

*Powered by Universal Speech Solutions LLC*



# Azure SS Plugin

## Administrator Guide

---

Revision: 8

Distribution: Red Hat / Cent OS

Created: November 11, 2017

Last updated: January 31, 2021

Author: Arsen Chaloyan

# Table of Contents

1 Overview.....	3
1.1 Applicable Versions.....	3
1.2 Supported Endpoints.....	3
1.3 Supported Distributions .....	3
1.4 Authentication.....	4
2 Installing RPMs Using YUM.....	5
2.1 Repository Configuration .....	5
2.2 Repository Verification.....	5
2.3 Azure SS Plugin Installation.....	6
3 Installing RPMs Manually .....	7
3.1 Package List.....	7
3.2 Package Installation Order.....	8
4 Obtaining License .....	9
4.1 License Type.....	9
4.2 Node Information.....	9
4.3 License Installation .....	9
5 Obtaining Service Credentials .....	10
5.1 Service Subscription .....	10
5.2 Installation of Credentials.....	10
Speech Service.....	10
Custom Speech Service .....	11
Speech Service Container .....	11
6 Configuring Server and Plugin .....	13
6.1 Plugin Factory Configuration .....	13
6.2 Logger Configuration .....	13
6.3 Azure SS Plugin Configuration .....	13
7 Validating Setup.....	14
7.1 Launching Server.....	14
7.2 Launching Client.....	14

# 1 Overview

This guide describes how to obtain and install binary packages for the Microsoft Azure Speech Synthesis (SS) 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 1.5.0 and above UniMRCP Azure SS Plugin 1.0.0 and above
------------------------------------------------------------------------------------------------------------------------------------------------------



## 1.2 Supported Endpoints

The plugin supports the following Speech Service endpoints.

Service Endpoint	Availability
Speech Service (regional)	Azure SS 1.5.0 and above
Custom Speech Service (regional)	Azure SS 1.7.0 and above
Speech Service (containerized)	Azure SS 1.12.0 and above
Custom Speech Service (containerized)	Azure SS 1.12.0 and above

## 1.3 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 7		
Red Hat / Cent OS 8		

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

## 1.4 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 Azure SS Plugin Installation

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

```
yum install unimrcp-azure-ss
```

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:

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

where

- *packagename* 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 Azure SS plugin.

Package Name	Description
<b>unimrcp-azure-ss</b>	Azure SS plugin to the server.
<b>unilibevent</b>	UniMRCP edition of the libevent library.
<b>umc-addons</b>	Sample en-US data files used with umc. [Optional]
<b>unilicnodegen</b>	Node information retrieval tool, required for license deployment.
<b>unimrcp-server</b>	Shared library and application of the server.
<b>unimrcp-client</b>	Shared libraries and sample applications of the client. [Optional]
<b>unimrcp-demo-plugins</b>	Set of demo plugins to the server. [Optional]
<b>unimrcp-common</b>	Data common for the client and the server.
<b>uniapr</b>	UniMRCP edition of the Apache Portable Runtime (APR) library.

<b>uniapr-util</b>	UniMRCP edition of the Apache Portable Runtime Utility (APR-Util) library.
<b>unisofia-sip</b>	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-$apuversion-$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 Azure SS plugin should follow.

```
rpm -ivh unimrcp-azure-ss-$universion-$packageversion.el$rhelversion.noarch.rpm
```



# 4 Obtaining License

The Azure SS 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](mailto: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 umsazuress_*.lic /opt/unimrcp/data
```

# 5 Obtaining Service Credentials



Instructions provided in this section are applicable to Azure SR 1.6.0 and above. Support for the old format remains in-tact, but is no longer documented.

In order to utilize either the deprecated Bing Speech API or the new Speech Service API, a corresponding service subscription key and an authentication endpoint need to be retrieved from the Microsoft Azure portal and further installed to the UniMRCP server.

## 5.1 Service Subscription

Navigate to the Microsoft Azure dashboard and create a new resource.

1. Navigate to the Dashboard.  
<https://portal.azure.com>
2. Create a new resource (+).
3. Type 'Speech' in the text box. (Note: for the deprecated Bing service, type 'Bing Speech')
4. Select and create the resource by filling out all the required parameters based on your needs.
5. Go to the created resource.
6. Collect one of the two keys (1) and the region-based authentication endpoint (2).

## 5.2 Installation of Credentials

Create a text file *cognitive.subscription.key* in the directory */opt/unimrcp/data*.

```
nano /opt/unimrcp/data/cognitive.subscription.key
```

Proceed with the instructions for one of:

- Speech Service (default)
- Custom Speech Service
- Speech Service Container

### Speech Service

Place the collected key and the authentication endpoint in the following JSON format. Leave the service endpoint empty by default.

```
{
  "auth-endpoint": "*****",
  "auth-key": "*****",
  "service-endpoint": ""
}
```

The field *auth-endpoint* is composed based on the following pattern.

Authentication Endpoint	URI
Speech Service (regional)	<a href="https://\$region.api.cognitive.microsoft.com/sts/v1.0/issuetoken">https://\$region.api.cognitive.microsoft.com/sts/v1.0/issuetoken</a>

The field *service-endpoint* is implicitly composed based on the following pattern.

Service Endpoint	URI
Speech Service (regional)	<a href="https://\$region.tts.speech.microsoft.com/cognitiveservices/v1">https://\$region.tts.speech.microsoft.com/cognitiveservices/v1</a>

Where *\$region* is one of *westus*, *eastus*, *westeurope* and others. See [Speech Service Regions](#).

## Custom Speech Service

Place the collected key and the authentication endpoint in the following JSON format. Note that the field *service-endpoint* is supported since Azure SS 1.7.0 and must be explicitly specified when a custom speech endpoint is used.

```
{
  "auth-endpoint": "*****",
  "auth-key": "*****",
  "service-endpoint": "*****"
}
```

## Speech Service Container

Speech Service containers are supported since Azure SS 1.12.0.

Install and run a container per instructions available in [this](#) guide. Collect and place the container service endpoint in the following JSON format. Leave the auth endpoint and key empty.

```
{
```

```
"auth-endpoint": "",
"auth-key": "",
"service-endpoint": "*****"
}
```

The field *service-endpoint* is composed based on the following pattern.

Service Endpoint	URI
Speech Service (containerized)	<a href="http://\$host:5000/speech/synthesize/cognitiveservices/v1">http://\$host:5000/speech/synthesize/cognitiveservices/v1</a>

# 6 Configuring Server and Plugin

## 6.1 Plugin Factory Configuration

In order to load the Azure SS 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 synthesizer 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="false"/>
  <engine id="Demo-Recog-1" name="demorecog" enable="true"/>
  <engine id="Demo-Verifier-1" name="demoverifier" enable="true"/>
  <engine id="Recorder-1" name="mrcprecorder" enable="true"/>
  <engine id="Azure-SS-1" name="umsazuress" enable="true"/>
</plugin-factory>
```

## 6.2 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="AZURESS-PLUGIN" priority="INFO" masking="NONE"/>
```

## 6.3 Azure SS Plugin Configuration

The configuration file of the plugin is located in */opt/unimrcp/conf/umsazuress.xml*. Default settings should be sufficient for generic use.

Refer to the *Usage Guide* for more information.

# 7 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.

## 7.1 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 [Azure-SS-1] [/opt/unimrcp/plugin/umsazuress.so]
```

Next, check for the license information.

```
[NOTICE] UniMRCP AZURESS License

-product name:  umsazuress
-product version: 1.0.0
-license owner:  -
-license type:  trial
-issue date:    2017-10-26
-exp date:     2017-11-25
-channel count: 2
-feature set:   0
```

## 7.2 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
```

Run a typical speech synthesis scenario by issuing the command *run bss1* from the console of the *umc* client application.

```
run bss1
```

This command sends a SPEAK request to the server and then records synthesized stream into a PCM file stored in the directory `/opt/unimrcp/var`.

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