Powered by Universal Speech Solutions LLC

UniMRCP

FreeSWITCH

Bing SR and SS

Usage Guide

Revision: 2

Created: October 30, 2017

Last updated: September 28, 2018

Author: Arsen Chaloyan

Table of Contents

1 Overview

This guide describes how to utilize the Microsoft Bing Speech services with FreeSWITCH.

Note that the FreeSWITCH and the UniMRCP server typically reside on different hosts in a LAN, although both might be installed on the same host.

Installation of the FreeSWITCH and the UniMRCP server with the BingSR and/or BingSS plugins is not covered in this document. Visit the corresponding web pages for more information.

https://freeswitch.org/confluence/display/FREESWITCH/mod_unimrcp
http://unimrcp.org/bingsr
http://unimrcp.org/bingss

1.1 Applicable Versions

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

FreeSWITCH 1.4 and above

UniMRCP BingSR Plugin 1.0.0 and above

UniMRCP BingSS Plugin 1.0.0 and above

2 UniMRCP Module

2.1 Overview

The module *mod_unimrcp.so* provides an implementation of the ASR and TTS interfaces of FreeSWITCH, based on the UniMRCP client library.

2.2 Configuration Steps

This section outlines major configuration steps required for use of the module *mod_unimrcp.so* with the UniMRCP server.

Create a new MRCP profile (or modify an existing one) in the configuration directory *mrcp_profiles* of FreeSWITCH. In the following example, the FreeSWITCH/UniMRCP client is located on 10.0.0.1 and the UniMRCP server is on 10.0.0.2.

```
<include>
  <!-- UniMRCP Server MRCPv2 -->
  <profile name="uni2" version="2">
    <!--param name="client-ext-ip" value="auto"-->
    <param name="client-ip" value="10.0.0.1"/>
    <param name="client-port" value="16090"/>
    <param name="server-ip" value="10.0.0.2"/>
    <param name="server-port" value="8060"/>
    <!--param name="force-destination" value="1"/-->
    <param name="sip-transport" value="udp"/>
    <!--param name="ua-name" value="FreeSWITCH"/-->
    <!--param name="sdp-origin" value="FreeSWITCH"/-->
    <!--param name="rtp-ext-ip" value="auto"/-->
    <param name="rtp-ip" value="auto"/>
    <param name="rtp-port-min" value="14000"/>
    <param name="rtp-port-max" value="15000"/>
    <!-- enable/disable rtcp support -->
    <param name="rtcp" value="0"/>
    <!-- rtcp bye policies (rtcp must be enabled first)
            0 - disable rtcp bye
            1 - send rtcp bye at the end of session
            2 - send rtcp bye also at the end of each talkspurt (input)
    -->
    <param name="rtcp-bye" value="2"/>
    <!-- rtcp transmission interval in msec (set 0 to disable) -->
    <param name="rtcp-tx-interval" value="5000"/>
    <!-- period (timeout) to check for new rtcp messages in msec (set 0 to disable) -->
    <param name="rtcp-rx-resolution" value="1000"/>
    <!--param name="playout-delay" value="50"/-->
    <!--param name="max-playout-delay" value="200"/-->
```

```
    <!--param name="ptime" value="20"/-->
    <param name="codecs" value="PCMU PCMA L16/96/8000"/>

    <!-- Add any default MRCP params for SPEAK requests here -->
    <synthparams>
    </synthparams>

    <!-- Add any default MRCP params for RECOGNIZE requests here -->
    <recogparams>
      <!--param name="start-input-timers" value="false"/-->
    </recogparams>
  </profile>
</include>
```

2.3 Usage Examples

Speech Recognition

Make use of a built-in speech grammar *transcribe* for recognition, by adding the following entry in the FreeSWITCH dialplan.

```
<action application="play_and_detect_speech" data="ivr/ivr-welcome_to_freeswitch.wav
detect:unimrcp:uni2 {start-input-timers=false}builtin:speech/transcribe"/>
```

Place a test call, listen to the file prompt and say something. Make sure recognition works as expected.

Speech Synthesis

Use the speak application for synthesis.

```
<action application="speak" data="unimrcp:uni2|BenjaminRUS|Welcome to FreeSWITCH"/>
```

Place a test call and listen to the synthesized message.

Speech Recognition and Synthesis

Play synthesized prompt and perform recognition.

```
<action application="set" data="tts_engine=unimrcp:uni2"/>
<action application="set" data="tts_voice=BenjaminRUS"/>
```

```
<action application="play_and_detect_speech" data=" say: Please say something detect:unimrcp:uni2 {start-input-timers=false}builtin:speech/transcribe"/>
```

Place a test call, listen to the synthesized prompt and say something. Make sure recognition works as expected.