JUNIPER NETWORKS

Network Configuration Example

Configuring and Troubleshooting Public Key Infrastructure

Release
11.3

Published: 2011-09-22

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Network Configuration Example Configuring and Troubleshooting Public Key Infrastructure
Release 11.3

Revision History
September 2011—R1 Junos OS 11.3

The information in this document is current as of the date listed in the revision history.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. The Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at http://www.juniper.net/support/eula.html. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

CHAPTER 1

Overview of PKI in Junos OS

This topic includes the following section

Introduction to PKI in Junos OS

This document provides an overview of PKI (Public Key Infrastructure) in Junos OS, architecture of the virtual private network (VPN) based on PKI authentication, step-by-step instructions on configuring and troubleshooting of PKI in Junos OS, and FAQ (frequently asked questions) on PKI with respect to your SRX series or J-Series devices.

This document is intended for network design and security engineers, as well as anyone that requires secure connectivity over public networks.

NOTE: For more details on digital certificates, see the *Junos System Basics Guide* available at: http://www.juniper.net/techpubs/software/junos/.

For information on crypto, RSA, and PKI, visit the website http://www.rsasecurity.com/rsalabs.

For a list of PKI-related technical terms, see the "Glossary of PKI Related Terms" on page 72.

This topic describes the basic elements of public key infrastructure (PKI) in Junos OS, including components of the PKI, certificate life cycle management, and usage within Internet Key Exchange (IKE) and includes the following sections:

Fundamentals of the PKI

Junos OS is the Juniper Networks single operating system and provides the following features:

- Powerful operating system with rich IP services toolkit.
- Unmatched IP dependability and security to ensure an efficient and predictable IP infrastructure.
- Enhanced security and VPN capabilities from Juniper Networks Firewall/IP Security (IPsec) VPN platforms including the SSG product family.

PKI Applications Overview

The Junos OS uses public/private keys in the following areas:

- SSH/SCP (for secure command-line interface [CLI]-based administration)
- Secure Sockets Layer (SSL) (for secure Web-based administration and for https-based webauth for user authentication)
- Internet Key Exchange (IKE) (for IPsec VPN tunnels)

NOTE: Note the following points:

- Currently the Junos OS supports only IKE (using public key infrastructure (PKI) certificates for public key validation).
- Support for identity binding with SSL is currently not available. A brief section on SSL is included in this example. For more information, see "Overview on Usage of SSL and IPsec/IKE Methods" on page 7.
- The SSH and SCP are used exclusively for system administration and depends on the use of out-of-band fingerprints for public key identity binding and validation. Details on SSH are not covered in this example.

Components for Administering PKI in Junos OS

The following components are required for administrating PKI in Junos OS:

- CA certificates and authority configuration
- Local certificates including the devices identity (example: IKE ID type and value) and private and public keys
- Certificate validation through a certificate revocation list (CRL)

Basic Elements of PKI in Junos OS

Junos OS supports three specific types of PKI objects:

- Private/public key pair

- Certificates
  - Local certificate—The local certificate contains the public key and identity information for the Juniper Networks device. The Juniper Networks device owns the associated private key. This certificate is generated based on a certificate request from the Juniper Networks device.
  - Pending certificate — A pending certificate contains a key pair and identity information that is generated into a PKCS10 certificate request and manually sent to a certificate authority (CA). While the Juniper Networks device waits for the certificate from the CA, the existing object (key pair and the certificate request) is tagged as a certificate request or pending certificate.

    NOTE: Junos OS Release 9.0 or later supports automatic sending of certificate requests through SCEP. For more information, see "Appendix D: Simple Certificate Enrollment Protocol" on page 70 .

  - CA certificate — When the certificate is issued by the CA and loaded into the Junos device, the pending certificate is replaced by the newly generated local certificate. All other certificates loaded into the device are considered CA certificates.
- Certificate revocation lists (CRLs)

Note the following points about certificates:

- Local certificates are generally used when a Junos device has VPNs in more than one administrative domain.
- All PKI objects are stored in a separate partition of persistent memory, apart from the Junos image and the system's general configuration.
- Each PKI object has a unique name or certificate-ID given to it when it is created and maintains that ID until its deletion. You can view the certificate-ID using the CLI command **show security pki local-certificate** .
- A certificate cannot be copied from a device (generally). The private key on a device must be generated on that device only, and it should never be viewed or saved from that device. So PKCS12 files (which contain a certificate with the public key and the associated private key) are not supported on Junos devices.
- CA certificates validate the certificates received by the IKE peer. If the certificate is valid, then it is verified in the CRL to see whether the certificate has been revoked.

  Each CA certificate includes a CA profile configuration that stores the following information:
  - CA identity, which is typically the domain name of the CA
  - E-mail address for sending the certificate requests directly to the CA

- Revocation settings:
  - Revocation check enable/disable option
  - Disabling of revocation check incase of CRL download failure.
  - Location of CDP (CRL Distribution Point) (for manual URL setting)
  - CRL refresh interval

Junos OS supports multiple local certificates, depending on the device size. See "Appendix A: Frequently Asked Questions" on page 49 for details.

Table 1 on page 4 provides information on possible PKI objects and their average sizes.

Table 1: PKI Objects and Average Sizes

PKI Objects	Average Sizes
Private/public key pair	1 KB
Local certificate	2 KB
CA certificate	2 KB
CA authority configuration	500 bytes
CRL (average size is a variable that depends on how many certificates have been revoked by that particular CA)	300 bytes up to 2 MB+

Example: Calculating flash memory requirements:

Assume the following settings in a Junos device:

- An average CRL of 10 KB
- One local certificate, one CA certificate, and CA authority configuration

The flash memory requirements for CRL =

2 KB (local certificate) + 1 KB (key pair) + 2 KB (CA certificate) + 0.5 (CA authority configuration) + 10 (CRL) = 15.5 KB

NOTE: Note the following points about certificate chains:

- For certificate chains, you must add additional CA certificates, additional CA profile configurations, and additional CRLs for each CA in the hierarchy or cross-certified chain.
- The high-end SRX Series (for example, SRX3400, SRX3600, SRX5600, SRX5800) or J-Series devices have more flash memory and can accommodate several local certificates and CA chains.
- The low-end SRX Series (for example, SRX100, SRX210, SRX 240) or J-Series devices have limited storage capacity and can include a limited number of certificates. It is recommended that the lower-end devices use only one local certificate/key pair, one CA (or one chain of CAs), and one CRL.

Related Documentation

- Certificate Life Cycle Management Overview on page 5
- Example: Configuring the PKI in Junos OS on page 13
- Verifying the PKI Configuration on page 43
- Appendix A: Frequently Asked Questions on page 49
- Appendix B: Administering Common Certificate Authorities on page 53
- Glossary of PKI Related Terms on page 72

Certificate Life Cycle Management Overview

The general life cycle for certificates includes the following phases:

1. Generation of public/private keys, identity information, and certificate request
2. Enrollment (request and retrieval)
3. Usage within Internet Key Exchange (IKE)
4. Certificate validation and revocation checks
5. Certificate renewal

Details on each phase of the certificate life cycle are given in the following topics:

- Generation of Public/Private Keys, Identity Information, and Certificate Request on page 6
- Certificate Enrollment on page 6
- Certificate Identity Usage Within IKE on page 6
- Certificate Validation and Revocation Checking on page 7
- Certificate Renewal on page 7

Generation of Public/Private Keys, Identity Information, and Certificate Request

The private key on a device must be generated on that device, and it should never be viewed or saved from that device. The user identity and private key form the basis of the certificate request and will continue to be used with the local certificate after enrollment. Therefore it is important to match the certificate-ID of the keys that are generated with the proper certificate request and eventually with the local certificate. The certificate request is available in PKCS10 format; the certificate request must be sent to the certificate authority (CA) either through an e-mail or some sort of web-based front-end site.

Certificate Enrollment

Junos OS Release 8.5 and earlier support only manual certificate request. This process includes generation of a PKCS10 request, submission to the certificate authority (CA), retrieval of the signed certificate, and manually loading of the certificate into the Junos device as the local certificate.

Automatic sending of certificate requests through Simple Certificate Enrollment Protocol (SCEP) is supported only in Junos OS Release 9.0 or later. For more information, see "Appendix D: Simple Certificate Enrollment Protocol" on page 70.

Certificate Identity Usage Within IKE

During the Internet Key Exchange (IKE) Phase 1 setup, a certificate can identify the peer by:

- IP address
- Fully qualified domain name (FQDN) (domain name)
- User fully qualified domain name (U-FQDN) (e-mail address)
- DN (distinguished name)

Most virtual private network (VPN) administrators use an IP address or FQDN for a VPN gateway identity type and use an e-mail address (U-FQDN) for a NetScreen-Remote (NSR) Client VPN laptop/user's identity type. The IKE IDs are added into the **SubjectAlternativeName** (a v3 extension) field of the certificate.

NOTE: **If the IP address of the VPN peer changes, then a new certificate must be issued with the new IP address, and the old certificate should be revoked.**

If the certificate authority (CA) does not support the signing of certificates with a **SubjectAlternativeName** field, then you must use the DN as the IKE ID on the Junos device or NSR configurations.

You must specify one of the following as the IKE ID in the certificate request:

- IP address
- Domain name

- E-mail address

Certificate Validation and Revocation Checking

Junos OS supports revocation checking by using the certificate revocation list (CRL) lookup method. The CRL can either be manually loaded in the Junos device or automatically retried online from the CRL Distribution Point (CDP), or it can be loaded on demand through Lightweight Directory Access Protocol (LDAP) or HTTP. Junos OS supports both DER and Privacy-Enhanced Mail (PEM) formats for CRL.

NOTE: **Junos OS does not support the Online Certificate Status Protocol (OCSP) method, which is a more scalable system than CRLs and CDP. Many other CAs also do not support an OCSP interface.**

Certificate Renewal

The certificate renewal process is similar to the initial certificate enrollment process. The renewal process includes replacing an old certificate (about to expire) on the virtual private network (VPN) device with a new certificate.

Currently the Junos OS supports only the manual renewal process. The Simple Certificate Enrollment Protocol (SCEP) can be used to re-enroll the local certificates automatically before they expire. For more information, see "Appendix D: Simple Certificate Enrollment Protocol" on page 70.

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Certificate Administration Overview on page 7
- Example: Configuring the PKI in Junos OS on page 13
- Appendix D: Simple Certificate Enrollment Protocol on page 70
- Glossary of PKI Related Terms on page 72

Certificate Administration Overview

This topic includes the following sections:

Overview on Usage of SSL and IPsec/IKE Methods

Currently, the Junos OS supports only the IP Security/Internet Key Exchange (IPsec/IKE) method for using certificates for public key validation and identity binding.

Table 2 on page 8 compares the authentication using the SSL method and IPsec/IKE method.

Table 2: Certificate Authentication Process (SSL and IPsec/IKE)

SSL Implementation	IPsec/IKE Implementation
Process is one-sided and simple.	Process is bidirectional and uses certificates.
The Security device acts as an SSL server, and the administrator's web browser (or for webauth, the user's web browser) acts as the SSL client.	Each VPN device sends its certificate to the other device.
The Security device sends a certificate to the web browser for identification, and the web browser starts a secured SSL session.	Each IPsec device validates the other device's certificate using the certificate authority (CA) certificate (or chain of CA certificates if a hierarchy of CAs is used).
The Security device authenticates the user or administrator through a login and password (secured with SSL encryption).	The Security device checks the other device's certificate through a certificate revocation list (CRL) to see whether it has been revoked.
The Security device does not require a user-based SSL certificate from the web browser and does not need CA and CRLs loaded on it.	The minimum PKI elements required to use certificate-based authentication are; a local certificate, a CA certificate, and the CA's CRL.

Process for Setting Up the PKI Elements

The minimum public key infrastructure (PKI) elements required for certificate-based authentication in Junos OS are a local certificate, certificate authority (CA) certificate, and the CA's certificate revocation list (CRL).

To manually load a local certificate, a CA certificate, and the CA's CRL on the device:

1. Configure the basic settings of the device as required by PKI:
   a. Set the accurate clock, date, time zone, and daylight savings.
   b. Set Domain Name System (DNS) to resolve hostnames that may be used in certificates and for CRL distribution points (CDPs).
   c. Set Network Time Protocol (NTP) to maintain accurate clocking because certificates have lifetimes based on a specific date and time.
2. Create a CA-profile for the certificate request and enrollment process.
3. Generate the certificate request (p10 file) and save it to a local file system or send it through a e-mail.
4. Submit the p10 file to the CA (through a Web server that acts as the front-end of the CA. Note that openSSL's CA supports only the command-line interface).
5. Retrieve the CA's own certificate (through the CA's Web server front-end interface).

6. Retrieve the Junos device's new local certificate after the CA has verified it (usually through the CA's Web server front-end interface).
7. Retrieve the CA's CRL (through a pre-specified URL to the CA's web server).
8. Load the CA certificate, local certificate, and CRL onto the Junos device.
9. Define the IKE policy and gateway to use the RSA-Signature authentication method (as opposed to pre-shared keys) and the local and CA certificates.

NOTE: The disadvantage of manual uploading of the certificates and CRLs are the extra time and effort needed to maintain an up-to-date CRL on all the devices.

Alternatively, you can define a CDP for the Junos device to retrieve the latest CRLs automatically. Most CAs have the CDP defined in the CA certificate itself, which the Junos device can use automatically. You can also define the CDP in the CA profile configuration.

For more information on certificates, see the Junos OS documentation at http://www.juniper.net/techpubs/software/junos/.

Choosing the IKE Identity to Use in the VPN and the Certificate

Junos OS supports the following four Internet Key Exchange (IKE) ID types that virtual private network (VPN) gateways can use to identify each other:

- IP address (example: 2.2.2.2)
- Fully qualified domain name (FQDN) (example: vpn1.juniper.net)
- User-fully qualified domain name (U-FQDN) or e-mail address (example: johndoe@juniper.net)
- Distinguished name (DN) (example: CN=John Doe, OU=sales, O=Juniper Networks, C=US)

NOTE: When configuring pre-shared keys, only the IP address, FQDN, and U-FQDN/e-mail address are used.

If you use IP address, FQDN, or U-FQDN/e-mail address as the IKE ID type, then it

- Must be defined in the **SubjectAlternativeName** field of the certificate
- Should match the peer configuration in the **security ike gateway <gateway-name>** hierarchy

You must define at least one IP address, FQDN, or U-FQDN/e-mail address when you generate a certificate request.

If your certificates do not have a **SubjectAlternativeName** field, then you should use DN for the IKE ID. To use DN, you have to define a distinguished name in dynamic IKE configuration by specifying a container (the DN must match exactly) or a wildcard (only a portion of the DN needs to match).

The following are the differences between the container method and the wildcard method:

- If the container keyword is used, then all the DN values specified must exactly match the values in the received certificate. In addition, the ordering of the values in both the remote certificate received and the gateway definition must match.
- If the wildcard keyword is used, then any one of the DN values specified must match a DN value in the received certificate. The ordering of the values in the remote certificate received and in the gateway definition can be different. The Junos device analyzes and compares only the defined DN fields; any unspecified DN fields are ignored.

NOTE: **If the certificate is using multiple fields of the same type (example: OU=sales, OU=engr, OU=central), then you must use the container method and ensure that the DN fields are defined exactly as they are received by the remote peer.**

Certificate Validation During the IKE Phase 1 Setup

During Internet Key Exchange (IKE), when two VPN peers are establishing a tunnel, each VPN device receives a certificate from the other IKE peer. On receiving the certificates, the device completes following tasks:

1. Pulls the IKE ID (defined in IKE Phase 1) from the peer and searches for the matching IKE definition on the gateway.
2. Validates the certificate by verifying the digital signature on the remote device's certificate using the public key in the certificate authority (CA) certificate.
3. Validates the current time mentioned in certificates in "Valid from" and "Valid to" fields.
4. Validates that the certificate contains the identity for the remote peer.
5. Performs the revocation check (unless revocation check is disabled) on the certificate to ensure that the certificate's serial number is not in the CA's certificate revocation list (CRL). This may cause the Junos device to automatically retrieve a new CRL from the CDP.

When all certificate checks are completed successfully, then IKE continues with the tunnel setup.

NOTE: Unless explicitly disabled in the configuration, CRLs are checked whenever the Junos device receives a certificate for a remote VPN gateway. If the CRL is not manually loaded, then the system may:

- Load the CRL from the CRL distribution point (CDP) defined in the certificate. Junos devices support HTTP and Lightweight Directory Access Protocol (LDAP) for CDPs.
- If the CDP is not available in the certificate, then the Junos device checks for a CDP setting in the CA profile.
- Use the global or default CDP (if already configured).

When verifying a certificate, a certificate chain is formed from the following:

- Remote server's local certificate
- Optional certificate chain from the IKE peer
- Locally stored CA certificates

Any CA certificate loaded from local storage during boot are considered as "trusted" certificates. The Junos OS supports certificate path validation upward through as many as seven levels of CA authorities in the hierarchy.

Junos OS currently supports only "partial" certificate path validation. Unlike "full" certificate path validation, "partial" validation does not require that the last certificate in the certificate chain be a root CA certificate (a self-signed CA). With "partial" certificate path validation, the last certificate can be a non-root CA certificate. However, the last certificate in the validation chain must be from local storage.

Unsupported PKI Protocols in Junos OS

Junos OS does not support the following public key infrastructure (PKI) protocols, which are primarily alternative forms of management protocols for communication between CAs and entities that use certificates:

- CMC—Certificate Management of Messages over CMS defined in RFC 2797.

  CMC is endorsed by Microsoft and VeriSign, but has limited support from other products.

- CMP—Certificate Management Protocol defined in RFC 2510.

  CMP is endorsed by Entrust, Baltimore, SSH, RSA, and OpenSSL. Even though this protocol works, it has limited deployment and interoperability with other interfaces from different vendors.

- XKMS—XML Key Management Specifications are defined in http://www.w3.org/TR/xkms/.

  XKMS is a relatively new protocol and currently not supported by Juniper Networks.

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Certificate Life Cycle Management Overview on page 5
- Example: Configuring the PKI in Junos OS on page 13
- Appendix A: Frequently Asked Questions on page 49
- Glossary of PKI Related Terms on page 72

CHAPTER 2

Configuring, Verifying, and Troubleshooting the PKI in Junos OS

This topic includes the following sections:

Example: Configuring the PKI in Junos OS

This example shows how to configure, verify, and troubleshoot the PKI. This topic includes the following sections:

Requirements

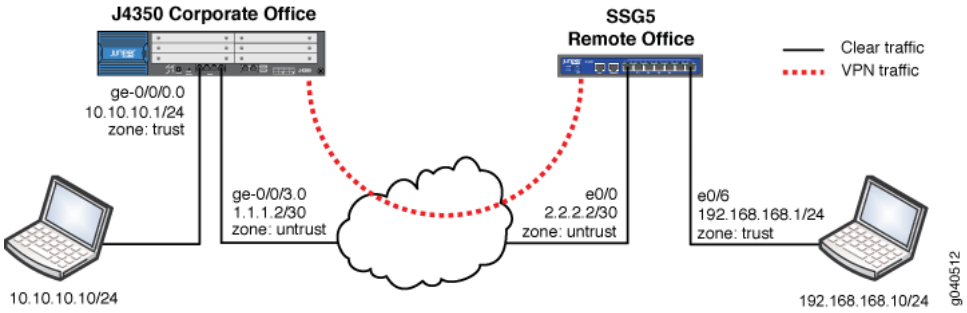
This example uses the following hardware and software components:

- Junos OS Release 9.4 or later
- Junos OS with Enhanced Services Release 8.5 through 9.3
- SRX Series devices or J Series devices

Network Topology

Figure 1 on page 14 shows the network topology used for this example to configure a policy-based VPN.

Figure 1: Network Topology Diagram



Required Settings

NOTE: The PKI administration is the same for both policy-based VPNs and route-based VPNs.

This example assumes the following settings:

- The remote VPN peer is a Juniper Networks SSG5 Firewall/VPN device (most commonly used for branch offices).
- The internal LAN interface of the device is **ge-0/0/0** in zone trust and has a private IP subnet.
- The Internet interface of the device is **ge-0/0/3** in zone untrust and has a public IP.
- All traffic between the local and remote LANs is permitted, and traffic can be initiated from either side.
- The SSG5 has already been pre-configured correctly and loaded with ready-to-use local certificate, CA-certificate, and CRL.
- The SSG5 device is configured to use FQDN as ssg5.juniper.net (IKE ID).
- PKI certificates with 1024–bit keys are used for the IKE negotiations on both sides.
- The CA is a standalone CA at the domain labdomain.com for the both VPN peers.

NOTE: For more information about the procedures for PKI administration and usage in Junos OS, see the *Junos OS Security Configuration Guide*.

NOTE: For more information on using a CA from PKI vendors such as Microsoft and OpenSSL, see "Appendix B: Administering Common Certificate Authorities" on page 53.

NOTE: More information about PKI configuration with SSG5 and other Juniper ScreenOS-based platforms. see the Juniper Networks PKI (Public Key Infrastructure) Primer & FAQ, Using X.509 Certificates in ScreenOS.

Configuration

This topic includes the following section:

Basic Configuration Steps for PKI

Step-by-Step Procedure

1. Configure an IP address and protocol family on the Gigabit Ethernet interfaces (**ge-0/0/0.0** and **ge-0/0/3.0**):

   ```
   [edit]
   user@host# set interfaces ge-0/0/0 unit 0 family inet address 10.10.10.1/24
   user@host# set interfaces ge-0/0/3 unit 0 family inet address 1.1.1.2/30
   ```

2. Configure a default route to the Internet nexthop:

   ```
   [edit]
   user@host# set routing-options static route 0.0.0.0/0 next-hop 1.1.1.1
   ```

   In this example, the VPN traffic is incoming on interface **ge-0/0/0.0** with the nexthop of **1.1.1.1**. Thus the traffic is outgoing on the interface **ge-0/0/3.0**. Any tunnel policy must consider incoming and outgoing interfaces.

   NOTE: Optionally you can use a dynamic routing protocol such as OSPF (not described in this document). When processing the first packet of a new session, the Junos device first performs a route lookup. The static route, which is also the default route, dictates the zone for the outgoing VPN traffic.

3. Set the system time and date.

   ```
   [edit]
   user@host# set system time-zone PST8PDT
   user@host# commit and-quit
   commit complete
   Exiting configuration mode
   ```

   ```
   user@host> set date ntp 130.126.24.24

   1 Nov 17:52:52 ntpdate[5204]: step time server 130.126.24.24 offset
   -0.220645 sec
   ```

When the configuration is committed, verify the clock settings:

```
user@host> show system uptime

Current time: 2007-11-01 17:57:09 PDT
System booted: 2007-11-01 14:36:38 PDT (03:20:31 ago)
```

```
Protocols started: 2007-11-01 14:37:30 PDT (03:19:39 ago)
Last configured: 2007-11-01 17:52:32 PDT (00:04:37 ago) by root
5:57PM up 3:21, 4 users, load averages: 0.00, 0.00, 0.00
```

4. Set the Domain Name System (DNS) configuration.

   Many CAs use hostnames (for example, FQDN) to specify various elements of the PKI. Since the CDP is usually specified using a URL containing an FQDN, you must configure a DNS resolver on the Junos device.

   ```
   [edit]
   user@host# set system name-server 4.2.2.1
   user@host# set system name-server 4.2.2.2
   user@host# commit and-quit
   commit complete
   Exiting configuration mode
   ```

5. Generate the certificate request by:

   - Create a CA profile to specify the CA settings.
   - Generate the PKCS10 certificate request.

     The PKCS10 certificate request process involves generating a public/private key pair and then generating the certificate request itself, using the key pair.

     NOTE: Note the following information about the CA profile:

     - The CA profile defines the attributes of a certificate authority.
     - Each CA profile is associated with a CA certificate. If a new or renewed CA certificate should be loaded without removing the older CA certificate, a new profile must be created. This profile can also be used for online fetching of the CRL.
     - There can be multiple such profiles present in the system created for different users.

   a. Create a trusted CA profile with the following mandatory values:

      - CA profile name — (ms-ca for this example) (any value)
      - CA identity— labdomain.com. (CA domain name)

        user@host# set security pki ca-profile ms-ca ca-identity labdomain.com

   b. Create a revocation check to specify a method for checking certificate revocation:

      user@host# set security pki ca-profile ms-ca revocation-check crl

      NOTE: You can use the option **disable** to disable the revocation check or select **crl** to configure the CRL attributes.

c. Set the refresh interval, in hours, to specify the frequency to update the CRL. The default values are: next-update time in CRL, or 1 week if no next-update time is specified.

```
user@host# set security pki ca-profile ms-ca revocation-check crl refresh-interval
  48
```

d. Specify the location (URL) to retrieve CRL (HTTP or LDAP). By default, the URL is empty and uses CDP information embedded in the CA certificate.

```
user@host# set security pki ca-profile ms-ca revocation-check crl url
  http://labsrv1.labdomain.com/CertEnroll/LABDOMAIN.crl
```

NOTE: The URL can include the server-name/port information such as, ldap://<ip-or-fqdn>:<port>). If the port number is missing, HTTP will use port 80, or LDAP will use port 443. Currently you can configure only one URL. Support for backup URL configuration is not available.

e. Specify an e-mail address to send the certificate request directly to a CA administrator.

```
user@host# set security pki ca-profile ms-ca administrator email-address
  certadmin@labdomain.com
```

NOTE: If you specify a CA administrator e-mail address to send the certificate request to, then the system composes an e-mail from the certificate request file and forwards it to the specified e-mail address. The e-mail status notification is sent to the administrator.

NOTE: The certificate request can be sent to the CA through an out-of-band method.

f. Commit the configuration:

```
user@host# commit and-quit
commit complete
Exiting configuration mode
```

g. Generate a key-pair request.

When the CA profile is configured, the next step is to generate a key pair on the Junos device. To generate the private and public key pair:

```
user@host> request security pki generate-key-pair certificate-id ms-cert size 1024

Generated key pair ms-cert, key size 1024 bits
```

The PKCS10 certificate request is generated and stored on the system as a pending certificate or certificate request. An e-mail notification will be sent to the administrator of the CA (in this example, certadmin@labdomain.com).

NOTE: Currently the Junos OS supports only the RSA algorithm and does not support the Digital Signature Algorithm (DSA). A unique identity called certificate-ID is used to name the generated key pair. This ID is also used in certificate enrollment and request commands to get the right key pair. The generated key pair is saved in the certificate store in a file with the same name as the certificate-ID. The file size can be 512, 1024, or 2048 bits.

NOTE:

A default (fallback) profile can be created if intermediate CAs are not preinstalled in the device. The default profile values are used in the absence of a specifically configured CA profile.

In the case of a CDP, the following order is followed:

- Per CA profile
- CDP embedded in CA certificate
- Default CA profile

We recommend using a specific CA profiles instead of a default profile.

6. Generate a local digital certificate request in the PKCS-10 format.

```
user@host> request security pki generate-certificate-request certificate-id ms-cert
subject "CN=john doe,CN=1.1.1.2,OU=sales,O=Juniper Networks,
L=Sunnyvale,ST=CA,C=US" email user@juniper.net filename ms-cert-req

Generated certificate request
-----BEGIN CERTIFICATE REQUEST-----
MIIB3DCCAUUCAQAwbDERMA8GA1UEAxMIam9obiBkb2UxDjAMBgNVBAsTBXNhbGVz
MRkwFwYDVQQKExBKdW5pcGVyIE5ldHdvcmtzMRIwEAYDVQQHEwlTdW5ueXZhbGUx
CzAJBgNVBAgTAkNBMQswCQYDVQQGEwJVUzCBnzANBgkqhkiG9w0BAQEFAAOBjQAw
gYkCgYEA5EG6sgG/CTFzX6KC/hz6Czal0BxakUxfGxF7UWYWHaWFFYLqo6vXNO8r
OS5Yak7rWANAsMob3E2X/1adlQIRi4QFTjkBqGI+MTEDGnqFsJBqrB6oyqGtdcSU
uOqUivMvgKQVCx8hpx99J3EBTurfWL1pCNlBmZggNogb6MbwES0CAwEAAaAwMC4G
CSqGSIb3DQEJDjEhMB8wHQYDVR0RBBYwFIESInVzZXJAanVuaXBlci5uZXQiMA0G
CSqGSIb3DQEBBQUAA4GBAI6GhBaCsXk6/1lE2e5AakFFDhY7oqzHhgd1yMjiSUMV
djmf9JbDz2gM2UKpI+yKgtUjyCK/lV2ui57hpZMvnhAW4AmgwkOJg6mpR5rsxdLr
4/HHSHuEGOF17RHO6x0YwJ+KE1rYDRWj3Dtz447ynaLxcDF7buwd4IrMcRJJI9ws
-----END CERTIFICATE REQUEST-----
Fingerprint:
47:b0:e1:4c:be:52:f7:90:c1:56:13:4e:35:52:d8:8a:50:06:e6:c8 (sha1)
a9:a1:cd:f3:0d:06:21:f5:31:b0:6b:a8:65:1b:a9:87 (md5)
```

NOTE: In the above sample of the PKCS10 certificate, the request starts with and includes the “BEGIN CERTIFICATE REQUEST” line and ends with and includes the “END CERTIFICATE REQUEST” line. This portion can be copied and pasted to your CA for enrollment. Optionally, you can also offload the “ms-cert-req” file and send that to your CA.

Generate the PKCS10 certificate request to be sent to the CA.

user@host> **request security pki generate-certificate-request certificate-id** ***id-name*** **subject subject-name (domain-name** ***domain-name*** **| ip-address** ***device-ip*** **| email** ***email-id*****) filename** ***filename***

The available options are:

- **certificate-id** — Name of the local digital certificate and the public/private key pair. This ensures that the proper key pair is used for the certificate request and ultimately for the local certificate.
- **subject** — Distinguished name format that contains the common name, department, company name, state, and country:
  - CN — Common name
  - OU — Department
  - O — Company name
  - L — Locality
  - ST — State
  - C — Country
  - CN — Phone
  - DC — Domain component

  NOTE: You are not required to enter all subject name components. Note also that you can enter multiple values of each type.

- **domain-name** — FQDN. The FQDN provides the identity of the certificate owner for IKE negotiations and provides an alternative to the subject name.
- **filename (path | terminal)** — (Optional) Location where the certificate request should be placed, or the login terminal.
- **ip-address** — (Optional) IP address of the device.
- **email** — (Optional) e-mail address of CA administrator.

NOTE: You must use one of these: domain-name, ip-address, or e-mail address.

This defines the IKE ID type and will need to be configured to the IKE gateway profile described later in this document.

The generated certificate request is stored in a specified file location. A local copy of the certificate request is saved in the local certificate storage. If the administrator reissues this command, the certificate request is generated once again.

The PKCS10 certificate request is stored in a specified file and location, from which you can download it and send it to the CA for enrollment. If you have not specified the file name or location, you can get PKCS10 certificate request details by using the **show security pki certificate-request certificate-id <id-name>** command in the CLI. You can copy the command output and paste it into a Web front-end for the CA server or into an e-mail.

7. Submit the certificate request to the CA, and retrieve the certificate.

   The administrator submits the certificate request to the CA. The CA administrator verifies the certificate request and generates a new certificate for the Junos device. The Junos device administrator retrieves it, along with the CA certificate and CRL.

   The process of retrieving the CA certificate, the device's new local certificate, and CRL from the CA depends on the CA configuration and software vendor in use.

   For more information on how retrieve the certificates, see "Appendix B: Administering Common Certificate Authorities" on page 53.

   NOTE:

   Junos OS supports the following CA vendors:

   - Entrust
   - Verisign
   - Microsoft

   Although other CA software services such as OpenSSL can be used to generate certificates, these certificates are not verified by Junos OS.

   Junos OS may extend support to other vendors that conform to X.509 certificate standards.

8. Load the local certificate, CA certificate, and CRL.

   The retrieved certificates (local certificate, CA certificate and CRL) should be loaded into the Junos device through FTP using the CLI.

   Assume the following names for certificates:

- local certificate — certnew.cer
- CA certificate — CA-certnew.cer
- CRL — certcrl.crl

```
user@host> file copy ftp://10.10.10.10/certnew.cer certnew.cer
  /var/tmp//...transferring.file.....crYdEC/100% of 1459 B 5864 kBps
user@host> file copy ftp:// 10.10.10.10/CA-certnew.cer CA-certnew.cer
  /var/tmp//...transferring.file.....UKXUWu/100% of 1049 B 3607 kBps
user@host> file copy ftp:// 10.10.10.10/certcrl.crl certcrl.crl
  /var/tmp//...transferring.file.....wpqnpA/100% of 401 B 1611 kBps
```

NOTE: You can verify that all files have been uploaded by using the command file list.

9. Load the certificate into local storage from the specified external file.

   You must also specify the certificate-ID in order to keep the proper linkage with the private/public key pair. This step loads the certificate into the RAM cache storage of the PKI module, checks the associated private key, and verifies the signing operation.

   ```
   user@host> request security pki local-certificate load certificate-id ms-cert filename
   certnew.cer

   Local certificate loaded successfully
   ```

10. Load the CA certificate from the specified external file.

    You must specify the CA profile to associate the CA certificate to the configured profile.

    ```
    user@host> request security pki ca-certificate load ca-profile ms-ca filename
    CA-certnew.cer

    Fingerprint:
    1b:02:cc:cb:0f:d3:14:39:51:aa:0f:ff:52:d3:38:94:b7:11:86:30 (sha1)
    90:60:53:c0:74:99:f5:da:53:d0:a0:f3:b0:23:ca:a3 (md5)
    Do you want to load this CA certificate ? [yes,no] (no) yes
    CA certificate for profile ms-ca loaded successfully
    ```

11. Load the CRL into the local storage.

    The allowed maximum size of the CRL is 5 MB. You must specify the associated CA profile in the command.

    ```
    user@host> request security pki crl load ca-profile ms-ca filename certcrl.crl

    CRL for CA profile ms-ca loaded successfully
    ```

12. Verify that all certificates are loaded.

    You can display the details of all local certificates in the CLI interface.

    ```
    user@host> show security pki local-certificate certificate-id ms-cert detail Certificate
    ```

```
identifier: ms-cert
Certificate version: 3
Serial number: 3a01c5a0000000000011
Issuer:
Organization: JuniperNetworks, Organizational unit: JTAC, Country: US,
State:
CA, Locality: Sunnyvale,
Common name: TACLAB
Subject:
Organization: Juniper Networks, Organizational unit: sales, Country: US,
State: CA, Locality: Sunnyvale,
Common name: john doe
Alternate subject: "user@juniper.net", fqdn empty, ip empty
Validity:
Not before: 11- 2-2007 22:54
Not after: 11- 2-2008 23:04
Public key algorithm: rsaEncryption(1024 bits)
30:81:89:02:81:81:00:e4:41:ba:b2:01:bf:09:31:73:5f:a2:82:fe
1c:fa:0b:36:a5:d0:1c:5a:91:4c:5f:1b:11:7b:51:66:16:1d:a5:85
15:82:ea:a3:ab:d7:34:ef:2b:39:2e:58:6a:4e:eb:58:03:40:b0:ca
1b:dc:4d:97:ff:56:9d:95:02:11:8b:84:05:4e:39:01:a8:62:3e:31
31:03:1a:7a:85:b0:90:6a:ac:1e:a8:ca:a1:ad:75:c4:94:bb:4a:94
8a:f3:2f:80:a4:15:0b:1f:21:a7:1f:7d:27:71:01:4e:ea:df:58:bd
69:08:d9:41:99:98:20:36:88:1b:e8:c6:f0:11:2d:02:03:01:00:01
Signature algorithm: sha1WithRSAEncryption
Distribution CRL:
ldap:///CN=TACLAB,CN=TACLABSRV1,CN=CDP,CN=Public%20Key%20Services,CN=Services,
CN=Configuration,DC=tacdomain,DC=com?certificateRevocationList?base?
objectclass=cRLDistributionPoint
http://taclabsrv1.tacdomain.com/CertEnroll/TACLAB.crl
Fingerprint:
c9:6d:3d:3e:c9:3f:57:3c:92:e0:c4:31:fc:1c:93:61:b4:b1:2d:58 (sha1)
50:5d:16:89:c9:d3:ab:5a:f2:04:8b:94:5d:5f:65:bd (md5)
```

NOTE: You can display the individual certificate details by specifying certificate-ID in the command line. Also, you can select the output format as either brief or detail.

13. View the CA certificates.

    Display all CA certificates or the CA certificates of an individual CA profile (specified).

    ```
    user@host> show security pki ca-certificate ca-profile ms-ca detail

    Certificate identifier: ms-ca
    Certificate version: 3
    Serial number: 44b033d1e5e158b44597d143bbfa8a13
    Issuer:
    Organization: JuniperNetworks, Organizational unit: JTAC, Country: US,
    State:
    CA, Locality: Sunnyvale,
    Common name: TACLAB
    Subject:
    Organization: JuniperNetworks, Organizational unit: JTAC, Country: US,
    State:
    CA, Locality: Sunnyvale,
    Common name: TACLAB
    ```

```
Validity:
Not before: 09-25-2007 20:32
Not after: 09-25-2012 20:41
Public key algorithm: rsaEncryption(1024 bits)
30:81:89:02:81:81:00:d1:9e:6f:f4:49:c8:13:74:c3:0b:49:a0:56
11:90:df:3c:af:56:29:58:94:40:74:2b:f8:3c:61:09:4e:1a:33:d0
8d:53:34:a4:ec:5b:e6:81:f5:a5:1d:69:cd:ea:32:1e:b3:f7:41:8e
7b:ab:9c:ee:19:9f:d2:46:42:b4:87:27:49:85:45:d9:72:f4:ae:72
27:b7:b3:be:f2:a7:4c:af:7a:8d:3e:f7:5b:35:cf:72:a5:e7:96:8e
30:e1:ba:03:4e:a2:1a:f2:1f:8c:ec:e0:14:77:4e:6a:e1:3b:d9:03
ad:de:db:55:6f:b8:6a:0e:36:81:e3:e9:3b:e5:c9:02:03:01:00:01
Signature algorithm: sha1WithRSAEncryption
Distribution CRL:
ldap:///CN=TACLAB,CN=TACLABSRV1,CN=CDP,CN=Public%20Key%20Services,CN=Services,
CN=Configuration,DC=tacdomain,DC=com?certificateRevocationList?base?
objectclass=cRLDistributionPoint
http://taclabsrv1.tacdomain.com/CertEnroll/TACLAB.crl
Use for key: CRL signing, Certificate signing, Non repudiation
Fingerprint:
1b:02:cc:cb:0f:d3:14:39:51:aa:0f:ff:52:d3:38:94:b7:11:86:30 (sha1)
90:60:53:c0:74:99:f5:da:53:d0:a0:f3:b0:23:ca:a3 (md5)
```

14. View the CRL.

    Display all loaded CRLs or the CRLs of the specified individual ca-profile.

    NOTE: You can select the output format as brief or detail to display CRL information; at present both options provide the same output.

    Syntax (operational mode):
    user@host> **show security pki crl ca-profile *ca-profile* [brief | detail]**

    Example:

    ```
    user@host> show security pki crl ca-profile ms-ca detail

    CA profile: ms-ca
    CRL version: V00000001
    CRL issuer: emailAddress = certadmin@juniper.net, C = US, ST = CA,
    L = Sunnyvale, O = JuniperNetworks, OU = JTAC, CN = TACLAB
    Effective date: 10-30-2007 20:32
    Next update: 11- 7-2007 08:52
    ```

15. Verify the certificate path for the local certificate and CA certificate.

    ```
    user@host> request security pki local-certificate verify certificate-id ms-cert

    Local certificate ms-cert verification success

    user@host> request security pki ca-certificate verify ca-profile ms-ca

    CA certificate ms-ca verified successfully
    ```

16. Use the certificates in an IPsec VPN.

NOTE: The steps for configuring a VPN using a certificate are similar to the steps for configuring a VPN using preshared keys. The only difference is the authentication method used for the IKE (Phase 1) policy. No changes are required for the IPsec (Phase 2) configuration because the use of certificates is part of Phase 1 negotiations.

The following configuration steps describe how to configure a policy-based VPN as this method is most commonly used for dial-up VPNs.

NOTE: For more information on VPN configuration, see the Junos Enhanced Services Configuration Guides.

NOTE: For more information on Junos Enhanced Services Application Notes, see KB10182 (http://kb.juniper.net/KB10182) available at Juniper Networks Knowledge Base.

To configure the IPsec VPN with the certificate:

NOTE: Refer to the network diagram shown in Figure 1 on page 14 to complete the following steps.

a. Configure the security zones, and bind the interfaces to the appropriate zones. Make sure that all necessary host-inbound services are enabled on the interfaces or zones. In this example, enable Internet IKE service either on the **ge-0/0/3** interface or on the untrust zone.

b. Configure address book entries for each zone for the tunnel policies.

c. Configure the IKE (Phase 1) proposals to use RSA encryption.

d. Configure an IKE policy specifying the RSA proposal (from step above 3), local certificate, CA certificate, and x.509 type peer certificate.

e. Configure the IKE gateway settings specifying the IKE policy (from step above 4), and a dynamic peer identified by the hostname.

   This step depends on how the certificate request was generated earlier. In this example, “CN=ssg5.juniper.net” was specified during the certificate request by the SSG5, which means that the IKE ID type is hostname.

f. Configure the IPsec (Phase 2) VPN settings.

Optionally you can also configure VPN monitor settings if required. In this example the Standard proposal set and PFS group 2 are used; however, you can create a different proposal if necessary.

g. Configure tunnel policies to permit remote office traffic into the host LAN and vice versa. Also configure an outgoing "trust" to "untrust" permit-all policy with source NAT for Internet traffic. Be sure that the tunnel policy is above the permit-all policy. Otherwise the policy lookup will never reach the tunnel policy.

h. Configure tcp-mss for IPsec traffic to eliminate the possibility of fragmented TCP traffic. This step reduces the resource usage on the device.

17. Configure security zones and assign interfaces to the zones.

The ingress (incoming) and egress (outgoing) zones are determined by the ingress and egress interfaces involved in the route lookup.

In this example packets are incoming on **ge-0/0/0**, and the ingress zone is the trust zone.

Following the route lookup, the egress interface is **ge-0/0/3**, and the egress zone is untrust zone. So the tunnel policy should be configured as "from-zone trust to-zone untrust" and vice versa.

```
[edit]
user@host# set security zones security-zone trust interfaces ge-0/0/0.0
user@host# set security zones security-zone untrust interfaces ge-0/0/3.0
```

18. Configure host-inbound services for each zone.

Host-inbound services are for traffic destined for the Junos device. These settings includes but are not limited to the FTP, HTTP, HTTPS, IKE, ping, rlogin, RSH, SNMP, SSH, Telnet, TFTP, and traceroute.

In this example we are assuming that all host-inbound services should be allowed from zone trust. For security reasons, we are allowing IKE only on the Internet-facing zone untrust which is required for IKE negotiations to occur. However, other services, such as services for management or troubleshooting can also be individually enabled if required.

```
[edit]
user@host# set security zones security-zone trust host-inbound-traffic
  system-services all
user@host# set security zones security-zone untrust host-inbound-traffic
  system-services ike
```

19. Configure the address book entries for each zone.

In this example we are using addressbook object names local-net and remote-net. There are some limitations with regard to which characters are supported for addressbook names. Please refer to the complete Junos OS documentation for more details.

```
[edit]
user@host# set security zones security-zone trust address-book address local-net
  10.10.10.0/24
```

```
user@host# set security zones security-zone untrust address-book address
  remote-net 192.168.168.0/24
```

20. Configure the IKE (Phase 1) proposal to use RSA encryption.

    This example uses 3DES encryption, the SHA1 authentication algorithm, and Diffie-Hellman Group 2 keys.

```
user@host# set security ike proposal rsa-prop1 authentication-method rsa-signatures
user@host# set security ike proposal rsa-prop1 encryption-algorithm 3des-cbc
user@host# set security ike proposal rsa-prop1 authentication-algorithm sha1
user@host# set security ike proposal rsa-prop1 dh-group group2
```

21. Configure an IKE policy.

    Main mode is typically used for site-to-site VPNs with static IP peers whereas aggressive mode is used for dynamic IP and dial-up peers.

    Tthis example uses main modebecause both sides have static IPs even though the hostname (typically used for dynamic tunnels) is used here for the IKE ID.

```
[edit]
user@host# set security ike policy ike-policy1 mode main
user@host# set security ike policy ike-policy1 proposals rsa-prop1
user@host# set security ike policy ike-policy1 certificate local-certificate ms-cert
user@host# set security ike policy ike-policy1 certificate peer-certificate-type x509-
  signature
user@host# set security ike policy ike-policy1 certificate trusted-ca use-all
```

22. Configure an IKE gateway.

    A remote IKE peer can be identified by IP address, FQDN/U-FQDN, or ASN1-DN (PKI certificates).

    In this example, we are identifying the peer by FQDN (hostname). Therefore the gateway IKE ID should be the remote peer's domain name. You must specify the correct external interface or peer ID to properly identify the IKE gateway during Phase 1 setup.

```
[edit]
user@host# set security ike gateway ike-gate external-interface ge-0/0/3.0
user@host# set security ike gateway ike-gate ike-policy ike-policy1
user@host# set security ike gateway ike-gate dynamic hostname ssg5.juniper.net
```

23. Configure the IPsec policy.

    In this example, we are using the Standard proposal set, which includes **esp-group2-3des-sha1** and **esp-group2- aes128-sha1** proposals. However, a unique proposal may be created and then specified in the IPsec policy if needed.

```
[edit]
user@host# set security ipsec policy vpn-policy1 proposal-set standard
[edit]
user@host# set security ipsec policy vpn-policy1 perfect-forward-secrecy keys
  group2
```

24. Configure the IPsec VPN with an IKE gateway and IPsec policy.

In this example, the VPN name ike-vpn must be referenced in the tunnel policy to create a security association. Additionally, if required, an idle time and a proxy ID can be specified if they are different from the tunnel policy addresses.

```
[edit]
user@host# set security ipsec vpn ike-vpn ike gateway ike-gate
user@host# set security ipsec vpn ike-vpn ike ipsec-policy vpn-policy1
```

25. Configure bidirectional tunnel policies for VPN traffic.

    In this example, traffic from the host LAN to the remote office LAN requires a "from-zone trust to-zone untrust" tunnel policy. However if a session needs to originate from the remote LAN to the host LAN, then a tunnel policy in the opposite direction "from-zone untrust to-zone trust" is also required. By specifying the policy in the opposite direction as the pair-policy, the VPN becomes bidirectional. Note also that in addition to action permit, we also need to specify the IPsec profile to be used. Furthermore source NAT can be enabled on the policy if desired, but that is beyond the scope of this application note. Note that for tunnel policies, the action is always permit. In fact if you are configuring a policy with action of deny, you will not see an option for specifying the tunnel.

```
[edit security policies from-zone trust to-zone untrust]
user@host# edit security policies from-zone trust to-zone untrust
user@host# set policy tunnel-policy-out match source-address local-net
user@host# set policy tunnel-policy-out match destination-address remote-net
user@host# set policy tunnel-policy-out match application any
user@host# set policy tunnel-policy-out then permit tunnel ipsec-vpn ike-vpn
  pair-policy tunnel-policy-in
user@host# top edit security policies from-zone untrust to-zone trust
user@host# set policy tunnel-policy-in match source-address remote-net
user@host# set policy tunnel-policy-in match destination-address local-net
user@host# set policy tunnel-policy-in match application any
user@host# set policy tunnel-policy-in then permit tunnel ipsec-vpn ike-vpn
  pair-policy tunnel-policy-out


[edit security policies from-zone untrust to-zone trust]
user@host# top
user@host# exit
```

26. Configure a security policy for Internet traffic.

    A security policy is required to permit all traffic from zone trust to zone untrust.

    The device uses the specified **source-nat interface**, and translates the source IP address and port for outgoing traffic, using the IP address of the egress interface as the source IP address and a random higher port for the source port. If required, more granular policies can be created to permit/deny certain traffic.

```
[edit]
user@host# edit security policies from-zone trust to-zone untrust
[edit security policies from-zone trust to-zone untrust]
user@host# set policy any-permit match source-address any
user@host# set policy any-permit match destination-address any
user@host# set policy any-permit match application any
user@host# set policy any-permit then permit source-nat interface
user@host# top
```

```
user@host# exit
```

27. Note that the security policy should be below the tunnel policy in the hierarchy because the policy list is read from top to bottom. If this policy were above the tunnel policy, then the traffic would always matche this policy and would not continue to the next policy. Thus no user traffic would be encrypted. To move the tunnel policy above the **any-permit** policy, use the **insert policy** command as shown below:

```
[edit]
user@host# edit security policies from-zone trust to-zone untrust
[edit security policies from-zone trust to-zone untrust]
user@host# insert policy tunnel-policy-out before policy any-permit
user@host# top
user@host# exit
```

28. Configure the **tcp-mss** (TCP maximum segment size) setting for TCP traffic across the tunnel.

    TCP-MSS is negotiated as part of the TCP 3-way handshake. It limits the maximum size of a TCP segment to accommodate the maximum transmission unit (MTU) limits on a network. This is very important for VPN traffic as the IPsec encapsulation overhead along with the IP and frame overhead can cause the resulting ESP packet to exceed the MTU of the physical interface, causing fragmentation. Fragmentation increases the bandwidth and device resources usage, and it should always be best avoided.

    The recommended value to use for tcp-mss is 1350 for most Ethernet-based networks with an MTU of 1500 or higher. This value may need to be altered if any device in the path has a lower value of MTU or if there is any added overhead such as PPP, Frame Relay, and so on. As a general rule, you may need to experiment with different **tcp-mss** values to obtain optimal performance.

    ```
    user@host# set security flow tcp-mss ipsec-vpn mss mss-value
    ```

    Example:

    ```
    [edit]
    user@host# set security flow tcp-mss ipsec-vpn mss 1350
    [edit]
    user@host# commit and-quit
    commit complete
    Exiting configuration mode
    ```

29. This step provides information on SSG device configuration. Because the focus of this example is on Junos OS configuration and troubleshooting, the SSG device configuration is explained briefly in this step.

    To show the configuration settings in Figure 1 on page 14, a sample of the relevant configurations is provided from an SSG5 device strictly for reference.

    However, the concepts with regard to configuration of policy-based VPNs for Juniper Networks Firewall/VPN products are available in the Concepts and Examples (C&E) guides. For more information, see the *Concepts & Examples ScreenOS Reference Guide* available at http://www.juniper.net/techpubs/software/screenos/.

    The following example is a relevant sample of an SSG5 configuration:

```
user@host# set interface ethernet0/6 ip 192.168.168.1/24
user@host# set interface ethernet0/6 route
user@host# set interface ethernet0/0 ip 2.2.2.2/30
user@host# set interface ethernet0/0 route
user@host# set flow tcp-mss 1350
user@host# set domain juniper.net
user@host# set hostname ssg5
user@host# set pki x509 default cert-path partial
user@host# set pki x509 dn country-name "US"
user@host# set pki x509 dn state-name "CA"
user@host# set pki x509 dn local-name "Sunnyvale"
user@host# set pki x509 dn org-name "Juniper Networks"
user@host# set pki x509 dn org-unit-name "Sales"
user@host# set pki x509 dn ip 2.2.2.2
user@host# set dns host dns1 4.2.2.1
user@host# set dns host dns2 4.2.2.2
user@host# set address "Trust" "local-net" 192.168.168.0 255.255.255.0
user@host# set address "Untrust" "corp-net" 10.10.10.0 255.255.255.0
user@host# set ike gateway "corp-ike" address 1.1.1.2 Main outgoing-interface
  ethernet0/0 proposal "rsa-g2- 3des-sha"
user@host# set vpn "corp-vpn" gateway "corp-ike" replay tunnel idletime 0 sec-level
  standard
user@host# set policy id 11 from "Trust" to "Untrust" "local-net" "corp-net" "ANY"
  tunnel vpn "corp-vpn" pairpolicy 10
user@host# set policy id 10 from "Untrust" to "Trust" "corp-net" "local-net" "ANY"
  tunnel vpn "corp-vpn" pairpolicy 11
user@host# set policy id 1 from "Trust" to "Untrust" "ANY" "ANY" "ANY" nat src
  permit
user@host# set ntp server "130.126.24.24"
user@host# set route 0.0.0.0/0 interface ethernet0/0 gateway 2.2.2.1
```

Verification

Verification and troubleshooting of the IKE and IPsec are similar to that site-to-site VPNs using preshared keys except for the use of the certificate for IKE identification, authentication, and encryption methods.

For more information, see the following:

- Juniper Networks' Knowledge Base at http://kb.juniper.net.

  Use Knowledge Base ID: KB10182KB10182 for list of application notes related to VPN configuration and troubleshooting information.

- Configuration Guides available at http://www.juniper.net/techpubs/.

Use the following steps to verify the IKE and IPsec configuration:

- Confirm IKE Phase 1 Status on page 30
- Get Details on Individual Security Associations on page 30
- Confirming IPsec Phase 2 Status on page 31
- Display IPsec Security Association Details on page 32
- Check IPsec SA Statistics on page 33

- Test Traffic Flow Across the VPN on page 34
- Confirm the Connectivity on page 34

Confirm IKE Phase 1 Status

Purpose To confirm the VPN status by checking any IKE Phase 1 security associations status.

PKI related to IPsec tunnels is formed during Phase 1 setup. Completion of Phase 1 indicates that PKI was successful.

Action

```
user@host> show security ike security-associations

Index Remote Address State Initiator cookie Responder cookie Mode
 202.2.2.2 UP af4f78bc135e4365 48a35f853ee95d21 Main
```

Meaning The output indicates that

- The remote peer is **2.2.2.2** and the status is **UP**, which means the successful association of Phase 1 establishment.
- The remote peer IKE ID, IKE policy, and external interfaces are all correct.
- **Index 20** is unique value for each IKE security association. You can use this output details to get further details on each security association. See "Get Details on Individual Security Associations" on page 30.

Incorrect output would indicate that:

- The remote peer status as **Down**.
- There are no IKE security associations .
- There are IKE policy parameters, such as the wrong mode type (Aggr or Main), PKI issues, or Phase 1 proposals (all must match on both peers). For more information, see "Troubleshooting IKE, PKI, and IPsec Issues" on page 35.
- External interface is invalid for receiving the IKE packets. Check the configurations for PKI-related issues or check kmd log for any other errors or run traceoptions to find the mismatch. For more information, see "Troubleshooting IKE, PKI, and IPsec Issues" on page 35.

Get Details on Individual Security Associations

Purpose Get details on individual IKE security associations (SAs).

Action

```
user@host> show security ike security-associations index 20 detail
IKE peer 2.2.2.2, Index 20,
Role: Responder, State: UP
Initiator cookie: af4f78bc135e4365, Responder cookie: 48a35f853ee95d21
Exchange type: Main, Authentication method: RSA-signatures
Local: 1.1.1.2:500, Remote: 2.2.2.2:500
Lifetime: Expires in 23282 seconds
Algorithms:
Authentication : sha1
```



```
Encryption : 3des-cbc
Pseudo random function: hmac-sha1
Traffic statistics:
Input bytes : 10249
Output bytes : 4249
Input packets: 10
Output packets: 9
Flags: Caller notification sent
IPsec security associations: 2 created, 1 deleted
Phase 2 negotiations in progress: 0
```

Meaning The output displays the details of the individual IKE SAs such as role (initiator or responder), status, exchange type, authentication method, encryption algorithms, traffic statistics, Phase 2 negotiation status, and so on.

You can use the output data to:

- Know the role of the IKE SA. Troubleshooting is easier when the peer has the responder role.
- Get the traffic statistics to verify the traffic flow in both directions.
- Get the number of IPsec security associations created or in progress.
- Get the status of any completed Phase 2 negotiations.

Confirming IPsec Phase 2 Status

Purpose View IPsec (Phase 2) security associations.

Phase 2 happens same was as it does with non-certificate-based VPNs.

When IKE Phase 1 is confirmed, view the IPsec (Phase 2) security associations.

Action

```
user@host> show security ipsec security-associations

total configured sa: 2
ID Gateway Port Algorithm SPI Life:sec/kb Mon vsys
<2 2.2.2.2 500 ESP:3des/sha1 bce1c6e0 1676/ unlim - 0
>2 2.2.2.2 500 ESP:3des/sha1 1a24eab9 1676/ unlim - 0
```

Meaning The output indicates that

- There is a configured IPsec SA pair available . The port number **500** indicates that a standard IKE port is used. Otherwise, it is Network Address Translation-Traversal (NAT-T), 4500, or random high port.
- The security parameter index (SPI) is used for both directions. The lifetime or usage limits of the SA is expressed either in seconds or in kilobytes. In the output, **1676/ unlim** indicates Phase 2 lifetime is set to expire in 1676 seconds and there is no specified lifetime size.
- The **ID** number shows the unique index value for each IPsec SA.

- A hyphen (-) in the **Mon** column indicates that VPN monitoring is not enabled for this SA.
- The virtual system (**vsys**) is zero, which is the default value.

NOTE: Phase 2 lifetime can be different from the Phase 1 lifetimebecause Phase 2 is not dependent on Phase 1 after the VPN is up.

NOTE: For information on VPN monitoring, refer to the complete documentation for Junos OS available at http://www.juniper.net/techpubs/.

Display IPsec Security Association Details

Purpose Display the individual IPsec SA details identified by the index number.

Action

```
user@host> show security ipsec security-associations index 2 detail
Virtual-system: Root
Local Gateway: 1.1.1.2, Remote Gateway: 2.2.2.2
Local Identity: ipv4_subnet(any:0,[0..7]=10.10.10.0/24)
Remote Identity: ipv4_subnet(any:0,[0..7]=192.168.168.0/24)
DF-bit: clear
Policy-name: tunnel-policy-out
Direction: inbound, SPI: bce1c6e0, AUX-SPI: 0
Hard lifetime: Expires in 1667 seconds
Lifesize Remaining: Unlimited
Soft lifetime: Expires in 1093 seconds
Mode: tunnel, Type: dynamic, State: installed, VPN Monitoring: -
Protocol: ESP, Authentication: hmac-sha1-96, Encryption: 3des-cbc
Anti-replay service: enabled, Replay window size: 32
Direction: outbound, SPI: 1a24eab9, AUX-SPI: 0
Hard lifetime: Expires in 1667 seconds
Lifesize Remaining: Unlimited
Soft lifetime: Expires in 1093 seconds
Mode: tunnel, Type: dynamic, State: installed, VPN Monitoring: -
Protocol: ESP, Authentication: hmac-sha1-96, Encryption: 3des-cbc
Anti-replay service: enabled, Replay window size: 32
```

Meaning The output displays the local Identity and the remote Identity.

Note that a proxy ID mismatch may cause Phase 2 completion to fail. The proxy ID is derived from the tunnel policy (for policy-based VPNs). The local address and remote address are derived from the address book entries, and the service is derived from the application configured for the policy.

If Phase 2 fails due to a proxy ID mismatch, verify which address book entries are configured in the policy and ensure that the correct addresses are sent. Also ensure that the ports are matching. Double-check the service to ensure that the ports match for the remote and local servers.

NOTE: If multiple objects are configured in a tunnel policy for source address, destination address, or application then the resulting proxy ID for that parameter is changed to zeroes.

For example, assume the following scenario for a tunnel policy:

- Local addresses of 10.10.10.0/24 and 10.10.20.0/24
- Remote address of 192.168.168.0/24
- Application as junos-http

The resulting proxy ID is local 0.0.0.0/0, remote 192.168.168.0/24, service 80.

The resulting proxy IDs can affect the interoperability if the remote peer is not configured for the second subnet. Also if you are employing a third-party vendor's application, you may have to manually enter the proxy ID to match.

If IPsec fails to complete, then check the kmd log or set traceoptions. For more information, see "Troubleshooting IKE, PKI, and IPsec Issues" on page 35.

Check IPsec SA Statistics

Purpose Check statistics and errors for an IPsec SA.

For troubleshooting purpose, check the Encapsulating Security Payload/Authentication Header (ESP/AH) counters for any errors with a particular IPsec SA.

Action

```
user@host> show security ipsec statistics index 2
ESP Statistics:
Encrypted bytes: 674784
Decrypted bytes: 309276
Encrypted packets: 7029
Decrypted packets: 7029
AH Statistics:
Input bytes: 0
Output bytes: 0
Input packets: 0
Output packets: 0
```

```
Errors:
AH authentication failures: 0, Replay errors: 0
ESP authentication failures: 0, ESP decryption failures: 0
Bad headers: 0, Bad trailers: 0
```

Meaning An error value of zero in the output indicates a normal condition.

We recommend running this command multiple times to observe any packet loss issues across a VPN. Output from this command also displays the statistics for encrypted and decrypted packet counters, error counters, and so on.

You must enable security flow traceoptions to investigate which ESP packets are experiencing errors and why. For more information, see "Troubleshooting IKE, PKI, and IPsec Issues" on page 35.

Test Traffic Flow Across the VPN

Purpose Test traffic flow across the VPN after Phase 1 and Phase 2 have completed successfully. You can test traffic flow by using the **ping** command. You can ping from local host to remote host. You can also initiate pings from the Junos device itself.

This example shows how to initiate a ping request from the Junos device to the remote host. Note that when pings are initiated from the Junos device, the source interface must be specified to ensure that the correct route lookup takes place and the appropriate zones are referenced in the policy lookup.

In this example, the **ge-0/0/0.0** interface resides in the same security zone as the local host and must be specified in the ping request so that the policy lookup can be from zone trust to zone untrust.

Action

```
user@host> ping 192.168.168.10 interface ge-0/0/0 count 5
PING 192.168.168.10 (192.168.168.10): 56 data bytes
64 bytes from 192.168.168.10: icmp_seq=0 ttl=127 time=8.287 ms
64 bytes from 192.168.168.10: icmp_seq=1 ttl=127 time=4.119 ms
64 bytes from 192.168.168.10: icmp_seq=2 ttl=127 time=5.399 ms
64 bytes from 192.168.168.10: icmp_seq=3 ttl=127 time=4.361 ms
64 bytes from 192.168.168.10: icmp_seq=4 ttl=127 time=5.137 ms
--- 192.168.168.10 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/stddev = 4.119/5.461/8.287/1.490 ms
```

Confirm the Connectivity

Purpose Confirm the connectivity between a remote host and a local host.

Action

```
ssg5-> ping 10.10.10.10 from ethernet0/6

Type escape sequence to abort
Sending 5, 100-byte ICMP Echos to 10.10.10.10, timeout is 1 seconds from
ethernet0/6
!!!!!
Success Rate is 100 percent (5/5), round-trip time min/avg/max=4/4/5 ms
```


Meaning You can confirm end-to-end connectivity by using the **ping** command from the remote host to the local host. In this example, the command is initiated from the SSG5 device.

Failed end-to-end connectivity may indicate an issue with routing, policy, end host, or encryption/decryption of the ESP packets. To verify the exact causes of the failure:

- Check IPsec statistics for details on errors as described in "Check IPsec SA Statistics" on page 33.
- Confirm end host connectivity by using the **ping** command from a host on the same subnet as the end host. If the end host is reachable by other hosts, then you can assume that the issue is not with the end host.
- Enable security flow traceoptions for troubleshooting the routing-and -policy-related issues.

  The details are not covered in this example, but you can get more details in VPN-related application notes for Junos OS available at http://kb.juniper.net/.

Troubleshooting IKE, PKI, and IPsec Issues

The basic troubleshooting steps are as follows:

1. Identifying and isolating the problem.
2. Debugging the problem.

The common approach of starting troubleshooting is with the lowest layer of the OSI layers and working your way up the OSI stack to confirm the layer in which the failure occurs. The steps for troubleshooting IKE, PKI, and IPsec are as follows:

- Confirm the physical connectivity of the Internet link at the physical and data link levels.
- Confirm that the Junos device has connectivity to the Internet next hop and connectivity to the remote IKE peer.
- Confirm IKE Phase 1 completion.
- Confirm IKE Phase 2 completion if IKE Phase 1 completion is successful.
- Confirm the traffic flow across the VPN (if the VPN is up and active).

Junos OS includes the traceoptions feature. Using this feature, you can enable a traceoption flag to write the data from the traceoption to a log file, which may be predetermined or manually configured and stored in flash memory. These trace logs can be retained even after a system reboot. Check the available flash storage before implementing traceoptions.

You can enable the traceoptions feature in configuration mode and commit the configuration to use the traceoptions feature. Similarly to disable traceoptions, you must deactivate traceoptions in configuration mode and commit the configuration.

Check the Free Disk Space on Your Device

Problem Check the statistics on the free disk space in your device file systems.

Solution

```
user@host> show system storage
Filesystem Size Used Avail Capacity Mounted on
/dev/ad0s1a 213M 74M 137M 35% /
devfs 1.0K 1.0K 0B 100% /dev
devfs 1.0K 1.0K 0B 100% /dev/
/dev/md0 180M 180M 0B 100% /junos
/cf 213M 74M 137M 35% /junos/cf
devfs 1.0K 1.0K 0B 100% /junos/dev/
procfs 4.0K 4.0K 0B 100% /proc
/dev/bo0s1e 24M 13K 24M 0% /config
/dev/md1 168M 7.6M 147M 5% /mfs
/cf/var/jail 213M 74M 137M 35% /jail/var
```

The **/dev/ad0s1a** represents the onboard flash memory and is currently at 35% capacity.

NOTE: You can view the available system storage in the J-Web interface under the **System Storage** option.

NOTE: You can enable traceoptions to log the trace data to the filenames specified or to the default log file to receive the output of the tracing operation.

The output of the traceoptions is placed in /var/log/kmd.

Check the Log Files to Verify Different Scenarios and Upload Log Files to an FTP

Problem View the log files to check security IKE debug messages, security flow debugs, and the state of logging to the syslog.

Solution

user@host> **show log kmd**
user@host> **show log pkid**
user@host> **show log security-trace**
user@host> **show log messages**

NOTE: You can view a list of all logs in the /var/log directory by using the **show log** command.

Log files can also be uploaded to an FTP server by using the **file copy** command.

(operational mode):
user@host> **file copy** ***path/filename dest-path/filename***
Example:

```
user@host> file copy /var/log/kmd ftp://10.10.10.10/kmd.log
ftp://10.10.10.10/kmd.log 100% of 35 kB 12 MBps
```

Enable IKE Traceoptions to View Messages on IKE

Problem To view success or failure messages for IKE or IPsec, you can view the key management process (kmd) log by using the **show log kmd** command. Because the kmd log displays some general messages, it may be useful to obtain additional details by enabling IKE and PKI traceoptions.

NOTE: Generally, it is best practice to troubleshoot the peer which has the responder role. You must obtain the trace output from the initiator and responder to understand the cause of a failure.

Configure IKE tracing options.

Solution

```
user@host> configure
Entering configuration mode

[edit]
user@host# edit security ike traceoptions
[edit security ike traceoptions]
```

```
user@host# set file ?
Possible completions:
<filename> Name of file in which to write trace information
files Maximum number of trace files (2..1000)
match Regular expression for lines to be logged
no-world-readable Don't allow any user to read the log file
size Maximum trace file size (10240..1073741824)
world-readable Allow any user to read the log file
```

```
[edit security ike traceoptions]
```

```
user@host# set flag ?
Possible completions:
all Trace everything
certificates Trace certificate events
database Trace security associations database events
general Trace general events
ike Trace IKE module processing
parse Trace configuration processing
policy-manager Trace policy manager processing
routing-socket Trace routing socket messages
timer Trace internal timer events
```

NOTE: If you do not specify file names for the <filename> field, then all IKE traceoptions are written to the kmd log.

You must specify at least one flag option to write trace data to the log. For example:

- **file size** — Maximum size of each trace file, in bytes. For example 1m or 1000000 can generate a maximum file size of 1 MB.
- **files** — Maximum number of trace files to be generated and stored in flash.

NOTE: You must commit your configuration to start the trace.

Enable PKI Traceoptions to View Messages on IPsec

Problem Enable PKI traceoptions to identify whether an IKE failure is related to the certificate or to a non-PKI issue.

Solution

```
user@host> configure
Entering configuration mode

[edit]
user@host# edit security pki traceoptions
[edit security pki traceoptions]
user@host# set file ?
Possible completions:
<filename> Name of file in which to write trace information
files Maximum number of trace files (2..1000)
match Regular expression for lines to be logged
no-world-readable Don't allow any user to read the log file
size Maximum trace file size (10240..1073741824)
world-readable Allow any user to read the log file

[edit security pki traceoptions]
user@host# set flag ?
Possible completions:
all Trace with all flags enabled
certificate-verification PKI certificate verification tracing
online-crl-check PKI online crl tracing
```

Setting up IKE and PKI Traceoptions to Troubleshoot IKE Setup Issues with Certificates

Problem Configure the recommended settings for IKE and PKI traceoptions.

NOTE: The IKE and PKI traceoptions use the same parameters, but the default filename for all PKI-related traces is found in the pkid log.

Solution

```
user@host> configure
Entering configuration mode

[edit]
user@host# edit security ike traceoptions
```

```
[edit security ike traceoptions]
user@host# set file size 1m
user@host# set flag ike
user@host# set flag policy-manager
user@host# set flag routing-socket
user@host# set flag certificates

[edit security ike traceoptions]
user@host# top edit security pki traceoptions
[edit security pki traceoptions]
user@host# set file size 1m
[edit security pki traceoptions]
user@host# set flag all
[edit security pki traceoptions]
user@host# commit and-quit
commit complete
Exiting configuration mode
```

Analyzing the Phase 1 Success Message

Problem Understand the output of the **show log kmd** command where the IKE Phase 1 and Phase 2 conditions are successful.

Solution

```
Nov 7 11:52:14 Phase-1 [responder] done for local=ipv4(udp:500,[0..3]=
1.1.1.2) remote=fqdn(udp:500,[0..15]=ssg5.juniper.net)
Nov 7 11:52:14 Phase-2 [responder] done for
p1_local=ipv4(udp:500,[0..3]=1.1.1.2)
p1_remote=fqdn(udp:500,[0..15]=ssg5.juniper.net)
p2_local=ipv4_subnet(any:0,[0..7]=10.10.10.0/24)
p2_remote=ipv4_subnet(any:0,[0..7]=192.168.168.0/24)
```

The sample output indicates:

- **1.1.1.2** — Local address.
- **ssg5.juniper.net** — Remote peer (hostname with FQDN).
- **udp: 500** — Indicates that no NAT-T was negotiated.
- **Phase 1 [responder] done** — Indicates the Phase 1 status, along with the role (initiator or responder).
- **Phase 2 [responder] done** — Indicates the Phase 1 status with proxy ID information.

  You can also confirm the IPsec SA status by using the verification commands mentioned in "Confirm IKE Phase 1 Status" on page 30.

Analyzing the Phase 1 Failure Message (Proposal Mismatch)

Problem Understand the output of the **show log kmd** command, where the IKE Phase 1 condition is a failure. This procedure helps in determining the reason for the VPN not establishing Phase 1.

Solution

```
Nov 7 11:52:14 Phase-1 [responder] failed with error(No proposal chosen) for
local=unknown(any:0,[0..0]=) remote=fqdn(udp:500,[0..15]=ssg5.juniper.net)
```

```
Nov 7 11:52:14 1.1.1.2:500 (Responder) <-> 2.2.2.2:500 { 011359c9 ddef501d -
2216ed2a bfc50f5f [-
1] / 0x00000000 } IP; Error = No proposal chosen (14)
```

The sample output indicates that

- **1.1.1.2** — Is the the local address
- **ssg5.juniper.net** — Indicates the remote peer (hostname with FQDN)
- **udp: 500** — Indicates that no NAT-T was negotiated.
- **Phase-1 [responder] failed with error (No proposal chosen)** — Indicates Phase 1 failure because of proposal mismatch.

To resolve this issue, ensure that the parameters for the IKE gateway Phase 1 proposals on both the responder and the initiator match. Also confirm that a tunnel policy exists for the VPN.

Analyzing the Phase 1 Failure Message (Authentication Failure)

Problem Understand the output of the **show log kmd** command when the IKE Phase 1 condition is a failure. This helps in determining the reason for the VPN not establishing Phase 1.

Solution

```
Nov 7 12:06:36 Unable to find phase-1 policy as remote peer:2.2.2.2 is not
recognized.
Nov 7 12:06:36 Phase-1 [responder] failed with error(Authentication failed) for
local=ipv4(udp:500,[0..3]=1.1.1.2) remote=ipv4(any:0,[0..3]=2.2.2.2)
Nov 7 12:06:36 1.1.1.2:500 (Responder) <-> 2.2.2.2:500 { f725ca38 dad47583 -
dab1ba4c ae26674b [-
1] / 0x00000000 } IP; Error = Authentication failed (24)
```

The sample output indicates that

- **1.1.1.2** — Is the local address
- **2.2.2.2** — Is the remote peer
- **Phase 1 [responder] failed with error (Authentication failed)** — Indicates Phase 1 failure due to the responder's not recognizing the incoming request originating from a valid gateway peer. In the case of IKE with PKI certificates, this failure typically indicates that an incorrect IKE ID type was specified or entered.

To resolve this issue, confirm that the correct peer IKE ID type is specified on the local peer based on the following:

- How the remote peer certificate was generated
- SubjectAlternativeName or DN information in the received remote peer certificate

Analyzing the Phase 1 Failure Message (Timeout Error)

Problem Understand the output of the **show log kmd** command when the IKE Phase 1 condition is a failure.

Solution

```
Nov 7 13:52:39 Phase-1 [responder] failed with error(Timeout) for
local=unknown(any:0,[0..0]=)
remote=ipv4(any:0,[0..3]=2.2.2.2)
```

The sample output indicates that:

- **1.1.1.2** — Is the local address
- **2.2.2.2** — Is the remote peer
- **Phase 1 [responder] failed with error(Timeout)** — Indicates Phase 1 failure.

  This error indicates that either the IKE packet is lost enroute to the remote peer or there is a delay or no response from the remote peer.

Because this timeout error is the result of waiting on a response from the PKI daemon, you must review the PKI traceoptions output to see whether there is a problem with PKI.

Analyzing the Phase 2 Failure Message

Problem Understand the output of the **show log kmd** command when the IKE Phase 2 condition is a failure.

Solution

```
Nov 7 11:52:14 Phase-1 [responder] done for local=ipv4(udp:500,[0..3]=
1.1.1.2) remote=fqdn(udp:500,[0..15]=ssg5.juniper.net)
Nov 7 11:52:14 Failed to match the peer proxy ids
p2_remote=ipv4_subnet(any:0,[0..7]=192.168.168.0/24)
p2_local=ipv4_subnet(any:0,[0..7]=10.10.20.0/24) for the remote
peer:ipv4(udp:500,[0..3]=2.2.2.2)
Nov 7 11:52:14 KMD_PM_P2_POLICY_LOOKUP_FAILURE: Policy lookup for Phase-2
[responder] failed for
p1_local=ipv4(udp:500,[0..3]=1.1.1.2) p1_remote=ipv4(udp:500,[0..3]=2.2.2.2)
p2_local=ipv4_subnet(any:0,[0..7]=10.10.20.0/24)
p2_remote=ipv4_subnet(any:0,[0..7]=192.168.168.0/24)
Nov 7 11:52:14 1.1.1.2:500 (Responder) <-> 2.2.2.2:500 { 41f638eb cc22bbfe -
43fd0e85 b4f619d5 [0]
/ 0xc77fafcf } QM; Error = No proposal chosen (14)
```

The sample output indicates that

- **1.1.1.2** — Is the local address.
- **ssg5.juniper.net** — Is the remote peer (IKE ID type hostname with FQDN).
- **Phase 1 [responder] done** — Indicates Phase 1 success.
- **Failed to match the peer proxy ids** — Indicates that the incorrect proxy IDs are received. In the previous sample, the two proxy IDs received are 192.168.168.0/24 (remote) and 10.10.20.0/24 (local) (for service=any). Based on the configuration given in this example, the expected local address is 10.10.10.0/24. This shows that there is a mismatch of configurations on the local peer, resulting in the failure of proxy ID match.

  To resolve this issue, correct the address book entry or configure the proxy ID on either peer so that it matches the other peer.

  The output also indicates the reason for failure is **No proposal chosen**. However in this case also we also see the message **Failed to match the peer proxy ids**.

Analyzing the Phase 2 Failure Message

Problem Understand the output of the **show log kmd** command when the IKE Phase 2 condition is a failure.

Solution

```
Nov 7 11:52:14 Phase-1 [responder] done for local=ipv4(udp:500,[0..3]=
1.1.1.2) remote=fqdn(udp:500,[0..15]=ssg5.juniper.net)
Nov 7 11:52:14 1.1.1.2:500 (Responder) <-> 2.2.2.2:500 { cd9dff36 4888d398 -
6b0d3933 f0bc8e26 [0]
/ 0x1747248b } QM; Error = No proposal chosen (14)
```

The sample output indicates that:

- **1.1.1.2** — Is the local address.
- **fqdn(udp:500,[0..15]=ssg5.juniper.net** — Is the remote peer.
- **Phase 1 [responder] done** — Indicates Phase 1 success.
- **Error = No proposal chosen** — Indicates that no proposal was chosen during Phase 2. This issue is due to proposal mismatch between the two peers.

  To resolve this issue, confirm that the Phase 2 proposals match on both peers.

Common Problems Related to IKE and PKI

Problem Troubleshoot common problems related to IKE and PKI.

Enabling the traceoptions feature helps you to gather more information on the debugging issues than is obtainable from the normal log entries. You can use the traceoptions log to understand the reasons for IKE or PKI failures.

For detailed analysis on the IKE and PKI problems and traceoptions outputs, contact Juniper Networks JTAC Support or visit the Juniper Networks Support Website at http://www.juniper.net/support for further assistance.

Solution Methods for troubleshooting the IKE -and-PKI-related issues:

- Ensure that the clock, date, time zone, and daylight savings settings are correct. Use NTP to keep the clock accurate.
- Ensure that you use a two-letter country code in the "C=" (country) field of the DN.

  For example: use “US” and not “USA” or “United States.” Some CAs require that the Country field of the DN be populated, allowing you to enter the country code value only with a two-letter value.
- Ensure that if a peer certificate is using multiple **OU=** or **CN=** fields, you are using the distinguished name with container method (the sequence must be maintained and is case sensitive).
- If the certificate is not valid yet, check the system clock and, if required, adjust the system time zone or just add a day in the clock for the quick test.

- Ensure that a matching IKE ID type and value are configured.
- PKI may fail due to a revocation check failure. To confirm this, temporarily disable revocation check and see whether IKE Phase 1 is able to complete.

  To disable revocation checking, use the following command in configure mode:

  set security pki ca-profile <ca-profile> revocation-check disable

  commit

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Verifying the PKI Configuration on page 43
- Certificate Life Cycle Management Overview on page 5
- Certificate Administration Overview on page 7
- Appendix A: Frequently Asked Questions on page 49
- Appendix B: Administering Common Certificate Authorities on page 53
- Appendix C: DoD PKI Usage on page 67
- Appendix D: Simple Certificate Enrollment Protocol on page 70
- Glossary of PKI Related Terms on page 72

Verifying the PKI Configuration

Purpose To verify the PKI configuration.

Action Use the **show configuration** command to verify PKI configuration.

```
user@host>show configuration
system {
    host-name host;
    time-zone PST8PDT;
    root-authentication {
        encrypted-password "$1$wUchK29B$IACQWVtsyF2PBlKtl1Air."; ## SECRET-DATA
    }
    name-server {
        4.2.2.1;
        4.2.2.2;
    }
    services {
        ssh;
        telnet;
        web-management {
            http {
                interface ge-0/0/0.0;
            }
        }
    }
    syslog {
        user * {
```

```
        any emergency;
      }
      file messages {
        any any;
        authorization info;
      }
      file interactive-commands {
        interactive-commands any;
      }
    }
  }
    interfaces {
      ge-0/0/0 {
        unit 0 {
          family inet {
            address 10.10.10.1/24;
          }
        }
      }
      ge-0/0/3 {
        unit 0 {
          family inet {
            address 1.1.1.2/30;
          }
        }
      }
    }
    routing-options {
      static {
        route 0.0.0.0/0 next-hop 1.1.1.1;
      }
    }
    security {
      ike {
        traceoptions {
          flag ike;
          flag policy-manager;
          flag routing-socket;
          flag certificates;
        }
        proposal rsa-prop1 {
          authentication-method rsa-signatures;
          dh-group group2;
          authentication-algorithm sha1;
          encryption-algorithm 3des-cbc;
        }
        policy ike-policy1 {
          mode main;
          proposals rsa-prop1;
          certificate {
            local-certificate ms-cert;
            trusted-ca use-all;
            peer-certificate-type x509-signature;
          }
        }
        gateway ike-gate {
```

```
        ike-policy ike-policy1;
        dynamic hostname ssg5.juniper.net;
        external-interface ge-0/0/3;
    }
}
ipsec {
    policy vpn-policy1 {
        perfect-forward-secrecy {
            keys group2;
        }
        proposal-set standard;
    }
    vpn ike-vpn {
        ike {
            gateway ike-gate;
            ipsec-policy vpn-policy1;
        }
    }
}
zones {
    security-zone untrust {
        address-book {
            address remote-net 192.168.168.0/24;
        }
        host-inbound-traffic {
            system-services {
                ike;
            }
        }
        interfaces {
            ge-0/0/3.0;
        }
    }
    security-zone trust {
        address-book {
            address local-net 10.10.10.0/24;
        }
        host-inbound-traffic {
            system-services {
                all;
            }
        }
        interfaces {
            ge-0/0/0.0;
        }
    }
}
policies {
    from-zone trust to-zone untrust {
        policy tunnel-policy-out {
            match {
                source-address local-net;
                destination-address remote-net;
                application any;
            }
            then {
```

```
            permit {
              tunnel {
                ipsec-vpn ike-vpn;
                pair-policy tunnel-policy-in;
              }
            }
          }
        }
        policy any-permit {
          match {
            source-address any;
            destination-address any;
            application any;
          }
          then {
            permit {
              source-nat {
                interface;
              }
            }
          }
        }
      }
      from-zone untrust to-zone trust {
        policy tunnel-policy-in {
          match {
            source-address remote-net;
            destination-address local-net;
            application any;
          }
          then {
            permit {
              tunnel {
                ipsec-vpn ike-vpn;
                pair-policy tunnel-policy-out;
              }
            }
          }
        }
      }
    }
    flow {
      tcp-mss {
        ipsec-vpn {
          mss 1350;
        }
      }
    }
    pki {
      ca-profile ms-ca {
        ca-identity labdomain.com;
        revocation-check {
          crl {
            url http://labsrv1.labdomain.com/CertEnroll/LABDOMAIN.crl;
          }
        }
```

```
        }
        traceoptions {
            file size 1m;
            flag all;
        }
    }
}
```

NOTE: In the above output sample of **show configuration** command, highlighted lines are for traceoption configurations for troubleshooting purposes.

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Example: Configuring the PKI in Junos OS on page 13

CHAPTER 3

Appendixes

This topic includes the following sections:

Appendix A: Frequently Asked Questions

- **Does Juniper Networks provide a CA with its products?**

  No. If you want to use a public key infrastructure (PKI), you must obtain third party certificate authority (CA) software to implement the PKI or use a service such as Verisign.

- **What version of X.509 certificates are supported (V1 or V3)?**

  Juniper Networks support both versions of X.509 certificates. However, you must use V3 if you want to use the **SubjectAlternativeName** extension field for a non-DN (distinguished name) Internet Key Exchange (IKE) ID type (for example, IP address, e-mail address, or fully qualified domain name [FQDN]).

- **Does the Junos device support multiple certificates?**

  Yes, the Junos device can generate multiple key pairs, and multiple certificate requests, and have multiple local certificates loaded. The specific quantity of certificates depends on the particular platform.

- **Can the Junos device use the same DN for different local certificates?**

  The Junos device does not support multiple certificates with the same subject (or DN) name on a single Junos device. Therefore, we recommend using a separate subject name for every key pair to avoid confusion. Some CAs also have limitations on supporting multiple key pairs for the same subject name.

- **Can the Junos device auto-generate common name (CN) field values such as FQDN and serial number in the DN?**

The Junos device does not auto-generate CN values such as FQDN and serial number. The FQDN or any other CN values must be specified during the certificate request procedure.

- **Does the Junos device support a hierarchical certificate authority (CA) chain?**

  Yes, the Junos device can validate certificates up through a chain of CA certificates.

- **How many levels of a certificate authority (CA) chain can the Junos device validate?**

  Seven.

- **I have many levels in my certificate authority (CA) chain, and my CRL Distribution Point (CDP) servers are slow; How do I keep IKE from timing out?**

  Try adjusting the refresh interval for the certificate revocation list (CRL) so that the CRL is not checked very frequently. This is at the expense of potentially allowing a certificate which may have been revoked by the CA.

- **Does Juniper Networks support PKCS10 for certificate requests?**

  Yes, PKCS10 certificate requests can be generated by the Junos device. These certificate requests can be copied using the command-line interface (CLI), sent through e-mail, or uploaded to an FTP server.

- **Does Juniper Networks support PKCS12 certificate packages?**

  No, the Junos device does not accept a PKCS12 file. The Junos device must generate its own private key. Also, a Junos device does not generate a PKCS12 file for exporting its private/public keys and certificate. This approach provides more protection and reduces the possibility that someone could steal a device's keys and thereby impersonate that device.

- **Does the private key ever leave the Junos device?**

  No, but in future Junos OS Releases, the private key may be copied from the active to the backup unit of a device if that device is part of chassis clustering or a Junos Services Redundancy Protocol (JSRP) pair as an RTO (run-time object).

- **What special characters should I avoid?**

  We support printable strings, minus reserved characters. We use as delimiters such as the comma. Names with an underscore (_) can also potentially cause problems.

- **What RFC does Juniper support for public key infrastructure (PKI)?**

  We follow RFC3280. We also have all the required security features of RFC2459 (the predecessor of RFC3280).

- **What are the PKI objects stored in flash and run-time memory?**

  Certificate authority (CA) certificate, CA certificate revocation list (CRL), CA profile configuration, local key pair, and local certificate or pending certificate.

- **How are these PKI objects related?**

  Each CA certificate typically uses three objects (CA certificate, CRL, and CA profile configuration). Each local certificate uses two objects (certificate and key pair). A pending certificate is a PKCS10 file that has been generated and sent to a CA. When

the signed certificate from the CA is installed the pending certificate object is replaced with the local certificate.

- **What are average sizes for PKI objects?**

  Average sizes of items:

  - CRLs vary, depending on how many certificates a particular CA has revoked: minimum of 300 bytes to a maximum of 5MB.
  - Certificates average 2K bytes each.
  - Key pairs average 1K bytes each.
  - CA profile configurations average 500 bytes each.

- **What is the maximum size of a CRL?**

  The maximum size supported in Junos OS Release 8.5 is 5 MB.

- **How do you disable CRL checking?**

  CRL checking is configurable per CA profile.

  The command syntax for disabling CRL checking is – **set security pki caprofile *ca-profile* revocation-check disable** followed by **commit**.

- **Why does the Junos device not use or support two sets of keys for a virtual private network (VPN)?**

  In general, while setting up a PKI for e-mail and file encryption and signing, you should use two sets of keys. While you certainly want two sets of keys when encrypting e-mails and files (one set for signing and one set for encryption) you do not need two sets for the VPN. RSA keys are used only for authentication in IPsec, and so you do not need the second set of keys for things like long-term storage of encrypted material.

- **Does Juniper Networks support CA Cross-certification? In other words, if one Junos device uses a certificate from one root CA, and another Junos device uses a certificate from a different root CA, are cross-certified. Can these two certificates validate each other's certificates and form the VPN tunnel properly?**

  Yes, it can be done by using the PKCS7 certificate type. Using cross-certification, we can form a full certificate path to the root certificate stored locally.

- **Which certificate formats does Junos OS supports?**

  Junos OS follows the PKI profile described in RFC 3280 and supports:

  - Installation of end-entity (EE) or CA certificate
  - Encode, including the X509 or PKCS7, DER or PEM
  - Compatibility with X.509 v3 and handling of extensions defined in RFC3280.

- **Does Junos OS support chassis clustering (high availability) for PKI certificates?**

  Junos OS Release 8.5 does not currently support high availability (HA) or JSRP with PKI. Future releases may support the transferring of a device key pair and local

certificates between two HA peers. Check release notes for upcoming releases to see whether this is supported in releases later than 8.5.

- **How is the public key of a key pair bound to a certificate request?**

  When generating a new key pair, a certificate-ID must be specified. This certificate-ID is also used for the certificate request and again when the local certificate is loaded. To completely delete a certificate request and key pair, use the **clear security pki** operational mode command. Two clear operations are needed: one to clear the certificate request and another to clear the key pair.

- **Why not delete both the certificate and the key pair at the same time?**

  Some administrators prefer the ability to keep the same key pair and use a new certificate with them. This allows deletion of the old certificate without destroying the old key pair.

- **Does Junos OS support Digital Signature Algorithm (DSA) keys?**

  No, currently only RSA keys are supported. DSA keys may be supported in future releases.

- **Is Junos ICSA certified?**

  Not yet, although many of the security features in Junos OS were sourced from Juniper Networks ScreenOS products which are certified for version 1.2.

  For more information regarding ICSA certification, see the ICSA Labs website at http://www.icsalabs.com/.

- **Is OCSP supported for revocation checking?**

  Not currently, but it may be supported in a future release.

- **Are there special characters to consider when doing PKI?**

  Yes, the comma (,) is a special character in ASN.1 DN and requires an escape character, to use which is the backslash (\).

  The UTF-8 encoded string should not have any of the following characters:

  - A space or pound (#) character occurring at the beginning of the string;
  - A space character occurring at the end of the string;
  - One of the characters comma (,), plus (+), double quote(""), backslash (\), less than or left triangle bracket (<), greater than or right triangle bracket (>), or semi-colon (;).

    If the comma (,) character needs to be escaped, then it should be prefixed by a backslash (\) (ASCII 92).

- **I want my CRL Distribution Point (CDP) function to communicate through a VPN tunnel. How do I set that up so the Junos device will source the IP from an internal interface that matches a tunnel definition and not source the packet from the egress (outgoing) interface which does not match a tunnel policy (even though that interface is the tunnel endpoint/gateway IP itself)?**

This is currently not supported in Junos OS.

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Example: Configuring the PKI in Junos OS on page 13
- Verifying the PKI Configuration on page 43
- Glossary of PKI Related Terms on page 72

Appendix B: Administering Common Certificate Authorities

This topic provides some basic concepts and examples of administrative procedures using a Microsoft certificate authority (CA) and an open source CA from OpenSSL. This information may help you to work with the administrator to enroll and use certificates on the Junos device.

The choice of CA depends on whether you want a standalone CA solution or will rely on a third party such as Verisign; this topic assumes that you want a standalone server for which you will be the CA administrator.

This topic includes the following sections:

Certificate Authorities Overview

Junos Networks supports the following vendors of CA:

- Verisign
- Entrust
- Microsoft Win2000 Advanced Server

NOTE: **Although Juniper Networks does not support the Open source code from OpenSSL officially, you can use it with Junos OS if set up properly.**

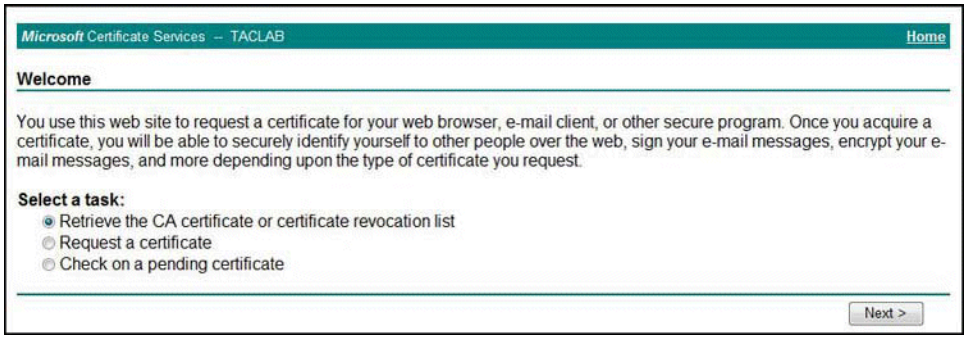
Microsoft Windows 2000 Certificate Authority

The Microsoft CA, provided on Windows 2000 advanced server, provides CA services through a web interface, including support for a CRL Distribution Point (CDP). Microsoft also supports a patch to activate Simple Certificate Enrollment Protocol (SCEP). Microsoft does not support OCSP.

1. In your Web browser, type the Web address http://<host.domain>/certsrv. where <host.domain> is the IP address of Microsoft CA server.

   The initial welcome page is displayed, as shown in Figure 2 on page 54.

Figure 2: Downloading Microsoft CA Certificate—Initial Welcome Page

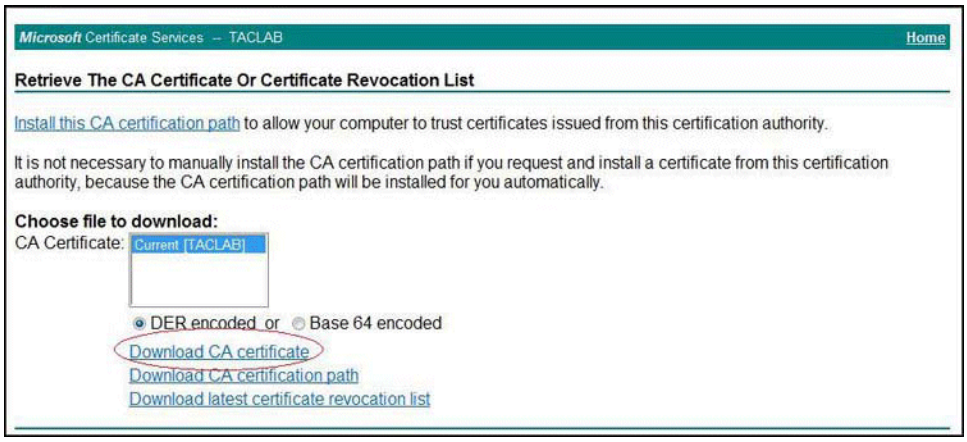


2. Select **Retrieve the CA certificate or certificate revocation list** and click **Next**.

   The Retrieve the CA Certificate or Certificate Revocation List page is displayed.

3. Select the CA you want to use, and then click **Download CA certificate**, as shown in Figure 3 on page 54.

Figure 3: Retrieve the CA Certificate or Certificate Revocation List



   A pop-up window appears that allows you to choose the location in which to save the certificate. Choose a location on your local file system and save the certificate with a .cer extension (for example, certnew.cer).

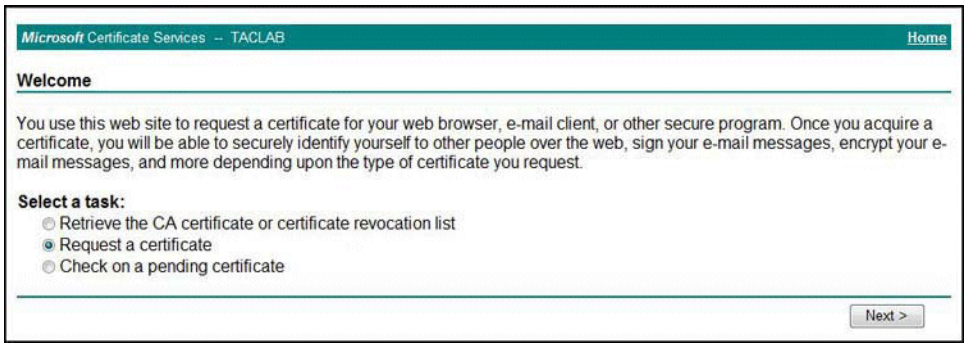
4. Select the **Download latest certificate revocation list** option on the Retrieve The CA Certificate or Certificate Revocation List page, as shown in Figure 4 on page 55.

Figure 4: Download Latest Certificate Revocation List

A pop-up window appears that allows you to choose the location in which to save the CRL. Choose a location on your local file system, and save the CRL with a .crl extension (for example, certcrl.crl).

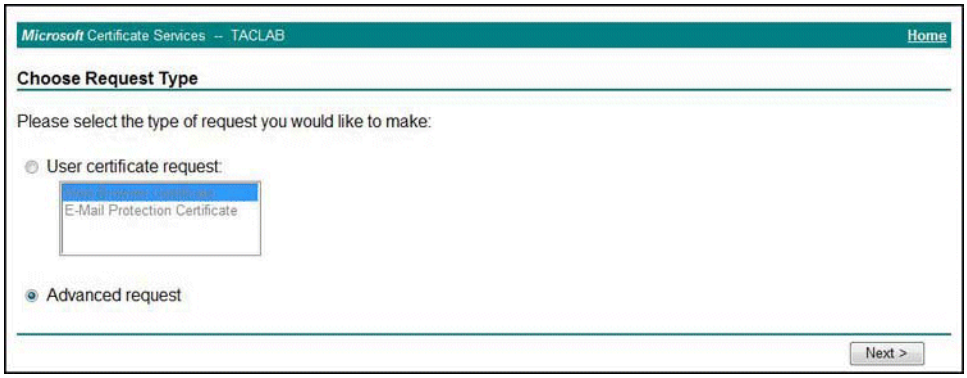
5. Return to the initial Welcome screen to request a certificate. Select **Request a Certificate** and click **Next**, as shown in Figure 5 on page 55.

Figure 5: Request Certificate



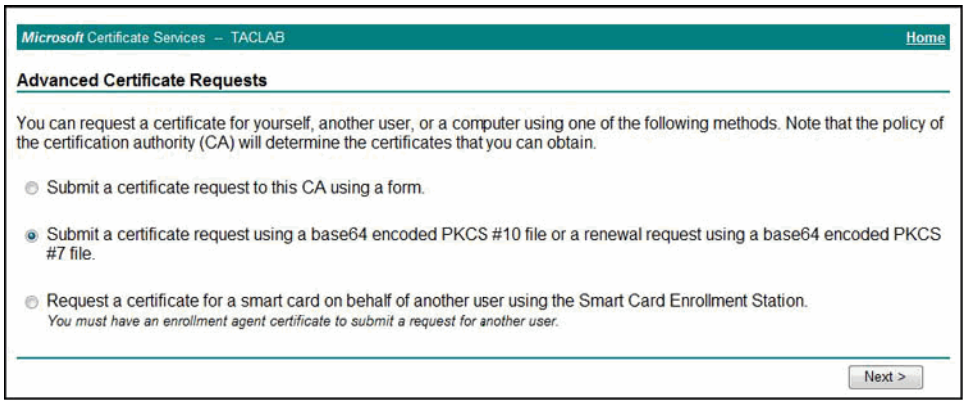
6. On the Choose Request Type page, select the **Advanced request** option, as shown in Figure 6 on page 55.

Figure 6: Select Advanced Request



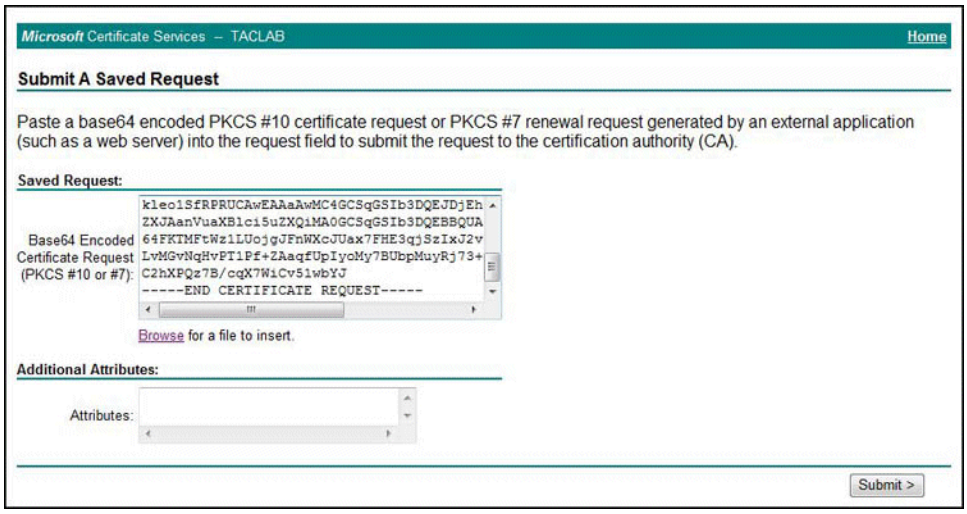
7. On the Advanced Certificate Requests page, select the **PKCS #10** option, as shown in Figure 7 on page 56.

Figure 7: Advanced Certificate Requests



8. On the Submit a Saved Request page, you can paste a copy of the certificate request into the page, as shown in Figure 8 on page 56, and click **Submit**.

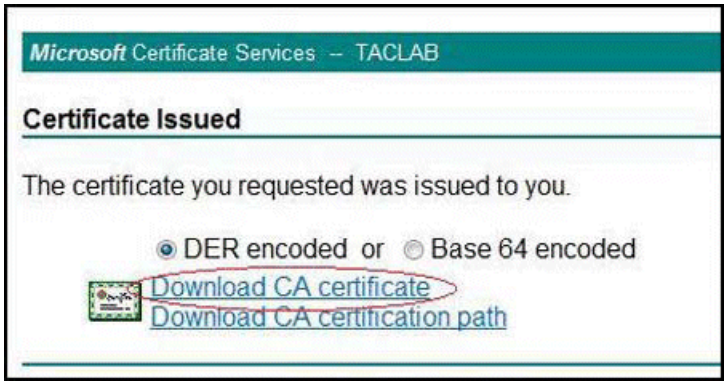
Figure 8: Submit a Saved Request



If the CA setup is configured to issue certificates automatically, then the Certificate Issued window is displayed as shown in Figure 9 on page 57.

9. Click **Download CA certificate** to download your new local certificate, as shown in Figure 9 on page 57.

Figure 9: Download New Local Certificate



NOTE: In Figure 9 on page 57, the downloaded certificate is your local Junos device certificate; not a CA certificate, as appears on the screen.

If the CA is not configured to issue certificates automatically, then you or a CA administrator must manually authorize the certificate request and generate the certificate.

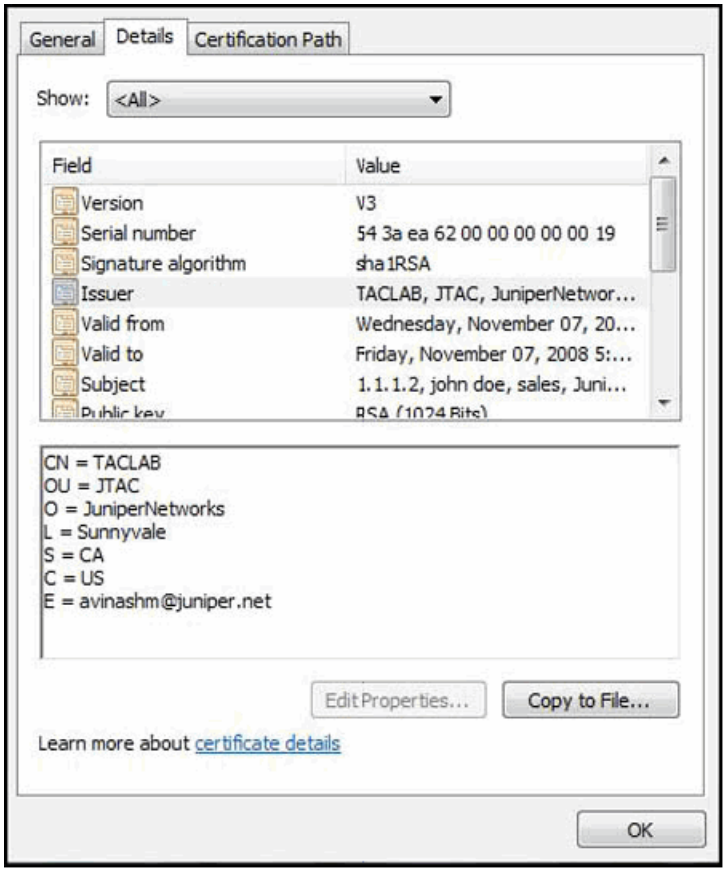
To retrieve a certificate that an administrator has issued, you can return to the Microsoft CA home page (http://servername/certsrv).

10. On the Microsoft CA home page, click **Check on a Pending Certificate**.

    If the certificate has been issued, the Certificate Issued Web page appears. From here you can view the certificates.

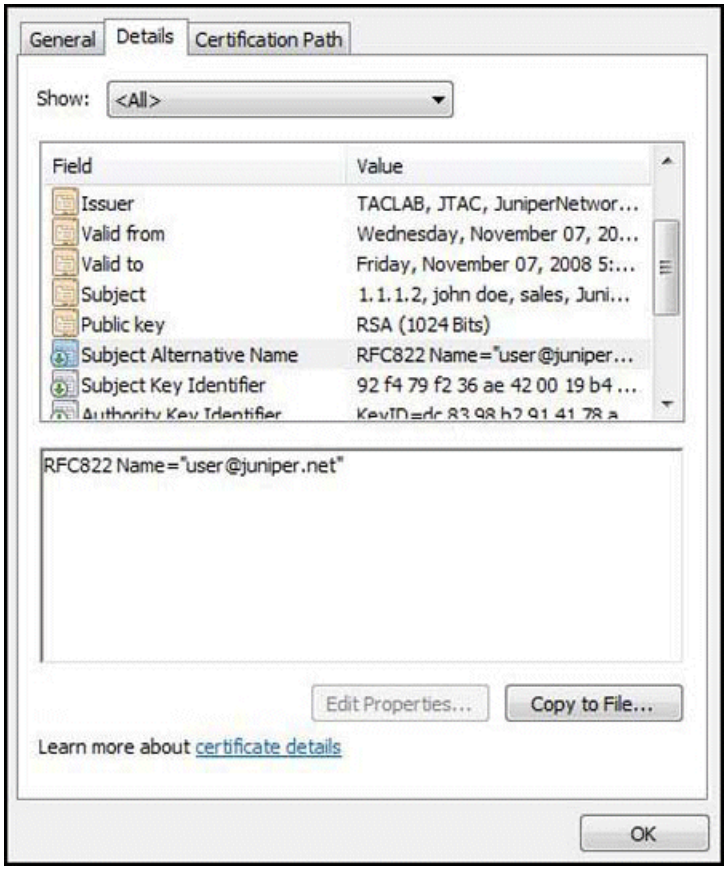
11. Double-click the **.cer** file, and then click the **Details** tab to see all the certificate fields and their values, as shown in Figure 10 on page 58.

Figure 10: View Certificate Details



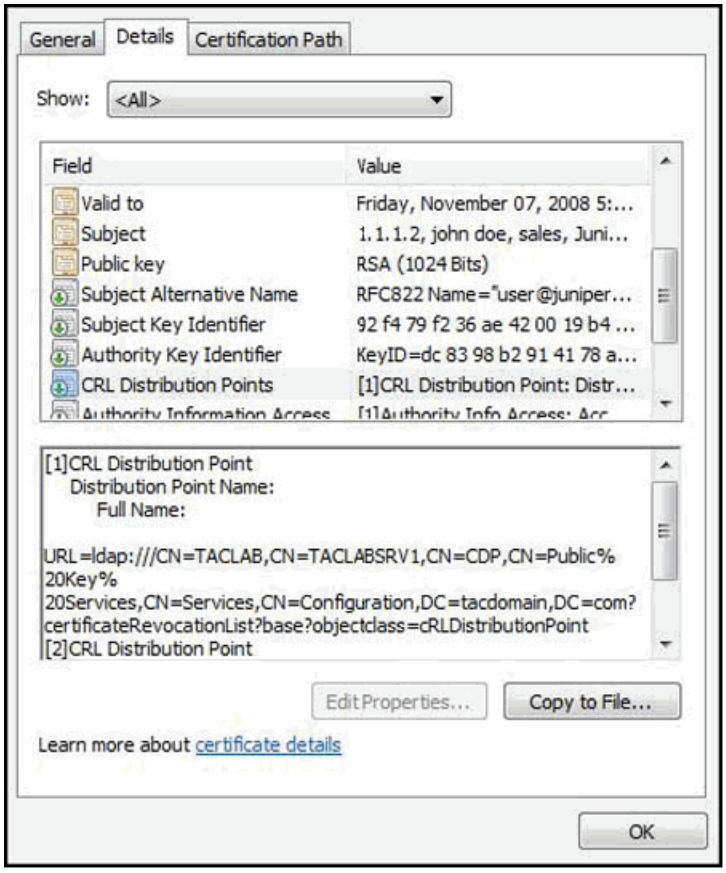
12. On the Details tab, validate the **SubjectAlternativeName** field for a certificate. Ensure that the **SubjectAlternativeName** includes the IKE ID types and values that are used in the Junos device IKE gateway definition, as shown in Figure 11 on page 59.

Figure 11: Validate SubjectAlternativeName Values



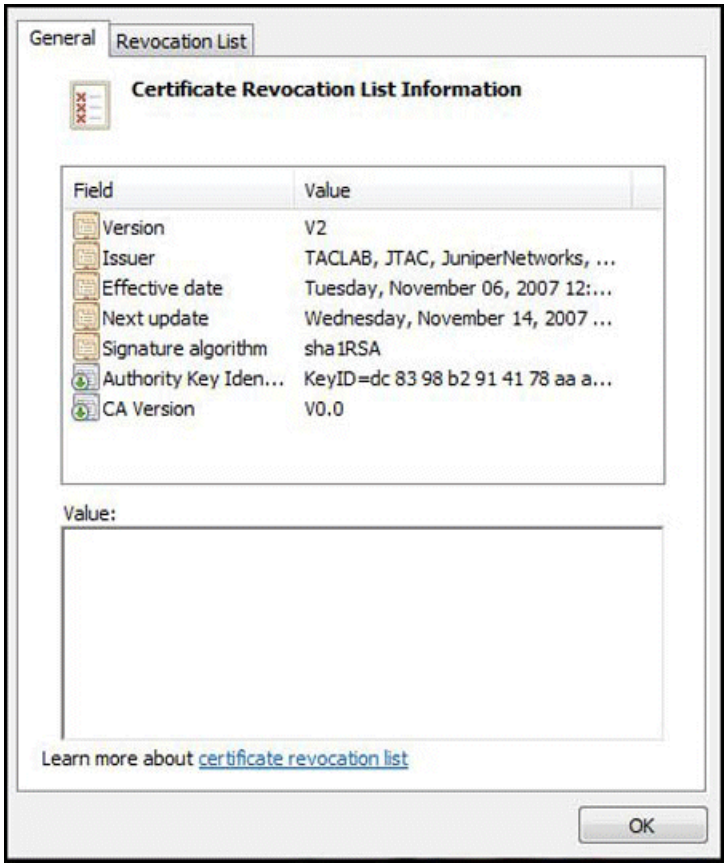
13. On the Details tab, you can also check the availability of a CDP. Figure 12 on page 60 shows the CDP field and value. Ensure that the hostname can be identified and reached by the Junos device.

Figure 12: View CDP Field and Value



14. View certificate revocation list (CRL) information on the Certificate Revocation List Information screen, as shown in Figure 13 on page 61.

Figure 13: View Certificate Revocation List Information



OpenSSL CA Overview

The OpenSSL code available at the http://www.openssl.org/ Web site, provides free and simple command-line functionality to use in certificate authorization applications. GUI or Web interface support is not available in this functionality. All input (forexample, p10 certificate requests) and all output (signed certificates and certificate revocation lists [CRLs]) are usually available in Privacy-Enhanced Mail (PEM)-encoded format.

To use OpenSSL, download and install the openssl.exe executable and perform the one-time CA setup. Here is an example using a Windows system:

1. Create a working directory, and use the **cd** command to make sure that you are in your home directory.
2. Copy the **openssl.exe** and **openssl.cfg** files to the home directory.

   To view a sample copy of the openssl.cfg file, see "OpenSSL.cfg File Sample" on page 63.
3. From the working directory, create some additional subdirectories as shown below:

   mkdir demoCA

   mkdir demoCA\private


```
Create the CA's own key pair and CA certificate.
```

openssl req -x509 -newkey rsa:1024 -keyout demoCA\private\key.pem \out demoCA\ca-cert.pem -config openssl.cfg

```
Warning: the private key for the CA will not be encrypted here.
```

4. Download the **ca-cert.pem** file into the Junos device as the certificate authority (CA) certificate.
5. Set up a "database" for the certificates that will be generated by this CA.

mkdir demoCA\certstore

echo 01 > demoCA\ca-cert.srl

6. Create a new but empty file called **index.txt** in the demoCA directory.

edit demoCA\index.txt

7. Save and exit the application.

   The CA is now initialized.

This procedure provides information about setting the basic configuration items and certificate request for each Junos device that needs a certificate:

1. When a PKCS10 file is generated, save that certificate request into a file called **jsNAME.pkcs10**.
2. Go to the OpenSSL CA's working directory (the parent directory of the **demoCA** subdirectory created earlier) to sign the certificate request (PKCS10 file) generated by the Junos device.
3. Although the **SubjectAlternativeName** field information is in the Junos device's PKCS10 certificate request, the OpenSSL CA cannot sign it as it is. The OpenSSL server may attempt to strip that part out of the certificate request. To have the certificate populated with a **SubjectAlternativeName** field, you must edit a setting in the **openssl.cfg** file itself. However, that file must be modified for every certificate you sign. Use the step below to edit the openssl.cfg file:

   edit openssl.cfg

4. Search for the **SubjectAltName** field. Reset the **SubjectAltName** field to the correct value for the particular Junos device certificate you are about to sign. For example:

   ```
   subjectAltName=DNS:ssg5.juniper.net
   ```

5. To create and sign the certificate, issue the command below, assuming
   - The certificate request from the Junos device is available in — **jsNAME.pkcs10**
   - The generated certificate will be stored in **jsNAME.cer**

   openssl ca -config openssl.cfg -in jsNAME.pkcs10 -out jsNAME.cer

   The Junos device's local certificate is now generated as the **jsNAME.cer** file and can be loaded into the Junos device. A copy of this certificate is also created in the

demoCA\certstore subdirectory with a name of NN.pem where NN is the serial number of this certificate.

NOTE: This certificate is in PEM format. To view the certificate with the Microsoft certificate viewer, the certificate needs to be converted to the DER encoding format by editing the jsNAME.cer file. Delete everything except the ----BEGIN/END certificate--- lines and all the data between those lines. This allows Microsoft Windows to decode the file properly to display its contents. The OpenSSL CLI can also convert the PEM-encoded certificate to DER encoding. See the OpenSSL documentation for details.

To revoke a certificate and generate a new CRL:

1. Find the serial number of the certificate. For example, to revoke a certificate with serial number 01, use the command below:

   openssl ca -config openssl.cfg -revoke demoCA\certstore\01.pem

2. If you encounter an error, then you can manually move the file by using the following command:

   mv demoCA\index.txt.new demoCA\index.txt

3. Next, generate the new CRL as shown below:

   openssl ca -config openssl.cfg -gencrl -out crl.crl

   The **crl.crl** file can now be loaded onto the Junos device. Load the CA certificate, CRL, and local certificate following the same steps as described in this document.

OpenSSL.cfg File Sample

A sample of an OpenSSL.cfg file is shown below:

```
#
# OpenSSL example configuration file.
# This is mostly being used for generation of certificate requests.
#
# This definition stops the following lines choking if HOME isn't
# defined.
HOME = .
RANDFILE = $ENV::HOME\\.rnd
# Extra OBJECT IDENTIFIER info:
#oid_file = $ENV::HOME\\.oid
oid_section = new_oids
# To use this configuration file with the "-extfile" option of the
# "openssl x509" utility, name here the section containing the
# X.509v3 extensions to use:
# extensions =
# (Alternatively, use a configuration file that has only
# X.509v3 extensions in its main [= default] section.)
[ new_oids ]
# We can add new OIDs in here for use by 'ca' and 'req'.
# Add a simple OID like this:
# testoid1=1.2.3.4
# Or use config file substitution like this:
```

```
# testoid2=${testoid1}.5.6
####################################################################
[ ca ]
default_ca = CA_default # The default ca section
####################################################################
[ CA_default ]
dir = .\\demoCA # Where everything is kept
certs = $dir\\certs # Where the issued certs are kept
crl_dir = $dir\\crl # Where the issued crl are kept
database = $dir\\index.txt # database index file.
new_certs_dir = $dir\\certstore # default place for new certs.
certificate = $dir\\ca-cert.pem # The CA certificate
serial = $dir\\ca-cert.srl # The current serial number
crl = $dir\\crl.pem # The current CRL
private_key = $dir\\private\\key.pem # The private key
RANDFILE = $dir\\private\\.rand # private random number file
x509_extensions = usr_cert # The extentions to add to the cert
# Extensions to add to a CRL. Note: Netscape communicator chokes
# on V2 CRLs so this is commented out by default to leave a V1 CRL.
# crl_extensions = crl_ext
default_days = 365 # how long to certify for
default_crl_days= 30 # how long before next CRL
default_md = md5 # which md to use.
preserve = no # keep passed DN ordering
# A few difference way of specifying how similar the request should
# look. For type CA, the listed attributes must be the same, and
# the optional and supplied fields are just that :-)
policy = policy_match
# For the CA policy
[ policy_match ]
countryName = optional
stateOrProvinceName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional
# For the 'anything' policy
# At this point in time, you must list all acceptable 'object'
# types.
[ policy_anything ]
countryName = optional
stateOrProvinceName = optional
localityName = optional
organizationName = optional
organizationalUnitName = optional
commonName = supplied
emailAddress = optional
####################################################################
[ req ]
default_bits = 1024
default_keyfile = privkey.pem
distinguished_name = req_distinguished_name
attributes = req_attributes
x509_extensions = v3_ca # extensions to add to the self signed cert
# Passwords for private keys if not present they will be prompted
# for input_password = secret
# output_password = secret
# This sets a mask for permitted string types of which there are
# several options.
#
# default: PrintableString, T61String, BMPString.
```

```
# pkix : PrintableString, BMPString.
# utf8only: only UTF8Strings.
# nombstr : PrintableString, T61String (no BMPStrings or
# UTF8Strings).
# MASK:XXXX a literal mask value.
# WARNING: current versions of Netscape crash on BMPStrings or
UTF8Strings
# so use this option with caution!
string_mask = nombstr
# req_extensions = v3_req # extensions to add to a cert request
[ req_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = US
countryName_min = 2
countryName_max = 2
stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = Illinois
localityName = City or local name
localityName_default = Chicago
0.organizationName = demo-company.com
0.organizationName_default = netscreen.com
# we can do this but it is not needed normally :-)
#1.organizationName = Second Organization Name (eg, company)
#1.organizationName_default = Sales
organizationalUnitName = Org Unit
organizationalUnitName_default = CSE
commonName = Common Name
commonName_default = test-CA
commonName_max = 64
emailAddress = Email Address
emailAddress_default = admin@juniper.net
emailAddress_max = 40
# SET-ex3 = SET extension number 3
[ req_attributes ]
challengePassword = secretkey
challengePassword_min = 4
challengePassword_max = 20
unstructuredName = juniper.net
[ usr_cert ]
# These extensions are added when 'ca' signs a request.
# This goes against PKIX guidelines. Some CAs do this and some
# software requires this to avoid interpreting an end user
# certificate as a CA.
basicConstraints=CA:FALSE
# Here are some examples of the usage of nsCertType. If it is
# omitted the certificate can be used for anything *except* object
# signing. This is OK for an SSL server.
#
# nsCertType = server
# For an object signing certificate this would be used.
# nsCertType = objsign
# For normal client use this is typical
#
# nsCertType = client, email
#
# and for everything including object signing:
# nsCertType = client, email, objsign
#
# This is typical in keyUsage for a client certificate.
# keyUsage = nonRepudiation, digitalSignature, keyEncipherment
# This will be displayed in Netscape's comment listbox.
```

```
nsComment = "OpenSSL Generated Certificate"
# PKIX recommendations harmless if included in all certificates.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
# This stuff is for subjectAltName and issuerAltname.
# Import the email address.
# subjectAltName=email:copy
subjectAltName=DNS:ssg5.juniper.net
# Copy subject details
# issuerAltName=issuer:copy
#nsCaRevocationUrl = http://www.domain.dom/ca-crl.pem
#nsBaseUrl
#nsRevocationUrl
#nsRenewalUrl
#nsCaPolicyUrl
#nsSslServerName
[ v3_req ]
# Extensions to add to a certificate request
basicConstraints = CA:FALSE
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
[ v3_ca ]
# Extensions for a typical CA
# PKIX recommendation.
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid:always,issuer:always
# This is what PKIX recommends but some broken software chokes on
# critical extensions.
#basicConstraints = critical,CA:true
# So we do this instead.
basicConstraints = CA:true
# Key usage: this is typical for a CA cert. However since it will
# prevent it being used as a test self-signed cert it is best
# left out by default.
# keyUsage = cRLSign, keyCertSign
# Some might want this also
# nsCertType = sslCA, emailCA
# Include email address in subject alt name: a PKIX recommendation
# subjectAltName=email:copy
# Copy issuer details
# issuerAltName=issuer:copy
# DER hex encoding of an extension: beware experts only!
# obj=DER:02:03
# Where 'obj' is a standard or added object
# You can even override a supported extension:
# basicConstraints= critical, DER:30:03:01:01:FF
[ crl_ext ]
# CRL extensions.
# Only issuerAltName and authorityKeyIdentifier make any sense in a
# CRL.
# issuerAltName=issuer:copy
authorityKeyIdentifier=keyid:always,issuer:always
```

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Example: Configuring the PKI in Junos OS on page 13
- Verifying the PKI Configuration on page 43
- Glossary of PKI Related Terms on page 72

Appendix C: DoD PKI Usage

The USA Federal Government Department of Defense (DoD) maintains a public key infrastructure (PKI) used by many entities, including the military.

DoD PKI Introduction

DoD PKI uses a custom PKI solution based on the Netscape iPlanet certificate authority (CA) server. Although Junos does not officially support Netscape iPlanet CA, it provides support to an extent required for DoD PKI. This custom PKI solution includes its own certificate profiles and security policies which may differ from other CAs.

You must analyze the DoD PKI profile to understand the requirements for operating with DoD PKI. Here is the summary of the features of DoD PKI:

- The DoD PKI is composed of a 3-layer certificate hierarchy and includes
  - Root CA — JITC DoD PKI Class 3 Root CA
  - Subordinate CA — JITC DoD PKI Class 3 ID CA. The subordinate CA issues all end-entity certificates.

NOTE: A server certificate (which the Junos device acquires) does not have a **SubjectAlternativeName** extension field (example — Certificate acquired through an e-mail).

A server certificate acquired by NS Remote client, will normally have a **SubjectAlternativeName** field containing an Internet Key Exchange (IKE) ID type of e-mail.

- The DN of all DoD PKI certificates can have multiple OU fields.

  A server certificate issued by DoD PKI has the DN form that includes the following fields:

  - CN=server <DNS name or IP>
  - OU=<military/government organization>
  - OU=PKI
  - OU=DoD
  - O=US. Government
  - C=US

A user certificate has the DN form that includes the following fields:

- Last Name
- First Name
- Middle Initial or Name (optional)
- Generation (Jr., Sr., II, III, and so forth) (optional)
- E-mail address
- Organization (military/government or contractor)
- City
- State
- Country

- The CA certificates can be downloaded from http://dodpki.c3pki.chamb.disa.mil/rootca.html. A certificate can be downloaded as two separate CA certificates (the root CA and the subordinate CA), or as a single PKCS7 envelope containing both.
- The root CRL is updated approximately every 30 days, where as the ID CRL is automatically updated every 24 hours. The CRL distribution point is in the CA certificate with attributes of **certificaterevocationlist;binary**. There is no scope or filter defined in the LDAP URL.

Junos OS needs to support multiple fields in order to interoperate with DoD PKI, It must support multiple OU fields to comply with the DN convention of the DoD PKI. Because Junos OS already supports multiple OU entries, this can be specified when generating a PKCS10 certificate request by adding multiple OU objects in the subject as shown in the example below:

```
request security pki generate-certificate-request certificate-id test hostname
user@idca.nit.disa.mil subject "CN=idca.nit.disa.mil,OU=DISA,OU=PKI,OU=DoD,
O=U.S. Government,C=US"
```

This requirement is not limited to just the OU or O fields of the DN, but also applies to all fields including S, L, and Country.

NOTE:

Note the following points about DoD PKI:

- DoD PKI supports a CRL Lightweight Directory Access Protocol (LDAP) search with default attributes and filters.
- The LDAP URL of the DoD PKI does not provide filters or a scope.
- DoD PKI supports certificate chaining and multilayer CRL verification.
- DoD PKI is a 2-layer CA hierarchy that is composed of a root CA and subordinate CA.
- DoD PKI supports DN as a peer gateway IKE ID type. Junos OS supports distinguished name as the IKE ID of a static or dynamic peer gateway.
- DoD PKI allows disabling of CRL-checking for easier viewing of debugging. Junos OS also supports this feature in ca-profile settings.

DoD PKI Setup

This section provides some notes on IKE configuration based on DoD PKI authentication. For more information, see "DoD PKI Introduction" on page 67.

- The DoD PKI uses a 2-tiered hierarchy of CAs; the device certificates are considered as a third or bottom tier. There are a couple of subordinate CAs and a root CA. You should retrieve and load the certificates for all these CAs in the Junos device.
- The DoD PKI supports one common name (CN) field in the DN.
- The DoD PKI requires multiple OU fields. The Junos device also supports multiple OU fields.
- You can download the DoD CRL files, or you can automatically use LDAP to retrieve the CRL. If you do use LDAP, make sure you have DNS set on the Junos device. This is required to resolve the name of the LDAP server. You can verify the connection by initiating the **ping** command from a Junos device to an LDAP server.
- The CRL file can be larger than 20 KB. Junos devices support up to 5 MB for the CRL.

Setting Up IKE Using DoD PKI Certificates

These steps are required to allow the Junos device to support Internet Key Exchange (IKE) tunnels based on DoD PKI authentication.

To set up an IKE using the DoD PKI certifications:

- Set up the IKE gateway by choosing a proposal which uses an RSA algorithm.

NOTE: You must specify a distinguished name as the IKE ID. Since the DoD PKI certificates do not support the **SubjectAlternativeName** V3 extensions, the default FQDN (hostname + domain name) may not work.

- In the IKE gateway configuration, select the appropriate preferred local certificate, and peer CA certificate. The peer type can be X.509 or PKCS7. Select the X.509 peer type. If the tunnels do not work, then try the other format.

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Example: Configuring the PKI in Junos OS on page 13
- Verifying the PKI Configuration on page 43
- Appendix A: Frequently Asked Questions on page 49
- Glossary of PKI Related Terms on page 72

Appendix D: Simple Certificate Enrollment Protocol

During the normal life cycle of a PKI certificate, managing a local certificate expiration involves following the tasks that an administrator has to perform :

- Delete the existing certificate, certificate request, and key pair.
- Generate a new key pair.
- Generate a new certificate request.
- Load the newly issued local certificate onto the device.

By using Simple Certificate Enrollment Protocol (SCEP), an administrator can manage the expiration of local certificates by automatically reenrolling and retrieving new certificates. Also, SCEP can ease the process of the initial certificate request and retrieval process by automatically retrieving the certificate from the certificate authority (CA) server (if the CA server supports SCEP).

The Junos OS Release with Enhanced Services 9.0 or later supports SCEP. Currently only the Microsoft SCEP is supported. Support for the Verisign SCEP and Entrust SCEP is not currently available.

NOTE: **You can troubleshoot SCEP related issues by enabling PKI traceoptions within the security PKI hierarchy.**

To configure SCEP:

1. Generate the key-pair:

```
user@host> request security pki generate-key-pair certificate-id mscert1 size 1024

Generated key pair mscert1, key size 1024 bits
```

2. Configure the SCEP server under the security PKI hierarchy:

```
user@host> configure
Entering configuration mode
[edit]
```

```
user@host# edit security pki

[edit security pki]
user@host# set ca-profile msca-profile ca-identity msca2000 enrollment url
  http://172.19.50.129/certsrv/mscep/mscep.dll retry 3 retry-interval 3

[edit security pki]
user@host# show
ca-profile msca-profile {
  ca-identity msca2000;
  enrollment {
    url http://172.19.50.129/certsrv/mscep/mscep.dll;
    retry 3;
    retry-interval 3;
  }
}

[edit security pki]
user@host# commit and-quit
```

3. Generate your local certificate by enrolling through SCEP.

```
user@host> request security pki local-certificate enroll ca-profile msca-profile
  certificate-id mscert1 challenge-password "" domain-name tacdomain.com subject
  "CN=testuser,OU=Support,O=Juniper Networks,L=Sunnyvale,ST=CA,C=US"
```

4. Enroll your CA certificate (if required).

```
user@host> request security pki ca-certificate enroll ca-profile msca-profile

Fingerprint:
1b:02:cc:cb:0f:d3:14:39:51:aa:0f:ff:52:d3:38:94:b7:11:86:30 (sha1)
90:60:53:c0:74:99:f5:da:53:d0:a0:f3:b0:23:ca:a3 (md5)
Do you want to load the above CA certificate ? [yes,no] (no) yes
CA certificate for profile msca-profile loaded successfully
```

5. When the certificate is loaded, configure auto-reenrollment.

```
Example:
user@host> configure
Entering configuration mode

[edit]
user@host# edit security pki

[edit security pki]
user@host# set auto-re-enrollment certificate-id mscert1 ca-profile-name msca-profile
  challenge-password "" re-enroll-triggertime- percentage 5 re-generate-keypair

[edit security pki]
user@host# show
ca-profile msca-profile {
  ca-identity msca-id;
  enrollment {
    url http://172.19.50.129/certsrv/mscep/mscep.dll;
    retry 3;
```

```
        retry-interval 3;
      }
    }
      auto-re-enrollment {
        certificate-id mscert1 {
          ca-profile-name msca-profile;
          challenge-password "$9$jx"; ## SECRET-DATA
          re-enroll-trigger-time-percentage 5;
          re-generate-keypair;
        }
      }

[edit security pki]
user@host# commit and-quit
```

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Example: Configuring the PKI in Junos OS on page 13
- Verifying the PKI Configuration on page 43
- Appendix A: Frequently Asked Questions on page 49
- Glossary of PKI Related Terms on page 72

Glossary of PKI Related Terms

C

CA certificate authority. In general, a CA is a trusted third-party organization that creates, enrolls, validates, and revokes digital certificates. The CA guarantees a user's identity and issues public and private keys for message encryption and decryption (coding and decoding).

In this example, CA is the server (or set of servers) that signs certificates for virtual private network (VPN) gateways and user systems (for client remote access server (RAS) VPN). CA also generates certificate revokation lists (CRLs) which are lists of revoked certificates. A CA acts as the trusted third party between two VPN gateways that are authenticating each other, using certificates.

CDP CRL Distribution Point (CRL-DP). A CDP is the place to retrieve a CA's latest CRLs. This is usually a Lightweight Directory Access Protocol (LDAP) server or HTTP (web) server. The CDP is normally expressed as an *ldap://host/dir* or *http://host/path* URL.

Certificates (certs) A certificates is the combination of an entity's identity and public key into one file. This file is digitally signed by a certificate authority (CA), and is validated by other servers that trust this CA. Certificates have a finite lifetime and are defined by a start time and an end time. The certificate becomes invalid when the life time expires. When the certificate expires, a certificate renewal or a new certificate request is required.

CRL certificate revocation list. The certificate server (CA) periodically publishes the CRL which includes a list of certificates that are prematurely invalid. This list also includes reasons for revocation and the names of the entities that issued certificates. A CRL prevents use of digital certificates and signatures that are expired or invalid.

D

DN distinguished name. A DN is the set of fields and values that uniquely define a certificate in a VPN gateway or remote access server (RAS) VPN client identity. The DN is also called the *subject* of the certificate. This DN identity can be used as the IKE ID.

The DN form includes the following fields:

- CN — Username, server DNS name, or almost any uniquely identifying string
- OU — Organizational unit (example: Sales)
- O — Organization (example: Juniper Networks)
- L — Locality (example: San Francisco)
- S — State (example: CA)
- C — Country (example: US)

DNS resolver Domain Name System resolver. Domain Name Server setting that needs to be set on the Juniper Networks virtual private network (VPN) device to help resolve fully qualified domain name (FQDN) names into IP addresses. Many CAs and certificates use hostnames which need to be resolved into IP addresses.

F

FQDN fully qualified domain name. The hostname and domain name for a specific system.

An FQDN is usually the name given to a device on the Internet, including the DNS-based zone that the device is in. Examples include *www.juniper.net*, *ocsp.chemistry.nwu.edu*, *nsgw1.dklein.org*, *ftp1.whitehouse.gov*, and *ca2.nit.disa.mil*.

I

IKE Internet Key Exchange. In general, IKE is the is part of IPsec that provides ways to securely negotiate the shared private keys that the authentication header (AH) and Encapsulating Security Payload (ESP) portions of IPsec need to function properly. IKE employs Diffie-Hellman methods and is optional in IPsec (the shared keys can be entered manually at the endpoints).

In this example, it is an Internet Security Association and Key Management Protocol (ISAKMP)/Oakley-based process used by two VPN gateways to identify and authenticate each other. In addition, key generation for packet-level authentication (integrity) and encryption (privacy) is handled during IKE.

The IKE RFC defines following two basic gateway authentication mechanisms:

- Preshared key (like a password)
- Digital certificates (certificates) based on RSA private/public key pairs

NOTE: **This document focuses on the digital certificates (certificates) based on RSA private/public key pairs.**

IKE ID IKE identity. The IKE ID defines how two VPN peers can identify each other.

The IKE ID can be one of the following:

- IP address (example: 21.62.2.252)
- fully qualified domain name (FQDN) (example: vpn1.juniper.net)
- U-FQDN or e-mail address (example: johndoe@juniper.net)
- distinguished name (DN) (example: CN=John Doe, OU=eng, O=Juniper, C=US).

IPsec IP Security. In general, IPsec is a standard way to add security to Internet communications. The secure aspects of IPsec are usually implemented in three parts: the authentication header (AH), the Encapsulating Security Payload (ESP), and the Internet Key Exchange (IKE).

In this example, IPsec is the protocol used to authenticate, encrypt, and encapsulate IP packets between two VPN/IKE peers, thereby creating a tunnel.

N

NSR NetScreen-Remote client is the software for Windows-based PCs or laptops, which allow clients to set up a personal VPN to a Junos or other IPsec gateway, opposed to a site-to-site VPN in which two VPN devices set up a VPN tunnel between two sites, containing many hosts.

O

OCSP Online Certificate Status Protocol. The OCSP protocol is used by a VPN device to contact a VA (validation authority) to check on the validity of a certificate. This is a more scalable alternative to the use of certificate revocation lists (CRL), CRL Distribution Points (CDPs).

P

PKCS Public-Key Cryptography Standards. PKCS are the series of standards established by RSA laboratories.

The PKCS are:

- PKCS7—The Cryptographic Message Syntax Standard defines how messages are encoded and digitally signed. The PKCS7 includes a certificate itself and it is also referred to as a p7 file.
- PKCS10—The Certificate Request Syntax Standard defines how a virtual private network (VPN) gateway can form a request for a certificate that can be sent to a certificate authority (CA). The request, also referred to as a p10 file, usually includes the VPN gateway's identity and a public key. The CA digitally signs the request with its own private key and return a p7 (certificate) file.
- PKCS11—The Cryptographic Token Interface Standard defines how to store certificates and private keys on a token card. This is not relevant for the SRX Series and J Series devices but can be relevant to the NSR devices.
- PKCS12—The Personal Information Exchange Syntax Standard defines how to bundle an entity's certificate and public/private key pair into a password-protected file. The PKCS12, also referred as a p12 file, facilitates moving a user from one machine to another for client VPNs. This standard is more frequently used by NSR devices.

PKI public key infrastructure. In general, PKI is a hierarchy of trust that enables users of a public network to securely and privately exchange data through the use of public and private cryptographic key pairs that are obtained and shared with peers through a trusted authority.

In this example, PKI is the set of objects that allows the use of digital certificates between two entities (such as virtual private network (VPN) gateways). This includes a certificate authority (CA), registration authority (RA), certificates, certificate revocation lists (CRLs), CRL Distribution Points (CDPs), and Online Certificate Status Protocol (OCSP).

S

SCEP Simple Certificate Enrollment Protocol. In general, SCEP is a protocol for digital certificates that supports certificate authority (CA) and registration authority (RA) public key distribution, certificate enrollment, certificate revocation, certificate queries, and certificate revocation list (CRL) queries.

In this example, SCEP is the protocol used to allow a device to generate a certificate request and automatically submit the request to a CA. You can use this protocol only if both the device and the CA support it. This protocol makes certificate enrollment and reenrollment easier than manually collecting a PKCS10 from the device and then submitting it to a CA.

Junos OS Release 9.0 or later supports SCEP. For more information on SCEP, see "Appendix D: Simple Certificate Enrollment Protocol" on page 70.

U

U-FQDN user-fully qualified domain name. This is usually the e-mail address given to users on the Internet or host network. For example: johndoe@juniper.net, user1@org.corp.com, and john.smith@jscorp.com.

Related Documentation

- Introduction to PKI in Junos OS on page 1
- Example: Configuring the PKI in Junos OS on page 13
- Verifying the PKI Configuration on page 43
- Appendix A: Frequently Asked Questions on page 49