Bria Desktop API

Developer Guide

CounterPath Corporation
Suite 300, One Bentall Centre
505 Burrard Street, Box 95
Vancouver, BC V7X 1M3
Tel: 604.320.3344
sales@counterpath.com www.counterpath.com

CounterPath® and the ® logo are trademarks of CounterPath Corporation.

This document contains information proprietary to CounterPath Corporation, and shall not be used for engineering, design, procurement, or manufacture, in whole or in part, without the consent of CounterPath Corporation. The content of this publication is intended to demonstrate typical uses and capabilities of the CounterPath Bria® Desktop API from CounterPath Corporation. Users of this material must determine for themselves whether the information contained herein applies to a particular IP-based networking system. CounterPath makes no warranty regarding the content of this document, including—but not limited to—implied warranties of fitness for any particular purpose. In no case will CounterPath or persons involved in the production of this documented material be liable for any incidental, indirect or otherwise consequential damage or loss that may result after the use of this publication.

Windows and Win32 are registered trademarks of Microsoft Corporation in the United States and other countries. All other products and services are the registered trademarks of their respective holders.

This manual corresponds to Bria® 5 Desktop.

Contents

List of Functions and Events

Function or Event	Purpose	Supported Bria Version	Page
GET /answer	Answer a Call		26
GET /audioProperties	Set Audio Settings		30
GET /bringToFront	Bring Phone to Front		20
GET /call	Place a Call		25
GET /callOptions	Set the Call Options		31
GET /checkVoiceMail	Check for Voicemail		37
GET /dtmf	Send DTMF		29
GET /endCall	End a Call		32
GET /hold	Hold a Call		28
GET /im	Start an IM Session		40
GET /resume	Resume a Call		28
GET /selectAudioDevices	Set Audio Devices	4.2+	23
GET /setPresence	Set Presence Status	4.2+	41
GET /showHistory	Bring the History Panel to Front		21
GET /startCallRecording	Start recording a call	4.5+	33
GET /startScreenShare	Start a Screen Sharing Session	4.2+	42
GET /stopCallRecording	Stop call recording	4.5+	33
GET /status "audioDevices"	Get Current Audio Devices	4.2+	22
GET /status "audioProperties"	Get Current Audio Settings		29
GET /status "call"	Get Information about Current Calls		26
GET /status "callHistory"	Get Call History Data		34
GET /status "callHistoryItem"	Get Call History Entry	4.7+	35
GET /status "callOptions"	Get the Call Options Modes		31
GET /status "contact"	Get Information for One Contact		39
GET /status "missedCall"	Get Count of Missed Calls		35
GET /status "phone"	Get Status of Phone		21
GET /status "presence"	Get Current Presence	4.2+	40
GET /status "screenShare"	Get Information about Ongoing Screenshare.	4.2+	43
GET /status "supportedPresence"	Get Information about Supported Presences	4.2+	40
GET /status "systemInformation"	Get Information on Bria Version	4.4+	22
GET /status "systemSettings"	Get Information on Bria Setup		22
GET /status "voiceMail"	Get MWI Count		37
GET /transferCall	Trasnfer a Call	4.5+	29

Function or Event	Purpose	Supported Bria Version	Page
POST /statusChange "audioDevices"	Change to Audio Devices	4.2+	17
POST /statusChange "audioProperties"	Change to Audio Settings		17
POST /statusChange "authentication"	Change to Authentication		18
POST /statusChange "call"	Change to Call Status		16
POST /statusChange "callHistory"	Change to Call History		16
POST /statusChange "callOptions"	Change to Call Options		18
POST /statusChange "missedCall"	Missed Call Occurred		17
POST /statusChange "phone"	Change to Phone Status		16
POST /statusChange "presence"	Change to Presence	4.2+	18
POST /statusChange "screenShare"	Change to Screenshare Sessions	4.2+	18
POST /statusChange "voicemail"	Change to MWI Count		17

1. About the Bria Desktop API

The Bria® Desktop API provides functions that let another application instruct Bria 5 Desktop *for Windows and Mac* to place phone calls and obtain call history information as well as be alerted about incoming calls and other events.

For example, your application may have a web page that provides information about customers for your employees to phone.

- The employee clicks on the customer phone number, which invokes a function from the Bria Desktop API to place the phone call via Bria.
- Bria executes the request and sends events to your application. These events provide information about the status of the request.
- The employee will manage the call, sometimes from your application and sometime using Bria. For example, the user can mute from your application but must use the Bria UI directly to transfer the call.
- When the call ends, your application will receive notification that the call has ended.

The API provides some control over Bria but not complete control: the user may have to interact with the Bria user interface in order to manage the call: place on hold, transfer, and so on.

This API guide assumes that you are familiar with the standard functionality of Bria 5 *for Windows or Mac*, as described in the Bria 5 User Guide.

2. Working with the API

Connecting to Bria via Web Socket

Connect to Bria by establishing a secure web socket connection ("wss") to Bria. This connection is full-duplex and message-oriented.

The Bria API supports two types of communications:

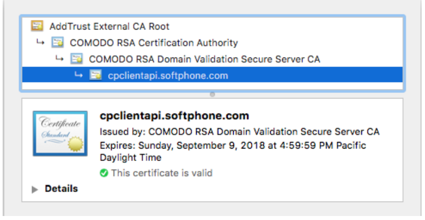
- Events from Bria. See page 8.
- Request/response messages. See page 20.

Events, requests and responses are in XML format with UTF-8 encoding.

Using Browsers and Javascript

CounterPath has provided the following support to enable connections to the client via popular browsers:

- A subdomain cpclientapi.softphone.com which re-directs to 127.0.0.1 enabling routing through browser local host access protection
- A browser-friendly certificate embedded in Bria which enables the setup of a trusted connection between your web application and the Bria client



AddTrust External CA Root
↳ COMODO RSA Certification Authority
↳ COMODO RSA Domain Validation Secure Server CA
↳ cpclientapi.softphone.com

cpclientapi.softphone.com
Issued by: COMODO RSA Domain Validation Secure Server CA
Expires: Sunday, September 9, 2018 at 4:59:59 PM Pacific Daylight Time
✓ This certificate is valid

▶ **Details**

- Support for secure web socket ("wss") connections via cross domain connections within Bria.

Here is an example of how the web socket session can be established within the Javascript language:

```
var ws = new WebSocket("wss://cpclientapi.softphone.com:9002/counterpath/socketapi/v1");
```

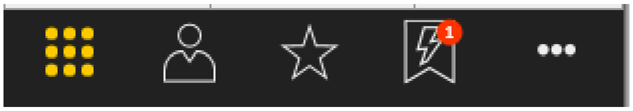
As of writing, this approach has been tested with Chrome, Internet Explorer 11 and newer, Microsoft Edge, Firefox, and Safari on Mac and Windows 7 and newer operating systems.

End User's Permission

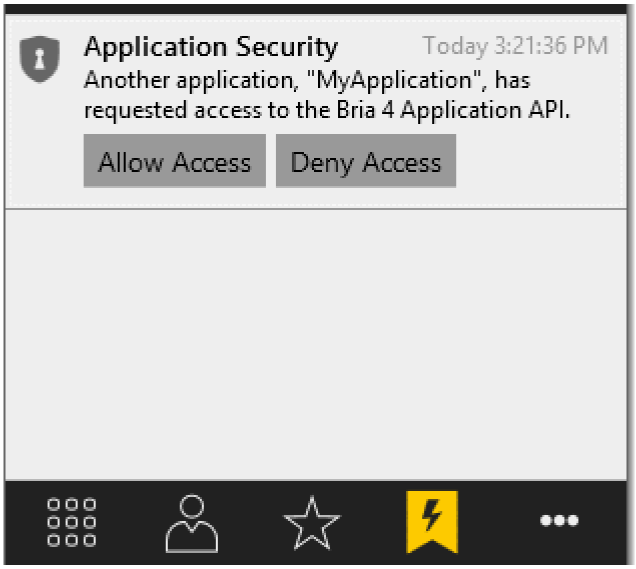
Bria 4.6 introduced a new notification mechanism which requires end user's permission before API calls gain control of Bria clients. You will need to inform your end users to give your application the permission to access their Bria clients.

The first time an API call tries to connect to Bria, Bria will show an alert on the Notification tab of the Bria client. End user needs to click the tab and allow an API application to access their Bria. Until end user takes an action, API calls have no control over their Bria.

1. Bring the Bria client to the foreground and click the Notification tab. The tab appears only when you have a notification.



2. Click Allow Access.



Application Security Today 3:21:36 PM
Another application, "MyApplication", has requested access to the Bria 4 Application API.

API Format

The message format resembles HTTP but does not follow all HTTP conventions.

The content-type is always application/xml, which means that the body always starts with <?xml version …>

Escape Character Handling

These special characters must be escaped in requests.

In responses, the characters will be represented by the escape sequence.

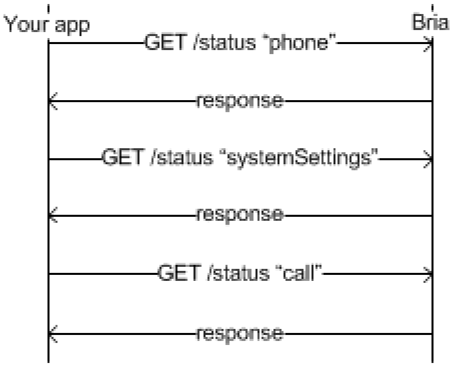
```
"    &quot;
'    &apos;
<    &lt;
>    &gt;
&    &amp;
```

3. Typical Sequences

3.1. Startup

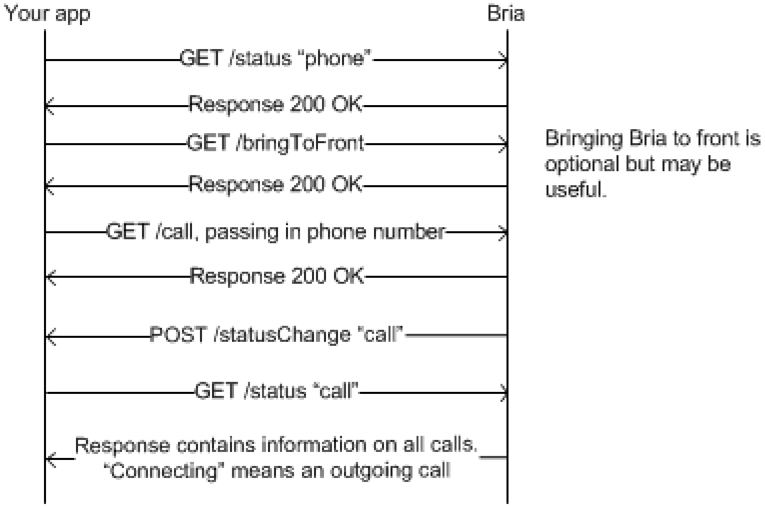
Startup

The following diagram shows a recommended startup sequence. Keep in mind that at any time before, during, or after this sequence, you may receive events completely independent of this sequence. For example, if a call ends.

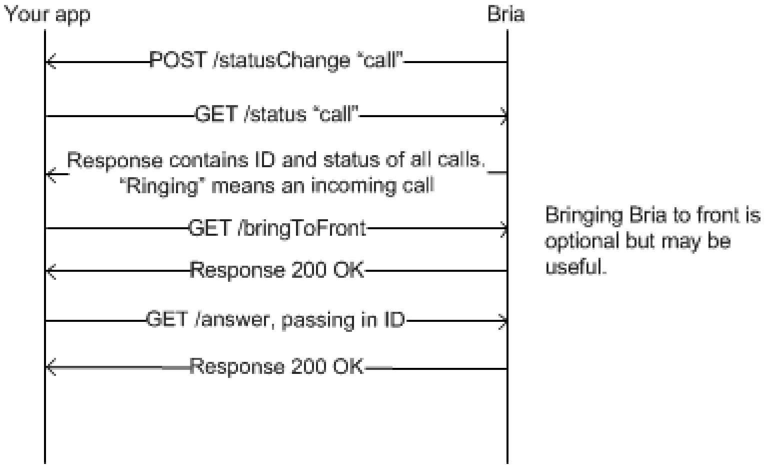


3.2. Placing and Answering Calls

Place an Outgoing Call


Your app
Bria
GET /status "phone"
Response 200 OK
GET /bringToFront
Response 200 OK
Bringing Bria to front is optional but may be useful.
GET /call, passing in phone number
Response 200 OK
POST /statusChange "call"
GET /status "call"
Response contains information on all calls. "Connecting" means an outgoing call

Answer a Call


Your app
Bria
POST /statusChange "call"
GET /status "call"
Response contains ID and status of all calls. "Ringing" means an incoming call
GET /bringToFront
Response 200 OK
Bringing Bria to front is optional but may be useful.
GET /answer, passing in ID
Response 200 OK

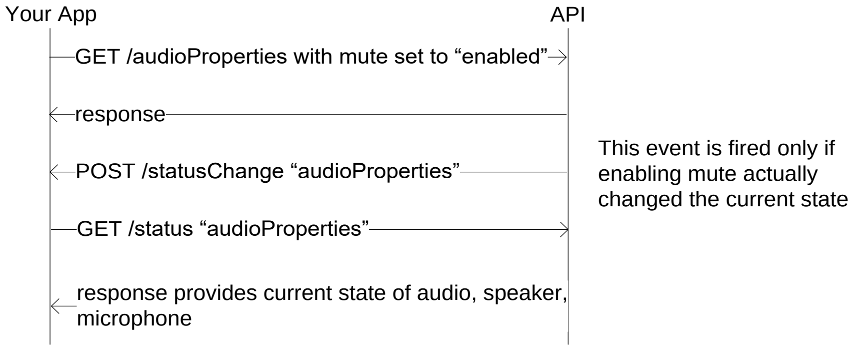
3.3. Monitoring and Handling Established Calls

You can use the API to handle the established call and to obtain information about the state of each call, for example, whether it is on hold, using GET /status "call".

Control Audio

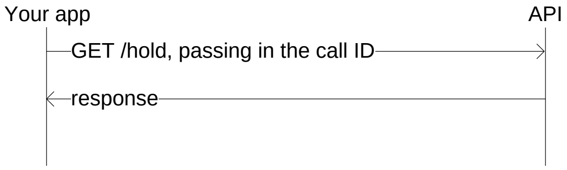
You can change the volume level on the speaker and microphone, and turn mute on or off. Changes to the volume affect the phone as a whole, not just the current call.

For example, to mute a call:

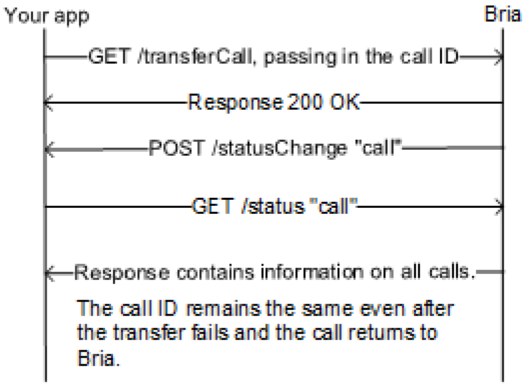

Your App
API
GET /audioProperties with mute set to “enabled”
response
POST /statusChange “audioProperties”
GET /status “audioProperties”
response provides current state of audio, speaker, microphone
This event is fired only if enabling mute actually changed the current state

Hold and Resume a Call

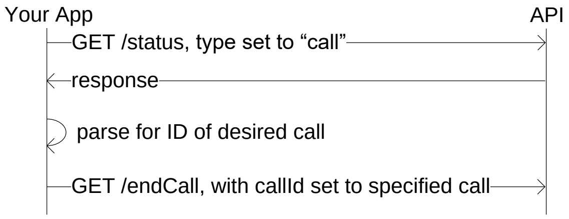
For example, to hold a call:



Transfer a Call


Your app
Bria
GET /transferCall, passing in the call ID
Response 200 OK
POST /statusChange "call"
GET /status "call"
Response contains information on all calls.
The call ID remains the same even after the transfer fails and the call returns to Bria.

End a Call

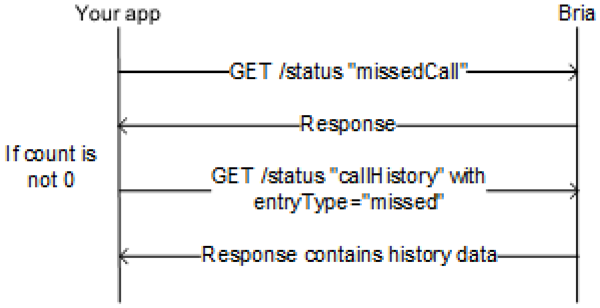

Your App
API
GET /status, type set to “call”
response
parse for ID of desired call
GET /endCall, with callId set to specified call

3.4. History and Voicemail

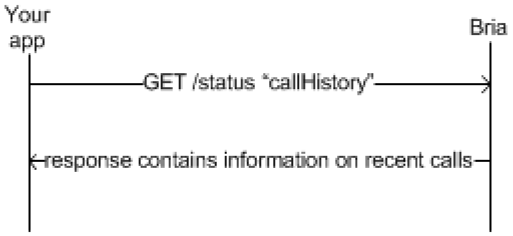
Obtain the Missed Call Count and Details

When a call is missed on Bria, you receive the POST /statusChange "missedCall" event.

You can also query for missed calls independently of this event (for example, at startup), as follows:

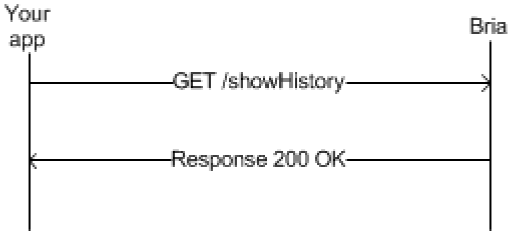

Your app
Bria
GET /status "missedCall"
Response
If count is not 0
GET /status "callHistory" with entryType="missed"
Response contains history data

Display Call History in your App

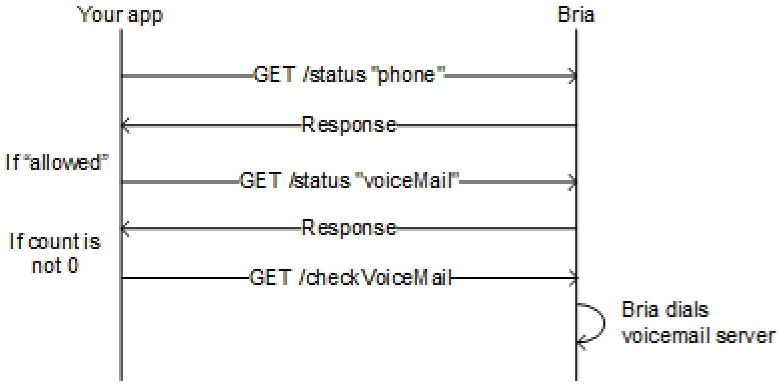

Your app
Bria
GET /status "callHistory"
response contains information on recent calls

Bring the Bria Call History Panel to the Front

You can also display the History tab on Bria. This function brings Bria to the front (if necessary) and brings the History tab to the front.

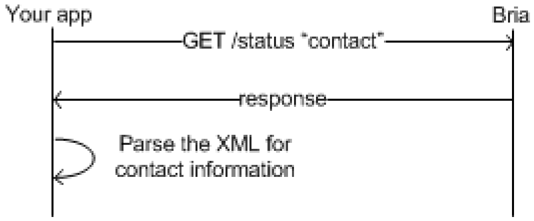


Get MWI Information and Connect to the Voicemail Server

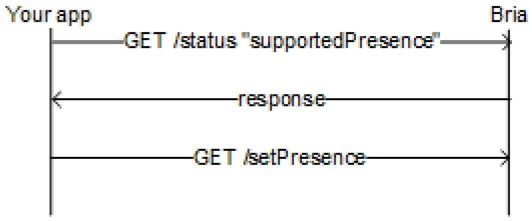


3.5. Contacts and Presence

Get Contact Information

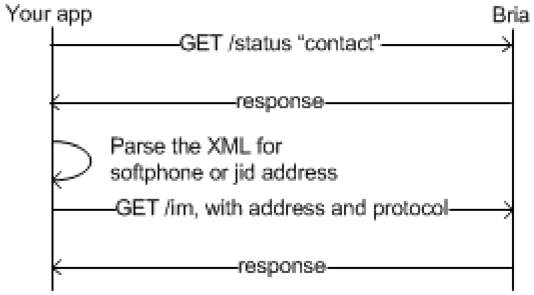


Set Presence



3.6. Instant Messages

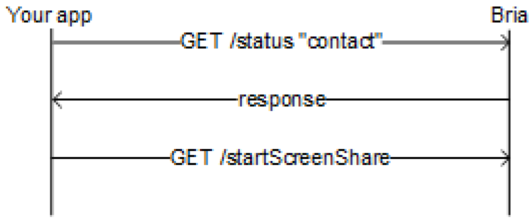
Start an IM Session


Your app
Bria
GET /status "contact"
response
Parse the XML for softphone or jid address
GET /im, with address and protocol
response

3.7. Screenshare

Start a Screenshare Session

Use GET /status "contact" to obtain the SIP address or XMPP jid of participants. Then use GET /startScreenShare to start screen share. The session ends when everyone leaves the session.



4. Events

As soon as you connect, you must be prepared to start receiving events from Bria. Events are generated for changes initiated from the Bria UI (or some other agent) as well as from the API.

You should ignore any event you are not interested in and expect that newer versions of Bria may send more events than what is listed in this documentation.

Phone Status Changed: POST /statusChange "phone"

The readiness of Bria has changed.

```
POST /statusChange
User-Agent: MyApplication
Content-Type: application/xml
Content-Length: 63
<?xml version="1.0" encoding="utf-8" ?>
<event type="phone"></event>
```

Next Action

Get details on the change by sending this request: Get /status "phone"

Call Status Changed: POST /statusChange "call"

The status of an existing call has changed.

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 62
<?xml version="1.0" encoding="utf-8" ?>
<event type="call"></event>
```

Next Action

Get /status "call"

Call History Changed: POST /statusChange "callHistory"

The contents of the call history has changed: a call has been added or the user has manually deleted calls.

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 69
<?xml version="1.0" encoding="utf-8" ?>
<event type="callHistory"></event>
```

Next Action

Get /status "callHistory"

Missed Call Occurred: POST /statusChange "missedCall"

The missed call count has changed.

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 68
<?xml version="1.0" encoding="utf-8" ?>
<event type="missedCall"></event>
```

Next Action

GET /status "missedCall" or GET /status "callHistory"

MWI Count Changed: POST /statusChange "voicemail"

Bria has received a MWI message from the voicemail server.

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 72
<?xml version="1.0" encoding="utf-8" ?>
<event type="voiceMail"></event>
```

Next Action

GET /status "voicemail" and GET /checkVoiceMail

Audio Devices Changed: POST /statusChange "audioDevices"

The presence status state of Bria has changed.

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 75
<?xml version="1.0" encoding="utf-8" ?>
<event type="audioDevices"></event>
```

Next Action

Get details on the change by sending this request: GET /status "audioDevices"

Audio Settings Changed: POST /statusChange "audioProperties"

This event fires when:

- the speaker is muted or unmute
- the speaker mode changes, and
- the volume changes.

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 62
<?xml version="1.0" encoding="utf-8" ?>
<event type="audioProperties"></event>
```

Next Action

Get /status "audioProperties"

Call Options Changed: POST /statusChange "callOptions"

One of the Bria "call options" has been turned on or off :

- Anonymous calling
- Auto answer
- Letters-to-numbers
- Call waiting

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 62
<?xml version="1.0" encoding="utf-8" ?>
<event type="callOptions"></event>
```

Next Action

Get /status "callOptions"

Presence Status Changed: POST /statusChange "presence"

The presence status state of Bria has changed.

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 71
<?xml version="1.0" encoding="utf-8" ?>
<event type="presence">
</event>
```

Next Action

Get details on the change by sending this request: GET /status "presence"

Screenshare Status Changed: POST /statusChange "screenShare"

The state of the screen sharing session of Bria has changed.

```
POST /statusChange
User-Agent: Bria Stretto
Content-Type: application/xml
Content-Length: 74
<?xml version="1.0" encoding="utf-8" ?>
<event type="screenShare"></event>
```

Next Action

Get details on the change by sending this request: GET /status "screenShare"

Authentication Changed: POST /statusChange "authentication"

Bria has logged on with its provisioning server or has logged out. This action was triggered by the user from the Bria UI.

```
POST /statusChange
User-Agent: Bria 5
Content-Type: application/xml
Content-Length: 62
<?xml version="1.0" encoding="utf-8" ?>
<event type="authentication"></event>
```

5. Requests and Reponses

5.1. Request and Response Syntax

The format and content of requests are illustrated in the following pages. The request can include a Transaction-ID; if this is included, then the response will also include this ID.

The order of elements within a tag is not important.

All requests receive an immediate response from Bria:

- 200 OK: Bria understands the request. This response may include details; see the information for the individual requests.
- 400 Bad Request: Bria does not understand the request. Check that the xml format and the request syntax are correct.
- 503 Service Unavailable: Bria is in the notReady or notAllowed state. See Get /status "phone".

If the response is 200 OK, then you may start receiving events that describe the progress of the request.

5.2. Getting Ready

Bring Phone to Front: GET /bringToFront

Bring the main window of Bria to front and give it focus.

```
GET/bringToFront
User-Agent: MyApplication
Transaction-ID: AE26f998027
Content-Type: application/xml
Content-Length: 0
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: AE26f998027
Content-Type: application/xml
Content-Length: 0
```

Bring the History Panel to Front: GET /showHistory

Bring the History panel to the front and give it focus, and show the specified types of calls.

- type: The types of calls to fetch: "all" or "missed".
- text: Optional. The text to enter in the History panel. The contents of the History panel will be filtered to show only entries that match this text.

```
GET/showHistory
User-Agent: MyApplication
Transaction-ID: FE881337
Content-Type: application/xml
Content-Length: 72
<?xml version="1.0" encoding="utf-8" ?>
<filter>
<type>missed</type>
<text>Frank Chan</text>
</filter>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: FE881337
Content-Type: application/xml
Content-Length: 0
```

Get Status of Phone: GET /status "phone"

Request the current phone status.

```
GET/status
User-Agent: MyApplication
Transaction-ID: GF832137
Content-Type: application/xml
Content-Length: 71
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>phone</type>
</status>
```

Response

- status: phone
- state: "ready" or "notReady" (no accounts are enabled).
- call: "allow" or "notAllow" (there is no free line available to make a call).
- account status: One of these values:
  - connected
  - connecting
  - failureContactingServer
  - failureAtServer
  - disabled
- account failure code: the SIP error code provided by the SIP server, if account status is "failureAtServer". If account status is another value, this parameter specifies "0".

- maxLines: The number of lines allowed in your brand of Bria. Typically 6.

```
HTTP/1.1 200 OK
Transaction-ID: GF832137
Content-Type: application/xml
Content-Length: 125
<?xml version="1.0" encoding="utf-8" ?>
<status type="phone">
<state>ready</state>
<call>allow</call>
<accountStatus>connected<accountStatus>
<accountFailureCode>0</accountFailureCode>
<maxLines>6</maxLines>
</status>
```

Get Information on Bria Setup: GET /status "systemSettings"

Request the current configuration of Bria.

```
GET/status
User-Agent: MyApplication
Transaction-ID: BS398809
Content-Type: application/xml
Content-Length: 80
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>systemSettings</type>
</status>
```

Response

- defaultCallType: The value currently selected in Bria in Preferences > Application > Default Action: "audio", "video" or "conference".
- callRightAwayOnceNumberSelected: "true" or "false". If false, then when a call is placed using GET /call, then the call entry field on Bria will be populated with the data you provide but the user will have to click the Call button on the Bria user interface in order to place the call. If true, the user will not have to intervene.

```
HTTP/1.1 200 OK
Transaction-ID: BS398809
Content-Type: application/xml
Content-Length: 184
<?xml version="1.0" encoding="utf-8" ?>
<status type="systemSettings">
<defaultCallType>audio</defaultCallType>
<callRightAwayOnceNumberSelected>true</callRightAwayOnceNumberSelected>
</status>
```

Get Information on Bria Version: GET /status "systemInformation"

Request the current version and build number of Bria.

```
GET /status
User-Agent: MyApplication
Transaction-ID: 1000
Content-Type: application/xml
Content-Length: 93
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>systemInformation</type>
</status>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: 1000
Content-Type: application/xml
Content-Length: 291
<?xml version="1.0" encoding="utf-8" ?>
<status type="systemInformation">
<systemCompanyName>CounterPath Corporation</systemCompanyName>
<systemProductName>Bria</systemProductName>
<systemProductVersion>4.4.0</systemProductVersion>
<systemProductBuild>79955</systemProductBuild>
</status>
```

Get Current Audio Devices: GET /status "audioDevices"

Get information on audio devices.

```
GET /status
User-Agent: MyApplication
Content-Type: application/xml
Content-Length: 14
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>audioDevices</type>
</status>
```

Response

- name: Device Name as it displays in Bria.
- id: The Device ID related to the device.
- type: type of usage either input or output.
- selected role: name of device role(s) currently selcetd for device.

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: 835
<?xml version="1.0" encoding="utf-8" ?>
<status type="audioDevices">
<devices>
<device>
<name>plantronics C320-M</name>
<id>1234</id>
<type>output</type>
<selected role="headset, speakPhone, ringOn">true</selected>
</device>
<device>
<name>plantronics C320-M</name>
<id>2345</id>
<type>input</type>
<selected role="headset, speakPhone">true</selected>
</device>
</devices>
</status>
```

Set Audio Devices: GET /selectAudioDevices

Set specific devices for audio input and output calling.

- name: Device Name as it displays in Bria.
- id: The Device ID.

- type: input or output.
- role: name of device role.

```
GET /selectAudioDevices
User-Agent: MyApplication
Content-Type: application/xml
Content-Length: 835
<?xml version="1.0" encoding="utf-8" ?>
<devices>
<device>
<name>Plantronics C710</name>
<id>283841905</id>
<type>output</type>
<role>headset</role>
</device>
<device>
<name>Plantronics C710</name>
<id>283841905</id>
<type>input</type>
<role>headset</role>
</device>
</devices>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID:
Content-Type: application/xml
Content-Length: 0
```

5.3. Handling Phone Calls

Place a Call: GET /call

Place a call of the specified type, to the specified number, and presenting the specified display name for the caller.

- dial type: "audio", "video".
- number: The phone number to dial. See "Details" below.
- displayName: Optional. The name of the local user; this is the name that will be presented to the other party.
- suppressMainWindow: When true, the Bria Main window remains in the background. When false, the Main window is brought to the foreground.

```
GET/call
User-Agent: MyApplication
Transaction-ID: GF8002137
Content-Type: application/xml
Content-Length: 135
<?xml version="1.0" encoding="utf-8" ?>
<dial type="audio">
<number>1440@zippy-voip.com</number>
<displayName>Joseph Santos</displayName>
<suppressMainWindow>true</suppressMainWindow>
</dial>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: GF8002137
Content-Type: application/xml
Content-Length: 0
```

Details

If the response is 200 OK, Bria populates the call entry field with the specified phone number. Depending on the value of callRightAwayOnceNumberSelected (GET /status "systemSettings") the call is either placed immediately or only after the user clicks the Call button.

To select the account to place the call on, Bria uses the account selection method that is currently selected in Bria: Auto Select or a specific account. If Auto Select is used and dial plans are defined, the phone number may be modified (by the dial plan) before it is placed. However, the call history shows the phone number as specified in the request (not as modified).

Once the call attempt starts, POST /statusChange "call" events are posted.

Answer a Call: GET /answer

Answer the specified incoming call. If another call is live, it is automatically put on hold and the newly answered call becomes live.

- callId: the ID of the call to answer.

```
GET /answer
User-Agent: MyApplication
Content-Type: application/xml
Content-Length: 104
<?xml version="1.0" encoding="utf-8" ?>
<answerCall>
   <callId> 391A64292F0A413aB14CBAB341621A3D </callId>
</answerCall>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: YUS002157
Content-Type: application/xml
Content-Length: 0
```

Get Information about Current Calls: GET /status "call"

Get information on all existing calls.

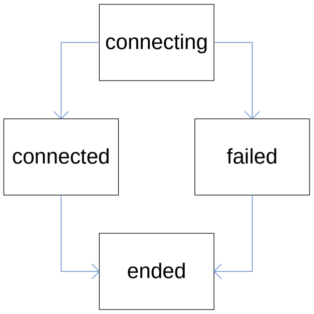
```
GET/status
User-Agent: MyApplication
Transaction-ID: EF855137
Content-Type: application/xml
Content-Length: 70
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>call</type>
</status>
```

Response

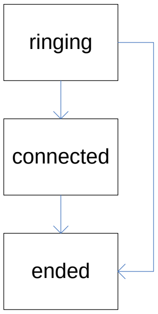
One or more <call> tags, each containing information about an incoming or outgoing call:

- id: the unique ID of this call.
- holdStatus: offHold, localHold, and remoteHold
- participants: One or more <participant> tags each containing:
  - number: The number of the other party
  - displayName: The display name of the other party
  - state: connected, connecting, ringing, failed, ended

States for an outgoing call



States for an incoming call



- timeInitiated: time the call attempt started, in Unix time or POSIX time (which is the number of seconds (not including leap seconds) since 00:00:00 January 1, 1970 UTC). The time is only available once a call is connected; it will be 0 in the connecting state.
- recordingStatus: "notRecording" or "recordingInProgress"
- recordingFile: this is the **target** filename for the recording. See details in Section 5.4.

If there are currently no existing calls, the <call> will be empty.

In the following example, the second call is a conference call (it would have been created from the Bria UI because the API does not support creation of conference calls).

```
HTTP/1.1 200 OK
Transaction-ID: EF855137
Content-Type: application/xml
Content-Length: 769
<?xml version="1.0" encoding="utf-8" ?>
<status type="call">
<call>
   <id>ea6c7664001343688013e3680013e368</id>
   <holdStatus>offHold</holdStatus>
   <recordingStatus>recordingInProgress</recordingStatus>
   <recordingFile>C:\Users\bsmith\Documents\MyRecordings\thisname.wav</recordingFile>
   <participants>
   <participant>
   <number>1440@zippy-voip.com</number>
   <displayName>Rita Santos</displayName>
   <state>connecting</state>
   <timeInitiated> 1303931552</timeInitiated>
   </participant>
   </participants>
</call>
<call>
   <id>ea6c7664001343688013e3680013e368</id>
   <holdStatus>onHold</holdStatus>
   <recordingStatus>notRecording</recordingStatus>
   <recordingFile></recordingFile>
   <participants>
   <participant>
   <number>2758@zippy-voip.com</number>
   <displayName>Kokila Perera</displayName>
   <state>connected</state>
   <timeInitiated> 1303931542</timeInitiated>
   </participant>
   <participant>
   <number>2764@zippy-voip.com</number>
   <displayName>Frank Chan</displayName>
   <state>connected</state>
   <timeInitiated> 1303931531</timeInitiated>
   </participant>
   </participants>
```

```
</call>
</status>
```

Hold a Call: GET /hold

Put a specific call on hold.

- callId: the ID of the call to hold.

```
GET /hold
User-Agent: Bria 5
Transaction-ID: 51ffe437439744219aab8ef2e0658ac0
Content-Type: application/xml
Content-Length: 113

<?xml version="1.0" encoding="UTF-8"?>
<holdCall>
 <callId>1310b86afcb44d6d9a00e93916dffee1</callId>
</holdCall>
```

Response

- 200 OK if Bria was able to parse the request, a 4xx reponse if the request was not able to be parsed.

```
HTTP/1.1 200 OK
Transaction-ID: 51ffe437439744219aab8ef2e0658ac0
Content-Type: application/xml
Content-Length: 0
```

Resume a Call: GET /resume

Take a specific call off hold.

- callId: the ID of the call to resume.

```
GET /resume
User-Agent: Bria 5
Transaction-ID: 5505d27c33564234b25a6c239603c0d4
Content-Type: application/xml
Content-Length: 117

<?xml version="1.0" encoding="UTF-8"?>
<resumeCall>
 <callId>1310b86afcb44d6d9a00e93916dffee1</callId>
</resumeCall>
```

Response

- 200 OK if Bria was able to parse the request, a 4xx reponse if Bria could not parse it.

```
HTTP/1.1 200 OK
Transaction-ID: 5505d27c33564234b25a6c239603c0d4
Content-Type: application/xml
Content-Length: 0
```

Send DTMF: GET /dtmf

Enter a DTMF digit.

- digit: possible values: 0,1,2,3,4,5,6,7,8,9,*,#
- tone: start or stop to control the DTMF tone.

```
GET /dtmf
User-Agent: Bria 5
Transaction-ID: ad87510b179745c7a610a4417f4c4604
Content-Type: application/xml
Content-Length: 84

<?xml version="1.0" encoding="UTF-8"?>
<dtmf>
 <tone digit="5">start</tone>
</dtmf>
```

Response

- 200 OK if Bria was able to parse the request, a 4xx reponse if Bria could not parse it..

```
HTTP/1.1 200 OK
Transaction-ID: ad87510b179745c7a610a4417f4c4604
Content-Type: application/xml
Content-Length: 0
```

Transfer a Call: GET /transferCall

Transfer a specific call to a specific number. A /statusChange "call" event will be generated when Bria attempts transfer. If the transfer succeeds, the call ends. If the transfer fails, the original call comes back to Bria.

- callId: the ID of the call to transfer.
- target: the number to transfer the call to.

```
GET /transferCall
User-Agent: MyApplication
Transaction-ID: ad87510b179745c7a610a4417f4c4604
Content-Type: application/xml
Content-Length: 128
<?xml version="1.0" encoding="utf-8" ?>
<transferCall>
  <callId>f57635b383da4861ad538371428c14bd</callId>
  <target>1331@zippy-voip.com</target>
</transferCall>
```

Response

- 200 OK if Bria was able to parse the request, a 4xx reponse if the request was not able to be parsed.

```
HTTP/1.1 200 OK
Transaction-ID: ad87510b179745c7a610a4417f4c4604
Content-Type: application/xml
Content-Length: 0
```

Get Current Audio Settings: GET /status "audioProperties"

Request the current settings for mute and volume.

```
GET/status
User-Agent:MyApplication
Transaction-ID:GF8002137
Content-Type:application/xml
Content-Length:88

<?xml version="1.0" encoding="UTF-8"?>
<status>
<type>audioProperties</type>
</status>
```

Response

- mute: whether or not the local microphone is muted: enabled/disabled.
- speakerMute: whether or not local sound output is muted: enabled/disabled.
- speaker: whether or not speaker mode is enabled: enabled/disabled.
- speaker volume: speakerphone volume, as a number from 0 to 100.
- microphone volume: microphone volume, as a number from 0 to 100.

```
  HTTP/1.1 200 OK
  Transaction-ID: GF8002137
  Content-Type: application/xml
  Content-Length: 835

  <?xml version="1.0" encoding="utf-8" ?>
  <status type="audioProperties">
    <mute>enabled</mute>
    <speakerMute>disabled</speakerMute>
    <speaker>disabled</speaker>
    <volume type="speaker">100</volume>
    <volume type="microphone">2</volume>
  </status>
```

Set Audio Settings: GET /audioProperties

Turn mute on or off and/or change the volume for the incoming and/or outgoing audio. One or more of the following can be included in the request.

- mute: whether or not the local microphone is muted: enabled/disabled.
- speakerMute: whether or not local sound output is muted: enabled/disabled.
- speaker: whether or not speaker mode is enabled: enabled/disabled.
- speaker volume: speakerphone volume, as a number from 0 to 100.
- microphone volume: microphone volume, as a number from 0 to 100.

```
GET /audioProperties
User-Agent:MyApplication
Content-Type:application/xml
Content-Length:835

<?xml version="1.0" encoding="utf-8"?>
<audioProperties>
<mute>enabled</mute>
```

```
<speakerMute>disabled</speakerMute>
<speaker>enabled</speaker>
<volume type="speaker">100</volume>
<volume type="microphone">2</volume>
</audioProperties>
```

Get State of Call Options: GET /status "callOptions"

Get the current state of the call options: anonymous calling, auto answer, letters-to-numbers, and call waiting.

```
GET /status
User-Agent: Bria 5
Transaction-ID: 86f57847379643b7a64a887d51f47103
Content-Type: application/xml
Content-Length: 84
<?xml version="1.0" encoding="UTF-8"?>
<status>
<type>callOptions</type>
</status>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: 86f57847379643b7a64a887d51f47103
Content-Type: application/xml
Content-Length: 225
<?xml version="1.0" encoding="utf-8" ?>
<status type="callOptions">
<anonymous>disabled</anonymous>
<lettersToNumbers>disabled</lettersToNumbers>
<autoAnswer>disabled</autoAnswer>
<callWaiting>enabled</callWaiting>
</status>
```

Set Call Options: GET /callOptions

Turn the specified options on or off. You can enter one request to set several options simultaneously.

- anonymous: Set to enable or disable anonymous calling. When anonymous calling is on, anonymity is enabled at the ID level, as per RFC 3325; the caller’s name will be removed everywhere in the SIP message.
- lettersToNumbers: Set to enable or disable letters-to-numbers. When letters-to-numbers is on, letters typed into the Bria call entry field or passed in the <number> element in GET /call will automatically be converted to numbers.
- autoAnswer: Set to enable or disable auto answer. When auto answer is on, phone calls will be automatically answered by Bria (after the delay specified in Preferences on the Calls panel).
- callWaiting: Set to enable or disable call waiting. When call waiting is enabled, incoming phone calls will be automatically displayed. When disabled, incoming calls will go directly to the voicemail (if configured) or play a busy tone to the caller.

```
GET /callOptions
User-Agent: MyApplication
Transaction-ID: GF8002137
Content-Type: application/xml
Content-Length: 135
<?xml version="1.0" encoding="utf-8" ?>
<callOptions>
<anonymous>enabled</anonymous>
<lettersToNumbers>enabled</lettersToNumbers>
<autoAnswer>enabled</autoAnswer>
```

```
<callWaiting>disabled</callWaiting>
</callOptions>
```

End a Call: GET /endCall

Hang up the call that has the specified call ID. You can obtain the call ID using GET /status "call".

```
GET/endCall
User-Agent: MyApplication
Transaction-ID: YW831137
Content-Type: application/xml
Content-Length: 104
<?xml version="1.0" encoding="utf-8" ?>
<endCall>
<callId>391A64292F0A413aB14CBAB341621A3D</callId>
</endCall>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: YW831137
Content-Type: application/xml
Content-Length: 0
```

5.4. Call Recording

Start Call Recording: GET /startCallRecording

Bria must have at least one established call. Call recording cannot be started for a call if recording is already in progress.

- callId: the call ID of the call to record. See “Details” below.
- filename: the target filename for the recording. Do not include any extension to the filename. Bria will automatically append the extension. Also see “Details” below.
- suppressPopup: “true”, “false” – whether or not Bria should show the regular Call Recording popup when the recording is stopped.

Request

```
GET /startCallRecording
User-Agent: MyApplication
Transaction-ID: 1234567890
Content-Type: application/xml
Content-Length: 241

<?xml version="1.0" encoding="utf-8" ?>
<startCallRecording>
 <callId>07efa3a1809e4ec08e46362e8916706d</callId>
 <filename>C:\Users\bsmith\Documents\MyRecordings\thename</filename>
 <suppressPopup>true</suppressPopup>
</startCallRecording>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: 1234567890
Content-Type: application/xml
Content-Length: 0
```

Details

If the callId parameter is omitted then the call recording will be started for the current “Live” call. If the call is on hold when call recording is started, recording will only begin when the call is taken off hold.

If a call is put on hold, the call recording will be split into multiple files, one for the first period when the call was on hold, another for the next period, and so on. The files will then be stored in a folder named using the given filename parameter. The files will be named <filename>.wav, <filename>-1.wav, <filename>-2.wav, and so on.

Stop Call Recording: GET /stopCallRecording

In order to stop call recording, a call must be established and call recording must be active for it.

- callId: the call ID of the call to stop recording. If omitted, the call recording will be stopped for the current “Live” call.

Request

```
GET /stopCallRecording
User-Agent: MyApplication
Transaction-ID: 1234567890
Content-Type: application/xml
Content-Length: 133

<?xml version="1.0" encoding="utf-8" ?>
<stopCallRecording>
 <callId>07efa3a1809e4ec08e46362e8916706d</callId>
</stopCallRecording>
```

Response

```
HTTP/1.1 200 OK
Transaction-ID: 1234567890
Content-Type: application/xml
Content-Length: 0
```

5.5. Call History

Get Call History Data: GET /status "callHistory"

Obtain the call data for the most recent phone calls (of the specified type), not including calls that are still established.

- type: "callHistory"
- count: the number of calls to query, starting with the most recent.
- entryType: The types of calls to fetch: "all" or "missed".

```
GET/status
User-Agent: MyApplication
Transaction-ID: CW839937
Content-Type: application/xml
Content-Length: 120
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>callHistory</type>
<count>10</count>
<entryType>all</entryType>
</status>
```

Response

One or more callHistory tags, each containing:

- type: "missed", "received" or "dialed"
- number: The number dialed as displayed in the call history.
- displayname: If available
- duration: In seconds
- timeInitiated: Time the call attempt started, in Unix time or POSIX time (which is the number of seconds (not including leap seconds) since 00:00:00 January 1, 1970 UTC).
- id: the call ID.

```
HTTP/1.1 200 OK
Transaction-ID: CW839937
```

```
Content-Type: application/xml
Content-Length: 401
<?xml version="1.0" encoding="utf-8" ?>
<status type="callHistory">
<callHistory>
   <type>dialed</type>
   <number>6045553344</number>
   <displayName>Frank Chan</displayName>
   <duration>128</duration>
   <timeInitiated>1303932552</timeInitiated>
   <id>6b9f485e39384307b01fb995dbc5a031</id>
</callHistory>
<callHistory>
   <type>missed</type>
   <number>demo</number>
   <displayName>demo</displayName>
   <duration>1440</duration>
   <timeInitiated>1303932432</timeInitiated>
   <id>bf692b1241604cf4b144d6a7219f0533</id>
</callHistory>
</status>
```

Get Call History Entry: GET /status "callHistoryItem"

Obtain the information about a call history entry by specifying a call ID.

- type: "callHistoryItem"
- id: the call ID.

```
GET/status
User-Agent: MyApplication
Transaction-ID: CC8322937
Content-Type: application/xml
Content-Length: 76
<?xml version="1.0" encoding="utf-8" ?>
<status>
   <type>callHistoryItem</type>
   <id>06edd3abe5504ac48eb896cb4b5f4f21</id>
</status>
```

Response

- type: "missed", "received" or "dialed"
- number: The number dialed as displayed in the call history.
- displayname: If available
- duration: In seconds
- timeInitiated: Time the call attempt started, in Unix time or POSIX time (which is the number of seconds (not including leap seconds) since 00:00:00 January 1, 1970 UTC).
- id: the call ID.

```
HTTP/1.1 200 OK
Transaction-ID:
Content-Type: application/xml
Content-Length: 286
<?xml version="1.0" encoding="utf-8" ?>
<status type="callHistory">
<callHistory>
   <type>dialed</type>
   <number>1335</number>
   <displayName>Joseph Santos</displayName>
   <duration>7</duration>
   <timeInitiated>1478557308</timeInitiated>
   <id>06edd3abe5504ac48eb896cb4b5f4f21</id>
</callHistory>
</status>
```

Get Count of Missed Calls: GET /status "missedCall"

Obtain the current count for missed calls. A missed call is an incoming call that the user did not answer (if the call was picked up by voicemail, it is still considered to be a missed call).

The "current count" increments each time a call is missed.

The count is reset to zero when:

- The user clicks on the Missed Calls icon in the Bria dashboard, bringing the History panel to the front.
- This GET /status "missed calls" is sent with a 200 OK response.

The count is not reset when the History panel is brought to the front without the user clicking on the Missed Calls icon.

```
GET/status
User-Agent: MyApplication
Transaction-ID: CC8322937
Content-Type: application/xml
Content-Length: 76
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>missedCall</type>
</status>
```

Response

- count: the current number of missed calls

```
HTTP/1.1 200 OK
Transaction-ID: CC8322937
Content-Type: application/xml
Content-Length:86
<?xml version="1.0" encoding="utf-8" ?>
<status type="missedCall">
<count>2</count>
</status>
```

5.6. Voicemail and MWI

Get MWI Count: GET /status "voiceMail"

```
GET/status
User-Agent: MyApplication
Transaction-ID: PI834137
Content-Type: application/xml
Content-Length: 75
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>voiceMail</type>
</status>
```

Response

One or more <voiceMail> tags (one for each account configured for voicemail), containing:

- accountId
- accountName
- count: The number of voicemail items.
    - If voicemail is supported and MWI is supported, the voicemail count: 0 or more.
    - If voicemail is not set up on a specific account, then 0 is specified.
    - If the voicemail server does not support MWI with a count, then -1 is specified.

```

HTTP/1.1 200 OK
Transaction-ID: PI834137
Content-Type: application/xml
Content-Length: 167
<?xml version="1.0" encoding="utf-8" ?>
<status type="voiceMail">
<voiceMail>
   <accountId>0</accountId>
   <accountName>Account1</accountName>
   <count>3</count>
</voiceMail>
</status>
```

Check for Voicemail: GET /checkVoiceMail

Phone the voicemail server for the specified account ID. Bria must be configured with the voicemail phone number for the specified account (Softphone > Account Settings > Voicemail).

- accountId
- suppressMainWindow: When true, the Bria Main window remains in the background. When false, the Main window is brought to the foreground.

```
GET/checkVoiceMail
User-Agent: MyApplication
Transaction-ID: MI839837
Content-Type: application/xml
Content-Length: 93
<?xml version="1.0" encoding="utf-8" ?>
<checkVoiceMail>
<accountId>0</accountId>
<suppressMainWindow>true</suppressMainWindow>
```

```
</checkVoiceMail>
```

Response

- 200 OK if the voicemail is configured in Bria on the specified account. Bria places a call to the voicemail server.
- 200 OK if voicemail is not configured on this account. However, no call is placed (which means that no call events are posted).

```
HTTP/1.1 200 OK
Transaction-ID: MI839837
Content-Type: application/xml
Content-Length: 0
```

5.7. Contacts and Presences

Get Information for One Contact: GET /status "contact"

Find a contact in the Bria contact list that has the specified email address and return all contact information for this contact. In addition, return the address that is considered the "primary presence address" (specified on the Contact Profile in the Bria UI) and indicate whether the address is SIP or XMPP.

- type: "contact"
- email: the email address of the contact whose information you want.

```
GET /status
User-Agent: MyApplication
Transaction-ID: EO58B
Content-Type: application/xml
Content-Length: 109

<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>contact</type>
<email>fchan@zippy-voip.com</email>
</status>
```

200 Response

If there is a contact whose email address matches the contact, the response contains one contact tag, containing:

- One or more <email> tags. One of these tags will match the query email.
- Zero or more of the following address tags: fax, home, jid, mobile, other, softphone, website, work. (jid is the address on an XMPP account, if any). The address itself can include spaces and characters such as - +.
- presenceStatus tag, where the possible values are: available, away, busy, doNotDisturb, idle, notAvailableForCalls, offline, onThePhone.
- presenceText tag, containing the text from the Bria user interface. This value may be in English or another language. It may be one of the "canned" values or it may be a custom value created by the user.
- presenceAddress, containing the address tag (above) that holds the presence information. This tag will correspond to the softphone address or jid address (above).
- presenceType, where the possible values are:
  - "sip" if the contact’s SIP account is the primary presence address for this contact.
  - Or "xmpp" if XMPP account is the primary presence address for this contact.

```
HTTP/1.1 200 OK
Transaction-ID: EO58B
Content-Type: application/xml
Content-Length: 400

<?xml version="1.0" encoding="utf-8" ?>
<status type="contact">
<contact>
<email>fchan@zippy-voip.com</email>
<work>+1 (604)555-3344</work>
```

```
<work>1111</work>
<work>+1 (604) 555-8888</work>
<mobile>+1 (514)555-2222</mobile>
<jid>frank@zippy-voip.com</jid>
<presenceStatus>away</presenceStatus>
<presenceText>Idle</presenceText>
<presenceAddress>frank@zippy-voip.com</presenceAddress>
<presenceType>xmpp</presenceType>
</contact>
</status>
```

204 Response

If there are no contacts with a matching email address:

```
HTTP/1.1 204 No Content
Transaction-ID: 001
Content-Type: application/xml
Content-Length: 0
```

Details

The search stops at the first contact that is encountered that has an email address that matches the query. The response only ever returns one contact.

Get Current Presence Status: GET /status "presence"

Get the current presences status and text.

```
GET /status
User-Agent: MyApplication
Content-Type: application/xml
Content-Length: 14
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>presence</type>
</status>
```

Response

- presenceStatus: The selected state icon to be presented
- presenceText: The text displayed beside the presence status icon.

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: 24
<?xml version="1.0" encoding="utf-8" ?>
<status type="presence">
<presenceStatus>available</presenceStatus>
<presenceText>at work</presenceText>
</status>
```

Get Information about Supported Presences: GET /status "supportedPresence"

Get information about supported presence options.

```
GET /status
User-Agent: MyApplication
Content-Type: application/xml
```

```
Content-Length: 14
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>supportedPresence</type>
</status>
```

Response

One or more <session> tags, each containing information about the supported presence and related states:

- presenceStatus : A list of available presence status states

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Length: 24
<?xml version="1.0" encoding="utf-8" ?>
<status type="supportedPresence">
<presenceStatus>available</presenceStatus>
<presenceStatus>busy</presenceStatus>
<presenceStatus>away</presenceStatus>
<presenceStatus>onThePhone</presenceStatus>
<presenceStatus>notAvailableForCalls</presenceStatus>
<presenceStatus>doNotDisturb</presenceStatus>
<presenceStatus>offline</presenceStatus>
</status>
```

Set Presence Status: GET /setPresence

Change the current presence status and displayed presence txt.

- presenceStatus: The selected state icon to be presented
- presenceText: The text displayed beside the presence status icon

```
GET /setPresence
User-Agent: MyApplication
Content-Type: application/xml
Content-Length: 835
<?xml version="1.0" encoding="utf-8" ?>
<presence>
<presenceStatus>away</presenceStatus>
<presenceText>lunch</presenceText>
</presence>
```

Response

200 OK if Bria was able to parse the request, a 4xx if Bria could not parse it.

```
HTTP/1.1 200 OK
Transaction-ID:
Content-Type: application/xml
Content-Length: 0
```

5.8. Handling Instant Messaging

Start an IM session: GET /im

Open the IM window in Bria and initiate an IM session to the specified address using the specified protocol.

- im type: "sip" or "xmpp"
- address: The address for the contact, a SIP address if the im type is "sip" or a jid if the im type is "xmpp".

```
GET /im
User-Agent: MyApplication
Transaction-ID: 4VEZ322
Content-Type: application/xml
Content-Length: 105

<?xml version="1.0" encoding="utf-8" ?>
<im type="xmpp">
<address>frank@zippy-voip.com</address>
</im>
```

Response

200 OK if Bria was able to parse the request, a 4xx if Bria could not parse it.

```
HTTP/1.1 200 OK
Transaction-ID: 4VEZ322
Content-Type: application/xml
Content-Length: 0
```

5.9. Handling Screen Sharing

Start a Screen Sharing session: GET /startScreenShare

Start screenshare on default monitor.

- invitees: A list of screen share attendees
- address type: The address for the contact, a SIP address if the address type is "simple" or a jid if the address type is "xmpp".

```
GET /startScreenShare
User-Agent: MyApplication
Content-Type: application/xml
Content-Length: 835
<?xml version="1.0" encoding="utf-8" ?>
<invitees>
<invitee>
<address type="xmpp">jeff@counterpath.com</address>
</invitee>
<invitee>
<address type="simple">john@counterpath.com</address>
</invitee>
</invitees>
```

Response

200 OK if Bria was able to parse the request, a 4xx if Bria could not parse it.

```
HTTP/1.1 200 OK
Transaction-ID:
Content-Type: application/xml
Content-Length: 0
```

Get Information about Current Screenshares: GET /status "screenShare"

Get information on active screenshare.

```
GET /status
User-Agent: MyApplication
Content-Type: application/xml
Content-Length: 14
<?xml version="1.0" encoding="utf-8" ?>
<status>
<type>screenShare</type>
</status>
```

Response

One or more <session> tags, each containing information about the screen sharing session:

- status: The status of screen share will return if active
- joinUrl: The url provided to invitees to attend the screenshare

```
HTTP/1.1 200 OK
Transaction-ID:
Content-Type: application/xml
Content-Length: 185
<?xml version="1.0" encoding="utf-8" ?>
<status type="screenShare">
<session>
<status>active</status>
<joinUrl>http://join.softphone.com/CLMSXF6KOEUQTZEE</joinUrl>
</session>
</status>
```

6. Connecting to Bria via Pipe

The pipe functionality of the API has been deprecated. Although the Bria Desktop API continues to support its use, no more development or changes will occur. CounterPath recommends that you avoid implementing any new features using apipipe.

You can connect to Bria by establishing a pipe connection to the named pipe "apipipe". This pipe is full-duplex and message-oriented. No port is used to connect to Bria; you only need a pipe name to establish a connection.

You can make unlimited connections to the same instance of Bria, but all connections will receive the same information (events and responses) from Bria.