

Powered by Universal Speech Solutions LLC



GoVivace SR Plugin

Administrator Guide

Revision: 2

Distribution: Debian / Ubuntu

Created: June 6, 2019

Last updated: March 15, 2021

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 Plugin Installation.....	5
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 Configuration	9
5.2 RTP Configuration.....	9
5.3 Logger Configuration	9
5.4 Plugin Configuration.....	10
6 Validating Setup.....	12
6.1 Launching Server.....	12
6.2 Launching Client.....	12
Gender Identification	13
Emotion Identification	13
Language and Accent Identification	14
Keyword Spotting	14

1 Overview

This guide describes how to obtain and install binary packages for the GoVivace Speech Recognition (GoVivaceSR) 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 1.6.0 and above
UniMRCP GoVivaceSR Plugin 1.0.0 and above

1.2 Supported Distributions


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

Operating System	Released	End of Support
Ubuntu 16.04 LTS (xenial)	June 2019	March 2021
Ubuntu 18.04 LTS (bionic)	June 2019	TBA
Ubuntu 20.04 LTS (focal)	March 2021	TBA

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

Supply login information by creating a file `/etc/apt/auth.conf.d/unimrcp.conf` containing the following entry.

```
machine unimrcp.org
login username
password password
```

Note: the *username* and *password* fields must be replaced with the corresponding account credentials.

Configure a repository by creating a file `/etc/apt/sources.list.d/unimrcp.list` containing the following entry.

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

Note: the *distr* field must be replaced with the corresponding distribution code name such as *xenial*, *bionic*, *focal*, etc. To determine the distribution code, use ``lsb_release -cs``.

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 Plugin Installation

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

```
sudo apt-get install unimrcp-govivace-sr
```

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 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 the GoVivaceSR plugin.

Package Name	Description
unimrcp-govivace-sr	GoVivaceSR plugin to the server.
unilibevent	UniMRCP edition of the libevent library.
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_${saprversion}-${distr}_${sarch}.deb
sudo dpkg --install uniapr-util_${sapuversion}-${distr}_${sarch}.deb
sudo dpkg --install unisofia-sip_${sofiaversion}-${distr}_${sarch}.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_${suniversion}-${distr}_${sarch}.deb
sudo dpkg --install unimrcp-server_${suniversion}-${distr}_${sarch}.deb
```

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

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

Next, a package containing the libevent library.

```
sudo dpkg --install unilibevent_${libeventversion}-${distr}_${sarch}.deb
```

Finally, a package containing the GoVivaceSR plugin should follow.

```
sudo dpkg --install unimrcp-govivace-sr_${suniversion}-${distr}_all.deb
```

4 Obtaining License

The GoVivaceSR 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 umsgovivacesr_*.lic /opt/unimrcp/data
```


5 Configuring Server and Plugin

5.1 Plugin Factory Configuration

In order to load the GoVivaceSR 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="mrcprecorder" enable="true"/>
  <engine id="GoVivace-SR-1" name="umsgovivacesr" enable="true"/>
</plugin-factory>
```

5.2 RTP Configuration

In order to support audio data sampled at 16 kHz, the corresponding codecs needs to be specified in the configuration file *unimrcpserver.xml* under the XML element *<rtp-settings>* as follows.

```
<rtp-settings id="RTP-Settings-1">
  <codecs own-preference="false"> PCMU PCMA L16/96/8000 telephone-event/101/8000
  PCMU/97/16000 PCMA/98/16000 L16/99/16000 telephone-
  event/102/16000</codecs>
</rtp-settings>
```

For the basic verification test to work, similar settings should be specified in the client configuration file *unimrcpclient.xml* as well.

```
<rtp-settings id="RTP-Settings-1">
  <codecs>PCMU PCMA L16/96/8000 telephone-event/101/8000 PCMU/97/16000
  PCMA/98/16000 L16/99/16000 telephone-event/102/16000</codecs>
</rtp-settings>
```

5.3 Logger Configuration

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="GOVIVACESR-PLUGIN" priority="INFO" masking="NONE"/>
```

5.4 Plugin Configuration

The configuration file of GoVivace plugin is *umsgovivacesr.xml*, located in the directory */opt/unimrcp/conf*.

While default settings should be sufficient for generic use, service endpoints need to be configured accordingly.

```
<!-- Server
Attributes:
* language
    This parameter specifies the language supported by the server.
* sampling-rate
    This parameter specifies the sampling rate supported by the server.
* uri
    This parameter specifies the Service URI of the server.
* method
    This parameter specifies the method supported by the server.
* secret-key
    This parameter specifies a secret key used for authentication to the server.
Elements:
* None
-->
<server name="gender-1" language="en-US" sampling-rate="8000"
    uri="wss://services.govivace.com:7684" method="GenderId" secret-key="****"/>
<server name="emotion-1" language="en-US" sampling-rate="8000"
    uri="wss://services.govivace.com:7687" method="EmotionId" secret-key="****"/>
<server name="language-1" language="en-US" sampling-rate="8000"
    uri="wss://services.govivace.com:7686" method="LanguageId" secret-key="****"/>
<server name="keyword-1" language="en-US" sampling-rate="8000"
    uri="wss://services.govivace.com:49149" method="telephony" secret-key="****"/>
<server name="grxml-1" language="en-US" sampling-rate="8000"
    uri="ws://198.199.70.106:49162" method="answer" secret-key="****"/>
```

```
<server name="grxml-2" language="en-US" sampling-rate="8000"  
  uri="ws://198.199.70.106:49162" method="location" secret-key="***/>  
  
<server name="grxml-3" language="en-US" sampling-rate="8000"  
  uri="ws://198.199.70.106:49162" method="duration" secret-key="***/>
```

Installation of the GoVivace server is not covered in this document.

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

Note: an instance of the GoVivace GStreamer server needs to be running prior starting the UniMRCP 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 [GoVivace-SR-1] [/opt/unimrcp/plugin/umsgovivacesr.so]
```

Next, check for the license information.

```
[NOTICE] UniMRCP GoVivaceSR License

-product name:  umsgovivacesr
-product version: 1.0.0
-license owner:  -
-license type:   trial
-issue date:    2019-05-31
-exp date:      2019-06-30
-channel count: 2
-feature set:   0
```

6.2 Launching Client

Note: the optional package *umc-addons* must be installed for this test to work.

Launch the sample UniMRCP client application *umc*.

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

Gender Identification

Run a typical scenario for gender identification by issuing the command `run gv1` from the console of the `umc` client application.

```
run gv1
```

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

Check for the NLSML results to be returned as expected.

```
<?xml version="1.0"?>
<result>
  <interpretation grammar="builtin:speech/GenderId" confidence="0.95">
    <instance>
      <status>0</status>
      <message>Gender identification is successful</message>
      <gender>male</gender>
      <string-confidence>0.952294</string-confidence>
      <processing-time>1.052527</processing-time>
      <input-speech-duration>0.810000</input-speech-duration>
    </instance>
    <input mode="speech"></input>
  </interpretation>
</result>
```

Emotion Identification

Run a typical scenario for emotion identification by issuing the command `run gv2` from the console of the `umc` client application.

```
run gv2
```

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

Check for the NLSML results to be returned as expected.

```
<?xml version="1.0"?>
<result>
  <interpretation grammar="builtin:speech/EmotionId" confidence="1.00">
```

```
<instance>
  <status>0</status>
  <message>Emotion identification is successful</message>
  <emotion>angry</emotion>
  <emotion-score>0.999981</emotion-score>
  <processing-time>0.344103</processing-time>
</instance>
<input mode="speech"></input>
</interpretation>
</result>
```

Language and Accent Identification

Run a typical scenario for language and accent identification by issuing the command `run gv3` from the console of the `umc` client application.

```
run gv3
```

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

Check for the NLSML results to be returned as expected.

```
<?xml version="1.0"?>
<result>
  <interpretation grammar="builtin:speech/LanguageId" confidence="0.81">
    <instance>
      <status>0</status>
      <message>Language and Accent identification is successful</message>
      <language>english</language>
      <score>0.805773</score>
      <processing-time>0.861588</processing-time>
      <enrollment-audio-time>0.000000</enrollment-audio-time>
    </instance>
    <input mode="speech"></input>
  </interpretation>
</result>
```

Keyword Spotting

Run a typical scenario for keyword spotting by issuing the command `run gv4` from the console of the `umc` client application.

```
run gv4
```

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

Check for the NLSML results to be returned as expected.

```
<?xml version="1.0"?>
<result>
  <interpretation grammar="builtin:speech/telephony" confidence="1.00">
    <instance>call study</instance>
    <input mode="speech">call study</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 *umsgovivacesr.xml* and/or requested by the client.