



Julius Plugin

Administrator Guide

Revision: 4

Distribution: Debian / Ubuntu

Created: February 3, 2017

Last updated: May 20, 2017

Author: Arsen Chaloyan

Table of Contents


1	Overview.....	3
1.1	Applicable Versions.....	3
1.2	Supported Distributions	3
1.3	Authentication.....	3
2	Installing Deb Packages Using Apt-Get	4
2.1	Repository Configuration	4
2.2	GnuPG Key.....	4
2.3	Repository Update	4
2.4	Julius Plugin Installation.....	4
3	Installing Deb Packages Manually.....	6
3.1	Package List.....	6
3.2	Package Installation Order.....	7
4	Obtaining License	8
4.1	License Type.....	8
4.2	Node Information.....	8
4.3	License Installation	8
5	Configuring Server and Plugin	9
5.1	Plugin Factory.....	9
5.2	Logger.....	9
5.3	Julius	9
6	Validating Setup.....	10
6.1	Launching Server.....	10
6.2	Launching Client.....	10

1 Overview

This guide describes how to obtain and install binary packages for the Julius plugin to the UniMRCP server on Debian-based Linux distributions. The document is intended for system administrators and developers.

1.1 Applicable Versions

Instructions provided in this guide are applicable to the following versions.

 UniMRCP 1.4.0 and above
UniMRCP Julius Plugin 1.0.0 and above

1.2 Supported Distributions


UniMRCP binary packages are currently available only for x86_64 (64-bit) architecture.

Operating System	32-bit	64-bit
Ubuntu 14.04 LTS (trusty)		✓
Ubuntu 16.04 LTS (xenial)		✓

Note: packages for other distributions can be made available upon request. For more information, contact services@unimrcp.org.

1.3 Authentication

UniMRCP binary packages are available to authenticated users only. In order to register a free account with UniMRCP, please visit the following page.

 <https://www.unimrcp.org/profile-registration>

Note: a new account needs to be verified and activated prior further proceeding.

2 Installing Deb Packages Using Apt-Get

Using the APT package handling utility (`apt-get`) is recommended for installation of UniMRCP binary packages.

2.1 Repository Configuration

The content of a typical configuration file of the APT repository, to be placed in `/etc/apt/sources.list.d/unimrcp.list`, is provided below.

```
deb [arch=amd64] https://username:password@unimrcp.org/repo/apt/ distr main
```

The *username* and *password* fields included in the HTTPS URI must be replaced with the corresponding account credentials.

The *distr* field must be replaced with the corresponding distribution code name such as *trusty*, *xenial*, etc.

2.2 GnuPG Key

For verification of binary packages, UniMRCP provides a public GnuPG key, which can be retrieved and installed as follows.

```
wget -O - https://unimrcp.org/keys/unimrcp-gpg-key.public | sudo apt-key add -
```

2.3 Repository Update

In order to check for updates and apply the changes in the APT configuration, use the following command.

```
sudo apt-get update
```

2.4 Julius Plugin Installation

In order to install the Julius plugin package, including all the dependencies, use the following command.

```
sudo apt-get install unimrcp-julius
```

As a result, *apt-get* will check and prompt to download all the required packages by installing them in the directory `/opt/unimrcp`.

In order to install the en-US model package for Julius, use the following command.

```
sudo apt-get install unimrcp-julius-model-en-us
```

Note: this package provides acoustic data files required for the Julius engine to perform recognition for the en-US language. Packages for other languages can be made available upon request.

In order to install the additional data files for the sample client application *umc*, the following command can be used.

```
sudo apt-get install umc-addons
```

Note: this package is optional and provides additional data which can be used for validation of basic setup.

3 Installing Deb Packages Manually

UniMRCP deb packages can be installed manually using the *dpkg* utility. Note, however, that the system administrator should take care of package dependencies and install all the packages in appropriate order.

The deb packages have the following naming convention:

```
$packagename_${universion}-${distr}_${arch}.deb
```

where

- *packagename* is the name of a package
- *universion* is the UniMRCP version
- *distr* is the distribution code name (trusty, xenial, ...)
- *arch* is the architecture (amd64, i386, all, ...)

3.1 Package List

The following is a list of UniMRCP deb packages required for the installation of Julius plugin.

Package Name	Description
unimrcp-julius	Julius plugin to the server.
unimrcp-julius-model-en-us	Julius en-US language files.
umc-addons	Sample en-US data files used with umc. [Optional]
unilicnodegen	Node information retrieval tool, required for license deployment.
unimrcp-server	Shared library and application of the server.
unimrcp-client	Shared libraries and sample applications of the client. [Optional]
unimrcp-demo-plugins	Set of demo plugins to the server. [Optional]
unimrcp-common	Data common for the client and the server.
uniapr	UniMRCP edition of the Apache Portable Runtime (APR) library.

uniapr-util	UniMRCP edition of the Apache Portable Runtime Utility (APR-Util) library.
unisofia-sip	UniMRCP edition of the Sofia SIP library.

3.2 Package Installation Order

Packages for APR, APR-Util and Sofia-SIP libraries must be installed first.

```
sudo dpkg --install uniapr_${aprversion}-${distr}_${arch}.deb
sudo dpkg --install uniapr-util_${apuverson}-${distr}_${arch}.deb
sudo dpkg --install unisofia-sip_${sofiaversion}-${distr}_${arch}.deb
```

Then, a package containing common data for the client and the server, and a package for the server should follow.

```
sudo dpkg --install unimrcp-common_${universion}-${distr}_${arch}.deb
sudo dpkg --install unimrcp-server_${universion}-${distr}_${arch}.deb
```

Next, a package containing the utility tool *unilicnodegen*, required for license deployment.

```
sudo dpkg --install unilicnodegen_${stoolversion}-${distr}_${arch}.deb
```

Next, a package containing the Julius plugin should follow.

```
sudo dpkg --install unimrcp-julius_${universion}-${distr}_${arch}.deb
```

Finally, one or more packages corresponding to a particular language should follow.

```
sudo dpkg --install unimrcp-julius-model-en-us_${universion}-${distr}_all.deb
```

4 Obtaining License

The Julius plugin to the UniMRCP server is a commercial product, which requires a license file to be installed.

4.1 License Type

The following license types are available:

- Trial
- Production
- Test and Development

4.2 Node Information

The license files are bound to a node the product is installed on. In order to obtain a license, the corresponding node information needs to be retrieved and submitted for generation of a license file.

Use the installed tool *unilicnodegen* to retrieve the node information.

```
sudo /opt/unimrcp/bin/unilicnodegen
```

As a result, a text file *uninode.info* will be saved in the current directory. Submit the file *uninode.info* for license generation to services@unimrcp.org by mentioning the product name in the subject.

4.3 License Installation

The license file needs to be placed into the directory */opt/unimrcp/data*.

```
sudo cp umsjulius_*.lic /opt/unimrcp/data
```


5 Configuring Server and Plugin

5.1 Plugin Factory

In order to load the Julius plugin into the UniMRCP server, open the file *unimrcpserver.xml*, located in the directory */opt/unimrcp/conf*, and add the following entry under the XML element *<plugin-factory>*. Disable other recognition plugins, if available. The remaining demo plugins might also be disabled, if not installed.

```
<!-- Factory of plugins (MRCP engines) -->
<plugin-factory>
  <engine id="Demo-Synth-1" name="demosynth" enable="true"/>
  <engine id="Demo-Recog-1" name="demorecog" enable="false"/>
  <engine id="Demo-Verifier-1" name="demoverifier" enable="true"/>
  <engine id="Recorder-1" name="mrcpreorder" enable="true"/>
  <engine id="JRecog-1" name="umsjulius" enable="true"/>
</plugin-factory>
```

5.2 Logger

In order to enable log output from the plugin and set filtering rules, open the configuration file *logger.xml*, located in the directory */opt/unimrcp/conf*, and add the following entry under the element *<sources>*.

```
<source name="JULIUS" priority="INFO" masking="NONE"/>
```

5.3 Julius

The configuration file of the plugin is located in */opt/unimrcp/conf/umsjulius.xml* and the data files are in the directory */opt/unimrcp/data/julius*.

Refer to the *Usage Guide* for more information.

6 Validating Setup

Validate your setup by using the sample UniMRCP client and server applications on the same host. The default configuration and data files should be sufficient for a basic test.

6.1 Launching Server

Launch the UniMRCP server application.

```
cd /opt/unimrcp/bin  
sudo ./unimrcpserver
```

In the server log output, check whether the plugin is normally loaded.

```
[INFO] Load Plugin [JRecog-1] [/opt/unimrcp/plugin/umsjulius.so]
```

Next, check for the license information.

```
[NOTICE] UniMRCP Julius License
```

```
-product name:  umsjulius  
-product version: 1.0.0  
-license owner:  Name  
-license type:   trial  
-issue date:    2017-01-28  
-exp date:     2017-02-27  
-channel count: 2  
-feature set:   0
```

Next, make sure that the Julius recognition server processes are normally started and the plugin can connect to them.

```
[INFO] Connected to JServer #0 Control Port localhost:10500 attempt 1  
[INFO] Connected to JServer #1 Control Port localhost:10500 attempt 1  
[INFO] Connected to JServer #2 Control Port localhost:10500 attempt 1  
[INFO] Connected to JServer #3 Control Port localhost:10500 attempt 1
```

6.2 Launching Client

Open the configuration file `/opt/unimrcp/conf/umscenarios.xml` and add a new section corresponding to

the use with the Julius engine.

```
<scenario name="jul" class="Recognizer" profile="uni2">
  <resource-discovery enable="0"/>
  <define-grammar enable="0"/>
  <recognize enable="1" content-type="text/uri-list" content-location="jgrammar.list" audio-
    source="callsteve.pcm"/>
  <termination enable="1">
    <capabilities>
      <codec name="LPCM" rates="8000 16000"/>
    </capabilities>
  </termination>
</scenario>
```

Launch the sample UniMRCP client application *umc*.

```
cd /opt/unimrcp/bin
./umc
```

Run a typical speech recognition scenario by issuing the command *run jul* from the console of the *umc* client application.

```
run jul
```

This command sends a RECOGNIZE request to the server and then starts streaming a sample audio input file *callsteve.pcm* to recognize.

Check for the NLSML results to be returned as expected.

```
<?xml version="1.0"?>
<result>
  <interpretation grammar="request1@form-level" confidence="1.00">
    <instance>call steve</instance>
    <input mode="speech">call steve</input>
  </interpretation>
</result>
```

Visually inspect the log output for any possible warnings or errors.

Note that utterances are stored in the *var* directory, if the corresponding parameter is enabled in the configuration file *umsjulius.xml* and/or requested by the client.