# Using the OpenTok connection and stream quality API

(Confidential and proprietary)


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

The OpenTok client libraries for JavaScript, Android, and iOS include APIs for checking network quality before connecting to a session and while connected to the a session (by checking the network usage of a subscriber's stream).

These are beta versions of these APIs, not documented publicly, and the details in this document are confidential and proprietary.

This document includes the following sections:




*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

# OpenTok Android SDK

Version 2.4 of the OpenTok Android SDK includes undocumented beta API additions for the following:

- Checking the connection quality before you connect to an OpenTok session:

  ```
  Session.testNetwork(String token, NetworkTestListener listener)
  NetworkTestListener
  ```

- Checking the quality of a stream the client is subscribed to:

  ```
  Subscriber.setAudioStatsListener(AudioStatsListener listener)
  Subscriber.setVideoStatsListener(VideoStatsListener listener)
  ```

## Session.testNetwork(String token, NetworkTestListener listener)

Tests the client network connection quality, reporting the following:

- The download bandwidth available to the client
- The upload bandwidth available to the client
- the ratio of lost packets per total packets sent
- The roundtrip time for media data to be sent and received

The test takes from 11 - 17 seconds, at which point the `onNetworkTestCompleted(Session session, SessionStats stats)` method of the `listener` object (passed into `Session.testNetwork()`) is called.

The SessionStats object (passed into the `onNetworkTestCompleted()` method) includes properties that define the network connection statistics. The results represent an estimate of network conditions at the time of testing.

If you call the `Session.testNetwork()` method, do so before you connect to the session (otherwise the call results in an error).

You may want to create a Publisher object with a frame rate and dimensions based on the results of the call to `Session.testNetwork()`. See Interpreting pre-connection quality statistics at the end of this document.


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

### Parameters

- `token` — A valid OpenTok token for the Session
- `listener` — The object that implements the NetworkTestListener interface.

If there is an error, the `onError(Session session, OpentokError error)` method of the object that implements NetworkTestListener is called. The error codes, defined in the ErrorCode enum in the OpentokError class, include the following:

| ***ErrorCode constant*** | ***Description*** |
|---|---|
| `AuthorizationFailure` | Authentication error — invalid token or API key. |
| `InvalidSessionId` | Invalid session ID. |
| `ConnectionFailed` | Connection to the HTTP server failed. |
| `SessionNullOrInvalidParameter` | No publisher (or invalid publisher) specified; or no token specified. |
| *TBD (1015)* | Session connected; cannot test network. |
| `SessionConnectionTimeout` | Request timeout |
| `SessionBlockedCountry` | Unable to connect to the session — terms of service violation: export compliance |
| `NoMessagingServer` | No TURN server found |
| *TBD (1553)* | ICE workflow failed — This can occur if the client does not have a camera available (and for other reasons) |
| `SubscriberWebRTCError` | WebRTC error |
| `SessionUnexpectedGetSession InfoResponse` | Unexpected server response |

*Note:* Some of these error conditions are not supported in the current beta version of the API.

The SessionStats object passed into the `onNetworkTestCompleted()`) method has the following properties:


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

- `downloadBitsPerSecond` (int) — The — The representative download bandwidth available to the client, in bits per second

- `packetLossRatio` — The  ratio of lost packets per total packets sent (between 0 and 1, inclusive)

- `roundTripTimeMilliseconds` — The  roundtrip time for media data to be sent and received, in milliseconds

- `uploadBitsPerSecond` (int) — The representative upload bandwidth available to the client, in bits per second


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

## Example

The following code checks the network quality before you connect to a session. It then uses the quality statistics to determine the settings for publisher based on the information in the Interpreting pre-connection quality statistics tables at the end of this document::

```
private boolean supportsPublishing = true;
private short publisherResolutionX;
private short publisherResolutionY;
private boolean publishVideo = true;

session.testNetwork(token, new NetworkTestListener() {
  @Override
  public void onNetworkTestCompleted(Session session,
                    SessionStats stats) {
    if (stats.packetLossRatio > 0.05 ||
        stats.roundTripTimeMilliseconds > 500 ||
        stats.uploadBitsPerSecond < 50 * 1000) {
      supportsPublishing = false;
    } else if (stats.packetLossRatio > 0.03 ||
        stats.uploadBitsPerSecond < 150 * 1000) {
      publishVideo = false;
    } else if (stats.uploadBitsPerSecond > 350 * 1000) {
      publisherResolutionX = 1280;
      publisherResolutionY = 720;
    } else if (stats.uploadBitsPerSecond > 250 * 1000) {
      publisherResolutionX = 640;
      publisherResolutionY = 480;
    } else {
      publisherResolutionX = 320;
      publisherResolutionY = 240;
    }

    if (supportsPublishing) {
      // Create an instance of PublisherKit based on
      // publisherResolutionX, publisherResolutionY, and publishVideo
      // settings.
    }
    // You can now connect to the session and
    // (if supportsPublishing == true) publish.
}
```


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

```
@Override
public void onError(Session session, OpentokError error) {
  Log.i(LOGTAG, "testNetwork error: " + exception.getMessage);
});
```

## SubscriberKit.setAudioStatsListener(AudioStatsListener listener)

Sets up a listener for subscriber audio statistics. The listener object implements the `onAudioStats(SubscriberKit subscriber, SubscriberAudioStats stats)` method of the AudioStatsListener interface. This method is called periodically to report the following:

- Total audio packets lost
- Total audio packets received
- Total audio bytes received

Note that you can only use this method on a subscriber that is subscribing to another client's stream (not to a stream that the local client is publishing).

### Parameters

The method has one parameter — `listener` — which is method that implements the `onAudioStats(SubscriberKit subscriber, SubscriberAudioStats stats)` method defined by the AudioStatsListener interface. This method is called periodically to to report audio statistics, which are passed in as the `stats` parameter. This `stats` object, defined by the SubscriberAudioStats class, has the following properties:

- `audioBytesReceived` — (int) The total audio bytes received by the subscriber
- `audioPacketsLost` — (int) The total audio packets that did not reach the subscriber
- `audioPacketsReceived` — (int) The total audio packets received by the subscriber
- `timestamp` — (double) The timestamp, in milliseconds since the Unix epoch, for when these stats were gathered


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

## SubscriberKit.setVideoStatsListener(VideoStatsListener listener)

Sets up a listener for subscriber video statistics. The listener object implements the `onVideoStats(SubscriberKit subscriber, SubscriberVideoStats stats)` method of the VideoStatsListener interface. This method is called periodically to report the following:

- Total video packets lost
- Total video packets received
- Total video bytes received

Note that you can only use this method on a subscriber that is subscribing to another client's stream (not to a stream that the local client is publishing).

### Parameters

The method has one parameter — `listener` — which is method that implements the `onVideoStats(SubscriberKit subscriber, SubscriberVideoStats stats)` method defined by the VideoStatsListener interface. This method is called periodically to to report video statistics, which are passed in as the `stats` parameter. This `stats` object, defined by the SubscriberAudioStats class, has the following properties:

- `videoBytesReceived` — (int) The total video bytes received by the subscriber
- `videoPacketsLost` — (int) The total video packets that did not reach the subscriber
- `videoPacketsReceived` — (int) The total video packets received by the subscriber
- `timestamp` — (double) The timestamp, in milliseconds since the Unix epoch, for when these stats were gathered


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

## Example

The following code checks the connection of a subscriber's stream every 5 seconds and calculates the bytes received per second and packet loss ratios for both audio and video:

```
SubscriberVideoStats lastVideoStats;
SubscriberVideoStats lastAudioStats;
Subscriber mSubscriber;
double audioBytesReceivedPerSecond;
double audioPacketLossRatio;
double videoBytesReceivedPerSecond;
double videoPacketLossRatio;

@Override
public void onStreamReceived(Session session, Stream stream) {
     mSubscriber = new Subscriber(this, stream);
     mSubscriber.setVideoStatsListener(this);
     mSubscriber.setAudioStatsListener(this);
     mSession.subscribe(mSubscriber);
}

@Override
onAudioStats(SubscriberKit subscriber, SubscriberAudioStats stats) {
    if (lastAudioStats) {
        audioBytesReceivedPerSecond =
         (stats.audioBytesReceived -
          lastAudioStats.audioBytesReceived) /
         (stats.timestamp - lastAudioStats.timestamp);
        audioPacketLossRatio =
          (stats.audioPacketsLost -
           lastAudioStats.audioPacketsLost) /
          (stats.audioPacketsReceived -
           lastAudioStats.audioPacketsReceived);
        }
    lastAudioStats = stats;
});
```


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

```
@Override
onAudioStats(SubscriberKit subscriber, SubscriberAudioStats stats) {
    if (lastVideoStats) {
        videoBytesReceivedPerSecond =
         (stats.videoBytesReceived -
          lastVideoStats.videoBytesReceived) /
         (stats.timestamp - lastVideoStats.timestamp);
        videoPacketLossRatio =
          (stats.videoPacketsLost -
           lastVideoStats.videoPacketsLost) /
          (stats.videoPacketsReceived -
           lastVideoStats.videoPacketsReceived);
    }
    lastVideoStats = stats;
});
```


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

# OpenTok iOS SDK

Version 2.4 of the OpenTok iOS SDK includes undocumented beta API additions for the following:

- Checking the connection quality before you connect to an OpenTok session:

```
[OTSession testNetworkWithToken:error:]
[OTSessionDelegate session: networkTestCompletedWithResult:]
```

- Checking the quality of a stream the client is subscribed to:

```
Subscriber.setAudioStatsListener(AudioStatsListener listener)
Subscriber.setVideoStatsListener(VideoStatsListener listener)
```

## [OTSession testNetworkWithToken: error:]

Tests the client network connection quality, reporting the following:

- The download bandwidth available to the client
- The upload bandwidth available to the client
- the ratio of lost packets per total packets sent
- The roundtrip time for media data to be sent and received

The test takes from 11 to 17 seconds, at which point the `[OTSessionDelegate session: networkTestCompletedWithResult:]` message is sent. The results represent an estimate of network conditions at the time of testing.

If you call `[OTSession testNetworkWithToken: error:]`, do so before you connect to the session (otherwise the call results in an error).

You may want to create a Publisher object with frame-rate and dimensions based on the results of the test. See Interpreting pre-connection quality statistics at the end of this document.


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

### Parameters

**token** (NSString) — A valid OpenTok token for the session.

**error** (OTError) — If there is a synchronous error in calling this method, this is set to an OTError object.

If there is an asynchronous error, the [`OTSessionDelegate didFailWithError:`] method is called. The error codes, defined in the OTSessionErrorCode enum, include the following:

| ***OTSessionErrorCode constant*** | ***Description*** |
|---|---|
| `OTAuthorizationFailure` | Authentication error — invalid token or API key. |
| `OTErrorInvalidSession` | Invalid session ID. |
| `OTConnectionFailed` | Connection to the HTTP server failed. |
| `OTNullOrInvalidParameter` | No publisher (or invalid publisher) specified; or no token specified. |
| *TBD (1015)* | Session connected; cannot test network. |
| `OTSessionConnectionTimeout` | Request timeout |
| *TBD (1026)* | Unable to connect to the session — terms of service violation: export compliance |
| `OTNoMessagingServer` | No TURN server found |
| *TBD (1553)* | ICE workflow failed — This can occur if the client does not have a camera available (and for other reasons) |
| `OTSubscriberWebRTCError` | WebRTC error |
| *TBD (2001)* | Unexpected server response |

*Note:* Some of these error conditions are not supported in the current beta version of the API.


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

### Example

The following code checks the network quality before you connect to a session:

```
- (void)testConnection
{
    OTError* error = nil;
    [mySession testNetworkWithToken:myToken error:&error];

    if (error) {
        NSLog(@"Cannot test network!");
    }
}

- (void)session:(OTSession*)session
networkTestCompletedWithResult:(OTSessionNetworkStats*)result
{
    NSLog(@"upbw: %f", result.uploadBitsPerSecond);
    NSLog(@"dnbw: %f", result.downloadBitsPerSecond);
    NSLog(@"plr: %f", result.packetLossRatio);
    NSLog(@"rtt: %f", result.latencyMilliseconds);
    // Now you can connect to the session.
}
```


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

## [OTSessionDelegate session: networkTestCompletedWithResult:]

Sent when the connection test is completed in response to the call to `[OTSession testNetworkWithToken: error:]`.

The properties of the result parameter define the connection quality statistics

### Parameters

**session** (OTSession) — The session the test results apply to

**result** (OTSessionNetworkStats) — An object with the following properties, defining the connection quality:

- `downloadBitsPerSecond` (double) — The representative download bandwidth available to the client, in bits per second.
- `packetLossRatio` (double) — The ratio of lost packets per total packets sent (between 0 and 1, inclusive)
- `roundTripTimeMilliseconds` (double) — The roundtrip time for media data to be sent and received, in milliseconds
- `uploadBitsPerSecond` (double) — The representative download bandwidth available to the client, in bits per second


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

## Checking the connection quality for a subscriber

In the OpenTok iOS SDK version 2.4, use the `[OTSubscriberKit setNetworkStatsDelegate:]` to get the following statistics for a subscriber's stream:

- The total audio and video packets lost
- The total audio and video packets received
- The total audio and video bytes received

You may use these statistics to have a subscriber subscribe to audio-only (if the audio packet loss reaches a certain threshold). If you choose to do this, for each stream you publish, you should call the `[OTPublisherKit setAudioFallbackEnabled:]` method, passing in `NO`, for each publisher you use. This prevents the OpenTok Media Router from using its own audio-only toggling implementation.

### [OTSubscriberKit setNetworkStatsDelegate:]

Sets up a delegate object for subscriber quality statistics. This object implements the OTSubscriberKitNetworkStatsDelegate protocol. This object is sent messages reporting the following:

- Total audio and video packets lost
- Total audio and video packets received
- Total audio and video bytes received

Note that you can only use this method on a subscriber that is subscribing to another client's stream (not to a stream that the local client is publishing).

### [OTSubscriberKitNetworkStatsDelegate subscriber: audioNetworkStatsUpdated:]

This message is sent periodically to report audio statistics for the subscriber. The second parameter, stats, which is defined by the OTSubscriberKitAudioNetworkStats interface, includes the following properties:

- `audioBytesReceived` (uint64_t) — The total number of audio bytes received by the subscriber
- `audioPacketsLost` (uint64_t) — The total number of audio packetsthat did not reach the subscriber
- `audioPacketsReceived` (uint64_t) — The total number of audio packets received by the subscriber
- `timestamp` (double) — The timestamp, in milliseconds since the Unix epoch, for when these stats were gathered


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

### [OTSubscriberKitNetworkStatsDelegate subscriber: videoNetworkStatsUpdated:]

This message is sent periodically to report video statistics for the subscriber. The second parameter, `stats`, which is defined by the OTSubscriberKitVideoNetworkStats interface, includes the following properties:

- `videoBytesReceived` (uint64_t) — The total number of video bytes received by the subscriber

- `videoPacketsLost` (uint64_t) — The total number of video packets lost by the subscriber

- `videoPacketsReceived` (uint64_t) — The total number of video packets received by the subscriber

- `timestamp` (double) — The timestamp, in milliseconds since the Unix epoch, for when these stats were gathered


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

### Example

```
- (void)subscribeToStream:(OTStream*)stream {
    if (doSubscribe && nil == mySubscriber) {
        mySubscriber = [[OTSubscriber alloc] initWithStream:stream
                                                   delegate:self];
        [mySubscriber setNetworkStatsDelegate:self];
        [mySession subscribe:mySubscriber];

        [mySubscriber.view setFrame:CGRectMake(0, 240, 320, 240)];
        [[self view] insertSubview:mySubscriber.view];
    }
}

- (void)subscriber:(OTSubscriberKit *)subscriber
videoNetworkStatsUpdated:(OTSubscriberKitVideoNetworkStats *)stats
{
    NSLog(@"Stats report received; %d", stats.timestamp);
    NSLog(@"Video packets lost: %d", result.videoPacketsLost);
    NSLog(@"Video packets received: %d",
result.videoPacketsReceived);
    NSLog(@"Video bytes received: %d", result.videoBytesReceived);
}

- (void)subscriber:(OTSubscriberKit *)subscriber
audioNetworkStatsUpdated:(OTSubscriberKitAudioNetworkStats *)stats
{
    NSLog(@"Stats report received; %d", stats.timestamp);
    NSLog(@"Audio packets lost: %f", result.audioPacketsLost);
    NSLog(@"Audio packets received: %f",
result.audioPacketsReceived);
    NSLog(@"Audio bytes received: %f", result.audioBytesReceived);
}
```


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

# OpenTok.js

Version 2.4 of OpenTok.js includes undocumented methods for the following:

- Checking the connection quality before you connect to an OpenTok session — `Session.testNetwork()`
- Checking the quality of a stream the client is subscribed to — `Subscriber.getStats()`

## Session.testNetwork(token, publisher, completionHandler)

Tests the client network connection quality, reporting the following:

- The download bandwidth available to the client
- The upload bandwidth available to the client
- The ratio of lost packets per total packets sent
- The roundtrip time for media data to be sent and received

The test begins once the user grants the Publisher object access to the camera and microphone. (If the user denies access, the completion handler is called with an error.) The test takes from 11 to 17 seconds, at which point the completion handler is called. The results are set as properties of the `stats` object of the completion handler, and they represent an estimate of network conditions at the time of testing.

If you call this method, you must do so before you connect to the session (otherwise the call results in an error).

You may want to create a new Publisher object optimized to your network conditions based on the results of the call to `Session.testNetwork()`. As an example, you can configure the Publisher to suitable frame-rate and video resolutions or control the media that you want to share (audio, audio and video, screen sharing, etc.). Your network conditions might change if you don't publish/subscribe after performing the network test. See Interpreting pre-connection quality statistics at the end of this document.


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

### Parameters

**`token`** — A valid OpenTok token for the Session

**`publisher`** — A Publisher object (see the documentation for OT.initPublisher())

**`completionHandler`** — A method that takes two parameters:

- `error` — Upon successful completion of the network test, this is undefined. Otherwise (on error), this property is set to an object with the following properties:
  - `code` — The error code
  - `message` — A description of the error

| | |
|---|---|
| 1004 | Authentication error — invalid token or API key. |
| 1005 | Invalid session ID. |
| 1006 | Connection to the HTTP server failed. |
| 1011 | No publisher (or invalid publisher) specified; or no token specified. |
| 1015 | Session connected; cannot test network. |
| 1021 | Request timeout |
| 1026 | Unable to connect to the session — terms of service violation: export compliance |
| 1503 | No TURN server found |
| 1553 | ICE workflow failed — This can occur if the client does not have a camera available (and for other reasons) |
| 1600 | WebRTC error |
| 2001 | Unexpected server response |

*Note:* Some of these error conditions are not supported in the current beta version of the API.


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

- `stats` — An object with the following properties:

  `downloadBitsPerSecond` — The representative download bandwidth available to the client, in bits per second.

  `packetLossRatio` — The ratio of lost packets per total packets sent (between 0 and 1, inclusive)

  `roundTripTimeMilliseconds` — The  roundtrip time for media data to be sent and received, in milliseconds

  `uploadBitsPerSecond` — The  representative upload bandwidth available to the client, in bits per second


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

### Example

The following code checks the network quality before you connect to a session. It then uses the quality statistics to determine the resolution of a publisher (or whether to publish audio only) based on the information in the Interpreting pre-connection quality statistics tables at the end of this document:

```
var apiKey = '',  // Replace with your OpenTok API key
  sessionId = '', // Replace with the OpenTok session ID
  token = '',     // Replace with a valid token for the session
  publisherOptions = {},
  supportsPublishing = true,
  publisher = OT.initPublisher(),
  session = OT.initSession(apiKey, sessionId);

session.testNetwork(token, publisher, function(error, stats) {
  if (error) {
    console.error('testNetwork error', error);
  } else {
    if (stats.packetLossRatio > 0.05 ||
        stats.roundTripTimeMilliseconds > 500 ||
        stats.uploadBitsPerSecond < 50 * 1000) {
      supportsPublishing = false;
    } else if (stats.packetLossRatio > 0.03 ||
        stats.uploadBitsPerSecond < 150 * 1000) {
      publisherOptions.publishVideo = false;
    } else if (stats.uploadBitsPerSecond > 350 * 1000) {
      publisherOptions.resolution = '1280x720';
    } else if (stats.uploadBitsPerSecond > 250 * 1000) {
      publisherOptions.resolution = '640x480';
    } else {
      publisherOptions.resolution = '320x240';
    }
    // Discard the publisher used for testNetwork():
    publisher.destroy();
    // Create a new publisher to use in the session:
    if (supportsPublishing) {
      publisher = OT.initPublisher('pubElementId', publisherOptions);
    }
    // You can now connect to the session and publish
  }
});
```


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

## Subscriber.getStats(completionHandler)

Returns the following details on the subscriber stream quality, including the following:

- Total audio and video packets lost
- Total audio and video packets received
- Total audio and video bytes received

You may use these statistics to have a Subscriber subscribe to audio-only (if the audio packet loss reaches a certain threshold). If you choose to do this, you should set the `audioFallbackEnabled` to false when you initialize Publisher objects for the session. This prevents the OpenTok Media Router from using its own audio-only toggling implementation. (See the documentation for the `OT.initPublisher()` method.)

Note that you can only use this method on a subscriber that is subscribing to another client's stream (not to a stream that the local client is publishing).


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

### Parameters

The method has one parameter — `completionHandler` — which is a function that takes the following parameters:

- `error` — Upon successful completion of the network test, this is undefined. Otherwise (on error), this property is set to an object with the following properties:

    - `code` — The error code, which is set to 1015
    - `message` — A description of the error

  The error results if the client is not connected or the stream published by your own client

- `stats` — An object with the following properties:

    - `audio.bytesReceived` — Total audio bytes received by the subscriber
    - `audio.packetsLost` — Total audio packets that did not reach the subscriber
    - `audio.packetsReceived` — Total audio packets received by the subscriber
    - `timestamp` — The timestamp, in milliseconds since the Unix epoch, for when these stats were gathered
    - `video.bytesReceived` — Total video bytes received by the subscriber
    - `video.packetsLost` — Total video packets that did not reach the subscriber
    - `video.packetsReceived` — Total video packets received by the subscriber


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

Example

The following code checks the connection of a subscriber's stream every 5 seconds and calculates the bytes received per second and packet loss ratios for both audio and video:

```
var lastStats;
session.on('streamCreated', function(event) {
  var subscriber = session.subscribe(event.stream, function(error) {
    if (!error) {
      setInterval(function() {
        subscriber.getStats(function(err, stats) {
          if (lastStats) {
            var audioBytesReceivedPerSecond =
               (stats.audio.bytesReceived -
                lastStats.audio.bytesReceived) /
               (stats.timestamp - lastStats.timestamp),
              videoBytesReceivedPerSecond =
               (stats.video.bytesReceived -
                lastStats.video.bytesReceived) /
               (stats.timestamp - lastStats.timestamp),
              audioPacketLossRatio =
                (stats.audio.packetsLost -
                 lastStats.audio.packetsLost) /
                (stats.audio.packetsReceived -
                 lastStats.audio.packetsReceived),
              videoPacketLossRatio =
                (stats.video.packetsLost -
                 lastStats.video.packetsLost) /
                (stats.video.packetsReceived -
                 lastStats.video.packetsReceived);
          }
          lastStats = stats;
        });
      }, 5000);
    }
  });
});
```


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

## Interpreting pre-connection quality statistics

When you test the client network connection before connecting to the session, you can use the results to determine the ability to send and receive streams. The following tables interpret results (for audio-video sessions and audio-only sessions), with the following quality designations:

- Excellent — none or imperceptible impairments in media.
- Acceptable — some impairments in media, leading to some momentaneous disruptions.

The video resolutions listed are representative of common resolutions. You can determine support for other resolutions by interpolating the results of the closest resolutions listed.

### Audio-video sessions

For the given qualities and resolutions, all the following conditions must met.

| Quality | Video resolution @fps | Upload bandwidth (kbps) | Download bandwidth / stream (kbps) | Packet loss (%) | Round-trip time (ms) < |
|---|---|---|---|---|---|
| Excellent | 1280x720 @30 | > 1000 | > 1000 | < 0.5% | < 250 |
| Excellent | 640x480 @30 | > 600 | > 600 | < 0.5% | < 250 |
| Excellent | 320x240 @30 | > 300 | > 300 | < 0.5% | < 250 |
| Acceptable | 1280x720 @30 | > 350 | > 350 | < 3% | < 500 |
| Acceptable | 640x480 @30 | > 250 | > 250 | < 3% | < 500 |
| Acceptable | 320x240 @30 | > 150 | > 150 | < 3% | < 500 |


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*

## Audio-only sessions

For the given qualities, all the following conditions must met:

| Quality | Upload bandwidth (kbps) | Download bandwidth / stream (kbps) | Packet loss (%) | Round-trip time (ms) |
|---|---|---|---|---|
| Excellent | > 60 | > 60 | < 0.5% | < 250 |
| Acceptable | > 50 | > 50 | < 5% | < 500 |


*Copyright 2015 TokBox, Inc. — Revision 2 (February 3, 2015) — Confidential and proprietary*