HCL MT CH-MSP Product Documentation

Configuring Mobile for the Multi-Tenant environment

Configuring Mobile for the Multi-Tenant environment

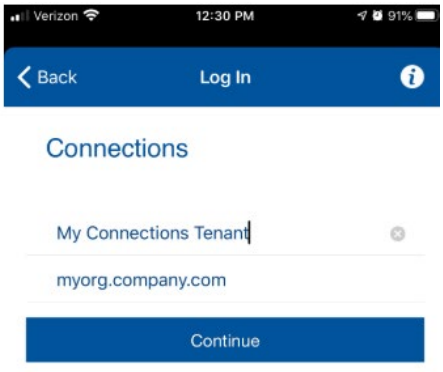
Overview

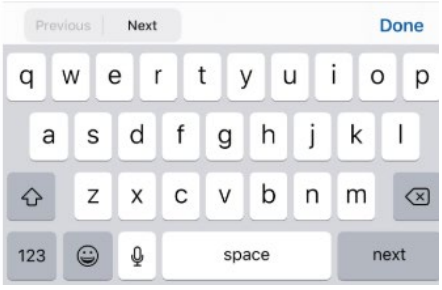
The Connections mobile app for iOS and Android fully supports the Connections MT environment. User's must be using version 6.5.4 or later, as these versions contain critical fixes for multi-tenant. By default, the server's mobile-config.xml has been provisioned to require a minimum version of 6.5.4, so user's with previous app versions will receive an error asking them to upgrade to the latest version before they can connect to the server.

The Connections Mobile app for iOS is available to any user from the Apple App Store and Connections Mobile app for Android is available from Google Play.

Using the Connections Mobile app

User's of the app must use the "My company's server" option in the account setup wizard and then enter the hostname of their MT deployment server. A full URL is not required. For example:





Touch Continue and the user will be presented with their company's login screen (or the hosting company's screen, depending on how Keycloak is setup for this user).

Administration

An organization administrator can access Connections Mobile policies and configuration, including setting up Connections mobile extensions by using the URL:
https://org_host/mobileAdmin

The user must have been assigned the role **org-admin** within the WebSphere Administration J2EE roles for the application named "Mobile Administration".

For more information on the policies and settings, see
https://help.hcltechsw.com/connectionsmobile/admin/overview/r_cloud_mobile_config_properties.html

Keycloak Setup

Certificates

The Connection mobile app requires that the SSL certificate used by all endpoints is valid and trusted. Self-signed certificates or expired certificates will cause the app to fail and there is no suitable workaround. Ensure that all public endpoints for the Connections server and for the Keycloak token provider are secured using certificate authorities that have trusted root certificates shipping with the Apple iOS and Android mobile operating systems.

Create client for Connections Mobile client

Using the Keycloak admin console, you must create an entry for a new client ID named **connections_social_mobile** in the realm that is being used by Connections.

Navigate to Clients and select Create to create a new client. Fill out the following fields, the rest can remain blank or unset.

Key	Value
Client ID	connections_social_mobile
Enabled	ON
Consent Required	OFF
Client Protocol	openid-connect
Access Type	public
Standard Flow Enabled	ON
Implicit Flow Enabled	OFF
Direct Access Grants Enabled	OFF
Valid Redirect URIs	com.ibm.ibmscp://com.ibm.mobile.connections/token

Special considerations for Keycloak based identity providers

If you are configuring customer specific Identity Providers via Keycloak, and looking to have mobile and plugin users automatically login with their customer specific IDP, see the topic Using Keycloak Identity Providers with Mobile and Plugins.

Authentication and Tokens

Connections mobile uses the Open ID Connect protocol with OAuth 2.0 to identify users and stores access and refresh tokens in secure storage on the mobile device. It has been tested with Keycloak as the token provider and Keycloak can be federated with a customer's Identity provider to provide per-customer validation forms and inputs. The mobile application uses the same user validation forms that are used on the web client.

As part of the OAuth and OIDC protocols, the mobile app will receive and use access and refresh tokens. Access tokens are typically short-lived tokens which are revalidated multiple times a day. A refresh token is used to silently "refresh" the access token. The mobile app uses a scope called "offline_access" which is used to ask the token provider for a refresh token that can be used for an extended time. Once the access token expires, the mobile app silently uses the refresh token to ask Keycloak for an updated access token. Keycloak then validates this token, checks the expiration time on the refresh token and ensures that the user is still valid and has not had their access to the system revoked. If all of these checks pass, then Keycloak will issue a new access token valid for a short duration.

There are a couple of differences in the session timeouts that should be noted.

1. By default, the access token expiration will inherit the same value as **Access Token Lifespan** that is defined in Keycloak under Realm Settings > (realm) > Tokens. It is possible to define a client unique Access Token Lifespan but it is not recommended.
2. The refresh token expiration period is controlled by 3 settings in Keycloak under Realm Settings > (realm) > Tokens.

   Offline Session Idle – Time an offline session is allowed to be idle. The client must use the refresh token at least once during this period or the session will expire, and the user must login again. For example, setting this to 7 days will require that the mobile user must use the Connections app at least once in 7 days and if not, they will be prompted to login again.
   Offline Session Max Limited – set this to on to force a maximum session lifetime limit.
   Offline Session Max – Maximum time that can pass before the session is expired, regardless of activity. For example, set this to 30 days to force all mobile users to login again even if they are actively using the application.
3. The desktop plugin app shares the Offline session timeouts with the Connections mobile apps, since both use offline tokens.

Using Keycloak Identity Providers with Mobile and Plugins

Overview

When the Connections Mobile and Connections Plugin applications request OAuth tokens from Keycloak, they will first authenticate the user by calling the Keycloak authentication endpoint. By default, Keycloak will display a user login form and optionally will provide a list of Identity Providers for which this user could choose to authenticate. However, in a Multi-Tenant environment, the tenant provider should either customize the login form to make it specific for this particular tenant that is logging in, or possibly route the login request to a tenant specific Identity Provider, which would perform the authentication of the user. In either of the cases however, Keycloak will need some sort of hint from the login request to understand which "tenant" this login should apply.

By default, there is no tenant information such as the hostname of the tenant embedded within the auth request made from the client apps to Keycloak. The apps make a REST call directly to the Keycloak server and provide only the minimum required information needed by an OAuth provider. For example, the apps would make a call similar to the following:

https://login.example.com/auth/realms/connmt/protocol/openid-connect/auth?response_type=code&code_challenge_method=S256&scope=openid%20offline_access&code_challenge=9aE4DTOwWk66U8Hh2k1TpSBelWggFnroXEkRmIbAAVY&redirect_uri=com.ibm.ibmscp://com.ibm.mobile.connections/token&client_id=connections_social_mobile&state=Q5t9rXggFR7VB0HyYOL2b_- vFMLIHPiO9OGRUE6O5a0

Note that Keycloak recognizes a Keycloak unique parameter called **kc_idp_hint**. If Keycloak detects this parameter, it checks the value against its list of Identity Provider aliases, and if a match is found, the login request will skip the Keycloak login form and be routed directly to that identity provider.

Solution

The Connections mobile server, mobile apps and plugins support additional custom parameters on the OAuth authentication URL.

These parameters must be added to the mobile-config.xml file on the Connections server, and then will be discovered by each of the

applications during their login process. This XML comment describes the OAuthAuthorizationURL_Parameter value:

```
<!-- OAuthAuthorizationURL_Parameters: Optional. If custom request parameters are required on the OAuthAuthorizationURL
request, add the parameters here. The mobile and plugins apps will append these parameters when requesting app and user
authorization.

Separate parameters using the "&amp;" string. This will be converted to a single "&" when used by the client apps.

Parameter values support the following substitution tokens. If these tokens are used as part of the value for a given parameter, they

will be replaced with a dynamic value at runtime. Supported substitution tokens:

    {MT_HOSTNAME} = The fully qualified tenant specific hostname for this instance. For example, "tenant.example.com"
    {MT_ORGNAME} = The tenant org name for this instance. For example, if the MT_HOSTNAME is "tenant.example.com", the
    MT_ORGNAME would be "tenant".

    For example:
    <OAuthAuthorizationURL_Parameters>parameter1=value1&amp;parameter2={MT_ORGNAME}</OAuthAuthorizationURL_Para
-   meters>
-
>
```

The parameter name that is added could be something generic like mt_hostname={MT_HOSTNAME} or mt_org={MT_ORGNAME}. Or, if you are already assigning an Identity provider alias in Keycloak that exactly matches the tenant org name, then you could directly add the parameter **kc_idp_hint=**
{MT_ORGNAME}. With this approach, there would not be a requirement for an NGINX or edge proxy rule to manipulate the tenant identity to the proper value required for kc_idp_hint.

To add tenant specific information to the authentication request, follow these steps:

1. On the Connections server, edit the mobile-config.xml file located in the DMgr profile. Typically this file structure would be similar to /opt/IBM/WebSphere/AppServer/profiles/Dmgr01/config/cells/mtdemo1Cell01/LotusConnections-config.
2. Find the key in the file called **<OAuthLogoutURL>**. Add a new line here and insert the following section. Note that the value is just an example and should be customized to your environment.
<OAuthAuthorizationURL_Parameters>mt_org={MT_ORGNAME}</OAuthAuthorizationURL_Parameters>
3. Save the file
4. Perform a Full Synchronization for all nodes using the WebSphere admin console
5. Restart the mobile application using the WebSphere admin console

If you are using the parameter kc_idp_hint directly, then no further steps should be necessary on the Keycloak side. However, if your Keycloak alias naming convention does not match the customer org name and/or you are using a different parameter name, it may be necessary to add an NGINX or proxy rule to modify the incoming auth requests.

Verifying the changes

Once the changes have been made, do the following:

1. Using a desktop browser and the hostname of one of your tenants, open this URL:
https://tenant_org_hostname/mobile/homepage/SecurityConfiguration?debug=true Verify that the returned JSON includes the correct key and value for OAuthAuthorizationURL_Parameters
2. Once #1 is checked, login to this tenant using a Connections Mobile app or Connections desktop plugin application to verify login works properly.

IMPORTANT

The behavior described in this article requires the Connections Mobile app version 6.5.5 or later, or the Connections plugins from late June 2020.

Configuring Desktop Plugins for the Multi-Tenant environment

Integrating plug-ins for HCL Connections

Access and update your Connections content from other applications. Use the plug-ins to share files and information between Microsoft™ Windows™ applications and HCL Connections 6.5 CR1a Multi-Tenant.

You must provide information about a Connections server before you can share files and information between Microsoft™ Windows™ and Connections.

Connecting to an HCL Multi-Tenant Connections. Follow the instructions under the section "**Procedure: Connecting to a multi-tenant Connections solution"**
https://help.hcltechsw.com/connections/v65/connectors/enduser/t_ms_plugins_connect.html

Managing HCL Connections for Mac accounts
https://help.hcltechsw.com/connections/v65/connectors/enduser/t_mac_plugins_connect.html

Using the HCL Connections desktop plug-ins for Microsoft Windows
https://help.hcltechsw.com/connections/v65/connectors/enduser/c_ms_plugins_win_explorer.html

Keycloak Setup

Create client for Connections Mobile client

Using the Keycloak admin console, you must create an entry for a new client ID named **conn-dsk-plugin** in the realm that is being used by Connections.

Navigate to Clients and select Create to create a new client. Fill out the following fields, the rest can remain blank or unset.

Key	Value
Client ID	conn-dsk-plugin
Enabled	ON
Consent Required	OFF
Client Protocol	openid-connect
Access Type	public
Standard Flow Enabled	ON
Implicit Flow Enabled	OFF
Direct Access Grants Enabled	OFF
Valid Redirect URIs	com.ibm.ibmscp://com.ibm.desktop.connections

Special considerations for Keycloak based identity providers

If you are configuring customer specific Identity Providers via Keycloak, and looking to have mobile and plugin users automatically login with their customer specific IDP, see the topic Using Keycloak Identity Providers with Mobile and Plugins.

Authentication and Tokens

With Connections MT, the Connections Desktop Plugins uses the Open ID Connect protocol with OAuth 2.0 to identify users. It has been tested with Keycloak as the token provider and Keycloak can be federated with a customer's Identity provider to provide per-customer validation forms and inputs. The desktop plugin application uses the same user validation forms that are used on the web client.

As part of the OAuth and OIDC protocols, the desktop plugin app will receive and use access and refresh tokens. Access tokens are typically short-lived tokens which are revalidated multiple times a day. A refresh token is used to silently "refresh" the access token. The desktop plugin app uses a scope called "offline_access" which is used to ask the token provider for a refresh token that can be used for an extended time. Once the access token expires, the desktop plugin app silently uses the refresh token to ask Keycloak for an updated access token. Keycloak then validates this token, checks the expiration time on the refresh token and ensures that the user is still valid and has not had their access to the system revoked. If all of these checks pass, then Keycloak will issue a new access token valid for a short duration.

There are a couple of differences in the session timeouts that should be noted.

1. By default, the access token expiration will inherit the same value as **Access Token Lifespan** that is defined in Keycloak under Realm Settings > (realm) > Tokens. It is possible to define a client unique Access Token Lifespan but it is not recommended.
2. The refresh token expiration period is controlled by 3 settings in Keycloak under Realm Settings > (realm) > Tokens.

   Offline Session Idle – Time an offline session is allowed to be idle. The client must use the refresh token at least once during this period or the session will expire, and the user must login again. For example, setting this to 7 days will require that the desktop plugin user must use the Connections app at least once in 7 days and if not, they will be prompted to login again.
   Offline Session Max Limited – set this to on to force a maximum session lifetime limit.
   Offline Session Max – Maximum time that can pass before the session is expired, regardless of activity. For example, set this to 30 days to force all desktop plugin users to login again even if they are actively using the application.
3. The desktop plugin app shares the Offline session timeouts with the Connections mobile apps, since both use offline tokens.

Validating OIDC Discovery URL

As part of the MT update scripts, the script will add the following stanza to the IBM HTTP Server configuration (where keycloak_hostname is really the hostname of your keycloak instance in your environment):

```
# OIDC discovery for the backend Keycloak OIDC server
Redirect "/.well-known/openid-configuration" "https://keycloak_hostname/auth/realms/connmt/.well-known/openid-configuration"
```

The desktop plugin clients will call the URL https://mt_org_hostname/.well-known/openid-configuration when an MT account is configured and will expect this URL to be unprotected. If your environment is using an edge proxy, make sure that this URL and the redirect URL are listed as unprotected.

If you are experiencing problems connecting a desktop plugin client, try the URL https://mt_org_hostname/.well-known/openid-configuration (using your organization hostname) in a private browser session and ensure that a json payload is returned containing links to Keycloaks auth and token endpoints. For example:


```
{
"issuer": "https://keycloak_hostname/auth/realms/connmt",
"authorization_endpoint": "https://keycloak_hostname/auth/realms/connmt/protocol/openid-
connect/auth",
"token_endpoint": "https://keycloak_hostname/auth/realms/connmt/protocol/openid-connect/token",
"token_introspection_endpoint": "https://keycloak_hostname/auth/realms/connmt/protocol/openid-
connect/token/introspect",
"userinfo_endpoint": "https://keycloak_hostname/auth/realms/connmt/protocol/openid-
connect/userinfo",
"end_session_endpoint": "https://keycloak_hostname/auth/realms/connmt/protocol/openid-
connect/logout",
"jwks_uri": "https://keycloak_hostname/auth/realms/connmt/protocol/openid-connect/certs",
"check_session_iframe": "https://keycloak_hostname/auth/realms/connmt/protocol/openid-
connect/login-status-iframe.html",
...
}
```

Configuring Sametime for the Multi-Tenant environment

About this topic
This topic contains general information and guidance on integrating Sametime with Connections.

1. Configure Sametime integration with Connections as per usual Connections and Sametime documentation (i.e. LotusConnections-config.xml changes)
2. If you want to share the same LDAP between Connections and Sametime then make sure you populate the ibm-socialOrganizationName attribute in the person record appropriately.
   Note: This attribute is ALREADY in the schema that you imported into LDAP and configured to use with Connections.