ISS Engineering

# Apriva® MESA VPN Common Criteria Guidance

Version 3.11

July 18, 2023

*Submitted by*

*Apriva ISS, LLC*
*7600 N 16th Street, Suite 230*
*Phoenix, Arizona 85020*



7600 N 16th Street, Suite 230 • Phoenix, Arizona 85020
Phone 480.421.1210 • Fax 480.421.1211

# Notice

## Important Legal Information

THIS DOCUMENT CONTAINS IMPORTANT TERMS AND CONDITIONS REGARDING THE ACCOMPANYING APRIVA PRODUCT. YOUR USE OF THE PRODUCT SHALL IRREVOCABLY INDICATE YOUR ACCEPTANCE OF THE ENCLOSED TERMS AND CONDITIONS. IF YOU DO NOT AGREE TO THE ENCLOSED TERMS AND CONDITIONS, PROMPTLY RETURN THE PRODUCT TO THE PLACE OF PURCHASE FOR A FULL REFUND. ALL RETURNED PRODUCTS MUST BE IN NEW CONDITION, IN THE ORIGINAL PACKAGING.

## Terms and Conditions

This publication is copyrighted by Apriva ISS, LLC and is intended solely for guidance use by Apriva customers of the Apriva MESA VPN product line. This publication may not be reproduced or distributed - in whole or in part - for any other purpose without the written permission of Apriva ISS, LLC.

The information Apriva furnished in this publication is believed to be accurate and reliable. However, Apriva assumes no responsibility for its use, and reserves the right to make changes to the publication at any time without notice. This document applies to Apriva products and supporting software.

## Trademarks

Apriva and Apriva logos are registered trademarks of Apriva. Apriva has attempted throughout this publication to use trademarks correctly and consistently with the wishes of the owners of such trademarks. Any error in identifying any proprietary marks or notices is inadvertent and unintentional.

All product names and services mentioned in this publication are the trademarks, service marks, registered marks, or registered service marks of their respective owners.

## Copyright

# Table of Contents

# Table of Figures

# 1 Introduction

The Apriva MESA VPN is a multi-faceted virtual private network. It provides administration at the local console, remote administration via Secure Shell (SSH), communication with an external audit server that can be protected by TLS and communication with VPN clients that are protected by IPsec. This document describes how to configure the Apriva MESA VPN server in the NIAP Evaluated configuration.

## 1.1 Purpose

This document provides the operational guidance for network engineers to configure and operate the Apriva MESA VPN (Virtual Private Network) Server as evaluated by the National Information Assurance Partnership (NIAP). Additionally, the NIAP evaluation provides assurance that the security requirements outlined by the product's Security Target are satisfied by the Apriva MESA VPN. The Security Target document can be found at the NIAP website (https://www.niap-ccevs.org/Product/index.cfm) and is entitled, "Apriva MESA VPN 3.0 Security Target".

## 1.2 Definitions

To provide a quick reference, the following is a list of acronyms, abbreviations and special terms used within this document.

| Term | Description |
|---|---|
| **CC** | Common Criteria (CC) is a set of guidelines and specifications for evaluating security products, specifically to ensure they meet an agreed-upon security standard for government deployments. |
| **COTS** | Commercial-Off-The-Shelf product (hardware or software), which are adapted aftermarket to the needs of the purchasing organization. |
| **FIPS** | Federal Information Processing Standards (FIPS) is a NIST cryptographic and crypto module standards unit that requires testing and validation in NIST approved labs. |
| **IPsec** | A protocol suite for securing Internet Protocol (IP) communications by authenticating and encrypting each IP packet of a communication session. IPsec also includes protocols for establishing mutual authentication between agents at the beginning of the session and negotiation of cryptographic keys to be used during the session. |
| **NIAP** | National Information Assurance Partnership (NIAP) is a U.S. Government, Common Criteria Evaluation and Validation process that ensures strong security in any validated product. The testing is performed by a small number of Government approved testing labs and the Protection Profiles are extremely comprehensive. |
| **NIC** | Network Interface Controller (NIC, also known as a network interface card, network adapter, LAN adapter, and by similar terms) is a computer hardware component that connects a computer to a computer network. |
| **NIST** | National Institute of Standards and Technology (NIST) is the U.S. Government standards and technology department. |

| | |
|---|---|
| **PFS** | Perfect Forward Secrecy (PFS) will ensure the same key will not be generated again, so forcing a new Diffie-Hellman key exchange. This would ensure if a hacker/criminal was to compromise a private key, they would only be able to access data in transit protected by that key and not any future data, as future data would not be associated with that compromised key. Both client and server require PFS support. |
| **PKI** | Public Key Infrastructure (PKI) describes the standards-based digital certificate, processes and supporting crypto algorithms used by the Apriva MESA VPN server. |
| **Suite B Cryptography** | Suite B is a set of cryptographic algorithms promulgated by the National Security Agency as part of its Cryptographic Modernization Program. It is to serve as an interoperable cryptographic base for both unclassified information and most classified information. Suite B was announced on 16 February 2005. |

| Acronym | Description |
|---|---|
| **ACL** | Access List |
| **AES** | Advanced Encryption Standard |
| **ARP** | Address Resolution Protocol |
| **CA** | Certificate Authority |
| **CBC** | Cipher Block Chaining |
| **CFGMODE** | Uses a push model to push attributes to the IPsec client. |
| **CIDR** | Classless Inter-Domain Routing |
| **CLI** | Command Line Interface |
| **COTS** | Commercial-Off-The-Shelf |
| **CPU** | Central Processing Unit |
| **CRL** | Certificate Revocation List |
| **CRLDP** | CRL Distribution Point |
| **CSfC** | Commercial Solutions for Classified |
| **CSR** | Certificate Signing Request |
| **CTR** | Counter Mode Encryption |
| **DER** | Distinguished Encoding Rules |
| **DH** | Diffie-Hellman |
| **DHCP** | Dynamic Host Configuration Protocol |
| **DMCC** | DoD Mobility Classified Capability |
| **DN** | Distinguished Name |
| **DNS** | Domain Name System |
| **DoS** | Denial-of-Service |

| Acronym | Description |
|---|---|
| **ECC** | Elliptical Curve Cipher |
| **ECDSA** | Elliptic Curve Digital Signature Algorithm |
| **ESP** | Encapsulating Security Payload |
| **FSO** | Facility Security Officer |
| **GRUB** | GRand Unified Bootloader |
| **IANA** | Internet Assigned Numbers Authority |
| **ICMP** | Internet Control Message Protocol |
| **IETF** | Internet Engineering Task Force |
| **IKE** | Internet Key Exchange |
| **KEX** | Key Exchange |
| **KMB** | Keyboard/Monitor/Mouse |
| **MAC** | Message Authentication Code |
| **MESA** | Mobile Environment Security Architecture |
| **MOBIKE** | IKEv2 Mobility and Multihoming |
| **MTU** | Maximum Transmission Unit |
| **NIAP** | National Information Assurance Partnership |
| **NSA** | National Security Agency |
| **NTP** | Network Time Protocol |
| **OCSP** | Online Certificate Status Protocol |
| **OID** | Object Identifier |
| **PEM** | Privacy Enhanced Mail |
| **PFS** | Perfect Forward Secrecy |
| **PKI** | Public Key Infrastructure |
| **RDN** | Relative Distinguished Name |
| **RFC** | Request For Comment |
| **RPM** | RPM Package Manager |
| **RSA** | Rivest, Shamir, and Adelman |
| **SA** | Security Association |
| **SCP** | Secure Copy |
| **SFR** | Security Function Requirement |
| **SSH** | Security Shell |
| **STIG** | Security Technical Implementation Guide |
| **TCP** | Transmission Control Protocol |
| **TLS** | Transport Layer Security |

| Acronym | Description |
|---|---|
| **UDP** | User Datagram Protocol |
| **UNDI** | Unique Device Identifier |
| **UUID** | Universally Unique Identifier |
| **VLAN/LAN** | Virtual Local Area Network/Local Area Network |
| **VPN** | Virtual Private Network |

## 1.3 Physical Platform

The Apriva MESA VPN server Version 3.0 is delivered on COTS, high-performance Dell hardware in a 2 RU rack mountable form factor.
The hardware is as follows:

- Dell PowerEdge R750
- Dual CPU Intel® Xeon® Silver 4310 2.1G processors
- RAM: 16 GB
- NICs:
    - Intel X710-T4L Quad Port 10GbE BASE-T Adapter
    - Broadcom NetXtreme BCM5720 2-port 1GbE
- Storage: Quantity 4, 480GB SSD SATA
- Power Supply: Hot-Plug, Power Supply, 1400W
- External DVD-ROM – Quantity 1, USB

**Operating Environment:**

- MESA OS 3.0
- Vendor proprietary user interface scripts and networking functionality scripts

### 1.3.1 Software Identification

The Apriva MESA VPN server Version 3.0 Build 3.00.1-7 is identified as such when executing the `show system version` command via the system's Command Line Interface (CLI). The command displays the Apriva operating system version number for reference. See the 'Show System Version' command in the 'Software Maintenance Commands' section of this document for more detail.

### 1.3.2 Product Label

The server also includes a label depicting the Apriva MESA VPN server version which will match the detail obtained in the previous step.

**Apriva MESA VPN**
Version 3.0
Build 3.00.1-7

### 1.3.3 Front View

**Figure 1 – Apriva MESA VPN Server: Front View**

### 1.3.4 Top View


**Figure 2 – Apriva MESA VPN Server: Top View**

### 1.3.5 Back View

**Figure 3 – Apriva MESA VPN Server: Back View**

### 1.3.6 Connection Diagram

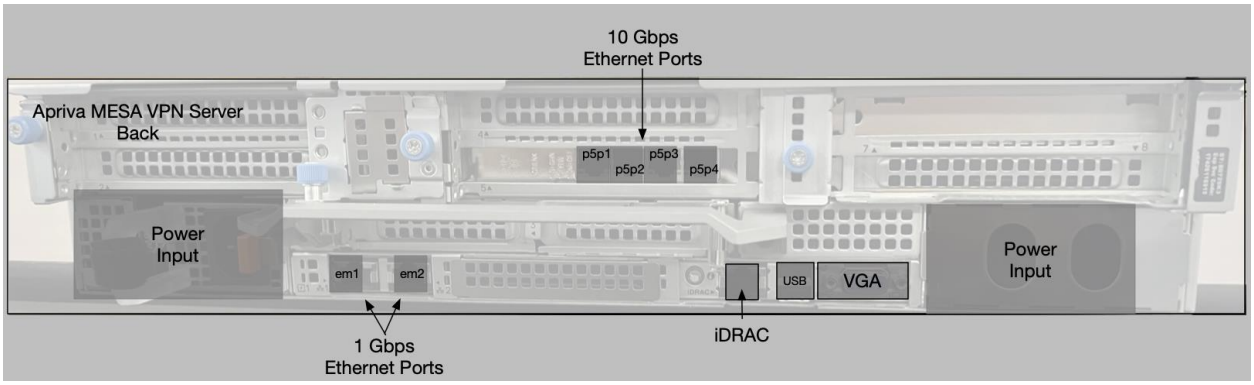



**Figure 4 – Apriva MESA VPN Server Connection Diagram**

## 1.4 Product Acceptance Procedure

The following secure acceptance procedures should be performed by the requisite authority.

- Ensure that the Apriva MESA VPN server is delivered using an insured and tracked commercial courier service.
- Inspect the delivery for signs of tampering. Contact Apriva immediately if signs of tampering are found.
- Verify that the purchase order and packing list contain the hardware name of the product listed on the Security Target document (Apriva VPN Server Security Target v1.0).
- After installation and configuration, login and display the version number using the CLI command show system version and match the displayed version number to the one described by the Security Target document.
- Verify configuration compliance with any organizationally specific requirements e.g., DoD Security Technical Implementation Guide (STIGs). Use the CLI command `show running-config` to display the configuration information.

## 1.5 Physical Installation

Once inspected, the Apriva VPN server must be installed in a physically secure environment:

- Install the Apriva MESA VPN server in a standard 19-inch rack and attach the power cables to the 2 power supplies.
- Attach external, internal and management RJ45 and Fibre interface cables to the NICs on the back of the unit.
- Attach a keyboard/video/mouse (KVM) device or a keyboard and monitor to the VGA and USB ports.

The following diagram shows the locations of each interface.

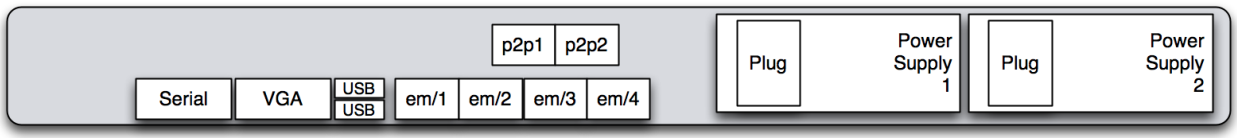



**Figure 5 – Apriva MESA VPN Server Connection Diagram**

Apriva recommends the following interface usage as a typical configuration:

| Interface | Description |
|---|---|
| Serial | Serial interface. Physically present but disabled. |
| VGA | Video signal for administrative console. |
| USB | For administrative console keyboard and mouse |
| Em/1 | RJ45 1GB ethernet port |
| Em/2 | RJ45 1GB ethernet port |
| Em/3 | RJ45 1GB ethernet port |
| Em/4 | RJ45 1GB ethernet port |
| p2p1 | Fiber 1GB ethernet port |
| p2p2 | Fiber 1GB ethernet port |
| Power | Dual power supply (2 plugs) |

Both the USB and VGA ports are used to access the console. The serial port is inactive.

The Apriva MESA VPN server allows the operator to select network interfaces for specific purposes, such as remote administration and VPN service operations.

Apriva strongly recommends that the external (device-facing) network interface be separate and not configured with VLANs, though all other services may be configured to use a single network interface and separate VLANs on that interface.

## 1.6 Cipher Profile Suites

The Apriva MESA VPN server is provided with two strong Cipher Profile Suites (RSA and ECC) for client authentication. If a different Cipher Profile is required for compatibility and testing, the CLI supports the creation of a newly named authentication suite for this purpose. An example might be the use of AES 128 vs. AES 256.

It is suggested that the digital certificates be obtained from the organization's authorized Certificate Authority (CA) in advance if generating a Certificate Signing Request (CSR) is not required. If a CSR is required, then a Superuser level user must use the CLI to generate a CSR. The two scenarios are:

- Generate a CSR on the Apriva MESA VPN server, deliver to the CA and receive the public Apriva MESA VPN server certificate along with the CA Chain certificates for import.

- Receive all certificate material from a CA for import without generating a CSR. PEM file format is supported.

## 1.7 Certificate Signing Request Commands

Establishing and maintaining secure communications with the Apriva MESA VPN is critical. It involves using encryption and other security measures to ensure data is securely transmitted between the Apriva MESA VPN and its clients. Doing so requires use of standard protocols to establish a secure and authenticated communication channel between two parties via the Apriva MESA VPN as well as the use of certificates. The Apriva MESA VPN utilizes both Ipsec and TLS to provide secure communications. The protocols ensure security for Apriva MESA VPN negotiation, remote host, and network access, while the certificates are used for authentication between the Apriva MESA VPN and its peers.

A certificate is a public key certificate, also known as a digital certificate or identity certificate, is an electronic document used to prove the validity of a public key and ensures that the communication between a client computer and a server is secure. The certificate includes information about the key, information about the identity of its owner (called the subject), and the digital signature of an entity that has verified the certificate's contents (called the issuer). If the signature is valid, and the software examining the certificate trusts the issuer, then it can use that key to communicate securely with the certificate's subject. In email encryption, code signing, and e-signature systems, a certificate's subject is typically a person or organization. However, in Transport Layer Security (TLS) a certificate's subject is typically a computer or other device, though TLS certificates may identify organizations or individuals in addition to their core role in identifying devices.

A Certificate Authority (CA) is a trusted entity that issues TLS certificates. These digital certificates are data files used to cryptographically link an entity with a public key.

A Certificate Signing Request (CSR) is an encoded file containing information about the Apriva MESA VPN, service, organization, and domain name. This information is used by a CA to create an X509 certificate specific to the VPN's Ipsec and TLS protocols to encrypt traffic to the VPN. A certificate CSR also contains the public key and signature, which helps to verify the identity and secure communications to the VPN.

The Apriva MESA VN provides commands to manage certificates.

### 1.7.1 Create a Certificate Signing Request Parameter Set

To create a certificate signing request, a CSR parameter set must be created. The parameter set establishes the contents of the CSR.

The following commands are available:

| Command | Description |
|---|---|
| csr-params | Create, modify, or remove a certificate signing request parameter set. |
| algorithm | Set the encryption algorithm to be used for the CSR. |
| common-name | Set the common name (CN) to be used for the CSR's distinguished name. |
| country | Set the country code (C) to be used for the CSR's distinguished name. |

| | |
|---|---|
| email | Set the email address to be used for the CSR's distinguished name. |
| locality | Set the locality (L) to be used for the CSR's distinguished name. |
| organization-name | Set an organization name (O) to be used in the CSR's distinguished name. |
| organization-unit | Set an organization unit (OU) to be used in the CSR's distinguished name. |
| serial | Set the serial number (SN) to be used for the CSR's distinguished name. |
| state | Set the state (ST) to be used for the CSR's distinguished name. |
| subject-alt-name | Set the subject alternative name property for the certificate to be requested. |
| usage | Set the usage type of the certificate to be requested in the CSR. |

The `algorithm`, `common-name`, `country`, `organization-name` and `organization-unit` are required fields. The remaining fields are optional and will not be included if not specified.
Enter the mode to create a certificate signing request parameter set.

### Usage

```
crypto csr-params <name>
```

### Options

| Option | Description | Details |
|---|---|---|
| name | Name of the parameter set. | |
| no | Remove the CSR parameter set. | |

### Examples

```
/localhost(config)# crypto csr-params rsaparams
/localhost(config)# no crypto csr-params rsaparams
/localhost(config)#
```

### Audit Log

The crypto csr-params command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields | |
|---|---|---|
| [<UUID>] User [<user>@<address>] executed command [crypto csr-params <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no crypto csr-params <name>]. | The crypto csr-params command generates an audit log that contains the following fields. | |
| | **Field** | **Description** |
| | UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| | user | This is the name of the account (aka userid) that issued the command. |
| | address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | name | The name of the csr parameter set to be edited. |

## 1.7.2 Set the CSR Parameter Set Encryption Algorithm

Set the encryption algorithm to use for the CSR parameter set. The encryption algorithm is a required field.

### Usage

```
algorithm <alg>
```

**Options**

| Option | Description | Details |
|---|---|---|
| alg | The encryption algorithm to use. | One of [RSA ECDSA] |

**Examples**

```
/localhost(config-csr)# algorithm RSA
/localhost(config-csr)#
```

**Audit Log**

The algorithm command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [algorithm <alg>]. | The algorithm command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| alg | The name of the algorithm to be used in the CSR (RSA or ECDSA). |

### 1.7.3 Set the CSR Parameter Set Common Name

Set the common name to use for the CSR parameter set's distinguished name (DN). The common name is a required field.

**Note:** The system treats the common name as a string and permits the operator to enter any string value. However, NIAP specifies that the common name shall not use an IP address or DNS address. NIAP specifies that any address information shall be included in the subject alternative name field. It is the responsibility of the operator to follow NIAP guidelines.

**Usage**

```
[no] common-name <name>
```

**Options**

| Option | Description | Details |
|---|---|---|
| name | The common name to use. | A literal, case-insensitive alphanumeric string, including special characters [ - _ . ] and spaces. If spaces are used, the string must be encased in quotes. |
| no | Remove the common name. | |

**Examples**

```
/localhost(config-csr)# common-name "John Doe"
/localhost(config-csr)# no common-name
```

```
/localhost(config-csr)#
```

### Audit Log

The common-name command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [common-name <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no common-name]. | The common-name command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the common name of the subject of the CSR. |

## 1.7.4 Set the CSR Parameter Set Country Code

Set the country code to use for the CSR parameter set's distinguished name (DN). The country code is a required field.

### Usage

```
country <code>
```

### Options

| Option | Description | Details |
|---|---|---|
| code | The country code to use. | A two-character country code as defined in ISO 3166-1 alpha-2 (reference http://en.wikipedia.org/wiki/ISO_3166-1_alpha-2 for a list of codes). |
| no | Remove the country code. | |

### Examples

```
/localhost(config-csr)# country US
/localhost(config-csr)# no country
/localhost(config-csr)#
```

### Audit Log

The country command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [country <name>]. | The country command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |

| | |
|---|---|
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the country of the subject of the CSR. |

### 1.7.5 Set the CSR Parameter Set Email Address

Set the email address to use for the CSR parameter set's distinguished name (DN).

**Usage**

```
email <email address>
```

**Options**

| Option | Description | Details |
|---|---|---|
| email | The email address to use. | Format <user@domain> |
| no | Remove the email address. | |

**Examples**

```
/localhost(config-csr)# email im.a.user@somedomain.com
/localhost(config-csr)# no email
/localhost(config-csr)#
```

### Audit Log

The email command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [email <emailaddress>]. | The email command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| emailaddress | The email address to be added into the subject of the certificate in the CSR. |

## 1.7.6 Set the CSR Parameter Set Locality

Set the locality to use for the CSR parameter set's distinguished name (DN). The locality is typically the city or region of a country in which one resides.

### Usage

```
locality <locale>
```

### Options

| Option | Description | Details |
|---|---|---|
| locality | The locale string to use. | A literal, case-insensitive alphanumeric string up to 40 characters, including special characters [ - _ . ] and spaces.<br>If spaces are used, the string must be encased in quotes. |
| no | Remove the locality. | |

### Examples

```
/localhost(config-csr)# locality Phoenix
/localhost(config-csr)# no locality
/localhost(config-csr)#
```

### Audit Log

The locality command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [locality <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no locality]. | The locality command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the locality of the subject of the CSR. |

## 1.7.7 Set a CSR Parameter Set Organization Name

Set an organization name to use for the CSR parameter set's distinguished name (DN). The organization name is a required field.

### Usage

```
organization-name <org-name>
```

### Options

| Option | Description | Details |
|---|---|---|
| org-name | An organization name to use. | Alphanumeric string up to 40 characters, including spaces and special characters [_ - .]. If spaces are used, the string must be encased in quotes. |
| no | Remove the organization name. | |

### Examples

```
/localhost(config-csr)# organization-name Apriva
/localhost(config-csr)# organization-name IBM
/localhost(config-csr)# no organization-name IBM
/localhost(config-csr)#
```

### Audit Log

The organization-name command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [organization-name <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no organization-name]. | The organization-name command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the organization of the subject of the CSR. |

## 1.7.8 Set a CSR Parameter Set Organization Unit

Set an organization unit to use for the CSR parameter set's distinguished name (DN). The organization unit is a required field. Multiple organization units can be specified.

### Usage

```
organization-unit <org-name>
```

### Options

| Option | Description | Details |
|---|---|---|
| org-unit | An organization unit to use. | Alphanumeric string up to 40 characters, including spaces and special characters [_ - .]. If spaces are used, the string must be encased in quotes. |
| no | Remove the organization unit. | The organization unit's value must be specified. |

### Examples

```
/localhost(config-csr)# organization-unit ISS
/localhost(config-csr)# organization-unit CSE
/localhost(config-csr)# no organization-unit CSE
/localhost(config-csr)#
```

### Audit Log

The organization-unit command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [organization-unit <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no organization-unit]. | The organization-unit command generates an audit log that contains the following fields.<br>**Field** — **Description**<br>UUID — This is a UUID number that specifies the session number of the user that issued the command.<br>user — This is the name of the account (aka userid) that issued the command.<br>address — The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from.<br>name — The name of the organization unit of the subject of the CSR. |

## 1.7.9 Set the CSR Parameter Set Serial Number

Set the serial number to use for the CSR parameter set's distinguished name (DN). The serial number is typically used to identify uniqueness between certificates issued to a user and is not the serial number assigned by the Certificate Authority.

### Usage

```
serial <serial-number>
```

### Options

| Option | Description | Details |
|---|---|---|
| serial | The serial number string to use. | Alphanumeric string up to 64 characters |
| no | Remove the serial number. | |

### Examples

```
/localhost(config-csr)# serial 12345
/localhost(config-csr)# no serial
/localhost(config-csr)#
```

### Audit Log

The serial command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [serial <number>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no serial]. | The serial command generates an audit log that contains the following fields.<br>**Field** — **Description**<br>UUID — This is a UUID number that specifies the session number of the user that issued the command.<br>user — This is the name of the account (aka userid) that issued the command.<br>address — The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP |

| Message Format | Description of Message and Fields | |
|---|---|---|
| | | address of the remote computer where the SSH client connected from. |
| | number | The serial number of the subject of the CSR. |

## 1.7.10 Set the CSR Parameter Set State/Province

Set a state or province identifier to use for the CSR parameter set's distinguished name (DN). In the United States this is typically a two-character state abbreviation.

### Usage

```
state <state>
```

### Options

| Option | Description | Details |
|---|---|---|
| state | The state or province string to use. | Alphabetic string up to 40 characters |
| no | Remove the state-or-province. | |

### Examples

```
/localhost(config-csr)# state AZ
/localhost(config-csr)# no state AZ
/localhost(config-csr)# state Arizona
/localhost(config-csr)#
```

### Audit Log

The state command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields | |
|---|---|---|
| [<UUID>] User [<user>@<address>] executed command [state <name>]. | The state command generates an audit log that contains the following fields. | |
| | **Field** | **Description** |
| OR | UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| [<UUID>] User [<user>@<address>] executed command [no state]. | user | This is the name of the account (aka userid) that issued the command. |
| | address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | name | The name of the state of the subject of the CSR. |

## 1.7.11 Set the CSR Parameter Set Subject Alternative Name

Some certificates, such as the Apriva MESA VPN server certificate, require that the subject alternative name property is specified. The server certificate requires that the ingress address of the VPN server is specified in the subject alternative name. The address can be either a DNS name or numeric IP address.

### Usage

```
subject-alt-name <values>
```

### Options

| Option | Description | Details |
|---|---|---|
| values | The subject alternative name values. | This value is in the format <type>:<value> where <type> is one of:<br>[IP, DNS, URI, RID, email].<br>Multiple type/value pairs of types and values are entered in a comma separated format. |
| No | Remove the subject alternative name. | |

### Examples

```
/localhost(config-csr)# subject-alt-name IP:1.2.3.4,IP:3.4.5.6,DNS:vpn.server.com
/localhost(config-csr)# no subject-alt-name
/localhost(config-csr)#
```

### Audit Log

The subject-alt-name command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [subject-alt-name <values>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no subject-alt-name]. | The subject-alt-name command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| values | The values in the subject alternative name property. |

## 1.7.12 Set the CSR Parameter Set Certificate Usage Type

To request a certificate that has the required properties for the certificate to be used successfully, certificate usage must be specified. The system currently supports two certificate usage types (VPN server usage and TLS client usage).

### Usage

```
usage <usage-type>
```

### Options

| Option | Description | Details |
|---|---|---|
| usage-type | The purpose for the certificate being requested. | One of:<br>tls-client (for use as a syslog TLS client)<br>vpn-server (for use as a VPN server certificate) |

### Examples

```
/localhost(config-csr)# usage tls-client
/localhost(config-csr)# usage vpn-server
/localhost(config-csr)#
```

### Audit Log

The state command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [usage <usage-type>]. | The usage command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| usage-type | The usage type specified for the certificate. |

## 1.7.13 Create a Certificate Signing Request (CSR)

Several steps must be taken to properly create a new CSR. These include viewing existing certificates, creating the CSR key, establishing the CSR parameter set, generating the CSR, submitting the CSR to the external Certificate Authority, importing the root certificate, and installing the certificate.

### 1.7.13.1 View Existing Certificates

Prior to creating a new CSR, execute the following command to view existing certificates.

```
/localhost# show crypto certificates
```

### *1.7.13.2* Create the CSR Key

Knowing what certificates already exist, create an internally generated CSR to be signed by an external Certificate Authority by using the crypto key generate command shown below.

**NOTE:** The key is user or organization defined based upon internally defined key naming conventions and rules.

#### Usage

```
crypto key generate <algorithm> <keysize> <name>
```

#### Options

| Option | Description | Details |
|---|---|---|
| algorithm | Key Algorithm | One of [RSA, ECDSA] |
| keysize | Size of the key in bits | One of [2048, 3072] for RSA<br>[256, 384, 521] for ECDSA |
| name | Name of the private key | Alphanumeric character string, excluding the following characters:<br>[ . , ~ / \ $ ! = + & % * ] |

#### Examples

```
/localhost# crypto key generate RSA 2048 myrsakey
```

```
Generated RSA private key 'mrsakey'.
/localhost#
```

### Audit Log

The crypto key generate command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [crypto key generate <algorithm> <keysize> <name>]. | The crypto key generate command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| algorithm | The key algorithm (RSA or ECDSA). |
| keysize | The size of the key in bits. |
| name | The name of the key. |

#### *1.7.13.3 Establish the CSR Parameter Set*

Use the following commands to establish the required CSR parameter set.

Enter the mode to create a certificate signing request parameter set.

### Usage

```
crypto csr-params <name>
```

### Options

| Option | Description | Details |
|---|---|---|
| name | Name of the parameter set. | |
| no | Remove the CSR parameter set. | |

### Examples

```
/localhost(config)# crypto csr-params rsaparams
/localhost(config)# no crypto csr-params rsaparams
/localhost(config)#
```

### Audit Log

The crypto csr-params command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [crypto csr-params <name>].<br><br>OR | The crypto csr-params command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |

| Message Format | Description of Message and Fields | |
|---|---|---|
| [<UUID>] User [<user>@<address>] executed command [no crypto csr-params <name>]. | address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | name | The name of the csr parameter set to be edited. |

Set the encryption algorithm to use for the CSR parameter set. The encryption algorithm is a required field.

### Usage

```
algorithm <alg>
```

### Options

| Option | Description | Details |
|---|---|---|
| alg | The encryption algorithm to use. | One of [RSA ECDSA] |

### Examples

```
/localhost(config-csr)# algorithm RSA
/localhost(config-csr)#
```

### Audit Log

The algorithm command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields | |
|---|---|---|
| [<UUID>] User [<user>@<address>] executed command [algorithm <alg>]. | The algorithm command generates an audit log that contains the following fields. | |
| | **Field** | **Description** |
| | UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| | user | This is the name of the account (aka userid) that issued the command. |
| | address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | alg | The name of the algorithm to be used in the CSR (RSA or ECDSA). |

Set the common name to use for the CSR parameter set's distinguished name (DN). The common name is a required field.

**NOTE**: The system treats the common name as a string and permits the operator to enter any string value. However, NIAP specifies that the common name shall not use an IP address or DNS address. NIAP specifies that any address information shall be included in the subject alternative name field. It is the responsibility of the operator to follow NIAP guidelines.

### Usage

```
[no] common-name <name>
```

### Options

| Option | Description | Details |
|---|---|---|
| name | The common name to use. | A literal, case-insensitive alphanumeric string, including special characters [ - _ . ] and spaces. If spaces are used, the string must be encased in quotes. |
| no | Remove the common name. | |

### Examples

```
/localhost(config-csr)# common-name "John Doe"
/localhost(config-csr)# no common-name
/localhost(config-csr)#
```

### Audit Log

The common-name command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [common-name <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no common-name]. | The common-name command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the common name of the subject of the CSR. |

Set the country code to use for the CSR parameter set's distinguished name (DN). The country code is a required field.

### Usage

```
country <code>
```

### Options

| Option | Description | Details |
|---|---|---|
| code | The country code to use. | A two-character country code as defined in ISO 3166-1 alpha-2 (reference http://en.wikipedia.org/wiki/ISO_3166-1_alpha-2 for a list of codes). |
| no | Remove the country code. | |

### Examples

```
/localhost(config-csr)# country US
/localhost(config-csr)# no country
/localhost(config-csr)#
```

### Audit Log

The country command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|

| | |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [country <name>]. | The country command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the country of the subject of the CSR. |

Set the email address to use for the CSR parameter set's distinguished name (DN).

### Usage

```
email <email address>
```

### Options

| Option | Description | Details |
|---|---|---|
| email | The email address to use. | Format <user@domain> |
| no | Remove the email address. | |

### Examples

```
/localhost(config-csr)# email im.a.user@somedomain.com
/localhost(config-csr)# no email
/localhost(config-csr)#
```

### Audit Log

The email command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [email <emailaddress>]. | The email command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| emailaddress | The email address to be added into the subject of the certificate in the CSR. |

Set the locality to use for the CSR parameter set's distinguished name (DN). The locality is typically the city or region of a country in which one resides.

### Usage

```
locality <locale>
```

### Options

| Option | Description | Details |
|---|---|---|
| locality | The locale string to use. | A literal, case-insensitive alphanumeric string up to 40 characters, including special characters [ - _ . ] and spaces.<br>If spaces are used, the string must be encased in quotes. |
| no | Remove the locality. | |

### Examples

```
/localhost(config-csr)# locality Phoenix
/localhost(config-csr)# no locality
/localhost(config-csr)#
```

### Audit Log

The locality command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [locality <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no locality]. | The locality command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the locality of the subject of the CSR. |

Set an organization name to use for the CSR parameter set's distinguished name (DN). The organization name is a required field.

### Usage

```
organization-name <org-name>
```

### Options

| Option | Description | Details |
|---|---|---|
| org-name | An organization name to use. | Alphanumeric string up to 64 characters, including spaces and special characters<br>[_ - .]. If spaces are used, the string must be encased in quotes. |
| no | Remove the organization name. | |

### Examples

```
/localhost(config-csr)# organization-name Apriva
/localhost(config-csr)# organization-name IBM
/localhost(config-csr)# no organization-name IBM
/localhost(config-csr)#
```

### Audit Log

The organization-name command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [organization-name <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no organization-name]. | The organization-name command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the organization of the subject of the CSR. |

Set an organization unit to use for the CSR parameter set's distinguished name (DN). The organization unit is a required field. Multiple organization units can be specified.

### Usage

```
organization-unit <org-name>
```

### Options

| Option | Description | Details |
|---|---|---|
| org-unit | An organization unit to use. | Alphanumeric string up to 64 characters, including spaces and special characters [_ - .]. If spaces are used, the string must be encased in quotes. |
| no | Remove the organization unit. | The organization unit's value must be specified. |

### Examples

```
/localhost(config-csr)# organization-unit ISS
/localhost(config-csr)# organization-unit CSE
/localhost(config-csr)# no organization-unit CSE
/localhost(config-csr)#
```

### Audit Log

The organization-unit command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [organization-unit <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no organization-unit]. | The organization-unit command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |

| | | |
|---|---|---|
| | name | The name of the organization unit of the subject of the CSR. |

Set the serial number to use for the CSR parameter set's distinguished name (DN). The serial number is typically used to identify uniqueness between certificates issued to a user and is not the serial number assigned by the Certificate Authority.

### Usage

```
serial <serial-number>
```

### Options

| Option | Description | Details |
|---|---|---|
| serial | The serial number string to use. | Alphanumeric string up to 64 characters |
| no | Remove the serial number. | |

### Examples

```
/localhost(config-csr)# serial 12345
/localhost(config-csr)# no serial
/localhost(config-csr)#
```

### Audit Log

The serial command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields | |
|---|---|---|
| [<UUID>] User [<user>@<address>] executed command [serial <number>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no serial]. | The serial command generates an audit log that contains the following fields. | |
| | **Field** | **Description** |
| | UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| | user | This is the name of the account (aka userid) that issued the command. |
| | address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | number | The serial number of the subject of the CSR. |

Set a state or province identifier to use for the CSR parameter set's distinguished name (DN). In the United States this is typically a two-character state abbreviation.

### Usage

```
state <state>
```

### Options

| Option | Description | Details |
|---|---|---|
| state | The state or province string to use. | Alphabetic string up to 40 characters |
| no | Remove the state-or-province. | |

### Examples

```
/localhost(config-csr)# state AZ
/localhost(config-csr)# no state AZ
/localhost(config-csr)# state Arizona
/localhost(config-csr)#
```

### Audit Log

The state command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [state <name>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no state]. | The state command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the state of the subject of the CSR. |

Some certificates, such as the Apriva MESA VPN server certificate, require that the subject alternative name property is specified. The server certificate requires that the ingress address of the VPN server is specified in the subject alternative name. The address can be either a DNS name or numeric IP address.

### Usage

```
subject-alt-name <values>
```

### Options

| Option | Description | Details |
|---|---|---|
| values | The subject alternative name values. | This value is in the format <type>:<value> where <type> is one of:<br>[IP, DNS, URI, RID, email].<br>Multiple type/value pairs of types and values are entered in a comma separated format. |
| No | Remove the subject alternative name. | |

### Examples

```
/localhost(config-csr)# subject-alt-name IP:1.2.3.4,IP:3.4.5.6,DNS:vpn.server.com
/localhost(config-csr)# no subject-alt-name
/localhost(config-csr)#
```

### Audit Log

The subject-alt-name command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [subject-alt-name <values>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no subject-alt-name]. | The subject-alt-name command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console |

| Message Format | Description of Message and Fields | |
|---|---|---|
| | | commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | values | The values in the subject alternative name property. |

To request a certificate that has the required properties for the certificate to be used successfully, the certificate usage must be specified. The system currently supports two certificate usage types (VPN server usage and TLS client usage).

### Usage

```
usage <usage-type>
```

### Options

| Option | Description | Details |
|---|---|---|
| usage-type | The purpose for the certificate being requested. | One of:<br>tls-client (for use as a syslog TLS client)<br>vpn-server (for use as a VPN server certificate) |

### Examples

```
/localhost(config-csr)# usage tls-client
/localhost(config-csr)# usage vpn-server
/localhost(config-csr)#
```

### Audit Log

The state command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields | |
|---|---|---|
| [<UUID>] User [<user>@<address>] executed command [usage <usage-type>]. | The usage command generates an audit log that contains the following fields. | |
| | **Field** | **Description** |
| | UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| | user | This is the name of the account (aka userid) that issued the command. |
| | address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | usage-type | The usage type specified for the certificate. |

Set a domain component (DC) to use for the CSR parameter set's distinguished name (DN). Multiple domain components can be specified. The order of the domain components is significant.

### Usage

```
[no] domain-component <component>
```

### Options

| Option | Description | Details |
|---|---|---|

| | | |
|---|---|---|
| component | A domain component to use. | Alphanumeric string up to 64 characters and special characters [ A-Za-z0-9_\-.]. Spaces will be converted to hyphens. |
| no | Remove the domain component. | The domain component's value must be specified. |

### Examples

```
/localhost(config-csr)# domain-component iss
/localhost(config-csr)# domain-component apriva
/localhost(config-csr)# domain-component com
/localhost(config-csr)# no domain-component iss
/localhost(config-csr)#
```

### Audit Log

The domain-component command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields | |
|---|---|---|
| [<UUID>] User [<user>@<address>] executed command [domain-component <component>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no domain-component <component>]. | The domain-component command generates an audit log that contains the following fields. | |
| | **Field** | **Description** |
| | UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| | user | This is the name of the account (aka userid) that issued the command. |
| | address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | component | The name of the domain component of the subject of the CSR. |

#### *1.7.13.4 Generate the CSR*

Create a Certificate Signing Request (CSR) to be signed by an external Certificate Authority.

### Usage

```
crypto csr generate <hash> <csr-param-set> <key-name>
```

### Options

| Option | Description | Details |
|---|---|---|
| hash | Hash algorithm to use | One of [SHA1, SHA256, SHA384, SHA512] |
| csr-param-set | CSR parameter set to use | |
| key-name | Private key to use | |

**NOTE:** The CSR data is displayed directly to the console and is to be copy/pasted from the console session to be sent to an external Certificate Authority.

The example given is not a valid CSR; its certificate values have been changed.

### Examples

```
/localhost# crypto csr generate sha256 rsaparams testkey
Certificate Request:
    Data:
        Version: 0 (0x0)
```

```
        Subject: CN=John Doe, C=US, ST=Arizona, L=Phoenix, O=Apriva, OU=ISS,
SN=12345/emailAddress=im.a.user@somedomain.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:b5:f8:c7:80:32:22:de:e8:c1:64:1e:7d:da:46:
                    db:42:23:98:5d:d1:6d:f9:80:ec:2e:43:4b:0f:9b:
                    41:05:42:38:6e:38:ab:d7:8f:56:3c:d8:2e:75:41:
                    69:76:d9:dc:e8:12:89:8f:62:06:ec:15:81:9d:1e:
                    bf:1b:dd:88:cb:56:0c:a6:4d:22:5c:a7:85:85:38:
                    e0:ff:9d:93:b8:81:64:73:ed:2c:57:3e:5d:69:da:
                    10:3e:11:91:71:8f:81:a8:e4:20:84:ea:7e:62:24:
                    83:21:44:58:39:e3:59:eb:d8:96:d2:7d:11:ce:ba:
                    ca:16:4e:ba:3e:31:19:20:6f:49:f7:d2:96:bb:dc:
                    67:71:f2:a5:55:c2:bd:05:9f:25:a1:76:c7:75:d6:
                    4d:80:88:85:13:f6:42:6a:c6:b3:ff:95:ef:c0:8e:
                    09:ba:fe:7e:0e:f4:82:e9:94:19:12:4f:b1:38:fe:
                    cb:e7:a1:b3:4b:ae:81:f0:85:63:ce:99:a5:3a:61:
                    cc:81:c2:1f:65:32:ab:72:98:f3:6b:e3:33:c5:8d:
                    2b:2c:74:bf:03:ea:1d:80:bd:2f:8d:38:9c:ec:2b:
                    f9:35:07:3f:ca:0d:25:5c:ad:e2:7c:9a:95:c0:78:
                    44:6d:9e:a4:6b:c9:4c:55:c1:58:82:3e:86:48:47:
                    ab:cf
                Exponent: 65537 (0x10001)
        Attributes:
            a0:00
    Signature Algorithm: sha256WithRSAEncryption
         46:91:8b:4f:71:6e:37:24:d4:c7:00:b4:d2:44:0c:3b:2b:c1:
         f6:1b:08:e5:ce:c2:18:c2:c6:65:9a:a1:af:c5:6e:82:a1:2f:
         b8:28:8e:67:c8:c3:37:47:b3:ef:2c:cb:26:2b:38:93:e9:29:
         f8:b0:06:a8:da:ce:a4:54:6b:45:38:02:4a:b5:d6:28:45:64:
         b6:23:bf:28:58:db:59:16:fa:78:1a:c3:0c:74:21:16:af:77:
         c6:e1:b2:80:85:40:99:24:56:66:a0:80:93:fb:f5:75:ea:97:
         7c:85:90:14:5f:68:db:74:1d:2f:3a:cc:89:a5:8b:7e:95:c3:
         6f:62:43:8e:f8:c8:d6:60:c4:14:f7:22:3f:4b:22:bf:d9:2a:
         ed:22:de:4d:d6:25:fc:d8:ac:6b:6b:6a:a1:b7:5c:0f:34:27:
         1f:38:aa:56:00:0f:d6:4e:2a:98:11:58:33:ba:9f:be:2d:01:
         8a:25:5a:1a:e9:5c:37:55:82:dd:75:25:2d:64:06:2b:98:c1:
         92:99:76:85:c0:36:7f:a7:98:73:0f:da:8a:7a:b2:1b:b3:da:
         2e:df:10:b7:f9:19:ce:53:99:4c:34:87:94:50:7d:e6:1d:c5:
         a6:55:a7:ac:44:0e:ba:f1:ef:d2:ce:df:08:1b:7f:2e:dd:1d:
         bb:40:94:2c
-----BEGIN CERTIFICATE REQUEST-----
MIICmTCCAYECAQAwVDELMAkGA1UEBhMCVVMxEDAOBgNVBAgMB0FyaXpvbmExEDAO
BgNVBAcMB1Bob2VuaXgxETAPBgNVBAoMCE1pbGl0YXJ5MQ4wDAYDVQQLDAVBbHBo
YTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALX4x4AyIt7owWQefdpG
20IsmF3RbfmA7C5DSw+bQQVCOG44q9ePVjzYLnVBaXbZ3OgSiY9iBuwVgZ0evxva
iMtWDKZNIlynhYU44P+dk7iBZHPtLFc+XWnaED4RkXGPgajkIITqfmIkgyNEWDnj
WevYltJ9Ec66yhZOuj4xGSBvSffSlrvcZ3HypVXCvQWfJaF2x3XWTYCIhRP2QmrG
s/+V78COCbrefg70gumUGRJPsTj+y+ehw0uugfCFY86ZpTphzIHCH2Uyq3KY82vj
M8WNKyx0vwPqHYC9L404nOwr+TUHP8oNJVyt4nyalcB4RG2epGvJTFXBWII+hkhH
q88CAwEAAaAAMA0GCSqGSIb3DQEBCwUAA4IBAQBGkYtPcW43JNTHALTSRAw7K8H2
GwjlzsIYwsZlmqGvxW6CoS+4KI+/V16pd8hZAUX2jbdB0vOsyJpYt+lcNvYkuO+M
5nyMM3R7PvLMsmKziT6Sn4sAao2s6kVGtFOAJKtdYoRWS2I78oWNtZFvp4GsMMdC
EWr3fG4bKAhUCZJFZmoICT jWYMQU9yI/SyK/2SrtIt5N1iX82Kxra2qht1wPNCc
fOKpmAA/WTiqYEVgzup++LQGKJVoa6Vw3VYLddSUtZAYrmMGSmXaFwDF/p5hzD9q
KerIbs9ou3xC3+RnOw5lMNIeUUH3mHcWmVaesRA668e/Szt8IG38u3R27QJQs
-----END CERTIFICATE REQUEST-----
```

### Audit Log

The crypto csr generate command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [crypto csr generate <hash> <csr-param-set> <key-name>]. | The crypto csr generate command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| hash | The hash algorithm to be used in the CSR. |
| csr-param-set | The CSR parameter set to be used for creating the CSR. |
| key-name | The name of key to be included in the CSR. |

#### *1.7.13.5 Submit the CSR to the External Certificate Authority*

Send the CSR via email to the external CA Certificate authority and they return a file containing the certificate.

The exact format of the certificates that are generated by the CA will depend on the Certificate Authority software. It may return a PKCS#7, DER, or PEM format file. It is the responsibility of the user to convert the certificates into PEM format before importing them into the Apriva MESA VPN server.

#### *1.7.13.6 Import the Root CA Certificate and Create a Trust Chain via Cut/Paste*

Install a certificate authority (CA) certificate by cutting and pasting from a file to the session console and assign to a trust chain. If the trust chain does not exist, it will be created.

### Usage

```
crypto import cacert <name> <trustchain>
```

### Options

| Option | Description | Details |
|---|---|---|
| name | Name to apply as the CA certificate | File source must be in DER format |
| trustchain | Name for the trust chain | The name of the trust chain to which the CA is to be added. If a root CA certificate is being installed, then this name is the name to be used for the new trust chain. |

### Examples

```
/localhost(config)#crypto import cacert myrootca mytlschain
Paste the cacert.
Enter a line with a single period ('.') at the end.>>
-----BEGIN CERTIFICATE-----
```

```
[encrypted certificate body]
-----END CERTIFICATE-----
.

//localhost(config)#crypto import cacert myinterca mytlschain
Paste the cacert.
Enter a line with a single period ('.') at the end.>>
-----BEGIN CERTIFICATE-----
[encrypted certificate body]
-----END CERTIFICATE-----
.

/localhost(config)#crypto import cacert mysubca mytlschain
Paste the cacert.
Enter a line with a single period ('.') at the end.>>
-----BEGIN CERTIFICATE-----
[encrypted certificate body]
-----END CERTIFICATE-----
.
```

### Audit Log

The crypto import cacert command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [crypto import cacert <name> <trustchain>]. | The crypto import cacert command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the certificate authority to be imported. |
| trustchain | The name of the trustchain for the certificate authority. |

#### *1.7.13.7 Install the Certificate (Cut/Paste)*

Install the certificate to be used for VPN device authentication and assign to a trust chain.

### Usage

```
import certificate <name> <trustchain>[<key>]
```

### Options

| Option | Description | Details |
|---|---|---|
| name | Name of the certificate | The certificate may be one of two formats:<br>1. A list of PEM encoded certificates and optionally the PEM encoded associated private key.<br>2. A PKCS#12 file converted to PEM format, which may include CA certificates and a private key. |

| | | |
|---|---|---|
| | | For CSR generated certificates, the private key is not included. |
| trustchain | Name of the trust chain | |
| key | The name of the private key associated when importing a certificate that was generated using a CSR. | The name of the private key that was associated with the CSR. This parameter only applies when importing a CSR certificate. |

## Examples

```
/localhost(config-vpnauth)# import certificate myvpncert myvpnchain
Paste the certificate.
Enter a line with a single period ('.') at the end.>>-----BEGIN ENCRYPTED PRIVATE KEY-
----
>>[encrypted key body]
>>-----END ENCRYPTED PRIVATE KEY-----
>>-----BEGIN CERTIFICATE-----
>>[encrypted certificate body]
>>-----END CERTIFICATE-----
>>.
PKCS12 Password: [pem passphrase]
/localhost(config-vpnauth)#

/localhost(config-vpnauth)# import certificate myvpncert myvpnchain mykey
Paste the certificate.
Enter a line with a single period ('.') at the end.>>>>-----BEGIN CERTIFICATE-----
>>[encrypted certificate body]
>>-----END CERTIFICATE-----
>>.
PKCS12 Password: [passphrase]
/localhost(config-vpnauth)#
```

## Audit Log

The import certificate command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [import certificate <name> <trustchain><key >]. | The import certificate command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the certificate to import. |
| trustchain | The name of the trust chain to import into or add. |
| key | The name of the private key associated with the certificate if this applies. |

## 1.7.14 Deleting Certificates

### *1.7.14.1 Delete a CA Certificate from a Trust Chain*

Delete a CA certificate from a trust chain. If the trust chain that contained this CA certificate is empty, it will be deleted. All CA certificates that are subordinate to the CA certificate being deleted are also removed.

**Usage**

```
delete crypto cacert <name>
```

**Options**

| Option | Description | Details |
|---|---|---|
| name | Name of the certificate | |

**NOTE:** When a CA certificate is removed, any other CAs or user certificates that were issued by the removed CA become invalid and will be removed. This process will continue until no certificates exist that are dependent on removed certificates.

**Examples**

```
/localhost# delete crypto cacert TestCA
/localhost#
```

**Audit Log**

The delete crypto cacert command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [delete crypto cacert <name>]. | The delete crypto cacert command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the certificate authority to be deleted. |

### *1.7.14.2 Delete a Certificate and Remove it from its Trust Chain*

Delete a certificate and remove it from its trust chain.

**Usage**

```
delete crypto certificate <name>
```

**Options**

| Option | Description | Details |
|---|---|---|
| name | Name of the certificate | |

### Examples

```
/localhost# delete crypto certificate tlscert
/localhost#
```

### Audit Log

The delete crypto certificate command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [delete crypto certificate <name>]. | The delete crypto certificate command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the certificate to be deleted. |

#### *1.7.14.3 Delete a System Private Key*

Delete a system private key.

### Usage

```
delete crypto key <name>
```

### Options

| Option | Description | Details |
|---|---|---|
| name | Name of the key to delete | |

### Examples

```
/localhost# delete crypto key testkey
Deleted private key 'testkey'.
/localhost#
```

### Audit Log

The delete crypto key command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [delete crypto key <name>]. | The delete crypto key command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For |

| | |
|---|---|
| | remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| name | The name of the key to be deleted. |

# 2 Procedure for Setup and Configuration

## 2.1 Configuration Steps

The following steps must be completed in sequence to configure the Apriva MESA VPN server to provide basic operations. Some steps are listed as optional. While not required for the server to operate, these steps should be executed in accordance with organization policy.

1. Login to the Server as Root (see section 2.1.2).
2. Change the Root User Password (see section 2.1.3).
3. Change the System Boot Password (see section 2.1.4).
4. Create Superuser Users (see section 2.1.5).
5. Log out and re-login as a Superuser (see section 2.1.6).
6. Modify CLI session timeout. (see section 2.1.7).
7. Configure the Management Network Interface (see section 2.1.8).
8. Configure the SSH service (see section 2.1.9).

   **NOTE:** The operator may now choose to transition to a remote shell for the remainder of these steps.

9. Set the System Host Name (optional) (see section 2.1.10).
10. Set the Login Banner (section 2.1.11).
11. Set Password Policies (optional) (section 2.1.12).
12. Configure the Internal and External Network Interfaces (section 2.1.13).
13. Configure System Logging (section 2.1.14).
    a. Syslog Server Requirements (section 2.1.14.1).
14. Set system date/time (section 2.1.15).
15. Configure the NTP Service (section 2.1.16).
16. Configure DNS (optional) (section 2.1.17).
17. Configure Distinguished Name Match Rules (optional) (section 2.1.18).
18. Set Access Policies (section 2.1.19).
19. Configure local redundancy settings (optional).
20. Create VPN Authentication Suites (section 2.1.20).
21. Install VPN Trust Chains and Certificates/Keys (section 2.1.21).
    a. Import VPN Trust Chains and Certificates/Keys (section 2.1.21.1).
    b. Syslog (section 2.1.21.2).
22. Configure Global VPN Settings (section 2.1.22).
23. Add Default Gateway and IP Routes (as required) (section 2.1.23).

24. Add Devices to the VPN Server (section 2.1.24).

Once these steps are completed, the Apriva MESA VPN server should be operational and able to accept connections from configured devices.

Once VPN connections can be made successfully, the backend network should be verified if services outside the Apriva MESA VPN server are to be accessed.

### 2.1.1 Initial Login

Configuring the system requires the administrator to complete an initial log in via the CLI. This initial log in is performed at the system console. To use the system console, attach a VGA monitor and a USB keyboard to the ports on the back of the Apriva MESA VPN server and use them to login to the system.

The system is shipped with a single **root** user and the root user password must be set before the system can be configured. The root user can only log in from the local (terminal) console and after initial configuration is intended for use only in a "break glass" (emergency) scenario where all other means of access to the Apriva MESA VPN server are unavailable.

During initial configuration, the user is responsible for generating a strong password per their organization's security requirements. Once a password policy is created, it will apply to future password changes. This password should be stored in a secure location.

After setting the root user password, the following additional initial configuration items are required before configuration can begin:

- Change the system boot password.
- Using the CLI, create one or more **Superuser** (superuser).
- Log off and log back in as a **Superuser** user.

All operations from this point forward will be associated with a person for proper audit logging. **DO NOT** use the root user for anything other than "break-glass" situations and **DO** reset the root user password after a "break-glass" situation is resolved.

Once logged in as a Superuser, configuration can begin. It is suggested that additional lower privilege level users be created, followed by configuration steps (e.g., network and route configuration, access policy configuration, CSR generation, certificate imports, etc.).

Please see Apriva MESA VPN Users Guide for details to finalize a typical Apriva MESA VPN server configuration.

### 2.1.2 Login to the Server as Root

The Apriva MESA VPN server ships in a factory default configuration with a root user and default password, accessible from a serial console. The default password will be provided in a separate document.

At the console:

```
localhost login: root
Password: My_password
/localhost>
```

**NOTE:** Passwords are not logged to any audit log and are not visible when using SSH per NIAP policy.

### 2.1.3 Change the System Boot Password

Before the system can be configured, the system boot password must be changed.

The password should be a strong password, containing mixed case letters, numbers, and special characters, and be not less than 15 characters. Because the system password policy has not yet been set, the Superuser is responsible for meeting password complexity requirements. At the console:

```
/localhost(init)> boot-passwd
Password:
Confirm:
Boot password updated successfully.
/localhost(init)>
```

### 2.1.4 Change the Root User Password

Before the system can be configured, the root user's password must be changed.

The password should be a strong password, containing mixed case letters, numbers, and special characters, and be not less than 15 characters. Because the system password policy has not yet been set, the Superuser is responsible for meeting password complexity requirements.

At the console:

```
/localhost(init)> passwd
Password:
Confirm:
Password updated successfully.
/localhost(init)>
```

**NOTE:** The root password is only accessible through the system console and its default is set to never expire.

**NOTE:** Passwords are never logged to any audit log and are unavailable when using SSH per NIAP policy.

### 2.1.5 Create Superuser Users

Once passwords are set, users can be created. The **root** user is not intended to be used for any system configuration; rather, it is a "break glass" user to allow for access at the serial console if all other means of accessing the server have failed.

At least one Superuser (superuser) user must be created for system configuration and each administrative user should have their own user account and password. The password for each Superuser should be a strong password, containing mixed case letters, numbers, and special characters, and be not less than 8 and no more than 64 characters. The Superuser is responsible

for meeting password complexity requirements because the system password policy has not yet been set.

Once a Superuser user has been created, that user will complete the configuration including the addition of lower privilege users. Additional users can be added now or later using the root account or any Superuser account using steps like those that follow.

```
/localhost> enable
/localhost# username <username> privilege superuser
Password:
Confirm:
User successfully added.
/localhost#
```

Password composition is as follows:

- The minimum length is defaulted to 12 characters but can be changed as needed.
- The following characters: [A-Z], [a-z], [0-9], spaces and special characters are permissible [!@$%^&*"#'+,-./:;<=>?[,\]_`{|}()~].

**NOTE:**
If all superuser (administrative) accounts are locked out due to failed login attempts, the system can still be accessed via the console using the root account and the locked accounts can be reset from there as well. Since the root account is only accessible from the console, it is protected from remote attacks via SSH.

Utilize the following commands to change the password for a user account.

```
/localhost> enable
/localhost# passwd <username>
Password:
Confirm:
Password updated successfully.
/localhost#
```

Normal administrative and user accounts can be created, and their passwords set later using the above examples.

### 2.1.6 Re-login as the Superuser User

Once the Superuser user has been created, log out of the root account and log back in as the Superuser. All commands will be logged as being executed by this user.

```
/localhost# logout
Logging out...
login: <superuser>
Password:
/localhost> exit
/localhost>
```

**NOTE:** A user with a CLI session (either using the local console or using a remote SSH connection) can terminate their interactive session using the “logout” command.

### 2.1.7 CLI Session Timeout

Local console CLI sessions can timeout due to user inactivity. The default timeout is 600 seconds and can be changed to a maximum of 999999 seconds to satisfy operator requirements. It can also be changed to a zero value. Sample commands to implement no time timeout and one that is 999999 seconds are shown immediately below.

```
[no] console-timeout <seconds>

console-timeout 999999
```

**NOTE:** The SSH timeout value is also defaulted to 600 seconds and can be separately adjusted to meet environmental needs. Refer to the section entitled “Configure the SSH Service” where the SSH timeout value can be specified.

### 2.1.8 Configure the Management Network Interface

Configuring the management network interface allows for remote access to the Apriva MESA VPN server, removing dependency on the system console.

**NOTE:** The system installs a default access policy that allows SSH connectivity on the management network interface. This policy can be modified or replaced as desired but must be assigned to the network interface to be used. Regardless, an access policy must be defined that allows communications on TCP port 22 for remote access to operate.

Before committing network configuration on the first interface (em1) on initial configuration, down the first interface as follows:

```
ifdown em1
```

If the interface is not taken down before the commit, the network restart will fail, and the system will roll back to the previous configuration.

```
/localhost> enable
/localhost# configure
/localhost(config)# interface em1
/localhost(config)# ifdown em1
/localhost(config-iface)# ip address <address/CIDR>
/localhost(config-iface)# mtu 1450
/localhost(config-iface)# access-list manage-input in
/localhost(config-iface)# access-list manage-output out
/localhost(config-iface)# commit
Restarting network services... (Remote connections will pause...)
Network restart complete. Restart complete.
Stopping service 'iptables': SUCCESS!
Starting service 'iptables': SUCCESS!Changes committed.
/localhost(config-iface)# exit
/localhost(config)#
```

### 2.1.9 Set Authentication Lockout Policy

Parameters are restricted in NIAP usage.
The [no] option of this command is outside the scope of NIAP testing. For NIAP configurations this setting must remain enabled and have a value above zero. The valid range of values accepted for the maximum number of failed attempts is between 2 and 10.

Set the maximum number of failed authentication attempts before a user account is locked.

**Usage**

```
[no] max-fail-attempts <count>
```

The following example is used to define the maximum number of successive failed login attempts to 3 for all user accounts. The failed login account will then be subsequently locked after the 3rd failed login attempt and that account will need to be reset via the Superuser account to permit a valid login. If the account is locked an SSH login for the locked account will fail, even at the system console.
Note: As previously stated, the root account is never locked out and is only available for use at the console.

```
/localhost(config)# authentication-policy
/localhost(config-authpol)# max-failed-attempts 3
/localhost(config-authpol)# exit
/localhost(config)# commit
Changes committed.
/localhost(config)#
```

### 2.1.10 Configure the SSH Service

Before using remote access via SSH, it is strongly recommended to configure the service timeout and authentication failure policy. The following commands are examples only and the operator is responsible for defining the values based on organization policy.

This example defines the following items:

- The ssh-server command places the user in the ssh command scope.
- The interface command specifies the network interface SSH will be available (em1).
- The no disable command enables the ssh to run.
- The login-grace command specifies the amount of time the user is allowed to complete their login.
- The timeout specifies the idle time in seconds before the user is automatically logged out.
- The ciphers command specifies the only cipher algorithms available for use as per NIAP certification (aes128-ctr and aes256-ctr).
- The commit command applies the previously entered changes.

```
/localhost(config)# ssh-server
/localhost(config-sshd)# interface em1
/localhost(config-sshd)# no disable
/localhost(config-sshd)# login-grace 60
/localhost(config-sshd)# timeout 600
/localhost(config-sshd)# ciphers aes128-ctr aes256-ctr
/localhost(config-sshd)# commit
Stopping service 'sshd': SUCCESS!
Starting service 'sshd': SUCCESS!
Changes committed.
```

```
/localhost(config-sshd)# exit
```

The Apriva MESA VPN public HostKey is automatically generated during system generation. There are, however, two commands that can be used to manage the SSH HostKey. The Hostkey Show command displays the current public HostKey value, whereas the HostKey Generate command replaces the existing public Hostkey with a new generated key.

### Usage

```
hostkey show
```

### Options

No options.

### Example

```
/localhost> hostkey show
ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAI8mlzdHAyNTYAAABBB5HkVpG9aQEX0LbRv7OMcjzKnA6NZc6+fs
uLa2Hi+hhd52X/zo8fUR2PA3ICbDf14XSSP0G01eXMJuqTAb3qs6g=
/localhost>
```

### Usage

```
hostkey generate <keysize>
```

### Options

| Option | Description | Details |
|---|---|---|
| keysize | Size of the key to generate in bits. | One of the following ECC key sizes: 256, 384, 521 |

### Example

```
/localhost> hostkey generate 384
Hostkey regenerated
/localhost>
```

The following items are supported by default and are not configurable within the Apriva MESA VPN.

- Key exchanges are hard coded to ECDH-SHA2-NISTP256, ECDH-SHA2-NISTP384, and ECDH-SHA2-NISTP521.
- MAC algorithms – HMAC-SHA2-256 and HMAC-SHA2-512.
- Public key algorithms – ECDSA-SHA2-NISTP256, ECDSA-SHA2-NISTP384, and ECDSA-SHA2-NISTP521.
- Rekey limits are set to 512 megabytes and 59 minutes.

The Apriva MESA VPN utilizes the NIAP evaluated configuration for TLS, syslog, and VPN sessions.

### 2.1.11 Set the System Host Name (optional)

The Apriva MESA VPN server's hostname is set by default to 'localhost'. This optional step sets the Apriva MESA VPN server's hostname and is recommended.

```
/localhost(config)# hostname NIAPVPN01.apriva.com
/localhost(config)# commit
Changes committed.
/NIAPVPN01.apriva.com(config)#
```

### 2.1.12 Set the Login Banner

A login banner is required on all United States government and similar type organization systems to provide a definitive warning to any possible intruders that may want to access the system that certain types of activity are illegal, but at the same time, also advise authorized and legitimate users of their obligations relating to acceptable use of the computerized or networked environment(s). This banner is displayed on the local console and all remote SSH logins.

The Apriva banner is shown below as an example and can be modified to address local requirements.

> You are accessing a U.S. Government (USG) Information System (IS) that is provided for USG-authorized use only. By using this IS (which includes any device attached to this IS), you consent to the following conditions: -The USG routinely intercepts and monitors communications on this IS for purposes including, but not limited to, penetration testing, COMSEC monitoring, network operations and defense, personnel misconduct (PM), law enforcement (LE), and counterintelligence (CI) investigations. -At any time, the USG may inspect and seize data stored on this IS. -Communications using, or data stored on, this IS are not private, are subject to routine monitoring, interception, and search, and may be disclosed or used for any USG-authorized purpose. -This IS includes security measures (e.g., authentication and access controls) to protect USG interests -- not for your personal benefit or privacy. -Notwithstanding the above, using this IS does not constitute consent to PM, LE or CI investigative searching or monitoring of the content of privileged communications, or work product, related to personal representation or services by attorneys, psychotherapists, or clergy, and their assistants. Such communications and work product are private and confidential. See User Agreement for details.

The system, by default, is configured without a login banner. This optional step shows how to create the Apriva MESA VPN server's login banner. The banner content implemented should conform to organization policy.

```
/NIAPVPN01.apriva.com(config)# banner
/NIAPVPN01.apriva.com(config-banner) # login
Please enter one or more lines.
Enter a line with a single period ('.') to end.
>> This is a login banner.
>> .
/localhost(config-banner)# commit
Changes committed.
/NIAPVPN01.apriva.com(config-banner)# show run banner
banner {
   login ^ This is a login banner.
}
/NIAPVPN01.apriva.com(config-banner)# exit
```

```
/NIAPVPN01.apriva.com(config)#
```

### 2.1.13 Set Password Policies (optional)

The system is configured without default password policies. This optional step sets the rules for the Apriva MESA VPN servers' user password selection. The password policy contents should conform to organizational policy.

There are three password policies that can be applied:

- 'Min-length' option allows the minimum length (size) allowed for passwords.
- 'Require mixed-case' option sets a mixed-case constraint on user passwords.
- 'Require special-character' option sets a use of special characters constraint on user passwords.

Password policies can be updated or removed.

```
/NIAPVPN01.apriva.com(config)# password-policy
/NIAPVPN01.apriva.com(config-passpol)# min-length 15
/NIAPVPN01.apriva.com(config-passpol)# require special-character
/NIAPVPN01.apriva.com(config-passpol)# require mixed-case
/NIAPVPN01.apriva.com(config-passpol)# commit
Changes committed.
/NIAPVPN01.apriva.com(config-passpol)# exit
/NIAPVPN01.apriva.com(config)#
```

### 2.1.14 Configure the Internal and External Network Interfaces

For the Apriva MESA VPN server to communicate with devices, and to forward and receive packets from other network destinations, the external (device-facing) and internal (network-facing) interfaces must be configured. The IP addresses and subnet masks must conform to the network infrastructure. These addresses are used when the Apriva MESA VPN server is not running the VPN service, to allow the interfaces to be tested but not pass VPN device traffic.

**NOTE:** An access policy must be defined to allow UDP ports 500 and 4500 to traverse the external network interface for devices to connect to the Apriva MESA VPN server.

```
/NIAPVPN01.apriva.com(config)# interface em4
/NIAPVPN01.apriva.com(config-iface)# ip address <IP address/CIDR>
/NIAPVPN01.apriva.com(config-iface)# exit
/NIAPVPN01.apriva.com(config)# interface em1
/NIAPVPN01.apriva.com(config-iface)# ip address <IP address/CIDR>
/NIAPVPN01.apriva.com(config-iface)# exit
/NIAPVPN01.apriva.com(config)# commit
Changes committed.
/NIAPVPN01.apriva.com(config)#
```

### 2.1.15 Configure System Logging

The Apriva MESA VPN server logs all user activities as well as providing logs for internal diagnostics and statistics. Logs are periodically archived based on configuration. In addition, a remote log destination may be configured. Due to how syslog performs TLS mutual authentication, the network destination hostname must be resolvable from the internal network of the Apriva MESA VPN server. If DNS is not resolvable from the internal network interface, then a host entry must be added for the destination server using the 'ip host' command at the config prompt.

**NOTE:** In order for logs to be sent to a log destination, an access policy must be defined to allow log traffic on UDP port 514 (plaintext) or TCP port 6514 (TLS encrypted). See "Access Policies" Under "Network Administration Commands" for details on creating access policies. Plaintext connections to the remote audit log server are not allowed in the NIAP evaluated configuration.

```
/NIAPVPN01.apriva.com(config)# logging
/NIAPVPN01.apriva.com(config-logging)# rotate archive 5
/NIAPVPN01.apriva.com(config-logging)# rotate interval weekly
/NIAPVPN01.apriva.com(config-logging)# rotate size 100M
/NIAPVPN01.apriva.com(config-logging)# commit
Changes committed.
/NIAPVPN01.apriva.com(config-logging)# exit
/NIAPVPN01.apriva.com(config)#
```

The following example is used to illustrate a means of creating a ruleset to log and use the log-prefix option to decide whether the packet is accepted or dropped as below.

Create one ruleset to accept and another ruleset to drop packets.

```
ruleset log_accept {
  match 10 action log log-prefix ACCEPTED_PACKET
  match 20 action permit
}

ruleset log_drop {
  match 10 action log log-prefix DROPPED_PACKET
  match 20 action deny
}
```

Now that rulesets have been defined, rules can be added to the access list. In the example below, the access list is established to only accept port 81 and drop Port 82 packets.

```
access-list main {
  match 10 proto tcp dstport 81 action log_accept
  match 20 proto tcp dstport 82 action log_drop
  ….
  match 30 action…..
}
```

This example can then be tested by attempting a connection to each port – 81 and 82. The iptables log reports:

VPN02 kernel: ACCEPTED_PACKET IN=enp1s0 OUT= MAC=00:e0:67:25:b7:ea:54:1e:56:18:23:11:08:00 SRC=172.16.57.126 DST=172.16.56.59 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=14851 DF PROTO=TCP SPT=46318 DPT=81 WINDOW=64240 RES=0x00 SYN URGP=0

VPN02 kernel: DROPPED_PACKET IN=enp1s0 OUT= MAC=00:e0:67:25:b7:ea:54:1e:56:18:23:11:08:00 SRC=172.16.57.126

DST=172.16.56.59 LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=14851 DF PROTO=TCP SPT=46318 DPT=82 WINDOW=64240 RES=0x00 SYN URGP=0

#### 2.1.15.1 Syslog Server Requirements

The Apriva MESA VPN communicates with an external syslog (audit) server by establishing a trusted channel between itself and the audit server that is protected using TLSv1.2.

The Apriva MESA VPN requires the TLS Syslog server to present an x509 certificate during TLS negotiations. The Apriva MESA VPN server will validate this certificate, ensure it matches the configuration and ensure that it has not been revoked using OCSP revocation checking. See section 2.1.21.2 for more information on the Syslog.

If an external syslog server is used, it must satisfy a minimum level of requirements and be sent the same audit log data that is recorded locally in the 'common' log. The minimum requirements for use of an external syslog server include the following elements.

1. During the TLS handshake, the syslog server must supply an x509 server certificate.
    a. The syslog server' certificate must not have expired, have the proper 'Server Authentication' extended key usage and ensure the certificate contains the Basic Constraints extension (cA flag set to TRUE).
    b. The TLS certificate must have a valid trust chain imported into the Apriva MESA VPN server.
    c. The syslog server's certificate must contain a Subject Alternative Name (SAN), or Subject Common Name (CN) field if the SAN is empty, that matches the specified host address of the syslog server. The SAN can be used to match either DNS names or numeric IP addresses. The common name supports DNS name matching. The common name does not support using a numeric IP address. This must be encoded in the preferred name syntax described in section 3.5 of RFC 1034. By default, wildcards are supported, and they match only in the left-most label; but they may match part of that label with an explicit prefix or suffix. For example, by default, the host **name** 'syslog.example.com' would match a certificate with a SAN or CN value of '*.example.com', 's*.example.com' or '*g.example.com'.
    **NOTE:** If the ***Common Name*** of the certificate contains a generic hostname, do not specify a specific hostname or an IP address in the ***subject_alt_name*** parameter.
    d. Apriva MESA VPN revocation checking uses OCSP URLs contained inside the syslog server's certificates and within all certificates in the certificate chain. The system requires:
        i. A valid OCSP URL is defined inside the certificate attributes.
        ii. The OCSP server is reachable.
        **NOTE:** If the OCSP response is received, it is always enforced. If the OCSP response is not received and the allow-failure option is enabled, it permits the connection.
2. The TLS syslog server must trust the Apriva MESA VPN TLS certificate that is imported into the Apriva MESA VPN server.
3. The TLS syslog server must be reachable from the Apriva MESA VPN network.
4. The TLS syslog server must support the following protocols and RFCs:

a. TCP RFC 1180 and TLS RFC 5246.
b. Syslog RFC 3164 and RFC 5424.
c. RFC 5425 Transport Layer Security (TLS) Transport Mapping for Syslog.

#### *2.1.15.2* Configure Log Archival Settings

By default, log archival is disabled. Recommended settings for archival are:

- Archive every 7 days.
- Archive when log reaches 10 MB in size or greater.

```
/NIAPVPN01.apriva.com(config)# logging
/NIAPVPN01.apriva.com(config-logging)# rotate size 10M
/NIAPVPN01.apriva.com(config-logging)# rotate interval weekly
/NIAPVPN01.apriva.com(config-logging)# commit
Changes committed.
/NIAPVPN01.apriva.com(config-logging)# exit
/NIAPVPN01.apriva.com(config)#
```

### 2.1.16 Set System Date/ Time

Display and set the system date and time. This example uses the date command to change the date and time to 09:30 am on Wednesday, July 31, 2013.

```
date <date-string>

/localhost# date 31 July 2013 09:30

Wed Jul 31 09:30:00 MST 2013
```

### 2.1.17 Configure the NTP Service

The Network Time Protocol (NTP) is a protocol used to synchronize computer system clocks automatically over a network. The machine can have the system clock use Coordinated Universal Time (UTC) rather than local time. Additionally, the Apriva MESA VPN only currently supports four algorithms – MD5, SHA256, SHA384, and SHA512.

The following sample command set activates and configures NTP on the Apriva MESA VPN. The Users Guide lists and describes available NTP command detail. Additional information on the NTP Service can be obtained in Section 6 of this document.

```
/localhost> ntp start
Starting service 'chronyd' : SUCCESS!
/localhost>
```

The following commands can be used to set up the initial service configuration as well as the required key configuration by adding the required input to each command.

Initial Service Configuration

```
configure (enter the configuration scope)
ntp (configure ntp)
server <address> (supply the address of the NTP server)
key 1 SHA256 <key> trusted (enter a key)
no disable (enable the service) commit (apply the changes)
```

Required Key Configuration

Establish a Key

```
key 1 SHA256 <key> trusted
```

Remove the Assigned Key

```
no key 1
```

### 2.1.18 Configure DNS (optional)

The Apriva MESA VPN server supports configuration of DNS to allow for connected devices to resolve services beyond the Apriva MESA VPN server. The domain and server may be set. In addition, DNS server addresses may be set within the global VPN settings to allow client devices to use DNS to resolve hostnames of services within the network protected by the Apriva MESA VPN server.

**NOTE:** In order for the DNS to operate, an access policy must be defined for UDP port 53 to allow DNS traffic. See "Access Policies" Under "Network Administration Commands" for details on creating access policies.

```
/NIAPVPN01.apriva.com(config)# dns domain apriva.com
/NIAPVPN01.apriva.com(config)# dns server 172.16.44.53
/NIAPVPN01.apriva.com(config)# commit
Changes committed.
/NIAPVPN01.apriva.com(config)#
```

### 2.1.19 Configure Distinguished Name Match Rules

The Apriva MESA VPN server supports authenticating device connection against Distinguished Name (DN) fields. The DN is part of the client device's certificate and is presented during the IPsec negotiation process.

A DN is composed of multiple fields (Relative Distinguished Names or RDNs). The field types that may be matched include:

- Country (C).
- State/Province (ST).
- Organization (O).
- Organization Unit (OU).
- Locate (L).
- Domain Component (DC).

A certificate's DN may contain one or more of these fields and fields may have multiple values within each field type.

The Apriva MESA VPN server supports the creation of DN 'sets'. DN sets are composed of one or more RDN fields. Multiple DN sets may be created to allow for different client organizational requirements; each client certificate's DN is compared against all defined DN sets.

The RDN format is:

```
<field-type>=<value>
```

- <field-type> is one of [C, ST, O, OU, L, DC].

- <value> is a literal, case-insensitive alphanumeric string which may also use dashes, underscores, periods, and spaces.
- No spaces between <field-type>, the '=' sign, and <value> are allowed.
- If <value> contains embedded spaces, it must be enclosed in double-quotes. The double-quotes are not part of the value.
- Each field type may only be specified once but may have multiple values. Multiple values for a given field-type may be specified by using a comma-separated list. The values' order is exact; 'OU=OU1,OU2,OU3' is not the same as 'OU=OU3,OU1,OU2'.
- Entering a new value for an existing field-type in a DN set replaces the existing field-type value.

The rules for DN matching are:

- RDNs can be specified in any order in the DN rule set and in the client certificate's DN.
- All fields in the client's DN must be in a DN rule set and the contents of all fields must match exactly for a match to be made against that set.
- Multiple fields of a given field type in the client certificate's DN must match the order of the values in the DN set for that field type for a match to be made against that set.
- If all the client certificate's DN fields match those in a configured DN set, the client will be allowed to connect.
- If there are no DN sets defined, this feature is disabled.

### Usage

```
dnmatch <set-name> [rdns]
no dnmatch <set-name> [dn-types]
```

### Options

| Option | Description | Details |
|---|---|---|
| set-name | The name of the DN set. | 1-16 alphanumeric characters |
| rdns | A list of RDN entries | Used when adding RDNs;<br>format <field-type>=<value><br><field-type> is one of [C, ST, O, OU, L, DC]<br><value> is a literal, case-insensitive alphanumeric string, including special characters [ - _ . ] and spaces<br>One or more RDNs must be specified |
| dn-types | A list of DN field types | Used when removing RDNs;<br>one of [C, ST, O, OU, L, DC] |
| no | Remove the specified RDN(s) or DN set | If no RDN field types are specified, the entire DN set is removed |

### Examples

```
/localhost(config-vpn)# dnmatch govset C=US O="US Government" OU="Govt Contractor"
/localhost(config-vpn)# dnmatch coolset C=US L=Hollywood
OU=VIP,Director,Producer,Gaffer O=Pixar
/localhost(config-vpn)# dnmatch coolset C=US L=Hollywood OU=Director,Producer O=Pixar
/localhost(config-vpn)# no dnmatch coolset L
/localhost(config-vpn)#
```

### Audit Log

The dnmatch command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields |
|---|---|

| | |
|---|---|
| [<UUID>] User [<user>@<address>] executed command [dnmatch <set-name> <rdns>].<br><br>OR<br><br>[<UUID>] User [<user>@<address>] executed command [no dnmatch <set-name> <dn-types>]. | The dnmatch command generates an audit log that contains the following fields. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| user | This is the name of the account (aka userid) that issued the command. |
| address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| set-name | The name of the DN set. |
| rdns | A list of RDN entries. |
| dn-types | A list of DN field types to remove. |

### 2.1.20 Set Access Policies

Access policies allow the Apriva MESA VPN server to permit or deny packet traffic based on traffic type, source, or destination. By default, the Apriva MESA VPN server will drop all unmatched inbound, outbound, or forwarded packets. The operator may add rules to provide enhanced traffic filtering. For more information and examples, see Section 6 (Access Policy Management).

**NOTE:** The system installs a default access policy that allows SSH connectivity on the management network interface. This policy can be modified or replaced as desired. However, an access policy must be defined that allows communications on TCP port 22 for remote access to operate.

It may be necessary to add access list rules to allow devices' application-specific traffic.

If an external syslog server is used, access list rules must be added to allow syslog/TLS traffic to and from the syslog server. The following example allows TLS traffic to and from the server.

```
/NIAPVPN01.apriva.com(config)# access-policy
/NIAPVPN01.apriva.com(config-apol)# access-list syslog-rules
/NIAPVPN01.apriva.com(config-apol-acl)# match 1 proto tcp dstport 6514 action permit
/NIAPVPN01.apriva.com(config-apol-acl)# match 2 proto tcp srcport 6514 action permit
/NIAPVPN01.apriva.com(config-apol-acl)# exit
/NIAPVPN01.apriva.com(config-apol)# exit
/NIAPVPN01.apriva.com(config)# interface em1
/NIAPVPN01.apriva.com(config-iface)# access-list syslog-rules in
/NIAPVPN01.apriva.com(config-iface)# exit
/NIAPVPN01.apriva.com(config)# commit
Changes committed.
/NIAPVPN01.apriva.com(config)#
```

### 2.1.21 Create VPN Authentication Suites

The Apriva MESA VPN server defines authentication parameters in authentication suites. Suites are identified by name and each one utilizes an encryption algorithm (RSA or ECDSA). The Apriva MESA VPN does not support the use of pre-shared keys for VPN authentication. Multiple suites may be defined of each type, but only one RSA suite or one ECDSA suite may be used at

any one time as configured in the server's global VPN settings. This restriction applies only to the IPsec portion and does not apply to TLS.

For convenience, one suite of each type is created by default and these suites cannot be modified. They are intended to be used as templates to create new suites to simplify Apriva MESA VPN server configuration. New suites can also be created manually if desired.

The NIAP evaluated configuration only allows the use of the following algorithms: AES-CBC-128, AES-CBC-256, AES-GCM-128, AES-GCM-256 as well as the required hash sizes SHA-256, SHA-384, and SHA-512. The TOE will ensure that the IKE algorithm key size must be equal to or stronger than the ipsec algorithm key size (i.e., the ESP algorithm key size). The NIAP evaluated configuration also allows use of DHgroup 14, 15, 19, 20 and 24 only.

This guide uses the suite templates to create new suites and displays the configuration settings for the newly created suites. The example below shows two different suites being created but in practice only one can be used at a time.

```
/NIAPVPN01.apriva.com(config)# vpn-suite SystemRSA copy suiteb-rsa
/NIAPVPN01.apriva.com(config-vpnsuite)# algorithm AES-256
/NIAPVPN01.apriva.com(config-vpnsuite)#  algorithm AES-GCM-256
/NIAPVPN01.apriva.com(config-vpnsuite)#  algorithm ike sha2 SHA-384
/NIAPVPN01.apriva.com(config-vpnsuite)#  algorithm ipsec sha2 SHA-512
/NIAPVPN01.apriva.com(config-vpnsuite)#  group ike 14
/NIAPVPN01.apriva.com(config-vpnsuite)#  group pfs 14
/NIAPVPN01.apriva.com(config-vpnsuite)#  keylife 3600
/NIAPVPN01.apriva.com(config-vpnsuite)#  ikelife 86400
/NIAPVPN01.apriva.com(config-vpnsuite)#  pfs
/NIAPVPN01.apriva.com(config-vpnsuite)# exit
/NIAPVPN01.apriva.com(config)# commit
Changes committed.
/NIAPVPN01.apriva.com(config)#

/NIAPVPN01.apriva.com(config)# vpn-suite SystemECDSA copy base-ecdsa
/NIAPVPN01.apriva.com(config-vpnsuite)# exit
/NIAPVPN01.apriva.com(config)# commit
Changes committed.
/NIAPVPN01.apriva.com(config)#
```

An RSA suite has the following configuration by default:

```
/NIAPVPN01.apriva.com(config)# show running-config
vpn-suite SystemRSA copy suiteb-rsa {
    algorithm AES-256
    algorithm AES-GCM-256
    algorithm ike sha2 SHA-384
    algorithm ipsec sha2 SHA-512
    group ike 14
    group pfs 14
    keylife 3600
    ikelife 86400
    pfs
}
```

An ECDSA suite has the following configuration by default:

```
vpn-suite SystemECDSA copy base-ecdsa {
    algorithm ike sha2 SHA-384
    algorithm ipsec sha2 SHA-512
    group ike 20
```

```
    group pfs 20
    keylife 3600
    ikelife 86400
}
```

The Apriva MESA VPN utilizes Internet Key Exchange version 2 (IKEv2) which is the protocol used to set up a security association (SA) in the IPsec protocol suite. IKEv2 and Network Address Traversal (NAT) are automatically enabled and there are no other configurations necessary.

The Apriva MESA VPN IKE lifetime default value is 86,400 seconds (24 hours) but can be changed to have no lifetime or the maximum value of 86400 seconds. Although the following command line can be adjusted to utilize the word no and omit the optional <value> entry, this option is not allowed in the evaluated configuration, so administrators should NEVER do this.

If an Ike lifetime is required omit [no] and enter a required numeric value equal to or less than 86400.

[no] ike-lifetime seconds <value>

Set the IKE Phase 2 (IPsec SA) key lifetime for the current VPN authentication suite. IKE Phase 2 key lifetimes may be specified in seconds, the number of kilobytes transmitted or received using the key, or both. If neither are specified, the system defaults to an IKE Phase 2 key lifetime of 8 hours (28800 seconds). The CC evaluated configuration must have a maximum of 8 hours configured. Parameters are restricted in NIAP usage. The [no] option of this command is outside the scope of NIAP testing. To meet the NIAP Protection Profile, the IPsec key lifetime must be configured.

**Usage**

```
[no] ipsec-lifetime <seconds | kilobytes> <value>
```

Perfect Forward Secrecy (PFS) Diffie-Hellman (DH) groups are used when negotiating the security association between two devices. DH Group numbers determine the strength of the key used in the key exchange process and are defined by Internet Assigned Numbers Authority (IANA). The following web site provides more in-depth information on IKE attributes: http://www.iana.org/assignments/ipsec-registry/ipsec-registry.xhtml

The Apriva MESA VPN permits the operator to configure settings for Set IKE and/or PFS Diffie-Hellman (DH) group settings used when authenticating a device against the VPN authentication suite. The following DH groups can be used.

| DH Group | Encryption | Description |
|---|---|---|
| 20 | ECDSA | 384-bit random ECP |
| 19 | RSA | 256-bit random ECP |
| 15 | RSA | 3072-bit MODP group |
| 24 | RSA | 2048-bit MODP group with 256-bit Prime Order Subgroup |
| 14 | RSA | 2048-bit MODP group |

The Groups may be entered in any order. The groups will be sorted according to their cryptographic complexity when stored and used. The user is responsible for ensuring the selected groups match the desired cryptographic capabilities.

The VPN authentication suite's IKE or PFS groups may be viewed using 'show running-config' (for current configuration) or 'show pending-config' (for pending configuration changes).

### 2.1.22 Install Trust Chains and Certificates/Keys

The Apriva MESA VPN server identifies network peers using x509 Certificates. These certificates are exchanged during the negotiation of network protocols which provide protected VPN communication and protected audit log transfers. Every certificate is issued by a Certificate Authority (CA). This trust relationship between a Certificate and the certificate belonging to the issuing CA forms a Trust Chain. A common Trust Chain is composed of a Root CA, an intermediate CA, and an end-entity Certificate. The end-entity certificate would be used by the Apriva MESA VPN server, a VPN peer, or a syslog server to identify themselves to their network peer within the context of the TLS or IPsec protocol negotiation. The intermediate CA would issue the end-entity certificate and the intermediate CA would have a certificate that was issued by the Root CA. The Root CA certificate is always self-signed. The Root CA inherits its trust by virtue of being installed by an administrator into the Apriva MESA VPN server.

The Apriva MESA VPN server must be configured to trust a Root CA. Once the Apriva MESA VPN server is configured to trust a specific Root CA, all certificates issued by that Root CA and all subordinate CA will be accepted as valid authentication for the network peer using the certificate. This allows the Apriva MESA VPN server to load one Root CA certificate and accept many certificates as valid authentication data.

The Apriva MESA VPN server can be configured to accept one RSA Root CA and one ECDSA Root CA that is used for authentication of peers during VPN communication. The Apriva MESA VPN server can also be configured to accept one RSA Root CA and one ECDSA Root CA to be used for authentication of peers during TLS communication.

The Apriva MESA VPN server management operations allow the importing and deleting of Root CA certificates. The Apriva MESA VPN server can obtain an end-entity certificate by directly importing a certificate and the private key associated with that certificate.

Rather than importing certificates with their private keys, the Apriva MESA VPN server can generate the private key internally, issue a Certificate Signing Request (CSR) to obtain a certificate from a CA and import the signed x509 certificate. (See section 1.7 for more information about Certificate Signing Requests.) Thus, management operations are available for generating and deleting keys, specifying values for a CSR, generating a CSR, and importing a certificate that is signed by an external CA. Once the certificate is imported into the Apriva MESA VPN, it can be deleted but not modified.

End-entity certificates are validated when they are imported, and when they are provided during TLS or IPsec session negotiation. TLS and IPsec session negotiation will typically include the end-entity certificate as well as all intermediate and root CA certificates necessary to verify the trust chain for the end-entity certificate being presented. However, during the management operation that imports the Apriva MESA VPN server's own certificates, it may be necessary to

import the Root CA as well as all intermediate CA certificates prior to importing the end-entity certificate.

If provided by an external authority, certificates must be bundled in PKCS#12 format with their private key, and may be converted to PEM format for installation through the Command Line Interface as a cut-and-paste operation, installed via SCP, or installed from CD-ROM.

If the certificate is the result of a Certificate Signing Request (CSR), the certificate is to be in PEM format, and the private key used to generate the CSR must be specified.

#### *2.1.22.1* Import VPN Trust Chains and Certificates/Keys

The Apriva MESA VPN server will not provide services without certificate material to be used in authenticating clients. This section details the criteria required for correct certificate/key importation.

- The Apriva MESA VPN server supports the use of a full chain of trust (root, intermediates, and user CA certificates). For certificates, the private key may be bundled with the certificate when provided by an external authority, or the certificate may use a locally generated private key used to create a Certificate Signing Request (CSR).
- All certificate/key material is imported to the Apriva MESA VPN server through one of three methods:
    1. Cut-and-paste operation of DER or PEM format file data using the session console.
    2. Import DER, PEM or PKCS#12 format file data using SCP.
    3. Import DER, PEM or PKCS#12 format file data from a CD-ROM.
- Each CA certificate and certificate may be imported separately (one at a time into a specified trust chain) or as part of a PKCS#12 file (PEM or binary).
- When importing CA certificates individually, the items must be imported into the same trust chain. For the certificate material to be properly validated during import the user must import the trust chain material in a certain order:
    1. First, the root CA of the trust chain must be imported, which creates the trust chain.
    2. Second, subordinate CA(s) must be imported. To import a subordinate CA, its issuing CA certificate must first be imported so the trust chain can be validated.
    3. Third, the relevant CRL(s) must be imported (if not using an external certificate distribution point).
    4. Finally, the private key/public certificate material must be imported.
- When importing a PKCS#12 file (PEM or binary), all necessary CA certificates must be present for the certificate within the bundle to be validated.
- At least one trust chain and certificate for either RSA or ECDSA authentication must be imported for the Apriva MESA VPN server to accept IPsec connections.
- Certificates are validated during the following three phases:
    1. During import.

2. During activation.
3. At the time of authentication of a tunnel or TLS connection.

The following commands are available within the 'vpn-auth' scope to import, delete, and show certificates, CRLs, and trustchains.

| Command | Description |
|---|---|
| import cacert | Create a VPN trust chain and install a CA certificate using cut/paste. |
| import cacert scp | Create a VPN trust chain and install a CA certificate using SCP. |
| import cacert cdrom | Create a VPN trust chain and install a CA certificate from a CD-ROM. |
| import certificate | Install a certificate into a VPN trust chain using cut/paste. |
| import certificate scp | Install a certificate into a VPN trust chain using SCP. |
| import certificate cdrom | Install a certificate into a VPN trust chain from a CD-ROM. |
| delete trust | Remove a VPN trust chain. |
| delete cacert | Remove a CA certificate from a VPN trust chain. |
| delete certificate | Remove a VPN certificate. |
| show trustchains | Show the available VPN trust chains. |
| show certificates | Show available VPN certificates. |

The following example shows the creation of an RSA Trust Chain for use with VPN authentication. It uses the cut/paste method to load individual CA certificates and the end-entity certificate to be used by the Apriva MESA VPN server.

```
/NIAPVPN01.apriva.com(config)# vpn-auth
/NIAPVPN01.apriva.com(config-vpnauth)# import cacert rsarootca rsachain
Paste the cacert. Enter a line with a single period ('.') at the end.>>
-----BEGIN CERTIFICATE-----
>> [body of certificate]
>>----END CERTIFICATE-----
>>.
Changes committed.
/NIAPVPN01.apriva.com(config-vpnauth)#
/NIAPVPN01.apriva.com(config-vpnauth)# import cacert rsainterca rsachain
Paste the cacert.
Enter a line with a single period ('.') at the end.>>-----BEGIN CERTIFICATE-----
>> [body of certificate]
>>-----END CERTIFICATE-----
>>.
Changes committed.
/NIAPVPN01.apriva.com(config-vpnauth)#
/NIAPVPN01.apriva.com(config-vpnauth)# import cacert rsavpnca rsachain
Paste the cacert.
Enter a line with a single period ('.') at the end.>>-----BEGIN CERTIFICATE-----
>> [body of certificate]
>>-----END CERTIFICATE-----
>>.
Changes committed.
/NIAPVPN01.apriva.com(config-vpnauth)#
/NIAPVPN01.apriva.com(config-vpnauth)# import certificate rsavpnmaterial rsachain
Paste the certificate.
Enter a line with a single period ('.') at the end.>>-----BEGIN ENCRYPTED PRIVATE KEY-
----
>> [body of certificate]
>>-----END ENCRYPTED PRIVATE KEY-----
>>-----BEGIN CERTIFICATE-----
>> [body of certificate]
```

```
>>-----END CERTIFICATE-----
>>.
PKCS12 Password:
Changes committed.
/NIAPVPN01.apriva.com(config-vpnauth)# exit
/NIAPVPN01.apriva.com(config)#
```

#### *2.1.22.2* Import TLS Trust Chains and Certificates/Keys

TLS is only used by the Apriva MESA VPN to connect to a remote log server via a secure connection and establishing that connection requires a TLS trust chain and certificates/key. This section details the criteria required for correct TLS trust chain and certificate/key importation.

- The Apriva MESA VPN server supports the use of a full chain of trust (root, intermediates, and certificate). For certificates, the private key may be bundled with the certificate when provided by an external authority or the certificate may use a locally generated private key used to create a Certificate Signing Request (CSR).
- All certificate/key material is imported to the Apriva MESA VPN server through one of three methods:
    1. A cut-and-paste operation of DER or PEM format file data using the session console.
    2. An import of DER, PEM or PKCS#12 format file data using SCP.
    3. An import of DER, PEM or PKCS#12 format file data from a CD-ROM.
- Importing a PKCS#12 file (PEM or binary) requires all necessary CA certificates be present for the certificate within the bundle to be validated.
- A certificate authority (CA) certificate can be installed by cutting and pasting from a file to the session console and assigned to a trust chain. If the trust chain does not exist, it will be created.
- A certificate authority (CA) certificate can be installed from a remote source via SCP and assigned to a trust chain. If the trust chain does not exist, it will be created.
- A certificate authority (CA) certificate can be installed to the certificate store from a file stored on a CD-ROM and assigned to a trust chain. If the trust chain does not exist, it will be created.
- A certificate can be installed by cutting and pasting from a file to the session console and assign to an existing trust chain.
- A certificate can be installed from a remote source via SCP and assigned to an existing trust chain.

The following commands are available to import TLS trust chains and certificates/keys.:

| Command | Description |
|---|---|
| crypto import cacert | Create a trust chain and install a CA certificate using cut/paste. |
| crypto import cacert scp | Create a trust chain and install a CA certificate using SCP. |
| crypto import cacert cdrom | Create a trust chain and install a CA certificate from a CD-ROM. |

| | |
|---|---|
| crypto import certificate | Install a certificate into a trust chain using cut/paste. |
| crypto import certificate scp | Install a certificate into a trust chain using SCP. |
| crypto import certificate cdrom | Install a certificate into a trust chain from a CD-ROM. |
| delete crypto trust | Remove a trust chain and all CA certificates it contains. |
| delete crypto cacert | Remove a CA certificate. |
| delete crypto certificate | Remove a certificate. |
| show crypto trustchains | Show the available trust chains. |
| show crypto certificates | Show available certificates. |

The following example shows the creation of a TLS Trust Chain for remote syslog authentication. It uses the cut/paste method to load individual CA certificates and the end-entity certificate to be used by the Apriva MESA VPN server.

```
/NIAPVPN01.apriva.com (config)#crypto import cacert myrootca mytlschain
Paste the cacert.
Enter a line with a single period ('.') at the end.>>
-----BEGIN CERTIFICATE-----
[encrypted certificate body]
-----END CERTIFICATE-----
.

/NIAPVPN01.apriva.com (config)#crypto import cacert myinterca mytlschain
Paste the cacert.
Enter a line with a single period ('.') at the end.>>
-----BEGIN CERTIFICATE-----
[encrypted certificate body]
-----END CERTIFICATE-----
.

/NIAPVPN01.apriva.com (config)#crypto import cacert mysubca mytlschain
Paste the cacert.
Enter a line with a single period ('.') at the end.>>
-----BEGIN CERTIFICATE-----
[encrypted certificate body]
-----END CERTIFICATE-----
```

### 2.1.23 Syslog

There are several system logging tools available that provide a high-performance tool containing key security features and a modular design that can be used to capture and record a list of user defined events. The Apriva MESA VPN uses syslog which includes these features to accept inputs from a wide variety of sources, transforms them, and outputs the results to diverse destinations. This information includes capturing relevant VPN specific operational events (aka audit records), End User Device (EUD) connections, and certificate usage.

There are internal commands to start, restart, and/or status syslog using the Apriva MESA VPN syslog service. Syslog stores the UUID number that specifies the session number of the user that issued the command, the user account name (userid) that issued the command, the IP address of the computer where the syslog command was issued from. For console commands this will be 127.0.0.1, and the command executed (start, restart, or status).

Syslog enables the following two cipher suites only. The list of TLS ciphersuites is not configurable.

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289.

If the plan is to use TLS for syslog, TLS must authenticate the syslog server and encrypt traffic to that server. It's important to note, however, while the Apriva MESA VPN must be able to provide TLS-protected syslog, the customer is not required to use it.

**NOTE:** The Apriva MESA VPN utilizes the open source rsyslog tool to fulfill its system logging requirements.

The following commands are used to start, restart, and obtain the operational status of syslog.

- **Start Command**

```
/localhost# syslog start
Starting service 'rsyslog': SUCCESS!
```

- **Restart Command**

```
/localhost# syslog restart
Stopping service 'rsyslog': SUCCESS!
Starting service 'rsyslog': SUCCESS!
```

- **Obtain Syslog Status Command**

```
/localhost# syslog status
Service 'rsyslog' is running.
```

There are specific system logging configurations required to properly log system operations and those commands are shown below.

**NOTE:** Before you can configure a remote network destination using TLS, you must import a trust chain and certificate used by the TOE as described in the previous section.

**NOTE:** Mutual authentication is supported but is not required. If mutual authentication is required by the remote syslog server, then it must be enabled using the import certificate command.

The following example depicts properly configuring the system for proper syslog usage and employs the commands described above.

```
/NIAPVPN01.apriva.com(config)# logging
/NIAPVPN01.apriva.com(config-logging)# network-destination
/NIAPVPN01.apriva.com(config-logging-dest)# host syslog-landingserver.testdomain.com
/NIAPVPN01.apriva.com(config-logging-dest)# port 6514
/NIAPVPN01.apriva.com(config-logging-dest)# no disable
/NIAPVPN01.apriva.com(config-logging-dest)# tls
/NIAPVPN01.apriva.com(config-logging-dest-tls)# cert syslogmaterial syslogkey
/NIAPVPN01.apriva.com(config-logging-dest-tls)# chain syslogchain
/NIAPVPN01.apriva.com(config-logging-dest-tls)# curve secp384r1
/NIAPVPN01.apriva.com(config-logging-dest-tls)# revocation
/NIAPVPN01.apriva.com(config-logging-dest-tls)# exit
/NIAPVPN01.apriva.com(config-logging-dest)# commit
Stopping service 'rsyslog': SUCCESS!
Starting service 'rsyslog': SUCCESS!
Changes committed.
/NIAPVPN01.apriva.com(config-logging-dest)# exit
/NIAPVPN01.apriva.com(config-logging)# exit
/NIAPVPN01.apriva.com(config)#
```

The value specified in the network destination host is considered the “reference identifier” that is matched against the SAN or CN as discussed in section “Syslog Server Requirements”.

The supported and evaluated curve used for an ECDHE key exchange is secp384r1.

The ‘cert’ command is used to identify the certificate that the Apriva MESA VPN will send to the syslog server during the TLS negotiation.

```
/NIAPVPN01.apriva.com(config-logging-dest-tls)# cert syslogmaterial syslogkey
```

**NOTE:** If a connection is interrupted, it will automatically attempt to reconnect.

**NOTE:** If certificate issue exists, the certificate issue then the certificate anomaly must be addressed.

Key notes about revocation checking include the following:

1. Revocation checking is performed differently on TLS syslog and VPN IPsec connections.
2. For TLS syslog,
    1. The system ALWAYS performs revocation checking on all certificates that contain an AIA specifying the OCSP URI.
    2. The system ALWAYS rejects connections if the OCSP response indicates the certificate is revoked.
    3. The administrator can determine if the system will accept or reject a certificate if the OCSP URI cannot be reached.
3. For VPN IPsec,
    1. The system ALWAYS performs revocation checking on all certificates that contain an AIA specifying the OCSP URI or a CRL Distribution Point.
    2. The system ALWAYS rejects connections if the OCSP response or CRL indicates the certificate is revoked.
    3. The admin can determine if the system will accept or reject a certificate if the OCSP URI **AND** the CRL cannot be contacted.
4. While the system has commands to completely disable revocation checking, revocation checking is mandatory for NIAP evaluated products.
5. CSfC also requires revocation checking but ALSO mandates that certificates be rejected if the system cannot determine the revocation status of the certificate by communicating with either an OCSP responder or CRL distribution point.

### 2.1.24 X.509 Certificate Key Usage OIDs

The internet security standard is based on a trust relationship model, also called “certificate chain of trust.” An X.509 certificate is a digital certificate based on the widely accepted International Telecommunications Union (ITU) X.509 standard, which defines the format of public key infrastructure (PKI) certificates. X.509 digital certificates validate the identity of a website, organization, or server and provide a trusted platform for the user to connect and share information

securely. The X.509 standard extension OIDs is used in the Apriva MESA VPN to access the associated certificate (and CRL) extension data.

Each certificate must be signed by the same issuer Certificate Authority (CA) named in its certificate. The client must be able to follow a hierarchical path of certification that recursively links back to at least one root CA listed in the client's trust store. In short, the VPN peer's certificate must have a valid trust chain to a trusted root CA.

Each Extended Key Usage defined is required in the VPN peer's certificate – Certificates are accepted from the following issuers.

| Extended Key Usage | Issuer | Protocol |
|---|---|---|
| clientAuth | 1.3.6.1.5.5.7.3.2 | TLS |
| IKEIntermediate | 1.3.6.1.5.5.8.2.2 | IKE |
| ipsecIKE | 1.3.6.1.5.5.7.3.17 | IKE |
| OCSP Signing | 1.3.6.1.5.5.7.3.9 | OCSP |
| CRL Signing | 1.3.6.1.5.5.x.x.x | CRL |

Certificate revocation is done in accordance with the Internet Engineering Task Force (IETF) Request for Comments (RFC) as detailed in section 7.1. The Apriva MESA VPN performs the following steps to validate a certificate: check the expiration, check the issuer, check the signature, and follow the chain to root before checking revocation. The revocation is checked in the following priority – OCSP, CRLDP, and then local CRL. If the revocation server cannot be reached and the allow-crl-failure is enabled, it permits the connection. If not enabled, the check fails.

In the event the revocation server cannot be reached, the system checks the allow-crl-failure option. If the option is enabled, then the system permits the certificate to validate. If the allow-crl-failure is disabled, the certificate is rejected.

The following vpn-setting subcommands can be used to set up a Target of Execution (TOE) configuration to perform the revocation checking required for a NIAP evaluation.

```
dns-servers <server-list>
ecdsa ciphersuite <suite-name>
address-pool <subnet>
dnmatch <name> <rdn> [rdns]
```

### 2.1.25 Notes about Certificate Revocation

1. The TOE performs revocation checking differently on TLS syslog and VPN IPsec connections.
2. For TLS syslog,
    a. The TOE ALWAYS performs revocation checking on all certificates that contain an AIA specifying the OCSP URI.
    b. The TOE ALWAYS rejects connections if the OCSP response indicates the certificate is revoked.

   c. The admin can determine if the TOE will accept or reject a certificate if the OCSP URI cannot be reached.
3. For VPN IPsec,
   a. The TOE ALWAYS performs revocation checking on all certificates that contain an AIA specifying the OCSP URI or a CRL Distribution Point.
   b. The TOE ALWAYS rejects connections if the OCSP response or CRL indicates the certificate is revoked.
   c. The admin can determine if the TOE will accept or reject a certificate if the OCSP URI **AND** the CRL cannot be contacted.
4. While the TOE has command to completely disable revocation checking, revocation checking is mandatory for NIAP evaluated products.
5. CSfC also requires revocation checking and mandates certificates be rejected if the TOE cannot determine the revocation status of the certificate by communicating with either an OCSP responder or CRL distribution point.

### 2.1.26 Configure Global VPN Settings

The Apriva MESA VPN server has both global settings as well as authentication suite specific settings. Global settings should be configured after creating an authentication suite, but the order is not critical if authentication suites are set as part of global VPN settings.

This guide provides example configurations only. The IP subnets for traffic selectors, the authentication carrier network(s) and the VPN IP address pool must conform to your network infrastructure.

The SSH and IPsec cryptographic algorithms utilized within the Apriva MESA VPN can be modified as detailed in section 2.1.9, however TLS is only permitted to utilize the NIAP approved algorithms.

The vpn-settings along with an access-list provide all the functionality of the SPD. An access-list defines the packets that are permitted or dropped. For packets that are permitted, those that match the vpn-settings traffic selector are encrypted, otherwise they bypass encryption.

**NOTE:** In order for clients to access the Apriva MESA VPN server, an access policy must be defined on the external network interface for IP protocols 50 and UDP ports 500 and 4500 to allow ESP/AH traffic. See “Access Policies” Under “Network Administration Commands” for details on creating access policies. The authrule specifies the policy that applies to an object and that is based on various conditions, such as context and environment. Each authorization rule has a unique name and can be applied to multiple objects in a domain. The example below uses authrule to enable restrictions on location, day, and time for VPN peer connections.

```
/NIAPVPN01.apriva.com(config)# vpn-settings
/NIAPVPN01.apriva.com(config-vpn)# authrule carrier 192.168.100.0/24
/NIAPVPN01.apriva.com(config-vpn)# authrule time "mon,tue,wed,thu,fri" "0000-2359"
/NIAPVPN01.apriva.com(config-vpn)# mobike
/NIAPVPN01.apriva.com(config-vpn)# client-traffic-selector 172.16.99.0/24
/NIAPVPN01.apriva.com(config-vpn)# rsa ciphersuite SystemRSA
/NIAPVPN01.apriva.com(config-vpn)# no rsa ciphersuite
/NIAPVPN01.apriva.com(config-vpn)# ecdsa ciphersuite SystemECDSA
/NIAPVPN01.apriva.com(config-vpn)# exit
/NIAPVPN01.apriva.com(config)# commit
Changes committed.
/NIAPVPN01.apriva.com(config)#
```

VPNsettings can also include restrictions on location, day, and time. In this example, the address-pool command assigns the VPN its IP address enables crl-failure, only permits from address range specified, defines the days of the week and the hours the VPN is available for connection, defines the DNS server, defines the cipher suite to be used, enables IKE fragmentation, enables mobike, sets the VPN server to be headend, specifies the use of no whitelist, no crl, and defines the IP addresses that it will route.

```
/NIAPVPN01.apriva.com(config)# vpn-settings
/NIAPVPN01.apriva.com(config-vpn)# address-pool 21.116.48.0/21
/NIAPVPN01.apriva.com(config-vpn)# allow-crl-failure
/NIAPVPN01.apriva.com(config-vpn)# authrule carrier "100.96.0.0/16"
/NIAPVPN01.apriva.com(config-vpn)# authrule                                    time
"sun","mon","tue","wed","thu","fri","sat" "0000-2359"
/NIAPVPN01.apriva.com(config-vpn)# dns-servers 172.16.53.36
/NIAPVPN01.apriva.com(config-vpn)# ecdsa ciphersuite ECC20
/NIAPVPN01.apriva.com(config-vpn)# ike-fragmentation
/NIAPVPN01.apriva.com(config-vpn)# interface external p5p2
/NIAPVPN01.apriva.com(config-vpn)# mobike
/NIAPVPN01.apriva.com(config-vpn)# mode headend
/NIAPVPN01.apriva.com(config-vpn)# whitelist
/NIAPVPN01.apriva.com(config-vpn)# vpn-device
/NIAPVPN01.apriva.com(vpndevice)# device device-example
/NIAPVPN01.apriva.com(vpndevice)# exit
/NIAPVPN01.apriva.com(config-vpn)# nocrl
/NIAPVPN01.apriva.com(config-vpn)# traffic-selector 172.16.52.0/26
/NIAPVPN01.apriva.com(config-vpn)# traffic-selector 21.116.48.0/21
/NIAPVPN01.apriva.com(config-vpn)# commit
```

#### *2.1.26.1 Set VPN Mode*

The Apriva MESA VPN can operate in various modes – headend, site to site, or both headend and site to site. Select the mode of the VPN. The User's Guide can be referenced for additional information.

- headend: Headend mode. The VPN assigns each client a tunnel IP address.
- site2site: Site to site mode. The VPN accepts tunnels from a remote site where the remote site determines the tunnel address.
- both: Support both headend and site to site mode. Allow the VPN client to decide whether to request an address or to provide the address it uses.

```
Mode [headend | site2site | both]
```

### 2.1.27 Add Default Gateway and IP Routes (as required)

Before the VPN service can start successfully, a default gateway will need to be added on the internal network ('int') interface. In addition, IP route(s) back to the VPN devices' carrier or wireless network on the 'ext' interface will need to be added so that tunnel negotiation can occur. Below is an example of adding a default gateway on the internal interface and an IP route back out the external interface.

```
/NIAPVPN01.apriva.com(config)# ip route 0.0.0.0/0 gw 172.16.54.1 dev em1
/NIAPVPN01.apriva.com(config)# ip route 100.0.0.0/8 gw 172.16.42.1
/NIAPVPN01.apriva.com(config)# commit
Changes committed.
/NIAPVPN01.apriva.com(config)# show ip route
  Destination          Netmask               Gateway               Iface
```

```
* 172.16.55.0            255.255.255.0          0.0.0.0              em2
* 172.16.54.0            255.255.255.0          0.0.0.0              em1
* 172.16.42.0            255.255.255.0          0.0.0.0              em4
  100.0.0.0              255.0.0.0              172.16.42.1          em4
  0.0.0.0                0.0.0.0                172.16.54.1          em1


* System (unmanaged) route
- Deleted route (pending)
+ New Route (pending)

/NIAPVPN01.apriva.com(config)#
```

### 2.1.28 Dead Peer Detection

To handle the case where there is an interruption in the IPsec connection, the system implements Dead Peer Detection. If the connection is idle, the system sends messages to the remote side to validate that the connection is still active. If responses to these messages are not received, the tunnel is declared dead and closed. The Dead Peer Detection setting is 5 minutes.

### 2.1.29 Add Devices to the Apriva MESA VPN Server

The system supports the use of a whitelist of authorized devices. The whitelist feature can be enabled (default mode) or disabled within the system. Whenever a whitelist is enabled, a device list must be created to serve as the index of approved devices that are granted access. The whitelist can also be modified to prevent a specific device from accessing the system before its certificate is revoked. When the whitelist is disabled, it is not necessary to add devices to the whitelist.

A whitelist contains the reference identifier for the TOE's VPN peer. This whitelist is used to implement the reference identifier – specifically the common name of the certificate for allowed IPsec connections. Therefore, to be in an evaluated configuration that satisfies the requirements of the ST, the whitelist feature must be enabled.

The Apriva MESA VPN supports authenticating device connection against Distinguished Name (DN) fields. The DN is part of the client device's certificate and is presented during the IPsec negotiation process. Certificates must meet the "ASN1DN" format and all other reference identifier formats like email, IP address, fully qualified domain name, etc. are rejected. A DN is composed of multiple fields known as Relative Distinguished Names (RDN) and they include Country (C), State/Province (ST), Organization (O), Organizational Unit (OU), Locate (L), and Domani Component (DC). A client certificate's DN may contain one or more of these fields and these fields may have multiple values within each field type.

The Apriva MESA VPN also supports the creation of DN sets. DN sets are composed of one or more RDN fields. Multiple DN sets may be created to allow for different client organizational requirements; each client certificate's DN is compared against all defined DN sets.

The RDN format is: <field-type>=<value>, where

- <field-type> is one of [C, ST, O, OU, L, DC].
- <value> is a literal, case-insensitive alphanumeric string which may also use dashes, underscores, periods, and spaces.
- No spaces between <field-type>, the '=' sign, and <value> are allowed.

- If <value> contains embedded spaces, it must be enclosed in double-quotes. The double-quotes are not part of the value.
- Each field type may only be specified once but may have multiple values. Multiple values for a given field-type may be specified by using a comma-separated list. The values' order is exact; 'OU=OU1,OU2,OU3' is not the same as 'OU=OU3,OU1,OU2'.
- Entering a new value for an existing field-type in a DN set replaces the existing field-type value.

The rules for DN matching are:

- RDNs can be specified in any order in the DN rule set and in the client certificate's DN.
- All fields in the client's DN must be in a DN rule set and the contents of all fields must match exactly for a match to be made against that set.
- Multiple fields of a given field type in the client certificate's DN must match the order of the values in the DN set for that field type for a match to be made against that set.
- If all the client certificate's DN fields match those in a configured DN set, the client will be allowed to connect.
- If there are no DN sets defined, this feature is disabled.

Refer to "Configure Distinguished Name Match Rules" section for commands to define specific DN matches.

To validate an identifier and successfully connect a tunnel, the Apriva MESA VPN extracts the fields from the DN to begin the process. It then looks for the common name (CN) in the Apriva MESA VPN approved device list. If there are zero or multiple common names, the request is rejected. If the common name value is not present in the list of approved device ids, the tunnel is rejected. The Apriva MESA VPN then looks for the other fields in the DN (such as O, OU, C) and looks for the values in the DN Matching using the rules shared above. These allow the operator to require specific values (for example O=U.S. Government). If any of the DN matching rules fail, the tunnel is rejected. If no DN Matching rules are specified, any value of O, OU, etc. are accepted.

From a TLS perspective, the reference identifier validation process centers on the remote syslog server's TLS certificate's Subject Alternative Name (SAN) or common name. When the Apriva MESA VPN's remote logging is configured, the configuration contains the address (name or IP number) of the remote syslog server. The remote syslog server sends its certificate in the TLS handshake. The Apriva MESA VPN checks the reference identifier by verifying the remote host address is contained inside the SAN or common name of the remote certificate. If the remote host address is not contained within the server's certificate, the tunnel is rejected.

Once the Apriva MESA VPN server is configured, devices can be provisioned to the server if a whitelist is required. Devices authenticate using the trust chain and certificate, but they must also be authorized by being provisioned into the Apriva MESA VPN server.

```
/NIAPVPN01.apriva.com# vpn-devices
/NIAPVPN01.apriva.com(vpndevice)#
```

Devices are individually provisioned to or removed from the Apriva MESA VPN server or may be provisioned in bulk using the device import function. Each device is identified to the server by the unique device identifier (UNDI) found in the common name of the device's certificate.

```
/NIAPVPN01.apriva.com(vpndevice)# device 001000000000001
/NIAPVPN01.apriva.com(vpndevice)# no device 001000000000001
Device '001000000000001' removed.
/NIAPVPN01.apriva.com(vpndevice)#
```

The Apriva MESA VPN server will automatically reconnect an Ipsec VPN connection that is interrupted.

## 2.2 Procedural Requirements

The Apriva MESA VPN server Superuser (superuser) must not install any additional software on the Apriva MESA VPN server. The Apriva MESA VPN server is intended to only be operated with the software provided by Apriva ISS LLC.

The Apriva MESA VPN server must be physically protected from unauthorized access. These countermeasures should be commensurate with the value of the data being protected by the Apriva MESA VPN server.

The Superuser must be a Trusted Agent and follow and apply all administrator guidance.
The Apriva MESA VPN server must be installed in the network in a manner that will allow the Apriva MESA VPN server to effectively enforce its policies on network traffic flowing among attached networks.

## 2.3 Software Update

Software Updates to the system are performed manually. The Superuser uses the command shell to execute the "update system repo" command to initiate a software update.

**NOTE:** The VPN should be considered unusable during an update as the system will automatically reboot once the installation is completed and validated.

No automatic software update method is provided.

### 2.3.1 Verifying Digital Signature

All software updates are signed using a digital signature. A software update consists of one or more RPM package files and a manifest file. Each RPM file contains a signature, and the manifest is also signed using a digital signature. The digital signatures are provided by the Linux GPG system.

The process of validating signatures is automated by the system and is automatically performed as part of a software update. When the system update command is executed, the system automatically validates the signatures of all the files in the update prior to beginning the installation of the updates and will only perform the update if all signatures are successfully validated.

The audit log is updated to reflect the status of the signature validation whether it succeeds or fails. The validation system writes audit log messages to indicate the results of the test regardless of whether the validation succeeds or fails. The administrator can view the audit logs to see the results of the validation.

### 2.3.2 Obtaining the Update

The software update files are obtained by contacting Apriva support. Software updates are available via download or DVD.

### 2.3.3 Initiating Update Process

The operator initiates the software update process by logging into the system as administrator and executing the "`update system repo`" command.

### 2.3.4 Determining if successful or not

The results of the update are both displayed on the Command Line Interface and in the audit log. If the update was successful, the system displays a message and then reboots automatically to apply the update.

### 2.3.5 Instructions for Validating the Digital Signature

There is no manual process for validating a signature. The process is automated and is performed as part of the software update command. Execute the software update command.

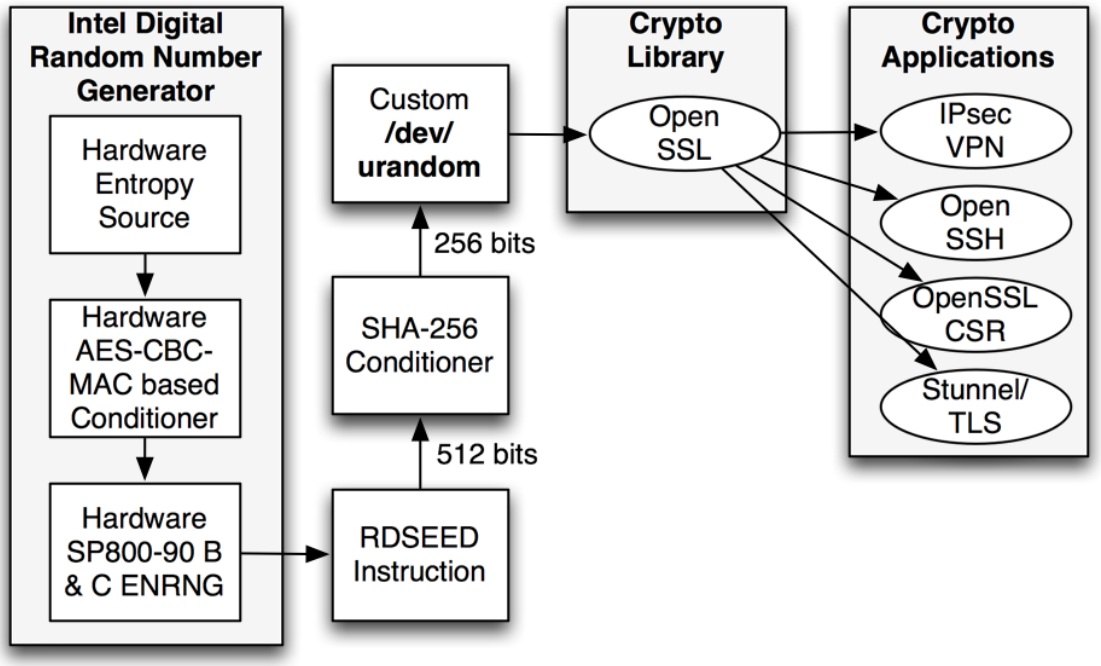
## 2.4 Warning for local storage space

The system generates warning messages in the audit log when the local storage space is more than 80% full. The operator can view the audit log using the "show logs" command.

# 3 Security Features

## 3.1 Entropy Source

The Apriva MESA VPN uses the Intel Secure Key (RDSEED instruction) as the noise source for entropy. The entropy is conditioned using SHA-256 as shown in the diagram below. See the Apriva MESA VPN Entropy Assessment Report for more details.




## 3.2 Self-Test

The self-test function works as follows:

- The system is powered on.
- The boot loader loads the kernel. Secure Boot is not implemented.
- The kernel loads the base operating environment. Measured Boot is not implemented.
- The kernel performs the FIPS test for the kernel cryptographic module.
- The system loads the self-test service.
- The self-test service performs the following steps:
    - Writes an audit log that the self-test has started.
    - Performs the FIPS test for the OpenSSL.
    - Checks the health of the entropy.
    - Validates all executable files using the GPG key and RPM file digital signatures. This includes file content, file ownership, and file permissions.
    - Validates all configuration files using hash values. This also includes file content, file permissions, and ownership.
    - Writes an audit log that the self-test has completed and the result.
- If the self-test fails, the system enters fail secure.
- If the self-test succeeds, the TOE software is loaded. The network, VPN, and other TOE services are loaded and begin to execute.

Periodic tests are performed:

- The FIPS self-test is performed again each time a cryptographic library is restarted.
- The entropy health is validated each time the entropy is read.
- The integrity of the executable code is validated each time a subcomponent is restarted (for example due to a configuration change).
- The integrity of the configuration files is validated using hash values.
- If the integrity fails, the system goes fail-secure.

## 3.3 User Actions

If the system experiences a fail-secure condition, the reason for the condition needs to be ascertained and corrected before it can be placed back in service.

In the case of a network hardware or service integrity check failure, a temporary failure may have occurred that does not impact any cryptographic operations or file integrity. The system may have to be restarted to clear the condition. Regardless, the system may be placed back in service once the condition is resolved.

If an entropy (noise source) integrity check failure occurs, restarting the Apriva MESA VPN server may clear the condition. If the condition recurs, the hardware entropy generator may be defective, and the system should be returned to Apriva for refurbishment after consultation with Apriva support.

If a runtime file check failure occurs, this is a more serious failure where one of the system's protected files has been disturbed. The system should be taken out of service and the root cause determined with the assistance of Apriva support. Once verified that the problem is not a security related issue, the system may be repaired with assistance from Apriva support and placed back in service. If a security incident has occurred, the system must not be placed back in service without servicing by and approval of Apriva support.

In cases where the fail-secure condition shuts down cryptographic services, the user will need to access the system console to view logs to determine the source of the problem.

## 3.4 National Information Assurance Partnership NIAP

The National Information Assurance Partnership (NIAP) is responsible for overseeing and monitoring the security of commercial IT products used in National Security Systems. NIAP certification is most applicable to the Department of Defense (DoD), the Intelligence community and any DoD contractors or affiliates. NIAP certification is also applicable and important to private sector companies.

NIAP certification is a commercial cybersecurity product certification that is mandated by federal procurement requirements (CNSSP 11) for use in U.S. National Security Systems (NSS). Its primary purpose is to certify commercial technology or products which will be used to handle sensitive data. NIAP testing and certification is a very stringent process using one or more NIAP common criteria Protection Profiles. These profiles help the evaluator ensure a minimal, baseline

set of requirements targeted at mitigating well defined and described threats has been established in the product being evaluated.

### 3.4.1 Protection Profiles

The Apriva MESA VPN version described here-in has been independently validated by a neutral third party and was then certified by the NIAP validation body who assessed the results of the security evaluation and issued Apriva a validation certificate. The NIAP provided Protection Profiles used to independently test against recognized criteria to a formalized methodology are listed below.

NIAP Protection Profiles (PP) are:

- Collaborative Protection Profile for Network Devices, Version 3.1 dated March 27, 2020 (CPP_ND_V2.2e)
- PP Module for Virtual Private Network Gateways Version 1.2 (MOD_VPNGW_V1.2)

In addition to the above PPs, additional features and controls were added in support of emerging COTS Solutions for Classified (CSfC) strengthening as defined in CSfC Selections for VPN Gateways.

Warning: Use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

### 3.4.2 Connection Diagram

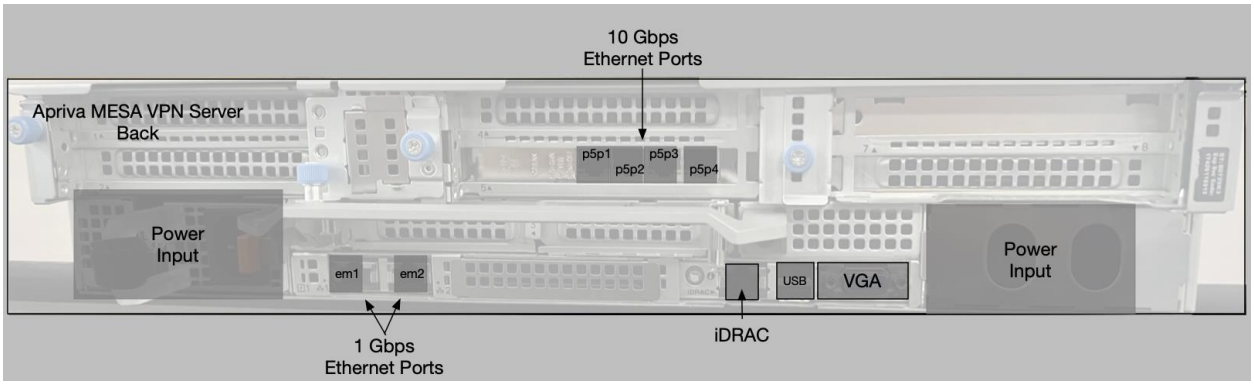



**Figure 6 – Apriva MESA VPN Server Connection Diagram**

The Apriva MESA VPN server allows the operator to select network interfaces for specific purposes, such as remote administration and VPN service operations.

Apriva strongly recommends that the external (device-facing) network interface be separate and not configured with VLANs, though all other services may be configured to use a single network interface and separate VLANs on that interface. Network Port Usage

The following network ports may be in active use by the Apriva MESA VPN server for its operations. Ports that are local-only are marked with 'lo (loopback)' as their interface. Ports marked with a specific interface name, or 'any' are accessible through other network interfaces and must be restricted through use of access-lists.

| Process Name | Port | Interface | Kernel Ring Level | Usage |
|---|---|---|---|---|
| | UDP 500 | any | | SA negotiation (no NAT-T). |
| | UDP 4500 | any | | SA negotiation (NAT-T). |
| sshd | TCP 22 | int | root | SSH services. |
| rsyslog | TCP 6514 | int | root | TLS connections to audit server (common usage) |
| | UDP 515 | int | | Plaintext connections to audit server (common usage). Plaintext connections to the remote audit log server is not allowed in the NIAP evaluated configuration. |

**NOTE**: The Superusers in the Apriva MESA VPN are defined as CLI users. When a CLI user is created, the system creates a userid with the same name. The userid executes the SSHD session and the Command Line Interpreters command prompt. All the components run in Linux user mode (ring-3) except for the Linux operating system

**NOTE**: User credentials, including passwords and private keys must be protected on any other non-TOE platform on which they reside.

# 4 Problem Resolution

The section describes the most common issues encountered during setup and operation of the Apriva MESA VPN server.

This answers the Security Target requirement FPT_TST_EXT.1: “The evaluator shall also ensure that the operational guidance describes the possible errors that may result from such tests and actions the Superuser should take in response; these possible errors shall correspond to those described in the TSS.”

This resolution guide discusses the following possible issues and means of troubleshooting:

- Apriva MESA VPN server user access issues.
- Apriva MESA VPN server service startup / PKI issues.
- Access Policy issues.
- Routing issues.

## 4.1 Apriva MESA VPN Server User Access Issues

The most likely causes of being unable to access the Apriva MESA VPN server are:

- Incorrect cabling or other network infrastructure issues.
- System not configured for remote access.
- Users were not added to the system.
- User credentials missing or invalid.

### 4.1.1 Network Infrastructure Issues

Verify the cabling to the Apriva MESA VPN server is correct. Verify that the management interface port on the Apriva MESA VPN server has an active link. If the link is not active, verify that the Apriva MESA VPN server’s management interface is configured with an IP address appropriate for the management LAN.

If the management interface still does not show an active link, check whatever the Apriva MESA VPN server is attached to (such as a switch) to ensure it has an active port configuration. Test or exchange the Ethernet cable to ensure it is good.

### 4.1.2 System Not Configured for Remote Access

The Apriva MESA VPN server is shipped in a factory default state that requires an initial setup prior to remote access and system configuration. The server must be configured with a management interface appropriate to the network infrastructure, as well as at least one Superuser before it can be accessed remotely.

If the server has not been configured, refer to the ‘Procedure for Setup and Configuration’ section of this Guide to complete the following procedures:

- Log in and change the root password.
- Create a Superuser.

- Configure the management network interface.
- Set access policies.

Once the management interface is accessible from a remote console (SSH), you can log out of the console terminal and continue Apriva MESA VPN server configuration from the remote console. Log in as one of the Superuser users and begin the Apriva MESA VPN server configuration.

Refer to the 'Procedure for Setup and Configuration' section of this guide for procedures on completing an initial system configuration.

### 4.1.3 Users Not Added to the System

If the management interface is operational, it is possible that the user has not been added to the system. Verify that the user attempting to access the system has an account on the system, and that the credentials assigned to the user are valid. See the Apriva MESA VPN User's Guide for information on how to list users and reset passwords.

**NOTE:** Passwords can only be reset at the local terminal console.

### 4.1.4 User Credentials Missing or Invalid

If the user has been added to the system, it is possible their credentials are invalid (lost or forgotten password or key). See the Apriva MESA VPN User's Guide for information on how to list users and how to reset passwords.

**NOTE:** Passwords can only be reset at the local terminal console.

## 4.2 Apriva MESA VPN Server Service Startup Issues / PKI Issues

There are several reasons the VPN might not accept connections from a VPN client. In most cases, there is an Apriva MESA VPN server configuration or PKI issue preventing the VPN service from starting.

The most likely causes of the VPN service failing to start are:

- Incorrect cabling or other network infrastructure issues.
- Interfaces not configured.
- Global Apriva MESA VPN server settings are not defined.
- VPN authentication suite(s) not defined.
- The VPN suite must have certificates available before setting it as an active suite.
- Incorrect or missing VPN trust chains and/or certificates/keys.
- Health monitor check failures.
- VPN client connection issues.
- Routing issues.
- Missing default IP gateway.

### 4.2.1 Network Infrastructure Issues

The Apriva MESA VPN server has two interfaces for device traffic: an external (device-facing) interface and an internal (network services-facing) interface. Both interfaces must be operational for the VPN service to start.

Verify the cabling to the Apriva MESA VPN server is correct. Reference the connection diagrams included in this guide. Verify that the internal and external interfaces ports each have an active link. If the link is not active, verify that the Apriva MESA VPN server's internal and external interfaces are configured with IP addresses appropriate for the management LAN.

If the interfaces still do not show an active link, check whatever the Apriva MESA VPN server is attached to (such as a switch) to ensure it has an active port configuration for each port. Test or swap out the Ethernet cables to ensure they are good.

### 4.2.2 VPN Server Interfaces Not Configured

If the internal and external network interfaces are not configured, refer to the 'Procedure for Setup and Configuration' section of this Guide to complete their configuration. Both external and internal network interfaces must be configured and operational regardless of whether the internal interface is intended to be used.

Verify that both external and internal network interfaces have valid IP addresses appropriate to the network infrastructure.

### 4.2.3 Global VPN Server Settings Not Defined

The Apriva MESA VPN server has both global and authentication-specific configuration parameters. Verify that the following global settings have been configured:

- Authentication rules for carrier ingress (cellular/internet) access.
- Authentication rule for days/times when devices may connect to the Apriva MESA VPN server.
- At least one traffic selector (subnet for devices to route traffic through the VPN tunnel).
- The VPN IP address pool (subnet to use for assigning IP addresses to devices).

### 4.2.4 VPN Service Settings Not Defined or Incorrect

The Apriva MESA VPN server requires that its service information be configured, and that its IP addresses match those of the target network environment. Verify that a VPN service has been created, and that the following service settings have been configured:

- The service id.
- The IP addresses for the internal and external service interfaces.
- The external service IP address is valid for the subnet used for the Apriva MESA VPN server's external interface.
- The internal service IP address is valid for the subnet used for the Apriva MESA VPN server's internal interface.

- The VPN IP address pool (subnet to use for assigning IP addresses to devices).

### 4.2.5 VPN Authentication Suite(s) Not Defined

For devices to authenticate to the Apriva MESA VPN server, the server must be configured to accept either an RSA or ECDSA negotiated connection, or one of each if a mixed population is to be served. These settings are defined by named authentication suites of each type, and one of each type of suite may be chosen for use (at least one suite must be chosen).

**NOTE:** Activating both RSA and ECDSA authentication suites is out of scope of the NIAP Protection profile. Only one authentication suite must be configured.

Verify that an authentication suite of the appropriate type has been defined and selected for use. Verify the settings within the authentication suite to ensure that the appropriate algorithms, ciphers and IKE/PFS group(s) have been defined.

Refer to the 'Procedure for Setup and Configuration' section for procedures on creating and completing VPN authentication suite configuration.

### 4.2.6 Incorrect or Missing Trust Chains or Certificates/Keys

The Apriva MESA VPN server requires a trust chain to be defined. Trust chains are typically composed of a root certificate authority (CA), followed by one or more intermediate CAs and a server CA. The trust chain may include certificate revocation lists (CRLs) for each step in the trust chain, and NIAP requirements enforce that at least one CRL is installed in the chain.

Once a trust chain has been defined, certificates that match that trust chain may be imported.

The trust chain files must be converted to PEM format and cut-and-pasted into the terminal window when instructed.

The certificate and private key must be bundled in PKCS#12 format, converted to PEM format, and cut-and-pasted into the terminal window when instructed.

### 4.2.7 VPN Client Connection Issues

Each device that is attempting to connect to the Apriva MESA VPN server has a certificate that is used to authenticate with the server and must be provisioned on the server. This two-step authentication/authorization sequence along with certificate trust chain and revocation lists provides multiple chokepoints where unauthorized devices can be prevented from accessing network services through the Apriva MESA VPN server.

The most likely client connection issues are:

- Authentication issues due to PKI or certificate mismatches.
- Device provisioning issues.
- Missing access policies.

#### *4.2.7.1 Authentication Mismatch Issues*

If the Apriva MESA VPN server has been configured with global and authentication suite settings, all PKI material and certificates have been imported, and a connection cannot be made, check to see if an authentication error is being reported.

```
/localhost# show log common active
```

The common log collects logs from multiple sources to provide a time-synchronized sequence of events. The logs of interest here are the ones identified by ‘vpn, which is the VPN routing module. Its logs will indicate why a connection failed to be made if the phase 1 (IKE) authentication succeeds.

If the IKE negotiation fails, no errors are logged to prevent a denial-of-service (DoS) attack on the Apriva MESA VPN server by flooding the log. There is a Command Line Interface command that allows the output of phase 1 and phase 2 (IPsec) negotiation packet headers:

```
/localhost> configure
/localhost# vpn-settings
/localhost(config-vpn)# log packets
/localhost(config-vpn)# commit
...
/localhost(config-vpn)# show log common active
```

This will log information about the headers for each packet in the authentication sequence. This will inform whether the client device is able to contact the Apriva MESA VPN server.

**NOTE:** Be sure to turn off this packet log after diagnosing this type of issue or the risk of a DoS attack through log flooding is present.

If the IKE authentication succeeds, but the IPsec authentication fails, then there are other checks to perform:

- Verify that the certificate type on the client matches the configured authentication suite.
- Verify that that client is an industry standard IPSec IKEv2 compatible VPN client. The Apriva MESA VPN server is an RFC compliant Apriva MESA VPN server. For example, strongSwan is a known compatible PC client and Mocana KeyVPN™ is a known compatible mobile client.
- Verify that the correct trust chain and certificate are installed on the VPN client.
- The PKI material for the client connection must be installed in the appropriate manner for the client. Some handsets rely on having a PKCS#12 file with the CA Chain, user certificate, and user key all included. Some clients require the CA Chain to be specified in a separate section than the key and the certificate. Depending on the type of client being used, the end user must make sure to convert the material in an appropriate way for their VPN client to utilize.
- Regarding the server material, the Apriva MESA VPN server only supports importing the CA chain, CRL(s), and certificate/key material separately. The key material must be issued from the same CA as the client, and in the appropriate format for the type of authentication suite that was chosen (Suite B RSA or Suite B ECDSA). To import the PKI material, the certificate/key, CA(s) and CRL(s) must be in PKCS#2 PEM format. It is the Superuser’s responsibility to make sure that the material they attempt to import matches this format.

NIAP requirements state that the CA Chain must have at least one CRL associated with it, in addition to the CA Chain and key material.

- Verify that the client certificate has not been revoked.
- Verify the client certificate has not expired.
- If the IPsec authentication fails due to an 'internal address failure', then the client device has not been provisioned in the system.
- If the IKE and IPsec authentications succeed, but the client still does not connect (or connects and immediately disconnects), it is almost certainly a mismatch in the certificate material between the client and the server.

#### *4.2.7.2 Client Provisioning Issues*

If the IPsec authentication fails due to an 'internal address failure', then the client device has not been provisioned in the system.

Client device certificates must contain a Common Name (CN) field that is set to a unique identifier like a mobile device IMEI, to further tie the identity of the device to a physical attribute. The device is provisioned in the Apriva MESA VPN server using the contents of the CN. The CN is used to identify the device as present within the Apriva MESA VPN server's internal client list, and if found an IP address is issued to the client to conclude the IPSec negotiation phase.

Verify the device is properly provisioned in the Apriva MESA VPN server. If not provisioned, provision the device, and try again to connect the device.

## 4.3 Routing Issues

If clients can make connections to the Apriva MESA VPN server but cannot communicate within the network behind the Apriva MESA VPN server, then it is likely either a routing or access list issue.

The most likely routing or access list issues are:

- Failure to configure a traffic selector in the VPN's global settings.
- Failure to configure access list rules that permit the desired traffic type and destination.
- Failure to configure internal routes in a multi-site network installation.

### 4.3.1 Traffic Selector Configuration

Traffic selectors are used by the Apriva MESA VPN server to tell clients which traffic IP subnets to route through the VPN connection. This includes traffic from networks for services that the Apriva MESA VPN server is protecting as well as other Apriva MESA VPN servers (in a multi-server configuration).

Traffic selectors are a global VPN option and can be configured with the 'client-traffic-selector' command in the server's vpn-settings mode.

### 4.3.2 Access List Rules Configuration

Another common cause of connection issues is a failure to allow specific traffic types with access lists. The Apriva MESA VPN server by default has a preconfigured set of system access rules but denies all other traffic types as a non-alterable policy. The user may add additional access list rules to allow or deny specific traffic by protocol, port, or source or destination IP address or subnet.

### 4.3.3 Routes Configuration

#### *4.3.3.1 Route Configuration*

The Apriva MESA VPN server is preconfigured with system routes, and some routes are set when interfaces are configured. These are labeled as 'unmanaged' routes when displaying the route table using 'show ip routes' and may not be modified by the user. The Apriva MESA VPN server supports the addition of routes through the 'ip route' command. These are 'managed' routes.

All VPN traffic uses the external (ext) interface to communicate with client devices, and the internal (int) interface to communicate with services behind the Apriva MESA VPN server. For VPN client traffic to be routed to and from the Apriva MESA VPN server, the user must set up routes to direct this traffic appropriately.

For the Apriva MESA VPN server to know where to send traffic to services behind the server, it must have routes for the networks on which those services reside, assigned to the internal network interface.

For the Apriva MESA VPN server to send traffic back to devices, it must have routes for the external networks from which the device traffic originates. The external interface is not a default destination for traffic for security reasons, so the server must be explicitly told where to route traffic back to devices.

### 4.3.4 Missing Default Gateway

If the system does not have the default IP gateway specified, the VPN service will not start. Use the ip route command to specify a default gateway. The default gateway should be assigned to the internal (int) network interface.

**DO NOT** assign the default gateway to the external (ext) network interface.

### 4.3.5 Missing Access Policies

For clients to access the Apriva MESA VPN server, an access policy must be defined on the external network interface for IP protocols 50 and UDP ports 500 and 4500 to allow ESP/AH traffic. See "Access Policies" Under "Network Administration Commands" for details on creating access policies.

For the Apriva MESA VPN server to use services such as Syslog and DNS, and access policies must be defined to allow this traffic to enter and leave the server.

For clients to access network services within the protected network zone behind the Apriva MESA VPN server, access policies must be defined to allow this traffic to enter and leave the server.

# 5 Audit Logs

Log files provide a crucial audit trail and can help monitor activity within the IT infrastructure, identify policy violations, pinpoint fraudulent or unusual activity, and highlight security incidents. Security teams can use them to detect and respond to indicators of compromise, investigate, and analyze where an attack is coming or came from, and establish how it has affected the system. As such, the Apriva MESA VPN only permits audit files to be viewed and they are automatically rotated out of real-time usage for archival.

**FAU_GEN.1:** The information below lists the auditable events and provides a format for the audit records. Each audit record format type must be covered, along with a brief description of each field. The evaluator shall check to make sure that every audit event type mandated by the Protection Profile is described and that the description of the fields contains the information required in FAU_GEN.1.2.

The evaluator shall verify that the operational guidance describes how to configure the Packet filter firewall rules to result in applicable network traffic logging.

**NOTE:** This activity should have been addressed with a combination of the guidance assurance activities.

The TSF shall be able to generate an audit record for the following auditable events.

**NOTE:** This list has been condensed by removing the "None" Auditable Event entries from the source data (Table 7: Auditable Events).

FAU_GEN.1.1 The TSF shall be able to generate an audit record for the following auditable events:

a) Start-up and shutdown of the audit functions,
b) All auditable events for the not specified level of audit,
c) All administrative actions; and
d) Specifically defined auditable events listed.

**Table 7: Auditable Events**

| # | SFR | Auditable Events | Additional Audit Record Contents |
|---|---|---|---|
| **5** | FAU_STG.3 /LocSpace | Log storage space for audit events. | No additional information. |
| **14** | FCS_IPSEC_EXT.1 | Session Establishment with peer. | Entire packet contents of packets transmitted/received during session establishment. |
| | | Failure to establish an IPsec SA.<br>Establishment/Termination of an IPsec SA. | Reason for failure.<br>Non-TOE endpoint of connection (IP address) for both success and failures. |

| Table 7: Auditable Events | | | |
|---|---|---|---|
| **#** | **SFR** | **Auditable Events** | **Additional Audit Record Contents** |
| **16** | FCS_SSH_EXT.1 | Failure to establish an SSH session.<br>Establishment/Termination of an SSH session. | Reason for failure.<br>Non-TOE endpoint of connection (IP address) for both successes and failures. |
| **17** | FCS_TLSC_EXT.2 | Failure to establish a TLS Session.<br>Establishment/Termination of a TLS session. | Reason for failure.<br>Non-TOE endpoint of connection (IP address) for both successes and failures. |
| **18** | FIA_AFL.1 | Unsuccessful login attempts limit is met or exceeded. | Origin of the attempt (e.g., IP address) |
| **20** | FIA_UIA_EXT.1 | All use of the identification and authentication mechanism. | Provided user identity, origin of the attempt (e.g., IP address) |
| **21** | FIA_UAU_EXT.2 | All use of the authentication mechanism. | Origin of the attempt (e.g., IP address). |
| **23** | FIA_X509.EXT.1 /Rev | Session Establishment with CA. | Entire packet content of packets transmitted/received during session establishment. |
| | | Unsuccessful attempt to validate a certificate. | Reason for failure. |
| **27** | FMT_MOF.1 /Functions | Modification of the behavior of the transmission of audit data to an external IT entity, the handling of audit data, the audit functionality when Local Audit Storage Space is full. | No additional information. |
| **28** | FMT_MOF.1 /ManualUpdate | Any attempt to initiate a manual update. | No additional information. |
| **29** | FMT_MTD.1/Core Data | All management activities of TSF data. | No additional information. |
| **30** | FMT_MTD.1 /CryptoKeys | Management of cryptographic keys. | No additional information. |
| **33** | FPT_RUL_EXT.1 | Application of rules configured with the ‘log’ operation. | Source and destination address.<br>Source and destination ports.<br>Transport Layer Protocol.<br>TOE interface. |
| | | Indication of packets dropped due to too much network traffic. | TOE interface that is unable to process packets. |
| **38** | FPT_TUD_EXT.1 | Initiation of update; result of the update attempt (success or failure). | No additional information. |
| **40** | FPT_STM_EXT.1 | Discontinuous changes to time – either Administrator actuated or changed via an automated process. (Continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1). | For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g. IP address). |

| Table 7: Auditable Events | | | |
|---|---|---|---|
| **#** | **SFR** | **Auditable Events** | **Additional Audit Record Contents** |
| **41** | FTA_SSL_EXT.1 | The termination of a local session by the session locking mechanism. | No additional information. |
| **42** | FTA_SSL.3 | The termination of a remote session by the session locking mechanism. | No additional information. |
| **43** | FTA_SSL.4 | The termination of an interactive session. | No additional information. |
| **47** | FTP_ITC.1 | Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions. | Identification of the initiator and target of failed trusted channels establishment attempt. |
| **49** | FTP_TRP.1 /Admin | Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions. | No additional information. |

**APPLICATION NOTE**: For session establishment, the expectation is that the TOE can audit all the packets associated with the establishment of a session; this would include the IKE phase 1 and phase 2 negotiations. The TOE must be able to log all the packets in a successful session establishment and can log any packets that were dropped or discarded.

## 5.1 Audit Log Components

To make messages easier to find, the system generates messages depending on the subsystem that generates the message. In this section, each of the auditable events is listed by the subsystem that generates the appropriate logs.

**Common Log - Network**

- FAU_STG.3 /LocSpace Log storage space for audit events.

**Common Log - Secure**

- FCS_SSH_EXT.1 Failure to establish an SSH session. Establishment/Termination of an SSH session. Reason for failure. Non-TOE endpoint of connection (IP address) for both successes and failures.
- FIA_AFL.1 Unsuccessful login attempts limit is met or exceeded. Origin of the attempt (e.g., IP address)
- FIA_UIA_EXT.1 All use of the identification and authentication mechanism. Provided user identity, origin of the attempt (e.g., IP address)
- FIA_UAU_EXT.2 All use of the authentication mechanism. Origin of the attempt (e.g., IP address)
- FTA_SSL.3 The termination of a remote session by the session locking mechanism.
- FTA_SSL.4 The termination of an interactive session.
- FTP_TRP.1 /Admin Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.

**Common Log – VPN and Stunnel**

- FIA_X509.EXT.1 /Rev Session Establishment with CA Entire packet content of packets transmitted/received during session establishment. Unsuccessful attempt to validate a certificate Reason for failure.
- FTP_ITC.1 Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions. Identification of the initiator and target of failed trusted channels establishment attempt.

**Common Log - VPN**

- FCS_IPSEC_EXT.1 Session Establishment with peer Entire packet contents of packets transmitted/received during session establishment. Failure to establish an IPsec SA. Establishment/Termination of an IPsec SA. Reason for failure. Non-TOE endpoint of connection (IP address) for both success and failures.

**Common Log - Stunnel**

- FCS_TLSC_EXT.2 Failure to establish a TLS Session. Establishment/Termination of a TLS session. Reason for failure. Non-TOE endpoint of connection (IP address) for both successes and failures.

**Common Log – AprivaCLI**

- FMT_MOF.1 /Functions Modification of the behavior of the transmission of audit data to an external IT entity, the handling of audit data, the audit functionality when Local Audit Storage Space is full.
- FMT_MOF.1 /ManualUpdate Any attempt to initiate a manual update.
- FMT_MTD.1/CoreData All management activities of TSF data.
- FMT_MTD.1 /CryptoKeys Management of cryptographic keys.
- FPT_TUD_EXT.1 Initiation of update; result of the update attempt (success or failure)
- FPT_STM_EXT.1 Discontinuous changes to time – either Administrator actuated or changed via an automated process. (No continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1) For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g. IP address).

**Common Log – Network Log, IPTables**

- FPT_RUL_EXT.1 Application of rules configured with the 'log' operation Source and destination addresses Source and destination ports Transport Layer Protocol TOE interface Indication of packets dropped due to too much network traffic TOE interface that is unable to process packets.

**Messages**

- FAU_GEN1.1 a Start and stop of the audit log subsystem.

## 5.2 Audit Log Format

This section describes the formats of messages generated by the Apriva MESA VPN server.

### 5.2.1 General Audit Log Format

The Apriva MESA VPN server relies on some open-source components such as OpenSSH, Linux, DNS, and rsyslog. Due to the differences in the development of this open-source software, there are some differences in the formats of the messages that are generated.
The general format of messages is as follows.

```
<Timestamp> <Hostname> <Subsystem> [<process ID] <Message>
```

| Field | Format | Description |
|---|---|---|
| Timestamp | YYYY-MM-DD THH:mm:SS+00:000 | YYYY is the year.<br>MM is the month.<br>DD is the day.<br>mm is minutes.<br>SS is seconds.<br>This timestamp is in the universal time zone (UTC). |
| Hostname | <name> | The name of the host computer that generated the message. |
| Subsystem | <subsystem> | The name of the subsystem that generated the message. |
| Process ID | #### | Process ID of the system or transaction ID. The precise meaning depends on the subsystem. For systems like SSH, this is a transaction ID that can be used to group multiple messages that are associated with the same transaction. |
| Message | <description> | Description of the event or error to be audited. |

As an example:

```
2015-02-16T16:16:50+00:00 SNA-NIAP01 sshd[13717]:
Connection from 172.16.23.20 port 60237 on 172.16.23.120 port 22

<Timestamp>      2015-02-16T16:16:50+00:00
<Hostname>       SNA-NIAP01
<Subsystem>      sshd
[<process ID]    13717
<Message>        Connection from 172.16.23.20 port 60237 on 172.16.23.120 port 22
```

### 5.2.2 Specific Audit Log Formats

Due to the differences in the development of open-source components used in the system, the formats of specific subsystem messages often differ. Specific subsystems include, but are not limited to:

1. sshd – OpenSSH remote login server
2. auditpd – audit service
3. rsyslog – syslog client service
4. aprivatest – periodic test service
5. networklog – network audit log service
6. vpn – VPN service module
7. iptables – iptables (access policy) manager

8. sysman – the command interface

For traceability, each audit log example includes the relevant SFR number(s) who which it applies.

**NOTE:** Audit logs do not include examples of a user locking and unlocking their session because the Apriva MESA VPN server does not support this functionality.

**NOTE**: For the special case of audit logs generated by the Command Line Interface, the subcomponent is reported as a UUID. This has the format as follows:

```
[aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee]
```

The UUID is a session identifier that indicates the associated session number. Each Command Line Interpreter command issued by a user within a command session (local or remote) will contain this UUID value in each of the audit log messages. This can help to track the commands issued by a user.

**NOTE**: Passwords are never written into the log and only the message that the password was changed is shown in the audit log.

## 5.3 Audit Log Rotation

The Log Rotation system runs once per day. If the log threshold is exceeded, the log rotation system rotates the file.

The system will force a rotate very quickly after the threshold is hit (within 30 seconds). A service runs every 30 seconds, checks the threshold, and forces a rotate on all logs if the threshold is exceeded. For each log type, the rotate compresses the current file, deletes the oldest (if above the maximum files kept), and creates a new, empty current file. This will bring the system below the threshold until additional logs are generated to exceed the threshold again. If not, we will repeat above every 30 seconds until compression and/or deletion of the oldest files brings the system below the threshold.

The maximum threshold is 90% and the log partition is almost 700 GB, there is plenty of headroom such that there will soon be a group of tiny files in rotation, with rather large, old files getting deleted from the end of the rotations.

In the case of packet logging, this is performed inside the kernel. When many kernel logs are generated, it is possible that the number of logs may exceed the maximum limit allowed by the kernel, and these logs may be dropped.

**NOTE:** From the Command Line Interpreter point-of-view, the operator can only see the latest log file, so the rest of the rotated files are not accessible. Rotation is there to manage saving as much logging as possible for the target syslog server, in the event there is an extended loss of connection.

## 5.4 Auditable Event Messages

NIAP has defined key Security Functional Requirements (SFR) to protect networked systems within their Common Criteria (CC) collaborative Protection Profile (cPP). The SFRs were created to highlight informational and potentially harmful auditable events for periodic investigation / evaluation. The Apriva MESA VPN satisfies this requirement by capturing auditable events and placing them into 12 logically separate log files. The sections that follow showcase the Apriva MESA VPN log files, identify the aligned SFRs with each log file, and detail the auditable events within the log file.

### 5.4.1 Common Log – Network

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FAU_STG.3 | Log storage space for audit events | |

**VPN Audit Record**

| Message Format | Description of Message and Fields |
|---|---|
| Stopping VPN because audit log is full. | (FAU_STG.3/LocSpace) Indicates that the system has stopped the critical VPN functions because the audit log is full. |
| Performing logrotate because audit log is full. | (FAU_STG.3/LocSpace) Indicates that the system detected that the audit log is nearly full and initiated a logrotate operation to free audit log disk space. |
| Warning: Audit log is <percent-full> % used. | (FAU_STG.3/LocSpace) Indicates that the audit log is becoming full.<br>**Field** — **Description**<br>percent-full — The percentage of the available disk space that is in use. It is full at 100% |
| Stopping service <service-name>. | (FAU_STG.3/LocSpace) Indicates the system has stopped the named service because the audit log is full.<br>**Field** — **Description**<br>service-name — The name of the service that was stopped. This may be VPN, SSH, or Network. |

### 5.4.2 Kernel Log

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| None | | |

**VPN Audit Record**

| Message Format | Description of Message and Fields |
|---|---|
| alg: self-tests for cryptd(__driver-gcm-aes-aesni) (__gcm-aes-aesni) passed | Indicates that the algorithm test for AES-GCM passed. |

| Message Format | Description of Message and Fields |
|---|---|
| alg: self-tests for ctr-aes-aesni (ctr(aes)) passed | Indicates that the algorithm test for AES-CTR passed. |
| alg: self-tests for rfc4106-gcm-aesni (rfc4106(gcm(aes))) passed | Indicates that the algorithm test for AES-GCM passed. |
| alg: self-tests for cryptd(__driver-cbc-aes-aesni) (__cbc-aes-aesni) passed | Indicates that the algorithm test for AES-CBC passed. |
| alg: self-tests for cbc-aes-aesni (cbc(aes)) passed | Indicates that the algorithm test for AES-CBC passed. |

### 5.4.3 Common Log – Secure

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FCS_SSH_EXT.1 | Failure to establish SSH connection | Reason for Failure<br>Non-TOE endpoint of attempted connection (IP Address), None |
| FCS_SSH_EXT.1 | Establishment of SSH connection, None | Non-TOE endpoint of connection (IP Address), None |
| FCS_SSH_EXT.1 | Termination of SSH connection, None | Non-TOE endpoint of connection (IP Address), None |
| FCS_SSH_EXT.1 | Dropping of packet(s) outside defined size limits, None | Packet size, None |
| FIA_AFL.1 | Unsuccessful login attempts limit is met or exceeded | Origin of the attempt (IP address) |
| FIA_UIA_EXT.1 | User Identification and Authentication | Provided user identity, origin of the attempt (IP address) |
| FIA_UIA_EXT.2 | All use of the authentication mechanism | Origin of the attempt (IP address) |
| FTA_SSL.3 | Termination of a remote session by the session locking mechanism | |
| FTA_SSL.4 | Termination of an interactive session | |
| FTP_TRP.1 /Admin | Initiation of the trusted path | |
| FTP_TRP.1 /Admin | Termination of the trusted path | |
| FTP_TRP.1 /Admin | Failure of the trusted path functions | |

**VPN Audit Record**

| Message Format | Description of Message and Fields |
|---|---|
| Accepted password for <user> from <address> port <port> ssh2 | (FIA_UIA_EXT.1) Indicates that a user successfully logged into the Apriva MESA VPN server using a password using the SSH protocol. |

| Field | Description |
|---|---|
| user | The name of the user account that was used to login to the system. |

| Message Format | Description of Message and Fields |
|---|---|
| | address — The IP address of the remote client computer where the login was initiated from.<br>port — The port number of the remote client computer where the login was initiated from. |
| Failed password for <user> from <address> port <port> ssh2 | (FCS_SSH_EXT.1) Indicates that a user failed to login to the Apriva MESA VPN server using a password using the SSH protocol.<br>**Field** — **Description**<br>user — The name of the user account that was used to attempt login to the system.<br>address — The IP address of the remote client computer where the login was initiated.<br>port — The port number of the remote client computer where the login was initiated. |
| Accepted publickey for <user> from <address> port <port> ssh2: <keytype> <key> | (FIA_UIA_EXT.1) A successful authentication using a public key method.<br>**Field** — **Description**<br>user — The name of the user account that was used to login to the system.<br>address — The IP address of the remote client computer where the login was initiated from.<br>port — The port number of the remote client computer where the login was initiated from.<br>keytype — The type of public key used for the authentication. This can be RSA or ECDSA.<br>key — The public key value that was used in the authentication. This is displayed in hexadecimal. |
| Connection from <client-address> port <port> on <server-address> port 22 | (FIA_UAU_EXT.2) Audits the SSH connection from the client at <client-address> and port <port> to the system. The SSH server's address that received the connection is <server-address>. This indicates only a connection was started and does not indicate that a SSH handshake or authentication was made. It will be necessary to look at subsequent messages to find details about the handshake or authentication of the connection.<br>**Field** — **Description**<br>client-address — The IP address of the remote computer that established the connection to the server.<br>port — The port number of the remote computer that established the connection to the server.<br>server-address — The IP address of the SSH server that received the connection from the remote computer. |
| Disconnecting: Too many authentication failures [preauth] | (FIA_AFL.1) The SSH server disconnected the client after receiving too many authentication failures for user <user>.<br>**Field** — **Description** |

| Message Format | Description of Message and Fields |
|---|---|
| error: maximum authentication attempts exceeded for <user> from <address> port <port> ssh2 [preauth] | user: The name of the user account that was used to attempt to login to the system.<br>address: The IP address of the remote client computer where the login was initiated from.<br>port: The port number of the remote client computer where the login was initiated from. |
| Failed publickey for <user> from <address> port <port> ssh2: <keytype> <key> | (FCS_SSH_EXT.1) A client failed to login using the public key method.<br>**Field** / **Description**<br>user: The name of the user account that was used to login to the system.<br>address: The IP address of the remote client computer where the login was initiated from.<br>port: The port number of the remote client computer where the login was initiated from.<br>keytype: The type of public key used for the authentication. This can be RSA or ECDSA.<br>key: The public key value that was used in the authentication. This is displayed in hexadecimal. |
| FIPS mode initialized [preauth] | Indicates that the self-test for FIPS cryptography for the SSH server has passed. |
| No matching cipher found: client <client-cipher-alg> server <server-cipher-alg> [preauth] | (FCS_SSH_EXT.1) The SSH handshake failed due to a cipher algorithm mismatch.<br>**Field** / **Description**<br>client-cipher-alg: The list of cipher algorithms that was proposed by the SSH client.<br>server-cipher-alg: The list of cipher algorithms that was proposed by the SSH server.<br>The handshake failed because no cipher algorithm could be found in common. Cipher algorithms include aes128-cbc or aes256-ctr, etc. This message can be reproduced by supplying a parameter to the SSH client such as "-c 3des-cbc". Since the 3des algorithm is not permitted, this will result in this log message being generated. |
| no matching mac found: client <client-mac-alg> server <server-mac-alg> [preauth] | (FCS_SSH_EXT.1) The SSH handshake failed due to a hash algorithm mismatch.<br>**Field** / **Description**<br>client-mac-alg: The list of mac algorithms that was proposed by the SSH client from the remote computer.<br>server-mac-alg: The list of mac algorithms that was proposed by the SSH server.<br>Mac algorithms include hmac-sha1, hmac-sha256, and hmac-sha384. The error is that no algorithms were found in common that both the client and server could accept. This message can be reproduced by adding a parameter such as "-m hmac-md5" to an SSH client and |

| Message Format | Description of Message and Fields |
|---|---|
| | attempting to connect. Since the MD5 hash algorithm is not permitted, the connection will result in this log message being generated. |
| Postponed publickey for <user> from <address> port <port> ssh2 [preauth] | Indicates the server received a request to begin a public key authentication. If the client proceeds, the administrator should expect to see a subsequent message indicating that the client succeeded or failed the public key authentication.<br>**Field** — **Description**<br>user — The name of the user account that was used to attempt to login to the system.<br>address — The IP address of the remote client computer where the login was initiated from.<br>port — The port number of the remote client computer where the login was initiated from. |
| Unable to negotiate a key exchange method | (FCS_SSH_EXT.1) The SSH handshake failed because the SSH server and client could not negotiate a common Diffie Hellman algorithm group to complete the handshake and key exchange. This error can be reproduced by attempting to use a Diffie Hellman algorithm suite that is not allowed. This can be reproduced by adding a parameter "-o KexAlgorithms=diffie-hellman-group1-sha1" to the SSH client (version 6.6 or better) and attempt a connection. Since DH group 1 is not permitted by the SSH client, this will result in this error message being generated. |
| Received disconnect from <address>: 11: disconnected by user | Indicates the client initiated the disconnection from the server.<br>**Field** — **Description**<br>address — The IP address of the remote client computer where the user disconnected from. |
| Starting session: shell on pts/0 for <user> from <address> port <port> | Audit log that a new session was created for the specified <user>.<br>**Field** — **Description**<br>user — The name of the user account that was associated with the session.<br>address — The IP address of the remote client computer where the login was initiated from.<br>port — The port number of the remote client computer where the login was initiated from. |
| Starting session: shell on pts/4 for <user> from <address> port <port> | (FTP_TRP.1/Admin) Audit log that a session was opened for user <user> from client address <address> and <port>.<br>**Field** — **Description**<br>user — The name of the user account that was associated with the session.<br>address — The IP address of the remote client computer where the login was initiated from.<br>port — The port number of the remote client computer where the login was initiated from. |

| Message Format | Description of Message and Fields |
|---|---|
| Remote SSH session timed out for user [<user>@<address>] | (FTA_SSL.3) (FTP_TRP.1 /Admin) This indicates a server initiated disconnect when the server closed the connection because the specified idle timeout had expired. |
| Error: maximum authentication attempts exceeded for <user> from <address> port <port> ssh2 [preauth]<br><br>Disconnecting: Too many authentication failures [preauth] | (FIA_AFL.1) Indicates the maximum number of authentication attempts has been exceeded.<br>**Field** — **Description**<br>user — The name of the user account that was associated with the session.<br>address — The IP address of the remote client computer where the login was initiated from.<br>port — The port number of the remote client computer where the login was initiated from. |
| pam_faillock(login:auth): Consecutive login failures for user <user> account temporarily locked | (FIA_AFL.1) Indicates the user account was locked because the number of failed logins was exceeded.<br>**Field** — **Description**<br>user — The name of the user account that was locked. |
| pam_unix(login:session): session opened for user <user> by LOGIN(uid=0) | (FIA_AFL.1) Indicates the user account was locked because the number of failed logins was exceeded.<br>**Field** — **Description**<br>user — The name of the user account that was successfully logged in. |
| pam_unix(login:auth): authentication failure; logname=LOGIN uid=0 euid=0 tty=tty1 ruser= rhost= user=<user><br>FAILED LOGIN SESSION FROM tty1 FOR <user>, Permission denied | (FIA_AFL.1) Indicates the user account was locked because the number of failed logins was exceeded.<br>**Field** — **Description**<br>user — The name of the user account that failed logged in. |

| Message Format | Description of Message and Fields |
|---|---|
| (sshd:session): session closed for user <user> | (FTA_SSL.4) (FTP_TRP.1/Admin) Audit log that the session was closed for the specified <user>.<br>**Field** — **Description**<br>user — The name of the user account that owned the session. |
| (login:auth): authentication failure; logname=LOGIN uid=0 euid=0 tty=tty1 ruser= rhost= user=<user> | (FCS_SSH_EXT.1) A user attempted to login from the console using a password and failed.<br>**Field** — **Description**<br>user — The name of the user account that attempted to login. |
| (sshd:session): Unable to negotiate with <address> <port>: no matching host | Indicates the host key type was not found.<br>**Field** — **Description** |

| Message Format | Description of Message and Fields | |
|---|---|---|
| key type found. Their offer: <host key> [preauth] | address | The IP address of the remote client computer that attempted to login. |
| | port | The port number of the remote client computer where the remote. |
| | host key | The host key value returned. |
| (sshd:session): Unable to negotiate with <address> <port>: no matching key exchange method found. Their offer:<key> [preauth] | Indicates the key exchange was not found. | |
| | Field | Description |
| | address | The IP address of the remote client computer where the user disconnected from. |
| | port | The port number of the remote client computer where the login was initiated from. |
| | key | The key value returned. |
| (sshd:session): Bad packet length <value> | Indicates a bad SSH protocol mismatch between the remote client computer and server (bad packet length was identified). | |
| | Field | Description |
| | value | The bad packet length value. |
| (sshd:session): error: maximum authentication attempts exceeded for <user> from <address> <port> [preauth] | Indicates the maximum number of user login authentication attempts was exceeded. | |
| | Field | Description |
| | user | The name of the user account that was associated with the session. |
| | address | The IP address of the remote client computer where the login was initiated from. |
| | port | The port number of the remote client computer where the login was initiated from. |
| (sysman:session) User <user> executed command [enable] | Indicates the enable command was executed. | |
| | Field | Description |
| | user | The name of the user account that was associated with the session. |
| | address | The IP address of the remote client computer where the login was initiated from. |
| (sysman:session) Successful login attempt from user <user> | Indicates a successful login attempt has occurred. | |
| | Field | Description |
| | user | The name of the user account that was associated with the session. |
| (login): FAILED LOGIN SESSION FROM <device> FOR <user> Permission denied | Indicates a failed login attempt has occurred. | |
| | Field | Description |
| | device | The name of the device the user was attempting to login from. |
| | user | The name of the user account that was associated with the failed login attempt. |
| | Indicates a failed login attempt because the password is incorrect. | |
| | Field | Description |

| Message Format | Description of Message and Fields | |
|---|---|---|
| (sysman:session): Failed password for <user> from <address> <port> | user | The name of the user account that was associated with the failed login attempt. |
| | address | The IP address of the remote client computer where the login was initiated from. |
| | port | The port number of the remote client computer where the login was initiated from. |

The passwd command will generate a message in the audit log that appears as follows:

| Message Format | Description of Message and Fields | |
|---|---|---|
| [<UUID>] User [<user>@<address>] executed command [passwd <username>]. | The passwd command generates an audit log that contains the following fields. | |
| | **Field** | **Description** |
| | UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| | user | This is the name of the account (aka userid) that issued the command. |
| | address | The IP address of the computer where the command was issued from. For console commands this will be 127.0.0.1. For remote SSH sessions this will be the IP address of the remote computer where the SSH client connected from. |
| | username | The name of the user account that will have the password updated. |

### 5.4.4 Common Log – VPN

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FIA_X509.EXT.1 / Rev | Session Establishment with CA Entire packet content of packets transmitted/received during session establishment | Reason for failure |
| FIA_X509.EXT.1 / Rev | Unsuccessful attempt to validate a certificate | Reason for failure |
| FTP_ITC.1 | Initiation of the trusted channel | Identification of the initiator and target of failed trusted channels establishment attempt |
| FTP_ITC.1 | Termination of the trusted channel | Identification of the initiator and target of failed trusted channels establishment attempt |
| FTP_ITC.1 | Failure of the trusted channel functions | Identification of the initiator and target of failed trusted channels establishment attempt |
| FCS_IPSEC_EXT.1 | Session Establishment with peer Entire packet contents of packets transmitted/received | Reason for failure<br><br>Non-TOE endpoint of connection (IP address) for both success and failures |

| | | |
|---|---|---|
| | during session establishment | |
| FCS_IPSEC_EXT.1 | Failure to establish an IPsec SA | Reason for failure<br><br>Non-TOE endpoint of connection (IP address) for both success and failures |
| FCS_IPSEC_EXT.1 | Establishment/Termination of an IPsec SA | Reason for failure<br><br>Non-TOE endpoint of connection (IP address) for both success and failures |
| FIA_X509_EXT.1 / Rev | Unsuccessful attempt to validate a certificate | Reason for failure |

## VPN Audit Record

<table>
<tr><th>Message Format</th><th>Description of Message and Fields</th></tr>
<tr><td>DAEMON ERROR: failed dhcp own_ip address pool attributes</td><td>(FTP_ITC.1) The VPN service is not configured correctly.</td></tr>
<tr><td>AUTH INFO: External key provider disabled</td><td>The authentication suite is configured incorrectly, or certificates are missing.</td></tr>
<tr><td>DAEMON ERROR: Reconfiguration failed4</td><td>The authentication suite is configured incorrectly, or certificates are missing.</td></tr>
<tr><td>TUNNEL INFO: Local IKE peer <server-address>:<server-port> ID <server-subject> (dn)<br>TUNNEL INFO: Remote IKE peer <client-address>:<client-port> ID <client-subject> (dn)<br>AUTH INFO: ESP [<esp>] | [cf3ee72a] | <cipher-alg>/<cipher-keysize> - <hash-alg></td><td>An IPSEC Tunnel was successfully created by the client at the specified address and port using the specified algorithms.
<table>
<tr><th>Field</th><th>Description</th></tr>
<tr><td>server-address</td><td>The IP address of the server that received the request to create a tunnel from the client.</td></tr>
<tr><td>server-port</td><td>The port number of the server that received the request to create a tunnel from the client. NOTE: For IKE, this will usually be the standard port 500.</td></tr>
<tr><td>server-subject</td><td>The subject name of the certificate presented by the server to the client.</td></tr>
<tr><td>client-address</td><td>The IP address of the server that received the request to create a tunnel from the client.</td></tr>
<tr><td>client-port</td><td>The port number of the server that received the request to create a tunnel from the client.</td></tr>
<tr><td>client-subject</td><td>The subject name of the certificate presented by the client to the server.</td></tr>
<tr><td>cipher-alg</td><td>The name of the cipher algorithm and mode used for the encryption. This will be aes-gcm or aes-cbc.</td></tr>
<tr><td>cipher-keysize</td><td>The size of the cipher algorithm key in bits. This is 128 or 256 bits.</td></tr>
<tr><td>hash-alg</td><td>The name of the hash algorithm used for authentication. For “combined” type ciphers, the hash value is “none”. For regular ciphers like AES-CBC, this is the hash algorithm like SHA-256 or SHA-384.</td></tr>
</table></td></tr>
<tr><td>TUNNEL INFO: SA deleted [IPSEC-SA-ESP-<esp>-R] [<id>] [<ip-</td><td>(FTP_ITC.1) The IPSEC VPN tunnel was closed by the client.</td></tr>
</table>

| Message Format | Description of Message and Fields |
|---|---|

address>] [<tunnel-address>] terminated-by-client

| Field | Description |
|---|---|
| Esp | The ESP identifier for the tunnel that was closed. |
| Id | The subject identifier of the client that closed the tunnel. |
| ip-address | The outside IP address of the client that closed the tunnel. |
| tunnel-address | The inside-tunnel assigned IP address of the client that closed the tunnel. |

AUTH INFO: IKEv2 SA [Responder] negotiation failed:
AUTH INFO: Local IKE peer <server-address>:<server-port> ID (null)
AUTH INFO: Remote IKE peer <client-address>:<client-port> ID (null)
AUTH INFO: Message: <message>
AUTH INFO: Reason:
AUTH INFO: <reason>
AUTH INFO: IKE SA negotiations: <total-count> done, <success-count> successful, <fail-count> failed

(FTP_ITC.1) (FIA_X509_EXT.1/Rev) The SA establishment failed. Note: The VPN interface associated with the SA is always the external network interface (ext). No other interfaces can support VPN tunnels.

| Field | Description |
|---|---|
| server-address | The IP address of the server that received the request to establish a VPN tunnel SA. |
| server-port | The port number of the server that received the request to establish a VPN tunnel SA. |
| client-address | The IP address of the VPN client that sent the request to establish a VPN tunnel SA. |
| client-port | The port number of the VPN client that sent the request to establish a VPN tunnel SA. |
| message | A textual description of the failure. The following values are possible:<br>• Unsupported critical payload<br>• Invalid ike SPI<br>• Invalid major version<br>• Invalid syntax<br>• Invalid message<br>• Invalid spi<br>• No proposal chosen<br>• Invalid KE payload<br>• Authentication failed<br>• Single pair required<br>• No additional SAs<br>• Internal address failure<br>• Failed CP required<br>• TS unacceptable<br>• Invalid selectors<br>• Unacceptable address<br>• Unexpected NAT detected<br>• Temporary failure<br>• Child SA not found<br>• Initial contact<br>• Additional TS possible<br>• ESP TFC padding not supported<br>• Encryption algorithm mismatch |
| reason | Some failures include additional data indicating the reason for the failure. The following values are possible:<br>• Algorithm mismatch between the certificate and the search constraints |

| Message Format | Description of Message and Fields | |
|---|---|---|
| | | • Algorithm or key not allowed (not strong enough)<br>• Certificate chain looped (did not find trusted root)<br>• Certificate contains critical extension that was not handled<br>• Certificate decoding failed<br>• Certificate is not valid<br>• Certificate issuer was not valid (CA specific information missing)<br>• Certificate signature was not verified correctly<br>• Certificate was not added to the cache<br>• Certificate was not found (anywhere)<br>• Certificate was not valid in the time interval<br>• Certificate was revoked by a CRL<br>• CRL contains duplicate serial numbers<br>• CRL decoding failed<br>• CRL is not currently valid, but in the future<br>• CRL is not valid<br>• CRL is too old<br>• CRL signature was not verified correctly<br>• CRL was not added to the cache<br>• CRL was not found (anywhere)<br>• Database method failed<br>• Database method failed due to timeout<br>• Key usage mismatch between the certificate and the search constraints<br>• Maximum path length reached<br>• Path was not verified<br>• Time information not available<br>• Time interval is not continuous |
| | total-count | A count of the total number of attempted SA connections. |
| | success-count | A count of the total number of successful SA connections. |
| | fail-count | A count of the total number of failed SA connections. |
| Opening <connect-type> connection to URL <url> : <result> | The Apriva MESA VPN Server attempted to open a connection to a CRL distribution point or OCSP server at the specified URL. | |
| | **Field** | **Description** |
| | connect-type | The type of connection. This can be HTTP. |
| | url | This is the URL that the system used to establish the connection to the CRLDP or OCSP server. |
| | result | This is a result of success or failure of the attempt to connect to the remote server. |

| Message Format | Description of Message and Fields |
|---|---|
| | Note: This message provides additional information about the CRLDP or OCSP server. The source and destination information are provided in a separate message described below. |
| Establishing <type> connection from <source-address>:<source-port> to <dest-address>:<dest-port> on interface <interface> : <status> | The Apriva MESA VPN server has established a connection to a CRLDP or OCSP server. |

| Field | Description |
|---|---|
| type | The type of connection. This is always TCP. |
| source-address | The source address of the connection. The source is the address of the VPN server. |
| source-port | The source port of the connection. |
| dest-address | The destination address of the connection. The destination address is the address of the remote CRLDP or OCSP server. |
| dest-port | The destination port of the connection |
| interface | The source interface of the server where the connection was initiated. |
| status | The status of the connection. This is success or failed. |

<packet-type> packet R(<src-address>:<src-port> <dest-address>:<dest-port): len= <length>, mID=0, HDR, N(NAT_DETECTION_SOURCE_IP), N(NAT_DETECTION_DESTINATION_IP)

<action> packet <src-address>:<src-port> to <dest-address>:<dest-port>. Packet dump:
<hex-contents>

(FCS_IPSEC_EXT.1) This is the result of the "log packet" command. The system logs the entire packet content.

| Field | Description |
|---|---|
| packet-type | The type of packet being logged. For example, IKEv2. |
| action | The direction of the packet. This can be Sending or Receiving |
| src-address | The source address of the packet |
| src-port | The source port of the packet |
| dest-address | The destination address of the packet |
| dest-port | The destination port of the packet |
| len | The packet length in bytes |
| hex-contents | The content of the packet displayed in hexadecimal. |

TLS connection from initiator <initiator> to destination <dest-address> was terminated.

(FTP_ITC.1) This reports that the secure TLS channel was terminated.

| Field | Description |
|---|---|
| initiator | This is the system that initiated the connection. This is always the TOE. |
| dest-address | The destination address of the secure connection. |

TLS connection from initiator <initiator> to destination <dest-address> failed.

(FTP_ITC.1) This reports that the secure TLS channel failed.

| Field | Description |
|---|---|
| initiator | This is the system that initiated the connection. This is always the TOE. |
| dest-address | The destination address of the secure connection. |

TLS connection from initiator <initiator> to destination <dest-address> was successfully initiated

(FTP_ITC.1) This reports that the secure TLS channel was successfully initiated.

| Field | Description |
|---|---|
| initiator | This is the system that initiated the connection. This is always the TOE. |
| dest-address | The destination address of the secure connection. |

Accepting to connect from initiator <initiator> to destination <dest-address>

(FTP_ITC.1) This reports that the system is attempting to initiate a connection to a remote system.

| Field | Description |
|---|---|
| initiator | This is the system that initiated the connection. This is always the TOE. |
| dest-address | The destination address of the secure connection. |

### 5.4.5 Common Log – Stunnel and TLS

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FIA_X509.EXT.1 / Rev | Session Establishment with CA Entire packet content of packets transmitted/received during session establishment | Reason for failure |
| FIA_X509.EXT.1 / Rev | Unsuccessful attempt to validate a certificate | Reason for failure |
| FTP_ITC.1 | Initiation of the trusted channel | Identification of the initiator and target of failed trusted channels establishment attempt |
| FTP_ITC.1 | Termination of the trusted channel | Identification of the initiator and target of failed trusted channels establishment attempt |
| FTP_ITC.1 | Failure of the trusted channel functions | Identification of the initiator and target of failed trusted channels establishment attempt |
| FCS_TLSC_EXT.2 | Failure to establish a TLS Session | Reason for failure<br><br>Non-TOE endpoint of connection (IP address) for both successes and failures |
| FCS_TLSC_EXT.2 | Establishment/Termination of a TLS session | Reason for failure<br><br>Non-TOE endpoint of connection (IP address) for both successes and failures |
| FIA_X509_EXT.1 / Rev | Unsuccessful attempt to validate a certificate | Reason for failure |

**VPN Audit Record**

| Message Format | (SFR #) Description of Message and Fields |
|---|---|
| OCSP: Connecting the AIA responder <ocsp-url> | Indicates the system is attempting to connect to an OCSP server. |

| Field | Description |
|---|---|
| ocsp-url | The URL of the OCSP server. |

<table>
<tr><th>Message Format</th><th>(SFR #) Description of Message and Fields</th></tr>
<tr><td>OCSP: Connected <ocsp-address>:<ocsp-port></td><td>Indicates the connection to the OCSP server is successful.<br><table><tr><th>Field</th><th>Description</th></tr><tr><td>ocsp-address</td><td>The address of the OCSP server.</td></tr><tr><td>ocsp-port</td><td>The port of the OCSP server. Normally 80.</td></tr></table></td></tr>
<tr><td>OCSP: Status: good</td><td>Indicates that the OCSP response is that the certificate is not revoked.</td></tr>
<tr><td>OCSP: Status: revoked<br>OCSP: Certificate revoked: 4: superseded</td><td>Indicates the OCSP response is that the certificate is revoked.</td></tr>
<tr><td>CERT: Host name <host-address> matched with <cert-address></td><td>Indicates the OCSP host and server certificate match.<br><table><tr><th>Field</th><th>Description</th></tr><tr><td>host-address</td><td>The address of the OCSP server.</td></tr><tr><td>cert-address</td><td>The address in the OCSP server’s certificate.</td></tr></table></td></tr>
<tr><td>(stunnel: session): CERT: Pre-verification error: unsupported certificate purpose<br>(stunnel: session): Rejected by CERT at depth=<depth>, <name></td><td>Indicates the OCSP has rejected the unsupported certificate.<br><table><tr><th>Field</th><th>Description</th></tr><tr><td>depth</td><td>The certificate depth.</td></tr><tr><td>name</td><td>The name of the certificate being checked.</td></tr></table></td></tr>
<tr><td>(stunnel: session): CERT: Pre-verification error: unable to get local issuer certificate</td><td>Indicates the OCSP has rejected the certificate because it is unable to obtain the local issuer certificate.<br><table><tr><th>Field</th><th>Description</th></tr><tr><td>depth</td><td>The certificate depth.</td></tr><tr><td>name</td><td>The name of the certificate being checked.</td></tr></table></td></tr>
<tr><td>(stunnel: session): Cert: No matching host name found<br>(stunnel: session): Rejected by CERT at depth=<depth>, <name></td><td>Indicates the OCSP has rejected the certificate because there is no matching host name found.<br><table><tr><th>Field</th><th>Description</th></tr><tr><td>depth</td><td>The certificate depth.</td></tr><tr><td>name</td><td>The name of the certificate being checked.</td></tr></table></td></tr>
<tr><td>(stunnel: session): CERT: Pre-verification error: invalid CA certificate<br>(stunnel: session): Rejected by CERT at depth=<depth>, <name></td><td>Indicates an invalid certificate was used.<br><table><tr><th>Field</th><th>Description</th></tr><tr><td>depth</td><td>The certificate depth.</td></tr><tr><td>name</td><td>The name of the certificate being checked.</td></tr></table></td></tr>
<tr><td>(stunnel: session): CERT: Pre-verification error: Certificate has expired<br>(stunnel: session): Rejected by CERT at depth=<depth>, <name></td><td>Indicates the certificate used has expired.<br><table><tr><th>Field</th><th>Description</th></tr><tr><td>depth</td><td>The certificate depth.</td></tr><tr><td>name</td><td>The name of the certificate being checked.</td></tr></table></td></tr>
<tr><td>(stunnel: session): TLS connection from initiator <user> to destination <dest-address> was successfully initiated</td><td>Indicates a TLS connection has been initiated.<br><table><tr><th>Field</th><th>Description</th></tr><tr><td>user</td><td>The name of the user account that was associated with the session.</td></tr><tr><td>destination</td><td></td></tr></table></td></tr>
</table>

| Message Format | (SFR #) Description of Message and Fields |
|---|---|
| | The destination address of the secure connection. |
| (stunnel: session): TLS connection from initiator <user> to destination <dest-address> was terminated | .<br>Indicates a TLS connected was terminated.<br>Field: Description<br>user: The name of the user account that was associated with the session.<br>destination: The destination address of the secure connection. |
| (stunnel: session): SSL_connect:<process id>: error:<process id>:SSL routines:ssl3_get_server_hello:wrong cipher returned | Indicates the wrong cipher value was returned.<br>Field: Description<br>process id: The process id. |
| (stunnel: session): SSL_connect:<process id>: error:<process id>:SSL routines:ssl3_get_server_hello:unknown cipher returned | Indicates an unknown cipher value was returned.<br>Field: Description<br>process id: The process id. |
| (stunnel: session): SSL_connect:< process id>: error:< process id>:SSL routines:ssl3_read_bytes:sslv3 alert handshake failure | Indicates a handshake failure occurred.<br>Field: Description<br>process id: The process id. |
| (stunnel: session): SSL_connect:< process id>: error:< process id>:SSL routines: ssl3_get_server_hello:wrong ssl version | Indicates the wrong SSL version was used.<br>Field: Description<br>process id: The process id. |
| (stunnel: session): SSL_connect:< process id>: error:< process id>:SSL routines: SSL3_GER_RECORD:decription failed or bad record mac | Indicates a bad record mac.<br>Field: Description<br>process id: The process id. |
| Rejected by OCSP at depth=<depth>:<name> | (FIA_X509_EXT.1/Rev) Indicates the OCSP server reported the certificate is revoked.<br>**Field**: **Description**<br>depth: The certificate depth.<br>name: The name of the certificate being checked. |
| Error resolving <ocsp-server-address>: Neither nodename nor servname known (EAI_NONAME) | Indicates that the attempt to connect to the OCSP server because the DNS name of the OCSP server could not be resolved or is invalid.<br>**Field**: **Description**<br>ocsp-server-address: The address of the OCSP server that failed. |
| SSL_connect:: error::SSL routines:ssl3_get_server_certificate:certificate verify failed | (FCS_TLSC_EXT.2) (FIA_X509_EXT.1/Rev) Indicates that the validation of the certificate failed. |
| | (FCS_TLSC_EXT.2) Indicates the TLS connection failed because the TLS server rejected the connection. |

| Message Format | (SFR #) Description of Message and Fields |
|---|---|
| SSL_connect: error: SSL routines:ssl3_read_bytes:tlsv1 alert internal error | |
| SSL_connect:: error: SSL routines:ssl3_read_bytes:sslv3 alert certificate expired | (FCS_TLSC_EXT.2) (FIA_X509_EXT.1/Rev) Indicates the certificate has expired. |

### 5.4.6 Common Log – AprivaCLI

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FMT_MOF.1 / Functions | Modification of the behavior of the transmission of audit data to an external IT entity, the handling of audit data | |
| FMT_MOF.1 / CoreData | All management activities of TSF data | |
| FMT_MOF.1 / CryptoKeys | Management of cryptographic keys | |

These audit logs are recorded using the following audit logs. All the commands related to manipulating TSF data, management of keys, and configuration of audit logs are audited by recording the full content of each command. See the user guide for the details of all available commands such as the logging scope, crypto import, crypto delete, and configuration of TSF data.

**VPN Audit Record**

| Message Format | Description of Message and Fields |
|---|---|
| [UUID] User [<user>@<address>] executed command [<command>] | (FMT_MTD.1/CoreData) Indicates an administrator has executed a command to modify or view configuration data. |

| Field | Description |
|---|---|
| user | The name of the user that issued the command |
| address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |
| command | The command that was executed |

**NOTE:** This guidance document contains audit logs for each successful CLI command in the section that describes the command. The Apriva MESA VPN server does not log failed command attempts; failures in syntax or parameter usage are reported to the operator's session window in an interactive manner. This satisfies FAU_GEN.1.1 b) and c).

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|

| | | |
|---|---|---|
| FMT_MOF.1 / Functions | Modification of the behavior of the transmission of audit data to an external IT entity, the handling of audit data | |
| FMT_MOF.1 / ManualUpdate | Any attempt to initiate a manual update | |
| FTA_SSL_EXT.1 | Termination of a local session by the session locking mechanism (timeout of the local console) | |

## VPN Audit Record

| Message Format | Description of Message and Fields |
|---|---|
| Software Update was requested and started | (FMT_MOF.1/ManualUpdate) (FPT_TUD_EXT.1) Indicates that a software update was initiated. |
| Software Update requested. Validating the software update files and signatures | Indicates that the software update digital signatures are being validated. |
| Repo update manifest does not exist on the CDROM | (FPT_TUD_EXT.1) Indicates that the software update failed because the update manifest file could not be found. |
| Digital Signature failed for manifest <file> | (FIA_X509_EXT.1/Rev) Indicates that the software update failed because the digital signature of the software update failed validation.<br>**Field** — **Description**<br>file — The name of the manifest file that failed. |
| Repo update manifest gpg does not exist on the CDROM | (FPT_TUD_EXT.1) Indicates the software update failed because the software manifest digital signature file is missing. |
| RPM folder does not exist | (FPT_TUD_EXT.1) Indicates the software update failed because the package folder could not be found. |
| Repodata folder does not exist | (FPT_TUD_EXT.1) Indicates the software update failed because the update data could not be found. |
| Manifest check failed | (FPT_TUD_EXT.1) Indicates that the software update failed because the manifest signature failed. |
| The update version number is not more recent than the current version. Update canceled | (FPT_TUD_EXT.1) Indicates the software update failed because the current system is newer or the same version as the update. The system does not support installing updates that are older than the current installed version. |
| Validation was successful | (FPT_TUD_EXT.1) Indicates the software update digital signature validation was successful. |
| Software Update version <version> passed validation. Installing… | (FPT_TUD_EXT.1) Indicates the software update digital signature validation was successful, and that the software is being installed.<br>**Field** — **Description** |

| Message Format | Description of Message and Fields |
|---|---|
| | version — The version number of the software update being installed. |
| Update is being installed, version number is <version> | (FPT_TUD_EXT.1) Indicates the software update digital signature validation was successful, and that the software is being installed.<br>**Field** — **Description**<br>version — The version number of the software update being installed. |
| Config file for CDROM repo update does not exist. | (FPT_TUD_EXT.1) Indicates the software update failed because of missing update data. |
| Gateway Repository Update did not complete | (FPT_TUD_EXT.1) Indicates the software update was attempted but failed with errors. |
| manifest gpg signature is not valid. | (FPT_TUD_EXT.1) Indicates the software update digital signature failed. |
| manifest verification: missing file <filename> on the CDROM | (FPT_TUD_EXT.1) Indicates the software update failed because of a missing manifest file.<br>**Field** — **Description**<br>filename — The name of the file that is missing from the software update data. |
| verification: invalid hash <filename> | (FPT_TUD_EXT.1) Indicates the software update failed because of an invalid signature hash.<br>**Field** — **Description**<br>filename — The name of the file that caused the digital signature to fail due to an integrity failure. |
| rpm signature not valid <filename> | (FPT_TUD_EXT.1) Indicates the software update failed because of an invalid or untrusted signature.<br>**Field** — **Description**<br>filename — The name of the file that caused the software update to fail because of an invalid digital signature. |
| rpm count <rpm-count> is not the same as the manifest rpm count <manifest-count> | (FPT_TUD_EXT.1) Indicates the software update failed because the number of package files does not match the expected number of files.<br>**Field** — **Description**<br>rpm-count — The number of RPM files in the software update<br>manifest-count — The number of expected RPM files in the update. |
| Repodata files are not the same as the manifest count | (FPT_TUD_EXT.1) Indicates the software update failed because the software update data does not match the digital signature. |
| Software Update is complete. Initiating system reboot | (FPT_TUD_EXT.1) Indicates the software update successfully completed and the system will restart to complete the installation. |
| Console session timed out for user [<user>@<address>] | (FTA_SSL_EXT.1) Indicates that the console session command interface for the console user was closed by the server (not by the user) due to an idle timeout where the timeout value was exceeded. |

| Field | Description |
|---|---|
| user | The name of the user that was previously logged in and the system forced to disconnect due to an idle timeout. |
| address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

Remote session closed for user [<user>@<address>]

Indicates that a user's session was closed. This message is reported regardless of whether the session was closed because the user logged out, the server disconnected the user due to an idle timeout, or if the connection was severed due to some network failure. To determine which of these cases occurred, the administrator should look for additional log messages that indicate why the session was closed.

| Field | Description |
|---|---|
| user | The name of the user account that that owned the session. |
| address | The IP address of the computer that was previously logged into the system. This may be 127.0.0.1, which indicates the local machine. For remote SSH, this address is the address of the remote computer that the user used to make the connection to the Apriva MESA VPN. |

User [<user>@<address>] logged out of the system

Indicates that the user logged out of the system (the user explicitly requested a logoff using the exit command).

| Field | Description |
|---|---|
| user | The name of the user account that requested the logoff |
| address | The IP address of the computer that was used to perform the logoff. For the console, this will be 127.0.0.1, which indicates the local machine. For remote SSH, this address is the address of the remote computer that the user used to make the connection to the Apriva MESA VPN. |

Successful login attempt from user [<user>@127.0.0.1]

Indicates a user successfully logged in from the console using proper password.

| Field | Description |
|---|---|
| user | The name of the user account that successfully logged in. |

(sysman: session) Started service 'chronyd'

Indicates a service was started.

(sysman: session) Stopped service 'chryond'

Indicates a service was started.

(sysman: session) User <user> executed command [login]

Indicates the system logon banner was displayed upon successful user login.

(sysman: session) User <user> executed command [console-timeout 60]

Indicates a console time-out occurred.

| | |
|---|---|
| | **Field** / **Description** |
| | user — The name of the user account that executed the command. |
| (sysman: session) User <user> executed command [max-failed-attempts 4] | Indicates a user reached the maximum failed login attempts. |
| | **Field** / **Description** |
| | user — The name of the user account that successfully logged in. |
| (sysman: session) Generated private <algorithm/key size> key <name> | Indicates a user generated a private key. |
| | **Field** / **Description** |
| | algorithim — The name of the private key algorithm used.<br>name — The name of the key (algorithm /size). |
| (sysman: session) host changed: old value = <old host>, new value = <new host> | Indicates host was changed. |
| | **Field** / **Description** |
| | old host — Old host.<br>new host — New host. |
| (sysman: session) ciphers changed: old value = <old cipher>, new value = <new cipher> | Indicates a cipher was changed. |
| | **Field** / **Description** |
| | old cipher — Old cipher.<br>new cipher — New cipher. |
| (sysman: session) User <user> executed command [crypto csr generate sha384 VPN_csr vpnKeyRSA] | Indicates a CSR was generated. |
| | **Field** / **Description** |
| | user — The name of the user account that created the CSR. |

## Security Functional Requirement

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FPT_STM_EXT.1 | Discontinuous changes to time – either Administrator actuated or changed via an automated process | For discontinuous changes to time: The old and new values for the time<br><br>Origin of the attempt to change time for success and failure (IP address) |

## VPN Audit Record

| Message Format | Description of Message and Fields |
|---|---|
| <UUID> System date changed by [<user>] from <time1> to <time2> | (FPT_STM_EXT.1) The time has been changed by the user. The previous time value and new time value are displayed. |

| Field | Description |
|---|---|
| UUID | This is a UUID number that specifies the session number of the user that issued the command. |
| time1 | The old-time value of the system before the change. |
| time2 | The new time value of the system after the change. |

| Message Format | Description of Message and Fields |
|---|---|
| Starting Apriva SYSMAN Service daemon | (FAU_GEN1.1. a) The Audit system has started. |
| Exiting the Apriva SYSMAN Service daemon | (FAU_GEN1.1. a) The Audit system has stopped. |

| Message Format | Description of Message and Fields |
|---|---|
| [UUID] User [<user>@<address>] executed command logging] | (FMT_MOF.1/Functions) Indicates the administrator has performed changes to the audit log behavior. |

| Field | Description |
|---|---|
| user | The name of the user that issued the command. |
| address | The IP address of the user. This address |

| Message Format | Description of Message and Fields |
|---|---|
| [UUID] CA Certificate '<cert-id>' was successfully imported into trustchain <trustchain> by user <user> at address <address> | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys. |

| Field | Description |
|---|---|
| cert-id | The certificate ID that was imported. The certificate ID has the format <type>-<id>. For example, VPN-id1. |
| trustchain | The name of the trust chain that the certificate was imported into. |
| user | The name of the user that issued the command. |
| address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

| Message Format | Description of Message and Fields |
|---|---|
| [UUID] Trustchain '<trustchain-id>' was successfully deleted by user <user> at address <address> | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys |

| Field | Description |
|---|---|
| trustchain-id | The trustchain ID that was deleted. The trustchain ID has the format <type>-<id>. For example, VPN-tc1. |
| user | The name of the user that issued the command. |
| address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

| Message Format | Description of Message and Fields |
|---|---|
| [UUID] CA Certificate '<cert-id>' was successfully removed from trustchain <trustchain> by user <user> at address <address> | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys |

| Field | Description |
|---|---|
| cert-id | The certificate ID that was deleted. The certificate ID has the format <type>-<id>. For example, VPN-id1. |
| trustchain | The name of the trust chain that the certificate was deleted from. |

| Message Format | Description of Message and Fields | |
|---|---|---|
| | user | The name of the user that issued the command. |
| | address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

| Message Format | Description of Message and Fields | |
|---|---|---|
| [UUID] CRL '<crl-id>' was successfully imported into trustchain <trustchain> by user <user> at address <address> | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys | |
| | **Field** | **Description** |
| | crl-id | The crl ID that was imported. The crl ID has the format <type>-<id>. For example, VPN-crl1. |
| | trustchain | The name of the trust chain that the certificate was imported into. |
| | user | The name of the user that issued the command. |
| | address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

| Message Format | Description of Message and Fields | |
|---|---|---|
| [UUID] Certificate '<cert-id>' was successfully imported into trustchain <trustchain> by user <user> at address <address> | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys | |
| | **Field** | **Description** |
| | cert-id | The certificate ID that was imported. The certificate ID has the format <type>-<id>. For example, VPN-id1. |
| | trustchain | The name of the trust chain that the certificate was imported into. |
| | user | The name of the user that issued the command. |
| | address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

| Message Format | Description of Message and Fields | |
|---|---|---|
| [UUID] Certificate '<cert-id>' was successfully deleted from trustchain <trustchain> by user <user> at address <address> | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys | |
| | **Field** | **Description** |
| | cert-id | The certificate ID that was deleted. The certificate ID has the format <type>-<id>. For example, VPN-id1. |
| | trustchain | The name of the trust chain that the certificate was imported into. |
| | user | The name of the user that issued the command. |
| | address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

| Message Format | Description of Message and Fields | |
|---|---|---|
| [UUID] SSH Key '<key-id> of type <type> was successfully imported by user <user> at address <address> | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys | |
| | **Field** | **Description** |

| Message Format | Description of Message and Fields | |
|---|---|---|
| | key-id | The key ID that was imported. This has the format 'SSH-<userid>-<name>', where the userid is the name of the user that was assigned the key and the name is the key name specified by that userid. |
| | type | The type of the key (the algorithm and key size) |
| | user | The name of the user that issued the command. |
| | address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

| Message Format | Description of Message and Fields | |
|---|---|---|
| [UUID] SSH Key '<key-id> was successfully deleted by user <user> at address <address> | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys. | |
| | **Field** | **Description** |
| | key-id | The key ID that was deleted. This has the format 'SSH-<userid>-<name>', where the userid is the name of the user that was assigned the key and the name is the key name specified by that userid. |
| | user | The name of the user that issued the command. |
| | address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |

**NOTE:** The audit log will contain a separate log message for each cryptographic change. See the user's guide for all the available commands such as import or delete.

**NOTE:** The audit log will contain multiple log messages for each command that was executed in relation to the audit log system and is applied when the commit command is executed. The changes are audited with the full command content of each command used to change the logging.

| Message Format | Description of Message and Fields | |
|---|---|---|
| [UUID] User [<user>@<address>] executed command crypto <command>] | (FMT_MOF.1/CryptoKeys) Indicates the administrator has performed changes to cryptographic keys. | |
| | **Field** | **Description** |
| | user | The name of the user that issued the command. |
| | address | The IP address of the user. This address is expected to be 127.0.0.1, which means the local machine of the console. |
| | command | The command that was executed. |

**NOTE:** The audit log will contain a separate log message for each cryptographic change. See the user's guide for all the available commands such as import or delete.

### 5.4.7 Common Log – Network Log and IPTables

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FPT_RUL_EXT.1 | Application of rules configured with the 'log' operation Source and destination addresses | Source and destination ports Transport Layer Protocol TOE interface Indication of packets dropped due to too much network traffic<br><br>TOE interface that is unable to process packets |

**VPN Audit Record**

| Message Format | Description of Message and Fields | |
|---|---|---|
| Dropped <number-packets> packets on interface <interface-name><br><br>OR<br><br>Dropped <number-packets> <type> packets on interface <interface-name> | (FPT_RUL_EXT.1) Indicates that packets were dropped due to too much network traffic. | |
| | **Field** | **Description** |
| | number-packets | The number of packets that were dropped. |
| | type | The packet type (send or receive) of packet that was dropped. |
| | interface-name | The name of the interface where the packets were dropped. This is the network interface name such as eth0. |

### 5.4.8 Messages Log

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FAU_GEN1.1 a | Start and stop of the audit log subsystem | |

**VPN Audit Record**

| Message Format | Description of Message and Fields | |
|---|---|---|
| node=<node> type=CRED_ACQ msg=audit(xxxx): user pid=<process> uid=0 auid=500 ses=<session> subj=unconfined_u :system_r:sshd_t:s0-s0:c0.c1023 msg='op=PAM:setcred acct="<userid>" exe="/usr/sbin/sshd" hostname=<host> addr=<address> terminal=ssh res=success' | Indicates a successful SSH login for user <user> using session <session> from client address <address>. | |
| | **Field** | **Description** |
| | userid | The name of the user account that was used to login with the Apriva MESA VPN. |
| | node | The node name where the login occurred. |
| | session | The session id of the session that was created for the login. |
| | host | The host name of the computer. |
| | port | The port number of the remote system that initiated the login. |
| | address | The address of the remote client computer that initiated the login. |

| Message Format | Description of Message and Fields |
|---|---|

| Message Format | Description of Message and Fields |
|---|---|
| audispd:node=localhost.localdomain.com type=SERVICE_STOP msg=audit(): pid=1 uid=0 msg='unit=rsyslog comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success' | (FAU_GEN1.1) Indicates the audit subsystem has stopped. |

| Message Format | Description of Message and Fields |
|---|---|
| localhost rsyslogd: [origin software="rsyslogd" swVersion="8.24.0" x-pid="21269" x-info="http://www.rsyslog.com"] start localhost audispd:node =localhost.localdomain.com type=SERVICE_START msg=audit(): pid=1 uid=0 msg='unit=rsyslog comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success' | (FAU_GEN1.1) Indicates the audit subsystem has started. |

### 5.4.9 Self Test Log

**Security Functional Requirement**

| Requirement | Auditable Events | Additional Audit Record Contents |
|---|---|---|
| FPT_RUL_EXT.1 | Application of rules configured with the 'log' operation | Source and destination addresses<br><br>Source and destination ports, Transport Layer Protocol, TOE Interface |
| FIA_X509_EXT.1 / Rev | Session establishment with CA | Entire packet contents of packets transmitted/received during session establishment |

**NOTE:** These two audit requirements share the same format. The session establishment to the CA is logged by adding a log rule to log all packets from and to the CA.

**VPN Audit Record**

| Message Format | Description of Message and Fields |
|---|---|
| FIPS test failed | Indicates that the FIPS test has failed. |
| SHA256 Fail | Indicates that the SHA 256 hash operation has failed. |
| FIPS test <result> | Indicates that the FIPS test result has been determined. This can report Passed or Failed.<br>**Field** — **Description**<br>result — The result of the test. This can be Passed or Failed. |
| AIDE configuration file <filename> not found | Indicates there is a failure because the AIDE configuration could not be found. |

| Message Format | Description of Message and Fields |
|---|---|
| | **Field** / **Description**<br>filename — The name of the configuration file that could not be opened. |
| AIDE database not present. Update required. | Indicates that the Configuration Integrity test failed because the AIDE database could not be found. |
| Integrity Test Failed - Filtered <count> failures <failed-count> | Indicates that the integrity test has failed. This will result in a fail-secure.<br>**Field** / **Description**<br>count — The number of failures found.<br>failed-count — The number of integrity failures found. |
| Entropy device <device-name> not found | Indicates that the entropy test failed because the entropy device could not be found.<br>**Field** / **Description**<br>device-name — The name of the entropy device that could not be found. |
| urandom device not found | Indicates the entropy test failed because the /dev/urandom device could not be located. |
| Apriva device is not urandom. inodes do not match | Indicates that the entropy test failed because the entropy device does not have the appropriate inode value. |
| Entropy device version check failed | Indicates the entropy test failed because the entropy device driver has the wrong version number. |
| Entropy failed | Indicates the entropy test failed. |
| Failed to read entropy data | Indicates that the entropy test failed because the request to read entropy failed. |
| Enropy test <result> | Indicates the result of the entropy test.<br>**Field** / **Description**<br>result — The result of the test. Can be Passed or Failed. |
| Power on Self Test Started | Indicates that the Power on Self-test has started. |
| Starting Integrity Test | Indicates the Integrity test has started. |
| Executable Integrity test <result> | Reports the result of the executable integrity test.<br>**Field** / **Description**<br>result — The result of the test. Can be Passed or Failed. |
| Configuration Integrity test <result> | Indicates the results of the configuration integrity test.<br>**Field** / **Description**<br>result — The result of the test. Can be Passed or Failed. |
| Self test has completed | Indicates that the self-test has completed. |
| Power on test <result> | Reports the result of the power on test.<br>**Field** / **Description** |

| Message Format | Description of Message and Fields | |
|---|---|---|
| | result | The result of the test. Can be Passed or Failed. |
| Periodic Test Started | Indicates the periodic self-test has started. | |
| Selftest has completed | Indicates the self-test has completed. | |

## 5.4.10 IPTables Logs

### VPN Audit Record

| Message Format | Description of Message and Fields | |
|---|---|---|
| kernel: <log-prefix> IN=<interface-name> OUT= MAC=<MAC-address> SRC=<source-ip> DST=<dest-ip> LEN=<packet-length> TOS=<tos> PREC=<prec> TTL=<ttl> ID=<id> <fragment> PROTO=<protocol> SPT=<source-port> DPT=<dest-port> WINDOW=<window> RES=<res> <tcp-flags> URGP=<urgp><br><br>kernel: <log-prefix> IN= OUT=<interface-name> SRC=<source-ip> DST=<dest-ip> LEN=<packet-length> TOS=<tos> PREC=<prec> TTL=<ttl> ID=<id> <fragment> PROTO=<protocol> SPT=<source-port> DPT=<dest-port> WINDOW=<window> RES=<res> <tcp-flags> URGP=<urgp><br><br>kernel: <log-prefix> IN=<interface-name> OUT=<interface-name> MAC=<MAC-address> SRC=<source-ip> DST=<dest-ip> LEN=<packet-length> TOS=<tos> PREC=<prec> TTL=<ttl> ID=<id> <fragment> PROTO=<protocol> SPT=<source-port> DPT=<dest-port> WINDOW=<window> RES=<res> <tcp-flags> URGP=<urgp><br><br>kernel: <log-prefix> IN= OUT=<interface-name> MAC=<MAC-address> SRC=<source-ip> DST=<dest-ip> LEN=<packet-length> TOS=<tos> PREC=<prec> TTL=<ttl> ID=<id> <fragment> PROTO=<protocol> TYPE=<icmp-type> CODE=<icmp-code> ERR=<icmp-err> ID=< icmp-id> SEQ=<icmp-seq> | (FPT_RUL_EXT.1) (FIA_X509_EXT.1/Rev) Audit log that a packet was logged through use of the 'log' option in an access policy match rule. | |
| | **Field** | **Description** |
| | log-prefix | The user-defined log prefix. |
| | interface-name | The Apriva MESA VPN server's internal interface name. This will be one of the Linux names eth0, eth1, eth2, etc. |
| | MAC-address | The MAC address of the outbound packet (only present for outbound packets). |
| | source-ip | The source IP address of the packet. |
| | dest-ip | The destination IP address of the packet. |
| | packet-length | The packet's length in bytes. |
| | tos | Packet IP type-of-service field as defined in RFCs 2474 and 3168. |
| | prec | Packet IP precedence. |
| | ttl | Packet IP time-to-live. |
| | id | Packet ID. |
| | fragment | Packet IP fragmentation option (may not be present in all logs). |
| | protocol | Packet IP protocol value. |
| | source-port | The packet's source port (for port-oriented protocols; may not be present in all logs). |
| | dest-port | The packet's destination port (for port-oriented protocols; may not be present in all logs). |
| | window | For TCP protocols, the TCP window size. |
| | res | For TCP protocols, TCP reserved bits. |
| | flags | For TCP protocols, TCP message flags. |
| | urgp | For TCP protocols, the URGENT pointer. |
| | icmp-type | For ICMP protocols, the ICMP packet type. |
| | icmp-code | For ICMP protocols, the ICMP packet code. |
| | icmp-err | For ICMP protocols, the ICMP error code (may not be present in all ICMP protocol logs). |

| Message Format | Description of Message and Fields | |
|---|---|---|
| | icmp-id | For ICMP protocols, the ICMP packet ID. |
| | icmp-seq | For ICMP protocols, the ICMP packet sequence number. |

### 5.4.11 Audit Log Discussion

In many cases, the system may generate multiple audit log messages containing different levels of information. To collect all the information available about a specific condition, it is often necessary to read multiple audit logs. For systems such as SSH, the transaction ID field of the message can be used to associate the multiple messages that related to the same transaction. For example:

1. Connection from <client-address> port <port> on <server-address> port 22.
2. Protocol error: no matching DH grp found.

In this example, the address and port of the connection is displayed in the first message, and then the result of the handshake (in this case, a handshake failure) is displayed in the second message. Collecting the information from both messages provides complete information about the source address and port of the connection attempt and the result.

1. Connection from <client-address> port <port> on <server-address> port 22
2. Postponed publickey for <user> from <address> port <port> ssh2 [preauth]
3. Failed publickey for <user> from <address> port <port> ssh2: <keytype> <key>

In this example, a single connection results in multiple log messages. The first message reports that a connection was started. The second indicates an initiation of a public key authentication, and the third message indicates the result of the authentication (in this case, a failure).

## 5.5 Relationship between local logs and remote audit server logs

The Apriva MESA VPN generates audit data, and all audit data is available locally. It can also forward audit data to a remote server and only forwards that data when the connection to the audit server is available. The Apriva MESA VPN does not implement queueing of audit data when a connection to an external audit server is not available. Audit data generated when a connection to an external audit server is unavailable will be available locally only.

# 6 Access Policy Management

This section describes access policy behavior and the mechanisms for managing access policies in the Apriva MESA VPN server.

The CLI in the Apriva MESA VPN server provides a flexible rule mechanism to manage access policies. The management mechanism was designed to provide the user with control over fine details such as the interface and direction to which the rules apply. The system will deny packets unless explicitly allowed in the access-policies for the interfaces in question and the default configuration is to deny / drop all packets (inbound, outbound, and forwarded).

In addition, the Apriva MESA VPN server enforces a restriction on the input/output paths for the external (device-facing) interface. Only IPsec (protocols 50/51, UDP ports 500/4500) and ICMP are allowed on that interface. This is non-configurable from the perspective of allowing additional traffic, but an access policy must be created to allow this traffic on the external network interface for device VPN connections to be established.

Section 6.1 provides a brief overview of the CLI commands used in managing access policies; please reference the sections that specify the commands for further details. Section 6.2 provides examples, including complete, real-world examples starting from creating an ACL rule set through assigning the ACL to an interface.

The following is a sample ICMP command set that creates rule sets and access control lists (ACLs), populating them with match rules, and assigning the ACLs to network interfaces.

```
access-list int-forward {
        match 10 action device-traffic
        match 30 proto icmp action permit
        match 31 action permit
    }
```

**NOTE:** In the above example, the match 30 lines says its ok to respond to ping requests. if that line were not there and that was the only policy applied to the interface in question, the ping would fail.

## 6.1 Access Policy Command Overview

Access policy management is accomplished using four CLI commands:

1. “ruleset” to create a rule set and enter the scope of the rule set,
2. “access-list” to create/modify a named ACL and enter the scope of that ACL,
3. “match” to create an ordered rule within a rule set or ACL,
4. “access-list” to assign an ACL to an interface (executed within the scope of the target interface).

Each of these commands can be prepended with “no” to remove the ACL, rule, or group mapping respectively.

### 6.1.1 Rule Sets

Rule sets are collections of match rules that perform basic actions (permit, deny, return, reject, log, monitor). Rule sets are commonly used when a collection of matches is required in multiple access lists to reduce duplication.

The “ruleset” command enters the scope of a named rule set to allow modification, or if the rule set does not exist, creates an empty one and enters its scope.
The format of the command is:

```
ruleset <name>
```

The “name” parameter is an alphanumeric string that can also include colons and underscores.
Once the user is in the ruleset scope, match rules can be entered using the “match” command described below.

### 6.1.2 Access Lists

Access lists are the collections of rule sets and match rules that are assigned to network interfaces to enact the access policy. Access lists are commonly called ‘ACLs’ (access control lists).
ACLs may use rule sets, match rules and other ACLs to create an access policy. ACLs are arranged by order of reference to other ACLs, regardless of the order in which they are created; in other words, an ACL that is referenced by another ACL will be installed first.
The “access-list” command enters the scope of a named ACL to allow modification, or if the ACL does not exist, creates an empty one and enters its scope.
The format of the command is:

```
access-list <name>
```

The “name” parameter is an alphanumeric string that can also include colons and underscores.
Once the user is in the scope of the ACL, match rules can be entered using the “match” command described below.

### 6.1.3 Match Rules

Match rules are individual rules that are applied to each packet to determine if an action should be taken. Rules may contain IP addresses, subnet masks, protocol values, and port values to be matched against those in the data packet. If a match is found, then a specified action is performed.

Actions consist of things to be done when a match is made, such as to allow the packet to traverse the server (permit), silently drop the packet (deny), drop the packet, and send a notification to the sender (reject), log the packet was seen (log), send a copy of the packet to a defined destination (monitor), or stop processing (return).

Match rules are ordered by an index value, which may be updated to rearrange the match rules within a rule set or ACL. There is a limit of 255 match rules per rule set or ACL, and the index is limited to a value between 1 and 999999 inclusive.

The “match” command is used to specify the rules that will be applied within a given rule set or ACL. The command requires an “action” parameter to specify the action (permit, deny, etc.) to take if the rule is matched. All other parameters are optional or conditional. The format of the command is:

```
match [index] [proto <protocol>] [src <cidr>] [srcport_<port>] [dst <cidr>]
              [dstport <port>] action <action> [log-prefix <prefix>]
              [log-level <0-7>] [monitor-target <target>]
```

Valid protocols are ip, hopopt, icmp, igmp, ggp, ipv4, st, tcp, cbt, egp, igp, bbn-rcc, nvp, pup, argus, emcon, xnet, chaos, udp, mux, dcn, hmp, prm, xns-idp, trunk-1, trunk-2, leaf-1, leaf-2, rdp,

irtp, iso-tp4, netblt, mfe-nsp, merit-inp, dccp, 3pc, idpr, xtp, ddp, idpr-cmtp, tp++, il, ipv6, sdrp, ipv6-route, ipv6-frag, idrp, rsvp, gre, dsr, bna, esp, ipv6-crypt, ah, ipv6-auth, i-nlsp, swipe, narp, mobile, tlsp, skip, ipv6-icmp, ipv6-nonxt, ipv6-opts, cftp, sat-expak, kryptolan, rvd,ippc, sat-mon, visa, ipcv, cpnx,cphb, wsn, pvp, br-sat-mon, sun-nd, wb-mon, wb-expak, iso-ip, vmtp, secure-vmtp, vines, ttp, nsfnet-igp, dgp, tcf, eigrp, ospf, sprite-rpc, larp, mtp, ax.25, ipip, micp, scc-sp, etherip, encap, gmtp, ifmp, pnni, pim, aris, scps, qnx, a/n, ipcomp, snp, compaq-peer, ipx-in-ip, vrrp, pgm, l2tp, ddx, iatp, stp, srp, uti, smp, sm, ptp, isis, fire, crtp, crudp, sscopmce, iplt, sps, pipe, sctp, fc, rsvp-e2e-ignore, mobility-header, udplite, mpls-in-ip, manet, hip, shim6, wesp, rohc

The Apriva MESA VPN does not support IPv6 within a tunnel, however it does support IPv4 within tunnels.

Iptables is the tool that governs access to the system. The iptables default is to block everything and only allow what is explicitly allowed in. As such, iptables are configurable to control the behavior for incoming packets (input), what packets get passed on (forwarded), and packets permitted to be sent out of the system (output).

The rules are evaluated in index order. If no index is specified, the first rule created is set to an index of 10. The index is then incremented by 10 for each additional rule created without a specified index. The default protocol is “ip” unless the “proto” parameter is specified. Network addresses and ports (src, srcport, dst, dstport parameters) that are not specified default to zeros, which indicate that “any” will match. The logging and monitoring parameters are not applicable, except for actions of “log” and “monitor” respectively.

#### *6.1.3.1 ACL Loops*

Since ACLs can be chained (one references another, which references another), it is possible to design an ACL topology with a loop (ACL 1 references ACL 2, which references ACL 1). The Apriva MESA VPN server checks for loops when ACL match rules are entered at the command line or when importing a configuration.

If a loop is found:

The rule is not created.

An error that a loop was detected is displayed.

A configuration-style display of the possible loops that would be created is displayed.

### 6.1.4 Access Policy Assignment

Once ACLs are defined, they must be assigned to a network interface to be applied. ACLs are applied to a network interface by name and direction (in, out or forward).
The “access-list” command within the interface scope is used to map an ACL to a network interface.
The format of the command is:

```
access-list <acl-name> <direction>
```

Both parameters are required. The “acl-name” parameter must be the name of an existing ACL. Valid values for the “direction” parameter are: “in”, “out” and “forward”. Once the ACL is mapped to the interface, the specified rules are active modifications to the firewall policy.

### 6.1.5 Display Configuration Commands

Information about the ACLs and the ACL interface mappings is included in the configuration output provided by:

```
show running-config
```

To filter the output to include only information about the ACLs use the “access-policy” parameter:

```
show running-config access-policy
```

To filter the output to include only information about the interfaces (including the ACL interface mappings) use the “interface” parameter:

```
show running-config interface
```

## 6.2 Access Policy Management Examples

### 6.2.1 NTP Service Example

This example shows how to set up the internal interface on a server to accept NTP traffic and allow ping activity. First a rule set named “ntp” is created and populated with the NTP rules. Then the ACL named “int-in” for the internal interface is created and populated with a call to the “ntp” rule set, along with a rule to accept ping requests. Finally, the “int-in” ACL is mapped for input to the internal interface. The setup details follow:

```
/NIAPVPN01.apriva.com(config)# access-policy
/NIAPVPN01.apriva.com(config-apol)# ruleset ntp
/NIAPVPN01.apriva.com(config-apol-rule)# match srcport 123 proto udp action permit
/NIAPVPN01.apriva.com(config-apol-rule)# match dstport 123 proto udp action permit
/NIAPVPN01.apriva.com(config-apol-rule)# exit
/NIAPVPN01.apriva.com(config-apol)# access-list int-in
/NIAPVPN01.apriva.com(config—apol-acl)# match action ntp
/NIAPVPN01.apriva.com(config-apol-acl)# match proto icmp action permit
/NIAPVPN01.apriva.com(config-apol-acl)# exit
/NIAPVPN01.apriva.com(config-apol)# exit
/NIAPVPN01.apriva.com(config)# interface int
/NIAPVPN01.apriva.com(config-iface)# access-list int-in in
/NIAPVPN01.apriva.com(config-iface)# exit
/NIAPVPN01.apriva.com(config)# commit
Stopping service 'iptables': SUCCESS!
Starting service 'iptables': SUCCESS!
Changes committed.
/NIAPVPN01.apriva.com(config)# show running-config access-policy
ruleset ntp {
match 1 proto udp srcport 123 action permit
match 2 proto udp dstport 123 action permit
}
access-list int-in {
match 1 action ntp
match 2 proto icmp action permit
}
/NIAPVPN01.apriva.com(config)# show running-config interface
interface int {
access-list int-in in
```

```
ip address 172.16.87.63/24
mtu 1450
}
/NIAPVPN01.apriva.com(config)#
```

# 7 Standards

This section provides international standards detail for a Commercial Apriva MESA VPN server. The MESA limits this to a limited number of strong security features per NIAP requirements.

## 7.1 Supported Requests for Comments (RFCs)

The Apriva® IPsec Server is a true international standard–based design to assist in removing vendor proprietary elements in gateway designs.
The following RFCs (Request For Comments), working documents of the Internet Engineering Task Force are supported. The RFC documents are available at www.ietf.org.
**NOTE**: The client devices that connect to the Apriva IPsec VPN server must follow the same RFCs and protocol definitions to successfully connect to the server.

| | |
|---|---|
| RFC 822 | Standard for the format of ARPA Internet text messages: Email Address |
| RFC 2003 | IP Encapsulation within IP |
| RFC 2104 | HMAC: Keyed-Hashing for Message Authentication |
| RFC 2314 | PKCS #10: Certification Request Syntax Version 1.5 |
| RFC 2315 | PKCS #7: Cryptographic Message Syntax Version 1.5 |
| RFC 2393 | IP Payload Compression Protocol (IPComp) |
| RFC 2394 | IP Payload Compression Using DEFLATE |
| RFC 2401 | Security Architecture for the Internet Protocol (superseded by RFC 4301) |
| RFC 2402 | IP Authentication Header |
| RFC 2403 | The Use of HMAC-MD5-96 within ESP and AH |
| RFC 2404 | The Use of HMAC-SHA-1-96 within ESP and AH |
| RFC 2405 | The ESP DES-CBC Cipher Algorithm with Explicit IV |
| RFC 2406 | IP Encapsulating Security Payload (ESP) |
| RFC 2407 | The Internet IP Security Domain of Interpretation for ISAKMP **NOTE:** This document has been superseded by RFC 4306 (‘Internet Key Exchange (IKEv2) Protocol’) |
| RFC 2408 | Internet Security Association and Key Management Protocol (ISAKMP) **NOTE:** This document has been superseded by RFC 4306 (‘Internet Key Exchange (IKEv2) Protocol’) |
| RFC 2409 | The Internet Key Exchange (IKE) **NOTE:** This document has been superseded by RFC 4306 (‘Internet Key Exchange (IKEv2) Protocol’) |
| RFC 2410 | The NULL Encryption Algorithm and Its Use With IPsec |
| RFC 2411 | IP Security Document Roadmap |
| RFC 2412 | The OAKLEY Key Determination Protocol |
| RFC 2437 | PKCS #1: RSA Cryptography Specifications Version 2.0 |
| RFC 2451 | ESP CBC-Mode Cipher Algorithms |
| RFC 2459 | Internet X.509 Public Key Infrastructure Certificate and CRL Profile |
| RFC 3280 | Internet X.509 Public Key Infrastructure Certificate and CRL Profile |
| RFC 2510 | Internet X.509 Public Key Infrastructure Certificate |
| RFC 2511 | Internet X.509 Certificate Request Message Format |
| RFC 2661 | Layer Two Tunneling Protocol (L2TP) |
| RFC 2663 | IP Network Address Translator (NAT) Terminology and Considerations |
| RFC 2694 | DNS extensions to Network Address Translators (DNS ALG) |

| | |
|---|---|
| RFC 3022 | Traditional IP Network Address Translator |
| RFC 3027 | Protocol Complications with the IP Network Address Translator |
| RFC 3193 | Securing L2TP using IPsec |
| RFC 3526 | More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE) |
| RFC 3554 | On the Use of Stream Control Transmission Protocol (SCTP) with IPsec |
| RFC 3566 | The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec |
| RFC 3602 | The AES-CBC Cipher Algorithm and Its Use with IPsec |
| RFC 3664 | The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol |
| RFC 3686 | Using AES Counter Mode With IPsec ESP |
| RFC 3706 | A Traffic-Based Method of Detecting Dead Internet Key Exchange (IKE) Peers |
| RFC 3748 | Extensible Authentication Protocol (EAP) |
| RFC 3947 | Negotiation of NAT-Traversal in the IKE |
| RFC 3948 | UDP Encapsulation of IPsec ESP Packets |
| RFC 4106 | The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP) |
| RFC 4109 | Algorithms for Internet Key Exchange version 1 (IKEv1) |
| RFC 4231 | Identifiers and Test Vectors for HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512 |
| RFC 4301 | Security Architecture for the Internet Protocol (excluding policy decorrelation) |
| RFC 4302 | IP Authentication Header |
| RFC 4303 | IP Encapsulating Security Payload (ESP) |
| RFC 4304 | On Extended Sequence Number (ESN) Addendum to IPsec Domain of Interpretation (DOI) for Internet Security Association and Key Management Protocol (ISAKMP) |
| RFC 4306 | Internet Key Exchange (IKEv2) Protocol<br>**NOTE:** This document has been superseded by RFC 5996 (‘Internet Key Exchange Protocol (IKEv2)’, also known as ‘IKEv2bis’).<br>**NOTE:** This RFC states that if message IDs grow too large to fit in 32 bits, the IKE SA must be closed. This functionality is not currently supported. |
| RFC 4307 | Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2) |
| RFC 4308 | Cryptographic Suites for IPsec |
| RFC 4346 | The Transport Layer Security (TLS) Protocol Version 1.1 |
| RFC 4434 | The AES-XCBC-PRF-128 Algorithm for the Internet Key Exchange Protocol (IKE) |
| RFC 4543 | The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH. |
| RFC 4555 | Mobility and Multihoming Protocol (MOBIKE) |
| RFC 4621 | Design of the IKEv2 Mobility and Multihoming (MOBIKE) Protocol |
| RFC 4634 | U.S. Secure Hash Algorithms (SHA and HMAC-SHA) |
| RFC 4718 | IKEv2 Clarifications and Implementation Guidelines |
| RFC 4739 | Multiple Authentication Exchanges in the Internet Key Exchange (IKEv2) Protocol |
| RFC 4753 | ECP Groups for IKE and IKEv2. (This specification has been superseded by RFC 5903.) |

| | |
|---|---|
| RFC 4754 | IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA) |
| RFC 4787 | Network Address Translation (NAT) Behavioral Requirements for Unicast UDP |
| RFC 4835 | Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH) |
| RFC 4868 | Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 With IPsec |
| RFC 4869 | Suite B Cryptographic Suites for IPsec |
| RFC 5114 | Additional Diffie-Hellman Groups for Use with IETF Standards. |
| RFC 5216 | The EAP-TLS Authentication Protocol |
| RFC 5282 | Using Authenticated Encryption Algorithms with the Encrypted Payload of the Internet |
| RFC 5903 | Elliptic Curve Groups modulo a Prime (ECP Groups) for IKE and IKEv2 |
| RFC 5930 | Using Advanced Encryption Standard Counter Mode (AES-CTR) with the Exchange version 02 (IKEv2) Protocol |
| RFC 5246 | The Transport Layer Security (TLS) Protocol Version 1.2 |
| RFC 5996 | Internet Key Exchange Protocol Version 2 (IKEv2) (also known as 'IKEv2bis') **NOTE:** This RFC states that if message IDs grow too large to fit in 32 bits, the IKE SA must be closed. This functionality is not currently supported. |
| RFC 5998 | An Extension for EAP-Only Authentication in IKEv2 |