Configure a Site-to-Site VPN Tunnel with ASA and Strongswan

Contents

Introduction

Prerequisites

- Requirements
- Components Used

Configure

- Scenario
- Network Diagram
- ASA Configuration
- strongSwan Configuration
- Useful Commands (strongswan)

Verify

- On ASA
- Phase 1 Verification
- Phase 2 Verification
- On strongSwan

Troubleshoot

- ASA Debugs
- strongSwan Debugs

Related Information

Introduction

This document describes how to configure Site-to-Site IPSec Internet Key Exchange Version 1 tunnel via the CLI between an ASA and a strongSwan server.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Cisco Adaptive Security Appliance (ASA)
- Basic Linux Commands
- General IPSec concepts

Components Used

The information in this document is based on these versions:

- Cisco ASAv running 9.18(4)22
- Ubuntu 20.04 running strongSwan U5.8.2

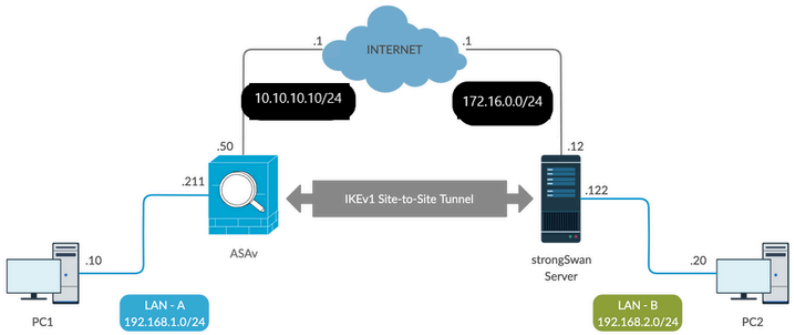
The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Configure

This section describes how to complete the ASA and strongSwan configurations.

Scenario

In this setup, PC1 in LAN-A wants to communicate with PC2 in LAN-B. This traffic needs to be encrypted and sent over an Internet Key Exchange Version 1 (IKEv1) tunnel between ASA and stongSwan server. Both peers authenticate each other with a Pre-shared-key (PSK).



Network Diagram

Note: Ensure that there is connectivity to both the internal and external networks, and especially to the remote peer that is used in order to establish a site-to-site VPN tunnel. You can use a ping in order to verify basic connectivity.

ASA Configuration

<#root>

```
!Configure the ASA interfaces


!
interface GigabitEthernet0/0
nameif inside
security-level 100
ip address 192.168.1.211 255.255.255.0
!
interface GigabitEthernet0/1
nameif outside
security-level 0
ip address 10.10.10.10 255.255.255.0
```

```
!
!Configure the ACL for the VPN traffic of interest


!
object-group network local-network
network-object 192.168.1.0 255.255.255.0
!
object-group network remote-network
network-object 192.168.2.0 255.255.255.0
!
access-list asa-strongswan-vpn extended permit ip object-group local-network object-group remote-networ
!
!Enable IKEv1 on the 'Outside' interface


!
crypto ikev1 enable outside
!
!Configure how ASA identifies itself to the peer


!
crypto isakmp identity address
!
!Configure the IKEv1 policy


!
crypto ikev1 policy 10
authentication pre-share
encryption aes-256
hash sha
group 14
lifetime 3600
!
!Configure the IKEv1 transform-set


!
crypto ipsec ikev1 transform-set tset esp-aes-256 esp-sha-hmac
!
!Configure a crypto map and apply it to outside interface


!
crypto map outside_map 10 match address asa-strongswan-vpn
crypto map outside_map 10 set peer 172.16.0.0
crypto map outside_map 10 set ikev1 transform-set tset
crypto map outside_map 10 set security-association lifetime seconds 28800
crypto map outside_map interface outside
!
!Configure the Tunnel group (LAN-to-LAN connection profile)


!
tunnel-group 172.16.0.0 type ipsec-l2l
tunnel-group 172.16.0.0 ipsec-attributes
ikev1 pre-shared-key cisco
```

```
!
```

Note: An IKEv1 policy match exists when both of the policies from the two peers contain the same authentication, encryption, hash, and Diffie-Hellman parameter values. For IKEv1, the remote peer policy must also specify a lifetime less than or equal to the lifetime in the policy that the initiator sends. If the lifetimes are not identical, then the ASA uses a shorter lifetime. Also, If you do not specify a value for a given policy parameter, the default value is applied.

Note: An ACL for VPN traffic uses the source and destination IP addresses after Network Address Translation (NAT).

NAT Exemption (optional):

Typically, there must be no NAT performed on the VPN traffic. In order to exempt that traffic, you must create an identity NAT rule. The identity NAT rule simply translates an address to the same address.

```
<#root>

nat (inside,outside) source static

local-network local-network

 destination static

remote-network remote-network

 no-proxy-arp route-lookup
```

strongSwan Configuration

On Ubuntu, you would modify these two files with configuration parameters to be used in the IPsec tunnel. You can use your favorite editor to edit them.

/etc/ipsec.conf

/etc/ipsec.secrets

```
<#root>

# /etc/ipsec.conf - strongSwan IPsec configuration file



# basic configuration



config setup
              strictcrlpolicy=no
              uniqueids = yes
              charondebug = "all"
```

VPN to ASA

```
conn vpn-to-asa
              authby=secret
              left=%defaultroute
              leftid=172.16.0.0
              leftsubnet=192.168.2.0/24
              right=10.10.10.10
              rightid=10.10.10.10
              rightsubnet=192.168.1.0/24
              ike=aes256-sha1-modp1536
              esp=aes256-sha1
              keyingtries=%forever
              leftauth=psk
              rightauth=psk
              keyexchange=ikev1
              ikelifetime=1h
              lifetime=8h
              dpddelay=30
              dpdtimeout=120
              dpdaction=restart
              auto=start
```

 # config setup

- Defines general configuration parameters.

 # strictcrlpolicy

- Defines if a fresh CRL must be available in order for the peer authentication based on RSA signatures to succeed.

 # uniqueids

- Defines whether a particular participant ID must be kept unique, with any new IKE_SA using an ID deemed to replace all old ones using that ID.

 # charondebug

- Defines how much charon debugging output must be logged.

 # conn <name>

- Defines a connection.

 # authby -

Defines how the peers must authenticate; acceptable values are secret or psk, pubkey, rsasig, ecdsasig

 # left -

Defines the IP address of the strongSwan's interface paricipating in the tunnel.

 # lefid -

Defines the identity payload for the strongSwan.

 # leftsubnet -

Defines the private subnet behind the strongSwan, expressed as network/netmask.

 # right -

Defines the public IP address of the VPN peer.

rightid -

Defines the identity payload for the VPN peer.

rightsubnet -

Defines the private subnet behind the VPN peer, expressed as network/netmask.

ike -

Defines the IKE/ISAKMP SA encryption/authentication algorithms. You can add a comma-separated list.

esp -

Defines the ESP encryption/authentication algorithms. You can add a comma-separated list.

keyingtries -

Defines the number of attempts that must be made to negotiate a connection.

keyexchange -

Defines the method of key exchange, whether IKEv1 or IKEv2.

ikelifetime -

Defines the duration of an established phase-1 connection.

lifetime -

Defines the duration of an established phase-2 connection.

dpddelay -

Defines the time interval with which R_U_THERE messages/INFORMATIONAL exchanges are sent to the peer. These are only sent if no other traffic is received.

dpdtimeout -

Defines the timeout interval, after which all connections to a peer are deleted in case of inactivity.

dpdaction -

Defines what action needs to be performed on DPD timeout. Takes three values as paramters :

clear

,

hold

, and

restart.

With

clear

the connection is closed with no further actions taken,

hold

installs a trap policy, which catches matching traffic and tries to re-negotiate the connection on demand and

restart

```
 immediately triggers an attempt
to re-negotiate the connection. The default is

none

 which disables the active sending of DPD messages.

# auto -

Defines what operation, if any, must be done automatically at IPsec startup (

start

 loads a connection and brings
it up immediately).
```

<#root>

```
/etc/ipsec.secrets -

This file holds shared secrets or RSA private keys for authentication.



# RSA private key for this host, authenticating it to any other host which knows the public part.



172.16.0.0 10.10.10.10 : PSK "cisco"
```

Useful Commands (strongswan)

Start / Stop / Status:

$ sudo ipsec up <connection-name>

<#root>

```
$ sudo ipsec up vpn-to-asa



generating QUICK_MODE request 656867907 [ HASH SA No ID ID ]
sending packet: from 172.16.0.0[500] to 10.10.10.10[500] (204 bytes)
received packet: from 10.10.10.10[500] to 172.16.0.0[500] (188 bytes)
parsed QUICK_MODE response 656867907 [ HASH SA No ID ID N((24576)) ]
selected proposal: ESP:AES_CBC_256/HMAC_SHA1_96/NO_EXT_SEQ
detected rekeying of CHILD_SA vpn-to-asa{2}
CHILD_SA vpn-to-asa{3} established with SPIs c9080c93_i 3f570a23_o and TS 192.168.2.0/24 === 192.168.1.
connection 'vpn-to-asa' established successfully
```

$ sudo ipsec down <connection-name>

<#root>

$ sudo ipsec down vpn-to-asa

```
generating QUICK_MODE request 656867907 [ HASH SA No ID ID ]
sending packet: from 172.16.0.0[500] to 10.10.10.10[500] (204 bytes)
received packet: from 10.10.10.10[500] to 172.16.0.0[500] (188 bytes)
parsed QUICK_MODE response 656867907 [ HASH SA No ID ID N((24576)) ]
selected proposal: ESP:AES_CBC_256/HMAC_SHA1_96/NO_EXT_SEQ
detected rekeying of CHILD_SA vpn-to-asa{2}
CHILD_SA vpn-to-asa{3} established with SPIs c9080c93_i 3f570a23_o and TS 192.168.2.0/24 === 192.168.1.
connection 'vpn-to-asa' established successfully
anurag@strongswan214:~$ sudo ipsec down vpn-to-asa
closing CHILD_SA vpn-to-asa{3} with SPIs c9080c93_i (0 bytes) 3f570a23_o (0 bytes) and TS 192.168.2.0/2
sending DELETE for ESP CHILD_SA with SPI c9080c93
generating INFORMATIONAL_V1 request 3465984663 [ HASH D ]
sending packet: from 172.16.0.0[500] to 10.10.10.10[500] (76 bytes)
deleting IKE_SA vpn-to-asa[2] between 172.16.0.0[172.16.0.0]...10.10.10.10[10.10.10.10]
sending DELETE for IKE_SA vpn-to-asa[2]
generating INFORMATIONAL_V1 request 2614622058 [ HASH D ]
sending packet: from 172.16.0.0[500] to 10.10.10.10[500] (92 bytes)
IKE_SA [2] closed successfully
```

$ sudo ipsec restart

```
Stopping strongSwan IPsec...
Starting strongSwan 5.8.2 IPsec [starter]...
```

$ sudo ipsec status

```
Security Associations (1 up, 0 connecting):
vpn-to-asa[1]: ESTABLISHED 35 seconds ago, 172.16.0.0[172.16.0.0]...10.10.10.10[10.10.10.10]
vpn-to-asa{1}: REKEYED, TUNNEL, reqid 1, expires in 7 hours
vpn-to-asa{1}: 192.168.2.0/24 === 192.168.1.0/24
vpn-to-asa{2}: INSTALLED, TUNNEL, reqid 1, ESP SPIs: c0d93265_i 599b4d60_o
vpn-to-asa{2}: 192.168.2.0/24 === 192.168.1.0/24
```

$ sudo ipsec statusall

```
Status of IKE charon daemon (strongSwan 5.8.2, Linux 5.4.0-37-generic, x86_64):
uptime: 2 minutes, since Jun 27 07:15:14 2020
malloc: sbrk 2703360, mmap 0, used 694432, free 2008928
worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 3
loaded plugins: charon aesni aes rc2 sha2 sha1 md5 mgf1 random nonce x509 revocation constraints pubkey
Listening IP addresses:
```

```
172.16.0.0
192.168.2.122
Connections:
vpn-to-asa: %any...10.10.10.10 IKEv1, dpddelay=30s
vpn-to-asa: local: [172.16.0.0] uses pre-shared key authentication
vpn-to-asa: remote: [10.10.10.10] uses pre-shared key authentication
vpn-to-asa: child: 192.168.2.0/24 === 192.168.1.0/24 TUNNEL, dpdaction=restart
Security Associations (1 up, 0 connecting):
vpn-to-asa[1]: ESTABLISHED 2 minutes ago, 172.16.0.0[172.16.0.0]...10.10.10.10[10.10.10.10]
vpn-to-asa[1]: IKEv1 SPIs: 57e24d839bf05f95_i* 6a4824492f289747_r, pre-shared key reauthentication in 4
vpn-to-asa[1]: IKE proposal: AES_CBC_256/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1536
vpn-to-asa{2}: INSTALLED, TUNNEL, reqid 1, ESP SPIs: c0d93265_i 599b4d60_o
vpn-to-asa{2}: AES_CBC_256/HMAC_SHA1_96, 0 bytes_i, 0 bytes_o, rekeying in 7 hours
vpn-to-asa{2}: 192.168.2.0/24 === 192.168.1.0/24
```

Get the Policies and States of the IPsec Tunnel:

$ sudo ip xfrm state

```
src 172.16.0.0 dst 10.10.10.10
proto esp spi 0x599b4d60 reqid 1 mode tunnel
replay-window 0 flag af-unspec
auth-trunc hmac(sha1) 0x52c84359280868491a37e966384e4c6db05384c8 96
enc cbc(aes) 0x99e00f0989fec6baa7bd4ea1c7fbefdf37f04153e721a060568629e603e23e7a
anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
src 10.10.10.10 dst 172.16.0.0
proto esp spi 0xc0d93265 reqid 1 mode tunnel
replay-window 32 flag af-unspec
auth-trunc hmac(sha1) 0x374d9654436a4c4fe973a54da044d8814184861e 96
enc cbc(aes) 0xf51a4887281551a246a73c3518d938fd4918928088a54e2abc5253bd2de30fd6
anti-replay context: seq 0x0, oseq 0x0, bitmap 0x00000000
```

$ sudo ip xfrm policy

```
src 192.168.2.0/24 dst 192.168.1.0/24
dir out priority 375423
tmpl src 172.16.0.0 dst 10.10.10.10
proto esp spi 0x599b4d60 reqid 1 mode tunnel
src 192.168.1.0/24 dst 192.168.2.0/24
dir fwd priority 375423
tmpl src 10.10.10.10 dst 172.16.0.0
proto esp reqid 1 mode tunnel
src 192.168.1.0/24 dst 192.168.2.0/24
dir in priority 375423
tmpl src 10.10.10.10 dst 172.16.0.0
proto esp reqid 1 mode tunnel
src 0.0.0.0/0 dst 0.0.0.0/0
socket in priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
socket out priority 0
src 0.0.0.0/0 dst 0.0.0.0/0
socket in priority 0
```

```
src 0.0.0.0/0 dst 0.0.0.0/0
socket out priority 0
src ::/0 dst ::/0
socket in priority 0
src ::/0 dst ::/0
socket out priority 0
src ::/0 dst ::/0
socket in priority 0
src ::/0 dst ::/0
socket out priority 0
```

Reload the secrets, while the service is running:

$ sudo ipsec rereadsecrets

Check if traffic flows through the tunnel:

$ sudo tcpdump esp

```
09:30:27.788533 IP 172.16.0.0 > 10.10.10.10: ESP(spi=0x599b4d60,seq=0x1e45), length 132
09:30:27.788779 IP 172.16.0.0 > 10.10.10.10: ESP(spi=0x599b4d60,seq=0x1e45), length 132
09:30:27.790348 IP 10.10.10.10 > 172.16.0.0: ESP(spi=0xc0d93265,seq=0x11), length 132
09:30:27.790512 IP 10.10.10.10 > 172.16.0.0: ESP(spi=0xc0d93265,seq=0x11), length 132
09:30:28.788946 IP 172.16.0.0 > 10.10.10.10: ESP(spi=0x599b4d60,seq=0x1e46), length 132
09:30:28.789201 IP 172.16.0.0 > 10.10.10.10: ESP(spi=0x599b4d60,seq=0x1e46), length 132
09:30:28.790116 IP 10.10.10.10 > 172.16.0.0: ESP(spi=0xc0d93265,seq=0x12), length 132
09:30:28.790328 IP 10.10.10.10 > 172.16.0.0: ESP(spi=0xc0d93265,seq=0x12), length 132
```

Verify

Before you verify whether the tunnel is up and that it passes the traffic, you must ensure that the traffic of interest is sent towards either the ASA or the strongSwan server.

Note: On the ASA, the packet-tracer tool that matches the traffic of interest can be used in order to initiate the IPSec tunnel (such as packet-tracer input inside tcp 192.168.1.100 12345 192.168.2.200 80 detailed for example).

On ASA

Phase 1 Verification

In order to verify whether IKEv1 Phase 1 is up on the ASA, enter the**show crypto ikev1 sa (**or, **show crypto isakmp sa)**command. The expected output is to see theMM_ACTIVEstate:

<#root>

```
ASAv#
```

```
show crypto ikev1 sa


IKEv1 SAs:

Active SA: 1
Rekey SA: 0 (A tunnel will report 1 Active and 1 Rekey SA during rekey)
Total IKE SA: 1

1 IKE Peer:

172.16.0.0


Type : L2L Role : responder
Rekey : no State :

MM_ACTIVE
```

Phase 2 Verification

In order to verify whether IKEv1 Phase 2 is up on the ASA, enter the **show crypto ipsec sa** command. The expected output is to see both the inbound and outbound Security Parameter Index (SPI). If the traffic passes through the tunnel, you must see the encaps/decaps counters increment.



Note: For each ACL entry there is a separate inbound/outbound SA created, which can result in a long **show crypto ipsec sa** command output (dependent upon the number of ACE entries in the crypto ACL).

```
<#root>

ASAv#

show crypto ipsec sa peer 172.16.0.0


interface:

outside


Crypto map tag: outside_map, seq num: 10, local addr: 10.10.10.10

access-list asa-strongswan-vpn extended permit ip 192.168.1.0 255.255.255.0 192.168.2.0 255.255.255.0
local ident (addr/mask/prot/port): (

192.168.1.0

/255.255.255.0/0/0)
remote ident (addr/mask/prot/port): (

192.168.2.0

/255.255.255.0/0/0)
current_peer:

172.16.0.0
```

```
#

pkts encaps: 37, #pkts encrypt: 37, #pkts digest: 37


#

pkts decaps: 37, #pkts decrypt: 37, #pkts verify: 37


#pkts compressed: 0, #pkts decompressed: 0
#pkts not compressed: 37, #pkts comp failed: 0, #pkts decomp failed: 0
#pre-frag successes: 0, #pre-frag failures: 0, #fragments created: 0
#PMTUs sent: 0, #PMTUs rcvd: 0, #decapsulated frgs needing reassembly: 0
#TFC rcvd: 0, #TFC sent: 0
#Valid ICMP Errors rcvd: 0, #Invalid ICMP Errors rcvd: 0
#send errors: 0, #recv errors: 0

local crypto endpt.: 10.10.10.10/0, remote crypto endpt.:

172.16.0.0

/0
path mtu 1500, ipsec overhead 74(44), media mtu 1500
PMTU time remaining (sec): 0, DF policy: copy-df
ICMP error validation: disabled, TFC packets: disabled

current outbound spi: C8F1BFAB


current inbound spi : 3D64961A


inbound esp sas:
spi: 0x3D64961A (1030002202)
SA State: active
transform: esp-aes-256 esp-sha-hmac no compression
in use settings ={L2L, Tunnel, IKEv1, }
slot: 0, conn_id: 31, crypto-map: outside_map
sa timing: remaining key lifetime (kB/sec): (4373997/27316)
IV size: 16 bytes
replay detection support: Y
Anti replay bitmap:
0x000001FF 0xFFFFFFFF
outbound esp sas:
spi: 0xC8F1BFAB (3371286443)
SA State: active
transform: esp-aes-256 esp-sha-hmac no compression
in use settings ={L2L, Tunnel, IKEv1, }
slot: 0, conn_id: 31, crypto-map: outside_map
sa timing: remaining key lifetime (kB/sec): (4373997/27316)
IV size: 16 bytes
replay detection support: Y
Anti replay bitmap:
0x00000000 0x00000001
```

Alternatively, you can make use of the command **show vpn-sessiondb** to verify the details for both Phases 1

and 2, together.

<#root>

ASAv#

show vpn-sessiondb detail l2l filter ipaddress 172.16.0.0

Session Type: LAN-to-LAN Detailed

Connection :

172.16.0.0

Index : 3 IP Addr : 172.16.0.0
Protocol :

IKEv1 IPsec

Encryption : IKEv1: (1)AES256 IPsec: (1)AES256
Hashing : IKEv1: (1)SHA1 IPsec: (1)SHA1
Bytes Tx : 536548 Bytes Rx : 536592
Login Time : 12:45:14 IST Sat Jun 27 2020
Duration : 1h:51m:57s

IKEv1 Tunnels: 1
IPsec Tunnels: 1

IKEv1:
Tunnel ID : 3.1
UDP Src Port : 500 UDP Dst Port : 500

IKE Neg Mode : Main Auth Mode : preSharedKeys

Encryption : AES256 Hashing : SHA1
Rekey Int (T): 3600 Seconds Rekey Left(T): 2172 Seconds
D/H Group : 14
Filter Name :

IPsec:
Tunnel ID : 3.2

Local Addr : 192.168.1.0/255.255.255.0/0/0

Remote Addr : 192.168.2.0/255.255.255.0/0/0

Encryption : AES256 Hashing : SHA1
Encapsulation: Tunnel
Rekey Int (T): 28800 Seconds Rekey Left(T): 22099 Seconds
Rekey Int (D): 4608000 K-Bytes Rekey Left(D): 4607476 K-Bytes
Idle Time Out: 30 Minutes Idle TO Left : 30 Minutes
Bytes Tx : 536638 Bytes Rx : 536676
Pkts Tx : 6356 Pkts Rx : 6389

On strongSwan

<#root>

#

sudo ipsec statusall

```
Status of IKE charon daemon (strongSwan 5.8.2, Linux 5.4.0-37-generic, x86_64):
uptime: 2 minutes, since Jun 27 07:15:14 2020
malloc: sbrk 2703360, mmap 0, used 694432, free 2008928
worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 3
loaded plugins: charon aesni aes rc2 sha2 sha1 md5 mgf1 random nonce x509 revocation constraints pubkey
Listening IP addresses:
172.16.0.0
192.168.2.122
Connections:
vpn-to-asa: %any...10.10.10.10 IKEv1, dpddelay=30s
vpn-to-asa:
```

local: [172.16.0.0]

```
 uses pre-shared key authentication
vpn-to-asa:
```

remote: [10.10.10.10]

```
 uses pre-shared key authentication
vpn-to-asa:
```

child: 192.168.2.0/24 === 192.168.1.0/24 TUNNEL

```
, dpdaction=restart
Security Associations (1 up, 0 connecting):
vpn-to-asa[1]:
```

ESTABLISHED

```
 2 minutes ago, 172.16.0.0[172.16.0.0]...10.10.10.10[10.10.10.10]
vpn-to-asa[1]: IKEv1 SPIs: 57e24d839bf05f95_i* 6a4824492f289747_r, pre-shared key reauthentication in 4
vpn-to-asa[1]: IKE proposal: AES_CBC_256/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1536
vpn-to-asa{2}:
```

INSTALLED, TUNNEL,

```
 reqid 1, ESP SPIs: c0d93265_i 599b4d60_o
vpn-to-asa{2}: AES_CBC_256/HMAC_SHA1_96, 0 bytes_i, 0 bytes_o, rekeying in 7 hours
vpn-to-asa{2}:
```

192.168.2.0/24 === 192.168.1.0/24

Troubleshoot

ASA Debugs

In order to troubleshoot IPSec IKEv1 tunnel negotiation on an ASA firewall, you can use thesedebugcommands:

Caution: On the ASA, you can set various debug levels; by default, level 1 is used. If you change the debug level, the verbosity of the debugs can increase. In, this case level 127 provides sufficient details to troubleshoot. Do this with caution, especially in production environments.

<#root>

```
debug crypto ipsec 127
debug crypto isakmp 127
debug ike-common 10
```

Note: If there are multiple VPN tunnels on the ASA, it is recommended to use conditional debugs (debug crypto condition peer A.B.C.D), in order to limit the debug outputs to include only the specified peer.

strongSwan Debugs

Ensure charon debug is enabled in ipsec.conf file:

<#root>

```
charondebug = "all"
```

Where the log messages eventually end up depends on how syslog is configured on your system. Common places are /var/log/daemon, /var/log/syslog, or /var/log/messages.

Related Information

- strongSwan User Documentation
- IKEv1/IKEv2 Between Cisco IOS® and strongSwan Configuration Example
- Configure a Site-to-Site IPSec IKEv1 Tunnel Between an ASA and a Cisco IOS® Router