

Solución de problemas de verificación de certificado de servidor de tráfico de Expressway para servicios MRA introducidos por CSCwc69661 / CSCwa25108

Contenido

[Introducción](#)

[Prerequisites](#)

[Requirements](#)

[Componentes Utilizados](#)

[Antecedentes](#)

[Cadena de CA de confianza](#)

[Comprobación de SAN o CN](#)

[Cambio de comportamiento](#)

[Versiones inferiores a X14.2.0](#)

[Versiones de X14.2.0 y superiores](#)

[Solucionar escenarios](#)

[1. La CA que firmó el certificado remoto no es de confianza](#)

[2. La dirección de conexión \(FQDN o IP\) no está incluida en el certificado](#)

[Cómo validarlo fácilmente](#)

[Solución](#)

Introducción

Este documento describe el cambio de comportamiento en las versiones de Expressway de X14.2.0 y posteriores vinculadas al Id. de bug Cisco [CSCwc6961](#) o al Id. de bug Cisco [CSCwa25108](#).

Prerequisites

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

- configuración básica de Expressway
- configuración básica de MRA

Componentes Utilizados

La información de este documento se basa en Cisco Expressway en la versión X14.2 y posteriores.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Si tiene una red en vivo, asegúrese de entender el posible impacto de cualquier comando.

Antecedentes

Con este cambio de comportamiento marcado por el Id. de bug Cisco [CSCwc69661](#) o ID de bug de Cisco [CSCwa25108](#), el servidor de tráfico de la plataforma Expressway realiza la verificación de certificados de los nodos de Cisco Unified Communication Manager (CUCM), Cisco Unified Instant Messaging & Presence (IM&P) y del servidor Unity para los servicios Mobile and Remote Access (MRA). Este cambio puede provocar errores de inicio de sesión de MRA después de una actualización en su plataforma de Expressway.

El protocolo seguro de transferencia de hipertexto (HTTPS) es un protocolo de comunicación segura que utiliza la seguridad de la capa de transporte (TLS) para cifrar la comunicación. Crea este canal seguro mediante el uso de un certificado TLS que se intercambia en el intercambio de señales TLS. De esta manera, tiene dos propósitos: autenticación (para saber a quién se conecta la persona remota) y privacidad (cifrado). La autenticación protege frente a ataques de intrusos y la privacidad evita que los atacantes intercepten y manipulen la comunicación.

La verificación de TLS (certificado) se realiza a la vista de la autenticación y le permite asegurarse de que se ha conectado a la parte remota correcta. La verificación consta de dos elementos individuales:

1. Cadena de autoridad certificadora de confianza (CA)
2. Nombre alternativo del sujeto (SAN) o nombre común (CN)

Cadena de CA de confianza

Para que Expressway-C confíe en el certificado que envía CUCM / IM&P / Unity, debe poder establecer un vínculo desde ese certificado a una entidad de certificación (CA) de nivel superior (raíz) en la que confíe. Este vínculo, una jerarquía de certificados que vincula un certificado de entidades a un certificado de CA raíz, se denomina cadena de confianza. Para poder verificar dicha cadena de confianza, cada certificado contiene dos campos : Emisor (o 'Emitido por') y Asunto (o 'Emitido para').

Los certificados de servidor, como el que CUCM envía a Expressway-C, tienen en el campo "Asunto" su nombre de dominio completo (FQDN) en el CN:

```
Issuer: DC=lab, DC=vngtp, CN=vngtp-ACTIVE-DIR-CA
Subject: C=BE, ST=Flamish-Brabant, L=Diegem, O=Cisco, OU=TAC, CN=cucm.vngtp.lab
```

Ejemplo de certificado de servidor para CUCM.cucm.vngtp.lab. Tiene el FQDN en el atributo CN del campo Asunto junto con otros atributos como País (C), Estado (ST), Ubicación (L), ... También podemos ver que el certificado del servidor es entregado (emitido) por una CA llamada vngtp-ACTIVE-DIR-CA.

Las CA de nivel superior (CA raíz) también pueden emitir un certificado para identificarse. En dicho certificado de CA raíz, vemos que el emisor y el sujeto tienen el mismo valor :

Issuer: DC=lab, DC=vngtp, CN=vngtp-ACTIVE-DIR-CA
Subject: DC=lab, DC=vngtp, CN=vngtp-ACTIVE-DIR-CA

Es un certificado emitido por una CA raíz para identificarse.

En una situación típica, las CA raíz no emiten directamente certificados de servidor. En su lugar, emiten certificados para otras CA. Estas otras CA se denominan CA intermedias. A su vez, las CA intermedias pueden emitir directamente certificados de servidor o certificados para otras CA intermedias. Podemos tener una situación en la que un certificado de servidor es emitido por la CA 1 intermedia, que a su vez obtiene un certificado de la CA 2 intermedia y así sucesivamente. Hasta que finalmente la CA intermedia obtiene su certificado directamente de la CA raíz :

Server certificate :

Issuer: DC=lab, DC=vngtp, CN=vngtp-intermediate-CA-1 Subject: C=BE, ST=Flamish-Brabant, L=Diegem, O=Cisco, OU=TAC, CN=cucm.vngtp.lab

Intermediate CA 1 certificate :

Issuer: DC=lab, DC=vngtp, CN=vngtp-intermediate-CA-2
Subject: DC=lab, DC=vngtp, CN=vngtp-intermediate-CA-1

Intermediate CA 2 certificate :

Issuer: DC=lab, DC=vngtp, CN=vngtp-intermediate-CA-3
Subject: DC=lab, DC=vngtp, CN=vngtp-intermediate-CA-2

...

Intermediate CA n certificate :

Issuer: DC=lab, DC=vngtp, CN=vngtp-ACTIVE-DIR-CA
Subject: DC=lab, DC=vngtp, CN=vngtp-intermediate-CA-n

Root CA certificate :

Issuer: DC=lab, DC=vngtp, CN=vngtp-ACTIVE-DIR-CA
Subject: DC=lab, DC=vngtp, CN=vngtp-ACTIVE-DIR-C

Ahora, para que Expressway-C confíe en el certificado de servidor que envía CUCM, debe poder generar la cadena de confianza desde ese certificado de servidor hasta un certificado de CA raíz. Para que esto ocurra, necesitamos cargar el certificado de CA raíz y también todos los certificados de CA intermedios (si los hay, lo que no es el caso si la CA raíz hubiera emitido directamente el certificado de servidor de CUCM) en el almacén de confianza de Expressway-C.

Nota: Aunque los campos Emisor y Asunto son fáciles de crear en la cadena de confianza de una manera legible por las personas, CUCM no utiliza estos campos en el certificado. En su lugar, utiliza los campos 'Identificador de clave de autoridad X509v3' e 'Identificador de clave de asunto X509v3' para crear la cadena de confianza. Esas claves contienen identificadores para los certificados que son más precisos que para utilizar los campos Asunto/Emisor : puede haber 2 certificados con los mismos campos Asunto/Emisor, pero uno de ellos ha caducado y otro sigue siendo válido. Ambos tendrían un identificador de clave de asunto X509v3 diferente, por lo que CUCM aún puede determinar la cadena de confianza correcta.

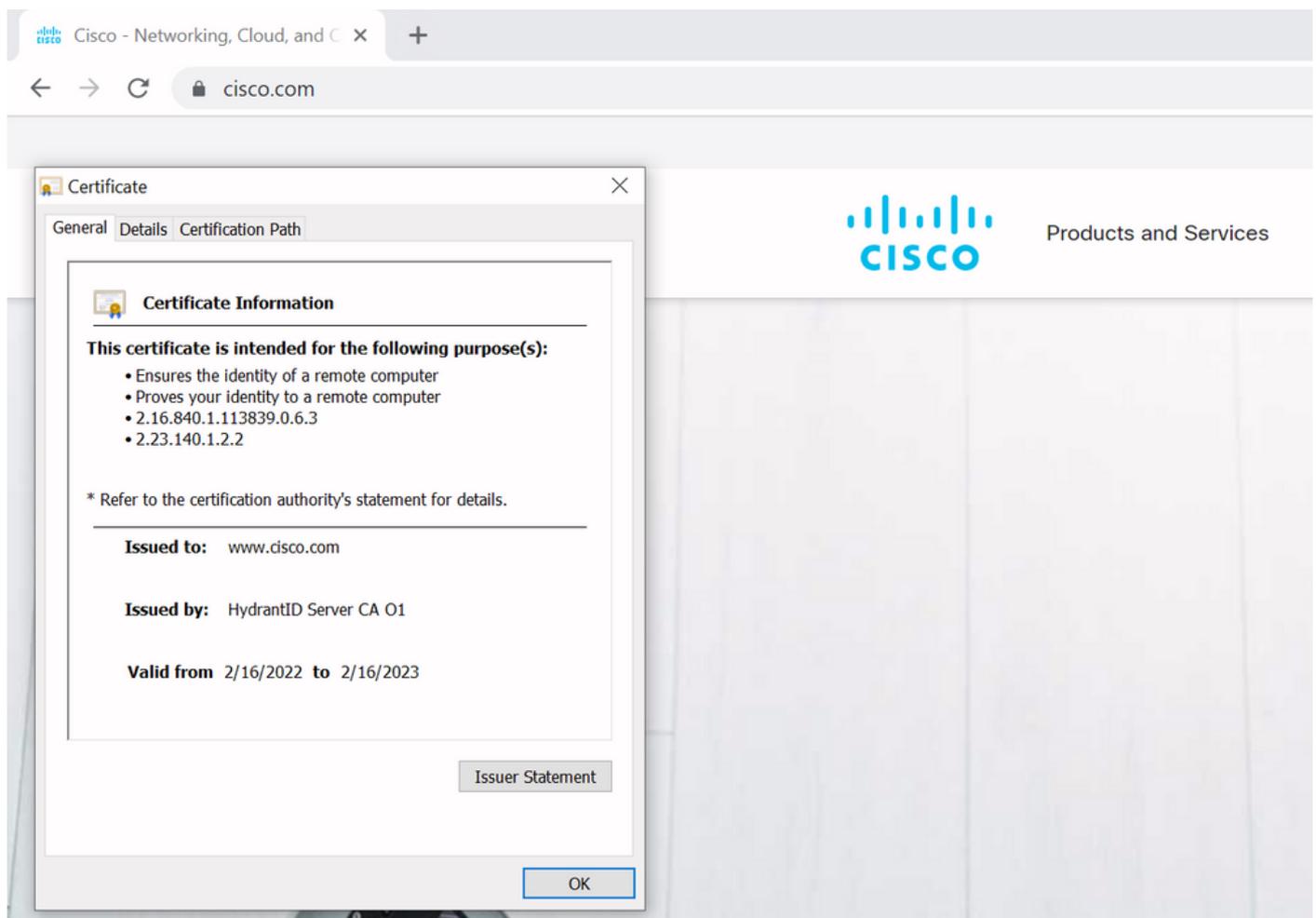
Este no es el caso de Expressway, aunque según el identificador de error de Cisco [CSCwa12905](#) y no es posible cargar dos certificados diferentes (autofirmados, por ejemplo) en el almacén de confianza de Expressway que tienen el mismo nombre común (CN). La manera de corregir esto es utilizar certificados firmados por CA o utilizar nombres comunes diferentes para ello o ver que utiliza siempre el mismo certificado (potencialmente a través de la función de certificado de reutilización en CUCM 14).

Comprobación de SAN o CN

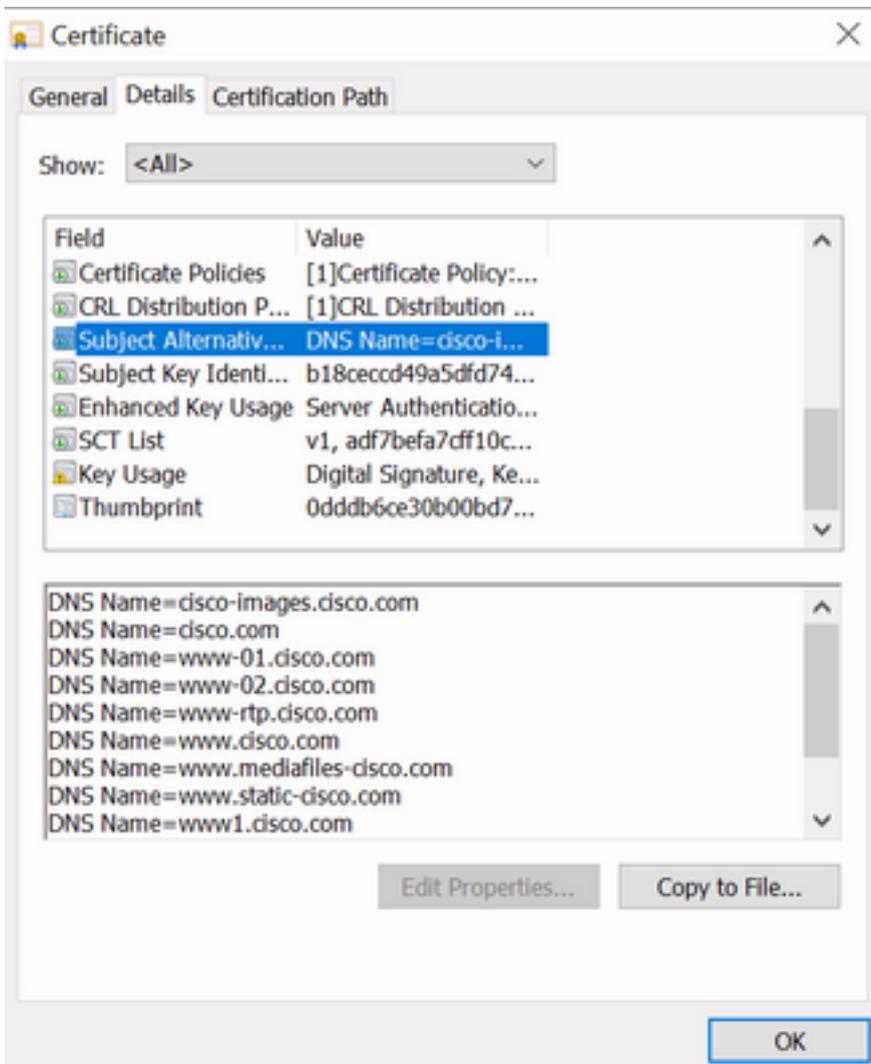
En el paso 1 se desprotege el almacén de confianza; sin embargo, cualquier persona que tenga

un certificado firmado por una CA en el almacén de confianza será válida en ese momento. Esto claramente no es suficiente. Por lo tanto, hay una comprobación adicional que valida que el servidor al que se conecta específicamente es el correcto. Lo hace basándose en la dirección para la que se formuló la solicitud.

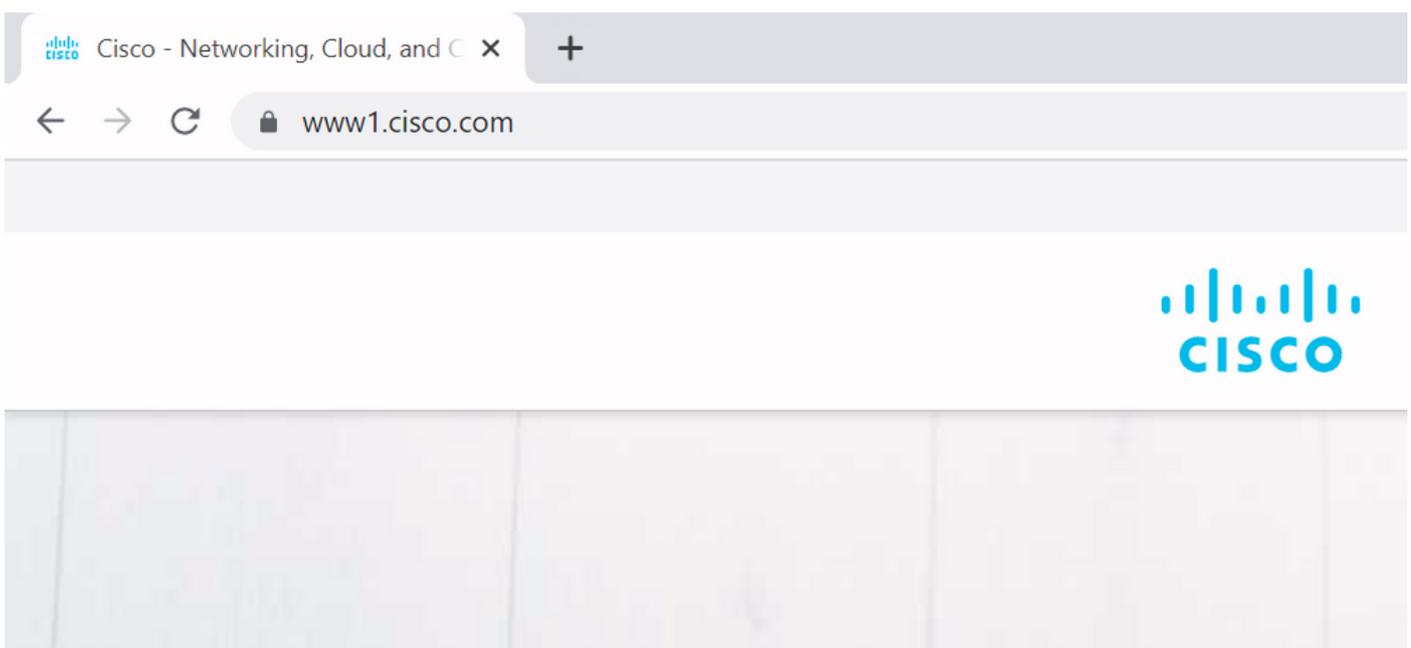
El mismo tipo de operación ocurre en su navegador, así que vamos a ver esto a través de un ejemplo. Si navega hasta <https://www.cisco.com> verá un icono de candado junto a la URL que ingresó y significa que se trata de una conexión confiable. Esto se basa tanto en la cadena de confianza de la CA (desde la primera sección) como en la comprobación de SAN o CN. Si abrimos el certificado (a través del navegador haciendo clic en el icono de candado), verá que el nombre común (que aparece en el campo 'Emitido para:') está configurado en www.cisco.com y que corresponde exactamente a la dirección a la que deseábamos conectarnos. De esta manera, podemos estar seguros de que nos conectamos al servidor correcto (porque confiamos en la CA que firmó el certificado y que realiza la verificación antes de entregar el certificado).



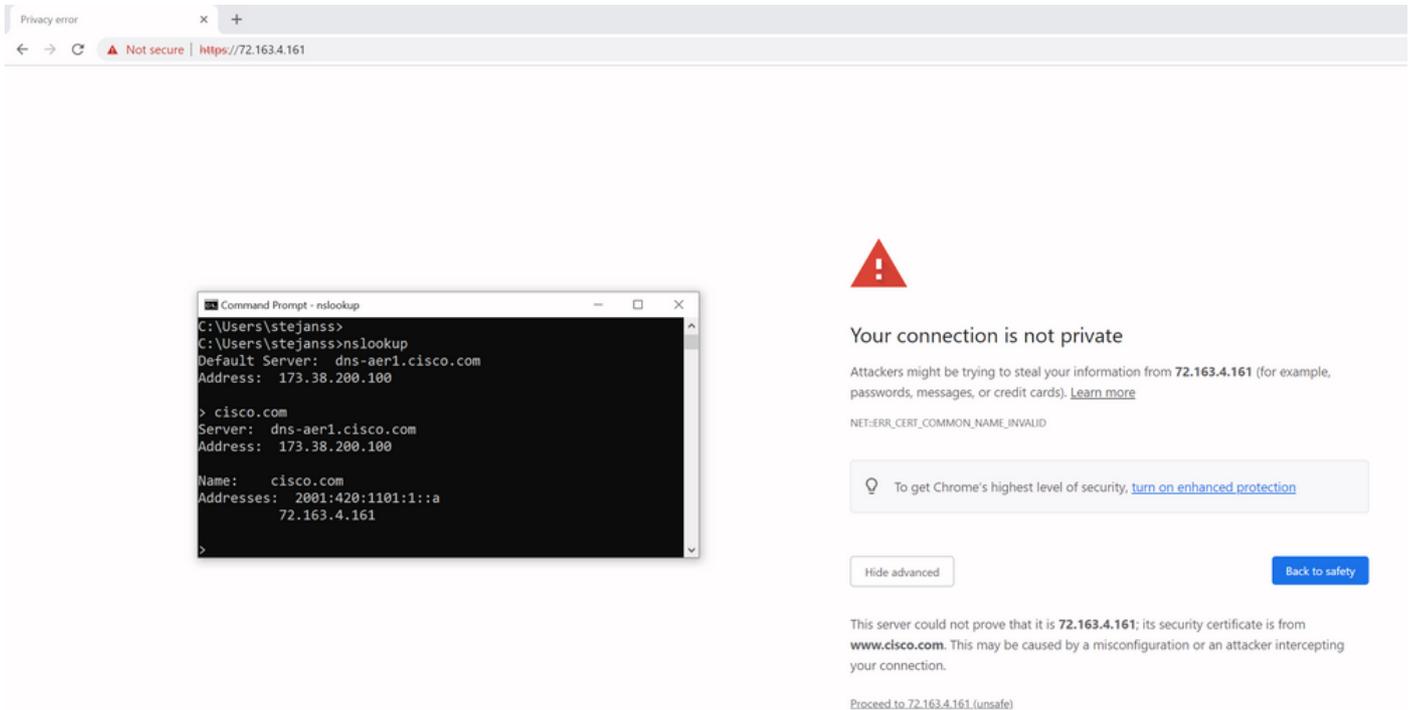
Cuando observamos los detalles del certificado y, en particular, las entradas de SAN, vemos que se repite lo mismo, así como algunos otros FQDN:



Esto significa que cuando solicitáramos la conexión a <https://www1.cisco.com>, por ejemplo, también se mostraría como una conexión segura porque está contenida en las entradas de SAN.



Sin embargo, cuando no navegamos a <https://www.cisco.com> sino directamente a la dirección IP (<https://72.163.4.161>), no aparece una conexión segura porque sí confía en la CA que la firmó, pero el certificado que se nos presentó no contiene la dirección (72.163.4.161) que usamos para conectarnos hacia el servidor.



En el navegador, puede omitir esto, sin embargo, es una configuración que puede habilitar en las conexiones TLS que no permite una omisión. Por lo tanto, es importante que sus certificados contengan los nombres CN o SAN correctos que la parte remota planea utilizar para conectarse a ella.

Cambio de comportamiento

Los servicios MRA dependen en gran medida de varias conexiones HTTPS a través de Expressway hacia los servidores CUCM / IM&P / Unity para autenticarse correctamente y recopilar la información correcta específica para el cliente que inicia sesión. Esta comunicación suele ocurrir en los puertos 8443 y 6972.

Versiones inferiores a X14.2.0

En versiones inferiores a X14.2.0, el servidor de tráfico de Expressway-C que administra esas conexiones HTTPS seguras no verificó el certificado presentado por el extremo remoto. Esto podría llevar a ataques de intrusos. En la configuración de MRA, hay una opción para la verificación de certificados TLS mediante la configuración del 'Modo de verificación TLS' a 'Activado' cuando agregaría servidores CUCM / IM&P / Unity en **Configuración > Unified Communications > Servidores Unified CM / nodos IM and Presence Service / Servidores Unity Connection**. La opción de configuración y el cuadro de información relevante se muestran como ejemplo, lo que indica que sí verifica el FQDN o la IP en la SAN, así como la validez del certificado y si está firmado por una CA de confianza.



Unified CM servers You are here: [Configuration](#)

Unified CM server lookup	
Unified CM publisher address	cucmpub.vngtp.lab
Username	* administrator i
Password	* i
TLS verify mode	On i
Deployment	Default deployment i
AES GCM support	Off i
SIP UPDATE for session refresh	Off i
ICE Passthrough support	Off i

Save Delete Cancel

Information x

If TLS verify mode is enabled, the Unified CM system's FQDN or IP address must be contained within the X.509 certificate presented by that system (in either the Subject Common Name or the Subject Alternative Name attributes of the certificate). The certificate itself must also be valid and signed by a trusted certificate authority.

Default: On

Esta comprobación de verificación del certificado TLS solo se realiza cuando se detectan los servidores CUCM / IM&P / Unity y no en el momento en que se consultan los distintos servidores durante el inicio de sesión de MRA. Un primer inconveniente de esta configuración es que sólo la comprueba para la dirección del editor que agrega. No valida si el certificado de los nodos del suscriptor se ha configurado correctamente, ya que recupera la información del nodo del suscriptor (FQDN o IP) de la base de datos del nodo del editor. Un segundo inconveniente de esta configuración también es que lo que se anuncia a los clientes MRA como la información de conexión puede ser diferente de la dirección del editor que se ha colocado en la configuración de Expressway-C. Por ejemplo, en CUCM, en **System > Server** podría anunciar el servidor con una dirección IP (10.48.36.215, por ejemplo) y los clientes de MRA lo utilizan (a través de la conexión de Expressway con proxy); sin embargo, podría agregar CUCM en Expressway-C con el FQDN de cucm.steven.lab. Por lo tanto, suponga que el certificado de Tomcat de CUCM contiene cucm.steven.lab como entrada de SAN pero no la dirección IP; a continuación, la detección con

'TLS Verify Mode' establecido en 'On' se realiza correctamente, pero las comunicaciones reales de los clientes de MRA pueden dirigirse a un FQDN o IP diferente y, por lo tanto, no pasar la verificación de TLS.

Versiones de X14.2.0 y superiores

A partir de la versión X14.2.0, el servidor de Expressway realiza la verificación de certificado TLS para cada solicitud HTTPS que se realiza a través del servidor de tráfico. Esto significa que también realiza esto cuando el 'Modo de verificación de TLS' se establece en 'Desactivado' durante la detección de los nodos CUCM / IM&P / Unity. Cuando la verificación no tiene éxito, el intercambio de señales TLS no se completa y la solicitud falla, lo que puede llevar a la pérdida de funcionalidad como problemas de redundancia o conmutación por fallas o fallas de inicio de sesión completas, por ejemplo. Además, con 'TLS Verify Mode' establecido en 'On', no garantiza que todas las conexiones funcionen correctamente como se describe en el ejemplo posterior.

Los certificados exactos que verifica Expressway hacia los nodos CUCM / IM&P / Unity son como se muestra en la sección de la [guía MRA](#).

Aparte de la verificación predeterminada de TLS, también hay un cambio introducido en X14.2 que podría anunciar un orden de preferencia diferente para la lista de cifrado, que depende de su trayectoria de actualización. Esto puede causar conexiones TLS inesperadas después de una actualización de software porque puede suceder que antes de la actualización solicitó el certificado de Cisco Tomcat o Cisco CallManager de CUCM (o cualquier otro producto que tenga un certificado independiente para el algoritmo ECDSA) pero que después de la actualización solicita la variante ECDSA (que es la variante de cifrado más seguro en realidad que RSA). Los certificados de Cisco Tomcat-ECDSA o Cisco CallManager-ECDSA podrían estar firmados por una CA diferente o simplemente ser certificados autofirmados (el valor predeterminado).

Este cambio en el orden de preferencia de cifrado no siempre es relevante para usted, ya que depende de la ruta de actualización, como se muestra en las [notas de la versión de](#) Expressway X14.2.1. En resumen, puede ver en **Mantenimiento > Seguridad > Cifras** para cada una de las listas cifradas si antepone "ECDHE-RSA-AES256-GCM-SHA384:" o no. Si no es así, prefiere el cifrado ECDSA más reciente sobre el cifrado RSA. Si es así, entonces tiene el comportamiento anterior con RSA que tiene la preferencia más alta.

Cipher Preferences - ECDSA Cipher Preference Over RSA

ECDSA certificates are preferred over RSA.



Important

The following points lists the various upgrade path(s) that are mandatory for upgrading ciphers.

1. When upgrading from version lower than 14.0 to 14.2, the ECDSA would be preferred. If you prefer RSA certificates over ECDSA, then prefix the cipher string with "ECDHE-RSA-AES256-GCM-SHA384:" using either Web User Interface (**Maintenance > Security > Ciphers**) or CLI command (**xConfiguration Ciphers**).
2. When upgrading from version equal or higher than 14.0 to 14.2 or higher version, you have appended "ECDHE-RSA-AES256-GCM-SHA384:" to the default Ciphers List to prefer RSA certificates over ECDSA. If you prefer ECDSA certificates over RSA, then remove "ECDHE-RSA-AES256-GCM-SHA384:" from the cipher string using Web User Interface (**Maintenance > Security > Ciphers**) or CLI command (**xConfiguration Ciphers**).
3. Any customer has a fresh install X14.2 image, ECDSA is being preferred. If you prefer RSA certificates over ECDSA, then prefix the cipher string with "ECDHE-RSA-AES256-GCM-SHA384:" using either Web User Interface (**Maintenance > Security > Ciphers**) or CLI command (**xConfiguration Ciphers**).

Hay dos maneras en que la verificación de TLS podría fallar en esta situación, que se tratan en detalle más adelante:

1. La CA que firmó el certificado remoto no es de confianza
 - a. Certificado con firma automática
 - b. Certificado firmado por CA desconocida
2. La dirección de conexión (FQDN o IP) no está incluida en el certificado

Solucionar escenarios

Los siguientes escenarios muestran un escenario similar en un entorno de laboratorio donde el inicio de sesión de MRA falló después de una actualización de Expressway de X14.0.7 a X14.2. Comparten similitudes en los registros, sin embargo la resolución es diferente. Los registros se recopilan mediante el registro de diagnóstico (de **Mantenimiento > Diagnóstico > Registro de diagnóstico**) que se inició antes del inicio de sesión de MRA y se detuvo después de que fallara el inicio de sesión de MRA. No se ha habilitado ningún registro de depuración adicional para él.

1. La CA que firmó el certificado remoto no es de confianza

El certificado remoto podría estar firmado por una CA que no está incluida en el almacén de confianza de Expressway-C o podría ser un certificado autofirmado (en esencia, una CA también) que no se agrega en el almacén de confianza del servidor de Expressway-C.

En el ejemplo aquí, puede observar que las solicitudes que van a CUCM (10.48.36.215 - cucm.steven.lab) se manejan correctamente en el puerto 8443 (respuesta 200 OK) pero arroja un error (respuesta 502) en el puerto 6972 para la conexión TFTP.

```
===Success connection on 8443===
```

```
2022-07-11T18:55:25.910+02:00 vcsc traffic_server[18242]: UTCTime="2022-07-11 16:55:25,910"
Module="network.http.trafficserver" Level="INFO": Detail="Receive Request" Txn-id="189"
TrackingID="6af9a674-9ebc-41ea-868e-90e7309a758c" Src-ip="127.0.0.1" Src-port="35764" Last-via-
addr="" Msg="GET
http://vcs_control.steven.lab:8443/c3RldmVuLmxhYi9odHRwcy9jdWNTLnN0ZXZlbi5sYWIVODQ0Mw/cucm-
uds/user/emusk/devices HTTP/1.1"
```

```
2022-07-11T18:55:25.917+02:00 vcsc traffic_server[18242]: Event="Request Allowed" Detail="Access
allowed" Reason="In allow list" Username="emusk" Deployment="1" Method="GET"
Request="https://cucm.steven.lab:8443/cucm-uds/user/emusk/devices"
```

```
Rule="https://cucm.steven.lab:8443/cucm-uds/user/" Match="prefix" Type="Automatically generated
rule for CUCM server" UTCTime="2022-07-11 16:55:25,916"
```

```
2022-07-11T18:55:25.917+02:00 vcsc traffic_server[18242]: UTCTime="2022-07-11 16:55:25,916"
Module="network.http.trafficserver" Level="INFO": Detail="Sending Request" Txn-id="189"
TrackingID="6af9a674-9ebc-41ea-868e-90e7309a758c" Dst-ip="10.48.36.215" Dst-port="8443" Msg="GET
/cucm-uds/user/emusk/devices HTTP/1.1"
```

```
2022-07-11T18:55:25.955+02:00 vcsc traffic_server[18242]: UTCTime="2022-07-11 16:55:25,955"
Module="network.http.trafficserver" Level="INFO": Detail="Receive Response" Txn-id="189"
TrackingID="" Src-ip="10.48.36.215" Src-port="8443" Msg="HTTP/1.1 200 "
```

```
2022-07-11T18:55:25.956+02:00 vcsc traffic_server[18242]: UTCTime="2022-07-11 16:55:25,955"
Module="network.http.trafficserver" Level="INFO": Detail="Sending Response" Txn-id="189"
TrackingID="" Dst-ip="127.0.0.1" Dst-port="35764" Msg="HTTP/1.1 200 "
```

```
===Failed connection on 6972===
```

```
2022-07-11T18:55:26.000+02:00 vcsc traffic_server[18242]: UTCTime="2022-07-11 16:55:26,000"
Module="network.http.trafficserver" Level="INFO": Detail="Receive Request" Txn-id="191"
TrackingID="bb0c8492-8c15-4537-a7d1-082dde781dbd" Src-ip="127.0.0.1" Src-port="35766" Last-via-
addr="" Msg="GET
http://vcs_control.steven.lab:8443/c3RldmVuLmxhYi9odHRwcy9jdWNTLnN0ZXZlbi5sYWIVNjk3Mg/CSFemusk.c
nf.xml HTTP/1.1"
```

```
2022-07-11T18:55:26.006+02:00 vcsc traffic_server[18242]: UTCTime="2022-07-11 16:55:26,006"
Module="network.http.trafficserver" Level="INFO": Detail="Sending Request" Txn-id="191"
TrackingID="bb0c8492-8c15-4537-a7d1-082dde781dbd" Dst-ip="10.48.36.215" Dst-port="6972" Msg="GET
/CSFemusk.cnf.xml HTTP/1.1"
```

2022-07-11T18:55:26.016+02:00 vscs traffic_server[18242]: UTCTime="2022-07-11 16:55:26,016" Module="network.http.trafficserver" Level="INFO": Detail="Sending Request" Txn-id="191" TrackingID="bb0c8492-8c15-4537-a7d1-082dde781dbd" Dst-ip="10.48.36.215" Dst-port="6972" Msg="GET /CSFemusk.cnf.xml HTTP/1.1"

2022-07-11T18:55:26.016+02:00 vscs traffic_server[18242]: [ET_NET 0] **WARNING: Core server certificate verification failed for** (cucm.steven.lab). **Action=Terminate Error=self signed certificate server=cucm.steven.lab(10.48.36.215) depth=0**

2022-07-11T18:55:26.016+02:00 vscs traffic_server[18242]: [ET_NET 0] **ERROR: SSL connection failed for** 'cucm.steven.lab': error:1416F086:SSL routines:tls_process_server_certificate:certificate verify failed

2022-07-11T18:55:26.024+02:00 vscs traffic_server[18242]: UTCTime="2022-07-11 16:55:26,024" Module="network.http.trafficserver" Level="INFO": Detail="Sending Response" Txn-id="191" TrackingID="" Dst-ip="127.0.0.1" Dst-port="35766" Msg="HTTP/1.1 502 connect failed"

El error 'certificate verify failed' indica el hecho de que Expressway-C no pudo validar el intercambio de señales TLS. La razón de esto se muestra en la línea de advertencia, ya que indica un certificado autofirmado. Si la profundidad se muestra como 0, es un certificado autofirmado. Cuando la profundidad es mayor que 0, significa que tiene una cadena de certificados y, por lo tanto, está firmado por una CA desconocida (desde la perspectiva de Expressway-C).

Cuando observamos el archivo pcap que se recopiló en las marcas de tiempo mencionadas en los registros de texto, puede ver que CUCM presenta el certificado con CN como cucm-ms.steven.lab (y cucm.steven.lab como SAN) firmado por steven-DC-CA a Expressway-C en el puerto 8443.

eth0_diagnostic_logging_tcpdump00_vscs_2022-07-11_16_55_44.pcap

No.	Time	Source	Dest port	Destination	Dest port	Protocol	DSCP	VLAN	Length	Info
4693	2022-07-11 16:55:25.916680	10.48.36.46	35622	10.48.36.215	8443	TCP	CS0		74	35622 → 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=878570435 TSecr=0 WS=128
4693	2022-07-11 16:55:25.916673	10.48.36.46	8443	10.48.36.46	35622	TCP	CS0		74	8443 → 35622 [SYN, ACK] Seq=0 Ack=1 Win=28968 Len=0 MSS=1460 SACK_PERM=1 TSval=343633238 TSecr=878570435 WS=128
4694	2022-07-11 16:55:25.917832	10.48.36.46	35622	10.48.36.215	8443	TLSv1.2	CS0		66	35622 → 8443 [ACK] Seq=1 Acks=1 Win=64256 Len=0 TSval=878570435 TSecr=343633238
4695	2022-07-11 16:55:25.938356	10.48.36.215	8443	10.48.36.46	35622	TLSv1.2	CS0		583	Client Hello
4696	2022-07-11 16:55:25.938390	10.48.36.46	35622	10.48.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=518 Ack=1449 Win=64218 Len=0 TSval=878570457 TSecr=343633251
4697	2022-07-11 16:55:25.938489	10.48.36.215	8443	10.48.36.46	35622	TLSv1.2	CS0		66	35622 → 8443 [ACK] Seq=518 Ack=2853 Win=63488 Len=0 TSval=878570457 TSecr=343633251
4698	2022-07-11 16:55:25.940187	10.48.36.46	35622	10.48.36.215	8443	TLSv1.2	CS0		192	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
4700	2022-07-11 16:55:25.943034	10.48.36.215	8443	10.48.36.46	35622	TLSv1.2	CS0		388	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
4701	2022-07-11 16:55:25.943051	10.48.36.46	35622	10.48.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=644 Acks=3095 Win=64218 Len=0 TSval=878570461 TSecr=343633256
4702	2022-07-11 16:55:25.943277	10.48.36.46	35622	10.48.36.215	8443	TLSv1.2	CS0		2543	Application Data
4703	2022-07-11 16:55:25.943476	10.48.36.215	8443	10.48.36.46	35622	TCP	CS0		66	8443 → 35622 [ACK] Seq=3095 Ack=3121 Win=35072 Len=0 TSval=343633256 TSecr=878570462
4707	2022-07-11 16:55:25.954796	10.48.36.215	8443	10.48.36.46	35622	TCP	CS0		1514	8443 → 35622 [ACK] Seq=3095 Ack=3121 Win=35072 Len=1448 TSval=343633268 TSecr=878570462 [TCP segment of a reassembled PDU]
4708	2022-07-11 16:55:25.954842	10.48.36.46	35622	10.48.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=644 Acks=4543 Win=64218 Len=0 TSval=878570473 TSecr=343633268
4709	2022-07-11 16:55:25.954861	10.48.36.215	8443	10.48.36.46	35622	TLSv1.2	CS0		1257	Application Data
4710	2022-07-11 16:55:25.954873	10.48.36.46	35622	10.48.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=3121 Acks=5734 Win=63488 Len=0 TSval=878570473 TSecr=343633268
4711	2022-07-11 16:55:25.955712	10.48.36.46	35622	10.48.36.215	8443	TLSv1.2	CS0		97	Encrypted Alert
4712	2022-07-11 16:55:25.955798	10.48.36.46	35622	10.48.36.215	8443	TCP	CS0		66	35622 → 8443 [FIN, ACK] Seq=3152 Acks=5734 Win=64218 Len=0 TSval=878570474 TSecr=343633268
4714	2022-07-11 16:55:25.956123	10.48.36.215	8443	10.48.36.46	35622	TLSv1.2	CS0		97	Encrypted Alert
4715	2022-07-11 16:55:25.956170	10.48.36.46	35622	10.48.36.215	8443	TCP	CS0		54	35622 → 8443 [RST] Seq=3153 Win=0 Len=0
4716	2022-07-11 16:55:25.956232	10.48.36.215	8443	10.48.36.46	35622	TCP	CS0		66	8443 → 35622 [FIN, ACK] Seq=5705 Acks=3153 Win=35072 Len=0 TSval=343633269 TSecr=878570474
4717	2022-07-11 16:55:25.956252	10.48.36.46	35622	10.48.36.215	8443	TCP	CS0		54	35622 → 8443 [RST] Seq=3153 Win=0 Len=0

Certificates (2429 Bytes)
Certificate Length: 1587
Certificate: 308205f308204c7a0030201020134500000122056050d3... (id-at-commonName=cucm-ms.steven.lab, id-at-organizationalUnitName=TAC, id-at-organizationName=Cisco, id-at-localityName=Olegon, id-at-stateOrProvinceName=Belgium, id-at-countryName=signedCertificate
version: v3 (2)
serialNumber: 0x450000012205605d3406084200020000122
> signature (sha1withRSAEncryption)
> issuer: rdnSequence (0)
> validity
> subject: rdnSequence (0)
> subjectPublicKeyInfo
> extensions: 9 items
> Extension (id-ce-extKeyUsage)
> Extension (id-ce-keyUsage)
> Extension (id-ce-subjectAltName)
Extension Id: 2.5.29.17 (id-ce-subjectAltName)
critical: True
> GeneralNames: 3 items
GeneralName: dNSName (2)
DNSName: cucm.steven.lab
GeneralName: dNSName (2)
DNSName: steven.lab
GeneralName: dNSName (3)
DNSName: cucm.steven.lab
> Extension (id-ce-subjectKeyIdentifier)
> Extension (id-ce-authorityKeyIdentifier)
> Extension (id-ce-cRLDistributionPoints)
> Extension (id-ce-authorityInfoAccessSyntax)
> Extension (id-ms-certificate-template)
> Extension (id-ms-application-certificate-policies)
> algorithmIdentifier (sha1withRSAEncryption)
padding: 0
encrypted: 9fb7f0741637a2a2071ef080f2270cc7cc48470c820...
Certificate Length: 910
> Certificate: 308203a30820272a00302010201062176f3f293908044... (id-at-commonName=steven-DC-CA, dc=steven, dc=lab)

Pero cuando inspeccionamos el certificado presentado en el puerto 6972, puede ver que es un certificado autofirmado (el emisor es él mismo) con CN configurado como cucm-EC.steven.lab. La extensión -EC indica que se trata del certificado ECDSA configurado en CUCM.

No.	Time	Source	Srv port	Destination	Dest port	Protocol	OSCP	VLAN	Length	Info
4730	2022-07-11 16:55:26.006408	10.40.36.46		11576 10.40.36.215	6972 TCP	C50			74	31576 → 6972 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=878578525 TSecr=0 WS=128
4731	2022-07-11 16:55:26.006853	10.40.36.215		6972 10.40.36.46	31576 TCP	C50			74	6972 → 31576 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=343633320 TSecr=878578525 WS=128
4732	2022-07-11 16:55:26.006892	10.40.36.46		31576 10.40.36.215	6972 TCP	C50			66	31576 → 6972 [ACK] Seq=1 Win=64256 Len=0 TSval=878578525 TSecr=343633320
4733	2022-07-11 16:55:26.007200	10.40.36.46		31576 10.40.36.215	6972 TLSv1.2	C50			583	Client Hello
4734	2022-07-11 16:55:26.013050	10.40.36.215		6972 10.40.36.46	31576 TLSv1.2	C50			1514	Server Hello, Certificate, Server Key Exchange
4735	2022-07-11 16:55:26.013351	10.40.36.46		31576 10.40.36.215	6972 TCP	C50			66	31576 → 6972 [ACK] Seq=518 Ack=1449 Win=64120 Len=0 TSval=878578535 TSecr=343633329
4736	2022-07-11 16:55:26.014040	10.40.36.215		6972 10.40.36.46	31576 TLSv1.2	C50			499	Certificate Request, Server Hello Done
4737	2022-07-11 16:55:26.014159	10.40.36.46		31576 10.40.36.215	6972 TCP	C50			66	31576 → 6972 [ACK] Seq=518 Ack=1882 Win=63744 Len=0 TSval=878578535 TSecr=343633329
4738	2022-07-11 16:55:26.014703	10.40.36.46		31576 10.40.36.215	6972 TLSv1.2	C50			73	Alert (Level: FATAL, Description: Unknown CA)
4739	2022-07-11 16:55:26.016221	10.40.36.46		31576 10.40.36.215	6972 TCP	C50			74	31576 → 6972 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=878578535 TSecr=0 WS=128
4740	2022-07-11 16:55:26.016965	10.40.36.46		31576 10.40.36.215	6972 TCP	C50			66	31576 → 6972 [RST, ACK] Seq=525 Ack=1882 Win=64120 Len=0 TSval=878578535 TSecr=343633329
4741	2022-07-11 16:55:26.016984	10.40.36.215		6972 10.40.36.46	31576 TCP	C50			74	6972 → 31576 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=343633330 TSecr=878578535 WS=128
4742	2022-07-11 16:55:26.017009	10.40.36.46		31578 10.40.36.215	6972 TCP	C50			66	31578 → 6972 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=878578535 TSecr=343633330
4743	2022-07-11 16:55:26.017101	10.40.36.215		6972 10.40.36.46	31576 TCP	C50			66	6972 → 31576 [SYN, ACK] Seq=1883 Ack=56 Win=30600 Len=0 TSval=878578535 TSecr=878578535
4744	2022-07-11 16:55:26.017123	10.40.36.46		31576 10.40.36.215	6972 TCP	C50			54	31576 → 6972 [RST] Seq=525 Win=0 Len=0
4745	2022-07-11 16:55:26.017218	10.40.36.46		31578 10.40.36.215	6972 TLSv1.2	C50			583	Client Hello
4746	2022-07-11 16:55:26.024226	10.40.36.215		6972 10.40.36.46	31578 TLSv1.2	C50			1514	Server Hello, Certificate, Server Key Exchange
4747	2022-07-11 16:55:26.024265	10.40.36.46		31578 10.40.36.215	6972 TCP	C50			66	31578 → 6972 [ACK] Seq=518 Ack=1449 Win=64120 Len=0 TSval=878578543 TSecr=343633337
4748	2022-07-11 16:55:26.024298	10.40.36.215		6972 10.40.36.46	31578 TLSv1.2	C50			590	Certificate Request, Server Hello Done
4749	2022-07-11 16:55:26.024309	10.40.36.46		31578 10.40.36.215	6972 TCP	C50			66	31578 → 6972 [ACK] Seq=518 Ack=1883 Win=63744 Len=0 TSval=878578543 TSecr=343633337
4750	2022-07-11 16:55:26.024548	10.40.36.46		31578 10.40.36.215	6972 TLSv1.2	C50			73	Alert (Level: Fatal, Description: Unknown CA)
4751	2022-07-11 16:55:26.024647	10.40.36.46		31578 10.40.36.215	6972 TCP	C50			66	31578 → 6972 [RST, ACK] Seq=525 Ack=1883 Win=64120 Len=0 TSval=878578543 TSecr=343633337
4752	2022-07-11 16:55:26.030359	10.40.36.46		31580 10.40.36.215	6972 TCP	C50			74	31580 → 6972 [SYN] Seq=0 Win=0 Len=0 MSS=1460 SACK_PERM=1 TSval=878578601 TSecr=0 WS=128

```

Secure Sockets Layer
  TLSv1.2 Record Layer: Handshake Protocol: Server Hello
  TLSv1.2 Record Layer: Handshake Protocol: Certificate
  Content Type: Handshake (22)
  Version: TLS 1.2 (0x0303)
  Length: 667
  Handshake Protocol: Certificate
  Handshake type: Certificate (11)
  Length: 663
  Certificates length: 660
  Certificates (600 Bytes)
  Certificate Length: 657
  Certificate: 3082020308202148008020210207470ee6271e1d1346... (id-at-localityName=Diegem,id-at-stateOrProvinceName=Belgium,id-at-commonName=cucm-ec.steven.lab,id-at-organizationalUnitName=TAC,id-at-organizationName=Cisco,id-at-countryName=BE)
    version: v3 (2)
    serialNumber: 00470ee6271e1d13461d09466a3b9f1d
    signature (ecdsa-with-SHA384)
    issuer: rdsnSequence (8)
    rdsnSequence: 6 items (id-at-localityName=Diegem,id-at-stateOrProvinceName=Belgium,id-at-commonName=cucm-ec.steven.lab,id-at-organizationalUnitName=TAC,id-at-organizationName=Cisco,id-at-countryName=BE)
    validity
    subject: rdsnSequence (8)
    subjectPublicKeyInfo
    extensions: 5 items
    Extension (id-ce-keyUsage)
    Extension (id-ce-extendedKeyUsage)
    Extension (id-ce-subjectKeyIdentifier)
    Extension (id-ce-basicConstraints)
    Extension (id-ce-subjectAltName)
    Extension (id-ce-subjectAltName)
    Extension ID: 2.5.29.17 (id-ce-subjectAltName)
    GeneralNames: 1 item
    GeneralName: dnName (2)
    dnName: cucm.steven.lab
    algorithmIdentifier (ecdsa-with-SHA384)
    padding: 0
    encrypted: 3064020121439565e74570b1171eb49f30b6ec0d08...
  TLSv1.2 Record Layer: Handshake Protocol: Server Key Exchange
  
```

En CUCM, en Administración de Cisco Unified OS, puede consultar los certificados en vigor en Seguridad > Administración de certificados, como se muestra, por ejemplo, aquí. Muestra un certificado diferente para tomcat y tomcat-ECDSA donde tomcat está firmado por CA (y es de confianza para Expressway-C) mientras que el certificado tomcat-ECDSA está firmado por sí mismo y no es de confianza para Expressway-C aquí.

Certificate	Common Name	Type	Key Type	Distribution	Issued by	Expiration	Description
authZ	AUTHZ_cucm.steven.lab	Self-signed	RSA	cucm.steven.lab	AUTHZ_cucm.steven.lab	07/21/2038	Self-signed certificate generated by system
CallManager	cucm.steven.lab	CA-signed	RSA	cucm.steven.lab	stevan-DC-CA	07/13/2022	Certificate Signed by stevan-DC-CA
CallManager-ECDSA	cucm-ec.steven.lab	Self-signed	EC	cucm.steven.lab	cucm-ec.steven.lab	02/18/2024	Self-signed certificate generated by system
CallManager-trust	stevan-DC-CA	Self-signed	RSA	stevan-DC-CA	stevan-DC-CA	06/01/2025	Signed Certificate
CallManager-trust	NOMAT-AD-CA	Self-signed	RSA	NOMAT-AD-CA	NOMAT-AD-CA	04/23/2028	Signed Certificate
CallManager-trust	CAP-RT9-002	Self-signed	RSA	CAP-RT9-002	CAP-RT9-002	10/10/2023	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	CAPP-ab26468	Self-signed	RSA	CAPP-ab26468	CAPP-ab26468	04/12/2020	Signed Certificate
CallManager-trust	ms-AD2-CA-1	Self-signed	RSA	ms-AD2-CA-1	ms-AD2-CA-1	09/11/2024	vntgc CA
CallManager-trust	CAP-RT9-001	Self-signed	RSA	CAP-RT9-001	CAP-RT9-001	02/07/2023	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	NOMAT-CA-10	Self-signed	RSA	NOMAT-CA-10	NOMAT-CA-10	08/11/2027	Signed Certificate
CallManager-trust	Cisco_Root_CA_M2	Self-signed	RSA	Cisco_Root_CA_M2	Cisco_Root_CA_M2	11/22/2037	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	ACT2_SUDD_CA	CA-signed	RSA	ACT2_SUDD_CA	Cisco_Root_CA_2048	05/14/2029	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	vntgc-ACTIVE-DIR-CA	Self-signed	RSA	vntgc-ACTIVE-DIR-CA	vntgc-ACTIVE-DIR-CA	02/10/2024	Signed Certificate
CallManager-trust	Cisco_Root_CA_2048	Self-signed	RSA	Cisco_Root_CA_2048	Cisco_Root_CA_2048	05/14/2029	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	Cisco_Manufacturing_CA	CA-signed	RSA	Cisco_Manufacturing_CA	Cisco_Root_CA_2048	05/14/2029	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	Cisco_Manufacturing_CA_SHA2	CA-signed	RSA	Cisco_Manufacturing_CA_SHA2	Cisco_Root_CA_H2	11/22/2037	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	dcomics-WONDERWOMAN-CA	Self-signed	RSA	dcomics-WONDERWOMAN-CA	dcomics-WONDERWOMAN-CA	09/19/2037	CA-variant
CallManager-trust	CAPP-616421bc	Self-signed	RSA	CAPP-616421bc	CAPP-616421bc	07/12/2025	Self-signed certificate generated by system
CallManager-trust	CAPP-616421bc	Self-signed	RSA	cucm.steven.lab	CAPP-616421bc	07/12/2025	Self-signed certificate generated by system
CallManager-trust	CAP-RT9-002	Self-signed	RSA	CAP-RT9-002	CAP-RT9-002	10/10/2023	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	CAPP-ab26468	Self-signed	RSA	CAPP-ab26468	CAPP-ab26468	04/12/2020	Signed Certificate
CallManager-trust	CAP-RT9-001	Self-signed	RSA	CAP-RT9-001	CAP-RT9-001	02/07/2023	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	Cisco_Root_CA_M2	Self-signed	RSA	Cisco_Root_CA_M2	Cisco_Root_CA_M2	11/22/2037	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	ACT2_SUDD_CA	CA-signed	RSA	ACT2_SUDD_CA	Cisco_Root_CA_2048	05/14/2029	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	Cisco_Root_CA_2048	Self-signed	RSA	Cisco_Root_CA_2048	Cisco_Root_CA_2048	05/14/2029	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	Cisco_Manufacturing_CA_SHA2	CA-signed	RSA	Cisco_Manufacturing_CA_SHA2	Cisco_Root_CA_H2	11/22/2037	This certificate was used to sign the MIC installed on Cisco endpoint. Presence of this certificate allows the end point to communicate securely with UCH using the MIC when associated with a secure profile.
CallManager-trust	CAPP-616421bc	Self-signed	RSA	CAPP-616421bc	CAPP-616421bc	07/12/2025	Self-signed certificate generated by system
ispcc	cucm.steven.lab	Self-signed	RSA	cucm.steven.lab	cucm.steven.lab	07/12/2025	Trust Certificate
ispcc-trust	cucm.steven.lab	Self-signed	RSA	cucm.steven.lab	cucm.steven.lab	07/12/2025	Trust Certificate
ITLRecovery	ITLRECOVERY_cucm.steven.lab	Self-signed	RSA	cucm.steven.lab	ITLRECOVERY_cucm.steven.lab	02/14/2039	Self-signed certificate generated by system
tomcat	cucm.steven.lab	CA-signed	RSA	cucm.steven.lab	stevan-DC-CA	07/10/2024	Certificate Signed by stevan-DC-CA
tomcat-ECDSA	cucm-ec.steven.lab	Self-signed	EC	cucm.steven.lab	cucm-ec.steven.lab	07/25/2023	Self-signed certificate generated by system
tomcat-ECDSA	cucm-ec.steven.lab	Self-signed	EC	cucm.steven.lab	cucm-ec.steven.lab	07/25/2023	Trust Certificate
tomcat-trust	stevan-DC-CA	Self-signed	RSA	stevan-DC-CA	stevan-DC-CA	06/01/2025	Trust Certificate
tomcat-trust	NOMAT-AD-CA	Self-signed	RSA	NOMAT-AD-CA	NOMAT-AD-CA	04/23/2028	Trust Certificate
tomcat-trust	cucm-ec.steven.lab	Self-signed	EC	cucm.steven.lab	cucm-ec.steven.lab	07/25/2023	Trust Certificate
tomcat-trust	cucm.steven.lab	CA-signed	RSA	cucm.steven.lab	stevan-DC-CA	07/10/2024	Trust Certificate
tomcat-trust	cucm-ec.steven.lab	Self-signed	EC	cucm.steven.lab	cucm-ec.steven.lab	07/25/2023	Trust Certificate
tomcat-trust	NOMAT-CA-10	Self-signed	RSA	NOMAT-CA-10	NOMAT-CA-10	08/11/2027	Signed Certificate
tomcat-trust	vntgc-ACTIVE-DIR-CA	Self-signed	RSA	vntgc-ACTIVE-DIR-CA	vntgc-ACTIVE-DIR-CA	02/10/2024	Trust Certificate
tomcat-trust	dcomics-WONDERWOMAN-CA	Self-signed	RSA	dcomics-WONDERWOMAN-CA	dcomics-WONDERWOMAN-CA	09/19/2037	CA Bruno
TVS	cucm.steven.lab	Self-signed	RSA	cucm.steven.lab	cucm.steven.lab	07/12/2025	Self-signed certificate generated by system

2. La dirección de conexión (FQDN o IP) no está incluida en el certificado

Aparte del almacén de confianza, el servidor de tráfico de MRA también verifica la dirección de conexión hacia la que realiza la solicitud el cliente MRA. Por ejemplo, cuando ha configurado en

CUCM en **System > Server** su CUCM con la dirección IP (10.48.36.215), Expressway-C anuncia esto como tal al cliente y las solicitudes posteriores del cliente (procesadas a través de Expressway-C) se dirigen hacia esta dirección.

Cuando esa dirección de conexión en particular no está contenida en el certificado del servidor, la verificación de TLS también falla y se arroja un error 502 que resulta en una falla de inicio de sesión de MRA, por ejemplo.

```
2022-07-11T19:49:01.472+02:00 vcsc traffic_server[3916]: UTCTime="2022-07-11 17:49:01,472"
Module="network.http.trafficserver" Level="DEBUG": Detail="Receive Request" Txn-id="144"
TrackingID="0a334fa8-41e9-4b97-adf4-e165372c38cb" Src-ip="127.0.0.1" Src-port="30044" Last-via-
addr=""
HTTPMSG:
|GET http://vcs_control.steven.lab:8443/c3RldmVuLmxhYi9odHRwcy8xMC40OC4zNi4yMTUvODQ0Mw/cucm-
uds/user/emusk/devices?max=100 HTTP/1.1
...

2022-07-11T19:49:01.478+02:00 vcsc traffic_server[3916]: UTCTime="2022-07-11 17:49:01,478"
Module="network.http.trafficserver" Level="INFO": Detail="Sending Request" Txn-id="144"
TrackingID="0a334fa8-41e9-4b97-adf4-e165372c38cb" Dst-ip="10.48.36.215" Dst-port="8443" Msg="GET
/cucm-uds/user/emusk/devices?max=100 HTTP/1.1"
2022-07-11T19:49:01.478+02:00 vcsc traffic_server[3916]: UTCTime="2022-07-11 17:49:01,478"
Module="network.http.trafficserver" Level="DEBUG": Detail="Sending Request" Txn-id="144"
TrackingID="0a334fa8-41e9-4b97-adf4-e165372c38cb" Dst-ip="10.48.36.215" Dst-port="8443"
HTTPMSG:
|GET /cucm-uds/user/emusk/devices?max=100 HTTP/1.1
...

2022-07-11T19:49:01.491+02:00 vcsc traffic_server[3916]: [ET_NET 2] WARNING: SNI (10.48.36.215)
not in certificate. Action=Terminate server=10.48.36.215(10.48.36.215)
2022-07-11T19:49:01.491+02:00 vcsc traffic_server[3916]: [ET_NET 2] ERROR: SSL connection failed
for '10.48.36.215': error:1416F086:SSL routines:tls_process_server_certificate:certificate
verify failed
```

Donde c3RldmVuLmxhYi9odHRwcy8xMC40OC4zNi4yMTUvODQ0Mw se traduce (base64 - <https://www.base64decode.org/>) a [steven.lab/https/10.48.36.215/8443](https://www.steven.lab/https/10.48.36.215/8443), que muestra que debe hacer la conexión hacia 10.48.36.215 como la dirección de conexión en lugar de a cucm.steven.lab. Como se muestra en las capturas de paquetes, el certificado Tomcat de CUCM no contiene la dirección IP en la SAN y, por lo tanto, se produce el error.

Cómo validarlo fácilmente

Puede validar si se encuentra con este cambio de comportamiento fácilmente con los siguientes pasos:

1. Inicie el registro de diagnóstico en los servidores de Expressway-E y C (idealmente con TCPDumps habilitado) desde **Mantenimiento > Diagnóstico > Registro de diagnóstico** (en caso de un clúster, es suficiente iniciarlo desde el nodo principal)
2. Intente iniciar sesión en MRA o pruebe la funcionalidad dañada después de la actualización
3. Espere hasta que falle y luego detenga el registro de diagnóstico en los servidores de Expressway-E y C (en caso de un clúster, asegúrese de recopilar los registros de cada nodo individual del clúster)
4. Cargue y analice los registros en la [herramienta Collaboration Solution Analyzer](#)

5. Si se encuentra con el problema, recoge las líneas de error y advertencia más recientes relacionadas con este cambio para cada uno de los servidores afectados

firma de diagnóstico de CA

Firma de diagnóstico SNI

Solución

La solución a largo plazo es asegurarse de que la verificación de TLS funcione correctamente. La acción que se debe realizar depende del mensaje de advertencia que se muestre.

Cuando observe la **ADVERTENCIA: Error al comprobar el certificado de servidor principal para (<FQDN-o-IP-servidor>)**. **Action=Terminate Error=self signed certificate**

server=cucm.steven.lab(10.48.36.215) depth=x message, entonces debe actualizar el almacén de confianza en los servidores de Expressway-C en consecuencia. Con la cadena de CA que firmó este certificado (profundidad > 0) o con el certificado autofirmado (profundidad = 0) de **Mantenimiento > Seguridad > Certificado de CA de confianza**. Asegúrese de realizar esta acción en todos los servidores del clúster. Otra opción sería firmar el certificado remoto por una CA conocida en el almacén de confianza de Expressway-C.

Nota: Expressway no permite cargar dos certificados diferentes (autofirmados, por ejemplo) en el almacén de confianza de Expressway que tienen el mismo nombre común (CN) que según el Id. de error de Cisco [CSCwa12905](#). Para corregir esto, mueva a certificados firmados por CA o actualice CUCM a la versión 14 donde puede volver a utilizar el mismo certificado (autofirmado) para Tomcat y CallManager.

Cuando observe la **ADVERTENCIA: SNI (<server-FQDN-or-IP> no está en el mensaje del certificado**, entonces indica que este FQDN o IP del servidor no está contenido dentro del certificado que se presentó. Puede adaptar el certificado para incluir esa información o puede modificar la configuración (como en CUCM en Sistema > Servidor para que se incluya en el certificado del servidor) y luego actualizar la configuración en el servidor de Expressway-C para que se tenga en cuenta.

La solución a corto plazo consiste en aplicar la solución alternativa según lo documentado para recurrir al comportamiento anterior anterior a X14.2.0. Puede realizar esto a través de la CLI en los nodos del servidor de Expressway-C con el comando recién introducido:

```
xConfiguration EdgeConfigServer VerifyOriginServer: Off
```

Acerca de esta traducción

Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).