Copyright IBM Corporation 2006, 2016

Customer and Network Analytics 9.0

Provisioning Guide

Copyright IBM Corporation 2006, 2016

Contents

Copyright IBM Corporation 2006, 2016

1 Overview

1.1 Intended Audience

Operations team working on customer site.

1.2 Document History

Date	Author	Version	CNA Version	Notes
27/12/2015	Georgy Turevich	0.1	9.0	First draft of Provisioning Ops Guide
11/01/2016	Georgy Turevich	0.2	9.0	Corrections and answers after Aubrey and Ivan reviews
20/01/2016	Georgy Turevich	0.3	9.0	Removed reference to Provision-And-Data-Loader-Tools-Guide.pdf from "2.1 Concepts" chapter and "4.2 Subscriber Provisioning" chapter from appendix as it is not Actual for 9.0
27/01/2016	Owen Lynch	0.5	9.0	Added info on export tool, and cleaned up Concepts Chapter.
28/01/2016	Owen Lynch	0.6	9.0	Removed all completed comments

1.3 Document Approval

Role	Name	Approval Date
Document Owner		

1.4 Related Documents

Document	Description
Provision-And-Data-Loader-Tools-Guide.pdf	Available on Jenkins Job http://9.162.178.97:8080/view/CNA%209.0/view/Framework/job/BIS-3.0-branch/lastSuccessfulBuild/artifact/bis-documentation/target/docbkx/pdf/Provision-And-Data-Loader-Tools-Guide.pdf
IBM Now Factory Rapid Analytics PROVISIONING Functional Spec.doc	Available in IBM Now Factory Internal Documents: https://w3-connections.ibm.com/files/app#/file/d5d6724c-2507-4a46-87d0-c91bfdc625ee
$TNF_BIS_MAIN	Common location of Analytics Directory
$TNF_PROV_TOOL	Location of Analytics Provisioning Tool Utilities (/opt/tnf/apps/bis-main/bis-tools/bis-provisioning-tool/)

1.5 Glossary

Acronyms and abbreviations used in this document are described in the table below.

Abbreviation	Description
Dimension Type	Example: CELL, APPLICATION, APN
Dimension Instance	Entity of Dimension Type like: CELL = M#I0341 APPLICATION = Facebook APN = xxx.operator.com
CNA	Customer and Network Analytics
IOP	IBM Open Platform (for Big Data) – Hadoop

2 General Provisioning

2.1 Concepts

Provisioning of data in CNA 9.0 is performed using a set of command line tools available on the Analytics Platform server (IOP master) of a CAN 9.0 installation. Data is provisioned from a plain text file containing a list of entities – such as Cells, Devices, and Applications.

Each entity, or row in the file, is a *Dimension Instance* in the application. The target table defines the *Dimension Type* (Cell, Devices, Application, etc.)

There is no separate control file to define the dimension type or attributes etc. All dimension tables in CNA 9.0 have a "*dim_*" prefix, and dimension type has a separate table.

When the tool runs and loads data, it is passed a table name (which identifies the dimension type) and a CSV file containing the records (entities).

A Provisioning file might also contain additional information about each dimension, or ***attributes***. For example – Cell is usually identified by Cell ID and LAC information. But it might also have several other ***attributes*** like location (Region, County) or type (BSC / RNC).

These attributes can be subsequently used to create hierarchy of dimension instances as groups. To match geographical locations – Region, County, City information can be used.

Attributes are represented in Analytics Framework as additional columns in Dictionary Tables. Each attribute has the following properties:

- Label – human readable description of attribute
- Alias - case insensitive alias of attribute name. So, you can load or map data from a different heading column in the csv import file, to column in the table.

The header row of the CSV file is parsed, and the columns to insert data in to are mapped from the header. If there is no file matching the column name in the CSV, an alias will be checked for values inserted to the field/column marked by the alias.

Aliases may also be complex – where an attribute is made up of a combination of several input columns of CSV files (see examples for Cells later). *Complex alias* can be "MCC:MNC:LAC:CI" and an import tool will search all columns by the headers and concatenate columns values into result string.

There are two types of Attributes – System or Custom.

- *System Attributes* are attributes which were defined by developers and created by default.
- *Custom attributes* are attributes which were created after initial installation using *attributor.sh* tool (described below)

Note: Provisionings should be performed, and entities loaded into the system before the Streams server starts to process data, as streams utilises auto-exported mapping files. The Streams server correlates data received in TDRs with loaded entities.

2.2 Generic Steps Required To Import Provisioning Information

Analytics Framework Provisioning Tool allows the user to:

- Import, update and delete dimension instances
- Provision additional attributes for dimensions
- Modify existing attribute labels

2.2.1 Viewing Existing Dimensions Information - attributor.sh

To see all defined Dimension Types and related tables – execute following shell command on Analytics master server:

```
$TNF_PROV_TOOL/attributor.sh -l
```

You will see similar output listing the tables and mapped dimensions:

Table	Dimension Type	Is Subscriber	Segment Field	Label
dim_aggregatedcarrierflag	AGGREGATEDCARRIERFLAG			Aggregated Carrier Flag
dim_apn	APN			APN
dim_application	APPLICATION			Applications (Discovered)
dim_application_protocol	APPLICATION_PROTOCOL			Application Protocol
dim_bsc	BSC			BSC
dim_bts	BTS			BTS
dim_cell	CELL			Cell
dim_ces_rule_type	CES_RULE_TYPE_NAME			Customer Experience Score Rule
dim_channelbandwidth	CHANNELBANDWIDTH			Channel Bandwidth
dim_channelcentrefrequency	CHANNELCENTREFREQUENCY			DL Channel Centre Frequency
dim_codec	CODEC			VoLTE RTP Codec
dim_collector	COLLECTOR			Data Collector
dim_cpedevice	CPEDEVICE			CPE Device
dim_device	DEVICE			Device
dim_domain	DOMAIN			Domain
dim_dslam	DSLAM			DSLAM
dim_earfcn	EARFCN			DL EARFCN
dim_enodeb	ENODEB			enode-B
dim_fixedline_cause_code	FIXEDLINE_CAUSE_CODE			Error Codes
dim_fixedline_qos	FIXEDLINE_QOS			Radius Class
dim_fixedline_transaction_type	FIXEDLINE_TRANSACTION_TYPE			Transaction Type
dim_frequencyband	FREQUENCYBAND			Frequency Band
dim_ggsn	GGSN			GGSN
dim_handovertype	HANDOVERTYPE			Handover Type
dim_hplmn	HPLMN			Home PLMN (roamers in)
dim_lte_cause_code	LTE_CAUSE_CODE			4G LTE Session Error Codes
dim_lte_transaction_type	LTE_TRANSACTION_TYPE			4G LTE Session Transaction Types
dim_mme	MME			MME
dim_modulationscheme	MODULATIONSCHEME			Modulation Scheme
dim_msc	MSC			Mobile Switching Center
dim_nodeb	NODEB			node-B
dim_other_party_group	OTHER_PARTY_GROUP			Other Party Group
dim_pdp_status	PDP_STATUS			2G/3G Error Codes
dim_protocol	PROTOCOL			Protocol
dim_qoscategory	QOSCATEGORY			QoS Category
dim_rat	RAT			RAT
dim_realm	REALM			VoLTE Realm
dim_rnc	RNC			RNC
dim_roaming_type	ROAMING_TYPE			Roaming Type
dim_sgsn	SGSN			SGSN
dim_sgw	SGW			SGW
dim_subscriber_fixedline	SUBSCRIBER	yes	csid	Subscriber
dim_subscriber_mobile	SUBSCRIBER	yes	imsi	Subscriber
dim_technology	TECHNOLOGY			Technology
dim_tier0node	TIER0NODE			Tier 0 Node
dim_tier1node	TIER1NODE			Tier 1 Node
dim_transaction_type	TRANSACTION_TYPE			2G/3G Transaction Type
dim_uarfcn	UARFCN			DL UARFCN

```
| dim_user_plane_status        | USER_PLANE_STATUS          |               |               | User Plane Status            |
| dim_voice_status_code        | VOICE_STATUS_CODE          |               |               | 2G/3G Voice/SMS Error Codes  |
| dim_voice_transaction_type   | VOICE_TRANSACTION_TYPE     |               |               | 2G/3G Voice/SMS Transaction  |
|                              |                            |               |               | Types                        |
| dim_volte_direction          | VOLTE_DIRECTION            |               |               | VoLTE MO/MT call direction   |
| dim_vplmn                    | VPLMN                      |               |               | Visited PLMN (roamers out)   |
+------------------------------+----------------------------+---------------+---------------+------------------------------+
```

In order to verify what attributes are defined for Dimension Type – execute following shell command:

```
$TNF_PROV_TOOL/attributor.sh -t <TABLE_NAME>
```

For example – Cell dimension

```
$TNF_PROV_TOOL/attributor.sh -t dim_cell
```

You will see similar output, listing all of the columns:

```
Table: dim_cell, Dimension Type: CELL, Label: null
+------------------+-----------+-------------+-------------+-----------+-------------+------------------+-----------------------------+
| Field            | Type      | Aggregation | Application | Attribute | Dimension   | Label            | Alias                       |
|                  |           | Key         | Key         |           | Description |                  |                             |
+------------------+-----------+-------------+-------------+-----------+-------------+------------------+-----------------------------+
| cell_key         | varchar   | yes         |             |           |             |                  | MCC:MNC:LAC:CI, MCC:MNC:ECI |
| label            | varchar   |             |             | system    | yes         | Cell Name        | LTE CELL NAME, CellSite2G3G_|
|                  |           |             |             |           |             |                  | ID                          |
| mcc              | varchar   |             |             | system    |             | MCC              |                             |
| mnc              | varchar   |             |             | system    |             | MNC              |                             |
| lac              | varchar   |             |             | system    |             | LAC/TA           |                             |
| ci               | varchar   |             |             | system    |             | CI               |                             |
| eci              | varchar   |             |             | system    |             | ECI              |                             |
| geolevel1        | varchar   |             |             | system    |             | GeoLevel1        | LTE CELL REGION, Region     |
| geolevel2        | varchar   |             |             | system    |             | GeoLevel2        | LTE CELL COUNTY, State      |
| geolevel3        | varchar   |             |             | system    |             | GeoLevel3        | LTE CELL CITY, tpim_market  |
| geolevel4        | varchar   |             |             | system    |             | GeoLevel4        | LTE CELL STATE, City        |
| sitename         | varchar   |             |             | system    |             | Site Name        | LTE CELL SITENAME           |
| rat              | varchar   |             |             | system    |             | RAT              | Network                     |
| cell_band        | varchar   |             |             | system    |             | Cell Band        |                             |
| vendor           | varchar   |             |             | system    |             | Vendor           |                             |
| zip              | varchar   |             |             | system    |             | Site ZIP/PostCode| LTE CELL ZIP                |
| latitude         | varchar   |             |             | system    |             | Latitude         |                             |
| longitude        | varchar   |             |             | system    |             | Longitude        |                             |
| azimuth          | varchar   |             |             | system    |             | Azimuth          |                             |
| last_updated_time| timestamp |             |             |           |             |                  |                             |
+------------------+-----------+-------------+-------------+-----------+-------------+------------------+-----------------------------+
```

We can see that System Attributes are marked by “system” keyword in Attribute column and Custom Attributes, if they exist, are marked by “custom” keyword.

2.2.2 General Provisioning File Format

Provisioning import files are text files. Usually, using CSV format (although delimiter can be different). General rule of provisioning file:

- delimited file, either comma or pipe
- Header contains list of fields (column names).
- Each line creates one dimension instance. Line contains values for dimension attributes.

Here is the example of provisioning file that is used for Devices import:

```
#TACFAC|BRAND|LABEL|TECHNOLOGY

44927561|Motorola|Motorola V50|GSM

44896150|Bang & Olufsen|Bang & Olufsen Beocom 9800 Db|GSM
```

Two things can be seen:

- 4 attributes are defined. Their names – TACFAC, BRAND, LABEL, TECHNOLOGY
- Values are separated by pipe character.

Name of these attributes must match column name in related dictionary table or match “alias” property value. Chapter Manage attribute definitions is describing steps needed to manage attributes.

2.2.3 Managing attribute definitions – command file

To add “custom” attributes or managing “system” attributes (changing label or alias) a command CSV file is used.

The command file is executed line by line. Each line describes one column operation. The format of the command file is "CSV" style and can contain 6 columns; 3 mandatory columns: table, operation, column and 3 optional columns: type, label, alias.

The possible operations are **“add”, “update”, “remove”.**

Operation	Parameters
Add	Type is mandatory. Other are optional (label, alias, …)
Update	User must set at least on parameter (type, label, alias, …)
Remove	Without parameters

Attributes added using attributor.sh will be marked as “custom”. For these attributes, attributor tool will show “custom” value in Attribute column. Attributes created by default install, can only update optional parameters (label and alias). These fields will contain “system” value in Attributor column (see example for DIM_CELL above)

First example of csv command file:

```
#table|operation|column|type|label|alias
DIM_SUBSCRIBER_MOBILE|add|ACCOUNT_NUMBER|VARCHAR|Account Number|ACCNUM
DIM_SUBSCRIBER_MOBILE|add|ADDRESS_2|VARCHAR||
DIM_SUBSCRIBER_MOBILE|update|DOB||Date Of Birth|
DIM_SUBSCRIBER_MOBILE|update|HANDSET||Handset|HNDST
```

Explanations of these commands:

Command	Explanation
DIM_SUBSCRIBER_MOBILE|add|ACCOUNT_NUMBER|string|Account Number|ACCNUM	Add ACCOUNT_NUMBER attribute to the DIM_SUBSCRIBER_MOBILE dimension with String type, property “label” equal to Account Number

	and property “alias” equal to ACCNUM
DIM_SUBSCRIBER_MOBILE|add|ADDRESS_2|string||	Add ADDRESS_2 attribute to the DIM_SUBSCRIBER_MOBILE dimension with String type. Without changing label and alias properties.
DIM_SUBSCRIBER_MOBILE|update|DOB|| Date Of Birth|	Update DOB attribute in the DIM_SUBSCRIBER_MOBILE dimension – set new value “Date Of Birth” to the “label” property for DOB attribute. Without changing data type and alias property
DIM_SUBSCRIBER_MOBILE|update|HANDSET||Handset|HNDST	Update SURNAME attribute in the DIM_SUBSCRIBER_MOBILE dimension – set new value “Handset” to the “label” property and new value “HNDST” to the “alias” property for the NAME dimension. Without changing data type of attribute

Second example of csv command file:

```
#operation|table|column|alias
update|DIM_SUBSCRIBER_MOBILE|ACCOUNT_NUMBER|ACCNUM
remove|DIM_SUBSCRIBER_MOBILE|ESTIMATED_INCOME
```

Command	Explanation
update|DIM_SUBSCRIBER_MOBILE|ACCOUNT_NUMBER|ACCNUM	Update ACCOUNT_NUMBER attribute in the DIM_SUBSCRIBER_MOBILE dimension – set new value “ACCNUM” to the “alias” property
remove|DIM_SUBSCRIBER_MOBILE|ESTIMATED_INCOME	Remove ESTIMATED_INCOME attribute from the DIM_SUBSCRIBER_MOBILE dimension. **Note**: You can only remove Custom Attributes

For submitting command files you should use following shell command:

```
$TNF_PROV_TOOL/attributor.sh -f /opt/tnf/path/to/commands.csv
```

2.2.4 Import provisioning files – import.sh

In order to run provisioning job we need to execute following:

```
$TNF_PROV_TOOL/load.sh -f /path/to/file -t dim_type_table
```

For example:

```
$TNF_PROV_TOOL/load.sh -f /opt/tnf/provdata/applications_20151114.csv -t
dim_application
```

It will take some time to import information – especially for large provisioning files. For current status monitor the following files:

- Console output of Provisioning too: **$TNF_PROV_TOOL/log/tools.log**
- Application Framework daemon log: $TNF_BIS_MAIN**/bis-demon/log/demon.log**

After provisioning finished you will see in log tools.log file similar lines

```
2016-01-13 16:56:51,725 [INFO ] main [com.tnf.bis.provisioning.Exporter.export]: Written to
CSV file from DIM_DEVICE table: 45248 records out of 45248
2016-01-13 16:56:51,726 [INFO ] main [com.tnf.bis.provisioning.Exporter.export]: CSV file from
DIM_DEVICE table written in : 4.168 seconds
2016-01-13 16:56:53,268 [INFO ] main [com.tnf.bis.provisioning.Loader.go]: Overall time spent
for loading records to DIM_DEVICE table: 16.502 seconds
```

You also can check number of Dimension Instances created and compare that with number of instances in provisioning file through Presto command. You should input "presto" in command line and then you can input sql-queries in the appeared prompt place. For example for viewing PROTOCOL dimensions:

```
$ presto
presto:default> use tnf;
presto:tnf> select * from dim_protocol;
```

You will see following output

```
 name | label | last_updated_time
------+-------+-------------------
 tcp  | tcp   | 20160113115046309
 udp  | udp   | 20160113115046309
(2 rows)

Query 20160113_172157_00160_r3ir3, FINISHED, 1 node
Splits: 2 total, 2 done (100.00%)
0:00 [2 rows, 617B] [13 rows/s, 4.18KB/s]
```

Another way to check is view audit tables. They are described in Appendix Provisioning Audit tables chapter.

2.2.5 Delete Dimensions – delete.sh

Dimensions may be deleted, for example,

```
$TNF_PROV_TOOL/delete.sh -f /path/to/file/delete.txt -t dim_type_table
```

The above command will reads records from input files and deletes records using key values.

If you want to delete all rows from the table, do not specify "-f". This wipes a table:

```
$TNF_PROV_TOOL/delete.sh -t dim_type_table
```

2.2.6 Export – export.sh

You can export all of the contents of a given dictionary table using the export tool. As with the import.sh command, you provide a filename and table name, and it exports the contents of that dictionary table to the specified file. This is useful if you wish to get a snapshot of the current values and use it as a basis to create an updated file.

```
$TNF_PROV_TOOL/export.sh -f /path/to/file/filename.csv -t dim_type_table
```

Exports contents of table to filename.csv

2.3 Subscribers CRM Import

Subscriber tables are also presented as dictionary tables and are very similar to regular dimension tables, and therefore the type of CSV import file follows the same format that Cells and Devices provisioning uses, and, the same tools with same command structures are used for performing provisioning operations on subscribers.

Two tables are used for Subscribers Import:

DIM_SUBSCRIBER_MOBILE and DIM_SUBSRIBER_FIXEDLINE (for Fixed line interface)

There is one mandatory columns needed in DIM_SUBSCRIBER_MOBILE

- IMSI

For fixed line provisioning there is also one mandatory columns required.

- CSID

2.3.1 Subscriber Attributes

Additional attributes can be defined for each customer. These attributes have to be created for Subscriber dimension types with the ***attributor.sh*** tool. This is performed in the same way as regular dimension provisioning. Let's assume that the provisioning file follows the example below:

```
#IMSI|PLAN|SUBSCRIBER_GENDER|DOB
XXXXYYYYZZZZAAAA|post-paid|Female|18/02/1980
```

And PLAN, SUBSCRIBER_GENDER, DOB are not existing as default attributes. A command file for adding these attributes might look like:

```
#table|operation|column|type|label
DIM_SUBSCRIBER_MOBILE|add|PLAN|string|Plan (Prepaid/Postpaid)
DIM_SUBSCRIBER_MOBILE|add|SUBSCRIBER_GENDER|string|Gender
DIM_SUBSCRIBER_MOBILE|add|DOB|string|Date Of Birth
```

The following table uses example above and explains what each line means.

ATTRIBUTE	TYPE	LABEL
PLAN	String	Plan (Prepaid/Postpaid)
GENDER	String	Gender
DOB	String	Date of Birth

For submitting this command files you should use following shell command:

```
$TNF_PROV_TOOL/attributor.sh -f /opt/tnf/path/to/add_subscr_attrs.csv
```

2.3.2 Run Subscribers Provisioning job

In order to run provisioning job we need to execute usual provisioning command for dim_subscriber_mobile table:

```
$TNF_PROV_TOOL/load.sh -f /path/to/file -t dim_subscriber_mobile
```

2.3.3 Delete Subscribers

Like usual dimensions deleting you can delete subscriber which are presented in specified file by following command:

```
$TNF_PROV_TOOL/delete.sh -f /path/to/file -t dim_subscriber_mobile
```

2.3.4 Provision certain segments

The ***load.sh*** tool has an attribute -- *subscriber-segment* for provisioning only certain segments. This attribute must contain a list of 1 or more integers delimited by comma corresponding to the segments.

Example:

```
$TNF_PROV_TOOL/load.sh -t DIM_SUBSCRIBER_MOBILE -f subscrbers.csv --
subscriber-segment="1,3,7"
```

2.3.5 Subscriber Provisioning for Fixed Line

Subscriber provisioning for fixed line is similar to usual Subscribers provisioning except two things. It is based on CSID and not on IMSI. So CSID is mandatory field in provisioning files. And separate dictionary table DIM_SUBSCRIBER_FIXEDLINE is used.

3 Interface Provisioning

3.1 Dimension Provisioning for Mobile UserPlane

3.1.1.1 Provisioned Out of the Box – Constant

The following dimensions are provisioned out of the box and should not be modified without a Vantage upgrade:

- Protocol
- Roaming_Type
- RAT

3.1.1.2 Provisioned Out of the Box - Modifiable

The following dimensions are provisioned out of the box as being operator independent but can be updated if a change to existing dimension instances or some additional dimension instances are required.

- User_Plane_Status
- HPLMN
- VPLMN
- APPLICATION_PROTOCOL
- QOSCATEGORY
- CES_RULE_TYPE_NAME

3.1.1.3 Manual Provisioning

The following dimensions should be provisioned manually as they are Operator specific.

- Collector
- Cell
- Application
- RNC
- ENODEB
- DEVICE
- SGSN
- GGSN
- SGW
- APN

3.2 Dimension Provisioning for Mobile Gn ControlPlane

3.2.1.1 Provisioned Out of the Box – Constant

The following dimensions are provisioned out of the box and should not be modified without a Vantage upgrade:

- PDP_STATUS
- TRANSACTION_TYPE
- Roaming_Type
- RAT

3.2.1.2 Provisioned Out of the Box - Modifiable

The following dimensions are provisioned out of the box as being operator independent but can be updated if a change to existing dimension instances or some additional dimension instances are required.

- HPLMN
- VPLMN
- QOSCATEGORY

3.2.1.3 Manual Provisioning

The following dimensions should be provisioned manually as they are Operator specific.

- APN
- Cell
- Collector
- GGSN
- SGSN
- Device

3.3 Dimension Provisioning for Mobile LTE ControlPlane

3.3.1.1 Provisioned Out of the Box – Constant

The following dimensions are provisioned out of the box and should not be modified without a Vantage upgrade:

- LTE_CAUSE_CODE
- LTE_TRANSACTION_TYPE
- Roaming_Type
- RAT

3.3.1.2 Provisioned Out of the Box - Modifiable

The following dimensions are provisioned out of the box as being operator independent but can be updated if a change to existing dimension instances or some additional dimension instances are required.

- HPLMN
- VPLMN
- QOSCATEGORY - For 4G and 2G/3G the same dimension are used. For 2G/3G "interface" attribute equals "2G/3G" and dimension instance has only ARPTHP attribute filled and for 4G "interface" attribute equals "4G" and dimension instance has QCI and Priority filled with ARPTHP blank

3.3.1.3 Manual Provisioning

The following dimensions should be provisioned manually as they are Operator specific.

- APN
- Cell
- Collector
- MME
- SGW
- DEVICE

3.4 Dimension Provisioning for RAN

3.4.1.1 Provisioned Out of the Box – Constant

The following dimensions are provisioned out of the box and should not be modified without a Vantage upgrade:

- RAT

3.4.1.2 Provisioned Out of the Box - Modifiable

The following dimensions are provisioned out of the box as being operator independent but can be updated if a change to existing dimension instances or some additional dimension instances are required.

- HPLMN
- VPLMN

3.4.1.3 Manual Provisioning

The following dimensions should be provisioned manually as they are Operator specific.

- AggregatedCarrierFlag
- APN
- BSC
- BTS
- Cell
- ChannelBandwidth
- ChannelCentreFrequency
- Collector
- EARFCN
- eNodeB
- FrequencyBand
- HandoverType
- MME
- ModulationScheme
- NodeB
- RNC
- Device
- UARFCN

3.5 Dimension Provisioning for Voice/SMS

3.5.1.1 Provisioned Out of the Box – Constant

The following dimensions are provisioned out of the box and should not be modified without a Vantage upgrade:

- Roaming_Type
- Voice_Status_Code
- Voice_Transaction_Type
- RAT

3.5.1.2 Provisioned Out of the Box - Modifiable

The following dimensions are provisioned out of the box as being operator independent but can be updated if a change to existing dimension instances or some additional dimension instances are required.

- HPLMN

- VPLMN
- Other_Party_Group

3.5.1.3 Manual Provisioning

The following dimensions should be provisioned manually as they are Operator specific.

- BSC
- Cell
- Collector
- MSC
- RNC
- Device

3.6 Dimension Provisioning for VoLTE

3.6.1.1 Provisioned Out of the Box – Constant

The following dimensions are provisioned out of the box and should not be modified without a Vantage upgrade:

- LTE_CAUSE_CODE
- LTE_TRANSACTION_TYPE

3.6.1.2 Provisioned Out of the Box - Modifiable

The following dimensions are provisioned out of the box as being operator independent but can be updated if a change to existing dimension instances or some additional dimension instances are required.

- Other_Party_Group
- QOSCATEGORY

3.6.1.3 Manual Provisioning

The following dimensions should be provisioned manually as they are Operator specific.

- APN
- CELL
- Codec
- Collector
- Device
- ENODEB

- REALM
- SGW
- Volte_Direction

3.7 Dimension Provisioning for Fixed Line

3.7.1.1 Provisioned Out of the Box – Constant

The following dimensions are provisioned out of the box and should not be modified without a Vantage upgrade:

- FixedLine_Transaction_Type
- FixedLine_Cause_Code
- Protocol
- Ces_Rule_Type
- User_Plane_Status

3.7.1.2 Provisioned Out of the Box - Modifiable

The following dimensions are provisioned out of the box as being operator independent but can be updated if a change to existing dimension instances or some additional dimension instances are required.

- Other_Party_Group

3.7.1.3 Manual Provisioning

The following dimensions should be provisioned manually as they are Operator specific.

- Application
- Collector
- CPEDevice
- Device
- DSLAM
- Fixedline_QoS
- Technology
- Tier0Node
- Tier1Node

4 Appendix

4.1 Dimension Provisioning

4.1.1 Location (Cells) Provisioning

A Cell's provisioning file contains information about an operator's network. 2G/3G and 4G cells are provisioned as the same dimension "CELL" and use the same table DIM_CELL.

Provisioning file for 2G/3G cells should contain at least 4 columns (MCC, MNC, LAC, CI) and a file for 4G cells should contain at least 3 columns (MCC, MNC, ECI). These fields are used for concatenation an aggregation key.

We can see there how Complex Aliases are used. For CELL_KEY aggregation key following "alias" property are configured:

```
MCC:MNC:LAC:CI,MCC:MNC:ECI
```

It means that provisioning tool will look for the headers MCC and MNC and LAC and CI (or MCC and MNC and ECI) and concatenate them with colon symbol and store to CELL_KEY.

For example if we have similar file:

```
#MCC|MNC|LAC|CI|Cell_name
22|01|12|123|cellABC
22|01|13|324|cellBCD
```

Following values will be inserted to the DIM_CELL table

CELL_KEY	CELL_NAME	MCC	MNC	LAC	CI
22:01:12:123	cellABC	22	01	12	123
22:01:12:324	cellBCD	22	01	13	324

Another example. In case we have following LTE cell file:

```
#MCC|MNC|ECI|Cell_name
22|01|123|cellABC
22|01|324|cellBCD
```

Following values will be inserted to the DIM_CELL table

CELL_KEY	CELL_NAME	MCC	MNC	ECI
22:01:123	cellABC	22	01	123
22:01:324	cellBCD	22	01	324

Error situation: Two sets of columns cannot relate to single field through different aliases. For example the tool will stop import and threw an error for a CSV file with following header:

```
#MCC|MNC|LAC|CI|ECI|Cell_name
```

Because it cannot understand what columns must be used for building complex attribute CELL_KEY. LTE Cells provisioning file contains information about operator's network. CELL will be provisioned in a similar way to main Cells. But for LTE Cells another 3 fields are mandatory in Provisioning file (MCC, MNC, ECI).

4.1.2 Devices Provisioning

For Devices – each TDR contains **TACFAC** value – that uniquely identifies Device. The problem here is that one Device can have multiple TACFACs. During data processing – Mediation sends request to Injector with TACFAC value. Injector responds with correct Device ID.

At minimum – following attributes are mandatory in Device provisioning file:

- **TACFAC** – 8 digits long value,
- **LABEL** – unique name of device. Also used as a label in GUI.

Example configuration for Devices import can be found in:

/opt/apps/interfaces/examples/config/example-import-tacfac.cfg.xml

This configuration expects following Device provisioning file format:

```
#TACFAC|LABEL
{tacfac value}|{device name}
```

For example:

```
#TACFAC|LABEL
44927595|Motorola V50
44896150|Bang & Olufsen Beocom 9800 Db
44895250|Motorola V2188
```

When run it will:

- Create DEVICE dimension instance with TACFAC **= 44927595 and label = Motorola V50**

Customer will most probably provide this file with more attributes – so configuration file has to be modified. High level overview of changes required can be found in Generic Steps Required To Import Provisioning Information.

4.1.3 Applications Provisioning

Application provisioning file contains the ID of applications that the Probe is able to discover. It also contains additional attributes defined for particular applications. These attributes might be used to create a hierarchy of Applications.

The customer is not defining this information. The number of Applications that is imported depends on the capability of the Service Detection mechanism in Sourceworks NG.

Application provisioning file has following format:

```
#NAME,LABEL,SERVICE

{application identifier as in TDR},{app label},{type of service}
```

For example

```
#NAME,LABEL,SERVICE

android market,Android Market,Content Stores

blackberry appworld,Blackberry AppWorld,Content Stores

playstation network,Playstation Network,Gaming
```

During installation process one group definition for Applications has been created. It's based on SERVICE attribute.

4.1.4 Roaming Support

Roaming functionality requires proper system configuration. This is done using provisioning functionality by assigning attribute values to existing or to create new dimension instances.

4.1.4.1 Home and Visited Operator

HPLMN and VPLMN represent operators that should be recognized, based on customer IMSIs values (HPLMN) or MCC/MNC fields in TDR feed (VPLMN).

To recognize operator – these also have to be provisioned. Following values are mandatory.

Attribute	Description	Example
MCCMNC	Identifies operator. In case if operator has multiple MCC/MNC pairs – multiple rows have to be created for the same operator.	20601 – VFIE
MCC	Separate MCC value	206
MNC	Separate MNC value	01
TYPE	Type of operator: One of 0 (international), 1 (home) or 2 (probe)	1
LABEL (HPLMN/VPLMN/NETWORK)	This should be the unique name of operator	VF Ireland
COUNTRY	Country of operator	Ireland
COUNTRY_CODE	Phone Country Code	353
GROUP	Operator's group	Vodafone

Note: You have to at least provision all HPLMN / VPLMN values for the local operator's traffic.

Note: It is very important to carefully provision these instances. This might result in increasing database size if all operators are loaded.

4.2 Provisioning Audit tables

Analytics Framework saves historical information about all provisioning imports and all changes which were performed through attributor.sh. It uses following two audit tables which are stored in Postgresql relational database:

Provisional Table Name	Description
PROVISIONING_IMPORT_AUDIT	This will contain a historical records of all provisioning imports.
PROVISIONING_ATTRIB_AUDIT	This will contain a historical records of all changes which were performed by attributor.sh utility.

We can view this information by using psql tool.

```
psql -d tnfbis -p 5433 -c "SELECT * FROM tnfdict.provisioning_attrib_audit" boss
# or
psql -d tnfbis -p 5433 -c "SELECT * FROM tnfdict.provisioning_import_audit" boss
```

4.2.1 PROVISIONING_IMPORT_AUDIT Schema

Column Name	Data type	Comment
ID	INT	
TIMEID	TIMESTAMP	When the operation was performed
END_TIME	TIMESTAMP	When the operation was completed
STATUS	String	Status of completion
TABLE	Varchar	Table name (dictionary/dimension)
INSERTED	INT	Count of inserted rows
UPDATED	INT	Count of updated rows
DELETED	INT	Count of deleted rows

For example for viewing about CELL provisioning you can use following sql-query:

```
psql -d tnfbis -p 5433 -c "SELECT * FROM tnfdict.provisioning_import_audit
WHERE upper(table_name)='dim_cell' order by timeid asc;" boss
```

Output:

```
id |         timeid          |        end_time         |      status       | table_name | inserted | updated | deleted
---+-------------------------+-------------------------+-------------------+------------+----------+---------+---------
  6 | 2016-01-13 16:57:11.739 | 2016-01-13 16:57:11.882 | FINISHED_SUCCESS | dim_cell   |        1 |       0 |
 42 | 2016-01-13 18:18:18.749 | 2016-01-13 18:18:18.915 | FINISHED_SUCCESS | dim_cell   |        0 |       0 |
 73 | 2016-01-13 18:35:33.639 | 2016-01-13 18:35:34.542 | FINISHED_SUCCESS | dim_cell   |     2548 |       0 |
 75 | 2016-01-13 18:40:12.177 | 2016-01-13 18:40:12.718 | FINISHED_SUCCESS | dim_cell   |        0 |       0 |
 80 | 2016-01-13 18:40:42.529 | 2016-01-13 18:40:43.166 | FINISHED_SUCCESS | dim_cell   |      572 |       0 |
 86 | 2016-01-13 22:53:33.496 | 2016-01-13 22:53:34.052 | FINISHED_SUCCESS | dim_cell   |        0 |       0 |
```

We can see that some rows contain zero values for “inserted” and “updated” fields. It means that provisioning file did not contain any new dimensions or changed values for existing dimensions.

4.2.2 PROVISIONING_ATTRIB_AUDIT Schema

Column Name	Data type	Comment
ID	INT	

TIMEID	DATE	When the operation was performed
END_TIME	DATE	When the operation was completed
STATUS	String	Status of completion
TABLE	Varchar	Table name (dictionary/dimension)
COLUMN	Varchar	Column name
OPERATION	Varchar	Operation name
DEFINITION	TEXT	Serialized string of operation parameters (type, label, alias, …)

For example if we perform such command file which we described in Manage attribute definitions chapter:

```
#table|operation|column|type|label|alias
DIM_SUBSCRIBER_MOBILE|add|ACCOUNT_NUMBER|VARCHAR|Account Number|ACCNUM
DIM_SUBSCRIBER_MOBILE|add|ADDRESS_2|VARCHAR||
DIM_SUBSCRIBER_MOBILE|update|DOB||Date Of Birth|
DIM_SUBSCRIBER_MOBILE|update|HANDSET||Handset|HNDST
```

Then we can see information in database by executing following query:

```
psql -d tnfbis -p 5433 -c "SELECT * FROM tnfdict.provisioning_attrib_audit
WHERE status='FINISHED_SUCCESS' AND
upper(table_name)='DIM_SUBSCRIBER_MOBILE' order by id" boss
```

Output:

```
 id |        timeid         |       end_time        |      status      |      table_name       |    column_name    | operation |       definition
----+-----------------------+-----------------------+------------------+-----------------------+-------------------+-----------+-------------------------
  3 | 2016-01-13 23:51:33.165 | 2016-01-13 23:51:33.216 | FINISHED_SUCCESS | DIM_SUBSCRIBER_MOBILE | dob               | UPDATE    | {OPERATION=UPDATE, ...}
  4 | 2016-01-13 23:56:00.731 | 2016-01-13 23:56:00.818 | FINISHED_SUCCESS | DIM_SUBSCRIBER_MOBILE | ACCOUNT_NUMBER    | ADD       | {OPERATION=ADD, ...}
  5 | 2016-01-13 23:56:08.478 | 2016-01-13 23:56:08.508 | FINISHED_SUCCESS | DIM_SUBSCRIBER_MOBILE | ADDRESS_2         | ADD       | {OPERATION=ADD, ...}
  7 | 2016-01-13 23:56:11.141 | 2016-01-13 23:56:11.174 | FINISHED_SUCCESS | DIM_SUBSCRIBER_MOBILE | handset           | UPDATE    | {OPERATION=UPDATE, ...}
```

4.3 Auto Export to Streams Server

Analytics Framework has got possibility to export dimensions data. It saves all content of specified Dictionary Table in a temporary CSV file. Analytics framework saves these files in “/opt/tnf/apps/bis-main-var/provisioning” directory by default. The set of columns of each file are configured as required by Streams server. Streams server periodically checks these files and uses them for internal “Streams provisioning”.