Konfigurieren von Radius DTLS auf der ISE und dem 9800 WLC

Inhalt

Einleitung

In diesem Dokument wird eine Methode zur Erstellung der erforderlichen Zertifikate für die Konfiguration von RADIUS DTLS zwischen der ISE und dem 9800 WLC beschrieben.

Hintergrund

RADIUS DTLS ist eine sichere Form des RADIUS-Protokolls, bei dem die RADIUS-Nachrichten über einen DTLS-Tunnel (Data Transport Layer Security) gesendet werden. Um diesen Tunnel zwischen dem Authentifizierungsserver und dem Authentifizierer zu erstellen, ist eine Reihe von Zertifikaten erforderlich. Für diese Zertifikatgruppe müssen bestimmte EKU-

Zertifikatserweiterungen (Extended Key Usage) festgelegt werden, insbesondere die Clientauthentifizierung für das WLC-Zertifikat und sowohl die Serverauthentifizierung als auch die Clientauthentifizierung für das ISE-Zertifikat.

Voraussetzungen

Anforderungen

Cisco empfiehlt, dass Sie über Kenntnisse in folgenden Bereichen verfügen:

- Konfigurieren des 9800 WLC, des Access Points (AP) für den Basisbetrieb
- Verwendung der OpenSSL-Anwendung
- Public Key Infrastructure (PKI) und digitale Zertifikate

Verwendete Komponenten

Die Informationen in diesem Dokument basierend auf folgenden Software- und Hardware-Versionen:

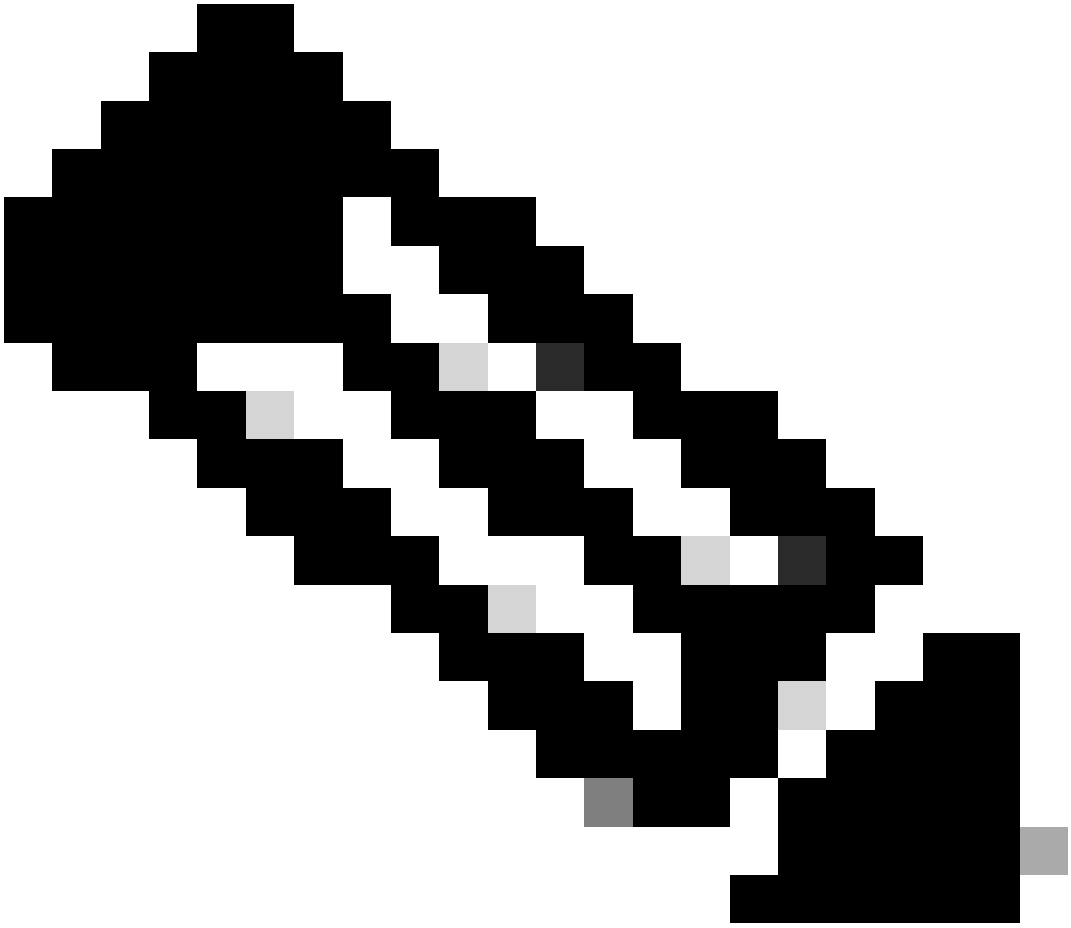
- OpenSSL-Anwendung (Version 3.0.2).
- ISE (Version 3.1.0.518)
- 9800 WLC (Version 17.12.3)

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die möglichen Auswirkungen aller Befehle kennen.

Konfigurieren

Überblick

Der Zweck besteht darin, eine Zertifizierungsstelle mit zwei Ebenen zu erstellen, die über eine Stammzertifizierungsstelle und eine intermediäre Zertifizierungsstelle verfügt, um Endpunktzertifikate zu signieren. Nach der Signatur werden die Zertifikate in den WLC und die ISE importiert. Schließlich werden die Geräte für die RADIUS-DTLS-Authentifizierung mit diesen Zertifikaten konfiguriert.



Hinweis: In diesem Dokument werden Linux-spezifische Befehle zum Erstellen und Anordnen von Dateien verwendet. Die Befehle werden erläutert, damit Sie die gleiche Aktion auf anderen Betriebssystemen ausführen können, auf denen OpenSSL verfügbar ist.

Optional - Erstellen eines WLC- und ISE RADIUS DTLS-Gerätezertifikats

Das RADIUS-DTLS-Protokoll muss Zertifikate zwischen ISE und WLC austauschen, um den DTLS-Tunnel zu erstellen. Wenn Sie noch keine gültigen Zertifikate besitzen, können Sie eine lokale Zertifizierungsstelle erstellen, um die Zertifikate zu generieren. Weitere Informationen finden Sie unter Konfigurieren einer mehrstufigen Zertifizierungsstelle auf OpenSSL zum Generieren von mit Cisco IOS® XE kompatiblen Zertifikaten und Durchführen der im Dokument beschriebenen Schritte vom Anfang bis zum Ende des Schritts Zwischenzertifikat erstellen.

Konfigurationsabschnitte zur Datei openssl.cnf hinzufügen

Öffnen Sie die Konfigurationsdatei openssl.cnf, und kopieren Sie unten die Abschnitte WLC und ISE, die zum Generieren einer gültigen Zertifikatsanforderung (Certificate Sign Request, CSR) verwendet werden, und fügen Sie sie ein.

Sowohl der Abschnitt ISE_device_req_ext als auch der Abschnitt WLC_device_req_ext verweisen jeweils auf eine Liste der SANs, die in den CSR aufgenommen werden sollen:

```
#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
```

Als Sicherheitsmaßnahme überschreibt die Zertifizierungsstelle alle SANs auf einem CSR, um diesen zu signieren, sodass nicht autorisierte Geräte kein gültiges Zertifikat für einen Namen erhalten, den sie nicht verwenden dürfen. Um die SANs wieder dem signierten Zertifikat hinzuzufügen, verwenden Sie den subjectAltName-Parameter, um auf die gleiche Liste von SANs zu verweisen, die auch für die CSR-Generierung verwendet werden.

Für die ISE sind serverAuth- und clientAuth-EKUs im Zertifikat erforderlich, während der WLC nur clientAuth benötigt. Sie werden dem signierten Zertifikat mit dem extendedKeyUsage-Parameter hinzugefügt.

Kopieren Sie die Abschnitte, die für das Zertifikatszeichen verwendet werden, und fügen Sie sie unten in der Datei openssl.cnf ein:

```
#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
```

WLC-Gerätezertifikat erstellen

Erstellen Sie einen neuen Ordner zum Speichern von WLC-Zertifikaten auf dem Computer, auf dem OpenSSL im Zertifizierungsstellenzertifizierungsordner "IntermCA.db.certs" installiert ist. Der neue Ordner heißt WLC:

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

Ändern Sie die DNS-Parameter im [WLC_alt_names] Abschnitt der Datei openssl.cnf. Ändern Sie die für die gewünschten Werte angegebenen Beispielnamen. Diese Werte werden im Feld SANs des WLC-Zertifikats eingetragen:

```
[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
```

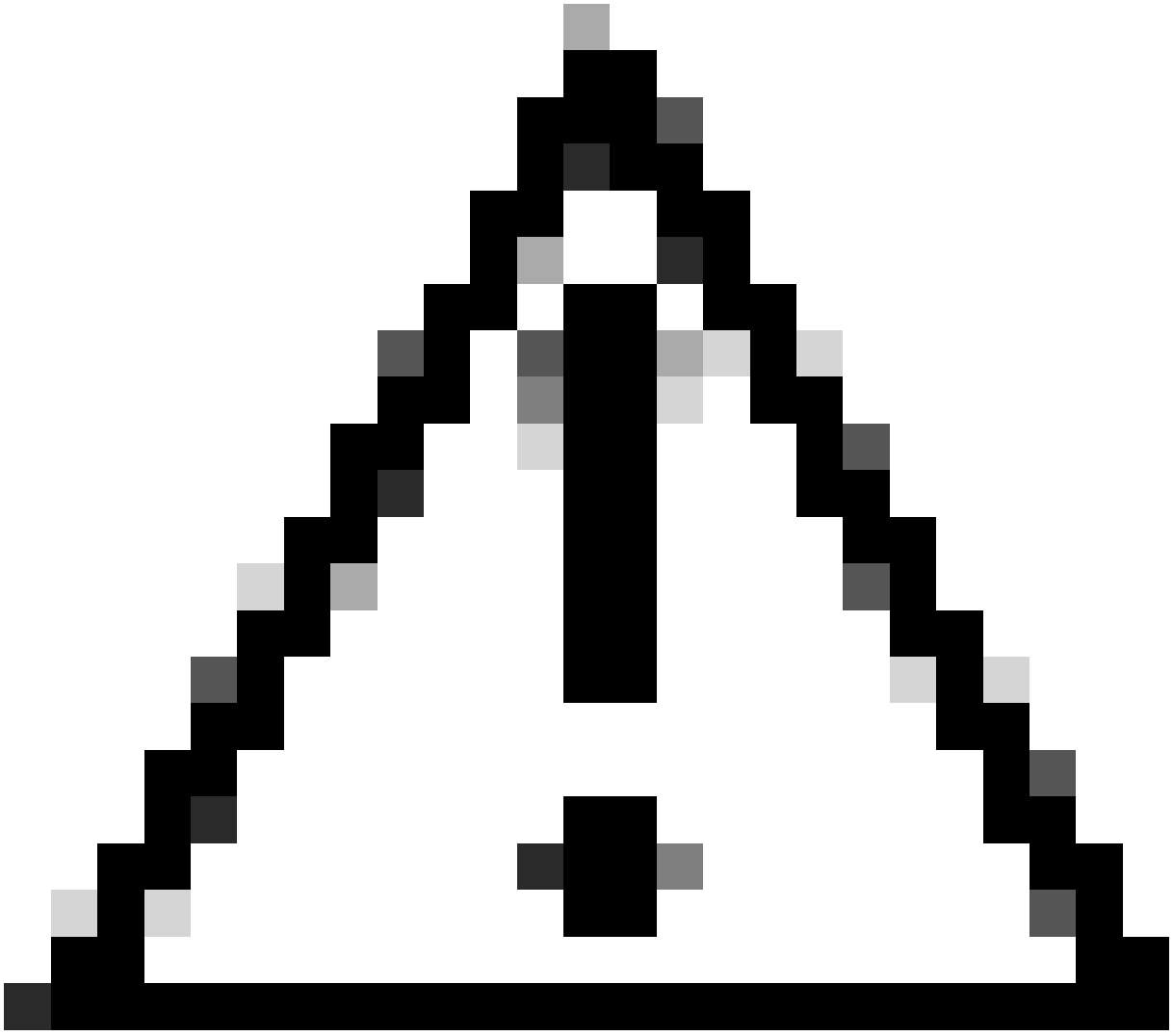
Erstellen Sie den privaten WLC-Schlüssel und den WLC-CSR mit Informationen aus dem Abschnitt WLC_device_req_ext für SANs:

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

OpenSSL öffnet eine interaktive Aufforderung zur Eingabe der DN-Details:

```
.........+..+.......+...........+...+......+.+...........+..++++++++++++
++++++
-----
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
```

WLC-Zertifikat Distinguished Name Interaktive Aufforderung



Achtung: Der Common Name (CN), den Sie an der interaktiven Eingabeaufforderung

angeben, muss mit einem der Namen im Abschnitt [WLC_alt_names] der Datei openssl.cnf identisch sein.

Verwenden Sie die Zertifizierungsstelle IntermCA, um den WLC-CSR WLC.csr mit den unter [WLC_cert] definierten Erweiterungen zu signieren und das signierte Zertifikat in ./IntermCA/IntermCA.db.certs/WLC zu speichern. Das WLC-Gerätezertifikat heißt WLC.crt:

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

Der 9800 WLC benötigt für den Import ein Zertifikat im PFX-Format. Erstellen Sie eine neue Datei, die die Kette von Zertifizierungsstellen enthält, die das WLC-Zertifikat signiert haben. Dies wird als "certfile" bezeichnet:

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

Führen Sie zum Erstellen der PFX-Datei einen dieser Befehle entsprechend der WLC-Version aus.

Bei älteren Versionen als 17.12.1:

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

Für Version 17.12.1 oder höher:

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

ISE-Gerätezertifikat erstellen

Erstellen Sie einen neuen Ordner zum Speichern von ISE-Zertifikaten auf dem Computer, auf dem OpenSSL im Zertifizierungsstellenzertifizierungsordner für die Zwischenzeit installiert ist, mit dem Namen IntermCA.db.certs. Der neue Ordner heißt ISE:

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

Ändern Sie die DNS-Parameter im [ISE_alt_names] Abschnitt der Datei openssl.cnf. Ändern Sie die Beispielnamen für die gewünschten Werte. Diese Werte werden im Feld SANs des WLC-Zertifikats eingetragen:

```
[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
```

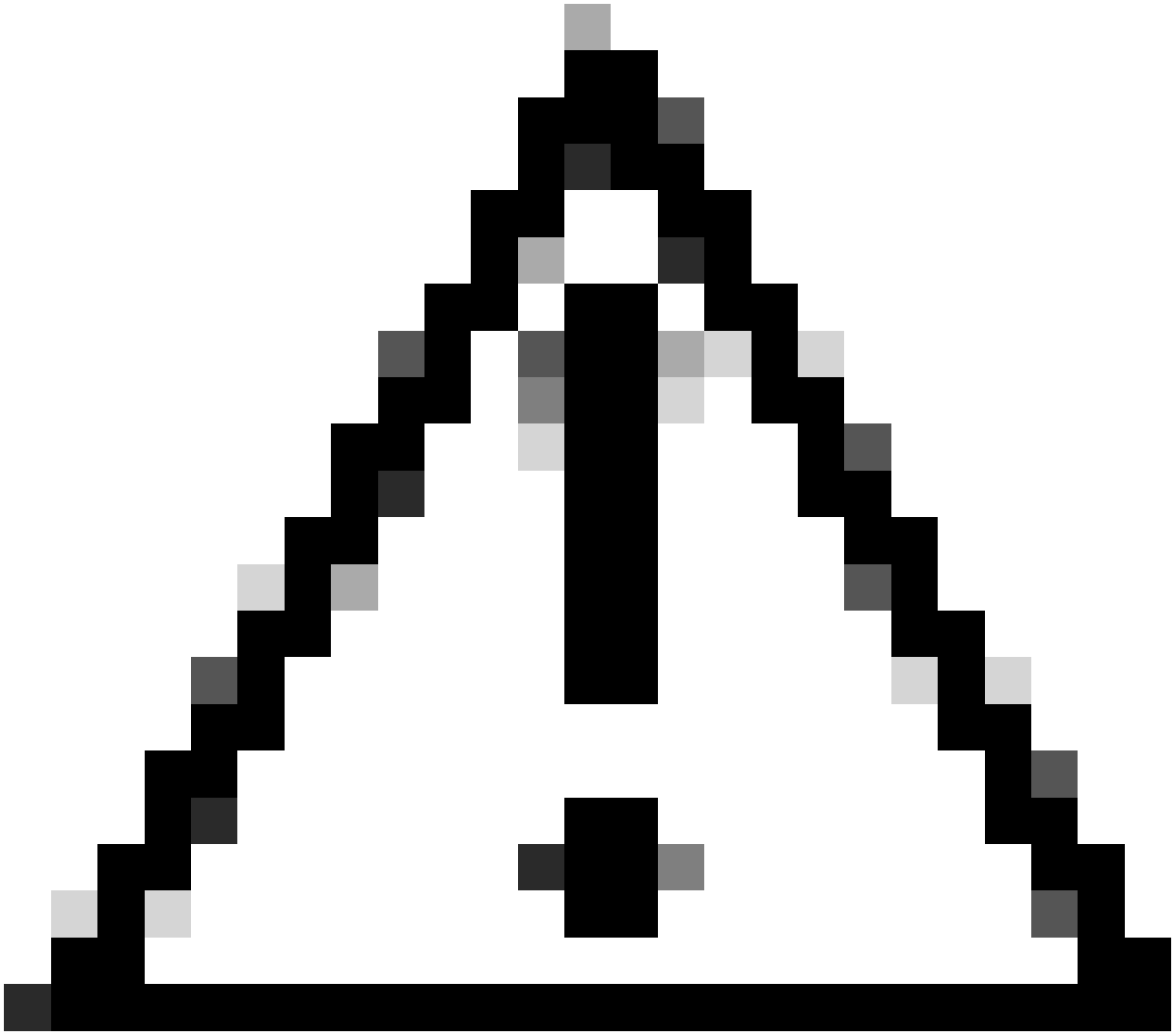
Erstellen Sie den privaten ISE-Schlüssel und den ISE-CSR mit Informationen aus dem Abschnitt ISE_device_req_ext für SANs:

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

OpenSSL öffnet eine interaktive Aufforderung zur Eingabe der DN-Details:

```
-----
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
```

ISE-Zertifikat Distinguished Name Interaktive Aufforderung



Achtung: Die in der interaktiven Eingabeaufforderung angegebene CN muss genau mit einem der Namen im Abschnitt [ISE_alt_names] der Datei openssl.cnf übereinstimmen.

Verwenden Sie die Zertifizierungsstelle mit dem Namen IntermCA, um den ISE-CSR mit dem Namen ISE.csr mit den unter [ISE_cert] definierten Erweiterungen zu signieren und das signierte Zertifikat in ./IntermCA/IntermCA.db.certs/WLC zu speichern. Das ISE-Gerätezertifikat heißt ISE.crt:

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

Zertifikate auf Geräte importieren

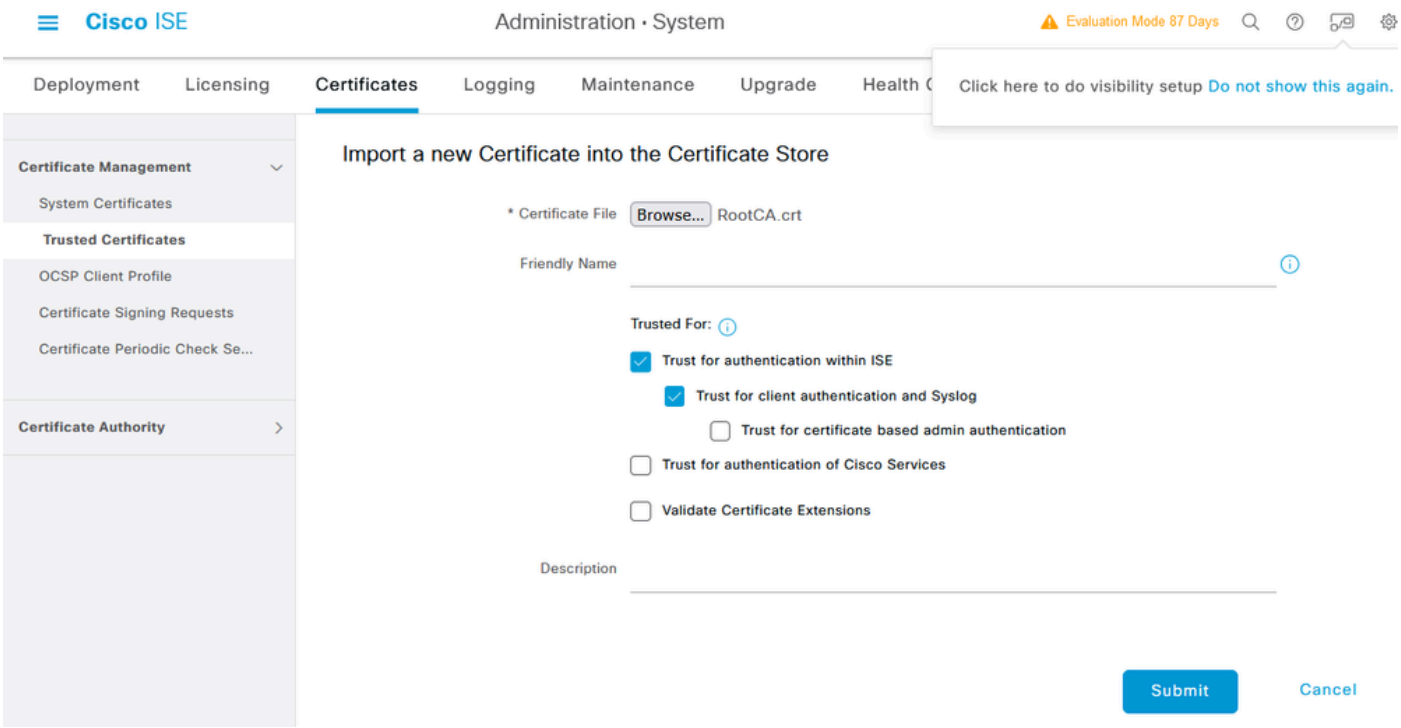
Zertifikate in ISE importieren

1. Importieren Sie das Zertifikat der Stammzertifizierungsstelle aus der ISE-Zertifikatskette in den Speicher für vertrauenswürdige Zertifikate.

2. Navigieren Sie zu Administration>System>Certificates>Trusted Certificates.

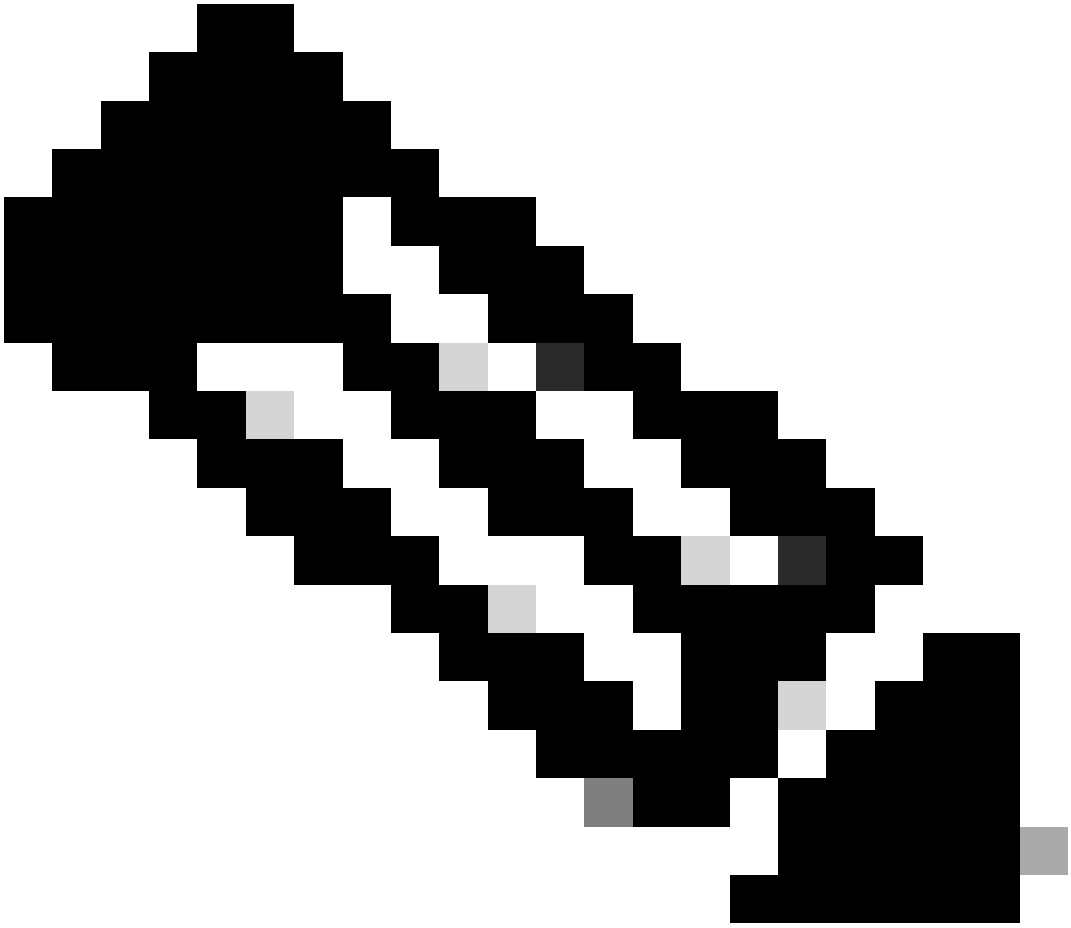
3. Klicken Sie auf Durchsuchen und wählen Sie die Datei Root.crt aus.

4. Aktivieren Sie die Kontrollkästchen Für die Authentifizierung innerhalb der ISE sowie Für die Client-Authentifizierung und Syslog vertrauen, und klicken Sie dann auf Senden:

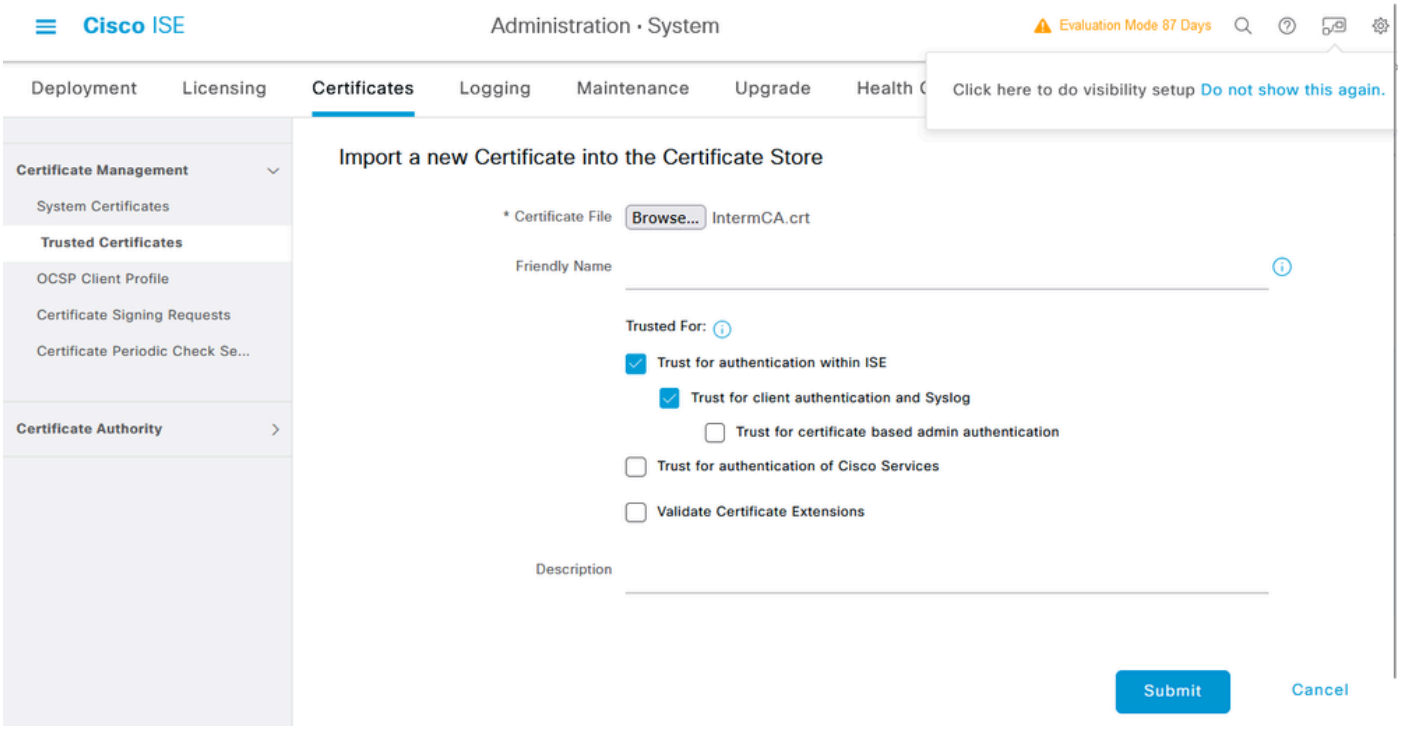


Dialogfeld zum Importieren von ISE-Stammzertifikaten der Zertifizierungsstelle

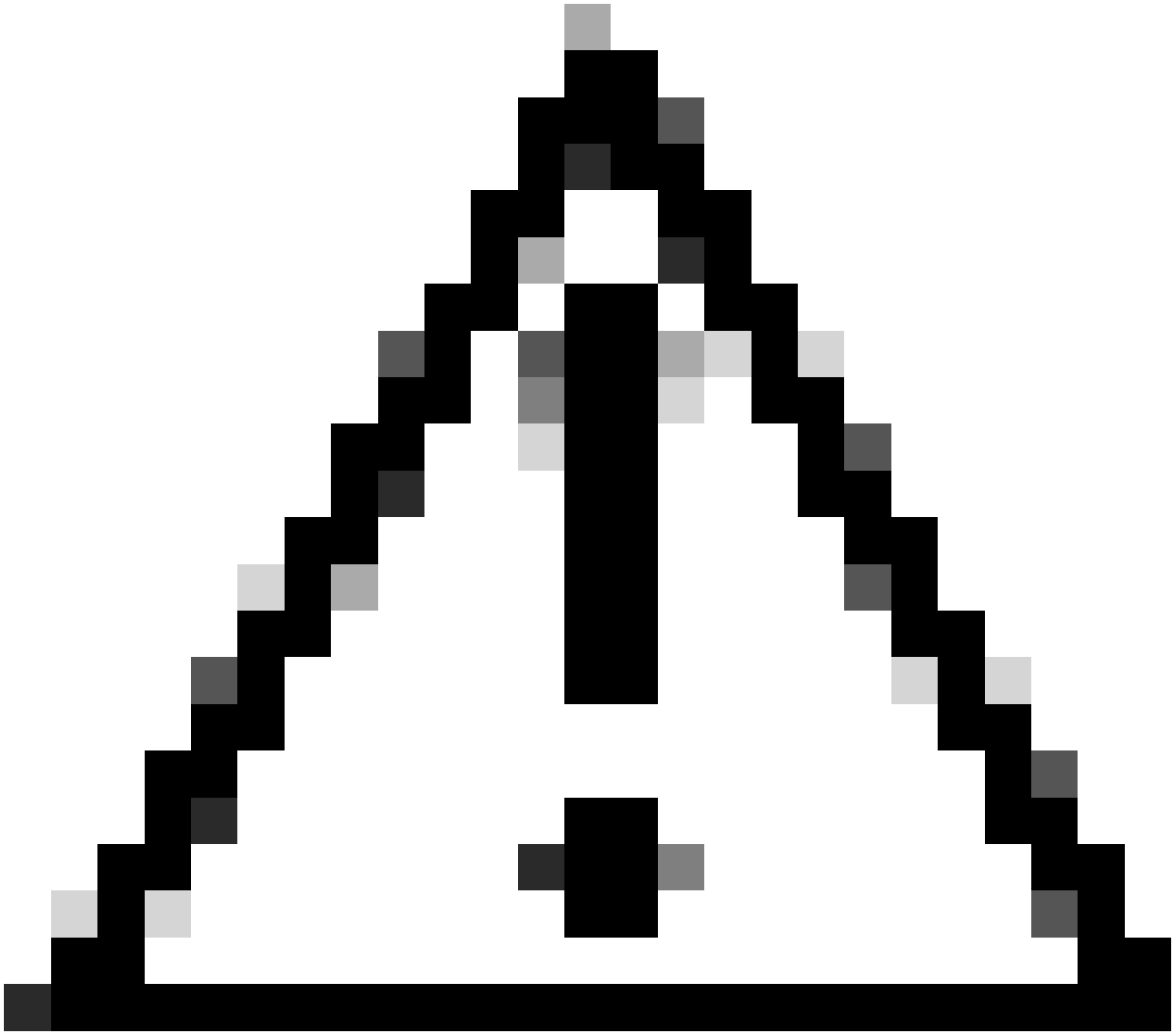
Führen Sie den gleichen Vorgang für das Zwischenzertifikat aus, sofern vorhanden.



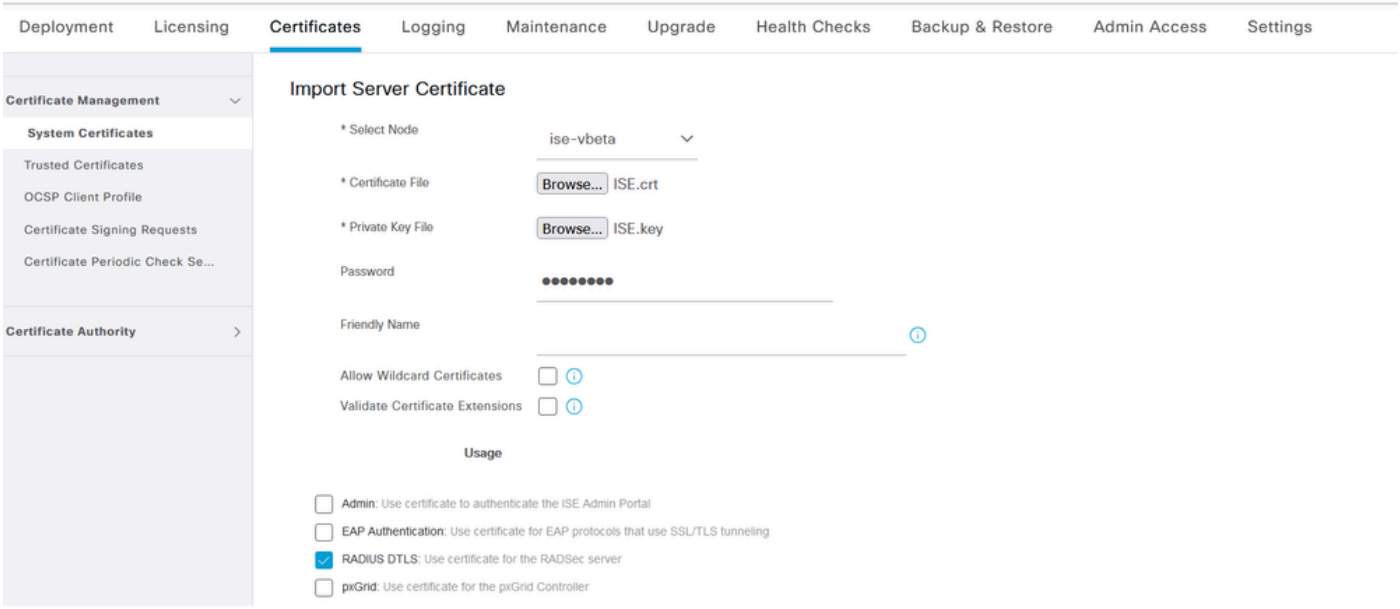
Hinweis: Wiederholen Sie die Schritte für alle Zertifizierungsstellenzertifikate, die Teil der ISE-Zertifikatvalidierungskette sind. Beginnen Sie immer mit dem Zertifikat der Stammzertifizierungsstelle, und beenden Sie es mit dem niedrigsten Zertifikat der Zwischenzertifizierungsstelle der Kette.



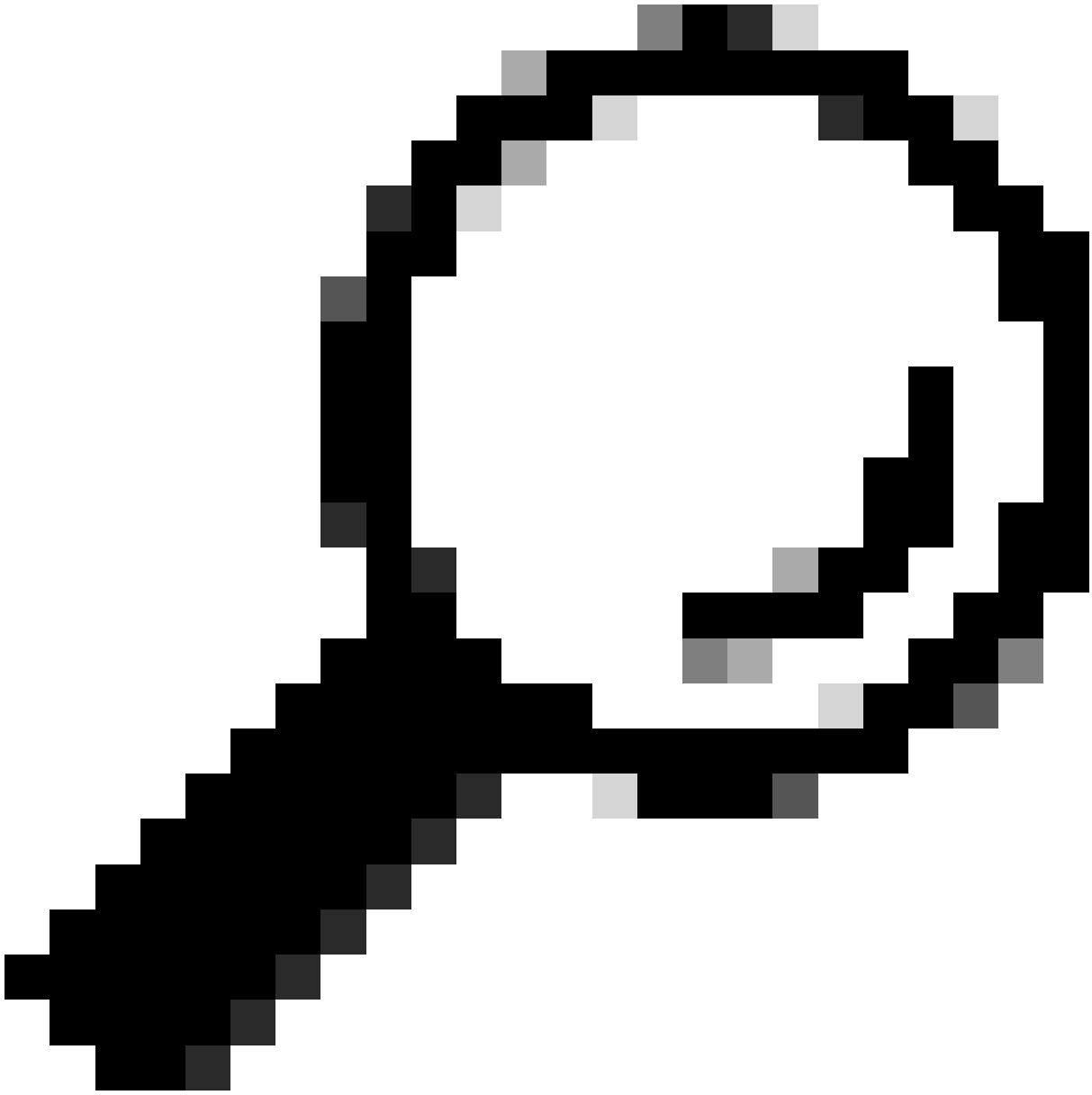
Dialogfeld zum Importieren von ISE-Zwischenzertifikaten



Vorsicht: Wenn das ISE- und das WLC-Zertifikat von verschiedenen Zertifizierungsstellen ausgestellt werden, müssen Sie auch alle Zertifizierungsstellenzertifikate importieren, die zur WLC-Zertifikatskette gehören. Die ISE akzeptiert das WLC-Zertifikat auf dem DTLS-Zertifikataustausch erst, wenn Sie diese Zertifizierungsstellenzertifikate importieren.



ISE-Menü für den Import von Gerätezertifikaten

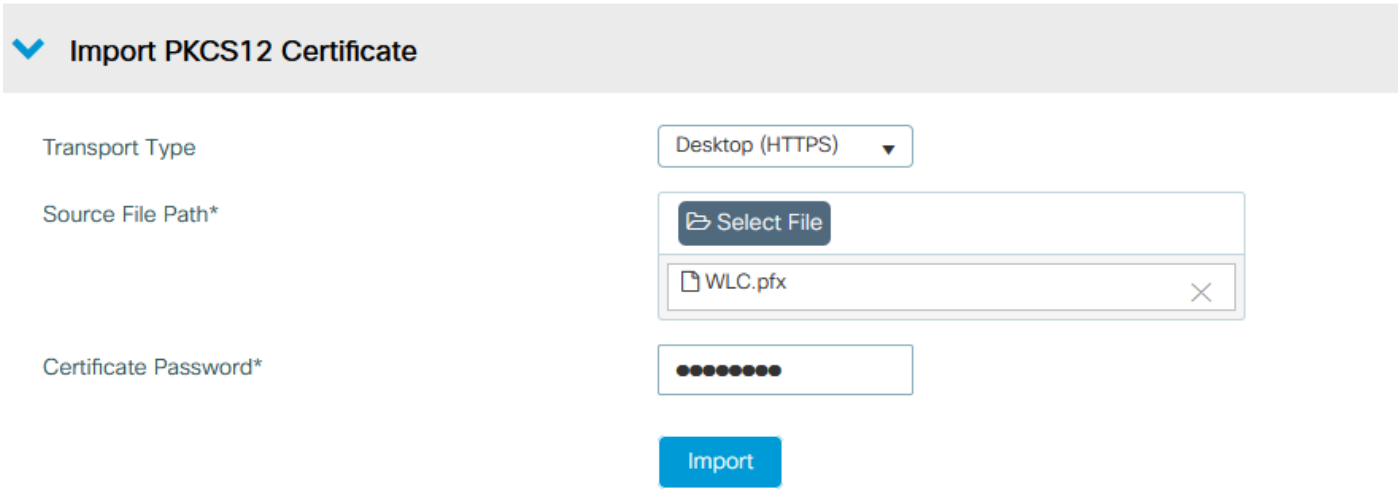


Tipp: Sie müssen in diesem Schritt nur das ISE-Gerätezertifikat importieren. Dieses Zertifikat ist die einzige ISE-Vermittlungsstelle, die den DTLS-Tunnel aufbaut. Das WLC-Gerätezertifikat und der private Schlüssel müssen nicht importiert werden, da das WLC-Zertifikat mit den zuvor importierten CA-Zertifikaten verifiziert wird.

Zertifikate in WLC importieren

1. Navigieren Sie auf dem WLC zu Configuration > Security > PKI Management, und gehen Sie zur Registerkarte Add Certificate.
2. Klicken Sie auf das Dropdown-Menü PKCS12-Zertifikat importieren, und legen Sie als Transporttyp Desktop (HTTPS) fest.
3. Klicken Sie auf die Schaltfläche Select File (Datei auswählen), und wählen Sie die zuvor vorbereitete .pfx-Datei aus.

4. Geben Sie das Importpasswort ein und klicken Sie abschließend auf Importieren.

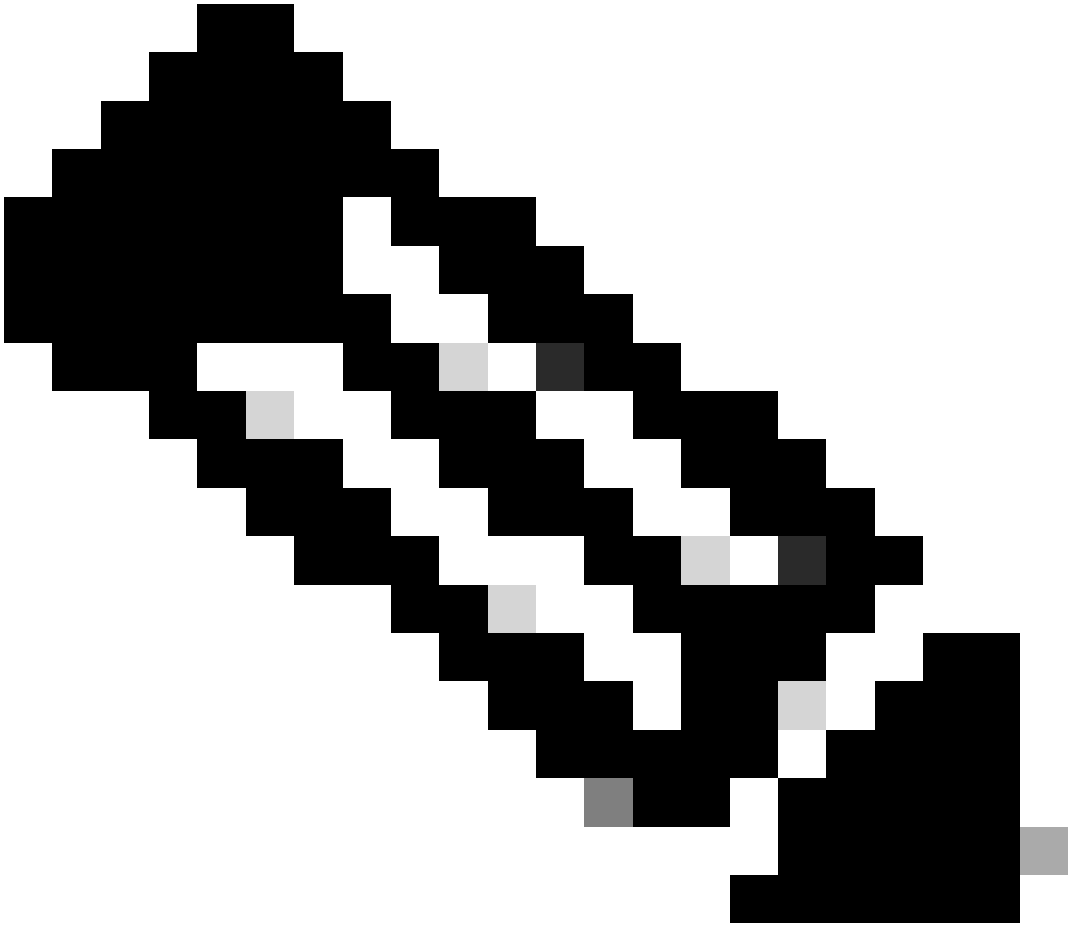


Dialog zum Importieren von WLC-Zertifikaten

Weitere Informationen zum Importvorgang finden Sie unter Generate and Download CSR Certificates on Catalyst 9800 WLCs (CSR-Zertifikate für Catalyst 9800 generieren und herunterladen).

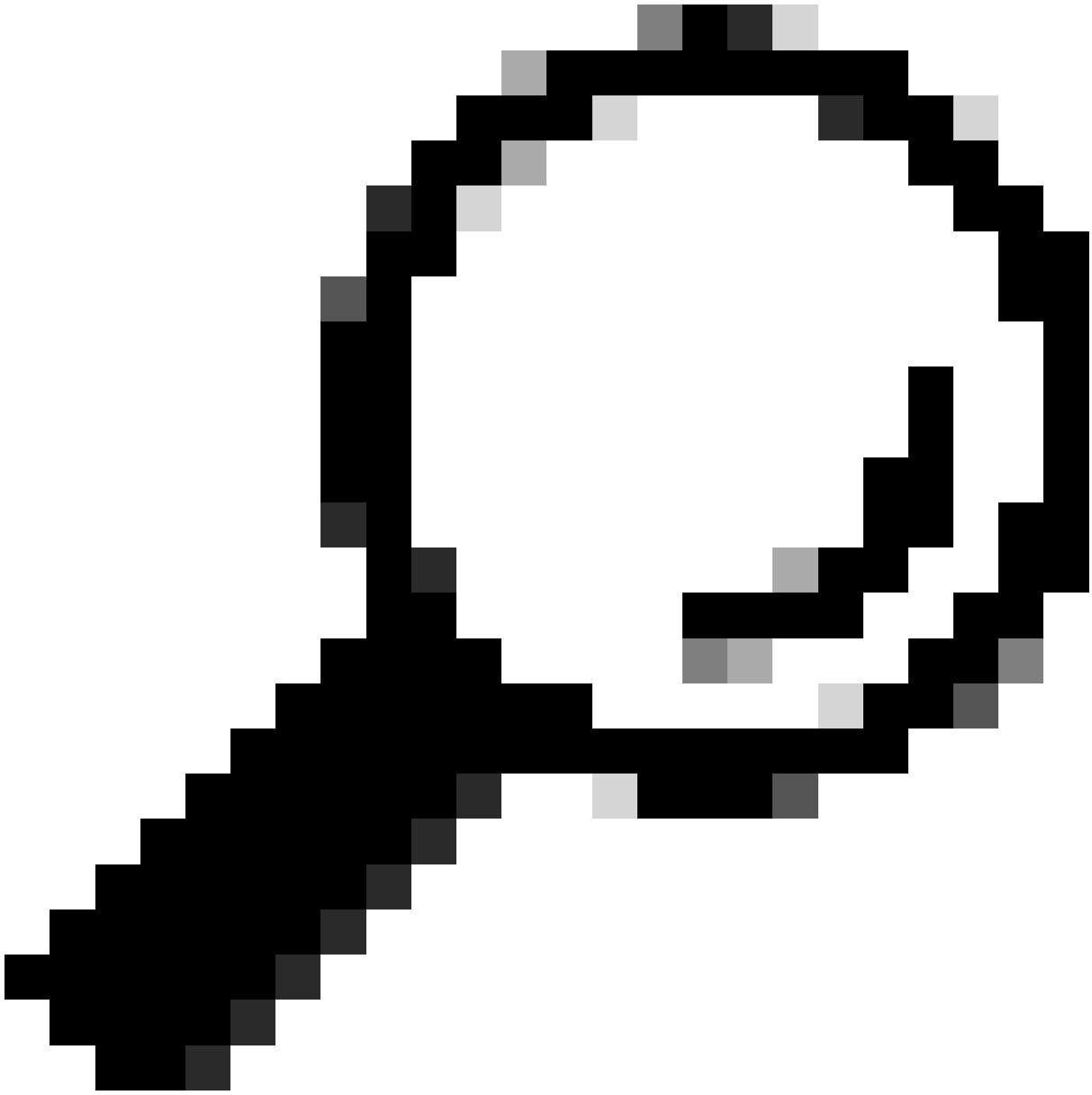
Deaktivieren Sie die Widerrufsprüfung in jedem automatisch erstellten Vertrauenspunkt, wenn der WLC keine Zertifikatsperrliste hat, die er über das Netzwerk überprüfen kann:

```
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
```



Hinweis: Wenn Sie eine mehrstufige Zertifizierungsstelle auf OpenSSL mit dem Dokument Configure Multi-Level CA on OpenSSL to Generate Cisco IOS XE Certificates erstellt haben, müssen Sie die Sperrprüfung deaktivieren, da kein Zertifikatsperrlisten-Server erstellt wurde.

Durch den automatisierten Import werden die erforderlichen Vertrauenspunkte für das WLC-Zertifikat und seine CA-Zertifikate erstellt.



Tipp: Wenn die WLC-Zertifikate von derselben CA wie die ISE-Zertifikate ausgestellt wurden, können Sie dieselben Vertrauenspunkte verwenden, die automatisch aus dem WLC-Zertifikatsimport erstellt wurden. ISE-Zertifikate müssen nicht separat importiert werden.

Wenn das WLC-Zertifikat von einer anderen Zertifizierungsstelle als das ISE-Zertifikat ausgestellt wird, müssen Sie auch die ISE-Zertifizierungsstellenzertifikate in den WLC importieren, damit der WLC dem ISE-Gerätezertifikat vertrauen kann.
Erstellen Sie einen neuen Vertrauenspunkt für die Stammzertifizierungsstelle, und importieren Sie die ISE-Stammzertifizierungsstelle:

```
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-----
```

Importieren Sie das nächste CA-Zwischenzertifikat in der ISE-Zertifizierungsstellenkette, d. h. das von der Stammzertifizierungsstelle ausgestellte Zertifizierungsstellenzertifikat:

```
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-------
```

Jede zusätzliche Zertifizierungsstelle in der Kette erfordert einen separaten Vertrauenspunkt. Jeder Vertrauenspunkt in der Kette muss auf den Vertrauenspunkt verweisen, der das Ausstellerzertifikat des Zertifikats enthält, das Sie mit dem Befehl chain-validation continue **<**Name des Ausstellervertrauenspunkts**>** importieren möchten.

Importieren Sie so viele Zertifizierungsstellenzertifikate, wie Ihre Zertifizierungsstellenkette enthält. Nachdem Sie die Ausstellerzertifizierungsstelle des ISE-Gerätezertifikats importiert haben, notieren Sie sich den Namen dieses Vertrauenspunkts.

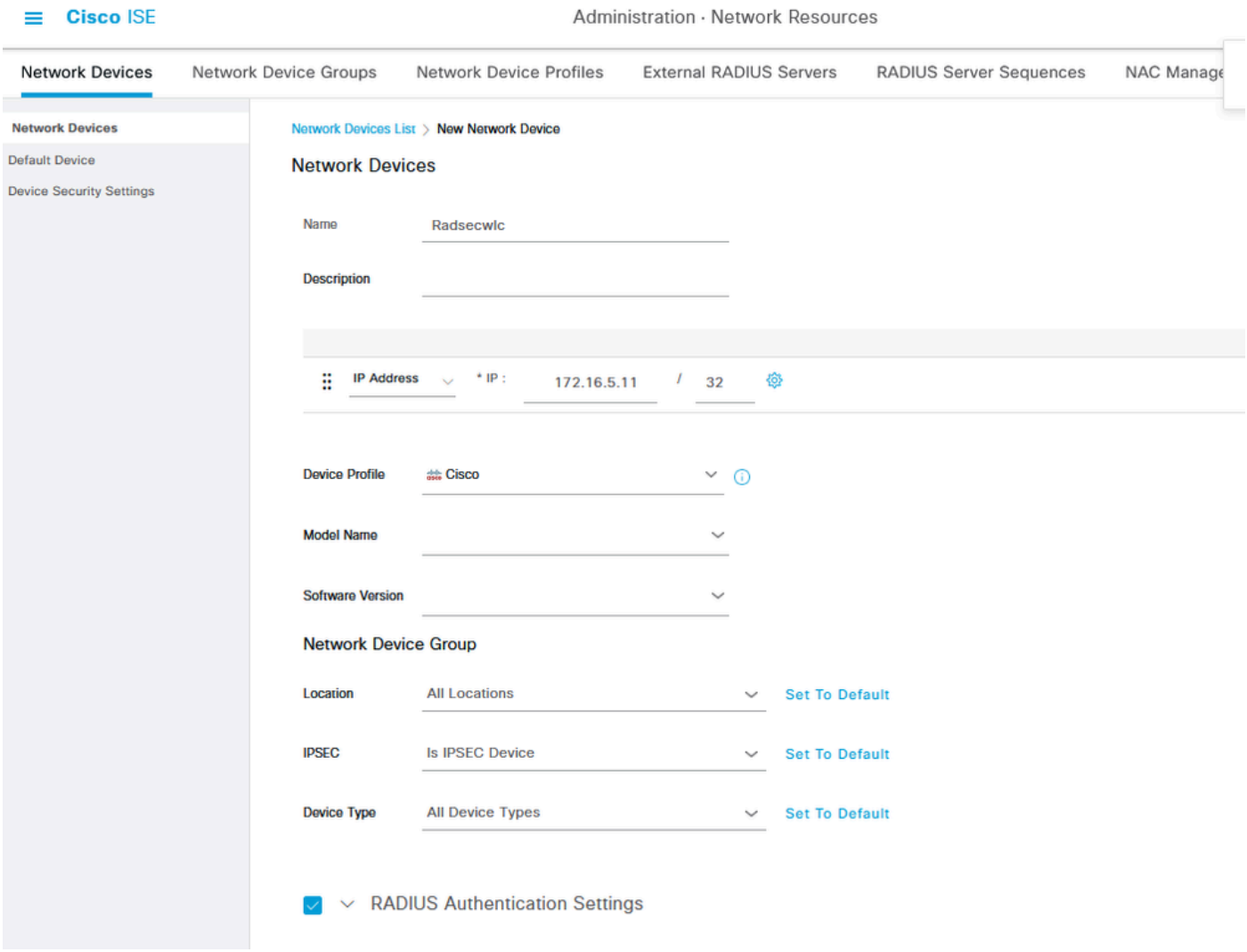
Das ISE-Gerätezertifikat muss nicht auf den WLC importiert werden, damit RADIUS DTLS funktioniert.

RADIUS-DTLS konfigurieren

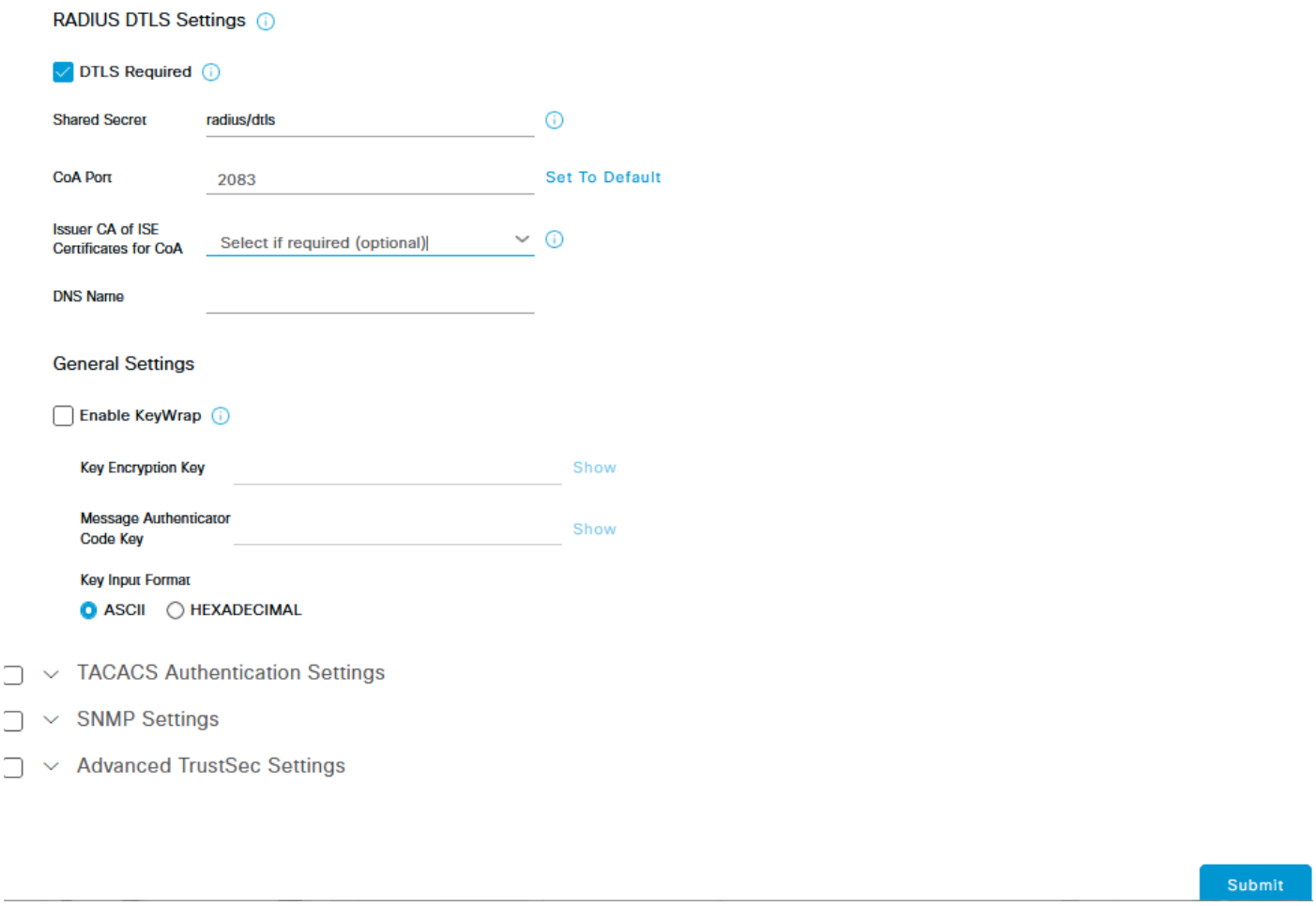
ISE-Konfiguration

Fügen Sie den WLC als Netzwerkgerät zur ISE hinzu. Navigieren Sie dazu zu Administration **>** Network Resources **>** Network devices **>** Add (Verwaltung **>** Netzwerkressourcen **>**

Netzwerkgeräte > Hinzufügen).
Geben Sie den Namen des Geräts und die IP der WLC-Schnittstelle ein, von der der RADIUS-Datenverkehr stammt. In der Regel die Wireless-Management-Schnittstelle IP. Blättern Sie nach unten, und überprüfen Sie die RADIUS-Authentifizierungseinstellungen sowie die Option DTLS erforderlich, und klicken Sie auf Senden:



Neue Netzwerkgerätekonfiguration



Radius-DTLS-Einstellungen für Netzwerkgerät auf der ISE

WLC-Konfiguration

Definieren Sie einen neuen Radius-Server zusammen mit der ISE-IP-Adresse und dem Standard-Port für Radius DTLS. Diese Konfiguration steht nur für die CLI zur Verfügung:

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

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

Radius DTLS muss den gemeinsamen geheimen Radius/dtls verwenden. Der 9800 WLC ignoriert alle konfigurierten Schlüssel, die nicht dieser ist:

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

Konfigurieren Sie mit dem`dtls trustpoint client`
Befehl den Vertrauenspunkt, der das WLC-Gerätezertifikat enthält, für den Austausch gegen den DTLS-Tunnel.
Konfigurieren Sie mit dem`dtls trustpoint server`
Befehl den Vertrauenspunkt, der die Ausstellerzertifizierungsstelle für das ISE-Gerätezertifikat enthält.

Der Client- und der Server-Vertrauenspunktname sind nur dann identisch, wenn das WLC- und das ISE-Zertifikat von derselben Zertifizierungsstelle ausgestellt wurden:

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

Konfigurieren Sie den WLC so, dass er nach einem der SANs (Subject Alternative Names) sucht, die im ISE-Zertifikat vorhanden sind. Diese Konfiguration muss genau einem der SANs entsprechen, die im Feld SANs des Zertifikats vorhanden sind.

Der 9800 WLC führt keine reguläre ausdrucksbasierte Übereinstimmung für das SAN-Feld aus. Dies bedeutet zum Beispiel, dass der Befehl `dtls match-server-identity hostname *.example.com` für ein Platzhalterzertifikat, das *.example.com auf seinem SAN-Feld hat, richtig ist, aber derselbe Befehl für ein Zertifikat, das www.example.com auf dem SAN-Feld enthält, nicht.

Der WLC vergleicht diesen Namen nicht mit einem Namenserver:

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

Erstellen Sie eine neue Servergruppe, um die neue Radius DTLS für die Authentifizierung zu verwenden:

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

Ab diesem Zeitpunkt können Sie diese Servergruppe wie jede andere Servergruppe auf dem WLC verwenden. Weitere Informationen zur Verwendung dieses Servers für die Wireless Client-Authentifizierung finden Sie unter Configure 802.1X Authentication on Catalyst 9800 Wireless Controller Series (Konfigurieren der 802.1X-Authentifizierung auf Catalyst 9800 Wireless Controller-Serien).

Überprüfung

Überprüfen der Zertifikatinformationen

Um die Zertifikatinformationen für die erstellten Zertifikate zu überprüfen, führen Sie auf dem Linux-Terminal den folgenden Befehl aus:

```
openssl x509 -in

      -text -noout
```

Es zeigt die vollständigen Zertifikatinformationen an. Dies ist nützlich, um die Ausstellerzertifizierungsstelle eines bestimmten Zertifikats zu ermitteln oder zu bestimmen, ob die Zertifikate die erforderlichen EKUs und SANs enthalten:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 2 (0x2)
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = MX, ST = CDMX, L = CDMX, O = Cisco lab, OU = Cisco Wireless, CN = Intermediate.example.com
        Validity
            Not Before: Jul 18 19:14:57 2024 GMT
            Not After : Apr 14 19:14:57 2027 GMT
        Subject: C = MX, ST = CDMX, L = CDMX, O = Cisco lab, OU = Cisco Wireless, CN = WLC.example.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:b1:10:7d:6c:6c:14:2f:18:a6:0b:69:d9:60:03:
                    56:2d:48:22:f0:42:10:65:44:24:3b:54:e1:4b:87:
                    b8:ab:c5:5f:f6:a1:a3:5e:f6:3c:c5:45:cc:01:6d:
                    df:e8:a7:81:28:50:44:54:4c:af:a0:56:cf:06:be:
                    10:7e:e2:46:42:ea:3c:b9:d4:03:75:08:84:70:36:
                    bb:3d:95:3b:e2:86:e6:f7:d9:4d:00:28:c4:3c:cb:
                    f8:6d:37:5c:89:28:c1:75:b1:7e:fa:bd:91:cf:8e:
                    5c:a2:37:4f:71:da:6a:04:ee:ba:68:bf:4d:f2:d3:
                    ae:aa:13:42:3b:ff:a0:b3:65:c9:ff:f6:9a:06:d7:
                    6c:08:10:e0:b9:d8:ca:93:2d:e5:5d:7b:74:cd:93:
                    68:b1:46:c7:35:d7:6b:0f:a6:ae:34:e6:23:d1:c8:
                    d3:bf:c0:85:ab:2d:02:a8:dd:54:77:e3:32:61:4e:
                    33:58:b0:62:12:82:42:ae:2b:69:f0:5f:0c:90:c7:
                    9c:ef:b9:9c:fc:29:e2:2c:cb:b4:a9:01:fa:5d:3c:
                    97:11:67:cc:25:96:01:3d:26:1a:43:34:bd:43:b0:
                    a0:f1:ec:a0:c7:98:ad:32:32:99:9c:6b:61:af:57:
                    53:ee:20:cc:d5:ed:db:1c:5c:65:51:42:8c:28:bf:
                    62:bf
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Subject Key Identifier:
                87:89:CA:28:06:95:D5:CE:7C:66:B4:75:81:AA:D4:19:EC:43:01:BB
            X509v3 Authority Key Identifier:
                keyid:2B:08:D8:4C:23:72:5B:62:03:EA:44:F6:9E:D9:F7:75:2E:64:97:DE
                DirName:/C=MX/ST=CDMX/L=CDMX/O=Cisco lab/OU=Cisco Wireless/CN=RootCA
                serial:01
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
            X509v3 Subject Alternative Name:
                DNS:WLC.example.com, DNS:WLC2.example.com
    Signature Algorithm: sha256WithRSAEncryption
    Signature Value:
```

Informationen zu Cisco IOS XE-Gerätezertifikaten wie in OpenSSL dargestellt

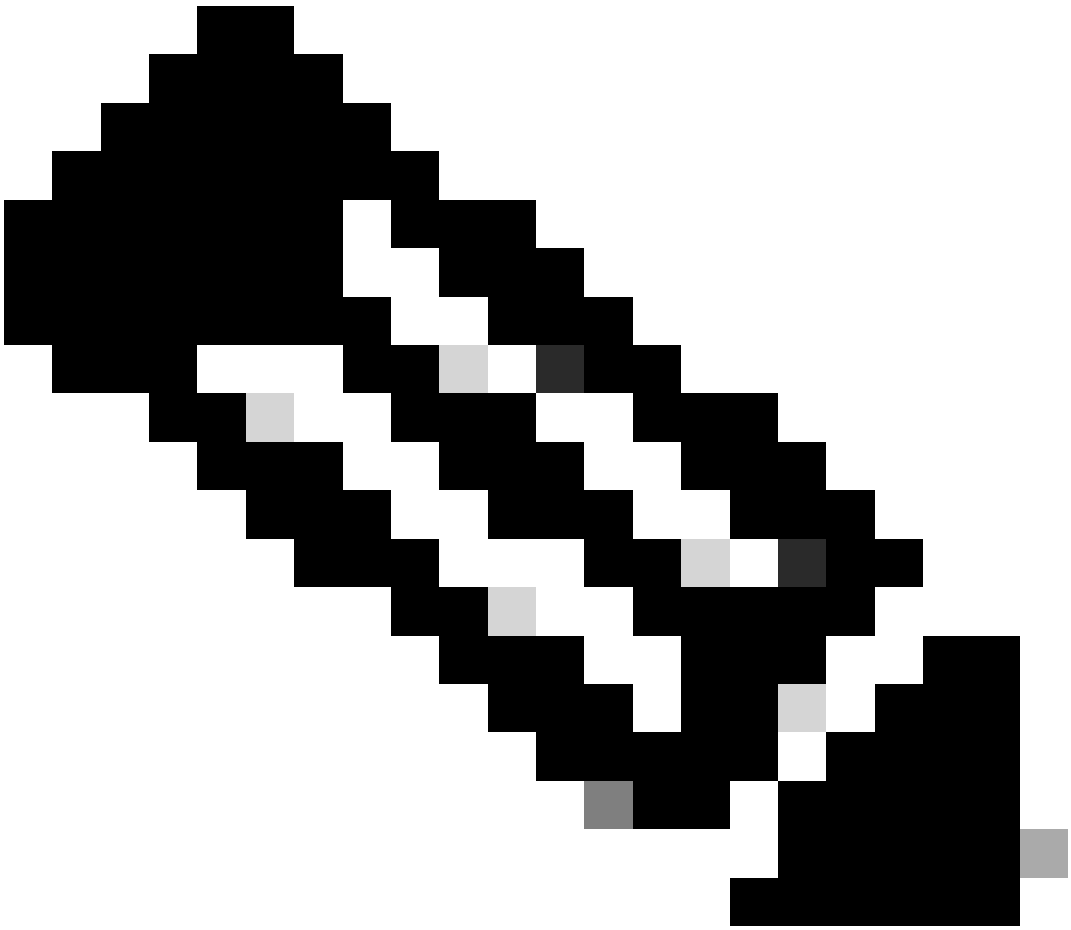
Testauthentifizierung durchführen

Über den WLC können Sie die Radius DTLS-Funktionalität mit folgendem Befehl testen: test aaa group

new-code

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

```
USER ATTRIBUTES

username              0   "testuser"
```



Hinweis: Die Ausgabe des Befehls access reject (Zugriff ablehnen) beim Testbefehl bedeutet, dass der WLC eine Access-Reject RADIUS-Nachricht erhalten hat, in welchem Fall RADIUS DTLS funktioniert. Es kann jedoch auch auf einen Fehler beim Einrichten des DTLS-Tunnels hinweisen. Der Testbefehl unterscheidet nicht zwischen beiden Szenarien. Im Abschnitt zur Fehlerbehebung finden Sie Hinweise, ob ein Problem vorliegt.

Fehlerbehebung

Um die Ursache einer fehlgeschlagenen Authentifizierung zu überprüfen, können Sie diese Befehle aktivieren, bevor Sie eine Testauthentifizierung durchführen.

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

Dies ist die Ausgabe einer erfolgreichen Authentifizierung mit aktiviertem Debugging:

```
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
```

Unbekannte CA von WLC gemeldet

Wenn der WLC die von der ISE bereitgestellten Zertifikate nicht validieren kann, erstellt er keinen DTLS-Tunnel, und die Authentifizierung schlägt fehl.
Dies ist ein Beispiel für die Debugmeldungen, die in diesem Fall angezeigt werden:

```
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:
```

Um dies zu korrigieren, stellen Sie sicher, dass die auf dem WLC konfigurierte Identität genau mit einem der SANs im ISE-Zertifikat übereinstimmt:

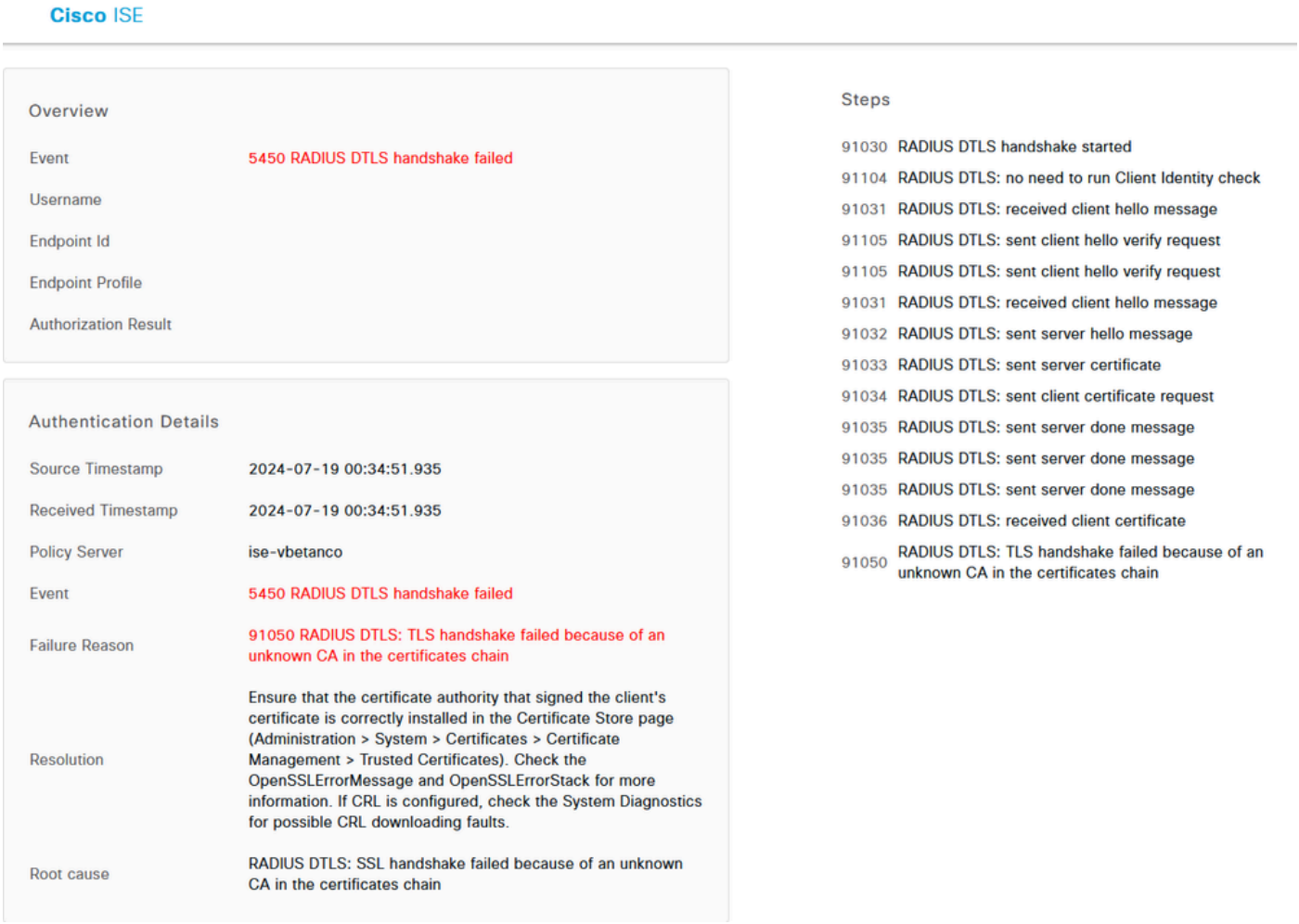
```
9800(config)#radius server
```

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

Stellen Sie sicher, dass die Zertifikatskette der Zertifizierungsstelle ordnungsgemäß auf den Controller importiert wurde und dass die **dtls trustpoint server** configuration uses the Issuer CA trustpoint.

Von ISE gemeldete unbekannte Zertifizierungsstelle

Wenn die ISE die vom WLC bereitgestellten Zertifikate nicht validieren kann, erstellt sie keinen DTLS-Tunnel, und die Authentifizierungen schlagen fehl. Dies wird in den RADIUS Live Logs als Fehler angezeigt. Navigieren Sie zu Operations > Radius > Live logs (Vorgänge > Radius > Live-Protokolle zur Überprüfung).



ISE Live Log meldet DTLS-Handshake-Fehler aufgrund unbekannter CA

Um dies zu korrigieren, stellen Sie sicher, dass Sie über Zwischen- und Stammzertifikate verfügen. Aktivieren Sie dazu die Kontrollkästchen Für Clientauthentifizierung vertrauen und Syslog unter Administration > System > Certificates > Trusted Certificates (Verwaltung > Zertifikate > Vertrauenswürdige Zertifikate).

Sperrprüfung wurde eingerichtet

Wenn die Zertifikate in den WLC importiert werden, ist für die neu erstellten Vertrauenspunkte die Sperrprüfung aktiviert. Dadurch versucht der WLC, nach einer Zertifikatssperrliste zu suchen, die nicht verfügbar oder erreichbar ist, und schlägt bei der Zertifikatsüberprüfung fehl.
Stellen Sie sicher, dass jeder Vertrauenspunkt im Prüfpfad für die Zertifikate den Befehl revocation-

check none enthält.

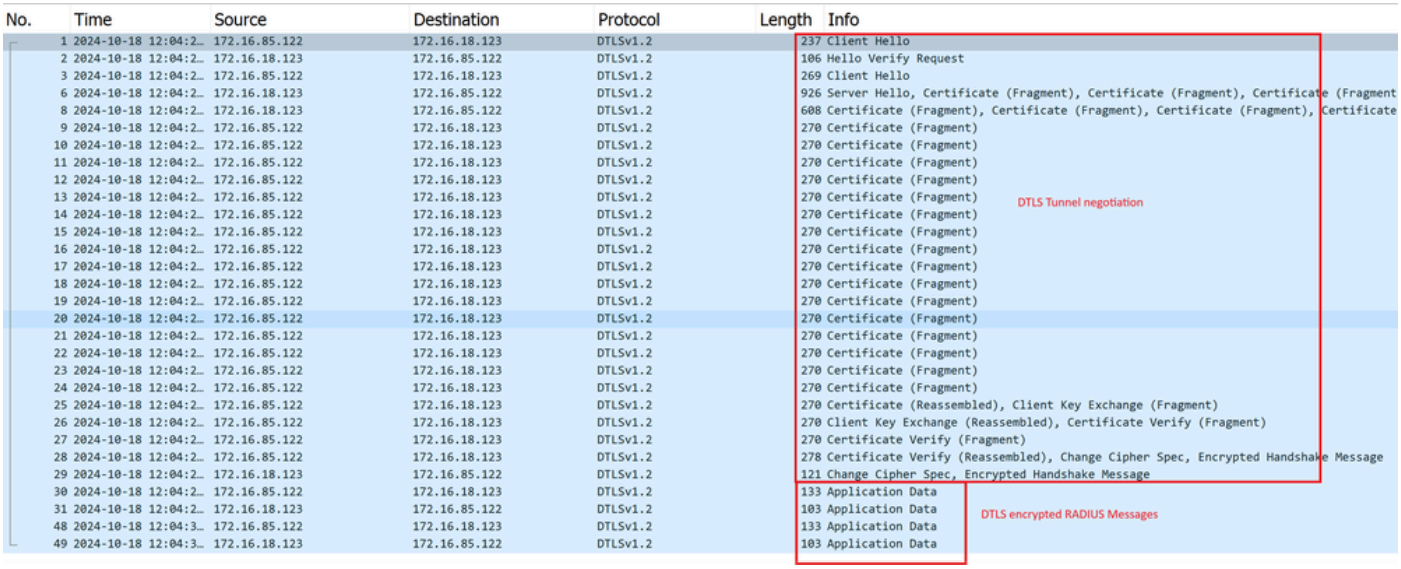
```
Jul 17 21:50:39.064: RADIUS_RADSEC_HASH_KEY_MATCH: hashkey1(0) matches hashkey2(0) TRUE
Jul 17 21:50:39.064: RADIUS_RADSEC_HASH_KEY_GET_CTX: radius radsec  sock_ctx(0x780FB0715978:0) get for
Jul 17 21:50:39.064: RADIUS_RADSEC_PROCESS_SOCK_EVENT: Handle socket event for TLS handshake(172.16.18.
Jul 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
```

Fehlerbehebung bei DTLS-Tunnelaufbau bei Paketerfassung

Der 9800 WLC bietet die Embedded Packet Capture (EPC)-Funktion, mit der Sie den gesamten gesendeten und empfangenen Datenverkehr für eine bestimmte Schnittstelle erfassen können. Die ISE bietet eine ähnliche Funktion namens TCP-Dump zur Überwachung von ein- und ausgehendem Datenverkehr. Wenn sie gleichzeitig verwendet werden, können Sie den Datenverkehr des DTLS-Sitzungsaufbaus aus der Perspektive beider Geräte analysieren.

Detaillierte Informationen zur Konfiguration des TCP-Dump auf der ISE finden Sie im Cisco Identity Services Engine-Administratorhandbuch. Informationen zur Konfiguration der EPC-Funktion auf dem WLC finden Sie auch unter Fehlerbehebung bei Catalyst 9800 Wireless LAN Controllern.

Dies ist ein Beispiel für die erfolgreiche Einrichtung eines DTLS-Tunnels.



Paketerfassung bei RADIUS DTLS Tunnelaushandlung und verschlüsselten Nachrichten

Paketerfassungen zeigen, wie der DTLS-Tunnel eingerichtet wird. Wenn bei der Aushandlung ein Problem auftritt, beispielsweise aufgrund von verlorenem Datenverkehr zwischen Geräten oder DTLS-verschlüsselten Warnpaketen, können Sie das Problem mithilfe der Paketerfassung identifizieren.

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.