



GoVivace SR Plugin

Administrator Guide

Revision: 1

Distribution: Red Hat / Cent OS

Created: June 6, 2019

Last updated: June 6, 2019

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 RPMs Using YUM.....	4
2.1	Repository Configuration	4
2.2	Repository Verification.....	4
2.3	Plugin Installation.....	5
3	Installing RPMs 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 Red Hat-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 binary packages are currently available only for x86_64 (64-bit) architecture.

Operating System	32-bit	64-bit
Red Hat / Cent OS 6		
Red Hat / Cent OS 7		

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 RPMs Using YUM

Using the Yellowdog Updater, Modifier (yum), a command-line package management utility for Red Hat-based distributions, is recommended for installation of UniMRCP binary packages.

2.1 Repository Configuration

The content of a typical yum configuration file, to be placed in `/etc/yum.repos.d/unimrcp.repo`, is provided below.

```
[unimrcp]
name=UniMRCP Packages for Red Hat / Cent OS-$releasever $basearch
baseurl=https://username:password@unimrcp.org/repo/yum/main/rhel$releasever/$basearch/
enabled=1
sslverify=1
gpgcheck=1
gpgkey=https://unimrcp.org/keys/unimrcp-gpg-key.public

[unimrcp-noarch]
name=UniMRCP Packages for Red Hat / Cent OS-$releasever noarch
baseurl=https://username:password@unimrcp.org/repo/yum/main/rhel$releasever/noarch/
enabled=1
sslverify=1
gpgcheck=1
gpgkey=https://unimrcp.org/keys/unimrcp-gpg-key.public
```

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

2.2 Repository Verification

In order to verify that yum can properly connect and access the UniMRCP repository, the following command can be used.

```
yum repolist unimrcp
yum repolist unimrcp-noarch
```

where *unimrcp* and *unimrcp-noarch* are names of the sections set in the yum configuration file above.

In order to retrieve a list of packages the UniMRCP repository provides, the following command can be used.

```
yum --disablerepo="*" --enablerepo="unimrcp" list available
```

```
yum --disablerepo="*" --enablerepo="unimrcp-noarch" list available
```

2.3 Plugin Installation

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

```
yum install unimrcp-govivace-sr
```

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

```
yum install umc-addons
```

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

3 Installing RPMs Manually

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

The RPM packages have the following naming convention:

```
$package-$universion-$packageversion.el$rhelversion.$arch.rpm
```

where

- *package* is the name of a package
- *universion* is the UniMRCP version
- *packageversion* is the RPM release version
- *rhelversion* is the Red Hat version
- *arch* is the architecture (x86_64, i686, ...)

3.1 Package List

The following is a list of UniMRCP RPM 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

Note that all the RPM packages provided by UniMRCP are signed by a GNU Privacy Guard (GPG) key. Before starting the installation, you may need to import the public key in order to allow the *rpm* utility to verify the packages.

```
rpm --import https://unimrcp.org/keys/unimrcp-gpg-key.public
```

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

```
rpm -ivh uniapr-$aprversion-$packageversion.el$rhelversion.$sarch.rpm
rpm -ivh uniapr-util-$apuverson-$packageversion.el$rhelversion.$sarch.rpm
rpm -ivh unisofia-sip-$sofiaversion-$packageversion.el$rhelversion.$sarch.rpm
```

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

```
rpm -ivh unimrcp-common-$universion-$packageversion.el$rhelversion.$sarch.rpm
rpm -ivh unimrcp-server-$universion-$packageversion.el$rhelversion.$sarch.rpm
```

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

```
rpm -ivh unilicnodegen-$toolversion-$packageversion.el$rhelversion.$sarch.rpm
```

Next, a package containing the libevent library.

```
rpm -ivh unilibevent-$libeventversion-$packageversion.el$rhelversion.$sarch.rpm
```

Finally, a package containing the GoVivaceSR plugin should follow.

```
rpm -ivh unimrcp-govivace-sr-$universion-$packageversion.el$rhelversion.noarch.rpm
```

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.

```
/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*.

```
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="mrcpreorder" 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

Launch the UniMRCP server application.

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