Cisco Meeting Server

Cisco Meeting Server リリース 3.7 以降

通話詳細レコードガイド

2024 年 9 月 27 日

目次

図:

変更履歴

日付	変更の概要
2023 年 3 月 16 日	バージョン 3.7 で更新
2022 年 8 月 23 日	バージョン 3.6 で更新。
2022 年 4 月 20 日	バージョン 3.5 で更新。
2021 年 12 月 15 日	バージョン 3.4 で更新。
2021 年 9 月 16 日	バージョン 3.3 で更新。
2021 年 5 月 12 日	callLegStart レコードコンテンツのサブタイプからディストリビューション リンクを削除。
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 で更新、X シリーズサーバーへの参照を削除。
2020 年 5 月 05 日	セクション 4.6 の記述を明確化。
2020 年 4 月 8 日	バージョン 2.9 で更新。
2020 年 1 月 7 日	軽微な修正
2019 年 9 月 16 日	callMove および displayName が callLegUpdate レコードにない。
2019 年 8 月 13 日	件名を「..... 2.6 以降」に変更、バージョン 2.7 での変更はなし。
2019 年 7 月 19 日	軽微な修正
2019 年 4 月 23 日	バージョン 2.6 で更新。 callLegStart レコードに canMove、movedCallLeg、movedCallLegCallBridge を追加。
2018 年 12 月 12 日	件名を「..... 2.4 以降」に変更、バージョン 2.5 での変更はなし。
2018 年 9 月 20 日	バージョン 2.4 で更新。 callEnd レコードに endpointRecorded を追加しました。
2017 年 12 月 20 日	バージョン 2.3 で再発行。 追加または変更はありません。
2017 年 6 月 28 日	callLegEnd レコードの mediaUsagePercentages に multiStreamVideo を追加しました。
2017 年 6 月 28 日	CDR レシーバーの作成に関する例を追加しました。
2017 年 5 月 3 日	バージョン 2.2 で更新。 ownerName フィールドを callStart レコードに追加。
2016 年 12 月 20 日	バージョン 2.1 で更新。 追加と変更が表示されます。
2016 年 8 月 03 日	Cisco Meeting Server 2.0 として再ブランド化

1 はじめに

Cisco Meeting Server ソフトウェアは、Cisco Unified Computing Server（UCS）テクノロジに基づく特定のサーバー、または仕様ベースの VM サーバーでホストできます。 本ドキュメントでは、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 以降は X シリーズサーバーをサポートしていません。

Meeting Server は、サーバーに到達する新しい SIP 接続、または通話のアクティブ化または非アクティブ化など、主要な通話関連イベントに対して、内部で通話詳細レコード（CDR）を生成します。

サーバーはこれらのレコードを収集および分析するためにリモートシステムに送信するように設定することができます。 Meeting Server 上にレコードを長期間保存するための規定はなく、Meeting Server 自体の CDR を参照する方法もありません。

CDR システムは Meeting Server API と組み合わせて使用できます。コール ID とコールレッグ ID の値は 2 つのシステム間で一貫しているため、イベントと診断の相互参照が可能になります。

Meeting Server は最大 4 つの CDR 受信者をサポートし、異なる管理ツールまたは同じ管理ツールの複数インスタンスを導入することができます。

注：また、『Cisco Meeting Server API リファレンスガイド』も参照してください。

1.1 本書の使用方法

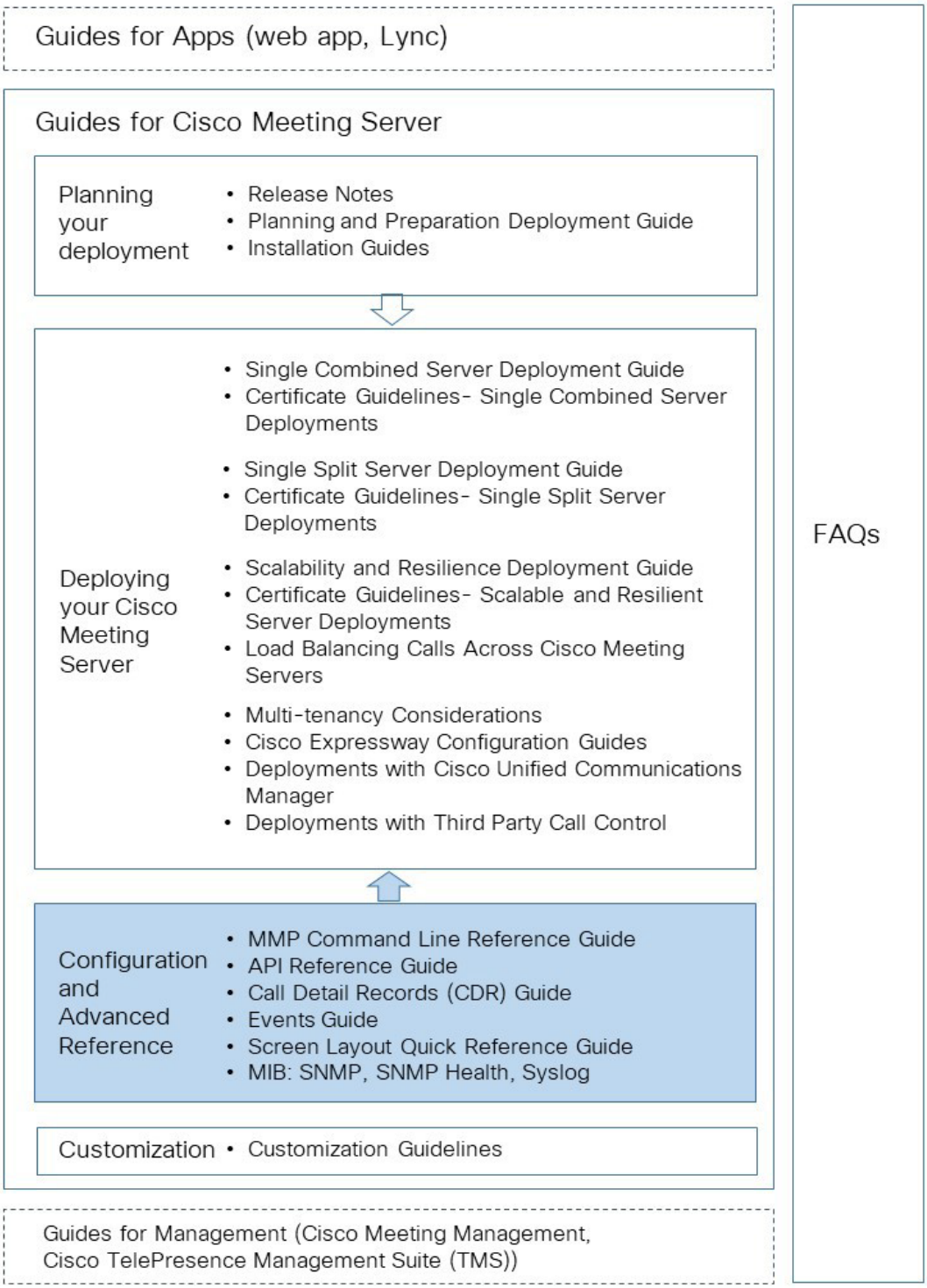
このドキュメントは、下図に示すような多数のリファレンスガイドのうちの 1 つです。

複数の項に分かれていて、前から後ろへと読むことで知識を積み上げることができます。さらに、第 3 章、第 4 章、および第 5 章は、必要に応じて参照できるものとして機能します。 各 CDR レコードタイプとそのフィールドの詳細が記載されています。

このドキュメントでは、「最小限の情報」について説明しています。レコードの XML の性質は、新しい要素が Call Bridge ソフトウェアの新しいバージョンで表示される可能性があることを意味します。そのため、生成されるレコードを解析するとき、新しい要素が現れる可能性に常に備えるべきです。 受信者は、既存のバージョンのドキュメントに記載されていない、追加の新しい要素に対処できなければなりません (同時に、ドキュメントに記載されている内容を、記載されている構造に従って提供することを約束します)。

これらのドキュメントは cisco.com で見つけることができます。

図 1：Cisco Meeting Server のドキュメント、リリース 3.10



2 一般的な仕組み

CDR は Meeting Server により一連の XML ドキュメントとして HTTP または HTTPS 経由で送信されます。 新しいレコードが生成されると、受信システムとの接続が確立され、受信システムはこの接続で 1 つまたは複数のレコードを受信することを予期する必要があります。 Meeting Server が受信者にレコードのグループを正常に送信すると、これらのレコードは Meeting Server によって保存されなくなり、長期保存の責任は受信者のデバイスに移されます。 HTTP または HTTPS 接続が正常に確立され、Meeting Server から XML レコードデータが送信され、受信者が「200 OK」でデータを確認応答した場合、Meeting Server はレコードが受信者に正常に送信されたとみなします。

Call Bridge はキープアライブ接続をサポートしているため、1 つの TCP または TLS 接続でレシーバーに複数の（バッチの）レコードを送信できます。

注：複数の Call Bridge が単一のエンティティとして機能する、スケーラブルでレジリエントな導入では、各 Call Bridge は実行しているコールレッグに CDR を提供します。 各 CDR はコールレッグの coSpace ID を識別します。 こうすることで、通話が複数の Call Bridge でホストされる場合、同じ通話を同じ coSpace ID によって異なる Call Bridge で識別することができます。

2.1 受信者のデバイスの設定

注：CDR 受信者のリストは、個々の Call Bridge に対してローカルに保持されます。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。

ウェブ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 受信者を設定することができます。

2.1.1 ウェブ管理者インターフェイスを使用して CDR 受信者を設定する

CDR の受信者を設定するには、以下の操作を行います。

1. ウェブ管理インターフェイスを開きます。
2. **[設定（Configure）] > [CDR]** 設定に移動します。
3. [CDR 受信者の設定] 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 を参照してください)。

2.1.2 API を使用して CDR 受信者を設定する

以下の API オブジェクトを使用して、Meeting Server に設定できる最大 4 つの CDR レシーバーを有効にします。

- /system/cdrReceivers/
- /system/cdrReceivers/<CDR receiver id>

/system/cdrReceivers ノードで POST を発行し、新しいレシーバーの完全な URI を設定します。 /system/cdrReceivers に対する GET リクエストは、現在設定されているレシーバーを示します。

CDR 受信機が設定されると、その詳細は、/system/cdrReceivers/<CDR receiver id> ノードで GET または PUT を使用することで、読み取りおよび更新できます。 CDR 受信機は、このノードの DELETE によって削除できます。

2.1.3 受信者の URI

Meeting Server 上で設定された受信者 URI は、いくつかの形式のいずれかを取ることができます:

- http://monitoring.example.com/cdr_receiver1
  シンプルな HTTP 接続の場合、リモートホスト「monitoring. example com」上の TCP ポート 80 への URI「/cdr_receiver1」に接続します。
- https://monitoring.example.com/cdr_receiver1
  上と同じ、ただし HTTPS、TCP ポート 443 を使用
- http://monitoring. example.com:8080/cdr_receiver1
  上と同じ、ただし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
- http://monitoring. example.com/cdr_receiver1?system_id=cms1
  上と同じ、ただしパラメータ「system_id」の値として「cms1」を宛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遠端に送信します。受信デバイスがその意味を理解する責任があります。

3 レコードタイプ

CDR は XML で 1 つ以上の「<record>」として送信されます。これらは親「<records>」要素内に存在します。 各レコードには、それが記述するものを識別し、レコード中に期待されるフィールドと属性を決定する関連する「type」値があります。

"<records>" の要素には以下が含まれます。

- そのセッションに固有の 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。特定の実行中の Call Bridge インスタンスのすべてのアクティブな CDR 受信者で同じ値になりますが、その Call Bridge が再起動されると変更されます。 これは、受信側のデバイスが、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たことを判断するために使用されます。
- Call Bridge がクラスタ内にある場合は、Call Bridge GUID が存在します。 これにより、レコードを送信しているクラスター内の Call Bridge を識別します。 これは、システムを再起動しても変わりません。 非クラスター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>" には次の項目が含まれます:

- Meeting Server 上でレコードが生成された時間の値。 このタイムスタンプは RFC 3339 / ISO 8601 形式です。たとえば、「2014-02-28T16:03:25Z (2014 年 2 月 28 日 4:03pm)」などです。 現在、Meeting Server は常にこれらの時間を UTC で提供します。
- 新しいレコードごとに 1 ずつインクリメントする「correlatorIndex」。 注：「session GUID」と「correlatorIndex」の組み合わせにより、すべての受信側でレコードが一意に識別され、受信側は重複するレコードを受信したかどうかを判断できます。

  「correlatorIndex」は、Call Bridge が起動後に生成する最初のレコードの「0」から開始します。 レコードの「correlatorIndex」は、すべての CDR 受信者で同じです。 そのため、Call Bridge が起動した後に設定されたレシーバの場合、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。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かった場合、Call Bridge はレコードを再送信し、受信者は重複したレコードを受信します。

  Meeting Server が内部で生成されたすべてのレコードを保存できないほど、リモートの受信者が一定の時間利用できなかった場合、リモートの受信者にプッシュされたレコードは、「correlatorIndex」にギャップを示します。
- 「recordIndex」は「correlatorIndex」に置き換えられました。 「recordIndex」は現在廃止されており、将来のリリースで削除される可能性があります。

完全性を保つため、以下では「recordIndex」の使用方法を説明します。

「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、受信者が異なると「recordIndex」値が異なる場合があります。

この説明は、受信者が 1 人しかいないことを前提としています。 「<record>」項目内の「recordIndex」は、レコードの順序を決定します。Call Bridge が生成する最初のレコードは「1」から始まり、新しいレコードが送信されるたびに 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 Bridge は、受信者から確認応答（「200 OK」HTTP レスポンス）を受信しなかった CDR を再送信します。 受信者がこのような積極的な応答を送信したが、その応答が Call Bridge で正常に受信されなかった場合、受信者は重複するレコードを受信する可能性があります。「recordIndex」により、受信者はこれを検出し、重複するレコードを処理しないようにすることができます。

Meeting Server が内部で生成されたすべてのレコードを保存することができないほど、リモートの受信者が一定期間利用できなかった場合、リモートの受信者にプッシュされたレコードには、「<record>」タグに「numPreceedingRecordsMissing」という数値が含まれます。 これは、この数のレコード（それが表示されるヘッダーの直前のレコード）が破棄され、もう利用できないことをリモートの受信側に知らせるものです。 CDR 受信者は、「numPreceedingRecordsMissing」の非ゼロ値の存在下でさえ、「recordIndex」シーケンスの不連続を見るべきではありません。

各レコードタイプについて、以下の表 1 で簡単に説明し、詳細については第 4 章で説明があります。

表 1: レコードタイプの概要

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、または最初に coSpace からインスタンス化されたときに生成されます。 このレコードには、通話の ID、名前、および関連付けられている coSpace の ID が含まれます。
callEnd	このレコードは、通話が終了したときに生成され、通常、通話の最後のコールレッグが切断された後に表示されます。 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、および通話の概要値（通話内で同時に機能したコールレッグの最大数など）が含まれます。

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、または Cisco ミーティング アプリケーションのユーザーが coSpace に接続したため、コールレッグが最初に作成され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、通話相手タイプ（SIP 接続または Cisco Meeting App デバイス）、通話相手の「名前」（SIP URI など）、および、意味のある場合は、そのコールレッグが着信か発信かどうかが含まれます。
callLegEnd	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に、コールレッグが終了すると、このレコードが生成されます。 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、および問題のコールレッグのライフタイムに関連する他の要約フィールド（例えば、どの音声とビデオの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、コールレッグに重大な変更が発生した場合に生成されます。例えば、コールレッグが通話に配置される、または（発信ケースの場合）応答され、「接続済み」状態に移行するなどです。
recordingStart	このレコードは、通話のレコーディング開始時に生成されます。 レコードには、開始するレコーディングの ID、レコーディングが保存されるパス（ディレクトリとファイル名）、レコーディングデバイスの URL、レコードされる通話の ID、および通話をレコードしているコールレッグの ID が含まれます。
recordingEnd	このレコードは、通話のレコーディングが終了したときに生成されます。 終了するレコーディングの ID が含まれます。
ストリーミング開始	このレコードは、通話でストリーミングが開始されると生成されます。 このレコードには、開始するストリーミングの ID、ストリーミングの 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。
ストリーミング終了	このレコードは、通話のストリーミングが終了すると生成されます。 終了するストリーミングの ID が含まれます。

4 レコードの詳細

この項では、各レコードタイプの「<record>」タグ内に表示されるパラメータ名と値の詳細を提供します。

4.1 callStart レコードコンテンツ

パラメータ	タイプ	説明
id	ID	開始する通話の ID です。 callStart レコードをカプセル化する「<call>」タグ内の「id」属性として伝えられます。
name	String	通話の名前;通話が通信スペースと関連付けられている場合、通常これは通信スペースの名前になります。
coSpace	ID	この通話に関連する通信スペースの ID です。 この通話が通信スペースに関連付けられていない場合 (たとえば、アドホック通話)、このフィールドは表示されません。
所有者名	String	この通話の所有者の名前。次のいずれかの降順の優先順位で指定されます。 coSpace の「meetingScheduler」フィールドの値、または 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、または空欄（上記のいずれも存在しないことを意味します）。
テナント	ID	マルチテナント導入において、テナントを指定します。
cdrTag	String	coSpace にタグ（API リファレンスを参照）が付けられた場合、これは callStart CDR に表示されます。 タグはオプションで、通話を識別するために使用される最大 100 文字のテキスト文字列です。
callType	coSpace| adHoc| lyncConferencing| forwarding	次のいずれか: *coSpace*：この呼び出しは、coSpace のインスタンス化です *adHoc*：アドホックの多者通話です *lyncConferencing*：この通話は Meeting Server から Lync で主催されている電話会議への接続です *転送* - これは転送された「ゲートウェイ」通話です

パラメータ	タイプ	説明
callCorrelator	ID	この値を使用して、複数のコールブリッジに分散されるが、すべてが同じ coSpace 内の同じ通話またはアドホック通話のいずれかのコールレッグを識別することができます。 注：coSpace 内の通話の場合、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	この通話が含まれる通信スペースでメタデータが設定されている場合、これは true に設定されます。通話がアドホック通話の場合、このフィールドは false になります。 (バージョン 3.2 以降)

注：分散通話で、複数の重複する「callStart」レコードが表示される場合、

- 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 でホストされる 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 コールですが、そうでない場合があります。たとえば、各コールレッグが異なる Call Bridge でホストされる「ポイントツーポイントコール」です。

4.2 callEnd レコードコンテンツ

パラメータ	タイプ	説明
id	ID	終了する通話の ID。以前の「callStart」レコードは、同じ通話に対して生成されています。 callEnd レコードをカプセル化する「<call>」タグ内の「id」属性として伝えられます。
callLegsCompleted	番号 (Number)	この通話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になったコールレッグの最大数。
durationSeconds	番号 (Number)	この通話がアクティブだった時間の長さ (秒)。
endpointRecorded	true|false	Skype または Lync クライアントなどのエンドポイントで通話がレコードされている場合、値は true になります。 (バージョン 2.4 以降)

4.3 callLegStart レコードコンテンツ

パラメータ	タイプ	説明
id	ID	開始するコールレッグの ID です。 callLegStart レコードをカプセル化する「<callLeg>」タグ内の「id」属性として伝えられます。
displayName	String	SIP エンドポイントの場合は「わかりやすい名前」、Cisco ミーティング アプリケーション接続の場合はユーザーの「本名」、ウェブクライアントのゲスト接続の場合はユーザーが入力します。 遠端がわかりやすい名前を提供しない場合、この値は空になります。
ローカルアドレス	String	コールレッグに関連するローカルの宛先（例：発信者が Meeting Server に接続するために使用したもの） この値の解釈は方向によって異なります (下記参照)。 そのため、これは着信コールの宛先アドレス、または発信コールの発信者 ID になります。 注：一部の通話シナリオでは、localAddress が適用されない場合があります。例えば、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 ユーザーへの発信などです。
remoteAddress	String	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（下記参照）。 これは発信コールの宛先 URI または着信コールのソース URI です。
remoteParty	String	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	String	コールレッグにタグが与えられた場合、これが 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。
ゲスト接続	true|false	（オプション）WebRTC アプリ経由で開始されたゲストログインと認識される接続は、ここで True の値を持ちます。
recording	true|false	通話をレコーディングしている接続には、ここで「true」の値があります。
streaming	true|false	通話をストリーミングしている接続には、ここで「true」の値があります。

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」。
subType	lync| avaya| lyncdistribution| webApp	コールレッグタイプのさらなる特殊化。コールレッグが「sip」の場合、可能な値は「lync」、「avaya」、「lyncdistribution」または「webApp」です。
lyncSubType	audioVideo| applicationSharing| instantMessaging	コールレッグのサブタイプが「lync」の場合の、コールレッグのタイプのさらなる特殊化。 *audioVideo*：これは、Call Bridge と Lync の間の音声とビデオの交換に使用される Lync コールレッグです。 *applicationSharing*：これは、Lync と Call Bridge の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 *instantMessaging*：これは、Lync と Call Bridge の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	着信|発信	sip および「acano」コールタイプ: *incoming*：リモート SIP デバイスが Meeting Server への接続を開始した場合、 *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ている場合。
call	ID	このコールレッグのコール ID。 コールレッグの開始時刻でわかっている場合、これが含まれる場合があります。そうでない場合、後の callLegUpdate レコードで通知されます
所有者 ID	ID	このコールレッグに指定するためにリモート管理システムが選択した ID です。この ID は、そのリモートシステムにのみ意味を持ちます。 このコールレッグにそのような所有者 ID が指定されていない場合、このフィールドは存在しません。
sipCallId	String	コールレッグが SIP 接続である場合、コールレッグの開始時刻にわかっていれば、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。

パラメータ	タイプ	説明
groupId	ID	Lync 通話のみ、このパラメータはプレゼンタのビデオコールレッグとコンテンツを共有する場合にプレゼンタの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 操作を実行するときに使用される ID でもあります。 Lync プレゼンテーションでは、CDR に追加のコールレッグ を作成できます。これらは、groupId パラメーターを使用して結合できます。 （callId はもちろん同じですが、同じユーザーによって所有されていない他の Lync コールレッグが通話に存在する可能性があります。Lync「接続」に固有なのは groupId です。） 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るものになりますが、groupID は同じになります。
replacesSipCallId	String	コールレッグが別の SIP コールを置き換える場合、このフィールドは置き換えられた通話の SIP プロトコルヘッダーからの一意の「Call-ID」値（文字列として）を保持します。
移動できる	true|false	このコールレッグを所有する参加者を、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 以降）
movedCallLeg	ID	このコールレッグが参加者の移動の一部として作成された場合、ID は移動元の参加者のコールレッグの GUID です。（バージョン 2.6 以降）
movedCallLegCallBridge	ID	このコールレッグが参加者の移動の一部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した電話会議をホストしている Call Bridge の GUID です。 （バージョン 2.6 以降）
confirmationStatus	required/notRequired/confirmed	- required：confirmation=true が設定され、ユーザー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。 - notRequired：confirmation=true が設定されなかったことを意味します。 - 確認: 参加者の通話への参加を確認するための DTMF シーケンスが入力されたことを意味します。

4.4 callLegEnd レコードコンテンツ

パラメータ	タイプ	説明
id	ID	終了するコールレッグの ID。 callLegEnd レコードをカプセル化する「<callLeg>」タグ内の「id」属性として伝えられます。
cdrTag	String	コールレッグにタグが与えられた場合、これが 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。
理由	String	コールレッグの終了理由（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の参加者によって開始されたことを示します *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
理由詳細	String	リモート終了によりコールレッグが終了した場合、パラメータは、通話が WebRTC 通話か SIP コールかを示します。
encryptedMedia unencryptedMedia		これらの値の一方または両方が存在し、コールレッグの存続期間中の暗号化または非暗号化メディアの有無（true または false の値に基づく）を示すために存在します。 表示されない場合、そのメディアタイプはこのコールレッグに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であった時間の長さ（秒）。
activatedDuration	番号（Number）	このコールレッグがアクティブであった時間の長さ（秒）。

<table>
<tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr>
<tr><td>mediaUsage Percentages</td><td></td><td>このコールレッグの有効期間中に、異なるタイプのメディアがアクティブだった割合に関する情報。メディアタイプは次のとおりです。

mainVideoViewer：ユーザーはメインビデオを受信

mainVideoContributor：ユーザーはメインのビデオに寄与

presentationViewer：ユーザーはプレゼンテーション情報を受信

presentationContributor：ユーザーはプレゼンテーションを共有

マルチストリームビデオ- マルチストリームビデオがアクティブであった時間の割合です。</td></tr>
<tr><td>マルチストリームビデオ</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>番号（Number）</td><td>マルチスクリーン メイン ビデオの最大数
このコールレッグの有効期間中にアクティブな画面。
例えば、デュアルビデオの場合は2になります。</td></tr></table></td></tr>
<tr><td>アラーム</td><td colspan="2">コールレッグがその存続期間中にアラーム状態が発生した場合、これらの要素が 1 つ以上存在します。

<table><tr><th>名前</th><th>タイプ</th><th>説明</th></tr><tr><td>タイプ</td><td>packetLoss|
RoundTripTime
ルでパケットロスが</td><td>packetLossexcessiveJitter|
このコールレッグにおいて、ローカ
観察されたか、遠端から報告

excessiveJitter：このコールレッグにおいて、ローカルで高いジッター値が観察されたか、遠端から報告

highRoundTripTime：このコールレッグにおいて、Meeting Server とリモート側との間の長い往復時間が検出</td></tr><tr><td>継続時間の割合</td><td>番号（Number）</td><td>この値は、
警告状態が発生した通話時間のパーセンテージを示します。</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
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 がリモート参加者から受信した音声）および送信した音声（「txAudio」）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ます:
<table><tr><th>パラメータ</th><th>タイプ</th><th>説明</th></tr><tr><td>コーデック</td><td>次のいずれか:
g711u
g711a
g722
g728
g729
g722_1
g722_1c
aac
speexNb
speexWb
speexUwb
isacWb
opus</td><td>使用された音声コーデック：
g711u - G.711 mu law
g711a - G.711 a law
g722 - G.722
g728 - G.728
g729 - G.729
g722_1 - G.722.1
g722_1c - G.722.1C (G.722.1 付録 C)
aac - AAC
speexNb - Speex ナローバンド
speexWb - Speex 広帯域
speexUwb - Speex ウルトラワイドバンド
isacWb - iSAC (internet Speech Audio Codec) ワイドバンド
isacSwb - iSAC（インターネットスピーチ音声コーデック）スーパーワイドバンド</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
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 がリモート参加者から受信したビデオ）および送信したビデオ（「txVideo」）の詳細を提供します。 rxVideo および txVideo セクションには次の要素が含まれます:
<table>
<tr><th>名前</th><th>タイプ</th><th>説明</th></tr>
<tr><td>コーデック</td><td>次のいずれか:
使用
h263
h263+
h264
h264Lync
vp8
rtVideo</td><td>ビデオコーデックが
h261 - H.261
h263 - H.263
h263+ - H.263+
h264 - H.264
h264Lync - H.264 SVC Lync 用
vp8 - VP8
rtVideo - RTVideo</td></tr>
<tr><td>maxSizeWidth</td><td>番号（Number）</td><td>使用される最大ビデオ解像度の幅</td></tr>
<tr><td>maxSizeHeight</td><td>番号（Number）</td><td>使用される最大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は、その方向にビデオが送信されていません。</td></tr>
<tr><td>所有者 ID</td><td>ID</td><td>管理しているリモートシステムがこのコールレッグに指定するために選択した ID です。この ID は、そのリモートシステムにのみ意味を持ちます。</td></tr>
<tr><td>sipCallId</td><td>String</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固有の「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、ユーザー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。
- notRequired：confirmation=true が設定されなかったことを意味します。
- confirmed：参加者の通話への参加を確認するための DTMF シーケンスが入力されたことを意味します。
- rejected：着信拒否の DTMF シーケンスが入力されたことを意味します。 ミーティング管理は参加者へのリダイヤルを停止します。</td></tr>
</table>

4.5 callLegUpdate レコードコンテンツ

パラメータ	タイプ	説明
id	ID	更新されているコールレッグの ID。 callLegUpdate レコードをカプセル化する「<callLeg>」タグ内の「id」属性として伝えられます。
cdrTag	String	コールレッグにタグが与えられた場合、これが CDR 表示されます。 タグは任意の、コールレッグの識別 使用される最大 100 文字のテキスト文字列です。
state	接続済みまたは値が指定されていません	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。
無効化されました	true|false	コールレッグが現在無効かどうかを示します
リモートアドレス	String	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 (下記参照)。 そのため、これは発信コールの宛先 URI、または着信コールのソース URI です。
call IVR		このコールレッグのコール ID、または現在コールレッグが IVR の場合は（空）「IVR」表示。
所有者 ID	ID	管理しているリモートシステムがこのコールレッグに指定するために選択した ID です。この ID は、そのリモートシステムにのみ意味を持ちます。
sipCallId	String	コールレッグが SIP 接続である場合、コールレッグの開始時にわかっていれば、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。
groupID	ID	Lync 通話の場合のみ、このパラメータはプレゼンタのビデオコールレッグ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	String	SIP エンドポイントの場合は「わかりやすい名前」、Cisco ミーティング アプリケーション接続の場合はユーザーの「本名」、ウェブクライアントのゲスト接続の場合はユーザーが入力します。 遠端がわかりやすい名前を提供しない場合、この値は空になります。
移動できる	true|false	このコールレッグを所有する参加者を、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、ユーザー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。 - notRequired：confirmation=true が設定されなかったことを意味します。 - 確認：参加者の通話への参加を確認するための 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が変更されたときに Meeting Server から送信されます。 たとえば、CDR 受信者は、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「所有者 ID」を変更した場合、そのような更新レコードを見ることを期待します。

4.6 recordingStart レコードコンテンツ

パラメータ	タイプ	説明
id	ID	開始しているレコーディングの ID です。 recordingStart レコードをカプセル化する「<recording>」タグ内の「id」属性として伝えられます。
パス	String	レコーディングのディレクトリとファイル名を保持する文字列です。（内蔵 XMPP レコーダーにのみ適用されます。）
recorderUri	String	レコーディングデバイスが SIP レコーダーの場合、その URI。（外部のサードパーティ SIP レコーダーにのみ適用されます。）
call	ID	レコードされている通話の ID。
callLeg	ID	通話をレコーディングしているコールレッグの ID。

4.7 recordingEnd レコードコンテンツ

パラメータ	タイプ	説明
id	ID	終了しているレコーディングの ID。 recordingEnd レコードをカプセル化する「<recording>」タグ内の「id」属性として伝えられます。

4.8 streamingStart レコードコンテンツ

パラメータ	タイプ	説明
id	ID	開始するストリーミングの ID です。 streamingStart レコードをカプセル化する「<streaming>」タグ内の「id」属性として伝えられます。
streamerUri	URL	ストリーミングデバイスの URL。（内部 SIP ストリーマ コンポーネントに適用されます。）
call	ID	ストリーミングされている通話の ID です。
callLeg	ID	通話をストリーミングしているコールレッグの ID。

4.9 streamingEnd レコードコンテンツ

パラメータ	タイプ	説明
id	ID	終了するストリーミングの ID です。 streamingEnd レコードをカプセル化する「<streaming>」タグ内の「id」属性として伝えられます。

5 コールレッグ終了レコードの理由コード

コールレッグ終了のレコードには、理由コード（「<reason>」タグ内）と、そのコールレッグの切断を Meeting Server またはリモート参加者のどちらが選択したかを示す個別の表示（「<remoteTeardown>」セクションで、「true または false」）。

切断を引き起こした側は切断理由によって判断できますが、別のリモートまたはローカル終了表示は、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、どちらの側が切断を開始したかの基本的な知識を得ることができるという点で、将来への備えができるようになります。

[<reason>] に指定可能な値コードは次のとおりです。

理由（Reason）	説明
apiInitiatedTeardown	コールレッグは、切断を行うための API リクエストへのレスポンスとして、Meeting Server により切断されました
callDeactivated	コールレッグの一部であった通話が無効化されたため、Meeting Server により切断されました。また、コールレッグの無効化アクションが「切断」に設定されました。 詳細については、API リファレンスを参照してください
callEnded	コールレッグが含まれていた通話が終了したため、コールレッグは Meeting Server によって切断されました（例えば、それを破棄する API コマンドへの応答とした場合など）
callMoved	コールレッグは、Call Bridge リソースの使用における効率を改善するために移動されました
clientInitiatedTeardown	コールレッグは、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 によって切断されました
確認タイムアウト	リモート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。 「通話に招待されました。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すが、相手方が 1 分以内にキーを押さなかったため、この理由コードによってコールレッグが切断されました。
dnsFailure	リモート宛先のホスト名解決の失敗。例えば、リモートシステムとの接続を確立するプロセスの一部として
暗号化必須	コールレッグが切断されました。暗号化されたメディアの要件が満たされないためです。
エラー	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、通話中に SIP エンドポイントの電源が切れるか、クラッシュ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てください。
incorrectPasscode	再試行の最大数に達した後、ユーザーは通話または参加する coSpace に正確な PIN を提供しなかった

理由（Reason）	説明
ivrTimeout	コールレッグは IVR に接続されたが、必要な時間内にコールに移行できなかった
ivrUnknownCall	最大再試行回数に達した後、IVR でユーザーが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終了
participantLimitReached	新しい参加者を追加しようとしましたが、通話で許可されている最大数を超えています。
remoteBusy	リモートの参加者が、通話中で接続を受け入れることができないという信号を送ったため、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側により拒否されました
remoteTeardown	コールレッグがリモート側により切断されました。 リモート終了中、通話が WebRTC 通話か SIP コールかを示すパラメータ **reasonDetails**。
ringingTimeout	コールレッグがリモートデバイスに到達し、リモートデバイスが鳴り、要求された時間内に応答がありませんでした。
tenantParticipantLimitReached	新しい参加者を追加しようとしましたが、所有するテナントに許可されている最大数を超えています。
タイムアウト	コールレッグは Meeting Server により切断されました。プロトコルタイムアウト、例えば SIP セッションタイムアウト、または SIP リクエストに対する必須のレスポンスの欠如などのためです。
unknownDestination	コールレッグは、有効な coSpace またはユーザーに解決されない宛先への着信接続です。

6 トラフィックフローの例

以下のトレースは、典型的なトラフィックフローの例を示します。 2 つの SIP クライアントがミーティングに接続し、そのうちの一つがミーティングを終了し、もう一つの SIP コールが切断されることについて説明します。 この例の XML は、読みやすくするために書式設定されています。

イベント投稿 #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>
```

イベント投稿 #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>接続済み</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>
```

イベント投稿 #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>
      <方向>受信</方向>
      <groupId>3420c93f-f33c-4c9e-be95-d0d1bfb207f0</groupId>
      <sipCallId>a939937c-8b5e-4376-92de-97635983d7ef</sipCallId>
    </callLeg>
  </record>
</records>
```

イベント投稿 #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>接続済み</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>
```

イベントポスト #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>
```

イベント投稿 #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>
        <パケット統計>
          <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 受信者を作成するためのサンプルスクリプト

次の Python スクリプトは、CDR 受信者を作成する方法を示します。 例は説明のみを目的としており、Cisco はコードの使用に関していかなるサポートも保証も提供しません。 Cisco はコードの著作権を保有しています。

```
#!/usr/bin/python

## Example CDR receiver code for Cisco Meeting Server
## Copyright - Cisco Systems (2013-2017)
## このコードにはサポート、保証、責任はありません



import BaseHTTPServer
import sys
import getopt
import ssl

class RequestHandler(BaseHTTPServer.BaseHTTPRequestHandler):
     handler = BaseHTTPServer.BaseHTTPRequestHandler
     handler.protocol_version = 'HTTP/1.1'
     print "使用しているプロトコルバージョン:",
     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
    例外:
        print 'usage: cdr_receiver.py -p <port> [-c <certfile path>] [-
        k <keyfile path>]'
        sys.exit(2)
    サーバアドレス = ('', 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、すべてユーザ側の責任になります。

対象製品のソフトウェア ライセンスおよび限定保証は、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。 シスコおよびこれら各社は、商品性の保証、特定目的への準拠の保証、および権利を侵害しないことに関する保証、あるいは取引過程、使用、取引慣行によって発生する保証をはじめとする、明示されたまたは黙示された一切の保証の責任を負わないものとします。

いかなる場合においても、シスコシステムズ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、あらゆる可能性がシスコシステムズ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。

★定型★このマニュアルで使用している 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。 ★定型★マニュアル内の例、コマンド出力、ネットワーク トポロジ図、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。

シスコは世界各国 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、シスコまたはその関連会社の米国およびその他の国における商標または登録商標です。 シスコの商標の一覧については、www.cisco.com/jp/go/trademarks をご覧ください。 記載されているサードパーティの商標は、それぞれの所有者に帰属します。 「パートナー」という用語の使用はシスコ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)