



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

SIP Endpoint SDK Developer's Guide

Configuring SIP Endpoint SDK for .NET

12/14/2025

Contents

- 1 Configuring SIP Endpoint SDK for .NET
 - 1.1 Basic Configuration Settings
 - 1.2 Advanced Settings and Configuration Options

Configuring SIP Endpoint SDK for .NET

Basic Configuration Settings

You may want to set certain basic SIP Endpoint configuration options in your application. The following sections contain information that may be helpful in doing that.

Most configuration settings described on this page can be found and changed in the `SipEndpoint.config` file that is included with your SIP Endpoint SDK installation. For more information about this file, see [Default Config Settings](#).

Timeout Issue

Note that the `reg_timeout` setting is valid only if it contains a numeric value. If the supplied value is a non-numeric value or is empty or null, it will be interpreted as 0 and the endpoint will operate in standalone mode.

IPv6

As of release 8.1.2, SIP Endpoint SDK for .NET supports TCP/IP version 6 (IPv6) on Windows Vista and higher.

The local address must be set to IPv6:

- By means of an explicit address specification (as a hostname resolving to IPv6 only, or by the use of a literal IP address)
- By setting the `ip-versions` option to favor IPv6
- Or by only having an IPv6 interface available on the host

The SIP proxy address must be either a hostname that primarily resolves to IPv6 or a literal IPv6 address. Note that the preferences in the `ip-versions` option do not currently affect external address resolution.

Note: IPv6 must be configured on both ends of a session — mixed IPv4 and IPv6 environments are not supported.

This feature is resolved at Endpoint creation time by applying the following configuration to the SIP Provider.

Settings:

```
<setting name="public_address" value="Address"/>
<setting name="ip_versions" value="ipv4,ipv6"/>
```

QoS Bits

Starting with Windows Vista, Quality of Service (QoS) settings are supported on the level of the Windows operating system. This support allows you to configure policies to mark all of the packets sent by a particular application with the desired level of service.

Microsoft has documented this support at <http://technet.microsoft.com/library/dd919203.aspx> and at <http://technet.microsoft.com/en-us/magazine/2007.02.cableguy.aspx>.

Diagnostics

In order to diagnose production issues, it is important to have access to diagnostic information that relates to your SIP endpoint and its environment. Because of this, the SIP Endpoint SDK provides logging with 5 possible logging levels.

Valid values for the SIP Endpoint SDK logging levels are:

```
Fatal = 0
Error = 1
Warning = 2
Info = 3
Debug = 4

<domain name="system">
  <section name="diagnostics">
    <setting name="log_level" value="from 0 to 4"/>
  </section>
</domain>
```

Endpoint DNs

The following fields of DN objects can be configured under the SIP Server Switch:

- Number — Contains a numeric-only DN number. This number can be used to dial from a phone to reach this DN directly. The "@" character and domain names are not allowed. This number constitutes the username part of the endpoint's Address of Record (AOR).
- Type — Should be set to "Extension".
- Annex TServer/refer-enabled — Valid values are "true" or "false":
 - "true" indicates that REFER can be used to make a call or a transfer. (This value is supported by the SIP Endpoint SDK starting in release 8.0.000.11.)
 - "false" indicates that REFER is disabled for the above usage.
- Annex TServer/sip-cti-control — Valid values are "talk", "hold", "dtmf":
 - "talk" and "hold" are used to support the talk and hold BroadSoft extensions.
 - "dtmf" is used to support the sending of DTMF tones during third-party call control of an active call.
 - **Note:** As of SIP Endpoint SDK 8.1.2 for .NET, beep tone generation is no longer supported.

Audio Device Configuration

You can use SIP Endpoint SDK for .NET to configure audio devices such as headsets and microphones.

If you are using Windows 7, Windows 8, or Windows 2008 Server, this includes the ability to use devices with long device names.

The following settings can be used to control the behavior of these devices:

```
<section name="device">
  <setting name="use_headset" value="1or0"/>
  <setting name="headset_name" value="Sennheiser USB"/>
  <setting name="audio_in_device" value="Sennheiser USB"/>
  <setting name="audio_out_device" value="Sennheiser USB"/>
</section>

<section name="session">
  <setting name="reject_session_when_headset_na" value="1or0"/>
  <setting name="sip_code_when_headset_na" value="480"/>
  <setting name="auto_answer" value="1or0"/>
<setting name="auto_accept_video" value="1or0"/>
</section>
```

The endpoint will automatically accept or reject a session, depending on whether or not a headset is available.

Configuration Notes

- If use_headset is set to 0 then the SDK will try to configure audio devices with the names that have been set in the audio_in_device and audio_out_device settings.
- If the configuration is not successfully completed, the default system devices will be selected.

Answer Actions and Dial Actions

The following tables describe answer actions and dial actions based on various combinations of settings.

Headset Availability

Use this table to determine what effect your headset settings will have on answer actions and dial actions. The last line indicates whether the action logic will consider the headset to be available or unavailable, or whether it will consider its state to be irrelevant.

Setting/State	Value		
use_headset	Yes	Yes	No
headset_name	Correctly specified	Incorrectly specified	Not applicable
Headset plugged in	Yes	Yes or No	Yes or No
Headset available	Yes	No	Irrelevant

Answer Actions

Depending on headset availability, as indicated in the previous table, the SDK provides the following options for answering an incoming session.

Setting/State	Value	
auto_answer	Yes	No

Setting/State		Value			
reject_session_when_headset_na	No	Yes	-	-	
sip_code_when_headset_na	-	Code	-	-	
Headset Available	Irrelevant	No	No	Yes	-
Answer	Auto	Auto	Reject	Auto	Manual

- If a headset is available, the session is answered as video or audio, depending on the media offer and the value of the auto_accept_video setting.
- If a headset is not available, the session is rejected with the SIP error code message specified in sip_code_when_headset_na.

Dial Actions

Outgoing sessions can be dialed using the headset availability criteria shown here, which are based on the inputs described in the Headset Availability table that appears above:

Setting/State		Value	
Headset Available	Yes	No	Irrelevant
Dial	Dial	Error	Dial

Defaults

If an endpoint is not configured in the configuration file, the following defaults will be used:

- sip_code_when_headset_na=480
- auto_answer=0
- use_headset=0

Ringer Device Configuration

Starting with SIP Endpoint SDK for .NET 8.5.200.57, you can configure ringer devices using the ringer_device and ringing_enabled settings:

```
<domain name="policy">
  <section name="device">
    <!-- Ringer -->
    <setting name="ringer_device" value="RingerDeviceName-HighPriority"/>
    <setting name="ringer_device" value="RingerDeviceName-LowPriority"/>
    . . .

  <section name="session">
    <!--
    settings "ringing_enabled" valid values:
    0 Ringing disabled
    1 event Ringing enabled (default value)
    2 built-in Ringing playback enabled
    3 built-in Ringing playback and event Ringing enabled
    NEW 4 built-in Ringing playback enabled using Ringer device
    NEW 5 built-in Ringing playback using Ringer device and event Ringing enabled
    -->
```

```
<setting name="ringing_enabled" value="1"/>
. . .
```

The SDK automatically chooses the ringer device according to configuration at startup and whenever the list of speaker devices changes. The procedure is the same as for audio devices. First, the SDK tries to find the device from the `ringer_device` setting. If this device is not found, the SDK uses the first system audio out device.

You can also configure the ringer device using the following methods for interface `IExtendedService`:

- `GsStatus SetRingerDevice(String^ name);`—use this method to programmatically set the ringer device.
- `String^ GetRingerName();`—returns the current ringer device name.

Advanced Settings and Configuration Options

The **default configuration file** does not contain all settings that may be used with the SIP Endpoint SDK; additional settings can be added to change certain behaviors. Some of these additional settings are discussed below.

Headset Connectivity Notification

The SIP Endpoint SDK only supports headset monitoring when the device is explicitly defined in the `SipEndpoint.config` file, as shown below:

```
<domain name="policy">
  <section name="device">
    <!-- Headset -->
    <setting name="use_headset" value="1"/>
    <setting name="headset_name" value="HeadsetName"/>
  </section>
</domain>
```

In this case, headset connectivity is fully supported and the application can receive events indicating that the device has been plugged or unplugged.

It is also possible to use your application to select audio devices automatically, by using the SIP Endpoint SDK API to look for IN and OUT devices. Using the `GetAllSystemAudioInDevices()` and `GetAllSystemAudioOutDevices()` methods allows you to check for attached devices, which can then be used to update the following settings before initializing the SIP Endpoint SDK.

```
<domain name="policy">
  <section name="device">
    <!-- Headset -->
    <setting name="use_headset" value="0"/>

    <!-- Mic -->
    <setting name="audio_in_device" value="MicName"/>

    <!-- Speaker -->
    <setting name="audio_out_device" value="SpeakerName"/>
  </section>
</domain>
```

However, the plugged or unplugged states are not monitored for the selected device in this case. If no devices are defined in the `SipEndpoint.config` file, then the SIP Endpoint SDK uses a default procedure to determine what device is attached, and will not monitor for plugged or unplugged states.

SIP Messaging and Media Encryption

SIP Endpoint SDK has settings that allow you to encrypt SIP messaging and the media channel. The SIP messaging can be encrypted using TLS, while the media channel can be encrypted using SRTP. The following diagram shows the SIP Endpoint SDK architecture with these features enabled.

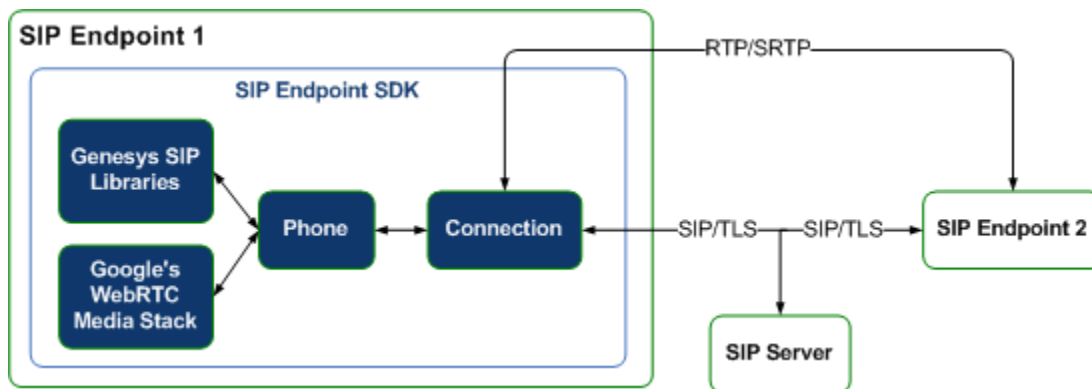


Figure 1: SIP Endpoint SDK Architecture with TLS and SRTP Enabled

To enable TLS support, set the protocol to `tls`, as shown here. The outgoing SIP messaging will be encrypted:

```
<Container name ="Basic">
  <Connectivity user ="DN" server="SipServerHost:Port" protocol="tls"/>
</Container>
...
<domain name="system">
  <section name="security">
    <setting name="cert_file" value="ValueOfCertificateThumbprint"/>
    <setting name="use_srtp" value="enabled"/>
  </section>
</domain>
```

Use the `use_srtp` setting to control media channel encoding. The setting `use_srtp` has the following valid values:

- `disabled`—SRTP is disabled (default)
- `optional` or `allowed`—SRTP is allowed
- `elective` or `both`—SRTP is allowed in both directions
- `force` or `mandatory`—SRTP is forced
- `enabled`—SRTP is enabled

Digest Authentication Support

SIP Endpoint SDK supports the RFC2617-style digest authentication that is currently used by SIP Server. This authentication is triggered by using the following DN configuration options: password and authenticate-requests. If these options are configured for a specific DN, then SIP Server enforces the authentication mechanism for that DN. During registration, the SIP Endpoint receives a 401 Unauthorized SIP challenge message request from SIP Server. The SIP Endpoint should then provide the encrypted password, along with the Digest username, in the next REGISTER message.

At this point, SIP Server compares the received password to the password associated with the DN object defined in Config Server. If the passwords match, then the SIP Endpoint can be registered and may proceed to access the call functionality provided by SIP Server. SIP Endpoint SDK provides the following method to set the password for a particular connection ID. This method has been added to the IExtendedService interface in the Genesyslab.Sip.Endpoint.Provider.Genesys namespace:

```
void SetPassword(int connectionId, String^ password);
```

Producing RTCP Extended Reports

You can use SIP Endpoint SDK to produce RTCP Extended Reports ([RFC 3611](#)) and publish them according to [RFC 6035](#) at the end of each call, using a collector address of your choice.

Important

The publish message is sent to the specified collector address only if the `vq_report_collector` parameter is configured with the `user@server:port;transport=udp` format. For example, `collector@127.0.0.1:5160;transport=udp`.

Settings:

```
<domain name="policy">
<section name="endpoint">
...
<!--
Valid values for Voice Quality (VQ) report publish setting (vq_report_publish):
0--VQ report is not published
1--VQ report is published to the collector at the end of the call--
    see the vq_report_collector setting information below
-->
<setting name="vq_report_publish" value="0"/>
<!--
Valid values for Voice Quality (VQ) report collector setting (vq_report_collector):
NULL or Empty--The VQ report is published to the proxy described in the
Connectivity section
FQDN or IP address along with port and transport--
    collector@SipServer.genesyslab.com:5060;transport=udp
-->
<setting name="vq_report_collector"
value="collector@SipServer.genesyslab.com:5060;transport=udp"/>
</section>
```

Endpoint:

The `vq_report_publish` and `vq_report_collector` settings can be read from the Endpoint Policy by using the following methods:

```
GetEndpointPolicy(EndpointPolicyQuery.VqReportCollector);
GetEndpointPolicy(EndpointPolicyQuery.VqReportPublish);
```

Audio Layer Selection

SIP Endpoint SDK for .NET release 8.1.2 and higher allows you to select an audio layer for WebRTC using the Windows environment variable `GCTI_AUDIO_LAYER`. If this variable is set to 1, SIP Endpoint SDK will use the Windows Wave audio layer. Otherwise, it will use the Windows Core audio layer. You can set this variable at Computer -> Properties -> Advanced System Settings -> Environment Variables.

refer-enabled Setting

This is a SIP Server setting configured under the SIPS or DN Object. Set to `true` to enable SIP REFER.

SIP::INFO Message Exchange

SIP Endpoint SDK can create and transmit In-Dialog SIP:INFO messages using updated Content-Type and Content headers. It can also receive messages with customer-provided information in the Content-Type and Content headers.

The following method has been added to the `ICallControl` interface in order to create and transmit an In-Dialog SIP:INFO message::

```
void SendSipInfo(int sessionId, String^ contentType, String^ content);
```

The following event has been added to Session Manager:

```
public event EventHandler<SessionEventArgs> SipInfoReceived;
```

The `SessionEvent` properties now also have these key value pairs:

```
Properties["ContentType"].ToString();
Properties["Content"].ToString();
```

Working with Codec Priorities

Codecs are listed by name in the codecs domain of the configuration file. Codecs are listed by priority with codecs closer to the top of the list having higher priority. To disable a codec, comment out or remove the codec from the file.

The setting `payload_type` is an integer. For codec *h264* the valid values are between 96 and 127.

The setting `fntp` is a string with valid values of RFC6184 for codec *h264* and RFC3555 for codec *g729*.

The codec *ulpfec/90000* supports the *vp8* codec with forward error correction.

Example

```
<domain name="codecs">
  <section name="PCMU/8000">
    <setting name="payload_type" value="0"/>
  </section>
  <section name="PCMA/8000">
    <setting name="payload_type" value="8"/>
  </section>
  <section name="G722/16000">
    <setting name="payload_type" value="9"/>
  </section>
  <section name="iLBC/8000">
    <setting name="payload_type" value="102"/>
  </section>
  <section name="iSAC/32000">
    <setting name="payload_type" value="104"/>
  </section>
  <section name="iSAC/16000">
    <setting name="payload_type" value="103"/>
  </section>
  <section name="vp8">
    <setting name="payload_type" value="100"/>
  </section>
  <section name="g729/8000">
    <setting name="payload_type" value="18"/>
    <setting name="fmtp" value="annexb=yes"/>
  </section>
  <section name="h264">
    <setting name="payload_type" value="108"/>
    <setting name="fmtp" value="profile-level-id=420028"/>
  </section>
  <section name="vp9">
    <setting name="payload_type" value="101"/>
  </section>
  <section name="ulpfec/90000">
    <setting name="payload_type" value="97"/>
  </section>
  <section name="opus/48000/2">
    <setting name="payload_type" value="120"/>
  </section>
</domain>
```

Tip

For information on which codecs are supported by SIP Endpoint SDK for .NET, see the list of [Supported Codecs](#) in the SIP Endpoint SDK Overview.

Configuring Capture Devices

In order to define the list of preferred capture devices and their priorities, the devices should be added to the device policy section in the **SipEndpoint.config** file:

```
<domain name="policy">
```

```
<section name="device">
<!--
  The priority of a device depends on its position in this section.
  The higher the position, the higher device priority.
  To disable a device it should be commented out or removed from the file.
-->
. . .
  <!-- Capture -->
  <setting name="capture_device" value="CaptureDeviceName-HighPriority"/>
  <setting name="capture_device" value="CaptureDeviceName-LowPriority"/>
</section>
</domain>
```

API

The `IVideoControl` Interface has been extended with a method that allows you to set up a capture device based on:

- Its configuration settings in the **policy:device** section
- Its priority in that section, as described above

This method can be called at any time while SIP Endpoint SDK is active in order to carry out this task, as shown here:

```
GsStatus SetCaptureDeviceFromConfig();
```

Application / SDK Interaction

The workflow between your app and the SIP Endpoint SDK is described in the following table:

App	SDK
Call: <code>ReleaseCaptureDevice()</code>	Releases currently used capture device
Call: <code>SetCaptureDeviceFromConfig()</code>	Finds matching capture device from the configuration file with the device from the array of all capture devices; if found, it sets this capture device as active
Call: <code>String^ GetCurrentCaptureDeviceName()</code>	Returns currently allocated capture device name
Compare with prioritized capture device name	