MeshPhoneGen XML to Asterisk Conversion Utility

Installation and operation

Written by:
Martin Alcock, M. Sc, VE6VH

For software revision: 1.2

Praebius Communications Inc,
43 Citadel Bay NW,
Calgary, AB T3G 3Y4
www.praebius.com

Table of Contents

List of Tables

Document Revision History

Date	Rev	Description
May 2023	1.2	Initial release after local testing

Table 1 Revision History

Distribution

Rev	Distribution
All	For amateur use only.

Table 2 Distribution

Reference Documents

[1] gnu.org, "General Public Licence," [Online]. Available: https://www.gnu.org/licenses/gpl-3.0.en.html. [Accessed 25th February 2018].

[2] w3.org, "Extensible Markup Language (XML) 1.0 (Fifth Edition)," 26 November 2008. [Online]. Available: https://www.w3.org/TR/xml/. [Accessed 26 05 2023].

Glossary of Terms

PBX	Private Branch Exchange. A node in a telephone network that provides connectivity for a series of local extensions to a set of trunks.
XML	Extensible mark up language. The source language that contains the Meshphone data.
Asterisk	An open source telephony toolkit.

Release Notes

Initial Release 1.0:

First release was limited to a disk file source only.

Update to Rev 1.1:

Added the ability to download the data from a web page, which is mutually exclusive with the disk file method.

Update to Rev 1.2:

1. Added the option to specify a stanza name on the command line.
2. Changed templates so that default and specified routes were mutually exclusive.
3. Added an error route to each dialplan entry.
4. Added a default route at the end of the file.

Installing the Software

It is recommended that the software be installed in a sub-directory of the asterisk configuration files, /etc/asterisk. Create a directory called MeshPhoneGen. From this directory, log in as a super-user and type the following:

```
https://github.com/ve6vh/meshphonegen.git
```

There are four files in this directory:

File	Purpose
MeshPhoneGen.jar	Executable java bytecode
Meshphone.tmpl	A generation template example
runMeshPhoneGen	A file to run the software from a remote URL
MeshPhoneGen.pdf	This documentation.

Table 3 MeshPhoneGen files

Next, add the following line to your extensions.conf in /etc/asterisk

```
#include "/etc/asterisk/MeshPhoneGen/Meshphone.conf"
```

Then execute the file 'runMeshPhoneGen' once from the directory to make the required file.

Weekly updates

For a weekly update, set up a CRON schedule to run the utility as follows:

```
crontab -e
```

Then add the following line:

```
0 2 * * 0  /etc/asterisk/MeshPhoneGen/runMeshPhoneGen
```

This will provide a weekly update to your Meshphone routing by executing the utility once a week on Sunday, at 2am. If for any reason it fails, no output file will be written, and the original will be preserved.

Running the software

The software is in a self-contained jar (java archive) file which can be executed directly from the command line. A minimum of the Open JDK must be installed. This software does not have require any native or additional libraries. All parameters are supplied on the command line.

The command line switches and arguments are listed in Table 4.

Switch	Usage	Argument
-h	None	Help on the command line arguments
-d	Debug flags	Specifies the debug modes
-o	Required	Name of the output file
-t		Template file name
-s		Asterisk stanza name in the configuration file, usually 'MeshPhone'
-X	Mutually exclusive	Name of the file containing XML data
-u		URL of the web page containing the XML data

Table 4 Command line switches

The debug values are specified in hexadecimal according to Table 5.

Flag	Hex	Debug data source
DEBUG_NONE	0x0000	Debugging not enabled.
DEBUGA_AREAS	0x0001	Dump area codes as they are encountered

Table 5 Debug mode flags

The -X switch and -u switch determine where the source file is to be found, the -X switch uses a local file, and the -u will fetch from a URL using an HTTP GET.

Example syntax of a using a local file is as follows:

```
java -jar MeshPhoneGen.jar -s MeshPhone -o <output> -t <templ> -X <XML filename>
```

and for a remote URL:

```
java -jar MeshPhoneGen.jar -s MeshPhone -o <output> -t <templ> -u <http://URL>
```

How it works

MeshPhoneGen reads the XML source file and extracts data from the <dialplan> nodes only, as shown in Figure 1.

```
<dialplan>
   <network_start>131-2165</network_start>
   <network_end>131-2165</network_end>
   <local_start>2165</local_start>
   <local_end>2165</local_end>
   <serving_office>97321</serving_office>
   <translation> 1312165</translation>
   <recno>1</recno>
   <last_upd>2018-12-01 01:17:00</last_upd>
</dialplan>
<dialplan>
   <network_start>131-6300</network_start>
   <network_end>131-6399</network_end>
   <local_start>6300</local_start>
   <local_end>6399</local_end>
   <serving_office>13163</serving_office>
   <translation>_13163xx</translation>
   <recno>2</recno>
   <last_upd>2018-12-01 01:18:00</last_upd>
</dialplan>
```

Figure 1 Sample XML dial plan

Once this has been completed, the data is then sorted into numerical order by the *network_start* entry, to begin the generation process, which uses the template to generate its output. From the template definition, the output of the two entries above will generate:

```
[MeshPhone]
;
exten =>  1312165,1,Dial(IAX2/n2mh_m/${EXTEN})          ;Last updated 2018-12-01 01:17:00
exten =>  1312165,n,Goto(Utilities,Sorry,1)             ;

exten => _13163xx,1,Dial(IAX2/n2mh_m/${EXTEN})          ;Last updated 2018-12-01 01:18:00
exten => _13163xx,n,Goto(Utilities,Sorry,1)             ;
```

Figure 2 Generated output

Generation template

Figure 3 illustrates a sample template.

```
#
# Template to generate meshphone fields. Each entry contains a
# match field, and a prototype field, separated by '=>'.
# The match field is tested against the translated code for
# each dialplan entry. If a match occurs, then the prototype
# field is evaluated and emitted. Only the number of characters
# in the match field are evaluated, hence 2044 matches 2044xxx.
# String substitution fields in the prototype contain a % + one character.
# %x emits exten => <xlated code>
# %u emits the time stamp
# %T emits the EOF marker
################################################################################
#
# Specified routes to local trunks or other dialplans. Modify to suit your
# installation here. All template matching data MUST be numeric, no
# asterisk shorthand is currently supported.
#
2044=>%x,1,Dial(IAX2/ve4drk_i/${EXTEN:3:4})             ;VE4DRK Manitoba
2046=>%x,1,Dial(IAX2/ve4drk_i/${EXTEN:3:4})             ;VE4DRK Manitoba
#
# this and hosted PBX codes
#
25054=>%x,1,Goto(LocalSets_250,${EXTEN:3},1)            ;BC mainland and interior
4035=>%x,Goto(LocalSets,${EXTEN:3},1)                   ;local extensions on this PBX
4065=>%x,1,Goto(LocalSets_406,${EXTEN:3},1)             ;Montana
#
# network numbers decoded and routed locally
#
5551212=>%x,Goto(LocalSets,5101,1)                              ;Information
#
################################################################################
#
# Default route. Mutually exclusive to all specified routes. If no match is found
# then this one is adopted. NNN matches any digit 0-9.
#
NNN=>%x,1,Dial(IAX2/n2mh_m/${EXTEN})            ;Last updated %u
#
################################################################################
#
# The following is added after either a specific or default route. XXX matches
# all area codes and templates.
#
XXX=>%x,n,Goto(Utilities,Sorry,1)               ; %n
#
################################################################################
#
# End of file default. Added at the end.
#
EOF=>%T,1,Goto(Utilities,Sorry,1)               ; %n
#
```

Figure 3 Sample template file

The format of the template is as follows:

```
<prototype> => <field> , <asterisk command>
```

The prototype consists of enough digits to uniquely identify a dial plan entry. In the example template, the following protypes are used to match numbers that are processed locally.

Prototype	Numbers matched
2044	2044000 to 2044999
2046	2046000 to 2046999
25054	2505400 to 2505499
4035	4035000 to 4035999
4065	4065000 to 4065999

Table 6 Prototypes and matches

The fields and their corresponding emitted values in the output file are listed in Table 7

Field	Field emitted
%x	exten => translated code
%T	exten => _xxxxxxx
%u	Date and time of last update
%n	A new line is emitted in the generated file

Table 7 Field specifications

Sections in the template

There are four sections of template entries:

1. Routing for extensions that are provisioned locally on this PBX.
2. Routing for extensions that are connected to other PBXs by an IAX2 trunk.
3. Default routing for all other area codes.
4. Error routing if the default fails or the code is not recognized.