

# **Standard Operating Procedure for OTA card profile setup and provisioning management**

**Version Control**

Document Name	Version	Date
SOP for OTA card profile setup and provisioning management	V1.0	12-08-2022

<b>Prepared by</b>	Technical Support Services	12-08-2022
<b>Approved by</b>	Saji Krishnan	<i>Saji Krishnan</i>

**Revision History**

Issue	Date	Reason for change
V1.0	11-08-2022	First Release

## Introduction

This document outlines the procedure for OTA card profile setup, provisioning and management. It pertains to OTA projects in which Kigen OTA platform has been deployed.

## Pre-requisites

- Commercial approval received from Workz finance department
- Profile for OTA campaign card profile has been agreed with the customer

## Procedure

1. Once the OTA platform software has been deployed by Workz IT team, they will share the credentials with TSS.  
The credentials will include IP address of the OTA platform and password.
2. For managing bulk campaigns, CLI interface of the OTA platform shall be used.
3. The documentation for CLI interface and commands is "Workz\_OTAPPlatform\_0120\_CLI" and placed in Onedrive at below path:  
01\_OTA\_Platform\_Documentation\_from\_ARM
4. Connect to the OTA platform using any terminal emulator, ex: Putty.
5. The high-level steps to preparing and executing an OTA campaign are listed below:
  - a. Create Profile
  - b. Provision Cards
  - c. Create Group
  - d. Create Campaign
  - e. Start Campaign
  - f. Retry Failed Campaign

Each of the above steps are explained in detail further in this document.

6. Create Profile -  
Creates a new profile, which can optionally include an applet. This applet can be the default Remote File Management (RFM) application, the default Remote Applet Management (RAM) application, or a custom application.

### Syntax:

```
{./cliota.sh | cliota.bat} -createProfile -name profile_name  
[-appType {rfm | ram | custom} -appName applet_name  
-tarList tar1 tar2 ... tarN]
```

### Arguments:

*name*

Mandatory. Specifies the name of the new profile. The name must be unique and must not contain spaces.

*appType*

Optional. Specifies the type of application to include with the profile. The available options are:

*rfm*: Include the default RFM applet (TAR: 0xB00010).

*ram*: Include the default RAM applet (TAR: 0x000000).

*custom*: Include a custom applet, the filename of which is specified by the *appName* argument.

*appName*

Conditional. For profiles with custom applets, specifies the name of the application. This value does not have to be the same as the filename of the applet, but it must end with *.cap* or *.ijc*.

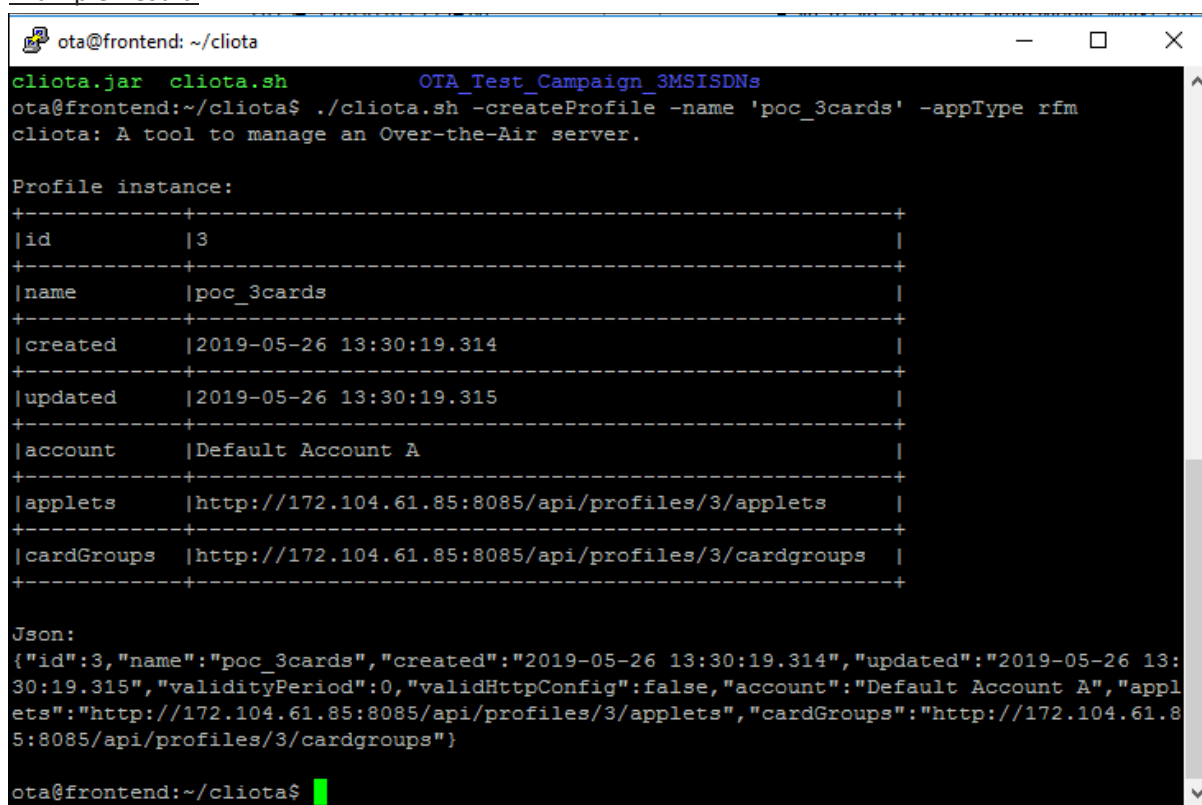
*tarList*

Conditional. For profiles with custom applets, specifies a list of TAR values for the application.

Example:

```
./cliota.sh -createProfile -name 'poc_3cards' -appType rfm
```

Example Result:



```
ota@frontend: ~/cliota
cliota.jar cliota.sh OTA_Test_Campaign_3MSISDNs
ota@frontend:~/cliota$ ./cliota.sh -createProfile -name 'poc_3cards' -appType rfm
cliota: A tool to manage an Over-the-Air server.

Profile instance:
+-----+-----+
|id      |3      |
+-----+-----+
|name    |poc_3cards|
+-----+-----+
|created |2019-05-26 13:30:19.314|
+-----+-----+
|updated |2019-05-26 13:30:19.315|
+-----+-----+
|account |Default Account A|
+-----+-----+
|applets |http://172.104.61.85:8085/api/profiles/3/applets|
+-----+-----+
|cardGroups|http://172.104.61.85:8085/api/profiles/3/cardgroups|
+-----+-----+

Json:
{"id":3,"name":"poc_3cards","created":"2019-05-26 13:30:19.314","updated":"2019-05-26 13:30:19.315","validityPeriod":0,"validHttpConfig":false,"account":"Default Account A","applets":"http://172.104.61.85:8085/api/profiles/3/applets","cardGroups":"http://172.104.61.85:8085/api/profiles/3/cardgroups"}

ota@frontend:~/cliota$
```

## 7. Provision Cards

Uploads to the Over-the-Air Platform database details of cards to manage from a card provisioning data file in CSV format. All cards must be associated with a profile.

Syntax

```
{./cliota.sh | cliota.bat} -provision -outputFilePath
```

`path_to_out_file -profile profile_id [-autoCG true | false]`

### Arguments

#### *outputFilePath*

Mandatory. Specifies the path, relative to the directory in which the CLI is running, to a .out file that contains details of the cards to provision in CSV format. Card data files must not contain:

- Cards that have already been provisioned
- Duplicate cards

#### *profile*

Mandatory. Specifies the ID number of the profile for the cards in the file to use. The ID must be a valid identifier for an existing profile that belongs to the account under which the command is executed.

#### *autoCG*

Optional. Specifies whether to create a new card group and add the cards to the group as they are provisioned. By default, the cards are not added to a new card group. Recommendation - a maximum of 400,000 cards in a single card group.

### Example:

```
./cliota.sh -provision -outputFilePath
'/home/ota/cliota/OTA_Test_Campaign_3MSISDNs/Provisioning_3TestSubs_FriendiKSA.out' -profile 2
```

### Example Result:

The action will continue and show in progress until completed.

### Sample provisioning file:

---

```
VAR_OUT:ICCID,IMSI,MSISDN,KIC,KID
8991234567788901234F,421234567890123,961234567890,12122343423234234234234234234,56756785675678568678678678678
8991234567788901235F,421234567890124,961234567891,23434564756786846345235345645745,23454576679803454767897890890678
```

## 8. Create Group

Creates a new card group that contains either all the cards that use the specified profile or a subset of cards.

To add a subset of cards to the group, you must provide a text file that contains a list of MSISDN or ICCID values for the cards to add. Recommendation - do not add more than 400,000 cards to a single card group.

### Syntax

```
{./cliota.sh | cliota.bat} -createGroup -profile profile_id -name
card_group_name [{-msisdnList path_to_msisdn_list | -iccidList
path_to_iccid_list}]
```

### Arguments

#### *profile*

Mandatory. Specifies the ID number of the profile for which to create the card group. The ID must be a valid identifier for an existing profile that belongs to the account under which the command is executed.

#### *name*

Mandatory. Specifies the name for the new card group. The name must be unique within the profile and must not contain spaces.

#### *msisdnList*

Conditional. Specifies the path, relative to the directory in which the CLI is running, to a plain text file that contains the MSISDNs of the cards to include in the new card group. The file must have the .txt extension and must contain one MSISDN on each line with no other text or formatting. The MSISDNs:

- Must be valid values for existing cards that belong to the account under which the command is executed.
- Must be for cards that are associated with the specified profile
- Must not be duplicated within the file

#### *iccidList*

Conditional. Specifies the path, relative to the directory in which the CLI is running, to a plain text file that contains the ICCIDs of the cards to include in the new card group. The file must have the .txt extension and must contain one ICCID on each line with no other text or formatting. The ICCIDs:

- Must be valid values for existing cards that belong to the account under which the command is executed
- Must be for cards that are associated with the specified profile
- Must not be duplicated within the file

#### Example:

```
./cliota.sh -createGroup -profile 2 -name 'poc_3cards' -msisdnList
'/home/ota/cliota/OTA_Test_Campaign_3MSISDNs/TargetMSISDN_3TestSubs_FriendiKSA.out'
```

#### Example Result:

```
ota@frontend:~/cliota$ ./cliota.sh -createGroup -profile 2 -name 'poc_3cards' -msisdnList '/home/ota/cliota/OTA_Test_Campaign_3MSISDNs/TargetMSISDN_3TestSubs_FriendiKSA.out'
cliota: A tool to manage an Over-the-Air server.

Card Group instance:
+-----+-----+
|id      |3      |
+-----+-----+
|name    |poc_3cards|
+-----+-----+
|profile |http://172.104.61.85:8085/api/profiles/2|
+-----+-----+
|cards   |http://172.104.61.85:8085/api/profiles/2/cardgroups/3/cards|
+-----+-----+

Json:
{"id":3,"name":"poc_3cards","created":"2019-05-26 14:00:38.0","updated":"2019-05-26 14:00:38.0","profile":
"http://172.104.61.85:8085/api/profiles/2","profile_id":2,"profile_name":"workz_virgin","cards":"http://17
2.104.61.85:8085/api/profiles/2/cardgroups/3/cards"}

ota@frontend:~/cliota$
```

#### Sample group file:

```
961234567890
961234567890
```

## 9. Create Campaign

Creates a new campaign. You can specify the settings for each campaign or create a campaign

from an existing template.

The URLs to which you want DLRs, PoRs, and TLS campaign responses sent are set in the CLI configuration file, `cliota.properties`. If set, these URLs are automatically added to the campaign that you create.

Recommendation - limit your campaigns to a maximum of 1,000,000 messages. For example, if you create a campaign that targets 250,000 cards, restrict the campaign to a maximum of four messages per card.

### Syntax

```

{./cliota.sh | cliota.bat} -createCampaign {-template template_id |
-type {RFM | TLS_TRIGGER_RFM | SMS | FLASH | RAM | TLS_TRIGGER_RAM}
-profile profile_id [-applet applet_id -tkp toolkit_parameter_id]
-name campaign_name -spi spi_value -tar tar_value -kicByte kic_value
-kidByte kid_value [-transmissionMode {seq | conc}] [-apduFilePath
path_to_apdu_file] [-message message_text] [-validityPeriod
time_in_minutes] [-includeSimRefresh {true | false}]
[-expectedLastSW status_word] [-isLoadIncluded {true | false}]
[-isInstallIncluded {true | false}] [-isDeleteIncluded
{true | false}]}

```

### Arguments

#### *template*

Conditional. Specifies the ID number of the template from which to create the campaign. The ID must be a valid identifier for an existing campaign template that belongs to the account under which the command is executed.

#### *type*

Mandatory. Specifies the campaign type. The available options are:

**RFM:** Create an RFM campaign, sending commands to the target cards by SMS.

**TLS\_TRIGGER\_RFM:** Create an RFM campaign, sending commands to the target cards over HTTPS. This type of campaign can only be used if the specified profile includes HTTP parameters.

**SMS:** Create a campaign to send an SMS message to the target cards.

**FLASH:** Create a campaign to send a flash message to the target cards. Flash messages are only displayed for a short time and are not stored on the card.

**RAM:** Create an RAM campaign, sending commands to the target cards by SMS.

**TLS\_TRIGGER\_RAM:** Create an RAM campaign, sending commands to the target cards over HTTPS. This type of campaign can only be used if the specified profile includes HTTP parameters.

#### *profile*

Mandatory. Specifies the ID number of the profile for which to create the campaign. The ID must be a valid identifier for an existing profile that belongs to the account under which the command is executed.

#### *applet*

Conditional. For RAM campaigns, specifies the application ID of an existing applet to use to automatically generate campaign commands. The applet must be associated with the specified profile.

### *tkp*

Conditional. For RAM campaigns, specifies the ID of an existing toolkit parameter to use to automatically generate campaign commands. The toolkit parameter must be associated with the specified profile.

### *name*

Mandatory. Specifies the name for the new campaign. The name must be unique within the profile and must not contain spaces.

### *spi*

Conditional. For RFM and RAM campaigns, specifies the Security Parameters Indication (SPI) value for the campaign.

### *tar*

Conditional. For RFM and RAM campaigns, specifies the TAR that is targeted by the campaign. The default value is 0xB00010 (default RFM applet).

### *kicByte*

Conditional. For RFM and RAM campaigns, specifies the KIC byte of the cards that are targeted by the campaign. The default value is 15.

### *kidByte*

Conditional. For RFM and RAM campaigns, specifies the KID byte of the cards that are targeted by the campaign. The default value is 15.

### *transmissionMode*

Conditional. For RFM and RAM campaigns, specifies how concatenated messages are transmitted. The available options are:

*seq*: Message parts are sent sequentially, that is, one at a time. A successful DLR for the previous message part must be received before the next message part can be sent.

*conc*: Message parts are sent concurrently, that is, all parts are sent at the same time.

### *apduFilePath*

Conditional. For RFM and RAM campaigns, specifies the path to a file that contains a list of APDU commands for execution by the campaign. The path must be specified relative to the directory in which the CLI is running. Omit this argument if automatically generating commands.

The file must contain one command on each line with no other text or formatting, unless the campaign is an RFM campaign with placeholder data. In this case, the commands can contain placeholders that are composed of the characters a–z, A–Z, 0–9, and \_ (underscore) enclosed by square brackets [ ]. For example, A0000009[Placeholder\_1]. The placeholders are replaced with data specific to each individual card when the campaign is executed.

### *message*

Conditional. For SMS and flash campaigns, specifies the SMS or flash message to send to the target cards.

### *validityPeriod*

Conditional. For campaigns where commands are sent by SMS, specifies the time in minutes for which the campaign must remain active on the SMSC. The default value is zero, which indicates that the validity period is not specified in the campaign and is inherited from elsewhere.

### *includeSimRefresh*

Conditional. For RFM campaigns, specifies whether the card is refreshed after the update is



applied. By default, the card is not refreshed.

*expectedLastSW*

Conditional. For RFM and RAM campaigns, specifies the last PoR status word that must be returned from the card before the campaign is marked complete. If provided, this value must match the status word that is returned from the card otherwise the campaign cannot be completed for that card. By default, no specific last status word is expected.

*isInstallIncluded*

Conditional. For RAM campaigns, specifies whether the applet that is attached to the campaign must be installed on the target cards. By default, the applet is not installed.

*isLoadIncluded*

Conditional. For RAM campaigns, specifies whether the applet that is attached to the campaign must be downloaded onto the target cards. By default, the applet is not downloaded.

*isDeleteIncluded*

Conditional. For RAM campaigns, specifies whether the applet that is attached to the campaign must be deleted from the target cards. By default, the applet is not deleted.

#### Example:

```
./cliota.sh -createCampaign -type RFM -profile 2 -name 'poc_3cards_campaign' -spi 1621 -tar 'B00010' -
transmissionMode conc -apduFilePath '/home/ota/cliota/OTA_Test_Campaign_3MSISDNs/Commands.txt'
```

#### Example Result:

```
Json:
{"id":4,"name":"poc_3cards_campaign","start":"2019-05-26 14:09:25.913","terminate":"2019-05-26 14:09:25.91
3","state":"WAITING","type":"RFM","created":"2019-05-26 14:09:26.517","updated":"2019-05-26 14:09:26.517",
"iteration":0,"validityPeriod":0,"spi":"1621","tar":"B00010","kicByte":"15","kidByte":"15","commands":["A0
A40000027F20","A0A40000026F46","A0D60000110056697267696E204D6F62696C65FFFFFF","A0A40000026FC5","A0DC010422
430D85D6B4FC9C76839A6F719A5D06FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF",
"D009810301010482028102"],"placeholders":null,"includeSIMRefresh":false,"sequentialTransmission":false,"profile":"http://172.104.61.85:8085/ap
i/profiles/2","profile_id":2,"profile_name":"workz_virgin","messages":"http://172.104.61.85:8085/api/campa
igns/4/messages","transmissionMode":"CONCURRENT"}
```

#### Example of Commands.txt file:

```
A0A40000027F20
A0A40000026F46
A0D60000110056697267696E204D6F62696C65FFFFFF
A0A40000026FC5
A0DC010422430D85D6B4FC9C76839A6F719A5D06FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
D009810301010482028102
```

## 10. Start Campaign

Executes an existing campaign or creates and starts a new campaign from an existing template.

Starting a campaign moves the campaign to the Running state and initiates message generation and transmission for the target cards.

To start a campaign, you must identify the cards that you want to target by specifying either an existing card group or card, or by provisioning a new card as part of the start command.

Recommendation - limit your campaigns to a maximum of 1,000,000 messages. For example, if you create a campaign that sends four messages to each card, restrict the campaign to a maximum of 250,000 cards.

## Syntax

```

{./cliota.sh | cliota.bat} -start {-campaign campaign_id | -template
template_id} [-cardGroup card_group_id | -card card_id] [-msisdn
msisdn_number] [-iccid iccid_number] [-imsi imsi_number -keyIndex
key_index_value -kicKey kic_key_value -kidKey kid_key_value]
[-dataFilePath path_to_map_file]

```

## Arguments

### *campaign*

Conditional. Specifies the ID number of the campaign to start. The ID must be a valid identifier for an existing campaign that is in the Waiting state and that belongs to the account under which the command is executed.

### *template*

Conditional. Specifies the ID number of the template from which to create the new campaign. The ID must be a valid identifier for an existing campaign template that belongs to the account under which the command is executed.

### *cardGroup*

Conditional. Specifies the ID number of the card group to target with the campaign. The ID must be a valid identifier for an existing card group that is associated with the same profile as the specified campaign or campaign template.

### *card*

Conditional. Specifies the ID number of the card to target with the campaign. The ID must be a valid identifier for an existing card that uses the same profile as the specified campaign or campaign template.

### *msisdn*

Conditional. Specifies the MSISDN number of the card to target with the campaign. For an existing card, the MSISDN must be a valid value for a card that uses the same profile as the specified campaign or campaign template. Where a new card is being provisioned as part of the start command, the MSISDN must be a unique value that does not match any existing MSISDNs in the Over-the-Air Platform database.

### *iccid*

Conditional. Specifies the ICCID number of the card to target with the campaign. For an existing card, the ICCID must be a valid value for a card that uses the same profile as the specified campaign or campaign template. Where a new card is being provisioned as part of the start command, the ICCID must be a unique value that does not match any existing ICCIDs in the Over-the-Air Platform database.

### *imsi*

Conditional. Specifies the IMSI number of the card to target with the campaign. For an existing card, the IMSI must be a valid value for a card that uses the same profile as the specified campaign or campaign template. Where a new card is being provisioned as part of the start command, the IMSI must be a unique value that does not match any existing IMSIs in the Over-the-Air Platform database.

### *keyIndex*

Conditional. Where a new card is being provisioned as part of the start command, specifies the key index number to which to add the specified KIC and KID for the new card. The default

value is one.

#### *kicKey*

Conditional. Where a new card is being provisioned as part of the start command, specifies the KIC to create for the new card. This value must match the configuration that is given by the spi and kicByte values of the specified campaign template. The key is only created if the card is not already present in the Over-the-Air Platform database. Otherwise, the value that is specified for this argument is discarded and the existing key is used.

#### *kidKey*

Conditional. Where a new card is being provisioned as part of the start command, specifies the KID to create for the new card. This value must match the configuration that is given by the spi and kidByte values of the specified campaign template. The key is only created if the card is not already present in the Over-the-Air Platform database. Otherwise, the value that is specified for this argument is discarded and the existing key is used.

#### *dataFilePath*

Conditional. For campaigns and campaign templates that include commands with placeholder values, specifies the path to a file that maps the placeholder value to specific values for each card. The path must be specified relative to the directory in which the CLI is running. The file must have the .txt extension and must contain data in CSV format, with each card on a separate line.

#### Example:

```
./cliota.sh -start -campaign 4 -cardGroup 3
```

Once a bulk campaign is started, you will see multiple SMSPPs being sent from the platform towards targeted MSSIDs in “bearerbox.acc” file.

This file is present at path ‘Kannel/logs’ in OTA platform.

The targeted cards will send back PORs back to the platform.

It is possible that not all PORs reach successfully back the platform.

Usually, for such scenarios, it is recommended to do 3 retries on every batch to ensure maximum success rate.

## 11. Retry Failed Campaign

Creates and starts a new iteration of an existing campaign that is in the Completed or Terminated state using the same settings. However, the new iteration includes only those cards that were not updated by the original campaign.

The retryFailed command regenerates all the campaign messages but only retransmits them to the target cards that did not return a successful delivery report for the original campaign.

Retrying only the failed cards allows you to quickly repeat a campaign without having to identify the cards that were not updated and create a new card group for them.

#### Syntax

```
{./cliota.sh | cliota.bat} -retryFailed -campaign campaign_id
```

#### Arguments

*campaign*

Mandatory. Specifies the ID number of the campaign to retry. The ID must be a valid identifier for an existing campaign that is in the Completed or Terminated state and that belongs to the account under which the command is executed.

<b>TITLE</b>	SOP_for_OTA_card_profile_setup_provisioning_management_v1.0
<b>FILE NAME</b>	SOP_for_OTA_card_...agement_v1.0.docx
<b>DOCUMENT ID</b>	3573de4e8033670aa63e848810179715e64ac73d
<b>AUDIT TRAIL DATE FORMAT</b>	DD / MM / YYYY
<b>STATUS</b>	● Signed

## Document history



SENT

**05 / 09 / 2022**

10:04:00 UTC+4

Sent for signature to Saji Krishnan  
(saji.krishnan@workz.com) from tss@workz.com  
IP: 94.200.229.6



VIEWED

**05 / 09 / 2022**

10:58:33 UTC+4

Viewed by Saji Krishnan (saji.krishnan@workz.com)  
IP: 86.99.233.49



SIGNED

**05 / 09 / 2022**

10:59:40 UTC+4

Signed by Saji Krishnan (saji.krishnan@workz.com)  
IP: 86.99.233.49



COMPLETED

**05 / 09 / 2022**

10:59:40 UTC+4

The document has been completed.