# Cisco Meeting Server

## Cisco Meeting Server リリース 3.7

## コール詳細レコード（CDR）ガイド

2023 年 3 月 16 日

# 目次


変更履歴 3

1 はじめに 4

1.1 このドキュメントの使用方法 4

2 一般的なメカニズム 6

2.1 受信者デバイスの構成 6

2.1.1 Web 管理インターフェイスを使用した CDR 受信者の設定 6

2.1.2 API を使用した CDR 受信者の設定 6

2.1.3 受信者 URI 7

3 レコード タイプ 8

4 レコードの詳細 11

4.1 callStart レコードの内容 11

4.2 callEnd レコードの内容 12

4.3 callLegStart レコードの内容 13

4.4 callLegEnd レコードの内容 16

4.5 callLegUpdate レコードの内容 20

4.6 recordingStart レコードの内容 21

4.7 recordingEnd レコードの内容 21

4.8 streamingStart レコードの内容 22

4.9 streamingEnd レコードの内容 22

5 コールレッグ終了レコードの理由コード 23

6 トラフィックフローの例 25

付録 A CDR 受信者を作成するスクリプト例 29

Cisco の法的情報 31

Cisco の商標または登録商標 32


## 図：


図 1：リリース 3.7 の Cisco Meeting Server のドキュメント14 5

# 変更履歴

| 日付 | 変更点 |
|---|---|
| 2023 年 3 月 16 日 | バージョン 3.7 に更新しました。 |
| 2022 年 8 月 23 日 | バージョン 3.6 用に更新されました。 |
| 2022 年 4 月 20 日 | バージョン 3.5 用に更新されました。 |
| 2021 年 12 月 15 日 | バージョン 3.4 用に更新されました。 |
| 2021 年 9 月 16 日 | バージョン 3.3 用に更新されました。 |
| 2021 年 5 月 12 日 | callLegStart レコードコンテンツの subType から distributionLink を削除。 |
| 2021 年 4 月 9 日 | バージョン 3.2 で更新。 |
| 2020 年 7 月 29 日 | バージョン 3.0 用に更新し、X シリーズサーバーへの参照を削除。 |
| 2020 年 5 月 5 日 | セクション 4.6 に説明を追加 |
| 2020 年 4 月 8 日 | バージョン 2.9 用に更新されました。 |
| 2020 年 1 月 7 日 | 軽微な修正 |
| 2019 年 9 月 16 日 | callLegUpdate レコードから callMove と displayName が欠落。 |
| 2019 年 8 月 13 日 | タイトルを「2.6 以降」に変更、バージョン 2.7 から変更なし。 |
| 2019 年 7 月 19 日 | 軽微な修正 |
| 2019 年 4 月 23 日 | バージョン 2.6 用に更新されました。callLegStart レコードに canMove、movedCallLeg、movedCallLegCallBridge を追加。 |
| 2018 年 12 月 12 日 | タイトルを「2.4 以降」に変更、バージョン 2.5 から変更なし。 |
| 2018 年 9 月 20 日 | バージョン 2.4 用に更新されました。endpointRecorded を callEnd レコードに追加。 |
| 2017 年 12 月 20 日 | バージョン 2.3 用に再発行しました。追加や変更はありません。 |
| 2017 年 6 月 28 日 | callLegEnd レコードの mediaUsagePercentages に multiStreamVideo を追加。 |
| 2017 年 6 月 28 日 | CDR 受信者の作成例を追加。 |
| 2017 年 5 月 3 日 | バージョン 2.2 用に更新されました。callStart レコードに ownerName フィールドを追加しました。 |
| 2016 年 12 月 20 日 | バージョン 2.1 用に更新されました。追加および変更点が記載されています。 |
| 2016 年 8 月 3 日 | Cisco Meeting Server 2.0 用にブランド変更 |

# 1 はじめに

Cisco Meeting Server ソフトウェアは、Cisco ユニファイド コンピューティング サーバー（UCS）テクノロジーに基づく特定のサーバー、または仕様に基づく VM サーバーにホストできます。本書では、Cisco Meeting Server を Meeting Server と呼びます。

**注：**Cisco Meeting Server ソフトウェア バージョン 3.0 以降では、X シリーズサーバをサポートしません。

Meeting Server では、サーバ側で接続される新しい SIP 接続や、アクティブ化または非アクティブ化されたコールなど、キーコール関連イベントに関するコール詳細レコード（CDR）が内部で生成されます。

これらのレコードをリモートシステムに送信して収集および分析するようにサーバーを構成できます。Meeting Server でレコードを長期間保存する規定や、Meeting Server 上の CDR を参照する方法はありません。

CDR システムは、イベントと診断を相互に参照できるよう、2 つのシステム間でコール ID とコールレッグ ID の値が一致する場合は、この 2 つのシステムを Meeting Server API と組み合わせて使用できます。

Meeting Server は CDR 受信者を最大 4 人までサポートし、さまざまな管理ツールや、同じ管理ツールの複数のインスタンスを展開できます。

**詳しくは『Cisco Meeting Server API リファレンスガイド』を参照してください。**

## 1.1 このマニュアルの使用方法

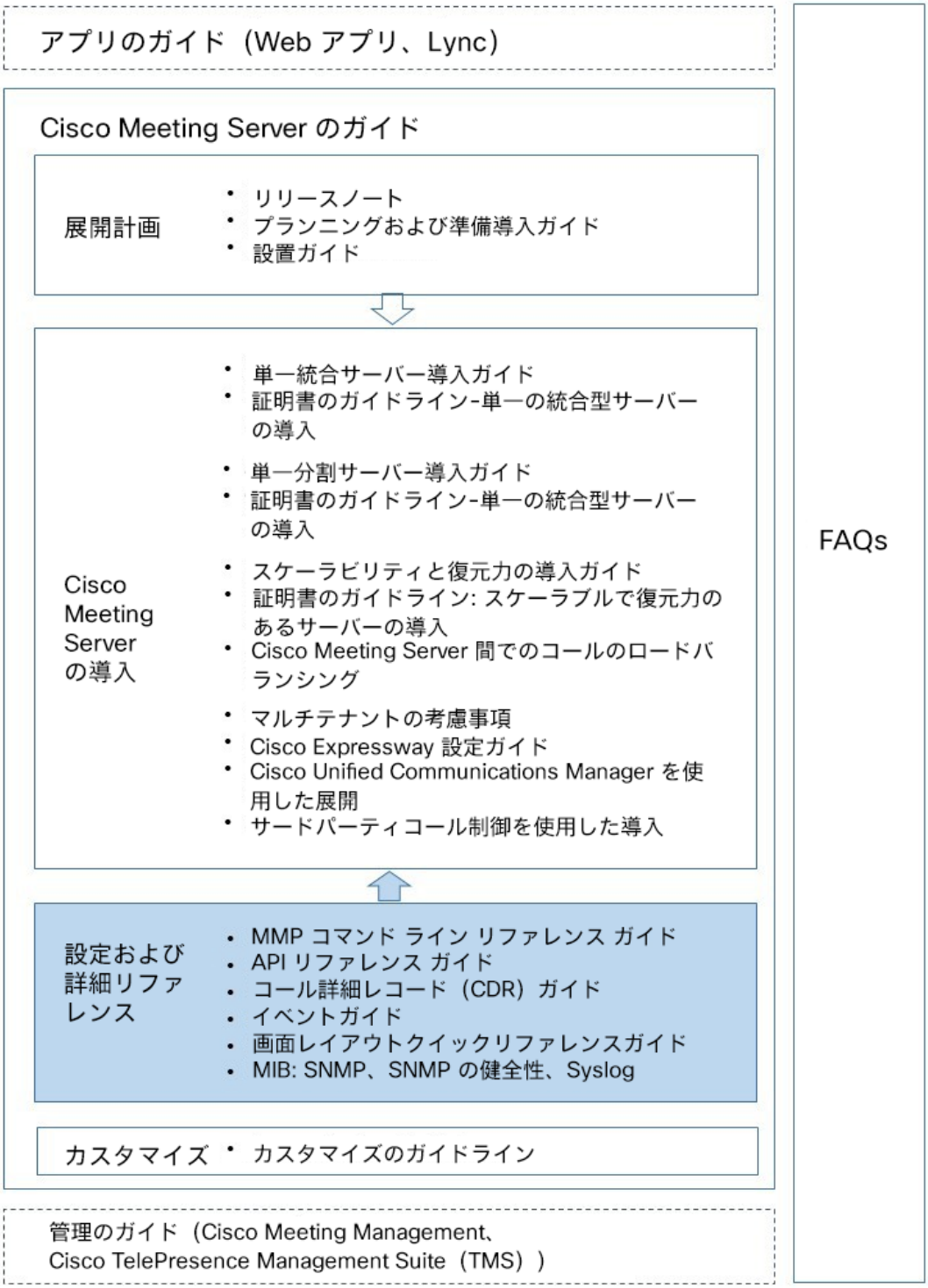
このドキュメントは、以下の図に示す多くのリファレンスガイドのうちの 1 つです。

ガイドはセクションに分かれており、前から順にお読みいただくと知識を習得できます。また、第 3 章、第 4 章、第 5 章は「必要なときにちょっと調べる」ことができるリファレンスとしてお使いいただけます。各 CDR レコードの種類とそのフィールドについて詳しく説明しています。

このドキュメントでは、情報の「最小限のセット」について説明しています。つまり、レコードの XML の性質によって、新たな要素が新規の Call Bridge ソフトウェアバージョンに表示される可能性があるため、生成されたレコードを解析する際は、必ずこれを許容する必要があります。受信者は、既存のドキュメントのバージョンに記載されていない新しい追加的な要素に対応する必要があります（同時に、ドキュメントに記載されていることを、ドキュメントに記載されている構造に従って提供することをお約束します）。

これらのドキュメントは cisco.com から入手できます。

**図 1：リリース 3.7 の Cisco Meeting Server のドキュメント**



# 2 一般的なメカニズム

CDR は、HTTP または HTTPS を介して、一連の XML ドキュメントとして Meeting Server によって送信されます。新しいレコードが生成されると、受信システムへの接続が確立され、この接続によって、受信システムでは 1 つまたは複数のレコードを受信することを想定する必要があります。Meeting Server によってレコードグループが受信者に正常に送信されると、そのレコードは Meeting Server では保存されなくなり、長期保存の責任が受信デバイスに移ります。HTTP または HTTPS 接続が正常に確立し、XML レコードデータが Meeting Server から送信され、受信者が「200 OK」の HTTP レスポンスでデータを確認した場合、Meeting Server ではレコードが受信者に正常に送信されたと見なします。

Call Bridge ではキープアライブ接続をサポートしており、1 つの TCP または TLS 接続で複数の（バッチの）レコードを受信者に送信できます。

**注：**複数の Call Bridge が単一のエンティティとして機能する拡張性と復元力がある展開では、各 Call Bridge によって実行中のコールレッグに CDR を提供します。各 CDR では、コールレッグの coSpace ID を識別します。次に、コールが複数の Call Bridge でホストされている場合は、異なる Call Bridge で同じコールを同じ coSpace ID で識別できます。

## 2.1 受信者デバイスの構成

**注：**CDR 受信者のリストは、個々の Call Bridge にローカルに保持され、クラスタ化された Call Bridge 間で共有されるデータベースには保存されません。

WEB 管理インターフェイスまたは API のいずれかを使用して、CDR 受信者を設定できます。

### 2.1.1 Web 管理インターフェイスを使用した CDR 受信者の設定

CDR の受信者を設定するには、次のステップを実行します。

1. [Web 管理インターフェイス（Web Admin Interface）] を開きます。
2. [**設定（Configure）**] > [CDR 設定（CDR settings）] の順に移動します。
3. [CDR 受信者設定（CDR Receiver Settings）] セクションで、各受信者について、受信者の HTTP または HTTPS URI を入力します（セクション 2.1.3 を参照）。

### 2.1.2 API を使用した CDR 受信者の設定

次の API オブジェクトを使用して、最大 4 人の CDR 受信者を Meeting Server に設定できるようにします。

- /system/cdrReceivers/ を使用して API で CDR 受信者の URI を設定します
- /system/cdrReceivers/ を使用して API で CDR 受信者の URI を設定します<CDR receiver id>

/system/cdrReceivers ノードで POST を発行して、新しい受信者の完全な URI を設定します。/system/cdrReceivers に GET リクエストを行うと、現在設定されている受信者が表示されます。

CDR 受信者が設定されると、/system/cdrReceivers/<CDR receiver id> ノードで GET または PUT を使用して、その詳細の読み取りや更新を行うことができます。CDR 受信者は、このノードの DELETE によって削除できます。

### 2.1.3 受信者 URI

Meeting Server で設定された受信者 URI は、次のいくつかの形式のうちの 1 つをとることができます。

- http://monitoring.example.com/cdr_receiver1
  リモートホスト「monitoring. example com」の TCP ポート 80 への単純な HTTP 接続の場合は、URI「/cdr_receiver1」に接続します。
- https://monitoring. example.com/cdr_receiver1
  上記と同じですが、HTTPS、TCP ポート 443 を使用します。
- http://monitoring. example.com:8080/cdr_receiver1
  上記と同じですが、デフォルトのポート番号（80）の代わりに TCP ポート 8080 を使用します。
- http://monitoring. example.com/cdr_receiver1?system_id=cms1
  上記と同じですが、パラメータ「system_id」と値「cms1」を接続先デバイスに提供します。Meeting Server は、URI フィールドで提供されたパラメータをそのまま相手に送信するだけで、その意味の解釈は受信デバイスに任されています。

# 3 レコード タイプ

CDR は、親の「<records>」要素であるアイテム内の「<record>」アイテムとして XML で送信されます。各レコードには、そのレコードに何が記述されているかを特定する「タイプ」値が関連付けられており、その中で想定されるフィールドや属性を決定しています。

「<records>」要素には次のものが含まれます。

- そのセッションに一意の GUID の形式をとる「セッション」値。セッション GUID は、Call Bridge の再起動時に作成されます。これは、実行中の特定の Call Bridge インスタンスでアクティブなすべての CDR 受信者で同じですが、その Call Bridge が再起動すると変更されます。これは、受信デバイスが、受信しているレコードが同じデバイス上の同じセッションから送信されていることを判断するために使用されます。
- Call Bridge GUID（Call Bridge がクラスタ内にある場合）。これは、クラスタ内のどの Call Bridge がレコードを送信しているかを識別します。これは、システムを再起動しても同じです。クラスタ化されていない展開には存在しないことに注意してください。Call Bridge GUID は、/callBridges API オブジェクトと同じです。

「<record>」の項目には次のものが含まれます。

- Meeting Server でレコードが生成された時刻の値。このタイムスタンプは RFC 3339/ISO 8601 形式です（たとえば、2014 年 2 月 28 日午後 4 時 3 分の場合は「2014-02-28T16:03:25Z」です）。現在、Meeting Server は常にこれらの時間を UTC で提供しています。
- 新しいレコードごとに 1 ずつ増加する「correlatorIndex」。注：「セッション GUID」と「correlatorIndex」の組み合わせにより、すべての受信者でレコードが一意に識別され、受信者は重複レコード受信の有無を判断できます。

  「correlatorIndex」は、起動後に Call Bridge が生成する最初のレコードの「0」から始まります。レコードの「correlatorIndex」は、すべての CDR 受信者で同じです。したがって、Call Bridge の起動後に設定された受信者の場合、受信する最初のレコードはインデックス 0 ではない場合があります。

  受信者がレコードを正常に受信すると、「200 OK」HTTP レスポンスを Call Bridge に送信する必要があります。次に、Call Bridge は次のレコードセットを受信者に送信します。「200 OK」HTTP レスポンスが Call Bridge によって正常に受信されない場合、Call Bridge はレコードを再送するため、受信者は重複レコードを受信することになります。

  リモート受信者が一定期間利用できず、Meeting Server によって生成されたすべてのレコードを内部に保存できなかった場合、リモート の受信者にプッシュされたレコードの「correlatorIndex」にギャップが生じます。

- 「recordIndex」は「correlatorIndex」に置き換えられています。「recordIndex」は現在推奨されておらず、将来のリリースで廃止される可能性があります。

万全を期すために、以下で「recordIndex」の使用方法について説明します。

「recordIndex」を使用すると、Meeting Server が重複レコードを受信したかどうかを判断できます。

**注：**複数の CDR 受信者がある場合、同じレコードでも、受信者によって「recordIndex」値が異なる場合があります。

この説明では、受信者が 1 人のみであることを前提としています。「<record>」項目内の「recordIndex」はレコードのシーケンスを決定します。これは、Call Bridge が生成する最初のレコードの「1」から始まり、送信される新しいレコードごとに 1 ずつ増加します。この「recordIndex」値により、CDR 受信者は重複レコードを受信したかどうかを判断できます。セッション GUID 値と recordIndex の組み合わせは一意です。Call Bridgee では、受信者から肯定確認応答（「200 OK」HTTP レスポンス）を受信していない CDR を再送します。受信者がそのような肯定の確認応答を送信しても、その応答が Call Bridge で正常に受信されなかった場合、受信者は重複レコードを受信する可能性があります。「recordIndex」により、受信者がこれを検出して、重複レコードを処理しないようにします。

リモート受信者が一定期間利用できず、Meeting Server によって生成されたすべてのレコードを内部に保存できなかった場合、リモート受信者にプッシュされたレコードには、「<record>」タグに「numPreceedingRecordsMissing」という数値が入ります。これにより、この数のレコード（そのヘッダー内の直前のレコード）が破棄され、使用できなくなったことをリモート受信者に通知します。CDR 受信者は「numPreceedingRecordsMissing」にゼロ以外の値が入っている場合でも「recordIndex」シーケンスの不連続性と認識すべきではありません。

レコードの種類については、以下の表 1 で簡単に説明し、第 4 章で詳しく説明します。

**表 1: レコードの種類の概要**

| レコード タイプ | 説明 |
|---|---|
| callStart | このレコードは、コールが作成されたとき、または coSpace から最初にインスタンス化されたときに生成されます。レコードには、コール ID、名前、関連付けられた coSpace ID が含まれます。 |
| callEnd | このレコードは、コールが終了したときに生成され、通常、コールの最後のコールレッグが切断された後に表示されます。このレコードには、以前の callStart レコードのコール ID と一致する必要があるコール ID と、コール内で同時にアクティブになったコールレッグの最大数など、コールの要約値が含まれています。 |

| レコード タイプ | 説明 |
|---|---|
| callLegStart | このレコードは、着信接続、発信コールレッグの確立、coSpace に接続する Cisco ミーティング アプリケーションのユーザーにより、コールレッグが最初に作成されたときに生成されます。レコードには、コールレッグ ID、リモートパーティのタイプ（SIP 接続または Cisco ミーティング アプリケーションデバイス）、リモートパーティの「名前」（SIP URI など）、および意味がある場合は、コールレッグが着信か発信かどうかが含まれます。 |
| callLegEnd | このレコードは、誰かが切断を選択したか、十分な権限を持つ別のユーザーが切断を選択したため、コールレッグが終了したときに生成されます。このレコードには、コールレッグ ID（以前の callLegStart レコードからのコールレッグ ID に対応する必要があります）、切断の理由、問題のコールレッグの有効期間に関連した特定の他の要約フィールド（どの音声やビデオコーデックが使用されていたかなど）が含まれます。 |
| callLegUpdate | このレコードは、コールレッグに大きな変更が発生した場合に生成されます。たとえば、そのコールレッグがコールの状態になった場合や、（発信の場合）応答があり「接続」状態に移行した場合などです。 |
| recordingStart | このレコードは、コールで録音が開始されたときに生成されます。レコードには、開始する録音 ID、録音の保存先のパス（ディレクトリとファイル名）、録音デバイスの URL、録音するコール ID、コールを録音するコールレッグ ID などが含まれます。 |
| recordingEnd | このレコードは、コールの録音が終了したときに生成されます。終了する録音の ID が含まれています。 |
| streamingStart | このレコードは、コールでストリーミングが開始されたときに生成されます。レコードには、開始するストリーミング ID、ストリーミング URL とストリーム名、ストリーミングデバイスの URL が含まれます。 |
| streamingEnd | このレコードは、コールのストリーミングが終了したときに生成されます。終了するストリーミングの ID が含まれています。 |

# 4 レコードの詳細

このセクションでは、レコードの種類ごとに「<record>」タグ内に表示されるパラメータ名や値の詳細を説明します。

## 4.1 callStart レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 開始中のコールの ID です。これは、callStart レコードをカプセル化する「<call>」タグ内の「ID」属性として伝送されます。 |
| name | 文字列 | コール名。通常は、コールが coSpace に関連付けられている場合の coSpace の名前です。 |
| coSpace | ID | このコールに関連付けられた coSpace の ID。このコールが coSpace に関連付けられていない場合（アドホックコールの場合など）は、このフィールドはありません。 |
| ownerName | 文字列 | このコールの所有者の名前です。優先順位の高い順に、次のいずれかから取得されます。<br>coSpace の「meetingScheduler」フィールドの値、<br>または<br>coSpace を所有するユーザーの名前、coSpace を所有するユーザーの jid、空白（これは、上記のいずれも存在しないことを意味します）。 |
| tenant | ID | マルチテナント展開では、テナントを指定します |
| cdrTag | 文字列 | coSpace にタグが指定されている場合（API リファレンスを参照）、これは callStart CDR に表示されます。タグはオプションで、コールの識別に使用される最大 100 文字のテキスト文字列です。 |
| callType | coSpace\|<br>adHoc\|<br>lyncConferencing\|<br>転送 | 次のいずれか：<br>*coSpace* - このコールは coSpace のインスタンス化です<br>*adHoc* - これはアドホック マルチパーティ コールです<br>*lyncConferencing* - このコールは、Lync がホストする会議への Meeting Server 接続です。<br>*forwarding* - これは、転送または「ゲートウェイ」コールです |

| パラメータ | タイプ | 説明 |
|---|---|---|
| callCorrelator | ID | この値は、複数の Call Bridge に分散している可能性がありますが、同じ coSpace またはアドホックコールのいずれかにあるコールレッグを識別するために使用できます。<br>注：coSpace 内のコールの場合、callCorrelator 値は coSpace の有効期間中は同じになります。アドホックコールごとに、値が動的に生成されます。 |
| coSpaceMetaDataConfigured | true \| false | このコールが含まれる coSpace でメタデータが設定されている場合、このフィールドは true に設定されます。コールがアドホックコールの場合、このフィールドは false になります。（バージョン 3.2 から） |

**注：**分散コールで、重複した「callStart」レコードが複数表示される場合は以下のようになります。

- 単一の coSpace ID の場合、これらのコールレッグは分散 coSpace コール、つまり複数の Call Bridge によってホストされる coSpace コールを構成します。API を使用して coSpace ID を検索できます。
- 単一の callCorrelator 値の場合、これらのコールレッグは分散コールを構成します。これは coSpace コールである場合もありますが、そうでない場合もあります。たとえば、各コールレッグが異なる Call Bridge によってホストされる「ポイントツーポイントコール」などです。

## 4.2 callEnd レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 終了するコールの ID です。以前の「callStart」レコードは、同じコールに対して生成されています。これは、callEnd record レコードをカプセル化する 「<call>」タグ内の「id」属性として伝送されます。 |
| callLegsCompleted | 番号（Number） | このコール内で完了したコールレッグの数。 |
| callLegsMaxActive | 番号（Number） | このコール内で同時にアクティブだったコールレッグの最大数。 |
| DurationSeconds | 番号（Number） | このコールがアクティブだった時間（秒単位）。 |
| endpointRecorded | true\|false | Skype や Lync クライアントなどのエンドポイントによって、常時コールが録音されている場合、値は true です。（バージョン 2.4 から） |

## 4.3 callLegStart レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 開始中のコールレッグの ID です。これは、callLegStart レコードをカプセル化する「<callLeg>」タグ内の「id」属性として伝送されます。 |
| displayName | 文字列 | SIP エンドポイントの「フレンドリ名」、Cisco ミーティング アプリケーション接続のユーザーの「実際の名前」、Web クライアントゲスト接続のユーザーが入力する名前。相手がフレンドリ名を提供しない場合、この値は空白になります。 |
| localAddress | 文字列 | コールレッグに関連する任意のローカルの接続先（発信者が Meeting Server に到達するために接続したものなど）。この値の解釈は、方向によって異なります（以下を参照）。したがって、着信コールの宛先アドレス、発信コールの場合は発信者 ID です。<br>注：一部のコールシナリオでは、localAddress は適用されません（URI が定義されていない coSpace から Cisco ミーティング アプリケーションのユーザーをコールする場合など）。 |
| remoteAddress | 文字列 | SIP コールの場合、コールレッグに関連するリモート URI。この値の解釈は、方向によって異なります（以下を参照）。これは、発信コールの接続先 URI または着信コールの送信元 URI です。 |
| remoteParty | 文字列 | このコールレッグのリモートパーティのアドレス。発信コールの場合、これはダイヤル変換の出力であり、ドメインが含まれていない場合があります。 |
| cdrTag | 文字列 | コールレッグにタグが付けられている場合、これは CDR に表示されます。タグはオプションで、コールレッグの識別に使用される最大 100 文字のテキスト文字列です。 |
| guestConnection | true\|false | （オプション）WebRTC アプリケーション経由で開始されたゲストログインであることがわかっている接続には、ここでは True の値になります。 |
| recording | true\|false | ここでは、コールを録音していることがわかっている接続の値は「true」です。 |
| streaming | true\|false | ここでは、通話をストリーミングしていることがわかっている接続の値は「true」です。 |

| パラメータ | タイプ | 説明 |
|---|---|---|
| type | sip\|acano | コールレッグのタイプ：Cisco ミーティング アプリケーション接続の場合は「acano」、SIP 接続の場合は「sip」です。 |
| subType | lync\|<br>avaya\|<br>lyncdistribution\|<br>webApp | コールレッグタイプをさらに分化するもので、コールレッグが「sip」の場合、ここで使用できる値は「lync」、「avaya」、「lyncdistribution」、「webApp」です。 |
| lyncSubType | audioVideo\|<br>applicationSharing\|<br>instantMessaging | コールレッグタイプをさらに分化するもので、コールレッグのサブタイプが「lync」の場合は以下のものが使用できます。<br>*audioVideo* - これは、Call Bridge と Lync の間でオーディオとビデオを交換するために使用される Lync コールレッグです。<br>*applicationSharing* - これは、Lync と Call Bridge 間のアプリケーションまたはデスクトップ共有に使用される Lync コールレッグです。<br>*InstantMessaging* - これは、Lync と Call Bridge 間のインスタントメッセージの交換に使用される Lync コール レッグです。 |
| direction | incoming\|<br>outgoing | sip と「acano」の両方のコールタイプの場合：<br>*incoming* - リモート SIP デバイスが Meeting Server への接続を開始した場合。<br>*outgoing* - Meeting Server からリモート SIP デバイスへのコールレッグが確立された場合。 |
| call | ID | このコールレッグのコール ID です。コールレッグの開始時に既知の場合は、これを含めることができます。それ以外の場合は、後の callLegUpdate レコードで通知されます。 |
| ownerId | ID | 管理するリモートシステムがこのコールレッグに割り当てるために選択した ID です。そのリモートシステムに限って意味があります。このコールレッグにそのような所有者 ID が割り当てられていない場合、このフィールドはありません。 |
| sipCallId | 文字列 | コールレッグが SIP 接続である場合、このフィールドがコールレッグの開始時に既知の場合、SIP プロトコルヘッダーからの一意の「Call-ID」値を保持します。 |

| パラメータ | タイプ | 説明 |
|---|---|---|
| groupId | ID | Lync 通話の場合のみ、コンテンツを共有している場合、このパラメータによってプレゼンターのビデオ callLeg とプレゼンテーション ストリームをリンクします。これは、このコールレッグに関連する「参加者」API の操作を実行する際に使用する必要がある ID でもあります。<br><br>Lync プレゼンテーションでは、CDR に追加の callLeg を作成でき、groupId パラメータを使用してこれらを結びつけることができます。（callId はもちろん同じですが、同じユーザーが所有していない他の Lync コールレッグがコール内に存在する可能性があります。これは、Lync の「接続」に固有の groupId です。）<br><br>Lync の発信者が共有を停止して再開した場合、コンテンツ共有接続のコールレッグ ID は最初のプレゼンテーションのものとは異なりますが、groupID は同じになります。 |
| replacesSipCallId | 文字列 | コールレッグによって別の SIP コールを置き換える場合、このフィールドには、置き換えられたコールの SIP プロトコルヘッダーから一意の「Call-ID」値（文字列として）が保持されます。 |
| canMove | true\|false | このコールレッグを所有している参加者を、movedParticipant API コマンドを使用して移動できるかどうかを示します。（バージョン 2.6 から） |
| movedCallLeg | ID | このコールレッグが参加者の移動の一環として作成された場合、ID はその参加者が移動したコールレッグの GUID になります。（バージョン 2.6 から） |
| movedCallLegCallBridge | ID | このコールレッグが参加者の移動の一環として作成された場合、ID は、移動した参加者のコールレッグがホームになっている会議をホストしている Call Bridge の GUID です。（バージョン 2.6 から） |
| confirmationStatus | required/notRequired/confirmed | - required：confirmation=true が設定されていて、ユーザが通話に参加するための DTMF 確認をまだ提供していないことを意味します。<br>- notRequired：は、confirmation=true が構成されていないことを意味します。<br>- confirmed：参加者が通話への参加を望んでいることを確認するために DTMF シーケンスが入力されたことを意味します。 |

## 4.4 callLegEnd レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 終了するコールレッグの ID です。これは、callLegEnd レコードをカプセル化する「<callLeg>」タグ内の「id」属性として伝送されます。 |
| cdrTag | 文字列 | コールレッグにタグが付けられている場合、これは CDR に表示されます。タグはオプションで、コールレッグの識別に使用される最大 100 文字のテキスト文字列です。 |
| reason | 文字列 | コールレッグが終了する理由（セクション 5 の表を参照）。 |
| remoteTeardown | true\|false | *true* - このコールレッグの終了がリモートパーティによって開始されたことを示します<br>*false* - このコールレッグの終了が Meeting Server によって開始されたことを示します |
| reasonDetails | 文字列 | リモート切断によってコールレッグが終了した場合、パラメータは、そのコールが WebRTC コールか SIP コールかを示します。 |
| encryptedMedia<br>unencryptedMedia | | これらの一方の値または両方の値は、コールレッグの有効期間中に暗号化メディアまたは非暗号化メディア有無（値が「true」または「false」に基づく）を示すために存在することがあります。存在しない場合、このコールレッグには該当のメディアタイプは存在しませんでした。 |
| DurationSeconds | 番号（Number） | このコールレッグがアクティブだった時間（秒単位）。 |
| activatedDuration | 番号（Number） | このコールレッグがアクティブ化された時間（秒単位）。 |

<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>mediaUsagePercentages</td><td></td><td>さまざまなタイプのメディアがアクティブだった、このコールレッグの有効時間の割合に関する情報です。メディアタイプは次のとおりです。<br>mainVideoViewer - ユーザーはメインビデオを受信していました<br>mainVideoContributor - ユーザーはメインビデオに投稿していました<br>presentationViewer - ユーザーはプレゼンテーション情報を受信していました<br>presentationContributor - ユーザーはプレゼンテーションを共有していました<br>multistreamVideo - multistreamVideo がアクティブだった時間の割合。</td></tr>
<tr><td>multistreamVideo</td><td colspan="2">このコールレッグの有効期間中に送信されたマルチストリームビデオに関する情報。
<table>
<tr><th>名前</th><th>タイプ</th><th>説明</th></tr>
<tr><td>maxScreens</td><td>数値</td><td>このコールレッグの有効期間中にアクティブなマルチスクリーンのメインビデオ画面 の最大数です。たとえば、デュアルビデオの場合は 2 になります。</td></tr>
</table></td></tr>
<tr><td>alarm</td><td colspan="2">コールレッグがその有効期間中にアラーム状態を検知した場合、これらの要素が 1 つまたは複数存在します。
<table>
<tr><th>名前</th><th>タイプ</th><th>説明</th></tr>
<tr><td>type</td><td>packetLoss|<br>excessiveJitter|<br>highRoundTripTime</td><td>packetLoss - ローカルでパケット損失が過剰に発生したか、またはこのコールレッグの相手からレポートがありました。<br>excessiveJitter - ローカルで高いジッター値が観察されたか、またはこのコールレッグの相手からレポートがありました<br>highRoundTripTime - このコールレッグで、Meeting Server とリモートパーティ間でラウンドトリップ時間の遅延が検出されました この値は、</td></tr>
<tr><td>durationPercentage</td><td>数値</td><td>アラーム状態が発生した通話時間の割合を示します。</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>rxAudio<br>txAudio</td><td colspan="2">このコールレッグの有効期間中に受信した音声（「rxAudio」、Meeting Server がリモートパーティから受信した音声）および送信した音声（「txAudio」）についての詳細を提供します。rxAudio および txAudio セクションには、次の要素が含まれる場合があります。
<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>codec</td><td>次のいずれか<br>g711u<br>g711a<br>g722<br>g728<br>g729<br>g722_1<br>g722_1c<br>aac<br>speexNb<br>speexWb<br>speexUwb<br>isacWb<br>opus</td><td>使用されるオーディオコーデック<br>g711u - G.711 mu law<br>g711a - G.711 a law<br>g722 - G.722<br>g728 - G.728<br>g729 - G.729<br>g722_1 - G.722.1<br>g722_1c - G.722.1C (G.722.1 Annex C)<br>aac - AAC<br>speexNb - Speex 狭帯域<br>speexWb - Speex 広帯域<br>speexUwb- Speex 超広帯域<br>isacWb- iSAC（インターネット スピーチ オーディオコーデック）広帯域<br>isacSwb - iSAC（インターネット スピーチ オーディオコーデック）超広帯域</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>rxVideo<br>txVideo</td><td colspan="2">このコールレッグの有効期間中に受信したビデオ（「rxVideo」、Meeting Server がリモートパーティから受信したビデオ）および送信したビデオ（「txVideo」）についての詳細を提供します。rxVideo セクションと txVideo セクションには、次の要素が含まれる場合があります。
<table>
<tr><th>名前</th><th>タイプ</th><th>説明</th></tr>
<tr><td>codec</td><td>次のいずれか<br>h261<br>h263<br>h263+<br>h264<br>h264Lync<br>vp8<br>rtVideo</td><td>使用されるビデオコーデック<br>h261 - H.261<br>h263 - H.263<br>h263+ - H.263+<br>h264 - H.264<br>h264Lync - H.264 SVC Lync 向け<br>vp8 - VP8<br>rtVideo- RTVideo</td></tr>
<tr><td>maxSizeWidth</td><td>数値</td><td>使用される最大のビデオ解像度の帯域幅</td></tr>
<tr><td>maxSizeHeight</td><td>数値</td><td>使用される最大のビデオ解像度の高さ</td></tr>
</table>
注：「rxVideo」または「txVideo」セクションがない場合、ビデオはその方向に送信されていません。</td></tr>
<tr><td>ownerId</td><td>ID</td><td>管理するリモートシステムがこのコールレッグに割り当てるために選択した ID です。そのリモートシステムに限って意味があります。</td></tr>
<tr><td>sipCallId</td><td>文字列</td><td>コールレッグが SIP 接続の場合、このフィールドは SIP プロトコルヘッダーからの一意の「Call-ID」値を保持します。</td></tr>
<tr><td>confirmationStatus</td><td>required/notRequired/confirmed/rejected</td><td>- required：confirmation=true が設定されていて、ユーザが通話に参加するための DTMF 確認をまだ提供していないことを意味します。<br>- notRequired：は、confirmation=true が構成されていないことを意味します。<br>- confirmed：参加者が通話への参加を望んでいることを確認するために DTMF シーケンスが入力されたことを意味します。<br>- rejected：コールを拒否するために DTMF シーケンスが入力されたことを意味します。会議の管理は、参加者への再ダイヤルを停止します。</td></tr>
</table>

## 4.5 callLegUpdate レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 更新中のコールレッグの ID です。これは、callLegUpdate レコードをカプセル化する「<callLeg>」タグ内の「id」属性として伝送されます。 |
| cdrTag | 文字列 | コールレッグにタグが付けられている場合、これは CDR に表示されます。タグはオプションで、コールレッグの識別に使用される最大 100 文字のテキスト文字列です。 |
| state | 接続されているか、値がありません | 存在する場合、コールレッグの状態の表示が含まれます。現在、「接続済み」の値のみがサポートされています。この値がない場合は、コールレッグがまだ接続状態に達していないことを示します。 |
| deactivated | true\|false | コールレッグが現在非アクティブ化されているかどうかを示します。 |
| remoteAddress | 文字列 | SIP コールの場合、コールレッグに関連するリモート URI。この値の解釈は、方向によって異なります（以下を参照）。したがって、これは発信コールの接続先 URI、または着信コールの送信元 URI です。 |
| call<br>IVR | | このコールレッグのコール ID、コールレッグが現在 IVR にある場合は（空の）「ivr」表示。 |
| ownerId | ID | 管理するリモートシステムがこのコールレッグに割り当てるために選択した ID です。そのリモートシステムに限って意味があります。 |
| sipCallId | 文字列 | コールレッグが SIP 接続である場合、このフィールドがコールレッグの開始時に既知の場合、SIP プロトコルヘッダーからの一意の「Call-ID」値を保持します。 |
| groupId | ID | Lync コールの場合のみ、このパラメータによってプレゼンターのビデオ callLeg と送信されるプレゼンテーション ストリームをリンクします。 |
| displayName | 文字列 | SIP エンドポイントの「フレンドリ名」、Cisco ミーティング アプリケーション接続のユーザーの「実際の名前」、Web クライアントゲスト接続のユーザーが入力する名前。相手がフレンドリ名を提供しない場合、この値は空白になります。 |
| canMove | true\|false | このコールレッグを所有している参加者を、movedParticipant API コマンドを使用して移動できるかどうか。 |

| パラメータ | タイプ | 説明 |
|---|---|---|
| confirmationStatus | required/notRequired/confirmed | - required：confirmation=true が設定されていて、ユーザが通話に参加するための DTMF 確認をまだ提供していないことを意味します。<br>- notRequired：は、confirmation=true が構成されていないことを意味します。<br>- confirmed：参加者が通話への参加を望んでいることを確認するために DTMF シーケンスが入力されたことを意味します。 |

callLegUpdate レコードは、参照するコールレッグの特性のいずれかがコールレッグに対して変更された場合に、Meeting Server によって送信されます。たとえば、コールレッグが IVR からコールに移動した場合、または外部管理システムがそのコールレッグに関連付けられた「ownerId」を変更した場合、CDR 受信者はそのような更新レコードが表示されることを想定します。

## 4.6 recordingStart レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 開始中のレコードの ID です。これは、recordingStart レコードをカプセル化する「<recording>」タグ内の「id」属性として伝送されます。 |
| path | 文字列 | 録音のディレクトリとファイル名を保持する文字列。（内蔵 XMPP レコーダーのみに適用されます。） |
| recorderUri | 文字列 | SIP レコーダーの場合は録音デバイスの URI です。（外部のサードパーティ SIP レコーダーのみに適用されます。） |
| call | ID | 録音中のコールの ID です。 |
| callLeg | ID | コールを録音中のコールレッグの ID です。 |

## 4.7 recordingEnd レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 終了中の録音の ID です。これは、recordingEnd レコードをカプセル化する「<recording>」タグ内の「id」属性として伝送されます。 |

## 4.8 streamingStart レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 開始中のストリーミングの ID です。これは、streamingStart レコードをカプセル化する「<streaming>」タグ内の「id」属性として伝送されます。 |
| streamerUri | URL | ストリーミングデバイスの URL です。（内部 SIP ストリーマコンポーネントに適用されます。） |
| call | ID | ストリーミング中のコールの ID です。 |
| callLeg | ID | コールをストリーミング中のコールレッグの ID です。 |

## 4.9 streamingEnd レコードの内容

| パラメータ | タイプ | 説明 |
|---|---|---|
| id | ID | 終了中のストリーミングの ID です。これは、streamingEnd レコードをカプセル化する「<streaming>」タグ内の「id」属性として伝送されます。 |

# 5 コールレッグ終了レコードの理由コード

コールレッグ終了レコードには、理由コードが含まれます（「<reason>」タグ内）と、Meeting Server またはリモートパーティがそのコールレッグの切断を選択したかどうかを示す個々の表示（「true」または「false」のいずれかを含む「<remoteTeardown>」セクション）が含まれます。

切断理由によって切断を引き起こしたパーティを判断できますが、個々のリモートまたはローカルの切断表示により、CDR 受信者が理解できない新しい理由コードが追加された場合でも、切断を開始した側の基本的な情報が得られる程度に、将来の変化に対応できます。

「<reason>」コードに指定できる値は次のとおりです。

| 理由 | 説明 |
|---|---|
| apiInitiatedTeardown | API リクエストに応じて、Meeting Server によってコールレッグを切断しました。 |
| callDeactivated | コールレッグの一部であるコールが非アクティブ化され、コールレッグの非アクティブ化アクションが「切断」に設定されていたため、Meeting Server によってコールレッグが切断されました。詳細については、『API リファレンス』を参照してください |
| callEnded | コールレッグの一部であったコールが終了したため（破棄するための API コマンドに応答した場合など）、 Meeting Server によってコールレッグが切断されました。 |
| callMoved | Call Bridge のリソースの使用効率を向上させるため、コールレッグが移動しました。 |
| clientInitiatedTeardown | 十分な権限を持つ Cisco ミーティング アプリケーションによるリクエストに応じて、Meeting Server によってコールレッグが切断されました。 |
| confirmationTimeOut | リモートの接続先が時間内に応答しなかったため、コールレッグが切断されました。「コールに招待されました。参加するには 1 を押してください」という音声プロンプトが再生されましたが、相手が 1 分以内にキーを押さなかったため、この理由コードを使用してコールレッグが切断されました。 |
| dnsFailure | リモートシステムへの接続を確立するプロセスの一部として、リモートの接続先ホスト名の解決に失敗した場合など |
| encryptionRequired | 暗号化メディアの要件が満たされなかったため、コールレッグが切断されました |
| error | SIP コール中にエラーが発生し、コールレッグが切断されました。これは、コール中に SIP エンドポイントの電源が切れたり、クラッシュしたりすることが原因である可能性があります。これが繰り返し発生する場合は、SIP トレースをオンにします。 |
| incorrectPasscode | 再試行の最大回数に達した後、ユーザーが参加するコールまたは coSpace の正しい PIN を入力しなかった場合 |

| 理由 | 説明 |
|---|---|
| ivrTimeout | コールレッグが IVR に接続されましたが、必要な時間内にコールに移行できませんでした |
| ivrUnknownCall | 最大再試行回数の後、ユーザーが IVR に参加するための有効なコール ID を提供しませんでした |
| localTeardown | Meeting Server によるコールレッグの通常の切断 |
| participantLimitReached | コールに許可されている最大数を超えて新しい参加者を追加しようとしました |
| remoteBusy | リモートパーティがビジーで接続を受け入れることができなかったため、コールレッグが切断されました |
| remoteRejected | コールレッグがリモートパーティによって拒否されました |
| remoteTeardown | コールレッグがリモートパーティによって接続解除されました。リモート切断中、**`reasonDetails`** パラメータは、コールが WebRTC コールか SIP コールかを示します。 |
| ringingTimeout | コールレッグがリモートデバイスに到達して呼び出し音は鳴動したが、必要な時間内に応答がありませんでした |
| tenantParticipantLimitReached | 所有テナントに許可されている最大数を超えて、新しい参加者を追加しようとしました |
| timeout | プロトコルタイムアウト（SIP セッションタイムアウトや、SIP リクエストに対する必須の応答がないなど）のため、Meeting Server によってコールレッグが切断されました。 |
| unknownDestination | このコールレッグは、有効な coSpace またはユーザーに解決されなかった接続先への着信接続でした |

# 6 トラフィックフローの例

次のトレースは、典型的なトラフィックフローの例を示しています。2 つの SIP クライアントが会議に接続し、一方が会議を終了し、もう一方が SIP コールが切断される例について取り上げています。この例にある XML は、読みやすいようにフォーマットされています。

Events post #1

```
<?xml version="1.0"?>
<records session="a865433a-4926-4549-a701-9bb5b93c75e6"
callBridge="158ba4f7-70eb-4a35-982c-71d4f1674277">
  <record type="callLegStart" time="2015-07-23T07:32:55Z" recordIndex="1"
correlatorIndex="0">
    <callLeg id="fc9c85ca-8c41-4a1a-9252-b16977d1e4e1">
      <remoteParty>sipclient1@example.com</remoteParty>
      <localAddress>access1@127.0.0.1</localAddress>
      <type>sip</type>
      <direction>incoming</direction>
      <groupId>18da80f3-8a71-4255-aa90-e1677b99b588</groupId>
      <sipCallId>b8a81da5-c24c-43db-ba58-742f587faec8</sipCallId>
    </callLeg>
  </record>
</records>
```

Events post #2

```
<?xml version="1.0"?>
<records session="a865433a-4926-4549-a701-9bb5b93c75e6"
callBridge="158ba4f7-70eb-4a35-982c-71d4f1674277">
  <record type="callStart" time="2015-07-23T07:32:55Z" recordIndex="2"
correlatorIndex="1">
    <call id="46d49cb4-8171-4abc-97f5-b88035b1da0a">
      <name>test564_1</name>
      <callType>coSpace</callType>
      <coSpace>50605235-60cf-484a-9fa1-278ad0646243</coSpace>
      <callCorrelator>5f3300c5-ca67-40e0-a503-
91baec70dbbe</callCorrelator>
    </call>
  </record>
  <record type="callLegUpdate" time="2015-07-23T07:32:55Z"
recordIndex="3" correlatorIndex="2">
    <callLeg id="fc9c85ca-8c41-4a1a-9252-b16977d1e4e1">
      <state>connected</state>
      <call>46d49cb4-8171-4abc-97f5-b88035b1da0a</call>
      <groupId>18da80f3-8a71-4255-aa90-e1677b99b588</groupId>
      <sipCallId>b8a81da5-c24c-43db-ba58-742f587faec8</sipCallId>
    </callLeg>
  </record>
</records>
```

Events post #3

```
<?xml version="1.0"?>
<records session="a865433a-4926-4549-a701-9bb5b93c75e6"
callBridge="158ba4f7-70eb-4a35-982c-71d4f1674277">
  <record type="callLegStart" time="2015-07-23T07:32:55Z" recordIndex="4"
correlatorIndex="3">
    <callLeg id="9cfdb064-3ae9-4b08-a003-6478187f375f">
      <remoteParty>sipclient2@example.com</remoteParty>
      <localAddress>access2@127.0.0.1</localAddress>
      <type>sip</type>
      <direction>incoming</direction>
      <groupId>3420c93f-f33c-4c9e-be95-d0d1bfb207f0</groupId>
      <sipCallId>a939937c-8b5e-4376-92de-97635983d7ef</sipCallId>
    </callLeg>
  </record>
</records>
```

Events post #4

```
<?xml version="1.0"?>
<records session="a865433a-4926-4549-a701-9bb5b93c75e6"
callBridge="158ba4f7-70eb-4a35-982c-71d4f1674277">
  <record type="callLegUpdate" time="2015-07-23T07:32:55Z"
recordIndex="5" correlatorIndex="4">
    <callLeg id="9cfdb064-3ae9-4b08-a003-6478187f375f">
      <state>connected</state>
      <call>46d49cb4-8171-4abc-97f5-b88035b1da0a</call>
      <groupId>3420c93f-f33c-4c9e-be95-d0d1bfb207f0</groupId>
      <sipCallId>a939937c-8b5e-4376-92de-97635983d7ef</sipCallId>
    </callLeg>
  </record>
</records>
```

Events post #5

```
<?xml version="1.0"?>
<records session="a865433a-4926-4549-a701-9bb5b93c75e6"
callBridge="158ba4f7-70eb-4a35-982c-71d4f1674277">
  <record type="callLegEnd" time="2015-07-23T07:33:05Z" recordIndex="6"
correlatorIndex="5">
    <callLeg id="9cfdb064-3ae9-4b08-a003-6478187f375f">
      <reason>remoteTeardown</reason>
      <remoteTeardown>true</remoteTeardown>
      <durationSeconds>10</durationSeconds>
      <mediaUsagePercentages>
        <mainVideoViewer>100.0</mainVideoViewer>
        <mainVideoContributor>100.0</mainVideoContributor>
      </mediaUsagePercentages>
      <unencryptedMedia>true</unencryptedMedia>
      <rxAudio>
        <codec>g722</codec>
        <packetStatistics>
```

```
            <packetLossBursts>
              <duration>0.000</duration>
              <density>0.00</density>
            </packetLossBursts>
            <packetGap>
              <duration>9.701</duration>
              <density>0.00</density>
            </packetGap>
          </packetStatistics>
        </rxAudio>
        <txAudio>
          <codec>g722_1c</codec>
        </txAudio>
        <rxVideo>
          <codec>h264</codec>
          <maxSizeWidth>768</maxSizeWidth>
          <maxSizeHeight>448</maxSizeHeight>
          <packetStatistics>
            <packetLossBursts>
              <duration>0.000</duration>
              <density>0.00</density>
            </packetLossBursts>
            <packetGap>
              <duration>8.597</duration>
              <density>0.00</density>
            </packetGap>
          </packetStatistics>
        </rxVideo>
        <txVideo>
          <codec>h264</codec>
          <maxSizeWidth>1280</maxSizeWidth>
          <maxSizeHeight>720</maxSizeHeight>
        </txVideo>
        <sipCallId>a939937c-8b5e-4376-92de-97635983d7ef</sipCallId>
      </callLeg>
    <record>
</records>
```

Events post #6

```
<?xml version="1.0"?>
<records session="a865433a-4926-4549-a701-9bb5b93c75e6"
callBridge="158ba4f7-70eb-4a35-982c-71d4f1674277">
  <record type="callLegEnd" time="2015-07-23T07:33:05Z" recordIndex="7"
correlatorIndex="6">
    <callLeg id="fc9c85ca-8c41-4a1a-9252-b16977d1e4e1">
      <reason>callDeactivated</reason>
      <remoteTeardown>false</remoteTeardown>
      <durationSeconds>10</durationSeconds>
      <mediaUsagePercentages>
        <mainVideoViewer>100.0</mainVideoViewer>
        <mainVideoContributor>100.0</mainVideoContributor>
      </mediaUsagePercentages>
```

```
      <unencryptedMedia>true</unencryptedMedia>
      <rxAudio>
        <codec>g711u</codec>
        <packetStatistics>
          <packetLossBursts>
            <duration>0.000</duration>
            <density>0.00</density>
          </packetLossBursts>
          <packetGap>
            <duration>9.702</duration>
            <density>0.00</density>
          </packetGap>
        </packetStatistics>
      </rxAudio>
      <txAudio>
        <codec>g722_1c</codec>
      </txAudio>
      <rxVideo>
        <codec>h264</codec>
        <maxSizeWidth>1280</maxSizeWidth>
        <maxSizeHeight>720</maxSizeHeight>
        <packetStatistics>
          <packetLossBursts>
            <duration>0.000</duration>
            <density>0.00</density>
          </packetLossBursts>
          <packetGap>
            <duration>8.484</duration>
            <density>0.00</density>
          </packetGap>
        </packetStatistics>
      </rxVideo>
      <txVideo>
        <codec>h264</codec>
        <maxSizeWidth>1024</maxSizeWidth>
        <maxSizeHeight>576</maxSizeHeight>
      </txVideo>
      <sipCallId>b8a81da5-c24c-43db-ba58-742f587faec8</sipCallId>
    </callLeg>
  </record>
  <record type="callEnd" time="2015-07-23T07:33:05Z" recordIndex="8"
correlatorIndex="7">
    <call id="46d49cb4-8171-4abc-97f5-b88035b1da0a">
      <callLegsCompleted>2</callLegsCompleted>
      <callLegsMaxActive>2</callLegsMaxActive>
      <durationSeconds>10</durationSeconds>
    </call>
  </record>
</records>
```

# 付録 A CDR 受信者を作成するスクリプト例

次の Python スクリプトは、CDR 受信者を作成する方法を示しています。この例で説明のみを目的としており、コードの使用に関してCisco はサポートや保証しておりません。Cisco はコードの著作権を留保します。

```
#!/usr/bin/python


## Cisco Meeting Server の CDR 受信者コードの例
## Copyright - Cisco SYSTEMS (2013-2017)
## このコードにはサポート、保証、義務はありません



import BaseHTTPServer
import sys
import getopt
import ssl

class RequestHandler(BaseHTTPServer.BaseHTTPRequestHandler):
     handler = BaseHTTPServer.BaseHTTPRequestHandler
     handler.protocol_version = 'HTTP/1.1'
     print "using protocol version:", handler.protocol_version


     def do_GET(self) :
          #print 'received request for GET', self.path
          self.send_response(200)
          self.end_headers()
     def do_POST(self) :
          print 'received request for POST', self.path
          length = int(self.headers['Content-Length'])
          post_data = self.rfile.read(length)
          print 'data:', post_data
          self.send_response(200)
          self.end_headers()
     def log_message(self, format, *args):
          return


def main(argv) :
     try:
```

```
            opts, args = getopt.getopt(argv, 'p:c:k:')
            port = [val for opt,val in opts if opt=='-p'][0]
            assert(len(port) > 0)
            certfile_name = ''
            keyfile_name = ''
            for opt,val in opts :
                  if opt=='-c' :
                        certfile_name = val
                  if opt=='-k' :
                        keyfile_name = val
      except:
            print 'usage: cdr_receiver.py -p <port> [-c <certfile path>] [-
            k <keyfile path>]'
            sys.exit(2)
      server_address = ('', int(port))
      httpd = BaseHTTPServer.HTTPServer(server_address, RequestHandler)
      if (len(certfile_name) > 0) :
            print 'HTTPS mode with certfile', certfile_name
            httpd.socket = ssl.wrap_socket (httpd.socket, keyfile=keyfile_
            name, certfile=certfile_name, server_side=True)
      try :
            httpd.serve_forever()
      except KeyboardInterrupt:
            pass
      httpd.server_close()

if name      == " main ":
      main(sys.argv[1:])
```

# Cisco の法的情報

このマニュアルに記載されている仕様および製品に関する情報は、予告なしに変更されることがあります。このマニュアルに記載されている表現、情報、および推奨事項は、すべて正確であると考えていますが、明示的であれ黙示的であれ、一切の保証の責任を負わないものとします。このマニュアルに記載されている製品の使用は、すべてユーザー側の責任となります。

対象製品のソフトウェア ライセンスと限定保証は、製品に添付された『Information Packet』に記載されています。添付されていない場合には、代理店にご連絡ください。

Cisco が採用している TCP ヘッダー圧縮機能は、UNIX オペレーティング システムの UCB（University of California, Berkeley）のパブリック ドメイン バージョンとして、UCB が開発したプログラムを採用したものです。All rights reserved. Copyright © 1981, Regents of the University of California.

ここに記載されている他のいかなる保証にもよらず、各社のすべてのマニュアルおよびソフトウェアは、障害も含めて「現状のまま」として提供されます。シスコおよび上記代理店は、商品性、特定目的適合、および非侵害の保証、もしくは取り引き、使用、または商慣行から発生する保証を含み、これらに限定することなく、明示または暗黙のすべての保証を放棄します。

いかなる場合においても、Cisco およびその供給者は、このマニュアルの使用または使用できないことによって発生する利益の損失やデータの損傷をはじめとする、間接的、派生的、偶発的、あるいは特殊な損害について、あらゆる可能性がCisco またはその供給者に知らされていても、それらに対する責任を一切負わないものとします。

このマニュアルで使用している Internet Protocol（IP）アドレスおよび電話番号は、実際のアドレスおよび電話番号を示すものではありません。マニュアルの中の例、コマンド出力、ネットワーク トポロジー図、およびその他の図は、説明のみを目的として使用されています。説明の中に実際の IP アドレスおよび電話番号が使用されていたとしても、それは意図的なものではなく、偶然の一致によるものです。

この文書の印刷されたハード コピーおよび複製されたソフト コピーは、すべて管理対象外と見なされます。最新版については、現在のオンライン バージョンを参照してください。

Cisco は世界各国 200 箇所にオフィスを開設しています。各オフィスの住所と電話番号は、当社の Web サイト www.cisco.com/go/offices をご覧ください。

# Cisco の商標または登録商標

Cisco および Cisco ロゴは、Cisco Systems, Inc. またはその関連会社の米国およびその他の国における登録商標または商標です。Cisco の商標の一覧については、https://www.cisco.com/c/ja_jp/about/legal/trademarks.html をご覧ください。本書に記載されているサードパーティの商標は、それぞれの所有者の財産です。「パートナー」という用語の使用は Cisco と他社との間のパートナーシップ関係を意味するものではありません。(1721R)