

# De configuratie van het SSL-algoritme van Expressway aanpassen

## Inhoud

---

[Inleiding](#)

[Voorwaarden](#)

[Vereisten](#)

[Gebruikte componenten](#)

[Achtergrondinformatie](#)

[De coderingsstring inspecteren](#)

[Inspecteer de cifeeronderhandeling in de TLS-handdruk met een pakketvastlegging](#)

[Configureren](#)

[Een specifiek algoritme uitschakelen](#)

[Een groep algoritmen uitschakelen met een gemeenschappelijk algoritme](#)

[Verifiëren](#)

[Inspecteer de lijst met cijfers die zijn toegestaan door de reeks coderingen](#)

[Test een TLS-verbinding door te onderhandelen over een uitgeschakeld algoritme](#)

[Inspecteer een pakketvastlegging van een TLSH en schud met behulp van een uitgeschakeld coderingssysteem](#)

[Gerelateerde informatie](#)

---

## Inleiding

Dit document beschrijft de stappen om de vooraf ingestelde algoritme koorden op Expressway aan te passen.

## Voorwaarden

### Vereisten

Cisco raadt kennis van de volgende onderwerpen aan:

- Cisco Express voor Cisco VCS.
- TLS-protocol.

### Gebruikte componenten

De informatie in dit document is gebaseerd op de volgende software- en hardware-versies:

- Cisco Expressway versie X15.0.2.1

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

De standaard Expressway-configuratie omvat pregeconfigureerde algoritme strings, die om compatibiliteitsredenen ondersteuning voor bepaalde algoritmen mogelijk maken die als zwak kunnen worden beschouwd in het kader van een bepaald bedrijfsbeveiligingsbeleid. Het is mogelijk om de algoritme koorden aan te passen om hen te verfijnen om het specifieke beleid van elke omgeving te passen.

In Expressway, is het mogelijk om een onafhankelijk cijferkoord voor elk van deze protocollen te vormen:

- HTTPS
- LDAP
- Reverse proxy
- SIP
- SMTP
- TMS-provisioning
- UCS-serverdetectie
- XMPP

De algoritme koorden gehoorzamen aan het OpenSSL-formaat dat in het [OpenSSL-coderingsmanpage wordt](#) beschreven. De huidige Expressway versie X15.0.2 wordt geleverd met de standaard string EECDH:EDH:HIGH:-AES256+SHA:!MEDIUM:!LOW:!3DES:!MD5:!PSK:!NULL:!aNULL:!aDH is vooraf geconfigureerd voor alle protocollen in gelijke mate. Van de web admin pagina, onder Onderhoud > Beveiliging > Cijfers, kunt u de algoritme wijzigen die aan elk protocol is toegewezen, om specifieke algoritmen of groepen van algoritmen toe te voegen of te verwijderen met behulp van een gemeenschappelijk algoritme.

### De coderingsstring inspecteren

Door de opdracht openssl-algoritmen -V "<algoritme string>" te gebruiken, kunt u een lijst met alle algoritmen uitvoeren die door een bepaalde tekenreeks zijn toegestaan, wat nuttig is voor de visuele inspectie van de algoritmen. Dit voorbeeld toont de uitvoer bij het inspecteren van de standaard expressway algoritme string:

```
<#root>
```

```
~ #
```

```
openssl ciphers -V "EECDH:EDH:HIGH:-AES256+SHA:!MEDIUM:!LOW:!3DES:!MD5:!PSK:!eNULL:!aNULL:!aDH"
```

```
0x13,0x02 - TLS_AES_256_GCM_SHA384 TLSv1.3 Kx=any Au=any Enc=AESGCM(256) Mac=AEAD
```

```
0x13,0x03 - TLS_CHACHA20_POLY1305_SHA256 TLSv1.3 Kx=any Au=any Enc=CHACHA20/POLY1305(256) Mac=AEAD
```

```

0x13,0x01 - TLS_AES_128_GCM_SHA256 TLSv1.3 Kx=any Au=any Enc=AESGCM(128) Mac=AEAD
0xC0,0x2C - ECDHE-ECDSA-AES256-GCM-SHA384 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESGCM(256) Mac=AEAD
0xC0,0x30 - ECDHE-RSA-AES256-GCM-SHA384 TLSv1.2 Kx=ECDH Au=RSA Enc=AESGCM(256) Mac=AEAD
0xCC,0xA9 - ECDHE-ECDSA-CHACHA20-POLY1305 TLSv1.2 Kx=ECDH Au=ECDSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
0xCC,0xA8 - ECDHE-RSA-CHACHA20-POLY1305 TLSv1.2 Kx=ECDH Au=RSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
0xC0,0xAD - ECDHE-ECDSA-AES256-CCM TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESCCM(256) Mac=AEAD
0xC0,0x2B - ECDHE-ECDSA-AES128-GCM-SHA256 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESGCM(128) Mac=AEAD
0xC0,0x2F - ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2 Kx=ECDH Au=RSA Enc=AESGCM(128) Mac=AEAD
0xC0,0xAC - ECDHE-ECDSA-AES128-CCM TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESCCM(128) Mac=AEAD
0xC0,0x24 - ECDHE-ECDSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AES(256) Mac=SHA384
0xC0,0x28 - ECDHE-RSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=RSA Enc=AES(256) Mac=SHA384
0xC0,0x23 - ECDHE-ECDSA-AES128-SHA256 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AES(128) Mac=SHA256
0xC0,0x27 - ECDHE-RSA-AES128-SHA256 TLSv1.2 Kx=ECDH Au=RSA Enc=AES(128) Mac=SHA256
0xC0,0x09 - ECDHE-ECDSA-AES128-SHA TLSv1 Kx=ECDH Au=ECDSA Enc=AES(128) Mac=SHA1
0xC0,0x13 - ECDHE-RSA-AES128-SHA TLSv1 Kx=ECDH Au=RSA Enc=AES(128) Mac=SHA1
0x00,0xA3 - DHE-DSS-AES256-GCM-SHA384 TLSv1.2 Kx=DH Au=DSS Enc=AESGCM(256) Mac=AEAD
0x00,0x9F - DHE-RSA-AES256-GCM-SHA384 TLSv1.2 Kx=DH Au=RSA Enc=AESGCM(256) Mac=AEAD
0xCC,0xAA - DHE-RSA-CHACHA20-POLY1305 TLSv1.2 Kx=DH Au=RSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
0xC0,0x9F - DHE-RSA-AES256-CCM TLSv1.2 Kx=DH Au=RSA Enc=AESCCM(256) Mac=AEAD
0x00,0xA2 - DHE-DSS-AES128-GCM-SHA256 TLSv1.2 Kx=DH Au=DSS Enc=AESGCM(128) Mac=AEAD
0x00,0x9E - DHE-RSA-AES128-GCM-SHA256 TLSv1.2 Kx=DH Au=RSA Enc=AESGCM(128) Mac=AEAD
0xC0,0x9E - DHE-RSA-AES128-CCM TLSv1.2 Kx=DH Au=RSA Enc=AESCCM(128) Mac=AEAD
0x00,0x6B - DHE-RSA-AES256-SHA256 TLSv1.2 Kx=DH Au=RSA Enc=AES(256) Mac=SHA256
0x00,0x6A - DHE-DSS-AES256-SHA256 TLSv1.2 Kx=DH Au=DSS Enc=AES(256) Mac=SHA256
0x00,0x67 - DHE-RSA-AES128-SHA256 TLSv1.2 Kx=DH Au=RSA Enc=AES(128) Mac=SHA256
0x00,0x40 - DHE-DSS-AES128-SHA256 TLSv1.2 Kx=DH Au=DSS Enc=AES(128) Mac=SHA256
0x00,0x33 - DHE-RSA-AES128-SHA SSLv3 Kx=DH Au=RSA Enc=AES(128) Mac=SHA1
0x00,0x32 - DHE-DSS-AES128-SHA SSLv3 Kx=DH Au=DSS Enc=AES(128) Mac=SHA1
0x00,0x9D - AES256-GCM-SHA384 TLSv1.2 Kx=RSA Au=RSA Enc=AESGCM(256) Mac=AEAD
0xC0,0x9D - AES256-CCM TLSv1.2 Kx=RSA Au=RSA Enc=AESCCM(256) Mac=AEAD
0x00,0x9C - AES128-GCM-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AESGCM(128) Mac=AEAD
0xC0,0x9C - AES128-CCM TLSv1.2 Kx=RSA Au=RSA Enc=AESCCM(128) Mac=AEAD
0x00,0x3D - AES256-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AES(256) Mac=SHA256
0x00,0x3C - AES128-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AES(128) Mac=SHA256
0x00,0x2F - AES128-SHA SSLv3 Kx=RSA Au=RSA Enc=AES(128) Mac=SHA1
~ #

```

## Inspecteer de cifeeronderhandeling in de TLS-handdruk met een pakketvastlegging

Door een TLS-onderhandeling in een pakketopname vast te leggen, kunt u de details van de algoritmeonderhandeling controleren met behulp van Wireshark.

Het TLS-handshake-proces omvat een ClientHello-pakket dat door het clientapparaat is verzonden en de lijst met de door het apparaat ondersteunde algoritmen bevat volgens de ingestelde algoritme voor het verbindingsprotocol. De server bekijkt de lijst, vergelijkt het met zijn eigen lijst van toegestane algoritmen (bepaald door zijn eigen algoritme string), en kiest een algoritme dat beide systemen ondersteunen, dat gebruikt moet worden voor de versleutelde sessie. Dan antwoordt het met een ServerHello pakket dat op het gekozen algoritme wijst. Er zijn belangrijke verschillen tussen de TLS 1.2 en 1.3 handshake dialogen, maar het algoritme onderhandelingsmechanisme gebruikt dit zelfde principe in beide versies.

Dit is een voorbeeld van een TLS 1.3 algoritme onderhandeling tussen een webbrowser en Expressway op poort 443 zoals gezien in Wireshark:

No.	Time	Source	Src port	Destination	Dst port	Protocol	Length	Info
3186	2024-07-14 23:28:55.675989	10.15.1.2	29986	10.15.1.7	443	TCP	66	29986 → 443 [SYN, ECE, CWR] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
3187	2024-07-14 23:28:55.676309	10.15.1.7	443	10.15.1.2	29986	TCP	66	443 → 29986 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
3188	2024-07-14 23:28:55.676381	10.15.1.2	29986	10.15.1.7	443	TCP	54	29986 → 443 [ACK] Seq=1 Ack=1 Win=4204800 Len=0
3189	2024-07-14 23:28:55.679410	10.15.1.2	29986	10.15.1.7	443	TLSv1.2	248	Client Hello
3190	2024-07-14 23:28:55.679651	10.15.1.7	443	10.15.1.2	29986	TCP	60	443 → 29986 [ACK] Seq=1 Ack=195 Win=64128 Len=0
3194	2024-07-14 23:28:55.686008	10.15.1.7	443	10.15.1.2	29986	TLSv1.2	1514	Server Hello
3195	2024-07-14 23:28:55.686008	10.15.1.7	443	10.15.1.2	29986	TLSv1.2	1514	Certificate
3196	2024-07-14 23:28:55.686097	10.15.1.2	29986	10.15.1.7	443	TCP	54	29986 → 443 [ACK] Seq=195 Ack=2921 Win=4204800 Len=0
3197	2024-07-14 23:28:55.686118	10.15.1.7	443	10.15.1.2	29986	TLSv1.2	547	Server Key Exchange, Server Hello Done
3198	2024-07-14 23:28:55.696856	10.15.1.2	29986	10.15.1.7	443	TCP	54	29986 → 443 [ACK] Seq=195 Ack=3414 Win=4204288 Len=0
3199	2024-07-14 23:28:55.702443	10.15.1.2	29986	10.15.1.7	443	TLSv1.2	147	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
3200	2024-07-14 23:28:55.702991	10.15.1.7	443	10.15.1.2	29986	TLSv1.2	312	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
3207	2024-07-14 23:28:55.712838	10.15.1.2	29986	10.15.1.7	443	TCP	54	29986 → 443 [ACK] Seq=288 Ack=3672 Win=4204032 Len=0

Voorbeeld van een TLS-handdruk in Wireshark

Eerst, verzendt browser een pakket ClientHello met de lijst van algoritmen het steunt:

eth0\_diagnostic\_logging\_tcpdump00\_exp-c1\_2024-07-15\_03\_54\_39.pcap

No.	Time	Source	Src port	Destination	Dst port	Protocol	Length	Info
270	2024-07-14 21:54:39.347430	10.15.1.2	26105	10.15.1.7	443	TCP	66	26105 → 443 [SYN, ECE, CWR] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
271	2024-07-14 21:54:39.347496	10.15.1.7	443	10.15.1.2	26105	TCP	66	443 → 26105 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
272	2024-07-14 21:54:39.347736	10.15.1.2	26105	10.15.1.7	443	TCP	60	26105 → 443 [ACK] Seq=1 Ack=1 Win=4204800 Len=0
273	2024-07-14 21:54:39.348471	10.15.1.2	26105	10.15.1.7	443	TCP	1514	26105 → 443 [ACK] Seq=1 Ack=1 Win=4204800 Len=0
274	2024-07-14 21:54:39.348508	10.15.1.7	443	10.15.1.2	26105	TCP	54	443 → 26105 [ACK] Seq=1 Ack=1 Win=4204800 Len=0
275	2024-07-14 21:54:39.348533	10.15.1.2	26105	10.15.1.7	443	TLSv1.3	724	Client Hello
276	2024-07-14 21:54:39.348544	10.15.1.7	443	10.15.1.2	26105	TCP	54	443 → 26105 [ACK] Seq=1 Ack=1 Win=4204800 Len=0

> Frame 275: 724 bytes on wire (5792 bits), 724 bytes captured (5792 bits)

> Ethernet II, Src: VMware\_b3:fe:d6 (00:50:56:b3:fe:d6), Dst: VMware\_b3:5c:7a (00:50:56:b3:5c:7a)

> Internet Protocol Version 4, Src: 10.15.1.2, Dst: 10.15.1.7

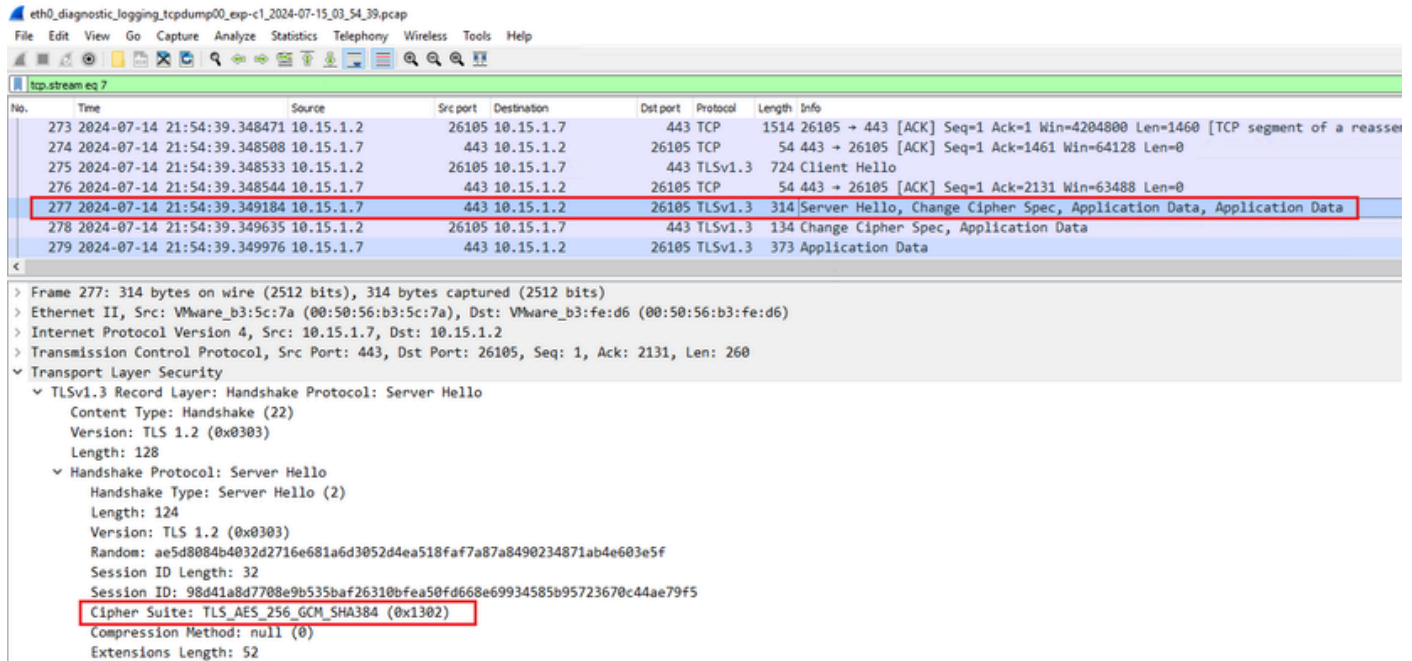
> Transmission Control Protocol, Src Port: 26105, Dst Port: 443, Seq: 1461, Ack: 1, Len: 670

> [2 Reassembled TCP Segments (2130 bytes): #273(1460), #275(670)]

Transport Layer Security

- TLV1.3 Record Layer: Handshake Protocol: Client Hello
  - Content Type: Handshake (22)
  - Version: TLS 1.0 (0x0301)
  - Length: 2125
  - Handshake Protocol: Client Hello
    - Handshake Type: Client Hello (1)
    - Length: 2121
    - Version: TLS 1.2 (0x0303)
    - Random: 7a61ba6edc3ff95c4b0672c7f1de5bf4542ced1f5eaa9147bef1cf2e54d83a50
    - Session ID Length: 32
    - Session ID: 98d41a8d7708e9b535baf26310bfea50fd668e69934585b95723670c44ae79f5
    - Cipher Suites Length: 32
    - Cipher Suites (16 suites)
      - Cipher Suite: Reserved (GREASE) (0xaeaa)
      - Cipher Suite: TLS\_AES\_128\_GCM\_SHA256 (0x1301)
      - Cipher Suite: TLS\_AES\_256\_GCM\_SHA384 (0x1302)
      - Cipher Suite: TLS\_CHACHA20\_POLY1305\_SHA256 (0x1303)
      - Cipher Suite: TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02b)
      - Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0xc02f)
      - Cipher Suite: TLS\_ECDHE\_ECDSA\_WITH\_AES\_256\_GCM\_SHA384 (0xc02c)
      - Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384 (0xc030)
      - Cipher Suite: TLS\_ECDHE\_ECDSA\_WITH\_CHACHA20\_POLY1305\_SHA256 (0xc0a9)
      - Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_CHACHA20\_POLY1305\_SHA256 (0xc0ab)
      - Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_128\_CBC\_SHA (0xc013)
      - Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_256\_CBC\_SHA (0xc014)
      - Cipher Suite: TLS\_RSA\_WITH\_AES\_128\_GCM\_SHA256 (0x009c)
      - Cipher Suite: TLS\_RSA\_WITH\_AES\_256\_GCM\_SHA384 (0x009d)
      - Cipher Suite: TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA (0x002f)
      - Cipher Suite: TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA (0x0035)
    - Compression Methods Length: 1

Expressway controleert zijn algoritme-string die geconfigureerd is voor het HTTPS-protocol en vindt een algoritme dat zowel door hem als door de client wordt ondersteund. In dit voorbeeld wordt het ECDHE-RSA-AES256-GCM-SHA384-algoritme geselecteerd. Expressway reageert met het ServerHello-pakket dat het geselecteerde algoritme aangeeft:



## Configureren

De OpenSSL-indeling voor algoritmen bevat verschillende speciale tekens om bewerkingen op de tekenreeks uit te voeren, zoals het verwijderen van een specifiek algoritme of een groep algoritmen die een gemeenschappelijke component delen. Aangezien het doel van deze aanpassingen meestal het verwijderen van algoritmen is, worden in deze voorbeelden de volgende tekens gebruikt:

- Het - teken, dat wordt gebruikt om algoritmen uit de lijst te verwijderen. Sommige of alle verwijderde algoritmen kunnen weer worden toegestaan door opties die later in de string verschijnen.
- Het ! teken, ook gebruikt om algoritmen uit de lijst te verwijderen. Wanneer het gebruiken van het, kunnen de verwijderde algoritmen niet opnieuw door een andere opties worden toegestaan die later in het koord verschijnen.
- Het : teken, dat fungeert als scheidingstekens tussen items in de lijst.

Beiden kunnen worden gebruikt om een algoritme van de string te verwijderen, maar ! wordt de voorkeur. Bekijk de [OpenSSL-coderingsmanpage](#) voor een volledige lijst met speciale tekens.



Opmerking: De OpenSSL-site verklaart dat bij gebruik van het !-teken "de verwijderde algoritmen nooit meer in de lijst kunnen verschijnen, zelfs als ze expliciet worden vermeld". Dit betekent niet dat de algoritmen permanent uit het systeem worden verwijderd, het verwijst naar de reikwijdte van de interpretatie van de algoritme string.

---

## Een specifiek algoritme uitschakelen

Als u een specifiek algoritme wilt uitschakelen, voegt u aan de standaardtekenreeks het : scheidingsteken, het ! of -teken en de naam van het algoritme die moet worden uitgeschakeld toe. De algoritmennaam moet voldoen aan de OpenSSL-naamgevingsindeling die beschikbaar is in het [OpenSSL-coderingsmanagement](#). Als u bijvoorbeeld het AES128-SHA-algoritme voor SIP-verbindingen moet uitschakelen, moet u een algoritme als volgt configureren:

```
<#root>
```

```
EECDH:EDH:HIGH:-AES256+SHA:!MEDIUM:!LOW:!3DES:!MD5:!PSK:!eNULL:!aNULL:!aDH
```

```
:!AES128-SHA
```

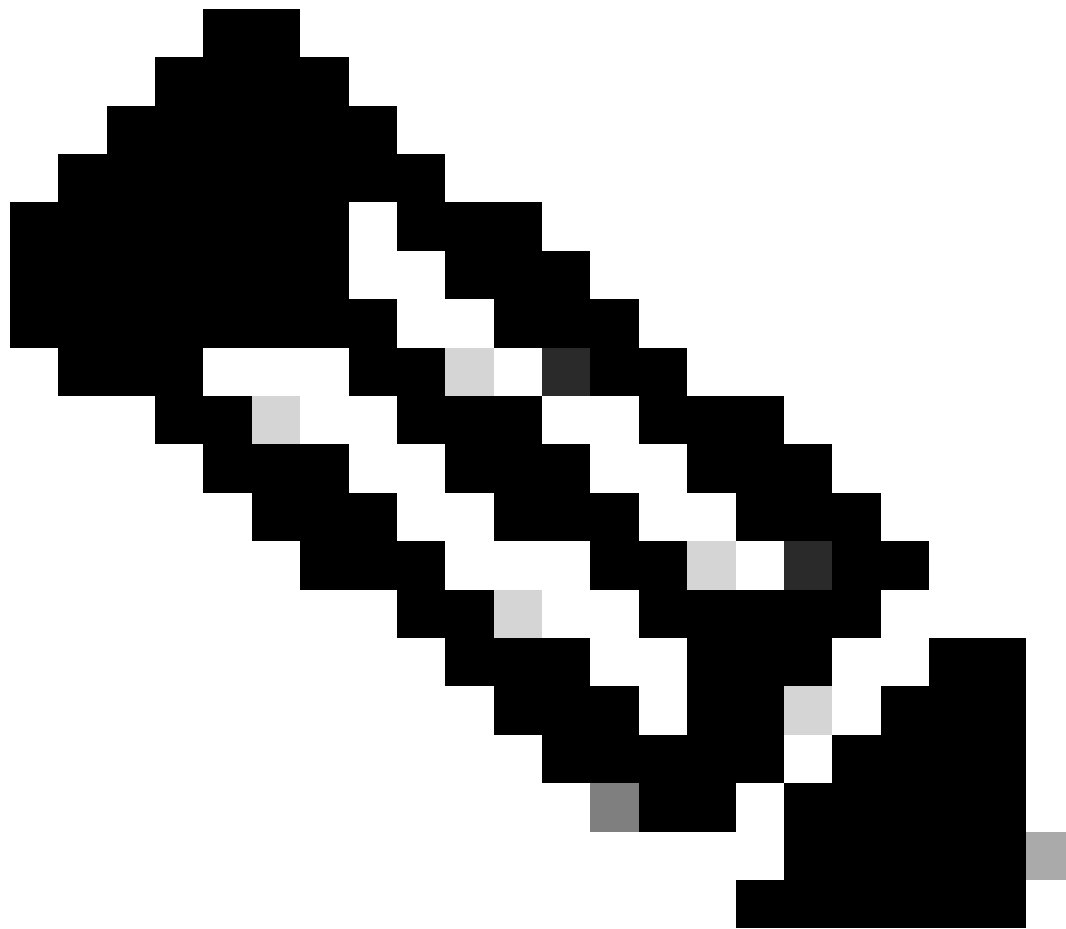


Ga vervolgens naar de pagina Expressway-webbeheer, navigeer naar Onderhoud > Beveiliging > Cijfers, wijs de aangepaste tekenreeks toe aan de vereiste protocollen en klik op Opslaan. De nieuwe configuratie kan alleen worden toegepast als het systeem opnieuw is opgestart. In dit voorbeeld wordt de aangepaste tekenreeks toegewezen aan het SIP-protocol onder SIP TLS-algoritmen:

The screenshot displays the 'Ciphers' configuration page in the Expressway web management interface. The page has a navigation bar at the top with links for Status, System, Configuration, Applications, Users, and Maintenance. The main content area is titled 'Ciphers' and contains a list of configuration items, each with a text input field and a dropdown menu for the minimum TLS version. The 'SIP TLS ciphers' field is highlighted with a red box and contains the value 'IMEDIUM:LOW:3DES:MD5:IPSK:!aNULL:!aNULL:!aDH:!AES128-SHA'. Below the configuration area, a 'Save' button is also highlighted with a red box.

Configuration Item	Value	Minimum TLS Version
HTTPS ciphers	EECDH:EDH:HIGH:-AES256+SHA:IMEDIUM:LOW:3DES:MD5:IPSK:h	TLS v1.2
HTTPS minimum TLS version		TLS v1.2
LDAP TLS Ciphers	EECDH:EDH:HIGH:-AES256+SHA:IMEDIUM:LOW:3DES:MD5:IPSK:h	TLS v1.2
LDAP minimum TLS version		TLS v1.2
Reverse proxy TLS ciphers	EECDH:EDH:HIGH:-AES256+SHA:IMEDIUM:LOW:3DES:MD5:IPSK:h	TLS v1.2
Reverse proxy minimum TLS version		TLS v1.2
<b>SIP TLS ciphers</b>	<b>IMEDIUM:LOW:3DES:MD5:IPSK:!aNULL:!aNULL:!aDH:!AES128-SHA</b>	TLS v1.2
SIP minimum TLS version		TLS v1.2
SMTP TLS Ciphers	EECDH:EDH:HIGH:-AES256+SHA:IMEDIUM:LOW:3DES:MD5:IPSK:h	TLS v1.2
SMTP minimum TLS version		TLS v1.2
TMS Provisioning Ciphers	EECDH:EDH:HIGH:-AES256+SHA:IMEDIUM:LOW:3DES:MD5:IPSK:h	TLS v1.2
TMS Provisioning minimum TLS version		TLS v1.2
UC server discovery TLS ciphers	EECDH:EDH:HIGH:-AES256+SHA:IMEDIUM:LOW:3DES:MD5:IPSK:h	TLS v1.2
UC server discovery minimum TLS version		TLS v1.2
XMPP TLS ciphers	EECDH:EDH:HIGH:-AES256+SHA:IMEDIUM:LOW:3DES:MD5:IPSK:h	TLS v1.2
XMPP minimum TLS version		TLS v1.2

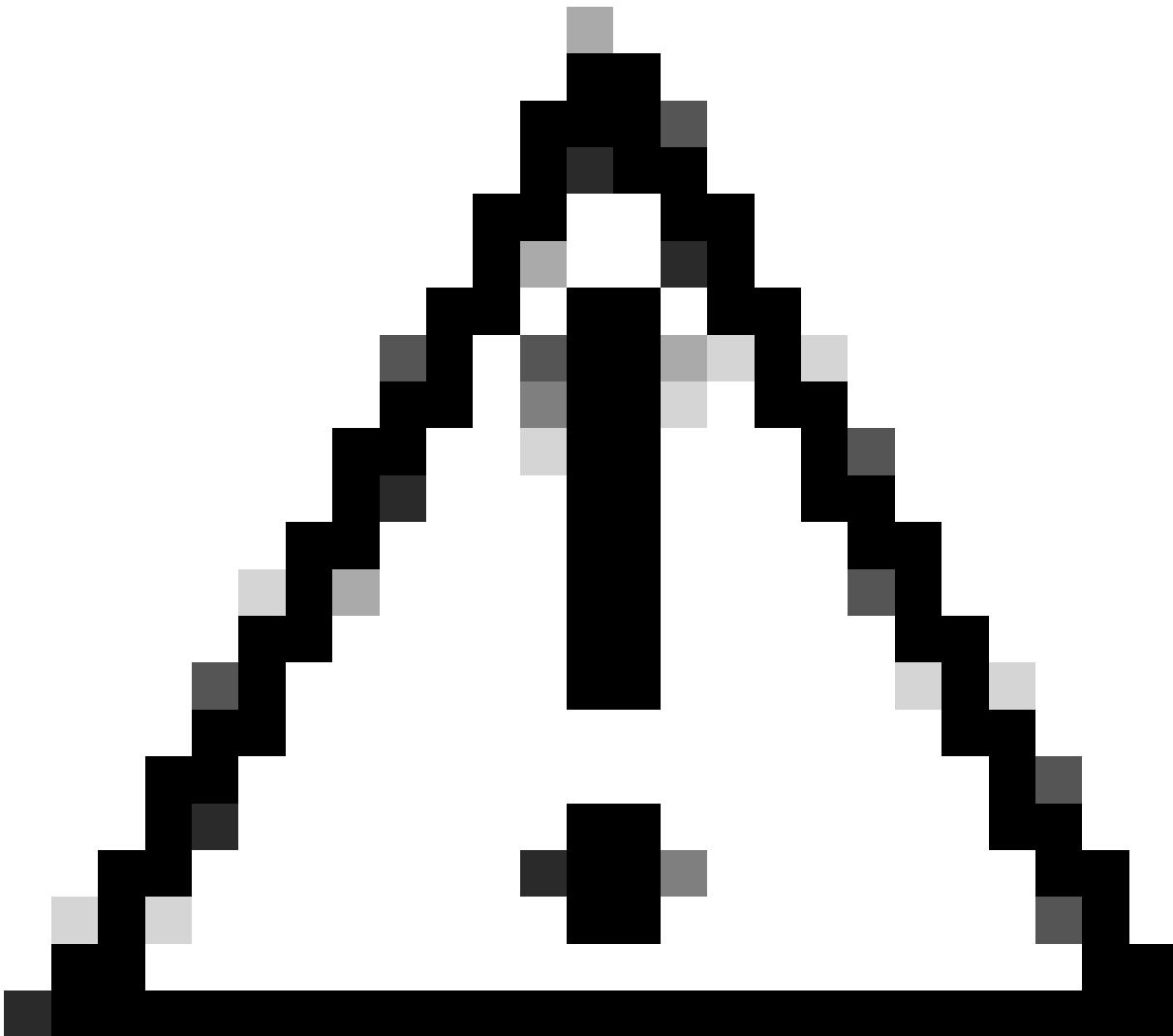
De pagina van de Instellingen van het algoritme op het Portaal van Admin van het Web Expressway



Opmerking: in het geval van een Expressway-cluster, breng de wijzigingen alleen aan op de primaire server. De nieuwe configuratie wordt gerepliceerd naar de rest van de clusterleden.

---





Waarschuwing: gebruik de aanbevolen volgorde voor het opnieuw opstarten van clusters die in de [implementatiehandleiding voor Cisco Expressway Cluster Creation and Maintenance wordt](#) geboden. Begin door de primaire server opnieuw te starten, wacht tot hij via de web interface toegankelijk is en doe hetzelfde met elke peer volgens de lijst die is ingesteld onder System > Clustering.

---

## Een groep algoritmen uitschakelen met een gemeenschappelijk algoritme

Als u een groep algoritmes wilt uitschakelen met behulp van een gemeenschappelijk algoritme, voegt u aan de standaardstring het : separator, het ! of -teken, en de algoritmenaam die moet worden uitgeschakeld toe. De ondersteunde algoritmenamen zijn beschikbaar in de [OpenSSL-coderingsmanpage](#). Als u bijvoorbeeld alle algoritmen die DHE gebruiken moet uitschakelen, moet u een algoritme als deze configureren:

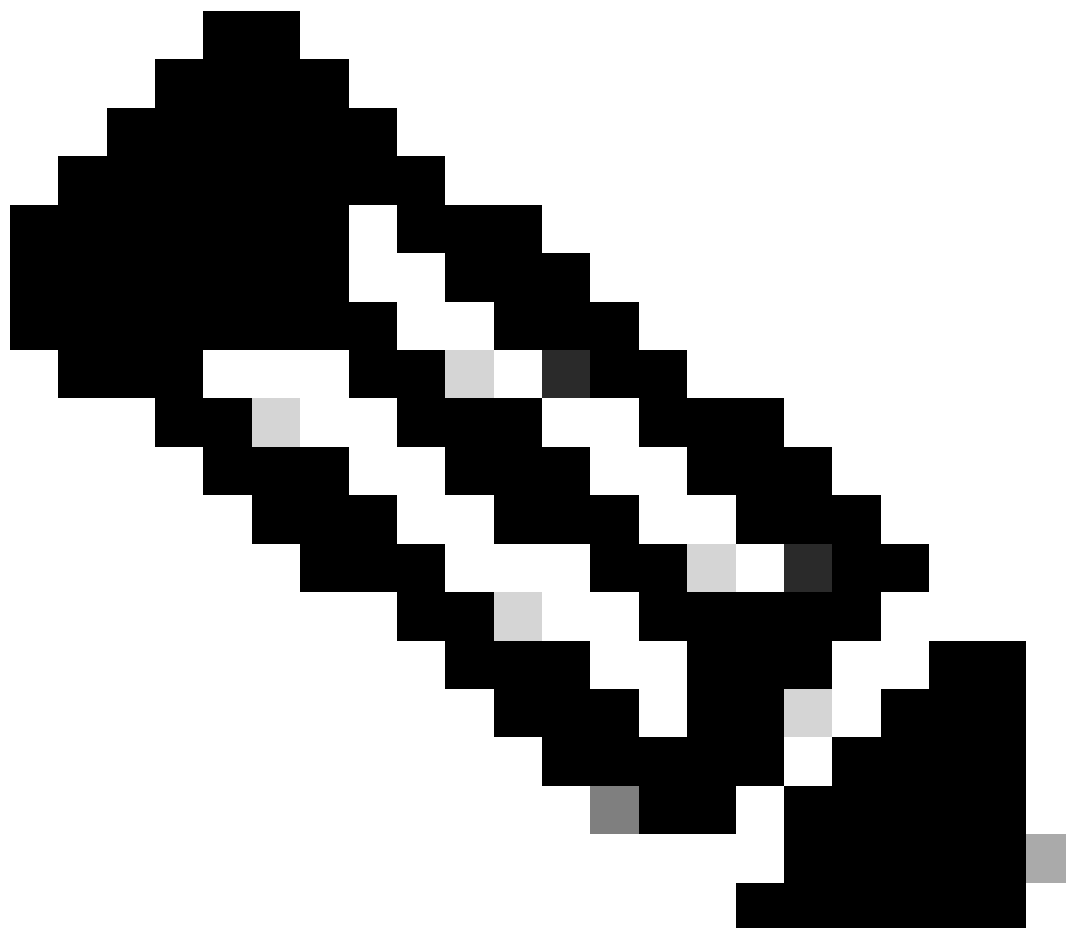
```
<#root>
```

```
EECDH:EDH:HIGH:-AES256+SHA:!MEDIUM:!LOW:!3DES:!MD5:!PSK:!eNULL:!aNULL:!aDH
```

: !DHE

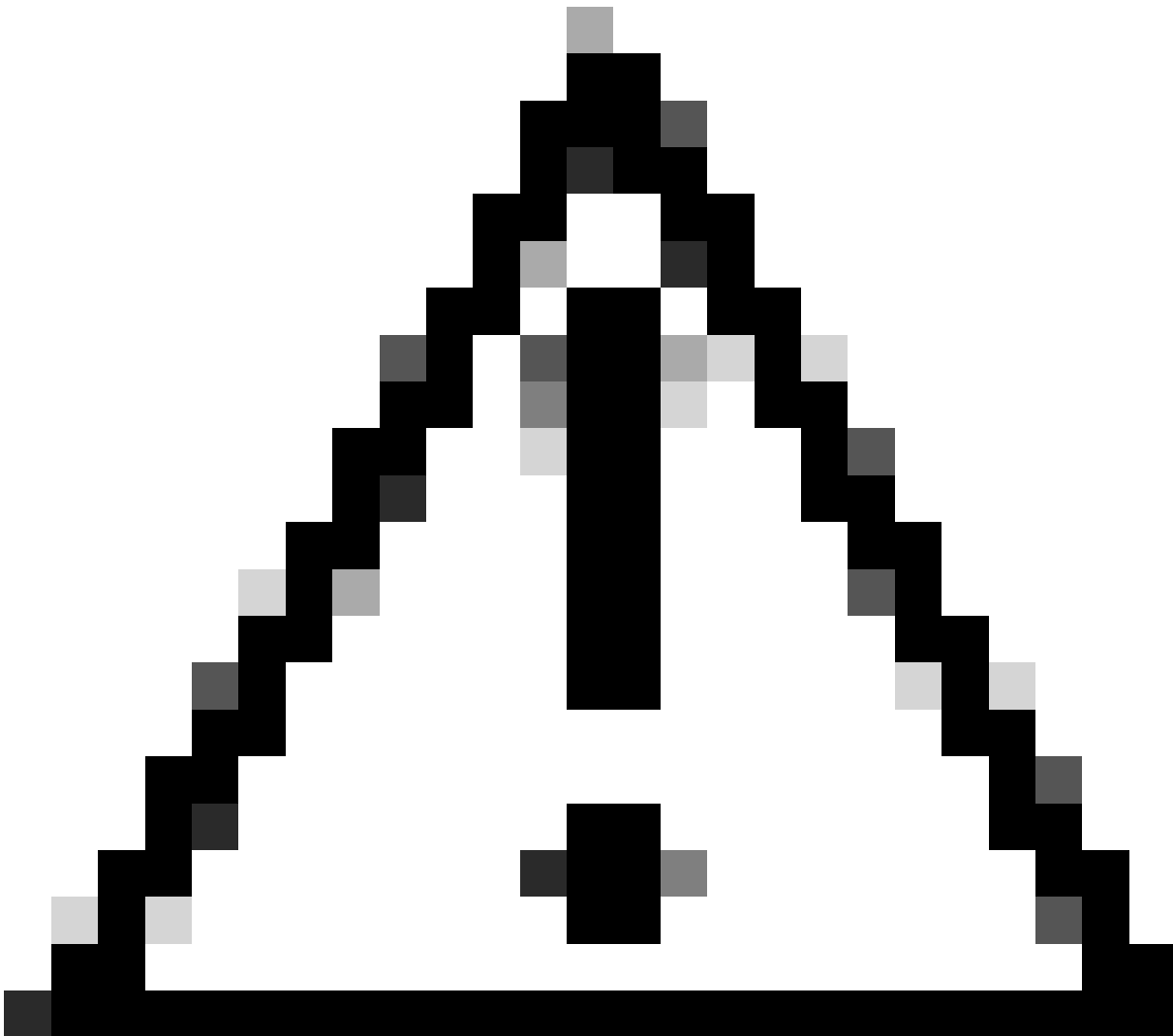
Navigeer naar de pagina Expressway web admin, navigeer naar Onderhoud > Beveiliging > Cijfers, wijs de aangepaste tekenreeks toe aan het/de vereiste protocol(n) en klik op Opslaan. De nieuwe configuratie kan alleen worden toegepast als het systeem opnieuw is opgestart.

---



Opmerking: in het geval van een Expressway-cluster, breng de wijzigingen alleen aan op de primaire server. De nieuwe configuratie wordt gerepliceerd naar de rest van de clusterleden.

---



Waarschuwing: gebruik de aanbevolen volgorde voor het opnieuw opstarten van clusters die in de [implementatiehandleiding voor Cisco Expressway Cluster Creation and Maintenance wordt](#) geboden. Begin door de primaire server opnieuw te starten, wacht tot hij via de web interface toegankelijk is en doe hetzelfde met elke peer volgens de lijst die is ingesteld onder System > Clustering.

---

## Verifiëren

Inspecteer de lijst met cifers die zijn toegestaan door de reeks coderingen

U kunt de aangepaste algoritme-string controleren met behulp van de opdracht `openssl -V "<algoritme string>"`. Bekijk de output om te bevestigen dat de ongewenste algoritmen niet meer vermeld na de veranderingen zijn. In dit voorbeeld, de `EECDH:EDH:HIGH:-AES256+SHA:!MEDIUM:!LOW:!3DES:!MD5:!PSK:!NULL:!ANULL:!aDH:!DHE-coderingsstring` wordt geïnspecteerd. De opdrachtoutput bevestigt dat de string geen algoritme toestaat die gebruik maakt van het DHE algoritme:

```
<#root>
```

```
~ # openssl ciphers -V "EECDH:EDH:HIGH:-AES256+SHA:!MEDIUM:!LOW:!3DES:!MD5:!PSK:!eNULL:!aNULL:!aDH
:!DHE
"
0x13,0x02 - TLS_AES_256_GCM_SHA384 TLSv1.3 Kx=any Au=any Enc=AESGCM(256) Mac=AEAD
0x13,0x03 - TLS_CHACHA20_POLY1305_SHA256 TLSv1.3 Kx=any Au=any Enc=CHACHA20/POLY1305(256) Mac=AEAD
0x13,0x01 - TLS_AES_128_GCM_SHA256 TLSv1.3 Kx=any Au=any Enc=AESGCM(128) Mac=AEAD
0xC0,0x2C - ECDHE-ECDSA-AES256-GCM-SHA384 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESGCM(256) Mac=AEAD
0xC0,0x30 - ECDHE-RSA-AES256-GCM-SHA384 TLSv1.2 Kx=ECDH Au=RSA Enc=AESGCM(256) Mac=AEAD
0xCC,0xA9 - ECDHE-ECDSA-CHACHA20-POLY1305 TLSv1.2 Kx=ECDH Au=ECDSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
0xCC,0xA8 - ECDHE-RSA-CHACHA20-POLY1305 TLSv1.2 Kx=ECDH Au=RSA Enc=CHACHA20/POLY1305(256) Mac=AEAD
0xC0,0xAD - ECDHE-ECDSA-AES256-CCM TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESCCM(256) Mac=AEAD
0xC0,0x2B - ECDHE-ECDSA-AES128-GCM-SHA256 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESGCM(128) Mac=AEAD
0xC0,0x2F - ECDHE-RSA-AES128-GCM-SHA256 TLSv1.2 Kx=ECDH Au=RSA Enc=AESGCM(128) Mac=AEAD
0xC0,0xAC - ECDHE-ECDSA-AES128-CCM TLSv1.2 Kx=ECDH Au=ECDSA Enc=AESCCM(128) Mac=AEAD
0xC0,0x24 - ECDHE-ECDSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AES(256) Mac=SHA384
0xC0,0x28 - ECDHE-RSA-AES256-SHA384 TLSv1.2 Kx=ECDH Au=RSA Enc=AES(256) Mac=SHA384
0xC0,0x23 - ECDHE-ECDSA-AES128-SHA256 TLSv1.2 Kx=ECDH Au=ECDSA Enc=AES(128) Mac=SHA256
0xC0,0x27 - ECDHE-RSA-AES128-SHA256 TLSv1.2 Kx=ECDH Au=RSA Enc=AES(128) Mac=SHA256
0xC0,0x09 - ECDHE-ECDSA-AES128-SHA TLSv1 Kx=ECDH Au=ECDSA Enc=AES(128) Mac=SHA1
0xC0,0x13 - ECDHE-RSA-AES128-SHA TLSv1 Kx=ECDH Au=RSA Enc=AES(128) Mac=SHA1
0x00,0x9D - AES256-GCM-SHA384 TLSv1.2 Kx=RSA Au=RSA Enc=AESGCM(256) Mac=AEAD
0xC0,0x9D - AES256-CCM TLSv1.2 Kx=RSA Au=RSA Enc=AESCCM(256) Mac=AEAD
0x00,0x9C - AES128-GCM-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AESGCM(128) Mac=AEAD
0xC0,0x9C - AES128-CCM TLSv1.2 Kx=RSA Au=RSA Enc=AESCCM(128) Mac=AEAD
0x00,0x3D - AES256-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AES(256) Mac=SHA256
0x00,0x3C - AES128-SHA256 TLSv1.2 Kx=RSA Au=RSA Enc=AES(128) Mac=SHA256
0x00,0x2F - AES128-SHA SSLv3 Kx=RSA Au=RSA Enc=AES(128) Mac=SHA1
~ #
```

## Test een TLS-verbinding door te onderhandelen over een uitgeschakeld algoritme

U kunt de opdracht `openssl s_client` gebruiken om te verifiëren dat een verbindingsooging met een uitgeschakeld algoritme wordt geweigerd. Gebruik de optie `-connect` om uw Expressway-adres en -poort op te geven en gebruik de optie `-algoritme` om het enkele algoritme op te geven waarover de client tijdens de TLS-handdruk moet onderhandelen:

```
openssl s_client -connect <adres>:<port> -algoritme <algoritme> -no_tls1_3
```

In dit voorbeeld wordt een TLS-verbinding naar Expressway geprobeerd vanaf een Windows-pc met geïnstalleerde `openssl`. De PC onderhandelt als client alleen over het ongewenste DHE-RSA-AES256-CCM algoritme, dat gebruik maakt van het DHE algoritme:

```
<#root>
```

```
C:\Users\Administrator>
```

```
openssl s_client -connect exp.example.com:443 -cipher DHE-RSA-AES256-CCM -no_tls1_3
```

```
Connecting to 10.15.1.7
```

```
CONNECTED(00000154)
```

```
D0130000:error:0A000410:SSL routines:ssl3_read_bytes:
```

ssl/tls alert handshake failure

...\ssl\record\rec\_layer\_s3.c:865:

SSL alert number 40

---

no peer certificate available

---

No client certificate CA names sent

---

SSL handshake has read 7 bytes and written 118 bytes

Verification: OK

---

New, (NONE), Cipher is (NONE)

Secure Renegotiation IS NOT supported

No ALPN negotiated

SSL-Session:

Protocol : TLSv1.2

Cipher : 0000

Session-ID:

Session-ID-ctx:

Master-Key:

PSK identity: None

PSK identity hint: None

SRP username: None

Start Time: 1721019437

Timeout : 7200 (sec)

Verify return code: 0 (ok)

Extended master secret: no

---

C:\Users\Administrator>

De opdrachtoutput laat zien dat de verbindingsooging mislukt met de foutmelding "ssl/tls alert handshake failure:...\ssl\record\rec\_layer\_s3.c:865:SSL alert number 40", omdat de Expressway is geconfigureerd om de EEC DH:EDH:HIGH:-

AES256+SHA:!MEDIUM:!LOW:!3DES:!MD5:!PSK:!NULL:!ANULL:!aDH: HTTDHE-algoritme voor PS-verbindingen, waardoor algoritmes worden uitgeschakeld die het DHE-algoritme gebruiken.



Opmerking: om te zorgen dat tests met de opdracht `openssl s_client` werken zoals uitgelegd, moet de optie `-no_tls1_3` worden doorgegeven aan de opdracht. Indien niet opgenomen, voegt de client automatisch TLS 1.3-algoritmen in het ClientHello-pakket in:

---

Urgent Pointer: 0

- > [Timestamps]
- > [SEQ/ACK analysis]
- TCP payload (247 bytes)
- Transport Layer Security
  - TLSv1.3 Record Layer: Handshake Protocol: Client Hello
    - Content Type: Handshake (22)
    - Version: TLS 1.0 (0x0301)
    - Length: 242
    - Handshake Protocol: Client Hello
      - Handshake Type: Client Hello (1)
      - Length: 238
      - Version: TLS 1.2 (0x0303)
      - Random: 19ec4e8994cc334599cf889d4e45a812029589923c4cfcf2cef6b6fc47ec2840
      - Session ID Length: 32
      - Session ID: e0d17cb402229aa46cab70b6a637ce38d9b5a228c7b360cb43f49086ce88d5df
      - Cipher Suites Length: 10
      - Cipher Suites (5 suites)
        - Cipher Suite: TLS\_AES\_256\_GCM\_SHA384 (0x1302)
        - Cipher Suite: TLS\_CHACHA20\_POLY1305\_SHA256 (0x1303)
        - Cipher Suite: TLS\_AES\_128\_GCM\_SHA256 (0x1301)
        - Cipher Suite: TLS\_DHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384 (0xc030)
        - Cipher Suite: TLS\_EMPTY\_RENEGOTIATION\_INFO\_SCSV (0x00ff)
      - Compression Methods Length: 1

Ciphers automatically inserted by the openssl s\_client command

Cipher passed with the -cipher option

ClientHello-pakket met automatisch toegevoegde coderingen

Als de doel Expressway ondersteunt die algoritmen, kan een van hen worden gekozen in plaats van het specifieke algoritme dat u moet testen. De verbinding is succesvol, wat u ertoe kan brengen te geloven dat een verbinding mogelijk was door het gehandicapte algoritme te gebruiken dat tot het bevel met de optie - algoritme wordt overgegaan.

## Inspecteer een pakketvastlegging van een TLS-handdruk met een uitgeschakeld coderingssysteem

U kunt een pakketopname, van het testapparaat of van de snelweg verzamelen terwijl u een verbindingstest uitvoert met behulp van een van de uitgeschakelde algoritmen. U kunt het dan inspecteren met Wireshark om de handshake gebeurtenissen verder te analyseren.

Zoek de ClientHello verzonden door het testapparaat. Bevestig dat alleen wordt onderhandeld over het ongewenste testalgoritme, in dit voorbeeld een algoritme met behulp van DHE:



The image shows a Wireshark capture of a network packet. The top pane displays a list of packets, with packet 327 highlighted in red. This packet is a Client Hello from source 10.15.1.2 to destination 10.15.1.7. The bottom pane shows the detailed structure of this packet:

- Acknowledgment number (raw): 3235581935
- 0101 .... = Header Length: 20 bytes (5)
- Flags: 0x018 (PSH, ACK)
- Window: 16425
- [Calculated window size: 4204800]
- [Window size scaling factor: 256]
- Checksum: 0x16b7 [unverified]
- [Checksum Status: Unverified]
- Urgent Pointer: 0
- [Timestamps]
- [SEQ/ACK analysis]
- TCP payload (118 bytes)
- Transport Layer Security
  - TLSv1.2 Record Layer: Handshake Protocol: Client Hello
    - Content Type: Handshake (22)
    - Version: TLS 1.0 (0x0301)
    - Length: 113
    - Handshake Protocol: Client Hello
      - Handshake Type: Client Hello (1)
      - Length: 109
      - Version: TLS 1.2 (0x0303)
      - Random: e5cb04a72ae567a0963c5a4a5901db3720fabc5980aa2ef5a5ecc099254c1bf8
      - Session ID Length: 0
      - Cipher Suites Length: 4
      - Cipher Suites (2 suites)
        - Cipher Suite: TLS\_DHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384 (0xc09f)
        - Cipher Suite: TLS\_EMPTY\_RENEGOTIATION\_INFO\_SCSV (0x00ff)
      - Compression Methods Length: 1

Voorbeeld van een ClientHello-pakket in Wireshark

:

Bevestig dat Expressway reageert met een fataal TLS-waarschuingspakket en de verbinding weigert. In dit voorbeeld, aangezien Expressway geen DHE-algoritmen ondersteunt per de ingestelde algoritme voor het HTTPS-protocol, reageert het met een fataal TLS-waarschuingspakket met foutcode 40.

Wireshark interface showing a network capture on 'Ethernet0'. The packet list pane displays several packets, with packet 329 highlighted in red. The packet details pane for packet 329 shows the following information:

No.	Time	Source	Src port	Destination	Dst port	Protocol	Length	Info
324	2024-07-14 23:00:32.459025	10.15.1.2	28872	10.15.1.7	443	TCP	66	28872 → 443 [SYN, ECE, CWR] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM
325	2024-07-14 23:00:32.459666	10.15.1.7	443	10.15.1.2	28872	TCP	66	443 → 28872 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
326	2024-07-14 23:00:32.459760	10.15.1.2	28872	10.15.1.7	443	TCP	54	28872 → 443 [ACK] Seq=1 Ack=1 Win=4204800 Len=0
327	2024-07-14 23:00:32.460733	10.15.1.2	28872	10.15.1.7	443	TLSv1.2	172	Client Hello
328	2024-07-14 23:00:32.461070	10.15.1.7	443	10.15.1.2	28872	TCP	60	443 → 28872 [ACK] Seq=1 Ack=119 Win=64128 Len=0
329	2024-07-14 23:00:32.461855	10.15.1.7	443	10.15.1.2	28872	TLSv1.2	61	Alert (Level: Fatal, Description: Handshake Failure)
330	2024-07-14 23:00:32.461855	10.15.1.7	443	10.15.1.2	28872	TCP	60	443 → 28872 [FIN, ACK] Seq=8 Ack=119 Win=64128 Len=0

The packet details pane for packet 329 shows the following information:

- Frame 329: 61 bytes on wire (488 bits), 61 bytes captured (488 bits) on interface \Device\NPF\_{122607A1-10A8-47F6-9069-936EB0CAAE1C}, id 0
- Ethernet II, Src: VMware\_b3:5c:7a (00:50:56:b3:5c:7a), Dst: VMware\_b3:fe:d6 (00:50:56:b3:fe:d6)
- Internet Protocol Version 4, Src: 10.15.1.7, Dst: 10.15.1.2
- Transmission Control Protocol, Src Port: 443, Dst Port: 28872, Seq: 1, Ack: 119, Len: 7
  - Source Port: 443
  - Destination Port: 28872
  - [Stream index: 2]
  - [Conversation completeness: Complete, WITH\_DATA (31)]
  - [TCP Segment Len: 7]
  - Sequence Number: 1 (relative sequence number)
  - Sequence Number (raw): 3235581935
  - [Next Sequence Number: 8 (relative sequence number)]
  - Acknowledgment Number: 119 (relative ack number)
  - Acknowledgment number (raw): 810929090
  - 0101 .... = Header Length: 20 bytes (5)
  - Flags: 0x018 (PSH, ACK)
  - Window: 501
  - [Calculated window size: 64128]
  - [Window size scaling factor: 128]
  - Checksum: 0x163f [unverified]
  - [Checksum Status: Unverified]
  - Urgent Pointer: 0
  - [Timestamps]
  - [SEQ/ACK analysis]
  - TCP payload (7 bytes)
- Transport Layer Security
  - TLSv1.2 Record Layer: Alert (Level: Fatal, Description: Handshake Failure)
    - Content Type: Alert (21)
    - Version: TLS 1.2 (0x0303)
    - Length: 2
    - Alert Message
      - Level: Fatal (2)
      - Description: Handshake Failure (40)

Een TLS-fataal waarschuwingspakket in Wireshark

## Gerelateerde informatie

- [OpenSSL-coderingsbeheer](#)
- [Cisco Expressway Administrator Guide \(X15.0\) - Hoofdstuk: Beveiligingsbeheer - Minimale TLS-versie configureren en aangepaste versies installeren](#)

## 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.