Finesse-fout "SSPeerUnverifiedException" voor gadgets die op CA-ondertekende servers worden gehost

Inhoud

[Inleiding](#)
[Voorwaarden](#)
[Vereisten](#)
[Gebruikte componenten](#)
[Achtergrondinformatie](#)
[Problemen](#)
[Scenario 1: De hostserver onderhandelt over onveilige TLS](#)
[Oplossing](#)
[Scenario 2: Het certificaat heeft een niet-ondersteund ondertekeningsalgoritme](#)
[Oplossing](#)

Inleiding

Dit document beschrijft de stappen om problemen op te lossen in het scenario waarbij een certificaatautoriteit (CA)-ondertekende certificaatketen wordt geüpload naar Finesse voor een externe webserver die een gadget host, maar de gadget niet te laden wanneer u inlogt bij Finesse en u de fout "SSLPeerUnverifiedException" ziet.

Bijgedragen door Gino Schweinsberger, Cisco TAC Engineer.

Voorwaarden

Vereisten

Cisco raadt kennis van de volgende onderwerpen aan:

- SSL-certificaten
- Finesse administratie
- Windows-serverbeheer
- Packet-opnameanalyse met Wireshark

Gebruikte componenten

De informatie in dit document is gebaseerd op de volgende softwareversies:

- Unified Contact Center Express (UCS X) 11.x
- Finesse 11.x

De informatie in dit document is gebaseerd op de apparaten in een specifieke

laboratoriumomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u zorgen dat u de potentiële impact van elke opdracht begrijpt.

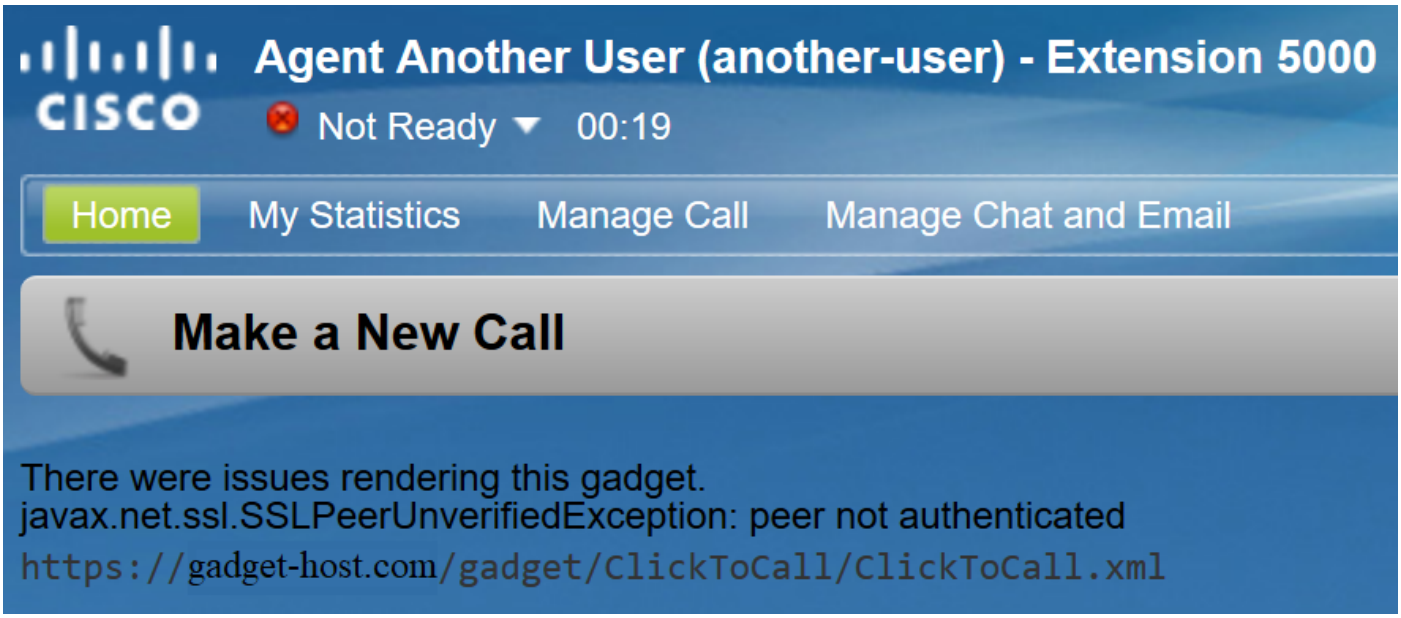
Achtergrondinformatie

Dit zijn de voorwaarden waaronder de fout kan optreden:

- Stel dat de certificaatvertrouwensketen is geüpload naar Finesse
- Zorg ervoor dat de juiste servers/services opnieuw zijn opgestart
- Stel dat de gadget is toegevoegd aan de Finesse-lay-out met een HTTPS URL en dat de URL bereikbaar is

Dit is de fout die is waargenomen wanneer de agent zich aanmeldt bij Finesse:

"Er waren problemen met het maken van dit gadget. javax.net.ssl.SLPeerUnverifiedException: peer niet geverifieerd"



Problemen

Scenario 1: De hostserver onderhandelt over onveilige TLS

Wanneer Finesse Server een verbindingsverzoek doet aan de Hosting-server, adverteert Finesse Tomcat een lijst met encryptie-algoritmen die zij ondersteunt.

Sommige algoritmen worden niet ondersteund vanwege beveiligingskwetsbaarheden,

Als de Hosting server een van deze algoritmen selecteert, wordt de verbinding geweigerd:

- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA

Deze algoritmen zijn gekend om zwakke efemere sleutels van Diffie-Hellman te gebruiken wanneer het over de verbinding onderhandelt, en de kwetsbaarheid Logjam maakt dit een slechte

keus voor verbindingen van TLS.

Volg het TLS-handshake-proces in een pakketopname om te zien welk algoritme wordt onderhandeld.

1. Finesse presenteert de lijst met ondersteunde algoritmen in de stap **Client Hello**:

```
▼ TLSv1 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 67
  ▼ Handshake Protocol: Client Hello
      Handshake Type: Client Hello (1)
      Length: 63
      Version: TLS 1.0 (0x0301)
    ▶ Random: 5cacb293b5efdb4cf1bb34464d7de9f5060b00a9beb81d29…
      Session ID Length: 0
      Cipher Suites Length: 24
    ▼ Cipher Suites (12 suites)
        Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
        Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
        Cipher Suite: TLS_DHE_DSS_WITH_AES_256_CBC_SHA (0x0038)
        Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
        Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x0033)
        Cipher Suite: TLS_DHE_DSS_WITH_AES_128_CBC_SHA (0x0032)
        Cipher Suite: TLS_RSA_WITH_RC4_128_SHA (0x0005)
        Cipher Suite: TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x000a)
        Cipher Suite: TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA (0x0016)
        Cipher Suite: TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA (0x0013)
        Cipher Suite: TLS_RSA_WITH_RC4_128_MD5 (0x0004)
        Cipher Suite: TLS_EMPTY_RENEGOTIATION_INFO_SCSV (0x00ff)
      Compression Methods Length: 1
    ▶ Compression Methods (1 method)
```

2. Voor deze verbinding werd **TLS_DHE_RSA_WITH_AES_256_CBC_SHA** geselecteerd door de Hosting server tijdens de stap **Server Hello** omdat dat hoger is op zijn lijst van voorkeursalgoritmen.

```
TLSv1 Record Layer: Handshake Protocol: Multiple Handshake Messages
  Content Type: Handshake (22)
  Version: TLS 1.0 (0x0301)
  Length: 2557
  Handshake Protocol: Server Hello
    Handshake Type: Server Hello (2)
    Length: 77
    Version: TLS 1.0 (0x0301)
    Random: 5cacb292c4d7183627f620a066f9b6ce6460dcb849b59cae…
    Session ID Length: 32
    Session ID: 4c290000ce66098cc994a33e193b0da1244cb9f083f69c26…
    Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
    Compression Method: null (0)
    Extensions Length: 5
    Extension: renegotiation_info (len=1)
  Handshake Protocol: Certificate
  Handshake Protocol: Server Key Exchange
    Handshake Type: Server Key Exchange (12)
    Length: 1032
    Diffie-Hellman Server Params
  Handshake Protocol: Server Hello Done
    Handshake Type: Server Hello Done (14)
    Length: 0
```

3. Finesse stuurt een fatale waarschuwing en beëindigt de verbinding:

```
TLSv1 Record Layer: Alert (Level: Fatal, Description: Internal Error)
  Content Type: Alert (21)
  Version: TLS 1.0 (0x0301)
  Length: 2
  Alert Message
```

Oplossing

Om het gebruik van deze algoritmen te voorkomen, moet de Hosting server geconfigureerd zijn om deze een lage prioriteit te geven, of ze moeten volledig uit de lijst met beschikbare algoritmen verwijderd worden. Dit kan worden gedaan op een Windows Server met de Windows Group Policy editor (gpedit.msc).

> **Opmerking**: Voor meer informatie over de effecten van Logjam in Finesse en het gebruik van gpedit, check:

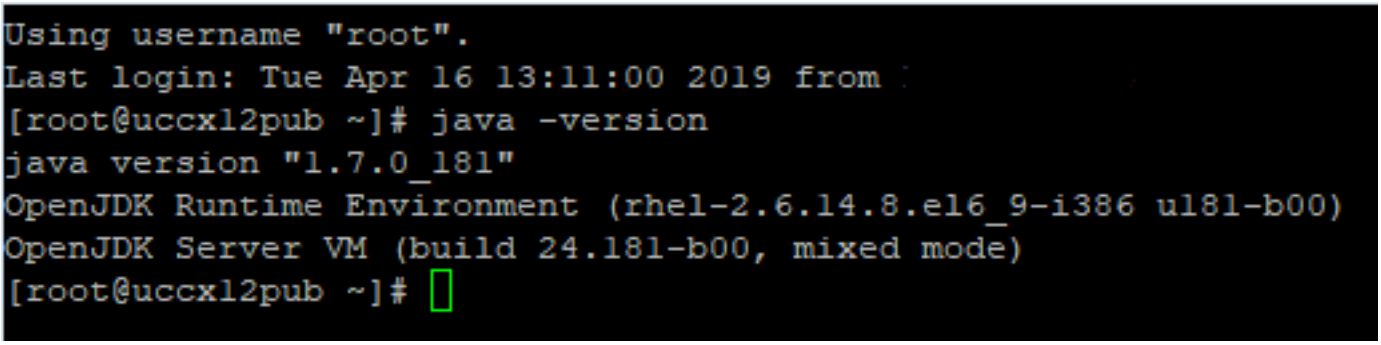
Scenario 2: Het certificaat heeft een niet-ondersteund ondertekeningsalgoritme

Windows Server-certificeringsinstanties kunnen nieuwere handtekeningsstandaarden gebruiken om certificaten te ondertekenen. Zelfs het biedt grotere veiligheid dan SHA, is de goedkeuring van deze normen buiten de producten van Microsoft laag en beheerders zullen waarschijnlijk in interoperabiliteitskwesties lopen.

Finesse Tomcat vertrouwt op de SunMSCAPI security provider van Java om ondersteuning mogelijk te maken voor de verschillende handtekeningsalgoritmen en cryptografische functies die door Microsoft worden gebruikt. Alle huidige versies van Java (1.7, 1.8 en 1.9) ondersteunen alleen deze handtekeningalgoritmen:

- MD5 met RSA
- MD2S met RSA
- GEEN met RSA
- SHA1 met RSA
- SHA256 met RSA
- SHA384 met RSA
- SHA512 met RSA

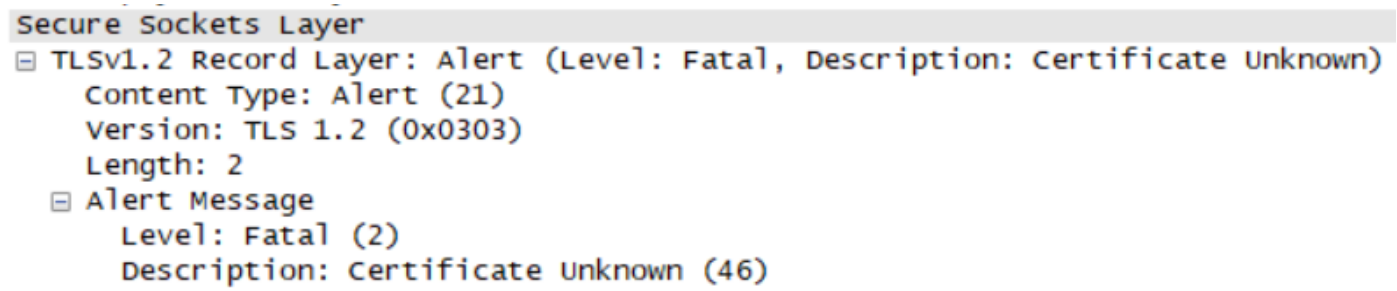
Het is een goed idee om de versie van Java die op de server van Finesse draait te controleren om te bevestigen welke algoritmen in die versie worden ondersteund. De versie kan via deze opdracht worden gecontroleerd vanuit root toegang: **java - versie**



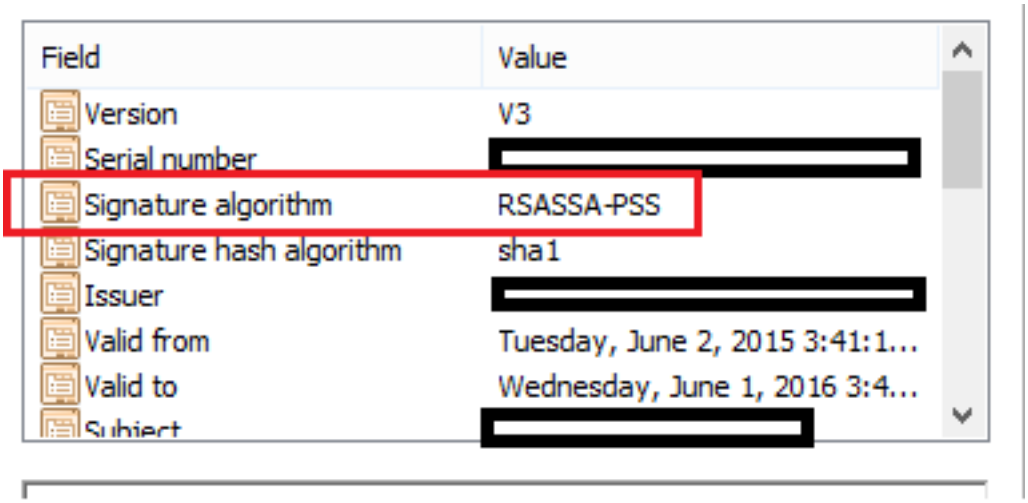
> **Opmerking**: voor meer informatie over de Java SunMSCAPI-provider raadpleegt u https://docs.oracle.com/javase/8/docs/technotes/guides/security/SunProviders.html#SunMSCAPI

Als een certificaat wordt voorzien van een andere handtekening dan de hierboven genoemde, kan Finesse het certificaat niet gebruiken om een TLS-verbinding te maken met de hostserver. Hiertoe behoren certificaten die zijn ondertekend met een ondersteund handtekeningtype, maar zijn afgegeven door certificeringsinstanties die hun eigen midden- en basiscertificaten hebben ondertekend met iets anders.

Als u een pakketopname bekijkt, sluit Finesse de verbinding met een "noodsignaal: Fout: certificaat onbekend", zoals in de afbeelding.



Op dit punt is nodig om de certificaten te controleren die worden aangeboden door de Hosting server en te zoeken naar niet-ondersteunde handtekeningsalgoritmen. Het is gebruikelijk om **RSASSA-PSS** als problematisch handtekeningsalgoritme te zien:



Als een certificaat in de keten is ondertekend met RSASSA-PSS, mislukt de verbinding. In dit geval toont de pakketopname aan dat de Root CA RSASSA-PSS gebruikt voor zijn eigen certificaat:

```
Certificates (3906 bytes)
    Certificate Length: 1728
  Certificate: 308206bc308205a4a00302010202137400000243b805da9... (id-at-commonName=[redacted]
    signedCertificate
    algorithmIdentifier (sha256WithRSAEncryption)
      Padding: 0
      encrypted: e6230df257be9d34c0f57bc2f88c081c4186aaad092c8155...
    Certificate Length: 1114
  Certificate: 308204563082033ea00302010202131600000000a93cd17d6... (id-at-commonName=[redacted]Issuing Authority[redacted]
    signedCertificate
    algorithmIdentifier (sha256WithRSAEncryption)
      Padding: 0
      encrypted: 889be6a1125c758cd0009b392d3b90a69b64546dcee09c84...
    Certificate Length: 1055
  Certificate: 3082041b308202cfa00302010202107b70dbb7c2760da74f... (id-at-commonName=[redacted]Root CA [redacted]
    signedCertificate
    algorithmIdentifier (id-RSASSA-PSS)
      Algorithm Id: 1.2.840.113549.1.1.10 (id-RSASSA-PSS)
    RSASSA-PSS-params
      Padding: 0
      encrypted: d8e9151adc76b4e55f9277fce916613ce26199e3b50dcb54...
```

Oplossing

Om dit probleem op te lossen, moet er een nieuw certificaat worden afgegeven door een CA-provider die alleen een van de ondersteunde SunMSCAPI-handtekeningstypen gebruikt die in de gehele certificaatketen worden vermeld, zoals eerder uitgelegd.

> **Opmerking**: voor meer informatie over het RSASSA-PSS-handtekeningsalgoritme, surf naar https://pkisolutions.com/pkcs1v2-1rsassa-pss/

> **Opmerking:** Deze kwestie wordt gevolgd in het gebrek CSCve79330

Over deze vertaling

Cisco heeft dit document vertaald via een combinatie van machine- en menselijke technologie om onze gebruikers wereldwijd ondersteuningscontent te bieden in hun eigen taal. Houd er rekening mee dat zelfs de beste machinevertaling niet net zo nauwkeurig is als die van een professionele vertaler. Cisco Systems, Inc. is niet aansprakelijk voor de nauwkeurigheid van deze vertalingen en raadt aan altijd het oorspronkelijke Engelstalige document (link) te raadplegen.