モバイルおよびリモートアクセス(MRA)証明書の要件とATS履歴の理解

内容

はじめに

このドキュメントでは、モバイルおよびリモートアクセス用のCUCMでの証明書のアップロード要件について説明します。

バックグラウンド情報

Cisco ExpresswayはApache Traffic Server(ATS)を使用します。 トラバーサルソリューションでは、トラフィックサーバは非常に重要なコンポーネントであり、主に次の機能に使用されます。

- 証明書検証：MRAサービスに対して、Cisco Unified Communications Manager(CUCM)、IM & Presence、およびUnityサーバノードの証明書検証を実行します。
- プロキシ化とキャッシング：HTTP/HTTPSトラフィックに対して、高速でスケーラブルなキャッシングプロキシサーバとして機能します。

Expressway バージョン14.0.2

トラフィックサーバ(ATS)がMRAのプロビジョニング中にCUCMと通信する際に、「証明書検証」のわずかな適用が見られるようになります。

この要件は、「CSCvz45074」で説明されています。この要件では、Expresswayコアサーバ証明書に署名したルート証明書を、CUCM上でTomcat-TrustおよびCallmanager Trust(https://cdetsng.cisco.com/summary/#/defect/CSCvz45074)としてアップロードする必要があります。

- トラフィックサーバは証明書検証を実行します。

- X14.0.2リリースにアップグレードする前に、この証明書の要件が満たされていることを確認してください。

要件：Expressway-C証明書に署名した認証局(CA)チェーン（ルート+中間者）は、Unified Communications Manager(UCM)が非セキュアモードであっても、CUCMのtomcat-trustおよびCallManager-trustリストに追加する必要があります。

理由：Expresswayのトラフィックサーバサービスは、サーバUCMが要求するたびに証明書を送信します。これらの要求は、8443以外のポート（たとえば、ポート6971、6972など）で実行されているサービスに対するものです。 これにより、UCMが非セキュアモードであっても、証明書の検証が適用されます。詳細については、『Expressway経由のモバイルおよびリモートアクセス導入ガイド』を参照してください。

14.0.8より前のバージョンの動作

Expressway-Cとユニファイドコミュニケーションノード間のセキュアなHTTPS双方向接続を処理するExpressway-C上のトラフィックサーバが、リモートエンドから提示された証明書を検証しませんでした。MRA設定では、CUCM、IM&P、またはUnityサーバがConfiguration > Unified Communications > Unified CM servers/IM and Presence Service nodes/Unity Connection serversの順に選択して追加されたときに、TLS Verify Modeの設定でTLS証明書の検証を「On」にするオプションがあります。次のスクリーンショットに示されている設定オプションは、SAN内のFQDNまたはIP、証明書の有効性、および信頼できるCAによって署名されているかどうかを確認します。

また、同じCN名を持つ2つの証明書をExpressway信頼ストアにロードできないという既知の問題がありました。この制限により、次の2つの問題が発生しました。

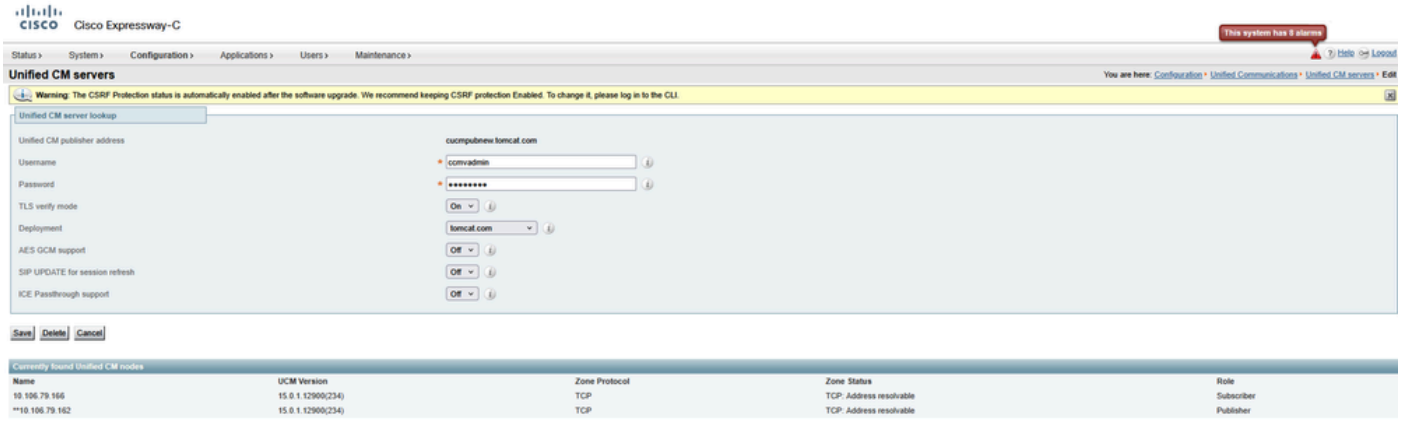
1. Expressway信頼ストアにCall Manager証明書をロードすることを選択した場合、CUCMの追加中にTLS検証「オン」が失敗します。

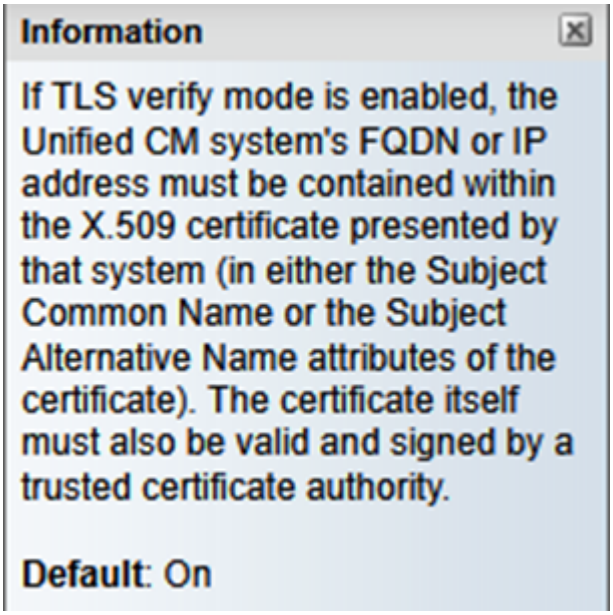
2: Expressway信頼ストアにTomcat証明書をロードすることを選択した場合、5061でのセキュアsip登録が失敗します。

この動作はCSCwa12894で文書化されています。

また、このTLS証明書検証チェックは、CUCM/IM&P/Unityサーバの検出時にのみ実行され、MRAクライアントプロビジョニング中には実行されません。

この設定の欠点は、追加したパブリッシャアドレスに対してのみ確認が行われることです。サブスクライバノード上の証明書が正しく設定されているかどうかは、パブリッシャノードのデータベースからサブスクライバノード情報（FQDNまたはIP）を取得するため、検証されません。





バージョン14.0.8以降での動作

X14.0.8バージョン以降、Expresswayサーバは、トラフィックサーバを介して行われるすべてのHTTPS要求に対してTLS証明書検証を実行します。これは、CUCM/IM&P/Unityノードの検出中にTLS検証モードが「オフ」に設定されている場合にも実行されることを意味します。検証が成功しない場合、TLSハンドシェイクが完了せず、要求が失敗します。その結果、冗長性、フェールオーバーの問題、または完全なログイン障害などの機能が失われる可能性があります。また、TLS検証モードを「On」に設定しても、すべての接続が正常に機能するとは限りません。これについては、後の例で説明します。

ExpresswayがCUCM/IM&P/Unityノードに対してチェックする正確な証明書については、『MRAガイド』のセクションを参照してください。

https://www.cisco.com/c/en/us/td/docs/voice_ip_comm/expressway/config_guide/X15-0/mra/exwy_b_mra-deployment-guide-x150.pdf

セクション

Certificate Requirements > Certificate Exchange Requirements

Expressway-CoreとCUCMの間で通信が行われる方法は次のように変更されているため、以下の点を確認する必要があります。

1. モバイルおよびリモートアクセス用にCA署名付き証明書を使用することをお勧めします。

2. 各Unified CMクラスタは、Expressway-C証明書を信頼する必要があります。各クラスタで、次のことを確認します。

- 混合モードが有効になっている場合：Expressway-C証明書をUnified CMのCallManager-trustおよびTomcat-trustストアにインストールする必要があります。
- 混合モードが無効の場合：Expressway-C証明書に署名するルートCA証明書をUnified CMのCallManager-trustおよびTomcat-trustストアにインストールする必要があります。次に、次のコマンドを再起動します。・ Tomcatサービス・ CallManagerサービス・ HAプロキシサービス（TomcatでTLSを使用している場合）。

Expressway-Coreで、次のアクションが実行されていることを確認します。

- Expressway-Cは、各Unified CMおよびIM and Presenceサービスクラスタによって提示される証明書を信頼する必要があります。

Expressway-Cの信頼ストアには、すべてのUCクラスタのUnified CMおよびIM and Presenceサービス証明書に署名するルートCA証明書が含まれている必要があります。

注:UCMが非セキュアモードで動作している場合でも、Expressway-C証明書の署名に使用されるすべてのルートおよび中間CA証明書、または完全なCAチェーンをCisco Unified Communications Manager(UCM)のTomcat-trustおよびCallManager-trustリストに追加していることを確認してください。

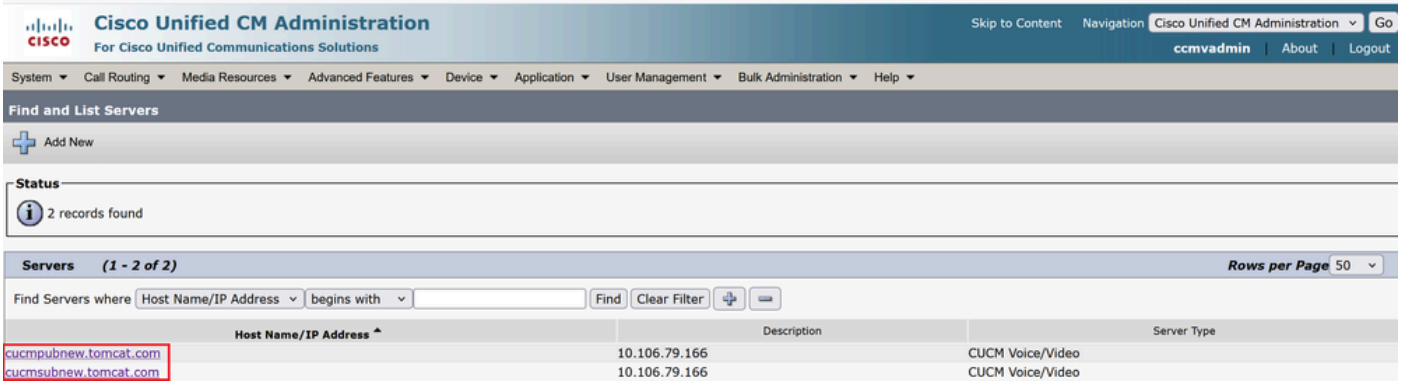
理由：Expresswayのトラフィックサーバサービスは、サーバ(UCM)が要求するたびに証明書を送信します。これらの要求は、8443以外のポート（たとえば、ポート6971、6972など）で実行されているサービスに対するものです。 これにより、UCMが非セキュアモードであっても、証明書の検証が適用されます。

System > Serverの下にCUCMアドレスを追加する方法は、Configuration > Unified Communications > Unified CM servers/IM and Presence Service nodesの下のExpresswayコアにCUCM/IMPを追加する際に非常に重要な役割を果たします。

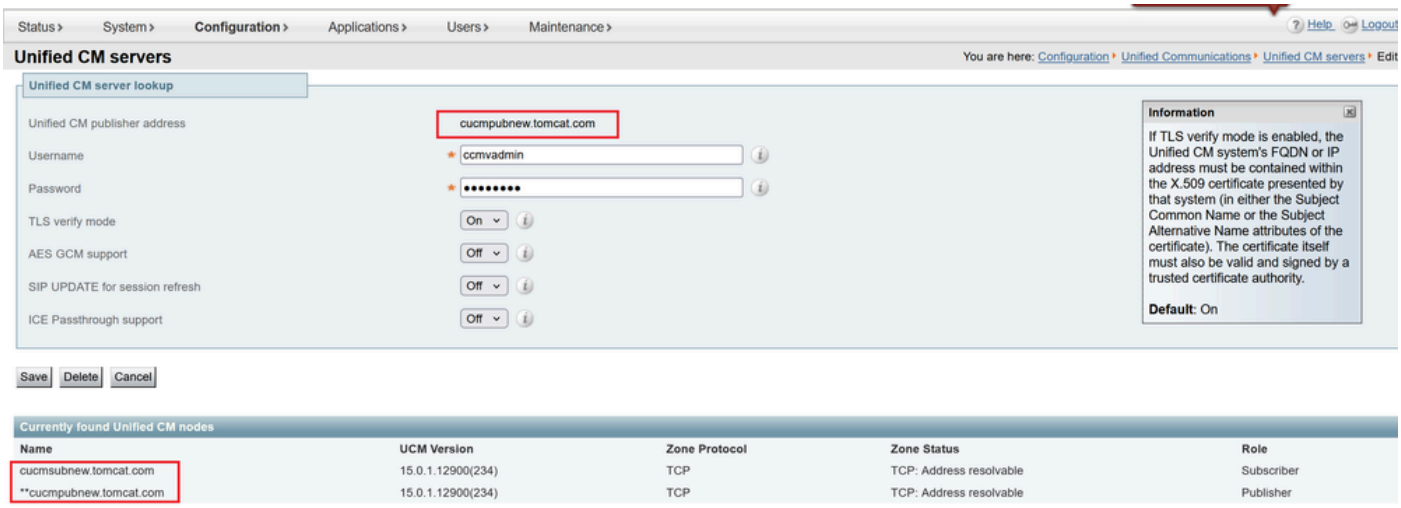
CUCMは、ホスト名やIPアドレスではなく、常にFQDNで追加する必要があります。CUCMがSystem > Serverの下にホスト名/IPアドレスとして追加されることを確認した場合、

tlsハンドシェイク中にTLS検証「On」が失敗し、CUCMクラスタがExpressway-Coreに追加されない

次の図に、ホスト名として追加されたCUCMを示します。



次の図は、TLS検証モード= ONのFQDNを使用してExpressway-Coreに追加されたCUCMを示します。

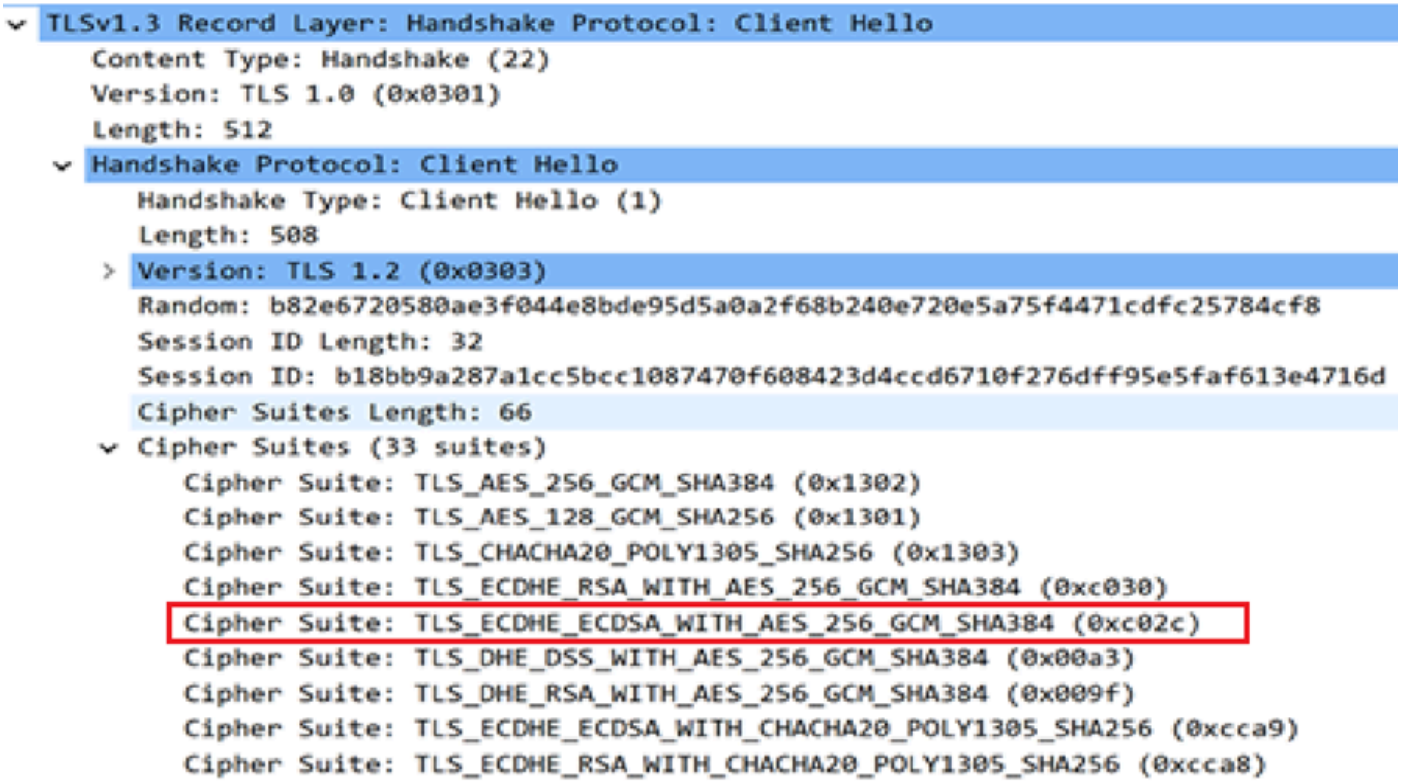


また、X14.2では、TLSハンドシェイク(client hello)中に異なる優先順位で暗号を提示する変更が導入されました。これはアップグレードパスに依存し、ソフトウェアアップグレード後に予期しないTLS接続が発生する原因となりました。TLSハンドシェイク中のアップグレードの前に、CUCMからCisco TomcatまたはCisco CallManager証明書を要求している可能性があります。ただし、アップグレード後に、ECDSAバリアント（RSAよりもセキュアな暗号バリアント）を要求しています。 Cisco Tomcat-ECDSA証明書またはCisco CallManager-ECDSA証明書は、別のCAによって署名することも、自己署名証明書のみで署名することもできます（デフォルト）。

この暗号の優先順位の変更は、Expressway X14.2.1のリリースノートで示されているようにアップグレードパスによって異なるため、常に関連するわけではありません。つまり、各暗号リストに対してECDHE-RSA-AES256-GCM-SHA384を付加するかどうかが、Maintenance > Security > Ciphersの順に選択されます。そうでない場合は、RSA暗号よりも新しいECDSA暗号が優先されます。存在する場合は、以前と同様にRSAの方が優先度が高い動作になります。

次のスクリーンショットは、Client helloのTLSネゴシエーションメッセージ中にExpresswayコアによってアドバタイズされ#IF赤いボックスのECDSA暗号で、サーバhelloのリモートレスポンダ(CUCM)によってTLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384が選択され、次の場合にTLSネゴシエーションが失敗します。

レスポンダからのルートCA証明書または実際のECDSA証明書。つまり、この場合、CUCMはExpressway信頼ストアにインストールされません。



または、ECDSAが優先されないようにExpressway暗号を変更することもできます。

1. GCM-Sha384オープンSSL文字列を付加してSIP暗号を変更する。

「ECDHE-RSA-AES256-GCM-SHA384:EECDH:EDH:HIGH:.....:!MD5:!PSK:!eNULL:!aNULL:!aDH」

2. 最後のプリファレンスで暗号を移動するには+を追加し、ECDSAを永久に無効にするには!を追加します。

暗号： "EECDH:EDH:HIGH:-AES256+SHA:!MEDIUM:!LOW:!3DES:!MD5:!PSK:!eNULL:!aNULL:!aDH:+ECDSA"

3. CUCMでECDSA証明書に署名したルートおよび中間CA証明書を追加するか、Expressway信頼ストア（場合によっては）でTomcat-ECDSA証明書を追加します。

ただし、暗号の優先順位の変更により、アップグレード後にMRAの展開が中断する可能性があるため、TACは前述の回避策を実行して、動作を再開する必要があります。

TLS 1.3の導入により、Wiresharkで交換される証明書を確認することはさらに困難になりました。

x15.3バージョンの動作

SIPインターフェイスの場合のみ、RSA暗号またはECDSA暗号を選択できます。

X15.xでは、TLS 1.3が適用されています。フィールドに表示されているように、RSAアルゴリズムは主にECDSA経由で選択されます。x15.2にアップグレードされたお客様は、次の製品を選択

できます

次のコマンドセットを使用したRSAアルゴリズムとECDSAアルゴリズムの間：

xConfiguration SIP Advanced TlsSignatureAlgoPrefRsa：オン/オフ

TlssignatureAlgoPrefRSAは、SIPインターフェイスにTLS 1.3が設定されている場合にのみ機能します

xConfiguration SIP Advanced SipTlsバージョン：「TLSv1.3」

注：現在、これはSIPインターフェイスにのみ適用されます。8443のTraffic ServerとTomcatの考慮事項は、前述のとおり変更されていません。

ExpresswayによってCUCMに「client hello」の間に送信される暗号スーツは、RSAが選択されたときに示されたとおりになります。

- 署名アルゴリズム： rsa_pss_rsae_sha512 (0x0806)
- 署名アルゴリズム：rsa_pss_rsae_sha384 (0x0805)
- 署名アルゴリズム： rsa_pss_rsae_sha256 (0x0804)
- 署名アルゴリズム：ecdsa_secp521r1_sha512(0x0603)
- 署名アルゴリズム：ecdsa_secp384r1_sha384 (0x0503)
- 署名アルゴリズム：ecdsa_secp256r1_sha256 (0x0403)

以前の設定は、CUCMのEnterprise Parameters > Security ParametersでTLS暗号に対して選択した設定と並行して機能します。

また、Expressway-CとCUCM間のTLS 1.3を介したTLSハンドシェイクが破損している場合、診断ログまたはPCAPに出力されるエラーはそれほど役に立ちません。TACの操作中にこれらのデバッグを有効にすると、コンポーネントがトラブルシューティングに明確なエラーを出力するようになります。

x構成ロガー開発者developer.trafficserver.httpレベル： "DEBUG"
x構成ロガー開発者developer.trafficserver.http_transレベル： "DEBUG"
x構成ロガー開発者developer.trafficserver.iocoreレベル： "DEBUG"
x構成ロガー開発者developer.trafficserver.sslレベル： "DEBUG"

Callmanagerが複数のサービスと1つの証明書を共有する場合の処理

CUCMで証明書を再利用すると、状況が少し変わります。

CUCM 14.0以降では、TomcatおよびTomcat ECDSA証明書をCall ManagerおよびCall Manager ECDSAとして再利用できます。

Tomcat証明書はCallmanager証明書として再利用できます。

Tomcat-ECDSA証明書は、Callmanager-ECDSA証明書として再利用できます。

これにより、生活が楽になります。

1. CUCMの複数のサービスが1つの証明書を使用するようになったため、証明書のコストが削減されました。

2. 証明書の管理が少ない。

3. Tomcat/CallmanagerまたはTomcat-ECDSA/Callmanager-ECDSA証明書を（何らかの理由で）Expressway-Core信頼ストアにアップロードする必要がある場合、アップロードする必要がある証明書は1つだけです。同じCN名の問題が発生しても問題はありません（このドキュメントで前述）。

注：証明書の再利用は、TomcatおよびTomcat-ECDSAがマルチSAN証明書である場合にのみ発生します。

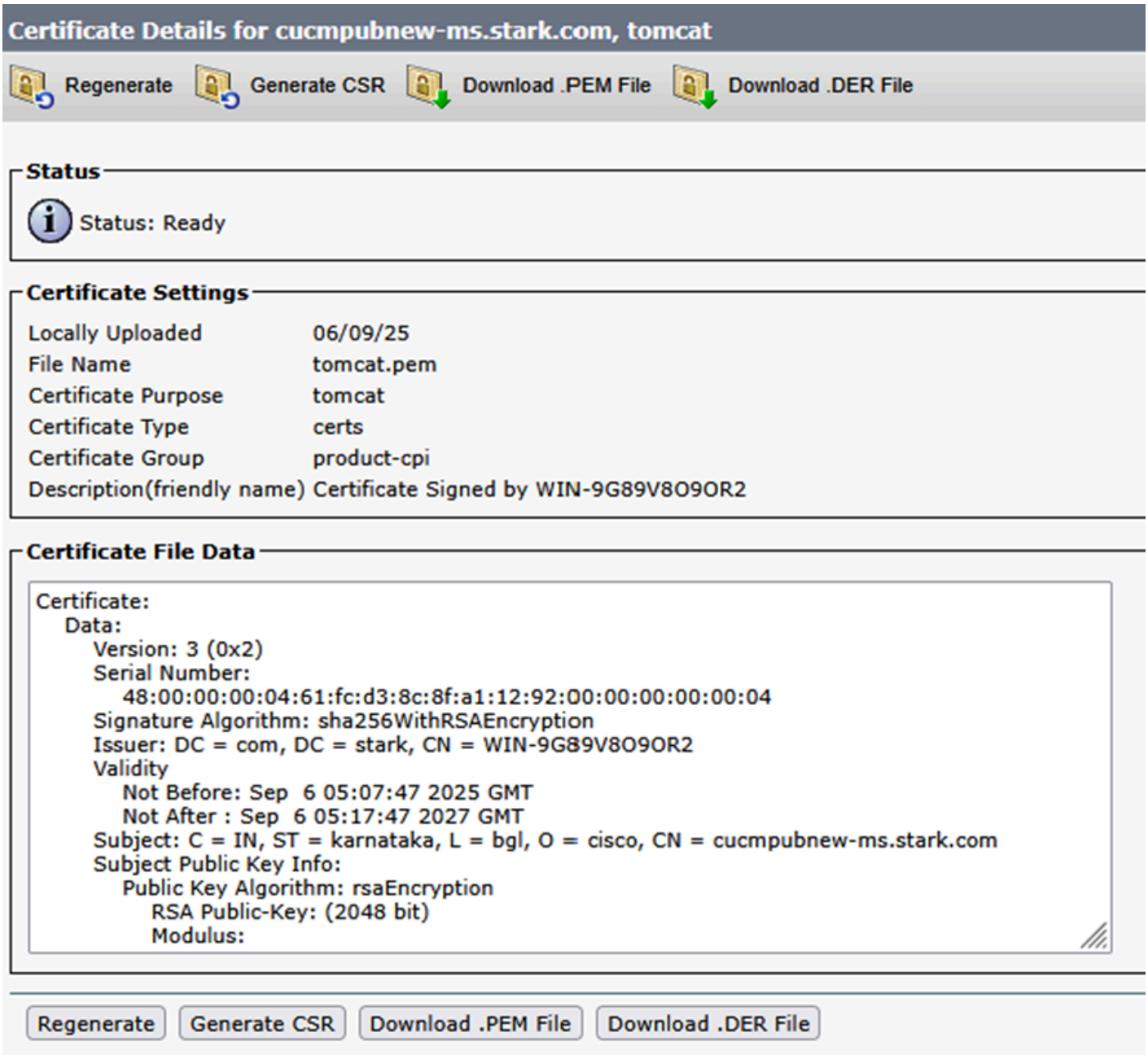
Post Reuse、Callmanager、およびCallmanager ECDSAサーバ証明書は、CUCM信頼ストアには表示されません。次のコマンドを実行することで、CLIから証明書の再利用を検証できます。

show cert own CallManager（登録ユーザ専用）

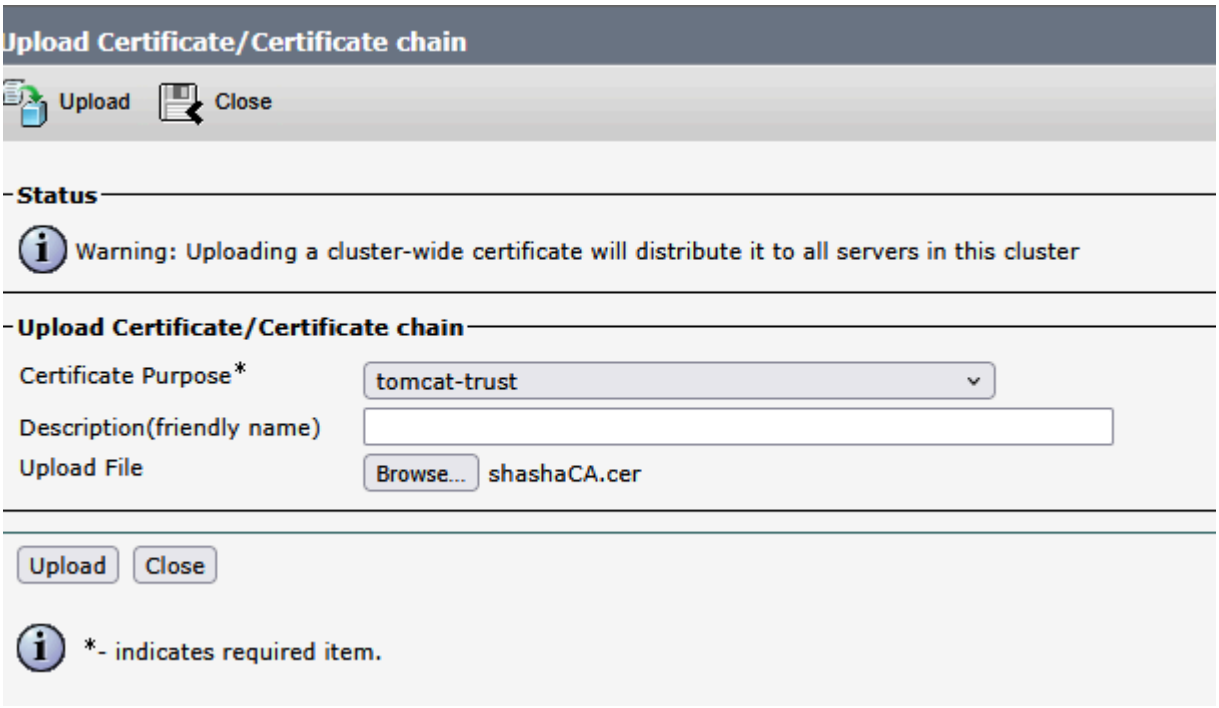
show cert own tomcat（隠しコマンド）

証明書を再利用する手順

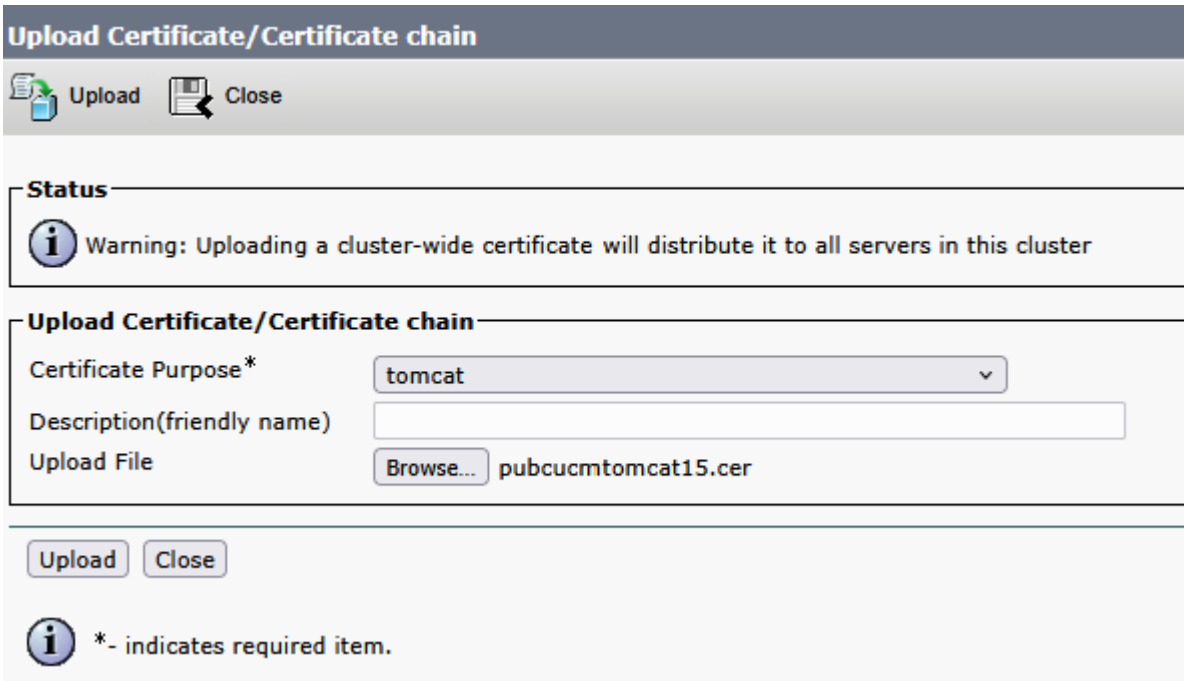
Tomcat CSR pub addを生成しています。



CUCMでTomcat証明書にTomcat-trustとして署名するCA証明書をアップロードします。



Tomcat証明書が署名されたら、パブリッシャにアップロードします。プロンプトに従って、関連するサービスを再起動します。



Tomcat証明書が署名されたら、パブリッシャにアップロードします。プロンプトに従って、関連するサービスを再起動します。

成功：証明書がアップロードされました。ディザスタリカバリのバックアップを実行して、アップロードされた証明書が最新のバックアップに含まれるようにします。
CLI「utils service restart Cisco Tomcat」をすべてのクラスタノード(UCM/IMP)で使用して、Cisco Tomcat Webサービスを再起動します。 CLI「utils service restart Cisco UDS Tomcat」お

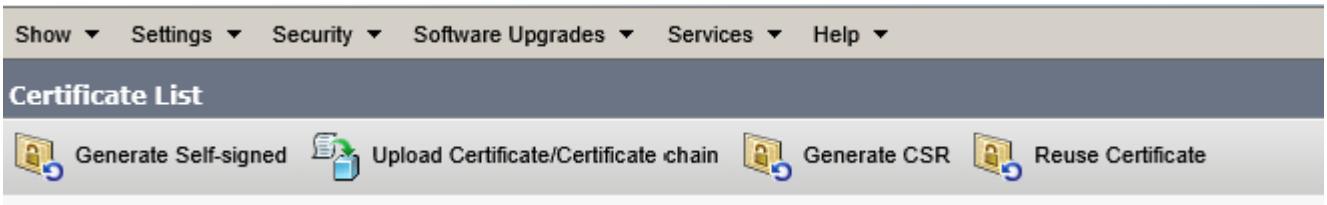
よび「utils service restart Cisco AXL Tomcat」をすべてのUCMクラスタノードで使用して、Cisco UDS TomcatおよびCisco AXL Tomcat Webサービスを再起動します。また、パブリッシャノードでCisco DRF Masterサービスと Cisco DRF Localサービスを再起動します。サブスクライバノードでCisco DRF Localサービスだけを再起動します。

Tomcat証明書がCAによって署名されます。

tomcat	cucmpubnew-ms.stark.com 51dc40f400000000000b	signed IdentityCA-signed	RSA	Multi-server(SAN)	RICKY200-TMS-CA	10/23/2027	Certificate Signed by RICKY200-TMS-CA

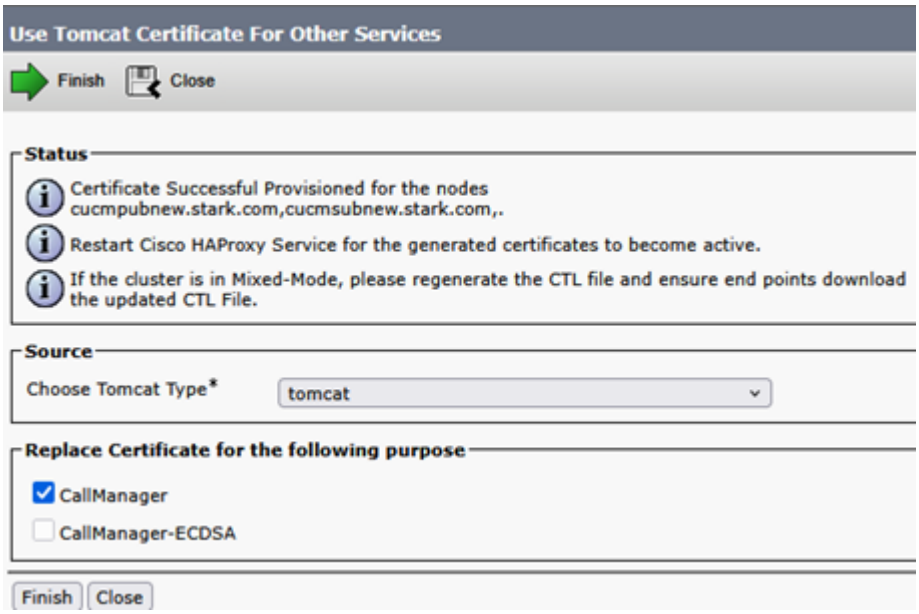
ここで、Tomcat証明書をCallmanager証明書として再利用します。

Reuse Certificateをクリックします。



ドロップダウンからTomcatを選択し、Callmanager証明書を確認します。

[Finish] をクリックします。



Tomcat証明書がCallmanager証明書として再利用されるようになりました。これはCLIから検証できます。

Callmanager証明書シリアル番号(SN):56:ff:6c:71:00:00:00:00:00:0d

```
admin:show cert own CallManager
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            56:ff:6c:71:00:00:00:00:00:0d
        Signature Algorithm: sha384WithRSAEncryption
        Issuer: DC = com, DC = RICKY200, CN = RICKY200-TMS-CA
        Validity
            Not Before: Oct 24 08:44:34 2025 GMT
            Not After : Oct 24 08:54:34 2027 GMT
        Subject: C = IN, ST = karnataka, L = bgl, O = cisco, CN = cucmpubnew-ms.
tomcat.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:b4:a6:fa:8f:9a:c3:32:02:74:fa:e9:92:30:de:
                    6e:3b:70:cd:d7:4e:64:e4:71:04:fe:17:80:0d:5b:
                    44:d1:7f:00:63:69:4a:5c:1a:1b:75:0c:1a:d6:ce:
                    10:3f:01:e2:d0:f1:75:33:57:b7:0a:71:e1:60:d1:
                    89:3c:e8:a4:8c:3e:30:69:4d:4e:98:da:b8:5d:dd:
                    23:8c:4d:69:90:69:9d:43:74:84:20:a8:9f:45:dc:
                    5a:aa:7b:c8:d1:d0:6f:05:13:d8:99:58:0e:49:7b:
Press <enter> for 1 line, <space> for one page, or <q> to quit
```

Tomcat証明書SN:56:ff:6c:71:00:00:00:00:00:0d

```
admin:show cert own tomcat
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            56:ff:6c:71:00:00:00:00:00:0d
        Signature Algorithm: sha384WithRSAEncryption
        Issuer: DC = com, DC = RICKY200, CN = RICKY200-TMS-CA
        Validity
            Not Before: Oct 24 08:44:34 2025 GMT
            Not After : Oct 24 08:54:34 2027 GMT
        Subject: C = IN, ST = karnataka, L = bgl, O = cisco, CN = cucmpubnew-ms.tomcat.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (2048 bit)
                Modulus:
                    00:b4:a6:fa:8f:9a:c3:32:02:74:fa:e9:92:30:de:
                    6e:3b:70:cd:d7:4e:64:e4:71:04:fe:17:80:0d:5b:
                    44:d1:7f:00:63:69:4a:5c:1a:1b:75:0c:1a:d6:ce:
                    10:3f:01:e2:d0:f1:75:33:57:b7:0a:71:e1:60:d1:
                    89:3c:e8:a4:8c:3e:30:69:4d:4e:98:da:b8:5d:dd:
                    23:8c:4d:69:90:69:9d:43:74:84:20:a8:9f:45:dc:
                    5a:aa:7b:c8:d1:d0:6f:05:13:d8:99:58:0e:49:7b:
Press <enter> for 1 line, <space> for one page, or <q> to quit
```

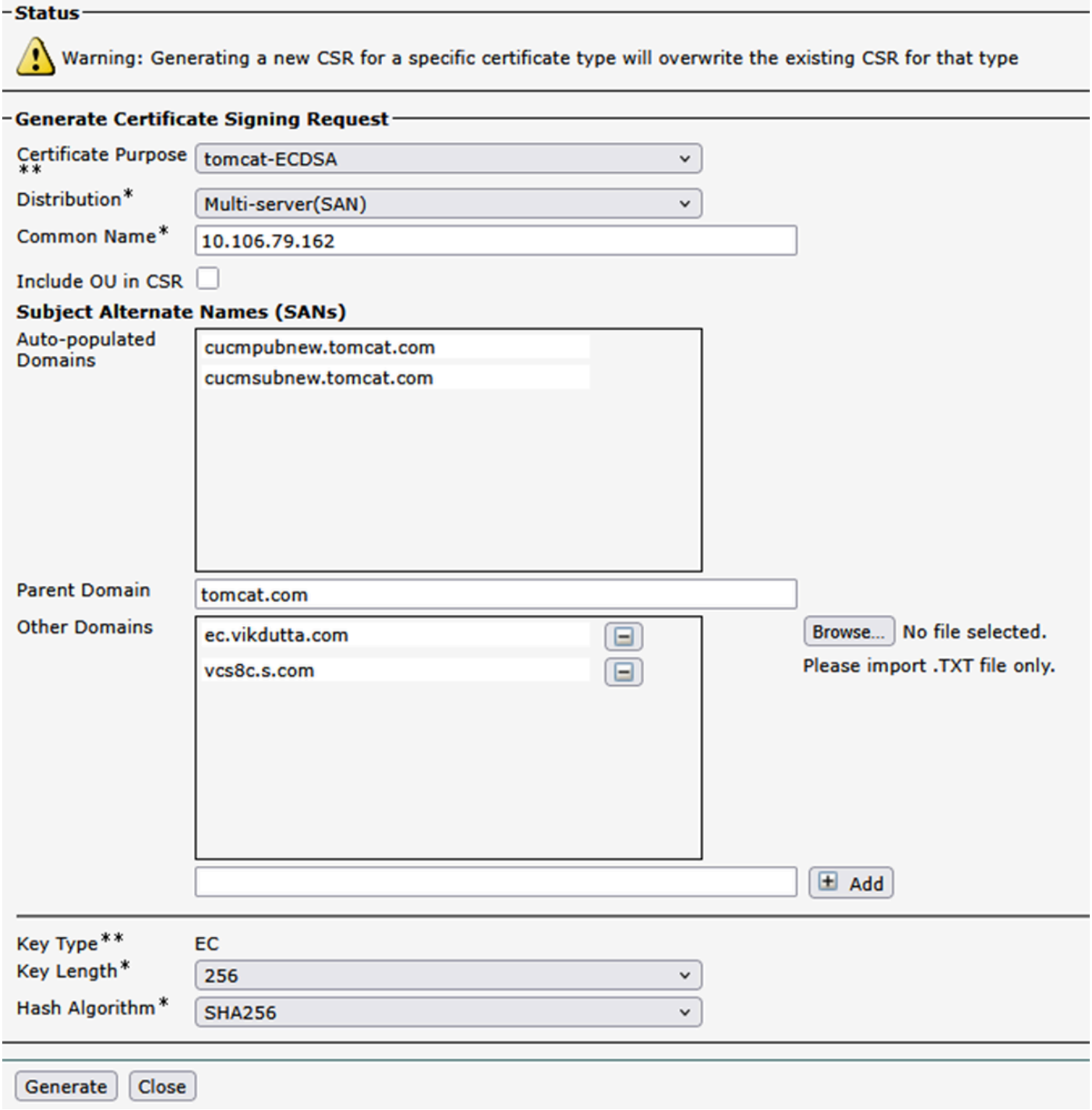
サブスクライバで同じ手順を実行します。

ECDSA証明書に署名して、Callmanager-ECDSAとして再利用できるようにします。

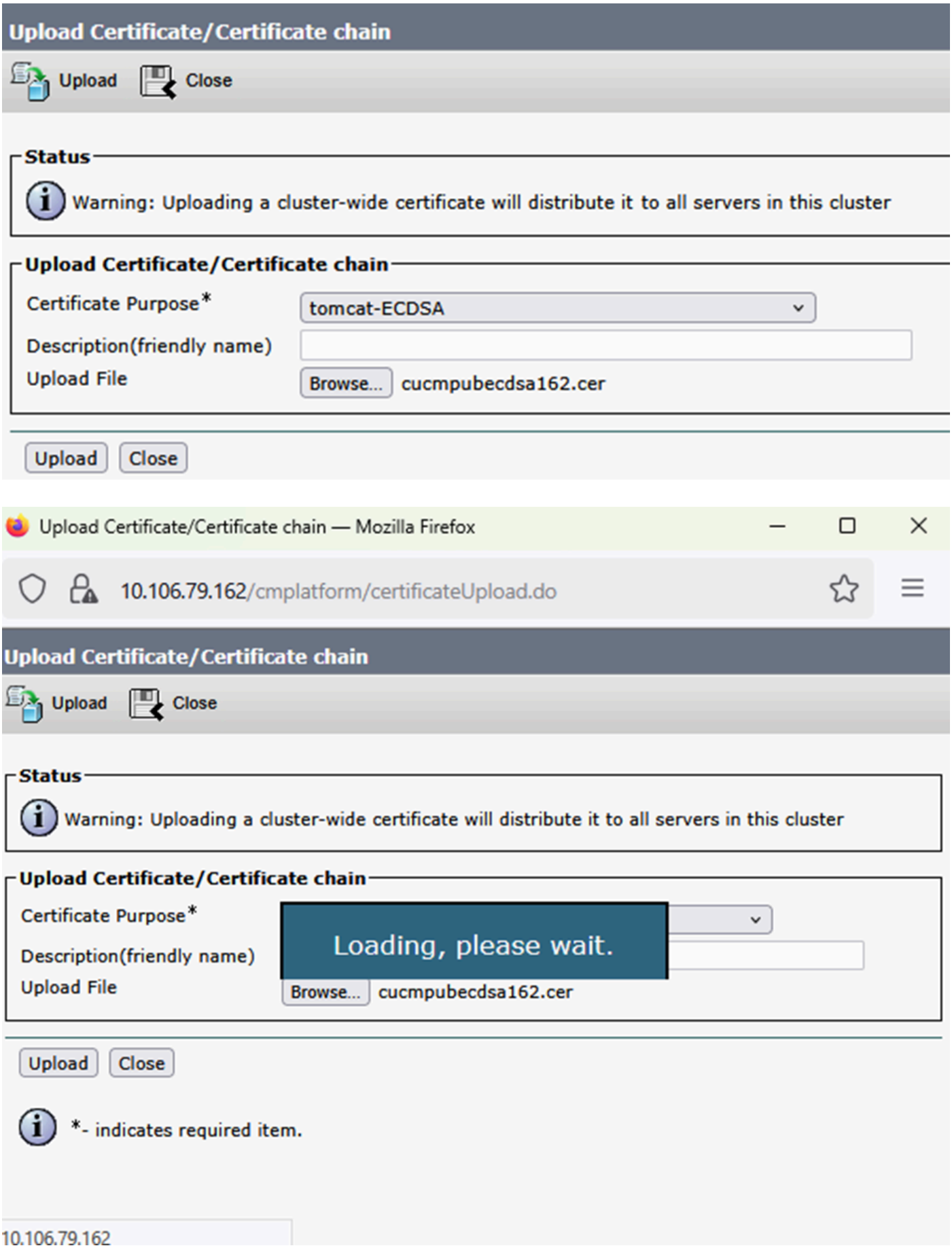
現在のTomcat-ECDSA証明書は自己署名です。

tomcat	10.106.79.162_5aceb67f00000000000f	Identity	CA-signed	RSA	Multi-server(SAN)	RICKY200-TMS-CA	10/25/2027	Certificate Signed by RICKY200-TMS-CA
tomcat-ECDSA	cucmpubnew-EC.tomcat.com_4b404cd202f547cabf8a6db78c715d46	Identity	self-signed	EC	cucmpubnew.tomcat.com	cucmpubnew-EC.tomcat.com	10/23/2030	self-signed certificate generated by system

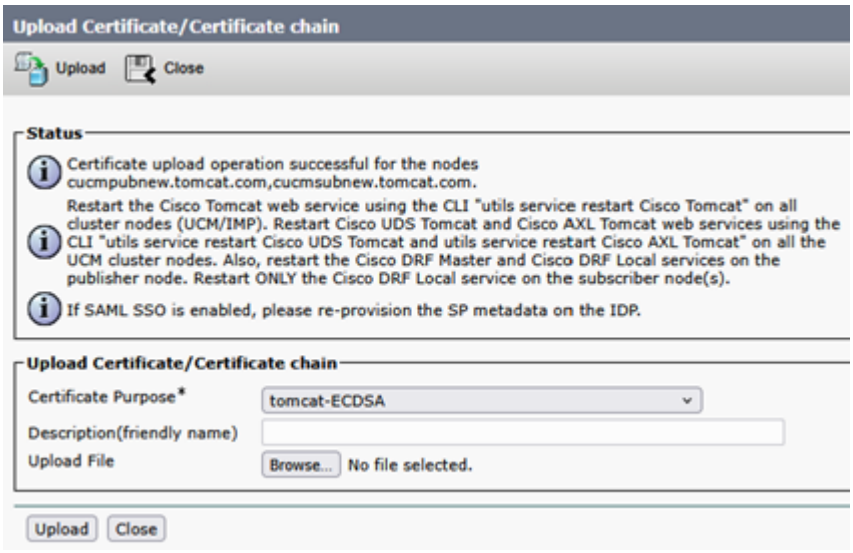
Tomcat-ECDSA証明書のマルチSAN CSRに署名します。



CSRを使用して証明書に署名し、アップロードします。

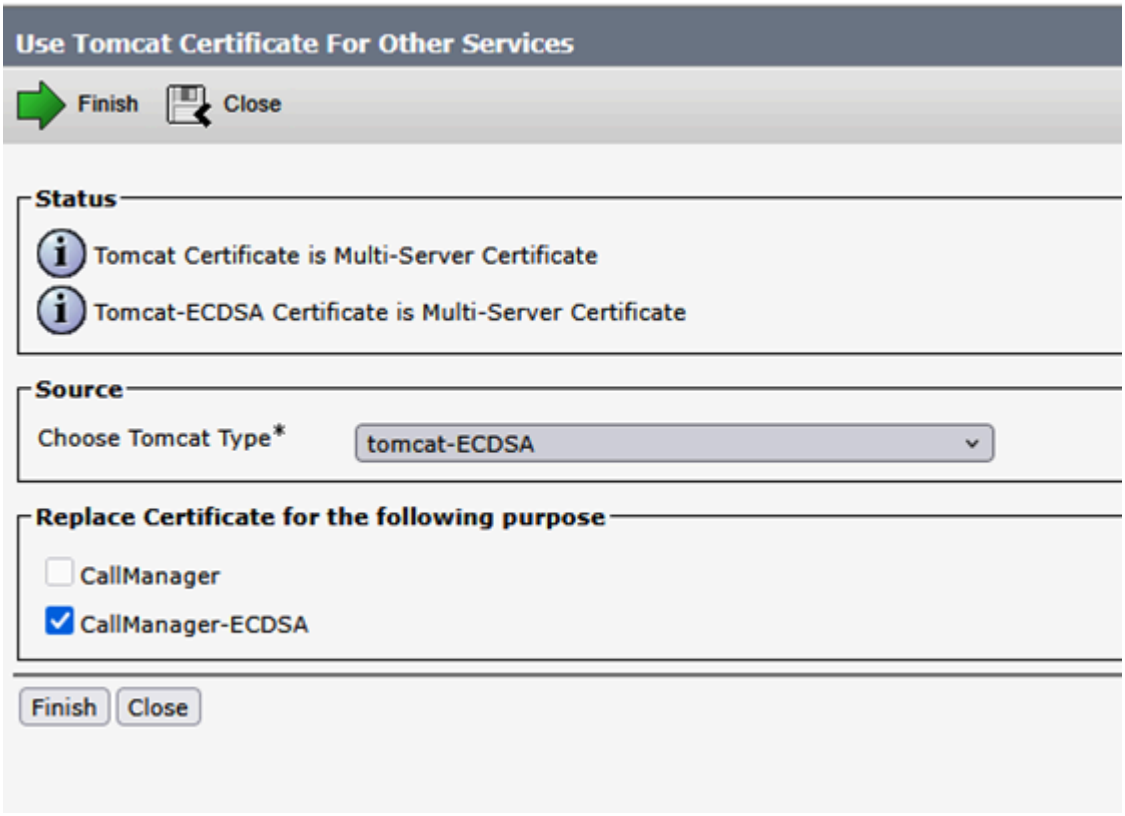


アップロードに成功しました。プロンプトに従って、関連するサービスを再起動します。

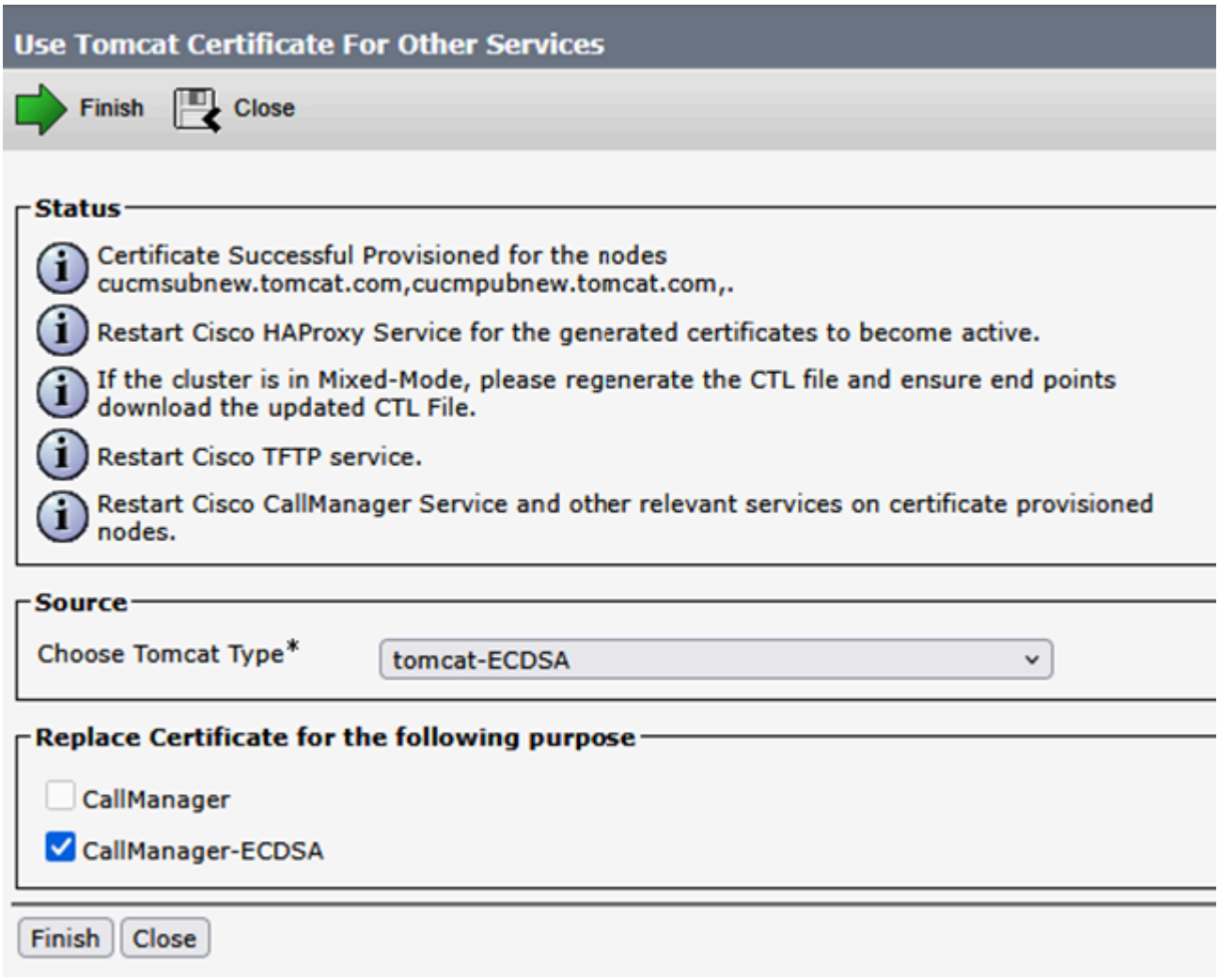


CAによって署名されたTomcatおよびTomcat-ECDSA。

ここで、Tomcat-ECDSAをCallmanager-ECDSA証明書として再利用します。



アップロードに成功しました。プロンプトに従って、関連するサービスを再起動します。



CLIから証明書を確認します。

Callmanager-ECDSA証明書SN:2f:00:00:00:38:80:be:cc:a8:a1:8e:8f:23:00:00:00:00:00:38

```
admin:show cert own CallManager-ecdsa
Invalid Certificate Name. Certificate Not Found.

admin:show cert own CallManager-Ecdsa
Invalid Certificate Name. Certificate Not Found.

admin:show cert own tomcat-ECDSA
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            2f:00:00:00:38:80:be:cc:a8:a1:8e:8f:23:00:00:00:00:00:38
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: DC = com, DC = bgluclab, CN = bgluclab-WIN-DC-01-CA
        Validity
            Not Before: Oct 25 06:46:37 2025 GMT
            Not After : Oct 25 06:46:37 2026 GMT
```

Tomcat-ECDSA証明書SN: 2f:00:00:00:38:80:be:cc:a8:a1:8e:8f:23:00:00:00:00:00:38

```
admin:show cert own tomcat-ECDSA
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            2f:00:00:00:38:80:be:cc:a8:a1:8e:8f:23:00:00:00:00:00:38
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: DC = com, DC = bgluclab, CN = bgluclab-WIN-DC-01-CA
        Validity
            Not Before: Oct 25 06:46:37 2025 GMT
            Not After : Oct 25 06:46:37 2026 GMT
        Subject: C = IN, ST = karnataka, L = bgl, O = cisco, CN = cucmpubnew-EC-ms.tomcat.com
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
```

現在、TomcatとCallmanagerのサービス用のTomcat証明書と、Tomcat-ECDSAとCallmanager-ECDSAのサービス用のTomcat-ECDSAの2つのサービス用の証明書を使用しているため、Expressway信頼ストア（アップロードが必要な場合）に証明書をアップロードする手間が軽減されます。

MRA用のExpressway-CoreでUCMを追加する際に、TLS検証を「オン」にすると、これまで以上に簡単になりました。1つのTomcat証明書CAまたはサーバ証明書を追加するだけで、この作業が実行されます（CallmanagerとTomcatサービス間で証明書が共有されるようになったため）。

x14.2以降へのアップグレードが原因でモバイルリモートアクセスが停止している場合は、この包括的なドキュメントで「問題のトラブルシューティング」も参照できます。

Apacheトラフィックサーバのバージョン履歴

サーバのバージョンを確認するには、rootにログインして ~ # /apache2/bin/httpd -vを実行します。

Expressway x8.11.4
サーババージョン：Apache/2.4.34(Unix)
構築サーバ：2018年11月12日19:04:23

Expressway x12.6

サーババージョン：Apache/2.4.43(Unix)
構築済みサーバ：2020年5月26日18:27:21

Expressway x14.0.8

サーババージョン：Apache/2.4.53(Unix)
構築済みサーバ：2022年5月4日08:52:57

Expressway x15.3

サーババージョン：Apache/2.4.62(Unix)
構築済みサーバ：2025年7月16日12:10:19

翻訳について
シスコは世界中のユーザにそれぞれの言語でサポート コンテンツを提供するために、機械と人による翻訳を組み合わせて、本ドキュメントを翻訳しています。ただし、最高度の機械翻訳であっても、専門家による翻訳のような正確性は確保されません。シスコは、これら翻訳の正確性について法的責任を負いません。原典である英語版（リンクからアクセス可能）もあわせて参照することを推奨します。