.
  - Navigate to `JOURNEY_HOME/Engine` directory.
  - Run Engine application by running the following command: java `-jar journeyEngine.jar`. You can optionally write script to start this as a service.

Verifying Journey Installation

If you have not already done, log in to Unica as a user that exists in the Unica Platform Administrators role (such as asm_admin). You require to define the user roles and permissions for the Unica Journey user by navigating to **Settings > User Roles and Permissions**. Under User Roles and Permissions, you must assign roles and permissions for Unica Journey application. You cannot assign new user roles and work with system provided user roles - JourneyAdmin and JourneyUser. You can review and edit the roles these two user roles can perform. Once your define user roles and permissions for

JourneyAdmin and JourneyUser, you can assign these roles to any Platform user which you require to provide access in Journey application to different application functionalities.

Setting properties for integration with Unica products

Unica Journey integrates with various applications.

See the documentation map in below tables to get more information on Unica Journey integration with other Unica suite products.

Installing and Configuring Unica Link

Task	Documentation
Installation and configuration of Unica Link	*Unica Link V12.1 Installation Guide*
Installing Unica Link connector app for Journey	*Unica Link V12.1 Installation Guide*
Installing Unica Link connector – MailChimp	*Unica Link Mailchimp Connector User Guide*
Installing Unica Link connector – Mandrill	*Unica Link Mandrill Connector User Guide*
Installing Unica Link connector – Twilio	*Unica Link Twilio Connector User Guide*
Installing Unica Link connector – Salesforce	*Unica Link Salesforce Connector User Guide*
Installing Unica Link connector – AdTech (Facebook, Google, Twitter)	*Unica Link Facebook Connector User Guide*
	Unica Link Google Connector User Guide
	Unica Link Twitter Connector User Guide

> **Note:** If you are configuring Unica Link with Unica Journey using a Safari browser, perform the following steps:
>
> 1. Select **Safari > Preferences**.
> 2. Select the **Privacy** tab.
> 3. For the **Website tracking** field, check if **Prevent cross-site tracking** is deselected. If it is selected, deselect it.

Integrate Unica Campaign with other HCL products

Task	Documentation
Integration of Unica Campaign and Unica Journey	*Unica Campaign Administration Guide* and *Unica Campaign User Guide*
Integration of Unica Campaign and Unica Interact	*Unica Interact Administration Guide*
Integration of Unica Deliver with Unica Journey	*Unica Journey User Guide*

Journey Proxy Integration

Proxy server has been integrated into journey web and Engine Projects, this gives user an upper hand in adding security and keeping application server behind proxy servers. Proxy server will interact with Deliver, Link and Platform servers.

Journey Web – Communicates with Deliver, Link and Platform server for fetching configuration details and while integrating Email/SMS/AdTech Point in Journey.

Journey Engine – Uses proxy to communicate with Deliver/Link Server for submitted Email/SMS/Adtech details to end servers.

Proxy Supported by Journey Web

1. SOCKS
2. HTTP
3. HTTPS

Proxy Supported by Journey Engine

1. HTTP

> **Note:** SOCKS and HTTPS proxy is not supported by SOAP (Apache Axis2) used by Engine to communicate with Deliver.

Property to be configured for Engine in Engine application.properties file

- journey.proxy.type=NONE
- spring.proxy.host=[IP]
- spring.proxy.port=[PORT]
- spring.proxy.username=[username]
- spring.proxy.password=[password]

Property to be configured for Web in Web application.properties file

- journey.proxy.type=NONE
- spring.proxy.host=[IP]
- spring.proxy.port=[PORT]
- spring.proxy.username=[USERNAME]

- spring.proxy.password=[PASSWORD]
- server.use-forward-headers=true

Note: Default value of property journey.proxy.type is NONE, when set to NONE proxy is disabled.

Journey Engine Connection Pool settings

- journey.datasource.maxpool.size=[MAX_POOL_SIZE] – set size of Db cnnection Pool
- journey.datasource.minIdle.size=[MIN_IDLE_SIZE] – sets size of minimum idle connections

Database Changes

MS SQL script to be executed for Email Unsubscribed events:

Please execute the below SQL scripts on Journey system tables, this is required for Email Unsubscribe details to be populated.

DROP TABLE IF EXISTS EmailUnsubscribedList;

CREATE TABLE EmailUnsubscribedList(

id BIGINT NOT NULL IDENTITY,

emailId NVARCHAR(200) NOT NULL,

status NVARCHAR(200) DEFAULT 0 NOT NULL,

channelAgent NVARCHAR(50),

eventID BIGINT NOT NULL,

audienceResponseId BIGINT,

audienceResponseExtendedId BIGINT,

createdBy NVARCHAR(200) DEFAULT 'SYSTEM' NOT NULL,

version BIGINT,

createdDate DATETIME2,

createdDateEpoch BIGINT NOT NULL,

modifiedDateTimeEpoch BIGINT,

FOREIGN KEY (eventID) REFERENCES AudienceResponseEventMaster(id),

FOREIGN KEY (audienceResponseId) REFERENCES AudienceResponse(id),

CONSTRAINT unique_emailId UNIQUE (emailId),

PRIMARY KEY (id)

);

DROP TABLE IF EXISTS AudienceResponseExtended;

CREATE TABLE AudienceResponseExtended(

id BIGINT NOT NULL IDENTITY,

audienceResponseId BIGINT NOT NULL,

associatedAttributes NVARCHAR(MAX),

isProcessed BIT DEFAULT 0 NOT NULL,

createdDate DATETIME2,

createdBy NVARCHAR(200),

version BIGINT,

responseTimeEpoch BIGINT NOT NULL,

createdDateEpoch BIGINT,

FOREIGN KEY (audienceResponseId) REFERENCES AudienceResponse(id),

CONSTRAINT ensure_attribute_json CHECK (ISJSON(associatedAttributes) > 0),

PRIMARY KEY (id)

);

Journey Engine Cluster Configuration

In production systems, Kafka is typically run as a cluster. This allows better load management and handling capabilities as well as providing fault tolerance.

The Journey application, and the Journey Engine specifically utilizes this ability of Kafka to be setup in a cluster for high volume production systems which also need redundancy in case of the failure of a single node. In this document we detail out the proposed layout and required configuration to host Journey Engine on a typical production cluster. The figures and number are indicative and will need to change as per actual requirements.

This is applicable only for Linux systems since installing and running Kafka on Windows is not recommended and has known issues.

- There is single shared Database Server for all applications and nodes.
- Servers 01 – 03 run combinations of Journey Engine and Kafka.
- It is recommended that the setup has a minimum of 2 instances of Journey Engine and 3 Kafka brokers.
- Server 04 runs Platform, Journey Web and Zookeeper. If other applications are also required and installed on this Server, the configuration shall need to be adjusted accordingly.

Typical Server Configuration

1. Journey Engine
   - CPU – 8 cores
   - RAM – 32 GB
   - Disk Space – 250 GB
2. Kafka

- CPU – At least 4 cores
- RAM – 32 GB
- Disk Space – 500 GB

3. Journey Web + Platform
    - CPU – 4 cores
    - RAM – 32 GB
    - Disk Space – 250 GB

4. Database
    - CPU – 4 cores
    - RAM – 32 GB
    - Disk Space – 500 GB

Required Configurations for Cluster

Journey Engine

1. Application properties
    - spring.ignite.ipFinder.List=< IPs of all system to be part of cluster>
        ◦ Example - spring.ignite.ipFinder.List =*IP-Address-1*,*IP-Address-2*,*IP-Address-3*,…,*IP-Address-n*
        ◦ This is to be done on all nodes where Journey Engine is running
        ◦ If new node is to be added in cluster then IP should be added to this property on all servers and nodes to be restarted.
    - journey.ignite.reload=<set to all, will reload all active journeys>
        ◦ Example - journey.ignite.reload=all
        ◦ Example - journey.ignite.reload=all This property is to be enabled on Master Node – the node that is started first and shutdown last - only
        ◦ Other Node(s) will get replicated data
    - journey.ignite.cluster=true – on all nodes
    - Journey.engine.secondary.node -- this property will have value as false for primary node, for rest of Engine nodes this property will have value as true.
    - Journey.cache.sync property to be enabled
        ◦ Only on Master Node or any one node which will be started first and closed in last
    - `journey.ignite.cluster.node.discovery.network.timeout` - Maximum network timeout in milliseconds for Ignite cluster node discovery. Default is 60000 milliseconds (60 seconds).
    - `journey.ignite.cluster.node.discovery.socket.timeout` - Sets socket operations timeout for Ignite node discovery. This timeout is used to limit connection time and write-to-socket time. Note that when running on Amazon EC2, socket timeout must be set to a value significantly greater than 30000 ms (30 seconds). Default is 60000 milliseconds (60 seconds).
    - `journey.ignite.cluster.node.communication.connect.timeout` - Sets connect timeout used when one Ignite node is establishing connection with remote Ignite nodes. Default is 60000 milliseconds (60 seconds).
    - `journey.ignite.cluster.node.communication.socket.timeout` - Sets socket write timeout for TCP connection. If Ignite node cannot write message to socket within this time, then connection is closed and reconnect is attempted. Default is 60000 milliseconds (60 seconds).

Kafka

1. Before starting Zookeeper and Kafka – increase (ulimit -n 100000, on linux)
2. Zookeeper port (2181 default) should be accessible from all servers in cluster
   - Test by using telnet <IP> 2181
3. Broker port (default 9092) should be accessible from all servers where engine is running
4. After starting Zookeeper, use the following command to check status if application is up and port is occupied
   - netstat –tulnp | grep 2181
5. Each Broker should point to same zookeeper
6. Under Broker property set replication to minimum 2,
   - If cluster contains 3 servers set the following values
     a. offsets.topic.replication.factor=2
     b. transaction.state.log.replication.factor=2
     c. transaction.state.log.min.isr=2

> **Note:** When user wants to use multiple kafka brokers and they start their engine with single kafka borker, then topics by default get created on this single broker. Later, when user updates the broker list then kafka doesn't replicate the topics on other brokers because at the time of installation the replication value was set to one. Hence, during installation only, user needs to specify whether they want to use kafka with multiple brokers or with single broker.
>
> Cluster Kafka:
>
> Kafka Broker list: <BROKER_HOST1>:<PORT>, <BROKER_HOST2>:<PORT>, <BROKER_HOST3>:<PORT>
>
> Information message on the broker list format.
>
> What it shall do:
> a. Update engine and web application properties with below parameter
>
>   spring.kafka.bootstrap-servers=<BROKER_HOST1>:<PORT>, <BROKER_HOST2>:<PORT>, <BROKER_HOST3>:<PORT>
> b. JourneyEngine :
>
>   We added below property in journey_master_config.properties file in Journey Engine. Need to update from installer according to number of broker.
>
>   **journey.kafka.replications** = <Number of Brokers>
> c. JourneyWeb :
>
>   We added below property in JounreyWeb/application.properties file in Journey web. Need to update from installer according to number of broker.
>
>   **journey.kafka.customtopic.replications** = <Number of Brokers>

Starting Journey Engine

The Journey Engine depends on Kafka and Ignite. The lifetime of Ignite is managed by Engine but Kafka needs to be started and stopped along with Journey Engine in a specific sequence to avoid Data corruption

- Start Zookeeper
- Start Kafka brokers on all servers
- Start Journey Engine nodes in order
- The Engine node that is started first – the Master Node – should be the last node whilst shutting down. This is important for Ignite Cluster to avoid corruption of cached data

Chapter 6. Deploying Unica Journey application

You can deploy the Unica Journey web application by using a WAR file or you can deploy the individual WAR files.

To deploy Unica Journey, follow the guidelines in this section and then start the Unica Journey server.

Unica Journey web application needs to be deployed in separate tomcat instance. It must not be included in Unica Platform (unica.war) deployment application server profile (tomcat instance)

Recommendation while deploying journey.war in Tomcat, Webpshere and JBOSS application server

It is recommended to deploy journey.war in the application server which does not have unica.war deployed. User can deploy the journey.war in separate application server.

For Journey application to start it require Platform application to be up and running. If user deploy journey application and platform application in same JVM they would face problem while starting the application server.

Deploying Unica Journey on Apache Tomcat application server

About this task

You can deploy or run the following Journey components.

- Journey Web -It is required to be deployed in Tomcat.
- Journey Engine - It is run as a standalone application.
- Kafka Server - It is run as a standalone application (Kafka server and Zookeeper).

Use the following guidelines when you deploy Unica Journey on Tomcat:

- Unica products customize the JVM used by Tomcat. You must create a new tomcat instance dedicated for Unica Journey web application deployment.
- If you are deploying in a production environment, set the JVM memory heap size parameters to at least 1024 by adding the following line to the setenv.bat/sh e.g : set `CATALINA_OPTS=%CATALINA_OPTS% -Xms1024m -Xmx1024m -XX:MaxPermSize=512m`.

  These are the suggested minimum values. Analyze your sizing requirements to determine correct values for your needs. Depending upon system load **-Xmx** value should be adjusted. Note that a 64-bit application server and JVM are usually necessary for values greater than 2048.
- Modify the `JAVA_OPTIONS` parameter to add the following value in `setenv.bat/sh`.

  set `JAVA_OPTS=%JAVA_OPTS% -DUNICA_PLATFORM_CACHE_ENABLED=true -Dclient.encoding.override=UTF-8`.

  `-Djourney.web.home=<Journeys_Install_location>/Web/`
- You must add Unica Journey deployment XML file with name `journey.xml` along with the path of journey.war to the Unica Journey Tomcat instance. For example:

```
  <?xml version="1.0"?>
<Context docBase="<Journeys_Install_Path>/Web/journey.war">
<Environment name="journey.web.home" value="<Journeys_Install_Path>/Web/" type="java.lang.String"/>
<Resource name="JourneyDS" type="javax.sql.DataSource"
  factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
```

```
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>"
 driverClassName="oracle.jdbc.OracleDriver"
url="jdbc:oracle:thin:@<Host>:<Port>:<SID_NAME>"/>
<Resource name="JourneyReportDS" type="javax.sql.DataSource"
 factory="com.hcl.journey.tomcat.util.JourneyTomcatDSFactory"
maxActive="30" maxIdle="10" maxWait="10000"
username="<your_db_user_name>" password="<your_db_user_password>"
 driverClassName="oracle.jdbc.OracleDriver"
url="jdbc:oracle:thin:@<Host>:<Port>:<SID_NAME>"/>
</Context>
```

> **Note:**
> ◦ docBase = path should point to journey web war
> ◦ {{You can encrypt DB password using }} `JourneyEncryptionUtility.sh` located in
> `<Journey_Install_Path>/tools`

- Restart the Tomcat application server.

If AWS ELB (Elastic Load Balancing) is on HTTPS and Journey product is on HTTP, if Journey Swagger API page is not loading or Swagger API are not executing, user needs to Configure Tomcat for SSL offloading by adding below **parameters connector tag in server.xml**:

Example: /opt/Tomcat/Journey_instance/conf/server.xml

<Connector port="7010" protocol="HTTP/1.1"

connectionTimeout="20000"

scheme="https" secure="true"

redirectPort="9010" />

Cleaning the Apache Tomcat application server cache

1. Access the instance location used for Unica Journey. For example, `/opt/Tomcat/instance1`.
2. Delete the content of the folders webapps and work.

Guidelines for deploying Unica Journey on WebSphere

You must follow a set of guidelines when you deploy Unica Journey on WebSphere.

Make sure that the version of WebSphere® meets the requirements as described in the Recommended Software Environments and Minimum System Requirements document, including fix packs (if necessary). Following are the guidelines for deploying Unica Journey on WebSphere®:

- Deploy the journey.war File as an enterprise application. When you deploy the journey.war file, ensure that the JDK source level of the JSP compiler is set to Java 18 for SDK 1.8 and the JSP pages are precompiled as per the following information:
  - In the form where you browse and select the WAR file, select **Show me all installation options and parameters**, so that the **Select Installation Options** wizard runs.
  - In step 1 of the **Select Installation Options** wizard, select **Precompile JavaServer Page files**.
  - In step 3 of the **Select Installation Options** wizard, ensure that the **JDK Source Level** is set to 18 for SDK 1.8.
  - In step 8 of the **Select Installation Options** wizard, select **JourneyDS** as the matching Target Resource.
  - In step 10 of the **Select Installation Options** wizard, the context root must be set to `/journey`, (all lower case).
    - Click on Finish and wait for the Application to be installed.
    - In WebSphere Enterprise Applications, select **Your Application ex:** journey.war
- In the **Web Container, Settings > Web Container > Session Management** section of the server enable cookies. Specify a different session cookie name for each application that is deployed. Use one of the following procedures to specify a cookie name:
  - Select the **Override session management** check box under **Session Management**. If you deployed separate WAR files for your Unica products, in the WebSphere console under **Applications > Enterprise Applications > [deployed_application] > Session Management > Enable Cookies > Cookie Name** section of the server, then specify a unique session cookie name.
  - If your installation must support non-ASCII characters, for example for Portuguese or for locales that require multi-byte characters, add the following arguments to **Generic JVM Arguments** at the server level:

    -Dfile.encoding=UTF-8

    -Dclient.encoding.override=UTF-8

    Add the **Journey Web Home** path variable like below where we used to put properties & config folder:

    -Djourney.web.home=<Journeys_Home>/Web/

    In case of a production setup, this Java option must be removed or set to `false`.

    Navigation tip: Select **Servers > Application Servers > Java and Process Management > Process Definition > Java Virtual Machine > Generic JVM Arguments**.

Additional Details:
- In the **Applications > Enterprise Applications** section of the server, select the WAR file that you deployed, then select **Class loading and update detection** and specify the following properties.

- In the **Applications > Enterprise Applications** section of the server, select the EAR file or WAR file that you deployed, then select **Class loading and update detection** and specify the following properties.
  - If you are deploying a WAR file:
    - For **Class loader order**, select **Classes loaded with local class loader first (parent last)**.
    - For **WAR class loader policy**, select **Single class loader for application**.
- In WebSphere Enterprise Applications, select *Your Application* **> Manage Modules >** *Your Application* **> Class Loader Order > Classes loaded with local class loader first (parent last)**..
- The recommended minimum heap size for the basic functioning of the application is 512 and the recommended maximum heap size is 1024.

Complete the following tasks to specify the heap size:

1. In WebSphere® Enterprise Applications, select **Servers > WebSphere application servers > server1 > Server Infrastructure > Java and Process Management > Process definition > Java Virtual Machine**.
2. Set the initial heap size to 512.
3. Set the maximum heap size to 1024

See the WebSphere® documentation for more information about sizing.

Add a specific web container custom property:

1. Click Servers > Server Types > Application Servers, and select the server initially created.
2. Click Web Container Settings > Web container.
3. Click Custom properties.
4. Click New.
5. Enter the property values:

Property	Value
Name	com.ibm.ws.webcontainer.invokeFlushAfterService
Value	False
Description	See http://www.ibm.com/support/docview.wss?uid=swg1PM50111

6. Click OK.
7. Click Save.

✏️ **Note:** For deployment on WebSphere you need to import the HTTPS certificates. As Journey is integrated with Link and Deliver, if these applications are deployed on HTTPS then you need to import HTTPS certificates in WebSphere application server, otherwise journey will not be able to access the Link and Deliver.

In case if need how to import SSL certificate then following URL will help: https://www.ibm.com/support/knowledgecenter/en/SSEKCU_1.1.2.1/com.ibm.psc.doc/rs_original/installer/rs_t_import_client_cert_was.html

✏️ **Note:** If you are using OneDB database with WebSphere and OneDB has DB_LOCALE set to en_us.57372 then in WebSphere console also set the datasource Custom properties locale to ifxDB_LOCALE ="en_us.57372" and ifxCLIENT_LOCALE="en_us.57372".
Cleaning the Websphere application server cache

1. Go to WAS profile location used for Journey installation. For example `/data/webservers/IBM/`
   `WebSphere85_ND/profiles/UMP9111`
2. There you will see two folders tmp and wstemp.
3. Delete the contents of these two folders.

- Restart WebSphere

   Note: Once platform starts successfully then we need to start Journey application manually in Websphere
   server.

- Start your deployment for journey.war (Journey Application).

Guidelines for deploying Unica Journey on JBoss

V12.1 Fixpack 4 onwards we can deploy Unica Journey on JBoss. You must follow a set of guidelines when you deploy Unica Journey on JBoss.

Make sure that the version of JBoss meets the requirements that are described in the HCL Enterprise Products Recommended Software Environments and Minimum System Requirements document. Use the following guidelines when you deploy Unica Journey on JBoss:

Use the following guidelines when you deploy the Unica Journey products on any supported version of JBoss:

1. Deploy unica.war file as an enterprise application.

   For example: deploy <Journey_Install>\unica.war

   See https://docs.jboss.org/jbossweb/3.0.x/deployer-howto.html for instructions on Deploying Web Server Application in JBoss.
2. Complete the following tasks if your installation must support non-ASCII characters, for example for Portuguese or for locales that require multi-byte characters:
   a. Edit the `standalone.conf` script in the bin directory under your JBOSS /bin directory to add

      `-Dfile.encoding=UTF-8`

      `-Dclient.encoding.override=UTF-8`

      `-Djourney.web.home=C:/WAS_FEDFP4/Journeys/Web/`

      `-Djboss.as.management.blocking.timeout=3600`

      to `JAVA_VENDOR`.

      In case of a production setup, this Java option must be removed or set to `false`.
   b. Restart JBoss server.
3. To ensure that the Scheduler works correctly, complete the following substeps.
   - Take a backup of <JBOSS_HOME>/standalone/configuration/standalone.xml file.
   - In <JBOSS_HOME>/standalone/configuration/standalone.xml, search the module name for driver.

```
<driver name="oracledriver" module="oracle.jdbc">
<xa-datasource-class>oracle.jdbc.OracleDriver</xa-datasource-class>
</driver>
```

- Installer will not update the datasources, so you need to manually configure the datasources.
- Add the following statement underneath `<subsystem xmlns="urn:jboss:domain:ee:4.0">` to make the module name global.

```
<global-modules>
<module name="oracle.jdbc"/>
</global-modules>
```

- Restart JBOSS server.

Cleaning the JBOSS application server cache

a. Go to JBOSS install location location used for Journey installation. For example, `/jboss-eap-7.1/ standalone`

b. There you will see two folders tmp and deployments.

c. Delete the contents of these two folders

Note: Journey Web component reads configurations from application.properties file. Users are required to update the Journey Web application properties file and Journey Engine Application properties file before the deployment of journey.war. Please refer Configuring Unica Journey on page 32

Kafka authentication using Kerberos

Clients (producers, consumers, connect workers, etc) will authenticate to the cluster with their own principal (usually with the same name as the user running the client), so obtain or create these principals as needed. Then configure the JAAS configuration property for each client. Different clients within a JVM may run as different users by specifying different principals.

Before starting Kafka kerberos configuration with Journey:

From Kerberos hosted machine copy all the files available at location `/var/kerberos/krb5kdc` to Journey installation location `<Journey-HOME>/Journey/Web/properties`

From Kerberos hosted machine copy **krb5.conf** file available at location /etc to Journey installation location -> `<Journey-HOME>/Journey/Web/properties`

Note: Host name of `kafka_Hosted` machine is added on Unica application hosted machine and vice versa and make sure that Kafka communication working good between two machines.

1. **Enable the Kafka Security for Journey Engine and Journey WEB**

   - **For Journey Engine** - Please replace the krb5 file location to the `<JOURNEY_HOME>/Engine/ journey_master_config.properties`

     kafka.security.enabled=Y

     kafka.security.protocols.enabled=GSSAPI

- **For Journey WEB** - Please replace the krb5 file location to the `<JOURNEY_HOME>/WEB/properties/`
  `application.properties`

  kafka.security.enabled=Y

  kafka.security.protocols.enabled=GSSAPI

2. Make sure the keytabs configured in the JAAS configuration are readable by the operating system user who is starting kafka client.

   - **For Journey Engine** - Please replace the kafka_server_jaas.conf file location to the `<JOURNEY_HOME>/Engine/`
     `journey_master_config.properties`

     `java.security.auth.login.config = <jass LOCATION> example - /etc/kafka_server_jaas.conf`

   - **For Journey WEB** - Please replace the kafka_server_jaas.conf file location to the `<JOURNEY_HOME>/WEB/`
     `properties/application.properties`

     `java.security.auth.login.config = <jass LOCATION> example - /etc/kafka_server_jaas.conf`

3. Pass the krb5 file locations as JVM parameters to each client JVM

   - **For Journey Engine** - Please replace the krb5 file location to the `<JOURNEY_HOME>/Engine/`
     `journey_master_config.properties`

     `java.security.krb5.conf = <krb5 LOCATION> example - /etc/krb5.conf`

   - **For Journey WEB** - Please replace the krb5 file location to the `<JOURNEY_HOME>/WEB/properties/`
     `application.properties`

     `java.security.krb5.conf = <krb5 LOCATION> example - /etc/krb5.conf`

4. While configuring Kerberos with Kafka add/update below changes in below file:

   Location - `/<Journey-HOME>/Journey/KafkaStandalone/config/server.properties`

```
sasl.enabled.mechanisms=GSSAPI
sasl.mechanism.inter.broker.protocol=GSSAPI
security.inter.broker.protocol=SASL_PLAINTEXT
listeners=SASL_PLAINTEXT://0.0.0.0:9092
advertised.listeners=SASL_PLAINTEXT://<Kafka_Kerberos_principle_host_name>:9092
#E.g.
 advertised.listeners=SASL_PLAINTEXT://ip-10-10-10-100.ap-south-1.compute.internal.nonprod.hclpnp.com:90
92
listener.name.sasl_plaintext.gssapi.sasl.jaas.config=com.sun.security.auth.module.Krb5LoginModule
 required useKeyTab=true storeKey=true
keyTab="/var/kerberos/krb5kdc/kfkserver.keytab" principal="kafka/<Kafka_Kerberos_principle_host_name>";
sasl.kerberos.service.name=kafka
#e.g. keyTab="/var/kerberos/krb5kdc/kfkserver.keytab"
 principal="kafka/ip-10-10-10-100.ap-south-1.compute.internal.nonprod.hclpnp.com";
sasl.kerberos.service.name=kafka
```

   📝 **Note:** This value will change as per your kafka principle

   `ip-10-10-10-100.ap-south-1.compute.internal.nonprod.hclpnp.com@HCLPNP.COM`

5. Add Kafka kerberos host name in Journey Engine and Journey WEB application.properties file

   `spring.kafka.bootstrap-servers=<Kafka_Kerbersos_Host_Name>:9092`

6. Create folders by name -> **krb5.conf.d** and **log** at location `/<Journey-HOME>/Journey/Web/properties`

7. update file `kafka_server_kerberos_jaas.conf` OR `equivalent Kafka Kerberos client conf` available at location `/<Journey-HOME>/Journey/Web/properties` with applicable kerberos details and recent paths as per Journey installation location

```
keyTab="/var/kerberos/krb5kdc/zkserver.keytab"
principal="zookeeper/ip-10-10-10-10.ec2.internal.nonprod.hclpnp.com";
keyTab="/var/kerberos/krb5kdc/kfkserver.keytab"
principal="kafka/ip-10-10-10-10.ec2.internal.nonprod.hclpnp.com"
```

8. update file **krb5.conf** available at location `/<Journey-HOME>/Journey/Web/properties` with applicable kerberos details and recent paths as per Journey installation location:

```
includedir /<Journey-HOME>/Journey/Web/properties/krb5.conf.d/
 [logging]
 default = FILE:/<Journey-HOME>/Journey/Web/properties/log/krb5libs.log
 kdc = FILE:/<Journey-HOME>/Journey/Web/properties/log/krb5kdc.log
 admin_server = FILE:/<Journey-HOME>/Journey/Web/properties/log/kadmind.log
```

JBOSS Setting for Kerberos Kafka Authentication

The below changes has been done in Jboss standalone.xml file to before start the Journey web:

1. Go to `<JBOSS_HOME>\standalone\configuration\`
2. Open standalone.xml
   a. **Confirm the servlet-container node** must have all the below values. If not please replace the node with below provided set of lines

   ```
   <servlet-container name="default" disable-caching-for-secured-pages="false">
   <jsp-config/>
   <websockets/>
   </servlet-container>
   ```

   b. **kafka_server_kerberos_jaas.conf and krb5.conf file path configure in standalone.xml as mention below**

   ```
   <system-properties>
    <property name="java.security.auth.login.config"
    value="<Journey-HOME>/Journey/Web/properties/kafka_server_kerberos_jaas.conf"/>
    <property name="java.security.krb5.conf"
    value="<Journey-HOME>/Journey/Web/properties/krb5.conf"/>
    <property name="java.security.krb5.debug" value="true"/>
    <property name="java.security.disable.secdomain.option" value="true"/>
    </system-properties>
   ```

   c. **Add the KafkaClient and KafkaServer conf in standalone.xml as shown below**

```
<security-domain name="KafkaClient" cache-type="default">
 <authentication>
 <login-module code="com.sun.security.auth.module.Krb5LoginModule" flag="required">
 <module-option name="storeKey" value="true"/>
 <module-option name="useKeyTab" value="true"/>
 <module-option name="refreshKrb5Config" value="true"/>
 <module-option name="principal" value="<CLIENT-PRINCIPLE>"/> example -
 "kafka/ip-10-10-10-10.ec2.nonprod.hclpnp.com"
 <module-option name="keyTab" value="<Journey-HOME>/Journey/Web/properties/kfkserver.keytab"/>
 <module-option name="doNotPrompt" value="true"/>
 </login-module>
 </authentication>
</security-domain>
<security-domain name="KafkaServer" cache-type="default">
```

```
<authentication>
<login-module code="com.sun.security.auth.module.Krb5LoginModule" flag="required">
<module-option name="storeKey" value="true"/>
<module-option name="useKeyTab" value="true"/>
<module-option name="refreshKrb5Config" value="true"/>
<module-option name="principal" value="<Server-PRINCIPLE>"/> example -
"zookeeper/ip-10-10-10-10.ec2.nonprod.hclpnp.com"
<module-option name="keyTab" value="<Journey-HOME>/Journey/Web/properties/zkserver.keytab"/>
<module-option name="doNotPrompt" value="true"/>
</login-module>
</authentication>
</security-domain>
```

> **Note:** The values of the above mentioned parameter keyTab and principle must be same which has been generated at the time of Kerberos Server setup.

3. After saving the above changes in `Jboss <JBOSS_HOME>\standalone\configurationstandalone.xml`, Journey Web can be started.After configuring Journey with Kafka Kerberos, on Journey while creating Rest Entry source if application gives error like below user needs to verify node configuration of Confirm the **servlet-container** node as below.

Errors: Unable to fetch rest Api Or Unable to fetch Kafka

- Confirm the **servlet-container** node must have all the below values. If not please replace the node with below provided set of lines at below location:

```
<JBOSS_HOME>\standalone\configuration\standalone.xml
<servlet-container name="default" disable-caching-for-secured-pages="false">
<jsp-config/>
<websockets/>
</servlet-container>
```

WAS Setting for Kerberos Kafka Authentication

There are 2 files which needs to set in the environment of WebSphere.

1. Set krb5 in System Env

   For Krb5 Server > Server Type > Select the server > Java and Process Management > Process Definition > Java Virtual Machine > Generic JVM arguments append the below mentioned argument in the Argument List

   -Djava.security.krb5.conf=<KRB File Location>\krb5.conf

2. jass entries

   Go to <WebSphere Home>\AppServer\profiles\<Profile>\properties\wsjaas.conf

   Append the following entries into the file

   KafkaServer

```
{ com.ibm.security.auth.module.Krb5LoginModule required useKeytab="<Zookeper Key Tab Path>
\zkserver.keytab" storeKey=true principal= <Zookeper Principle> eg- "zookeeper/ip-10-10-10-10.ap-
south-1.compute.internal.nonprod.hclpnp.com" credsType = both; }
```

   KafkaClient

```
{ com.ibm.security.auth.module.Krb5LoginModule required useKeytab="<Kafka Server Kay
Tab Path>\kfkserver.keytab" principal= <Kafka Principle> eg - "kafka/ip-10-10-10-10.ap-
south-1.compute.internal.nonprod.hclpnp.com" credsType = both debug=true; }
```

Restart Application Server.

Configuring Kafka Kerberose for Deliver Responses:

Login to Unica application and navigate to below location:

1. **Location > Settings > configuration > HCL Unica > Journey > Kafka > Configurations**

   Set:

   ```
   KafkaBrokerURL:      <Priciple_Hostname>:9092
   CommunicationMechanism:     GSSAPI
   sasl.mechanism: GSSAPI
   sasl.jass.config.location:
    <JOURNEY_HOME>/WEB/properties/application.properties/kafka_server_Kerberose_jaas.conf
   java.security.krb5.conf.location: <JOURNEY_HOME>/WEB/properties/application.properties/krb5.conf
   ```

   .

2. Location: **Settings > configuration > HCL Unica > Deliver > Kafka > RCT**

   Set:

   ```
   Set
   KafkaBrokerURL:      <Priciple_Hostname>:9092
   CommunicationMechanism:     SASL_PLAINTEXT
   sasl.mechanism: GSSAPI
   ```

Kafka Configuration for Configuring External resources in Journey (AssetPicker and Journey configuration)

Login to Unica application and navigate to below location:

Location > Settings > configuration > HCL Unica > Journey > Integration > dataSource > <System_Configuration Template Name> > Kafka Configurations

```
Bootstrap servers (comma separated list of hosts)          <Karberos_Kafka_Principle_Host:>7001
(e.g. ip-10-10-10-100.nonprod.hclpnp.com:7001){}
Security protocol            SASL_PLAINTEXT
SASL mechanism            KERBEROS
Kerberos - Configuration file path (e.g. /etc/krb5.conf, C:/Windows/krb5.ini)
 <JOURNEY_HOME>/WEB/properties/application.properties/krb5.conf
Kerberos - Keytab file path            <JOURNEY_HOME>/WEB/properties/kfkserver.keytab
Kerberos - Principal      kafka/<Karberos_Kafka_Principle_Host>@HCLPNP.COM
e.g. kafka/ip-10-10-10-100.nonprod.hclpnp.com@HCLPNP.COM
Kerberos - Service Name            kafka
```

Chapter 7. Uninstalling Unica Journey

Run the Unica Journey uninstaller to uninstall Unica Journey. When you run the uninstaller, the files that were created during the installation process are removed. For example, files such as configuration files, installer registry information, and user data are removed from the computer. Stop the processes that are related to Unica Journey before uninstallation.

About this task

When you install Unica products, an uninstaller is included in the `Uninstall_Product` directory, where *Product* is the name of your product. On Windows™, an entry is also added to the **Add or Remove Programs** list in the Control Panel.

If you manually remove the files in your installation directory instead of running the uninstaller, the result might be an incomplete installation if you later reinstall a product in the same location. After uninstalling a product, its database is not removed. The uninstaller only removes default files that are created during installation. Any file that is created or generated after installation is not removed.

> **Note:** On UNIX™, the same user account that installed Unica Journey must run the uninstaller.