Profile Client

Workcloud Communication

ZEBRA

Configuration and Programmer Guide

MN-004351-10EN Rev A

Terms of Use

Proprietary Statement

Product Improvements

Liability Disclaimer

Limitation of Liability

Contents

About this Guide

This guide provides Zebra Profile Client configuration information and is intended for third-party launcher developers, MDM administrators, system engineers, and other parties that administer the Profile Client.

Configuration information is communicated to the Profile Client through JSON elements as described in this guide. This guide also describes Android intents that comprise the application programming interface (API) for the Profile Client, including intents that trigger the Profile Client to consume the JSON configuration information. This guide assumes the reader is familiar with the Profile Client and Android intents.

This document is part of the Profile Manager Solution documentation set available at www.zebra.com/us/en/support-downloads/software/productivity-apps/profile-manager.html. Refer to these documents for more information on the Profile Manager (PFM) Solution and the Profile Client in particular.

Chapter Descriptions

This guide include the following sections:

- About this Guide explains document conventions and provides service information.
- Profile Client Configuration provides information about starting the Profile Client with configuring and without configuring Profile Client.
- Intents and Actions explains how to log in, and log out of Profile Client.
- Additional Support for Third Party Launcher explains how to log in to the Profile Client without intents.

Document Conventions

- Extras: These are standard part of Android intents and are indicated in the tables accompanying each intent as “Extra 0”, “Extra 1”, etc., along with the type, name, and acceptable values.
- This guide specifies a minimum version for each intent. However, it is strongly recommended to upgrade to the latest Profile Client version in order to get current fixes and new features.

Notational Conventions

The following notational conventions make the content of this document easy to navigate.

- **Bold** text is used to highlight the following:
  - Dialog box, window, and screen names
  - Dropdown list and list box names
  - Checkbox and radio button names
  - Icons on a screen
  - Key names on a keypad
  - Button names on a screen
- Bullets (•) indicate:
  - Action items
  - List of alternatives
  - Lists of required steps that are not necessarily sequential.
- Sequential lists (for example, those that describe step-by-step procedures) appear as numbered lists.

Service Information

If you have a problem with your equipment, contact Zebra Global Customer Support for your region. Contact information is available at: zebra.com/support.

When contacting support, please have the following information available:

- Serial number of the unit
- Model number or product name
- Software type and version number

Zebra responds to calls by email, telephone, or fax within the time limits set forth in support agreements.

If your problem cannot be solved by Zebra Customer Support, you may need to return your equipment for servicing and will be given specific directions. Zebra is not responsible for any damages incurred during shipment if the approved shipping container is not used. Shipping the units improperly can possibly void the warranty.

If you purchased your Zebra business product from a Zebra business partner, contact that business partner for support.

Revision History

Table 1 Revision History

Revision	Date	Description
MN-004351-01EN Rev A	12/2021	Initial Release
MN-004351-02EN Rev A	03/2022	Updated Proxy URL information.
MN-004351-03EN Rev A	03/2022	Added intent for enrolling the device.

Table 1 Revision History (Continued)

Revision	Date	Description
MN-004351-04EN Rev A	04/2022	Added parameters to retrieve store RO Property.
MN-004351-05EN Rev A	05/2022	Updated the default value for store_ro_property and store_regex_pattern in JSON configuration variable table.
MN-004351-06EN Rev A	07/2023	Added the Selected Role Response intent.
MN-004351-07EN Rev A	01/2024	Add JSON Configuration parameter and an intent in Starting the Profile Client with Configuration.
MN-004351-08EN Rev A	04/2024	Added Suppress Data Consent Dialog and rebranded Workforce Connect and WFC to Workcloud Communication and Zebra.
MN-004351-09EN Rev A	06/2024	Updated login_info_response payload parameters.
MN-004351-10EN Rev A	06/2024	Added Data Collection feature.

Profile Client Configuration

This section explains how a JSON file on the device conveys configuration elements to the Zebra Profile Client, or how the user configures parameters manually using Profile Client settings through intents if the Client operates without configuring JSON element. The Profile Client User Guide describes how to configure Profile Client manually.

This section provides information the following information on the following:

- Profile Manager connectivity
- Customer and site ID
- Third-party launcher
- Profile Client display settings
- Enable/disable for other Profile Client features

JSON Sample Configuration

The following JSON configuration sample may be contained in a file or in a JSON structure sent as an extra, as described in Starting the Profile Client with Configuration.

```
{
"customer_id": "3001",
"sfs_url": "https://<   PFM_server.com  >",
"site_id": 5000,
"log_level": "debug",
"confirm_role":true,
"power_connected_logout":false,
"log_file": false,
"dnd_switch":true
"config_settings":0,
"disable_signout_btn_bluefletch":false,
"supress_network_disconnect_bar":true,
"secret_key": "my_secret_key",
"key_user_name":"userNameInput",
"key_user_pwd":"passwordInput",
"key_submit":"submitButton",
"key_domain":"pttpro",
"username_layout":"usernamelayout",
"password_layout":"passwordlayout",
"max_tryouts_for_finding_user":20,
"proxy_url":" ",
"proxy_url_api_key":" ",
"force_login_after_os_kill":false,
"enableGA":0,
}
```

Table 2 describes the valid JSON elements.

Table 2 JSON Configuration Variables for the Zebra Profile Client

Label	Description	Type	Default	Config via UI
customer_id	Maps to the Profile Manager's Tenant ID provided by the system administrator.	string	null	Yes
sfs_url	The FQDN of the Profile Manager secure web socket connection provided by the system administrator.	string	null	Yes
site_id	`site_id` value is correlated with the Extension Manager's Store ID value. Upon a match, it aligns the group of extensions available to the user based on the roles configured.	string	null	Yes
log_level	Log options are: • Info (default) • Debug • Warning • Error • Verbose.	string	info	Yes
confirm_role	When set to `True`, the user is prompted to confirm the selected roles.	Boolean	TRUE	Yes
power_connected_logout	When enabled, signs out the Profile Client user, when the device is put in a cradle for charging.	Boolean	FALSE	Yes
log_file	Enable the application to record logs and store the device at the following location: • /sdcard/Android/data/com.zebra.dfs/files/DFS/ for WPC Version 2.0.21200 and above. • /sdcard/DFS in A11 • /sdcard/DFS/ for all other versions.	Boolean	FALSE	Yes

Table 2 JSON Configuration Variables for the Zebra Profile Client (Continued)

Label	Description	Type	Default	Config via UI
dnd_switch	Allows the user to display/ remove the `dnd_switch` in Profile Client.	Boolean	TRUE	No
config_settings	Configures the settings menu visibility: • 0 = (default) Setting Menu is visible and editable. • 1 = Setting Menu is visible but not editable. • 2 = Setting Menu is not visible.	integer	0	No
disable_signout_btn_bluefletch	Configures whether the **Sign Out** button is visible when the auth type is BlueFletch. • `false` = (default) **Sign Out** the button is visible. • `true` = **Sign Out** button is not visible.	Boolean	FALSE	No
supress_network_disconnect_bar	Configures whether the network disconnect bar is visible at the bottom of the screen during a network disconnection. • `true` = (default) Network disconnect bar is not visible. • `false` = Network disconnect bar is visible.	Boolean	TRUE	No
secret_key	Key used to decrypt the login blob delivered from the launcher app through intent to the Profile Client.	string	null	
key_user_name	Used with third-party launcher apps. This tag identifies the Username ID field and allows input on the current login web page. Sample HTML: <input type="`text`" name="`username`" id="`username`" class="textinput" value="">	string	null	No

Table 2 JSON Configuration Variables for the Zebra Profile Client (Continued)

Label	Description	Type	Default	Config via UI
key_user_pwd	Used with third-party launcher amax_tryouts_for_finding_user apps. This tag identifies the Password ID field and provides input on the current login web page. Sample HTML <input type="`password`" name="`password`" id="password" class="textinput" autocomplete="off">.	string	null	No
key_submit	Used with third-party launcher apps. This tag identifies the Submit ID field for acceptance of credentials on the current login web page. Sample.html: <input type=" submitButton" value="Login" id="submit" class="submit" tabindex="4" role="button" >	string	null	No
key_domain	Used in conjunction with third-party launcher apps. This tag inserts a Domain name string to add before the username. The slash separators are automatically added. For example: "pttpro" becomes "pttpro\\<username>"	string	null	No
username_layout	Used to configure the user name field of the customized EKB layout in EC30 devices. May require change if the layout encryption file uses a different name in the future.	string	usernamelayout	No
password_layout	Used to configure the password field of the customized EKB layout in EC30 devices. May require change if the layout encryption file uses a different name in the future.	string	passwordlayout	No
max_tryouts_for_finding_user	The pop-up is displayed for the maximum time, even if the user does not belong to the respective department while applying for the Zebra PTT Pro role.	integer	20	No
proxy_url	The Proxy URL is used to configure the Proxy PFM when the environment is a multi-PFM instance or device enrollment is required.	String	“”	Yes
proxy_url_api_key	The API key for Proxy PFM URL.	String	“”	No

Table 2 JSON Configuration Variables for the Zebra Profile Client (Continued)

Label	Description	Type	Default	Config via UI
force_login_after_os_kill	Re-login automatically if Profile Client is restarted.	Boolean	FALSE	No
store_ro_property	The RO property is read to retrieve the store from the device at the time of enrolling when the store is not passed.	string	"dhcp.wlan0.domain"	No
store_regex_pattern	The regex pattern retrieves the store from the RO property value.	string	"\\d{4}"	No
store_regex_index	The index within the group is searched for regex and matched with the store_regex_pattern value.	string	“0”	No
store_regex_group	The index searches the group for regex and matches it with the store_regex_pattern value.	string	“0”	No
proxy_pfm_timeout_in_sec	The proxy server timeout for proxy rest API is called for enrolling in seconds.	integer	180	No
customUserAgentString	To enable the Google IDP Login, you must pass any valid string, such as Zebra/Zebra, to enable it.	string	“”	No
launch_on_imprivata_unlock	To enable Profile Client to be in the foreground. • true: The Profile Client appears in the foreground. • The Profile Client does not appear in the foreground	Boolean	FALSE	No
enableGA	To enable data collection in the Profile Client.	string	1	No

NOTE: Zebra collects Workcloud Communication usage and performance data to ensure the quality of the products delivered to our customers. This feature has been integrated and enabled by default with Profile Client from version 2.0.24200. This can be disabled by setting enableGA = 0.

Intents and Actions

Profile Client Intents Overview

The API consists of a series of intents that can be invoked via the following methods:

- **StageNow**: This staging software can generate an Android intent as described at Zebra TechDocs. The intent can be sent immediately, scheduled, or triggered by pressing a button or sensor.
- **MDM**: An Android intent can be generated by various MDMs, such as the following:
  - **Soti Mobicontrol**: SOTI MOBICONTROL.
  - **Airwatch/Workspace ONE** VMWARE Workspace One.
- **Android application**: An Android application can generate an intent to another application on the device. For more information, refer to Android Developers Documentation.

When Zebra Profile Client Android receives the intent, it takes its indicated action.

This guide describes the flow of intents and each intent in detail.

Third-party applications can use Profile Manager intents to start the Profile Client, load configuration, log in and log out, and request status. In addition, the Profile Client can send two intents back to the third-party application to signal login completion and respond to a status request.

The following steps describe typical intent usage, also illustrated in the diagram that follows:

1. The ACTION_NEW_CONFIG intent starts (or restarts) and configures the Profile Client. The location of a JSON configuration file or a JSON configuration blob accompanies this intent. Typically, configuration setup through this intent only happens once.

   The LoginActivity intent can start (or restart) the Profile Client without configuration.

2. After starting the Profile Client, the ACTION_LOGIN intent can be sent to log it in to the Profile Manager-

   If the login information sent with this intent includes a request ID and package name, the Profile Client sends an ACTION_LOGIN_RESPONSE when login is completed. The response is not sent until the user selects a role (if applicable) and the Zebra Voice and/or PTT Pro clients are logged in.

3. The third-party application may periodically request login status from the Profile Client through ACTION_LOGIN_STATUS intent. Upon receipt, the Profile Client responds with ACTION_LOGIN_STATUS_RESPONSE.

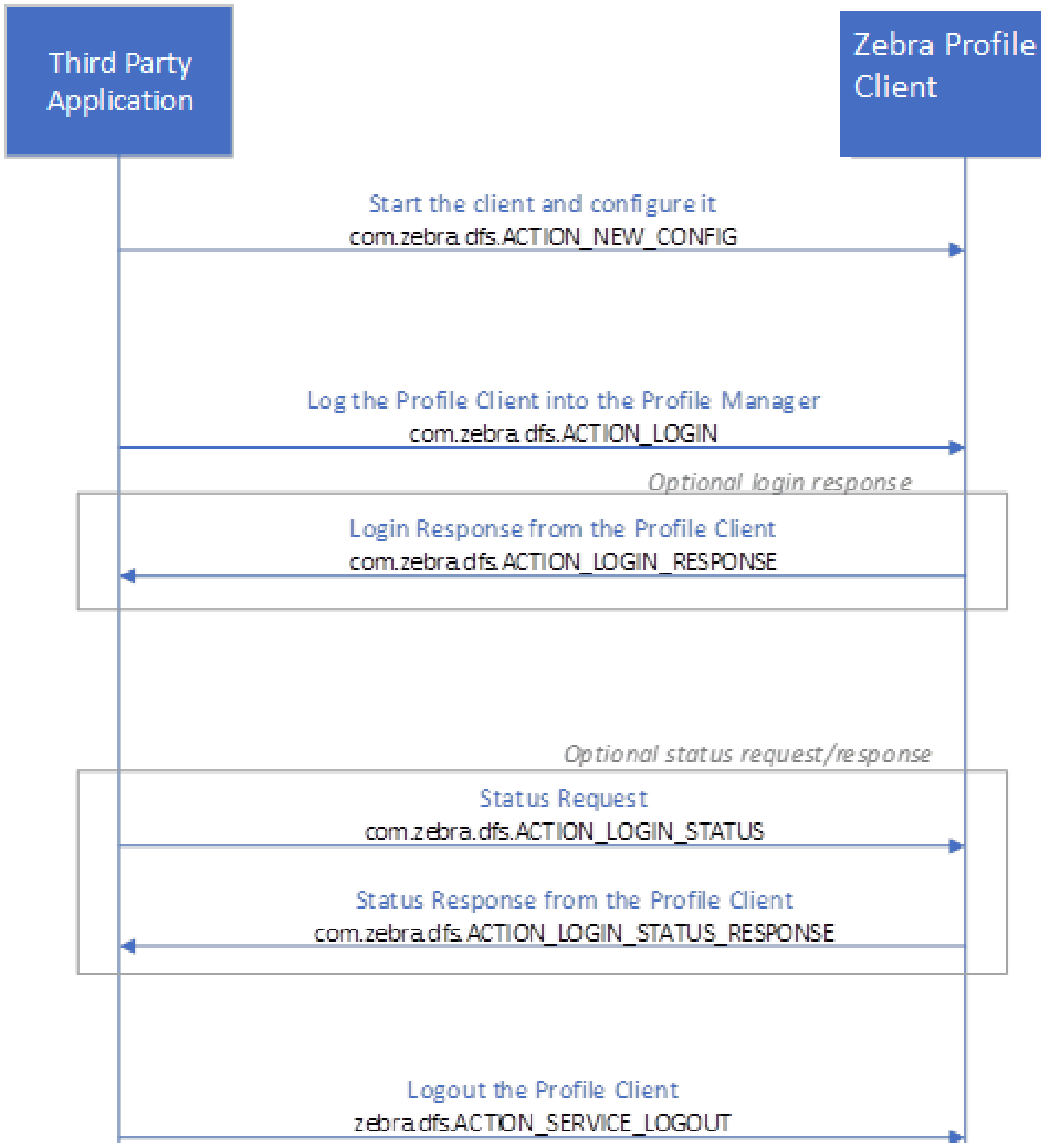
NOTE:

- The `ACTION_LOGIN_STATUS` is not intended to be used while waiting for an ACTION_LOGIN_RESPONSE. The returned status may indicate that the Profile Client is logged in while the login procedure is still in process.

- It is recommended to send all broadcast intents with `package` set to `com.zebra.dfs` as a security measure.
- ACTION_SERVICE_LOGOUT logs out the Profile Client.

The following diagram illustrates the flow of intents.

Figure 1 Profile Client Inter Overview


Third Party Application
Zebra Profile Client
Start the client and configure it
com.zebra.dfs.ACTION_NEW_CONFIG
Log the Profile Client into the Profile Manager
com.zebra.dfs.ACTION_LOGIN
Optional login response
Login Response from the Profile Client
com.zebra.dfs.ACTION_LOGIN_RESPONSE
Optional status request/response
Status Request
com.zebra.dfs.ACTION_LOGIN_STATUS
Status Response from the Profile Client
com.zebra.dfs.ACTION_LOGIN_STATUS_RESPONSE
Logout the Profile Client
zebra.dfs.ACTION_SERVICE_LOGOUT

Starting the Profile Client without Configuration

The startActivity intent is used to start or restart the Profile Client without configuration.

Prerequisites

- The minimum required Profile Client Android version is 2.0.19234

Intent Definition

Name	Description
Component	`com.zebra.dfs/.LoginActivity`
Intent Type	`startActivity`

NOTE:

If this intent is delivered to a currently running client, the client restarts. The client retains the existing configuration, if any. Also, the user is logged out of the Zebra PTT Pro and Voice applications and returned to the PFM Sign-in window.

Starting the Profile Client with Configuration

The startActivity intent starts the Profile Client with configuration. Sending this intent overwrites the Profile Client existing configuration with the parameters in the new configuration. Valid configuration parameters are defined in Profile Client Configuration on page 8.

Use one of the following two methods to configure the Profile Client. Both methods support the same parameters.

Configuration via a JSON File

Load the configuration file using the `profile_uri` , which is defined in the Intent Definition table. The `profile_uri` is a part of the Extra field, The file must be located on the device and the path/name specified in the intent unless using the following default file path/name:

- /enterprise/device/settings/WFCDFSConfig.json for WPC Version 2.0.21201 and above.
- /sdcard/WFCDFSConfig.json for all other versions.

While sending the intent, always set the package `-p com.zebra.dfs` for WPC Version 2.0.23300 and above.

Intent:

```
adb shell am start -a com.zebra.dfs.ACTION_NEW_CONFIG -p com.zebra.dfs --es
 profile_uri /enterprise/device/settings/WFCDFSConfig.json
```

Configuration via a JSON structure

To send the configuration information, use the `config_profile` , which is defined in the Intent Definition table.

Prerequisites

- When configuring through a file, ensure that the JSON file is downloaded to the device.
- The minimum required Profile Client Android version is 2.0. 19234.
- When configuring through a JSON structure, the minimum required Profile Client Android version is 2.0.20205.

Intent Definition

Name	Description
Action	`com.zebra.dfs.ACTION_NEW_CONFIG`

Name	Description
Intent Type	`startActivity`
Extra 0	This extra specifies the name of a configuration file. Either Extra 1 or Extra 0 must be present, but not both.
Type	String
Name	`profile_uri`
Value	Path/name of the configuration file.
Extra 1	This extra specifies a set of configuration parameters. Either Extra 1 or Extra 0 must be present, but not both.
Type	String
Name	`config_profile`
Value	JSON string

NOTE:

1. When a configuration file is used, once the file is successfully ingested, the JSON file is deleted from the folder.
2. If the client is running, the user is logged out of the PTT Pro and Voice applications returned to the PFM Sign-in screen, and the Profile Client re-starts.

Logging a User into Profile Client

This intent is used to log in a user to the Profile Client. Login information is sent in this intent, including parameters that indicate if a response is desired.

There are two methods to log in through intent:

- Include `user_name` and `user_pwd` in the `login_info` JSON parameter, or specify them as separate extra strings. With this method, the IDP provides the access code, refresh token, and refresh token expiration information. The access code manages the Profile Client and logs out the user when the refresh token expires. This method of login requires a previous configuration of the key fields (e.g. `key_user_name, key_user_pwd`) through the `com.zebra.dfs.ACTION_NEW_CONFIG` intent.
- Include `user_name` and `user_accesscode` in the `login_info` JSON parameter. With this method, the key fields are irrelevant. The third-party application is responsible for refreshing the access code when necessary. The `refresh_token_expiration` field in `login_info` is optional:
  - `refresh_token_expiration` specified as non-zero: If a new access code is not sent by the refresh token expiration time using the `com.zebra.dfs.ACTION_LOGIN` intent, the Profile Client logs out.
  - `refresh_token_expiration` specified as 0 or if not specified: Profile Client is not automatically logged out.

Access tokens (`user_accesscode`) take precedence over the `user_name` and `user_pwd`.

A third method to log in without intent is described in Additional Support for Third-Party Launcher on page 26.

Prerequisites

- The user is logged out of the Profile Client.

- The minimum required Profile Client Android version is 2.0.20205.

NOTE: The `request_id` and `package` parameters are supported in 2.0.21200 and later.

Intent Definition

Name	Description
Action	`com.zebra.dfs.ACTION_LOGIN`
Intent Type	`broadcast`
Extra 0	This extra specifies the username. This is a mandatory parameter if the `login_info` parameter is not included.
Type	String
Name	`user_name`
Value	Username with domain (i.e. `sample.user@domain`)
Extra 1	This extra specifies the user password. This is a mandatory parameter if the `login_info` parameter is not included.
Type	String
Name	`user_pwd`
Value	User password
Extra 2	This extra specifies if the JSON login information is encrypted. This is an optional parameter.
Type	Boolean
Name	`json_encrypted`
Value	True indicates the login information is encrypted. False or absent indicates the information is not encrypted.
Extra 3	This extra `login_info` specifies the JSON login information. See below for the payload definition. This extra may or may not be encrypted.
Extra 3 - sent unencrypted	This extra specifies the JSON login information when not encrypted. This is an optional parameter.
Type	String
Name	`login_info`
Value	String in JSON format
Extra 3 – sent encrypted	This extra specifies the JSON login information when encrypted. This is an optional parameter.
Type	Byte array
Name	`login_info`
Value	See code snippet

ABD Examples

Sending user_name and user_pwd as separate extras:

```
adb shell am broadcast -a com.zebra.dfs.ACTION_LOGIN --es user_name --es
```

```
user_pwd --es <other login parameters>
```

Sending login_info unencrypted:

```
adb shell am broadcast -a com.zebra.dfs.ACTION_LOGIN –es json_encrypted false
--es login_info {user_name:sample.user@domain, user_pwd:<password>}
```

login_info payload

The `login_info` parameter contains JSON formatted data with the following fields:

```
{
"user_name": "username",          //mandatory
"user_pwd": "userpwd",           //mandatory
"user_accesscode": "xxxxxxxx",  //optional; access token from IDP.
"site_id": "xxxxxx",            //optional; client uses most recent-
                               // site_id received.
"refresh_token": "xxxxx",      //optional; specifies-
                               //refresh token.
"refresh_token_expiration": "xxxxxx",        //optional; specifies expiration
                                            // of refresh token.
"request_id": <unique id for the request>,      //optional; required if a
                                                //response is desired.
"package": "<third_party_package_name>"        //optional; required if a
                                               //response is desired.
}
```

Encyrption/Decryption

Encryption and decryption of the `login_info` parameter is accomplished using the Google Tink Library at //github.com/google/tink. Following is a sample code to accomplish encryption and decryption.

```
String plainText = "This is a plain text which needs to be encrypted!";
String aad = "These are additional authenticated data (optional)";
String secret_key = "5ecded3e-9562-11ea-bb37-0242ac130002"; // UUID

// Encryption
AesGcmJce agjEncryption = new AesGcmJce(secret_key.getBytes());
byte[] encrypted = agjEncryption.encrypt(plainText.getBytes(),
 aad.getBytes());

// Decryption
AesGcmJce agjDecryption = new AesGcmJce(secret_key.getBytes());
byte[] decrypted = agjDecryption.decrypt(encrypted, aad.getBytes());
```

The `secret_key` must be configured to encrypt/decrypt the `login_info`. If the Profile Client does not have the `secret_key` or `json_encrypted=false`, it does not decrypt the information.

Login Response

This intent is sent from the Profile Client to the third-party application in response to a Login Request.

Prerequisites

- The `request_id` and `package` parameters were received in the Login Request `login_info` parameter.
- The minimum required Profile Client Android version is 2.0.21200.

Intent Definition

Name	Description
Action	`com.zebra.dfs.ACTION_LOGIN_RESPONSE`
Intent Type	`broadcast`
Extra 0	This extra `login_info_response` contains the login response information. This is a mandatory parameter. See below for the payload definition. This extra may or may not be encrypted.
Extra 0 - sent unencrypted	This extra specifies the JSON login information when not encrypted. This is an optional parameter.
Type	String
Name	`login_info_response`
Value	String in JSON format
Extra 0 – sent encrypted	This extra specifies the JSON login information when encrypted. This is an optional parameter.
Type	Byte array
Name	`login_info_response`
Value	See the code snippet under Encryption/Decryption

login_info_response payload

The `login_info` parameter contains JSON formatted data with the following fields:

```
{
"user_name": "username",    //mandatory
"status": "LOGGED_IN" or “LOGGED_OUT”, // mandatory
"selected_roles":"[{"name":"qapfm_v2","description":"qapfm_v2","is_swap_role":true}]
"user_selected_roles":"[{"name":"qapfm_v2","description":"qapfm_v2",
"is_swap_role":true},{"name":"voice","description":"voice_role",
"is_swap_role":false}]"
}
```

NOTE: The Profile Client encrypts the `login_info_response` payload if the client is configured with a `secret_key`. Otherwise, it is sent in the clear.

Selected Role Response

This intent is sent from the Profile Client to the third-party application in response to a role application when there is a switch role, transfer role, or in case of multi-device login.

Prerequisites

- The request_id and package parameters were received in the login_info parameter of the Login Request.
- The minimum required Profile Client Android version is 2.0.23100.

Intent Definition

Name	Description
Action	`com.zebra.dfs.ACTION_SELECTED_ROLES_RESPONSE`
Intent Type	`broadcast`
Extra 0	This extra selected_roles_response contains selected role information. This is a mandatory parameter. Go to the payload definition. This extra may or may not be encrypted.
Extra 0 - sent unencrypted	This extra specifies the JSON selected role information when not encrypted. This is an optional parameter.
Type	String
Name	selected_roles_response
Value	String in JSON format
Extra 0 - sent encrypted	This extra specifies the JSON selected role information when encrypted. This is an optional parameter.
Type	Byte array
Name	selected_roles_response
Value	Go to the code snippet under Encryption/Decryption.

login_info_response payload

The login_info parameter contains JSON formatted data with the following fields:

```
{
"user_name": "username",     //mandatory
"status": "LOGGED_IN" or “LOGGED_OUT”, // mandatory
"selected_roles":"[{"name":"qapfm_v2","description":"qapfm_v2","is_swap_role":true}]
"user_selected_roles":"[{"name":"qapfm_v2","description":"qapfm_v2",
"is_swap_role":true},{"name":"voice","description":"voice_role",
"is_swap_role":false]"
            }
```

NOTE: The Profile Client encrypts the selected_roles_response payload if the client is configured with a secret_key. Otherwise, it is sent in the clear.

Logging a User in Profile Client on Proxy URL Availability

The intent is sent, when Proxy Profile Manager URL is available during login. First, it moves the user to the respective department through Proxy Profile Manager server. If the user is successfully moved, then only it proceeds for login. The parameters are same with login intent. The site_id decides which Profile Manager instance to be moved. Once the response is received, sfs_url and customer_id may be changed. For more information, refer to Profile Manager Proxy Deployment Guide.

Logging a User Out of the Profile Client

This intent is used to log a user out from the Profile Client.

Prerequisites

- The user is logged into the Profile Client.
- The minimum required Profile Client Android version is 2.0.19234.

Intent Definition

Name	Description
Action	`zebra.dfs.ACTION_SERVICE_LOGOUT`
Intent Type	`start`
Extra 0	This extra specifies the exit parameter. This is an optional parameter that minimize the client after logout.
Type	String
Name	Exit
Value	Boolean

ADB Example

```
adb shell am start -a com.zebra.dfs.ACTION_SERVICE_LOGOUT --es Exit true
```

NOTE:

- With the extra string Exit set to `true`, the Profile Client logs out and the UI minimizes and the service is still running.
- With no extra string Exit, or if Exit is set to `false`, the Profile Client logs out and the UI does not minimize and the service is still running.

Status Request

This intent is used to request a login status update from the Profile Client.

Prerequisites

- The Profile Client is running.
- The minimum required Profile Client Android version is 2.0.21200.

Intent Definition

Name	Description
Action	`com.zebra.dfs.ACTION_LOGIN_STATUS`
Intent Type	`broadcast`
Extra 0	This extra specifies if the JSON status request information is encrypted. This is an optional parameter.
Type	Boolean
Name	`json_encrypted`
Value	`True` indicates the status request information is encrypted. `False` or absent means the information is not encrypted.
Extra 1	This extra `login_info_status` specifies information necessary to query the client's login status. See the payload definition below. This extra may or may not be encrypted.
Extra 1 – sent encrypted	This extra specifies the status request information when not encrypted. This is an optional parameter.
Type	String
Name	`login_info_status`
Value	String in JSON format
Extra 1 – sent encrypted	This extra specifies the status request information when encrypted. This is an optional parameter.
Type	Byte array
Name	`login_info_status`
Value	See code snippet under Encryption/Decryption

ADB Examples

login_info_status Payload

The `login_info_status` parameter contains JSON formatted data with the following fields:

```
{
 "user_name": "username",                 //mandatory.
 "request_id": <unique_number>,           //mandatory
 "package" : "<third_party_package_name>"  //mandatory
}
```

NOTE: If `user_name` does not match with the current logged in user by ignoring the upper/lower case and domain name, Profile Client sends `ERROR_MISMATCH_USERNAME`.

Status Request Response

The Profile Client sends this intent in response to a status request (Profile Client originated) from the third-party application.

Prerequisites

- The Profile Client has received the `com.zebra.dfs.ACTION_LOGIN_STATUS` intent.
- The minimum required Profile Client Android version is 2.0.21200.

Intent Definition

Name	Description
Action	`com.zebra.dfs. ACTION_LOGIN_STATUS_RESPONSE`
Intent Type	`broadcast`
Extra 0	This extra specifies`login_info_status_response` information necessary to query the client's login status. See the payload definition below. This extra may or may not be encrypted.
Extra 0 - sent unencrypted	This extra specifies `login_info_status_response` when not encrypted. This is an optional parameter.
Type	String
Name	`login_info_status_response`
Value	String in JSON format
Extra 0 – sent encrypted	This extra specifies `login_info_status_response` when encrypted. This is an optional parameter.
Type	Byte array
Name	`login_info_status_response`
Value	See the code snippet under Encryption/Decryption

ADB Example

login_info_status_response Payload

The `login_info_status_response` parameter contains JSON formatted data with the following fields:

```
{
   "user_name": "username",            //mandatory
   "request_id": <unique_number>,      //mandatory
   "status" : "<user_login_status>"    //mandatory - LOGGED_IN, LOGGED_OUT or
                                       // ERROR_MISMATCH_USERNAME
}
```

NOTE:

- `LOGGED_IN` indicates that the Profile Client is logged in. It does not indicate that a role is selected or that the Zebra Voice and/or PTT Pro Clients are logged in.

- `ERROR_MISMATCH_USERNAME` is returned when the `user_name` from `login_info_status` It does not match the currently logged-in user by ignoring the upper/ lower case and domain name.
- The Profile Client encrypts the `login_info_status_response` payload if the client is configured with a `secret_key`. Otherwise, it is sent in the clear. Even if the `login_info_status` is sent in the clear, `login_info_status_response` is encrypted and sent if the `secret_key` has been configured.

Enrolling a Device to PFM and ESN When Proxy URL is Available

- To enroll a device, Profile Client must be configured with Proxy URL and API key. Once it is configured, the device can be enrolled with the following command:

```
adb shell am start -a com.zebra.dfs.ACTION_ENROLL_DEVICE --es store
 <storename>
```

- If the store parameter is not passed, then the store is retrieved from the device based on the value of the parameter set for the following parameters:
  - `store_ro_property`: The RO property is retrieved from the device. The default value is dhcp.wlan0.domain
  - `store_regex_pattern`: This regex pattern is applied on RO property to extract four continuous digits. The default value is d{4}.

NOTE: After enrolling, the Profile Client is updated with the correct pfm url and tenant id which is received from the proxy-pfm server.

Table 3 Intent Definition

Name	Description
Action	`adb shell am start -a com.zebra.dfs.ACTION_ENROLL_DEVICE`
Intent Type	`startActivity`
Extra 0	This parameter defines the names of the store where the device is enrolled.
Type	String
Name	`Store`
Value	Store ID/Store Name.

For more information about Proxy URL and API key configuration, refer to Profile Manager Proxy Installation and Deployment Guide.

Suppress Data Consent Dialog

- This intent is used to suppress the data consent dialog in Profile Client.

```
adb shell am start -a com.zebra.dfs/com.zebra.dfs.LoginActivity --ez
 showDisclosure false
```

- If the showDisclosure parameter is not passed, then the Data Consent dialog is not suppressed.

Table 4 Intent Definition

Name	Description
Action	`adb shell am start -a com.zebra.dfs/ com.zebra.dfs.LoginActivity --ez showDisclosure false`
Intent Type	`startActivity`
Extra 0	This parameter defines the names of the stores where the device is enrolled.
Type	Boolean
Name	`ShowDisclosure`
Value	true/false

Downgrading the Profile Client

If we downgrade Profile Client in a signed-in state, it displays the "connection timeout" pop-up. We need to press **Try Again** to launch the page. If `secret_key` is configured earlier, it needs to be re-configured. This is because of security enhancement done for Profile Client.

Additional Support for Third-Party Launcher

Logging in a User to the Profile Client describes two ways to log into the Profile Client using intents.

While these are the preferred methods, a third method that does not use intents allows the customer to keep their existing launcher for the user sign-on process and then pass the user information to the Profile Client for authentication with the Profile Manager.

This method uses the following four tags (described in JSON Configuration Variable for Zebra Profile Client table):

- `Key_user_name`
- `Key_user_pwd`
- `Key_submit`
- `Key_domain`

The values in these tags identify the input fields to automate the login process.

Following is a sample HTML of a sample login screen:

```
<div class="input-row"
<table>
  <tr>
    <td>
       <p> <label style="margin-top:-14px"  for="username">Login ID:</
label></p>
    </td>
    <td>
        <span class="ctrl">
           <input type="text" name="username" id="username" class="textinput"
 value=""/>
        </span>
    </td>
  </tr>
</table>
</div>
<div class="input-row">
<table>
  <tr>
    <td>
     <p> <label style="margin-top:-14px" for="password">Password:</label></p>
    </td>
    <td>
       <span class="ctrl">
        <input type="password" name="password" id="password"
 class="textinput" autocomplete="off"/>
       </span>
    </td>
   </tr>
</table>
</div>

<div class="button-row">
<span class="ctrl">
<input type="submit" value="Login" id="submit" class="formButton"
 onclick="this.disabled=true;document.body.style.cursor = 'wait';
 this.className='formButton-disabled';form.submit();return false;"/>
```

The highlighted fields in the HTML example are the content of the login screen sent to the mobile device from the customer's authentication system. Once authenticated, the Profile Client receives the credentials through an intent from the third-party launcher application and passes the credentials to the appropriate tagged fields. The Profile Client receives the input from the launcher through intent and then provides the input to the User ID = id="username", and Password = id="password" entries. These fields are returned to the authorizing system with id=" submit."

By correctly identifying the HTML entry ID Fields, the third-party application can pass the credentials to the Profile Client to log in with these credentials.

- `Key_user_name`: "`username`"
- `Key_user_pwd`: "`password`"
- `Key_submit`: "submit"

NOTE: The domain prefix is not shown in this example.

Third-party launchers can send an access token (`user_accesscode`), refresh token (`refresh_token`) and the refresh token expiration time (`refresh_token_expiration`) in seconds. Access tokens take precedence over the user name and password fields.

When the `refresh_token_expiration` time elapses, and the user is signed out. Third-party launchers must send a new intent with the refreshed `user_accesscode`, `refresh_token`, and `refresh_token_expiration` before the expiration time of the previous toke elapses.

Sample Project

A sample source code project showcasing some of the APIs and configuration methods described in this document are available upon request.