Cisco Meeting Server

Cisco Meeting Server リリース 3.13

証明書のガイドライン

2026 年 5 月 4 日

目次

変更履歴

日付	変更の概要
2026 年 5 月 4 日	Cisco Meeting Server バージョン 3.13 のドキュメントを更新しました。 第 2 章：導入に必要な証明書：TLS 証明書ポリシーの更新と Cisco Meeting Server 3.13 互換性についての注記を追加しました。

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 以降は X シリーズサーバーをサポートしていません。

Cisco Meeting Server は安全性が高く、サーバー上で実行されるほとんどのサービスおよびアプリケーションは、通信に TLS 暗号プロトコルを使用します。TLS により、通信する当事者は、相手を認証するために X.509 証明書と公開キーを交換し、当事者間で送信されるデータを暗号化するための暗号化アルゴリズムを交換できます。

この証明書ガイドラインのドキュメントでは、スケーラブルで復元力のある導入のための証明書を作成してインストールする方法について説明しています。単一の分割または単一の統合など、その他の導入で違いがある場合は、これらの違いがハイライトされます。

メモ: 以降、本書では Cisco Meeting Serverソフトウェアを指すことを「Meeting Server」と呼びます。

1.1 このガイドの使い方

この章の残りの部分では、Meeting Server の導入全体に証明書を適用するために理解する必要のある概念について説明します。PKI、証明書、および信頼ストアにすでに精通している場合は、これをスキップしてください。

第 2 章 では、スケーラブルで復元力のあるサーバモデル内で証明書が必要な場所と、必要な証明書のタイプについて詳しく説明します。

第 3 章 では証明書の作成方法について説明しています。

第 4 章 では、Meeting Server への証明書のインストールについて説明しています。

第 5 章 では、典型的な証明書関連の問題のトラブルシューティング情報を提供しています。

第 6 章 では、自己署名証明書を簡単に作成する方法について説明しています。

OpenSSL を使用する場合、Meeting Server の pki コマンドではなく、OpenSSL の使用について説明します。

付録 B では、証明書ファイルと秘密鍵で許可されるファイル拡張子の概要を説明しています。

付録 C は MMP pki コマンドの一覧です。

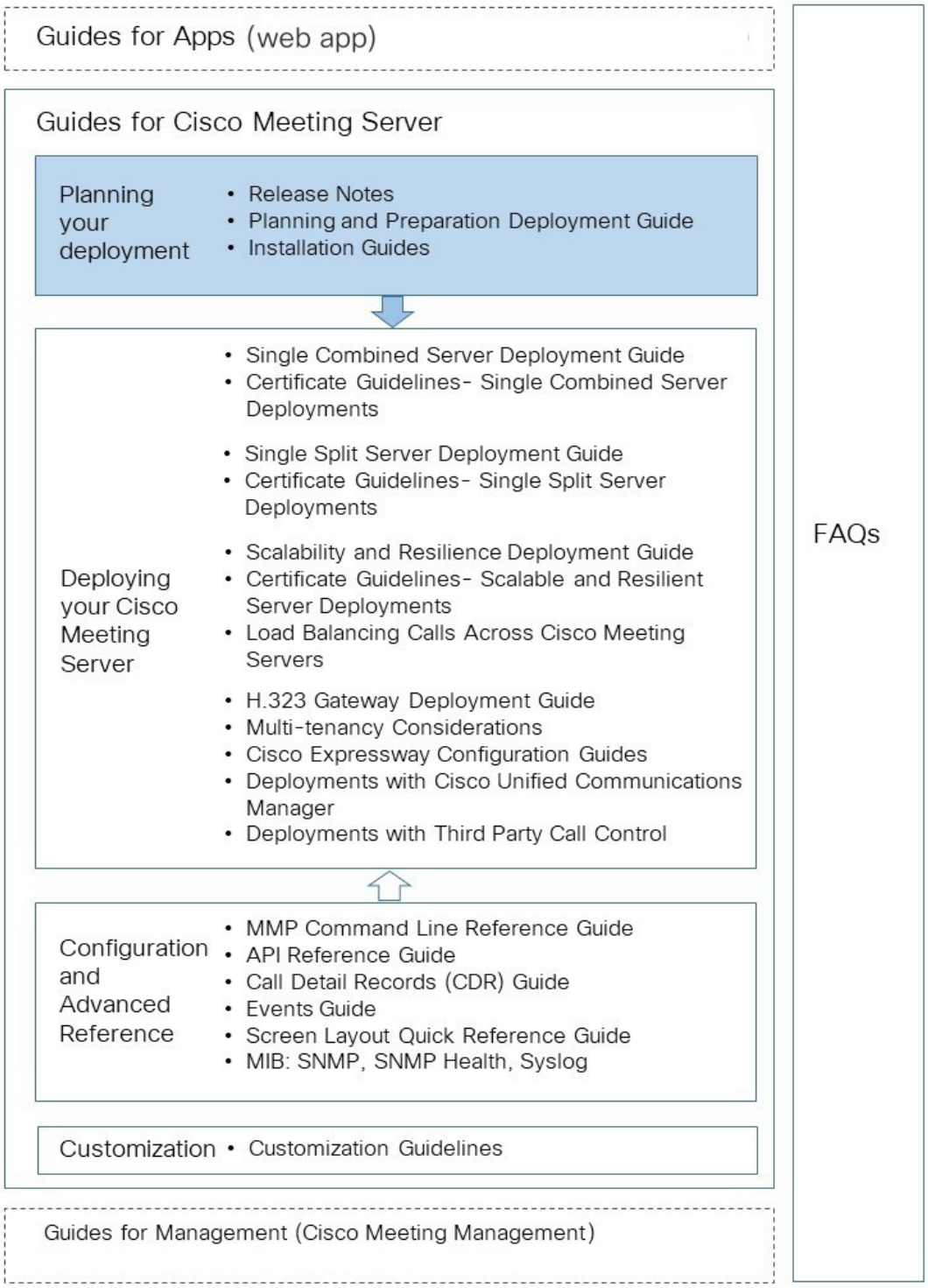
付録 D では、Web Bridge 3 を展開して Cisco Meeting Server ウェブアプリを使用する場合の、証明書と構成情報を提供します。

注：WebRTC 用 Cisco Meeting App（Web Bridge 2）は Cisco Meeting Server バージョン 3.0 から削除されました。ソフトウェアバージョン 3.0 以降を使用している場合、WebRTC 用 Cisco Meeting App の代わりに、Cisco Meeting Server web app を使用する必要があります。これを行うには、Web Bridge 3 を導入する必要があります。Web Bridge 3 の導入と設定の詳細については、3.0 以降の導入ガイド を参照してください。

重要情報: バージョン 3.0 から、XMPP サーバー、ロードバランサー、SIP Edge および H.323 ゲートウェイのコンポーネントが Cisco ミーティングサーバーソフトウェアから削除されました。さらに、新しい SIP レコーダーおよびストリーマーコンポーネントは、サーバーソフトウェアから削除された以前の XMPP クライアントバージョンのレコーダーおよびストリーマーに置き換わるものです。TURN サーバーはバージョン 3.0 ソフトウェアのままで、ブラウザベースの Cisco Meeting Server web app を Meeting Server 電話会議に接続するために使用される場合があります。ネイティブおよびブラウザベースの Cisco ミーティングアプリクライアントはバージョン 3.0 ではサポートされていません。

このガイドは、Meeting Server 用のドキュメントセット（1 に示されています）の一部です。

図 1：Meeting Server に関するガイドの概要



これらのドキュメントは、cisco.com でご確認いただけます。

1.2 PKI の概要

公開キー基盤 (PKI) は、通信をセキュリティ保護し、通信する当事者のアイデンティティを検証するメカニズムを提供します。通信は暗号化により安全にされ、アイデンティティは公開/秘密鍵ペアおよびデジタル証明書を使用して検証されます。

1.2.1 公開/秘密鍵ペア

公開キーと秘密キーのペアは、数学的に関連した 2 つの一意に関連する暗号キーで構成されます。公開キーで暗号化されたものは何でも、それに対応する秘密キー（秘密にしておく必要があります）によってのみ復号化でき、その逆も同様です。

1.2.2 証明書

証明書は公開鍵のラッパーであり、公開鍵の所有者に関する情報を提供します。これには通常、証明書が発行されるエンティティの名前、所有者の連絡先詳細、有効日（証明書が有効な期間）、および発行者（証明書を発行した機関）が含まれます。証明書は、所有者が本人であることを確認できる信頼できる機関によって署名されている必要があります。認証局 (CA) は、ネットワーク上の個人、組織、およびコンピュータの身元を証明する信頼できる機関です。

エンティティが証明書を要求する場合、まず公開/秘密鍵のペアを生成します。次に、エンティティの公開鍵とエンティティを識別する情報を含む証明書署名リクエスト (.csr) ファイルを作成します (表 1) を参照してください。エンティティは秘密鍵を使用して .csr ファイルに署名し、処理のために .csr ファイルを CA に送信します。必要な検証のレベルに応じて、エンティティは Verisign などの公開 CA に .csr ファイルを送信するか、または内部 CA、たとえば、Active Directory 証明書サービスの役割がインストールされた Active Directory サーバを使用する場合があります。

CA は .csr ファイルと公開キーを使用して、エンティティのアイデンティティを確認します。検証に成功すると、CA はデジタル ID 証明書をエンティティに発行します。これは、証明書に記載されているエンティティが公開鍵と秘密鍵のセットの所有者であることを証明するものです。デジタル ID 証明書は、公開鍵が本当に秘密鍵の所有者に属していることを、ネットワーク上の他のエンティティに高レベルの保証を与えるために、エンティティによって使用されます。

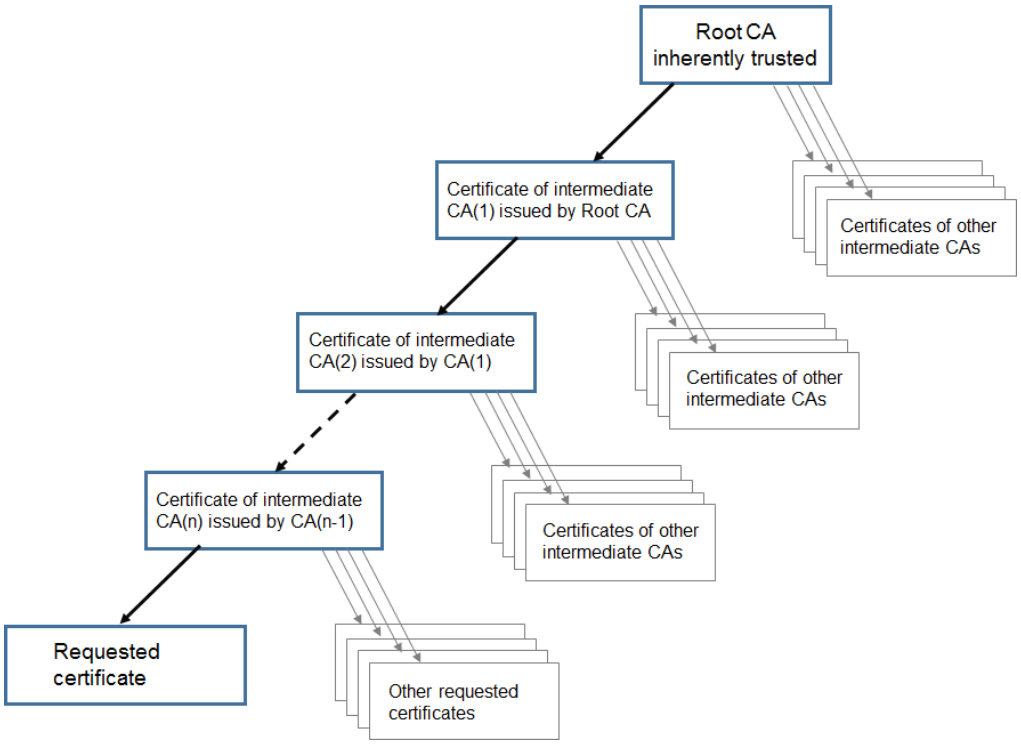
表 1: .csr ファイルの情報

情報	説明
共通名 (CN)	これは、保護する完全修飾ドメイン名です（「www.example.com」など）。
組織名または会社名 (O)	通常は会社の正式な法人名です。これには、Ltd.、Inc.、または Corp. などのサフィックスを含める必要があります。
組織単位または部門名 (OU)	例えば、サポート、IT、エンジニアリング、財務など。
場所 (L)	市区町村を指定します。たとえば、ロンドン、ボストン、ミラノ、ベルリンなどです。
地方、地域、郡または州（ST）	たとえば、バッキンガムシャー、ニュージャージー州です。短縮形を使用しないでください。
国 (C)	組織が所在する国を表す 2 文字の ISO コードです。US、GB、FR など。
メールアドレス	組織に連絡するためのメールアドレスです。通常は証明書管理者または IT 部門のメールアドレスです。
サブジェクト代替名 (subjectAltName)	X509 バージョン 3（RFC 2459）から、SSL 証明書は証明書が一致する必要がある複数の名前を指定することが許可されています。subjectAltName（SAN）には、メールアドレス、IP アドレス、通常の DNS ホスト名などを含めることができます。

1.2.3 信頼チェーン

あるエンティティが別のエンティティから認証用の証明書を提供するよう要求された場合、そのエンティティは、自身の証明書に加えて、要求する側が信頼する Certificate Authority へのリンクを確立する他の証明書シリーズ（通常、ルート Certificate Authority と呼ばれます）を提示する必要があります。エンティティの証明書をルート CA にリンクする、この証明書の階層は、「トラストのチェーン」と呼ばれます。ルート CA が別の Certificate Authority（中間 CA として知られています）の証明書に署名し、中間 CA がエンティティの証明書に署名することはよくあることです。その場合、エンティティはそれ自体の証明書と、ルート CA によって発行されたこの中間 CA の証明書の両方を提示する必要があります。エンティティが、信頼できるルート CA へのリンクを確立せずに、自身の証明書のみを提示した場合、チャレンジする当事者は提示された証明書を信頼しません。エンティティの証明書をルート CA にリンクする一連の証明書は、中間 CA に対して発行されるため、「中間証明書」と呼ばれます。

図 2: 証明書の信頼チェーン



接続しているデバイスが信頼のチェーンを確認できるようにするために、すべての証明書には「発行先」と「発行者」の 2 つのフィールドが含まれます。中間 CA はこれらの 2 つのフィールドに異なる情報を表示し、必要に応じて信頼を確立するためにどこを継続して確認するかを接続デバイスに示します。ルート CA 証明書は「発行先」および「発行者」自体であるため、これ以上のチェックは不可能です。

たとえば、エンティティ A（ウェブサーバー www.example.com）がエンティティ B（ウェブクライアント）によって認証を要求された場合、エンティティ A は証明書と証明書チェーンをエンティティ B に提示する必要があります。

図 3: エンティティ A の証明書チェーン

証明書 1 - 発行先: example.com;発行者: 中間 CA 1
証明書 2 - 発行先: 中間 CA 1;発行者: 中間 CA 2
証明書 3 - 発行先: 中間 CA 2;発行者: ルート CA

エンティティ B が信頼ストアにルート CA の証明書を格納している場合、エンティティ A とエンティティ B の間でセキュアな接続を確立できます。エンティティ B は、エンティティ A の公開キーを使用してメッセージを暗号化し、エンティティ A に送信できます。秘密キーにはエンティティ A のみがアクセスできるため、エンティティ A のみがメッセージを復号化できます。

注意: このプロセスは「証明書のチェーン」と呼ばれ、中間 CA 証明書は「チェーンされた証明書」と呼ばれることがあります。

1.2.4 証明書バンドル

証明書バンドルは、ルート CA の証明書およびチェーン内のすべての中間証明書のコピーを保持する単一のファイル (拡張子 .pem、.cer または.crt) です。証明書は、証明書バンドルでルート CA の証明書が最後になるように順に配置する必要があります。ウェブブラウザなどの外部クライアントは、安全な接続を設定する際に、ウェブブリッジ 3 が証明書と証明書バンドルを提示することを要求します。Call Bridge が SIP ピアへの TLS トランクを確立する場合、Call Bridge は証明書と証明書バンドルを SIP エンドポイントに提示する必要があります。

メモ帳などのプレーンテキストエディタを使用して、証明書バンドルを作成することができます。-----BEGIN CERTIFICATE----- および -----END CERTIFICATE----- タグを含むすべての文字をドキュメントに挿入する必要があります。証明書の間にはスペースを入れないでください。たとえば、証明書 1 -----の-----END CERTIFICATE および証明書 2 の -----BEGIN CERTIFICATE----- の間にスペースや余分な行は入れないでください。証明書 1 は、-----END CERTIFICATE----- で終了し、その次の行に証明書 2 の -----BEGIN CERTIFICATE-----期間挿入されます。ファイルの最後には余分な行が 1 行必要です。ファイルを .pem、.cer、または .crt の拡張子で保存します。

注：Web Bridge 3 では、すべての証明書定義で証明書のバンドル（つまり、フルチェーンファイル）を使用する必要があります。これはウェブ ブリッジ 2 の証明書の実装とは異なります。

1.2.5 トラストストア

ウェブブラウザおよびその他のクライアントは、信頼する署名機関のリストを保持します。したがって、「信頼チェーン」によって、信頼できるサーバーを決定します。これらの信頼できる CA は、クライアントの「信頼ストア」に保持されます。信頼できる CA が失効リストを発行すると、クライアントはトラストストアを更新し、ストアから失効リスト中のエンティティを削除します。

接続するクライアント (またはデバイス) が証明書を信頼する場合、クライアントは証明書の CA がクライアントの信頼ストアに保持されているかどうかを確認します。証明書が信頼できる CA によって発行されたものではない場合、接続するクライアントは、発行する CA の証明書が信頼できる CA によって発行されたものかどうかを確認します。信頼できる CA が見つかるまで、または信頼できる CA が見つからなくなるまで、これを繰り返します。信頼できる CA が見つかった場合、クライアントとサーバ間で安全な接続が確立されます。信頼できる CA が見つからない場合、接続しているクライアントは通常エラーメッセージを表示します。

バージョン 3.0 ではウェブ ブリッジ 3 の C2W 接続証明書に変更が加えられたため、2.9 の場合のように信頼ストアはルート証明書を必要としなくなりました。これにより、管理者は信頼できる証明書をより柔軟に選択することができます。たとえば、会社の内部ポリシーにより、C2W 接続を保護するために公開証明書を使用する必要がある場合、パブリック CA によって署名されたすべての証明書を信頼するのではなく、相手側で使用されるクライアントまたはサーバの C2W 証明書のみを信頼できるようになりました。これは証明書のピン留めと呼ばれます。

2 展開に必要な証明書

この章では、安全な接続を確立するために証明書が必要な場所と、必要な証明書のタイプについて説明します。

Cisco Meeting Server 3.0 ソフトウェアから削除されたコンポーネントについてのメモ：次のコンポーネントはソフトウェアバージョン 3.0 から削除されました：Web Bridge 2、H.323 ゲートウェイ、SIP Edge、XMPP サーバー、ロードバランサ。

注記： Meeting Server 3.13 では、クライアントの証明書に対する拡張鍵使用法（EKU）検証は強制されなくなります。ミーティングサーバ内のすべてのサービスとコンポーネントは、既存の証明書を引き続き使用して動作するため、現在の展開環境における証明書の変更は一切必要ありません。
ただし、データベースなど、クライアント認証拡張使用権限（EKU）付きの証明書を必要とするコンポーネントについては、内部またはプライベートの認証局（CA）を通じてこれらの証明書を生成するか、クライアント認証 EKU なしで証明書を発行する外部の認証局を使用することをお勧めします。

2.1 パブリック CA または内部 CA 署名付き証明書

外部デバイスのインターフェイスとなる Meeting Server 上のアプリケーションは、外部デバイスから信頼され、パブリック CA の署名入りの証明書を必要とします。Meeting Server 内部でインターフェイスとなるアプリケーションは、パブリック CA または内部 CA により署名された証明書を使用できます。内部 CA 署名付き証明書は、Active Directory 証明書サービスの役割がインストールされた Active Directory サーバーなど、ローカルまたは組織の Certificate Authority によって生成されます。セクション 3を参照してください。

パブリック CA の署名付き証明書を必要とするアプリケーションを 表 2 に示します。内部 CA 署名証明書のみを必要とするアプリケーションを 表 3 に示します。

ミーティングサーバでのワイルドカード証明書の使用に関する情報およびその他の証明書に関する FAQ は、リンクに移動してください。

注：WebRTC 通話を使用する導入では、Call Bridge 証明書に、digitalSignature ビットが設定された KeyUsage 拡張機能が含まれている必要があります。それ以外の場合は、Call Bridge とウェブアプリクライアント間のメディアの DTLS ネゴシエーションが失敗する場合があります。この KeyUsage は通常、クライアントおよび/またはサーバー証明書に CA 設定を使用する場合に含まれます。

表 2: パブリック CA の署名付き証明書 (スケーラブルで復元力のあるサーバモデル)

パブリック CA 署名付き証明書を要求するアプリケーション	証明書の使用	理由
Web Bridge 3 ウェブアプリが使用される場合のみ		ウェブブラウザはパブリック CA の署名付き証明書を要求します
Call Bridge（Meeting Server が Lync の直接フェデレーション用のパブリックネットワークに接続されている場合のみ)	callbridge - TLS ウェブサーバー認証、TLS ウェブクライアント認証	直接フェデレーションを行う場合、Lync Edge サーバは Call Bridge からのパブリック CA 署名付き証明書を必要とします。
TURN サーバー	TURN - TLS ウェブサーバ認証	TURN サーバーで TLS を設定する場合、WebRTC クライアントが接続を信頼できるように、TURN サーバーは Web Bridge 用に作成されたものと同様の証明書/キーのペアを要求します。証明書は、ウェブブリッジの証明書に使用されたものと同じ認証局によって署名されている必要があります。

表 3: 内部 CA の署名付き証明書 (スケーラブルで復元力のあるサーバモデル)

内部 CA 署名付き証明書を使用できるアプリケーション	証明書の使用	理由
Web 管理（Web Admin)	WebAdmin - TLS ウェブサーバ認証	ミーティングサーバはウェブ管理のインターフェイスへの HTTPS 接続のみを許可します。そのため、ウェブ管理には証明書が必要です。 注：Meeting Server API はウェブ管理のインターフェイスを通じてルーティングされるため、ウェブ管理インターフェイスではなく API を通じて Call Bridge を設定する場合でも、証明書が必要です。 さらに、クラスタ内の Call Bridge は、ウェブ管理を介して、HTTPS 経由で相互に接続します。バージョン 2.4 から、クラスタ内の Call Bridge を確認するための Call Bridge 信頼ストアを使用することで、Call Bridge クラスタのセキュリティを向上させることができます。セクション 4.9 を参照してください。

内部 CA 署名付き証明書を使用できるアプリケーション	証明書の使用	理由
Call Bridge	TLS ウェブクライアント認証および TLS ウェブサーバ認証	Web Bridge 3 c2w 接続は、Call Bridge からの証明書を信頼する必要があります。 また、Active Directory サーバは Call Bridge からの証明書を信頼する必要があります。 さらに、展開に TLS を使用する SIP トランクがある場合、Call Bridge には SIP 通話制御デバイスとの相互認証用の証明書が必要です。
データベースクライアント	データベース - TLS ウェブ クライアント認証	データベースクラスタリングは、機密性と認証の両方のために公開/秘密鍵暗号化を使用します。データベースをホストする各サーバは、同じ CA によって署名された証明書のセットを必要とします。次を参照してください。セクション 3.1.4。
データベースサーバ	データベース - TLS ウェブサーバ認証	
レコーダ	レコーダー - TLS ウェブサーバ認証	ミーティング サーバでレコーダーを有効にすると、Call Bridge はレコーダーからの署名済み証明書を必要とし、レコーダーは Call Bridge からの証明書を要求し、また信頼する必要があります。
ストリーマー	Streamer - TLS ウェブサーバ認証	ミーティング サーバでストリーマを有効にすると、Call Bridge はストリーマからの署名付き証明書を必要とし、ストリーマは Call Bridge からの証明書を必要とし、信頼する必要があります。
c2w	web ブリッジ 3 c2w - TLS ウェブサーバ認証	C2W 証明書は、Call Bridge とウェブブリッジ 3 の間の接続に使用されます。

メモ: 証明書が CA によって署名されている場合、証明書のプロパティを指定するのに役立つように、証明書の使用状況が記録されます。ExtendedKeyUsages が有効になっていない場合、証明書はすべての用途に対して有効です。いずれかが有効になっている場合、証明書の ExtendedKeyUsages には少なくとも表で定義されている使用箇所が含まれている必要があります。

注： Call Bridge 証明書の KeyUsage ビットには、KeyUsage デジタル署名を含める必要があります。そうしないと、Call Bridge とウェブアプリクライアント間のメディアの DTLS ネゴシエーションが失敗する場合があります。この KeyUsage は通常、クライアントおよび/またはサーバー証明書に CA 設定を使用する場合に含まれます。

2.2 Meeting Server が証明書の拡張子を確認する方法

Meeting Server は以下で説明するように証明書の拡張子を確認します。

クライアント接続の場合:

- ExtendedKeyUsage 拡張機能が存在する場合、TLS ウェブクライアント認証ビットが設定されている必要があります
- KeyUsage 拡張機能が存在する場合、デジタル署名と keyAgreement ビットが少なくとも 1 つ設定されている必要があります
- Netscape 拡張機能が存在する場合、SSL クライアントビットがセットされている必要があります

サーバ接続の場合:

- ExtendedKeyUsage 拡張機能が存在する場合、TLS ウェブサーバ認証ビットがセットされている必要があります
- KeyUsage 拡張機能が存在する場合、デジタル署名、keyEncipherment、および keyAgreement ビットが少なくとも 1 つ設定されている必要があります
- Netscape 拡張機能が存在する場合、SSL サーバビットが設定されている必要があります

さらに、証明書のチェーンを検証するとき、リーフ証明書を除くすべての証明書は、以下のオプションの少なくとも 1 つで CA としてマークされる必要があります。

- KeyUsage 拡張機能が存在する場合、keyCertSign ビットが設定されている必要があります
- 基本的な制約の拡張機能が存在する場合、CA ビットが設定されている必要があります
- Netscape 拡張機能が存在する場合、SSL CA ビットが設定されている必要があります

3 証明書を取得する

セクション 2.1 では、展開でセキュアな接続を確立するために証明書が必要な場所、必要な証明書のタイプ (パブリック CA または内部 CA の署名) について説明します。この章では、さまざまな種類の証明書を取得する方法に焦点を当て、第 4 章 で証明書をインストールする場所について説明します。

注：Lync 導入を Meeting Server に接続する場合、Lync フロントエンドサーバーにより信頼されているものと同じ Certificate Authority（CA）を使用することをお勧めします。CA の詳細および Meeting Server と Lync の統合に関するサポートについては、Lync アドバイザに問い合わせてください。

すべての証明書は、3 つのステップのプロセスに従う必要があります。

1. 特定の Meeting Server コンポーネント用の秘密キーおよび証明書署名リクエスト（.csr）ファイルを生成します。

   メモ: 公開鍵は .csr ファイル内に作成され、保持されます。

   ---

2. .csr ファイルを署名のために CA (パブリック CA または内部 CA) に提出します。
3. CA から Meeting Server に署名済み証明書と中間バンドル（ある場合）をアップロードするために SFTP を使用します。

この章の後半では、手順 1 と 2 の例を示します。第 4 章 で手順 3 を示します。

メモ: ミーティングサーバの MMP コマンドを使用して自己署名証明書を生成する方法は、セクション 6 に記載されています。これらはラボでの構成のテストに役立ちます。ただし、本番環境ではCertificate Authority（CA）によって署名された証明書を使用することをお勧めします。

メモ: ミーティングサーバは、SHA2 アルゴリズムを使用して署名された証明書に対応しています。ミーティングサーバが証明書署名要求を作成する際、現在 CA が運用しているルールに従って、SHA256 を使用して署名されます。

3.1 秘密鍵と証明書署名リクエスト (.csr ファイル) の生成

このセクションでは、Meeting Server の MMP **`pki`** コマンドを使って、公開キーと .csr ファイルを作成する方法について説明します。サードパーティのツールを使用してこれを行う場合

は、サードパーティからの手順に従って、このガイドのセクション 3.2 から再開してください。OpenSSL を使用して秘密鍵と .csr ファイルを作成する場合、付録 1 に手順の概要が記載されています。

pki csr <key/cert basename> コマンドを使用して、秘密キー <basename>.key と証明書署名リクエストファイル <basename>.csr の 2 つのファイルを生成することができます。SFTP を使用することで、ミーティングサーバから直ちに取得することができます。

注：ベース名には「.」または「_」を含むことはできません。たとえば、**pki csr basename** は有効ですが、**pki csr base.name** または **pki csr base_name** は許可されていません。

秘密鍵と証明書署名リクエストファイルを生成するには:

1. MMP にログインします。
2. 次の構文を使って **pki csr** コマンドを入力します
   pki csr <key/cert basename> <CN:value> [OU:<value>] [O:<value>] [ST:<-value>] [C:<value>] [subjectAltName:<value>]

   説明

   <key/cert basename> は新しいキーと CSR を識別する文字列です。英数字、ハイフン、アンダースコア文字を使用できます。

   CN,OU,O,ST,C,subjectAltName は 表 4 に記載されています。ローカル Certificate Authority による署名のために pki csr コマンドを使用して証明書要求ファイルを作成している場合、これらのマークが付いたオプションは省略できます。パブリック Certificate Authority が署名するための証明書リクエストファイルを作成する場合、すべての属性を提供することをお勧めします。

表 4: .csr ファイルの属性

属性（Attribute）		説明	任意/必須
CN	一般名	これは、ドメインネームシステム（DNS）内のサーバーの正確な場所を指定する完全修飾ドメイン名（FQDN）です。たとえば、ホスト名が webBridge1、親ドメインが example.com のコンポーネントの完全修飾ドメイン名は webBridge1.example.com です。FQDN は、コンポーネントを他のドメインの webBridge1 と呼ばれる他のコンポーネントから一意に区別します。	必須、以下のメモを参照
O	組織名または会社名	通常は会社の正式な法人名です。これには、Ltd.、Inc.、Corp. などのサフィックスを含める必要があります。属性が複数の単語である場合は、" " で囲む必要があります（例：“Example Inc."）。	オプション

属性（Attribute）		説明	任意/必須
OU	組織単位または部署名	たとえば、サポート、IT、エンジニアリング、ファイナンスなどです。属性が複数の単語からなる場合は "" で囲みます（例：“人事”）	オプション
L	Location	市区町村を指定します。たとえば、ロンドン、ボストン、ミラノ、ベルリンなどです。	>オプション
ST	地方、地域、郡または州	たとえば、バッキンガムシャー、カリフォルニア州です。短縮形を使用しないでください。属性が複数の単語からなる場合は "" で囲みます（例：“New Jersey“）	オプション
C	国	組織が所在する国を表す 2 文字の ISO コードです。US、GB、FR など。	オプション
メールアドレス		組織に連絡するためのメールアドレスです。通常は証明書管理者または IT 部門のメールアドレスです。	オプション
SAN	サブジェクト代替名	X509 バージョン 3（RFC 2459）以降、SSL 証明書は証明書が一致する必要がある複数の名前を指定することが可能です。このオプションのフィールドにより、生成された証明書で複数のドメインをカバーできます。IP アドレス、ドメイン名、メールアドレス、通常の DNS ホスト名などをコンマで区切って指定します。このリストを指定する場合は、**CN** もこのリストに含める必要があります。	必須（単一の証明書を複数のコンポーネントで使用する場合）。下記のメモを参照

注意すべき点：

- Web Bridge 3 に専用の証明書を使用する場合は、CN フィールドで指定し、Web Bridge 3 の DNS A レコードで定義されている FQDN を指定します。FQDN の指定に失敗すると、ブラウザの証明書エラーが発生する場合があります。
- 複数のコンポーネント（たとえば、Web Bridge 3、Call Bridge、TURN サーバー、ウェブ管理、リオーダー、およびストリーマーなど）で同じ証明書を使用する場合は、CN フィールドでドメイン名（DN）を指定し、SAN フィールドでドメイン名（DN）と、証明書を使用する各コンポーネントの FQDN を指定します。
- [SAN] フィールドで、デリミタの「,」とリスト内の項目の間にスペースが入っていないことを確認します。

次に例を示します。

CN=**example.com**

SAN=**callbridge1.example.com,callbridge2.example.com,callbridge3.example.com,webbridge3.example.com,example.com**

pki csr コマンドを使用する場合:

```
pki csr <key/cert basename> <CN:value> [OU:<value>] [O:<value>] [ST:<value>]
[C:<value>] [<subjectAltName:value>]
```

コマンドは次のとおりです。

```
pki csr onecert CN:example.com
subjectAltName:callbridge1.example.com, callbridge2.example.com,
callbridge3.example.com, webbridge3.example.com
```

注： **pki** コマンドを使用すると、上記の例に示されているように、CN は自動的に SAN リストに追加され、SAN リストには CN は表示されません。

3.1.1 Call Bridge の CSR

導入内での各 Call Bridge の使用方法に応じて、以下を行うために、秘密キー/証明書のペアが必要になる場合があります。

- Web Bridge 3 との通信を確立。導入のセキュリティのために、信頼された Call Bridge からの設定のみを受け入れることが重要です。
- SIP 通話コントロールデバイスとの TLS 接続を確立。
- Lync フロントエンド（FE）サーバーへの TLS 接続を確立。証明書が Lync FE サーバーにより信頼されるようにするには、以下を確認します。
  - 単一の分割および単一の統合導入の場合：証明書の **CN** は、Meeting Server を Lync FE サーバーの信頼されたアプリケーションおよび統計ルートとして設定するときに追加された FQDN と同じである必要があります。
  - スケーラブルで復元力のある導入の場合：**CN** は、Meeting Server を Lync FE サーバー上の信頼できるアプリケーションとして構成するときに追加された FQDN と同じである必要があります。この値を確認するには、Lync Powershell コマンド **get-cstrustedapplicationcomputer** を使用します。
  - 証明書に **SubjectAltName** リストにある場合、FQDN もリストに追加する必要があります。
  - Lync FE サーバの証明書を発行した CA など、信頼された CA サーバーを使用して証明書に署名します。

次に例を示します。

```
pki csr callbridge CN:www.example.com O:"Example Inc."
```

または

```
pki csr callbridge CN:www.example.com O:"Example Inc."
subjectAltName:callbridge.example.com
```

この例では、callbridge.key と callbridge.csr の 2 つのファイルが生成されます。ファイルは SFTP を使用してミーティングサーバから直ちに取得することができます。署名用に .csr ファイルをパブリック CA に送信します。セクション 3.2 を参照してください。

セクション 4.4 では、Call Bridge の証明書のアップロードについての詳細を記載しています。

3.1.2 ウェブブリッジ 3 の CSR

ウェブブラウザは **CN** フィールドを調べて、ウェブブリッジ 3 の FQDN を決定します。ウェブブラウザの証明書エラーを回避するには、次のアドバイスに従ってください:

- Web Bridge 3 の専用証明書を使用している場合：**CN** フィールドで、Web Bridge 3 の DNS A レコードで定義されている FQDN を指定します。FQDN を指定しないと、ブラウザの証明書エラーが発生する場合があります。**subjectAltName** フィールドが使用されている場合、**CN** フィールドで指定されている FQDN は、自動的に追加されなければ **subjectAltName** フィールドに含める必要があります。注意: **pki csr** は、**CN** を **SAN** リストに自動的に追加します。**SAN** リストが存在します。

  注： Cisco Expressway ウェブプロキシをWeb Bridgeに接続する必要がある導入の場合、Web Bridge証明書の SAN フィールドが、Web Bridgeに接続する Expressway-C で使用される A レコードが含まれていることを確認してください。そうしないと、接続に失敗します。例えば、Expressway が join.example.com のWeb Bridgeに接続するように設定されている場合、A レコードがこの FQDN に対して存在している必要があり、Web Bridge証明書の SAN フィールドには join.example.com が含まれている必要があります。

- 複数のコンポーネント (ウェブブリッジ 3、Call Bridge、および TURN サーバ) で同じ証明書を使用する予定の場合: [**CN**] フィールドでドメイン名 (DN) を指定し、[**SAN**] フィールドに、ドメイン名 (DN) と、証明書を使用する各コンポーネントの FQDN を指定します。

次に例を示します。

```
pki csr webbridge3 CN:www.example.com O:"Example Inc."
```

または

```
pki csr webbridge3 CN:www.example.com O:"Example Inc."
subjectAltName:guest.example.com
```

この例では、webbridge3.key および webbridge3.csr の 2 つのファイルを生成します。ファイルは SFTP を使用してミーティングサーバから直ちに取得することができます。署名用に .csr ファイルをパブリック CA に送信します。セクション 3.2 を参照してください。

セクション 4.5 には、Web Bridge 3 の証明書のアップロードの詳細が記載されています。

3.1.3 MeetingApps の CSR

ウェブブラウザは MeetingApps と直接通信するため、ブラウザの信頼できる CA 証明書を使用する必要があります。内部 CA 署名付き証明書を使用する予定の場合、ウェブ アプリで使用する各ブラウザーで証明書を検証する必要があります。

メモ: 内部 CA 署名付き証明書を使用している場合、ファイル共有機能は一部のブラウザおよびオペレーティングシステムでは動作しない可能性があります。

MeetingApps には専用の証明書を使用することをお勧めします。ウェブブラウザは **CN** フィールドを見て、MeetingApps の FQDN を決定します。**CN** フィールドで、MeetingApps の DNS A レコードで定義されている FQDN を指定します。FQDN の指定に失敗すると、ブラウザの証明書エラーが発生する場合があります。 **subjectAltName** フィールドを使用する場合、**CN** フィールドに指定された FQDN が自動的に追加されない場合は、**subjectAltName** フィールドに含める必要があります。

注：pki csr は、**SAN** リストが存在する場合は、**CN** を **SAN** リストに自動的に追加します。

次に例を示します。

```
pki csr meetingapps CN:www.example.com O:"Example Inc."
```

または

```
pki csr MeetingApps CN:www.example.com O:”Example Inc.”
subjectAltName:guest.example.com
```

この例では、meetingapps.key と meetingapps.csr の 2 つのファイルが生成されます。ファイルは SFTP を使用してミーティングサーバから直ちに取得することができます。署名用に .csr ファイルをパブリック CA に送信します。セクション 3.2 を参照してください。

セクション 4 には、MeetingApp の証明書のアップロードに関する詳細が記載されています。

3.1.4 データベースクラスタリングの CSR

注：このセクションは、スケーラブルで復元力のある導入にのみ適用されます。

バージョン 2.7 以降、データベースクラスタでは、クラスタ内のデータベースを保持している、または接続している各ミーティングサーバに設定されている、同一の認証局（CA）によって署名されたクライアント証明書とサーバ証明書が必要となります。証明書の使用を義務付けることで、クラスター全体における機密性と認証の両方が確保されます。

2.7 から、MMP から直接すべての証明書を生成できます。その後、証明書とキーを SFTP 経由でダウンロードし、同じデータベースクラスタに属する各ミーティングサーバにアップロードする必要があります。

警告：データベースクラスタを形成するノードは、信頼できるルート CA 証明書で設定する必要があります。これにより、正当なノードだけがクラスタに接続できます。ノードは、信頼できるルート証明書で終わる証明書チェーンを提示する接続を信頼します。そのため、各データベースクラスタは専用のルート証明書を使用する必要があり、ルート証明書または中間証明書は他の目的には使用されてはいけません。

---

openssl など、その他の方法を使用して証明書を作成することもできます。詳細をご確認ください。

データベースクラスタリングの場合：

1. dbca 自己署名証明書を pki selfsigned コマンドで作成します。

   次に例を示します。

   ```
   pki selfsigned dbca CN:"My company CA"
   ```

   dbca.key という名前のローカル秘密キーを作成し、**CN=My company CA** という共通名を持つ自己署名証明書を dbca.crt に作成します

2. データベースサーバ用の秘密鍵と証明書リクエストファイルを作成します。データベースクラスタ内のすべてのサーバーで同じ証明書を使用できます。CN フィールドにサーバーの 1 つの FQDN を指定し、SAN フィールドに他のサーバーの FQDN を指定します。「拡張キーの使用状況」を使用している場合、データベースサーバーで「サーバー認証」が許可されていることを確認してください。

   次に例を示します。

   ```
   pki csr dbserver CN:server.db.example.com
   subjectAltName:server02.db.example.com
   ```

   dbserver.csr という名前の CSR ファイルと dbserver.key という名前の秘密鍵が生成されます。

3. データベースクライアント用の秘密鍵と証明書リクエストファイルを作成します。データベースクライアントの共通名（CN）は「postgres」と等しくなければなりません。「拡張キー使用法」を使用している場合、「クライアント認証」がデータベースクライアントに許可されていることを確認します。

   次に例を示します。

   ```
   pki csr dbclient CN:postgres
   ```

   dbclient.csr という名前の CSR ファイルと dbclient.key という名前の秘密鍵を生成します

4. 内部 CA を使用して dbserver.csr および dbclient.csr 証明書署名リクエストファイルに署名し、対応する dbserver.crt および dbclient.crt 証明書、および内部 CA 証明書（バンドル）を取得してください。

   次に例を示します。

   ```
   pki sign dbserver dbca
   pki sign dbclient dbca
   ```

1. データベースサーバ用の秘密鍵と証明書リクエストファイルを作成します。データベースクラスタ内のすべてのサーバーで同じ証明書を使用できます。CN フィールドにいずれかのサーバーの FQDN を指定し、SAN フィールドにその他のサーバの FQDN を指定します。「拡張キーの使用状況」を使用している場合、データベースサーバーで「サーバー認証」が許可されていることを確認してください。

   次に例を示します。

   ```
   pki csr db01server CN:www.example.com
   ```

   db01server.csr という名前の CSR ファイルと db01server.key という名前の秘密鍵が生成されます。

2. データベースクライアント用の秘密鍵と証明書リクエストファイルを作成します。データベースクライアントの共通名（CN）は「postgres」と等しくなければなりません。「拡張キー使用法」を使用している場合、「クライアント認証」がデータベースクライアントに許可されていることを確認します。

   次に例を示します。

   ```
   pki csr db01client CN:postgres
   ```

   db01client.csr という名前の CSR ファイルと db01client.key という名前の秘密鍵を生成します

3. 内部 CA を使用して db01server.csr および db01client.csr 証明書署名リクエストファイルに署名し、対応する db01server.crt および db01client.crt 証明書、および内部 CA 証明書（バンドル）を取得してください。セクション 3.3 を参照してください。

セクション 4.4 では、データベースクラスタリングの証明書のアップロードの詳細について説明します。

3.1.5 TURN サーバの CSR

TURN サーバーで TLS を使用する場合、TURN サーバーは Web Bridge 3 用に作成されたものと同様の証明書/キーペアを要求し、それにより WebRTC クライアントは接続を信頼します。証明書は、ウェブブリッジ 3 証明書に使用されるものと同じ認証局によって署名されている必要があります。

次に例を示します。

```
pki csr turnserver CN:www.example.com O:"Example Inc."
```

この例では、turn.key と turn.csr の 2 つのファイルが生成されます。ファイルは SFTP を使用してミーティングサーバから直ちに取得することができます。署名用に .csr ファイルをパブリック CA に送信します。セクション 3.2 を参照してください。

セクション 4.6 には、TURN サーバの証明書のアップロードに関する詳細が記載されています。

3.2 パブリック Certificate Authority を使用した CSR の署名

ミーティングサーバに必要な公開 CA 署名証明書のリストについては、セクション 2.1 を参照してください。

パブリック CA 署名付き証明書を取得するには、生成された .csr ファイルを Verisign などの希望の Certificate Authority に送信します。CA で ID を確認し、Meeting Server とその使用要件に必要なパブリック CA 署名付き証明書のリストに対して署名付き証明書を発行します。証明書ファイルの拡張子は .pem、.cer または .crt です。付録 B では、証明書ファイルに使用されるファイル拡張子の概要を説明します。

署名済み証明書と秘密キーを Meeting Server に転送する前に、証明書ファイルを確認してください。CA が証明書のチェーンを発行している場合、チェーンから証明書を抽出する必要があります。証明書ファイルを開き、BEGIN CERTIFICATE および END CERTIFICATE の行を含む特定の証明書テキストをコピーして、テキストファイルに貼り付けます。.crt、.cer または .pem の拡張子を持つ証明書としてファイルを保存します。残りの証明書チェーンをコピーして別のファイルに貼り付けます。中間証明書チェーンと認識できるように明確な名前を付け、同じ拡張子（.crt、.cer または .pem）を使用します。中間証明書チェーンは順番通りである必要があります。チェーンを発行した CA の証明書が最初で、ルート CA の証明書がチェーンの最後です。

ミーティングサーバへの署名付き証明書と秘密鍵のインストールについては、第 4 章 を参照してください。

注：証明書を展開する前に、**`pki inspect`** コマンドまたは openssl ツールを使用して、すべての証明書の CN、SAN、KeyUsage、および ExtendedKeyUsage の値が正しいことを確認することをお勧めします。

3.3 内部認証局を使用して CSR に署名する

ミーティングサーバに必要な内部 CA 署名付き証明書のリストとその使用要件は、セクション 2.1 を参照してください。

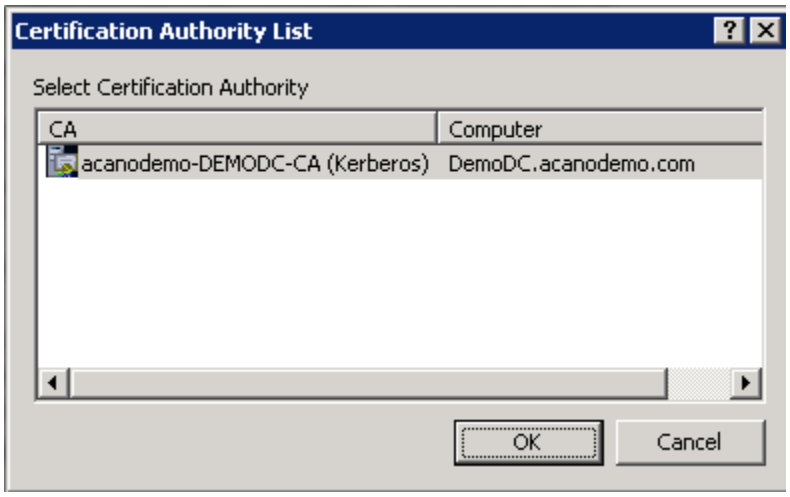
注：Meeting Server が Cisco Unified Communications Manager にトランク接続されている導入では、Cisco Unified Communications Manager は、TLS ウェブクライアント認証および TLS ウェブサーバー認証の両方を含む拡張キーの使用状況を許可するテンプレートを使用して Call Bridge 証明書に署名することを要求します。Microsoft Active Directory 証明書サービスはこのタイプの証明書を発行できます。

注：証明書を展開する前に、**`pki inspect`** コマンドまたは openssl ツールを使用して、すべての証明書の CN、SAN、KeyUsage、および ExtendedKeyUsage の値が正しいことを確認することをお勧めします。

このセクションは、Microsoft Active Directory を内部 CA として使用している場合に適用されます。別の内部 CA を使用している場合は、対応する手順に従って、このガイドの第 4 章から再開してください。

内部 CA の署名付き証明書を取得するには、以下の手順に従います。

1. 生成された .csr ファイルを CA に転送します。たとえば、Active Directory 証明書サービスの役割がインストールされた Active Directory サーバなど。
2. CA サーバのコマンド ライン管理シェルで次のコマンドを発行し、パスと CSR ファイル名を自分の情報に置き換えます。
   `certreq -submit -attrib "CertificateTemplate:Computer" <path\csrfilename>`
   次に例を示します。
   `certreq -submit -attrib "CertificateTemplate:Computer" C:\Users\Administrator\Desktop\example.csr`
3. コマンドを入力すると、以下のような CA 選択リストが表示されます。正しい CA を選択し、[OK] をクリックします。



Windows アカウントに証明書を発行する権限がある場合、生成された証明書を保存するようにプロンプトが表示されます。.crt、.cer、または .pem の拡張子を付けてファイルを保

存します (例: example.crt)。ステップ 4 に進みます。証明書ファイル拡張子の概要については、付録 B を参照してください。

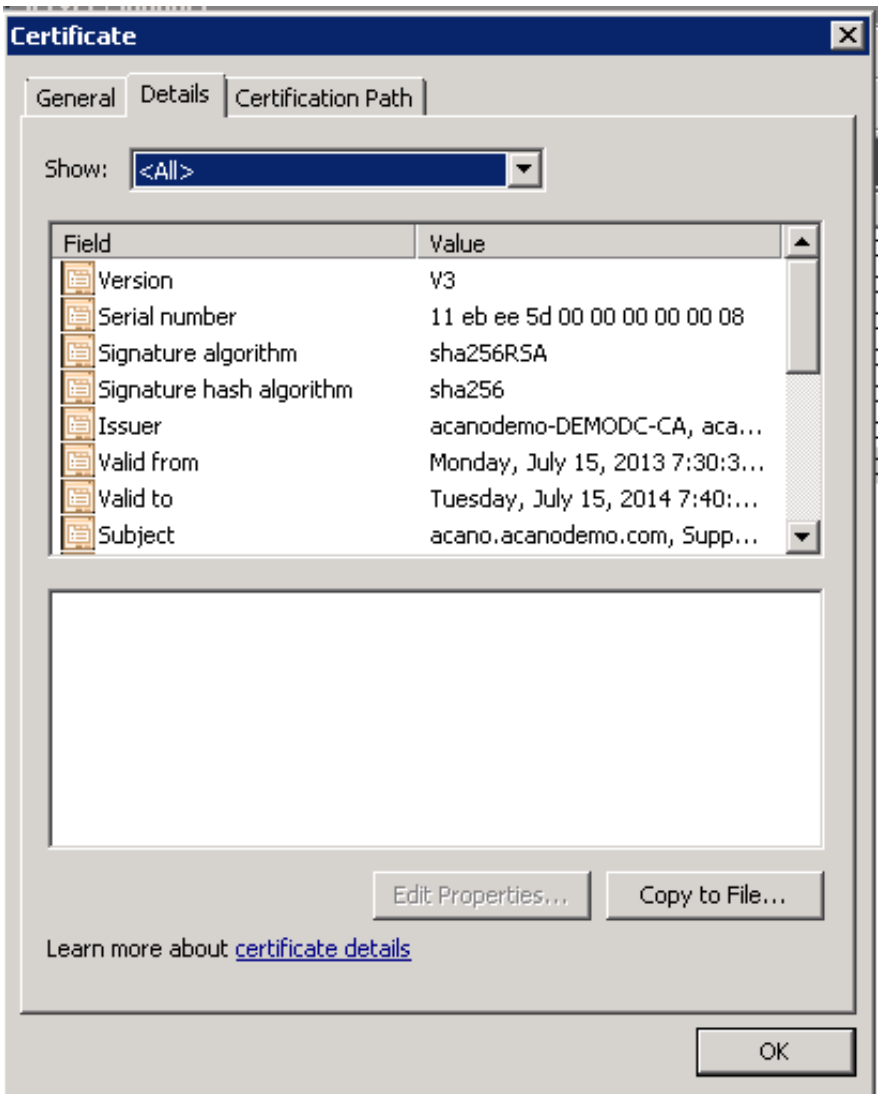
生成された証明書を発行するためのプロンプトが表示されず、代わりにコマンドプロンプトウィンドウに「Certificate request is pending: taken under submission」というメッセージ、およびリクエスト ID のリストが表示される場合は、リクエスト ID をメモします。

以下の手順に従って、発行された証明書を取得します。

a. CA の **サーバ マネージャー** ページを使用して、CA ロールの下にある **保留中のリクエスト** フォルダーを探します。

b. **cmd** ウィンドウに表示されたリクエスト ID に一致する保留中のリクエストを右クリックし、[**すべてのタスク**] > [**発行**] を選択します。

c. 署名付き証明書は [発行済み証明書] フォルダに保存されます。証明書をダブルクリックして開き、[**詳細**] タブを開きます。



d. [**ファイルにコピー**] をクリックして、証明書のエクスポートウィザードを開始します。

e. [Base-64 エンコード X.509 (.CER)] を選択して [**次へ**] をクリックします。

f. 証明書を保存する場所を参照し、名前を入力し、
`callbridge` などの名前を入力して、[**次へ**] をクリックします。

g. 生成された証明書に .crt、.cer、.pem などの拡張子を付けて保存します。
`callbridge.crt` という名前の自己署名証明書を作成します

4. ミーティングサーバへの署名付き証明書と秘密鍵のインストールについては、第 4 章 を参照してください。

4 ミーティングサーバへの署名証明書と秘密鍵のインストール

スケーラブルで復元力のあるミーティングサーバの展開では、以下に対してパブリック CA 署名証明書が必要です。

- セキュアな通信のために TLS 接続を使用する場合は、**TURN サーバー**。
- パブリックネットワーク経由での直接 Lync フェデレーションが必要な場合は、**Call Bridge**。Lync Edge サーバは、接続を信頼するために、Call Bridge からのパブリック CA 署名付き証明書を必要とします。
- ウェブアプリを使用するために Web Bridge 3 を導入する場合は、**Web Bridge 3**。接続を信頼するために Call Bridge からのパブリック CA 署名付き証明書が必要です。

以下に対するパブリック CA または内部 CA 署名付き証明書：

- **ウェブ管理**。Meeting Server API はウェブ管理のインターフェイスを通じてルーティングされるため、ウェブ管理インターフェイスではなく API を通じて Call Bridge を設定する場合でも、証明書が必要です。

  注：このガイドは、ユーザーがウェブ管理インターフェイス用の秘密キー/証明書のペアをすでにインストールしていることを前提としています（詳細は、『Meeting Server インストールガイド』を参照してください）。まだ行っていない場合は、今すぐ行ってください。

- **Call Bridge**。Web Bridge 3 c2w 接続は、Call Bridge からの証明書を信頼する必要があります。また、Active Directory サーバは、Call Bridge からの証明書も必要とします。さらに、展開に SIP トランクがある場合、Call Bridge には SIP 通話制御デバイスとの相互認証用の証明書が必要です。
- 各データベースサーバーとデータベースクライアント（データベースと共存していない Call Bridge を含む）が、秘密キーと、同じ Certificate Authority によって署名された証明書を必要とする、**データベース**クラスタリング。(スケーラブルで復元力のある展開のみ。)
- Meeting Server で**レコーダー**を有効にすると、Call Bridge はレコーダーからの証明書を要求し、レコーダーは Call Bridge からの証明書を要求し、信頼する必要があります。証明書のアップロードとレコーダーの設定の詳細については、『Cisco Meeting Server スケーラブルで復元力のある展開ガイド』のレコーダーのセクションを参照してください。

- ミーティングサーバで **ストリーマー** を有効にすると、Call Bridge はストリーマからの証明書を必要とし、ストリーマーは Call Bridge からの証明書を要求し、信頼する必要があります。証明書のアップロードとストリーマの設定の詳細については、『Cisco ミーティング サーバ スケーラブルで復元力のある展開ガイド』のストリーマのセクションを参照してください。
- **アップローダー**を有効にする場合、Vbrick Rev Server と Cisco Meeting Server の Web 管理インターフェイスの両方のルート CA と中間証明書を含む証明書バンドルが必要です。録画データを Vbrick Rev にアップロードする際に、信頼性の問題なく安全な HTTPS 通信を有効にするには、このバンドルが Cisco Meeting Server によって信頼されている必要があります。Cisco Meeting Server Single Split Server 導入ガイドの「Vbrick と連携するように Meeting Server を設定する」セクションを参照してください。
- Web Bridge 3 を有効にすると、C2W 証明書が Call Bridge と Web Bridge 3 の間の接続に使用されます。Call Bridge が Web Bridge 3 への C2W 接続を行う場合、証明書を検証するための C2W 用の信頼ストアを指定する必要があります。
- MeetingApps で使用する証明書は、MMP コマンドを使って MeetingApps を設定する際に割り当てる必要があります。MeetingApps の内部 CA 署名付き証明書を使用している場合、各ブラウザで MeetingApps アドレスに接続して証明書を検証する必要があります。

4.1 秘密鍵と証明書を再利用する

証明書をインストールするたびに、異なる秘密鍵/証明書のペアを用意する必要はありません。状況によっては、秘密鍵と証明書をコピーして複数のサービスで再利用することができます。以下は、秘密鍵/証明書のペアを再利用する場合のアドバイスです。

- Lync 導入を Meeting Server に接続する場合、Lync 導入によって信頼されている Certificate Authority（CA）を使用することをお勧めします。
- 証明書と秘密キーのファイル名は、それらが使用される場所を反映するものにします。例：**`webadmin.crt`** および **`webadmin.key`**。

4.2 秘密鍵と証明書を MMP にアップロードする

1. SSH で MMP に接続し、ログインする
2. SFTP を使用して、各秘密鍵/証明書のペアと証明書バンドルをアップロードします
3. MMP PKI コマンド：**`pki list`** を使用して、アップロードされたファイルを確認します。**`pki list`** では、MMP にアップロードされた SSH キーと CSR ファイルも一覧表示します。

注：ファイル拡張子の直前を除き、秘密キーおよび証明書のファイル名には「.」を含めてはいけません。たとえば、callbridge.key は有効ですが、call.bridge.key は許可されません。

4.3 ファイルの種類を検査し、証明書と秘密鍵が一致することを確認する

Meeting Server に秘密キー/証明書のペアをインストールする前に、インストールする正しいファイルがあることを確認してください。このセクションでは、MMP コマンドの使用方法について簡単に概説します。インストール予定のファイルの真正性を確認するには、pki inspect、pki match、および pki verify を実行します。

ファイルを確認して、ファイルがまだ有効（有効期限内）であるかどうかを特定するには、次のコマンドを入力します。

```
pki inspect <filename>
```

証明書が秘密キーと一致することを確認するには、次のコマンドを入力します。

```
pki match <keyfile> <certificatefile>
```

証明書が CA によって署名され、証明書バンドルを使用してこれを表明できることを確認するには:

```
pki verify <cert> <certbundle/CAcert>
```

次に例を示します。

1. SSH で MMP に接続し、ログインする
2. コマンドを入力してください：

   ```
   pki inspect callbridge.crt
   ```

   ファイルの内容を検査するには、たとえば、証明書がまだ有効かどうかを確認します。
3. コマンドを入力してください：

   ```
   pki match callbridge.key callbridge.crt
   ```

   ファイル `callbridge.key` がファイル `callbridge.crt` と一致し、一緒になって 1 つの使用可能な ID を形成することを確認します。
4. コマンドを入力してください：

   ```
   pki verify callbridge.crt callbridgebundle.crt
   ```

   callbridge.crt が信頼できる CA によって署名されており、callbridge.crt の中間証明書のチェーンを通じて確立された信頼チェーンがあることを確認します。

4.4 Call Bridge の証明書と秘密鍵のインストール

セクション 3.1.1 で説明されているように、導入内で各 Call Bridge がどのように使用されているかによって、秘密キー/証明書のペアが必要になる場合があります。

以下の手順は、各 Call Bridge がリッスンに使用するネットワーク インターフェイスをすでに設定していることを前提としています。証明書を割り当てる前に、MMP コマンド listen を使用してインターフェイスを設定する方法については、『スケーラブルで復元力のあるサーバーの導入ガイド』を参照してください。

各 Call Bridge:

1. Meeting Server の MMP に SSH で接続します。
2. 次のコマンドを使用して、秘密鍵/証明書のペアを割り当てます。

   `callbridge certs <keyfile> <certificatefile>[<cert-bundle>]`

   `keyfile` と **`certificatefile`** は、一致する秘密キーと証明書のファイル名です。CA が証明書バンドルを提供している場合は、バンドルも証明書とは別のファイルとして含めます。

   次に例を示します。

   `callbridge certs callbridge.key callbridge.crt callbridgebundle.crt`
3. 変更を適用するために、Call Bridge インターフェイスを再起動します。

   `callbridge restart`

証明書が Call Bridge に正常にインストールされると、次のメッセージが表示されます。

```
SUCCESS: listen interface configured
SUCCESS: Key and certificate pair match
```

証明書のインストールに失敗すると、次のエラーメッセージが表示されます。

失敗: キーと証明書の問題: 証明書とキーが一致しません

注：Web Bridge 3 の設定が完了したら、すべての Web Bridge 3 の信頼ストアに Call Bridge 証明書を追加する必要があります。

注：MMP コマンド **`callbridge certs none`** を使用して、Call Bridge から証明書の設定を削除します。

4.4.1 Call Bridge と Web Bridge 3 の間の信頼の確立

Web Bridge 3 では、ゲストログインの設定と画像のカスタマイズを Call Bridge からプッシュできます（カスタマイズのガイドライン）を参照してください。導入のセキュリティにとって、設定は信頼できる Call Bridge からのみ受け入れることが重要です。

C2W 証明書は、Call Bridge とウェブブリッジ 3 の間の接続に使用されます。Call Bridge がウェブブリッジ 3 への C2W 接続を行うには、証明書を確認するための C2W トラストストアを指定する必要があります。詳細は セクション 4.5 および 付録 Dを参照してください。

4.5 Web Bridge 3 の証明書と秘密鍵をインストールする

Cisco Meeting Server ウェブアプリを使用するために Web ブリッジ 3 を展開して証明書を構成する方法の詳細は、付録 D を参照してください。以下は、ウェブ アプリを使用できるように Web Bridge 3 を設定するのに役立つ概要情報です。

- Web Bridge 3 では、すべての証明書定義で証明書のバンドル（つまり、フルチェーンファイル）を使用する必要があります。これは、ウェブ ブリッジ 2 に証明書が実装される方法とは異なります。
- 「Call Bridge to Web Bridge」プロトコル（C2W）は、Call Bridge と Web Bridge 3 間のリンクです。
- Call Bridge が Web Bridge 3 に接続できるように、（**`webbridge3 c2w listen`** を使用して）インターフェイス上でポートを開く必要があります（Web Bridge 3 はそのポートでリッスンします）。このため、API 要求を実行して Call Bridge にこの Web Bridge 3 について伝えるときに、このポートでアドレスを指定する必要があります。この接続は証明書で保護する必要があります。
- その開いたポートを外部アクセスから保護することをお勧めします。Call Bridge からのみ到達可能である必要があります。
- Call Bridgeは **`callbridge certs`** を使用した証明書セットを使用し、Web Bridge 3 は **`webbridge3 c2w certs`** を使用した証明書セットを使用します。
- Web Bridge 3 は、**`webbridge3 c2w trust`** で設定された信頼ストアの中のものによって署名された Call Bridge の証明書を信頼します。
- Call Bridge は、**`callbridge trust c2w`** で設定された信頼ストアの中のものによって署名された証明書を持つ Web Bridge 3 を信頼します。
- 公的機関によって署名された証明書は必要ありません。MMP 内で作成された自己署名証明書を使用できます。
- SAN/CN は、Call Bridge API の Web Bridge 3 を登録するための c2w:// url で使用される FQDN または IP アドレスと一致する必要があります。（これが一致しない場合、Call Bridge は TLS ネゴシエーションに失敗し、Web Bridge 3 によって提示された証明書を拒否し、Web Bridge 3 との接続に失敗します。）
- Call Bridge および Web Bridge 3 の証明書拡張子の要件の詳細については、第 2.1 章を参照してください。

4.6 TURN Server の証明書と秘密鍵をインストールする

セキュアな通信に TLS を使用する予定の場合は、TURN サーバ用の署名済み証明書をインストールする必要があります。その際、ウェブブリッジで使用したものと同じ CA を使用して証明書に署名します。この証明書は、ブラウザがミーティングサーバとの接続を信頼するかどうかを判断する際に使用されます。

以下の手順は、TURN サーバが待機するために使用するネットワークインタフェースをすでに構成していることを前提としています。証明書を指定する前に、**listen** MMP コマンドを使ってインターフェースを設定する方法について、『Scalable and Resilient Server 導入ガイド』を参照してください。

各 TURN サーバ:

1. ホストサーバーの MMP に SSH で接続します
2. 証明書を割り当てる前に、TURN サーバインターフェイスを無効にする

   turn disable
3. CA から Meeting Server に署名済み証明書と中間バンドル（ある場合）をアップロードするために SFTP を使用します。
4. 証明書（および証明書バンドル）と秘密キーが一致していることを確認します

   pki verify <certicate> <cert bundle/CA cert> [<CA cert>]
5. 証明書（および証明書バンドル）と秘密キーのペアを Turn サーバーに割り当てます

   turn certs <keyfile> <certificatefile> [<cert-bundle>]

   ここで、keyfile と certificatefile は、対応する秘密鍵と証明書のファイル名です。CA が証明書バンドルを提供している場合は、バンドルも証明書とは別のファイルとして含めます。

   例

   turn certs turn.key turn.crt turnbundle.crt
6. TURN サーバを再度有効にする

   turn enable

4.7 ミーティングアプリの証明書をインストールする

参加者がウェブアプリミーティングでファイルを共有する前に、MeetingApps の証明書を設定する必要があります。公的に署名されたブラウザの信頼できる CA 証明書の使用を推奨します。ただし、内部 CA 署名付き証明書を使用している場合、ウェブ アプリはミーティングに参加するために使用する各ブラウザーでこれらの証明書を検証するようにプロンプトを表示します。

証明書を割り当てる前に、MeetingApps が通信に使用するインターフェイスとポートを設定する必要があります。MeetingApps インターフェイスとポートの設定に関する詳細については、*Cisco Meeting Server 導入ガイド*を参照してください。

MeetingApps 証明書を設定するには:

1. ホストサーバの MMP に SSH 接続します。
2. 証明書を割り当てる前に MeetingApps を無効にします

   meetingapps disable
3. コマンドを使用して MeetingApp の証明書キーペアを割り当てます

   meetingapps https certs <key-file> <crt-fullchain-file> を使用して MeetingApp の証明書キーペアを設定します。
4. ミーティングアプリを有効にする

   meetingapps enable

4.7.1 MeetingApps 証明書の検証

接続を信頼するには、MeetingApp に使用される内部 CA 署名付き証明書が、ウェブ アプリ ミーティングに使用する各ブラウザーで検証される必要があります。内部 CA 署名付き証明書が確認されない場合、参加者はミーティングでファイルを共有できず、ウェブアプリが参加者に MeetingApps 証明書を確認するために使用できるリンクを提示します。

証明書を確認するには、ブラウザーの新しいタブを開き、MeetingApps アドレスの後に：とポート番号を入力します。プロンプトが表示されたら、証明書を受け入れます。MeetingApps へのアクセスを継続してください。MeetingApps に使用されるアドレスとポートは、**meetingapps** コマンドを使用して取得できます。

メモ: ウェブ アプリ ミーティングを使用するすべてのブラウザーでプロセスを繰り返す必要があります。

MeetingApps 証明書を検証した後、ウェブ アプリ ミーティングでファイルを共有できます。

4.7.2 信頼されたルート認証局に証明書を追加する

または、内部 CA 署名付き証明書を、ウェブアプリミーティングで使用するすべてのブラウザの信頼されたルート証明機関ストアにインストールすることもできます。

内部 CA の署名付き証明書を信頼されたルート証明機関ストアに追加するには:

1. MeetingApp で使用される中間証明書とルート証明書をダウンロードします。

2. すべての内部 CA 署名付き証明書を信頼されたルート証明機関にインポートします。

証明書を信頼ストアにインポートしたら、ウェブ アプリ ミーティングに参加してファイルを共有できます。

4.8 TLS 証明書の検証

リモート証明書が信頼できるものであることを確認するために、SIP および LDAP の相互認証を有効にすることができます。有効にすると、Call Bridge は常にどちらの側で接続を開始したかに関係なく、リモート証明書を要求し、提示された証明書を、アップロードされ、サーバーに定義されている信頼ストアと比較します。

利用できる MMP コマンドは以下の通りです。

- 現在の構成を表示します

  tls<sip|ldap>
- 信頼する認証局を定義する

  tls <sip|ldap> trust <crt bundle>
- 証明書の検証を有効/無効にするか、検証に OCSP を使用するかどうかを指定します。

  tls <sip|ldap> verify enable|disable|ocsp

詳細については、『MMP コマンドリファレンス』ガイドを参照してください。

4.9 Call Bridge クラスタの検証

注：このセクションは、スケーラブルで復元力のある導入にのみ適用されます。

クラスタ内の Call Bridge を検証するための Call Bridge 信頼ストアを使用することで、Call Bridge クラスタのセキュリティを向上させることができます。Call Bridge はウェブ管理の前面にある HTTPS を介してお互いに接続するため、クラスター化された Call Bridge のウェブ管理証明書を保持する証明書バンドルを作成し、クラスタ内のそれぞれの Call Bridge の信頼ストアに証明書バンドルをアップロードする必要があります。MMP コマンドを使用します。

callbridge trust cluster <bundle name>

Call Bridge がクラスタ内の別の Call Bridge に接続するとき、信頼ストア内の証明書のバンドルをチェックして、接続先の Call Bridge のアイデンティティを検証します。これにより、Call Bridge が安全ではないミーティング サーバに接続するリスクを排除できます。証明書バンドルは、証明書チェーンまたは信頼できる証明書の許可リストのいずれかになります。

注：バージョン 3.4 から、「**callbridge trust cluster**」が有効な場合に証明書名の検証が実装されました。このため、クラスタリングで設定されたピアは、対応するウェブ管理証明書の FQDN と完全に一致し、この設定に失敗すると、Call Bridge クラスタエラーが発生します。

トラストストアが使用されない場合、クラスター化された Call Bridge 間で証明書の検証が行われず、Call Bridge はリモート Call Bridge への接続を継続しますが、アイデンティティの確認は行いません。

Call Bridge 信頼ストアから Call Bridge クラスタ証明書バンドルを削除するには、次の MMP コマンドを使用します。

```
callbridge trust cluster none
```

5 証明書に関する問題のトラブルシューティング

このセクションでは、いくつかの一般的なトラブルシューティングの問題について説明します。Meeting Server ナレッジベースを参照して、 証明書に関するその他のよくある質問を確認してください。

5.1 サービスが信頼されていないという警告メッセージ

このメッセージは、次の場合に表示されます。

- 信頼ストアにない内部 CA を使用している。
- 公開または内部 CA 署名証明書が必要な自己署名証明書を使用した。証明書を再発行し、信頼できる CA で署名してもらいます。このコンポーネントにパブリックアクセスを希望しない限り、内部 CA を使用することができます。

5.2 Lync フロントエンドサーバへの接続の問題

Call Bridge 証明書に署名した CA が、Lync フロントエンドサーバーの証明書に署名するために使用されたのと同じ CA であることを確認してください。Call Bridge 証明書が、Lync フロントエンドサーバの証明書の署名に使用されたのと同じ CA によって署名されていない場合、Call Bridge の信頼できる CA 証明書を Lync サーバーのルート信頼ストアにアップロードして、Lync サーバーが Call Bridge 証明書を信頼できるようにしてください。

Lync に追加された FQDN が Call Bridge の証明書の CN としても存在していることを確認してください。

5.3 証明書の期限切れ間近または期限切れのメッセージ

証明書の更新は、新しい証明書セットの展開と同じプロセスに従います。証明書の取得とインストールに関する前のセクションを参照してください。

クラスター環境の場合、証明書の有効期限が切れると、クラスター内のデータベース ノードは相互の通信を停止します。クラスタがコマンド **`database cluster remove`** を使用して削除される場合を除き、証明書は、Meeting Server データベースクラスタノードで更新できません。そのため、データベースのクラスタ化を解除し、証明書を更新して、もう一度クラスタを作成してください。詳細については Cisco ミーティングサーバを使って期限切れのデータベースクラスタ証明書を更新する方法を参照してください。

6 テスト環境での証明書の作成と使用

pki selfsigned コマンドを使用して、Meeting Server 上で秘密キーと自己署名証明書を作成できます。

自己署名証明書は、「クラスターキー」には使用できません（CA 証明書を取得できないため（スケーラブルで復元力のある導入の場合のみ））または Lync 認証には使用できません（CA が信頼できる機関ではないため）。自己署名証明書は、ブラウザが証明書エラーを表示しても、ウェブ管理、および Call Bridge とウェブブリッジ間の相互認証に使用できます。本番環境ではなく、テスト環境で自己署名証明書を使用することを強く推奨します。

ミーティングサーバ上でローカル秘密鍵と自己署名証明書を生成するには:

1. MMP にログインし、次のコマンドを入力します。

   pki selfsigned <key/cert basename>

   ここで、**<key/cert basename>** は、生成されるキーと証明書を識別します。

   次に例を示します。

   pki selfsigned callbridge

   は、**callbridge.key** という名前のローカル秘密キーと **callbridge.crt** という名前の自己署名証明書を作成します

付録 A 証明書を生成するための OpenSSL コマンド

セクション 1.2 に記載されている MMP `pki` コマンドの代わりに、OpenSSL を使用して秘密キー、証明書署名リクエスト、および証明書を生成することができます。この付録では、使用する OpenSSL コマンドについて詳しく説明します。この例では、OpenSSL が Windows で実行されていることを前提としていますが、他のプラットフォームでも OpenSSL を使用することができます。

メモ: OpenSSL は管理者モードで実行してください。

メモ: この章の例では、ウェブブリッジ 3 を使用します。

A.1 RSA 秘密鍵と CSR ファイルの生成

お使いのコンピュータで OpenSSL ツールキットを使用してください。

新しい RSA 秘密鍵および CSR ファイルを生成するには、次のコマンドを使用します:

```
openssl req -out webbridge3.csr -new -newkey rsa:2048 -nodes -keyout
webbridge3.key
```

webbridge3.csr という名前の CSR ファイルと、webbridge3.key という名前の RSA 2048 ビット秘密キーを生成します。

注：キー名と証明書名には、「.」または「_」を含めてはいけません。たとえば、`webbridge3` は有効ですが、`web.bridge 3` または `web_bridge_3` は使用できません。

既存の秘密鍵に対する証明書署名リクエスト CSR を生成するには、次のコマンドを使用します:

```
Openssl req -out <certname>.csr -key <keyname>.key -new
```

次に例を示します。

```
openssl req -out webbridge3.csr -key webbridge3.key -new
```

は、webbridge3.key という名前の既存の秘密キーに基づいて、webbridge3.csr という名前の CSR ファイルを生成します。

OpenSSL を使用して証明書に自己署名する場合、中間 CSR ファイルは必要ないため、次のセクションに進んでください。

A.1.1 CSR ファイルに署名する

パブリック CA を使用して CSR に署名するには、セクション 3.2 の手順に従ってください。

内部 CA を使用して CSR に署名するには、セクション 3.3 の手順に従ってください。

証明書に自己署名するには、OpenSSL コマンドを使用します。

```
openssl req -x509 -nodes -days 100 -newkey rsa:2048 -keyout <keyname>.key –out
<certname>.crt
```

次に例を示します。

```
openssl req -x509 -nodes -days 100 -newkey rsa:2048 -keyout callbridge.key -
out callbridge.crt
```

は、call ブリッジ.key という名前の新しい秘密鍵と、call ブリッジ.crt という名前の(最終的な)証明書を生成します。

A.2 ECDSA 秘密鍵と CSR ファイルの生成

A.2.1 証明書リクエスト署名用のルート CA 証明書の生成

ルート秘密鍵を生成するには、次のコマンドを使用します。

```
openssl ecparam -genkey -name prime256v1 -out root.key
```

このコマンドは、楕円曲線 prime256v1 を使用して、「root.key」という名前の秘密鍵を生成します。

ECDSA ルート CA 証明書を生成するには、次のコマンドを使用します

```
openssl req -new -x509 -SHA256 -key root.key -out root.crt -days 3650
-
subj"/C=<country>/ST=<state>/L=<location>/O=<organization>/OU=<organiz
ational unit>/CN=<authorityname>"
```

次に例を示します。

```
Openssl req -new -x509 -SHA256 -key root.key -out root.crt -days 3650 -
subj"/C=UK/ST=London/L=London/O=Example/OU=/CN=example"
```

このコマンドは、3650 日間 (約 10 年間) 有効な「root.crt」という名前の自己署名証明書を生成します。

A.2.2 証明書リクエスト署名用の中間 CA 証明書の生成

中間証明書の秘密鍵を生成するには、次のコマンドを使用します。

```
openssl ecparam -genkey -name prime256v1 -out intermediate.key
```

このコマンドは、楕円曲線 prime256v1 を使用して、「intermediate.key」という名前の秘密鍵を作成します。

サーバ証明書署名要求（CSR）を生成するには

1. 次の属性で拡張機能ファイル（.conf）を作成します：

   ```
   [req]
   distinguished_name = req_distinguished_name
   req_extensions = v3_req
   prompt = no

   [req_distinguished_name]
   C = <country>
   ST = <state>
   L = <location>
   O = <organization>
   OU = <organization unit>
   CN = <commonname/hostname>

   [v3_req]
   basicConstraints = CA:TRUE, pathlen:0
   ```

2. 次のコマンドを使用してください。

   `openssl req -new -sha256 -key intermediate.key -out intermediate.csr <.conf file>`

   このコマンドは、中間 CA 秘密鍵を使用して、「intermediate.csr」という名前の CSR を作成します。

A.2.3 証明書リクエスト署名用のサーバー証明書の生成

サーバ証明書の秘密鍵を生成するには、次のコマンドを使用します。

`openssl ecparam -genkey -name prime256v1 -out server.key`

このコマンドは、楕円曲線 prime256v1 を使用して、「server.key」という名前の秘密鍵を作成します。**サーバー証明書署名リクエスト（CSR）を生成するには**

1. 次の属性で拡張機能ファイル（.conf）を作成します：

   ```
   [req]
   distinguished_name = req_distinguished_name
   req_extensions = v3_req
   prompt = no

   [req_distinguished_name]
   C = <country>
   ```

ST = <state>

L = <location>

O = <organization>

OU = <organization unit>

CN = <commonname/hostname>

[v3_req]

keyUsage=digitalSignature、keyEncipherment、dataEncipherment

extendedKeyUsage=serverAuth、clientAuth

subjectAltName=@alt_names

[alt_names]

DNS.1 = <DNS ホスト名>

2. 次のコマンドを使用してください。
   openssl req -new -SHA256 -key server.key -out server.csr -config <.conf file>
   ステップ 1 で作成した .conf ファイルを使用します。
   このコマンドは、サーバの秘密鍵を使用して、「server.csr」という名前の CSR を生成します。

中間 CA でサーバ証明書に署名するには

```
openssl x509 -req -in server.csr -CA intermediate.crt -CAkey
intermediate.key -CAcreateserial -out server.crt -days 365 -SHA256 -
extfile <.conf file> -extensions v3_req
```

このコマンドは、中間 CA 証明書と秘密鍵でサーバ CSR に署名します。これにより、365 日間有効な「server.crt」という名前の証明書が生成されます。「extensions.txt」ファイルで指定された拡張機能はサーバー証明書に含まれています。

証明書に自己署名するには、次のコマンドを使用します。

```
openssl x509 -req -in server.csr -CA root.crt -CAkey root.key -
CAcreateserial -out server.crt -days 365 -SHA256 -extfile <.conf file>
-extensions v3_req
```

このコマンドは、ルート CA 証明書と秘密鍵でサーバ CSR に署名します。これにより、365 日間有効な「server.crt」という名前の証明書が生成されます。「extensions.txt」ファイルで指定された拡張機能はサーバー証明書に含まれています。

A.3 データベースクラスタリングの証明書の作成

このセクションは、スケーラブルで復元力のある展開のみを対象としており、OpenSSL コマンドを使用してデータベース クラスタリング用の証明書を作成する方法について詳しく説明します。

メモ: 2.7 からデータベースクラスタ証明書は必須になるため、ミーティングサーバのデータベースクラスタリングのセットアップを容易にするために、pki コマンドを使用してデータベースクラスタ用の署名付き証明書を作成できます。詳細については、セクション 3.1.4 を参照してください。

データベースクラスタリング用に作成された証明書は、同じ認証局 (CA) によって署名されている必要があります。データベースのクラスタリングにはユーザがアクセスできないため、CA、キー、および証明書は OpenSSL を使用して内部で生成できます。

データベースクラスタは、クラスタ内のデータベースを保持または接続する各ミーティングサーバで設定された同じ CA によって署名されたクライアントとサーバ証明書を必要とします。証明書の使用を強制することで、クラスター全体の機密性と認証の両方が保証されます。

警告：データベースクラスタが、証明書を必要としない以前のバージョンの Meeting Server ソフトウェアを使用して、証明書なしで構成された場合、バージョン 2.7 へのアップグレード時に、データベースは停止し、証明書が構成されてデータベースクラスタが再作成されるまで到達不能になります。

メモ: OpenSSL は管理者権限で実行してください。

注：キー名と証明書名には、「.」または「_」を含めてはいけません。たとえば、**db01_ca** または **db01.ca** は使用できません。

手順は以下のとおりです。

1. CA を定義し、CA の公開/秘密鍵のペアと証明書を作成します。

   a. 指定した CA に対して、秘密鍵と証明書要求 (.csr) のペアを生成します。次の OpenSSL 構文を使用します

   ```
   openssl req -new -text -nodes -keyout <keyname>.key -out <certname>.csr
   -subj /C=<country>/ST=<state>/L=<location>
   /O=<組織>/OU=<組織部門>/CN=<authorityname>
   ```

   次に例を示します。

   ```
   openssl req -new -text -nodes -keyout db01ca.key -out db01ca.csr -subj
   /C=UK/ST=London/L=London/O=Example/OU=/CN=example
   ```

では、秘密キー **db01ca.key** と証明書署名リクエストファイル **db01ca.csr** を **-subj** に続く属性で定義された CA に対して作成します

b. ステップ 1a で生成された秘密鍵と証明書リクエスト (.csr) を使用して、CA の証明書を作成します。

```
openssl req -x509 -text -in db01ca.csr -key db01ca.key -out db01ca.crt -
days 3650
```

が証明書を作成します。**db01ca.crt**

2. 手順 1 で生成した CA 資格情報を使用して、データベースサーバとデータベースクライアントの秘密鍵と署名付き証明書を出力します。

a. データベース サーバの秘密鍵と証明書の要求 (.csr) を生成します。

```
openssl req -new -nodes -keyout <keyname>.key -out <certname>.csr -subj
/C=<国>/ST=<州>/L=<地域>
/O=<組織>/OU=<組織部門>/CN=<ノード名>
```

ここで、**nodename** はデータベースをホストしているサーバーの実際の名前です。次に例を示します。

```
openssl req -new -nodes -keyout db01server.key -out db01server.csr -subj
/C=英国/ST=ロンドン/L=ロンドン/O=Example/OU=/CN=server1
```

では、キー **db01server.key** と証明書署名リクエストファイル **db01server.csr** を作成します

b. データベースの CA 署名付き証明書を生成します。次に例を示します。

```
openssl x509 -req -CAcreateserial -in db01server.csr -CA db01ca.crt -
CAkey db01ca.key -out db01server.crt -days 3650
```

では、証明書 **db01server.crt** を作成します。

c. データベースの秘密鍵と証明書の要求 (.csr) を生成します。データベースクライアントの CommonName (CN) は「postgres」と同じでなければなりません。

```
openssl req -new -nodes -keyout <keyname>.key -out <certname>.csr
-subj /C=<country>/ST=<state>/L=<locality>/O=<organization>
/OU=<organizational unit>/CN=postgres
```

次に例を示します。

```
openssl req -new -nodes -keyout db01client.key -out db01client.csr -subj
/C=UK/ST=London/L=London/O=Example
/OU=/CN=postgres
```

では、キー **db01client.key** および証明書署名リクエストファイルを作成します **db01client.csr.**

d. データベースクライアント用の CA 署名付き証明書を生成します。次に例を示します。

```
openssl x509 -req -CAcreateserial -in db01client.csr -CA db01ca.crt -
CAkey db01ca.key -out db01client.crt -days 3650
```

が証明書を作成します `db01client.crt`

3. セクション 1.8 のステップに従って、データベースの証明書と秘密キーをアップロードして指定します。

   a. データベースをホストする各サーバは、次のキーと証明書をアップロードする必要があります。

      - データベースクラスタサーバ証明書 (手順 2 で生成)
      - データベースクラスタサーバキー (手順 2 で生成)
      - データベースクラスタークライアント証明書 (ステップ 2 で生成)
      - データベースクラスタクライアントキー (ステップ 2 で生成)
      - データベースクラスタ CA 証明書バンドル (ステップ 1 で生成)

   b. データベースと同じ場所にない各 Call Bridge は、次のキーと証明書をアップロードする必要があります。

      - データベースクラスタークライアント証明書 (ステップ 2 で生成)
      - データベースクラスタクライアントキー (ステップ 2 で生成)
      - データベースクラスタ CA 証明書バンドル (ステップ 1 で生成)

A.4 データベースクラスタリング用の ECDSA 証明書を作成する

クラスタデータベース用の ECDSA 証明書を生成するには、セクションに記載されているステップに従ってくださいA.2。しかし、DNS は以下で説明するように構成する必要があります。

データベースサーバーの場合、DNS はデータベースをホストするすべてのサーバーの**ノード名**に設定する必要があります。

データベースクライアントの場合、DNS を「postgres」に設定する必要があります。

A.5 証明書と秘密鍵のペアのインストール

Meeting Server への証明書と秘密キーペアのインストールの詳細については、第 4 章の手順に従ってください。

付録 B 証明書ファイルと秘密鍵で許可されている拡張子

以下の表は、証明書ファイルと秘密鍵に許可されているファイル拡張子を示しています。

表 5: 証明書ファイルで許可されている拡張子

内線番号	ファイル形式に関する情報
.pem	PEM はエンコード (ASCII base64) であり、ファイル拡張子として使用されます。通常、Unix ベースの Apache ウェブサーバからインポートされ、OpenSSL アプリケーションと互換性があります。 PEM 証明書ファイルは自動的に生成されます。一部の安全なウェブサイトでは、身元を認証するために、ユーザーに PEM ファイル（メールで送信される可能性があります）をアップロードするように求める場合があります。
.der	識別エンコード ルール (DER) はエンコードであると同時に、ファイル拡張子としても使用されます。DER 形式で作成された証明書のバイナリ表現が含まれます。通常、公開暗号化で X.509 証明書を保存するために使用されます。
.cer	ウェブサイトの信頼性を確認する、VeriSign や Thwate などのサードパーティの Certificate Authority が提供するセキュリティファイル。サーバ上にホストされている特定のウェブサイトの有効性を確認するために、ウェブサーバにインストールされます。証明書は、バイナリ DER または ASCII (Base64) PEM としてエンコードされる場合があります。
.crt	安全なウェブサイト ("https://" で始まる) が信頼性を確認するために使用する証明書。Verisign や Thawte などの企業から配布されています。証明書は、バイナリ DER または ASCII (Base64) PEM としてエンコードされる場合があります。 証明書ファイルは、ユーザが安全なサイトにアクセスすると、ウェブブラウザにより自動的に認識されます。証明書に保存されている情報は、ブラウザウィンドウ内のロックアイコンをクリックすることで表示できます。

表 6: 秘密鍵ファイルで許可されている拡張子

内線番号	ファイル形式に関する情報
.key	PKCS#8 公開鍵と秘密鍵の両方に使用されます。キーは、バイナリ DER または ASCII PEM としてエンコードされる場合があります。
.pem	キーが PEM (ASCII base64) を使用してエンコードされたことを示します。
.der	キーがバイナリ DER を使用してエンコードされたことを示します。

付録 C MMP PKI コマンド

MMP pki コマンドのリストを次に示します。

コマンド/例	説明/メモ
pki	現在の PKI の使用状況を表示します。
pki list	秘密鍵、証明書、証明書署名要求 CSR などの PKI ファイルを一覧表示します。
pki inspect <filename>	ファイルを検査し、ファイルが秘密鍵、証明書、CSR または不明のいずれであるかを表示します。証明書の場合、様々な詳細が表示されます。ファイルに証明書のバンドルが含まれている場合、バンドルの各要素に関する情報が表示されます。 PEM および DER 形式のファイルの両方が処理されます。
pki match <key> <certificate>	このコマンドは、指定されたキーとシステム上の証明書が一致するかどうかを確認します。プライベートキーと証明書は、1 つの使用可能な ID の 2 等分であり、サービス（例：callbridge）で使用される場合は一致する必要があります。
pki verify <cert> <cert bundle/CA cert> [<CA cert>] **pki verify server.pem bundle.pem rootca.pem pki verify server.pem bundle.pem**	証明書は Certificate Authority（CA）によって署名されている場合があり、CA は中間 CA 証明書の「証明書バンドル」と、場合によっては独自のファイルで CA 証明書を提供します。証明書が CA によって署名され、証明書バンドルを使用してこれを表明できることを確認するには、このコマンドを使用します。
pki unlock <key>	多くの場合、秘密鍵はパスワードで保護されています。Meeting Server で使用するには、キーのロックを解除する必要があります。このコマンドにより、ターゲットファイルのロックを解除するためのパスワードの入力が求められます。ロックされた名前は、同じ名前のロック解除されたキーによって置き換えられます

コマンド/例	説明/メモ
pki csr <key/cert basename> [<attribute>:<value>] **pki csr example CN:www.example.com OU:"My Desk" O:"My Office" L:"San Jose" ST:California C:US**	プライベート キー マテリアルの生成に関する要件を満たしている Cisco を信頼することに問題ないユーザーの場合、プライベートキーと関連する証明書署名要求（CSR）を生成できます。 <key/cert basename>は新しいキーと CSR を識別する文字列です (例えば、「new」は「new.key」と「new.csr」ファイルになります) CSR の属性は、コロン (":") で区切られた属性名と値のペアで指定できます。属性は次のとおりです。 CN：証明書に記載する CommonName です。CommonName はシステムの DNS 名である必要があります。 OU: 組織単位 O: 組織 L: 地域 ST: 州 C: 国 emailAddress：メールアドレス CSR ファイルは SFTP でダウンロードし、Certificate Authority（CA）に署名してもらうことができます。(または、CSR ファイルを「pki sign」コマンドで使用して、ローカルで証明書を生成することもできます。) 戻ったら SFTP 経由でアップロードする必要があります。その後、証明書として使用できます。 メモ：1.6.11 以降から、**pki csr <key/cert basename> [<attribute>:<value>]** は、属性として、subjectAltName を取得します。IP アドレスとドメイン名は、コンマ区切りリストの subjectAltName でサポートされています。次に例を示します。 **pki csr test1 CN:example.exampledemo.com subjectAltName:exampledemo.com** **pki csr test2 CN:example.exampledemo.com C:US L:Purcellville O:Example OU:Support ST:Virginia subjectAltName:exampledemo.com** **pki csr test3 CN:example.exampledemo.com C:US L:Purcellville O:Example OU:Support ST:Virginia subjectAltName:exampledemo.com, 192.168.1.25,exampledemo.com, server.exampledemo.com,join.exampledemo.com, test.exampledemo.com** 証明書のサイズと数をチェーンを最小限に抑えます。そうしないと、TLS ハンドシェイクの往復時間が長くなります。

コマンド/例	説明/メモ
pki selfsigned <key/cert basename> [<attribute>:<value>]	このコマンドを使用して、自己署名証明書を生成できます。 <key/cert basename> は、生成されるキーと証明書を示します。たとえば、「pki selfsigned new」は、「new.key and new.crt」（自己署名）を作成します。 CSR の属性は、コロン (":") で区切られた属性名と値のペアで指定できます。属性は次のとおりです。 CN：証明書に記載する CommonName です。CommonName はシステムの DNS 名である必要があります。 OU: 組織単位 O: 組織 L：地域 ST：州 C：国 emailAddress：メールアドレス CSR ファイルは SFTP でダウンロードし、Certificate Authority（CA）に署名してもらうことができます。戻ったら SFTP 経由でアップロードする必要があります。その後、証明書として使用できます。 証明書のサイズとチェーン内の証明書の数を最小限に維持します。維持しないと、TLS ハンドシェイクのラウンドトリップタイムが長くなります。
pki sign <csr/cert basename> <CA キー/証明書ベース名>	このコマンドは、<csr/cert basename> が識別する CSR に署名し、<CA key/cert basename> が識別する CA 証明書とキーを使用して、署名された同じベースネームを持つ証明書を作成します。 <csr/cert basename> ファイルおよび <CA key/cert basename> ファイルは、「pki csr」および「pki selfsigned」のコマンドでそれぞれ生成されます。
pki pkcs12-to-ssh <username> **pki pkcs12-to-ssh john**	PKCS#12 ファイルに保存されたパブリック SSH キーを使用できますが、最初に処理する必要があります。このコマンドは、<username>.pub という名前でアップロードされた PKCS#12 ファイルから使用可能な公開鍵を抽出します。pkcs#12 ファイルのパスワードの入力を求めるプロンプトが表示されます。完了後、pkcs#12 ファイルはパスワード保護なしの使用可能なキーに置換されます。 注：pkcs#12 ファイルに含まれるその他のデータはすべて失われます。 ジョンというユーザーにアップロードされた PKCS#12 ファイルである john.pub のキーは、コマンドを実行可能にすることで、使用できるようになります。

付録 D Cisco Meeting Server ウェブアプリを使用するように Web Bridge 3 を導入するための証明書と設定情報

D.1 ウェブブリッジ 3 を使用するためのミーティングサーバの設定

ウェブ ブリッジ 3 では、HTTPS ポートと C2W ポートを設定する必要があります。Web Bridge 3 を使用するように Meeting Server を設定するには：

1. MMP に SSH でログインします。
2. MMP で **`webbridge3`** コマンドを使用して、webbridge3 を設定します。webbridge3 の使用状況を表示するには、**`help webbridge3`** を入力します。

```
> help webbridge3

    Usage:
    webbridge3
    webbridge3 restart
    webbridge3 enable
    webbridge3 disable
    webbridge3 https listen <interface:port allowed list>
    webbridge3 https certs <key-file> <crt-fullchain-file>
    webbridge3 https certs none
    webbridge3 http-redirect (enable [port]|disable)
    webbridge3 c2w listen <interface:port allowed list>
    webbridge3 c2w certs <key-file> <crt-fullchain-file>
    webbridge3 c2w certs none
    webbridge3 c2w trust <crt-bundle>
    webbridge3 c2w trust none
    webbridge3 options <space-separated options>
    webbridge3 options none
    webbridge3 status
```

3. (オプション) HTTP 接続用のポートをセットアップします。このポートは、Web App が設定されているすべての Meeting Server インターフェイスに対して解放されます。着信 HTTP 接続は、到達したインターフェイスに一致する HTTPS ポートに自動的にリダイレ

クトされます。**webbridge3 http-redirect enable [port]** でポートを指定していない場合、デフォルトのポートは 80 になります。

4. HTTPS サービスがリッスンするポートを構成します。それをインターフェイス a のポート 443 でリッスンするように設定するには、次のコマンドを入力します。

   webbridge3 https listen a:443

5. HTTPS 証明書を設定します。これらはウェブブラウザに提示される証明書であるため、認証局によって署名され、ホスト名/目的などが一致する必要があります。(証明書ファイルは、エンドエンティティ証明書で開始し、ルート証明書で終了する証明書の完全なチェーンです。) 次のコマンドを入力します。

   webbridge3 https certs wb3-https.key wb3-https-fullchain.crt

6. C2W 接続を構成します。このアドレス/ポートには、Call Bridge からのみアクセスできるようにすることをお勧めします。次のコマンドは、それをインターフェイス a のポート 9999 に設定します。

   webbridge3 c2w listen a:9999

   ここではポート 9999 の例を使用しますが、ネットワーク上で利用可能なポート でもかまいません。443 とは異なり、固定ポートではありません。

7. C2W 接続証明書を構成します。C2W 接続に使用される SSL サーバ証明書を設定する必要があります。（証明書の要件については、以下の「C2W 接続をするための Call Bridge の設定 」を参照してください。詳細については、FAQ に記載されています。）

   webbridge3 c2w certs wb3-c2w.key wb3-c2w-fullchain.crt

8. Web Bridge 3 C2W サーバーでは、Call Bridge がクライアント証明書を提示することを期待しています。このサーバーは、次のコマンドで提供される信頼バンドルを使用して、Call Bridge を信頼するかどうかを確認します。

   webbridge3 c2w trust wb3-c2w-trust-bundle.crt

9. ここで Web Bridge 3 を有効にします。

   webbridge3 enable

D.2 C2W 接続を使用するための Call Bridge の設定

C2W 証明書は、Call Bridge と Web Bridge 3 の間の接続に使用されます。Call Bridge が Web Bridge 3 への C2W 接続を行うには、C2W 信頼ストアを指定して証明書を確認する必要があります、つまり、上記のステップ 7 で設定された Web Bridge 3 によって提示された証明書です。

1. MMPで **callbridge** コマンドを使用して Call Bridge の使用状況を表示します。表示するには、**help callbridge** と入力します。

   ```
   > help callbridge
   Configure CMS callbridge
   ```

   使用法：

   ```
   callbridge listen <interface allowed list>
   callbridge prefer <interface>
   callbridge certs <key-file> <crt-file> [<cert-bundle>]
   callbridge certs none
   callbridge trust c2w <bundle>
   callbridge trust c2w none
   callbridge add edge <ip address>:<port>
   callbridge del edge
   callbridge trust edge <trusted edge certificate bundle>
   callbridge trust cluster none
   callbridge trust cluster <trusted cluster certificate bundle>
   callbridge restart
   ```

2. Call Bridge の証明書を設定します。

   ```
   callbridge certs cert.key cert.crt
   ```

3. Web Bridge 3 により提示される SSL サーバ証明書を確認するために使用される C2W 信頼ストアを設定します（詳細については、FAQ を参照してください）。

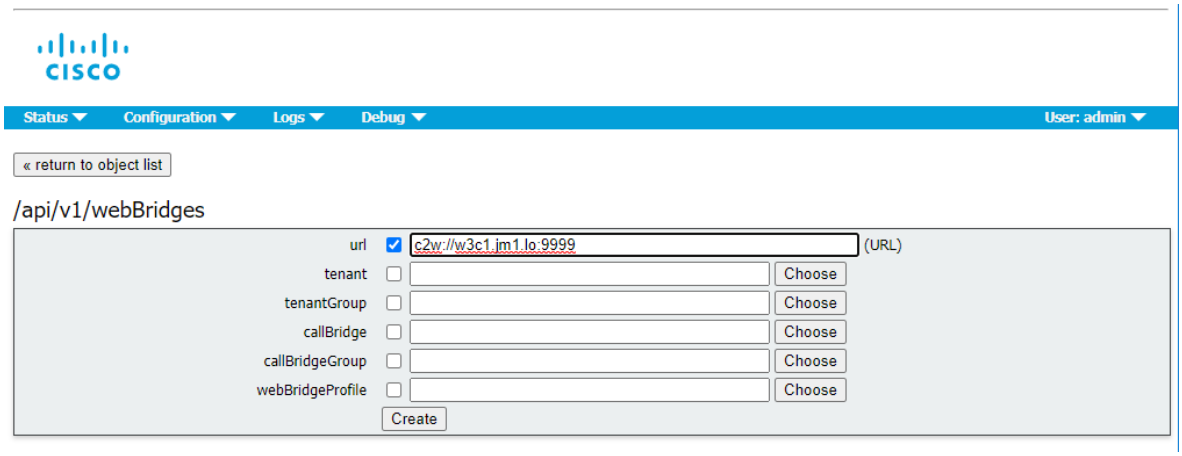
   ```
   callbridge trust c2w c2w-callbrige-trust-store.crt
   ```

4. ここで Call Bridge を再起動します。

   ```
   callbridge restart
   ```

5. ウェブ管理ユーザーインターフェイスに進み、**[設定（Configuration）] > [API]** を選択し、**/api/v1/webBridges** を選択して、下記のように、Web Bridge 3 の URL を実行中の callbridge REST API に登録します。URL プロトコルは、それが webbridge3 であることを示しています。つまり、URL に c2w:// プロトコルを指定することで、webbridge3 接続として処理されるようになります。

図 4: ウェブブリッジ 3 URL を Call Bridge API に登録する



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式、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。

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