Solucionar problemas de erro do Finesse "SSLPeerUncheckedException" para gadgets hospedados em servidores assinados por CA

Contents

Introduction

Este documento descreve as etapas para solucionar problemas no cenário em que uma cadeia de certificados assinada por uma autoridade de certificação (CA) é carregada no Finesse para um servidor Web externo que hospeda um gadget, mas o gadget não é carregado quando você faz login no Finesse e você vê o erro "SSLPeerUncheckedException".

Contribuição de Gino Schweinsberger, engenheiro do Cisco TAC.

Prerequisites

Requirements

A Cisco recomenda que você tenha conhecimento destes tópicos:

- Certificados SSL
- Administração Finesse
- Administração do Windows Server
- Análise de captura de pacotes com o Wireshark

Componentes Utilizados

As informações neste documento são baseadas nestas versões de software:

- Unified Contact Center Express (UCCX) 11.X
- Finesse 11.X

The information in this document was created from the devices in a specific lab environment. All of

the devices used in this document started with a cleared (default) configuration. Se a rede estiver ativa, certifique-se de que você entenda o impacto potencial de qualquer comando.

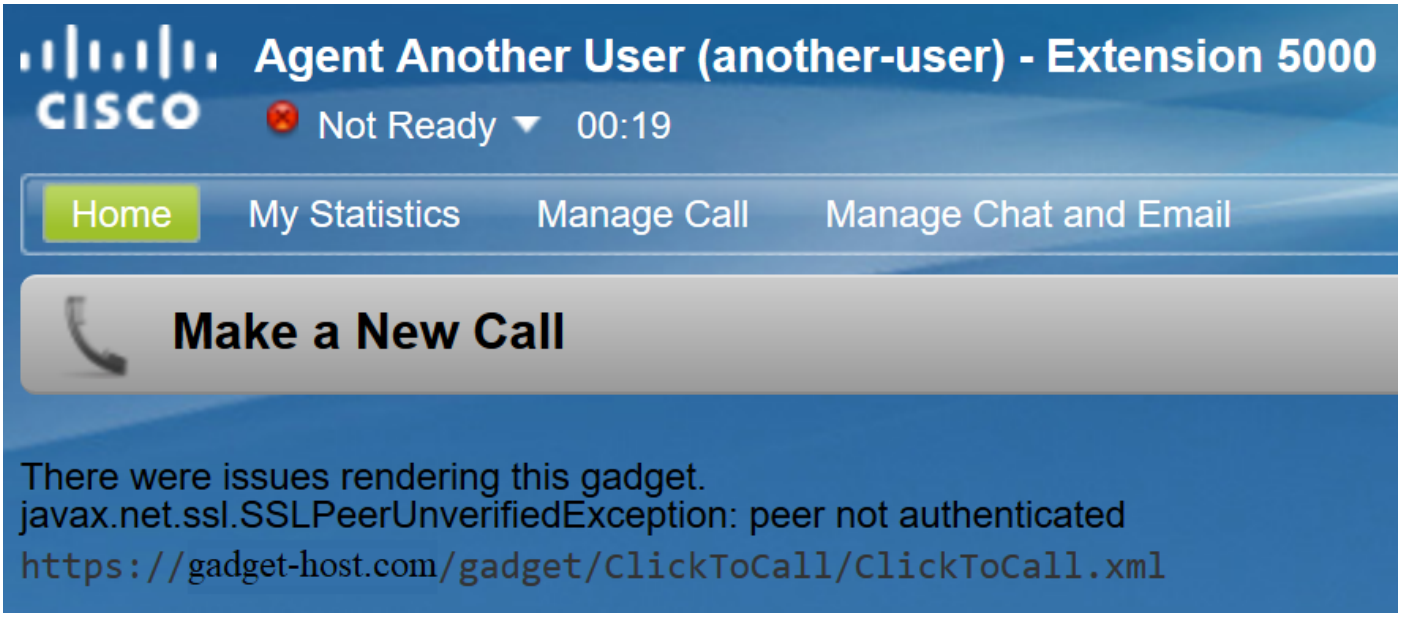
Informações de Apoio

Estas são as condições para que o erro ocorra:

- Assumir que a cadeia de certificados confiáveis foi carregada no Finesse
- Verifique se os servidores/serviços corretos foram reiniciados
- Suponha que o gadget tenha sido adicionado ao layout Finesse com uma URL HTTPS e que a URL esteja acessível

Este é o erro observado quando o agente faz logon no Finesse:

"Houve problemas ao processar este gadget. javax.net.ssl.SSLPeerUncheckedException: peer not authenticated"



Problemas

Cenário 1: O servidor host negocia TLS não seguro

Quando o Finesse Server faz uma solicitação de conexão ao servidor de hospedagem, o Finesse Tomcat anuncia uma lista de cifras de criptografia que ele suporta.

Algumas cifras não são suportadas devido a vulnerabilidades de segurança,

Se o servidor de hospedagem selecionar uma dessas cifras, a conexão será recusada:

- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA

Essas cifras são conhecidas por usar chaves Diffie-Hellman efêmeras fracas quando negociam a conexão, e a vulnerabilidade Logjam torna isso uma má escolha para conexões TLS.

Siga o processo de handshake TLS em uma captura de pacote para ver qual cifra é negociada.

1. A Finesse apresenta sua lista de cifras suportadas na etapa **Hello do cliente**:

```
▼ 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. Para esta conexão, o **TLS_DHE_RSA_WITH_AES_256_CBC_SHA** foi selecionado pelo servidor de hospedagem durante a etapa **Hello do servidor**, porque ele está no topo de sua lista de cifras preferenciais.

```
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. O Finesse envia um alerta Fatal e encerra a conexão:

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

Solução

Para evitar o uso dessas cifras, o servidor de hospedagem deve ser configurado para dar a elas uma prioridade baixa ou elas devem ser removidas da lista de cifras disponíveis completamente. Isso pode ser feito em um Windows Server com o editor de Diretiva de Grupo do Windows (gpedit.msc).

> **Nota**: Para mais detalhes sobre os efeitos do Logjam no Finesse e o uso do gpedit, verifique:

Cenário 2: O certificado tem um algoritmo de assinatura sem suporte

As autoridades de certificação do Windows Server podem usar padrões de assinatura mais recentes para assinar certificados. Mesmo oferecendo mais segurança que o SHA, a adoção desses padrões fora dos produtos da Microsoft é baixa e os administradores provavelmente terão problemas de interoperabilidade.

O Finesse Tomcat conta com o provedor de segurança SunMSCAPI do Java para permitir o suporte a vários algoritmos de assinatura e funções criptográficas usados pela Microsoft. Todas as versões atuais do Java (1.7, 1.8 e 1.9) suportam somente estes algoritmos de assinatura:

- MD5com RSA
- MD2com RSA
- NENHUMcomRSA
- SHA1com RSA
- SHA256com RSA
- SHA384com RSA
- SHA512com RSA

Éuma boa ideia verificar a versão do Java que é executada no servidor Finesse para confirmar quais algoritmos são suportados nessa versão. A versão pode ser verificada a partir do acesso raiz com este comando: **java -version**

```
Using username "root".
Last login: Tue Apr 16 13:11:00 2019 from
[root@uccxl2pub ~]# java -version
java version "1.7.0_181"
OpenJDK Runtime Environment (rhel-2.6.14.8.el6_9-i386 u181-b00)
OpenJDK Server VM (build 24.181-b00, mixed mode)
[root@uccxl2pub ~]#
```

> **Observação**: para obter mais detalhes sobre o provedor Java SunMSCAPI, consulte https://docs.oracle.com/javase/8/docs/technotes/guides/security/SunProviders.html#SunMSCAPI

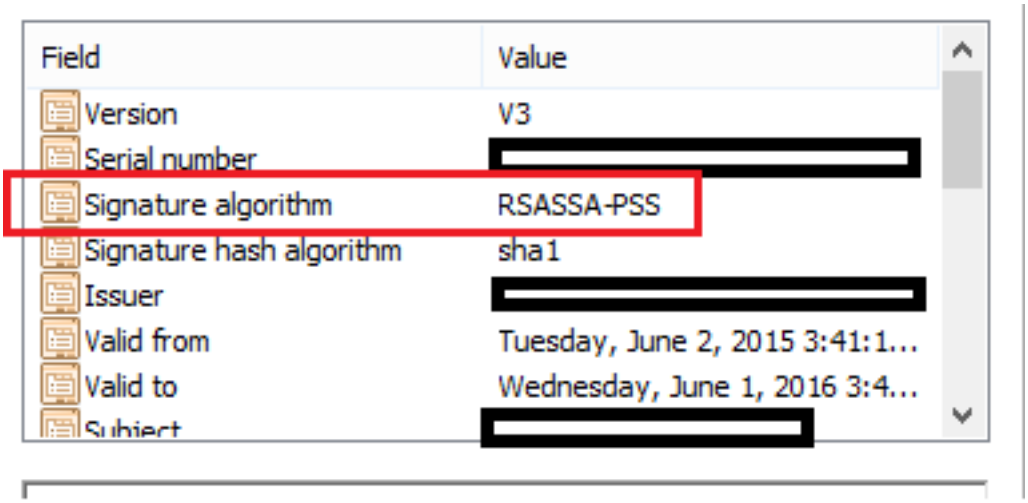
Se um certificado for fornecido com uma assinatura diferente das listadas acima, o Finesse não poderá usar o certificado para criar uma conexão TLS com o servidor de hospedagem. Isso inclui certificados assinados com um tipo de assinatura suportado, mas emitidos por autoridades de certificado que têm seus próprios certificados raiz e intermediários assinados com outra coisa.

Se você observar uma captura de pacote, o Finesse fecha a conexão com um "alerta fatal: Certificate Unknown", conforme mostrado na imagem.

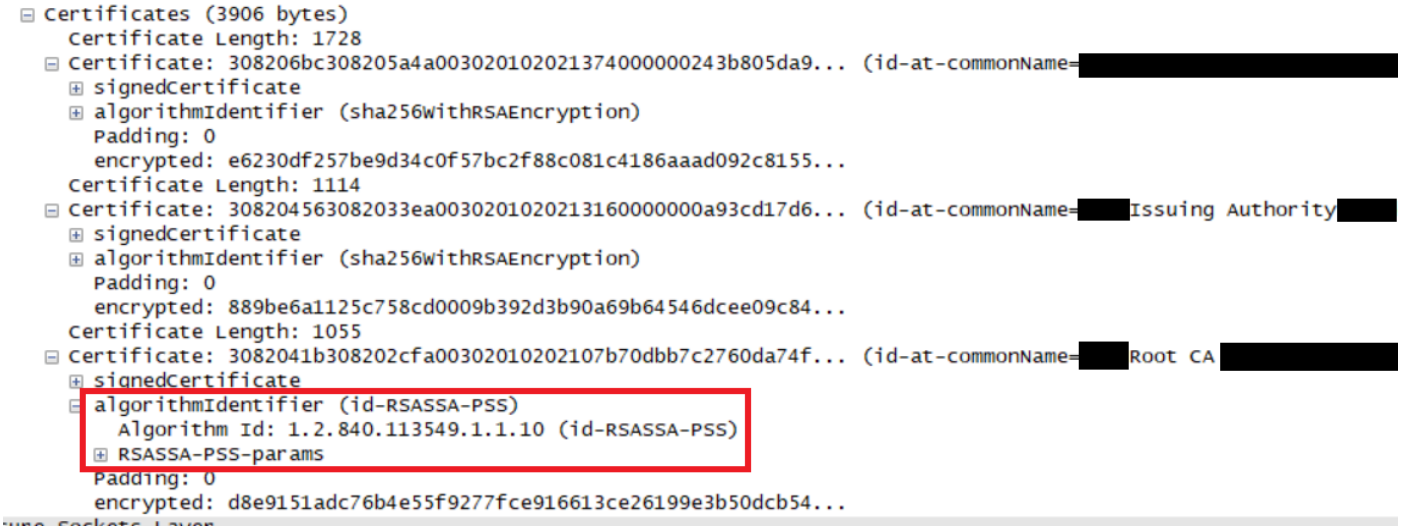
```
Secure Sockets Layer
⊟ TLSv1.2 Record Layer: Alert (Level: Fatal, Description: Certificate Unknown)
    Content Type: Alert (21)
    Version: TLS 1.2 (0x0303)
    Length: 2
  ⊟ Alert Message
      Level: Fatal (2)
      Description: Certificate Unknown (46)
```

Neste ponto, é necessário verificar os certificados apresentados pelo servidor de hospedagem e

procurar algoritmos de assinatura não suportados. É comum ver o **RSASSA-PSS** como o algoritmo de assinatura problemático:



Se algum certificado da cadeia estiver assinado com RSASSA-PSS, a conexão falhará. Nesse caso, a captura do pacote mostra que a CA raiz usa RSASSA-PSS para seu próprio certificado:



Solução

Para resolver esse problema, um novo certificado deve ser emitido por um provedor de CA que use apenas um dos tipos de assinatura SunMSCAPI com suporte listados em toda a cadeia de certificados, conforme explicado anteriormente.

> **Observação**: para obter mais detalhes sobre o algoritmo de assinatura RSASSA-PSS, consulte https://pkisolutions.com/pkcs1v2-1rsassa-pss/

> **Note**: Esse problema é rastreado no CSCve com defeito79330

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.