HCLSoftware

Unica Deliver V12.1.7.1 Startup and Administrator's Guide

Contents

Chapter 1. Hosted messaging using Unica Campaign and Unica Deliver

When Unica Campaign is integrated with Unica Deliver, you can use Deliver to conduct highly personalized digital marketing campaigns.

📝 **Note:** Deliver supports the following channels along with email. In this guide, the term message applies to all channels.

- SMS
- WhatsApp
- Push

Deliver provides access to resources hosted by Unica so you can design, send, and monitor individually customized messages based on information stored in your customer datamart.

- In Campaign, use flowcharts to create lists of message recipients and select personalization data for each recipient.
- In Deliver, use message design, transmission, and deliverability resources hosted by HCL to conduct digital marketing campaigns.

Establishing a hosted account with Unica

When you purchase a message subscription, Unica creates a hosted account on your behalf and sends you the account credentials that you will need to use message features. You apply these credentials when you configure your local HCL Unica applications to access the hosted environment over secure connections.

You must have a valid account to access the message resources that Unica provides as a software service. If your HCL Unica installation includes multiple partitions and you plan to use message in more than one partition, you need a hosted account and atleast one service provider for SMS, Push, and Whatsapp depending on services you need for each partition. You cannot share accounts across installations or partitions.

Establishing a hosted account is the beginning of the startup process, which lasts for approximately 90 days. You can subscribe to SMS, Push and Whatsapp depending on services you need as per the requirements. For a general description of the process, see the next topic.

Overall view of the startup process

You can activate message features in Unica Campaign to conduct highly targeted and trackable digital marketing campaigns. Campaign uses message functions that are provided by Unica Deliver through resources that are hosted in data centers in the US, India, and Europe. An account to access the email resources is included with your Deliver subscription. You can also opt for the WhatsApp, Push, or SMS channels as per the requirements.

Unica begins the startup process after it creates your hosted email account. Unica helps you to become familiar with Deliver, connect to message resources, and establish your reputation as a legitimate digital marketer among leading Internet Service Providers (ISPs).

The process proceeds in three phases. The Unica Professional Services and Deliver Services teams guide you along the way.
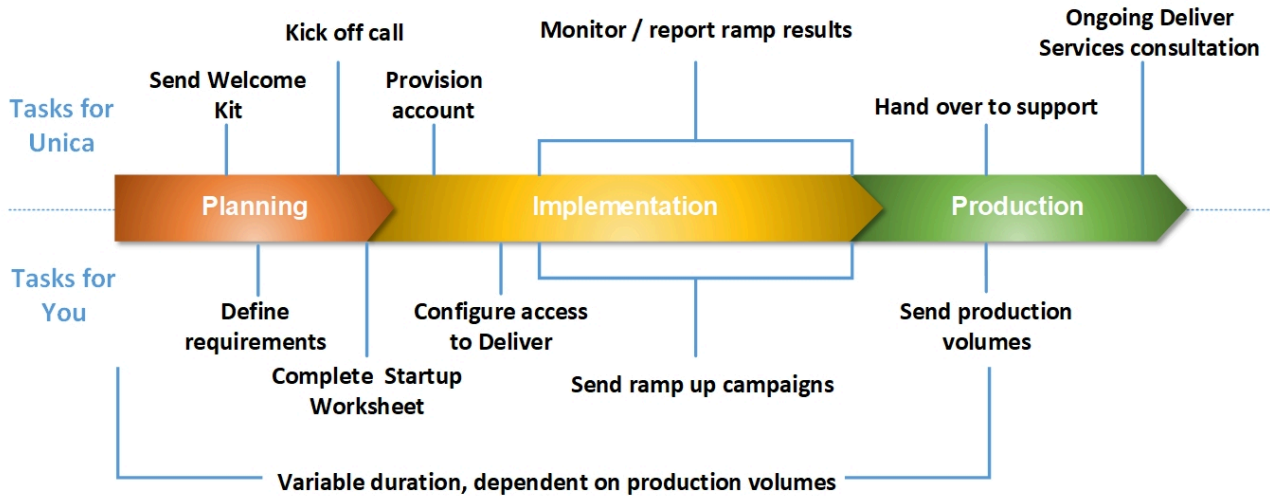
The Professional Services consultant is your primary point of contact with Unica during the startup process. When the account startup process completes, the Professional Services consultant transfers primary support responsibility to the Unica Product Support team.

A dedicated Deliver Services consultant provides special assistance for message-related issues. Creating a favorable message reputation among major Internet Service Providers (ISPs) is critical to ensuring that your digital marketing campaigns consistently reach their target recipients. When you start to run mailings, the EAS consultant reviews the mailing deliverability performance and suggests the best ways to gradually build your message reputation.

Startup activities and milestones

Planning

	What happens	Who is responsible
	Send email account credentials and the Welcome Kit, including the Email Startup Worksheet.	Unica Deliver Services
	Schedule a conference call to introduce all involved parties, review the startup schedule, and understand the email marketing objectives.	Unica Professional Services
	Complete the Email Startup Worksheet to specify your email domain requirements and mailing projections.	Your organization

Create your email reputation

What happens	Who is responsible
Provision the email account using information provided during the conference call and in the Email Startup worksheet.	Unica email operations
Begin warm up mailings to selected test accounts with major ISPs. This phase requires approximately 30 days to complete.	Unica email operations
Activate Deliver in Unica Campaign.	Your organization (with support from Unica)
Configure access to hosted email resources. Consult with the EAS consultant about which data center to specify.	Your organization (with support from Unica)
Start sending mailings. To build a favorable email reputation, send small mailings initially, followed over time by larger and more frequent mailings. ISPs often try to limit spam by blocking large or frequent mailings from email domains they do not recognize as legitimate.	Your organization (with support from Unica)
Provide deliverability results and reputation guidance as mailing volumes and frequency gradually increase.	Unica Deliver Services

Production

What happens	Who is responsible
Send mailings at typical volume and frequency.	Your organization
Transfer primary contact responsibility to the Unica Support team.	Unica Professional Services
Maintain engagement for consultation on email issues. Make contact on a regular basis for continuing email account support.	Unica Deliver Services

Before you begin working with Deliver

Before you begin the messaging startup process, consider the following issues.

- Some configurations require restarting the web application server. Plan Deliver configuration activity to avoid interfering with large flowchart runs and other activities in Campaign.
- Unica asks you to name one individual to serve as the primary contact point during the startup process.
- Request hosted email account credentials before you begin the startup process. You use these credentials to configure your systems to access the account.

- Consult with your network administration staff. Deliver requires specific port ranges when communicating with HCL Unica.
- Confirm that you have the appropriate network permissions to make configuration changes.

Chapter 2. Configuring the local HCL Unica environment for Deliver

Using Deliver to send messages requires changes in the local installation of HCL Unica. Complete the steps described in the following sections.

If your environment contains multiple partitions, repeat these steps for each Campaign partition in which you use Unica Deliver. For more information about creating and working with multiple partitions, see the *Unica Campaign Installation Guide*.

Confirming Deliver registration

Unica Deliver must be registered with Unica Platform. To confirm that Deliver is registered successfully, you must examine the configuration for Platform.

1. Log in to HCL Unica.
2. Navigate to **Settings** > **Configuration**.
3. Look for the Deliver configuration category.

   Unica Deliver is registered with Platform when the Deliver category appears in the configuration properties hierarchy.

What to do next

If the Deliver category does not appear in the properties hierarchy, see the *Unica Campaign Installation Guide* for information about how to register Deliver manually.

If the Deliver category is available, you must enable Deliver functions in Campaign.

Registering Deliver manually

During the installation process if the Deliver installer cannot access the Platform system tables, you must run the configTool utility to register it manually.

About this task

By default, the Campaign installer automatically registers Deliver with the Platform system tables without enabling Deliver. In some situations, the Campaign installer does not connect with the Platform system tables to automatically register Deliver.

If the installer does not register Deliver automatically, you must register Deliver manually with the `configTool` utility that is provided with the HCL Unica installation. The `configTool` utility is in the `tools\bin` directory under your Platform installation.

To register Deliver manually, use the following command to run the `configTool` utility:

configTool -r Deliver -f "*full_path_to_Deliver_installation_directory*\conf\Deliver_configuration.xml"

The Deliver installation directory is a subdirectory of the Campaign installation directory.

Enable Deliver functions in Campaign

About this task

When you install Campaign, the installer also installs Deliver in the default partition, but does not enable it. Deliver functions are not available until you enable Deliver.

You enable Deliver with the following configuration property, in the Unica Platform.

```
Campaign > partitions > partition[n] > server > internal > deliverInstalled
```

To enable Deliver, change the value to `yes`.

What to do next

Registration requirement

Registering Deliver with Unica Platform is required to operate Deliver. You register Deliver with Platform when you install Unica Campaign.

After you have enabled Deliver, confirm that Deliver is properly registered with Unica Platform. For details, see Confirming Deliver registration on page 5.

Displaying Deliver menu options

To use Unica Deliver, you must update the system configuration so that menu options for Deliver display in the Unica Platform interface. When you install Campaign, the installer also installs Deliver menus in the default partition. In case Campaign installer do not connect with the Platform system tables, you must configure them manually using the following steps. To display the required options, use the `configTool` utility that is supplied with your HCL Unica installation.

You must run `configTool` with specific parameters for each Deliver menu option. Running `configTool` updates system configuration settings. You must restart the web application server to apply the changes. Although Deliver is installed with Campaign, menu options for Deliver do not appear until after you run `configTool` and restart the web application server. In the `tools` directory of the Platform installation, the `configTool` utility is located in the `bin` folder.

> **Note:** You must specify a path to the Deliver installation directory as a `configTool` parameter. The Deliver installation directory is a subdirectory of the Campaign installation directory.

- To display **Deliver Settings** in the **Settings** menu.

  ```
  configTool.bat -v -i -p "Affinium|suite|uiNavigation|settingsMenu" -f
  "full_path_to_Deliver_installation_directory\conf\deliver_op_odsettings_navigation.xml"
  ```
- To display **Deliver Mailings** in the **Campaign** menu.

```
configTool.bat -v -i -p "Affinium|suite|uiNavigation|mainMenu|Campaign" -f
```

```
"full_path_to_Deliver_installation_directory\conf\deliver_op_mailings_navigation.xml"
```

- To display **Quick Builder** in the **Campaign** menu.

```
configTool.bat -v -i -p "Affinium|suite|uiNavigation|mainMenu| Campaign" -f
```

```
"full_path_to_Deliver_installation_directory\conf\deliver_op_new_documents_navigation.xml"
```

- To display **Deliver Documents** in the **Campaign** menu.

```
configTool.bat -v -i -p "Affinium|suite|uiNavigation|mainMenu|Campaign" -f
```

```
"full_path_to_Deliver_installation_directory\conf\deliver_op_documents_navigation.xml"
```

- To display **Deliver Analytics** in the **Analytics** menu.

```
configTool.bat -v -i -p "Affinium|suite|uiNavigation|mainMenu|Analytics" -f
```

```
"full_path_to_Deliver_installation_directory\conf\deliver_op_analytics_navigation.xml"
```

To verify that you successfully added the menu options, after you restart the web application server, log in to HCL Unica and open the **Settings**, **Campaign**, and **Analytics** menus to verify that the Deliver options appear.

Specifying Deliver system table characteristics

Unica Deliver requires information that describes the type, schema, and JDBC connection for the Deliver system tables in your installation. The Deliver system tables are created in the Campaign schema as part of the Campaign installation process.

- Navigate to **Settings > Configuration > Deliver > Partitions > Partition[n] > Datasources > System tables**.
- Review and update information for the following parameters.

> 📝 **Note:** The Campaign system table information is expected here.

Database Type

Schema Name

jdbcBatchSize

jdbcClassName

jdbcURI

asmDataSourceForDBCredentials - must be UA_SYSTEM_TABLES

- Provide the required information in the following configuration properties.

See the Platform online help for each property to learn more about setting the configuration properties.

   ○ Deliver > partitions > partition [n] < dataSources > systemTables > type

   ○ Deliver > partitions > partition [n] < dataSources > systemTables > schemaName

   ○ Deliver > partitions > partition [n] < dataSources > systemTables > jdbcBatchSize

   ○ Deliver > partitions > partition [n] < dataSources > systemTables > jdbcClassName

   ○ Deliver > partitions > partition [n] < dataSources > systemTables > jdbcURI

What to do next

For additional information about configuration properties and configuring Deliver, see Configurations for Unica Deliver on page 64.

Configuring access to local Deliver system tables

Unica Deliver requires access to the Deliver system tables in the Campaign schema. To allow Unica Deliver components to access system tables in the Campaign schema without requesting a manual database login, you must specify an Deliver system user to provide the necessary database access credentials.

About this task

The system user that accesses the database is associated with an Unica Platform data source that contains the login credentials for the database that hosts the Campaign schema.

For more information about system table configuration properties, see Deliver | partitions | partition[n] | dataSources | systemTables on page 78.

1. Specify the system user that you defined in the Unica Platform. Edit the following configuration property.

   ```
   Deliver > partitions > partition [n] < dataSources > systemTables > asmUserForDBCredentials
   ```
2. Specify the login credentials for the database that contains the Campaign schema and Deliver system tables. Edit the following configuration property.

   ```
   Deliver > partitions > partition [n] < dataSources > systemTables > amDataSourceForDBCredentials
   ```

Required mapping for Deliver system tables in Campaign

You must map Deliver system tables in the Campaign schema to corresponding Deliver database tables. The Deliver system tables have **Deliver** in the table name.

In Campaign, map the following Deliver system tables.

- Deliver Output List Table
- Deliver Output List Audience Fields Mapping Table
- Deliver Mailing Table
- Deliver Mailing Instance Table
- Deliver Data Table Column Mapping Table
- Deliver Personalization Field Mapping Table
- Deliver Personalization Field Usage Table

For information about mapping tables, see the *Unica Campaign Administrator's Guide*.

Required restart of the web application server for Campaign

After making changes to the Campaign and Deliver configurations, you must restart the web application server that hosts Campaign.

Consult the documentation for your web application server for restart instructions.

Chapter 3. Connections to message services

To access message services provided by Unica, you must configure a connection between the local HCL Unica installation and HCL Unica hosted services.

Marketers access Deliver features through the Campaign interface. Working with Deliver requires that you establish a secure, automatic Internet connection that Campaign can use to upload message recipient lists to HCL Unica hosted services. Deliver components installed with Campaign also use this connection to download contact and response data to the Deliver system tables in the Campaign schema.

> ✏️ **Note:** Each instance of Campaign requires a unique connection to HCL Unica. If the Campaign installation includes multiple partitions, each partition requires a separate hosted account. The accounts can share the IP connection to HCL Unica.

All communication between HCL Unica and HCL Unica hosted services is done over SSL. Each communication from HCL Unica hosted services is a response to a request from the local environment. HCL Unica hosted services never attempts to initiate a connection with your corporate network. All communication with HCL Unica hosted services originates from behind your corporate firewall.

Requirements for configuring connection to HCL Unica hosted services

Configuring a connection to HCL Unica hosted services requires administrative permissions and information about the hosted account established for your organization.

To configure a hosted email connection, you need the following.

- User name and password provided by Unica for the hosted account
- Permissions to create or modify system users in the Unica Platform
- Administrative access to configuration properties maintained in the local Unica Platform installation
- Administrative access to the web application server on which the Unica Platform and Campaign are deployed

You must know, or be able to consult with people who know, your corporate data security requirements. Before you begin, review these procedures to understand how to create the necessary connection in compliance with your corporate firewall restrictions.

You must be familiar with how to configure trusted connections on your web application server, IBM WebSphere®, Oracle WebLogic, Apache Tomcat, and JBOSS.

Configuring addresses for connecting to HCL Unica hosted services

To ensure proper connection to HCL Unica hosted services, you must enter the addresses as values for configuration properties in the Deliver configuration. The connection addresses that you enter depend on whether you are connecting to the Unica data center in US, Europe or India.

Before you begin

Consult with Unica to confirm which data center your hosted email account uses.

In Unica Platform, navigate to **Settings > Configuration**. In the Deliver configuration, navigate to the following Deliver configuration properties and confirm or update the connection settings, depending on data center your account uses.

- `Deliver > serverComponentsAndLocations > hostedServices> uiHostName`

  To connect to Unica data center in US, change this value to `em.unicadeliver.com`.

  To connect to the Unica data center in Europe, change this value to

  `em-eu.unicadeliver.com`.

  To connect to the Unica data center in India, change this value to

  `em-in.unicadeliver.com`

- `Deliver > serverComponentsAndLocations > hostedServices> dataHostName`

  To connect to Unica data center in US, change this value to `em.unicadeliver.com`.

  To connect to the Unica data center in Europe, change this value to

  `em-eu.unicadeliver.com`.

  To connect to the Unica data center in India, change this value to

  `em-in.unicadeliver.com`

- `Deliver > serverComponentsAndLocations > hostedServices> ftpHostName`

  To connect to Unica data center in US, change this value to `ftp-em.unicadeliver.com`.

  To connect to the Unica data center in Europe, change this value to

  `ftp-eu.unicadeliver.com`.

  To connect to the Unica data center in India, change this value to

  `ftp-in.unicadeliver.com`

What to do next

If you change a configuration property, restart the web application server to apply the changes.

IP address of Deliver host names

If you require to whitelist IP address for Deliver host names on your corporate firewall, use the following IP addresses.

`em.unicadeliver.com`: `13.248.215.130` and `76.223.84.165`

`em-eu.unicadeliver.com`: `75.2.15.173` and `99.83.137.137`

`em-in.unicadeliver.com`: `75.2.92.153` and `99.83.224.139`

`ftp-em.unicadeliver.com`: `192.190.152.236`

`ftp-eu.unicadeliver.com`: `192.190.153.236`

`ftp-in.unicadeliver.com`: `192.175.4.236`

`tms-us.unicadeliver.com`: `13.248.172.132` and `76.223.38.158`

`tms-eu.unicadeliver.com`: `75.2.31.132` and `99.83.164.171`

`tms-in.unicadeliver.com:` `15.197.234.141` and `3.33.199.128`

Requirements for uploading data to HCL Unica hosted services

A Deliver component called the Recipient List Uploader (RLU) is part of your Unica Campaign installation. The RLU uses SFTP as a preferred mechanism to manage the upload of recipient lists and associated metadata to HCL Unica hosted services.

Deliver uses SFTP to upload data. When using SFTP, the RLU initiates all upload connection requests as the local client. HCL Unica hosted services never initiates a connection request to your network.

Connection and port requirements

To communicate with HCL Unica hosted services, you must have a connection to the Internet. HCL Unica hosted services use specific ports.

The local HCL Unica installation and HCL Unica hosted services use the following ports to communicate.

HTTPS: port 443

SFTP port: port 2222

HCL Unica hosted services never initiates a connection with your local network. It only responds to connection requests initiated from behind your firewall.

Whitelisting IP

To upload the recipient list (OLT) to FTP Deliver server, the external IP of the server on which Campaign Web is running must be whitelisted at Deliver On Demand server side.

You must get the external IP using the following commands and provide it to the Onboarding team. The Onboarding team will request to whitelist the IP so that FTP requests from your server to FTP Deliver server is allowed.

- On Unix system, run `curl ifconfig.me` command to get the external IP of your server.
- On Windows system, you can access `http://ifconfig.me` to get the external IP of your server.

Upload connection by SFTP

The Recipient List Uploader (RLU) uses SFTP protocol as a preferred mechanism to securely upload recipient lists. The RLU establishes a connection with HCL Unica hosted services over SFTP port 2222. Over the secure connection, the RLU negotiates authentication details with the SFTP server and uploads recipient list once authentication is successful.

The following diagram illustrates this method for uploading recipient data from Campaign to HCL Unica hosted services.
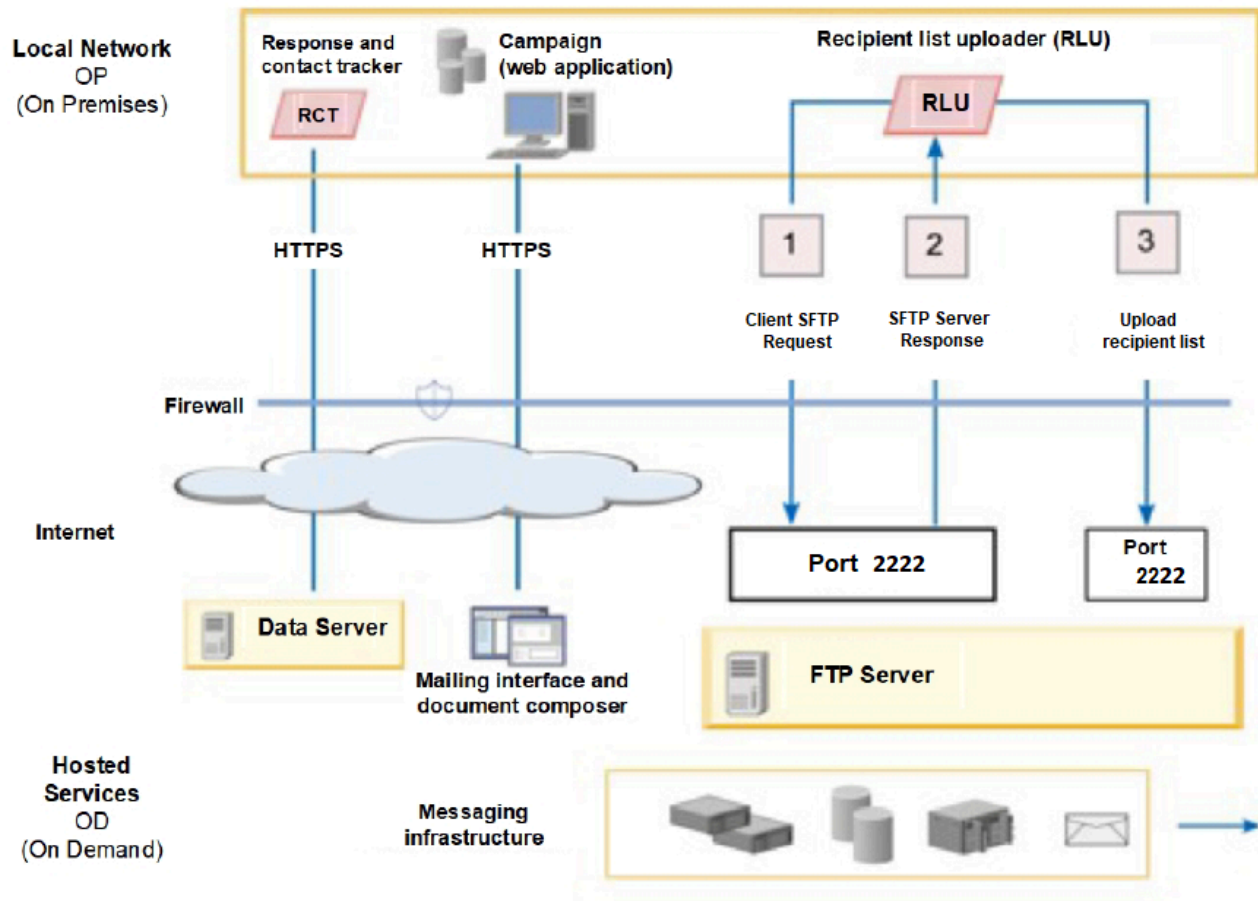
In the Configuration page, you can view the SFTP option under ftpProtocol (serverComponentsAndLocations -> hostedServices).

The RLU connects to SFTP server and uploads the recipient list to SFTP server. A certificate-based authorization is used to connect to auth server. It uses the SSH private key configured in PEM file and SSH RSA fingerprint configured in known_hosts file to establish connection with SFTP server. Customers are required to configure separate PEM file per Deliver account, so that it can be configured separately for each Deliver partition. In addition to the PEM file, RLU requires known_hosts file containing SSH server fingerprint.This is configured globally at the following path `Affinium|Deliver|serverComponentsAndLocations|hostedServices`. and a flag to control whether RLU requires a pre-configured SSH server fingerprint in known_hosts file.

Once authenticated, recipient list uploads are done using SFTP and there is no impact to Deliver process box execution from where this will be triggered.

In case if public or private keys are generated using `passPhrase`, create a new datasource with name "`SFTP_PASSPHRASE_DATASOURCE`" under Platform user specified at "`amUserForAcctCredentials`". For example: asm_admin and specify same password or `Passphrase` to this datasource which you have used while generating public or private keys. Data source login can be mentioned as any text.

If public or private keys are not generated using `passPhrase`, the datasource must not be created.

To generate keys, perform the following steps.

Steps to generate public / private key pair for SFTP authentication.

1. Create the Key Pair

   The first step is to create a key pair on the machine, where Campaign web is installed.

   Log in to the machine, where Campaign web is installed. Open the command prompt and execute the following command:

   ```
   ssh-keygen -m PEM
   ```

2. Specify the location to save the keys.

   You can press ENTER here to save the files to the default location in the .ssh directory of your home directory. Alternately, you can choose another file name or location by typing it after the prompt and pressing ENTER.

3. Create a passphrase.

   The second and final prompt from ssh-keygen will ask you to enter a passphrase. It depends on your requirements, whether you want to use a passphrase or not.

   For example

   ```
   [root@Host bin]# ssh-keygen
   Generating public/private rsa key pair.
   Enter file in which to save the key (/root/.ssh/id_rsa):
   Enter passphrase (empty for no passphrase):
   Enter same passphrase again:
   Your identification has been saved in /root/.ssh/id_rsa.
   Your public key has been saved in /root/.ssh/id_rsa.pub.
   The key fingerprint is:
   61:ca:14:c2:7a:71:e2:aa:bd:2e:ff:25:b8:b1:fd:ac root@Host
   The following is the key's randomart image.
   .. .
   +...
   o +. o
   . oo o .
   o o S
   ..
   oo . .
   o .= +
   +*oEoo
   [root@Host bin]#
   ```

Your public key is saved in – `/root/.ssh/id_rsa.pub`. Send the public key generated to Deliver Dev Ops team through HCL support for configuration.

Configuring SFTP

To configure SFTP, complete the following steps.

1. Execute the following command by navigating to `<Deliver_Home>/tools` from command prompt to expose ftpProtocol property on UI.

   ```
   ./switch_config_visibility.sh / bat -p "Affinium|Deliver|serverComponentsAndLocations|hostedServices|
   ftpProtocol" -v true
   ```

2. Log in to Platform and navigate to **Settings > Configuration** and select **SFTP** for ftpProtocol at `Affinium|Deliver|serverComponentsAndLocations|hostedServices`.

3. Execute the following command by navigating to `<Deliver_Home>/tools` from command prompt to expose ftpPort property on UI.

   ```
   ./switch_config_visibility.sh / bat -p "Affinium|Deliver|serverComponentsAndLocations|hostedServices|
   ftpPort" -v true
   ```

4. Mention port number as `2222` for ftpPort at `Affinium|Deliver|serverComponentsAndLocations|hostedServices`.

5. Keep the value for `enforceKnownHostsValidation` to `false`, update the path as `<Deliver_HOME>/Conf/known_hosts` for `knowHostsPath` property.

   For example: `knowHostsPath - /opt/HCL/Campaign/Deliver/conf/known_hosts`

   `enforceKnownHostsValidation - False`

6. Optional. In case you have the known_hosts file, update its complete path for `knowHostsPath` property at `Affinium|Deliver|serverComponentsAndLocations|hostedServices` and set `enforceKnownHostsValidation` to `true`.

7. Copy private certificate file (id_rsa) to `<DELIVER_HOME>/conf` and update the complete Path for this private certificate file in `pemFilePath` property at `Affinium|Deliver|partitions|partition1|hostedAccountInfo`.

   For example

   pemFilePath – `/opt/HCL/Campaign/Deliver/conf/id_rsa`

   amDataSourceForSftpPassPhrase-- `SFTP_PASSPHRASE_DATASOURCE`

8. In case you had specified passphrase, while generating public/private keys, create a datasource with name `SFTP_PASSPHRASE_DATASOURCE` under Platform user specified at `amUserForAcctCredentials` (example: asm_admin) and specify same password / passphrase to this datasource, which you have used while generating public/private keys. Datasource login can be mentioned as any text.

9. In case you did not specify any passphrase while generating public/private keys, you are not required to create this datasource `SFTP_PASSPHRASE_DATASOURCE` to asm_admin user or any other user.

10. Restart App server for Campaign.

11. Open command prompt, navigate to `<Deliver_home>/bin` and test SFTP connectivity using rlu , as following.

    ```
    rlu.sh / bat -c
    ```

Note: All file paths specified above must be complete including the file Name. Repeat Steps 7 to 9 for all the partitions, where Deliver is configured.

Connection through an HTTP proxy

If your corporate firewall rules prohibit direct communication with the public Internet, you can connect to HCL Unica through an HTTP proxy server. Deliver supports connecting through a SOCKS proxy server that allows both HTTPS and SFTP traffic.

Deliver supports SOCKS Protocol Version 5.

The following diagram illustrates communication between the local and hosted environments when you use a SOCKS proxy.
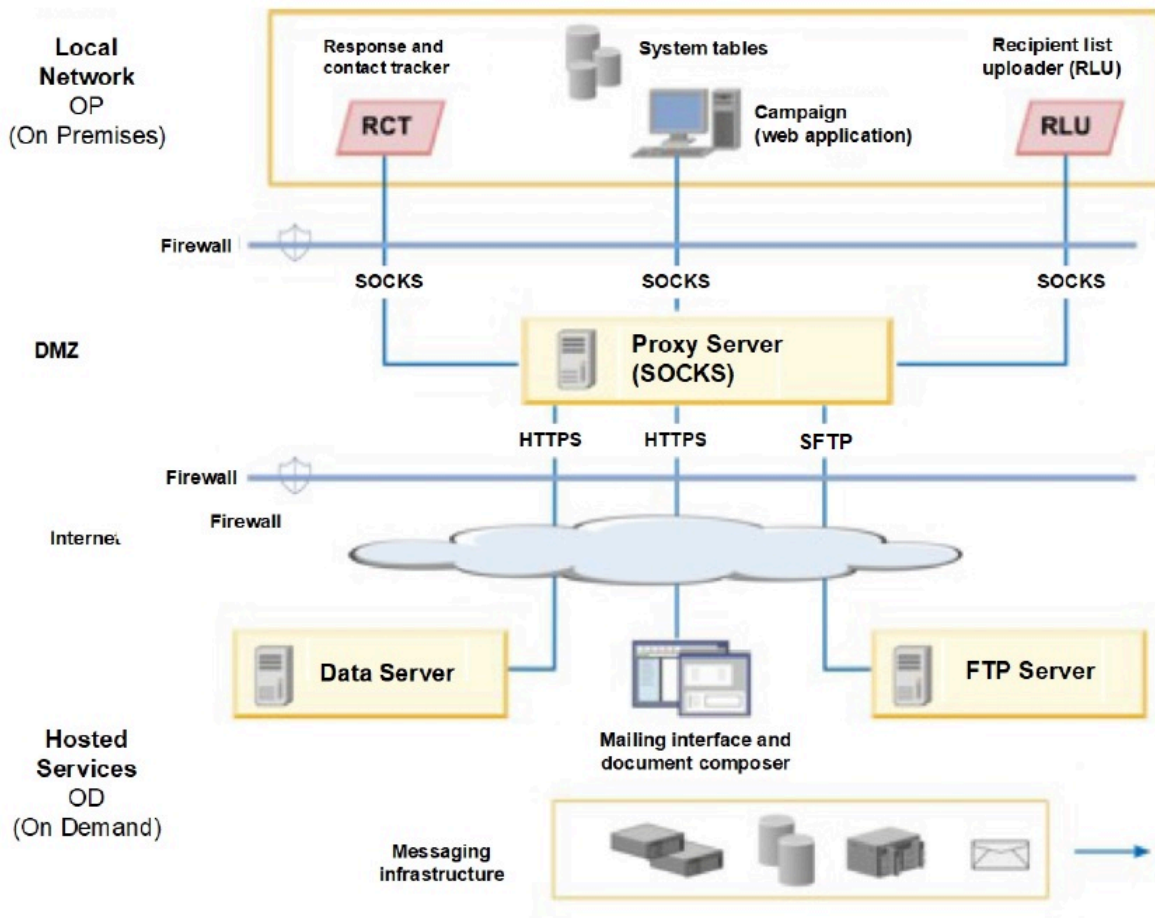
You configure the SOCKS proxy server in the local On Premises environment. Before you begin configuring the proxy server, confirm that you satisfy the following requirements.

- The proxy server must be a SOCKS proxy server.
- The proxy server must be able to access the Deliver OD environment. The server must allow traffic to and from the ports that are configured for the data center that is used by your hosted email account. Unica maintains data centers in the United States, Europe, and India.
- The Deliver OP environment must be able to access the SOCKS proxy server.

Configuring routing for SFTP and HTTPS traffic through a SOCKS proxy

To use a SOCKS proxy to access the hosted email resources, you must update the web application server where you deployed Campaign. You must also modify the startup scripts for the Deliver RCT and RLU.

• For SFTP traffic, apply the following configurations to the RLU and the web application server.

Setting	Description
`-Dhcl.unica.deliver.ftp.proxy.host = <socksHost>`	Host name or IP of the SOCKS proxy.
`-Dhcl.unica.deliver.ftp.proxy.port = <socksPort>`	The port on which SOCKS proxy is running.
`-Dhcl.unica.deliver.ftps.proxy.match.hosts=` `<comma separated list of host names and IP addresses>`	Host names and IP addresses that are used when routing traffic through the SOCKS proxy. Provide values specific to the data center used by your account.

When the local and hosted environments establish a data connection, the IP address that is specified for `-Dhcl.unica.deliver.ftps.proxy.match.hosts` is the IP address that the remote FTP server sends to the local FTP client.

Set `-Dhcl.unica.deliver.ftps.proxy.match.hosts` to one of the following values. The value that you enter depends on the data center that is used by your hosted email account.

Host name and IP addresses for the US data center:

```
-Dhcl.unica.deliver.ftps.proxy.match.hosts=

ftp-em.unicadeliver.com
```

Host name and IP addresses for the Europe data center:

```
-Dhcl.unica.deliver.ftps.proxy.match.hosts=

ftp-eu.unicadeliver.com
```

Host name and IP addresses for the India data center:

```
-Dhcl.unica.deliver.ftps.proxy.match.hosts=

ftp-in.unicadeliver.com
```

• For HTTPS traffic, apply the following configurations to the RCT and the web application server.

Setting	Description
`-Dhcl.unica.deliver.https.proxy.host=<socksHost>`	Host name or IP of the SOCKS proxy
`-Dhcl.unica.deliver.https.proxy.port=<socksPort>`	The port on which SOCKS proxy is running
`-Dhcl.unica.deliver.https.proxy.type=SOCKS`	The type of proxy server. You must use a SOCKS proxy server.

Configuring authentication to access a SOCKS proxy

If your SOCKS proxy requires authentication, you must configure the web application server, RLU, and RCT to provide the access credentials.

Configure the following for the web application server, RLU, and RCT. The values for `username` and `password` must be the credentials that are required to authenticate against the proxy.

```
-Dhcl.unica.deliver.proxy.auth.user = <username>
```

```
-Dhcl.unica.deliver.proxy.auth.password = <password>
```

Configuring the RCT to use a SOCKS proxy

You must modify the RCT to communicate through a SOCKS proxy server. The required settings depend on your operating system.

- For the RCT in Windows™ environments, add the following proxy arguments to `common.bat`.

    The `common.bat` file is in the `\deliver\bin` directory of your local Deliver installation.

    ```
    set RCT_PROXY_ARGS=

    -Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>

    -Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>

    -Dhcl.unica.deliver.https.proxy.type=SOCKS

    -Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUTH_USER>

    -Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_PASSWORD>

    set RCT_JAVA_ARGS=%BASE_VM_ARGS% %RCT_MEM_ARGS%

    %RCT_EXTRA_VM_ARGS% %RCT_PROXY_ARGS%
    ```

- For the RCT in UNIX™ environments, add the following proxy arguments to `common.sh`.

    The `common.sh` file is in the `/deliver/bin` directory of your local Deliver installation.

Note: Do not directly modify `rlu.sh`, `rct.sh`, or `setenv.sh`. The system overrides the changes.

```
RCT_PROXY_ARGS="

-Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.https.proxy.type=SOCKS

-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUTH_USER>

-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_PASSWORD>"

RCT_JAVA_ARGS="${BASE_VM_ARGS} ${RCT_MEM_ARGS} ${RCT_EXTRA_VM_ARGS} ${RCT_PROXY_ARGS}"
```

Configuring the RLU to use a SOCKS proxy

You must modify the RLU to communicate through a SOCKS proxy server. The required settings depend on your operating system.

- For the RLU in Windows™ environments, add the following proxy arguments to `common.bat`. The `common.bat` file is in the `\deliver\bin` directory of your local Deliver installation.

```
set RLU_PROXY_ARGS=

-Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>

-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUTH_USER>

-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_PASSWORD>

set RLU_JAVA_ARGS=%BASE_VM_ARGS% %RLU_MEM_ARGS% %RLU_EXTRA_VM_ARGS%

%RLU_PROXY_ARGS%
```

- For the RLU in UNIX™ environments, add the following proxy arguments to `common.sh`. The `common.sh` file is in the `/deliver/bin` directory of your local Deliver installation.

Note: Do not directly modify `rlu.sh`, `rct.sh`, or `setenv.sh`. The system overrides the changes.

```
RLU_PROXY_ARGS="

-Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>

-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUTH_USER>
```

```
            -Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_PASSWORD>"

        RLU_JAVA_ARGS="${BASE_JAVA_ARGS} ${RLU_MEM_ARGS} ${RLU_EXTRA_VM_ARGS}

        ${RLU_PROXY_ARGS}"
```

Configuring the web application server to use a SOCKS proxy

To connect to HCL Unica through a SOCKS proxy, you must modify the configuration of the web application server. For Unica WebSphere® servers, you modify the generic JVM arguments. For Oracle Weblogic servers, you modify the `SetDomainEnv` script.

- If your web application server is Unica WebSphere®, add the following to the WebSphere® generic JVM arguments.

  ```
  -Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>

  -Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>

  -Dhcl.unica.deliver.https.proxy.type=SOCKS

  -Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>

  -Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>

  -Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>

  -Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUTH_USER>

  -Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_PASSWORD>
  ```

- If your web application server is Oracle Weblogic, modify the `setDomainEnv` script. The required settings depend on your operating system.

  In Windows™ environments, make the following changes:

  ```
  JAVA_OPTIONS =%{JAVA_OPTIONS}

  -Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>

  -Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>

  -Dhcl.unica.deliver.https.proxy.type=SOCKS

  -Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>

  -Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>

  -Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>

  -Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUTH_USER>

  -Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_PASSWORD>
  ```

  In UNIX™ environments, make the following changes:

  ```
  JAVA_OPTIONS ='${JAVA_OPTIONS}
  ```

```
-Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.https.proxy.type=SOCKS

-Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.ftp.proxy.port=<POXY_PORT>

-Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>

-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUTH_USER>

-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_PASSWORD>'
```

Data download frequency and port setting

A Deliver component called the Response and Contact Tracker (RCT) is installed as part of your Unica Campaign installation. The RCT regularly requests email response and tracking data from HCL Unica. By default, the RCT issues a data request every 5 minutes.

The RCT issues data requests over HTTPS (HTTP over SSL). HCL Unica hosted services accept HTTPS connection requests on port 443, and only from hosts that you have specified during the hosted email account startup process.

Configuring a system user to access HCL Unica hosted services

Deliver components must be able to access the HCL Unica hosted services without requiring manual entry of login credentials. To establish automatic login, define a system user in Platform that can provide the required access credentials.

To simplify user administration and troubleshooting, you can modify an existing system user to access hosted services and local system tables. You can configure a single system user to provide credentials for multiple systems. For example, modifying the configuration of the Campaign system user creates a single user that can automatically access HCL Unica hosted services and the Deliver system tables in the Campaign schema.

The credentials required to access HCL Unica hosted services are the user name and password that Unica provides for your hosted messaging account. The credentials that you use depend on whether you are connecting to the Unica data center in US, Europe or India. Consult with Unica to determine which data center you use.

For specific information about how to configure a system user to communicate with HCL Unica hosted services, see the *Unica Deliver Startup and Administrator's Guide.*

For general information about how to create system users and data sources, see the *Unica Platform Administrator's Guide*.

Configuring partition access to HCL Unica hosted services

Unica Deliver components in the partition must be allowed to automatically provide valid login credentials when attempting to communicate with HCL Unica hosted services. To do this, you must add the HCL Unica hosted services login credentials to a Platform user. This user becomes the Deliver system user.

You can add the platform data source containing the HCL Unica hosted services credentials to the Deliver system user. This user can be the same system user that accesses the Campaign system tables in the partition.

The steps for configuring system users for a partition are the same as those followed during the initial Deliver installation, which created the first partition. For details about how to add HCL Unica hosted services login credentials to a system user, see the *Unica Deliver Startup and Administrator's Guide*.

The credentials required to access HCL Unica hosted servuces are the user name and password that Unica provided during the initial startup process.

⚠ **Important:** For each additional partition, you must request a separate user name and password from Unica.

Configuring the system user that accesses HCL Unica hosted services

Deliver components in Campaign must be able to access HCL Unica hosted services automatically, without prompting for a login. System users that are configured in Unica Platform can reference a data source that provides the required user name and password. You can add the data source to a new system user or to an existing system user. To simplify user administration, you can update a system user that is already configured to access the Campaign schema so that it can also access HCL Unica hosted services.

Before you begin

To complete this task, you must know the HCL Unica hosted services user name and password that Unica assigned to your hosted email account. Receiving the user name and password is part of the account startup process.

You must have the appropriate access permissions and know how to create system users and data sources in the Unica Platform.

About this task

📝 **Note:** If your installation contains multiple partitions, you must complete this task for each partition. You cannot share system users across partitions.

1. Create a Platform data source to contain the user name and password that is required to access HCL Unica hosted services. For best results and easier maintenance, name this data source UNICA_HOSTED_SERVICES. Configure this data source as follows.

   For **Data Source Login**, enter the user name that you received from Unica during account startup.

   For **Data Source Password**, enter the password that you received Unica during account startup.

2. Specify the data source in the Deliver configuration. Use the **amDataSourceForAcctCredentials** configuration property.

   The configuration property is at `Deliver > partitions > partition[n] > hostedAccountInfo >` **amDataSourceForAcctCredentials.**

   By default, the specified data source is `UNICA_HOSTED_SERVICES`.

3. Specify a system user to access HCL Unica hosted services. You can specify an existing user or create a user. In the Deliver configuration, use the **amUserForAcctCredentials** configuration property.

The configuration property is at `Deliver > partitions > partition[n] > hostedAccountInfo >` **`amUserForAcctCredentials.`**

By default, the specified user is `asm_admin`.

4. Add the data source that is configured in Step 1 to the system user specified in Step 3.

What to do next

You must restart the web application server for the configuration changes to take effect.

Configuring secure communication for hosted email

Communications between the email marketer and HCL Unica hosted services occur over Secure Sockets Layer (SSL). You must change the web application server configuration to use SSL. Making the required changes requires using the Java™ `keytool` utility.

Configuring the secure communication involves the following actions.

- Generate a trusted keystore.
- Obtain a digital certificate from HCL Unica hosted services.
- Add the trusted keystore to the web application server.
- Import the HCL Unica hosted services digital certificate into the trusted keystore.

The exact steps and sequence required to configure SSL depend on the type and version of web application server (WebSphere®, WebLogic, Tomcat, JBOSS) on which you deployed Unica Platform and Unica Campaign.

For WebLogic, see Configuring SSL when using WebLogic on page 25.

For WebSphere®, see Configuring SSL when using WebSphere on page 27.

Generating a trusted keystore

Follow this procedure to create an identity keystore and a trusted keystore for configuring Unica Deliver to communicate with HCL Unica hosted services over SSL. You add the keystores to the web application server when you configure SSL.

About this task

HCL uses the following sample values in the procedures contained in this section.

- Identity keystore: `HCLUnicaClientIdentity.jks`
- Alias for the identity keystore: `HCLUnicaClientIdentity`
- Password (`-storepass`) for the identity keystore: `clientPwd`
- The security key (`-keypass`) for the identity keystore: `clientPwd`
- Certificate based on the identity keystore: `ClientCertificate.cer`
- Trusted keystore: `HCLUnicaTrust.jks`
- Password (`-storepass`) for the trusted keystore: `trustPwd`

The actual values that you enter must be specific to your installation.

To complete steps in this procedure, run the Java™ `keytool` utility from the command line.

1. Generate an identity keystore. Use the `genkey` command, as shown in the following example.

   The example creates an identity keystore named `HCLUnicaClientIdentity.jks`. You can use a different name for the identity keystore that you create.

   Example

   ```
   keytool -genkey -alias HCLUnicaClientIdentity -keyalg RSA -keystore <HCLUnicaClientIdentity.jks> -keypass
   <clientPwd> -validity 1000 -dname "CN=hostName, O=myCompany" -storepass <clientPwd>
   ```

   Note the following.

   - You use the values for `alias`, `keystore`, `keypass`, and `storepass` later in this procedure and when you configure SSL in the web application server.
   - For WebSphere®, the keystore password (`-storepass`) and the key password (`-keypass`) must be the same.
   - In the distinguished name (`-dname`) the common name (`CN`) is the same as the host name used to access HCL Unica hosted services. For example, if the URL for HCL Unica hosted services is `https://hostName.example.com:7002/unica/jsp`, then the CN is `hostName.example.com`. The CN portion of the distinguished name is the only required portion; Organization (`O`) and Organizational Unit (`OU`) are not required.

2. Generate a certificate based on the identity keystore. Use the `export` command, as shown in the following example.

   The example generates a certificate named `ClientCertificate.cer`. You can use a different name for the certificate that you create.

   The values for `keystore`, `storepass`, and `alias` must match the values you specified for the identity keystore.

   Example

   ```
   keytool -export -keystore <HCLUnicaClientIdentity.jks> -storepass <clientPwd> -alias HCLUnicaClientIdentity
   -file <ClientCertificate.cer>
   ```

3. Generate the trusted keystore. Use the `import` command, as shown in the following example.

   The example generates a trusted keystore named `HCLUnicaTrust.jks`. You can use a different name for the trusted keystore that you create.

   Example

   ```
   keytool -import -alias HCLUnicaClientIdentity -file <ClientCertificate.cer> -keystore <HCLUnicaTrust.jks>
   -storepass <trustPwd>
   ```

   Type **Y** when prompted to trust the certificate.

What to do next

Note the values that you defined for the following variables. Your values can be different from the values given in the example.

- `alias` (in the example: `HCLUnicaClientIdentity`)
- `identity keystore` (in the example: `HCLUnicaClientIdentity.jks`)

- `storepass` (in the example: `trustPwd`) The `storepass` value for the trusted keystore can be different from the `storepass` value for the identity keystore and certificate.
- `keystore` (in the example: `HCLUnicaTrust.jks`) Depending on your web application server, you also specify the identity keystore.

You specify these installation-specific values when you configure SSL on the web application server for your HCL Unica installation.

Configuring SSL when using WebLogic

This section describes the steps required to configure SSL if you deploy HCL Unica components on Oracle WebLogic. This change is required to allow Deliver components that operate inside Campaign to communicate with HCL Unica hosted services over SSL.

About this task

For specific guidance regarding navigation and working with the Oracle WebLogic user interface, consult the documentation for the specific Oracle WebLogic version you are using.

Complete the following tasks.

- Modify the WebLogic startup script
- Modify the WebLogic configuration
- Obtain a digital certificate from HCL Unica hosted services
- Create a trusted keystore and import the Unica digital certificate

Modifying the WebLogic startup script

If you deployed Campaign on WebLogic, you must modify the WebLogic startup script and the WebLogic configuration for SSL so that WebLogic recognizes and accepts secure communication between locally installed Deliver components and HCL Unica hosted services.

Add the following arguments to the JAVA_OPTIONS in the WebLogic startup script.

- `-Dweblogic.security.SSL.allowSmallRSAExponent=true`
- WebLogic version 12c or higher: `-Dweblogic.security.SSL.protocolVersion=TLS1`

  All previous versions: `-Dweblogic.security.SSL.nojce=true`

Modify the WebLogic configuration

You must change the SSL configuration in WebLogic.

Use the WebLogic console to make the following change in the WebLogic SSL configuration for your domain.

Change the **Hostname Verification** setting to `None`.

Obtain a certificate from HCL Unica hosted services

To configure SSL communication, you must download a digital certificate from HCL Unica. The certificate details are saved to a file with the `.cer` extension that you can import into the web application server keystore.

About this task

You lose access to HCL Unica hosted services when your existing SSL certificate expires. Use this procedure to download a new certificate.

1. In Internet Explorer, log into the address for HCL Unica hosted services that has been configured for your hosted email account.
     - For the US data center, go https://em.unicadeliver.com
     - For the Europe data center, go to https://em-eu.unicadeliver.com
     - For the India data center, go to https://em-in.unicadeliver.com

   The login attempt results in a failed login, but allows you to use the browser to submit the certificate request.
2. Click the lock icon and select **View Certificate**.
3. Select the Details tab and select **Copy to File**.

   Save the file with a `.cer` extension to a location that is accessible to the web application server. The file that you create is the digital certificate that you insert into the keystore on the web application server.

   For example, save the certificate as `HCLHosted.cer`.

Create a trusted keystore for WebLogic and import the Unica certificate

For Weblogic, you must create a trusted keystore that accepts the Unica certificate.

Before you begin

Before you begin, use a web browser to download the HCL Unica hosted services digital certificate and save it as a `.cer` file. For example, the certificate can be named `HCLHosted.cer` (your file name can be different). For additional details, see Obtain a certificate from HCL Unica hosted services on page 26.

About this task

HCL uses the following sample values in the procedures contained in this section.

- Identity keystore: `HCLUnicaClientIdentity.jks`
- Password for the identity keystore: `clientPwd`
- Trusted keystore: `HCLUnicaTrust.jks`
- Alias for the trusted keystore: `HCLUnicaHostedIdentity`
- Password (`-storepass`) for the trusted keystore: `trustPwd`
- Digital certificate (`-file`) from Unica: `HCLHosted.cer`

The actual values that you enter must be specific to your installation.

To complete steps in this procedure, run the Java™ `keytool` utility from the command line.

1. Generate a trusted keystore for WebLogic.

   For details, see Generating a trusted keystore on page 23.

   You specify the *identity keystore* and the *trusted keystore* in the WebLogic configuration.

2. Use the `import` command in the `keytool` utility to add the HCL Unica hosted services certificate to the trusted keystore created in Step 1, as shown in the following example.
   Use the digital certificate that you downloaded from Unica.

   In this procedure, you also define an alias for the trusted keystore.

   Example

   ```
   keytool -import -alias HCLUnicaHostedIdentity -file <HCLHosted.cer> -keystore <HCLUnicaTrust.jks> -storepass
   <trustPwd>
   ```

   Type **Y** when prompted to trust the certificate.

3. In the WebLogic administration console, configure the keystores for the server.

   To specify the configuration rules, select the option for Custom Identity and Custom Trust keystores from the available choices. For the Custom Identity, you specify the identity keystore. For the Custom Trust, you specify the trusted keystore.

   For example, in the administration console, specify the following (using the example values from the trusted keystore you created in Step 1).

   - For the **Identity**: specify the identity keystore and associated password.

     For example, `HCLUnicaClientIdentity.jks` and `clientPwd`.
   - For the **Trust**: specify the trusted keystore and associated password.

     For example, `HCLUnicaTrust.jks` and `trustPwd`.

   Specify the full path to both keystores.

4. Restart WebLogic. WebLogic does not implement the configuration changes until you restart the web application server.

5. To test the SSL connection, log in to Unica Campaign and access various messaging features menus. Confirm that you can create email, landing pages, and mailings.

Configuring SSL when using WebSphere

This section describes the general steps required to configure SSL if you have deployed HCL Unica components on WebSphere®. This change is required to allow Deliver components that operate inside Campaign to communicate with HCL Unica hosted services over SSL.

Before you begin

Before you begin, you will need to know the value for the configuration property `uiHostName`. The value for `uiHostName` is the URL for HCL Unica hosted services . For details, see Configuring addresses for connecting to HCL Unica hosted services on page 10.

About this task

You must access the WebSphere® security console to modify settings for SSL certificate and key management. This task requires a restart of the Campaign web application server to implement the changes.

If you have deployed Campaign on WebSphere®, you must modify the WebSphere® security configuration to retrieve the signer certificate from HCL Unica and add it to the WebSphere® trust store. If you receive an error message indicating that your current signer certificate has expired, delete the current certificate and add a new one.

For specific guidance regarding navigation and working with the WebSphere® user interface, consult the documentation for the specific Unica WebSphere® version you are using.

1. Generate a trusted keystore.

   For additional details, see .

   To configure SSL, you need to specify the values that you define for the following variables. The values shown are for example only. Your values can be different.

   - `alias`: *UnicaClientIdentity* (example)
   - `keystore`: *HCLUnicaTrust.jks* (example)
   - `storepass`: *trustPwd* (example)

2. Select the new keystore in the WebSphere® security console.

   For example, if you followed the example in Step 1, select `HCLUnicaTrust.jks`.

3. Obtain a security certificate from HCL Unica and import it into WebSphere®, as described in the following steps.

   a. In the WebSphere® security console, navigate to **SSL certificate and key management** > **Key stores and certificates** > **NodeDefaultTrustStore** > **Signer certificates**. Select the option to **Retrieve from port**.

   b. Configure WebSphere® to establish a test connection to retrieve the signer certificate from HCL Unica. Enter the following values for the HCL Unica signer certificate.
      - **Host** The value that is defined for `Deliver >serverComponentsAndLocations > hostedServices >uiHostName`
      - **Port** 443
      - **SSL configuration for outbound connection** `NodeDefaultSSLSettings`
      - **Alias** The value that you entered for **Host**

   When you have finished, WebSphere® communicates with HCL Unica hosted services to retrieve the information required to create a signer certificate for HCL Unica hosted services.

4. After WebSphere® finishes creating the signer certificate, select the new certificate in the security console.

   The web application server uses the new certificate when establishing connections to HCL Unica.

5. Restart WebSphere®

   WebSphere® does not implement the configuration changes until you restart the web application server.

   For additional information about supported WebSphere® versions for deploying Unica products, see the *Recommended Software Environments and Minimum System Requirements* document for each product.

Deploying Campaign in Tomcat or JBOSS

There are no extra configurations required for Deliver, if Campaign is deployed in Tomcat or JBOSS. You do not require to obtain and configure any hosted services certificates.

Chapter 4. Response and Contact Tracker operation

The Response and Contact Tracker (RCT) is installed in your local environment and communicates with HCL Unica hosted services to retrieve and process data for email contacts, email delivery, and recipient responses, such as link clicks and opens. The RCT must be running in order to retrieve link tracking and email delivery notification data from HCL Unica hosted services .

You can use RCT with or without Kafka. If you do not wish to use Kafka, go to Deliver settings and set the value of **IsKafkaEnabled** to `False`. For more information, see Deliver | serverComponentsAndLocations | Kafka | RCT on page 74.

Customers can use RCT without Kafka if they don't have required infrastructure support. Without Kafka, RCT will run in single instance mode and process responses sequentially. For faster processing, scaling RCT on multiple instances is recommended by running it with Kafka configured.

If you use RCT with Kafka, it processes the responses independently of download mechanism. When RCT is run for the first time, multiple Kafka topics are created if not already existing. Each type of response is processed using a separate topic per Campaign partition. Each topic has two Kafka partitions and two consumers by default allowing faster response processing even with single instance of RCT. The significance of each topic created for Deliver RCT under RCT with Kafka integration is as follows:

Sl No	Topic Name	Description	Events Processed
1	`AP_rct_topic_{partition_name}`	alert publication	Alert notification
2	`CT_rct_topic_{partition_name}`	contact tracker	Email, SMS, and WhatsApp message sent events
3	`INET_rct_topic_{partition_name}`	inet response	Email Open, Link click, and all responses for SMS and WhatsApp
4	`IMT_rct_topic_{partition_name}`	inbound mail	All type of bounce responses (response type 5 to 13)
5	`LPT_rct_topic_{partition_name}`	landing page	Landing page form submission
6	`MT_rct_topic_{partition_name}`	mailing tracker	Container events
7	`MAET_rct_topic_{partition_name}`	mobile app event	All mobile push related events
8	`MAKT_rct_topic_{partition_name}`	mobile app key	Mobile app events
9	`OT_rct_topic_{partition_name}`	offer tracker	Deliver offer integration
10	`TST_rct_topic_{partition_name}`	treatment set	Deliver offer integration
11	`TT_rct_topic_{partition_name}`	treatment tracker	Deliver offer integration
12	`VT_rct_topic_{partition_name}`	variation tracker	Deliver offer integration
13	`EHT_rct_topic_{partition_name}`	execution history tracker	Detailed execution history events

It is recommended to run Kafka as a shared service on separate machine(s) so it is not specific to RCT and can also process messages from other Unica applications like Campaign, Journeys and Interact.

Restart of RCT is required in following conditions:

1. Execution history flag is toggled
2. Kafka configurations are changed
3. Kafka partitions are increased

You must configure Kafka in Unica Platform for the RCT. To access Kafka configurations, complete the following steps. For details on configuring Kafka, see the Configuration properties for Deliver on page 67 topic.

1. On Unica Platform, navigate to **Settings > Configuration**.
2. Expand the Deliver node.
3. Navigate to **Deliver|serverComponentsAndLocations|Kafka|RCT**.
4. Select **RCT**.
5. Select **Edit settings**.

The following are the mandatory configurations based on CommunicationMechanism value.

In the Kafka configurations page, you can select one of the following values for the CommunicationMechanism field:

- `NO_SASLPLAINTEXT_SSL`
- `SASL_PLAINTEXT`
- `SSL`
- `SASL_PLAINTEXT_SSL`

Based on your selection, the following fields become mandatory:

Field name	NO_SASLPLAIN TEXT_SSL	SASL_PLAIN TEXT	SSL	SASL_PLAIN TEXT_SSL
KafkaBrokerURL	Yes	Yes	Yes	Yes
TopicName	Yes	Yes	Yes	Yes
sasl.mechanism		Yes		Yes
UserForKafkaData		Yes	Yes	Yes

-

Field name	NO_SASLPLAIN TEXT_SSL	SASL_PLAIN TEXT	SSL	SASL_PLAIN TEXT_SSL
sasl.jaas.config.data Source		Yes		Yes
truststore.location			Yes	Yes
truststore.password.data Source			Yes	Yes
keystore.location			Yes	Yes
keystore.password.data Source			Yes	Yes
key.password.dataSource			Optional	Optional
ssl.endpoint.identification. algorithm			Yes	Yes

Make the required configurations and click Save.

Note: Due to large size of kafka log files, you may run out of storage and abrupt shut down of kafka server.

Steps to start RCT:

1. Start Zookeper, wait 10 sec

2. Start Kafka

3. Then Start RCT as usual

Steps to stop RCT:

1. Stop RCT
2. Stop Kafka
3. Stop Zookeper

You can start the RCT in either of the following ways.

- Start the RCT manually
- Start the RCT as a service

> ⚠️ **Important:** You must start the RCT manually the first time you use Deliver, even if you registered the RCT as a service.

You must restart the RCT when you make changes to configuration properties for Deliver. You can restart the RCT at any time, even if you have configured it to run as a service. HCL Unica hosted services continue to store tracking data if the RCT is shut down or restarting. When it resumes operation, the RCT downloads the queued information.

Manual operation of the Response and Contact Tracker

To operate the Response and Contact Tracker (RCT) manually, run the `rct` script in the `bin` directory in your Deliver installation.

- To start the RTC, run the `rct` script in the `bin` directory under your Deliver installation, as follows.

  `rct start`
- To stop the RCT, run the `rct` script as follows.

  `rct stop`

What to do next

For more information about this script, see .

Adding the Response and Contact Tracker as a service

You can configure the Response and Contact Tracker (RCT) to start automatically by adding it as a service.

About this task

Register the RCT service by running the `MKService_rct` script that is provided with the Deliver software.

To add the Response and Contact Tracker (RCT) as a service, run the `MKService_rct -install` script from the `bin` directory under your Deliver installation.

The `bin` directory is created as a subdirectory in the Campaign installation directory when you install or upgrade to the latest version of Unica Campaign.

In UNIX™ or Linux™, run this script with a user that has root permissions or permissions to create daemon processes.

In Windows™, the name of the service is **Response & Contact Tracker**.

What to do next

After you run the `MKService_rct` script, start the RCT manually with the `rct` script. You manually restart the RCT only once. After you start the RCT manually the first time, the RCT restarts automatically every time you restart the operating system of the computer where you installed the RCT.

After configuring the RCT service, you can prevent the RCT from starting automatically by running the `MKService_rct` script with the `-remove` option.

Removing the Response and Contact Tracker service

If you have installed the Response and Contact Tracker (RCT) as a service, the RCT restarts each time you restart the system where you installed the RCT. To prevent the RCT from restarting automatically, you must remove the Response and Contact Tracker (RCT) service.

To remove the RCT as a service, run the `MKService_rct` script with the `-remove` option.

From a Windows™ command line, in your HCL Unica home directory, run `Deliver\bin\MKService_rct.bat -remove`.

In UNIX™ or Linux™, in your HCL Unica home directory, run `Deliver/bin/MKService_rct.sh -remove`.

What to do next

For more information about this script, see

Chapter 5. Deliver Response Processing

From version 12.1.5, Deliver adopts a push model from On-Demand side so that reply messages arriving from WhatsApp and SMS from On-Demand server are pushed by Deliver to RCT as and when they appear. This significantly reduces data transfer between On-Demand and On-Premises because the responses will be pushed only when they arrive.

In the earlier versions, RCT was downloading responses of messages, sent via Unica Deliver and user interactions, to those messages, periodically, by polling OD dataapp server. This was not an effective model as response availability depended on the download frequency. Additionally, if there were no pending responses, RCT kept polling for responses resulting in bandwidth consumption and increase in load on OD dataapp server, which was responsible for serving responses to RCT.

Instead of the earlier mechanism, Deliver pushes these responses as and when possible. To effectively utilize resources like bandwidth and processing capabilities of downstream systems, the download frequency for different type of responses are as mentioned in the following points:

- Message send events for all channels and responses for email channel uses the current configuration of 1000 responses or 10 minutes whichever is earliest to send them together to OP.
- All type of delivery receipts for other channels, for example, SMS, WhatsApp, and Push are sent more often, like at an interval of 5 minutes, as these channels are more interactive.
- Reply messages for SMS and WhatsApp are pushed to OP as soon as they enter the OD server. This ensures that conversations can happen without a signifcant delay.

ResponseOut module

The ResponseOut module on Deliver OD side pushes only WhatsApp and SMS replies to the customers on-premises network. This simplifies the requirement of having static IP down to single module. Each module which processes response relays it to ResponseOut module via separate Kafka topics for each customer account and type of response. This allows ResponseOut module to scale horizontally to process responses coming from all modules faster and push them to customers on-premises network.

Since there are issues in communication like network issues over internet or inside customers network, RCT not running on customers side, a retry mechanism ensures that responses are not due to these issues, for example, if RCT is down and Deliver gets a 404 error for responses sent over web-hooks, Deliver retries few times to send the responses again after which these responses are routed to existing response files and RCT can download them using existing pull model when it is run again. However, for this approach, both new web-hook listen, and existing response download mechanisms must be kept active in RCT.

Changes on RCT to accept responses over webhook

RCT can be enhanced to accept incoming responses as it already handles further processing like saving those responses in Deliver OP tables or handing over responses to Journey for messages sent from Journey.

Deliver has changed RCT to a Spring Boot standalone application which can be triggered from command line or in dockerized manner. REST controllers are exposed in RCT which can handle responses pushed by Deliver OD. For example, inet responses can be pushed to the endpoint `/inetresp`. This endpoint is exposed at a common path. **Example**: `https://`

`CAMPAIGN_HOST:CAMPAIGN_PORT/deliver/responses`. Use this as a proxy URL instead of directly exposing the host which runs RCT (which is either Campaign web or listener node).

Securing communication between OD and OP

Customers must provide inbound access to above mentioned URL so Deliver can push responses to that URL. Inbound requests into customer's OP network must be secured so RCT will process responses originating only from Deliver datacenters. To secure the communication, Whitelist Deliver IP address which pushes responses into customer's network.

Performance and Scaling considerations for RCT

Based on typical response numbers (example 1 million contact, 1 million responses) determine if a single RCT instance can handle load of all incoming response events and and recommend scaling based on contact and response data getting processed by Deliver for each customer.

Chapter 6. Startup verification

To ensure access to all hosted email features, test the configurations and connections for your Campaign and Deliver installations after you enable Deliver, expand your Deliver installation, or upgrade the Campaign installation.

Verify configurations and connections after you do any of the following.

- Enable Deliver for the first time
- Upgrade your current Unica Campaign installation
- Add a new partition to the Deliver configuration maintained in Unica Platform

Confirmation for system configurations

To ensure that startup preparations are complete, confirm that the following configuration properties are set and that the settings meet the requirements for your Deliver and Campaign installations.

Configuration property	Setting
`Campaign | partitions | partition[n] | Deliver | DeliverPluginJarFile`	Complete path to the location of the plug-in file that operates as the Recipient List Uploader (RLU). Enter the full local directory path in the file system for the computer that hosts the Campaign web application server.
	The Unica installer populates this setting automatically for the default partition when you run the installer. For other partitions, configure this property manually.
`Campaign | partitions | partition[n] | server | internal | deliverInstalled`	Indicates that Deliver is installed.
	Set this property to **Yes** in each partition where you want to enable Deliver, including the default partition. When you set this property to **Yes**, Deliver features become available in the Campaign interface.
`Deliver | serverComponentsAndLocations | hostedServices | uiHostName`	Address to HCL Unica for all communication except uploading lists.
	The default setting is `em.unicadeliver.com`, for the US data center.
	If you are connecting to the data center in Europe, change this value to `em-eu.unicadeliver.com`.
	If you are connecting to the data center in India, change this value to `em-in.unicadeliver.com`.
`Deliver | serverComponentsAndLocations | hostedServices | dataHostName`	The address for the connection that Deliver uses for uploading metadata that is related to recipient lists to HCL Unica.

Configuration property	Setting
	The default setting is `em.unicadeliver.com`, for the US data center.
	If you are connecting to the data center in Europe, change this value to `em-eu.unicadeliver.com`.
	If you are connecting to the data center in India, change this value to `em-in.unicadeliver.com`.
`Deliver | serverComponentsAndLocations | hostedServices | ftpHostName`	The address for the connection that Deliver uses for uploading recipient list data (except list metadata) to HCL Unica.
	For the host names, see the Configuring addresses for connecting to HCL Unica hosted services on page 10 topic.
`Deliver | partitions | partition[n] | hostedAccountInfo | amUserForAcctCredentials`	The HCL Unica user that references the data source that contains the HCL Unica hosted services access credentials.
	You configure this value when you create a system user to access the email resources that are hosted by Unica.
`Deliver | partitions | partition[n] | hostedAccountInfo | amDataSourceForAcctCredentials`	The Platform data source that contains the HCL Unica hosted services login credentials.
	You configure this value when you create a system user to access the email resources that are hosted by Unica.
`Deliver | partitions | partition [n] < dataSources | systemTables | type`	Type of database that hosts the system tables.
	Provide the correct value for your database.
`Deliver | partitions | partition [n] < dataSources | systemTables | schemaName`	Name of the database schema for the system tables.
	Set to the appropriate schema name for your database.
`Deliver | partitions | partition [n] < dataSources | systemTables | jdbcClassName`	JDBC driver for system tables.
	Provide the correct value for your environment.
`Deliver | partitions | partition [n] < dataSources | systemTables | jdbcURI`	JDBC connection URI for system tables.
	Provide the correct value for your environment.
	Specify the database type, database driver, host, port, and database name. For example: `jdbc:oracle:thin:@yourdb.example.com:1234:DBname`
	Consult your database documentation for specific instructions regarding how to construct the JDBC URL.
	The value that you enter must exactly match the value that is defined in your Campaign web server.

Configuration property	Setting
`Deliver	partitions	partition [n] < dataSources	systemTables	asmUserForDBCredentials`	The HCL Unica user that references the data source that contains the system tables login credentials.
	You create this user when you configure access to the local Deliver system tables.
`Deliver	partitions	partition [n] < dataSources	systemTables	asmDataSourceForDBCredentials`	The Platform data source that contains login credentials to the database that contains the system tables.
	You create this data source when you create a user to access the Deliver system tables.

Testing upload to HCL Unica hosted services

To test the ability to upload data to HCL Unica hosted services from your local environment, run the **rlu** script in check mode.

In the `bin` directory under your Deliver installation, run the `rlu` script in either of the following ways.

- `rlu -c`
- `rlu --check`

Testing download from HCL Unica hosted services

To test the ability to download information from HCL Unica hosted services, run the rct script in check mode.

In the `bin` directory under your Deliver installation, run the `rct` script as follows.

`rct check`

Testing the connection to the hosted messaging interface

Unica hosts the messaging interface from its data centers in the US, India, and Europe. Test the connection to the hosted mailing interface by attempting to access an Deliver feature.

Log in to HCL Unica and select **Deliver mailings** from the **Campaign** menu.
Result

If the connection to the Deliver user interface is established correctly, the Deliver mailings page opens and displays a list of mailings and related mailing characteristics.

If the connection to the user interface is not established properly, you see an error.

Chapter 7. Unica Deliver REST APIs

Deliver has exposed REST APIs which can be used to perform variety of operations like working with Email, SMS, Push, WhatsApp messages and communication templates, working with batch production and test message sending as well as transactional message sending. This chapter provides details of these APIs, how to authenticate and generate API token which is needed to call these APIs.

Since these APIs directly work using Deliver hosted account credentials, all the operations like view, create, update, delete are accessible via these APIs. So, these APIs should be typically used by a user who is Deliver Administrator. If you are integrating these APIs in a third-party application to work with Unica Deliver Hosted Services, you should allow operations based on these APIs with proper authentication and authorization of users accessing this third-party application.

The host name needed to use for invoking all the APIs is value of configuration parameter `uiHostName` under Deliver >> serverComponentsAndLocations >> hostedServices configuration setting page. Typically, it is one of the Deliver datacenter hosts where your Deliver account is provisioned. For information related to connections to message services for exact value of host name you need to use, see Connections to message services on page 10.

Generating an authentication token

Before using any API, generate an authentication token using login API as mentioned in the following menthod:

```
POST https://<deliver_host>/deliverui/api/deliver/rest/v1/login
```

Request Header:

```
m_user_name: {deliver_account_user_name}
m_user_password: {deliver_account_password}
```

Response:

The earlier mentioned API will return a token which should be used in subsequent APIs.

Example: A CURL request for login API for Unica Deliver US datacenter:

```
curl --location --request POST 'https://em.unicadeliver.com/deliverui/api/deliver/rest/v1/login' \
--header 'm_user_name: <USER>' \
--header 'm_user_password: <PASSWORD>'
```

If the username and password combination is valid, it returns a 200 OK response with payload similar to the following example:

```
{
    "m_tokenId": "1661234885892-69-zQEqQ6n5-XmV3-W3PhU3p4-eOQ7-0ohb1tKA",
    "createDate": "Tue Aug 23 06:08:05 UTC 2022",
    "m_user_name": "a38test"
}
```

This token is typically valid for 30 minutes.

Working with Unica Deliver REST APIs

Once you have obtained an authentication token, you can invoke APIs by passing that token as part of authentication header and rest of the parameters as needed for each API.

You can refer to API Documentation or Transactional Message Service API documentation in Settings -> Messaging Settings menu in Unica application.
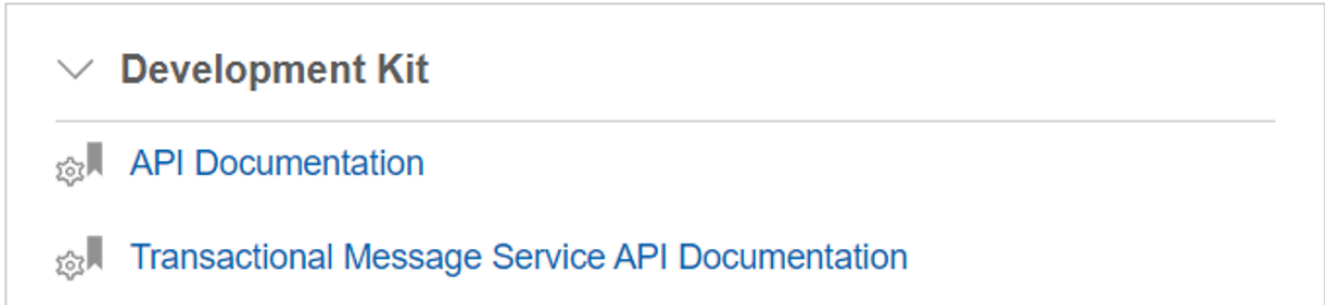
The following is an example of email communication template details API invocation:

```
curl --location --request GET 'https://em.unicadeliver.com/deliverui/api/deliver/rest/v1/email-templates' \
--header 'm_tokenid: <TOKEN>' \
--header 'api_auth_mode: manager' \
--header 'm_user_name: <USER>'
```

The following is a sample output of this API (considering a single email communication template):

```
[
    {
        "hasHTMLContent": true,
        "id": 201,
        "name": "Email 1",
        "policyId": -1,
        "fromDomainName": null,
        "hasTextContent": false,
        "associatedOffersCount": 0,
        "thumbnailURL": null,
        "landingPageCount": 0,
        "folder": false
    }
]
```

Like this example, all the other APIs can be invoked using a REST client, like Postman, for trying it out before integrating in a third-party application.

Chapter 8. Configurations for implementing Push Notifications

Unica supports push notifications by using the Deliver MX SDK, provided in the form of a framework to ease the integration into your iOS and Android app. This guide provides an overview of setting up your app with in both your Apple Production and Firebase Cloud Messaging accounts before detailing the specific integration steps for the appropriate SDK for your project.

For integration with Android, complete the following procedures:

For integration with iOS, complete the following procedures:

For creating an App in Deliver, complete the following procedure:

Android Push integration with Firebase

If application is already enabled in Firebase and Message API enabled and `.json` is added to project, then skip to the step to generate the server key and provide to Unica Deliver Administrator.

About this task

Perform the following procedure to integrate Android Push with Firebase for sending and receiving Push from Unica Deliver SDK library and framework:

1. Open the Firebase Console.
2. Click **Add Project or Create New Project**.
   Result

   The **Create Project** window opens.
3. Perform the following steps to create a project:

   a. Provide an appropriate value for the **Enter your project name** field. The name is the proect name of your app in the Firebase console.

   b. **Optional Step**: Perform this step only if you want to disable Google Analytics. If not, click **Continue** and move to the next step. To disable Google Analytics for the project, toggle the **Enable Google Analytics for this project** option (by default, the option is enabled) and click **Continue**.

   c. For the **Choose or create a Google Analytics account** field, select the project name and click **Create project**.

   d. When your project is ready, click **Continue**.

Result

The start screen of the Firebase Console opens.

4. Perform the following steps to add Firebase to your Android apps:

    a. Select the Android icon.

       Result

       The **Add Firebase to your Android app** panel opens

    b. In the **Register app** section, provide values for the following fields:

Field name	Mandatory?	Description
Android package name	Yes	Enter the package name of the mobile app. Type the package name that you entered when creating the project in Android Studio. **Example**: `com.example.testapp`
App nickname	No	A short name to quickly identify the package.
Debug signing certificate SHA-1	No	Add this only if you want to use FCM features like Dynamic Links, Invites, and Google Sign In. If you do not want to use FCM features, click the **Register app**.

    c. In the **Download and then add config file** section, click **Download google-services.json** to download the file to your system. In Android Studio, set up the `oogle-services.json` file and click **Next**.

    d. In the **Add Firebase SDK** section,
       • if your project is in Kotlin, select **Kotlin DSL (build.gradle.kts)**.
       • if your project is in Java, select **Groovy (build.gradle)**

       Add the provided dependencies in Android Studio.

    e. Click **Next**.

    f. Click **Continue**.

       Result

       The Firebase console page opens.

5. Select the created project and select the Settings icon.

   Result

   The **Project settings** page opens.

6. Select the **Cloud Messaging** tab.

Result

By default, the Server Key (**Cloud Messaging API (Legacy)**) is disabled.

7. To enable **Cloud Messaging API (Legacy)**, select the three dots succeeding it.

Result

A small panel appears.

8. Select **Manage API in Google Cloud Console**.

Result

A Google Cloud screen appears.

9. Within **Cloud Messaging**, click **Enable** and close the screen.

Result

You are back in the **Cloud Messaging** tab within the **Project Settings** screen.

10. The **Cloud Messaging API (Legacy)** is now enabled and Server key field is seen.

11. Provide the Server Key to the Unica Deliver administrator to configure the application for Push Notifications on Android.

Android SDK integration for Push notification

To receive Push notifications on Unica Deliver, you must integrate the required SDK. To integrate the SDK, you must execute the following procedures:

Downloading the appropriate Android SDK

About this task

You will first need the SDK to perform the integration. For accessing the SDK, complete the following steps:

1. Use the provided Unica download link and download the SDK.
2. Save the `kony_sdk-release.aar` on your local system.

Configuring the `kony_sdk_release.aar` file

1. Launch the Android Studio.
2. Select **Start a new Android Studio Project**.
3. Enter all the required data and select **Next**.
4. To import SDK libraries in your project, complete the following steps:
    a. Select the `kony_sdk-release.aar`.
    b. Copy the `kony_sdk-release.aar` and paste it to the `/app/libs/` folder in your project.
    c. Navigate to **Android Studio > File > Project Structure**.
    d. Click **Dependencies**.
    e. To the right of **All Dependencies**, select **+ > JAR/AAR Dependencies**.
    f. Select the `kony_sdk-release.aar` file path.
    g. Click **OK**.

Adding permissions in the `AndroidManifest.xml` file

About this task

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

Adding the `MyFirebaseMessagingService` class

About this task

To build the notification, add the `MyFirebaseMessagingService` class (from Google site) to the application.

The `fcm` token subscribes the device to console for receiving notifications. The `FirebaseMessaging.getInstance().getToken()` call receives the `fcm` token.

Initializing the Android Client SDK

1. Declare the following parameters globally:
   - **KonyClient myClient;**
   - **MessagingService messagingClient = null;**
2. Add the following line within the `onCreate()` function.

```
myClient = new KonyClient();
```

Calling the `initAsync` method

About this task

Contact the Unica Deliver administrator for Appkey, AppSecret and ServiceURL.

> 📝 **Note:** For API authentication pass the following headers:
>
> - `header_key_for_API = "X-Voltmx-App-API-Key"`
> - `header_value_for_API = "<Api Key Header>"`

```
String appkey = "<Primary App Key>";
String appsecret = "<Primary App secret>";
String serviceURL = "<Service URL>";
```

Unica Deliver administrator can access Appkey, AppSecret and ServiceURL from the following screen:

```
Context context = getApplicationContext();
try
{
 myclient.initAsync(getApplicationContext(), appkey, appsecret, serviceURL new InitCallback()
 {
  @Override
  public void onSuccess(JSONObject response)
  {
   Log.d("Init", "Success");
   // In the response object will get the baseURL and appID
   //call the messagingService() here and should pass the
   //FCM token here
   initMessaging(fcmToken)
         }
  @Override
  public void onFailure(KonyException error)
  {
   Log.d("Init", "Failure"); ]
  }
     });
}
catch (KonyException exception)
{
    Log.d("Init", "Exception");
}
```

Writing the `initMessaging()` function

About this task

Add the following code to `initMessaging()`

```
public void initMessaging(String FCmToken)
{
 try
 {
  messagingClient = myClient.getMessagingService();
     }
 catch (KonyException exception)
 {
  Log.e("Exception", Arrays.toString(exception.getStackTrace()));
     }
 @SuppressLint("HardwareIds")
 String deviceId =  Settings.Secure.getString(getApplicationContext().getContentResolver(),
 Settings.Secure.ANDROID_ID);
 messagingClient.registerInBackground(FCmToken,"UserId", deviceId, new MessagingCallback()
 {
  @Override
  public void onSuccess(JSONObject successresponse)
  {
   Log.e("Msg_Registration", "success");
   String ksid = successresponse.getString("id"); //You get the KSID here which will be used in Response API
            }
  catch (JSONException e)
  {
   throw new RuntimeException(e);
            }
         }
```

```
@Override
public void onFailure(MessagingServiceException messagingServiceException) {
Log.e("Registration Failure",messagingServiceException.getMessage());
        }
});
```

The device becomes ready for receiving notifications.

Testing Simple and RichPush notifications

About this task

You can now send Simple and Rich Push to device.

📝 **Note:** For notification to work in the App foreground, App background, or App kill state, we have removed the **notification key** from the payload and we are using only the **data** key.

```
String notificationData = remoteMessage.getData().toString();
```

```
{
 "priority":"normal",
 "data":
 {
  "responseData":"
  {
   "miId":"498",
   "appId":"8e25b50a-de26-4cd5-9c85-1e37cdb8ee70",
   "appRef":"unica"
  }",
  "title":"Hi",
  "mid":"8756756118946356591",
  "msgEntryId":2546,
  "seqNum":"0",
  "body":"Hello Username"
 }
}
```

1. For Simple push notifications, when the device receives the notification, use the `MyFirebaseMessagingService` class to build the notification.

   Example for Simple push notification payload

   ```
   {
    seqNum=0,
    mid=8608644054117202283,
    body=Hello Test,
    title=Hi,
    responseData=
    {
     "miId":"498",
     "appid":"8e25b50a-de26-4cd5-9c85-1e37cdb8ee70",
     "appRef":"unica"
    },
    msgEntryId=2270
   }
   ```

Notification with Image

To show Notification with Image, use the `image` key to get the image URL.

Example: `remoteMessage.getData().get("image");`

Payload

```
{
  seqNum=0,
  mid=8754907663368125996,
  body=Hello,
  image=https://<image-source-url>/<image-name>.<extension>,
  title=Hi,
  responseData=
  {
   "miId":"498",
   "appId":"8e25b50a-de26-4cd5-9c85-1e37cdb8ee70",
   "appRef":"unica"
  },
  msgEntryId=2530
}
```

Notification with clickAction

Use the `click_action` key to fetch the URL to open the URL in the browser.

Example: `remoteMessage.getData().get("click_action")`

Payload

```
{
  seqNum=0,
  mid=8755956681328623439,
  body=Hello,
  title=Hi,
  click_action=https://<URL>/,
  responseData=
  {
   "miId":"498",
   "appId":"aa8c669a-031a-453a-b59d-156eebcc629f",
   "appRef":"unica"
  },
  msgEntryId=2543
}
```

2. For RichPush Notifications, when the device receives the notification, use the `MyFirebaseMessagingService` class to build the notification. To receive the in-app content, you must execute the `getRichPushdDataAPICall( )` API.

Example for In-app push notification payload

```
{
  seqNum=0,
  mid=8392748706469346156,
  body=,
  title=,
  responseData=
  {
   "miId":"499",
   "appId":"8e25b50a-de26-4cd5-9c85-1e37cdb8ee70",
   "appRef":"unica"
```

```
  },
  msgEntryId=2273,
  isRichPush=true
}
```

When user clicks the notification, call the `getRichPushdDataAPICall( )` API.

`mID` = Receives the Push Notification payload that must be passed to the following method:

```
getRichPushdDataAPICall( )
{
 final String url = baseUrl+"/messages/richcontent/"+mID;

 //Here you will get the HTML response and this response should pass to Webview.
 //For proper in app view, we are showing the webview inside the AlertBox.
 //call AlertCustom(Pass the HTML response Here ) method.
}
```

```
Public void AlertCustom(HTMl DATA)
{
 //Here create a Webview and pass the html data to it.
}
```

Verifying custom events

About this task

The following are the custom events:

- `push.opened`
- `inApp.opened`
- `push.dismissed`
- `inApp.dismissed`
- `push.enabled`
- `push.disabled`
- `app.installed`
- `app.uninstalled`
- `custom`
- `push.delivered`
- `inApp.delivered`

Note: For the In-App notifications to work, the app must be open on the mobile device or in the memory. If you kill the app, the service class will stop working and the app will not work as expected.

1. `push.opened`, `inApp.opened`, `push.dismissed`, `inApp.dismissed`, `push.enabled`, `push.disabled`, `app.installed`, `app.uninstalled`, and `custom`

   For these events, the API remains the same and only the POST body **type** and **data** parameter changes:

   ```
   final String url = baseUrl+"/response";
   ```

The API posts the body as shown in the following response and the data object receives notification payload as `responseData`.

`type` - Depends on the Notification type.

- For simple notification it is `push.opened` / `push.dismissed` / `push.delivered`.
- For in-app notification, it is `inApp.opened` / `inApp.dismissed` / `inApp.delivered`.

2. **Push-related events**: `push.opened`, `inApp.opened`, `push.dismissed`, `inApp.dismissed`, `push.delivered`, and `inApp.delivered`

```
{
 "type": "push.opened / inApp.opened / push.dismissed / inApp.dismissed / push.delivered /
 inApp.delivered",
 "uuid": "UUID.randomUUID().toString()",
 "happenedAt": "2024-02-09, 06:08:42", // Use the date format yyyy-MM-dd, hh:mm:ss (GMT) for responses
 to be correctly tracked from Unica Deliver
 "ksid": "7743349406375345629",
 "userId": "name@customer.com",
       // Content for the data object is populated upon receiving the notification payload
 "data":
 {
  "messageId": <mID>
  "seqNum":1,
  "miId":3,
  "appId":"ef7c9e33-09c1-4d05-8248-351537f453bd",
  "appRef":"unicaPartitionName",
  "msgEntryId":"2273"
 }
}
```

3. **App-related events**: `push.enabled`, `push.disabled`, `app.installed`, and `app.uninstalled`

```
{
 "type": "push.enabled / Push.disabled / app.installed / app.uninstalled ",
 "uuid": "UUID.randomUUID().toString()",
 "happenedAt": "2024-02-09, 06:08:42", // Use the date format yyyy-MM-dd, hh:mm:ss (GMT) for responses
 to be correctly tracked from Unica Deliver,
 "ksid": "7743349406375345629",
 "userId": "name@customer.com",
 "data":
 {
  "appId":"ef7c9e33-09c1-4d05-8248-351537f453bd", // appID retrieved from initAsync method.
  "appRef":"unicaPartitionName", // For appRef, contact Unica Deliver DevOps team.
 }
}
```

4. **custom**

Once the device receives the in-App push notification, and if the user clicks the notification, call the `getRichPushdDataAPICall` API for HTML response to show the pop-up. In the response, check whether the `data-custom-event` tag is available or not. If it is available, parse the `data-custom-event` tag.

If user clicks the custom event, POST the body as shown in the following sample:

Example HTML:

```
<a data-custom-event="\&quot;RESP_CLICK_EVENT\&quot;" href="\" none="
 style="\&quot;text-decoration:">Click</a></div>
```

Payload

```
{
 "type": "RESP_CLICK_EVENT", (data-custom-event)
 "uuid": "UUID.randomUUID().toString()",
 "happenedAt": "2024-02-09, 06:08:42", // Use the date format yyyy-MM-dd, hh:mm:ss (GMT) for responses
 to be correctly tracked from Unica Deliver,",
 "ksid": "7743349406375345629",
 "userId": "name@customer.com",
 "data":
 {
  "messageId": 1, <mID>
  "seqNum":1,
  "miId":3,
  "appId":"ef7c9e33-09c1-4d05-8248-351537f453bd", // appID retrieved from initAsync method.
  "appRef":"appRef":"unicaPartitionName", // For appRef, contact Unica Deliver DevOps team.
  "msgEntryId":"2404"
 }
}
```

5. **Deep link**

   Once the device receives the in-App push notification, and if the user clicks the notification, call the `getRichPushdDataAPICall` API for HTML response to show the pop-up. In the response, check whether the `data-deeplink` tag is available or not. If it is available, parse the `data-deeplink` tag. If the user clicks the deeplink button, it opens the screen based on deeplink `JSON`.

   If user clicks the Deep link button, POST the body as shown in the following sample:

```
<a data-custom-event="\&quot;RESP_DEEPLINK_EVENT\&quot;"
data-deeplink="\&quot;
{&quot;address&quot;:{&quot;pin&quot;:&quot;411&quot;},&quot;name&quot;:&quot;test&quot;}\&quot;"
 href="\" none=" style="\&quot;text-decoration:">Submit</a></div>
```

   📝 **Note:** Deliver does not support personalization field in Deep Link.

iOS Push integration with APNS

Apple Push Notification Service, also known as Apple Notification Service or APNS, is a platform service from Apple Inc. Third-party application developers use APNS to send push notifications to iOS users. A paid Apple Developer account is mandatory for creating certificates. The steps to create a `.p12` certificate, for sending Push notifications, are as follows:

Creating an Apple ID

About this task

If you already have an Apple ID, proceed to the procedure.

1. Open https://developer.apple.com/and log in using your credentials.
2. Select **Certificates, Identifiers and Profiles**.
3. From the dropdown menu, select `ios`.
4. From the side menu, select **App IDs** and create a new App ID.
5. To complete the app registration, click **Register > Done**.

Results

The newly created App apears in the list of App IDs.

Generating a certificate from Keychain access

1. On your Mac OS X, launch the **Keychain Access** application.
2. Select **Keychain Access > Certificate Assistant > Request a Certificate from a Certificate Authority**.
3. Save the certificate.

Generating a Production+SSL certificate

About this task

> ✏ **Note:** Currently, Unica Deliver only supports Production certificate.

1. Access the https://developer.apple.com/ again and log in using your credentials.
2. Select the app that you created in procedure Creating an Apple ID on page 51 and click **Edit**.
3. Scroll down to the **Production+SSL certificates** section and click **Create Certificate**.
4. Click **Continue**.
5. Select the certificate that you created using the Generating a certificate from Keychain access on page 52 procedure and click **Continue**.
6. Download the Production Certificate.
7. To finish the procedure, click **Done**.

Generating an APNS .p12 certificate

1. Double-click the Production Certificate that you created in the Generating a Production+SSL certificate on page 52 procedure to add it to the Keychain Access.
2. From the side menu, navigate to **Keychain Access > Login Keychain > My Certificate**.
3. Locate the certificate and right-click the certificate to export it.
4. Type the certificate name and click **Save**.
5. Type the password for the certificate and click **OK**.
6. To finish the procedure, type your system's administrator password and press **Enter**.

Results

Provide this Certificate key to the Unica Deliver Administrator as part of the Application configuration for Push Notification for iOS.

> ✏ **Note:** Create Separate Certificates for Production mode and Development mode for APNS Interaction.

Basic Push with image and customizing the Notification

About this task

You can customize a specific push notification. To customize a specific push notification, use the `UNNotificationCategory` to define a type of notification that the executable can receive.

In the configureNotification method, add the `UNNotificationCategory` and `UNNotificationAction` to the custom push.

Create a Swift class and add the Extension class with the Key Name to Send an Image with Simple Push.

Use the following extension for adding the user information with the Key required for customization:

```
extension UNNotificationRequest
{
 var attachment: UNNotificationAttachment?
 {
  guard let attachmentURL = content.userInfo["image"] as? String, let imageData = try? Data(contentsOf:
 URL(string: attachmentURL)!)
  else
  {
   return nil
  }
  return try? UNNotificationAttachment(data: imageData, options: nil)
 }
}
```

API calls inside the project

About this task

Use the default iOS framework NSUrl Session for all API calls and post the data back to server.

Sample code for NSurl session for a data task:

```
NSString *strUrl = [NSString stringWithFormat:yourUrl];
NSURL *url = [NSURL URLWithString:strUrl];
NSURLRequest *request = [NSURLRequest requestWithURL:url];
[NSURLConnection sendAsynchronousRequest:request
 queue:[NSOperationQueue mainQueue]
 completionHandler:^(NSURLResponse *response,
 NSData *data, NSError *connectionError)
 {
  if (data.length > 0 && connectionError == nil)
  {
   NSDictionary *dicResponse = [NSJSONSerialization JSONObjectWithData:data options:0 error:NULL];
   NSLog(@"%@",dicResponse);
  }
 }
];
```

iOS SDK integration for Push Notification

To receive Push notifications on Unica Deliver, integrate the required SDK. The prerequisites for integrating the SDK are as follows:

- Maverick OS X
- XCode 9 or later
- If you are using an untrusted self-signed (SSL) enable the `KNYClient acceptSelfSignedCertificates` API. By default, native apps do not allow untrusted SSL certificates for HTTPS connection.

- For notifications to work in your project, in XCode, navigate to **Add Push Notification capabilities > Build signing and capabilities > + Capabilities**, and add the **Push Notifications** capability.
- If you are developing an iOS app extension using the `KonySDK.xcframework`, add the following frameworks in your app extension project build phases:
  - `libsqlite3.framework`
  - `libz.framework`
  - `libc++.framework`
  - `VoltmxSDK.framework`
  - `Task.framework`
  - `SDKCommons.framework`
  - `VoltmxLogger.framework`
  - `Binary.framework`
  - `FMDB.framework`
  - `CMS.framework`
  - `VoltmxSyncV2.framework`
  - `SQLCipher.framework`
  - `Security.framework`
  - `CoreGraphics.framework`
  - `UIKit.framework`
  - `Foundation.framework`
  - `MobileCoreServices.framework`
  - `SystemConfiguration.framework`

After adding the earlier mentioned frameworks, add the `KNYSharedContainerIdentifier` key to the extension app `info.plist`.

To integrate the SDK, execute the following procedures:

Downloading and Configuring the SDK into the Project

About this task

To configure VoltMXSDK Framework to project, complete the following steps:

1. Use the provided Unica download link and download the SDK.
2. Extract the downloaded SDK file.
3. Drag `VoltMXSDK.framework` to your Framework group in Xcode project.
4. Select **Copy items into destination group's folder** check box. Ensure that the libraries added to your main target.
5. Click **Finish**.

Adding the Framework Dependency into the Xcode Project

1. In the **Editor**, select your app target.
2. Navigate to **Build Phases > Link Binary With Libraries**.
3. Click **Add (+)** button.
4. In the **Choose frameworks and libraries to add** dialog, type `systemconf`.

5. Select `SystemConfiguration.framework` and click**Add**.

Result

The system adds the selected frameworks to the project under the**Link Binary With Libraries** `section.`

6. Under Build Phases, in the Project Directory, add the following script:

```
# Type a script or drag a script file from your workspace to insert its path.
echo "Target architectures: $ARCHS"
APP_PATH="${TARGET_BUILD_DIR}/${WRAPPER_NAME}"
find "$APP_PATH" -name '*.framework' -type d | while read -r FRAMEWORK
do
FRAMEWORK_EXECUTABLE_NAME=$(defaults read "$FRAMEWORK/Info.plist" CFBundleExecutable)
FRAMEWORK_EXECUTABLE_PATH="$FRAMEWORK/$FRAMEWORK_EXECUTABLE_NAME"
echo "Executable is $FRAMEWORK_EXECUTABLE_PATH"
echo $(lipo -info "$FRAMEWORK_EXECUTABLE_PATH")
FRAMEWORK_TMP_PATH="$FRAMEWORK_EXECUTABLE_PATH-tmp"
#remove simulator's archs if location is not simulator's directory
case "${TARGET_BUILD_DIR}" in
*"iphonesimulator")
echo "No need to remove archs"
;;
*)
if $(lipo "$FRAMEWORK_EXECUTABLE_PATH" -verify_arch "i386") ; then
lipo -output "$FRAMEWORK_TMP_PATH" -remove "i386" "$FRAMEWORK_EXECUTABLE_PATH"
echo "i386 architecture removed"
rm "$FRAMEWORK_EXECUTABLE_PATH"
mv "$FRAMEWORK_TMP_PATH" "$FRAMEWORK_EXECUTABLE_PATH"
fi
if $(lipo "$FRAMEWORK_EXECUTABLE_PATH" -verify_arch "x86_64") ; then
lipo -output "$FRAMEWORK_TMP_PATH" -remove "x86_64" "$FRAMEWORK_EXECUTABLE_PATH"
echo "x86_64 architecture removed"
rm "$FRAMEWORK_EXECUTABLE_PATH"
mv "$FRAMEWORK_TMP_PATH" "$FRAMEWORK_EXECUTABLE_PATH"
fi
;;
esac
echo "Completed for executable $FRAMEWORK_EXECUTABLE_PATH"
echo $(lipo -info "$FRAMEWORK_EXECUTABLE_PATH")
done
cp Resources/* "$TARGET_BUILD_DIR/$UNLOCALIZED_RESOURCES_FOLDER_PATH" > /dev/null 2> /dev/null
exit 0
```

Initializing the SDK into the App Delegate File

1. **Note:** For API authentication pass the following headers:

   - `header_key_for_API = "X-Voltmx-App-API-Key"`

   - `header_value_for_API = "<Api Key Header>"`

The following code executes the Subscription with the Kony SDK and posts the Success Message along with KSID and Message Successful Payload for Simple Push Notification:

```
- (void)application:(UIApplication *)app didRegisterForRemoteNotificationsWithDeviceToken:(NSData
 *)deviceToken
{
 [client initializeInBackgroundWithAppKey: @"<Primary App Key>"
```

```
appSecret: @"<Primary App secret>"
serviceURLString: @"<Service URL>"
completion: ^ (BOOL succeeded, NSError * error)
}
```

Unica Deliver administrator can access Appkey, AppSecret and ServiceURL from the following
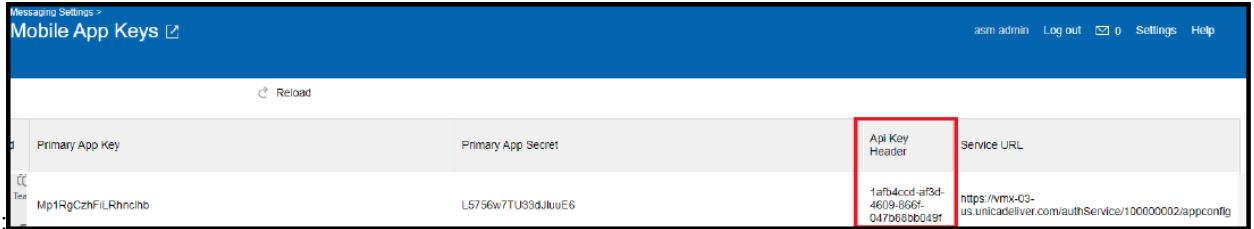
screen:

2. The following sample is the Basic Push Notification APNS payload:

```
APNS Payload:
{
 aps =
 {
  alert =
  {
   body = "Hello ";
   title = Hi;
  };
  badge = 0;
  sound = default;
 };
 mid = 4895850506347263356;
 msgEntryId = 2470;
 responseData =
 {
  appId = 01234;
  appRef = qtest;
  miId = 482;
 };
 seqNum = 18;
}
```

The following sample is the In-App Push Notification APNS payload:

```
{
 aps =
 {
  alert =
  {
   body = "Hello Username";
   title = Hi;
  };
  badge = 0;
  sound = default;
 };
 isRichPush = 1;
 mid = 4895850843398559662;
 msgEntryId = 2471;
 responseData =
 {
  appId = 01234;
  appRef = qtest;
```

```
  miId = 482;
 };
 seqNum = 18;
}
```

For Basic Push with Image, add mutable content within the Alert Body so that the Image is attached with the Push Notification.

```
{
 "aps":
 {
  "category": "content_added_notification",
  "alert":
  {
   "title": "Photos",
   "body": "Antoine added something new - take a look"
  },
  "mutable-content": 1
 },
 "image_url": "https://www.example.com/image_url"
}
```

3. When user clicks on the Notification call the following method and pass the HTML content through the Webview:

```
-(void)updatePushDetailsWithNetworkCallMesagesRichcontent:(NSNotification *)notification
{
 NSString* myString = [[NSUserDefaults standardUserDefaults] valueForKey:@"mid"];
 NSString *pushUrlStr = [NSString stringWithFormat:@"%@/messages/richcontent/%@",
 networkUrlStr ,myString];
 //Here We need to pass the HTML content along With the WEBVIEW and We need to use Dispatch Main Queue
 to pass the Webview inside.
}
```

Verifying custom events

About this task

The following are the custom events:

- `push.opened`
- `inApp.opened`
- `push.dismissed`
- `inApp.dismissed`
- `push.enabled`
- `push.disabled`
- `app.installed`
- `app.uninstalled`
- `custom`
- `push.delivered`
- `inApp.delivered`

1. `push.opened`, `inApp.opened`, `push.dismissed`, `inApp.dismissed`, `push.enabled`, `push.disabled`, `app.installed`, `app.uninstalled`, and `custom`

   For these events, the API remains the same and only the POST body **type** parameter and **data** parameter changes:

   ```
   static NSString *networkUrlStr =  @"Base URL";
   NSString *urlString = [NSString stringWithFormat:@"%@/response", networkUrlStr];
   ```

   The API posts the body as shown in the following response and the data object receives notification payload as `responseData`.

   `type` - Depends on the Notification type.

   - For simple notification it is `push.opened` / `push.dismissed` / `push.delivered`.
   - For in-app notification, it is `inApp.opened` / `inApp.dismissed` / `inApp.delivered`.

2. **Push-related events**: `push.opened`, `inApp.opened`, `push.dismissed`, `inApp.dismissed`, `push.delivered`, and `inApp.delivered`

   ```
   {
    type= push.opened / inApp.opened / push.dismissed / inApp.dismissed / push.delivered / inApp.delivered

    data =
    {
     appId = ef7c9e33-09c1-4d05-8248-351537f453bd;
     appRef = UnicaPartitionName;
     messageId = <mid>;
     miId = 3;
     msgEntryId = 2273;
     seqNum = 1;
    };
    happenedAt = "2024-02-09, 06:08:42"; // Use the date format yyyy-MM-dd, hh:mm:ss (GMT) for responses to
    be correctly tracked from Unica Deliver
    ksid = "7743349406375345629";
    type = "inApp.dismissed";
    userId = "name@customer.com";
    uuid = "UUID.randomUUID().toString()";
   }
   ```

   Response of `api/v1/response :/n`

   ```
   {
    id = "";
    message = "Response received by server";
   }
   ```

3. **App-related events**: `push.enabled`, `push.disabled`, `app.installed`, and `app.uninstalled`

   ```
   {
    data:
    {
     appId: ef7c9e33-09c1-4d05-8248-351537f453bd;
     appRef: unicaPartitionName;
    }
    happenedAt: 2024-02-09, 06:08:42; // Use the date format yyyy-MM-dd, hh:mm:ss (GMT) for responses to be
    correctly tracked from Unica Deliver;
    ksid: 8206033607781965226;
    type: push.enabled / push.disabled / app.installed / app.uninstalled;
    userId: name@customer.com;
    uuid: 56D43AD6-83E9-40ED-9952-A5B56558563B;
   }
   ```

Response of `api/v1/response :/n`

```
{
  id = "56D43AD6-83E9-40ED-9952-A5B56558563B";
  message = "Response received by server";
}
```

4. **custom**

   Once the device receives the in-App push notification, and if the user clicks the notification, call the `getRichPushdDataAPICall` API for HTML response to show the pop-up. In the response, check whether the `data-custom-event` tag is available or not. If it is available, parse the `data-custom-event` tag.

   Example HTML:

   ```
   <a data-custom-event="\&quot;RESP_CLICK_EVENT\&quot;" href="\" none="
     style="\&quot;text-decoration:">Click</a></div>
   ```

   Payload:

   ```
   {
    "type": "RESP_CLICK_EVENT", (data-custom-event)
    "uuid": "UUID.randomUUID().toString()",
    "happenedAt": "2024-02-09, 06:08:42", // Use the date format yyyy-MM-dd, hh:mm:ss (GMT) for responses
    to be correctly tracked from Unica Deliver ,
    "ksid": "7743349406375345629",
    "userId": "name@customer.com",
    "data":
    {
     "messageId": 1, <mID>
     "seqNum":1,
     "miId":3,
     "appId":"ef7c9e33-09c1-4d05-8248-351537f453bd",
     "appRef":"unicaPartitionName",
     "msgEntryId":"2404"
    }
   }
   ```

5. **Deep Link**

   Once the device receives the in-App push notification, and if the user clicks the notification, call the `getRichPushdDataAPICall` API for HTML response to show the pop-up. In the response, check whether the `data-deeplink` tag is available or not. If it is available, parse the `data-deeplink` tag.

   If user clicks the Deep link button, POST the body as shown in the following sample:

   ```
   <a data-custom-event="deeplinktest" data
     deeplink="{&quot;name&quot;:&quot;main_window&quot;,&quot;width&quot;:500,&quot;title&quot;:&quot;Sampl
   e Konfabulator Widget&quot;,&quot;height&quot;:500}" href="" style="text-decoration:
     none;">Button</a></div>
   ```

   ✏️ **Note:** Deliver does not support personalization field in Deep Link.

6. **Getting user information details Like Badge, Sound, and any other payload from APNS**

   Execute the following method:

   ```
   - (void)application:(UIApplication *)application didReceiveRemoteNotification:(NSDictionary *)userInfo
     fetchCompletionHandler:(void (^)(UIBackgroundFetchResult))completionHandler {
     NSLog(@"APNS Payload **: %@",userInfo);
   ```

```
   NSError *error;
   NSDictionary *jsonDictionary = [NSDictionary dictionaryWithDictionary:userInfo];
   if (!error)
   {
    [self handleAPns: jsonDictionary];
   }
   else
   {
    NSLog(@"message_id not found in the dictionary");
   }
}
```

Checking Push on iOS Simulator

1. Use an APNS Extensions file and place the earlier mentioned notifications payload inside the file and drag and drop inside the simulator.
2. Push Notification works on iOS 10 and above. Simulators pops up the Push messages.

Creating Certificates to Run on Real time devices

1. In the developer account, within the App Identifier Section, create an App ID using Bundle Identifier for each app, or Project.
2. Create an Adhoc and Distribution certification using the App ID and Install it inside the keychain in the local system.
3. Create a `.p12` Certificate for push notification and install it in the Keychain.
4. Export the `.p12` format cert and install it in the XCode.
5. Enable the Push Notification under Capabilities.
6. Under the Push Notification section, create a separate certificate for Development mode and for Production mode, like Sandbox and Production mode.
7. Using Keychain access, create a CSR certificate from the local system.

Creating apps using Unica Deliver

You can create an app in Unica Deliver for sending Push notifications using Android or iOS.

Before you begin

To create an app, you must have the `DeliverAdmin` role. For details, see the Unica Platform Administrator's Guide.

About this task

To create an App complete the following steps:

1. Navigate to **Settings > Messaging Settings**.
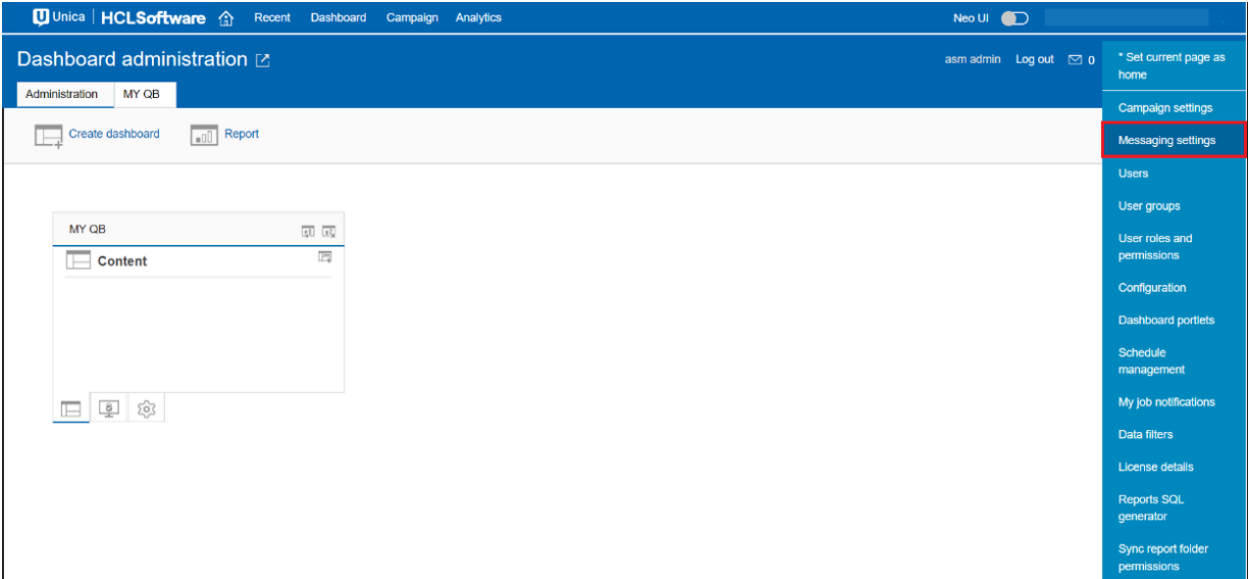
   Result

   The **Messaging Settings** page appears.

2. Within the **Mobile Push Notification Settings** section, select **Display the list of mobile app keys**.

   Result

   The **Mobile App Keys** page appears.

3. Select **Add a mobile application key**.

4. Provide values for the following fields:

- **App name** - A unique app name. Also used to identify the app in Deliver MX (Mobile eXperience) Push.
- **Description** - A description for the app.
- **App key type** - By default, the value is `production`.
- **App operating system** - The platform for which the app is created. Select either `Android` or `iOS`.
- **Provider Account** - The onboarding service provider. Select `kony` for onboarding via Deliver MX Push.
- **Default time zone** - The applicable time zone.

5. In *Step 4*, for the field App operating system, if the you select:

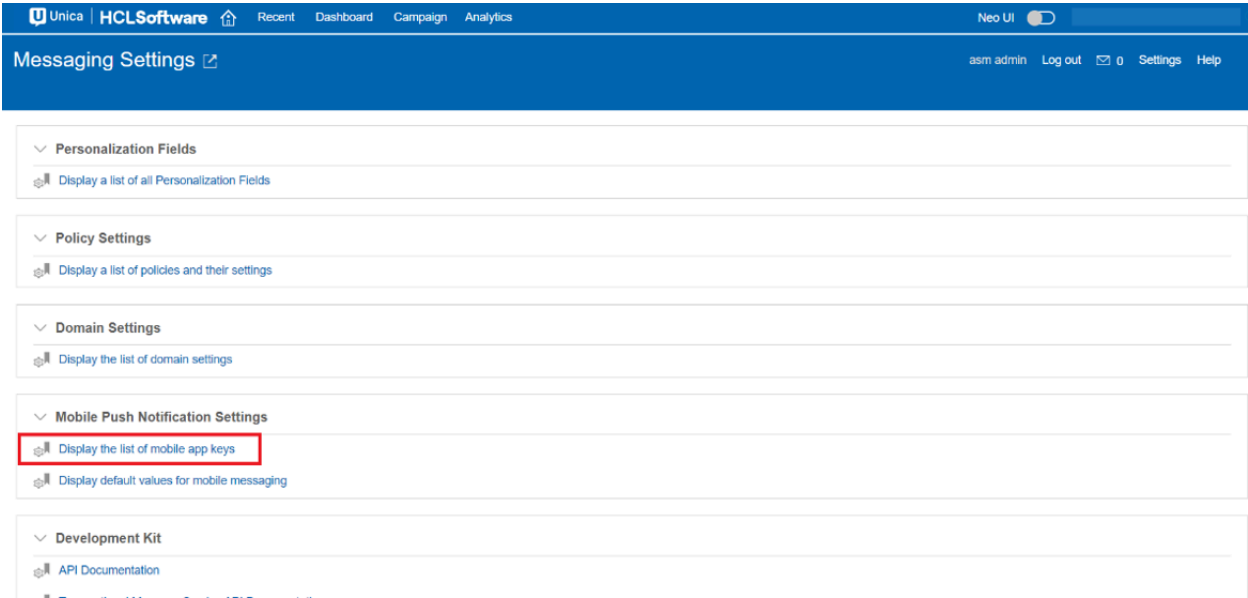a. Android, provide values for the following fields:

- **Google API key** - Mandatory. The API key required for Android apps that you must retrieve from Google Developer Tools.

b. iOS, provide values for the following fields:

- **Certificate file** - Mandatory. The iOS app certificate file. Click the **Choose File** button, navigate to the location on the system where the iOS certificate file exists and select the file. The certificate file requires a certificate password for access.

> 📝 **Note:** Currently, Unica Deliver only supports Production certificate.

- **Certificate password** - Mandatory. The password to access the certificate file.

6. For the field Security Policies, select one of the following options (the selection makes the app available to all users having these security policies assigned to their role):

- All existing and future policies
- Only the selected policies
  - If you select **Only the selected policies**, you are selecting policies listed in **Available policies** and adding the required/appropriate policies to the **Selected policies** list box. The policies added to the **Selected policies** list box are the policies applied to the app.
7. Click **Save Changes** to create an app.

Results

Upon saving the changes, an app with the same name is created on Deliver MX Push as a Deliver MX Push app. Once the Deliver app creation is successful, it appears in the Mobile App Keys page. You will see the following details, related to the app, in the Mobile App Keys page:

- **App Name** - Name that identifies the app.
- **Description** - Description about the app.
- **App Key Type** - By default, the value is `production`.
- **Operating System** - Displays the operating system compatibility of the app. Values are either `Android` or `iOS`.
- **Policy Name** - Displays the policies assigned to the app.
- **App Key** - An alphanumeric identifier for the app.
- **iOS Certificate Status** - Shows the iOS Certificate Status as `Valid` or `Invalid`.
- **File Upload Status** - Shows the upload status as `Success` or `Failed`.
- **Primary App Key** - The value needed for Android or iOS SDK integration.
- **Primary App Secret** - The value needed for Android or iOS SDK integration.
- **App Key Header** - The value needed to authenticate REST APIs from the app.
- **Service URL** - The value needed for Android or iOS SDK integration.

Chapter 9. Configurations for Unica Deliver

The Unica Platform provides various configuration properties to modify the behavior and appearance of Deliver. Some configuration properties are set during installation. You can change configuration properties at any time.

After you update the Campaign or Deliver configurations, you must restart the Response and Contact Tracker (RCT) and the web application server that hosts Campaign.

Characteristic or feature	Configuration property (including path)
Enable or disable Deliver in the Campaign partition. See Campaign | partitions | partition[n] | server | internal on page 69.	`Campaign	partitions	partition[n]	server	internal`
Characteristics of email recipient lists. See Campaign | partitions | partition[n] | Deliver on page 67.	`Campaign	partitions	partition[n]	Deliver`
URLs required to connect to HCL Unica hosted services. See Deliver | serverComponentsAndLocations | hostedServices on page 73.	`Deliver	serverComponentsAndLocations	hostedServices`
Database and account access credentials for connecting to HCL Unica hosted services. See Deliver | partitions | partition[n] | hostedAccountInfo on page 77	`Deliver	partitions	partition[n]	hostedAccountInfo`
Database access and schema settings for the Deliver system tables. See Deliver | partitions | partition[n] | dataSources | systemTables on page 78	`Deliver	partitions	partition[n]	dataSources	systemTables`
Location of a script that runs in response to the actions or status of the Recipient List Uploader. (optional) See Deliver | partitions | partition[n] | recipientListUploader on page 81	`Deliver	partitions	partition[n]	recipientListUploader`
Settings related to data download, processed by the Response and Contact Tracker (RCT). See Deliver | partitions | partition[n] | responseContactTracker on page 82	`Deliver	partitions	partition[n]	responseContactTracker`

Characteristic or feature	Configuration property (including path)
Support for presenting lists of personalized data in Deliver based on dimension tables in Campaign. See Configuring support for dimension tables on page 66.	`Campaign	partitions	partition[n]	Deliver	oltDimTableSupport`
Support for tracking mailing execution history. See Deliver	partitions	partition[n]	responseContactTracker on page 82	`Deliver	partitions	partition[n]	responseContactTracker` See the **enableExecutionHistoryDataTracking** parameter.

For more information about working with configuration properties, see the Unica Platform Administrator's Guide.

Configuring access to additional mailing execution history

You can request that Unica provide additional data for mailing execution history. Access to additional mailing execution history data is available by request from Unica and by updating the Deliver configuration. Data for mailing execution history is recorded your local Deliver system tables in the `UACE_ExecHistory` table to describe completed mailing runs.

Before you begin

To download additional mailing run data, you must update the configuration property `enableExecutionHistoryDataTracking`. By default, `enableExecutionHistoryDataTracking` is not exposed in the Deliver configuration properties.

You can display this configuration property in your local Deliver installation by running the `switch_config_visibility.bat` script, which is in the `Deliver\tools` directory. The following types of records are available in additional mailing execution history.

About this task

- Message subject line
- From address
- User that updated the mailing
- Document description
- Mailing save date

1. Request access to additional mailing execution history data. To request access, contact your Unica Deliver Services team via HCL technical support.
2. Update the Deliver configuration. Configure the following configuration property.

    `Affinium|deliver|partitions|partition1|responseContactTracker|` **enableExecutionHistoryDataTracking**

    Set **enableExecutionHistoryDataTracking** to **True**.

What to do next

You can query the Deliver system tables to retrieve mailing run information from the `UACE_ExecHistory` table.

For more information about the Deliver system tables, see the *Unica Deliver System Tables and Data Dictionary*.

Support for Campaign offer integration

Unica Deliver supports adding offers that are configured in Campaign to personalized email created in Deliver .

The offers are based on offer templates that are configured in Unica Campaign. To support integrating Campaign offers into personalized email, you must update the `contactAndResponseHistTracking` property in the Campaign configuration and complete other configurations in Campaign.

For more information about how to configure support for offer integration, see topics for Deliver offer integration in the Unica Campaign Administrator's Guide.

Configuring support for dimension tables

To support certain features that are provided by advanced scripts for email., the `oltDimTableSupport` configuration property must be set to **True**.

About this task

Deliver provides advanced scripts to create email messages that display lists of personalized information. These lists require associating dimension tables created in Campaign with an Output List Table (OLT) that defines the email recipient list. Output List Tables are created in the Deliver schema.

The `oltDimTableSupport` configuration property controls support for creating dimension tables in the Deliver schema. When the value for this property is set to `True`, an OLT can use information provided in a dimension table.

Complete the following steps to update the `oltDimTableSupport` property.

For more information about how marketers use advanced scripts to create data tables, see the *Unica Deliver User's Guide*.

1. Go to **Settings** > **Configuration** > **Campaign** > **Partitions** > **partition[n]** > **Deliver**
2. Click **Edit Settings** and set the value of the `oltDimTableSupport property` to `True`.

Configuring access to local Deliver system tables

Deliver components must be able to access the Deliver system tables in the Campaign schema. You must create and configure a system user that can access the system tables automatically. The system user that was configured during the installation of Campaign already has the necessary access to the Campaign schema.

About this task

> **Note:** If your installation contains multiple partitions, you must complete this task for each partition. You cannot share system users across partitions.

If you want to use a different system user to access the Deliver system tables, you must create a new system user in the Platform and create new platform data source with access to the Campaign schema.

1. In the Deliver configuration, specify a system user that accesses the database that hosts the Campaign schema.

   You can create a new user or specify an existing user. The system user that you set up for Campaign already has access to the Campaign schema.

   Use the configuration property `Deliver > partitions > partition [n] < dataSources > systemTables > asmUserForDBCredentials`.

   By default the specified user is `asm_admin`.

2. In the Deliver configuration, specify the data source that is configured to contain the user name and password that is required to access the database that hosts the Campaign schema.

   You can use the data source that was created to access the Campaign schema when you installed Campaign.

   Use the configuration property `Deliver > partitions > partition [n] < dataSources > systemTables > amDataSourceForDBCredentials`.

Configuration properties for Deliver

You access Deliver configuration properties from the Settings menu in the Platform. Properties for configuring Deliver are contained in the Campaign and Deliver configuration categories.

To access the configuration properties, navigate to **Settings > Configurations**. The Configurations page lists all of the available configuration properties for your HCL Unica installation.

Campaign | partitions | partition[n] | Deliver

Define properties in this category to define characteristics of recipient lists and specify the location of resources that upload the lists to HCL Unica.

DeliverPluginJarFile

> **Description**
>
> Complete path to the location of the file that operates as the Recipient List Uploader (RLU). This plug-in to Campaign uploads OLT data and associated metadata to the remote services hosted by Unica. The location that you specify must be the full local directory path in the file system for the computer that hosts the Campaign web application server.
>
> The Unica installer populates this setting automatically for the default partition when you run the installer. For other partitions, you must configure this property manually. Because there is only one RLU for each Deliver installation, all partitions must specify the same location for the RLU.
>
> Do not change this setting unless Unica instructs you to do so.
>
> **Default value**
>
> No default value defined.
>
> **Valid Values**
>
> Full local directory path to where you installed the Campaign web server.

defaultSeedInterval

Description

The number of messages between seed messages if `defaultSeedType` is `Distribute list`.

Default value

`1000`

defaultSeedType

Description

The default method that Deliver uses to insert seed addresses into a recipient list.

Default value

`Distribute IDS`

Valid Values

- `Distribute IDS` - Distribute IDs evenly, based on the size of the recipient list and the number of seed addresses available, inserts seed addresses at equal intervals throughout the entire recipient list.
- `Distribute list` - Insert seed address for every `defaultSeedInterval` IDs in main list. Inserts the entire list of available seed addresses at specified intervals throughout the recipient list. You must specify the interval between insertion points.

oltTableNamePrefix

Description

Used in the generated schema for the output list table. You must define this parameter.

Default value

OLT

Valid Values

The prefix can contain no more than 8 alphanumeric or underscore characters, and must start with a letter.

oltDimTableSupport

Description

This configuration parameter controls the ability to add dimension tables to output list tables (OLT) created in the Deliver schema. Dimension tables are required to use advanced scripting for email to create data tables in email messages.

You must set this property to `True` (by default it is `True`) so that marketers can create dimension tables when they use the Deliver process to define a recipient list. For more information about creating data tables and working with advanced scripts for email, see the Unica Deliver User Guide.

You must set this property to `False`, if you are using dimension table fields to output in olt and want to use these dimension fields in communication as a personalisation field.

Default value

`True`

Valid Values

True | False

Campaign | partitions | partition[n] | server | internal

Properties in this category specify integration settings and the internalID limits for the selected Campaign partition. If your Campaign installation has multiple partitions, set these properties for each partition that you want to affect.

internalIdLowerLimit

Configuration category

`Campaign|partitions|partition[n]|server|internal`

Description

The `internalIdUpperLimit` and `internalIdLowerLimit` properties constrain the Campaign internal IDs to be within the specified range. Note that the values are inclusive: that is, Campaign may use both the lower and upper limit.

Default value

`0` (zero)

internalIdUpperLimit

Configuration category

`Campaign|partitions|partition[n]|server|internal`

Description

The `internalIdUpperLimit` and `internalIdLowerLimit` properties constrain the Campaign internal IDs to be within the specified range. The values are inclusive: that is, Campaign may use both the lower and upper limit. If Unica Collaborate is installed, set the value to `2147483647`.

Default value

`4294967295`

deliverInstalled

Configuration category

`Campaign|partitions|partition[n]|server|internal`

Description

Indicates that Deliver is installed. When you select `Yes`, Deliver features are available in the Campaign interface.

The Unica installer sets this property to `Yes` for the default partition in your Deliver installation. For additional partitions where you installed Deliver, you must configure this property manually.

Default value

```
No
```

Valid Values

```
Yes | No
```

Legacy_campaigns

Configuration category

```
Campaign|partitions|partition[n]|server|internal
```

Description

For this partition, enables access to campaigns created before Unica Plan and Campaign were integrated. Applies only if **MO_UC_integration** is set to `Yes`. Legacy campaigns also include campaigns created in Campaign 7.x and linked to Plan 7.x projects. For more information, see the *Unica Unica Plan and Campaign Integration Guide*.

Default value

No

Valid Values

Yes | No

Campaign | partitions | partition[n] | Deliver | contactAndResponseHistTracking

Use the properties in this category to configure Deliver offer integration with Unica Campaign for the current partition.

etlEnabled

Description

Campaign uses its own ETL process to extract, transform, and load offer response data from the Deliver tracking tables into the Campaign contact and response history tables.

The ETL process coordinates information across the necessary tables, including `UA_UsrResponseType` (Campaign response types) and `UA_RespTypeMapping` (mapping of response types between Campaign and Deliver).

Setting the value to `Yes` ensures that information about Deliver offer contact and response history is coordinated between Campaign and Deliver. For example, email response data will be included in Campaign reports.

> **Note:** You must also set `Campaign | partitions | partition[n] | server | internal | DeliverInstalled`
> to `Yes` for this partition or the ETL process will not run.

> **Tip:** If you want to monitor the progress of the ETL, enable `Campaign | monitoring |`
> `monitorEnabledForDeliver`.

Default value

`No`

Valid values

`Yes | No`

runOnceADay

Description

Indicate whether the ETL process should run only once a day.

If the value is `Yes`: You must specify a **startTime**; the ETL job then runs until all of the records are processed;
and the **sleepIntervalInMinutes** is ignored.

If the value is `No`: The ETL job starts as soon as the Campaign web server starts. The ETL job stops after all of
the records are processed, then waits for the time specified by **sleepIntervalInMinutes**.

Default value

`No`

Valid values

`Yes | No`

batchSize

Description

The ETL process uses this parameter to fetch records that have been downloaded by the RCT into the local
Deliver system tables. Because large values can impact performance, the list of available values is restricted to
the valid values shown below. If you anticipate large volumes of records, adjust the **batchSize** together with the
sleepIntervalInMinutes to process records at regular intervals.

Default value

`100`

Valid values

100 | 200 | 500 | 1000

sleepIntervalInMinutes

Description

Specify the interval in minutes between ETL jobs. This option determines the wait time after a job finishes. The ETL process waits for this duration before starting the next job. Multiple jobs can run synchronously and there may be multiple ETL jobs per partition.

If **runOnceADay** is `Yes`, you cannot set a sleep interval.

Default value

`60`

Valid values

Positive integers

startTime

Description

Specify a time to start the ETL job. You must use the English locale format to specify the start time.

Default value

`12:00:00 AM`

Valid values

Any valid time in the format `hh:mm:ss AM/PM`

notificationScript

Description

An optional executable or script file that is run after each ETL job is done. For example, you might want to be notified of the success or failure of each ETL job, for monitoring purposes. The notification script runs every time the ETL job for a given partition finishes running.

The parameters passed in to this script are fixed and cannot be changed. The following parameters can be used by the script:

- etlStart: The start time of ETL in number of milliseconds.
- etlEnd: The end time of ETL in number of milliseconds.
- totalCHRecords: Total number of contact records processed.
- totalRHRecords: Total number of response histroy records processed.
- executionStatus: Execution status of the ETL with value 1 (failed) or 0 (succeeded).

Default value

No default value defined.

Valid values

Any valid path that the Campaign server can access with Read and Execute permissions. For example: `D:\myscripts\scriptname.exe`

Deliver | serverComponentsAndLocations | hostedServices

Define properties to specify the URLs for connecting to HCL Unica hosted services. Deliver uses separate connections for uploading recipient lists, metadata that describes recipient lists, and for general communication sent to the hosted environment.

You must change the default values if you are connecting to HCL Unica hosted services through the data center that is established by Unica in the Europe or India. Consult Unica to determine the data center to which you are connected.

uiHostName

Description

The address that Deliver uses for all communication to HCL Unica hosted services, except uploading recipient lists and related metadata.

Default value

`em.unicadeliver.com`

If you are connecting to the Europe data center, change this value to `em-eu.unicadeliver.com`.

If you are connecting to the India data center, change this value to `em-in.unicadeliver.com`.

dataHostName

Description

The address that Deliver uses for uploading metadata that is related to recipient lists to HCL Unica hosted services.

Default value

`em.unicadeliver.com`

If you are connecting to the Europe data center, change this value to `em-eu.unicadeliver.com`.

ftpHostName

Description

The address that Deliver uses for uploading recipient list data (except list metadata) to HCL Unica hosted services.

Default value

`ftp-em.unicadeliver.com`

If you are connecting to Europe datacenter, change this value to `ftp-eu.unicadeliver.com`.

If you are connecting to the India data center, change this value to `em-in.unicadeliver.com`.

If you are connecting to India datacenter, change this value to `ftp-in.unicadeliver.com`.

Deliver | serverComponentsAndLocations | Kafka | RCT

KafkaBrokerURL

Description

Use this property to define IP and port on which Zookeeper or Kafka is running.

Default value

No default value defined.

Valid Values

Any valid Kafka broker URL.

CommunicationMechanism

Description

Specifies the configuration for Kafka client authentication.

Default value

No default value defined.

Valid Values

In the Kafka configurations page, you can select one of the following values for the CommunicationMechanism field depending on your organization's Kafka server's stream security.

- NO_SASLPLAINTEXT_SSL
- SASL_PLAINTEXT
- SSL
- SASL_PLAINTEXT_SSL

sasl.mechanism

Description

Specifies the Kafka client authentication.

Default value

No default value defined.

Valid Values

You can select one of the following values depending on Kafka server's authentication configurations.

- SASL_PLAINTEXT
- SASL_PLAINTEXT_SSL
- SSL

UserForKafkaDataSource

Description

Specifies the HCL Unica user that references the data source that contains the Kafka services access credentials. You configure this value when you create a system user.

Default value

No default value defined.

Valid Values

Any valid user that reference Kafka datasource

sasl.jaas.config.dataSource

Description

Kafka uses the Java Authentication and Authorization Service (JAAS) for SASL configuration. You must provide JAAS configurations for all SASL authentication mechanisms.

Default value

No default value defined.

Valid Values

Refer "listener.name.sasl_ssl.plain.sasl.jaas.config" as in kafka_home/server.properties

truststore.location

Description

Specifies the path of "kafka.server.truststore.jks"

Default value

No default value defined.

Valid Values

Path of "kafka.server.truststore.jks" file as mentioned in kafka_home/server.properties/ssl.truststore.location.

truststore.password.dataSource

Description

Specifies the Platform data source that contains the Kafka truststore login credentials. You configure this value when you create a system user.

Default value

No default value defined.

Valid Values

Path of "kafka.server.keystore.jks" as mentioned in `kafka_home/server.properties/ssl.keystore.location.`

keystore.password.dataSource

Description

Specifies the Platform data source that contains the Kafka keystore login credentials. You configure this value when you create a system user.

Default value

No default value defined.

Valid Values

Datasource which contains the kafka keystore login credentials

key.password.dataSource

Description

Specifies the Platform data source that contains the Kafka key login credentials. You configure this value when you create a system user.

Default value

No default value defined.

Valid Values

Datasource which contains the kafka key login credentials

ssl.endpoint.identification.algorithm

Description

Specifies the endpoint identification algorithm used by clients to validate server host name. Disable server host name verification by setting ssl.endpoint.identification.algorithm to an empty string.

Default value

empty

Valid Values

Refer ssl.endpoint.identification.algorithm as mentioned in kafka_home/server.properties.

KafkaPartitionCount

Description

Partitions are the main concurrency mechanism in Kafka. A topic is divided into one or more partitions, enabling producer and consumer loads to be scaled. Specifically, a consumer group supports as many consumers as partitions for a topic.

Default value

2

Valid values

Number of RCT instance * 2

Each RCT instance will have only two consumers per topic. This can be increased by increasing Kafka partitions from configuration and initiating multiple RCT instances.

For example: If you there are four Kafka partitions, then two RCT instances must be started. For six Kafka partitions there must be three RCT instances and so on. Each RCT instances must run on different nodes. If Kafka partitions are increased all the RCT instances must be restarted.

Replicafactor

Description

A replication factor is the number of copies of data over multiple brokers. The replication factor value should be greater than 1 always. This helps to store a replica of the data in another broker from where the user can access it.

Default value

1

Valid values

The number of copies of data you need to keep over multiple brokers.

IsKafkaEnabled

Description

To use RCT with Kafka, set the value to `True`. To use RCT without Kafka, set the value to `False`.

Default value

`False`

Valid values

`True | False`

Deliver | partitions | partition[n] | hostedAccountInfo

Define properties in this category to define user credentials for the database that contains account information that is required to access HCL Unica hosted services. Values that you specify here must be defined as user settings in the Platform.

amUserForAcctCredentials

Description

Use this property to specify the Platform user that contains a Platform data source that specifies the account access credentials that are required to access HCL Unica hosted services.

Default value

asm_admin

Valid Values

>   Any Platform user.

amDataSourceForAcctCredentials

Description

>   Use this property to specify the Platform data source that defines login credentials for HCL Unica hosted services.

Default value

>   UNICA_HOSTED_SERVICES

Valid Values

>   A data source that is associated with the user you specify in `amUserForAcctCredentials`

Deliver | partitions | partition[n] | dataSources | systemTables

This category contains configuration properties that define the schema, connection settings, and login credentials for the database that contains the Deliver system tables in your network environment.

type

Description

>   Type of database that hosts the Deliver system tables.

Default value

>   No default value defined. You must define this property.

Valid Values

>   - `SQLSERVER`
>   - `ORACLE`
>   - `DB2`
>   - `MARIADB`
>   - `ONEDB`
>   - `POSTGRESQL`

schemaName

Description

>   Name of the database schema for the Deliver system tables. This is the same as the schema name for the Campaign system tables.
>
>   You must include this schema name when referencing system tables in scripts.

Default value

>   dbo

jdbcBatchSize

Description

The number of execution requests JDBC runs on the database at a time.

Default value

10

Valid Values

An integer greater than 0.

jdbcClassName

Description

JDBC driver for system tables as defined in your Campaign web server.

Default value

No default value defined. You must define this property.

Example

```
org.postgresql.Driver
```

jdbcURI

Description

JDBC connection URI for system tables as defined in your Campaign web server. The format is as follows:

```
jdbc:<database-identifier>://<host-name>:<port-number>/<database-name>
```

Default value

No default value defined. You must define this property.

Example

```
jdbc:postgresql://example-system:4321/postgres
```

asmUserForDBCredentials

Description

Use this property to specify an HCL Unica user that will be allowed to access the Deliver system tables.

Default value

No default value defined. You must define this property.

Valid Values

Any user defined in the Platform. This should typically be the name of the system user for Campaign

amDataSourceForDBCredentials

Description

Use this property to specify the data source that defines login credentials for the database that contains the Deliver system tables. This can be the same as the data source for the Campaign system tables.

Default value

UA_SYSTEM_TABLES

Valid Values

A Platform data source associated with the HCL Unica user you specify in `asmUserForDBCredentials`

The data source specifies a database user and credentials used to access the Deliver system tables. If the default schema for the database user is not the schema that contains the system tables you must specify the system table schema in the JDBC connection used to access the system tables.

poolAcquireIncrement

Description

When the database connection pool runs out of connections, the number of new connections Deliver creates for the system tables. Deliver creates new connections up to the number specified in `poolMaxSize`.

Default value

1

Valid Values

An integer greater than 0.

poolIdleTestPeriod

Description

The number of seconds Deliver waits between testing idle connections to the Deliver system tables for activity.

Default value

100

Valid Values

An integer greater than 0.

poolMaxSize

Description

The maximum number of connections Deliver makes to the system tables. A value of zero (0) indicates there is no maximum.

Default value

100

Valid Values

> An integer greater than or equal to 0.

poolMinSize

Description

> The minimum number of connections Deliver makes to the system tables.

Default value

> 10

Valid Values

> An integer greater than or equal to 0.

poolMaxStatements

Description

> The maximum number of statements that Deliver stores in the PrepareStatement cache per connection to the system tables. Setting poolMaxStatements to zero (0) disables statement caching.

Default value

> 0

Valid Values

> An integer equal to or greater than 0.

timeout

Description

> The number of seconds Deliver maintains an idle database connection before dropping the connection.
>
> If `poolIdleTestPeriod` is greater than 0, Deliver tests all idle, pooled, but unchecked-out connections, every `timeout` number of seconds.
>
> If `poolIdleTestPeriod` is greater than `timeout`, the idle connections are dropped.

Default value

> 100

Valid Values

> An integer equal to or greater than 0.

Deliver | partitions | partition[n] | recipientListUploader

This configuration category contains an optional property for the location of a user-defined script that runs in response to the actions or status of the Recipient List Uploader.

pathToTriggerScript

Description

You can create a script that triggers an action in response to the upload of a recipient list to HCL Unica hosted services. For example, you can create a script to send an email alert to the list designer when the list upload completes successfully.

If you define a value for this property, Deliver passes status information about the Recipient List Uploader to the specified location. Delivertakes no action if you leave this property blank.

Default value

No default value defined.

Valid Values

Any valid network path.

Deliver | partitions | partition[n] | responseContactTracker

Properties in this category specify behavior for the Response and Contact Tracker (RCT). The RCT retrieves and processes data for email contacts, email delivery, and recipient responses, such as link clicks and opens.

pauseCustomerPremisesTracking

Description

Deliver stores contact and response data in a queue in HCL Unica hosted services. This property allows you to instruct the RCT to temporarily stop retrieving data from HCL Unica hosted services. When you resume tracking, the RCT downloads the accumulated data.

Default value

False

Valid Values

True | False

waitTimeToCheckForDataAvailability

Description

The RCT periodically checks for new data regarding email contacts or recipient responses. This property allows you to specify how often, in seconds, the RCT checks for new data in HCL Unica hosted services. The default value is 300 seconds, or every 5 minutes.

Default value

300

Valid Values

Any integer greater than 1.

perfLogInterval

Description

This property allows you to specify how often the RCT logs performance statistics to a log file. The value you enter determines the number of batches between log entries.

Default value

10

Valid Values

An integer greater than 0.

enableSeparatePartialResponseDataTracking

Description

This property determines if Deliver forwards partial email response data to the tracking tables in your local Deliver installation.

Deliver requires the Mailing Instance ID and Message Sequence Number to properly attribute email responses. When you enable separate partial response data tracking, Deliver places the incomplete responses in separate local tracking tables where you can review them or perform additional processing.

Default value

True

Valid Values

True | False

enableExecutionHistoryDataTracking

Description

This property controls whether you can download additional mailing execution history data from HCL Unica.

By default, this property is set to **False**, to prevent download of additional data. When you set this property to **True**, you can download data about mailing runs that is not ordinarily entered to the Deliver system tables. You can use this supplementary information to help automate mailing and database management.

This property is hidden by default. You can display this configuration property in your local Deliver installation by running the `switch_config_visibility.bat` script, located in the `Deliver\tools` directory.

Access to mailing execution history data is available by request from Unica. To request access to additional mailing execution history data, contact Unica Deliver Services team via HCL technical support.

Default value

False

Valid Values

True | False

Deliver | serverComponentsAndLocations | internetResponseServers | contactAndResponseEventsTracking

Following configurations are added to global configuration in OD Platform - Category

getPartitionTopicName

Description

The topic name for partition names from Runapp to ResponseEvent module.

Default value

–

Valid values

–

failedResponseEventTrackingName

Description

The topic name for events which failed to be sent over webhook to on-premises.

Default value

–

Valid values

–

Chapter 10. Utilities for Deliver

Deliver provides several scripts that you use to administer Deliver functions.

You can use the software utilities described in this section for a variety of startup and administration functions. In addition to software utilities used with Unica Platform, Unica Deliver uses utilities are specific to Deliver and you use them only to manage Deliver components.

For more information about other utilities available for your HCL Unica installation, see the *Unica Platform Administrator's Guide*.

The RLU script

Use the RLU script to check the status of the Recipient List Uploader (RLU).

📝 **Note:** You cannot use this script to start or stop the RLU. Use this script to check connectivity between on premise and on demand components.

RLU script is located at `<Deliver Install Home>/bin` folder. The Deliver directory is a subdirectory in the Campaign directory.

In UNIX™ or Linux™ environments, run the script as `rlu.sh`.

In Windows™, run the script from the command prompt as `rlu.bat`.

Syntax

```
rlu -c | --check [-h]
```

Commands

-c, --check

Check that the RLU is correctly configured, and that is it connected to HCL Unica.

Options

-h, --help

Display syntax for the script

Example

In a Linux™ environment, determine whether the RLU is connected to HCL Unica hosted services.

```
rlu.sh --check
```

Depending on the status of your system, the output of this command might look like this sample.

```
Configuring Data Source [systemTables]...
Testing configuration for partition partition1
Testing connectivity for partition partition1
Testing user accessibility for partition partition1
```

```
Succeeded. List uploader config and connectivity test
succeeded for partition partition1
```

Deliver Response and Contact Tracker (RCT) script

In order to resolve existing issues in previous version of RCT, Kafka layer is introduced.

Use this script to run and check the status of the Response and Contact Tracker (RCT).

This script is located in the `bin` directory under your Deliver installation. The Deliver directory is a sub-directory in the Campaign directory.

In UNIX™ or Linux™ environments run the script as `rct.sh`.

In Windows™, run the script from the command line as `rct.bat`.

Syntax

```
rct [ start | stop | check ]
```

Commands

start

Start the RCT

stop

Stop the RCT

Options

check

Check the status of the connection between the RCT and HCL Unica hosted services.

Examples

- To start the RCT on Windows™.

  ```
  rct.bat start
  ```

- To stop the RCT on Windows™.

  ```
  rct.bat stop
  ```

- In a Linux™ environment, to determine if the RCT is connected to HCL Unica hosted services.

  ```
  rct.sh check
  ```

  Depending on the status of your system, the output of this command might look like this:

  ```
  C:\<UNICA_HOME>\Campaign\Deliver\bin>rct check
  Testing config and connectivity for partition partition1
  Succeeded | Partition: partition1 - Hosted Services Account ID:
  asm_admin
  ```

The MKService_rct script

The MKService_rct script adds or removes the Response and Contact Tracker (RCT) as a service. Adding the RCT as a service restarts the RCT every time you restart the computer where you have installed the RCT. Removing the RCT as a service prevents the RCT from restarting automatically.

This script is located in the `bin` directory under your Deliver installation.

In UNIX™ or Linux™ environments run `MKService_rct.sh.` with a user that has root permissions or permissions to create daemon processes.

In Windows™, run the script from the command line as `MKService_rct.bat`.

Syntax

```
MKService_rct -install

MKService_rct -remove
```

Commands

-install

Add the RCT as a service

-remove

Remove the RCT service

Examples

- To add the RCT as a Windows™ service.

  ```
  MKService_rct.bat -install
  ```
- To remove the RCT service on UNIX™ or Linux™.

  ```
  MKService_rct.sh -remove
  ```

configTool

The properties and values on the **Configuration** page are stored in the Platform system tables. You can use the `configTool` utility to import and export configuration settings to and from the system tables. For more details, see the Platform Administrator Guide.

Chapter 11. About troubleshooting Deliver

Unica Deliver provides various tools and techniques that you can use to investigate issues related to your Campaign and Deliver installations.

Log files for Deliver

HCL Unica provides several log files that you can review to monitor your Deliver installation and investigate issues.

Deliver log file

This log contains the following types of information regarding information downloaded from HCL Unica hosted services. Located in the `logs` directory under your Deliver installation.

- general mailing information
- mailing instance ID
- link click data
- data for bounced email

Deliver temporary files

This directory contains the data being uploaded.

Located in the `temp` directory under your Deliver installation.

Campaign log files

You can review log files in the following locations for information related to mailing-related activity in Campaign.

- `Campaign\partitions\<partitionN>\logs`

  Various log files relating to flowchart runs, including log entries from any Deliver process contained in the flowchart.

- `Campaign\logs`

  This directory contains `campaignweb.log` that contains information about upload activity performed by the Recipient List Uploader.

Using log4j with Deliver

Deliver uses the Apache log4j utility for logging configuration, debugging, and error information related to the Response and Contact Tracker (RCT) and the Recipient List Uploader (RLU).

For information about changing the system log settings, see:

- The comments in the log4j.xml file.
- The log4j documentation on the Apache web site: https://logging.apache.org/log4j/2.x/manual/index.html

Using log4j with the Recipient List Uploader

When you run the Recipient List Uploader (RLU) utility from the command line, it uses default logger settings.

To change those settings, you modify the `deliver_rlu_log4j.xml` file.

Modify `deliver_rlu_log4j.xml` as instructed by the comments in that file. You must not modify this file unless suggested by HCL support.

When the RLU is invoked automatically by a flowchart, it uses the Campaign web application's logging, which is configured in `campaign_log4j.xml` under your Campaign installation directory.

Using log4j with the Response and Contact Tracker

When you run the Response and Contact Tracker (RCT) utility, it uses default logger settings.

To change those settings, you modify the `deliver_rct_log4j.xml` file.

Modify `deliver_rct_log4j.xml` as instructed by the comments in that file.

Landing page

In case, unexpected or no values appears in UCC_RESPONSEATTR table for any of the form field after creating a particular

landing page, perform the following steps for resolution.

1. Open Message Editor and locate the landing page.

2. Right- click **Edit content**, click on **Link** tab, select the form defined under "Submit form (optional) drop down, click **OK**.

3. Save and publish the Landing page.

4. Send the mail again. This ensures that all the form field values are added in UCC_RESPONSEATTR table for Landing page.

> 📝 **Note:**
>
> These steps are not required in Quick builder.

Chapter 12. Management of user access to messaging features

Campaign and Deliver use roles and permissions provided by the Unica Platform to control user access to messaging features in Deliver and Campaign. You must have permissions in Unica Platform and Campaign to make the required changes. You must also be familiar with how to configure roles and permissions in the Platform and how to define security policies for Campaign.

To conduct email marketing campaigns, email marketers access Deliver mailing features in Unica Campaign.

To create personalized communications and hosted landing pages, marketers work with features and content in the Deliver Document Composer.

For general information about how to configure roles, permissions, and policies, see the sections of the *Unica Platform Administrator's Guide* that describe how to manage security in the Unica Platform and Unica Campaign.

Role and policy assignment for mailing access

To log in to the HCL Unica system, email marketers enter a system user name and password. The permissions that are granted to the system user determine how the marketer can access mailing features, personalized communications, and content in Deliver and Campaign.

Permissions are associated with roles that are defined in the Unica Platform. To control access to mailing features in Campaign, you can define roles within one or more security policies. All system users that access mailing features, communications, and content must be assigned an Deliver role within a Campaign security policy. Through the policy, you selectively apply permissions for mailing features in Campaign and communications and content in the Deliver Document Composer.

Users that access mailing features must also be assigned the Deliver user and admin roles. These roles are separate from the Deliver roles available in Campaign security policies.

Roles and permissions in Platform and Campaign

Roles in Platform and Campaign are a configurable collection of permissions. For each role in Platform and Campaign, you can specify permissions that control access to the application.

You can use the default roles or create new roles. The set of available permissions is defined by the system; you cannot create a new permission.

About role assignment

Generally, you should give users roles with permissions that reflect the functions that users perform in your organization when they use HCL Unica. You can assign roles to a group or to an individual user. The advantage of assigning roles by group is that you can assign a combination of roles to the group, and if you later want to change that combination, you can do it in one place rather than having to do it multiple times for multiple users. When you assign roles by group, you add and remove users from your groups to control user access.

How the system evaluates roles

If a user has multiple roles, the system evaluates permissions from all those roles together. The ability to perform a function on a particular object is then granted or denied based on the aggregated permissions from all roles. In the case of Campaign, the ability to perform a function on a particular object is granted or denied based on the security policy of the object.

How security policies work

Security policies are the "rule books" that govern security for folders and objects in Campaign. They are consulted each time a user performs an action in the application.

You can create your own security policies or use the default global security policy included with Campaign.
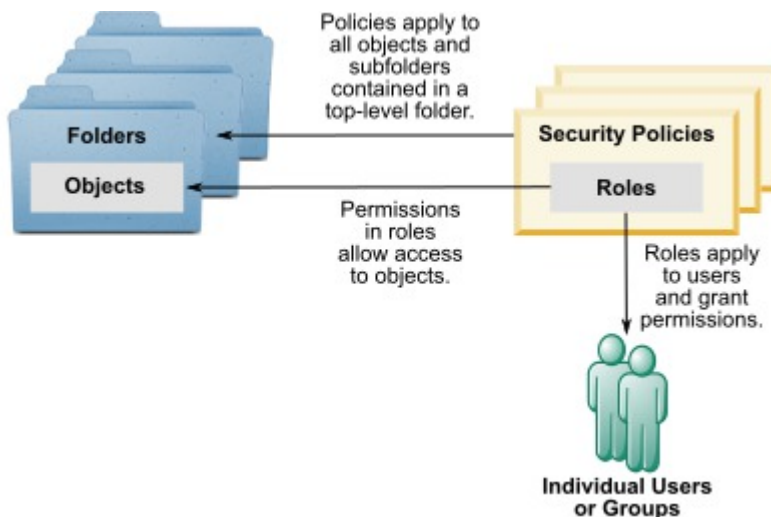
In Campaign, security policies are assigned to folders. When you create a top-level folder, you are required to apply a security policy to the folder. Any objects or subfolders within that folder inherit the folder's security policy.

Because the top-level folder determines the security policy of the objects in the folder, you cannot directly assign a security policy to objects. To change the security policy of an object, you must move the object into a folder with the desired security policy or into the top-level root folder.

You also cannot directly assign a security policy to a user. Unlike objects and folders, which are assigned to security policies as a whole, users are assigned to roles within security policies. To control what users can do, you assign users to roles within security policies. In this way, you control user access to objects within folders that use those security policies.

If a user is not explicitly assigned to at least one role in a security policy, that user cannot create folders and objects under a top-level folder that uses that policy, and that user has no access to objects under that folder or its sub folders.

The following diagram illustrates the relationship between security policies, folders, objects, roles, and users.

Top-level Administrative roles

Administrative roles in Unica Campaign are assigned for each partition. Users with these roles can perform the allowed actions on any objects within the partition, regardless of the security policy used in the folders that contain the objects.

Security policies and partitions

Security policies are created per partition. There is no sharing of security policies across partitions.

Each partition in Unica Campaign can have multiple security policies.

Security policy changes when folders and objects are moved or copied.

Objects and folders can be moved or copied across security policies, but the user performing the move or copy must have permissions to do so, in both the source and destination policies.

After an object or folder is moved or copied to a folder that is assigned to a different security policy from its source, the security policy of the lower-level objects or subfolders is automatically changed to the security policy of the new folder.

The global security policy

Campaign includes a default global security policy. You cannot delete this policy; it always applies. However, you can customize your security scheme as follows.

- Modify the roles and permissions in the global policy to suit the needs of your organization.
- Create custom policies and assign users only to your custom policies rather than the global policy.
- Use both custom policies and the global policy.

Any custom policy you create exists under the global policy. If you choose not to create your own security policies, the global security policy is applied by default to the folders and objects that users create in Campaign.

The global security policy contains six pre-defined roles You cannot delete the pre-defined roles, but you can modify their permissions.

The pre-defined roles in the global security policy are:

- **Folder Owner** - All permissions enabled for the folders a user has created. All users have this role; you do not need to assign users to it.
- **Owner** - All permissions enabled for the objects a user has created. All users have this role; you do not need to assign users to it.
- **Admin** - All permissions enabled. The default user `asm_admin` has this role.
- **Execute** - All permissions enabled.
- **Design** - Read and write permissions on all objects. This role cannot schedule flowcharts or sessions.
- **Review** - Read-only permissions.

Deliver roles in the Global Policy

In addition to the pre-defined Campaign roles, the Global Policy includes several roles that are specific to Deliver.

The Global Policy includes the following Deliver roles.

- **Deliver_admin** - Able to access to all mailing features, all content, and all documents.
- **Deliver_execute** - Able to access to all mailing features, all content, and all documents.

- **Deliver_design** - Able to access to all content, all documents, and most mailing features. However, it is explicitly not granted permission to send production mailings.
- **Deliver_review** - Able only to view content and documents, and has limited permissions for working with mailings. It is explicitly not granted permission to add, edit, or delete mailings. It is allowed to view and send test and production mailings.

Note: Deliver does not support the Owner and Folder Owner roles that are created by default for Campaign.

Messaging permissions in Campaign

Campaign controls user access to mailing features by enabling or disabling specific permissions that are defined in roles that are assigned to a user or group. These roles are associated with one or more security policies. You can define multiple Campaign security policies and assign multiple roles to each policy. Each combination of policy and roles can define a specific set of permissions.

For more information about how to manage security permissions, including sample security scenarios, see the *Unica Campaign Administrator's Guide.*

Under Roles and Permissions for the Unica Platform, you assign user permissions for mailing features and content in the Campaign section, as follows.

1. Define user roles.

   System defined user roles for Deliver are created by default under the Global Policy.

   You can also define custom roles and add them to the Global Policy or to other policies that you define.
2. Define security policies and add user roles to the policies.

   The Global Policy is defined by default. You can define additional policies for Campaign.
3. Define specific permissions for each role in each policy.

   You can define additional policies and custom roles with various sets of permissions for more control over access to mailing features in Campaign and the Deliver Document Composer.

Changes to permissions, roles, and policies are applied when the user logs in to HCL Unica. After you assign or change mailing permissions for a user, the user must log out and then log back in for the changes to be observed.

Making roles and permissions available

Depending on your Unica Platform installation, the administrative controls that are required to define and apply roles and permissions might not be immediately visible. You can make the necessary controls visible by accessing the Deliver Document Composer or a mailing in Campaign.

About this task

Perform the following procedure if you do not see all of the following permissions under the Campaign Global Policy.

- Permissions for mailings in the Campaigns category
- Permissions for the Content Library in the Digital Assets category
- Permissions for Deliver documents in the Documents category

1. Log in to HCL Unica.

   If you have multiple users that are configured, log in as a user with the most limited permissions. For example, log in as a user with only View permissions.

2. Navigate to **Campaign** > **Deliver Documents** to access the Document Composer.

   Wait for the Document Composer to finish loading.

3. Navigate to **Settings** > **User Roles and Permissions** > **Campaign** > **partition [n]** > **Global Policy**

   When prompted, confirm that you want to leave the Document Composer by leaving the page.

4. Click **Add Roles and Assign Permissions**. The following Deliver roles are visible.
   - deliver_admin
   - deliver_execute
   - deliver_design
   - deliver_review

5. Click **Save and Edit Permissions**.

   Mailing permissions are visible in the Campaigns, Digital Assets, and Documents categories.

   For more information about the specific permissions that are available, see the following topics.

How Campaign evaluates permissions

When a user performs a task or tries to access an object, Campaign performs the following steps.

1. Identifies all groups and roles to which this user belongs within the global security policy.

   Users can belong to one, many, or no roles. Users belong to the Owner role if they own an object; they belong to the Folder Owner role if they own the folder in which an object resides.

   Users belong to other roles only if they have been specifically assigned to that role (either directly or because they belong in a group assigned to that role).

2. Identifies whether the object being accessed is assigned to a custom-defined policy. If so, the system identifies all groups and roles to which the user belongs within this custom policy.

3. Aggregates the permissions for all roles to which the user belongs, based on results from steps 1 and 2. Using this composite role, the system evaluates the permissions for the action are evaluated as follows:

   a. If any roles have **Denied** permission for this action, then the permissions are aggregated as follows:

      i. Consider a Global Policy, 1 Custom Policy, and a permission DENIED for the Custom Policy role. Then, DENIAL of any permission for a Custom policy Role takes precedence over permissions assigned to the Global Policy Role.

      ii. Consider a Global Policy, 2 or more Custom Policies, a permission DENIED for one of the Custom policy roles, and the same permission GRANTED to the other Custom policy role. Then, GRANT of any permission of Custom policy takes precedence over DENIAL of permission of the Custom policy.

b. If no roles have **Denied** permission for this action, then it checks to determine whether any roles have **Granted** permission for this action. If so, the user is allowed to perform the action.

c. If neither a nor b is true, the user is denied the permission.

Example for one custom policy

Consider one custom policy under Global Policy : CustomPolicyA. CustomPolicyA has CustomPolicyARole, that has Add/Edit Campaign permission DENIED.

Consider UserA who has CustomPolicyARole assigned. DENIAL of Add/Edit Campaign permission for a CustomPolicyARole takes precedence over permissions assigned to the Global Policy Role. Hence, the Add/Edit Campaign objects are not visible to UserA.

Example for two custom policies

Consider two custom policies under Global Policy: CustomPolicyA and CustomPolicyB. Both CustomPolicyA and CustomPolicyB have CustomPolicyARole and CustomPolicyBRole respectively. CustomPolicyARole has Add/Edit Campaign permission GRANTED. CustomPolicyBRole has Add/Edit Campaign permission DENIED.

UserA has both CustomPolicyARole and CustomPolicyBRole assigned. GRANT of Add/Edit permission of CustomPolicyARole takes precedence over DENIAL of permission of the CustomPolicyBRole. Hence, the Add/Edit Campaign objects are visible to UserA.

Definitions of permission states

For each role, you can specify which permissions are granted, not granted, or denied. You set these permissions on the **Settings > User roles and permissions** page.

These states have the following meanings.

- **Granted** - indicated with a check mark ⊘. Explicitly grants permission to perform this particular function as long as none of the user's other roles explicitly denies permission.

- **Denied** - indicated with an "X" ⊗. Explicitly denies permission to perform this particular function, regardless of any other of the user's roles which might grant permission.

- **Not granted** - indicated with a circle ◯. Does not explicitly grant nor deny permission to perform a particular function. If this permission is not explicitly granted by any of a user's roles, the user is not allowed to perform this function.

Permissions for mailings in Campaign

In Campaign, you create, configure, run, and monitor Deliver mailings with controls on Deliver mailing tabs. You manage each mailing on a separate tab.

The following permissions control user access to the Deliver mailing tabs. They are in the **Campaigns** category.

Permission	Description
View Mailings	Allows a user to view an Deliver mailing tab in a campaign. The user cannot edit or change the mailing.
Edit Mailings	Allows a user to configure or change an Deliver mailing tab in a campaign.
Delete Mailings	Allows a user to remove an Deliver mailing from a campaign.
Add Mailings	Allows a user to create a mailing in a campaign.
Send production mailing	Allows a user to initiate a production run of the mailing, enable a mailing for transactional email, or schedule a production mailing run. Production mailings can include many messages. Email messages are sent to every individual identified as a production recipient on the recipient list that is associated with the mailing.
Execute test run	Allows a user to initiate a test run of the mailing. Test mailings usually involve a few messages. During a test run, an email message is sent to every address identified as a test recipient on the recipient list that is associated with the mailing.

Permissions for the Digital Assets category

The Digital Assets permissions control user access to content elements in the Deliver Content Library, and to folders and subfolders in which they are stored.

The Content Library is a repository for content elements (also called digital assets) that are used in communications that users create in the Deliver Document Composer.

Permissions	Description
View Deliver digital assets	Allows a user to open content elements to view properties and preview the content that can be added to a personalized communication.
Create new digital assets in the Deliver content library	Allows a user to create a content element and add it to the Content Library.
Edit existing digital assets in the Deliver content library	Allows a user to open and edit existing content elements.
Delete digital assets from the Deliver content library	Allows a user to remove a content element from the Content Library.
Move digital assets from one folder to another	Allows a user to move content elements within the Content Library. Moving a content element requires assigning this permission to the source and destination folders.

Permissions for the Documents category

The permissions in the **Documents** category control user access to create, edit, and manage personalized communications in the Deliver Document Composer.

Permissions	Description
View Deliver documents	Allows a user to view a document that is used to create an email, Inbox push notification, or hosted landing page.
Create new Deliver documents	Allows a user to create a new personalized communication.
Edit existing Deliver documents	Allows a user to change an existing personalized communication.
Delete Deliver documents	Allows a user to remove a personalized communication.
Publish Deliver document, making content available on the public Internet	Allows a user to publish a personalized communication. Publishing a communication makes the document and all added content available for use in an Deliver mailing.
Copy Deliver documents from one folder to another	Allows a user to copy a personalized communication between folders in the Content Library. Copying a communication requires assigning this permission to the source and destination folders.
Move Deliver documents from one folder to another	Allows a user to move a personalized communication from one folder to another folder in the Content Library. Moving a communication requires assigning this permission to the source and destination folders.

Permissions for the Email Administration category

The permissions in the Email Administration category of the Campaign Global Policy provide Deliver administrators with access to settings that control user access to various messaging domains and messaging features.

Administrators assign domain and feature access in the Policy Settings section of the Deliver Settings window. For example, the administrator can restrict the list of email domains that a user can select as a **From:** domain in an email communication that is created in the Communication Editor. The Policy Settings section does not display unless appropriate permission is explicitly granted to the administrator in the Campaign Global Policy.

Administrators can also control access to administrative interfaces for registering mobile apps with Deliver and for configuring locations for use with location triggered delivery. Links to the administrative pages appear in the Mobile Notification Settings section of the Deliver Settings page. Mobile messaging must be enabled for the hosted messaging account to display the Mobile Notification Settings section of the Deliver Settings page.

Permissions	Description
Configure Domains	Controls access to the Policy Settings section of the Deliver Settings page. If the administrator's role is not granted permission to configure email domains, the administrator cannot see the Policy Settings section. This permission is also required to administer short link domains.

Messaging permissions for Deliver

Unica Deliver controls access to mailing features outside of the mailing tab in Campaign through the following pre-defined security roles.

- `Deliver_admin`
- `Deliver_user`

Users must have both roles to have access to Deliver mailing features.

Assigning Deliver roles

To provide a user with full access to Deliver mailing features, assign the pre-defined Deliver roles to the user.

1. In Unica Platform, navigate to `Settings > User roles & Permissions > Deliver > partition [n] > Deliver_admin`.
2. Click **Assign Users**.
3. Select the user from the list of available users. Click **Add** to assign the role to the user.
4. Repeat steps 1-3 for the `Deliver_user` role.
5. Save changes.

Controlling email domains and short link domains

Upon request, Unica configures one or more email domains for your hosted email account. Unica can also assign domains that marketers use to create shortened links in various types of messages. System administrators with appropriate permissions control the messaging domains that are available to marketers.

About this task

Depending on your business requirements, it might be desirable to restrict the list of messaging domains that are available to specific marketers. Deliver administrators restrict the list of available domains through security policies that are applied to folders in the Document Composer. The ability of marketers to create and edit email communications depends on the security policy that is applied to the folder that contains the communication.

Deliver administrators with appropriate permissions can control the list of email domains that Deliver users can use as the **From:** domain in email communications. Administrators can also control the list of short link domains that is presented to marketers when they configure communications that use shortened links. For example, you can specify which short link domains are available when marketers add a social sharing link to marketing messages.

Deliver administrators use the **Policy Settings** page to grant permissions to use specific messaging domains. Access to the **Policy Settings** page is controlled by the Email Administration permissions that are granted through the Campaign

Global Policy. Only administrators with the appropriate permissions can restrict access to email domains through the **Policy Settings** page.

1. In the **Settings** menu, select **Messaging Settings**.
   Result
   If you have the appropriate administrative permissions, the Policy Settings section displays on the Deliver Settings page.
2. Click **Display a list of policies and their settings**.
   Result
   A list of security policies that are configured for your Deliver installation displays.
3. Click a security policy that is associated with the system user whose messaging domain access you want to control.

   The Domains section displays the email domains that are configured for your hosted messaging account.

   The Short Link Domains section displays the short link domains that are configured for your hosted messaging account.

   - In either section, click **Use all domains** to allow users that are associated with the policy to use any of the email domains that Unica configured for your hosted email account.

     This option is the default.
   - Click **Use specific domains** to select specific domains.

     > **Note:** If you select **Use specific domains**, you must update the domain permissions when you register a new email or short link domain for your hosted messaging account. The system does not assign permissions for the new domain automatically.

Results

For the users associated with the security policy, only the selected email domains appear as an option for the **From:** address in email communications. For communications that require shortened links, marketers can choose only from the specific short link domains that you select.

After you save the new settings, the Document Composer updates the domain options that are available to marketers.

For more information about how Deliver marketers create and manage communications, see the *Unica Deliver User's Guide.*

Maintenance of hosted email domains

To send email messages you must register at least one email domain with Unica. To improve message deliverability, Unica works with you to establish and maintain the email reputation of the domain with leading Internet Service Providers (ISPs) around the world. You can establish multiple email domains with Unica.

When you configure the header in an email communication, the system populates the From address with the email domain that you have registered with Unica. If you establish multiple email domains with Unica, the available domains display in a drop-down list. System administrators can control the email domains that email marketers can select or modify.

You can request that Unica add or delete email domains established for your hosted messaging account. After Unica completes the change, the system updates the list of available email domains. The change is reflected in the list of available email domains the next time you create or edit an email communication.

📝 **Note:** Email domain changes for your account do not update email communications that you created before the change request. To change the email domain for a communication created previously, you must reopen the email communication and update the email domain selection.

For more information about how to register an email domain with Unica, see *HCL Unica Domain Name Options for Email.*

To request changes related to your email domains, contact Unica Deliver Services team via HCL technical support.

Configuring default sender address and display names

For each email domain that you have registered with Unica, you can define a default email address and a default friendly name. The combination of email address or friendly name and the email domain appears as the From: address for the email messages that you send.

About this task

Administrators can configure the default sender and display names on the Domain Settings page. The domain settings are part of the Deliver Settings interface. Access to the Domain Settings page is controlled by the Email Administration permissions that are granted through the Campaign Global Policy. Only administrators with the appropriate permissions can restrict access to email domains through the Policy Settings page.

1. Go to **Settings** > **Deliver Settings**. In the Domain Settings section, click **Display** the list of domain settings.

   The Domain Settings page lists the default display names and email addresses associated with email domains that are registered to your hosted email account. The list includes only the domains that your user permissions allow you to modify.

   The Default column indicates the combination of display name, address, and domain that appears as the default From address for new email communications.

2. Click **Edit** 📝 . The Edit Domain Settings window opens.

   The Domain Name column lists the available email domains. You can do the following for any of the domains.

       • In the From Display Name column, enter a friendly name to appear as the default for an email domain in the list.
       • In the From Address column, enter the local part of the email address to appear as the default for an email domain in the list.
3. Optionally, in the Default column, select one combination of display name address, and domain to appear as the default From address for new email communications.

   If you do not select a default, the system uses the first domain on the list to create the default From address for new email communications.
4. Save changes.

Results

The new address settings apply to all new email communications that you create. The settings do not change address information for email communications that you created previously. To update previous email communications, you must reopen and modify each communication.

Controlling access to the list of sent messages

Deliver provides a list of messages that have been sent from your Deliver environment. Because the list includes links to messaging configurations, your security plans might require that you restrict access to the list.

About this task

The list of messages is presented on the **Message Overview** page. By default, all users in your Campaign and Deliver environment can see the list of sent messages. However, when you enable the access restriction, you can prevent specific users from seeing the menu option to open the page that contains the list.

Restricting access to the list of sent messages affects all partitions in your Campaign installation. If your Campaign installation includes multiple partitions, you must update user permissions separately in each partition to explicitly grant or deny permission to access the list.

Controlling who can access the list of sent messages requires a series of tasks to change user permissions and the system configuration.

Task	More information
Identify users who can access the list of messages. At first, all users are granted access.	Granting access to the list of sent messages on page 101
Identify users who are not allowed to access the list of messages.	Denying access to the list of sent messages on page 102
Enable the access restriction.	Enabling the restriction to the sent message list on page 103

Results

When you complete these tasks, the **Message Overview** option on the **Campaign** menu is visible only to users with roles that explicitly grant permission to access the list of mailings.

Granting access to the list of sent messages

If you restrict access to the list of sent messages, you must specifically grant access to users who must access the list.

About this task

Users access the list of sent messages by clicking the **Message Overview** link on the **Campaign** menu. You can grant a user access to the list of all sent messages by assigning the user a top-level administrative role that is explicitly granted permission to see the **Message Overview** link.

The default top-level roles include **Admin**, **Execute**, **Design**, and **Review**. Permissions that you grant through the top-level roles apply to all objects in the partition.

1. Go to `Settings > User Roles and Permissions > Campaign > partition (n).`
2. Click **Save and Edit Permissions**.
   A list of permissions for the partition opens. The available top-level roles are listed across the top to the page.
3. In the **Administration** section, explicitly grant the **View Mailing List Page** permission to every role.

Results

When you enable access restrictions for the list of sent messages, users with roles that are explicitly granted the **View Mailing List Page** permission can see the **Message Overview** link on the **Campaign** menu.

What to do next

Create a role to deny access to the list of sent messages.

Denying access to the list of sent messages

If you restrict access to the list of sent messages, you must specifically deny access to users who must not be allowed to access the list.

About this task

Users access the list of sent messages by clicking the **Message Overview** link on the **Campaign** menu. You can prevent a user from accessing the list of all sent messages by assigning the user a top-level administrative role that is explicitly denied permission to see the **Message Overview** link.

The default top-level roles include **Admin**, **Execute**, **Design**, and **Review**. Permissions that you grant through the top-level roles apply to all objects in the partition. You can create new top-level roles to supplement the default top-level roles. The new roles can grant or deny specific permissions.

1. Go to **Settings** > **User Roles and Permissions** > **Campaign** > **partition (n)**. The **partition <n>** page opens.
2. Click **Add Role**. Assign a name to the role and enter a brief description. Save the changes and return to the **partition <n>** page.
3. Configure the new role to deny access to the list of sent mailings.

   a. Click **Add Roles and Assign Permissions**. The **Properties for Administrative Roles** page opens. The new role displays in the list of roles.

   b. Click **Save and Edit Permissions**.
   A list of permissions for the partition displays as a matrix of selection icons that indicate the state of each permission for each role. The new role displays next to the other top-level roles across the top to the matrix.

   c. In the **Administration** section, explicitly deny the **View Mailing List Page** permission for the new role. Save changes.
4. Assign the new role to the users that you want to prevent from accessing the mailing list page.

a. Go to **Settings** > **Users**. Select the user that you want to prevent from accessing the list of sent messages.

b. Click **Edit Roles**. The new role that you created in the previous step (a role that is configured to deny access) appears in the list of **Available Roles**.

c. Move the new role from **Available Roles** to **Roles**. Save changes.

Results

When you enable access restrictions for the list of sent messages, a user that is assigned the new role cannot see the **Message Overview** link.

What to do next

Update the configuration to enable access restrictions for the list of sent messages.

Enabling the restriction to the sent message list

Users access the list of sent messages through the **Message Overview** option on the **Campaign** menu. If you restrict access to the list of sent messages, the Security Function ID property controls the display of this menu option and, therefore, controls access to the list of sent messages.

About this task

To restrict access to the list of sent messages, you must update the Security Function ID property in the Platform configuration. This property applies to all partitions in your Campaign installation.

When you populate the Security Function ID with the correct value, the **Message Overview** option is available to only those users with a role that explicitly grants the View Mailing List Page permission. Users with roles where the View Mailing List Page permission is either denied or not granted, cannot see the **Message Overview** option.

1. Go to **Settings** > **Configuration** > **Platform** > **Platform-wide navigation** > **Main navigation menu** > **Campaign** > **Deliver Mailings**. Click **Deliver Mailings** to display the configuration settings.
2. Click **Edit Settings**.
3. In the **Security Function ID** field, enter `7000`. Save changes.

   To see the results of the configuration change, log out of the system and log in again.

Results

Only users with roles that explicitly grant the View Mailing List Page permission can see the **Message Overview** link to access the list of sent messages.

Permissions for Deliver reports

Your user permissions determine your ability to view Deliver reports.

For information about setting permissions to access standard Deliver reports, see the section in the *Unica Insights Reports Installation and Configuration Guide* for reporting and security.

Chapter 13. Technote (troubleshooting)

Problem (Abstract)

To use the Deliver components installed with Unica Campaign and send personalized marketing messages, you must connect the local Campaign installation to remote message resources hosted by HCL. This section describes how to configure such a connection when your corporate firewall rules prohibit direct communication with the hosted environment.

Resolving the problem

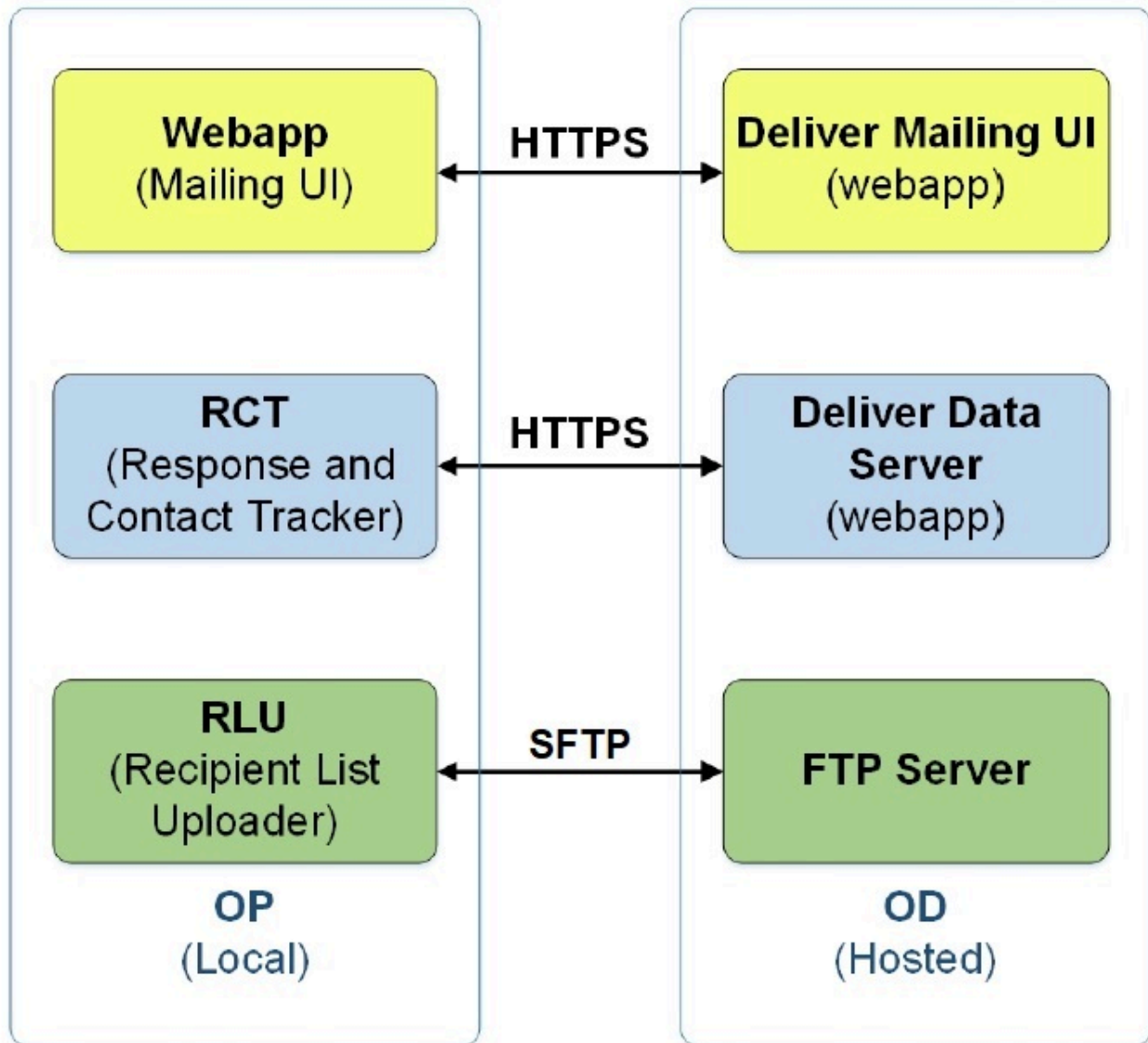**Typical communication with the hosted email resources environment**

The following diagram illustrates the standard configuration for communication between the On Premises (OP) environment and On Demand (OD) environment.

The local Deliver OP environment requires external communication with Deliver OD environment using HTTPS and SFTP.

The OP environment includes a web application server (either IBM WebSphere or Oracle WebLogic) on which you have deployed Campaign. Campaign hosts the Deliver components (RCT and RLU) that communicate with the hosted email resources in the OD environment.

The Response and Contact Tracker (RCT) downloads response data from the OD environment.

The Response List Uploader (RLU) uploads mailing lists and other required mailing data to the OD environment.

When the machine where Unica Deliver is installed cannot communicate directly with OD environment, Deliver supports communication with the hosted OD resources through a SOCKS proxy.

Connecting to message services through a proxy

The following diagram illustrates communication between the OP and OD environments when using a SOCKS proxy. Note that SOCKS proxy is configured on the local "On Premises" environment.
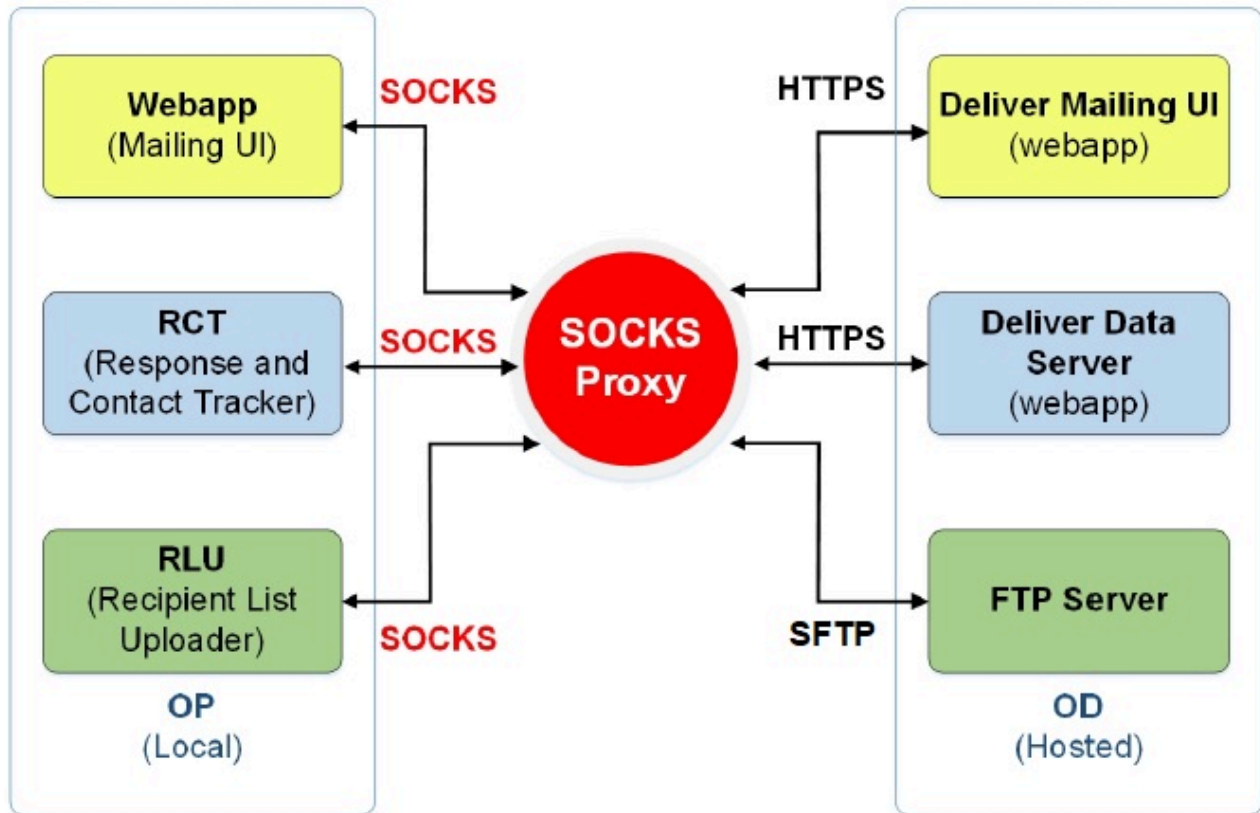
Verify the following points before you enable the proxy options.

- The proxy server is a SOCKS proxy.
- The proxy server can access the Deliver OD environment and allows the traffic to and from the ports configured in the HCL Data center used by your hosted email account.
- You have installed the SOCKS proxy in such a way that the Deliver OP environment can access the proxy.

Changes required for routing SFTP and HTTPS traffic through a SOCKS proxy

To use a SOCKS proxy to access email resources hosted by HCL, you must make changes to the web application where you have deployed Campaign and to the startup scripts for the Deliver RCT and RLU.

In case of SOCKS proxy for RLU, ensure that you enable the SOCKS5 IP feature on.

Making changes for SFTP

For SFTP traffic, apply the following configurations to the RLU and the web application server.

- `– Dhcl.unica.deliver.ftp.proxy.host = <socksHost>`
- `– Dhcl.unica.deliver.ftp.proxy.port = <socksPort>`
- `– Dhcl.unica.deliver.ftps.proxy.match.hosts = <comma separated list of host names and IP addresses>`

socksHost is the host name or IP of the SOCKS proxy.

socksPort is the port on which SOCKS proxy is running.

`-Dhcl.unica.deliver.ftps.proxy.match.hosts` matches host names and IPs used when routing traffic through the SOCKS proxy.

The IP address specified for `-Dhcl.unica.deliver.ftps.proxy.match.hosts` is the IP address that FTP seiver in the hosted OD environment sends to the FTP client in the local OP environment as part of the SFTP protocol during data transfer.

Set `-Dhcl.unica.deliver.ftps.proxy.match.hosts` to one of the following values (depends on the data center used by your hosted email account).

US data center: `-Dhcl.unica.deliver.ftps.proxy.match.hosts=ftp-em.unicadeliver.com`

India data center: `-Dhcl.unica.deliver.ftps.proxy.match.hosts=ftp-in.unicadeliver.com`

Europe data center: `-Dhcl.unica.deliver.ftps.proxy.match.hosts=ftp-eu.unicadeliver.com`

Making changes for HTTPS

For HTTPS traffic, apply the following configurations to the RCT and the web application server.

`-Dhcl.unica.deliver.https.proxy.host= <socksHost>`

`-Dhcl.unica.deliver.https.proxy.port= <socksPort>`

`-Dhcl.unica.deliver.https.proxy.type=SOCKS`

socksHost is the host name or IP of the SOCKS proxy.

socksPort is the port on which SOCKS proxy is running.

Authentication requirements when using a SOCKS proxy

If your SOCKS proxy requires authentication, configure the following for the web application servers, RLU and RCT.

- `-Dhcl.unica.deliver.proxy.auth.user = <username>`
- `-Dhcl.unica.deliver.proxy.auth.password = <password>`

Where username and password are the credentials required to authenticate against the proxy.

To configure the RCT using a SOCKS proxy

Configure the RCT to work through a SOCKS proxy, follow the procedure for your operating system.

For the RCT in Windows environment

Add the following proxy arguments to `common.bat`, located in the `//deliver/bin` directory of your local Deliver installation.

`set RCT PROXY ARGS=`

`-Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>`

`-Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>`

`-Dhcl.unica.deliver.https.proxy.type=SOCKS`

`-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUIH USER>`

```
-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUIH_PASSWORD>
```

```
set RCT_JAVA_ARGS=%BASE<em>_VM_ARGS% %RCT_MEM_ARGS%
```

```
%RCT_EXTRA_VM_ARGS% %RCT_PROXY_ARGS
```

For the RCT in UNIX environments

Add the following proxy arguments to `common.sh` located in the `\\deliver\bin` directory of your local Deliver installation.

> **Note:** Do not make changes directly to `rlu.sh.rct.sh` or `setenv.sh` because they will be overridden.

```
RCT PROXY ARGS="
```

```
-Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>
```

```
-Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>
```

```
-Dhcl.unica.deliver.https.proxy.type=SOCKS
```

```
-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUIH USER>
```

```
-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUIH_PASSWORD>
```

```
RCT_JAVA_ARGS="${BASE_VM_ARGS} ${RCT_MEM_ARGS} ${RCT_EXTRA_VM_ARGS}
```

```
${RCT_PROXY_ARGS}"
```

To configure RLU using a SOCKS proxy

To configure the RLU to work through a SOCKS proxy, follow the procedure for your operating system.

For the RLU in Windows environment

Add the following proxy arguments to common.bat located in the `//deliver/bin` directory of your local Deliver installation.

```
set RLU PROXY ARGS=
```

```
-Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>
```

```
-Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>
```

```
-Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>
```

```
-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUIH USER>
```

```
-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUIH_PASSWORD>
```

```
set RLU_JAVA_ARGS=%BASE_VM_ARGS% %RLU_MEM_ARGS% %RLU_EXTRA_VM_ARGS%
```

```
%RLU_PROXY_ARGS%
```

For the RLU in UNIX environments

Add the following proxy arguments to common.sh, located in the `\\deliver\bin` directory of your local Deliver installation.

> 📝 **Note:** Do not make changes directly to `rlu.sh.rct.sh` or `setenv.sh` because they will be overridden.

```
RLU PROXY ARGS=

-Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>

-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUIH USER>

-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUIH_PASSWORD>

RLU_JAVA_ARGS="${BASE_VM_ARGS} ${%RLU_MEM_ARGS%} ${%RLU_EXTRA_VM_ARGS%}

${RLU_PROXY_ARGS}"
```

Changes to the WebSphere configuration

Add the following to the WebSphere generic JVM arguments (see the screenshot):

```
-Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.https.proxy.type=SOCKS

-Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>

-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUIH USER>

-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_ PASSWORD>
```
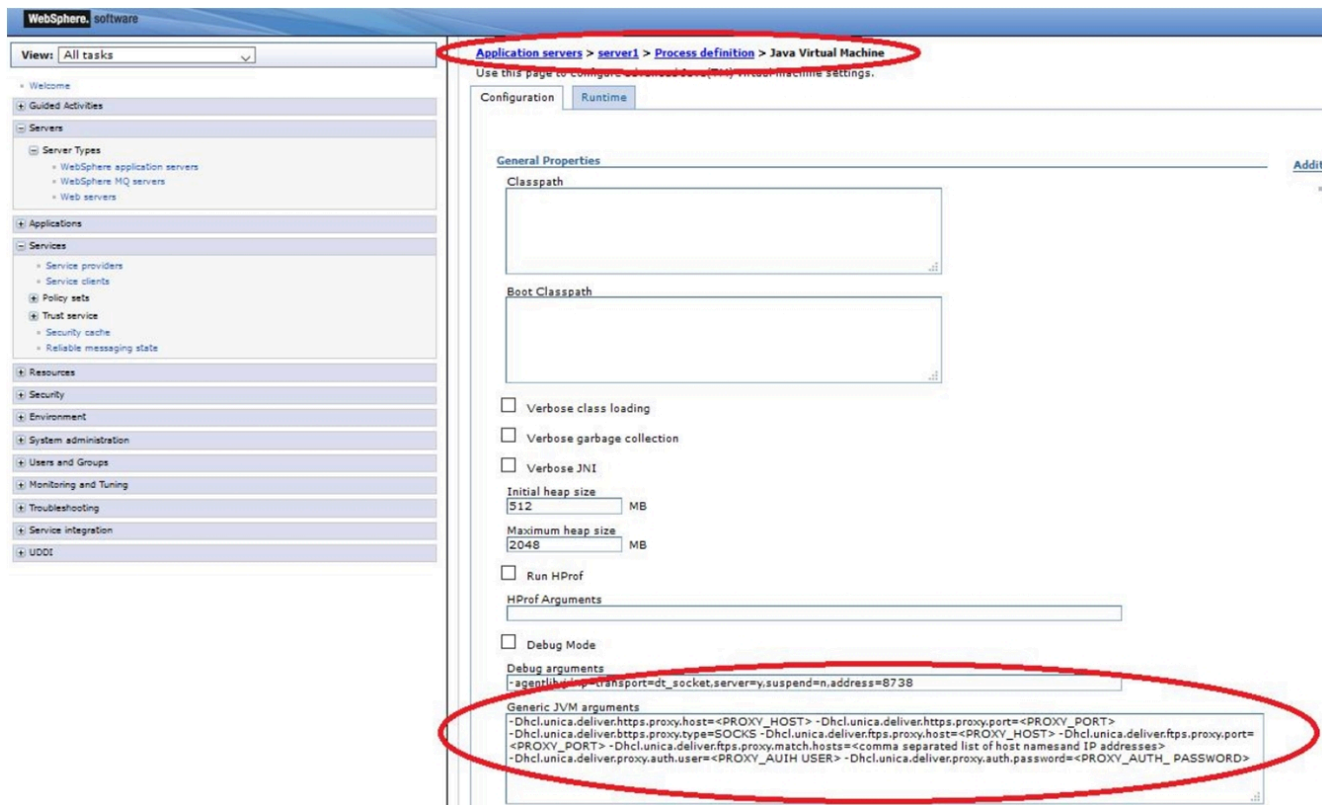
Changes to the Oracle WebLogic configuration

For WebLogic, modify the script.

In Windows environment

```
JAVA_OPTIONS=%(JAVA_OPTIONS)

-Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.https.proxy.type=SOCKS

-Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>

-Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>

-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUIH USER>

-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_ PASSWORD>%
```

In UNIX environments

```
JAVA_OPTIONS='(JAVA_OPTIONS)

-Dhcl.unica.deliver.https.proxy.host=<PROXY_HOST>

-Dhcl.unica.deliver.https.proxy.port=<PROXY_PORT>
```

```
-Dhcl.unica.deliver.https.proxy.type=SOCKS
```

```
-Dhcl.unica.deliver.ftp.proxy.host=<PROXY_HOST>
```

```
-Dhcl.unica.deliver.ftp.proxy.port=<PROXY_PORT>
```

```
-Dhcl.unica.deliver.ftps.proxy.match.hosts=<comma separated list of host names and IP addresses>
```

```
-Dhcl.unica.deliver.proxy.auth.user=<PROXY_AUIH USER>
```

```
-Dhcl.unica.deliver.proxy.auth.password=<PROXY_AUTH_ PASSWORD>'
```

Chapter 14. Deliver channel vendor account configuration

Deliver supports SMS, WhatsApp and Push as delivery channels in addition to Email. Deliver supports SMS, WhatsApp and Push as delivery channels in addition to Email. SMS is supported using different vendors, so customer has a choice to select a SMS partner based on geographical and cost aspects. Whenever a customer decides to use a specific vendor for SMS or Whatsapp messages, HCL works with the customer and required vendor to onboard him smoothly to Deliver. As part of this process, customer's account is created with each vendor. This account enables Deliver to send messages on behalf of that customer and process responses from users.

Deliver supports the following channels.

- SMS using Karix vendor
- SMS using RML vendor (for both referral and reseller type of licenses)
- Whatsapp using RML vendor

The following document explains these steps for each of these channels. These steps must be performed by or for each customer account as part of the onboarding process.

Karix SMS account configuration

Perform the following steps to configure the customer's Karix SMS account for it to work with Deliver.

1. Log in to Karix Console (www.karix.solutions) and click the API Keys button on dashboard to create API key and configure with Deliver.

2. In the top right corner, under the **Open My Account** list, select **Edit My Info** option and note the Sender ID configured for your Karix account to configure in Deliver.
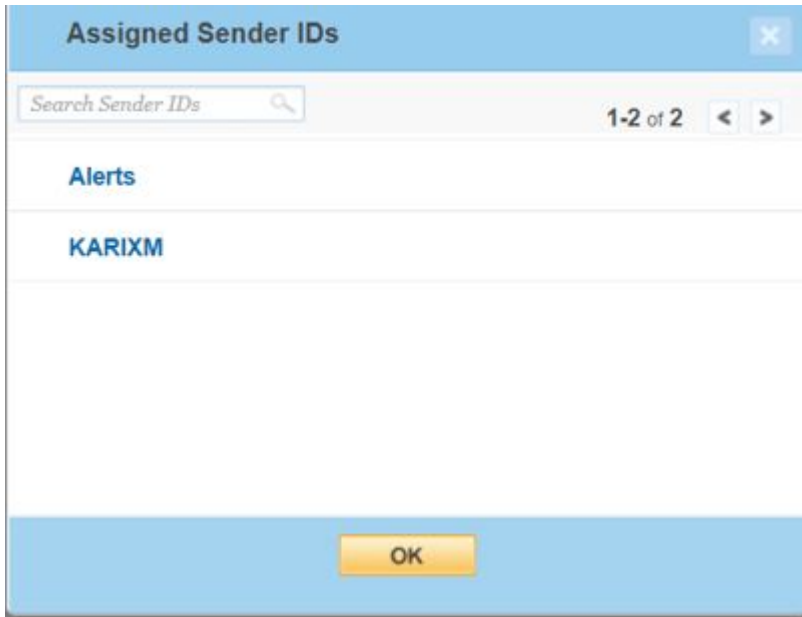
3. Provide the API Key created in Step 1 and the Sender ID noted in Step 2 for it to be configured in the account.

4. Set the callback URL in the Karix console.

- For US datacenter: https://smsin-us.unicadeliver.com/deliversmsib/sms?
  partition=<account>&provider=karix&dummy=1
- For EU datacenter: https://smsin-eu.unicadeliver.com/deliversmsib/sms?
  partition=<account>&provider=karix&dummy=1
- For India: https://smsin-in.unicadeliver.com/deliversmsib/sms?partition=%3Caccount
  %3E&provider=karix&dummy=1

Note: Once the account is configured in Deliver, <account> in above URL must be replaced with the account name provided by Deliver Provisioning team.

RML SMS account configuration

The customer's RML SMS account must have the following configurations for it to work with Deliver.

RML account onboarding

- You must work with RML team to create a SMS account for India or World Wide location based on the customer's location.
- The senderid must to be provided by the customer configured by RML.
- SMS templates also require to be whitelisted based on target SMS sending location. For example:

◦ Indian customers require to provide Principal Entity ID (PE ID) to Deliver services team and configure templates registered on DLT platform as directed in User Guide.

◦ Pre-approved templates required for USA and Canada.

RML has different login URLs based on geography, which they provide as part of account creation email from their side. For example, URLs for India and Worldwide datacenters

- India account: https://ems.rmlconnect.net/
- Worldwide account: https://client.rmlconnect.net/login

Perform the following steps.

1. Log in to RML Console as per the above URLs and navigate to **Utilities> DLR Push URL**.

   Note: You must ask RML to add this menu while provisioning account.

2. Set the callback URL in RML console.
   - For US datacenter: https://smsin-us.unicadeliver.com/deliversmsib/sms?partition=<account>&provider=RML
   - For EU datacenter: https://smsin-eu.unicadeliver.com/deliversmsib/sms?partition=<account>&provider=RML
   - For India datacenter: https://smsin-in.unicadeliver.com/deliversmsib/sms?partition=%3Caccount%3E&provider=RML&dummy=1

   Note: Once the account is configured in Deliver, <account> in above URL must be replaced with the account name provided by Deliver Provisioning team.

Setting up two-way SMS replies with RML

Deliver supports two-way SMS replies (example: STOP requests) with RML since v12.1.1. Two-way SMS requires a senderid supporting replies, which will be provisioned by RML based on request.

RML supports either dedicated or shared short code. For shared short code, user need to put brand name at the start of message (example: HCL STOP) for RML to delegate request back to Deliver correctly. There's no such requirement for dedicated short code (example: user can reply just STOP) and RML will associate reply correctly to required Deliver account.

For RML to provide replies to correct prod environment and customer account, Deliver inbound reply webhook URL needs to be configured in customer's shared/dedicated senderid as below.

- For US datacenter: https://smsin-us.unicadeliver.com/deliversmsib/mo/<account>?provider=RML
- For EU datacenter: https://smsin-eu.unicadeliver.com/deliversmsib/mo/<account>?provider=RML
- For India datacenter: https://smsin-in.unicadeliver.com/deliversmsib/mo/<account>?provider=RML

📝 **Note:** <account> in above URL needs to be replaced with account name provided by Deliver Services team once the account is configured in Deliver.

RML WhatsApp account configuration

About this task

The customers who are provisioned with the WhatsApp feature (provided by RML) must perform the following steps for configuration.

1. Create a verified Facebook Business Manager account setup for WhatsApp. The RML team will guide customers to configure Facebook Business Manager account as required.
2. Create a WhatsApp account with RML. RML will create this account after the Facebook Business Manager account verification is complete.
3. Create Message templates approved by WhatsApp. The RML team will work with customers to prepare message templates in required format and getting it approved from customer.
4. Once RML provides approved templates to the customers, they require to upload them to the Deliver Message Editor using **New > WhatsApp** Content menu item. All the details must be exactly provided as per approved template, since WhatsApp does not allow any changes to template post approval.
5. Configure callback URL in RML WhatsApp account. For this, need to provide below URL to RML so that they can configure it as callback URL for WhatsApp message Deliver reports.
   - For US datacenter:https://smsin-us.unicadeliver.com/deliversmsib/wa/<account>
   - For EU datacenter: https://smsin-eu.unicadeliver.com/deliversmsib/wa/<account>
   - For India datacenter:https://smsin-in.unicadeliver.com/deliversmsib/wa/<account>

   📝 **Note:** <account> in above URL needs to be replaced with account name provided by Deliver Services team once the account is configured in Deliver.

Chapter 15. RCT Reverse Proxy Configurations

Configuring Nginx to use with Deliver clustered environment as reverse proxy and load balancer

To configure Nginx to use with Deliver clustered environment as reverse proxy and load balancer, complete the following steps:

1. Edit the Nginx configuration record with elevated rights.
2. Define an upstream element and list every node in your backend cluster.
3. Map a URI to the upstream cluster with a proxy_pass area setting.
4. Restart or reload the Nginx server to incorporate the config modifications.
5. Verify the configuration of the Nginx load balancer setup.

Defining an upstream element
Example:

```
upstream samplecluster

{
server <backend server1>:2001;
server <backend server2>:2001;
}
```

> ✏️ **Note:** Deliver RCT listens on port 2001 for API requests and receives JSON payload containing response data. You can also mention the weight of the server based on the requirement. **Example server**: `backendserver1.example.com weight=5`. For more details, see http://nginx.org/en/docs/http/ngx_http_upstream_module.html.

Defining an upstream element

The Nginx load balancer can act as a reverse proxy. To do that, Nginx needs to recognize the URLs it need to forward requests to the workload controlled cluster. You must configure a region detail with an Nginx proxy_pass entry in the default configurations. Check out the following example:

```
location  /deliver/responses/inetresp

{
client_max_body_size 100M;
proxy_set_header content-type "application/json";
proxy_method POST;
proxy_pass http://samplecluster/deliver/responses/inetresp ;
}
```

> ✏️ **Note:** `–delivercluster` is the name that needs to be used while calling the Deliver API. **Example**: `http:// delivercluster/deliver/responses/inetresp` along with JSON payload. Caller machine must contain an entry of IP of the machine where Nginx is running against the name of upstream element. In above example the host file must contain the following entry:

```
<Nginx server IP> delivercluster
```

Restarting/Reloading the Nginx server

Run the following commands:

```
systemctl stop nginx
systemctl start nginx
nginx -s reload
```

Verifying the configuration

Run the following command:

```
systemctl status nginx
```

Index

C

configTool 87
configTool utility 87

U

utilities
    configTool 87