Cisco BroadWorks

Busy Lamp Field

Interface Specification

Release 23.0

Document Version 2

Notification

The BroadSoft BroadWorks has been renamed to Cisco BroadWorks. Beginning in September 2018, you will begin to see the Cisco name and company logo, along with the new product name on the software, documentation, and packaging. During this transition process, you may see both BroadSoft and Cisco brands and former product names. These products meet the same high standards and quality that both BroadSoft and Cisco are known for in the industry.

Copyright Notice

Trademarks

Document Revision History

Release	Version	Reason for Change	Date	Author
14.0	1	Created document.	February 19, 2007	Martin Perron
14.0	1	Edited document.	March 26, 2007	Patricia Renaud
15.0	1	Updated document for Release 15.0.	June 22, 2008	Martin Perron
15.0	1	Edited changes and published document.	July 15, 2008	Andrea Fitzwilliam
16.0	1	Updated document for Release 16.0.	June 8, 2009	Martin Perron
16.0	1	Edited changes and published document.	July 16, 2009	Andrea Fitzwilliam
16.0	2	Updated the following sections for EV 95026: ▪ *8.1 IP Phone Power Up* ▪ *8.1.3 Step 1.3: AC Busy Lamp Field Resource List Subscription*	July 22, 2009	Roberta Boyle
16.0	2	Edited changes and published document.	September 29, 2009	Andrea Fitzwilliam
17.0	1	Updated document for Release 17.0.	February 23, 2010	Martin Perron
17.0	1	Edited changes and published document.	March 24, 2010	Margot Hovey-Ritter
18.0	1	Updated document for Release 18.0.	October 6, 2011	Eric Bernier
18.0	1	Edited changes and published document.	November 22, 2011	Jessica Boyle
18.0	2	Changed section *8.2.1 Step 2.1: A-1 Calls B* for EV 164144.	June 8, 2011	Joël Collin
18.0	2	Edited changes and published document.	June 13, 2012	Jessica Boyle
19.0	1	Updated document for Release 19.0. This is a major change to include the now-supported redundancy for this feature.	September 24, 2012	Joël Collin
19.0	1	Edited changes and published document.	October 12, 2012	Patricia Renaud
19.0	2	Updated references to release number and republished document.	June 10, 2013	Andrea Fitzwilliam
20.0	1	Updated document for Release 20.0.	August 19, 2013	Joël Collin
20.0	1	Edited changes and published document.	October 8, 2013	Joan Renaud
21.0	1	Updated document for Release 21.0 and integrated changes requested for EV 235263.	October 7, 2013	Joël Collin

Release	Version	Reason for Change	Date	Author
21.0	1	Updated copyright notice and trademarks. Edited changes.	October 14, 2014	Joan Renaud
21.0	1	Updated BroadSoft and Cisco BroadWorks logos. Published document.	October 22, 2014	Joan Renaud
21.0	2	Updated server icons and published document.	February 26, 2015	Joan Renaud
22.0	1	Updated document for Release 22.0.	August 18, 2016	Joël Collin
22.0	1	Edited changes and published document.	November 24, 2016	Joan Renaud
23.0	1	Updated document for Release 23.0. Added changes from FR-17622.	August 22, 2018	Joël Collin
23.0	1	Rebranded document for Cisco. Edited changes.	September 17, 2018	Michael Boyle
23.0	1	Reviewed changes and published document.	September 26, 2018	Patricia Renaud
23.0	2	Completed rebranding for Cisco and republished document.	March 4, 2019	Jessica Boyle

Table of Contents

Table of Figures

1 Summary of Changes

This section describes the changes to this document for each release and document version.

1.1 Changes for Release 23.0, Document Version 2

This version of the document contains the following changes:

- Completed rebranding for Cisco.

1.2 Changes for Release 23.0, Document Version 1

This version of the document contains the following changes:

- Updated references to the new release number.
- Modified section *6.2.2 Redundancy* to include the system-level parameters introduced in Release 23.0 by *Busy Lamp Field and Application Server Redundancy Container Option Conversions Feature Description* [6].

1.3 Changes for Release 22.0, Document Version 1

This version of the document contains the following change:

- Updated references to the new release number.

1.4 Changes for Release 21.0, Document Version 2

This version of the document contains the following change:

- Updated server icons.

1.5 Changes for Release 21.0, Document Version 1

This version of the document contains the following changes:

- Updated references to the new release number.
- Updated the document to clarify the behavior on server startup.
- Updated BroadSoft and BroadWorks logos.

1.6 Changes for Release 20.0, Document Version 1

This version of the document contains the following change:

- Updated references to the new release number.

1.7 Changes for Release 19.0, Document Version 2

This version of the document contains the following change:

- Updated references to the new release number.

1.8 Changes for Release 19.0, Document Version 1

This version of the document contains the following change:

- In this release, support for redundancy was added to the Busy Lamp Field feature. This document was updated to include provisioning of redundancy support as well as explanations of supported redundancy scenarios.

1.9 Changes for Release 18.0, Document Version 2

This version of the document contains the following change:

- Updated section *8.2.1 Step 2.1: A-1 Calls B* for EV 164144.

1.10 Changes for Release 18.0, Document Version 1

This version of the document contains the following change:

- Added support for transport override to Transmission Control Protocol (TCP) for large notifications.

1.11 Changes for Release 17.0, Document Version 1

There were no changes to this document for Release 17.0.

1.12 Changes for Release 16.0, Document Version 2

This version of the document contains the following changes:

- Updated the following sections for EV 95026:
  - *8.1 IP Phone Power Up*
  - *8.1.3 Step 1.3: AC Busy Lamp Field Resource List Subscription*

1.13 Changes for Release 16.0, Document Version 1

There were no changes to this document for Release 16.0.

1.14 Changes for Release 15.0, Document Version 1

There were no changes to this document for Release 15.0.

1.15 Changes for Release 14.sp5

This version of the document contains the following change:

- Support for the dialog event package (*RFC 4235*) is enhanced, specifically with the addition of direction attribute.

2 Introduction

The ability to support Busy Lamp Field (BLF) is one of the building blocks for the Attendant Console application. Although SIP (*RFC 3261*) by itself offers no inherent semantics for supporting Busy Lamp Field, when coupled with the appropriate instantiation and the appropriate extension of the "SIP Specific Event Notification" framework (*RFC 3265*), the Busy Lamp Field feature can be deployed quite easily in a distributed network.

The purpose of this document is to present general requirements for access devices to support the Busy Lamp Field feature hosted on a Cisco BroadWorks Application Server (AS) using SIP as the line-side access protocol. Note that the Cisco BroadWorks Application Server uses a SIP back-to-back user agent model for service delivery, and this model is assumed in the mechanism proposed. The Cisco BroadWorks Application Server is the customer-facing and endpoint-hosting component of the overall Cisco BroadWorks service delivery platform as shown in the following figure.

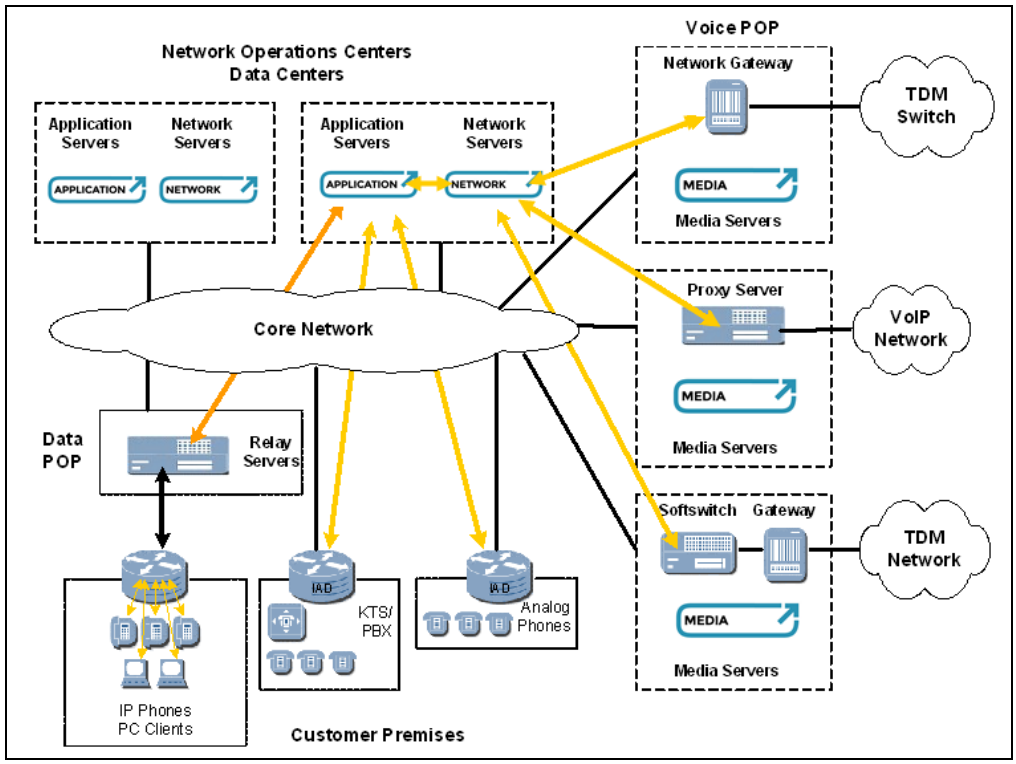

Figure 1 Cisco BroadWorks Service Delivery Platform

This document provides an overview of the Attendant Console application, which requires the Busy Lamp Field functionality. This application is part of Cisco BroadWorks, but it requires corresponding IP phone support before it can be delivered. The remainder of this document serves to outline these requirements on IP phones and how they must interoperate with Cisco BroadWorks to enable the Busy Lamp Field.

3 Application Overview

The primary application that drives the requirements for Busy Lamp Field functionality on Cisco BroadWorks is the Attendant Console. To understand Busy Lamp Field requirements, the Attendant Console application is summarized in the following sections. Note that the Busy Lamp Field feature is one of the building blocks of the Attendant Console application. The *Cisco BroadWorks Shared Call Appearance Interface Specification* [5] defines the Shared Call Appearance (SCA) features and requirements. The Busy Lamp Field and the Shared Call Appearance features contribute to building a complete Attendant Console application in a distributed network.

3.1 Attendant Console

The Attendant Console application allows a station in the network to monitor the call state of other stations in the network. Conventionally, an executive assistant or "front desk" operator uses this application. The operator is equipped with an enhanced station that offers enough line keys to adequately monitor a large set of lines in the network.

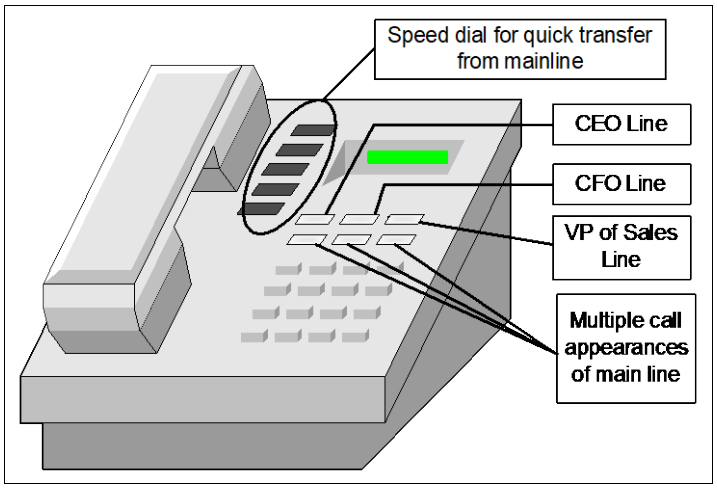

Figure 2 Attendant Console

An Attendant Console can have one or more lines that are configured to roll over from one to another. The purpose is to use these lines for all incoming calls that are managed by the administrator. Calls can be transferred from one of the main lines to the destination extension in the enterprise. Speed dial buttons can be configured to facilitate transfer interactions. When calls arrive for a user that has a line that is being "monitored" by the Attendant Console, the operator can easily determine if the user is busy (by looking at the lamp associated with the line key of that user) and make appropriate call routing decisions.

An Attendant Console can also host expansion modules with additional keys. Keys on an expansion module are typically configured for speed dial and allow the attendant to monitor a larger number of users.

In legacy systems, Attendant Consoles are typically attached to a CPE-based PBX that is connected to all the other stations in the enterprise. An Attendant Console cannot monitor stations that are not directly attached to the same PBX. Using the hosted features in BroadWorks, it is possible to not only offer the classic Attendant Console feature set, but also to offer them across geographically dispersed networks with heterogeneous access devices.

4 IP Phone

The following figure provides a hypothetical layout of an IP phone. It is not intended to represent a specific make or model, but rather a generic IP phone that incorporates the variety of features found on emerging IP phones in the industry. The definitions of the various keys and displays are given below and these terms are used throughout this document.

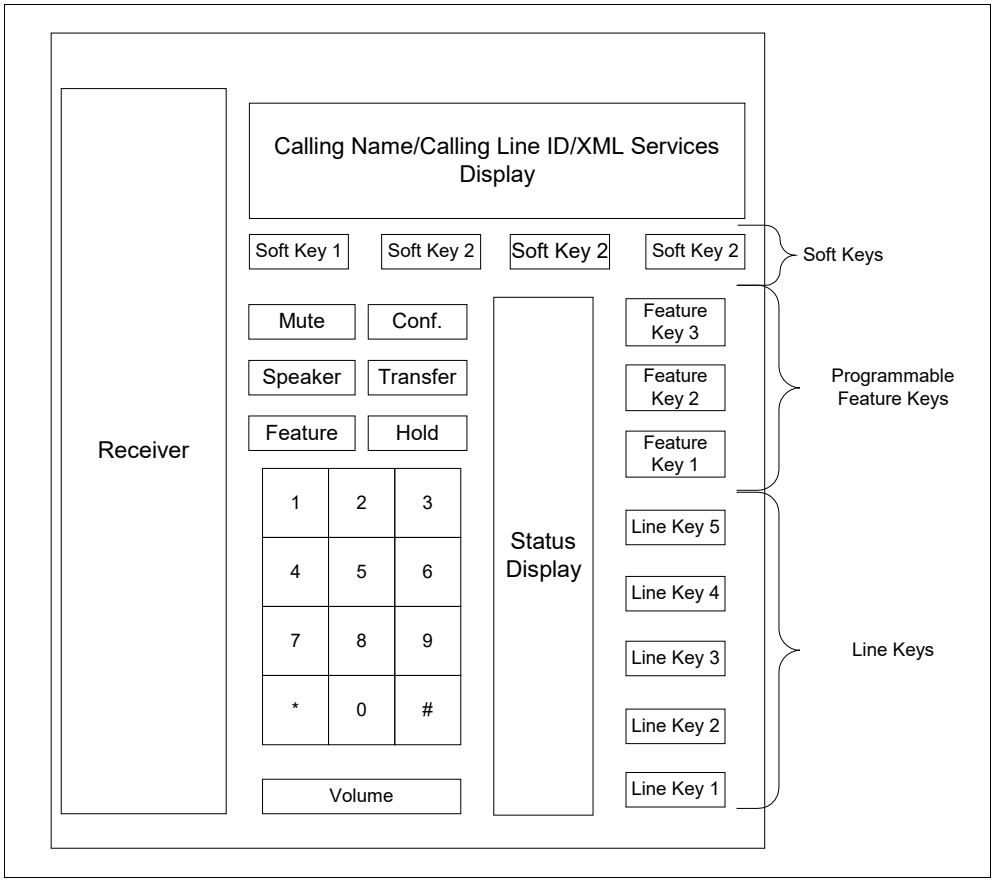

Figure 3 Layout of an IP Phone

Built-In Feature Keys: IP phones are smart endpoints with a variety of local features that can be invoked directly on the phone through “built-in feature keys”. The most common built-in features are “hold”, “transfer”, “mute” and “speaker” (or “hands-free”). These features are said to be “built-in” because the service function itself is implemented locally on the phone. Standard mechanisms for supporting these types of services are well defined in the SIP community and are not described in this document.

Programmable Feature Key: Some IP phones have "programmable feature keys" that allow the user to either invoke a service, or toggle it on and off. Because these keys are programmable, they have removable caps or labels that can be changed to reflect the associated service. Feature keys also have a "status indicator" that provides a visual queue of the state of the corresponding service. The status indicator is either a lamp or backlighting against the key itself, or a line on an LCD display adjacent to the key. Some IP phones allow feature keys to be programmed with services implemented locally on the phone or with remote services implemented in the network. Mechanisms for supporting programmable feature keys are outside of the scope of this document.

Soft Key: Almost all IP phones have some form of "soft key". Soft keys are like programmable feature keys, but instead of having labels or caps that must be physically replaced on the phone, the keys are unmarked, and a corresponding label is dynamically displayed on an adjacent LCD display based on the context of the user interaction with the phone. Soft keys are tightly integrated with an XML presentation language used to manage the contents of the LCD display. This document describes a mechanism for supporting soft keys and the corresponding XML presentation language.

Line: The concept of a "line" originates from circuit switched networks. Lines do not really exist in packet switched networks, but the concept is still useful when describing phones. For the purpose of this document, one address-of-record as defined in *RFC 3261* represents one line. When a user registers with a SIP network, the *To* header containing their address-of-record identifies their line. Lines can have multiple call appearances. Lines can be qualified as being either "private" or "shared".

Call Appearance: Every call is associated with a specific line. The presentation of a call on a line is a "call appearance".

Line Key: This is a bit of a misnomer, but it is commonly accepted that a line key is any key on an IP phone that is used to present and manage call appearances for a specific line or address-of-record. Some phones can have multiple line keys per line, making management of multiple call appearances intuitive, as each new call on the line has a separate line key associated with it. Some phones use only one line key per line and use soft keys to manage multiple call appearances. Line keys can also have a status indicator that provides a visual queue of the state of the call appearance on the given line. The status indicator can be a lamp, an LED, or backlighting against the key. Alternatively, an LCD display adjacent to the key is used.

Private Call Appearance: A private call appearance is any call appearance that is only visible and accessible through the original endpoints involved in setting up the call.

Private Line: A private line is a line that is only presented with private call appearances.

Shared Call Appearance: A shared call appearance is any call appearance that is visible and optionally accessible through the original endpoints in the call as well as an authorized set of other endpoints in the network. The mechanisms for supporting shared call appearances in SIP are described in the *Cisco BroadWorks Shared Call Appearance Interface Specification* [5].

Shared Line: A shared line is a line that is only presented with shared call appearances. The mechanisms for supporting shared call appearances in SIP are described in the *BroadWorks Shared Call Appearance Interface Specification* [5].

5 General Protocol Requirements

5.1 Assumptions

The mechanism described assumes that all users in a Busy Lamp Field list are hosted by an Application Server acting as a third-party call control element. "Hosted" means that:

- All user endpoints register their locations with the third-party controlling element (directly or indirectly).
- All outbound calls traverse the third-party controlling element immediately after leaving the user's endpoint.
- All inbound calls traverse the third-party controlling element before being delivered to the user's endpoint.

The general approach is focused on keeping the protocol elements and the semantics for using them simple. SIP endpoints tend to be constrained by memory and processing power – so the introduction of unnecessary signaling requirements should be avoided. Another primary objective is to keep the approach as standard as possible. No new methods, headers, or document types are introduced. The two primary protocol elements that are used are an extension (*RFC 4662 – Resource Lists*) and an instantiation (*RFC 4235 – Dialog event package*) of the SIP Specific Event Notification Framework (defined in *RFC 3265*).

5.2 RFC 4235 – Dialog Event Package

RFC 4235 defines the mechanism for an endpoint to subscribe to the state of any dialog involving another endpoint. *RFC 4235* is an instantiation of the SIP Specific Event Notification Framework (*RFC 3265*) and defines the dialog event package. It also defines the behavior of the device (subscriber) and the Application Server (notifier) in the context of subscriptions to the dialog event package.

This document provides specific information regarding the Application Server requirements (section 6) and device requirements (section 7) related to the transmission and handling of SUBSCRIBE and NOTIFY requests.

5.3 RFC 4662 – Resources Lists

RFC 4662 defines an aggregating mechanism that allows for subscribing and notifying for a list of resources. This is useful in the context of the Attendant Console application where the device needs to subscribe to a large number of resources. The overhead of generating individual SUBSCRIBE requests and the overhead of processing individual NOTIFY requests can be reduce by aggregating resources using a resource list concept.

This document provides specific information regarding the Application Server requirements (section *6 Application Server Requirements*) and device requirements (section *7 Device Requirements*) related to the transmission and handling of SUBSCRIBE and NOTIFY requests when using resource lists.

6 Application Server Requirements

6.1 Configuration

The Application Server configuration requirements are as follows:

The Application Server must allow an administrator to configure a resource list (Busy Lamp Field list) with a unique resource list URI and a list of users.

The Application Server must allow an administrator to override the transport for BLF notifications when TCP is supported at the SIP layer.

NOTE: The Application Server does not generate phone configuration information based on Busy Lamp Field service configured against the user. It is a requirement of the device to dynamically assign resources to soft keys based on the notifications sent by the Application Server at run time.

6.1.1 Redundancy

Support for full redundancy in the Busy Lamp Field is achieved by configuring the address of the redundant peer in the *bw.sip.peernetworkinterfacehost* system parameter on both servers involved in the redundancy functionality. Additionally, the following system-level parameters can be set to further customize the redundancy.

Name	Description
enableRedundancy	This parameter provides a configuration option to enable or disable redundancy support for BLF. If the parameter is set to “true”, then BLF redundancy is enabled on the Application Server. If the parameter is set to “false”, then BLF redundancy is disabled. The default value for the parameter is “false”. Replaces the former *bw.execution.blf.disableredundancy* execution container option.
redundancyTaskDelayMilliseconds	This parameter determines the period of delay the Application Server observes following a redundancy link “up” or “down” event before it starts processing to manage BLF subscriptions. The parameter takes a value with units of milliseconds and within the range from 10,000 through 180,000. The default value is “10,000”. Replaces the former *bw.execution.blf.taskDelay* execution container option.
maxNumberOfSubscriptionsPer-RedundancyTaskInterval	This parameter determines the maximum number of BLF subscriptions that the Application Server can process in one round of processing following a redundancy link state change event. The parameter takes a value within the range from 1 through 50,000. Thedefault value is “50”. Replaces the former *bw.execution.blf.maxNumberOfNotificationsPerSecondOnLinkDown* execution container option.

Name	Description
redundancyTaskIntervalMilliseconds	This parameter determines the time delay between processing rounds for BLF subscriptions for a redundancy link state change event (that is, “up” or “down”). The parameter takes a value with units of milliseconds and within the range from 50 through 20,000. The default value is “1,000”. Replaces the former *bw.execution.blf.taskWaitPeriod* execution container option.

6.2 Execution

The Application Server execution requirements are as follows:

- The Application Server must allow no more than 20 endpoints to SUBSCRIBE to the “Dialog” event package for a particular resource list. In practice, a single endpoint (Attendant Console) will SUBSCRIBE to the “Dialog” event package for a particular resource list. The resource list URI is associated with a single user account on BroadWorks.
  - The Application Server must send a *421 Extension Required* response if the SUBSCRIBE does not include a Supported header with the *eventlist* option tag.
  - The Application Server must send a *404 Not Found* response if the resource list URI is not associated with the endpoint’s Busy Lamp Field service profile.
  - The Application Server must send a *403 Forbidden* response if maximum number of subscriptions for a resource list is exceeded.
- If the Application Server is configured to challenge REGISTER requests from endpoints, then it must also challenge SUBSCRIBE requests from endpoints.
- If the Application Server is configured to perform access control on INVITE requests from endpoints, then it must also perform access control on SUBSCRIBE requests from endpoints.
- After granting a subscription to the “Dialog” event package for a resource list, the Application Server must initialize the IP phone to the current state of all resources by sending a NOTIFY request with corresponding state information for all resources identified by the resource list. The state information is included in a multipart body that identifies all subscribed resources and that provides the current state for each resource.
- When a subscribed resource transitions to the progressing, confirmed, or terminated state, the Application Server must send a NOTIFY request to all endpoints that have subscribed to the “Dialog” event package for this resource. The NOTIFY request MUST contain a message body indicating the state of that resource.
- When the device refreshes a subscription by sending a SUBSCRIBE request, the Application Server must send a NOTIFY request with a message body that contains the identity and state of all subscribed resources.
- When the system parameter *forceUseOfTCP* is enabled, the Application Server will force the transport to be TCP for BLF notifications, even if the initial subscription’s dialog is explicitly, implicitly, or by DNS resolution indicated to send the SIP NOTIFY using UDP. This is true if the SIP configuration supports the TCP transport. The *forceUseOfTCP* flag has no effect if the subscription originally indicated TCP transport.

6.2.1 Startup

When the Application Server starts, it issues a NOTIFY request indicating that the subscription is terminated (Subscription-State header is set to “terminated”) for the BLF subscriptions belonging to a user hosted on the server starting up.

If a user was migrated to the collocated server, then no NOTIFY requests are sent by the server starting up (for more information, see section *6.2.2 Redundancy)*.

The notification is interpreted by the device as a trigger to re-SUBSCRIBE and realign the resource list and dialog version sequence numbers.

6.2.2 Redundancy

Starting in Release 19.0, comprehensive redundancy support was added to the Busy Lamp Field feature. Prior to this release, when a monitored user was not hosted on the same server as the monitoring user, NOTIFY requests were sent to the subscribing device from both the primary and the secondary servers, leading to the following potential issues:

- Cseq header value out of sequence
- Resource list and dialog version out of sequence
- Non-existent support for fallback to peer when hosting Application Server (AS) dies

Redundancy behavior is as follows:

- Busy Lamp Field NOTIFY requests are always sent from the Application Server where the original SUBSCRIBE message has been handled:
  - NOTIFY requests are always sent with increasing CSeq parameters, no matter if the monitored user at the origin of the notification is hosted on the same server as the monitoring user.
  - NOTIFY requests are always sent with increasing resource list and/or dialog versions.
  - Exception: If the server hosting the monitoring user dies, the peer will send a NOTIFY request indicating that the subscription is terminated (“Subscription-state” header is set to “terminated”). This triggers a new subscription to be established by the device towards the available server.
- When one of the Application Server dies, its peer shall terminate the Busy Lamp Field subscription for any monitoring user it does not host.
  - These terminations are throttled to a maximum of 50 terminations per second.
- When one of the Application Server dies, its peer shall send a notification to indicate that monitored users it does not host now have a dialog state value **terminated** (when the prior dialog state was not terminated).
- When the user owning a BLF list is migrated to the peer, the peer shall send a NOTIFY request with state terminated. The device shall establish a new subscription on the peer to realign the resource list and dialog versions.

Note that redundancy applies for a variety of Busy Lamp Field notifications, including Call Park notifications.

7 Device Requirements

7.1 Configuration

The device configuration requirements are as follows:

- The device must securely retrieve configuration information from a server in the hosted provider network at startup time.
- The device configuration information must contain a resource list URI. The resource list can be associated with a specific set of keys on the IP phone.
- The device should be configured to use TCP as the transport protocol for SIP signaling to accommodate for the larger packet sizes that are implied by the aggregation of resources for the Busy Lamp Field feature.

NOTE: The device actually populates the keys upon receiving the initial NOTIFY request when the sends a SUBSCRIBE to the resource list URI.

7.2 Execution

The device execution requirements are as follows:

- The device must send SUBSCRIBE request to the configured resource list URI as per *RFC 3265*. The subscription is for "Dialog" event package (*RFC 4235*) that provides state information associated with all resources associated with the resource list URI. The SUBSCRIBE request must include the Supported header with the "eventlist" option tag (*per RFC 4662*).
- The device must continually refresh the "Dialog" subscriptions before they expire as per *RFC 3265*.
- The device must be prepared to provide corresponding credentials for authentication challenges for any SUBSCRIBE transactions.
- When a NOTIFY arrives, the phone must perform access control and only accept the request if the source IP address corresponds to one of the IP addresses of the Application Server.
- After a NOTIFY request with a message body has been accepted, the device should modify the state and display of the associated resource(s) based on the body of the NOTIFY request. The NOTIFY request contains a multipart body.
  - The first body part has a content-type of application/rlmi+xml and uniquely identifies the subscribed resource(s). The identification contains a resource URI and a display name.
  - The second body part has a content-type of application/dialog-info+xml and contains the identity and the current state for each dialog associated with a subscribed resource(s).
    - The identity includes their display name, DN, extension, and SIP-URI. The extension for a particular resource in the list is provided in accordance with *RFC 3966* as part of the tel URI. An example is "tel:+12403645555;ext=5555". If a user does not have an extension, then the *ext* parameter is not present. If a user does not have a DN, then the tel URI is not included.

  - If the dialog state is not “terminated”, then the identity of the remote party can also be included in the message body.
  - The dialog identifier can include the direction attribute. The possible direction values are “initiator” and “recipient”. The direction attribute of the dialog identifier never changes for a given dialog. The direction attribute is not included in an initial NOTIFY request if the dialog state is “terminated”.

- The initial NOTIFY request contains the identity and state of all resources associated with the resource list URI. The device should use the identity of the resources to auto-populate the keys associated with the resource list. The information included for an individual subscribed resource allows the device to program an available soft key such that it can be used for monitoring and speed dial purposes.
- If the NOTIFY request arrives with a “Subscription-State” value of “terminated”, the device must attempt to re-subscribe by sending a SUBSCRIBE request. The device must honor the “retry-after” header if present. The device should handle the subsequent NOTIFY request as an initial NOTIFY request and re-populate the soft keys accordingly.
- The device should use the following guidelines for managing the state of each subscribed resource:
  - Terminated

    When there is no active call for a specific resource or when a call to or from a subscribed resource is just being released, then the resource is considered idle/terminated.

    If applicable, the LED for the resource should be OFF.
  - Proceeding

    When a new call arrives for a subscribed resource or if a call is originated from a subscribed resource, the resource is considered alerting/proceeding.

    If applicable, the LED for the call resource should be blinking fast green.
  - Confirmed

    When a call to or from a subscribed resource is answered, then the resource is said to be in the “active” state.

    When there are one or more active calls for a subscribed resource, then the resource is also considered active.

    IP phones should indicate that the line is active visually. If applicable, the LED of the resource on the endpoint should be solid red.

8 Functional Overview

This section presents a functional overview of how the dialog event package (*RFC 3245*) and resource lists (*RFC 4662*) are used to deliver Busy Lamp Field functionality in a hosted communications environment. The intent is to show an application neutral set of call flows that cover the most common interactions that can be expected between the SIP IP phones and the Application Server. For simplicity, the Busy Lamp Field is limited to two monitored users, but the basic principles can be directly extrapolated to any number of users.

The following figure shows the network configuration for the examples that follow. IP phone A-1 (a1.foo.com) and IP phone A-2 (a2.foo.com) each have a private line hosted on the Application Server (as.foo.com), respectively: private-sip:a1@a.foo.com and sip:private-a2@a.foo.com.

Phone B is behind a SIP network gateway and represents the off-network party in the call flows.

Phone AC is the Attendant Console that is monitoring the private phone lines.

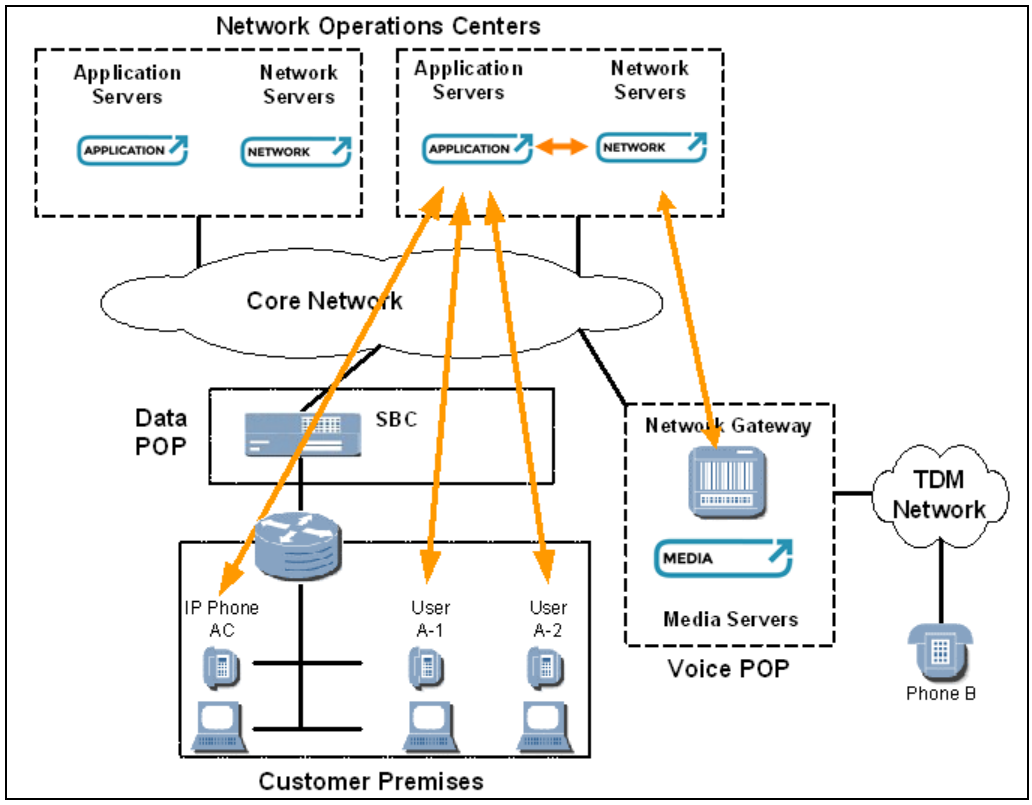

Figure 4 Functional Overview

8.1 IP Phone Power Up

The following figure shows the message flow as a result of powering up IP phone AC.

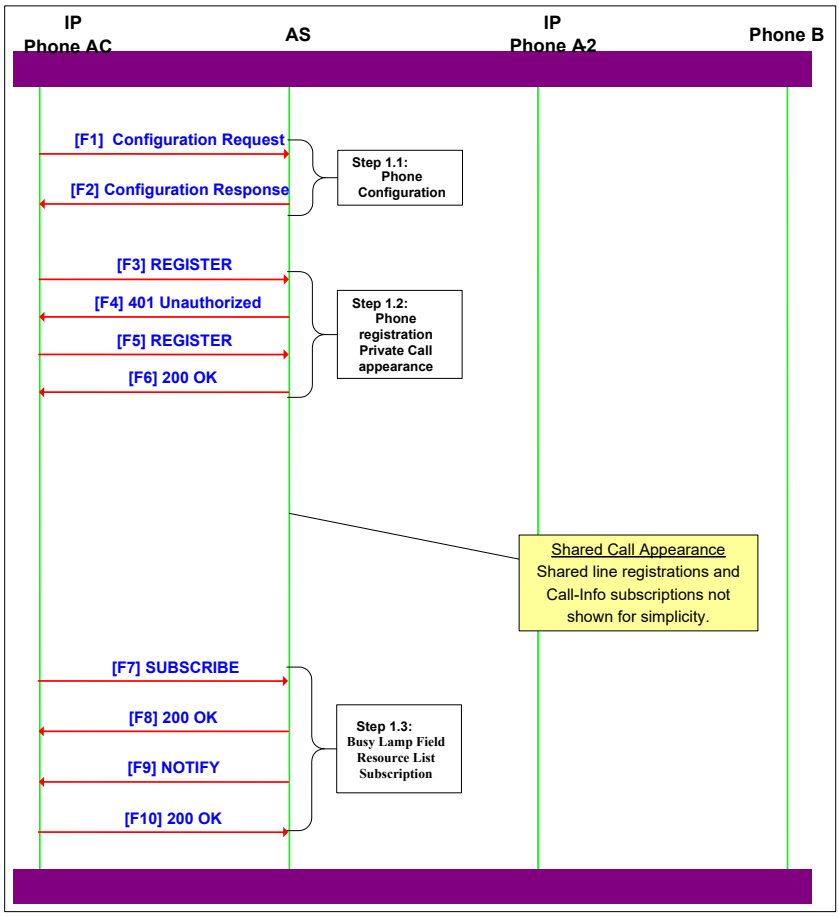

Figure 5 Phone AC Powers Up

8.1.1 Step 1.1: AC Phone Configuration

The phones must first retrieve network topology and device configuration information. This is done through a "Configuration Request" in [F1]. The transport for the configuration request can be any number of options: FTP, TFTP, HTTP, or HTTPS. HTTPS is preferred as it offers confidential transport of potentially sensitive configuration information. The flow shows the Application Server (AS) as the target of the configuration request. In practice, this can be any centralized server in the network, and does not have to be the Application Server. The "Configuration Response" in [F2] provides any number of device specific configuration parameters, as well as the following critical information:

- The Busy Lamp Field resource list URI and potentially a set of keys that can be associated with the resources
- Which line keys are shared call appearances
- Which line keys are local/private call appearances
- The display name for each line
- The address-of-record for each line
- The credentials for each line

- The registrar for each line
- The proxy for each line

The details related to shared call appearances are not provided in this document. For more information, see the *Cisco BroadWorks Shared Call Appearance Interface Specification* [5].

8.1.2 Step 1.2: AC Phone Registration for Local Private Lines

[F3] to [F6] is performed for each unique private line. This is the standard registration process as described in *RFC 3261*. SIP MD-5 digest authentication is used to authenticate the endpoint.

```
[F3] REGISTER AC -> Application Server

REGISTER sip:as.foo.com SIP/2.0
Via: SIP/2.0/UDP ac.foo.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=a73kszlfl
To: "Private-AC" <sip:private-ac@as.foo.com>
Call-ID: 1j9FpLxk3uxtm8tn@a.foo.com
CSeq: 1 REGISTER
Contact: <sip:private-ac@a.foo.com>
Content-Length: 0

[F4] 401 Unauthorized Application Server → AC

SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP a1.foo.com:5060;branch=z9hG4bKnashds7
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=a73kszlfl
To: "Private-AC" <sip:private-ac@as.foo.com>;tag=dgt2883jjkd
Call-ID: 1j9FpLxk3uxtm8tn@a.foo.com
CSeq: 1 REGISTER
WWW-Authenticate: Digest realm="foo.com", qop="auth",
nonce="ea9c8e88df84f1cec4341ae6cbe5a359", opaque="", stale=FALSE,
algorithm=MD5
Content-Length: 0

[F5] REGISTER AC -> Application Server

REGISTER sip:as.foo.com SIP/2.0
Via: SIP/2.0/UDP a1.foo.com:5060;branch=asdf32nashds7
Max-Forwards: 70
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=a73kszlfl
To: "Private-AC" <sip:private-ac@as.foo.com>;tag=dgt2883jjkd
Call-ID: 1j9FpLxk3uxtm8tn@a.foo.com
CSeq: 2 REGISTER
Contact: <sip:private-ac@a.foo.com>
Authorization: Digest username="private-a1", realm="foo.com"
nonce="ea9c8e88df84f1cec4341ae6cbe5a359", opaque="",
uri="sip:as.foo.com", response="dfe56131d1958046689d83306477ecc"
Content-Length: 0

[F6] 200 OK Application Server → AC

SIP/2.0 200 OK
Via: SIP/2.0/UDP a1.foo.com:5060;branch=asdf32nashds7
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=a73kszlfl
To: "Private-AC" <sip:private-ac@as.foo.com>;tag=dgt2883jjkd
Call-ID: 1j9FpLxk3uxtm8tn@a.foo.com
```

```
CSeq: 2 REGISTER
Contact: <sip:private-ac@a.foo.com>;expires=3600
Content-Length: 0
```

8.1.3 Step 1.3: AC Busy Lamp Field Resource List Subscription

[F7] to [F10] shows how the AC subscribes for the dialog event package and for a particular resource list. This process allows the endpoint to SUBSCRIBE to the call state of multiple users. The Request-URI for these transactions is the resource list URI as configured on the Application Server.

Note that the call flow does not show the SUBSCRIBE being challenged. The Application Server may choose to use access control (using the IP address authenticated in the REGISTER transaction) or optionally it could challenge the transaction. If the transaction is challenged, then the IP phone should use the same credentials it used in the corresponding registration process for that call appearance. Throughout the rest of this example, it is assumed that standard access control is being applied to all transactions.

After accepting the SUBSCRIBE, the Application Server initializes the phone by sending a NOTIFY with a message body that describes all monitored users referred by the resource list and that provides the call state for each monitored user. This NOTIFY serves as a synchronization event between the Application Server and the IP phone.

The IP phone typically uses this information to initialize the soft keys associated with the Busy Lamp Field resource list. Each key is associated with one user described in the NOTIFY event. The IP phone uses the user information to automatically program an available soft key. The display is updated to reflect the identity of the monitored user and a speed dial number is associated with the key. In addition, the status indicator of the key is updated to reflect the current call state received in the NOTIFY event. In the case of a non-terminated dialog state, the remote party information is included and may be displayed against the soft key on the device.

[F7] SUBSCRIBE AC -> Application Server

```
SUBSCRIBE sip:blf-resource-list@as.foo.com SIP/2.0
Via: SIP/2.0/UDP a1.foo.com:5060;branch=dsdf32nashds7
Max-Forwards: 70
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
To: <sip:blf-resource-list@as.foo.com>
Call-ID: 5j9FpLxk3uxtm8to@a.foo.com
CSeq: 6 SUBSCRIBE
Event: dialog
Supported: eventlist
Expires: 3600
Contact: <sip:private-ac@a.foo.com>
Accept: application/dialog-info+xml
Accept: application/rlmi+xml
Accept: multipart/related
Content-Length: 0
```

[F8] 200 OK Application Server → AC

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP a1.foo.com:5060;branch=dsdf32nashds7
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
To: <sip:blf-resource-list@as.foo.com>;tag=323423233
Call-ID: 5j9FpLxk3uxtm8tn@a.foo.com
CSeq: 6 SUBSCRIBE
Event:  dialog
```

```
Contact: <sip:as.foo.com>
Expires: 3600
Content-Length: 0

[F9] NOTIFY Application Server -> AC

NOTIFY sip: private-ac@a.foo.com SIP/2.0
Via: SIP/2.0/UDP as.foo.com:5060;branch=gsdf32nashds7
Max-Forwards: 70
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
To: <sip:blf-resource-list@as.foo.com>;tag=323423233
Call-ID: 5j9FpLxk3uxtm8tn@a.foo.com
CSeq: 1344 NOTIFY
Event: dialog
Require: eventlist
Subscription-State: active; expires=3599
Contact:  <sip:as.foo.com>
Content-Length: 1666
Content-Type:multipart/related;boundary=UniqueBroadWorksBoundary

--UniqueBroadWorksBoundary
Content-Type:application/rlmi+xml
Content-Length:429
Content-ID:<sczbLo@as.foo.com>

<?xml version="1.0" encoding="UTF-8"?>
<list xmlns="urn:ietf:params:xml:ns:rlmi"
      uri="sip:blf-resource-list@as.foo.com" version="0"
      fullState="true">
  <resource uri="sip:user-a1@as.foo.com">
    <name>User A-1</name>
    <instance id="FQJN17kNZT" state="active" cid="vZIKOJ@as.foo.com"/>
  </resource>
  <resource uri="sip:user-a2@as.foo.com">
    <name>User A-2</name>
    <instance id="lFUjuZjfxM" state="active" cid="5U3W4o@as.foo.com"/>
  </resource>
</list>

--UniqueBroadWorksBoundary
Content-Type:application/dialog-info+xml
Content-Length:366
Content-ID:<vZIKOJ@as.foo.com>

<?xml version="1.0" encoding="UTF-8"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
             version="0"
             state="full"
             entity="user-a1@as.foo.com">
  <dialog id="aHRFSG9R">
    <state>terminated</state>
    <local>
      <identity display="User A-1">sip:user-a1@as.foo.com</identity>
      <identity display="User A-1">tel:+12403645132;ext=5132</identity>
    </local>
  </dialog>
</dialog-info>

--UniqueBroadWorksBoundary
Content-Type:application/dialog-info+xml
Content-Length:476
Content-ID:<5U3W4o@broadworks>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
             version="0"
             state="full"
             entity="user-a2@as.foo.com">
  <dialog id="bG9jYWxIb3N0NTow" direction="recipient">
    <state>confirmed</state>
    <local>
      <identity display="User A-2">sip:user-a2@as.foo.com</identity>
      <identity display="User A-2">tel:+12403645131;ext=5131</identity>
    </local>
    <remote>
      <identity display="User A-3">sip:5130@as.foo.com;user=phone
      </identity>
    </remote>
  </dialog>
</dialog-info>

--UniqueBroadWorksBoundary--
```

[F10] 200 OK AC → Application Server

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP as.foo.com:5060;branch=gsdf32nashds7
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
To: <sip:blf-resource-list@as.foo.com>;tag=323423233
Call-ID: 5j9FpLxk3uxtm8tn@a.foo.com
CSeq: 1344 NOTIFY
Content-Length: 0
```

8.2 Dialog State Transitions

The following figure shows the call flow as a result of A-1 originating a call to B.

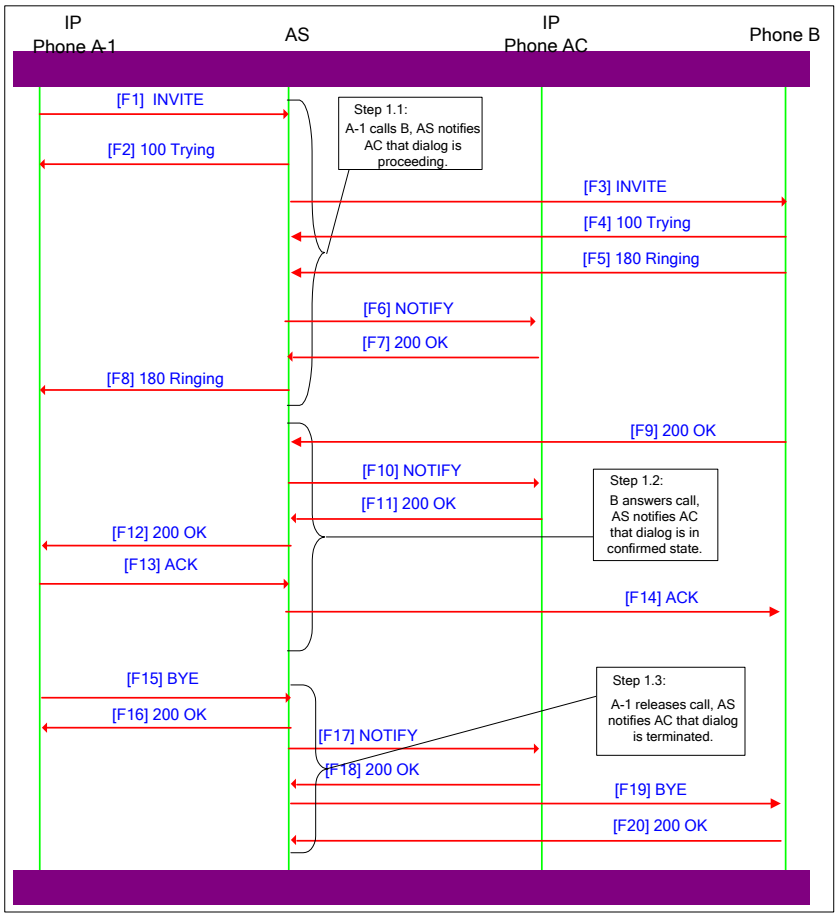

Figure 6 A-1 Calls B

8.2.1 Step 2.1: A-1 Calls B

In [F1 to F8], the user on IP phone A-1 has entered digits that originates a call to B. The Application Server processes the INVITE and simple treats the call like a standard outbound call to an off-network device. After creating the session for the outbound call, the Application Server sends a NOTIFY request to the Attendant Console (without waiting for a response to the INVITE sent to B).

The NOTIFY request has a message body that describes the state change (that is, "proceeding") for the monitored resource (that is, user A-1). The Attendant Console processes the NOTIFY request and updates the state and/or display of the soft key associated with the resource.

[F1] INVITE A-1 → Application Server

```
INVITE sip:2405555130@as.foo.com SIP/2.0
Via: SIP/2.0/UDP a.foo.com:5060;branch=m9hG4bK74bf9
Max-Forwards: 70
From: "Private-A1" <sip:private-a1@as.foo.com>;tag=dsa3dszlfl
To: "2405555130" <sip:2405555130@as.foo.com>
Call-ID: 98482762982201885l1@a.foo.com
Call-Info:  <sip:as.foo.com>;appearance-index=1
CSeq: 43234 INVITE
Contact: <sip:private-a1@a.foo.com>
Content-Type: application/sdp
Content-Length: 143

v=0
o=UserA 2890844526 2890844526 IN IP4 a.foo.com
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

[F2] 100 Trying Application Server → A-1

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP a1.foo.com:5060;branch=m9hG4bK74bf9
From: "Private-A1" <sip:private-a1@as.foo.com>;tag=dsa3dszlfl
To: "2405555130" <sip:2405555130@foo.com>
Call-ID: 98482762982201885l1@a.foo.com
Call-Info:  <sip:as.foo.com>;appearance-index=1
CSeq: 43234 INVITE
Contact: <sip:@as.foo.com>
Content-Length: 0
```

[F3] INVITE Application Server → B

```
INVITE sip:B@b.foo.com SIP/2.0
Via: SIP/2.0/UDP as.foo.com:5060;branch=234adfrjksd
Max-Forwards: 70
From: "2403645132" <sip:2403645132@as.foo.com>;tag=234wdkfj
To: "B" <sip:B@as.foo.com>
Call-ID: asdf2220188511@as.foo.com
CSeq: 12345 INVITE
Contact: <sip:as.foo.com>
Content-Type: application/sdp
Content-Length: 143

v=0
```

```
o=BroadWorks 345679504 0 IN IP4 as.foo.com
s=-
c=IN IP4 192.0.2.101
t=0 0
m=audio 49172 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

[F4] 100 Trying B → Application Server

```
SIP/2.0 100 Trying
Via: SIP/2.0/UDP as.foo.com:5060;branch=234adfrjksd
From: "2403645132" <sip: 2403645132@as.foo.com>;tag=234wdkfj
To: "B" <sip:B@as.foo.com>
Call-ID: asdf2220188511@as.foo.com
CSeq: 12345 INVITE
Contact: <sip:B@b.foo.com>
Content-Length: 0
```

[F5] NOTIFY Application Server -> AC

```
NOTIFY sip: private-ac@a1.foo.com SIP/2.0
Via: SIP/2.0/UDP as.foo.com:5060;branch=gsdf32nashds7
Max-Forwards: 70
From: <sip:blf-resource-list@as.foo.com>;tag=323423233
To: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
Call-ID: 5j9FpLxk3uxtm8tn@a.foo.com
CSeq: 1345 NOTIFY
Event: dialog
Require: eventlist
Subscription-State: active; expires=3599
Contact:  <sip:as.foo.com>
Content-Length: 1666
Content-Type:multipart/related;boundary=UniqueBroadWorksBoundary

--UniqueBroadWorksBoundary
Content-Type:application/rlmi+xml
Content-Length:429
Content-ID:<sczbLo@as.foo.com>

<?xml version="1.0" encoding="UTF-8"?>
<list xmlns="urn:ietf:params:xml:ns:rlmi"
      uri="sip:blf-resource-list@as.foo.com" version="0"
      fullState="true">
  <resource uri="sip:user-a1@as.foo.com">
    <name>User A-1</name>
    <instance id="FQJN17kNZT" state="active" cid="vZIKOJ@as.foo.com"/>
  </resource>
</list>

--UniqueBroadWorksBoundary
Content-Type:application/dialog-info+xml
Content-Length:366
Content-ID:<vZIKOJ@as.foo.com>

<?xml version="1.0" encoding="UTF-8"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
             version="0"
             state="full"
             entity="user-a1@as.foo.com">
  <dialog id="aHRFSG9R" direction="initiator">
    <state>proceeding</state>
```

```
    <local>
      <identity display="User A-1">sip:user-a1@as.foo.com</identity>
      <identity display="User A-1">tel:+12403645132;ext=5132</identity>
    </local>
    <remote>
      <identity>sip:2405555130@as.foo.com;user=phone
      </identity>
    </remote>
  </dialog>
</dialog-info>

--UniqueBroadWorksBoundary--
```

[F6] 200 OK AC → Application Server

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP as.foo.com:5060;branch=gsdf32nashds7
From: <sip:blf-resource-list@as.foo.com>;tag=323423233
To: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
Call-ID: 5j9FpLxk3uxtm8tn@a.foo.com
CSeq: 1345 NOTIFY
Content-Length: 0
```

[F7] 180 Ringing B → Application Server

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP as.foo.com:5060;branch=234adfrjksd
From: "2403645132" <sip: 2403645132@as.foo.com>;tag=234wdkfj
To: "B" <sip:B@as.foo.com>;tag=1234dsf2
Call-ID: asdf2220188511@as.foo.com
CSeq: 12345 INVITE
Contact: <sip:B@b.foo.com>
Content-Length: 0
```

[F8] 180 Ringing Application Server → AC

```
SIP/2.0 180 Ringing
Via: SIP/2.0/UDP a1.foo.com:5060;branch=m9hG4bK74bf9
From: "Private-A1" <sip:private-a1@as.foo.com>;tag=dsa3dszlfl
To: "2405555130" <sip:2405555130@as.foo.com>;tag=3564hhhjd742
Call-ID: 98482762982201885l1@a.foo.com
Call-Info:  <sip:as.foo.com>;appearance-index=1
CSeq: 43234 INVITE
Contact: <sip: as.foo.com>
Content-Length: 0
```

8.2.2 Step 2.2: B Answers Call

[F9 to F14] shows B answering the call using standard back-to-back user agent call flow. A-1 and B are now active in a call. Upon receiving the 2xx response from B, the Application Server sends a NOTIFY request to the Attendant Console.

The NOTIFY request has a message body that describes the state change (that is, "confirmed") for the monitored resource (that is, user A-1). The Attendant Console processes the NOTIFY request and updates the state and/or display of the soft key associated with the resource.

[F9] 200 OK B → Application Server

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP as.foo.com:5060;branch=234adfrjksd
From: "2403645132" <sip: 2403645132@as.foo.com>;tag=234wdkfj
To: "B" <sip:B@as.foo.com>;tag=1234dsf2
Call-ID: asdf2220188511@as.foo.com
CSeq: 12345 INVITE
Contact: <sip:B@b.foo.com>
Content-Length: 143

v=0
o=UserB 2890844526 2890844526 IN IP4 b.foo.com
s=-
c=IN IP4 192.0.4.101
t=0 0
m=audio 49174 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

[F10] NOTIFY Application Server -> AC

```
NOTIFY sip: private-ac@a1.foo.com SIP/2.0
Via: SIP/2.0/UDP as.foo.com:5060;branch=gsdf32nashds7
Max-Forwards: 70
From: <sip:blf-resource-list@as.foo.com>;tag=323423233
To: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
Call-ID: 5j9FpLxk3uxtm8tn@a.foo.com
CSeq: 1346 NOTIFY
Event: dialog
Require: eventlist
Subscription-State: active; expires=3599
Contact:  <sip:as.foo.com>
Content-Length: 1666
Content-Type:multipart/related;boundary=UniqueBroadWorksBoundary

--UniqueBroadWorksBoundary
Content-Type:application/rlmi+xml
Content-Length:429
Content-ID:<sczbLo@as.foo.com>

<?xml version="1.0" encoding="UTF-8"?>
<list xmlns="urn:ietf:params:xml:ns:rlmi"
      uri="sip:blf-resource-list@as.foo.com" version="0"
      fullState="true">
  <resource uri="sip:user-a1@as.foo.com">
    <name>User A-1</name>
    <instance id="FQJN17kNZT" state="active" cid="vZIKOJ@as.foo.com"/>
  </resource>
</list>
```

```
--UniqueBroadWorksBoundary
Content-Type:application/dialog-info+xml
Content-Length:366
Content-ID:<vZIKOJ@as.foo.com>

<?xml version="1.0" encoding="UTF-8"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
             version="0"
             state="full"
             entity="user-a1@as.foo.com">
  <dialog id="aHRFSG9R" direction="initiator">
    <state>confirmed</state>
    <local>
      <identity display="User A-1">sip:user-a1@as.foo.com</identity>
      <identity display="User A-1">tel:+12403645132;ext=5132</identity>
    </local>
    <remote>
      <identity>sip:2405555130@as.foo.com;user=phone
      </identity>
    </remote>
  </dialog>
</dialog-info>

--UniqueBroadWorksBoundary--
```

[F11] 200 OK AC → Application Server

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP as.foo.com:5060;branch=gsdf32nashds7
From: <sip:blf-resource-list@as.foo.com>;tag=323423233
To: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
Call-ID: 5j9FpLxk3uxtm8tn@ac.foo.com
CSeq: 1346 NOTIFY
Content-Length: 0
```

[F12] 200 OK Application Server → A-1

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP a1.foo.com:5060;branch=m9hG4bK74bf9
From: "Private-A1" <sip:private-a1@as.foo.com>;tag=dsa3dszlfl
To: "2405555130" <sip:2405555130@as.foo.com>;tag=3564hhhjd742
Call-ID: 9848276298220188511@a.foo.com
Call-Info:  <sip:as.foo.com>;appearance-index=1
CSeq: 43234 INVITE
Contact: <sip: as.foo.com>
Content-Length: 143

v=0
o=BroadWorks 345679505 0 IN IP4 as.foo.com
s=-
c=IN IP4 192.0.4.101
t=0 0
m=audio 49174 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

[F13] ACK A-1 → Application Server

```
ACK sip:2405555130@as.foo.com SIP/2.0
Via: SIP/2.0/UDP a.foo.com:5060;branch=m9hG4bK74bf9
Max-Forwards: 70
From: "Private-A1" <sip:private-a1@as.foo.com>;tag=dsa3dszlfl
To: "2405555130" <sip:2405555130@as.foo.com>;tag=3564hhhjd742
```

```
Call-ID: 9848276298220188511@a.foo.com
CSeq: 43234 ACK
Contact: <sip:private-a1@a.foo.com>
Content-Length: 0

[F14] ACK Application Server → B

ACK sip:B@b.foo.com SIP/2.0
Via: SIP/2.0/UDP as.foo.com:5060;branch=23dsdf2ksd
Max-Forwards: 70
From: "2403645132" <sip: 2403645132@as.foo.com>;tag=234wdkfj
To: "B" <sip:B@as.foo.com>;tag=1234dsf2
Call-ID: asdf2220188511@a.foo.com
CSeq: 12345 ACK
Contact: <sip:as.foo.com>
Content-Length: 0
```

8.2.3 Step 2.3: A-1 Releases Call

In [F15 to F20] shows A-1 releasing the call using standard back-to-back user agent call flow. The dialog is not terminated. Upon receiving the BYE request from A-1, the Application Server sends a NOTIFY request to the Attendant Console.

The NOTIFY request has a message body that describes the state change (that is, "terminated") for the monitored resource (that is, user A-1). The Attendant Console processes the NOTIFY request and updates the state and/or display of the soft key associated with the resource.

```
[F15] BYE A-1 → Application Server

BYE sip:2405555130@as.foo.com SIP/2.0
Via: SIP/2.0/UDP a.foo.com:5060;branch=m9hG4bK74bf9
Max-Forwards: 70
From: "Private-A1" <sip:private-a1@as.foo.com>;tag=dsa3dszlfl
To: "2405555130" <sip:2405555130@as.foo.com>;tag=3564hhhjd742
Call-ID: 9848276298220188511@a.foo.com
CSeq: 43235 BYE
Contact: <sip:private-a1@a.foo.com>
Content-Length: 0

[F16] 200 OK Application Server → A-1

SIP/2.0 200 OK
Via: SIP/2.0/UDP a1.foo.com:5060;branch=m9hG4bK74bf9
From: "Private-A1" <sip:private-a1@as.foo.com>;tag=dsa3dszlfl
To: "2405555130" <sip:2405555130@as.foo.com>;tag=3564hhhjd742
Call-ID: 9848276298220188511@a.foo.com
CSeq: 43235 BYE
Contact: <sip:as.foo.com>
Content-Length: 0

[F17] NOTIFY Application Server -> AC

NOTIFY sip: private-ac@a1.foo.com SIP/2.0
Via: SIP/2.0/UDP as.foo.com:5060;branch=gsdf32nashds7
Max-Forwards: 70
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
To: <sip:blf-resource-list@as.foo.com>;tag=323423233
Call-ID: 5j9FpLxk3uxtm8tn@ac.foo.com
```

```
CSeq: 1347 NOTIFY
Event: dialog
Require: eventlist
Subscription-State: active; expires=3599
Contact:  <sip:as.foo.com>
Content-Length: 1666
Content-Type:multipart/related;boundary=UniqueBroadWorksBoundary

--UniqueBroadWorksBoundary
Content-Type:application/rlmi+xml
Content-Length:429
Content-ID:<sczbLo@as.foo.com>

<?xml version="1.0" encoding="UTF-8"?>
<list xmlns="urn:ietf:params:xml:ns:rlmi"
      uri="sip:blf-resource-list@as.foo.com" version="0"
      fullState="true">
  <resource uri="sip:user-a1@as.foo.com">
    <name>User A-1</name>
    <instance id="FQJN17kNZT" state="active" cid="vZIKOJ@as.foo.com"/>
  </resource>
</list>

--UniqueBroadWorksBoundary
Content-Type:application/dialog-info+xml
Content-Length:366
Content-ID:<vZIKOJ@as.foo.com>

<?xml version="1.0" encoding="UTF-8"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
             version="0"
             state="full"
             entity="user-a1@as.foo.com">
  <dialog id="aHRFSG9R" direction="initiator">
    <state>terminated</state>
    <local>
      <identity display="User A-1">sip:user-a1@as.foo.com</identity>
      <identity display="User A-1">tel:+12403645132;ext=5132</identity>
    </local>
  </dialog>
</dialog-info>

--UniqueBroadWorksBoundary--
```

[F18] 200 OK AC → Application Server

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP as.foo.com:5060;branch=gsdf32nashds7
From: "Private-AC" <sip:private-ac@as.foo.com>;tag=dsa3dszlfl
To: <sip:blf-resource-list@as.foo.com>;tag=323423233
Call-ID: 5j9FpLxk3uxtm8tn@ac.foo.com
CSeq: 1347 NOTIFY
Content-Length: 0
```

[F19] BYE Application Server → B

```
BYE sip:B@b.foo.com SIP/2.0
Via: SIP/2.0/UDP as.foo.com:5060;branch=23dsdf2ksd
Max-Forwards: 70
From: "2403645132" <sip: 2403645132@as.foo.com>;tag=234wdkfj
To: "B" <sip:B@as.foo.com>;tag=1234dsf2
Call-ID: asdf2220188511@a.foo.com
```

```
CSeq: 12346 BYE
Contact: <sip:as.foo.com>
Content-Length: 0

[F20] 200 OK B → Application Server

SIP/2.0 200 OK
Via: SIP/2.0/UDP as.foo.com:5060;branch=234adfrjksd
From: "2403645132" <sip: 2403645132@as.foo.com>;tag=234wdkfj
To: "B" <sip:B@as.foo.com>;tag=1234dsf2
Call-ID: asdf2220188511@a.foo.com
CSeq: 12346 BYE
Contact: <sip:B@b.foo.com>
Content-Length: 0
```

References

[1] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J, Sparks R., Handley, M., and Schooler, E., "SIP: Session Initiation Protocol", Request for Comments 3261, Internet Engineering Task Force, June 2002. Available from http://www.ietf.org/.

[2] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", Request for Comments 3265, Internet Engineering Task Force, June 2002. Available from http://www.ietf.org/.

[3] Rosenberg, J., Schulzrinne, H., Mahy, R., "An INVITE-Initiated Dialog Event Package for the Session Initiation Protocol (SIP)", Request for Comments 4235, Internet Engineering Task Force, November 2005. Available from http://www.ietf.org/.

[4] Roach, A. B., Campbell, B., Rosenberg, J., "A Session Initiation Protocol (SIP) Event Notification Extension for Resource Lists", Request for Comments 4662, Internet Engineering Task Force, August 2006. Available from http://www.ietf.org/.

[5] BroadSoft, Inc. 2018. *Cisco BroadWorks Shared Call Appearance Interface Specification*, *Release 23.0*. Available from the BroadSoft Xchange at xchange.broadsoft.com.

[6] BroadSoft, Inc. 2018. *Busy Lamp Field and Application Server Redundancy Container Option Conversions Feature Description*, *Release 23.0*. Available from the BroadSoft Xchange at xchange.broadsoft.com.