Configurar Radius DTLS no ISE e no 9800 WLC

Contents

Introdução

Este documento descreve um método para criar os certificados necessários para configurar o RADIUS DTLS entre o ISE e o 9800 WLC.

Background

RADIUS DTLS é uma forma segura do protocolo RADIUS em que as mensagens RADIUS são enviadas por um túnel Data Transport Layer Security (DTLS). Para criar esse túnel entre o servidor de autenticação e o autenticador, é necessário um conjunto de certificados. Esse conjunto de certificados exige que determinadas extensões de certificado de Uso Estendido de Chave (EKU) sejam definidas, especificamente, a autenticação de cliente no certificado WLC e a autenticação de servidor, bem como a autenticação de cliente para o certificado ISE.

Pré-requisitos

Requisitos

A Cisco recomenda que você tenha conhecimento destes tópicos:

- Como configurar a WLC 9800, o ponto de acesso (AP) para operação básica
- Como usar a aplicação de OpenSSL
- Infraestrutura de Chave Pública (PKI) e certificados digitais

Componentes Utilizados

As informações neste documento são baseadas nestas versões de software e hardware:

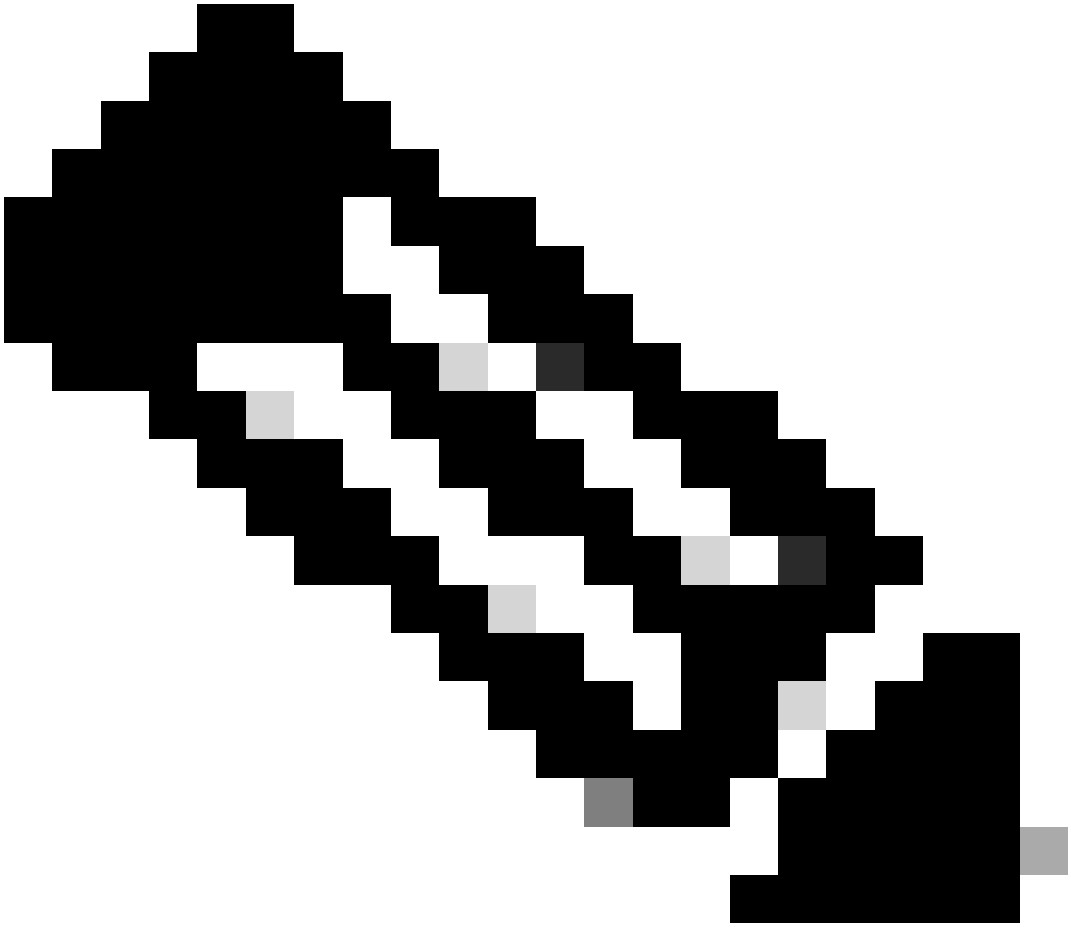
- Aplicativo OpenSSL (versão 3.0.2).
- ISE (versão 3.1.0.518)
- 9800 WLC (versão 17.12.3)

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. Todos os dispositivos utilizados neste documento foram iniciados com uma configuração (padrão) inicial. Se a rede estiver ativa, certifique-se de que você entenda o impacto potencial de qualquer comando.

Configurar

Overview

A finalidade é criar uma autoridade de certificação de dois níveis com uma CA raiz e uma CA intermediária para assinar certificados de ponto de extremidade. Uma vez assinados, os certificados são importados para a WLC e o ISE. Por fim, os dispositivos são configurados para executar a autenticação RADIUS DTLS com esses certificados.



Observação: este documento usa comandos específicos do Linux para criar e organizar arquivos. Os comandos são explicados para que você possa executar a mesma ação em outros sistemas operacionais em que o OpenSSL esteja disponível.

Opcional - Criar certificado de dispositivo DTLS RADIUS WLC e ISE

O protocolo RADIUS DTLS precisa trocar certificados entre o ISE e a WLC para criar o túnel DTLS. Se você ainda não tiver certificados válidos, poderá criar uma autoridade de certificação local para gerar os certificados, consulte Configurar uma autoridade de certificação multinível no OpenSSL para gerar certificados compatíveis com o Cisco IOS® XE e execute as etapas descritas no documento desde o início até o fim da etapa Criar certificado CA intermediário.

Adicionar seções de configuração no arquivo openssl.cnf

Abra o arquivo de configuração openssl.cnf e, na parte inferior, copie e cole as seções WLC e ISE usadas para gerar uma CSR (Certificate Sign Request, Solicitação de assinatura de certificado)

válida.

As seções ISE_device_req_ext e WLC_device_req_ext apontam cada uma para uma lista de SANs a serem incluídas no CSR:

```
#Section used for CSR generation, it points to the list of subject alternative names to add them to CSR
[ ISE_device_req_ext ]
subjectAltName = @ISE_alt_names

[ WLC_device_req_ext ]
subjectAltName = @WLC_alt_names

#DEFINE HERE SANS/IPs NEEDED for **ISE** device certificates
[ISE_alt_names]
DNS.1   = ISE.example.com
DNS.2   = ISE2.example.com

#DEFINE HERE SANS/IPs NEEDED for **WLC** device certificates
[WLC_alt_names]
DNS.1   = WLC.example.com
DNS.2   = WLC2.example.com
```

Como medida de segurança, a CA substitui qualquer SAN presente em um CSR para assiná-lo, de modo que dispositivos não autorizados não possam receber um certificado válido para um nome que não têm permissão para usar. Para adicionar as SANs de volta ao certificado assinado, use o parâmetro subjectAltName para apontar para as mesmas SANs de lista que as usadas para a geração de CSR.

O ISE requer EKUs serverAuth e clientAuth presentes no certificado, enquanto o WLC precisa apenas de clientAuth. Eles são adicionados ao certificado assinado com o parâmetro extendedKeyUsage.

Copie e cole as seções usadas para assinatura de certificado na parte inferior do arquivo openssl.cnf:

```
#This section contains the extensions used for the device certificate sign
[ ISE_cert ]
basicConstraints=CA:FALSE
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
#EKU client and server is needed for RADIUS DTLS on ISE
extendedKeyUsage = serverAuth, clientAuth
subjectAltName = @ISE_alt_names

[ WLC_cert ]
basicConstraints=CA:FALSE
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
#EKU client is needed for RADIUS DTLS on WLC
extendedKeyUsage = clientAuth
```

```
subjectAltName = @WLC_alt_names
```

Criar Certificado de Dispositivo WLC

Crie uma nova pasta para armazenar certificados de WLC no computador que tem o OpenSSL instalado dentro da pasta de certificados de CA intermediário chamada IntermCA.db.certs. A nova pasta é chamada de WLC:

```
mkdir ./IntermCA/IntermCA.db.certs/WLC
```

Modifique os parâmetros DNS na seção [WLC_alt_names] do arquivo openssl.cnf. Altere os nomes de exemplo fornecidos para os valores desejados. Esses valores preenchem o campo SANs do certificado da WLC:

```
[WLC_alt_names]
DNS.1   = WLC.example.com    <-----Change the values after the equals sign
DNS.2   = WLC2.example.com   <-----Change the values after the equals sign
```

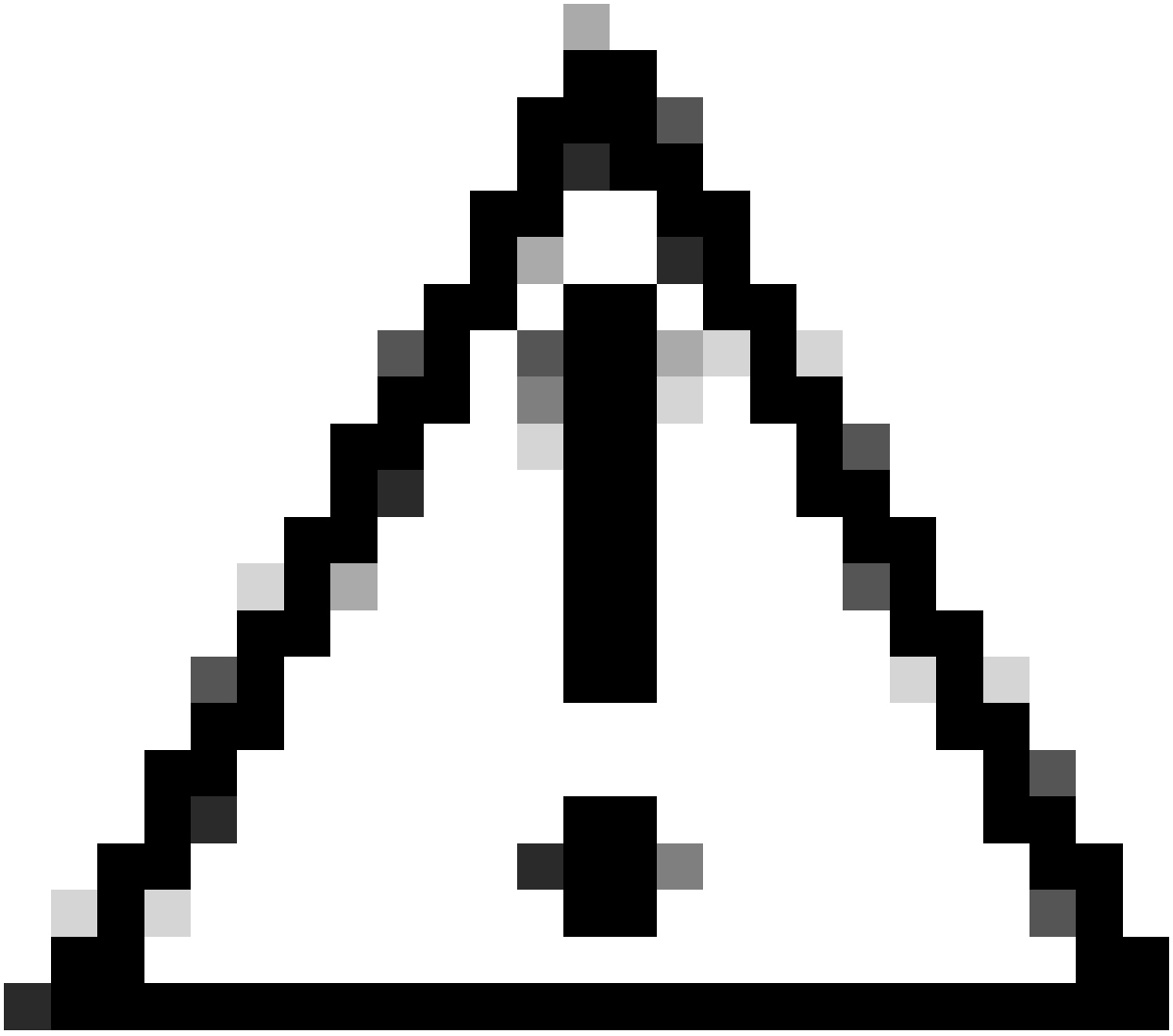
Crie a chave privada da WLC e o CSR da WLC com informações da seção WLC_device_req_ext para SANs:

```
openssl req -newkey rsa:4096 -keyout ./IntermCA/IntermCA.db.certs/WLC/WLC.key -nodes -config openssl.cn
```

O OpenSSL abre um prompt interativo para que você insira os detalhes do DN (Distinguished Name - Nome Distinto):

```
.........+..+.......+...........+...+......+.+...........+..++++++++++++++
++++++
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name [MX]:
State or province [CDMX]:
Locality [CDMX]:
Organization name [Cisco lab]:
Organizational unit [Cisco Wireless]:
Common name []:WLC.example.com
```

Prompt Interativo de Nome Distinto do Certificado WLC



Cuidado: o Nome Comum (CN) fornecido no prompt interativo deve ser idêntico a um dos

---

Nomes na seção [WLC_alt_names] do arquivo openssl.cnf.

Use a CA chamada IntermCA para assinar o CSR da WLC chamado WLC.csr com as extensões definidas em [WLC_cert] e armazenar o certificado assinado dentro de ./IntermCA/IntermCA.db.certs/WLC. O certificado do dispositivo WLC é chamado de WLC.crt:

```
openssl ca -config openssl.cnf -extensions WLC_cert -name IntermCA -out ./IntermCA/IntermCA.db.certs/WL
```

A WLC 9800 precisa que o certificado esteja no formato pfx para importá-lo. Crie um novo arquivo que contenha a cadeia de CAs que assinaram o certificado da WLC. Isso é chamado de arquivo de certificado:

```
cat ./RootCA/RootCA.crt ./IntermCA/IntermCA.crt > ./IntermCA/IntermCA.db.certs/WLC/certfile.crt
```

Para criar o arquivo .pfx, execute um desses comandos de acordo com a versão da WLC.

Para versões anteriores a 17.12.1:

```
openssl pkcs12 -export -macalg sha1 -legacy -descert -out ./IntermCA/IntermCA.db.certs/WLC/WLC.pfx -ink
```

Para a versão 17.12.1 ou posterior:

```
openssl pkcs12 -export -out ./IntermCA/IntermCA.db.certs/WLC/WLC.pfx -inkey ./IntermCA/IntermCA.db.cert
```

Criar certificado do dispositivo ISE

Crie uma nova pasta para armazenar certificados ISE no computador que tem o OpenSSL instalado dentro da pasta de certificados CA intermediários chamada IntermCA.db.certs. A nova pasta é chamada de ISE:

```
mkdir ./IntermCA/IntermCA.db.certs/ISE
```

Modifique os parâmetros DNS na seção [ISE_alt_names] do arquivo openssl.cnf. Altere os nomes de exemplo fornecidos para os valores desejados, esses valores preenchem o campo SANs do certificado WLC:

```
[ISE_alt_names]
DNS.1   = ISE.example.com   <-----Change the values after the equals sign
DNS.2   = ISE2.example.com  <-----Change the values after the equals sign
```

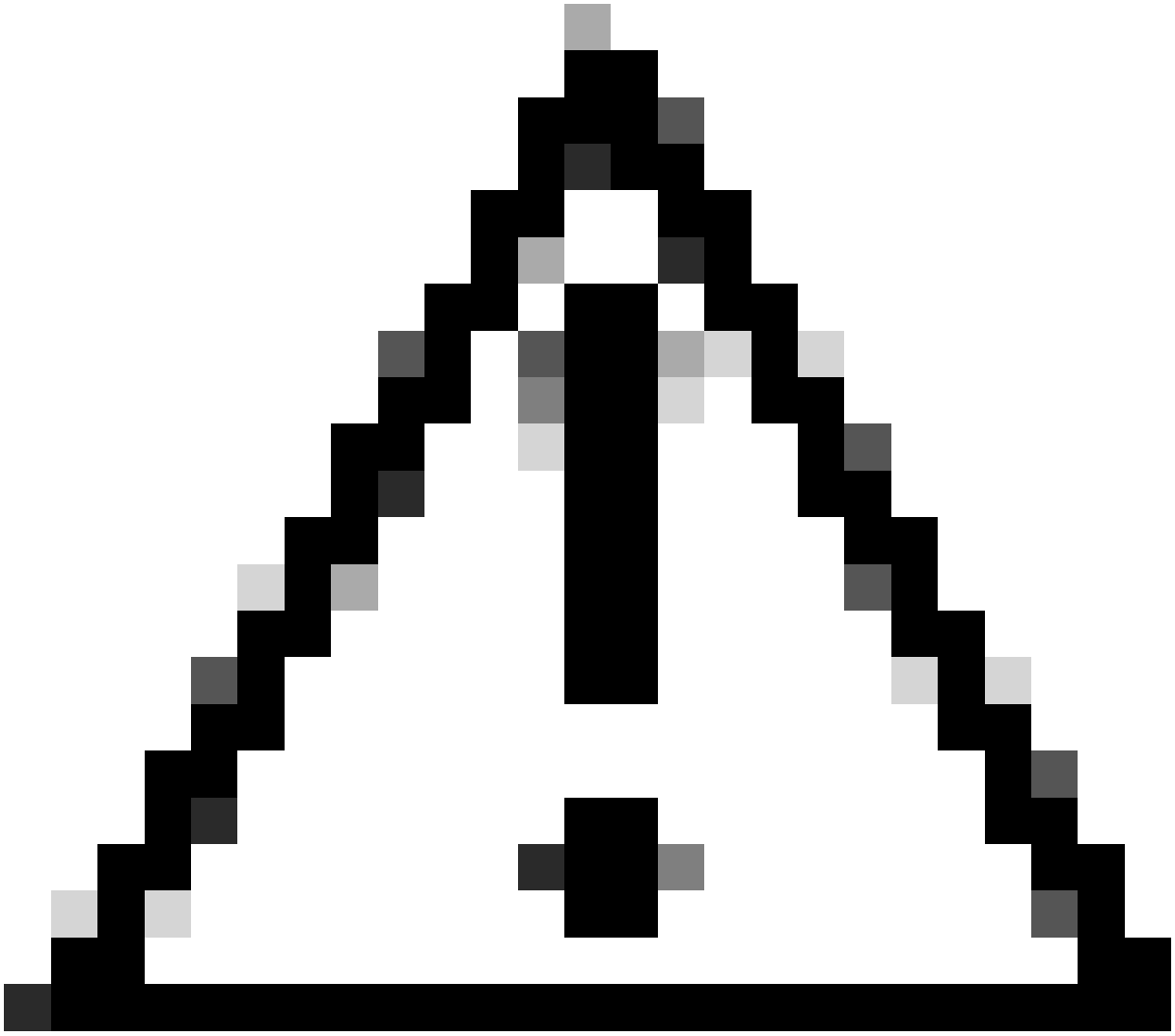
Crie a chave privada do ISE e o ISE CSR com informações da seção ISE_device_req_ext para SANs:

```
openssl req -newkey rsa:2048 -sha256 -keyout ./IntermCA/IntermCA.db.certs/ISE/ISE.key -nodes -config op
```

O OpenSSL abre um prompt interativo para que você insira os detalhes do DN (Distinguished Name - Nome Distinto):

```
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name [MX]:
State or province [CDMX]:
Locality [CDMX]:
Organization name [Cisco lab]:
Organizational unit [Cisco Wireless]:
Common name []:ISE.example.com
```

Prompt Interativo de Nome Distinto do Certificado ISE



Cuidado: o CN fornecido no prompt interativo deve ser exatamente o mesmo que um dos Nomes na seção [ISE_alt_names] do arquivo openssl.cnf.

Use a CA denominada IntermCA para assinar o ISE CSR denominado ISE.csr com as extensões definidas em [ISE_cert] e armazenar o certificado assinado em ./IntermCA/IntermCA.db.certs/WLC. O certificado do dispositivo ISE é chamado de ISE.crt:

```
openssl ca -config openssl.cnf -extensions ISE_cert -name IntermCA -out ./IntermCA/IntermCA.db.certs/IS
```

Importar certificados para dispositivos

Importar certificados para o ISE

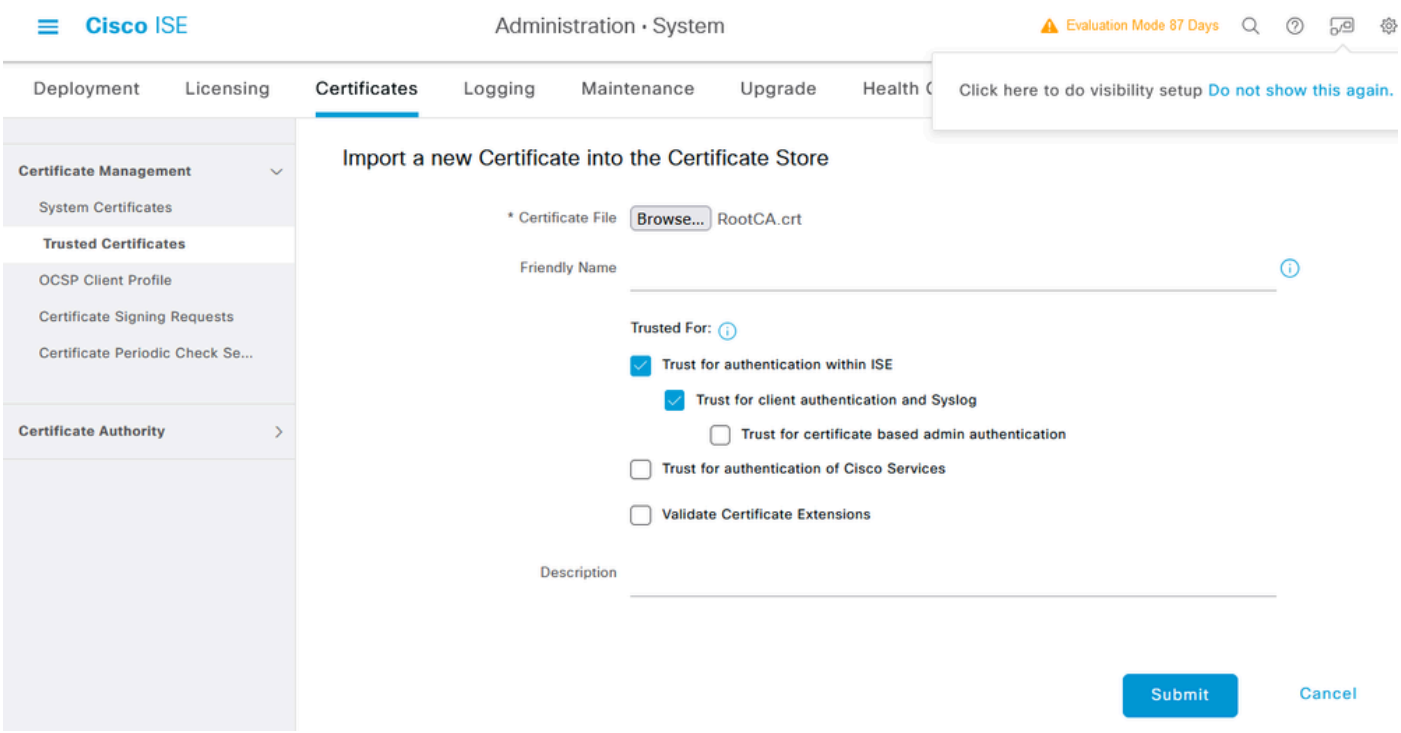
1. Importe o certificado de CA Raiz da cadeia de certificados ISE para o repositório de certificados

confiáveis.

2. Navegue até Administração>Sistema>Certificados>Certificados de Confiabilidade.

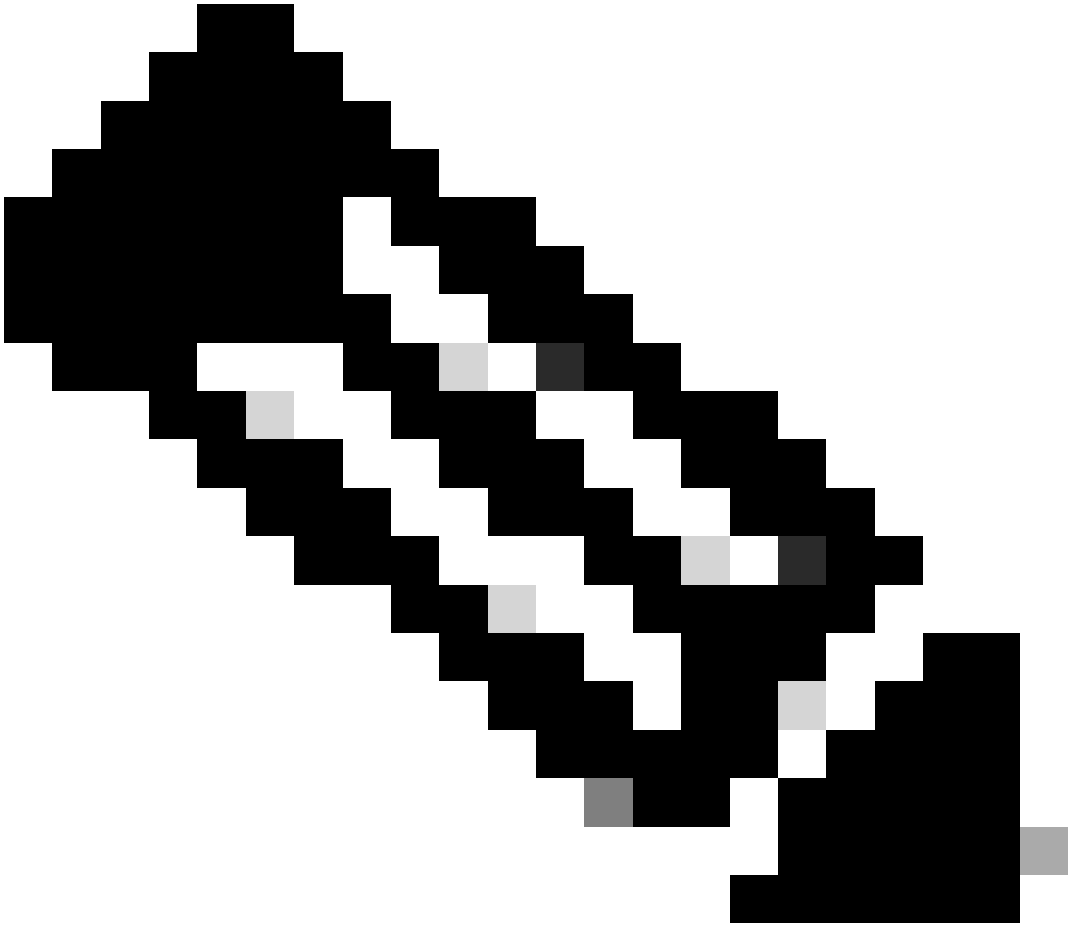
3. Clique em Procurar e selecione o arquivo Root.crt.

4. Marque as caixas de seleção Trust for authentication within ISE, bem como Trust for client authentication e Syslog e clique em Submit:

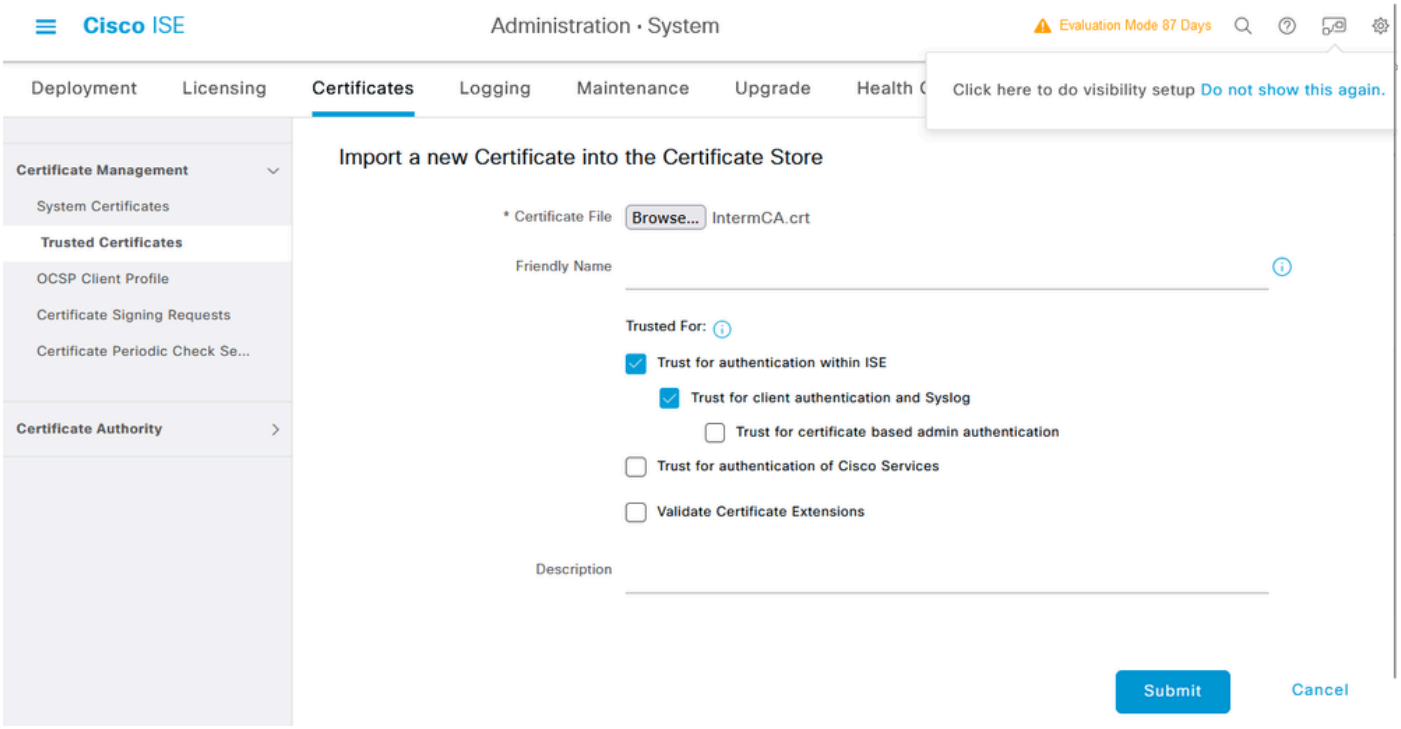


Diálogo de importação de certificado de CA raiz do ISE

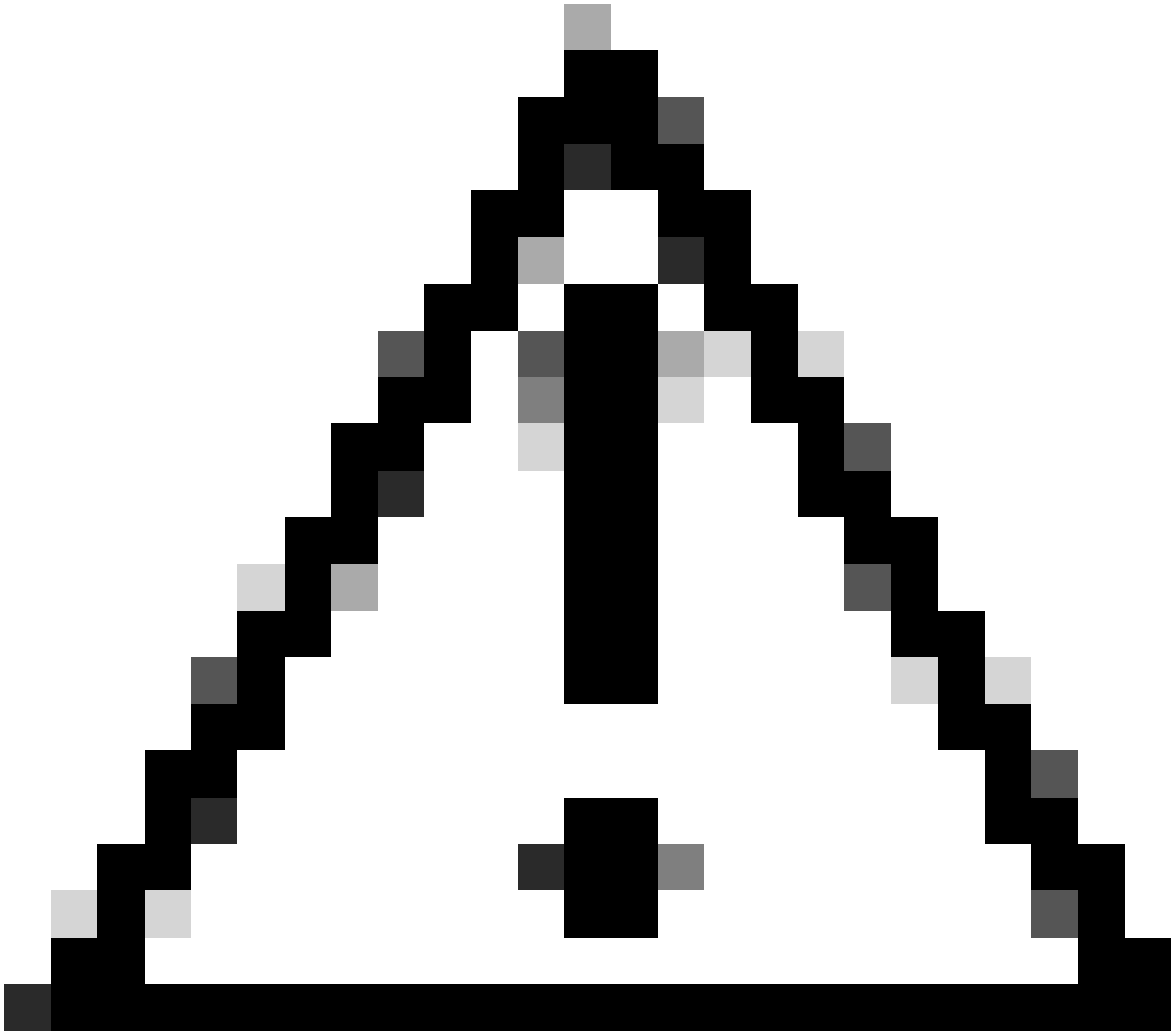
Faça o mesmo para o certificado intermediário, se ele existir.



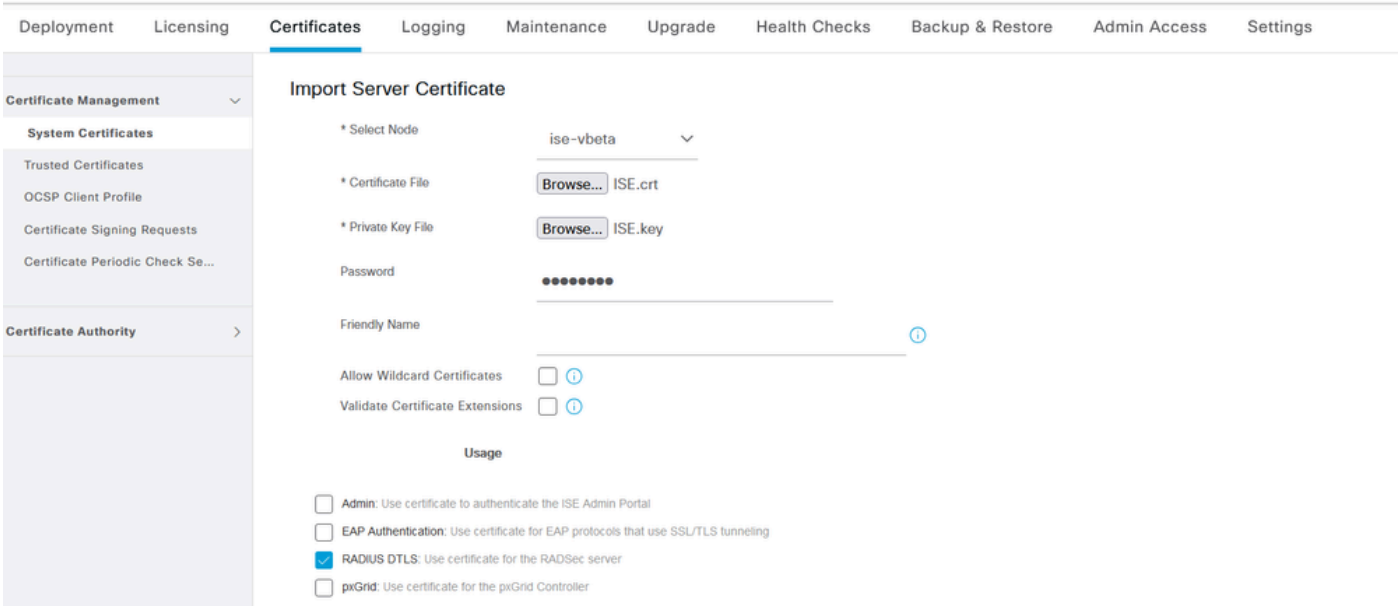
Observação: repita as etapas para qualquer certificado CA que faça parte da cadeia de validação do certificado ISE. Sempre comece com o certificado de CA raiz e termine com o certificado de CA intermediário mais baixo da cadeia.



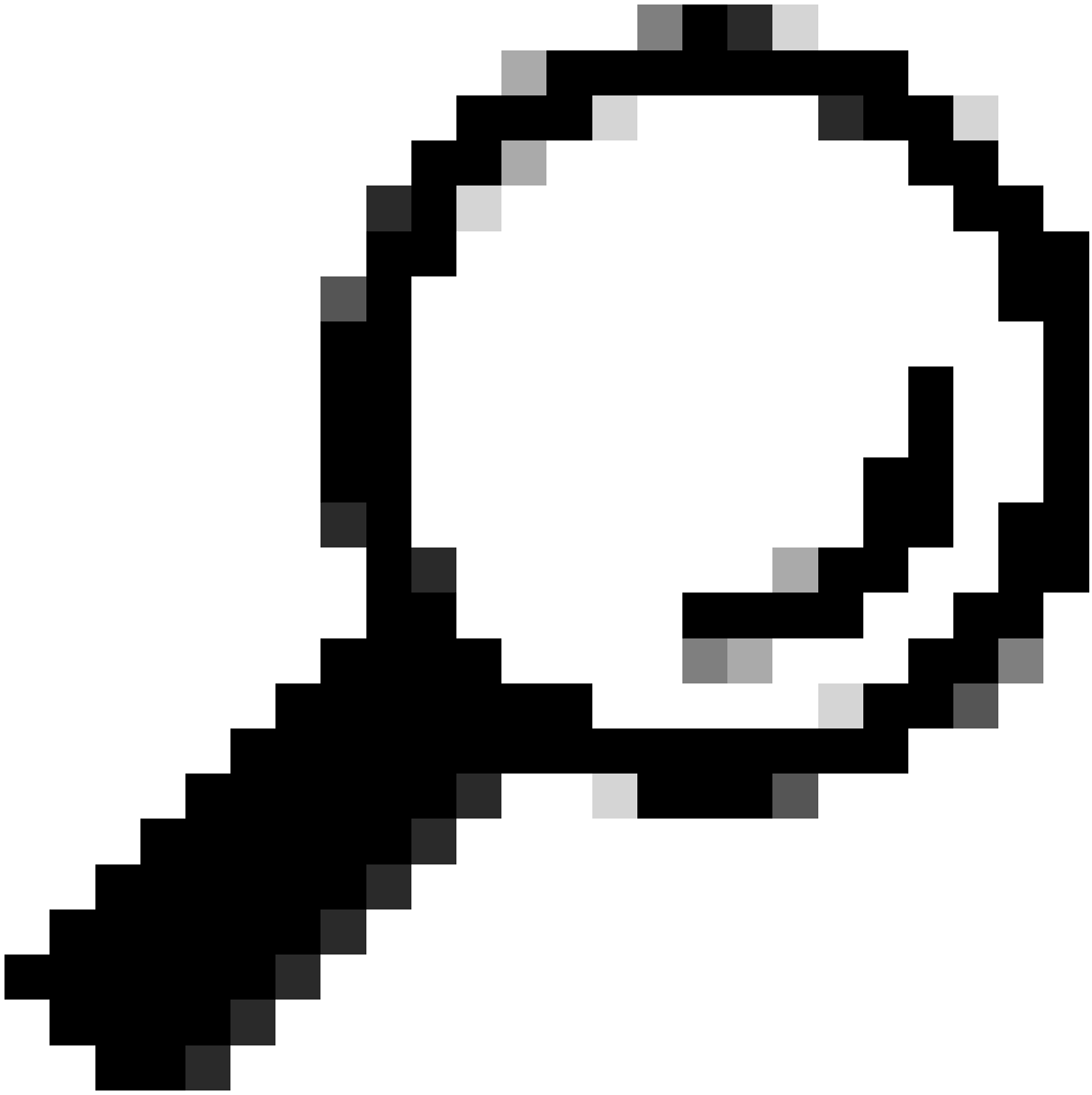
Caixa de Diálogo Importação de Certificado de CA Intermediário do ISE



Cuidado: se o certificado ISE e o certificado WLC forem emitidos por CAs diferentes, você também deverá importar todos os certificados CA que pertencem à cadeia de certificados WLC. O ISE não aceita o certificado WLC na troca de certificados DTLS até que você importe esses certificados CA.



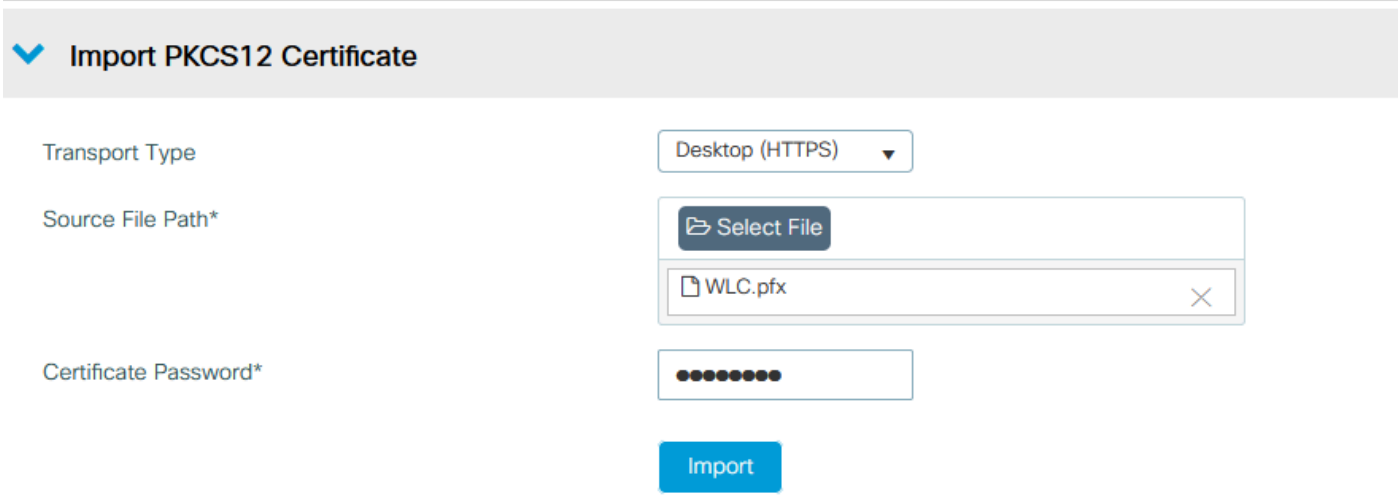
Menu de Importação de Certificado de Dispositivo do ISE



Dica: você só precisa importar o certificado do dispositivo ISE nesta etapa. Esse certificado é o único intercâmbio ISE para estabelecer o túnel DTLS. Não é necessário importar o certificado do dispositivo WLC e a chave privada, pois o certificado WLC é verificado com o uso dos certificados CA importados anteriormente.

Importar certificados para WLC

1. Navegue para Configuration > Security > PKI Management no WLC e vá para a guia Add Certificate.
2. Clique no menu suspenso Import PKCS12 Certificate e defina o tipo de transporte como Desktop (HTTPS).
3. Clique no botão Select File e selecione o arquivo .pfx que você preparou anteriormente.
4. Digite a senha de importação e clique em Import.

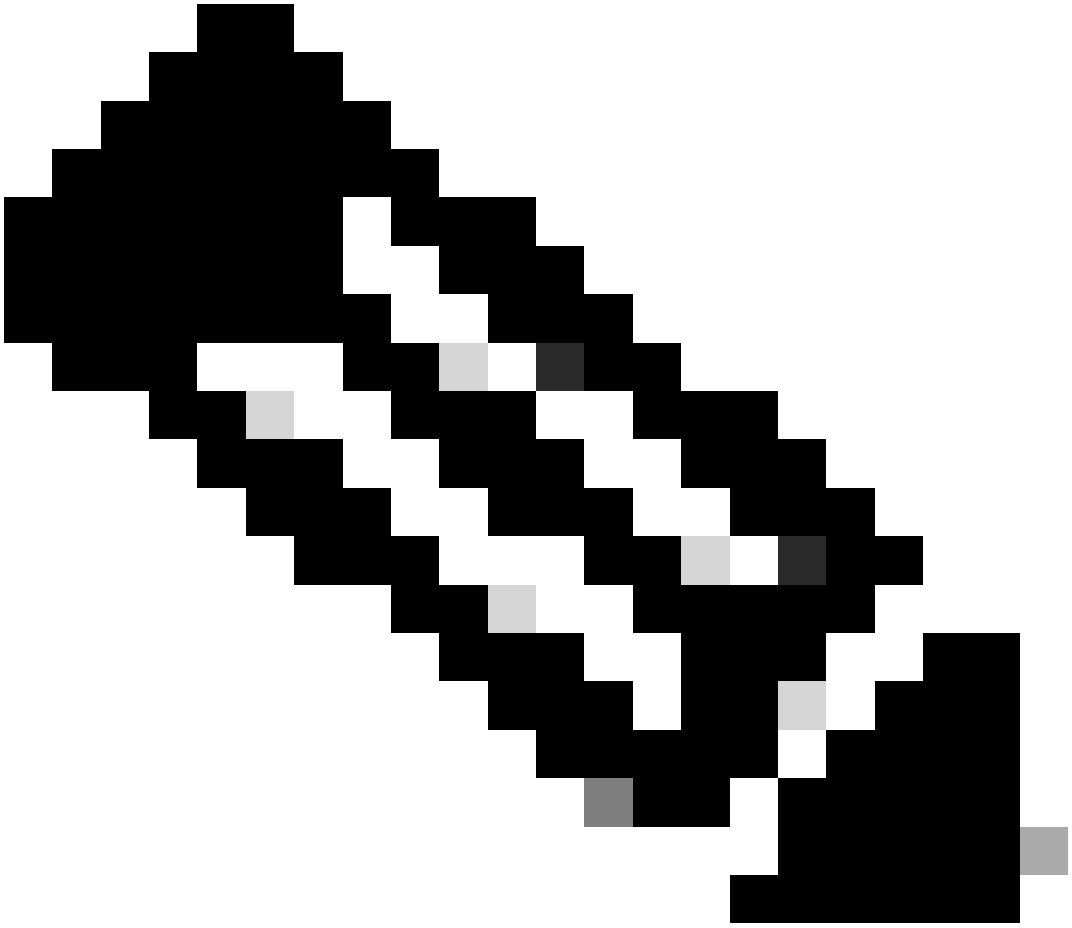


Caixa de Diálogo Importação de Certificado WLC

Para obter informações detalhadas sobre o processo de importação, consulte Gerar e fazer download de certificados CSR em WLCs do Catalyst 9800.

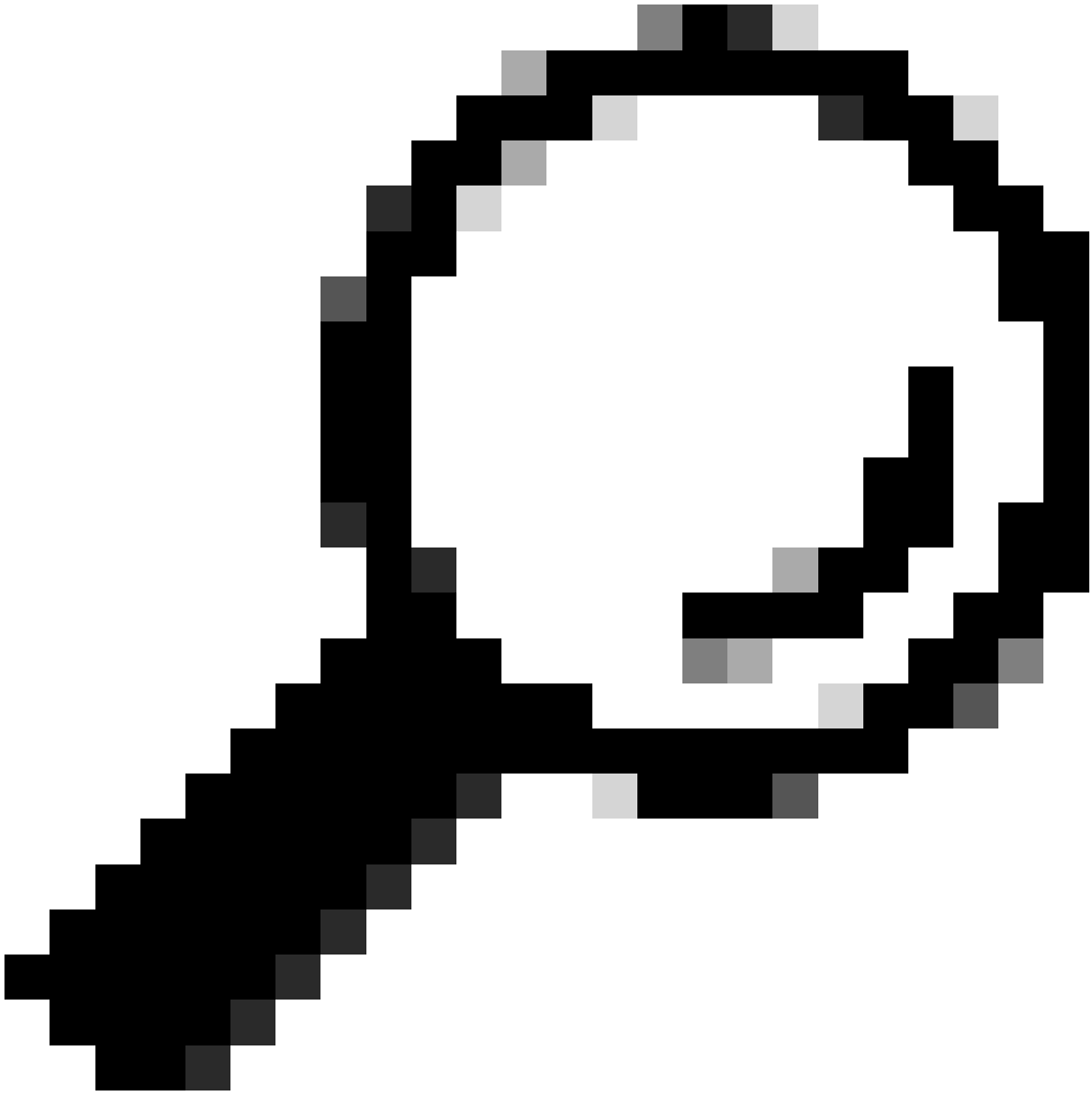
Desabilite a verificação de revogação dentro de cada ponto de confiança criado automaticamente se a WLC não tiver uma Lista de Revogação de Certificados que possa verificar através da rede:

```
9800#configure terminal

9800(config)#crypto pki trustpoint WLC.pfx
9800(config)#revocation-check none
9800(config)#exit

9800(config)#crypto pki trustpoint WLC.pfx-rrr1
9800(config)#revocation-check none
9800(config)#exit
```



Observação: se você criou uma CA de vários níveis no OpenSSL com o documento Configurar CA de vários níveis no OpenSSL para gerar certificados Cisco IOS XE, desabilite a verificação de revogação, pois nenhum servidor de CRL é criado.

A importação automatizada cria os pontos de confiança necessários para conter o certificado WLC e seus certificados CA.



Dica: se os certificados WLC tiverem sido emitidos pela mesma CA que os certificados ISE, você poderá usar os mesmos pontos confiáveis criados automaticamente a partir da importação do certificado WLC. Não há necessidade de importar os certificados ISE separadamente.

Se o certificado da WLC for emitido por uma CA diferente do certificado do ISE, você também precisará importar os certificados da CA do ISE para a WLC para que a WLC confie no certificado do dispositivo do ISE.
Crie um novo ponto confiável para a CA raiz e importe a CA raiz do ISE:

```
9800(config)#crypto pki trustpoint ISEroot
```

```
9800(ca-trustpoint)#revocation-check none
9800(ca-trustpoint)#enrollment terminal
9800(ca-trustpoint)#chain-validation stop
9800(ca-trustpoint)#exit
9800(config)#crypto pki authenticate ISEroot

Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

------Paste the ISE root CA-----
```

Importe o próximo certificado de CA intermediário na cadeia de CA do ISE, em outras palavras, o certificado de CA Emitido pela CA raiz:

```
hamariomed1(config)#crypto pki trustpoint ISEintermediate
hamariomed1(ca-trustpoint)#revocation-check none
hamariomed1(ca-trustpoint)#chain-validation continue ISErootCA
hamariomed1(ca-trustpoint)#enrollment terminal
hamariomed1(ca-trustpoint)#exit

hamariomed1(config)#crypto pki authenticate ISEintermediate

Enter the base 64 encoded CA certificate.
End with a blank line or the word "quit" on a line by itself

------Paste the ISE intermediate CA-------
```

Cada CA adicional na cadeia requer um ponto de confiança separado. Cada ponto confiável na cadeia deve fazer referência ao ponto confiável que contém o certificado do emissor do certificado que você deseja importar com o comando chain-validation continue <Nome do ponto confiável do emissor>.

Importe quantos certificados CA sua cadeia de CA contiver. Você terminou depois de importar a CA do emissor do certificado do dispositivo ISE, anote o nome deste ponto de confiança.

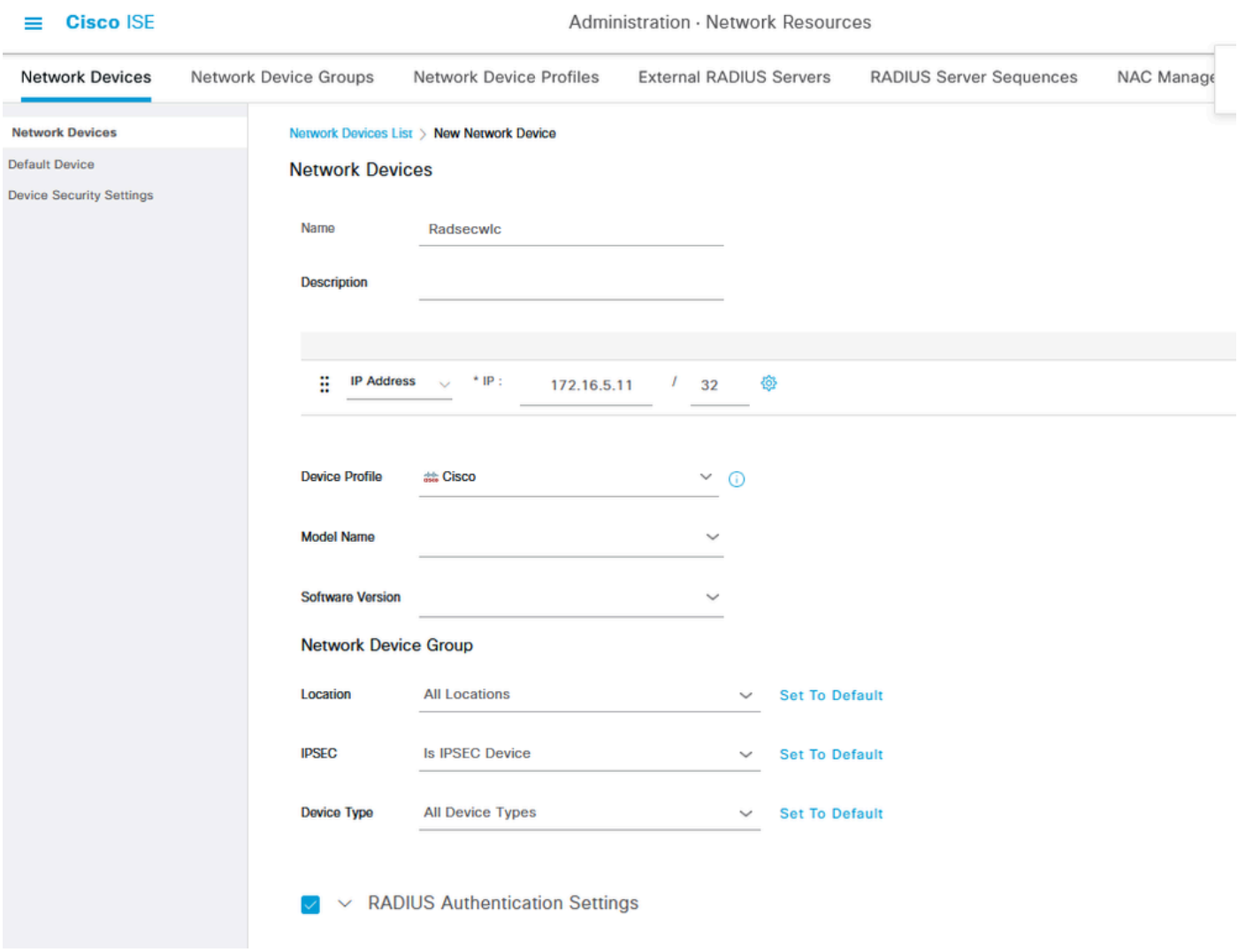
Você não precisa importar o certificado do dispositivo ISE no WLC para que o RADIUS DTLS funcione.

Configurar RADIUS DTLS

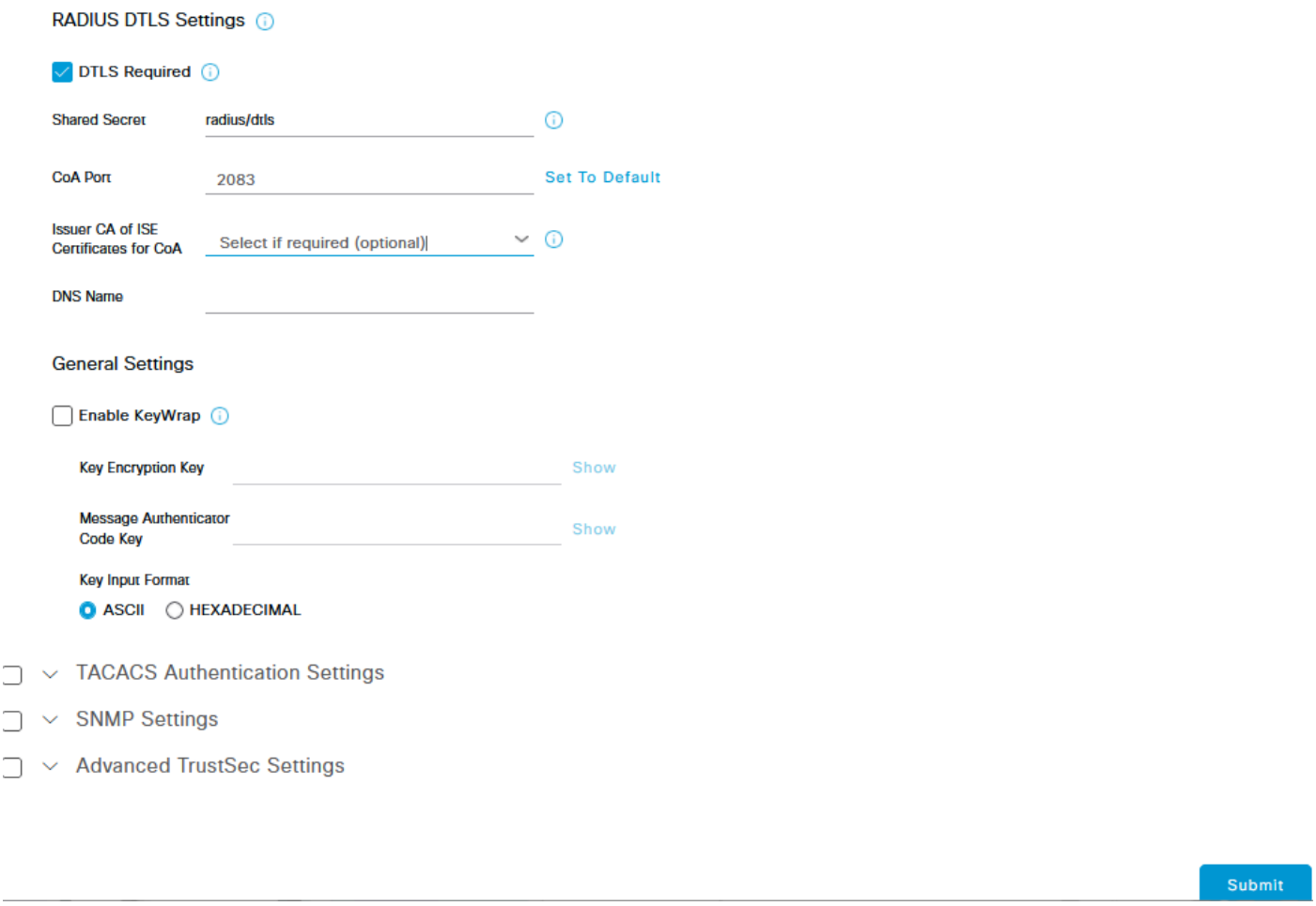
Configuração do ISE

Adicione a WLC como um dispositivo de rede ao ISE, para fazer isso, navegue até Administração>Recursos de rede>Dispositivos de rede>Adicionar
Insira o nome do dispositivo e o IP da interface da WLC que origina o tráfego RADIUS. Normalmente o IP da interface de gerenciamento sem fio. Role para baixo e verifique RADIUS

Authentication Settings, bem como DTLS Required e clique em Submit:



Nova configuração de dispositivo de rede



Configurações DTLS Radius para o dispositivo de rede no ISE

Configuração de WLC

Defina um novo servidor Radius junto com o endereço IP do ISE e a porta padrão para Radius DTLS. Esta configuração está disponível somente na CLI:

```
9800#configure terminal
9800(config)#radius server ISE
9800(config-radius-server)#address ipv4
```

```
9800(config-radius-server)#dtls port 2083
```

O DTLS Radius deve usar o radius/dtls secreto compartilhado, a WLC 9800 ignora qualquer chave configurada diferente desta:

```
9800(config-radius-server)#key radius/dtls
```

Use o comando`dtls trustpoint client`
para configurar o ponto confiável que contém o certificado do dispositivo WLC a ser trocado pelo túnel DTLS.
Use o comando`dtls trustpoint server`
para configurar o ponto confiável que contém a CA do emissor para o certificado do dispositivo ISE.

Os nomes dos pontos de confiança do cliente e do servidor serão os mesmos somente se os certificados WLC e ISE forem emitidos pela mesma CA:

```
9800(config-radius-server)#dtls trustpoint client WLC.pfx
9800(config-radius-server)#dtls trustpoint server WLC.pfx
```

Configure a WLC para verificar se há um dos SANs (Subject Alternative Names, nomes alternativos do assunto) presentes no certificado do ISE. Essa configuração deve corresponder exatamente a uma das SANs presentes no campo SANs do certificado.

A WLC 9800 não executa uma correspondência regular baseada em expressões para o campo SAN. Isso significa, por exemplo, que o comando `dtls match-server-identity hostname *.example.com` para um certificado curinga que tem *.example.com em seu campo SAN está correto, mas o mesmo comando para um certificado que contém www.example.com no campo SAN não está.

A WLC não verifica este nome em relação a nenhum servidor de nomes:

```
9800(config-radius-server)#dtls match-server-identity hostname ISE.example.com
9800(config-radius-server)#exit
```

Crie um novo grupo de servidores para usar o novo DTLS Radius para autenticação:

```
9800(config)#aaa group server radius Radsec
9800(config-sg-radius)#server name ISE
9800(config-sg-radius)#exit
```

A partir desse ponto, você pode usar esse grupo de servidores como qualquer outro grupo de

servidores na WLC. Consulte Configurar a Autenticação 802.1X no Catalyst 9800 Wireless Controller Series para usar este servidor para autenticação de cliente sem fio.

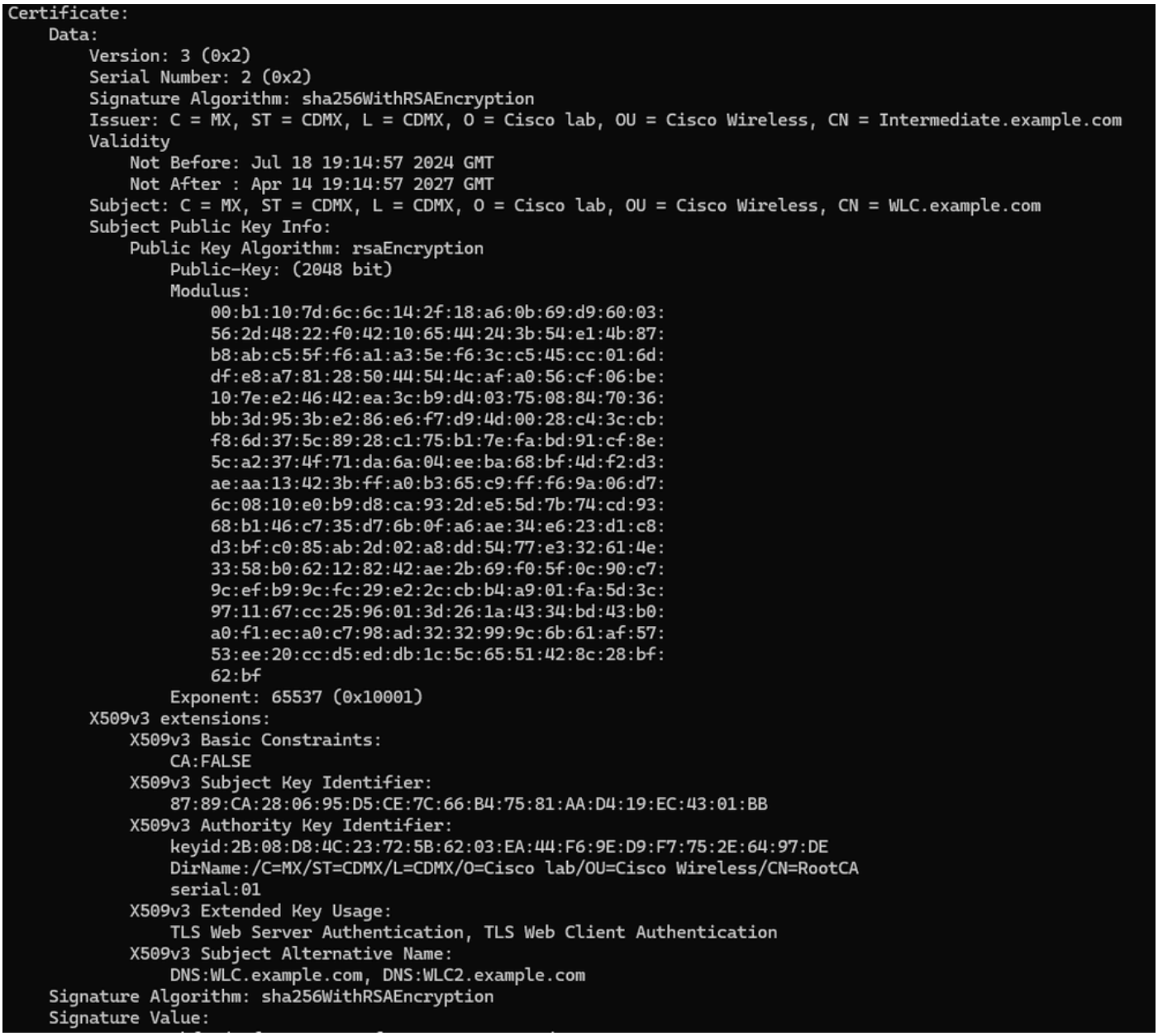
Verificar

Verificar informações do certificado

Para verificar as informações de certificado para os certificados criados, no terminal Linux, execute o comando:

```
openssl x509 -in

      -text -noout
```

Ela mostra as informações completas do certificado. Isso é útil para determinar a CA do emissor de um determinado certificado ou se os certificados contêm as EKUs e SANs necessárias:



Informações de certificado do dispositivo Cisco IOS XE conforme mostrado pelo OpenSSL

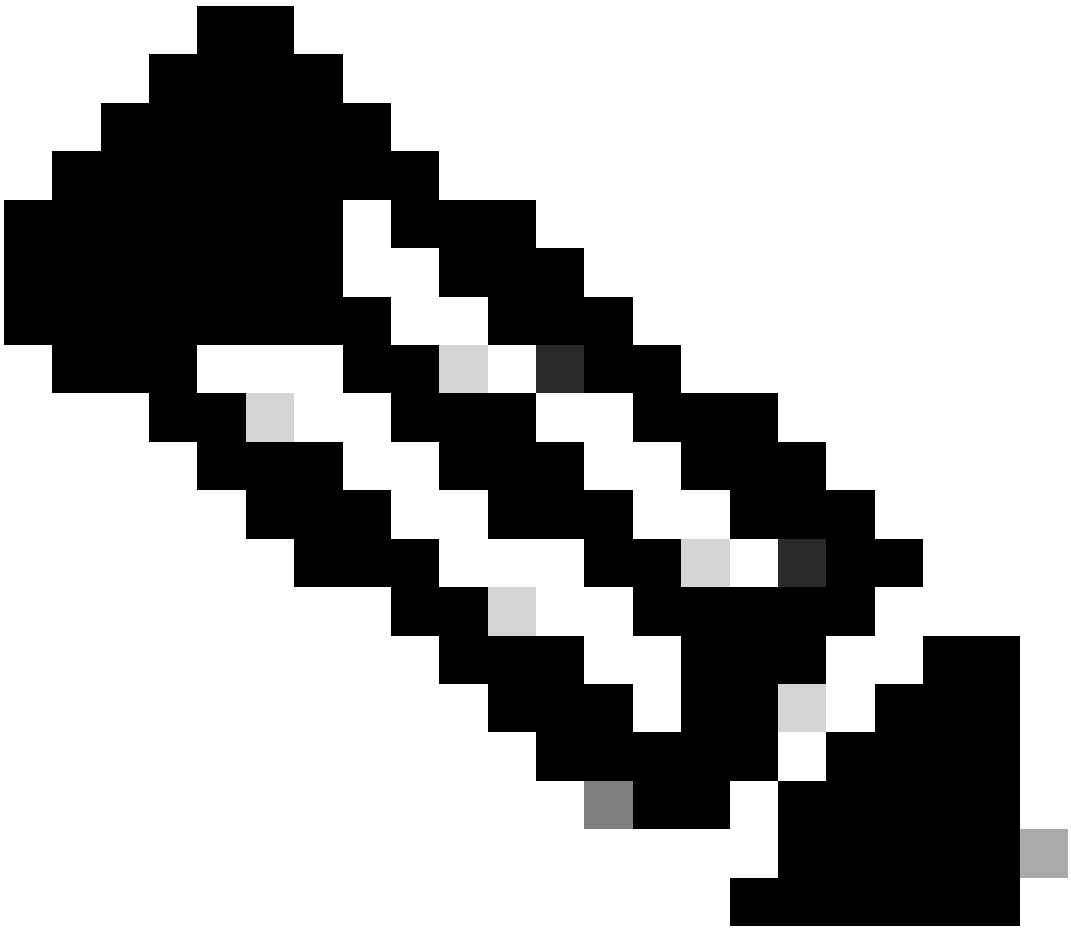
Executar Autenticação de Teste

Na WLC, você pode testar a funcionalidade do Radius DTLS com o comando test aaa group

new-code

```
9800#test aaa group Radsec testuser Cisco123 new-code
User successfully authenticated
```

```
USER ATTRIBUTES

username              0   "testuser"
```



> Observação: uma saída de rejeição de acesso no comando de teste significa que a WLC recebeu uma mensagem RADIUS Access-Reject, caso em que o RADIUS DTLS está funcionando. No entanto, também pode indicar uma falha ao estabelecer o túnel DTLS. O comando de teste não diferencia os dois cenários; consulte a seção de solução de problemas para identificar se há algum problema.

Troubleshooting

Para revisar a causa de uma autenticação com falha, você pode habilitar esses comandos antes de executar uma autenticação de teste.

```
9800#debug radius
9800#debug radius radsec
9800#terminal monitor
```

Esta é a saída de uma autenticação bem-sucedida com depurações habilitadas:

```
9800#test aaa group Radsec testuser Cisco123 new-code
User successfully authenticated

USER ATTRIBUTES

username             0   "testuser"
9800#
Jul 18 21:24:38.301: %PARSER-5-HIDDEN: Warning!!! ' test platform-aaa group server-group Radsec user-na
Jul 18 21:24:38.313: RADIUS/ENCODE(00000000):Orig. component type = Invalid
Jul 18 21:24:38.313: RADIUS/ENCODE(00000000): dropping service type, "radius-server attribute 6 on-for-
Jul 18 21:24:38.313: RADIUS(00000000): Config NAS IP: 0.0.0.0
Jul 18 21:24:38.313: vrfid: [65535]  ipv6 tableid : [0]
Jul 18 21:24:38.313: idb is NULL
Jul 18 21:24:38.313: RADIUS(00000000): Config NAS IPv6: ::
Jul 18 21:24:38.313: RADIUS(00000000): sending
Jul 18 21:24:38.313: RADIUS/DECODE(00000000): There is no General DB. Want server details may not be sp
Jul 18 21:24:38.313: RADSEC: DTLS default secret
Jul 18 21:24:38.313: RADIUS/ENCODE: Best Local IP-Address 172.16.5.11 for Radius-Server 172.16.18.123
Jul 18 21:24:38.313: RADSEC: DTLS default secret
Jul 18 21:24:38.313: RADIUS(00000000): Send Access-Request to 172.16.18.123:2083 id 53808/10, len 54
RADIUS:  authenticator C3 4E 34 0A 91 EF 42 53 - 7E C8 BB 50 F3 98 B3 14
Jul 18 21:24:38.313: RADIUS:  User-Password       [2]   18  *
Jul 18 21:24:38.313: RADIUS:  User-Name           [1]   10  "testuser"
Jul 18 21:24:38.313: RADIUS:  NAS-IP-Address      [4]   6   172.16.5.11
Jul 18 21:24:38.313: RADIUS_RADSEC_ENQ_WAIT_Q: Success Server(172.16.18.123)/Id(10)
Jul 18 21:24:38.313: RADIUS_RADSEC_CLIENT_PROCESS: Got DATA SEND MSG
Jul 18 21:24:38.313: RADIUS_RADSEC_SOCK_SET:  0 Success
Jul 18 21:24:38.313: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 18 21:24:38.313: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 18 21:24:38.313: RADIUS_RADSEC_HASH_KEY_ADD_CTX: add [radius_radsec ctx(0x7522CE91BAC0)] succeedd f
Jul 18 21:24:38.313: RADIUS_RADSEC_GET_SOURCE_ADDR: Success
Jul 18 21:24:38.313: RADIUS_RADSEC_GET_SOCK_ADDR: Success
Jul 18 21:24:38.313: RADIUS_RADSEC_SET_LOCAL_SOCK: Success
Jul 18 21:24:38.313: RADIUS_RADSEC_SOCK_SET: Success
Jul 18 21:24:38.314: RADIUS_RADSEC_BIND_SOCKET: Success
Jul 18 21:24:38.314: RADIUS_RADSEC_CONN_SET_LPORT: Success
Jul 18 21:24:38.314: RADIUS_RADSEC_CONN_SET_SERVER_PORT: Success
Jul 18 21:24:38.314: RADIUS_RADSEC_CLIENT_HS_START: local port = 54509
Jul 18 21:24:38.314: RADIUS_RADSEC_SOCKET_CONNECT: Success
Jul 18 21:24:38.315: RADIUS_RADSEC_UPDATE_SVR_REF_CNT: Got radsec_data
Jul 18 21:24:38.315: RADIUS_RADSEC_UPDATE_SVR_REF_CNT: Got valid rctx, with server_handle B0000019
Jul 18 21:24:38.316: RADIUS_RADSEC_CLIENT_HS_START: TLS handshake in progress...(172.16.18.123/2083)
Jul 18 21:24:38.316: RADIUS_RADSEC_START_CONN_TIMER: Started (172.16.18.123/2083) for 5 secs
Jul 18 21:24:38.316: RADIUS_RADSEC_CONN_STATE_UPDATE: Success - State = 2
Jul 18 21:24:38.318: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 18 21:24:38.318: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 18 21:24:38.318: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 18 21:24:38.318: RADIUS_RADSEC_HASH_KEY_MATCH: hashkey1(0) matches hashkey2(0) TRUE
Jul 18 21:24:38.318: RADIUS_RADSEC_HASH_KEY_GET_CTX: radius radsec  sock_ctx(0x7522CE91BAC0:0) get for
Jul 18 21:24:38.318: RADIUS_RADSEC_PROCESS_SOCK_EVENT: Handle socket event for TLS handshake(172.16.18.
Jul 18 21:24:38.318: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 18 21:24:38.318: RADIUS_RADSEC_START_CONN_TIMER: Started (172.16.18.123/2083) for 5 secs
```

```
Jul 18 21:24:38.318: RADIUS_RADSEC_HS_CONTINUE: TLS handshake in progress...(172.16.18.123/2083)
Jul 18 21:24:38.318: RADIUS_RADSEC_SOCK_TLS_EVENT_HANDLE: Success
Jul 18 21:24:38.318: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 18 21:24:38.327: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 18 21:24:38.327: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 18 21:24:38.327: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 18 21:24:38.327: RADIUS_RADSEC_HASH_KEY_MATCH: hashkey1(0) matches hashkey2(0) TRUE
Jul 18 21:24:38.327: RADIUS_RADSEC_HASH_KEY_GET_CTX: radius radsec  sock_ctx(0x7522CE91BAC0:0) get for
Jul 18 21:24:38.327: RADIUS_RADSEC_PROCESS_SOCK_EVENT: Handle socket event for TLS handshake(172.16.18.
Jul 18 21:24:38.327: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 18 21:24:38.391: RADIUS_RADSEC_START_CONN_TIMER: Started (172.16.18.123/2083) for 5 secs
Jul 18 21:24:38.391: RADIUS_RADSEC_HS_CONTINUE: TLS handshake in progress...(172.16.18.123/2083)
Jul 18 21:24:38.391: RADIUS_RADSEC_SOCK_TLS_EVENT_HANDLE: Success
Jul 18 21:24:38.391: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 18 21:24:38.397: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 18 21:24:38.397: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 18 21:24:38.397: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 18 21:24:38.397: RADIUS_RADSEC_HASH_KEY_MATCH: hashkey1(0) matches hashkey2(0) TRUE
Jul 18 21:24:38.397: RADIUS_RADSEC_HASH_KEY_GET_CTX: radius radsec  sock_ctx(0x7522CE91BAC0:0) get for
Jul 18 21:24:38.397: RADIUS_RADSEC_PROCESS_SOCK_EVENT: Handle socket event for TLS handshake(172.16.18.
Jul 18 21:24:38.397: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 18 21:24:38.397: RADIUS_RADSEC_HS_CONTINUE: TLS handshake success!(172.16.18.123/2083) <-------- TL
Jul 18 21:24:38.397: RADIUS_RADSEC_CONN_STATE_UPDATE: Success - State = 3
Jul 18 21:24:38.397: RADIUS_RADSEC_UPDATE_SVR_REF_CNT: Got radsec_data
Jul 18 21:24:38.397: RADIUS_RADSEC_UPDATE_SVR_REF_CNT: Got valid rctx, with server_handle B0000019
Jul 18 21:24:38.397: RADIUS-RADSEC-HS-SUCCESS: Negotiated Cipher is ECDHE-RSA-AES256-GCM-SHA384
Jul 18 21:24:38.397: RADIUS_RADSEC_START_DATA_SEND: RADSEC HS Done, Start data send (172.16.18.123/2083
Jul 18 21:24:38.397: RADIUS_RADSEC_UNQUEUE_WAIT_Q: Success Server(172.16.18.123)/Id(10)
Jul 18 21:24:38.397: RADIUS_RADSEC_MSG_SEND: RADSEC Write SUCCESS(id=10)
Jul 18 21:24:38.397: RADIUS(00000000): Started 5 sec timeout
Jul 18 21:24:38.397: RADIUS_RADSEC_UNQUEUE_WAIT_Q: Empty Server(172.16.18.123)/Id(-1)
Jul 18 21:24:38.397: RADIUS_RADSEC_START_DATA_SEND: no more data available
Jul 18 21:24:38.397: RADIUS_RADSEC_IDLE_TIMER: Started (172.16.18.123/2083)
Jul 18 21:24:38.397: RADIUS-RADSEC-HS-SUCCESS: Success
Jul 18 21:24:38.397: RADIUS_RADSEC_SOCK_TLS_EVENT_HANDLE: Success
Jul 18 21:24:38.397: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 18 21:24:38.453: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 18 21:24:38.453: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 18 21:24:38.453: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 18 21:24:38.453: RADIUS_RADSEC_HASH_KEY_MATCH: hashkey1(0) matches hashkey2(0) TRUE
Jul 18 21:24:38.453: RADIUS_RADSEC_HASH_KEY_GET_CTX: radius radsec  sock_ctx(0x7522CE91BAC0:0) get for
Jul 18 21:24:38.453: RADIUS_RADSEC_MSG_RECV: RADSEC Bytes read= 20, Err= 0
Jul 18 21:24:38.453: RADIUS_RADSEC_SOCK_READ_EVENT_HANDLE: Radius length is 113
Jul 18 21:24:38.453: RADIUS_RADSEC_SOCK_READ_EVENT_HANDLE: Going to read rest 93 bytes
Jul 18 21:24:38.453: RADIUS_RADSEC_MSG_RECV: RADSEC Bytes read= 93, Err= 0
Jul 18 21:24:38.453: RADIUS_RADSEC_SOCK_READ_EVENT_HANDLE: linktype = 7 - src port = 2083 - dest port =
Jul 18 21:24:38.453: RADIUS: Received from id 54509/10 172.16.18.123:2083, Access-Accept, len 113 <----
RADIUS:  authenticator 4E CE 96 63 41 4B 43 04 - C7 A2 B5 05 C2 78 A7 0D
Jul 18 21:24:38.453: RADIUS:  User-Name           [1]   10  "testuser"
Jul 18 21:24:38.453: RADIUS:  Class               [25]  83
RADIUS:   43 41 43 53 3A 61 63 31 30 31 32 37 62 64 38 74  [CACS:ac10127bd8t]
RADIUS:   47 58 50 47 4E 63 6C 57 76 2F 39 67 44 66 51 67  [GXPGNclWv/9gDfQg]
RADIUS:   63 4A 76 6C 35 47 72 33 71 71 47 36 4C 66 35 59  [cJvl5Gr3qqG6Lf5Y]
RADIUS:   52 42 2F 7A 57 55 39 59 3A 69 73 65 2D 76 62 65  [RB/zWU9Y:ise-vbe]
RADIUS:   74 61 6E 63 6F 2F 35 31 30 34 33 39 38 32 36 2F  [tanco/510439826/]
RADIUS:   39                  [ 9]
Jul 18 21:24:38.453: RADSEC: DTLS default secret
Jul 18 21:24:38.453: RADIUS/DECODE(00000000): There is no General DB. Reply server details may not be r
Jul 18 21:24:38.453: RADIUS(00000000): Received from id 54509/10
```

CA Desconhecida Relatada pelo WLC

Quando a WLC não pode validar os certificados fornecidos pelo ISE, ela falha ao criar o túnel DTLS e as autenticações falham.
Este é um exemplo das mensagens de depuração apresentadas quando este é o caso:

```
9800#test aaa group Radsec testuser Cisco123 new-code

Jul 19 00:59:09.695: %PARSER-5-HIDDEN: Warning!!! ' test platform-aaa group server-group Radsec user-na
Jul 19 00:59:09.706: RADIUS/ENCODE(00000000):Orig. component type = Invalid
Jul 19 00:59:09.707: RADIUS/ENCODE(00000000): dropping service type, "radius-server attribute 6 on-for-
Jul 19 00:59:09.707: RADIUS(00000000): Config NAS IP: 0.0.0.0
Jul 19 00:59:09.707: vrfid: [65535]  ipv6 tableid : [0]
Jul 19 00:59:09.707: idb is NULL
Jul 19 00:59:09.707: RADIUS(00000000): Config NAS IPv6: ::
Jul 19 00:59:09.707: RADIUS(00000000): sending
Jul 19 00:59:09.707: RADIUS/DECODE(00000000): There is no General DB. Want server details may not be sp
Jul 19 00:59:09.707: RADSEC: DTLS default secret
Jul 19 00:59:09.707: RADIUS/ENCODE: Best Local IP-Address 172.16.5.11 for Radius-Server 172.16.18.123
Jul 19 00:59:09.707: RADSEC: DTLS default secret
Jul 19 00:59:09.707: RADIUS(00000000): Send Access-Request to 172.16.18.123:2083 id 52764/13, len 54
RADIUS:  authenticator E8 09 1D B0 72 50 17 E6 - B4 27 F6 E3 18 25 16 64
Jul 19 00:59:09.707: RADIUS:  User-Password       [2]   18  *
Jul 19 00:59:09.707: RADIUS:  User-Name           [1]   10  "testuser"
Jul 19 00:59:09.707: RADIUS:  NAS-IP-Address      [4]   6   172.16.5.11
Jul 19 00:59:09.707: RADIUS_RADSEC_ENQ_WAIT_Q: Success Server(172.16.18.123)/Id(13)
Jul 19 00:59:09.707: RADIUS_RADSEC_CLIENT_PROCESS: Got DATA SEND MSG
Jul 19 00:59:09.707: RADIUS_RADSEC_SOCK_SET:  0 Success
Jul 19 00:59:09.707: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 19 00:59:09.707: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 19 00:59:09.707: RADIUS_RADSEC_HASH_KEY_ADD_CTX: add [radius_radsec ctx(0x7522CE91BAC0)] succeedd f
Jul 19 00:59:09.707: RADIUS_RADSEC_GET_SOURCE_ADDR: Success
Jul 19 00:59:09.707: RADIUS_RADSEC_GET_SOCK_ADDR: Success
Jul 19 00:59:09.707: RADIUS_RADSEC_SET_LOCAL_SOCK: Success
Jul 19 00:59:09.707: RADIUS_RADSEC_SOCK_SET: Success
Jul 19 00:59:09.707: RADIUS_RADSEC_BIND_SOCKET: Success
Jul 19 00:59:09.707: RADIUS_RADSEC_CONN_SET_LPORT: Success
Jul 19 00:59:09.707: RADIUS_RADSEC_CONN_SET_SERVER_PORT: Success
Jul 19 00:59:09.707: RADIUS_RADSEC_CLIENT_HS_START: local port = 49556
Jul 19 00:59:09.707: RADIUS_RADSEC_SOCKET_CONNECT: Success
Jul 19 00:59:09.709: RADIUS_RADSEC_UPDATE_SVR_REF_CNT: Got radsec_data
Jul 19 00:59:09.709: RADIUS_RADSEC_UPDATE_SVR_REF_CNT: Got valid rctx, with server_handle B0000019
Jul 19 00:59:09.709: RADIUS_RADSEC_CLIENT_HS_START: TLS handshake in progress...(172.16.18.123/2083)
Jul 19 00:59:09.709: RADIUS_RADSEC_START_CONN_TIMER: Started (172.16.18.123/2083) for 5 secsUser reject

uwu-9800#
Jul 19 00:59:09.709: RADIUS_RADSEC_CONN_STATE_UPDATE: Success - State = 2
Jul 19 00:59:09.711: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 19 00:59:09.711: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 19 00:59:09.711: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 19 00:59:09.711: RADIUS_RADSEC_HASH_KEY_MATCH: hashkey1(0) matches hashkey2(0) TRUE
Jul 19 00:59:09.711: RADIUS_RADSEC_HASH_KEY_GET_CTX: radius radsec  sock_ctx(0x7522CE91BAC0:0) get for
Jul 19 00:59:09.711: RADIUS_RADSEC_PROCESS_SOCK_EVENT: Handle socket event for TLS handshake(172.16.18.
Jul 19 00:59:09.711: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 19 00:59:09.711: RADIUS_RADSEC_START_CONN_TIMER: Started (172.16.18.123/2083) for 5 secs
Jul 19 00:59:09.711: RADIUS_RADSEC_HS_CONTINUE: TLS handshake in progress...(172.16.18.123/2083)
Jul 19 00:59:09.711: RADIUS_RADSEC_SOCK_TLS_EVENT_HANDLE: Success
Jul 19 00:59:09.713: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
```

```
Jul 19 00:59:09.720: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 19 00:59:09.720: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 19 00:59:09.720: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 19 00:59:09.720: RADIUS_RADSEC_HASH_KEY_MATCH: hashkey1(0) matches hashkey2(0) TRUE
Jul 19 00:59:09.720: RADIUS_RADSEC_HASH_KEY_GET_CTX: radius radsec  sock_ctx(0x7522CE91BAC0:0) get for
Jul 19 00:59:09.720: RADIUS_RADSEC_PROCESS_SOCK_EVENT: Handle socket event for TLS handshake(172.16.18.
Jul 19 00:59:09.720: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 19 00:59:09.722: RADIUS_RADSEC_HS_CONTINUE: TLS handshake failed!
Jul 19 00:59:09.722: RADIUS_RADSEC_UNQUEUE_WAIT_Q: Success Server(172.16.18.123)/Id(13)
Jul 19 00:59:09.722: RADIUS_RADSEC_FAILOVER_HANDLER:Failng-over to new server = 0x0
Jul 19 00:59:09.722: RADIUS_RADSEC_UNQUEUE_WAIT_Q: Empty Server(172.16.18.123)/Id(-1)
Jul 19 00:59:09.722: RADIUS_RADSEC_FAILOVER_HANDLER: no more data available
Jul 19 00:59:09.722: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 19 00:59:09.722: RADIUS_RADSEC_CONN_CLOSE: Cleaned up timers for Radius RADSEC ctx
Jul 19 00:59:09.722: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 19 00:59:09.722: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 19 00:59:09.722: RADIUS_RADSEC_HASH_KEY_DEL_CTX: remove [radius_radsec  ctx(0x7522CE91BAC0)] succee
Jul 19 00:59:09.722: RADIUS_RADSEC_CONN_CLOSE: Hash table entry removed for RADSEC sock ctx
Jul 19 00:59:09.723: RADIUS_RADSEC_CONN_CLOSE: Success
Jul 19 00:59:09.723: RADIUS_RADSEC_SOCK_TLS_EVENT_HANDLE: Failed to complete TLS handshake <----------D
Jul 19 00:59:09.723: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 19 00:59:09.723: RADIUS_RADSEC_CONN_CLOSE: Cleaned up timers for Radius RADSEC ctx
Jul 19 00:59:09.723: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(-1) generated for sock(-1)
Jul 19 00:59:09.723: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(-1) generated for sock(-1)
uwu-9800#
Jul 19 00:59:09.723: RADIUS_RADSEC_HASH_KEY_DEL_CTX: remove [radius_radsec  ctx(0x7522CE91BAC0)] succee
Jul 19 00:59:09.723: RADIUS_RADSEC_CONN_CLOSE: Hash table entry removed for RADSEC sock ctx
Jul 19 00:59:09.723: RADIUS_RADSEC_CONN_CLOSE: Success
Jul 19 00:59:09.723: RADIUS_RADSEC_SOCK_TLS_EVENT_HANDLE: Error
Jul 19 00:59:09.723: RADIUS_RADSEC_PROCESS_SOCK_EVENT: failed to hanlde radsec hs event
Jul 19 00:59:09.723: RADIUS/DECODE: No response from radius-server; parse response; FAIL
Jul 19 00:59:09.723: RADIUS/DECODE: Case error(no response/ bad packet/ op decode);parse response; FAIL
Jul 19 00:59:09.723: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
Jul 19 00:59:10.718: %RADSEC_AUDIT_MESSAGE-3-FIPS_AUDIT_FCS_RADSEC_SERVER_CERTIFICATE_VALIDATION_FAILUR
Jul 19 00:59:10.718: %RADSEC_AUDIT_MESSAGE-3-FIPS_AUDIT_FCS_RADSEC_SERVER_IDENTITY_CHECK_FAILURE: Chass
Jul 19 00:59:10.718: %RADSEC_AUDIT_MESSAGE-6-FIPS_AUDIT_FCS_DTLSC_DTLS_SESSION_CLOSED: Chassis 1 R0/0:
```

Para corrigi-lo, certifique-se de que a identidade configurada no WLC corresponda exatamente a uma das SANs incluídas no certificado ISE:

```
9800(config)#radius server
```

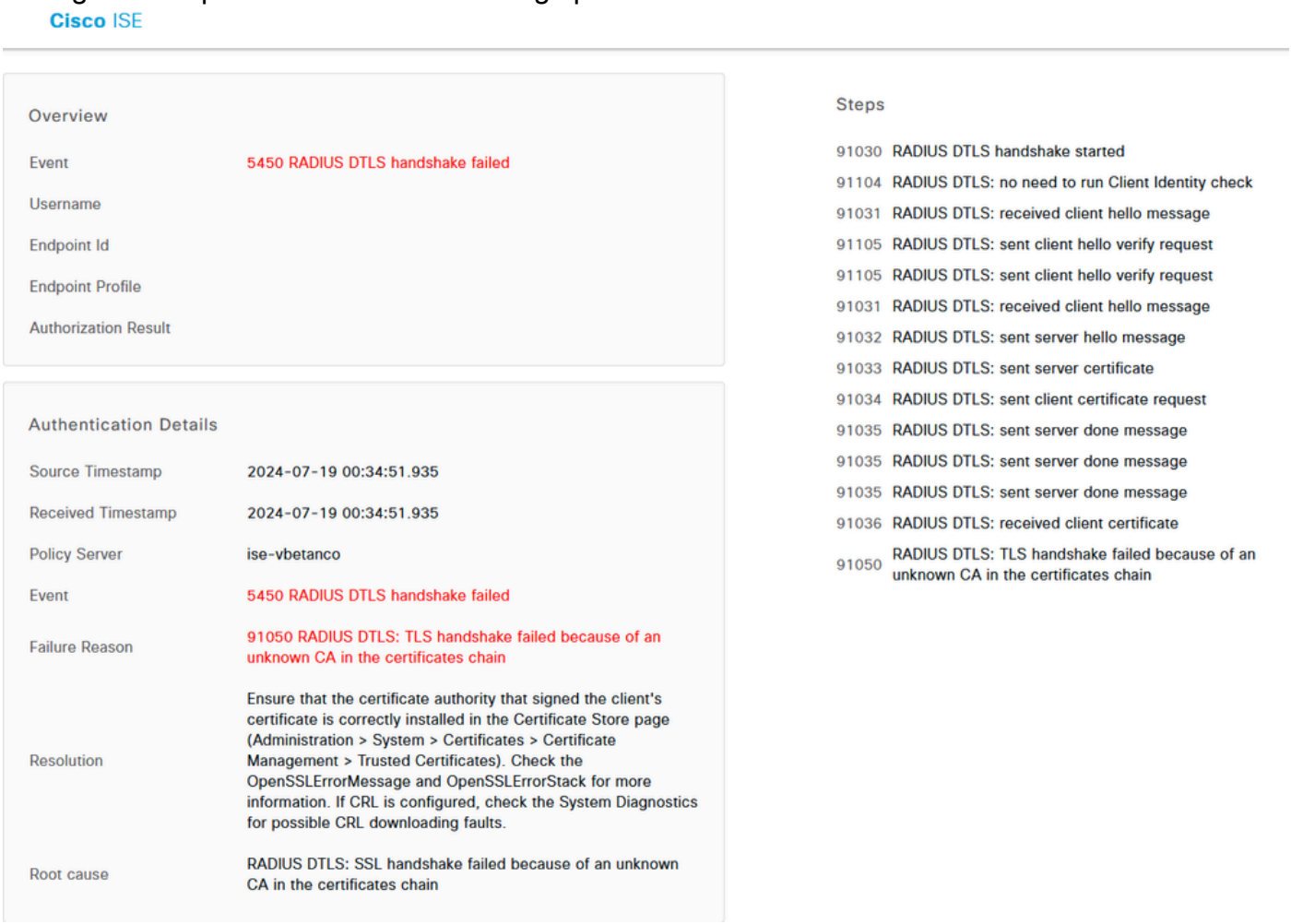
```
      9800(config)#dtls match-server-identity hostname
```

Verifique se a cadeia de certificados da autoridade de certificação foi importada corretamente no controlador e se o **dtls trustpoint server**

configuration uses the Issuer CA trustpoint.

CA desconhecido relatado pelo ISE

Quando o ISE não pode validar certificados fornecidos pela WLC, ele não cria o túnel DTLS e as autenticações falham. Isso aparece como um erro nos registros em tempo real do RADIUS. Navegue até Operations>Radius>Live logs para verificar.



O log do ISE reporta falha de handshake DTLS devido a CA desconhecida

Para corrigi-lo, assegure-se de que os certificados intermediário e raiz, marque as caixas de seleção Confiar para autenticação de cliente e Syslog em Administração>Sistema>Certificados>Certificados confiáveis.

Verificação de Revogação em Vigor

Quando os certificados são importados para a WLC, os pontos confiáveis recém-criados têm a verificação de revogação habilitada. Isso faz com que a WLC tente procurar uma lista de revogação de certificados que não esteja disponível ou acessível e falha na verificação do certificado.
Certifique-se de que cada ponto confiável no caminho de verificação para os certificados contenha o comando revocation-check none .

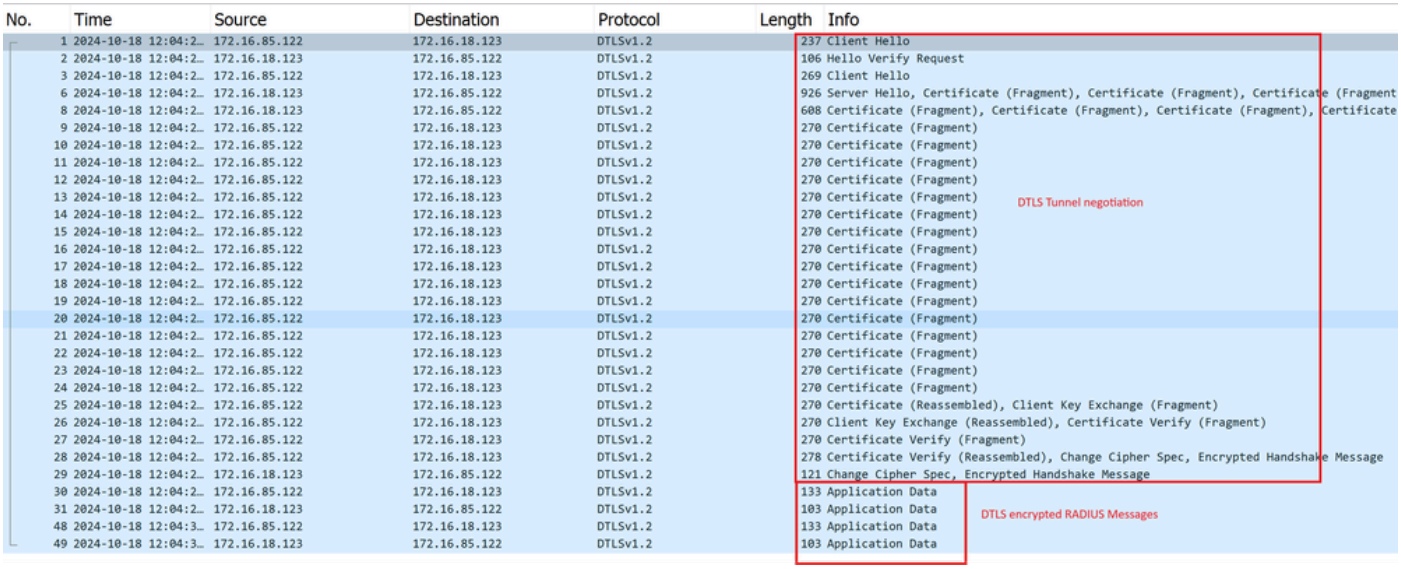
```
Jul 17 21:50:39.064: RADIUS_RADSEC_HASH_KEY_MATCH: hashkey1(0) matches hashkey2(0) TRUE
Jul 17 21:50:39.064: RADIUS_RADSEC_HASH_KEY_GET_CTX: radius radsec  sock_ctx(0x780FB0715978:0) get for
Jul 17 21:50:39.064: RADIUS_RADSEC_PROCESS_SOCK_EVENT: Handle socket event for TLS handshake(172.16.18.
Jul 17 21:50:39.064: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 17 21:50:39.068: %PKI-3-CRL_FETCH_FAIL: CRL fetch for trustpoint WLC1.pfx failed
                      Reason : Enrollment URL not configured. <------ WLC tries to perform revocation ch
Jul 17 21:50:39.070: RADIUS_RADSEC_HS_CONTINUE: TLS handshake failed!
Jul 17 21:50:39.070: RADIUS_RADSEC_UNQUEUE_WAIT_Q: Success Server(172.16.18.123)/Id(2)
Jul 17 21:50:39.070: RADIUS_RADSEC_FAILOVER_HANDLER:Failng-over to new server = 0x0
Jul 17 21:50:39.070: RADIUS_RADSEC_UNQUEUE_WAIT_Q: Empty Server(172.16.18.123)/Id(-1)
Jul 17 21:50:39.070: RADIUS_RADSEC_FAILOVER_HANDLER: no more data available
Jul 17 21:50:39.070: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 17 21:50:39.070: RADIUS_RADSEC_CONN_CLOSE: Cleaned up timers for Radius RADSEC ctx
Jul 17 21:50:39.070: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(0) generated for sock(0)
Jul 17 21:50:39.070: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(0) generated for sock(0)
Jul 17 21:50:39.070: RADIUS_RADSEC_HASH_KEY_DEL_CTX: remove [radius_radsec  ctx(0x780FB0715978)] succee
Jul 17 21:50:39.070: RADIUS_RADSEC_CONN_CLOSE: Hash table entry removed for RADSEC sock ctx
Jul 17 21:50:39.070: RADIUS_RADSEC_CONN_CLOSE: Success
Jul 17 21:50:39.070: RADIUS_RADSEC_SOCK_TLS_EVENT_HANDLE: Failed to complete TLS handshake
Jul 17 21:50:39.070: RADIUS_RADSEC_STOP_TIMER: Stopped (172.16.18.123/2083)
Jul 17 21:50:39.070: RADIUS_RADSEC_CONN_CLOSE: Cleaned up timers for Radius RADSEC ctx
Jul 17 21:50:39.070: RADIUS_RADSEC_GENERATE_HASHKEY: hash key(-1) generated for sock(-1)
Jul 17 21:50:39.070: RADIUS_RADSEC_GENERATE_HASHBUCKET: hash bucket(-1) generated for sock(-1)
Jul 17 21:50:39.070: RADIUS_RADSEC_HASH_KEY_DEL_CTX: remove [radius_radsec  ctx(0x780FB0715978)] succee
Jul 17 21:50:39.070: RADIUS_RADSEC_CONN_CLOSE: Hash table entry removed for RADSEC sock ctx
Jul 17 21:50:39.070: RADIUS_RADSEC_CONN_CLOSE: Success
Jul 17 21:50:39.070: RADIUS_RADSEC_SOCK_TLS_EVENT_HANDLE: Error
Jul 17 21:50:39.070: RADIUS_RADSEC_PROCESS_SOCK_EVENT: failed to hanlde radsec hs event
Jul 17 21:50:39.070: RADIUS_RADSEC_CLIENT_PROCESS: Got Socket Event
```

Solucionar problemas de estabelecimento de túnel DTLS na captura de pacotes

A WLC 9800 oferece o recurso Embedded Packet Capture (EPC), que permite capturar todo o tráfego enviado e recebido de uma determinada interface. O ISE oferece um recurso semelhante chamado despejo TCP para monitorar o tráfego de entrada e saída. Quando usados ao mesmo tempo, eles permitem analisar o tráfego de estabelecimento de sessão DTLS da perspectiva de ambos os dispositivos.

Consulte o Guia do Administrador do Cisco Identity Services Engine para obter as etapas detalhadas para configurar o despejo TCP no ISE. Consulte também Troubleshoot Catalyst 9800 Wireless LAN Controllers para obter informações sobre como configurar o recurso EPC no WLC.

Este é um exemplo de um estabelecimento de túnel DTLS bem-sucedido.



Captura de pacotes de uma negociação de túnel DTLS RADIUS e mensagens criptografadas

As capturas de pacotes mostram como o estabelecimento de túnel DTLS acontece. Se houver um problema com a negociação, causado por tráfego perdido entre dispositivos ou pacotes de alertas criptografados DTLS, a captura de pacotes o ajudará a identificar o problema.

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.