

# Utilize EEM to Automate Secure Emails to User

## Contents

---

[Introduction](#)

[Use Case](#)

[Background](#)

[Gmail Account Setup](#)

[Base EEM Configuration](#)

[Issue Seen with Only Default Certificates Installed](#)

[Certificates for Securing SMTP](#)

[An Easier Way to Find the Certificates](#)

[Testing EEM with Secure SMTP Again](#)

[Other Caveats and Considerations](#)

[Usernames with @ Symbols](#)

[Conclusion](#)

---

## Introduction

This document describes the process needed to utilize the "mail server" action in the Embedded Event Manager (EEM) within Cisco IOS® XE to send secure emails to a Simple Mail Transfer Protocol (SMTP) server using Transport Layer Security (TLS) on port 587.

There are many caveats that you can encounter during this process, which is why this article was written to document the steps needed to accomplish this.

## Use Case

Many customers see value in receiving an email notification automatically after a certain event occurs. The EEM subsystem is a powerful tool for network event detection and onboard automation, and it can provide an efficient way to automate email notifications on a Cisco IOS XE device. For example, you might want to monitor an IPSLA track, and in response to a syslog indicating a state change, take some kind of action and alert the network administrators of the event via email. This "email notification" idea could be applied to many other scenarios as a means of bringing attention to any particular event that you want to highlight.

## Background

PEM stands for "Privacy Enhanced Mail", and it is a format often used to represent certificates and keys. This is the certificate format that Cisco IOS XE devices utilize. Secure applications (like HTTPS or secure SMTP) often have a "Stacked PEM", where there are multiple certificates involved, including:

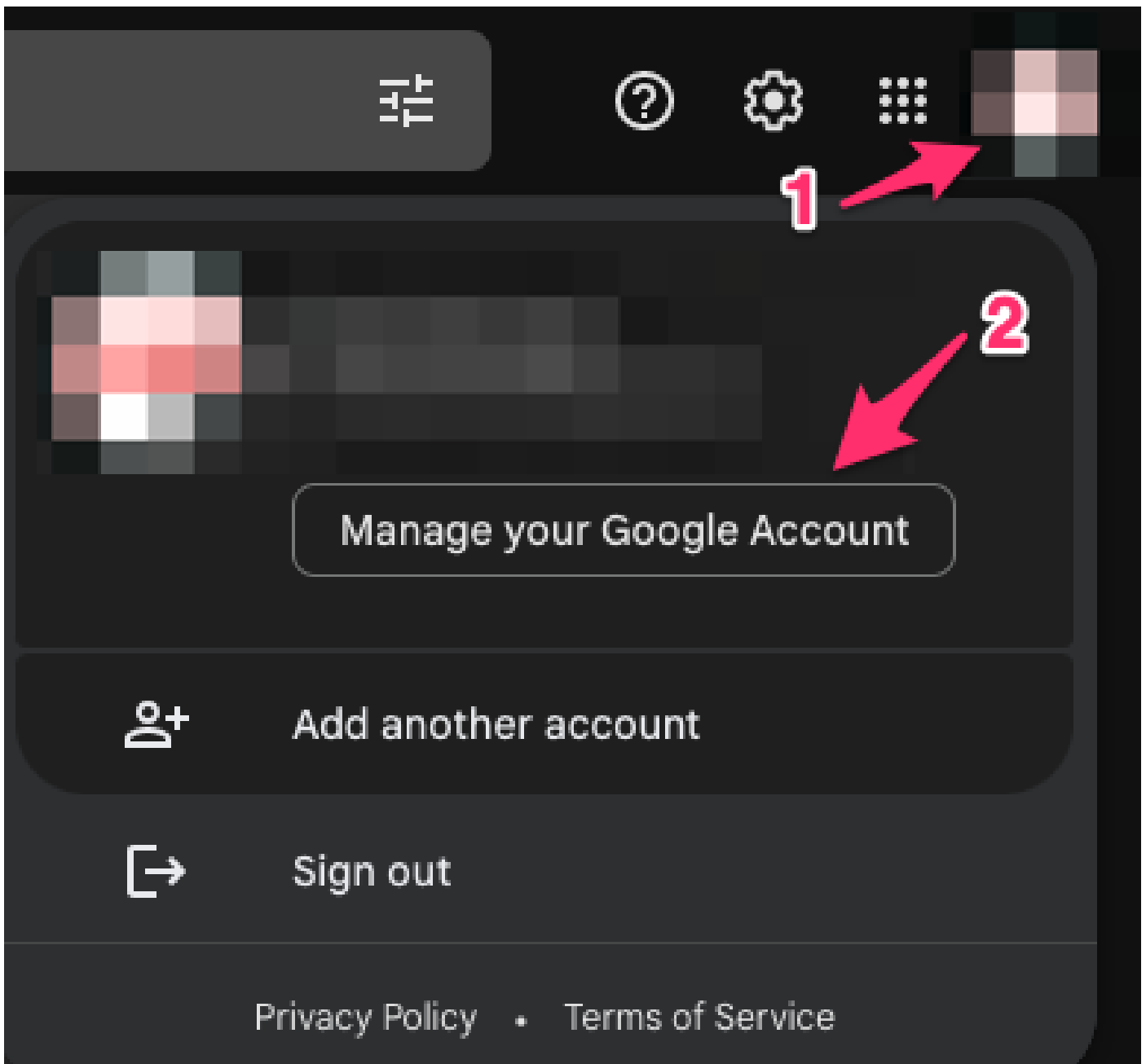
- Root Certificate
- Signing (Intermediate) Certificate
- End-user (or server) certificate

## Gmail Account Setup

Google's SMTP services will be used as an example in this article. Pre-requisites are that you have a Gmail account previously set up.

Google allows you to send emails from remote clients into Gmail. There used to be a setting in Gmail for "unsecured apps", and the application would face an error if this setting was not allowed on Google's end. That setting has been removed, and in its place is a "Secure Applications" option, which is accessible via:

mail.google.com > Click on Your Profile (#1) > Manage your Google Account (#2) > Security (#3) > How you sign in to Google > 2-Step Verification (#4)



- Home
- Personal info
- Data & privacy
- Security**
- People & sharing
- Payments & subscriptions
- About

## Security

Settings and recommendations to help you keep your account secure

### You have security tips

Security tips found in the Security Checkup



[Review security tips](#)

### Recent security activity

New sign-in on Mac

3:55 PM



[Review security activity](#)

### How you sign in to Google

Make sure you can always access your Google Account by keeping this information up to date

2-Step Verification



On since Jul 20, [blurred]



From this page, ensure that 2-Step Verification is ON.

# ← 2-Step Verification

**2-Step Verification is ON** since Jul 20, [blurred]

You can then scroll down to "App passwords" to have Gmail generate a password that can be used to sign in to your Google Account from an application that does not support 2-Step verification.

## App passwords

App Passwords aren't recommended and are unnecessary in most cases. To help keep your account secure, use "Sign in with Google" to connect apps to your Google Account.

### App passwords

None



## ← App passwords

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)

You don't have any app passwords.

Select the app and device you want to generate the app password for.

Mail



Select device

iPhone

iPad

BlackBerry

Mac

Windows Phone

Windows Computer

Other *(Custom name)*

GENERATE

## ← App passwords

---

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)

You don't have any app passwords.

Select the app and device you want to generate the app password for.


MyRouter ×

GENERATE

## ← App passwords

App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)

### Your app passwords

Name	Created	Last used	
MyRouter	4:03 PM	-	

Select the app and device you want to generate the app password for.

Select app



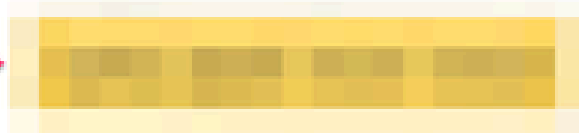
Select device



GENERATE

### Generated app password

#### Your app password for your device



#### How to use it

Go to the settings for your Google Account in the application or device you are trying to set up. Replace your password with the 16-character password shown above. Just like your normal password, this app password grants complete access to your Google Account. You won't need to remember it, so don't write it down or share it with anyone.

DONE

The 16-character application password in this screenshot was blurred out as it is tied to a personal Gmail account.

Now that you have an application password for Gmail, you can use this, along with your Gmail account name, as the e-mail server to use for forwarding the e-mail. The format to specify the server is "username:password@host".

## Base EEM Configuration

There are many ways that you can customize an EEM script to fit your exact needs, but this example is a basic EEM script to get the secure email functionality running:

```
(config)# event manager environment _email_from <username@gmail.com>
(config)# event manager environment _email_to <EMAIL@domain.com>
(config)# event manager environment _email_server <username>:<password>@smtp.gmail.com

(config)# event manager applet SendSecureEmailEEM
(config-applet)# event none
(config-applet)# action 0010 mail server "$_email_server" to "$_email_to" from "$_email_from" cc "$_
```

The configurations first create three EEM environment variables: `_email_from`, `_email_to`, and `_email_server`. Each one is defined in a variable to make configuration changes easier. You then create the `SendSecureEmailEEM` script. The triggering event here is "none" so that you can manually run the EEM script at-will using "# event manager run SendSecureEmailEEM" (rather than waiting for a specific event to trigger). Next, you have just a single "mail server" action which takes care of the email generation. The "secure tls" and "port 587" options tell the device to negotiate TLS on port 587, which the Gmail servers will be listening on.

You also need to ensure that your "from" field is valid. If you are authenticating as "Alice" but are trying to send an email from "Bob", then it will error out because Alice is spoofing someone else's email address. The "from" field needs to align with the account being used to send the email on the server.

## Issue Seen with Only Default Certificates Installed

EEM utilizes openssl to make a connection with the SMTP server. For secure communication, the server sends back a certificate to openssl running in Cisco IOSd. IOSd will then look for a trustpoint associated with that certificate.

On a Cisco IOS XE device, the certificates for the Gmail SMTP servers are not installed by default. They must be manually imported for trust to be established. Without the certificates installed, the TLS handshake will fail due to a "bad certificate".

These debugs are extremely useful for debugging any certificate issues:

```
debug event manager action mail
debug crypto pki API
debug crypto pki callbacks
debug crypto pki messages
debug crypto pki scep
debug crypto pki server
debug crypto pki transactions
debug crypto pki validation
debug ssl openssl errors
debug ssl openssl ext
```

```
debug ssl openssl msg
debug ssl openssl states
```

You can start an Embedded Packet Capture (EPC) on the router to capture any traffic to or from the email server as the EEM is triggered:

```
! Trigger the EEM:
```

```
# event manager run SendSecureEmailEEM
```

```
<SNIP>
```

```
*Mar 15 21:51:32.798: CRYPTO_PKI: (A0693) Check for identical certs
```

```
*Mar 15 21:51:32.798: CRYPTO_PKI(Cert Lookup) issuer="cn=GlobalSign Root CA,ou=Root CA,o=GlobalSign nv-
```

```
*Mar 15 21:51:32.798: CRYPTO_PKI: looking for cert in handle=7F41EE523CE0, digest=
94 40 D1 90 A0 A3 5D 47 E5 B5 31 F6 63 AD 1B 0A
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: Cert record not found for issuer serial.
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI : (A0693) Validating non-trusted cert
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: (A0693) Create a list of suitable trustpoints
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: crypto_pki_get_cert_record_by_issuer()
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: Unable to locate cert record by issuername
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: No trust point for cert issuer, looking up cert chain
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: crypto_pki_get_cert_record_by_subject()
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: (A0693) No suitable trustpoints found
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: (A0693) Removing verify context
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: destroying ca_req_context type PKI_VERIFY_CHAIN_CONTEXT,ident 32, ref
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: ca_req_context released
```

```
*Mar 15 21:51:32.799: CRYPTO_OPSSL: Certificate verification has failed
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: Rcvd request to end PKI session A0693.
```

```
*Mar 15 21:51:32.799: CRYPTO_PKI: PKI session A0693 has ended. Freeing all resources.
```

```
*Mar 15 21:51:32.800: >>> ??? [length 0005]
```

```
*Mar 15 21:51:32.800: 15 03 03 00 02
```

```
*Mar 15 21:51:32.800:
```

```
*Mar 15 21:51:32.800: >>> TLS 1.2 Alert [length 0002], fatal bad_certificate
```

```
*Mar 15 21:51:32.800: 02 2A
```

```
*Mar 15 21:51:32.800:
```

```
*Mar 15 21:51:32.800: SSL3 alert write:fatal:bad certificate
```

```
*Mar 15 21:51:32.801: P11:OpenSession slot 1 flags 6
```

```
*Mar 15 21:51:32.801: SSL_connect:error in error
```

```
*Mar 15 21:51:32.801: 0:error:1416F086:SSL routines:tls_process_server_certificate:certificate verify f
```

Ultimately, openssl cannot establish the secure TLS session with the SMTP server, so it throws a "bad certificate" error, which causes the EEM to stop running:

```
*Mar 15 21:51:32.801: %HA_EM-3-FMPD_SMTP: Error occurred when sending mail to SMTP server: username:pas
```

```
*Mar 15 21:51:32.802: %HA_EM-3-FMPD_ERROR: Error executing applet SendSecureEmailEEM statement 0010
```

The packet capture documented from this exchange is attached as "NoCertificateInstalled.pcap". The final TLS packet from the router (10.122.x.x) to the Gmail SMTP server (142.251.163.xx) shows that the TLS



negotiation was terminated due to the same "Bad Certificate" message seen in the debugs previously.

```
Frame 33: 61 bytes on wire (488 bits), 61 bytes captured (488 bits)
Ethernet II, Src: Cisco_a3:c5:f0 (74:86:0b:a3:c5:f0), Dst: Cisco_f0:44:45 (00:08:30:f0:44:45)
Internet Protocol Version 4, Src: 10.122.xx.xx, Dst: 142.251.163.xx
Transmission Control Protocol, Src Port: 13306, Dst Port: 587, Seq: 189, Ack: 4516, Len: 7
Transport Layer Security
TLV1.2 Record Layer: Alert (Level: Fatal, Description: Bad Certificate)
Content Type: Alert (21)
Version: TLS 1.2 (0x0303)
Length: 2
Alert Message
Level: Fatal (2)
Description: Bad Certificate (42)
```

## Certificates for Securing SMTP

Because the certificates that allow the Cisco IOS XE device to trust Gmail's servers are missing, the fix is to install one/all of those certificates in a trustpoint on the device.

For example, the full debugs of the previous test show these certificate lookups that took place:

```
CRYPTO_PKI(Cert Lookup) issuer="cn=GTS CA 1C3,o=Google Trust Services LLC,c=US" serial number= 52 87 E0
CRYPTO_PKI(Cert Lookup) issuer="cn=GTS Root R1,o=Google Trust Services LLC,c=US" serial number= 02 03 B
CRYPTO_PKI(Cert Lookup) issuer="cn=GlobalSign Root CA,ou=Root CA,o=GlobalSign nv-sa,c=BE" serial number=
```

A certificate for each of these issuers needs to be installed under a trustpoint so the device can establish a secure session with the Gmail SMTP servers. You can create a trustpoint for each issuer using these configurations:

```
crypto pki trustpoint CA-GTS-1C3
  enrollment terminal
  revocation-check none
  chain-validation stop
```

```
crypto pki trustpoint CA-GTS-Root-R1
  enrollment terminal
  revocation-check none
  chain-validation stop
```

```
crypto pki trustpoint CA-GlobalSign-Root
  enrollment terminal
  revocation-check none
  chain-validation stop
```

```
crypto pki trustpoint CA-gmail-SMTP
  enrollment terminal
  revocation-check none
  chain-validation stop
```

You now have a trustpoint for each issuer set up; however, there are no actual certificates associated with them just yet. They are essentially blank trustpoints:

```
# show run | sec crypto pki certificate chain CA-  
crypto pki certificate chain CA-GTS-1C3  
crypto pki certificate chain CA-GTS-Root-R1  
crypto pki certificate chain CA-GlobalSign-Root  
crypto pki certificate chain CA-gmail-SMTP
```

You must track down where those certificates are and then install them on the device.

Searching online for "Google Trust Services 1C3", we quickly come across the Google Trust Services Repository of certificates:

<https://pki.goog/repository/>

After expanding all the certificates on that page, you can search to find "1C3", click the "Action" dropdown, and download the PEM certificate:



Label	Key Size	Serial Number	Expiration	Action
GTS CA 1C3	RSA	23:ec:b0:3e:ec:17:33:8c:4e:33:a6:b4:8a:41:dc:3c:da:12:28:1b:bc:3f:8:13:c0:58:9d:6c:c2:38:75:22	2027-09-30	Action ^
GTS CA 1D4	RSA	64:e2:86:b7:60:63:60:2a:37:2e:fd:60:cd:e8:db:26:56:a4:9e:e1:5e:825:4b:3d:6e:b5:fe:38:f4:28:8b		Preview Certificate View Certificate Details
GTS CA 1D8	RSA	c0:e8:b1:c1:95:cd:ff:7b:51:37:b9:ad:35:13:a6:12:0b:1d:bf:f4:9e:5e:8c:ea:32:73:bc:8d:76:18:77		Downloads Certificate (PEM) Certificate (DER) Partitioned CRLs (JSON)
GTS CA 1P5	RSA	97:d4:20:03:e1:32:55:29:46:09:7f:20:ef:95:5f:5b:1c:d5:70:aa:43:727:80:03:3a:65:ef:be:69:75:8d		
		11:c6:97:87:87:32:05:6d:e1:7c:1d:a1:34:e9:d2:b6:d2:3c:f1:de:95:b		

Opening the downloaded PEM file with a text editor shows that this is just a certificate that can be imported onto the Cisco IOS XE device under the trustpoint you created previously:

```
-----BEGIN CERTIFICATE-----  
MIIF1jCCA36gAwIBAgINAg08U11rNmCY9QFQZjANBgkqhkiG9w0BAQsFADBHMQsw  
CQYDVQQGEwJVUzEiMCAGA1UEChMZR29vZ2x1IFRydXN0IFN1cnZpY2VzIEExMQZEU  
<snip>  
AJ2xDx8hcFH1mt0G/FX0Kw4zd8NLQsLxdxP8c4CU6x+7Nz/OAipmsHMdMqUybdKw  
juDEI/9bfU11cKwrmz302+BtjKAvpafkm0817tdufThcV4q508DirGKZTqPwJN1  
1IXNDw9bg1kWRxYtnCQ6yICmJhSFm/Y3m6xv+cXDB1Hz4n/FsRC6UfTd  
-----END CERTIFICATE-----
```

You can import it under the "CA-GTS-1C3" trustpoint using the configuration commands:

```
(config)# crypto pki authenticate CA-GTS-1C3
```

Enter the base 64 encoded CA certificate.

End with a blank line or the word "quit" on a line by itself

```
MIIFlJCCA36gAwIBAgINAg08U1lrNmCY9QFQZjANBgkqhkiG9w0BAQsFADBHMQsw
CQYDVQQGEwJVUzEiMCAGA1UEChMZR29vZ2x1IFRydXN0IFN1cnZpY2VzIExMQzEU
<snip>
juDEI/9bfU1lcKwrmz302+BtjjKAvpafkm0817tdufThcV4q508DIrGKZTqPwJN1
1IXNDw9bg1kWRxYtnCQ6yICmJhSFm/Y3m6xv+cXDB1Hz4n/FsRC6UfTd
```

```
Certificate has the following attributes:
Fingerprint MD5: 178EF183 43CCC9E0 ECB0E38D 9DEA03D8
Fingerprint SHA1: 1E7EF647 CBA15028 1C608972 57102878 C4BD8CDC
Certificate validated - Signed by existing trustpoint CA certificate.
```

```
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

```
(config)#
```

And then you can confirm the certificate was installed:

```
# show run | sec crypto pki certificate chain CA-GTS-1C3
crypto pki certificate chain CA-GTS-1C3
certificate ca 0203BC53596B34C718F5015066
 30820596 3082037E A0030201 02020D02 03BC5359 6B34C718 F5015066 300D0609
 2A864886 F70D0101 0B050030 47310B30 09060355 04061302 55533122 30200603
 55040A13 19476F6F 676C6520 54727573 74205365 72766963 6573204C 4C433114
<snip>
E1715E2A E4EF0322 B18A653A 8FC09365 D485CD0F 0F5B8359 1647162D 9C243AC8
80A62614 859BF637 9BAC6FF9 C5C30651 F3E27FC5 B110BA51 F4DD
quit
```

```
#show crypto pki certificates verbose CA-GTS-1C3
CA Certificate
Status: Available
Version: 3
Certificate Serial Number (hex): 0203BC53596B34C718F5015066
Certificate Usage: Signature
Issuer:
  cn=GTS Root R1
  o=Google Trust Services LLC
  c=US
Subject:
  cn=GTS CA 1C3
  o=Google Trust Services LLC
  c=US
CRL Distribution Points:
  http://crl.pki.goog/gtsr1/gtsr1.crl
Validity Date:
  start date: 00:00:42 UTC Aug 13 2020
  end date: 00:00:42 UTC Sep 30 2027
Subject Key Info:
  Public Key Algorithm: rsaEncryption
  RSA Public Key: (2048 bit)
Signature Algorithm: SHA256 with RSA Encryption
Fingerprint MD5: 178EF183 43CCC9E0 ECB0E38D 9DEA03D8
Fingerprint SHA1: 1E7EF647 CBA15028 1C608972 57102878 C4BD8CDC
```

```
X509v3 extensions:
  X509v3 Key Usage: 86000000
    Digital Signature
    Key Cert Sign
    CRL Signature
  X509v3 Subject Key ID: 8A747FAF 85CDEE95 CD3D9CD0 E24614F3 71351D27
  X509v3 Basic Constraints:
    CA: TRUE
  X509v3 Authority Key ID: E4AF2B26 711A2B48 27852F52 662CEFF0 8913713E
  Authority Info Access:
    OCSP URL: http://ocsp.pki.goog/gtsr1
    CA ISSUERS: http://pki.goog/repo/certs/gtsr1.der
  X509v3 CertificatePolicies:
    Policy: 2.23.140.1.2.2
    Policy: 2.23.140.1.2.1
    Policy: 1.3.6.1.4.1.11129.2.5.3
      Qualifier ID: 1.3.6.1.5.5.7.2.1
      Qualifier Info: https://pki.goog/repository/
  Extended Key Usage:
    Client Auth
    Server Auth
  Cert install time: 02:31:20 UTC Mar 16 2023
  Cert install time in nsec: 1678933880873946880
  Associated Trustpoints: CA-GTS-1C3
```

Next, you can install the certificates for the other two issuers.

CA-GTS-Root-R1:

Configuration:

[Spoiler](#) (Highlight to read)

```
(config)# crypto pki authenticate CA-GTS-Root-R1

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

MIIFVzCAAz+gAwIBAgINAgPlk28xsBNJiGuiFzANBgkqhkiG9w0BAQwFADBHMQsw
CQYDVQQGEwJVUzEiMCAGA1UEChMZR29vZ2x1IFRydXN0IFN1cnZpY2VzIExMQzEU
<snip>
2tIMPNUzjSmhDYAPexZ3FL//2wmUsp08IFgV6dtxQ/PeEMMA3Kgq1bbC1j+Qa3bb
bP6MvPJwNQzcmRk13NFIRmPVNnGuV/u3gm3c

Certificate has the following attributes:
Fingerprint MD5: 05FED0BF 71A8A376 63DA01E0 D852DC40
Fingerprint SHA1: E58C1CC4 913B3863 4BE9106E E3AD8E6B 9DD9814A

% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Certificate successfully imported

(config)# end
```

```
(config)# crypto pki authenticate CA-GTS-Root-R1Enter the base 64 encoded CA certificate.End with a
blank line or the word "quit" on a line by
itselfMIIFVzCAAz+gAwIBAgINAgPlk28xsBNJiGuiFzANBgkqhkiG9w0BAQwFADBHMQswCQYDVQQGEwJVUzEiMCAGA1UEChMZR29vZ2x1IFRydXN0IFN1cnZpY2VzIExMQzEU
has the following attributes:Fingerprint MD5: 05FED0BF 71A8A376 63DA01E0 D852DC40 Fingerprint
```

SHA1: E58C1CC4 913B3863 4BE9106E E3AD8E6B 9DD9814A% Do you accept this certificate?  
[yes/no]: yesTrustpoint CA certificate accepted.% Certificate successfully imported(config)# end

Running-config Verification:

[Spoiler](#) (Highlight to read)

```
# show run | sec crypto pki certificate chain CA-GTS-Root-R1
crypto pki certificate chain CA-GTS-Root-R1
certificate ca 0203E5936F31B01349886BA217
 30820557 3082033F A0030201 02020D02 03E5936F 31B01349 886BA217 300D0609
 2A864886 F70D0101 0C050030 47310B30 09060355 04061302 55533122 30200603
<snip>
6775C119 3A2B474E D3428EFD 31C81666 DAD20C3C DBB38EC9 A10D800F 7B167714
BFFFDB09 94B293BC 205815E9 DB7143F3 DE10C300 DCA82A95 B6C2D63F 906B76DB
6CFE8CBC F270350C DC991935 DCD7C846 63D53671 AE57FBB7 826DDC
quit
```

```
# show run | sec crypto pki certificate chain CA-GTS-Root-R1crypto pki certificate chain CA-GTS-Root-R1
certificate ca 0203E5936F31B01349886BA217 30820557 3082033F A0030201 02020D02 03E5936F
31B01349 886BA217 300D0609 2A864886 F70D0101 0C050030 47310B30 09060355 04061302
55533122 30200603 <snip> 6775C119 3A2B474E D3428EFD 31C81666 DAD20C3C DBB38EC9
A10D800F 7B167714 BFFFDB09 94B293BC 205815E9 DB7143F3 DE10C300 DCA82A95 B6C2D63F
906B76DB 6CFE8CBC F270350C DC991935 DCD7C846 63D53671 AE57FBB7 826DDC quit
```

Show Crypto Verification:

[Spoiler](#) (Highlight to read)

```
# show crypto pki certificates verbose CA-GTS-Root-R1
CA Certificate
Status: Available
Version: 3
Certificate Serial Number (hex): 0203E5936F31B01349886BA217
Certificate Usage: Signature
Issuer:
  cn=GTS Root R1
  o=Google Trust Services LLC
  c=US
Subject:
  cn=GTS Root R1
  o=Google Trust Services LLC
  c=US
Validity Date:
  start date: 00:00:00 UTC Jun 22 2016
  end date: 00:00:00 UTC Jun 22 2036
Subject Key Info:
  Public Key Algorithm: rsaEncryption
  RSA Public Key: (4096 bit)
Signature Algorithm: SHA384 with RSA Encryption
Fingerprint MD5: 05FED0BF 71A8A376 63DA01E0 D852DC40
Fingerprint SHA1: E58C1CC4 913B3863 4BE9106E E3AD8E6B 9DD9814A
X509v3 extensions:
  X509v3 Key Usage: 86000000
    Digital Signature
    Key Cert Sign
    CRL Signature
  X509v3 Subject Key ID: E4AF2B26 711A2B48 27852F52 662CEFF0 8913713E
  X509v3 Basic Constraints:
```

CA: TRUE  
Authority Info Access:  
Cert install time: 14:39:38 UTC Mar 13 2023  
Cert install time in nsec: 1678718378546968064  
Associated Trustpoints: CA-GTS-Root-R1 Trustpool

# show crypto pki certificates verbose CA-GTS-Root-R1CA Certificate Status: Available Version: 3  
Certificate Serial Number (hex): 0203E5936F31B01349886BA217 Certificate Usage: Signature Issuer:  
cn=GTS Root R1 o=Google Trust Services LLC c=US Subject: cn=GTS Root R1 o=Google Trust Services  
LLC c=US Validity Date: start date: 00:00:00 UTC Jun 22 2016 end date: 00:00:00 UTC Jun 22 2036  
Subject Key Info: Public Key Algorithm: rsaEncryption RSA Public Key: (4096 bit) Signature Algorithm:  
SHA384 with RSA Encryption Fingerprint MD5: 05FED0BF 71A8A376 63DA01E0 D852DC40  
Fingerprint SHA1: E58C1CC4 913B3863 4BE9106E E3AD8E6B 9DD9814A X509v3 extensions: X509v3  
Key Usage: 86000000 Digital Signature Key Cert Sign CRL Signature X509v3 Subject Key ID: E4AF2B26  
711A2B48 27852F52 662CEFF0 8913713E X509v3 Basic Constraints: CA: TRUE Authority Info Access:  
Cert install time: 14:39:38 UTC Mar 13 2023 Cert install time in nsec: 1678718378546968064 Associated  
Trustpoints: CA-GTS-Root-R1 Trustpool

CA-GlobalSign-Root:

This certificate was found at this location:

<https://support.globalsign.com/ca-certificates/root-certificates/globalsign-root-certificates>

Configuration:

Spoiler (Highlight to read)

```
(config)# crypto pki authenticate CA-GlobalSign-Root
```

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

```
MIIDdTCCA12gAwIBAgILBAAAAAABFUtaW5QwDQYJKoZIhvcNAQEFBQAwwVzELMAkG  
A1UEBhMCQkUxGTAXBgNVBAoTEEdsb2JhbFNpZ24gbnYtc2ExEDAOBgNVBAsTB1Jv  
<snip>  
DKqC5J1R3XC321Y9YeRq4VzW9v493kHMB65jUr9TU/Qr6cf9tveCX4XSQRjbgbME  
HMUfpIBvFSDJ3gyICh3WZ1Xi/EjJKSZp4A==
```

Certificate has the following attributes:  
Fingerprint MD5: 3E455215 095192E1 B75D379F B187298A  
Fingerprint SHA1: B1BC968B D4F49D62 2AA89A81 F2150152 A41D829C

% Do you accept this certificate? [yes/no]: yes  
Trustpoint CA certificate accepted.  
% Certificate successfully imported

```
(config)# end
```

```
(config)# crypto pki authenticate CA-GlobalSign-RootEnter the base 64 encoded CA certificate.End with a  
blank line or the word "quit" on a line by  
itselfMIIDdTCCA12gAwIBAgILBAAAAAABFUtaW5QwDQYJKoZIhvcNAQEFBQAwwVzELMAkGA1UEBhMCQkUxGTAXBgNVBAoTEEdsb2JhbFNpZ24gbnYtc2ExEDAOBgNVBAsTB1Jv  
has the following attributes:Fingerprint MD5: 3E455215 095192E1 B75D379F B187298A Fingerprint  
SHA1: B1BC968B D4F49D62 2AA89A81 F2150152 A41D829C% Do you accept this certificate? [yes/no]:  
yesTrustpoint CA certificate accepted.% Certificate successfully imported(config)# end
```

## Running-config Verification:

[Spoiler](#) (Highlight to read)

```
# show run | sec crypto pki certificate chain CA-GlobalSign-Root
crypto pki certificate chain CA-GlobalSign-Root
certificate ca 040000000001154B5AC394
 30820375 3082025D A0030201 02020B04 00000000 01154B5A C394300D 06092A86
 <snip>
 2AC45631 95D06789 852BF96C A65D469D 0CAA82E4 9951DD70 B7DB563D 61E46AE1
 5CD6F6FE 3DDE41CC 07AE6352 BF5353F4 2BE9C7FD B6F7825F 85D24118 DB81B304
 1CC51FA4 806F1520 C9DE0C88 0A1DD666 55E2FC48 C9292669 E0
 quit
```

```
# show run | sec crypto pki certificate chain CA-GlobalSign-Rootcrypto pki certificate chain CA-
GlobalSign-Rootcertificate ca 040000000001154B5AC394 30820375 3082025D A0030201 02020B04
00000000 01154B5A C394300D 06092A86 <snip> 2AC45631 95D06789 852BF96C A65D469D
0CAA82E4 9951DD70 B7DB563D 61E46AE1 5CD6F6FE 3DDE41CC 07AE6352 BF5353F4
2BE9C7FD B6F7825F 85D24118 DB81B304 1CC51FA4 806F1520 C9DE0C88 0A1DD666 55E2FC48
C9292669 E0 quit
```

## Show Crypto Verification:

[Spoiler](#) (Highlight to read)

```
#show crypto pki certificates verbose CA-GlobalSign-Root
CA Certificate
Status: Available
Version: 3
Certificate Serial Number (hex): 040000000001154B5AC394
Certificate Usage: Signature
Issuer:
cn=GlobalSign Root CA
ou=Root CA
o=GlobalSign nv-sa
c=BE
Subject:
cn=GlobalSign Root CA
ou=Root CA
o=GlobalSign nv-sa
c=BE
Validity Date:
start date: 12:00:00 UTC Sep 1 1998
end date: 12:00:00 UTC Jan 28 2028
Subject Key Info:
Public Key Algorithm: rsaEncryption
RSA Public Key: (2048 bit)
Signature Algorithm: SHA1 with RSA Encryption
Fingerprint MD5: 3E455215 095192E1 B75D379F B187298A
Fingerprint SHA1: B1BC968B D4F49D62 2AA89A81 F2150152 A41D829C
X509v3 extensions:
X509v3 Key Usage: 60000000
Key Cert Sign
CRL Signature
X509v3 Subject Key ID: 607B661A 450D97CA 89502F7D 04CD34A8 FFFCFD4B
X509v3 Basic Constraints:
CA: TRUE
```

Authority Info Access:  
Cert install time: 03:03:01 UTC Mar 16 2023  
Cert install time in nsec: 1678935781942944000  
Associated Trustpoints: CA-GlobalSign-Root

```
#show crypto pki certificates verbose CA-GlobalSign-RootCA CertificateStatus: AvailableVersion:
3Certificate Serial Number (hex): 040000000001154B5AC394Certificate Usage: SignatureIssuer:
cn=GlobalSign Root CAou=Root CAo=GlobalSign nv-sac=BESubject: cn=GlobalSign Root CAou=Root
CAo=GlobalSign nv-sac=BEValidity Date: start date: 12:00:00 UTC Sep 1 1998end date: 12:00:00 UTC
Jan 28 2028Subject Key Info:Public Key Algorithm: rsaEncryptionRSA Public Key: (2048 bit)Signature
Algorithm: SHA1 with RSA EncryptionFingerprint MD5: 3E455215 095192E1 B75D379F B187298A
Fingerprint SHA1: B1BC968B D4F49D62 2AA89A81 F2150152 A41D829C X509v3 extensions:X509v3
Key Usage: 6000000Key Cert SignCRL SignatureX509v3 Subject Key ID: 607B661A 450D97CA
89502F7D 04CD34A8 FFFCFD4B X509v3 Basic Constraints:CA: TRUEAuthority Info Access:Cert install
time: 03:03:01 UTC Mar 16 2023 Cert install time in nsec: 1678935781942944000Associated Trustpoints:
CA-GlobalSign-Root
```

CA-gmail-SMTP:

The TLS certificate for Gmail's servers (CA-gmail-SMTP) was found using the steps documented here: [Use TLS certificates for secure transport](#)

Configuration:

[Spoiler](#) (Highlight to read)

```
(ca-trustpoint)# crypto pki authenticate CA-gmail-SMTP
```

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

```
MIIEnhjCCA26gAwIBAgIQUofgQKT+9wcSaLBP3d3w9DANBgkqhkiG9w0BAQsFADBG
MQswCQYDVQQGEWJVVzEiMCAGA1UEChMZR29vZ2x1IFRydXN0IFN1cnZpY2VzIEExM
<snip>
b1J2gZAyjyd4nffRG1jeL5KrsfUR9hIXufqySv1PuPuKS3i3fvsIS21BYEXEe8uZ
gBxJaeTUjncvow==
```

```
Trustpoint 'CA-gmail-SMTP' is a subordinate CA.
but certificate is not a CA certificate.
Manual verification required
Certificate has the following attributes:
Fingerprint MD5: 19651FBE 906A414D 6D57B783 946F30A2
Fingerprint SHA1: 4EF392CB EEB46D5E 47433953 AAEF313F 4C6D2825
```

```
% Do you accept this certificate? [yes/no]: yes
Trustpoint CA certificate accepted.
% Certificate successfully imported
```

```
(config)#
```

```
(ca-trustpoint)# crypto pki authenticate CA-gmail-SMTPEnter the base 64 encoded CA certificate.End with
a blank line or the word "quit" on a line by
itselfMIIEnhjCCA26gAwIBAgIQUofgQKT+9wcSaLBP3d3w9DANBgkqhkiG9w0BAQsFADBG
MQswCQYDVQQGEWJVVzEiMCAGA1UEChMZR29vZ2x1IFRydXN0IFN1cnZpY2VzIEExM
'CA-gmail-SMTP' is a subordinate CA.but certificate is not a CA certificate.Manual verification
requiredCertificate has the following attributes:Fingerprint MD5: 19651FBE 906A414D 6D57B783
946F30A2 Fingerprint SHA1: 4EF392CB EEB46D5E 47433953 AAEF313F 4C6D2825% Do you accept
this certificate? [yes/no]: yesTrustpoint CA certificate accepted.% Certificate successfully
```



imported(config)#

Running-config Verification:

[Spoiler](#) (Highlight to read)

```
# show run | sec crypto pki certificate chain CA-gmail-SMTP
crypto pki certificate chain CA-gmail-SMTP
certificate ca 5287E040A4FEF7071268B04FDDDDF0F4
30820486 3082036E A0030201 02021052 87E040A4 FEF70712 68B04FDD DDF0F430
0D06092A 864886F7 0D01010B 05003046 310B3009 06035504 06130255 53312230
<snip>
92ABB1F5 11F61217 B9FAB24A F94F5283 EE2928B7 7EFB084B 6D416045 C47BCB99
801C4969 E4D48E77 2FA3
quit
```

```
# show run | sec crypto pki certificate chain CA-gmail-SMTP crypto pki certificate chain CA-gmail-SMTP
certificate ca 5287E040A4FEF7071268B04FDDDDF0F4 30820486 3082036E A0030201 02021052
87E040A4 FEF70712 68B04FDD DDF0F430 0D06092A 864886F7 0D01010B 05003046 310B3009
06035504 06130255 53312230 <snip> 92ABB1F5 11F61217 B9FAB24A F94F5283 EE2928B7 7EFB084B
6D416045 C47BCB99 801C4969 E4D48E77 2FA3 quit
```

Show Crypto Verification:

[Spoiler](#) (Highlight to read)

```
# show crypto pki certificates verbose CA-gmail-SMTP
CA Certificate
Status: Available
Version: 3
Certificate Serial Number (hex): 5287E040A4FEF7071268B04FDDDDF0F4
Certificate Usage: Signature
Issuer:
cn=GTS CA 1C3
o=Google Trust Services LLC
c=US
Subject:
cn=smtp.gmail.com
CRL Distribution Points:
http://crls.pki.goog/gts1c3/moVDfISia2k.crl
Validity Date:
start date: 09:15:03 UTC Feb 20 2023
end date: 09:15:02 UTC May 15 2023
Subject Key Info:
Public Key Algorithm: ecEncryption
EC Public Key: (256 bit)
Signature Algorithm: SHA256 with RSA Encryption
Fingerprint MD5: 19651FBE 906A414D 6D57B783 946F30A2
Fingerprint SHA1: 4EF392CB EEB46D5E 47433953 AAEF313F 4C6D2825
X509v3 extensions:
X509v3 Key Usage: 80000000
Digital Signature
X509v3 Subject Key ID: 5CC36972 D07FE997 510E1A67 8A8ECC23 E40CFB68
X509v3 Basic Constraints:
CA: FALSE
X509v3 Subject Alternative Name:
smtp.gmail.com
IP Address :
OtherNames :
```

```
X509v3 Authority Key ID: 8A747FAF 85CDEE95 CD3D9CD0 E24614F3 71351D27
Authority Info Access:
OCSP URL: http://ocsp.pki.goog/gts1c3
CA ISSUERS: http://pki.goog/repo/certs/gts1c3.der
X509v3 CertificatePolicies:
Policy: 2.23.140.1.2.1
Extended Key Usage:
Server Auth
Cert install time: 03:10:41 UTC Mar 16 2023
Cert install time in nsec: 1678936241822955008
Associated Trustpoints: CA-gmail-SMTP
```

```
# show crypto pki certificates verbose CA-gmail-SMTPCA CertificateStatus: AvailableVersion: 3Certificate
Serial Number (hex): 5287E040A4FEF7071268B04FDDDDF0F4Certificate Usage: SignatureIssuer:
cn=GTS CA 1C3o=Google Trust Services LLCc=USSubject: cn=smtp.gmail.comCRL Distribution Points:
http://crls.pki.goog/gts1c3/moVDfISia2k.crlValidity Date: start date: 09:15:03 UTC Feb 20 2023end date:
09:15:02 UTC May 15 2023Subject Key Info:Public Key Algorithm: ecEncryptionEC Public Key: (256
bit)Signature Algorithm: SHA256 with RSA EncryptionFingerprint MD5: 19651FBE 906A414D
6D57B783 946F30A2 Fingerprint SHA1: 4EF392CB EEB46D5E 47433953 AAEF313F 4C6D2825
X509v3 extensions:X509v3 Key Usage: 80000000Digital SignatureX509v3 Subject Key ID: 5CC36972
D07FE997 510E1A67 8A8ECC23 E40CFB68 X509v3 Basic Constraints:CA: FALSEX509v3 Subject
Alternative Name:smtp.gmail.com IP Address : OtherNames : X509v3 Authority Key ID: 8A747FAF
85CDEE95 CD3D9CD0 E24614F3 71351D27 Authority Info Access:OCSP URL:
http://ocsp.pki.goog/gts1c3CA ISSUERS: http://pki.goog/repo/certs/gts1c3.derX509v3
CertificatePolicies:Policy: 2.23.140.1.2.1Extended Key Usage:Server AuthCert install time: 03:10:41 UTC
Mar 16 2023 Cert install time in nsec: 1678936241822955008Associated Trustpoints: CA-gmail-SMTP
```

## An Easier Way to Find the Certificates

Alternatively, you can try using an openssl call from a server/laptop as an easier way to get the certificates from an SMTP server without having to use debugs and searching Google to track them down:

```
openssl s_client -showcerts -verify 5 -connect gmail-smtp-in.l.google.com:25 -starttls smtp
```

You could use smtp.gmail.com as well:

```
openssl s_client -showcerts -verify 5 -connect smtp.gmail.com:25 -starttls smtp
```

The outputs from that call will include the actual certificates themselves that can be used for the “crypto pki authenticate <trustpoint>” configurations.

## Testing EEM with Secure SMTP Again

Now that the certificates are applied to the Cisco IOS XE device, the EEM script will send the secure SMTP messages as expected.

# event manager run SendSecureEmailEEM

Check the Spoiler for the full crypto and ssl debug outputs:

[Spoiler](#) (Highlight to read)

# event manager run SendSecureEmailEEM

```
*Mar 16 03:28:50.673: CRYPTO_OPSSL: Allocated the memory for OPSSLContext
*Mar 16 03:28:50.673: CRYPTO_OPSSL: Set cipher specs to mask 0x02FC0000 for version 128
*Mar 16 03:28:50.674: Set the Default EC Curve list: 0x70Set the EC curve list: secp521r1:secp384r1:prim
*Mar 16 03:28:50.674: opssl_SetPKIInfo entry
*Mar 16 03:28:50.674: CRYPTO_PKI: (A069B) Session started - identity selected (TP-self-signed-486541296
*Mar 16 03:28:50.674: CRYPTO_PKI: Begin local cert chain retrieval.
*Mar 16 03:28:50.674: CRYPTO_PKI(Cert Lookup) issuer="cn=IOS-Self-Signed-Certificate-486541296" serial
*Mar 16 03:28:50.674: CRYPTO_PKI: looking for cert in handle=7F41EE523CE0, digest=
1C 7F 3D 52 67 66 D5 59 E2 66 58 E7 8B E7 9B 8E
*Mar 16 03:28:50.675: CRYPTO_PKI: Done with local cert chain fetch 0.
*Mar 16 03:28:50.675: CRYPTO_PKI: Rcvd request to end PKI session A069B.
*Mar 16 03:28:50.675: CRYPTO_PKI: PKI session A069B has ended. Freeing all resources.TP-self-signed-486
*Mar 16 03:28:50.675: opssl_SetPKIInfo done.
*Mar 16 03:28:50.675: CRYPTO_OPSSL: Common Criteria is disabled on this session.Disabling Common Criter
*Mar 16 03:28:50.675: CRYPTO_OPSSL: ciphersuites ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA
*Mar 16 03:28:50.676: Handshake start: before SSL initialization
*Mar 16 03:28:50.676: SSL_connect:before SSL initialization
*Mar 16 03:28:50.676: >>> ??? [length 0005]
*Mar 16 03:28:50.676: 16 03 01 00 95
*Mar 16 03:28:50.676:
*Mar 16 03:28:50.676: >>> TLS 1.2 Handshake [length 0095], ClientHello
*Mar 16 03:28:50.676: 01 00 00 91 03 03 26 4B 9F B3 44 94 FD 5F FD A1
<snip>
*Mar 16 03:28:50.679: 03 03 01 02 01
*Mar 16 03:28:50.679:
*Mar 16 03:28:50.679: SSL_connect:SSLv3/TLS write client hello
*Mar 16 03:28:50.692: <<< ??? [length 0005]
*Mar 16 03:28:50.692: 16 03 03 00 3F
*Mar 16 03:28:50.692:
*Mar 16 03:28:50.692: SSL_connect:SSLv3/TLS write client hello
*Mar 16 03:28:50.692: <<< TLS 1.2 Handshake [length 003F], ServerHello
*Mar 16 03:28:50.692: 02 00 00 3B 03 03 64 12 7E 05 25 F6 7A BD A0 2E
*Mar 16 03:28:50.692: 58 83 12 7F 90 CD F4 AB E2 69 53 A8 C7 FC 44 4F
*Mar 16 03:28:50.692: 57 4E 47 52 44 01 00 C0 2B 00 00 13 00 17 00 00
*Mar 16 03:28:50.693: FF 01 00 01 00 00 0B 00 02 01 00 00 23 00 00
*Mar 16 03:28:50.693: TLS server extension "unknown" (id=23), len=0
TLS server extension "renegotiate" (id=65281), len=1
*Mar 16 03:28:50.693: 00
*Mar 16 03:28:50.693: TLS server extension "EC point formats" (id=11), len=2
*Mar 16 03:28:50.693: 01 00
*Mar 16 03:28:50.693: TLS server extension "session ticket" (id=35), len=0
*Mar 16 03:28:50.693: <<< ??? [length 0005]
*Mar 16 03:28:50.693: 16 03 03 0F 9A
*Mar 16 03:28:50.694:
*Mar 16 03:28:50.702: SSL_connect:SSLv3/TLS read server hello
```

\*Mar 16 03:28:50.702: <<< TLS 1.2 Handshake [length 0F9A], Certificate  
\*Mar 16 03:28:50.702: 0B 00 0F 96 00 0F 93 00 04 8A 30 82 04 86 30 82  
\*Mar 16 03:28:50.702: 03 6E A0 03 02 01 02 02 10 52 87 E0 40 A4 FE F7  
<snip>  
\*Mar 16 03:28:50.763: 82 35 CF 62 8B C9 24 8B A5 B7 39 0C BB 7E 2A 41  
\*Mar 16 03:28:50.763: BF 52 CF FC A2 96 B6 C2 82 3F  
\*Mar 16 03:28:50.763:  
\*Mar 16 03:28:50.765: CC\_DEBUG: Entering shim layer app callback function  
\*Mar 16 03:28:50.765: CRYPTO\_PKI: (A069C) Session started - identity not specified  
\*Mar 16 03:28:50.765: CRYPTO\_PKI: (A069C) Adding peer certificate  
\*Mar 16 03:28:50.767: CRYPTO\_PKI: Added x509 peer certificate - (1162) bytes  
\*Mar 16 03:28:50.767: CRYPTO\_PKI: (A069C) Adding peer certificate  
\*Mar 16 03:28:50.768: CRYPTO\_PKI: Added x509 peer certificate - (1434) bytes  
\*Mar 16 03:28:50.768: CRYPTO\_PKI: (A069C) Adding peer certificate  
\*Mar 16 03:28:50.770: CRYPTO\_PKI: Added x509 peer certificate - (1382) bytes  
\*Mar 16 03:28:50.770: CRYPTO\_OPSSL: Validate Certificate Chain Callback  
\*Mar 16 03:28:50.770: CRYPTO\_PKI(Cert Lookup) issuer="cn=GTS CA 1C3,o=Google Trust Services LLC,c=US" s  
  
\*Mar 16 03:28:50.770: CRYPTO\_PKI: looking for cert in handle=7F41EE523CE0, digest=  
A7 CC 4B 0F 36 C3 AC D1 2F 77 DD 1D 9A 37 DC FC  
  
\*Mar 16 03:28:50.770: CRYPTO\_PKI(Cert Lookup) issuer="cn=GTS Root R1,o=Google Trust Services LLC,c=US" s  
  
\*Mar 16 03:28:50.771: CRYPTO\_PKI: looking for cert in handle=7F41EE523CE0, digest=  
03 9F CF 59 82 EE 09 CC 4F 53 AE D8 02 7E 4B AF  
  
\*Mar 16 03:28:50.771: CRYPTO\_PKI(Cert Lookup) issuer="cn=GlobalSign Root CA,ou=Root CA,o=GlobalSign nv-  
  
\*Mar 16 03:28:50.771: CRYPTO\_PKI: looking for cert in handle=7F41EE523CE0, digest=  
94 40 D1 90 A0 A3 5D 47 E5 B5 31 F6 63 AD 1B 0A  
  
\*Mar 16 03:28:50.771: CRYPTO\_PKI: Cert record not found for issuer serial.  
\*Mar 16 03:28:50.772: CRYPTO\_PKI: crypto\_pki\_get\_cert\_record\_by\_subject()  
\*Mar 16 03:28:50.772: CRYPTO\_PKI: Found a subject match  
\*Mar 16 03:  
#28:50.772: CRYPTO\_PKI: ip-ext-val: IP extension validation not required:Incrementing refcount for cont  
\*Mar 16 03:28:50.773: CRYPTO\_PKI: create new ca\_req\_context type PKI\_VERIFY\_CHAIN\_CONTEXT,ident 35  
\*Mar 16 03:28:50.773: CRYPTO\_PKI: (A069C)validation path has 1 certs  
  
\*Mar 16 03:28:50.773: CRYPTO\_PKI: (A069C) Check for identical certs  
\*Mar 16 03:28:50.773: CRYPTO\_PKI(Cert Lookup) issuer="cn=GlobalSign Root CA,ou=Root CA,o=GlobalSign nv-  
  
\*Mar 16 03:28:50.774: CRYPTO\_PKI: looking for cert in handle=7F41EE523CE0, digest=  
94 40 D1 90 A0 A3 5D 47 E5 B5 31 F6 63 AD 1B 0A  
  
\*Mar 16 03:28:50.774: CRYPTO\_PKI: Cert record not found for issuer serial.  
\*Mar 16 03:28:50.774: CRYPTO\_PKI : (A069C) Validating non-trusted cert  
\*Mar 16 03:28:50.774: CRYPTO\_PKI: (A069C) Create a list of suitable trustpoints  
\*Mar 16 03:28:50.774: CRYPTO\_PKI: crypto\_pki\_get\_cert\_record\_by\_issuer()  
\*Mar 16 03:28:50.774: CRYPTO\_PKI: Found a issuer match  
\*Mar 16 03:28:50.774: CRYPTO\_PKI: (A069C) Suitable trustpoints are: CA-GlobalSign-Root,  
\*Mar 16 03:28:50.775: CRYPTO\_PKI: (A069C) Attempting to validate certificate using CA-GlobalSign-Root p  
\*Mar 16 03:28:50.775: CRYPTO\_PKI: (A069C) Using CA-GlobalSign-Root to validate certificate  
\*Mar 16 03:28:50.775: CRYPTO\_PKI(make trusted certs chain)  
\*Mar 16 03:28:50.775: CRYPTO\_PKI: Added 1 certs to trusted chain.  
\*Mar 16 03:28:50.775: CRYPTO\_PKI: Prepare session revocation service providers  
\*Mar 16 03:28:50.776: P11:C\_CreateObject:  
\*Mar 16 03:28:50.776: CKA\_CLASS: PUBLIC KEY  
\*Mar 16 03:28:50.776: CKA\_KEY\_TYPE: RSA  
\*Mar 16 03:28:50.776: CKA\_MODULUS:  
DA 0E E6 99 8D CE A3 E3 4F 8A 7E FB F1 8B 83 25  
6B EA 48 1F F1 2A B0 B9 95 11 04 BD F0 63 D1 E2

<snip>

\*Mar 16 03:28:50.780: CKA\_PUBLIC\_EXPONENT: 01 00 01  
\*Mar 16 03:28:50.780: CKA\_VERIFY\_RECOVER: 01  
\*Mar 16 03:28:50.780: CRYPTO\_PKI: Deleting cached key having key id 45  
\*Mar 16 03:28:50.781: CRYPTO\_PKI: Attempting to insert the peer's public key into cache  
\*Mar 16 03:28:50.781: CRYPTO\_PKI:Peer's public inserted successfully with key id 46  
\*Mar 16 03:28:50.781: P11:C\_CreateObject: 131118  
\*Mar 16 03:28:50.781: P11:C\_GetMechanismInfo slot 1 type 3 (invalid mechanism)  
\*Mar 16 03:28:50.781: P11:C\_GetMechanismInfo slot 1 type 1  
\*Mar 16 03:28:50.781: P11:C\_VerifyRecoverInit - 131118  
\*Mar 16 03:28:50.781: P11:C\_VerifyRecover - 131118  
\*Mar 16 03:28:50.781: P11:found pubkey in cache using index = 46  
\*Mar 16 03:28:50.781: P11:public key found is :  
30 82 01 22 30 0D 06 09 2A 86 48 86 F7 0D 01 01  
01 05 00 03 82 01 0F 00 30 82 01 0A 02 82 01 01  
<snip>  
CF 02 03 01 00 01

\*Mar 16 03:28:50.788: P11:CEAL:CRYPTO\_NO\_ERR  
\*Mar 16 03:28:50.788: P11:C\_DestroyObject 2:2002E  
\*Mar 16 03:28:50.788: CRYPTO\_PKI: Expiring peer's cached key with key id 46  
\*Mar 16 03:28:50.788: CRYPTO\_PKI: (A069C) Certificate is verified  
\*Mar 16 03:28:50.788: CRYPTO\_PKI: Remove session revocation service providers  
\*Mar 16 03:28:50.788: CRYPTO\_PKI: Remove session revocation service providersCA-GlobalSign-Root:validat  
\*Mar 16 03:28:50.788: CRYPTO\_PKI: (A069C) Certificate validated without revocation check:cert refcount  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: Populate AAA auth data  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: Unable to get configured attribute for primary AAA list authorization  
\*Mar 16 03:28:50.790: PKI: Cert key-usage: Digital-Signature , Certificate-Signing , CRL-Signing  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: (A069C)chain cert was anchored to trustpoint CA-GlobalSign-Root, and  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: (A069C) Removing verify context  
  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: destroying ca\_req\_context type PKI\_VERIFY\_CHAIN\_CONTEXT,ident 35, ref  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: ca\_req\_context released  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: (A069C) Validation TP is CA-GlobalSign-Root  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: (A069C) Certificate validation succeeded  
\*Mar 16 03:28:50.790: CRYPTO\_OPSSL: Certificate verification is successful  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: Rcvd request to end PKI session A069C.  
\*Mar 16 03:28:50.790: CRYPTO\_PKI: PKI session A069C has ended. Freeing all resources.:cert refcount aft  
\*Mar 16 03:28:50.791: <<< ??? [length 0005]  
\*Mar 16 03:28:50.791: 16 03 03 00 93  
\*Mar 16 03:28:50.791:  
\*Mar 16 03:28:50.791: SSL\_connect:SSLv3/TLS read server certificate  
\*Mar 16 03:28:50.791: <<< TLS 1.2 Handshake [length 0093], ServerKeyExchange  
\*Mar 16 03:28:50.791: 0C 00 00 8F 03 00 17 41 04 3D 49 34 A3 52 D4 EB  
\*Mar 16 03:28:50.791: DE A2 9E CC B0 91 AA CB 1B 39 D0 26 1B 7D FF 31  
\*Mar 16 03:28:50.792: E0 D7 D5 9C 75 C0 7D 5B D6 B2 0A B5 CC EA E1 4B  
\*Mar 16 03:28:50.792: 4E E5 72 7B 54 5D 9B B2 95 91 E0 CC D6 A5 8E CE  
\*Mar 16 03:28:50.792: 8D 36 C9 83 42 B0 4D AC 0C 04 03 00 46 30 44 02  
\*Mar 16 03:28:50.792: 20 67 B3 F1 DA D1 BF 13 72 DD B6 B2 11 3B 6E 6F  
\*Mar 16 03:28:50.793: 87 52 D9 00 F7 44 31 C3 C2 5E BE 2D FF 93 4E F0  
\*Mar 16 03:28:50.793: A8 02 20 24 42 91 BE B7 10 1C D1 C0 12 28 FB 1F  
\*Mar 16 03:28:50.793: E4 DE 81 0B AA 66 19 CD 28 5A A0 30 7D 3C 4A 56  
\*Mar 16 03:28:50.793: 0D 94 E2  
\*Mar 16 03:28:50.793:  
\*Mar 16 03:28:50.794: P11:C\_FindObjectsInit:  
\*Mar 16 03:28:50.794: CKA\_CLASS: PUBLIC KEY  
\*Mar 16 03:28:50.794: CKA\_KEY\_TYPE: : 00 00 00 03  
  
\*Mar 16 03:28:50.794: CKA\_ECDSA\_PARAMS:  
30 59 30 13 06 07 2A 86 48 CE 3D 02 01 06 08 2A  
86 48 CE 3D 03 01 07 03 42 00 04 63 B6 D3 1A 28

<snip>

```
*Mar 16 03:28:50.796: P11:C_FindObjectsFinal
*Mar 16 03:28:50.796: P11:C_VerifyInit - Session found
*Mar 16 03:28:50.796: P11:C_VerifyInit - key id = 131073
*Mar 16 03:28:50.796: P11:C_Verify
*Mar 16 03:28:50.800: P11:CEAL:CRYPTO_NO_ERR
*Mar 16 03:28:50.800: <<< ??? [length 0005]
*Mar 16 03:28:50.800: 16 03 03 00 04
*Mar 16 03:28:50.800:
*Mar 16 03:28:50.800: SSL_connect:SSLv3/TLS read server key exchange
*Mar 16 03:28:50.800: <<< TLS 1.2 Handshake [length 0004], ServerHelloDone
*Mar 16 03:28:50.801: 0E 00 00 00
*Mar 16 03:28:50.801:
*Mar 16 03:28:50.801: SSL_connect:SSLv3/TLS read server done
*Mar 16 03:28:50.810: >>> ??? [length 0005]
*Mar 16 03:28:50.810: 16 03 03 00 46
*Mar 16 03:28:50.811:
*Mar 16 03:28:50.811: >>> TLS 1.2 Handshake [length 0046], ClientKeyExchange
*Mar 16 03:28:50.811: 10 00 00 42 41 04 26 C3 EF 02 05 6C 82 D1 90 B3
*Mar 16 03:28:50.811: 17 31 9A CD DD 8C 81 91 BA E8 C0 86 40 7B 2C E4
*Mar 16 03:28:50.811: 9A 2C 18 9D D1 6A C0 56 A0 98 2E B7 3B AB B3 EB
*Mar 16 03:28:50.811: BB CD 5E 42 C5 76 C0 C4 BF 15 F4 87 F2 7C AD 74
*Mar 16 03:28:50.812: 97 0A 97 2B 06 B5
*Mar 16 03:28:50.812:
*Mar 16 03:28:50.812: SSL_connect:SSLv3/TLS write client key exchange
*Mar 16 03:28:50.812: >>> ??? [length 0005]
*Mar 16 03:28:50.812: 14 03 03 00 01
*Mar 16 03:28:50.812:
*Mar 16 03:28:50.812: >>> TLS 1.2 ChangeCipherSpec [length 0001]
*Mar 16 03:28:51.116: >>> ??? [length 0005]
*Mar 16 03:28:51.116: 17 03 03 00 35
*Mar 16 03:28:51.116:
*Mar 16 03:28:51.116: >>> ??? [length 0005]
*Mar 16 03:28:51.116: 17 03 03 00 1A
*Mar 16 03:28:51.116:
*Mar 16 03:28:51.116: >>> ??? [length 0005]
*Mar 16 03:28:51.116: 17 03 03 00 30
*Mar 16 03:28:51.116:
*Mar 16 03:28:51.116: >>> ??? [length 0005]
*Mar 16 03:28:51.116: 17 03 03 00 1B
*Mar 16 03:28:51.117:
*Mar 16 03:28:51.713: <<< ??? [length 0005]
*Mar 16 03:28:51.713: 17 03 03 00 6D
*Mar 16 03:28:51.713:
*Mar 16 03:28:51.714: >>> ??? [length 0005]
*Mar 16 03:28:51.714: 17 03 03 00 1E
*Mar 16 03:28:51.714:
*Mar 16 03:28:51.732: <<< ??? [length 0005]
*Mar 16 03:28:51.732: 17 03 03 00 71
*Mar 16 03:28:51.732:
```

```
# event manager run SendSecureEmailEEM*Mar 16 03:28:50.673: CRYPTO_OPSSL: Allocated the
memory for OPSSLContext*Mar 16 03:28:50.673: CRYPTO_OPSSL: Set cipher specs to mask
0x02FC0000 for version 128*Mar 16 03:28:50.674: Set the Default EC Curve list: 0x70Set the EC curve
list: secp521r1:secp384r1:prime256v1*Mar 16 03:28:50.674: opssl_SetPKIInfo entry*Mar 16 03:28:50.674:
CRYPTO_PKI: (A069B) Session started - identity selected (TP-self-signed-486541296)xTP-self-signed-
486541296:refcount after increment = 1*Mar 16 03:28:50.674: CRYPTO_PKI: Begin local cert chain
retrieval.*Mar 16 03:28:50.674: CRYPTO_PKI(Cert Lookup) issuer="cn=IOS-Self-Signed-Certificate-
486541296" serial number= 01*Mar 16 03:28:50.674: CRYPTO_PKI: looking for cert in
handle=7F41EE523CE0, digest=1C 7F 3D 52 67 66 D5 59 E2 66 58 E7 8B E7 9B 8E*Mar 16
```

03:28:50.675: CRYPTO\_PKI: Done with local cert chain fetch 0.\*Mar 16 03:28:50.675: CRYPTO\_PKI: Rcvd request to end PKI session A069B.\*Mar 16 03:28:50.675: CRYPTO\_PKI: PKI session A069B has ended. Freeing all resources.TP-self-signed-486541296:unlocked trustpoint TP-self-signed-486541296, refcount is 0\*Mar 16 03:28:50.675: opssl\_SetPKIInfo done.\*Mar 16 03:28:50.675: CRYPTO\_OPSSL: Common Criteria is disabled on this session.Disabling Common Criteria mode functionality in CiscoSSL on SSL CTX 0x7F41F28EAFF8\*Mar 16 03:28:50.675: CRYPTO\_OPSSL: ciphersuites ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-SHA384:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-SHA256:AES256-GCM-SHA384:AES256-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-SHA256:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES128-SHA256:AES128-GCM-SHA256:AES128-SHA256\*Mar 16 03:28:50.676: Handshake start: before SSL initialization\*Mar 16 03:28:50.676: SSL\_connect:before SSL initialization\*Mar 16 03:28:50.676: >>> ??? [length 0005]\*Mar 16 03:28:50.676: 16 03 01 00 95\*Mar 16 03:28:50.676: \*Mar 16 03:28:50.676: >>> TLS 1.2 Handshake [length 0095], ClientHello\*Mar 16 03:28:50.676: 01 00 00 91 03 03 26 4B 9F B3 44 94 FD 5F FD A1<snip>\*Mar 16 03:28:50.679: 03 03 01 02 01\*Mar 16 03:28:50.679: \*Mar 16 03:28:50.679: SSL\_connect:SSLv3/TLS write client hello\*Mar 16 03:28:50.692: <<< ??? [length 0005]\*Mar 16 03:28:50.692: 16 03 03 00 3F\*Mar 16 03:28:50.692: \*Mar 16 03:28:50.692: SSL\_connect:SSLv3/TLS write client hello\*Mar 16 03:28:50.692: <<< TLS 1.2 Handshake [length 003F], ServerHello\*Mar 16 03:28:50.692: 02 00 00 3B 03 03 64 12 7E 05 25 F6 7A BD A0 2E\*Mar 16 03:28:50.692: 58 83 12 7F 90 CD F4 AB E2 69 53 A8 C7 FC 44 4F\*Mar 16 03:28:50.692: 57 4E 47 52 44 01 00 C0 2B 00 00 13 00 17 00 00\*Mar 16 03:28:50.693: FF 01 00 01 00 00 0B 00 02 01 00 00 23 00 00\*Mar 16 03:28:50.693: TLS server extension "unknown" (id=23), len=0TLS server extension "renegotiate" (id=65281), len=1\*Mar 16 03:28:50.693: 00\*Mar 16 03:28:50.693: TLS server extension "EC point formats" (id=11), len=2\*Mar 16 03:28:50.693: 01 00\*Mar 16 03:28:50.693: TLS server extension "session ticket" (id=35), len=0\*Mar 16 03:28:50.693: <<< ??? [length 0005]\*Mar 16 03:28:50.693: 16 03 03 0F 9A\*Mar 16 03:28:50.694: \*Mar 16 03:28:50.702: SSL\_connect:SSLv3/TLS read server hello\*Mar 16 03:28:50.702: <<< TLS 1.2 Handshake [length 0F9A], Certificate\*Mar 16 03:28:50.702: 0B 00 0F 96 00 0F 93 00 04 8A 30 82 04 86 30 82\*Mar 16 03:28:50.702: 03 6E A0 03 02 01 02 02 10 52 87 E0 40 A4 FE F7<snip>\*Mar 16 03:28:50.763: 82 35 CF 62 8B C9 24 8B A5 B7 39 0C BB 7E 2A 41\*Mar 16 03:28:50.763: BF 52 CF FC A2 96 B6 C2 82 3F\*Mar 16 03:28:50.763: \*Mar 16 03:28:50.765: CC\_DEBUG: Entering shim layer app callback function\*Mar 16 03:28:50.765: CRYPTO\_PKI: (A069C) Session started - identity not specified\*Mar 16 03:28:50.765: CRYPTO\_PKI: (A069C) Adding peer certificate\*Mar 16 03:28:50.767: CRYPTO\_PKI: Added x509 peer certificate - (1162) bytes\*Mar 16 03:28:50.767: CRYPTO\_PKI: (A069C) Adding peer certificate\*Mar 16 03:28:50.768: CRYPTO\_PKI: Added x509 peer certificate - (1434) bytes\*Mar 16 03:28:50.768: CRYPTO\_PKI: (A069C) Adding peer certificate\*Mar 16 03:28:50.770: CRYPTO\_PKI: Added x509 peer certificate - (1382) bytes\*Mar 16 03:28:50.770: CRYPTO\_OPSSL: Validate Certificate Chain Callback\*Mar 16 03:28:50.770: CRYPTO\_PKI(Cert Lookup) issuer="cn=GTS CA 1C3,o=Google Trust Services LLC,c=US" serial number= 52 87 E0 40 A4 FE F7 07 12 68 B0 4F DD DD F0 F4\*Mar 16 03:28:50.770: CRYPTO\_PKI: looking for cert in handle=7F41EE523CE0, digest=A7 CC 4B 0F 36 C3 AC D1 2F 77 DD 1D 9A 37 DC FC\*Mar 16 03:28:50.770: CRYPTO\_PKI(Cert Lookup) issuer="cn=GTS Root R1,o=Google Trust Services LLC,c=US" serial number= 02 03 BC 53 59 6B 34 C7 18 F5 01 50 66\*Mar 16 03:28:50.771: CRYPTO\_PKI: looking for cert in handle=7F41EE523CE0, digest=03 9F CF 59 82 EE 09 CC 4F 53 AE D8 02 7E 4B AF\*Mar 16 03:28:50.771: CRYPTO\_PKI(Cert Lookup) issuer="cn=GlobalSign Root CA,ou=Root CA,o=GlobalSign nv-sa,c=BE" serial number= 77 BD 0D 6C DB 36 F9 1A EA 21 0F C4 F0 58 D3 0D\*Mar 16 03:28:50.771: CRYPTO\_PKI: looking for cert in handle=7F41EE523CE0, digest=94 40 D1 90 A0 A3 5D 47 E5 B5 31 F6 63 AD 1B 0A\*Mar 16 03:28:50.771: CRYPTO\_PKI: Cert record not found for issuer serial.\*Mar 16 03:28:50.772: CRYPTO\_PKI: crypto\_pki\_get\_cert\_record\_by\_subject()\*Mar 16 03:28:50.772: CRYPTO\_PKI: Found a subject match\*Mar 16 03:28:50.772: CRYPTO\_PKI: ip-ext-val: IP extension validation not required:Incrementing refcount for context id-35 to 1\*Mar 16 03:28:50.773: CRYPTO\_PKI: create new ca\_req\_context type PKI\_VERIFY\_CHAIN\_CONTEXT,ident 35\*Mar 16 03:28:50.773: CRYPTO\_PKI: (A069C)validation path has 1 certs\*Mar 16 03:28:50.773: CRYPTO\_PKI: (A069C) Check for identical certs\*Mar 16 03:28:50.773: CRYPTO\_PKI(Cert Lookup) issuer="cn=GlobalSign Root CA,ou=Root CA,o=GlobalSign nv-sa,c=BE" serial number= 77 BD 0D 6C DB 36 F9 1A EA 21 0F C4 F0 58 D3 0D\*Mar 16 03:28:50.774: CRYPTO\_PKI: looking for cert in handle=7F41EE523CE0, digest=94 40 D1 90 A0 A3 5D 47 E5 B5 31 F6

63 AD 1B 0A\*Mar 16 03:28:50.774: CRYPTO\_PKI: Cert record not found for issuer serial.\*Mar 16  
03:28:50.774: CRYPTO\_PKI : (A069C) Validating non-trusted cert\*Mar 16 03:28:50.774: CRYPTO\_PKI:  
(A069C) Create a list of suitable trustpoints\*Mar 16 03:28:50.774: CRYPTO\_PKI:  
crypto\_pki\_get\_cert\_record\_by\_issuer()\*Mar 16 03:28:50.774: CRYPTO\_PKI: Found a issuer match\*Mar  
16 03:28:50.774: CRYPTO\_PKI: (A069C) Suitable trustpoints are: CA-GlobalSign-Root,\*Mar 16  
03:28:50.775: CRYPTO\_PKI: (A069C) Attempting to validate certificate using CA-GlobalSign-Root  
policy\*Mar 16 03:28:50.775: CRYPTO\_PKI: (A069C) Using CA-GlobalSign-Root to validate  
certificate\*Mar 16 03:28:50.775: CRYPTO\_PKI(make trusted certs chain)\*Mar 16 03:28:50.775:  
CRYPTO\_PKI: Added 1 certs to trusted chain.\*Mar 16 03:28:50.775: CRYPTO\_PKI: Prepare session  
revocation service providers\*Mar 16 03:28:50.776: P11:C\_CreateObject:\*Mar 16 03:28:50.776:  
CKA\_CLASS: PUBLIC KEY\*Mar 16 03:28:50.776: CKA\_KEY\_TYPE: RSA\*Mar 16 03:28:50.776:  
CKA\_MODULUS: DA 0E E6 99 8D CE A3 E3 4F 8A 7E FB F1 8B 83 25 6B EA 48 1F F1 2A B0 B9 95  
11 04 BD F0 63 D1 E2 <snip>\*Mar 16 03:28:50.780: CKA\_PUBLIC\_EXPONENT: 01 00 01\*Mar 16  
03:28:50.780: CKA\_VERIFY\_RECOVER: 01\*Mar 16 03:28:50.780: CRYPTO\_PKI: Deleting cached key  
having key id 45\*Mar 16 03:28:50.781: CRYPTO\_PKI: Attempting to insert the peer's public key into  
cache\*Mar 16 03:28:50.781: CRYPTO\_PKI:Peer's public inserted successfully with key id 46\*Mar 16  
03:28:50.781: P11:C\_CreateObject: 131118\*Mar 16 03:28:50.781: P11:C\_GetMechanismInfo slot 1 type 3  
(invalid mechanism)\*Mar 16 03:28:50.781: P11:C\_GetMechanismInfo slot 1 type 1\*Mar 16 03:28:50.781:  
P11:C\_VerifyRecoverInit - 131118\*Mar 16 03:28:50.781: P11:C\_VerifyRecover - 131118\*Mar 16  
03:28:50.781: P11:found pubkey in cache using index = 46\*Mar 16 03:28:50.781: P11:public key found is :  
30 82 01 22 30 0D 06 09 2A 86 48 86 F7 0D 01 01 01 05 00 03 82 01 0F 00 30 82 01 0A 02 82 01 01  
<snip>CF 02 03 01 00 01\*Mar 16 03:28:50.788: P11:CEAL:CRYPTO\_NO\_ERR\*Mar 16 03:28:50.788:  
P11:C\_DestroyObject 2:2002E\*Mar 16 03:28:50.788: CRYPTO\_PKI: Expiring peer's cached key with key  
id 46\*Mar 16 03:28:50.788: CRYPTO\_PKI: (A069C) Certificate is verified\*Mar 16 03:28:50.788:  
CRYPTO\_PKI: Remove session revocation service providers\*Mar 16 03:28:50.788: CRYPTO\_PKI:  
Remove session revocation service providersCA-GlobalSign-Root:validation status -  
CRYPTO\_VALID\_CERT\_WITH\_WARNING\*Mar 16 03:28:50.788: CRYPTO\_PKI: (A069C) Certificate  
validated without revocation check:cert refcount after increment = 1\*Mar 16 03:28:50.790: CRYPTO\_PKI:  
Populate AAA auth data\*Mar 16 03:28:50.790: CRYPTO\_PKI: Unable to get configured attribute for  
primary AAA list authorization.\*Mar 16 03:28:50.790: PKI: Cert key-usage: Digital-Signature , Certificate-  
Signing , CRL-Signing\*Mar 16 03:28:50.790: CRYPTO\_PKI: (A069C)chain cert was anchored to  
trustpoint CA-GlobalSign-Root, and chain validation result was:  
CRYPTO\_VALID\_CERT\_WITH\_WARNING\*Mar 16 03:28:50.790: CRYPTO\_PKI: (A069C) Removing  
verify context\*Mar 16 03:28:50.790: CRYPTO\_PKI: destroying ca\_req\_context type  
PKI\_VERIFY\_CHAIN\_CONTEXT,ident 35, ref count 1:Decrementing refcount for context id-35 to 0\*Mar  
16 03:28:50.790: CRYPTO\_PKI: ca\_req\_context released\*Mar 16 03:28:50.790: CRYPTO\_PKI: (A069C)  
Validation TP is CA-GlobalSign-Root\*Mar 16 03:28:50.790: CRYPTO\_PKI: (A069C) Certificate  
validation succeeded\*Mar 16 03:28:50.790: CRYPTO\_OPSSL: Certificate verification is successful\*Mar  
16 03:28:50.790: CRYPTO\_PKI: Rcvd request to end PKI session A069C.\*Mar 16 03:28:50.790:  
CRYPTO\_PKI: PKI session A069C has ended. Freeing all resources.:cert refcount after decrement = 0\*Mar  
16 03:28:50.791: <<< ??? [length 0005]\*Mar 16 03:28:50.791: 16 03 03 00 93\*Mar 16 03:28:50.791: \*Mar  
16 03:28:50.791: SSL\_connect:SSLv3/TLS read server certificate\*Mar 16 03:28:50.791: <<< TLS 1.2  
Handshake [length 0093], ServerKeyExchange\*Mar 16 03:28:50.791: 0C 00 00 8F 03 00 17 41 04 3D 49 34  
A3 52 D4 EB\*Mar 16 03:28:50.791: DE A2 9E CC B0 91 AA CB 1B 39 D0 26 1B 7D FF 31\*Mar 16  
03:28:50.792: E0 D7 D5 9C 75 C0 7D 5B D6 B2 0A B5 CC EA E1 4B\*Mar 16 03:28:50.792: 4E E5 72 7B  
54 5D 9B B2 95 91 E0 CC D6 A5 8E CE\*Mar 16 03:28:50.792: 8D 36 C9 83 42 B0 4D AC 0C 04 03 00 46  
30 44 02\*Mar 16 03:28:50.792: 20 67 B3 F1 DA D1 BF 13 72 DD B6 B2 11 3B 6E 6F\*Mar 16  
03:28:50.793: 87 52 D9 00 F7 44 31 C3 C2 5E BE 2D FF 93 4E F0\*Mar 16 03:28:50.793: A8 02 20 24 42  
91 BE B7 10 1C D1 C0 12 28 FB 1F\*Mar 16 03:28:50.793: E4 DE 81 0B AA 66 19 CD 28 5A A0 30 7D  
3C 4A 56\*Mar 16 03:28:50.793: 0D 94 E2\*Mar 16 03:28:50.793: \*Mar 16 03:28:50.794:  
P11:C\_FindObjectsInit:\*Mar 16 03:28:50.794: CKA\_CLASS: PUBLIC KEY\*Mar 16 03:28:50.794:  
CKA\_KEY\_TYPE: : 00 00 00 03\*Mar 16 03:28:50.794: CKA\_ECDSA\_PARAMS: 30 59 30 13 06 07 2A  
86 48 CE 3D 02 01 06 08 2A 86 48 CE 3D 03 01 07 03 42 00 04 63 B6 D3 1A 28 <snip>\*Mar 16  
03:28:50.796: P11:C\_FindObjectsFinal\*Mar 16 03:28:50.796: P11:C\_VerifyInit - Session found\*Mar 16  
03:28:50.796: P11:C\_VerifyInit - key id = 131073\*Mar 16 03:28:50.796: P11:C\_Verify\*Mar 16

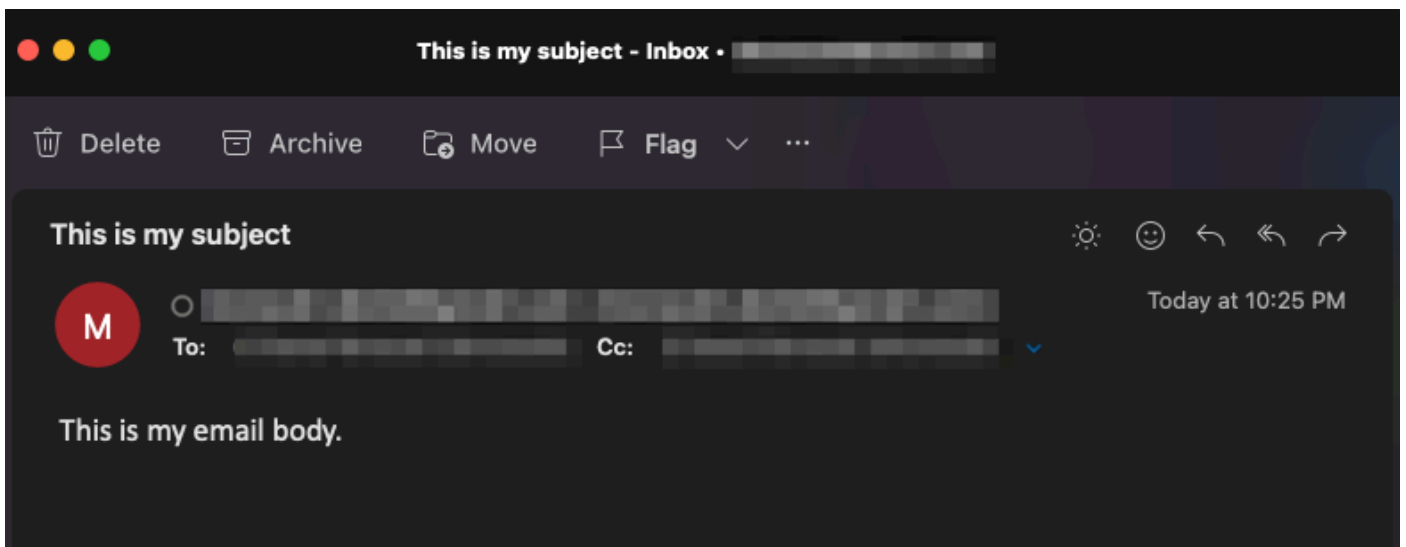


```

03:28:50.800: P11:CEAL:CRYPTO_NO_ERR*Mar 16 03:28:50.800: <<< ??? [length 0005]*Mar 16
03:28:50.800: 16 03 03 00 04*Mar 16 03:28:50.800: *Mar 16 03:28:50.800: SSL_connect:SSLv3/TLS read
server key exchange*Mar 16 03:28:50.800: <<< TLS 1.2 Handshake [length 0004], ServerHelloDone*Mar
16 03:28:50.801: 0E 00 00 00*Mar 16 03:28:50.801: *Mar 16 03:28:50.801: SSL_connect:SSLv3/TLS read
server done*Mar 16 03:28:50.810: >>> ??? [length 0005]*Mar 16 03:28:50.810: 16 03 03 00 46*Mar 16
03:28:50.811: *Mar 16 03:28:50.811: >>> TLS 1.2 Handshake [length 0046], ClientKeyExchange*Mar 16
03:28:50.811: 10 00 00 42 41 04 26 C3 EF 02 05 6C 82 D1 90 B3*Mar 16 03:28:50.811: 17 31 9A CD DD
8C 81 91 BA E8 C0 86 40 7B 2C E4*Mar 16 03:28:50.811: 9A 2C 18 9D D1 6A C0 56 A0 98 2E B7 3B
AB B3 EB*Mar 16 03:28:50.811: BB CD 5E 42 C5 76 C0 C4 BF 15 F4 87 F2 7C AD 74*Mar 16
03:28:50.812: 97 0A 97 2B 06 B5*Mar 16 03:28:50.812: *Mar 16 03:28:50.812: SSL_connect:SSLv3/TLS
write client key exchange*Mar 16 03:28:50.812: >>> ??? [length 0005]*Mar 16 03:28:50.812: 14 03 03 00
01*Mar 16 03:28:50.812: *Mar 16 03:28:50.812: >>> TLS 1.2 ChangeCipherSpec [length 0001]*Mar 16
03:28:51.116: >>> ??? [length 0005]*Mar 16 03:28:51.116: 17 03 03 00 35*Mar 16 03:28:51.116: *Mar 16
03:28:51.116: >>> ??? [length 0005]*Mar 16 03:28:51.116: 17 03 03 00 1A*Mar 16 03:28:51.116: *Mar 16
03:28:51.116: >>> ??? [length 0005]*Mar 16 03:28:51.116: 17 03 03 00 30*Mar 16 03:28:51.116: *Mar 16
03:28:51.116: >>> ??? [length 0005]*Mar 16 03:28:51.116: 17 03 03 00 1B*Mar 16 03:28:51.117: *Mar 16
03:28:51.713: <<< ??? [length 0005]*Mar 16 03:28:51.713: 17 03 03 00 6D*Mar 16 03:28:51.713: *Mar 16
03:28:51.714: >>> ??? [length 0005]*Mar 16 03:28:51.714: 17 03 03 00 1E*Mar 16 03:28:51.714: *Mar 16
03:28:51.732: <<< ??? [length 0005]*Mar 16 03:28:51.732: 17 03 03 00 71*Mar 16 03:28:51.732:

```

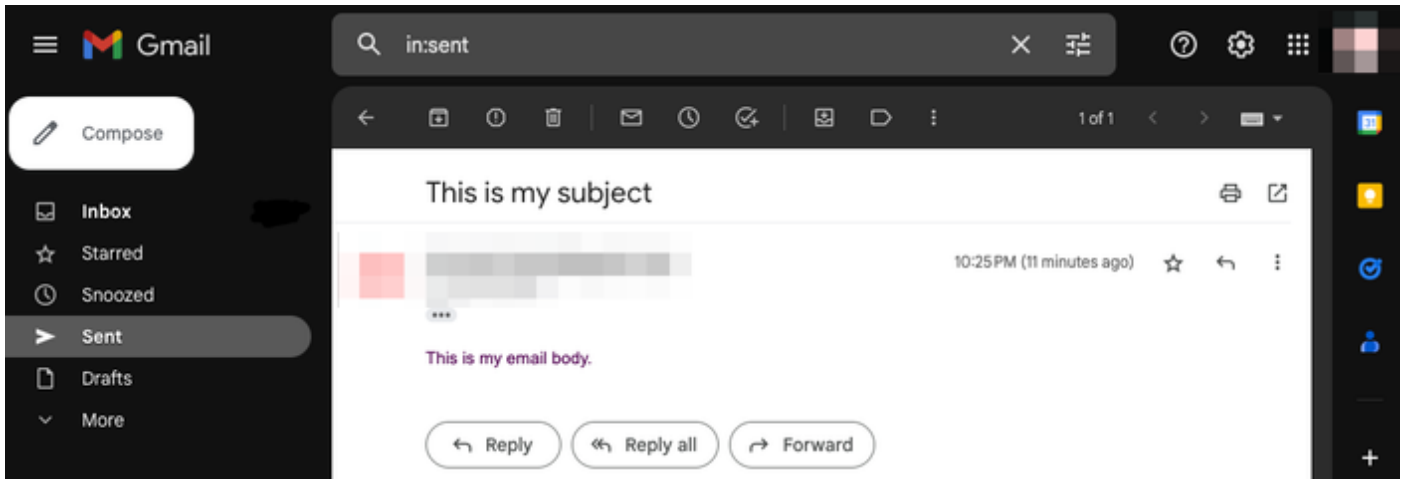
You can verify that the email was received and all fields (to, from, cc, subject, body) are all correctly populated:



You can also verify the TLS handshake and session took place from the packet capture on the Cisco IOS XE device (attached as "WorkingSMTPwithTLS.pcap"):

No.	Time	Source	Destination	Protocol	Length	ID	Info
11	2023-03-16 03:28:50...	10.122.144.150	142.251.163.109	TLSv1.2	208	0x8790 (34704)	Client Hello
12	2023-03-16 03:28:50...	142.251.163.109	10.122.144.150	TLSv1.2	590	0x7641 (30273)	Server Hello
32	2023-03-16 03:28:50...	142.251.163.109	10.122.144.150	TLSv1.2	439	0x7649 (30281)	Certificate, Server Key Exchange, Server Hello Done
33	2023-03-16 03:28:50...	10.122.144.150	142.251.163.109	TLSv1.2	180	0x879d (34717)	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
34	2023-03-16 03:28:50...	142.251.163.109	10.122.144.150	TLSv1.2	349	0x764a (30282)	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
36	2023-03-16 03:28:50...	10.122.144.150	142.251.163.109	TLSv1.2	107	0x879f (34719)	Application Data
38	2023-03-16 03:28:50...	142.251.163.109	10.122.144.150	TLSv1.2	306	0x764c (30284)	Application Data
39	2023-03-16 03:28:50...	10.122.144.150	142.251.163.109	TLSv1.2	116	0x87a0 (34720)	Application Data
41	2023-03-16 03:28:50...	142.251.163.109	10.122.144.150	TLSv1.2	101	0x764e (30286)	Application Data
42	2023-03-16 03:28:50...	10.122.144.150	142.251.163.109	TLSv1.2	109	0x87a1 (34721)	Application Data

You can even verify the emails are reflected in the "Sent" folder of the email account used:



## Other Caveats and Considerations

### Username with @ Symbols

Issues can be seen when attempting to utilize an SMTP relay. Because of the SMTP relay, the server string has this format (an "@" in the username):

```
event manager environment _email_server email.relay@My.Domain.Name:MyPasswordString@smtp-relay.gmail.com
```

The code to parse the username and password splits the string on the first occurrence of the "@" symbol. As a result, the system thinks the server hostname starts immediately after the first "@" symbol through the rest of the string, and it interprets everything before that as the "username:password".

The TCL implementation of SMTP uses a Regular Expression (regex) which handles this username/password/server information differently. Because of that difference, TCL allows usernames with an "@" symbol; however, Cisco IOS XE TCL does not support crypto, so there is no option for sending secure emails over TLS.

To summarize:

- If the email needs to be secure, then you cannot send it with TCL.
- If there is an "@" in your username, then you cannot send it with an EEM.

Cisco bug ID [CSCwe75439](#) was filed to address this opportunity to improve the EEM email feature; however, there is no roadmap for this enhancement request currently.

## Conclusion

As shown here, it is possible to send secure emails via SMTP with TLS using the Embedded Event Manager (EEM) applet. It requires some setup on the server side, as well as configuring the necessary certificates to allow trust, but it is feasible if you want to generate automated, secure, email notifications.