Client Hub

SAP Mobile Platform 3.0

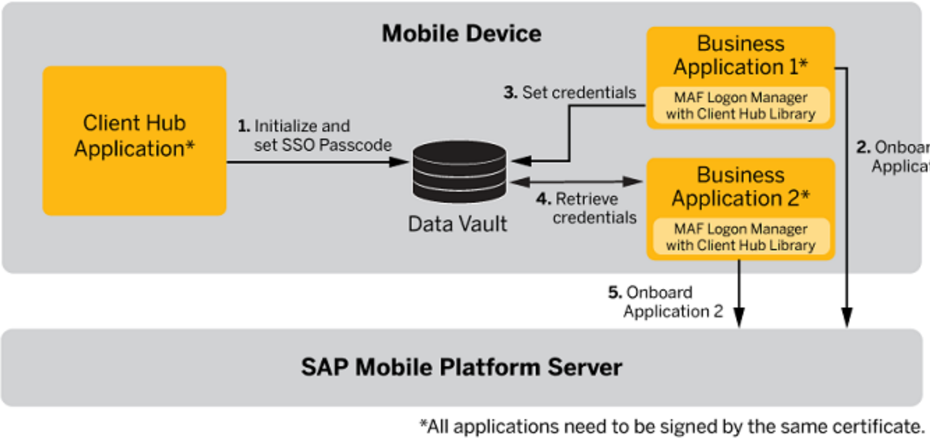
DOCUMENT ID: DC-01-0300-01
LAST REVISED: November 2013

Contents

Client Hub

Client Hub is a native application a user installs on the device, that allows multiple business applications to share common credentials enabling an SSO-style behavior. The Client Hub, integrated with Logon Manager, simplifies user onboarding and configuration to enable easier and faster enterprise-wide deployments. Client Hub is an additional layer of management on top of the shared keychain, and allows end-users to control which applications are using their credentials. The Client Hub reduces the effort required by the end user to manage multiple passwords for mobile applications and improves the user experience. Client Hub supports both OData and Kapsel applications.

Figure 1: Client Hub Overview


Mobile Device
Client Hub Application*
1. Initialize and set SSO Passcode
Data Vault
3. Set credentials
4. Retrieve credentials
Business Application 1*
MAF Logon Manager with Client Hub Library
Business Application 2*
MAF Logon Manager with Client Hub Library
2. Onboard Application 1
5. Onboard Application 2
SAP Mobile Platform Server
*All applications need to be signed by the same certificate.

Open the Client Hub application to set the SSO passcode. The SSO passcode is a unique password used for Client Hub applications. The same passcode should be entered in the business application when credentials are requested from Client Hub applications. Once the SSO Passcode is created in the Client Hub application, a shared DataVault is created. The Business Application 1 (OData or Kapsel) chooses to work with Client Hub. For the first time, user enters the credentials, and on successful onboarding, these credentials are stored in the shared DataVault using the SSO passcode. If a Business Application 2 (OData or Kapsel) also chooses to work with Client Hub, it can retrieve the same stored credentials using the same SSO passcode, and onboard successfully.

Note: The SDK installer includes the source code for Client Hub. SAP® does not support customer modifications to the source code after new versions of the template are released. Intellectual property for the template code belongs to SAP. The main purpose for including the source code is to enable code-signing and branding by customers or partners.

The Client Hub simplifies application onboarding by:

- Onboarding multiple applications to SAP® Mobile Platform. Once the end-user has onboarded the first application, all subsequent applications that are enabled for Client Hub onboard automatically.
- Managing the single sign-on (SSO) vault on the device.
- Enabling cosigned business applications with the same security configuration to securely share credentials on the device.
- Supporting multiple security configurations per device.
- Supporting backend password change events. If the backend password changes, once the end-user updates in a single application, all other applications automatically pick up the update

Note: For information about supported Apple iOS and Google Android device platforms, see *SAP Note 1901995*

Prerequisites

Complete the prerequisites before using Client Hub in your applications. The Client Hub feature set is only available for cosigned enterprise store applications.

- Install Client Hub (along with the API libraries) on your Android or iOS device. Before installation, compile or build the Client Hub application, cosigned with the same developer certificate as the application.
- Install the business applications that implement Client Hub on the device.
  - Use the same certificate to cosign applications and Client Hub.
  - Compile the applications using SAP Mobile Platform SDK version 3.0 or later.
  - Use the Logon Manager for application initialization.
- Mobile apps should contain a configuration file (plist or properties) from developer. The `clienthub.plist or clienthub.properties` contains connection settings for the application, and its security configuration.
- The applications must share the same security configuration setting:
  - Security configuration is a setting defined by the administrator on the SAP Mobile Platform Server, which maps mobile apps to an Identity Provider.
  - You can have multiple security configurations in a single Client Hub application: the apps access the credentials for their own security configuration.

Managing iOS Application Registration Using Client Hub

Use Client Hub, integrated with Logon Manager to register applications for iOS devices. SAP provides client-side credentials and a connection settings sharing mechanism for applications that are based on MAF Logon. Client Hub supports both OData and Kapsel applications.

Note:

- The SDK installer includes the source code for Client Hub. SAP does not support customer modifications to the source code after new versions of the template are released. Intellectual property for the template code belongs to SAP. The main purpose for including the source code is to enable code-signing and branding by customers or partners.
- This topic covers the Client Hub application installation and deployment for Eclipse environment only. You can use any other third-party party as required.

---

1. Getting Started with Client Hub Application Installation and Initialization

The following tasks describe the steps to install and initialize the Client Hub application.

Installing the Client Hub Application

Install SAP Mobile Platform Native SDK - Client Hub component. By default, SAP Mobile Platform SDK components are installed in the `..\SAP\MobileSDKXXX` directory. In this guide, `SDK_HOME` represents the SAP Mobile Platform SDK installation directory, down to the `MobileSDKXXX` folder. Client Hub gets installed under the `ClientHub` directory, where the project files for Client Hub applications, used for registering applications on iOS devices is available. Ensure that you uncompress the `ClientHub.zip` file before importing the Client Hub project into Xcode.

Setting Up the Development Environment

The Client Hub application is shipped as a source code project. Set up the iOS Development Environment before registering your application using Client Hub.

1. Download and install Xcode from the Apple Developers Web site: *http://developer.apple.com/downloads/*.
2. Log in using your Apple Developer credentials.
3. Download the appropriate Xcode.
4. Navigate to folder `SDK_HOME > ClientHub > src > xcode` and open the project `ClientHub.xcodeproj..`

Note: You can also download the latest version of Xcode using the App store. It is a free download that installs directly into the `Applications` folder. By default, Xcode downloads developer documentation in the background for offline reading, and automatically downloads documentation updates as well.

1. Open the Mac App Store.
2. Under **categories**, select **App development**.
3. Select the **Xcode Developer Tools** and provide **Install app**.
4. Enter your App Store credentials.
5. Download Xcode.

Customizing or Branding the Client Hub User Interface

Open the `ClientHub` project in Xcode to customize the look and feel of the Client Hub application. For example: the splash or welcome screen can be customized to include your company logo or image. Browse through `ClientHub > Targets` and replace the icons and launch image files compliant with iOS standards as per your requirement.

Client Hub Application Signing

1. Create a certificate signing request file to use for authenticating the creation of the SSL certificate:
   - **a.** Launch the Keychain Access application on your Mac (usually found in the **Applications** > **Utilities** folder.
   - **b.** Select **Keychain Access** > **Certificate Assistant** > **Keychain Access**.
   - **c.** Enter your e-mail address and name, then select **Save to disk**, and click **Continue**. This downloads the `.certSigningRequest` file to your desktop.
2. Create a new App ID for the application:

   Note: As a convention, the App ID is in the form of a reversed addresse, for example, com.example.MyPushApp. The App ID must not contain a wildcard character ("*").

   - **a.** Go to the *Apple Developer Member Center* Web site, log in, if required, and select *Certificates, Identifiers & Profiles*.
   - **b.** Select **Identifiers** > **App IDs**, and click the **+**.
   - **c.** Enter a name for your App ID, and, under App Service, select **Push Notifications**. This string should match the Bundle Identifier in your iOS app's `Info.plist`.
   - **d.** Accept the default App ID prefix, or choose another one.
   - **e.** Under App ID Suffix, select **Explicit App ID**, and enter your iOS app's Bundle ID. Verify that all the values are correct.
   - **f.** Click **Submit**.
3. Create a provisioning profile to authenticate your device to run the app you are developing:

   Note: If you create a new App ID or modify an existing one, you must regenerate and install your provisioning file.

   - **a.** Navigate to the *Apple Developer Member Center* Web site, and select *Certificates, Identifiers & Profiles*.
   - **b.** From the iOS Apps section, select **Povisioning File**, and select the + button to create a new provisioning file.
   - **c.** Choose **iOS App Development** as your provisioning profile type, then click **Continue**.
   - **d.** From the drop-down, choose the App ID you created and click **Continue**.
   - **e.** Select your iOS Development certificate in the next screen, and click **Continue**.
   - **f.** Select which devices to include in the provisioning profile, and click **Continue**.
   - **g.** Choose a name for your provisioning profile, then click **Generate**.
   - **h.** Click **Download** to download the generated provisioning file.
   - **i.** Double-click the downloaded provisioning file to install it. Xcode's Organizer opens in the Devices pane. Your new provisioning profile appears in the Provisioning Profiles

section of your Library. Verify that the status for the profile is "Valid profile." If the profile is invalid, verify that your developer certificate is installed in your Keychain.

4. Deploy the Client Hub application on the Device:
   a. In the Client Hub Xcode project, change the bundle identifier in your iOS app's `Info.plist` to the App ID created in Apple Developer Member Center.
   b. In the **TARGETS** > **Build Settings** > **Code Signing**, make sure that appropriate provisioning profile created in Step 3 is selected.

Note: The SSO passcode created is not a single-sign-on credential. Setting up the SSO passcode ensures that you are approving an application to access the stored credentials on the device. This behavior is similar to the SharedKeychain concept in iOS. This requires the apps to be signed by the same developer certificate for sharing the keychain.

Setting the SSO Passcode in Client Hub Application

You must set your SSO passcode in the Client Hub application, and use this passcode in all your applications. Ensure that the SSO passcode is at least 8 characters, and contains at least one uppercase, lowercase, and numeric character.

1. Launch the Client Hub application on your device.

   The **Create SSO Passcode** window is displayed.
2. Enter the SSO passcode, then reenter the passcode to confirm the change.
3. Click **Submit**.

   A success message is displayed if the passcode is accepted and set correctly. Use this SSO passcode for all the applications.
4. Exit the Client Hub application.

Resetting the Client Hub SSO Passcode

(Optional) If you forget the SSO passcode, platform security prevents you from using the applications. You must reset your SSO passcode and use the new passcode in all your applications. Resetting the passcode deletes all data from the secure store.

1. Click **Reset**, then click **OK** to confirm.

   An alert box is displayed for confirmation. If you click **OK**, you are redirected to the **Set passcode** screen.
2. In the **Create SSO Passcode** screen, enter the new passcode, then reenter the passcode to confirm the change.
3. Click **Submit**.

   Use this new passcode for all the applications.

2. Configuring Business Application With Client Hub

The following tasks describe the steps to configure the business application (OData or Kapsel) using Client Hub.

Registering a New Application

Prepare your applications using MAF Logon to work with Client Hub. Applications use a shared keychain. The keychain can be shared only between applications that are signed by the same certificate. Either use the same certificate that you used to sign your version of the Client Hub application, or re-sign the Client Hub using your application certificate.

1. To share a common keychain across two applications, add an entitlements file to your Xcode project:
   a. Create an entitlements file `<PROJECT_NAME>.entitlement` using `Project target > Summary > Entitlements`. Select **Use Entitlements file**.
   b. Add `clienthubEntitlements` keychain group to the entitlements file using `Project target > Summary > Entitlements`. Add `clienthubEntitlements` keychain group.
2. To register your application to the Client Hub, add a configuration descriptor file to your Xcode project.
   a. Create a file named `clienthub.plist`.
   b. In Xcode, go to `File > New > File`.
   c. In the **Choose a template for your new file** modal view, choose `Resource > Property List`.
   d. Right-click the new **Property List > Open As > "Source Code"**.
   e. Add this XML snippet:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<!-- Properties file to provide the application settings. Do
not change the key names. -->
<dict>
 <!--Mandatory Settings-->
        <!--Hostname of the server, example: xyz.sap.corp-->
        <key>Host</key>
        <string>FULLY_QUALIFIED_HOST_NAME</string>
        <!--Port of the server, example: 8080-->
        <key>Port</key>
        <string>PORT</string>
       <!--Security configuration of the application, example:
SSO-->
        <key>SecurityConfiguration</key>
        <string>SECURITY_CONFIGURATION</string>
        <!--Property to set the user creation policy. The user
creation policy defines the authentication method for the user:
automatic, manual or certificate.
        The manual and automatic is for the password based
authentication. The certificate is for the X.509 based
authentication.
        If no value is set, default is certificate. -->
        <key>UserCreationPolicy</key>
        <string>automatic/manual/certificate</string>
 <!--Optional Settings-->
        <!--URL suffix of the relay server or reverse proxy -->
        <key>URLSuffix</key>
        <string>URL_SUFFIX</string>
        <!--Farm ID of the relay server in case it is used, 
example: xyz.farm -->
        <key>FarmID</key>
        <string>FARM_ID</string>
        <!--Domain of the application. Used in SAP Mobile 
Platform older versions. -->
        <key>Domain</key>
        <string>DOMAIN</string>
       <!--Connection type - HTTP or HTTPS. If no value is set, 
default is true (HTTPS)-->
        <key>HTTPS</key>
        <true/>
       <!--Property to set whether the credentials can be shared 
or not. If no value is set, default is true-->
        <key>ShareCredentials</key>
        <true/>
</dict>
</plist>
```

Replace the values (for example, `SECURITY_CONFIGURATION`) with values that are specific to your enterprise. If any of the optional settings are not applicable to your enterprise, leave the string value blank.

3. Deploy your project to your device.
4. Open your MAF Logon-based application. MAF Logon checks if you have Client Hub installed on your device and if the SSO password is specified by the user.
5. MAF Logon displays the Client Hub Logon UI screen, where you can either enter your Client Hub password, or choose skip:
   - To use the app with Client Hub, enter your SSO passcode and tap **Next**. Once all the prerequisites are fulfilled, the **Set Passcode** screen appears, which indicates that the registration is successful. The registration is preformed based on the credentials stored in the Client Hub application shared Data Vault, and the connection data is read using the Client Hub libraries built into the application.
   - If you do not want to use your application with Client Hub, tap **Skip**. You are opted out from using Client Hub to share credentials and connection data with this application. MAF Logon does not present the SSO Passcode UI on subsequent application starts, unless the application is reinstalled.
6. If you enter the SSO Passcode, MAF Logon checks whether it can open Client Hub with the specified password, then stores the password in its own Secure Store.
7. MAF Logon opens Client Hub and requests credentials and connection data from the Client Hub libraries. If the `UserCreationPolicy`, `HTTPS`, and `ShareCredentials` values are not provided, the Client Hub libraries use the default values for the application, from the `clienthub.plist` file.

   If there are no shared credentials yet, MAF Logon presents the Logon UI with only two fields for providing the back-end username and password. When the registration succeeds with these new credentials and the connection data provided by the `clienthub.plist`, it stores the credentials in Client Hub.

Enabling an Application Registered Using Client Hub

To reenable an application that is registered with Client Hub, relaunch the application.

1. MAF Logon checks whether the Client Hub is still present on the device.
    - If it is not, MAF Logon decouples your application from the Client Hub. If you intentionally skip the Client Hub screen, your application never again checks for Client Hub, and cannot share credentials or connection information, even via a new Client Hub installation. To recouple your application with Client Hub, you must delete, then reinstall, the application on the device. When the Client Hub is detected after re-launch, the application shares its credential with Client Hub.
    - If the Client Hub is still available, MAF Logon checks whether the SSO passcode is still valid.

      If the SSO Passcode is invalid, the MAF Logon UI prompts the device user for a new SSO passcode. MAF Logon then opens Client Hub and fetches the credentials stored there.
2. MAF Logon compares the back-end user name and password with the user name and password stored in the secure store of the application.
    - MAF Logon writes the credentials into Client Hub application if:
        - Client Hub does not contain any credentials, or
        - credentials stored in the secure store of the application are newer than those in Client Hub.
    - MAF Logon writes the credentials into the secure store of the application if the credentials stored in the secure store of the application are older than those in Client Hub version.
3. Once the passwords are identical, MAF Logon launches the application process.

Changing the Back-end Password

If there is an authentication error or when the backend password is changed, follow these steps to update the back-end password.

1. MAF Logon presents the `Backend Password` screen to get the new password.
2. Provide the new password.
3. MAF Logon verifies the password, then shares the new password with other applications through the Client Hub.

Maintaining a Private Data Vault

If a business application needs to maintain a private data vault, then you should add `(CFBundleIdentifier)` as the first keychain group prior to `clienthubEntitlements` keychain group in the entitlements file. Ensure to set the access group to default access group `<bundleseedID.bundleID>`, before performing any operations on the private data vault like creating or retrieving data vault. Use the following code snippet example to set the default access group programmatically:

```
NSDictionary *query = [NSDictionary dictionaryWithObjectsAndKeys:
                           kSecClassGenericPassword, kSecClass,
                           @"bundleSeedID", kSecAttrAccount,
                           @"", kSecAttrService,
                           (id)kCFBooleanTrue, kSecReturnAttributes,
                           nil];
    CFDictionaryRef result = nil;
    OSStatus status = SecItemCopyMatching((CFDictionaryRef)query,
(CFTypeRef *)&result);
    if (status == errSecItemNotFound)
        status = SecItemAdd((CFDictionaryRef)query, (CFTypeRef
*)&result);
    NSString *accessGroup = [(NSDictionary *)result
objectForKey:kSecAttrAccessGroup];
    NSArray *components = [accessGroup
componentsSeparatedByString:@"."];
    NSString *bundleSeedID = [[components objectEnumerator]
nextObject];
    NSString *bundleIdentifier = [[NSBundle mainBundle]
bundleIdentifier];
    NSString *defaultaccessGroup = [NSString
stringWithFormat:@"%@.%@",bundleSeedID,bundleIdentifier];

#if !TARGET_IPHONE_SIMULATOR && !TARGET_IPAD_SIMULATOR
    [DataVault setAccessGroup:defaultaccessGroup];
#endif
    CFRelease(result);
```

Managing Android Application Registration Using Client Hub

Use Client Hub, integrated with Logon Manager to register applications for Android devices. SAP provides client-side credentials and a connection settings sharing mechanism for applications that are based on MAF Logon. Client Hub supports both OData and Kapsel applications.

Note:

- The SDK installer includes the source code for Client Hub. SAP does not support customer modifications to the source code after new versions of the template are released. Intellectual property for the template code belongs to SAP. The main purpose for including the source code is to enable code-signing and branding by customers or partners.
- This topic covers the Client Hub application installation and deployment for Eclipse environment only. You can use any other third-party party as required.

1. Getting Started with Client Hub Application Installation and Initialization

The following tasks describe the steps to install and initialize the Client Hub application.

Installing the Client Hub Application

Install SAP Mobile Platform Native SDK - Client Hub component. By default, SAP Mobile Platform SDK components are installed in the `..\SAP\MobileSDKXXX` directory. In this guide, `SDK_HOME` represents the SAP Mobile Platform SDK installation directory, down to the `MobileSDKXXX` folder. Client Hub gets installed under the `ClientHub` directory, where the project files for Client Hub applications, used for registering applications on Android devices is available. Ensure that you unzip the `ClientHub.zip` file before importing the Client Hub project into Eclipse.

Setting Up the Development Environment

The Client Hub application is shipped as a source code project. Set up the Android Development Environment before registering your application using Client Hub.

- Download the Java Standard Edition (6 Update 24 and above versions) Development Kit (JDK) from *http://www.oracle.com/technetwork/java/javase/downloads/index.html*.
- Download the Android Developer Tool (ADT) Bundle from *http://developer.android.com/sdk/index.html*. With the single download of the ADT Bundle, you can get all the essential Android SDK components and Eclipse IDE installed.

  For more information on supported Google Android development environment in SAP Mobile Platform 3.0, see *http://service.sap.com/pam*.

Importing Client Hub Project into Eclipse

1. Open **Eclipse WorkSpace**.
2. Click `File > Import... > Android > Existing Android Code Into Workspace`. Click **Next**.
3. In the Root Directory, click **Browse...**, select `C:\SAP\MobileSDKXXX directory \ClientHub`. Click **OK**. The Client Hub project is selected by default.
4. Click **Finish**. The `ClientHub` folder is displayed in your Eclipse Package Explorer.

Client Hub Application Signing

Build the Client Hub application, cosigned with the same developer certificate as the application using these steps:

1. Right-click `ClientHub` folder in the package explorer. Navigate to `Android Tools > Export Signed Application Package`.
2. In the **Export Android Application** wizard, the `ClientHub` project appears. Click **Next**.
3. In the **Keystore selection** window, select:

   a. Select **existing keystore** if you already have a keystore:
      1. Enter the location of the existing keystore.
      2. Enter the password.
      3. Click **Next**.
      4. Choose **Use existing key**. Enter the alias and password, click **Next**.
   b. Select **new keystore** to create a new key store.
      1. Enter the location where the new keystore should be created.
      2. Enter the password, then reenter the password for confirmation.
      3. Click **Next**.
      4. In the **Key Creation** window, enter the details. Click **Next**.
4. In the **Destination and key/certificate checks** window, enter the destination APK file name.
5. Click **Finish**.

   This process creates an Android Client Hub executable (.apk) that can be deployed on the device or emulator.

Note: The SSO passcode created is not a single-sign-on credential. Setting up the SSO passcode ensures that you are approving an application to access the stored credentials on the device.

Customizing or Branding the Client Hub User Interface

After importing the `ClientHub` project, you can customize the look and feel of the Client Hub application. For example: the splash or welcome screen can be customized to include your company logo or image. Browse through the `ClientHub > res` folder and replace the resource files as per your requirement.

Setting the SSO Passcode in Client Hub Application

You must set your SSO passcode in the Client Hub application, and use this passcode in all your applications. Ensure that the SSO passcode is at least 8 characters, and contains at least one uppercase, lowercase, and numeric character.

1. Launch the Client Hub application on your device.

   The **Create SSO Passcode** window is displayed.
2. Enter the SSO passcode, then reenter the passcode to confirm the change.
3. Click **Submit**.

   A success message is displayed if the passcode is accepted and set correctly. Use this SSO passcode for all the applications.
4. Exit the Client Hub application.

Resetting the Client Hub SSO Passcode

If you forget the SSO passcode, platform security prevents you from using the applications. You must reset your SSO passcode in the Client Hub application, and use the new passcode in all your applications. Resetting the passcode deletes all data from the secure store.

1. Click **Reset**, then click **OK** to confirm.

   An alert box is displayed for confirmation. If you click **OK**, you are redirected to the **Set passcode** screen.
2. In the **Create SSO Passcode** screen, enter the new passcode, then reenter the passcode to confirm the change.
3. Click **Submit**.

   Use this new passcode for all the applications.

2. Configuring Business Application With Client Hub

The following tasks describe the steps to configure the business application using Client Hub.

Registering a New Application Using Client Hub

1. To get connection settings for Client Hub, add a configuration descriptor file to your Eclipse project.
   a. Create a file named `clienthub.properties` and place it into the `/res/raw` folder of your Android project.
   b. Add this content:

```
#Properties file to provide the application settings. Do not
change the key names.

#Mandatory Settings


#Hostname of the server, example:xyz.sap.corp
Host=<FULLY_QUALIFIED_HOSTNAME>
#Port of the server, example: 8080
Port=<PORT>
#Farm ID of the relay server in case it is used. Example:
xyz.farm. If relay server is not used, set the value to 0.
FarmID=<FARM_ID>
#Security configuration of the application, example: SSO
SecurityConfiguration=<SECURITY_CONFIGURATION>
#Property to set the user creation policy. The user creation
policy defines the authentication method for the user:
automatic, manual or certificate.
The manual and automatic is for the password based
authentication. The certificate is for the X.509 based
authentication.
If no value is set, default is certificate.
UserCreationPolicy=<automatic/manual/certificate>

#Optional Settings


#URL suffix of the relay server or reverse proxy.
URLSuffix=<URL_SUFFIX>
#Domain of the application. Used in SAP Mobile Platform older
versions.
Domain=<DOMAIN>
```

```
#Connection type, HTTP or HTTPS. If no value is set, default is
true (HTTPS).
HTTPS=<true/false>
#Property to set whether the credentials can be shared or not.
If no value is set, default is true.
ShareCredentials=<true/false>
```

Replace the values (for example, SECURITY_CONFIGURATION) with values that are specific to your enterprise.

2. Ensure that the following permission is present in the MAF Logon-based application's `androidManifest.xml` file within the `<manifest>` tag. If not, add the permission:

```
 <uses-permission
android:name="com.sap.mobile.clientHub.CLIENTHUB_ACCESS_PERMISSIO
N"/>
```

3. Deploy the MAF Logon-based application to your device.
4. Open your MAF Logon-based application. MAF Logon checks whether Client Hub is installed on your device, and if the SSO password is specified by the user.
5. MAF Logon displays the Client Hub Logon UI screen, where you can either enter your Client Hub SSO password, or choose skip:
    - To use the app with Client Hub, enter your SSO passcode and tap **Next**. Once all the prerequisites are fulfilled, the **Set Passcode** screen appears, which indicates that the registration is successful. The registration is preformed based on the credentials stored in the Client Hub application shared Data Vault, and the connection data is read using the Client Hub libraries built into the application.
    - If you do not want to use your application with Client Hub, tap **Skip**. You are opted out from using Client Hub to share credentials and connection data with this application. MAF Logon does not present the SSO Passcode UI on subsequent application starts, unless the application is reinstalled.
6. If you enter the SSO Passcode, MAF Logon checks whether it can open Client Hub with the specified password, then stores the password in its own Secure Store.
7. MAF Logon opens Client Hub and requests credentials and connection data from the Client Hub libraries. If the `UserCreationPolicy`, `HTTPS`, and `ShareCredentials` values are not provided, the Client Hub libraries use the default values for the application, from the `clienthub.properties` file.

    If there are no shared credentials yet, MAF Logon presents the Logon UI with only two fields for providing the back-end username and password. When the registration succeeds with these new credentials and the connection data provided by the `clienthub.properties`, it stores the credentials in Client Hub.

Enabling an Application Registered Using Client Hub

To reenable an application that is registered with Client Hub, relaunch the application.

1. MAF Logon checks whether the Client Hub is still present on the device.
    - If it is not, MAF Logon decouples your application from the Client Hub. If you intentionally skip the Client Hub screen, your application never again checks for Client

Hub, and cannot share credentials or connection information, even via a new Client Hub installation. To recouple your application with Client Hub, you must delete, then reinstall, the application on the device. When the Client Hub is detected after re-launch, the application shares its credential with Client Hub.

- If the Client Hub is still available, MAF Logon checks whether the SSO passcode is still valid.

  If the SSO Passcode is invalid, the MAF Logon UI prompts the device user for a new SSO passcode. MAF Logon then opens Client Hub and fetches the credentials stored there.

2. MAF Logon compares the back-end user name and password with the user name and password stored in the secure store of the application.
   - MAF Logon writes the credentials into Client Hub application if:
     - Client Hub does not contain any credentials, or
     - credentials stored in the secure store of the application are newer than those in Client Hub.
   - MAF Logon writes the credentials into the secure store of the application if the credentials stored in the secure store of the application are older than those in Client Hub version.
3. Once the passwords are identical, MAF Logon launches the application process.

Changing the Back-end Password

If there is an authentication error or when the backend password is changed, follow these steps to update the back-end password.

1. MAF Logon presents the `Backend Password` screen to get the new password.
2. Provide the new password.
3. MAF Logon verifies the password, then shares the new password with other applications through the Client Hub.

Index