Bria 3.0 Provisioning Guide
Retail Deployments

Contents

About this Manual

This manual describes the *mechanism* of remote login/provisioning. It describes how to set up a server (or servers) for the remote login and optionally the remote provisioning, remote update and remote upgrade features of Bria:

* Remote login controls access to the application; the softphone will not start until the user has logged in.
* Remote provisioning lets you configure the softphone remotely.
* Remote update lets you change the configuration of a given deployment of Briaat runtime (outside of the login process).
* Remote upgrade lets you deploy upgrades of the software remotely.

This manual is intended for:

* System administrators who have purchased Bria from the CounterPath website and are deploying Bria in their enterprise using remote logging and (optionally) remote provisioning. (See Bria Administrator Guide for other ways of deploying.)
* Service providers who have purchased Bria from CounterPath Sales, without further customization or engineering changes.

This manual is intended to be read in conjunction with:

* "Bria 3.0 Configuration Guide - *Retail Deployments*", which describes the features that can be configured through remote provisioning.

1 About Provisioning

1.1 Provisioning Functions

Provisioning of Bria includes the following features:

- Controlling access to the VoIP service through a remote login.
- The ability to provide a license key remotely. below
- Updating the Bria configuration (changing the factory defaults). Bria can be configured differently for each user. This feature is optional. See page 15.
- Providing upgrades to the executable by making new versions of Bria available to each Bria installation to download. This feature is optional. See page 15.

1.2 Licenses

When you obtain Bria, you purchase a license with a specified number of seats.

Scope of Licenses

A license can be shared by users on the same computer if the users are using the Windows administrator or regular user accounts. However, a user who uses this computer with the Windows guest account and starts Bria will automatically draw down the license count (assuming that a license key has already been entered).

Drawing Down of Licenses

Each time a user enters the license key, the license count is drawn down on the CounterPath license database. When the count is drawn down to 0, then the next time the key is entered, an error message appears for that user.

You can either increase your license count or revoke unused seats. To revoke seats, go to ww.counterpath.com, click the Store link, click the Your Account link, and log in.

As described above, use of a Windows guest account draws down the license count. Therefore, if you seem to have drawn down more license counts than expected, the problem may be that one or more guests have used seats. You can request that CounterPath revoke these licenses in order to reinstate the number of seats actually in use.

Provisioning and Managing Licenses

If you use one of the remote login profiles, you can provision the license at initial login, then manage the license at further logins, as desired. See page 13 for more details.

Setting up your Service for the Licensing Server

Periodically, Bria connects to CounterPath's license server in order to verify that a valid license is being used. Therefore, at all times, Bria will need to have an internet connection.

Bria connects to https://secure.counterpath.com via port 443; make sure your firewall allows this HTTPS traffic to this URL. In addition, if you have explicitly set a web proxy (Start > Control Panel > Internet Options > Connections) then Bria will use this proxy; make sure the proxy allows this traffic.

1.3 What Provisioning Does: Writing to Settings

Each provisioning function involves writing to settings stored on Bria computer. These settings control the behavior of various features of Bria. For example, a successful login request will result in the creation of new settings representing the account. A remote update may result in changing the value of existing settings.

For detailed information on settings and the features they control, see "*Bria 3.0 Configuration Guide - Retail Deployments*".

1.3.1 Provisioned Settings Overwrite GUI Settings

Settings are assigned values in several ways:

- A setting has a default "factory" value.
- Some settings can be changed by the user on the GUI.
- Remote provisioning lets you can change the value of any setting.

At startup, the factory values are loaded, then the user overrides are loaded (overwriting factory values), and finally values that you send through the provisioning response are loaded (overwriting factory or user values). At shutdown, the current user overrides and provisioning overrides are persisted to the user file.

Keep in mind that provisioned settings override user settings. A user may complain that they change a value on the GUI but each time they restart Bria, their changes are lost: you are probably overwriting their value when you provision.

The Bria Settings reference documentation (a Microsoft® Excel® document) includes a column that identifies settings that are represented on the softphone GUI.

1.3.2 Syntax of Settings

Each setting has a fully qualified name: <domain>:<section>:<setting>

For example, proxies:proxy0:register.

The syntax for setting values via provisioning is:

<domain>:<section>:<setting>=<"value">

For example, proxies:proxy0:domain="domainA.com"

- The value of the variable must appear in double quotes.
- Always a string. True is represented by "true" or "1". False is represented by "false" or "0".
- The Bria process that interprets the settings ignores the case of the value (uppercase or lowercase), except for literals such as display names.

1.4 The Mechanism of Remote Provisioning

Each remote provisioning service involves an exchange between the login server and an individual Bria client. The exchange is performed over HTTP or HTTPS.

1.4.1 Servers

You must deploy servers to handle the provisioning requests:

- The "login server": a server to handle login requests, if you decide to implement login. This server is simply a web server that, at a minimum, can serve one plaintext or XML file.
- The "update server": a server to handle remote update.
- The "upgrade executable server": a server to handle remote upgrades of the Bria application.

These server roles may in fact all be deployed on the same physical server: that is your decision.

You must set these servers in Bria.

- The login server is either discovered through DHCP or manually entered by the user on the Login dialog, as described in "Specifying the Login Server" on page 10.
- You will set the update server and upgrade executable server (if they are being used) in the provisioning response that you set the first time the user logs in.

1.4.2 Bria-to-Server Exchange

The exchange between Bria and the appropriate server involves the following:

- When the appropriate trigger occurs, Bria sends an HTTP or HTTPS request to the server. For login, the trigger is the user pressing OK on the Bria login dialog. For remote upgrade, the trigger is startup of the softphone.
- The server responds.
- Bria reads the response and takes the appropriate action: starts the softphone and registers with the SIP proxy, or finds and installs the upgrade.

Use of Scripts and Macros

You may want to run an appropriate script on the given web server, to provide the information required by Bria. To run a script, include it in the URL for that server.

Running scripts usually requires information about the user's deployment. The URL for the appropriate server can include macros. When Bria contacts the server, it replaces the macros with the real data and includes this information in the HTTPS request.

Your script must understand the names assigned to the macros.

For example a URL of:

```
https://mycustomloginserver.com/login.php?platform=$platform$&lic=$license$
```

might become this POST used to log in the user:

```
https://mycustomloginserver.com/login.php
--------------------------------------------------------------------------------
Username=21187
Password=rosebud
platform=win32
lic=d3874ihfd8t23975v1iu5182ruity3iusapor236u545uye0r9qwjj
```

Note that "Username" and "Password" (with initial capitals) are always sent in a login POST; the URL does not have to include macros for this data.

See "Script Samples" on page 21 for samples of some of the scripts that are mentioned in this manual.

See "Macros" on page 22 for a list of macros that Bria supports.

1.4.3 Communication Mechanism

All communications between Bria and the login server are performed over HTTP or HTTPS, as follows:

*   Custom login uses POST.
*   Remote update and remote upgrade use GET.

The remote provisioning mechanism does not support redirect.

If using HTTPS, you need a trusted certificate (not self signed). Bria will only accept certificates whose authenticity can be verified through the trust chain.

1.4.4 Data Format

All the data included in the GET or POST response is in a specific format. This format is similar to that of Microsoft® Windows®.ini files.

The information is organized into three portions, which must appear in this order:

- [DATA]
- [SETTINGS]
- [##MEMORY##]

Example

```
[DATA]
Success=1
[SETTINGS]
proxies:proxy0:display_name="kokila"
proxies:proxy0:enabled="1"
proxies:proxy0:username="kperera"
proxies:proxy0:password="dfher43d89dhferuieo98375uy8"
proxies:proxy0:domain="domainA.com"
```

[DATA]

This section contains the response to requests:

Success=<value>, a boolean. This data is required.

Failure=<message>, which is optional if the success is 0. For login, the string you enter here will be displayed in the Login dialog.

[SETTINGS]

This section contains settings to be written to persistent memory. The values will be used immediately.

At shutdown, these settings will be written to the local settings file on the Bria computer.

[##MEMORY##]

This section contains settings to be written to non-persistent memory. The values will be used immediately, but only for the current session.

At shutdown, these settings will not be written to the local settings file.

CRLF

The response must end with a CRLF. If this is missing, the last line of the response is ignored.

Handling and Encryption of Passwords

All "password" settings in any domain/section are handled as follows:

- Bria does not interpret passwords in any way, so the value the login server passes to Bria can be encrypted.
- Bria encrypts the value before storing it, regardless of whether or not it is already encrypted. When a stored value is read in order to pass it to the login server, it is first decrypted.
- When a password that the user has been entered into a dialog is then passed to the login server, Bria does not encrypt the value.

1.4.5 Example of an Implementation

The hardware requirements of the login server depend on what the server will do. If it will have a complicated backend database and processing in order to retrieve the settings that are to be provisioned, then the server should be of higher processing capabilities. Regardless, the login server is simply a web server and it only needs to serve one file for provisioning; this file is in plaintext or XML format.

The login server could be a Linux® machine with an Apache™ web server or a Microsoft® Windows® machine with an IIS web server.

For their internal deployment, CounterPath uses Debian® Linux with Apache version 2. The login server is a Pentium® 4 with 3GHz processor. This server scales to thousands of requests per second. We use the internal database of the SIP proxy (this can be a MySQL® database) which contains all usernames and passwords. The provisioning response is constructed based on login information retrieved from Bria via the login PHP script.

2   Login and Configuring

2.1 Credentials Required

2.1.1 Types of Credentials

Login Credentials

Login refers to the process of signing into the VoIP service. The Bria user must enter login credentials – user name and password – in order to access to Bria.

Login credentials cannot be changed through provisioning.

SIP Account Credentials

Once the user has logged in, the SIP account credentials allow the user to register for your VoIP service; they are known to your SIP registrar. These credentials are user name, password, and the optional authorization user name.

SIP credentials are always required. They can be changed through provisioning.

XMPP Account Credentials

Once the user has logged in, the XMPP credentials allow the user to access the XMPP service; they are known to the XMPP server. These credentials are user name and password.

XMPP credentials are required only if you support XMPP accounts. They can be changed through provisioning.

2.1.2 Providing Credentials

When setting up a new user, give the user their login credentials, outside of Bria. You do not give the user the account credentials; instead, these credentials will be sent down through provisioning.

The account user name and login user name can be identical or different.

- The login user name is meaningful to the user (for example, their own name).
- The account user name follows the syntax for your accounts – it may be a number or words.

The account password and the login password are typically different for security reasons.

- The login password should not be encrypted, because the user will enter it manually.
- The account password does not have to be human-readable.

The authorization user name (one of the account credentials but not a login credential) is optional.

- An authorization user name is useful, for example, if you allow usernames that are short and therefore easy to guess. The authorization user name is used in place of the user name to register the account with the proxy. It provides an added layer of security.

- The authorization user name is used only to register the account. It does not replace the account user name in identifying the user to the outside world.
- If you use an authorization name, make sure it is different from the user name!

2.2 Specifying the Login Server

You can identify the login server in one of these ways:

- Bria discovers login server via DHCP.
- Users enter login server manually. You might choose to use this scenario, for example, in the following situations:
    - You do not want to set up DHCP in your enterprise.
    - You have set up DHCP but one of your users is using Bria for the first time on a computer that is not on the enterprise LAN; for example, the user is traveling and using a new laptop.

Login with DHCP Discovery

Your DHCP server must be set up so that option 120 specifies the URL of the login server.

When Bria starts, Bria detects that a value is specified in option 120. Therefore, the Login dialog appears with "DHCP" specified in the Profile field. The user should enter their login credentials and click Login. Bria attempts to contact the server specified in option 120. If the server is contacted, then Bria attempts to log into that server, as described in "The Login Process" on page 12. If the login is successful, then account credentials and other data are sent through provisioning.


Fallback Behavior

The next time the user logs in, Bria goes through DHCP discovery. If discovery fails (for example, the user is temporarily not on the enterprise LAN), then the Login profile reappears. The user should choose "Manually configure" and enter the server URL and their login credentials in order to log in. The server URL is saved for use the next time the user starts Bria outside the network.

The next time the user logs on from inside the network, they should remember to set the Profile back to DHCP. In this way, if you ever change the server URL, Bria automatically finds the new server.

Login with Manually Entered Server

In this case, you must provide users with the URL of the provisioning server. The URL should include any scripts and macros you are using.

When Bria starts, Bria performs DHCP discovery and fails to find a value (because you have not set up DHCP on your network). Therefore, the Login dialog appears with "Manually configure" specified in the Profile field.
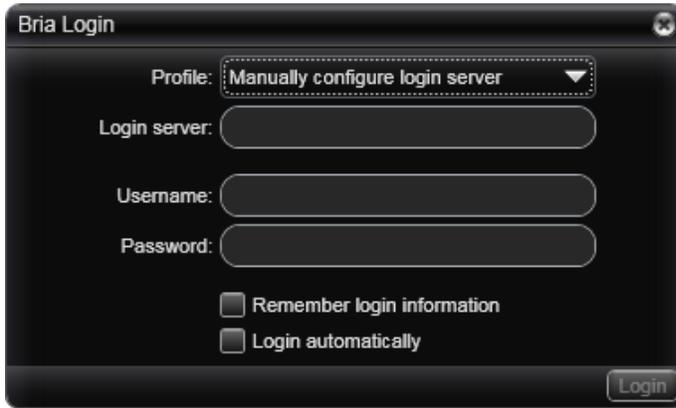
The user should enter the server URL and their login credentials and click Login. Bria attempts to login to that server, as described in "The Login Process" on page 12.

If the login is successful, then account credentials and other data are sent through provisioning. This data, along with the manually entered URL for the login server, is stored locally on the Bria computer. The next time the user logs in, the locally stored server URL is used.

2.3 Changing the Login Profile

The Preferences > Application panel in Bria lets user change the login behavior by checking the Enable login screen field.

Keep in mind that this field is intended to allow you, the administrator, to experiment with login during the deployment phase; it is not intended to allow users to skip login, for example, by displaying the Login dialog and choosing the "No login required" option.

If a user first logs on using DHCP or by manually entering the server URL and then later changes to "No login required", Bria will start but none of the user's account credentials or account settings will be available, so Bria will not be usable.

2.4 The Login Process

The login server must be set up to handle the following procedure.
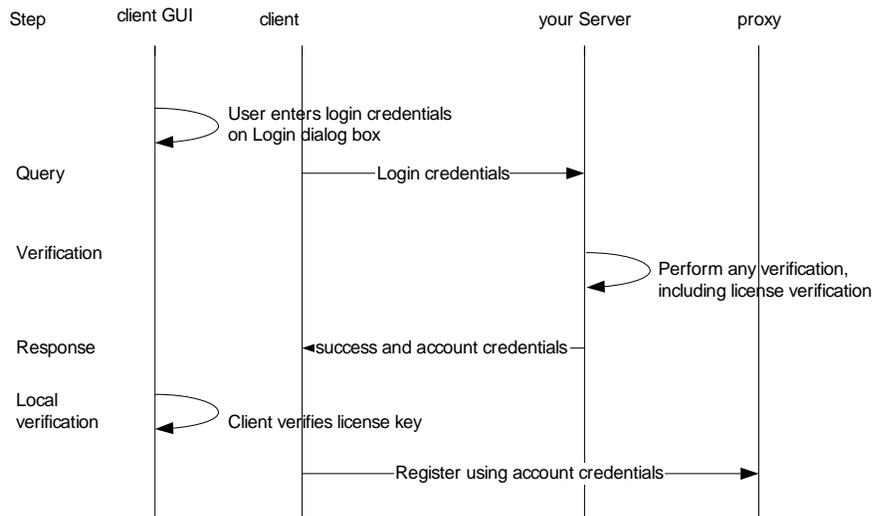
Login Procedure Is Invoked

The Login dialog box is displayed. The user enters the required information and clicks Login.

Query Step

Bria sends the data from the Login dialog. The data is encoded application/x-www-form-urlencoded.

The data is sent to the login server (the server specified in feature:custom_login:server) in an HTTP POST. The value will be blank if the branded Login dialog box does not include the corresponding field.

For example a URL of:

```
https://mycustomloginserver.com/login.php?platform=$platform$&cid=$computerid$
```

might become this POST used to log in the user:

```
https://mycustomloginserver.com/login.php
-----------------------------------------------------------------------------------
Username=21187
Password=rosebud
platform=win32
cid=d3874ihfd8t23975v1iu5182ruity3iusapor236u545uye0r9qwjj
```

where:

- "Username" and "Password" (with initial capitals) are always sent in a login POST; the URL does not have to include macros for this data.
- platform and computerid are macros used by the login script; see "Use of Scripts and Macros" on page 5 and see page 22 for a list of macros.

License Key Management

You may want to maintain license key information such as the distribution of licenses to users and computers.

To help you maintain this information, you can capture data by including the appropriate macro in the login URL.

The key information that you should maintain is the username and the computer ID. Capturing the computer ID lets you match the license key information you maintain with the license key information on the CounterPath Store.

For example, when a user logs on, you can use the username and computer ID to determine whether or not you need to send down a license key in the provisioning response: Is this a valid user? Is there already a license key on this computer? If you decide yes, then you can add the username and the computer ID to your database.

As another example, if some users log on at various computers and you have a policy that a user can draw down a maximum of 3 keys, then capturing the combination of the username and computer ID lets you implement that policy.

Finally, if you revoke the license keys for a user who no longer valid, you can go to your login database and look up the computer IDs associated with all the license keys that this user has obtained. You then go to the Store and revoke the licenses for these computer IDs.

Verification Step

Your login server should perform any suitable verification on the sent data, according to your business rules.

Typically, this verification will include a check for the license key, using the username and computer ID, as described above. You may decide to send down a license key or (for an existing deployment) you will determine that there is no need to send down a license key.

Response Step: Failure

If there is a problem with any of the data, the login server should return failure data in the following format:

```
[DATA]
Success=0
Failure="<message> "
<CRLF>
```

Example that includes a license key being sent down to the computer:

```
[DATA]
Success=1
LicenseKey="e48jey45379ryeioo8a7e934q8dhfudufoladskiuwb"
[SETTINGS]
proxies:proxy:user_name="KPerera"
[##MEMORY##]
proxies:proxy0:password="rosebud"
<CRLF>
```

where:

- Success: this line is required.
- LicenseKey: the license key for the computer. This data should be sent for a new deployment or if you want to change the existing license key.
- Settings: the username will be saved at shutdown.
- ##Memory##: the password will not be saved at shutdown.
- The response must end with a CRLF.

License Key Verification by Bria

Bria next takes one of these actions, depending on the response received from the server:

- If the response was a failure, then the Login dialog appears again. The process goes back to "Login Procedure Is Invoked" on page 12.
- If the response was a success, then Bria checks if there is a license key in the response.
    - If the response includes a key, the key is validated and then Bria starts.
    - If the response includes a key, and validation fails, then the Enter License Key dialog appears.
    - If the response does not include a license key, then Bria checks if there is already a key stored on this computer.
        - If yes, then Bria verifies that the key is valid and then starts.
        - If no, then the Enter License Key dialog appears. When the user enters the license key (obtained outside of Bria, for example in an email sent to all new customers), the Bria verifies that the entered license key is valid. If the key is valid, Bria starts.
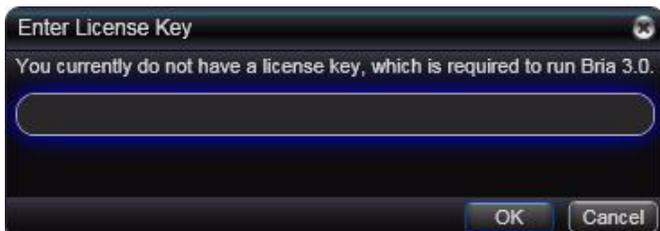
3   Updates and Upgrades

Remote Updates

You can configure Bria to check with the update server at specified intervals for changes to the user's settings.

Remote Upgrades

There are two ways to support upgrades to Bria.

Use CounterPath Upgrade Server

The default behavior is to obtain upgrades from the CounterPath upgrade server. Whenever CounterPath puts a new executable on its upgrade server, your users' Bria will automatically download the upgrade and prompt the user to install it.

No work on your part is required for this option. Bria is already configured to do this. You do not have to set up any servers.

Set up your Own Upgrade Server

You can set up an upgrade server and deploy upgrades yourself. One of the advantages of this option is that you can test the new executable on a few deployments before rolling it out to all your users. See page 18.

3.1 General Setup

Remote update and remote upgrade are controlled by Bria settings. To change the default values to values suitable for your deployment, change them through remote provisioning, the first time the user logs in. After that, change them as necessary through remote provisioning or remote update.

Domain:Section	Setting	Comment
feature:auto_update	code_server_url	The web server for remote upgrades of the executable. Default is empty.
feature:auto_update	config_server_url	The web server for remote update. Default is empty.
feature:auto_update	block_timer_t3_s	See below for a description. Default is 10 seconds. Typically, leave the default.
feature:auto_update	deffer_timer_t2_s	See below for a description. Default is 60 seconds. Typically, leave the default.
feature:auto_update	update_check_initial_t1_s	See below for a description. Default is 20 seconds. Typically, leave the default.

feature:auto_update	update_check_t1_s	See below for a description.
		Default is 86400 seconds (24 hours).
		Typically, leave the default.
feature:auto_update	timer_factor	See below for a description.
		Default is 1.00

Timer Settings

Remote upgrades and remote updates rely on four timers in the user's settings. The timers control how frequently Bria contacts the update and upgrade executable servers.

All values are in seconds. You can use the timer_factor setting to convert the values on your side into seconds.

- update_check_initial_t1_s.
- update_check_t1_s.
- deffer_timer_t2_s.
- block_timer_t3_s.

Automatic checks for remote upgrades and automatic checks for remote updates are performed at the same point: when the user starts Bria. The interaction of the timers occurs as follows:

1. Bria starts.

2. The timer update_check_initial_t1_s starts.

3. When update_check_initial_t1_s expires, Bria checks its state of business (whether or not the user is busy with a call or instant message session).

   - If Bria is busy, deffer_timer_t2_s starts. When this timer expires, Bria checks its business again. deffer_timer_t2_s continues restarting and expiring until Bria is no longer busy (when the user hangs up from an active call).

   - If Bria is not busy, it contacts the servers.

4. The timer block_timer_t3_s is set when a check is initiated at Bria. Another check is not allowed as long as block_timer_t3_s is still active. This timer ensures that provisioning checks are not performed too often, and is especially useful for protecting against potential hacker requests (which may arrive with frequency). The timer block_timer_t3_s is typically shorter in duration than update_check_t1_s.

5. After the first check, the cycle starts over at step 1 using the timer update_check_t1_s (not update_check_initial_t1_s).

Changing Timer Settings

New timer settings will take effect as follows:

- If the timer is not running when the server sends new settings (and they are saved on the Bria computer), then the setting take effect immediately. The next time the timer is loaded, the new setting will be used.
- If the timer is running, the new setting takes effect after the timer expires and is reloaded. This means if update_check_t1_s still has 23 hours to go, it will be changed only after 23 hours. However, if the session is restarted, the new setting will take effect.

3.2 Remote Update

3.2.1 Setting Up

• Set up Bria as described on page 15.

• Set up the update server to handle the procedure described below.

3.2.2 How Remote Update Is Performed

Assuming that the timers are not all set to zero, this procedure runs "in the background" for as long as Bria is running.

1. When triggered by the timer, Bria checks for remote updates by sending a GET to the update server

   For example, the value of feature:auto_update:config_server_url might be:

   https://myupdatesettingsserver.com?language=$language$&build=$build$&name=$loginame$

   This URL could result in a GET to your web server of:

   myupdatesettingsserver.com?language=EN&build=16835&name=kperera

2. The update server must response with the following:

   [DATA]
   Success=0
   <CRLF>

   or

   [DATA]
   Success=1
   [SETTINGS]
   feature:auto_update:update_check_t1_s="3600"
   <CRLF>

   where:

   • success: 1=true (there are updates) or 0=false (there are no updates).

   • The [SETTINGS] section contains the changed settings. See "Data Format" on page 7 for details.

   • The response must end with a CRLF.

3.3 Remote Upgrade

3.3.1 Setting Up

• Set up Bria as described on page 15.

• Set up an upgrade server as follows:

  • You can use a script to include logic that determines a given deployment needs an upgrade. See below for an example. Obtain the sample upgrade script from CounterPath and modify it to suit your needs.

    Or you can skip the script and manually set up your upgrade server to simply provide a success response when an upgrade is available and a failure response at other times.

  • If you are using scripts, set the URL for the upgrade server to include the script and any macros (for example, the language and the build macros).

  • When you want to deploy an upgrade, place it on the "upgrade location".

3.3.2 How Remote Upgrade Is Performed

Assuming that the timers are not all set to zero, this procedure runs "in the background" for as long as Bria is running.

Bria Sends a GET

When triggered by the timer, Bria checks for available upgrades by sending a GET to the upgrade executable server.

• For example, if you are using scripts, the value of feature:auto_update:code_server_url might be:

  https://executablepgradeserver.com/exe_upgrade.php?build=$build$&language=$language&name=$loginame$

  This URL could result in a GET to your webserver of:

  https://executablepgradeserver.com/exe_upgrade.php?build=38740&language=USEnglish&name=kperera

• Or if you are not using scripts, the value is simply the URL of the upgrade server:

  https://executablepgradeserver.com

Server Response

The upgrade executable server must respond with the following:

```
[DATA]
Success=0
<CRLF>
```

  or

```
[DATA]
Success=1
Mandatory=1
version=60000
url=https://executableupgradeserver.com/newversion.exe
<CRLF>
```

where:

- `Success`: 1=true (there is an upgrade) or 0=false (there is no upgrade).

- `Mandatory`: 1=true. This response is optional; the default is "0". Bria handles the upgrade differently depending on this response; see below.

- `version`: identifies a build stamp set by Bria during build time. Bria uses this version to determine whether to prompt the user to install the upgrade; see step .

- `url`: the absolute path to the installer software for the new version.

- The response must end with a CRLF.

The response **cannot** include a [SETTINGS] section. In other words, none of the user's current settings can be changed via this response.

If no upgrades are found, Bria will recheck periodically for available upgrades. See "Timer Settings" on page 16 for details.

Handling of the Upgrade

If an upgrade is available, Bria compares the build number of the application on the user's computer to the build number specified in the response (60000 in the above example).

- If the response has the same number, Bria does not prompt the user to download
- If the response has a *different* number, Bria prompts the user to download the upgrade.
  - If the user initiates the download, Bria will download the installer and save it to the local Bria program folder. Bria will also prompt the user to exit in order to install the new version. The user can install immediately or postpone installation.
  - If the user declines the upgrade and the upgrade is optional, Bria will enter its timing cycle and display the download prompt again at the appropriate time. See "Timer Settings" on page 16.
  - If the user declines a mandatory upgrade, Bria shuts down
  - If the user declines with "do not ask me again" (possible only with an optional upgrade), Bria will not check again for upgrades during the session.

"Install Later" Handling

If the user declines to install the downloaded version, then the next time Bria is started the user will be prompted to install the newer version.One of the following will occur:

- If the user initiates the installation, Bria will install the new (local) version.
- If the user declines, Bria will start the original version and will enter its timing cycle, displaying the download prompt again at the appropriate time. See "Timer Settings" on page 16.
- If the user declines with "do not ask me again", Bria will start the original version and will not prompt to install again during the session.

Bria starts the version installed most recently. The automatic check scenario will be initiated as described in the previous section. The downloaded installer will not be deleted, to enable manual rollback, if required.

A Script Samples

Contact CounterPath to obtain sample scripts.

> These sample scripts, written in PHP, are intended to illustrate a possible implementation. They are not intended to be used without modification. You should write scripts suitable to your environment, in an appropriate scripting language.

login.php

Custom login script. Bria passes in the username and password. After verification, if the login credential is correct, the server will write the proper settings into the response and send it back to Bria.

See "Use of Scripts and Macros" on page 5 for an example that uses this script.

exe_upgrade.php

Bria passes in the buildstamp. You may want to revise the script to also pass in the platform. It returns success is true or false, plus the URL where the upgrade of the Bria executable is located.

See "Remote Upgrade" on page 18.

B Macros

Macro	Description	Value
$acc_passwdn$	where n is an account. The password for the specified SIP account (for deployments that support more than one SIP account). Stored as a setting.	
acc_usern	where n is an account. The username for the specified SIP account (for deployments that support more than one SIP account). Stored as a setting.	
$build$	The unique buildstamp.	For example, 12345
$computerid$	Unique ID for this computer	
$computername$	From the operating system	
$hashlicense$	A hash of the license key. This allows the license key to be sent in a secure way	
IP	The IP address of this computer	
$language$	The language of the installed application.	USEnglish, French, German, Italian, Portuguese-Brazil, Spanish,
$license$	The license key.	
$loginname$	The login username. This is the username the user enters in the Login dialog and is not necessarily the same as the SIP username. See page 9.	
$loginpassword$	The login password. This is the password the user enters in the Login dialog and is not necessarily the same as the SIP password. See page 9.	
MAC	The MAC address of the machine running Bria.	
$platform$	The operating system platform.	win32, mac, UNDEFINED.
$release$	The two-digit release.	For example, 2.5.
$winusername$	The Microsoft Windows user name	
$winversion$	The Microsoft Windows version.	WINNT4, WIN2K, WINXP, WINVISTA, OTHEROS.