

Solucionar problemas da verificação de certificado do servidor de tráfego Expressway para serviços MRA introduzidos pelo CSCwc69661

Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Componentes Utilizados](#)

[Informações de Apoio](#)

[Cadeia de CA confiável](#)

[Verificação de SAN ou CN](#)

[Alteração de comportamento](#)

[Versões anteriores a X14.2.0](#)

[Versões do X14.2.0 e posterior](#)

[Solucionar problemas de cenários](#)

[1. A autoridade de certificação que assinou o certificado remoto não é confiável](#)

[2. O endereço de conexão \(FQDN ou IP\) não está contido no certificado](#)

[Como validá-lo facilmente](#)

[Solução](#)

Introduction

Este documento descreve a alteração de comportamento nas versões do Expressway do X14.2.0 e superior vinculada ao bug da Cisco ID [CSCwc69661](#). Com essa alteração, o servidor de tráfego na plataforma Expressway executa a verificação de certificado do Cisco Unified Communication Manager (CUCM), do Cisco Unified Instant Messaging & Presence (IM&P) e dos nós de servidor do Unity para os serviços de acesso remoto e móvel (MRA). Essa alteração pode levar a falhas de login de MRA após uma atualização na plataforma Expressway.

Prerequisites

Requirements

A Cisco recomenda que você tenha conhecimento destes tópicos:

- Configuração básica do Expressway
- configuração básica MRA

Componentes Utilizados

As informações neste documento são baseadas no Cisco Expressway na versão X14.2 e superior.

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. Se a rede estiver ativa, certifique-se de que você entenda o impacto potencial de qualquer comando.

Informações de Apoio

O protocolo HTTPS é um protocolo de comunicação seguro que usa o TLS (Transport Layer Security) para criptografar a comunicação. Ele cria esse canal seguro usando um certificado TLS que é trocado no handshake TLS. Dessa forma, ela serve a duas finalidades: autenticação (para saber a quem o participante remoto está conectado) e privacidade (a criptografia). A autenticação protege contra ataques de intermediários e a privacidade impede que os invasores interceptem e interfiram na comunicação.

A verificação de TLS (certificado) é realizada aos olhos da autenticação e permite ter certeza de que você se conectou à parte remota certa. A verificação consiste em dois elementos individuais:

1. Cadeia da Autoridade de Certificação Confiável (AC)
2. Nome Alternativo do Assunto (SAN) ou Nome Comum (CN)

Cadeia de CA confiável

Para que o Expressway-C confie no certificado que o CUCM / IM&P / Unity envia, ele precisa ser capaz de estabelecer um link desse certificado para uma Autoridade de Certificação (CA) de nível superior (raiz) em que ele confia. Esse link, uma hierarquia de certificados que vincula um certificado de entidades a um certificado de CA raiz, é chamado de cadeia de confiança. Para poder verificar essa cadeia de confiança, cada certificado contém dois campos: Emitente (ou 'Emitido por') e Assunto (ou 'Emitido para').

Os certificados de servidor, como o que o CUCM envia para o Expressway-C, têm no campo "Assunto" normalmente seu FQDN (Fully Qualified Domain Name, Nome de domínio totalmente qualificado) no 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
```

Exemplo de um certificado de servidor para CUCM cucm.vngtp.lab. Ele tem o FQDN no atributo CN do campo Assunto junto com outros atributos como País (C), Estado (ST), Local (L), ... Também podemos ver que o certificado do servidor é distribuído (emitido) por uma CA chamada vngtp-ACTIVE-DIR-CA.

As CAs de nível superior (CAs raiz) também podem emitir um certificado para se identificarem. Nesse certificado CA raiz, vemos que Emissor e Assunto têm o mesmo valor :

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

É um certificado passado por uma CA raiz para se identificar.

Em uma situação típica, as CAs raiz não emitem diretamente certificados de servidor. Em vez disso, emitem certificados para outras autoridades de certificação. Essas outras CAs são chamadas de CAs intermediárias. Por sua vez, as autoridades de certificação intermediárias podem emitir diretamente certificados de servidor ou certificados para outras autoridades de certificação intermediárias. Podemos ter uma situação em que um certificado de servidor é emitido pela CA 1 intermediária, que por sua vez recebe um certificado da CA 2 intermediária e assim por diante. Até que a CA intermediária obtenha seu certificado diretamente da CA raiz:

```
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
```

Agora, para que o Expressway-C confie no certificado de servidor que o CUCM envia, ele precisa ser capaz de criar a cadeia de confiança desse certificado de servidor até um certificado de CA raiz. Para que isso aconteça, precisamos carregar o certificado de CA raiz e também todos os certificados de CA intermediários (se houver, o que não é o caso se a CA raiz teria emitido diretamente o certificado de servidor do CUCM) no armazenamento confiável do Expressway-C.

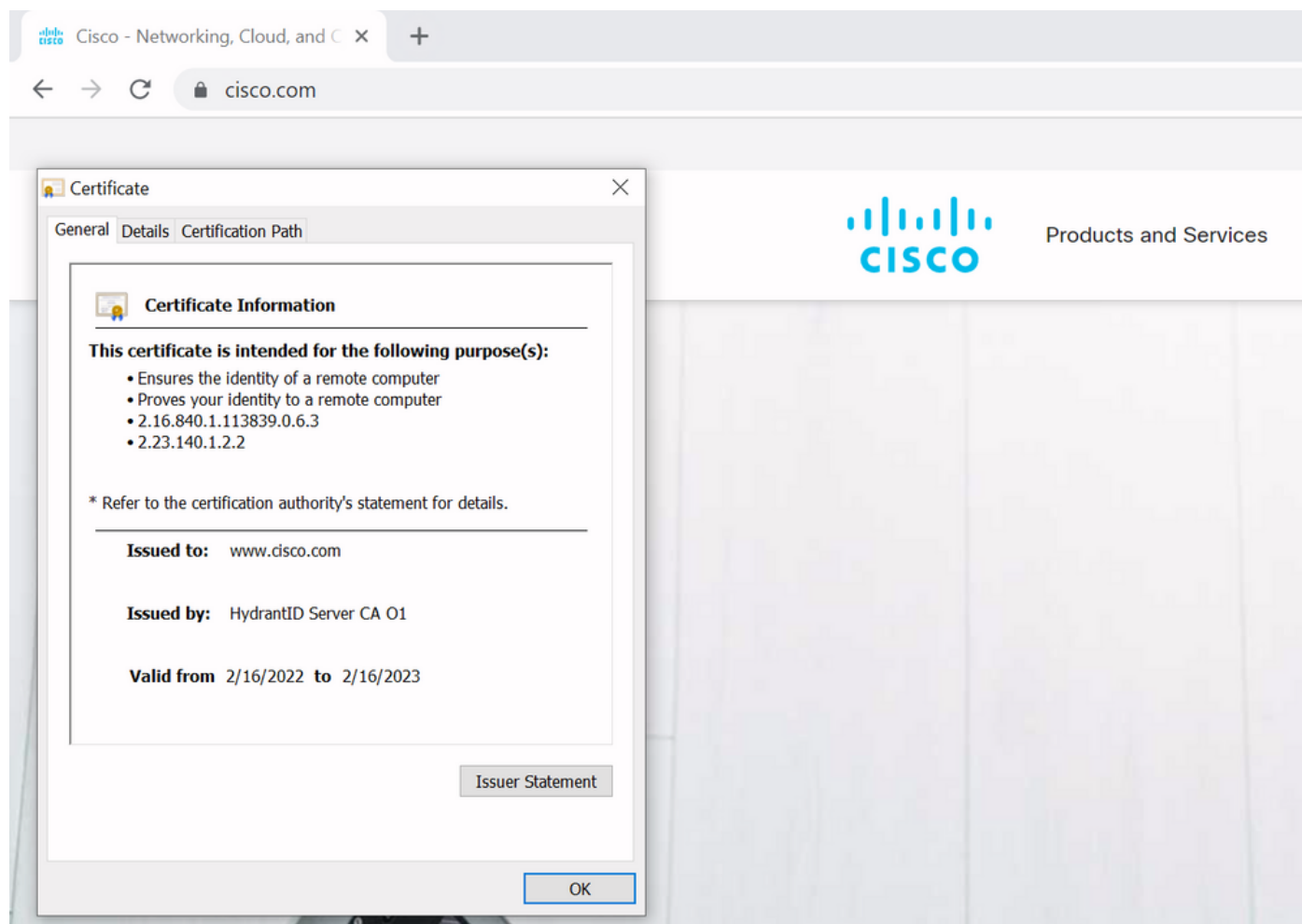
Note: Embora os campos Emissor e Assunto sejam fáceis de criar a cadeia de Confiança de forma legível por humanos, o Expressway-C e o CUCM não usam esses campos no certificado. Em vez disso, eles usam os campos 'X509v3 Authority Key Identifier' e 'X509v3 Subject Key Identifier' para criar a cadeia de confiança. Essas chaves contêm identificadores para os certificados que são mais precisos do que para usar os campos Assunto/Emissor : pode haver 2 certificados com os mesmos campos Assunto/Emissor, mas um deles expirou e o outro ainda é válido. Ambos teriam um identificador de chave de assunto X509v3 diferente para que o Expressway/CUCM ainda possa determinar a cadeia de confiança correta.

Verificação de SAN ou CN

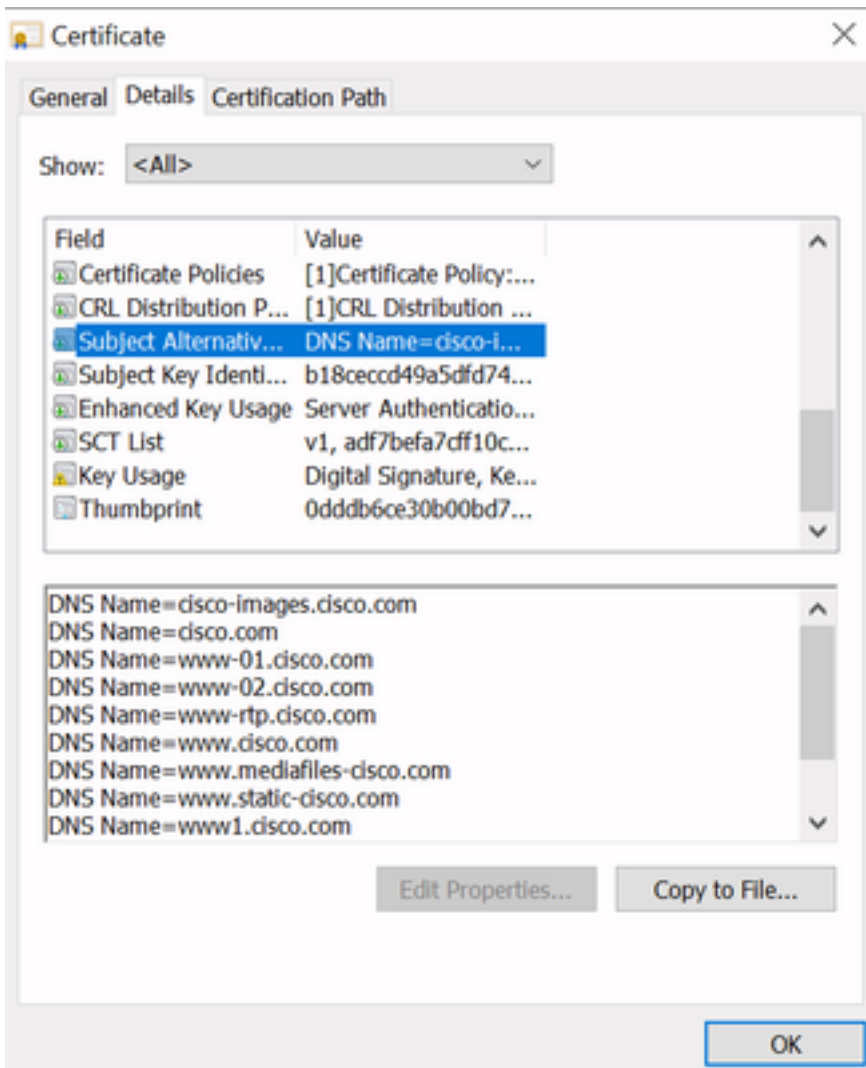
A Etapa 1 faz o check-out do armazenamento confiável, mas qualquer pessoa que tiver um certificado assinado por uma CA no armazenamento confiável será válida. É evidente que isto não é suficiente. Portanto, há uma verificação adicional que valida se o servidor ao qual você se conecta especificamente é realmente o correto. Ele faz isso com base no endereço para o qual o pedido foi feito.

O mesmo tipo de operação acontece em seu navegador, então vamos analisar isso por meio de um exemplo. Se você navegar até <https://www.cisco.com>, verá um ícone de cadeado ao lado do URL inserido e isso significa que a conexão é confiável. Isso é baseado na cadeia de confiança da CA (da primeira seção) e na verificação SAN ou CN. Se abrirmos o certificado (através do navegador com um clique no ícone de cadeado), você verá que o Nome comum (visto no campo

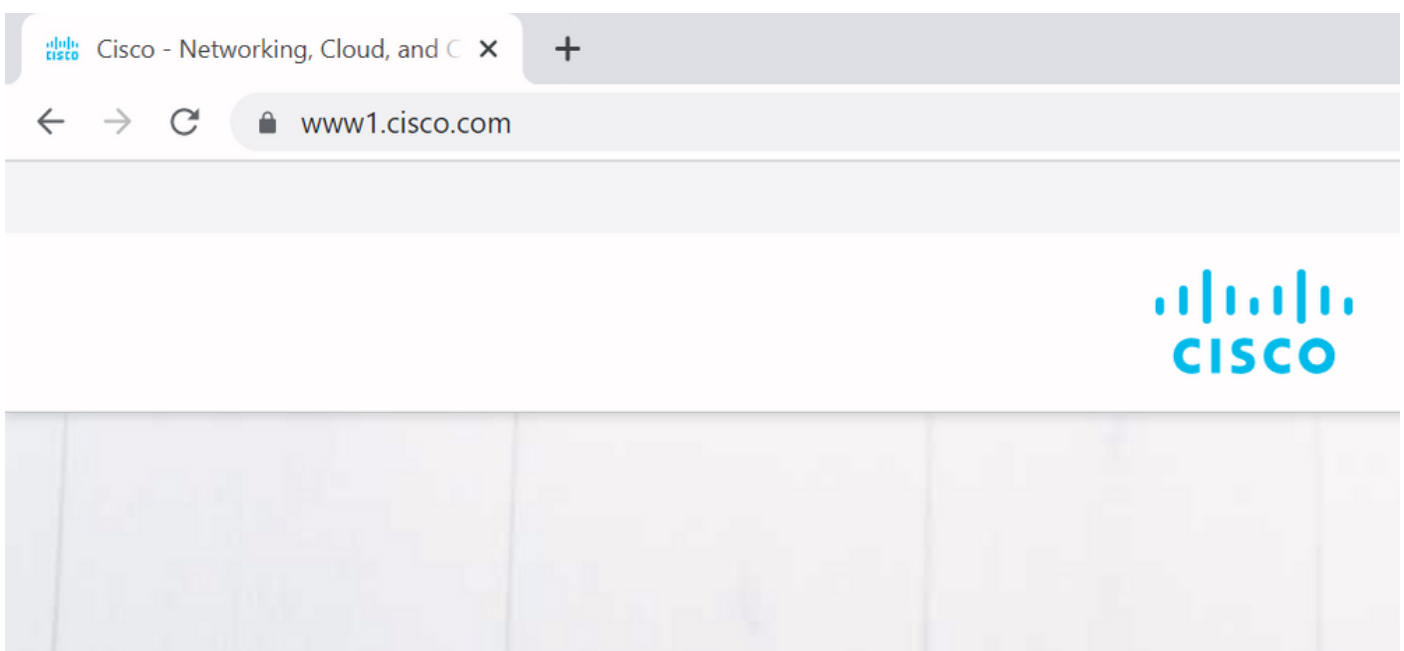
'Emitido para:') está definido como www.cisco.com e que corresponde exatamente ao endereço ao qual queríamos nos conectar. Dessa forma, podemos ter certeza de que nos conectamos ao servidor correto (porque confiamos na CA que assinou o certificado e que executa a verificação antes de entregar o certificado).



Quando observamos os detalhes do certificado e, em particular, as entradas SAN, vemos que o mesmo se repete, assim como alguns outros FQDNs:



Isso significa que, quando solicitamos a conexão com <https://www1.cisco.com>, por exemplo, ela também seria exibida como uma conexão segura porque está contida nas entradas SAN.



No entanto, quando não navegamos até <https://www.cisco.com>, mas diretamente até o endereço IP (<https://72.163.4.161>), ela não mostra uma conexão segura porque confia na CA que a assinou, mas o certificado apresentado a nós não contém o endereço (72.163.4.161) que usamos para conexão com o servidor.

The image shows a browser window with a 'Privacy error' tab and a 'Not secure' warning. The address bar shows 'https://72.163.4.161'. To the left, a Command Prompt window displays the output of an 'nslookup' command for 'cisco.com', showing the server as 'dns-aer1.cisco.com' and the address as '173.38.200.100'. The browser page displays a red warning triangle with an exclamation mark and the text 'Your connection is not private'. Below this, it states 'Attackers might be trying to steal your information from 72.163.4.161 (for example, passwords, messages, or credit cards). Learn more'. The error code 'NET:ERR_CERT_COMMON_NAME_INVALID' is visible. A button 'To get Chrome's highest level of security, turn on enhanced protection' is present. At the bottom, there are buttons for 'Hide advanced' and 'Back to safety'. A note at the bottom states 'This server could not prove that it is 72.163.4.161; its security certificate is from www.cisco.com. This may be caused by a misconfiguration or an attacker intercepting your connection.' and a link 'Proceed to 72.163.4.161 (unsafe)'.

No navegador, você pode ignorar isso, mas essa é uma configuração que você pode habilitar em conexões TLS que um desvio não é permitido. Portanto, é importante que seus certificados contenham os nomes CN ou SAN corretos que a parte remota planeja usar para se conectar a eles.

Alteração de comportamento

Os serviços MRA dependem muito de várias conexões HTTPS nos Expressways em direção aos servidores CUCM / IM&P / Unity para se autenticar corretamente e coletar as informações certas específicas para o cliente que faz login. Essa comunicação geralmente acontece nas portas 8443 e 6972.

Versões anteriores a X14.2.0

Em versões anteriores a X14.2.0, o servidor de tráfego no Expressway-C que lida com essas conexões HTTPS seguras não verificou o certificado apresentado pela extremidade remota. Isso pode levar a ataques de intermediários. Na configuração MRA, há uma opção para verificação de certificado TLS pela configuração do 'Modo de verificação TLS' como 'Ativado' quando você adicionaria servidores CUCM / IM&P / Unity em **Configuração > Comunicações Unificadas > servidores Unified CM / nós de Serviço IM e Presence / servidores Unity Connection**. A opção de configuração e a caixa de informações relevantes são mostradas como exemplo, indicando que ela verifica o FQDN ou o IP na SAN, bem como a validade do certificado e se ele está assinado por uma CA confiável.



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 verificação de certificado TLS só é feita na descoberta dos servidores CUCM / IM&P / Unity e não no momento em que, durante o login MRA, os vários servidores são consultados. Uma primeira desvantagem dessa configuração é que ela apenas verifica o endereço do publicador adicionado. Ele não valida se o certificado nos nós do assinante foi configurado corretamente, pois recupera as informações do nó do assinante (FQDN ou IP) do banco de dados do nó do publicador. Uma segunda desvantagem dessa configuração também é que o que é anunciado para os clientes MRA como as informações de conexão podem ser diferentes do endereço do publicador que foi colocado na configuração Expressway-C. Por exemplo, no CUCM, em **System > Server**, você pode anunciar o servidor com um endereço IP (10.48.36.215, por exemplo) e isso é usado pelos clientes MRA (através da conexão do Expressway com proxy); no entanto, você pode adicionar o CUCM no Expressway-C com o FQDN de cucm.steven.lab. Suponha que o certificado tomcat do CUCM contenha cucm.steven.lab como entrada de SAN, mas não o endereço IP, então a descoberta com 'TLS Verify Mode' definido como 'On' é bem-sucedida, mas

as comunicações reais dos clientes MRA podem ter como destino um FQDN ou IP diferente e, portanto, falhar na verificação de TLS.

Versões do X14.2.0 e posterior

A partir da versão X14.2.0, o servidor Expressway executa a verificação de certificado TLS para cada solicitação HTTPS feita através do servidor de tráfego. Isso significa que ele também executa isso quando o 'Modo de verificação TLS' está definido como 'Desligado' durante a descoberta dos nós CUCM / IM&P / Unity. Quando a verificação não é bem-sucedida, o handshake TLS não é concluído e a solicitação falha, o que pode levar à perda de funcionalidade, como problemas de redundância ou failover ou falhas de login completas, por exemplo. Além disso, com o 'Modo de verificação de TLS' definido como 'Ativado', ele não garante que todas as conexões funcionem bem, conforme abordado no exemplo mais adiante.

Além do padrão na verificação TLS, há também uma alteração introduzida no X14.2 que anuncia uma ordem de preferência diferente para a lista de cifras. Isso pode causar conexões TLS inesperadas após uma atualização de software, pois pode acontecer que, antes da atualização, ele solicite o certificado Cisco Tomcat ou Cisco CallManager do CUCM (ou qualquer outro produto que tenha um certificado separado para o algoritmo ECDSA), mas que, após a atualização, ele solicite a variante ECDSA. Os certificados Cisco Tomcat-ECDSA ou Cisco CallManager-ECDSA podem ser assinados por uma CA diferente ou apenas certificados ainda autoassinados (o padrão).

Há duas maneiras de a verificação de TLS falhar neste cenário, que serão abordadas em detalhes posteriormente:

1. A autoridade de certificação que assinou o certificado remoto não é confiável
 - a. Certificado autoassinado
 - b. Certificado assinado por CA desconhecida
2. O Endereço de Conexão (FQDN ou IP) não está contido no certificado

Solucionar problemas de cenários

Os próximos cenários mostram um cenário semelhante em um ambiente de laboratório onde o login de MRA falhou após uma atualização do Expressway de X14.0.7 para X14.2. Eles compartilham semelhanças nos logs, no entanto, a resolução é diferente. Os logs são coletados pelo log de diagnóstico (de **Manutenção > Diagnóstico > Log de diagnóstico**) que foi iniciado antes do logon de MRA e interrompido depois que o logon de MRA falhou. Nenhum log de depuração adicional foi habilitado para ele.

1. A autoridade de certificação que assinou o certificado remoto não é confiável

O certificado remoto pode ser assinado por uma CA que não esteja incluída no armazenamento confiável do Expressway-C ou pode ser um certificado autoassinado (em essência, uma CA também) que não é adicionado no armazenamento confiável do servidor Expressway-C.

No exemplo aqui, você pode observar que as solicitações que vão para o CUCM (10.48.36.215 - cucm.steven.lab) são tratadas corretamente na porta 8443 (resposta 200 OK), mas ela gera um

erro (resposta 502) na porta 6972 para a conexão 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/c3RldmVuLmxhYi9odHRwcy9jdWNTLnN0ZXZlbi5sYWVvODQ0Mw/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/c3RldmVuLmxhYi9odHRwcy9jdWNTLnN0ZXZlbi5sYWVvNjk3Mg/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 vcsc 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 vcsc 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 vcsc 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 vcsc 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"
```

O erro de 'falha na verificação do certificado' indica o fato de que o Expressway-C não pôde validar o handshake TLS. A razão para isso é mostrada na linha de aviso, pois indica um certificado autoassinado. Se a profundidade for mostrada como 0, é um certificado autoassinado. Quando a profundidade é maior que 0, isso significa que ela tem uma cadeia de certificados e, portanto, é assinada por uma CA desconhecida (da perspectiva do Expressway-C).

Quando observamos o arquivo pcap que foi coletado nos carimbos de data e hora mencionados

nos logs de texto, você pode ver que o CUCM apresenta o certificado com CN como cucm-ms.steven.lab (e cucm.steven.lab como SAN) assinado por steven-DC-CA ao Expressway-C na porta 8443.

No.	Time	Source	Src port	Destination	Dest port	Protocol	DSCP	VLAN	Length	Info
4693	2022-07-11 16:55:25.916640	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		74	35622 → 8443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=878570435 TSecr=0 WS=128
4692	2022-07-11 16:55:25.916953	10.40.36.215	8443	10.40.36.46	35622	TCP	CS0		74	8443 → 35622 [ACK] Seq=0 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=843633230 TSecr=878570435 WS=128
4693	2022-07-11 16:55:25.916973	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=878570435 TSecr=843633230
4694	2022-07-11 16:55:25.917032	10.40.36.46	35622	10.40.36.215	8443	TLSv1.2	CS0		583	Client Hello
4695	2022-07-11 16:55:25.938356	10.40.36.215	8443	10.40.36.46	35622	TLSv1.2	CS0		1514	Server Hello
4696	2022-07-11 16:55:25.938390	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=518 Ack=1449 Win=64128 Len=0 TSval=878570457 TSecr=343633251
4697	2022-07-11 16:55:25.938489	10.40.36.215	8443	10.40.36.46	35622	TLSv1.2	CS0		1470	Certificate, Server Key Exchange, Server Hello Done
4698	2022-07-11 16:55:25.938419	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=518 Ack=2853 Win=63488 Len=0 TSval=878570457 TSecr=343633251
4699	2022-07-11 16:55:25.940187	10.40.36.46	35622	10.40.36.215	8443	TLSv1.2	CS0		192	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
4700	2022-07-11 16:55:25.943034	10.40.36.215	8443	10.40.36.46	35622	TLSv1.2	CS0		308	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
4701	2022-07-11 16:55:25.943051	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=644 Ack=3095 Win=64128 Len=0 TSval=878570461 TSecr=343633256
4702	2022-07-11 16:55:25.943277	10.40.36.46	35622	10.40.36.215	8443	TLSv1.2	CS0		2543	Application Data
4703	2022-07-11 16:55:25.943476	10.40.36.215	8443	10.40.36.46	35622	TCP	CS0		66	8443 → 35622 [ACK] Seq=3095 Ack=3121 Win=35072 Len=0 TSval=843633256 TSecr=878570462
4704	2022-07-11 16:55:25.943476	10.40.36.215	8443	10.40.36.46	35622	TCP	CS0		1514	8443 → 35622 [ACK] Seq=3095 Ack=3121 Win=35072 Len=1440 TSval=843633260 TSecr=878570462 [TCP segment of a reassembled PDU]
4705	2022-07-11 16:55:25.943476	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=3121 Ack=543 Win=63488 Len=0 TSval=878570473 TSecr=343633268
4706	2022-07-11 16:55:25.954842	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		1257	Application Data
4707	2022-07-11 16:55:25.954873	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		66	35622 → 8443 [ACK] Seq=3121 Ack=5734 Win=63488 Len=0 TSval=878570473 TSecr=343633268
4711	2022-07-11 16:55:25.955712	10.40.36.46	35622	10.40.36.215	8443	TLSv1.2	CS0		97	Encrypted Alert
4712	2022-07-11 16:55:25.955710	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		66	35622 → 8443 [FIN, ACK] Seq=3152 Ack=5734 Win=64128 Len=0 TSval=878570474 TSecr=343633268
4714	2022-07-11 16:55:25.956123	10.40.36.215	8443	10.40.36.46	35622	TLSv1.2	CS0		97	Encrypted Alert
4715	2022-07-11 16:55:25.956170	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		54	35622 → 8443 [RST] Seq=3153 Win=0 Len=0
4716	2022-07-11 16:55:25.956222	10.40.36.215	8443	10.40.36.46	35622	TCP	CS0		66	8443 → 35622 [SYN, ACK] Seq=3153 Win=64128 Len=0 MSS=1460 SACK_PERM=1 TSval=843633260 TSecr=878570474
4717	2022-07-11 16:55:25.956252	10.40.36.46	35622	10.40.36.215	8443	TCP	CS0		54	35622 → 8443 [RST] Seq=3153 Win=0 Len=0

```
Certificates (2423 bytes)
Certificate Length: 1507
Certificate: 308205f208202c7a0030201020113450000012205060803... (id-at-commonName=cucm-ms.steven.lab,id-at-organizationalUnitName=TAC,id-at-organizationName=Cisco,id-at-localityName=Diegen,id-at-stateOrProvinceName=Belgium,id-at-countryName=BE)
  signedCertificate
  version: v3 (3)
  serialNumber: 0x450000012205060803480044200000000122
  signature (sha1WithRSAEncryption)
  issuer: rdSequence (0)
  validity
  subject: rdSequence (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: cups.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-pe-authorityInfoAccessSyntax)
  Extension (id-ms-certificate-template)
  Extension (id-ms-application-certificate-policies)
  algorithmIdentifier (sha1WithRSAEncryption)
  padding: 0
  encrypted: 9f0a7f874637a2a92071ef608f2270ccc7ec4a470c82b...
Certificate Length: 910
Certificate: 308203a30820272a0030201020110740e62271e3d1346... (id-at-commonName=steven-DC-CA,dc=steven,dc=lab)
Secure Sockets Layer
```

Mas quando inspecionamos o certificado apresentado na porta 6972, você pode ver que ele é um certificado autoassinado (o próprio Emissor) com CN configurado como cucm-EC.steven.lab. A extensão -EC dá a indicação de que este é o certificado ECDSA configurado no CUCM.

No.	Time	Source	Src port	Destination	Dest port	Protocol	DSCP	VLAN	Length	Info
4730	2022-07-11 16:55:26.000600	10.40.36.46	31576	10.40.36.215	6972	TCP	CS0		74	31576 → 6972 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=878570525 TSecr=0 WS=128
4731	2022-07-11 16:55:26.000851	10.40.36.215	6972	10.40.36.46	31576	TCP	CS0		74	6972 → 31576 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=843633320 TSecr=878570525 WS=128
4732	2022-07-11 16:55:26.000892	10.40.36.46	31576	10.40.36.215	6972	TCP	CS0		66	31576 → 6972 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=878570515 TSecr=343633320
4733	2022-07-11 16:55:26.007100	10.40.36.46	31576	10.40.36.215	6972	TLSv1.2	CS0		583	Client Hello
4734	2022-07-11 16:55:26.016350	10.40.36.215	6972	10.40.36.46	31576	TLSv1.2	CS0		1514	Server Hello, Certificate, Server Key Exchange
4735	2022-07-11 16:55:26.016391	10.40.36.46	31576	10.40.36.215	6972	TCP	CS0		66	31576 → 6972 [ACK] Seq=518 Ack=1449 Win=64128 Len=0 TSval=878570535 TSecr=343633329
4736	2022-07-11 16:55:26.016400	10.40.36.215	6972	10.40.36.46	31576	TLSv1.2	CS0		499	Certificate Request, Server Hello Done
4737	2022-07-11 16:55:26.016419	10.40.36.46	31576	10.40.36.215	6972	TCP	CS0		66	31576 → 6972 [ACK] Seq=518 Ack=1882 Win=63744 Len=0 TSval=878570535 TSecr=343633329
4738	2022-07-11 16:55:26.016703	10.40.36.46	31576	10.40.36.215	6972	TLSv1.2	CS0		73	Alert (Level: Fatal), Description: Unknown CA
4739	2022-07-11 16:55:26.016821	10.40.36.46	31578	10.40.36.215	6972	TCP	CS0		74	31578 → 6972 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=878570535 TSecr=0 WS=128
4740	2022-07-11 16:55:26.016945	10.40.36.215	31576	10.40.36.46	6972	TCP	CS0		66	31576 → 6972 [RST] Seq=525 Ack=1882 Win=0 Len=0 TSval=878570535 TSecr=343633329
4741	2022-07-11 16:55:26.017015	10.40.36.215	6972	10.40.36.46	31578	TCP	CS0		74	6972 → 31578 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=843633330 TSecr=878570535 WS=128
4742	2022-07-11 16:55:26.017009	10.40.36.46	31578	10.40.36.215	6972	TCP	CS0		66	31578 → 6972 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=878570515 TSecr=343633330
4743	2022-07-11 16:55:26.017101	10.40.36.215	6972	10.40.36.46	31576	TCP	CS0		66	6972 → 31576 [FIN, ACK] Seq=1882 Ack=525 Win=30800 Len=0 TSval=843633330 TSecr=878570535
4744	2022-07-11 16:55:26.017111	10.40.36.46	31576	10.40.36.215	6972	TCP	CS0		54	31576 → 6972 [RST] Seq=525 Win=0 Len=0
4745	2022-07-11 16:55:26.017210	10.40.36.46	31578	10.40.36.215	6972	TLSv1.2	CS0		583	Client Hello
4746	2022-07-11 16:55:26.020266	10.40.36.215	6972	10.40.36.46	31578	TLSv1.2	CS0		1514	Server Hello, Certificate, Server Key Exchange
4747	2022-07-11 16:55:26.020265	10.40.36.46	31578	10.40.36.215	6972	TCP	CS0		66	31578 → 6972 [ACK] Seq=518 Ack=1449 Win=64128 Len=0 TSval=878570543 TSecr=343633337
4748	2022-07-11 16:55:26.020298	10.40.36.215	6972	10.40.36.46	31578	TLSv1.2	CS0		500	Certificate Request, Server Hello Done
4749	2022-07-11 16:55:26.020309	10.40.36.46	31578	10.40.36.215	6972	TCP	CS0		66	31578 → 6972 [ACK] Seq=518 Ack=1883 Win=63744 Len=0 TSval=878570543 TSecr=343633337
4750	2022-07-11 16:55:26.020667	10.40.36.46	31578	10.40.36.215	6972	TLSv1.2	CS0		73	Alert (Level: Fatal), Description: Unknown CA
4751	2022-07-11 16:55:26.020667	10.40.36.46	31576	10.40.36.215	6972	TCP	CS0		66	31576 → 6972 [RST] Seq=525 Ack=1882 Win=0 Len=0 TSval=878570543 TSecr=343633337
4767	2022-07-11 16:55:26.083159	10.40.36.46	31580	10.40.36.215	6972	TCP	CS0		74	31580 → 6972 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=878570601 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 (660 bytes)
  Certificate Length: 657
  Certificate: 308203a30820214e0030201020110740e62271e3d1346... (id-at-localityName=Diegen,id-at-stateOrProvinceName=Belgium,id-at-commonName=cucm-EC.steven.lab,id-at-organizationalUnitName=TAC,id-at-organizationName=Cisco,id-at-countryName=BE)
  signedCertificate
  version: v3 (3)
  serialNumber: 0x7470ee62271e3d1346109946f0a3f0fd
  signature (ecdsa-with-SHA384)
  issuer: rdSequence (0)
  rdSequence: 6 items (id-at-localityName=Diegen,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: rdSequence (0)
  subjectPublicKeyInfo
  extensions: 5 items
  Extension (id-ce-keyUsage)
  Extension (id-ce-extKeyUsage)
  Extension (id-ce-subjectKeyIdentifier)
  Extension (id-ce-basicConstraints)
  Extension (id-ce-subjectAltName)
  Extension Id: 2.5.29.17 (id-ce-subjectAltName)
  GeneralNames: 1 item
  GeneralName: dNSName (2)
  dNSName: cucm.steven.lab
  algorithmIdentifier (ecdsa-with-SHA384)
  padding: 0
  encrypted: 30642020434050e74570b11710e489ff030e6cd00d9...
  TLSv1.2 Record Layer: Handshake Protocol: Server Key Exchange
```

No CUCM, em Cisco Unified OS Administration, você pode ver os certificados em vigor em Segurança > Gerenciamento de certificado, como mostrado aqui, por exemplo. Ele mostra um certificado diferente para tomcat e tomcat-ECDSA em que o tomcat é assinado pela CA (e confiável pelo Expressway-C) enquanto o certificado tomcat-ECDSA é autoassinado e não confiável pelo Expressway-C aqui.

Certificate	Common Name	Type	Key Type	Distribution	Issued By	Expiration	Description
avt2	ALTHZ_cucm.steven.lab	Self-signed	RSA	cucm.steven.lab	ALTHZ_cucm.steven.lab	07/12/2038	Self-signed certificate generated by system
CallManager	cucm.steven.lab	CA-signed	RSA	cucm.steven.lab	stevenc-CA	07/13/2022	Certificate Signed by steven-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	stevenc-CA	Self-signed	RSA	stevenc-CA	stevenc-CA	06/01/2025	Trust Certificate
CallManager-trust	NOMAT-AD-CA	Self-signed	RSA	NOMAT-AD-CA	NOMAT-AD-CA	04/23/2028	Signed Certificate
CallManager-trust	CAP-RTP-002	Self-signed	RSA	CAP-RTP-002	CAP-RTP-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	CAPF-4b204648	Self-signed	RSA	CAPF-4b204648	CAPF-4b204648	04/12/2020	
CallManager-trust	ms-402-CA-1	Self-signed	RSA	ms-402-CA-1	ms-402-CA-1	09/11/2024	vngtp-CA
CallManager-trust	CAP-RTP-001	Self-signed	RSA	CAP-RTP-001	CAP-RTP-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	11/12/2037	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	vngtp-ACTIVE-DRR-CA	Self-signed	RSA	vngtp-ACTIVE-DRR-CA	vngtp-ACTIVE-DRR-CA	02/10/2024	VNGTP-CA
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_M2	11/12/2037	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-bantrum
CallManager-trust	CAPF-6164213c	Self-signed	RSA	CAPF-6164213c	CAPF-6164213c	07/12/2025	Self-signed certificate generated by system
CAPF	CAPF-6164213c	Self-signed	RSA	cucm.steven.lab	CAPF-6164213c	07/12/2025	Self-signed certificate generated by system
CAPF-trust	CAP-RTP-002	Self-signed	RSA	CAP-RTP-002	CAP-RTP-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.
CAPF-trust	CAPF-4b204648	Self-signed	RSA	CAPF-4b204648	CAPF-4b204648	04/12/2020	
CAPF-trust	CAP-RTP-001	Self-signed	RSA	CAP-RTP-001	CAP-RTP-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.
CAPF-trust	Cisco_Root_CA_M2	Self-signed	RSA	Cisco_Root_CA_M2	Cisco_Root_CA_M2	11/12/2037	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.
CAPF-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.
CAPF-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.
CAPF-trust	Cisco_Manufacturing_CA_SHA2	CA-signed	RSA	Cisco_Manufacturing_CA_SHA2	Cisco_Root_CA_M2	11/12/2037	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.
CAPF-trust	CAPF-6164213c	Self-signed	RSA	CAPF-6164213c	CAPF-6164213c	07/12/2025	Self-signed certificate generated by system
ipsec	cucm.steven.lab	Self-signed	RSA	cucm.steven.lab	cucm.steven.lab	07/12/2025	Self-signed certificate generated by system
ipsec-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	ITLRECOVERY_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	stevenc-CA	07/10/2024	Certificate Signed by steven-DC-CA
tomcat-ECDSA	cucm-EC.steven.lab	CSR Only	EC	cucm.steven.lab	---	---	---
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-trust	stevenc-CA	Self-signed	RSA	stevenc-CA	stevenc-CA	06/01/2025	Trust Certificate
tomcat-trust	NOMAT-AD-CA	Self-signed	RSA	NOMAT-AD-CA	NOMAT-AD-CA	04/23/2028	Signed 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-EC.steven.lab	CA-signed	EC	cucm.steven.lab	stevenc-CA	07/10/2024	Trust Certificate
tomcat-trust	cups-EC.steven.lab	Self-signed	EC	cups.steven.lab	cups-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	vngtp-ACTIVE-DRR-CA	Self-signed	RSA	vngtp-ACTIVE-DRR-CA	vngtp-ACTIVE-DRR-CA	02/10/2024	Trust Certificate
tomcat-trust	dcomics-WONDERWOMAN-CA	Self-signed	RSA	dcomics-WONDERWOMAN-CA	dcomics-WONDERWOMAN-CA	09/19/2037	CA-brane
TVS	cucm.steven.lab	Self-signed	RSA	cucm.steven.lab	cucm.steven.lab	07/12/2025	Self-signed certificate generated by system

2. O endereço de conexão (FQDN ou IP) não está contido no certificado

Além do armazenamento confiável, o servidor de tráfego também verifica o endereço de conexão para o qual o cliente MRA faz a solicitação. Por exemplo, quando você tiver configurado no CUCM em **System > Server** seu CUCM com o endereço IP (10.48.36.215), o Expressway-C anunciará isso como tal ao cliente e as solicitações subsequentes do cliente (com proxy através do Expressway-C) serão direcionadas a esse endereço.

Quando esse endereço de conexão específico não está contido no certificado do servidor, a verificação TLS também falha e um erro 502 é lançado, resultando em falha de login de MRA, por exemplo.

```
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 vscs 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 vscs 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
```

Em que `c3RldmVuLmxhYi9odHRwcy8xMC40OC4zNi4yMTUvODQ0Mw` converte (base64 - <https://www.base64decode.org/>) para `steven.lab/https/10.48.36.215/8443`, que mostra que deve fazer a conexão em direção a `10.48.36.215` como o endereço de conexão, em vez de para `cucm.steven.lab`. Como mostrado nas capturas de pacote, o certificado tomcat CUCM não contém o endereço IP na SAN e, portanto, o erro é lançado.

Como validá-lo facilmente

É possível validar se você se depara com essa mudança de comportamento facilmente com as próximas etapas:

1. Inicie o log de diagnóstico no(s) servidor(es) Expressway-E e C (idealmente com TCPDumps ativados) de **Manutenção > Diagnóstico > Log de Diagnóstico** (no caso de um cluster, é suficiente iniciá-lo a partir do nó mestre)
2. Tente um login de MRA ou teste a funcionalidade interrompida após a atualização
3. Aguarde até que haja falha e pare o log de diagnóstico no(s) servidor(es) Expressway-E e C (no caso de um cluster, certifique-se de coletar os logs de cada nó individual do cluster individualmente)
4. Carregue e analise os logs na [ferramenta Collaboration Solution Analyzer](#)
5. Se você encontrar o problema, ele selecionará as linhas de aviso e de erro mais recentes relacionadas a essa alteração para cada um dos servidores afetados

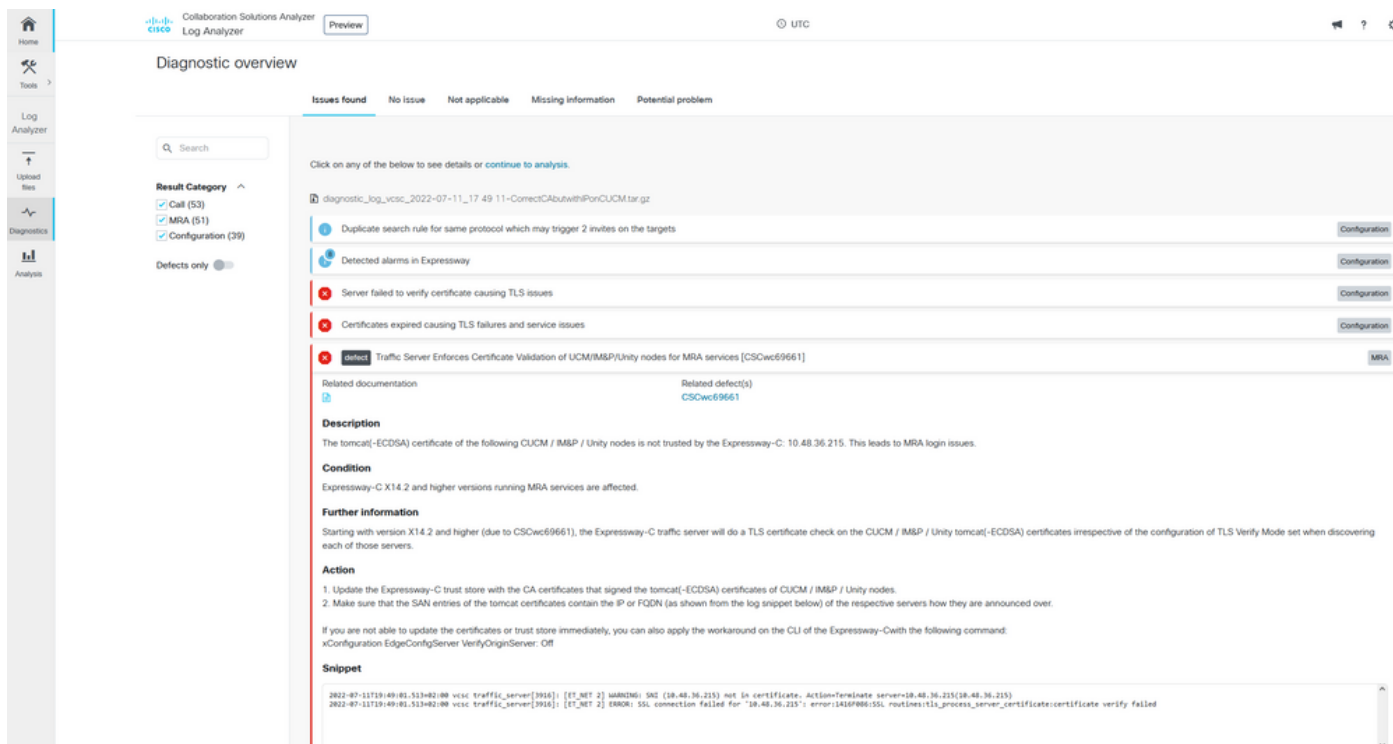
The screenshot shows the Cisco Collaboration Solutions Analyzer Log Analyzer interface. The main panel displays a 'Diagnostic overview' with a search bar and a list of issues. The selected issue is 'Traffic Server Enforces Certificate Validation of UCM/IMP/Unity nodes for MRA services [CSCw69661]'. The interface includes a sidebar with navigation options like Home, Tools, Log Analyzer, and Analysis. The main content area shows the following details for the selected issue:

- Description:** The tomcat-(ECDSA) certificate of the following CUCM / IMP / Unity nodes is not trusted by the Expressway-C: cucm.steven.lab, 10.48.36.215. This leads to MRA login issues.
- Condition:** Expressway-C X14.2 and higher versions running MRA services are affected.
- Further information:** Starting with version X14.2 and higher (due to CSCw69661), the Expressway-C traffic server will do a TLS certificate check on the CUCM / IMP / Unity tomcat-(ECDSA) certificates irrespective of the configuration of TLS Verify Mode set when discovering each of those servers.
- Action:**
 1. Update the Expressway-C trust store with the CA certificates that signed the tomcat-(ECDSA) certificates of CUCM / IMP / Unity nodes.
 2. Make sure that the SAN entries of the tomcat certificates contain the IP or FQDN (as shown from the log snippet below) of the respective servers how they are announced over.

If you are not able to update the certificates or trust store immediately, you can also apply the workaround on the CLI of the Expressway-C with the following command:
`>Configuration EdgeConfigServer VerifyOriginServer Off`

Snippet:

```
2022-07-11T19:33:06.740+02:00 vscs traffic_server[3956]: [ET_NET 0] WARNING: Core server certificate verification failed for (10.48.36.215). Action=Terminate Error=self signed certificate in certificate chain server=10.48.36.215(10.48.36.215) depth=1
2022-07-11T19:33:06.740+02:00 vscs traffic_server[3956]: [ET_NET 0] ERROR: SSL connection failed for '10.48.36.215': error:1416F086:SSL routines:tls_process_server_certificate:certificate verify failed
2022-07-11T19:33:06.160+02:00 vscs traffic_server[3956]: [ET_NET 1] WARNING: Core server certificate verification failed for (cucm.steven.lab). Action=Terminate Error=self signed certificate in certificate chain server=cucm.steven.lab(10.48.36.215) depth=1
2022-07-11T19:33:06.160+02:00 vscs traffic_server[3956]: [ET_NET 1] ERROR: SSL connection failed for 'cucm.steven.lab': error:1416F086:SSL routines:tls_process_server_certificate:certificate verify failed
```

Assinatura de diagnóstico SNI

Solução

A solução de longo prazo é garantir que a verificação TLS funcione bem. A ação a ser executada depende da mensagem de aviso exibida.

Quando você observar o **AVISO: Falha na verificação do certificado do servidor núcleo para (<server-FQDN-or-IP>)**. Action=Terminate Error=self signed certificate server=cucm.steven.lab(10.48.36.215) **depth=x, então você precisa atualizar o armazenamento confiável nos servidores Expressway-C adequadamente.** Com a cadeia de CAs que assinou este certificado (profundidade > 0) ou com o certificado autoassinado (profundidade = 0) em **Manutenção > Segurança > Certificado de CA Confiável.** Certifique-se de executar esta ação em todos os servidores do cluster. Outra opção seria assinar o certificado remoto por uma CA conhecida no repositório de confiança do Expressway-C.

Quando você observar o **AVISO: SNI (<server-FQDN-or-IP>) não na mensagem de certificado,** indica que esse FQDN ou IP do servidor não está contido no certificado que foi apresentado. Você pode adaptar o certificado para incluir essas informações ou pode modificar a configuração (como no CUCM em Sistema > Servidor para algo que esteja contido no certificado do servidor) e, em seguida, atualizar a configuração no servidor Expressway-C para que ela seja levada em conta.

A solução de curto prazo é aplicar a solução alternativa conforme documentado para fazer fallback para o comportamento anterior antes de X14.2.0. Você pode executar isso através da CLI nos nós do servidor Expressway-C com o comando recém-introduzido:

```
xConfiguration EdgeConfigServer VerifyOriginServer: Off
```

Sobre esta tradução

A Cisco traduziu este documento com a ajuda de tecnologias de tradução automática e humana para oferecer conteúdo de suporte aos seus usuários no seu próprio idioma, independentemente da localização.

Observe que mesmo a melhor tradução automática não será tão precisa quanto as realizadas por um tradutor profissional.

A Cisco Systems, Inc. não se responsabiliza pela precisão destas traduções e recomenda que o documento original em inglês ([link fornecido](#)) seja sempre consultado.