Cloud Certificate Manager

User Guide

Issue	01
Date	2025-10-29

HUAWEI CLOUD COMPUTING TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Service Overview

1.1 What Is Cloud Certificate Manager?

Cloud Certificate Manager (CCM) is a service that issues certificates and manages the lifecycle of certificates in the cloud. CCM includes the SSL Certificate Manager (SCM) and Private Certificate Authority (PCA) services.

What Is SCM?

SCM is a platform to centrally manage your Secure Sockets Layer (SSL) certificates.

- What Is an SSL Certificate?

  An SSL certificate is an SSL-compliant digital certificate issued by a trusted CA.

  After an SSL certificate is deployed on a server, HTTPS is enabled on the server. The server uses HTTPS to establish encrypted links to the client, ensuring data transmission security.
- SSL certificates can help you:
  - Authenticate websites and ensure that data is sent to the correct clients and servers.
  - Set up encrypted connections between clients and servers, preventing data from being stolen or tampered with during transmission.

What Is PCA?

Private Certificate Authority (PCA) is a private certificate and CA management platform. You can use CCM to set up a complete CA hierarchy and use it to issue and manage private certificates for your organization. It is used to authenticate application identities and encrypt and decrypt data within your organization.

Certificates issued by a private CA are trusted only within your organization, but not the Internet.

1.2 Features

With CCM, you can easily authenticate application identities and encrypt data within your organization.

SSL Certificate Manager (SCM)

Feature	Description
Centralized SSL certificate management	CCM provides you with a one-stop management platform. You can upload certificates and private keys to our platform to centrally manage certificates, apply for review, view the domain names bound to certificates and certificate expiration time, change certificate names, and delete expired certificates, helping you improve certificate O&M efficiency.

Private Certificate Authority (PCA)

Feature	Description
Hosting CAs	PCA provides CAs and supports multiple key algorithms, including RSA_2048, RSA_4096, EC_P256, and EC_P384. It supports X.509 v3 certificates, as well as multi-level extension and multi-level authentication of CAs. It uses symmetric and asymmetric algorithms which are internationally used and comply with the PKI and CA international standards.
Private certificate lifecycle management	PCA allows you to apply for, download, and revoke private certificates. It can manage more than 10 million certificates.
Key lifecycle management	PCA uses Key Management Service (KMS) and Hardware Security Modules (HSMs) to protect CA keys. It supports the generation, update, deletion, and restoration of key pairs for software and hardware.
Certificate Revocation List (CRL) management	PCA periodically releases and updates a private certificate revocation list (CRL) to your OBS buckets for downloading. Applications, services, and devices can use CRLs to periodically check certificate status.
Automated API integration	PCA provides APIs to help you efficiently develop and deploy products.

1.3 Advantages

One-Stop SSL certificate management

You can upload SSL certificates you have bought from third parties to CCM and manage all your certificates in one place. For those external certificates, you can query and manage them.

Quickly Deploying Certificates to Cloud Products

You can deploy an SSL certificate to other services (such as ELB and WAF) you subscribe in just a few clicks.

Private CA Hosting

You can easily manage CAs and certificates without having to build or maintain complex CA infrastructures.

Complete CA Hierarchy

You can create a flexible CA hierarchy, including root CAs and subordinate CAs. External CAs are also supported to meet the deployment requirements of more applications.

Managing the Private Certificate Lifecycle

PCA allows you to centrally manage certificates and keys. It can manage millions of certificates, and quickly notify tenants of certificate status using the CRL to prevent certificate expiration.

Varied Key Algorithms for Private Certificates

PCA supports different key algorithms, such as RSA_2048, RSA_4096, EC_P256, and EC_P384. It supports the x.509 v3 certificate format and complies with the PKI and CA international standards.

Secure and Reliable Storage of Private Certificate Keys

PCA uses Key Management Service (KMS) to store keys securely.

Flexible Integration of Private Certificate APIs

PCA provides you with great flexibility through abundant APIs that allow you to efficiently integrate and deploy products in the development environment.

1.4 Application Scenarios

Enabling of HTTPS on Services such as WAF and ELB

CCM enables you to quickly deploy SSL certificates to your services, such as WAF and ELB.

Internal Application Data Security Control

You can use PCA to establish an internal certificate management system for your enterprise and issue and manage self-signed private certificates to authenticate identities, encrypt and decrypt data, and secure data transmission within the enterprise.

IoV

Telematics Service Providers (TSPs) can use PCA to issue a certificate to each vehicle terminal, thereby providing security capabilities such as authentication and encryption during vehicle-vehicle, vehicle-cloud, and vehicle-road interaction.

IoT

The Internet of Things (IoT) platform can use PCA to issue a certificate to each IoT device to implement IoT device identity verification and authentication, ensuring device access security in IoT scenarios.

1.5 Basic Concepts

1.5.1 Related Concepts in SCM

This topic describes the concepts related to SSL Certificate Manager (SCM).

Digital Certificate

A digital certificate is a file digitally signed by a CA and contains information about the owner of a public key and the public key. It is a trusted certificate issued by an authority to a website. The simplest certificate contains a public key, name, and digital signature of the CA. Another important feature of a digital certificate is that it is valid only within a specific period of time.

SSL Protocol

SSL is an encryption protocol that secures communication over a computer network. It establishes an encrypted channel between the browser and website to prevent information from being stolen or tampered with during transmission.

Certificate Authority

A Certificate Authority (CA) is an authority responsible for issuing and managing digital certificates. As a trusted third party in e-commerce transactions, the CA verifies the validity of public keys in the public key system.

HTTPS

Hypertext Transfer Protocol Secure (HTTPS) is a website encryption transmission protocol based on the SSL protocol. HTTPS activates an SSL encrypted channel between a web browser and a website server for a user to visit the website where an SSL certificate has been installed. The channel enables high-strength bidirectional encrypted transmission to prevent leakage or tampering of the data in transit. HTTPS is the secure version of HTTP.

CSR

A certificate signing request (CSR) is a message sent from an applicant to a CA to apply for an SSL certificate. A CSR file contains a public key and a distinguished name (DN). Typically, a CSR file is generated by a web server, and a pair of public and private keys are created along with the CSR file.

SSL Certificate Validity Period

From September 1, 2020, only one-year SSL certificates can be issued by CAs around the world.

1.5.2 PCA-related Concepts

This topic describes the concepts related to Private Certificate Authority (PCA) service.

Root CA

The public key certificate of a CA. A root certificate is the trust anchor in the public key infrastructure (PKI) system. It can issue subordinate CAs, private certificates, and certificate revocation lists (CRLs). After a root CA is imported into the client trust list, the certificates issued by it can be validated as trusted.

Subordinate CA

A subordinate CA, or intermediate CA or child CA, is used to isolate the root CA from the private certificates. It is the key to divide the CA hierarchy. A subordinate CA validates certificates at the next layer in the certificate chain. If the path length of a subordinate CA is greater than 0, it can issue lower-layer subordinate CAs.

NOTE

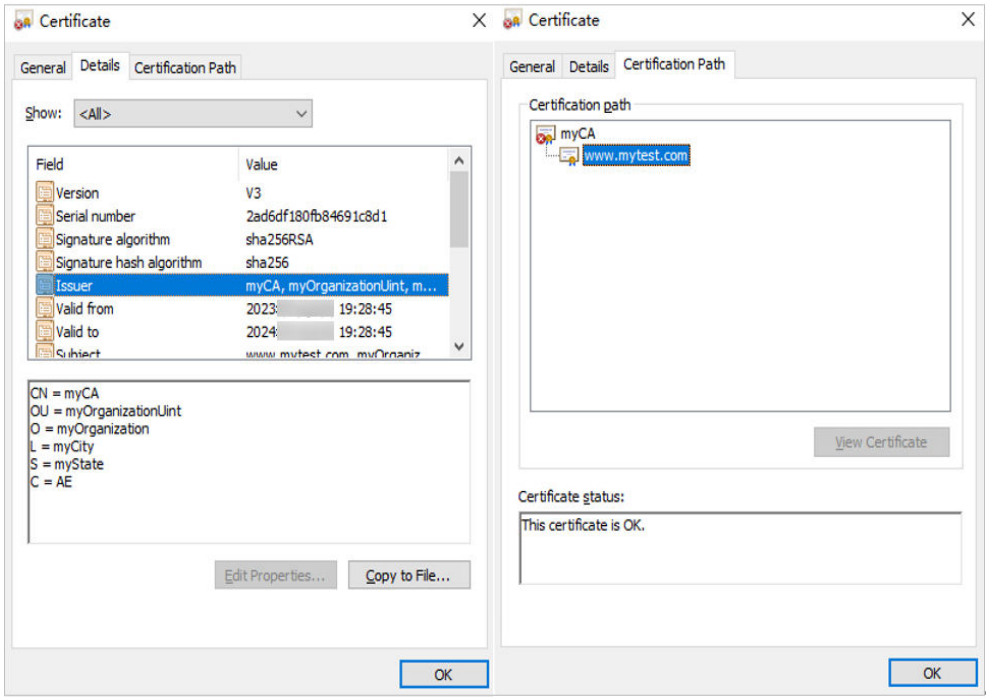
The path depth of a subordinate CA controls how many layers of subordinate CAs the current CA can issue. (The last layer of the certificate chain is a private certificate).

Private certificate

A private certificate is an end-entity certificate, which is installed on an end entity, including certificates used for the client (or client certificates) and certificates used

for the server (or server certificates). An end-entity certificate is at the bottom layer of a certificate chain and is used to authenticate an entity. It cannot be used to issue a certificate and is a credential for HTTPS communication between the entity that owns the certificate and other entities. **Figure 1-1** shows the content of a private certificate.

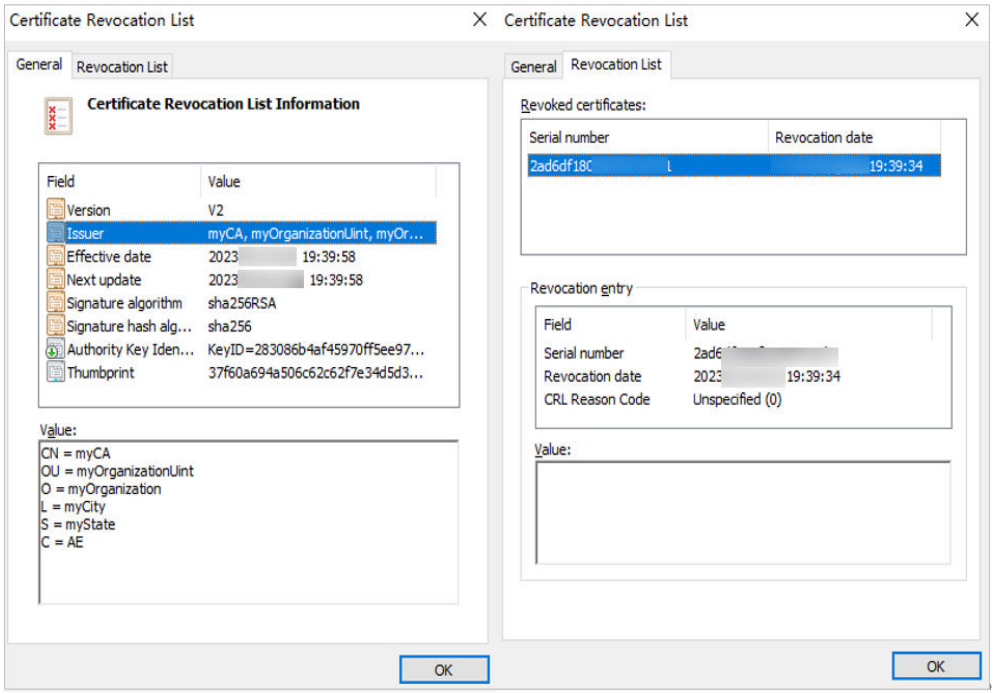
Figure 1-1 Private certificate



Certificate Revocation List (CRL)

A certificate revocation list (CRL) is a list of certificates revoked by the parent CA when they are still valid. The revoked certificates include subordinate CAs and private certificates. A CRL is a structured data file in a fixed format. It contains the issuer information, time when the CRL takes effect, time when the CRL is updated next time, issuing algorithm, fingerprint, as well as the serial number, revocation time, and revocation reason code of a revoked certificate. **Figure 1-2** provides more details.

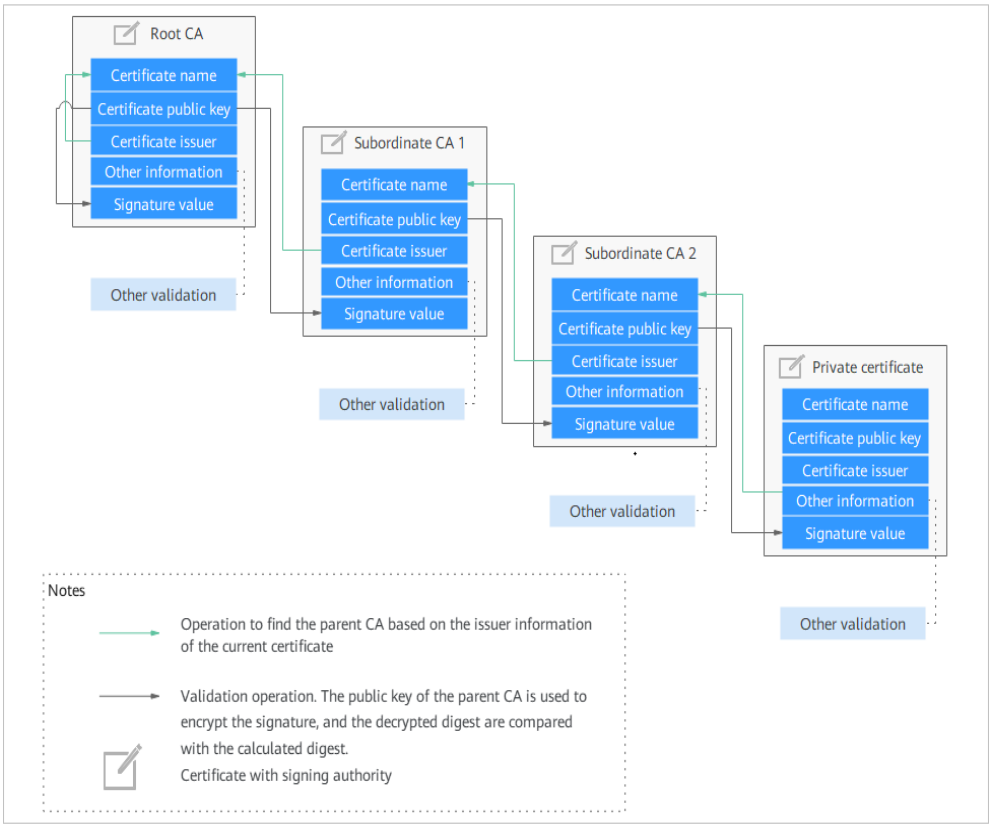
Figure 1-2 Certificate Revocation List (CRL)



Certificate chain

A certificate chain is a file that combines all certificates from the root CA to the private certificates in a fixed sequence. A certificate chain is used to validate certificates layer by layer. **Figure 1-3** shows an example certificate chain.

Figure 1-3 Certificate chain



Certificate validation involves the following aspects:

- Integrity of the certificate chain and validity of certificates
- Validity of the root CA, which is preinstalled in its trust store.

The following information is validated during the validation process:

- Subject the certificate owner claims, such as the domain name of the server
- Certificate validity period
- Key usage, such as key negotiation and digital signatures.
- Digital signature
- Whether the certificate has been revoked.

> **NOTE**
>
> Not all validation items are listed here. The X.509 certificate allows users to add multiple customized extension items. For details, see related international standards.

PCA Certificate Validity Period

In a certificate chain, the root CA is the trust anchor for all of the subordinate CAs and the end-entity certificates below it. Once the root CA expires, all certificates issued by the root CA and its subordinate CAs are no longer trusted. The validity period of the root CA is the upper limit of the validity period of all lower-layer certificates. Even if the validity period of a lower-layer certificate can be set to a value greater than that of the root CA (if not mandated), the certificate chain validation fails as long as the root CA in the chain expires.

In the PCA service, the validity period of a certificate cannot be longer than that of its parent CA. This ensures that the validity periods decrease gradually in the certificate chain from the root CA to the private certificate. **Table 1-1** lists the restrictions PCA places on validity periods of certificates.

The validity periods of different types of certificates vary depending on their roles. The more frequently a certificate is used, the higher the risk of key leakage is. Therefore, the validity period of frequently used certificate should be as short as possible. A root CA is used only to issue subordinate CAs. Root CAs are infrequently used, and the tightest protection measures are used for them. (KMS is used for CA key management in PCA). The validity period of a root CA is about 10 to 30 years. The lower the layer of a subordinate CA, the shorter the validity period. The subordinate CA at the lowest layer is used to issue private certificates, so its validity period is usually set to 2 to 5 years. A private certificate is frequently used during communications. The validity period of a private certificate can be set to several hours, months, or one or two years based on the security requirements of application scenarios.

Table 1-1 Certificate validity period constraints

Certificate Type	Min. Validity Period	Max. Validity Period	Extension Supported	Remarks
Root CA	1 hour	30 years	No	None
Subordinate CA	1 hour	20 years	No	The root CA must within the validity period.
Private certificate	1 hour	20 years	No	The root CA must within the validity period.

1.6 Billing Description

Billing Items

You will be billed based on how many private CAs and private certificates you have.

Billing

Private CAs and private certificates are billed on a pay-per-use basis. A root CA is billed from the moment it is created. Subordinate CAs are not billed until they are activated. **Table 1-2** provides details about how private CAs are billed.

Table 1-2 Private CA billing description

CA Status	Billed or Not	Remarks
Pending activation	No	A private certificate can be used only after being activated.
Activated	Yes	An activated CA can issue certificates, revoke certificates, and sign CRLs. **NOTICE** How an activated CA works depends on what type of key it owns.
Disabled	Yes	A disabled CA cannot be used to issue certificates, but it can still revoke certificates and sign CRLs. **NOTICE** How an activated CA works depends on what type of key it owns.
Pending deletion	• If a private CA in the **Pending deletion** status is finally deleted as scheduled, no additional fee is incurred for the pending deletion period. • If the deletion is canceled for a private CA in the **Pending deletion** status, the pending period for the private CA will be billed. For example, if you delete a private CA at 00:00 on January 1, 2022 and the private CA is deleted seven days later as scheduled, you will not be billed for the seven days. If you cancel the scheduled deletion at 00:00 on January 4, 2022 and the private CA is not deleted, you will still be billed for the CA for the period from 00:00 on January 1, 2022 to 00:00 on January 4, 2022. **NOTICE** Only **Disabled** or **Expired** private CAs can enter into the **Pending deletion** status when they are deleted. This means when you delete a disabled or expired certificate, it cannot be deleted immediately. It takes at least 7 days for a scheduled deletion to take effect (depending on the delay time you configured).	Only the deletion cancellation is provided.

CA Status	Billed or Not	Remarks
Expired	Yes	An expired private CA is no longer trusted and cannot issue or revoke certificates or sign CRLs, but it still uses the CA quota and can be exported. **CAUTION** If you no longer need it, delete it as soon as possible to stop the billing.
Revoked	No	Only subordinate CAs can be revoked. If the CRL function is enabled for their parent CA, the revocation information will be published in the CRL of subordinate CAs. Revoked private CA will no longer be trusted.

Changing Billing Options

Private CAs and private certificates are billed on a pay-per-use basis.

To stop billing for a private CA or certificate, delete it.

1.7 Permissions Management

If you need to assign different permissions to employees in your enterprise to access your CCM resources, IAM is a good choice for fine-grained permissions management. IAM provides identity authentication, permissions management, and access control, helping you secure the access to your cloud resources.

With IAM, you can use your account to create IAM users for your employees, and assign permissions to control their access to specific resource types. For example, if you have software developers and you want to assign them the permission to access CCM but not to delete CCM or its resources, then you can create an IAM policy to assign the developers the permission to access CCM but prevent them from deleting CCM related data.

If your account does not need individual IAM users for permissions management, then you may skip over this chapter.

CCM Permissions

By default, new IAM users do not have permissions assigned. You need to add a user to one or more groups, and attach permissions policies or roles to these

groups. Users inherit permissions from the groups to which they are added and can perform specified operations on cloud services based on the permissions.

CCM is a global service deployed for all physical regions. Therefore, CCM permissions are assigned to users in the Global project, and the users do not need to switch regions when accessing CCM.

You can grant users permissions by using roles and policies.

- Roles: A type of coarse-grained authorization mechanism that defines permissions related to users responsibilities. This mechanism provides a limited number of service-level roles for authorization. If one role has a dependency role required for accessing CCM, assign both roles to the users. Roles are not an ideal choice for fine-grained authorization and secure access control.
- Policies: A fine-grained authorization mechanism that defines permissions required to perform operations on specific cloud resources under certain conditions. This mechanism allows for more flexible policy-based authorization and meets secure access control requirements. For example, you can grant CCM users the permissions to manage only a certain type of resources.

Table 1-3 lists the system-defined roles of CCM.

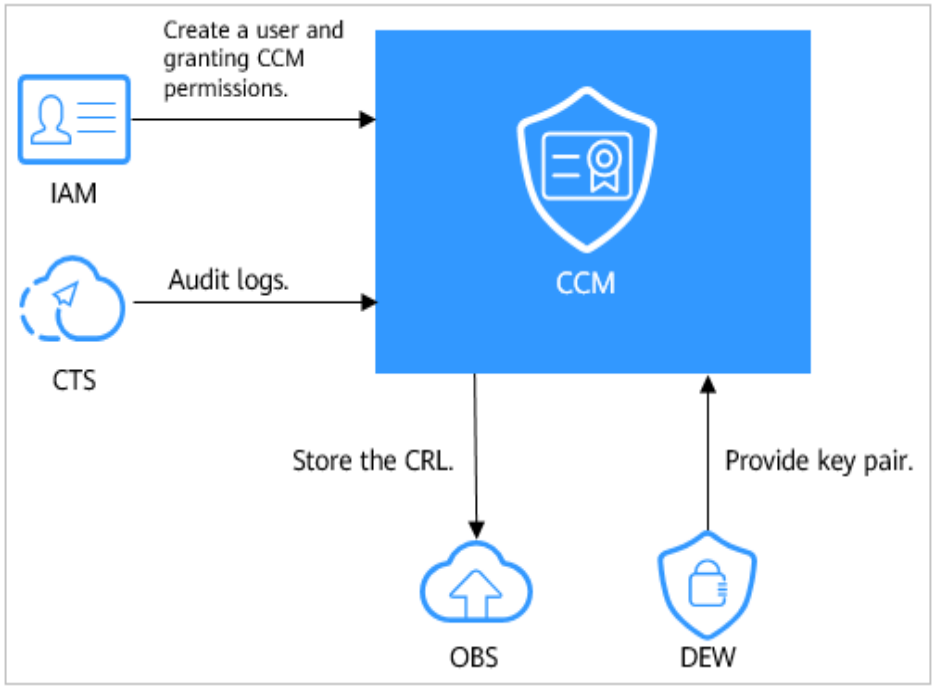
Table 1-3 System role supported by CCM

Role/Policy	Description	Type	Dependency
SCM FullAccess	All permissions for SCM	System-defined policy	BSS Administrator: a system role, which is the administrator of the billing center (BSS) and has all permissions for the service. WAF FullAccess: system policy, which is the Web Application Firewall (WAF) administrator. ELB FullAccess: a system policy that has all permissions for Elastic Load Balance (ELB). EPS FullAccess: a system-defined policy that has all Enterprise Project Management Service (EPS) permissions. OBS Administrator: a system policy, which is the Object Storage Service (OBS) administrator. DNS FullAccess: a system policy that has all permissions for Domain Name Service (DNS), including creating, deleting, querying, and modifying DNS resources.
PCA FullAccess	All permissions for PCA	System policy	BSS Administrator role is required for creating a private CA or private certificate. EPS FullAccess: a system-defined policy that has all Enterprise Project Management Service (EPS) permissions. OBS Administrator: a system policy, which is the Object Storage Service (OBS) administrator.

1.8 Related Services

Figure 1-4 shows the dependencies between CCM and other services.

Figure 1-4 CCM and related services



Elastic Load Balance (ELB)

You can purchase SSL certificates on the SCM console and deploy them on load balancers provided by ELB in just a few clicks.

Web Application Firewall (WAF)

You can purchase SSL certificates on the SCM console and deploy them on WAF in just a few clicks.

Object Storage Service (OBS)

OBS is an object-based cloud storage service. It provides massive, secure, highly reliable, and low-cost data storage capabilities. When you revoke a certificate in CCM, the CRL of the revoked certificate is stored in your OBS bucket for query.

Data Encryption Workshop (DEW)

DEW provides key pair generation and protection for CCM.

Cloud Trace Service (CTS)

You can use CTS to record CCM operations for querying, auditing, or backtracking later.

Identity and Access Management (IAM)

IAM provides the permission management function for CCM.

Only users who have PCA FullAccess and SCM FullAccess permissions can use CCM.

To obtain the permissions, contact the users who have the Security Administrator permissions.

1.9 Personal Data Protection

To ensure that your personal data, such as the username, password, and mobile phone number, will not be obtained by unauthorized or unauthenticated entities or people and to prevent data leakage, CCM encrypts your personal data before storing it to control access to the data and records logs for operations performed on the data.

Personal Data

Table 1-4 lists the personal data generated or collected by CCM.

Table 1-4 Personal data

Type	Collection Method	Can Be Modified	Mandatory
Tenant ID	• Tenant ID in the token when an operation is performed on the console • Tenant ID in the token when an API is invoked	No	Yes. The tenant ID is the certificate resource ID.
Email Address	Email address entered when applying for the private certificate	No	No

Storage

CCM uses encryption algorithms to encrypt your sensitive data and stores encrypted data.

- Tenant IDs: Tenant IDs are not sensitive data and are stored in plaintext.
- email address: encrypted for storage

Access Control

Token authentication is required for accessing your personal data in the CCM database.

Logging

CCM logs all operations involving personal data, such as editing, querying, and deleting personal data. The logs are uploaded to Cloud Trace Service (CTS). You can view only the logs for your operations.

2 SSL Certificate Manager (SCM) User Guide

2.1 Installing an SSL Certificate

2.1.1 Installing an SSL Certificate on a Web Server

2.1.1.1 Downloading an SSL Certificate

After an SSL certificate is issued, you need to download it. Then, you can install it on your web server and modify server configuration to let the SSL certificate work.

This topic describes how to download an SSL certificate on the SCM platform.

Prerequisites

The certificate is in the **Hosted** status.

Constraints

- A certificate can only be downloaded when it is in its validity period.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 In the **Operation** column of the row containing the desired certificate, click **Download**.

Step 5 On the certificate details page, click **Download**.

Step 6 Install the certificate on the corresponding server for the SSL certificate to work.

The procedure for installing an SSL certificate varies depending on the web server. The following describes how to install an SSL certificate on mainstream web servers.

- For details about how to install an SSL certificate on a Tomcat server, see **Installing an SSL Certificate on a Tomcat Server**.
- For details about how to install an SSL certificate on an Nginx server, see **Installing an SSL Certificate on an Nginx Server**.
- For details about how to install an SSL certificate on an Apache server, see **Installing an SSL Certificate on an Apache Server**.
- For details about how to install an SSL certificate on an IIS server, see **Installing an SSL Certificate on an IIS Server**.

----End

Description of Downloaded Certificate Files

Different types of certificate files can be downloaded depending on if you select **System generated CSR** or **Upload a CSR** when you applied for the certificate.

- **System generated CSR**

  The downloaded certificate package contains **Apache**, **IIS**, **Nginx**, and **Tomcat** folders as well as the **domain.csr** file. See **Table 2-1** for details. **Figure 2-1** shows an example.

Figure 2-1 Decompressing an SSL certificate package

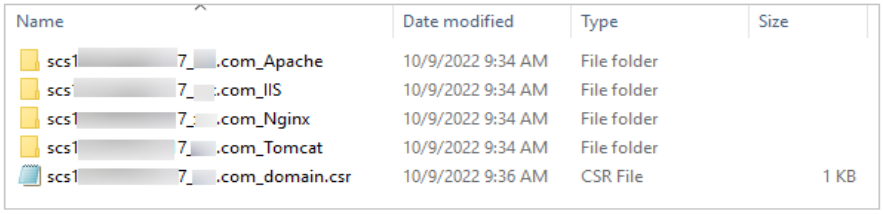


Table 2-1 Description of files/folders in the downloaded certificate

File/Folder Name	Content
Tomcat	**keystorePass.txt**: certificate password **server.jks**: certificate file
Nginx	**server.crt**: certificate file, which contains two segments of certificate code (server certificate and intermediate CA certificate respectively) **server.key**: certificate's private key file, which contains a segment of private key code of the certificate

File/Folder Name	Content
Apache	**ca.crt**: certificate chain file, which contains a segment of intermediate CA code. **server.crt**: certificate file, which contains a segment of server certificate code **server.key**: certificate's private key file, which contains a segment of private key code of the certificate
IIS	**keystorePass.txt**: certificate password **server.pfx**: certificate file
domain.csr	Certificate signing request.

- **Upload a CSR**

  The downloaded certificate package contains only the **server.pem** file. The file contains two segments of certificate code, namely, the server certificate and intermediate CA certificate.

  SCM does not store your private keys. Keep them properly, so keep them safe. When installing the certificate on a server, you will need to provide the file path to the location of your private keys.

2.1.1.2 Installing an SSL Certificate on a Tomcat Server

This section describes how to install an SSL certificate on a Linux Tomcat 7 server. The installation process is similar for other Tomcat servers. When the certificate is installed, it secures communication between your server and the client through SSL.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Hosted** status.
- You have downloaded the certificate. For details, see **Downloading an SSL Certificate**.
- You have installed the OpenSSL tool.

  Download the latest OpenSSL installation package from **https://www.openssl.org/source/**. The OpenSSL must be 1.0.1g or later.
- You have installed Keytool.

  Keytool is typically included in the Java Development Kit (JDK) tool package.

Constraints

- Before installing the certificate, enable port 443 on the server where the SSL certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- The domain name to be run on the target server must be the same as the one associated with the certificate. Otherwise, the web browser will display a message indicating that the connection to the domain name is insecure.

Procedure

The installation process is as follows (for Tomcat 7 servers):

Step 1: Obtaining Files → **Step 2: Creating a Directory** → **Step 3: Modifying Configuration Files** → **Step 4: Restarting the Tomcat** → **Verifying the Result**

Step 1: Obtaining Files

Before installing a certificate, obtain the certificate file and password file. Perform the following operations based on the value selected for **CSR** when applying for a certificate:

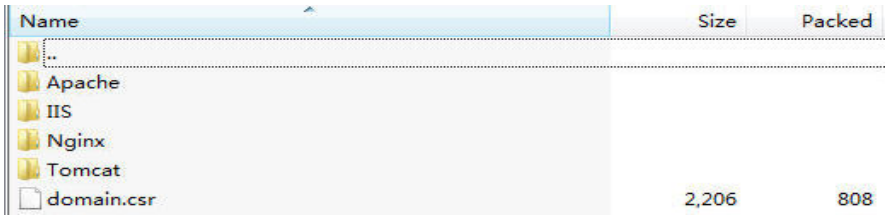
- If you select **System generated CSR** for **CSR** when applying for a certificate, perform the operations according to the instructions in **System generated CSR**.
- If you select **Upload a CSR** for **CSR** when applying for a certificate, perform the operations according to the instructions in **Upload a CSR**.

Detailed operations are as follows:

- **System generated CSR**
  a. Decompress the downloaded certificate file on your local PC.

     The downloaded file contains the **Apache**, **IIS**, **Nginx**, and **Tomcat** folders as well as the **domain.csr** file. **Figure 2-2** shows an example.

     Figure 2-2 Decompressing an SSL certificate package on a local computer

  b. Obtain ***Certificate ID_Domain name bound to the certificate*_server.jks** and ***Certificate ID_Domain name bound to the certificate*_keystorePass.txt** from ***Certificate ID_Domain name bound to the certificate*_Tomcat**.
- **Upload a CSR**
  a. Decompress the downloaded certificate package to obtain the ***Certificate ID_Domain name bound to the certificate*_server.pem** file.

The ***Certificate ID_Domain name bound to the certificate*_server.pem** file contains two segments of certificate codes **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**, which are the server certificate and intermediate CA certificate respectively.

b. Use OpenSSL to convert the PEM certificate into a PFX certificate and obtain the **server.pfx** file.

i. Save the PEM certificate and the private key **server.key** generated during CSR generation to the **bin** directory in the OpenSSL installation directory.

ii. In the **bin** directory of the OpenSSL installation directory, run the following command to convert the PEM certificate into a PFX certificate and press **Enter**:

openssl pkcs12 -export -out server.pfx -inkey server.key -in *Certificate ID_Domain name bound to the certificate*_**server.pem**

The command output is as follows:

```
Enter Export Password:
```

iii. Enter the password of the PFX certificate and press **Enter**.

The password is user-defined. Set it as required.

The command output is as follows:

```
Verifying - Enter Export Password:
```

NOTE

Record the password of the PFX certificate. The password of the JKS certificate must be the same as that of the PFX certificate. Otherwise, the Tomcat service may fail to start.

To improve password security, set the password based on the following rules:

- Consists of 8 to 32 characters.
- Must contain at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters ~`!@#$%^&*()_+|{}:"<>?-=\[];',./

iv. Re-enter the password of the PFX certificate and press **Enter**.

If no error information is displayed, the **server.pfx** file has been generated in the OpenSSL installation directory.

c. Use Keytool to convert the PFX certificate into a JKS certificate and obtain the **server.jks** file.

i. Copy the **server.pfx** file generated in **b** to the **%JAVA_HOME%/jdk/bin** directory.

ii. In the **%JAVA_HOME%/jdk/bin** directory, run the following command and press **Enter**:

keytool -importkeystore -srckeystore server.pfx -destkeystore server.jks -srcstoretype PKCS12 -deststoretype JKS

The following message is displayed.

```
Enter the destination keystore password:
```

iii. Enter the password of the JKS certificate and press **Enter**.

NOTICE

Set the password of the JKS certificate to the same as that of the PFX certificate. Otherwise, Tomcat may fail to start.

The command output is as follows:

```
Re-enter the new password:
```

iv. Re-enter the password of the JKS certificate and press **Enter**.

The command output is as follows:

```
Enter the source keystore password:
```

v. Enter the password of the PFX certificate set in **b.iii** and press **Enter**.

If information similar to the following is displayed, the conversion is successful and the **server.jks** file has been generated in the OpenSSL installation directory.

```
Entry for alias 1 imported successfully.
Import command completed: 1 entry successfully imported, 0 entries failed or canceled
```

vi. Create a **keystorePass.txt** file in the **%JAVA_HOME%/jdk/bin** directory and save the password of the JKS certificate in the file.

d. Place the converted certificate file **server.jks** and the new password file **keystorePass.txt** in the same directory.

Step 2: Creating a Directory

Create a **cert** directory in the Tomcat installation directory, and copy the **server.jks** and **keystorePass.txt** files to the **cert** directory.

Step 3: Modifying Configuration Files

NOTICE

Before modifying the configuration file, back up the configuration file. You are advised to deploy the configuration file in the test environment and then configure it on the production environment to avoid service interruptions caused by incorrect configurations.

The installation process is as follows (for Tomcat 7 servers):

1. Find the following parameters in the **server.xml** file in the Tomcat installation directory **conf**:

```
<!--
   <Connector port="8443" protocol="org.apache.coyote.http11.Http11Protocol"
              maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
              clientAuth="false" sslProtocol="TLS" />
   -->
```

2. Find the preceding parameters and delete the comment characters **<!--** and **-->**.
3. Add the following parameters. Change the values of the parameters according to **Table 2-2**.

```
keystoreFile="cert/server.jks"
keystorePass="Certificate key"
```


The complete example configuration is as follows. Modify other parameters based on your needs.

```
<Connector port="443" protocol="org.apache.coyote.http11.Http11Protocol"
               maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
    keystoreFile="cert/server.jks"
  keystorePass="Certificate key"
    clientAuth="false" sslProtocol="TLS" />
```

NOTICE

Do not directly copy all configuration. Only parameters **keystoreFile** and **keystorePass** need to be added. Set other parameters based on site requirements.

Table 2-2 Parameter description (1)

Parameter	Description
port	Port number to be used on the server. You are advised to set the value to **443**.
protocol	HTTP protocol. Retain the default value.
keystoreFile	Path for storing the **server.jks** file. The value can be an absolute path or a relative path. Example: **cert/server.jks**
keystorePass	Password of **server.jks**. Set this parameter to the password provided in the **keystorePass.txt** file. **NOTICE** If the password contains **&**, replace it with **&** to avoid configuration failure. An example command is provided as follows: If the password is **keystorePass="lx6&APWgcHf72DMu"**, change it to **keystorePass="lx6&APWgcHf72DMu"**.
clientAuth	Whether to require all customers to show the security certificate and authenticate their identity. Retain the default value.

4. Find the following parameters in the **server.xml** file in the Tomcat installation directory **conf**:

   ```
   <Host name="localhost"  appBase="webapps"
       unpackWARs="true" autoDeploy="true">
   ```

5. Change the value of **Host name** to the domain name bound to the certificate.

   The complete configuration is as follows (**www.domain.com** is used as an example):

   ```
   <Host name="www.domain.com"  appBase="webapps"
       unpackWARs="true" autoDeploy="true">
   ```

6. Save the configuration file.

Step 4: Restarting the Tomcat

Run the **./shutdown.sh** command in the **bin** directory of Tomcat to stop the Tomcat service.

After 10 seconds, run the **./startup.sh** command to start the Tomcat service. If the process is automatically started by the daemon process, you do not need to manually start the process.

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

2.1.1.3 Installing an SSL Certificate on an Nginx Server

This section describes how to install an SSL certificate on an Nginx 1.7.8 server running CentOS 7. The installation process is similar for other Nginx servers. When the certificate is installed, it secures communication between your server and the client through SSL.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Hosted** status.
- You have downloaded the certificate. For details, see **Downloading an SSL Certificate**.

Constraints

- Before installing the certificate, enable port 443 on the server where the SSL certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- The domain name to be run on the target server must be the same as the one associated with the certificate. Otherwise, the web browser will display a message indicating that the connection to the domain name is insecure.

Procedure

The installation process is as follows (for Nginx 1.7.8 servers running CentOS 7):

Step 1: Obtaining Files → **Step 2: Creating a Directory** → **Step 3: Modifying Configuration Files** → **Step 4: Verifying the Configuration** → **Step 5: Restarting Nginx** → **Verifying the Result**

Step 1: Obtaining Files

Before installing a certificate, obtain the certificate file and password file. Perform the following operations based on the value selected for **CSR** when applying for a certificate:

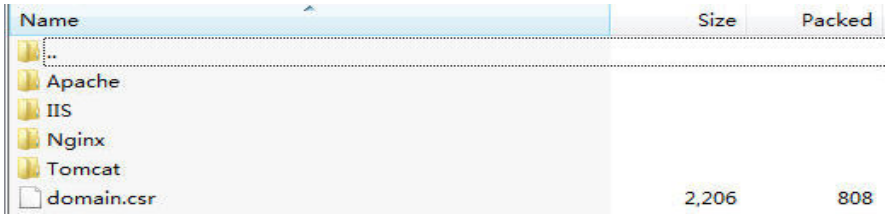
- If you select **System generated CSR** for **CSR** when applying for a certificate, perform the operations according to the instructions in **System generated CSR**.
- If you select **Upload a CSR** for **CSR** when applying for a certificate, perform the operations according to the instructions in **Upload a CSR**.

Detailed operations are as follows:

- **System generated CSR**
  a. Decompress the downloaded certificate file on your local PC.

     The downloaded file contains the **Apache**, **IIS**, **Nginx**, and **Tomcat** folders as well as the **domain.csr** file.

     Figure 2-3 Decompressing an SSL certificate package on a local computer

  b. Obtain the certificate file ***Certificate ID_Domain name bound to the certificate*_server.crt** and private key file ***Certificate ID_Domain name bound to the certificate*_server.key** from ***Certificate ID_Domain name bound to the certificate*_Nginx**.

     - The ***Certificate ID_Domain name bound to the certificate*_server.crt** file contains two segments of certificate codes **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**, which are the server certificate and intermediate CA certificate respectively.
     - The ***Certificate ID_Domain name bound to the certificate*_server.key** file contains a segment of private key code **-----BEGIN RSA PRIVATE KEY-----** and **-----END RSA PRIVATE KEY-----**.

- **Upload a CSR**
  a. Decompress the downloaded certificate package to obtain the ***Certificate ID_Domain name bound to the certificate*_server.pem** file.

     The ***Certificate ID_Domain name bound to the certificate*_server.pem** file contains two segments of certificate codes **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**, which are the server certificate and intermediate CA certificate respectively.

  b. Change the suffix of ***Certificate ID_Domain name bound to the certificate*_server.pem** to **crt**, that is, **server.crt**.

c. Place **server.crt** and the **server.key** private key generated during CSR generation in the same folder.

Step 2: Creating a Directory

Create a **cert** directory in the Nginx installation directory **conf**, and copy the **server.key** and **server.crt** files to the **cert** directory.

Step 3: Modifying Configuration Files

NOTICE

Before modifying the configuration file, back up the configuration file. You are advised to deploy the configuration file in the test environment and then configure it on the production environment to avoid service interruptions caused by incorrect configurations.

Configure the **nginx.conf** file in the **conf** directory of Nginx.

1. Find the following configuration:

```
#server {
#    listen       443 ssl;
#    server_name  localhost;
#    ssl_certificate      cert.pem;
#    ssl_certificate_key  cert.key;
#    ssl_session_cache    shared:SSL:1m;
#    ssl_session_timeout  5m;
#    ssl_ciphers  HIGH:!aNULL:!MD5;
#    ssl_prefer_server_ciphers  on;
#    location / {
#        root   html;
#        index  index.html index.htm;
#    }
#}
```

2. Delete comment tags (#) at the beginning of the lines.

```
server {
            listen              443 ssl;
            server_name         localhost;
            ssl_certificate     cert.pem;
            ssl_certificate_key cert.key;
            ssl_session_cache   shared:SSL:1m;
            ssl_session_timeout 5m;
            ssl_ciphers         HIGH:!aNULL:!MD5;
            ssl_prefer_server_ciphers  on;
            location / {
                    root     html;
                    index    index.html index.htm;
                     }
}
```

3. Modify the following parameters according to **Table 2-3**.

```
ssl_certificate          cert/server.crt;
ssl_certificate_key      cert/server.key;
```

The complete configuration is as follows. Modify other parameters based on your needs.

```
server {
       listen       443 ssl; # Set the default HTTPS port to 443. If the default HTTPS port is not configured, Nginx may fail to start.
       server_name  www.domain.com; #Replace www.domain.com with the domain name associated with your certificate.
```



```
    ssl_certificate      cert/server.crt; #Replace cert/server.crt with the path of the certificate file.
    ssl_certificate_key  cert/server.key; #Replace cert/server.key with the path of the private key.
    ssl_session_cache    shared:SSL:1m;
    ssl_session_timeout  5m;
    ssl_ciphers  HIGH:!aNULL:!MD5; #Encryption suite
    ssl_prefer_server_ciphers  on;
    location / {
        root   html; #Site directory
        index  index.html index.htm; #Add attributes.
            }
}
```

NOTICE

Do not directly copy all configuration. Only attributes starting with **ssl** are directly related to the certificate configuration. Modify other parameters based on site requirements.

Table 2-3 Parameters

Parameter	Description
listen	SSL access port number. Set the value to **443**. Set the default HTTPS port to 443. If the default HTTPS port is not configured, Nginx may fail to start.
server_name	Domain name which the certificate is used for. Example: www.domain.com
ssl_certificate	Certificate file **server.crt** Set the value to the path of the **server.crt** file. An example of the path is **cert/server.crt**.
ssl_certificate_key	Private key file **server.key** Set the value to the path of the **server.key** file. An example of the path is **cert/server.key**.

4. Save the configuration file.

Step 4: Verifying the Configuration

Go to the execution directory of Nginx and run the following command:

sbin/nginx -t

If the following information is displayed, the configuration is correct.

```
nginx.conf syntax is ok
nginx.conf test is successful
```

Step 5: Restarting Nginx

Run the following command to restart Nginx to make the configuration take effect:

cd /usr/local/nginx/sbin

./nginx -s reload

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

2.1.1.4 Installing an SSL Certificate on an Apache Server

This section describes how to install an SSL certificate on an Apache 2.4.6 server running CentOS 7. The installation process is similar for other Apache servers. When the certificate is installed, it secures communication between your web server and the client through SSL.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Hosted** status.
- You have downloaded the certificate. For details, see **Downloading an SSL Certificate**.
- You have installed the **mod_ssl.so** module (for enabling SSL) on the Apache server.

Constraints

- Before installing the certificate, enable port 443 on the server where the SSL certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- The domain name to be run on the target server must be the same as the one associated with the certificate. Otherwise, the web browser will display a message indicating that the connection to the domain name is insecure.

Procedure

The installation process is as follows (for Apache 2.4.6 servers running CentOS 7):

Step 1: Obtaining Files → **Step 2: Creating a Directory** → **Step 3: Modifying Configuration Files** → **Step 4: Restarting Apache** → **Verifying the Result**

Step 1: Obtaining Files

Before installing a certificate, obtain the certificate file and password file. Perform the following operations based on the value selected for **CSR** when applying for a certificate:

- If you select **System generated CSR** for **CSR** when applying for a certificate, perform the operations according to the instructions in **System generated CSR**.
- If you select **Upload a CSR** for **CSR** when applying for a certificate, perform the operations according to the instructions in **Upload a CSR**.

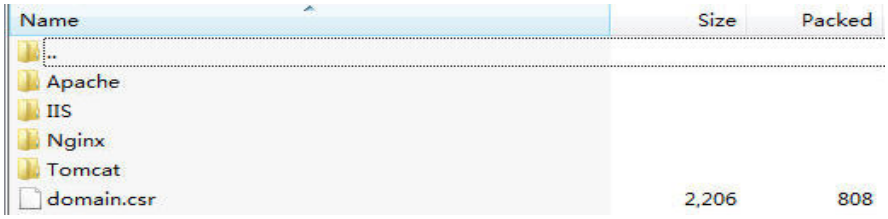
Detailed operations are as follows:

- **System generated CSR**

  a. Decompress the downloaded certificate file on your local PC.

  The downloaded file contains the **Apache**, **IIS**, **Nginx**, and **Tomcat** folders as well as the **domain.csr** file. **Figure 2-4** shows an example.

  Figure 2-4 Decompressing an SSL certificate package on a local computer

  b. Obtain the certificate files ***Certificate ID_Domain name bound to the certificate*_ca.crt** and ***Certificate ID_Domain name bound to the certificate*_server.crt**, and private key file ***Certificate ID_Domain name bound to the certificate*_server.key** from ***Certificate ID_Domain name bound to the certificate*_Apache**.

  - The ***Certificate ID_Domain name bound to the certificate*_ca.crt** file contains a segment of intermediate CA certificate code **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**.
  - The ***Certificate ID_Domain name bound to the certificate*_server.crt** file contains a segment of server certificate code **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**.
  - The ***Certificate ID_Domain name bound to the certificate*_server.key** file contains a segment of private key code **-----BEGIN RSA PRIVATE KEY-----** and **-----END RSA PRIVATE KEY-----**.

- **Upload a CSR**

  a. Decompress the downloaded certificate package to obtain the ***Certificate ID_Domain name bound to the certificate*_server.pem** file.

  The ***Certificate ID_Domain name bound to the certificate*_server.pem** file contains two segments of certificate codes **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**, which are the server certificate and intermediate CA certificate respectively.

  b. Copy the first segment of certificate code (server certificate) in the ***Certificate ID_Domain name bound to the certificate*_server.pem** file and save it as the **server.crt** file.

c. Copy the second segment of certificate code (intermediate CA certificate) in the ***Certificate ID_Domain name bound to the certificate*_server.pem** file and save it as the **ca.crt** file.

d. Place **ca.crt**, **server.crt**, and the **server.key** private key generated during CSR generation in any folder.

Step 2: Creating a Directory

Create a **cert** directory in the Apache installation directory, and copy the **server.key**, **server.crt**, and **ca.crt** files to the **cert** directory.

Step 3: Modifying Configuration Files

NOTICE

Before modifying the configuration file, back up the configuration file. You are advised to deploy the configuration file in the test environment and then configure it on the production environment to avoid service interruptions caused by incorrect configurations.

1. Open the **conf.d/ssl.conf** file in the Apache root directory.
2. Configure the domain name associated with the certificate.

   Find and modify the following parameter:

   ```
   ServerName www.example.com:443
   ```

   The complete configuration is as follows (**www.domain.com** is used as an example):

   ServerName *www.domain.com*:443 #Replace *www.domain.com* with the domain name of your server.
3. Configure the public key for the certificate.

   Find and modify the following parameter:

   ```
   SSLCertificateFile "${SRVROOT}/conf/server.crt"
   ```

   Set the value to the path of the **server.crt** file. An example of the path is **cert/server.crt**.

   The complete configuration is as follows:

   ```
   SSLCertificateFile "cert/server.crt"
   ```
4. Configure the private key for the certificate.

   Find and modify the following parameter:

   ```
   SSLCertificateKeyFile "${SRVROOT}/conf/server.key"
   ```

   Set the value to the path of the **server.key** file. An example of the path is **cert/server.key**.

   The complete configuration is as follows:

   ```
   SSLCertificateKeyFile "cert/server.key"
   ```
5. Configure the certificate chain.

   Find and modify the following parameter:

   ```
   #SSLCertificateChainFile "${SRVROOT}/conf/server-ca.crt"
   ```

   Delete the comment tag **#** at the beginning of the line. Set this parameter to the path of the **ca.crt** file. An example of the path is **cert/ca.crt**.

The complete configuration is as follows:

```
SSLCertificateChainFile "cert/ca.crt"
```

6. Save the **ssl.conf** file and exit.

Step 4: Restarting Apache

Restart the Apache service for the configuration to take effect:

1. Run the **service named stop** command to stop the Apache server.
2. Run the **service httpd start** command to start the Apache server.

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

2.1.1.5 Installing an SSL Certificate on an IIS Server

This topic describes how to install an SSL certificate on an IIS server. When the certificate is installed, it secures communication between your server and the client through SSL.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Hosted** status.
- You have downloaded the certificate. For details, see **Downloading an SSL Certificate**.

Constraints

- Before installing the certificate, enable port 443 on the server where the SSL certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- The domain name to be run on the target server must be the same as the one associated with the certificate. Otherwise, the web browser will display a message indicating that the connection to the domain name is insecure.

Procedure

To install an SSL certificate on an IIS server, perform the following steps:

Step 1: Obtaining Files → **Step 2: Configuring IIS** → **Verifying the Result**

Step 1: Obtaining Files

Before installing a certificate, obtain the certificate file and password file. Perform the following operations based on the value selected for **CSR** when applying for a certificate:

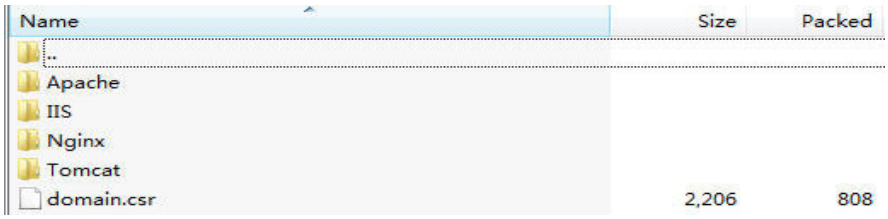
- If you select **System generated CSR** for **CSR** when applying for a certificate, perform the operations according to the instructions in **System generated CSR**.
- If you select **Upload a CSR** for **CSR** when applying for a certificate, perform the operations according to the instructions in **Upload a CSR**.

Detailed operations are as follows:

- **System generated CSR**
  a. Decompress the downloaded certificate file on your local PC.

     The downloaded file contains the **Apache**, **IIS**, **Nginx**, and **Tomcat** folders as well as the **domain.csr** file. **Figure 2-5** shows an example.

     Figure 2-5 Decompressing an SSL certificate package on a local computer

  b. Obtain the SSL certificate file ***Certificate ID_Domain name bound to the certificate*_server.pfx** and password file ***Certificate ID_Domain name bound to the certificate*_keystorePass.txt** from ***Certificate ID_Domain name bound to the certificate*_IIS**.
- **Upload a CSR**
  a. Decompress the downloaded certificate package to obtain the ***Certificate ID_Domain name bound to the certificate*_server.pem** file.

     The ***Certificate ID_Domain name bound to the certificate*_server.pem** file contains two segments of certificate codes **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**, which are the server certificate and intermediate CA certificate respectively.

  b. Use OpenSSL to convert the PEM certificate into a PFX certificate and obtain the **server.pfx** file.

     i. Save the PEM certificate and the private key **server.key** generated during CSR generation to the **bin** directory in the OpenSSL installation directory.

     ii. In the **bin** directory of the OpenSSL installation directory, run the following command to convert the PEM certificate into a PFX certificate and press **Enter**:

     openssl pkcs12 -export -out server.pfx -inkey server.key -in *Certificate ID_Domain name bound to the certificate*_**server.pem**

     Information similar to the following is displayed.


```
Enter Export Password:
```

iii. Enter the password of the PFX certificate and press **Enter**.

The password is user-defined. Set it as required.

Information similar to the following is displayed.

```
Verifying - Enter Export Password:
```

NOTE

Record the password of the PFX certificate. The password of the JKS certificate must be the same as that of the PFX certificate. Otherwise, the IIS service may fail to start.

To improve password security, a password must:

- Consist of 8 to 32 characters.
- Contain at least three types of the following characters: uppercase letters, lowercase letters, digits, and special characters ~`!@#$%^&*()_+|{}:"<>?-=\[];',./

iv. Re-enter the password of the PFX certificate and press **Enter**.

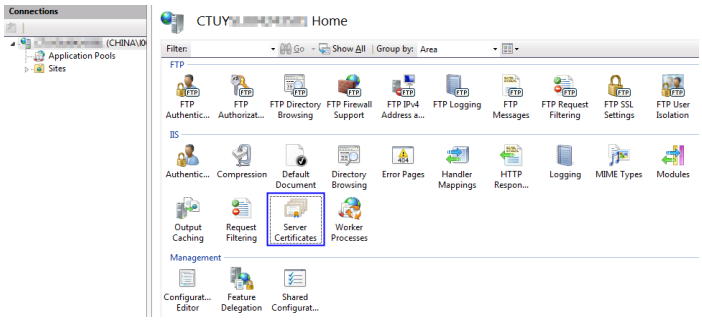
If no error information is displayed, the **server.pfx** file has been generated in the OpenSSL installation directory.

v. Create a **keystorePass.txt** file in the OpenSSL installation directory and save the password of the PFX certificate in the file.

Step 2: Configuring IIS

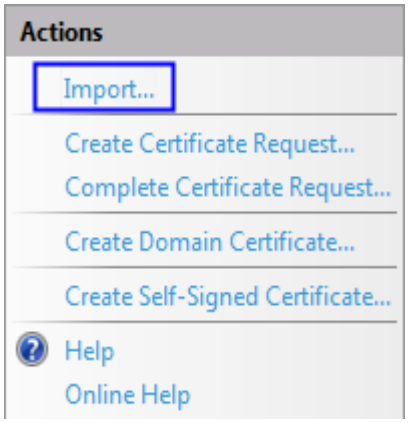
1. Install IIS as instructed by IIS guides.
2. Open the IIS management console, double-click **Server Certificates**.

Figure 2-6 Double-clicking Server Certificates



3. In the displayed dialog box, click **Import**.

Figure 2-7 Import

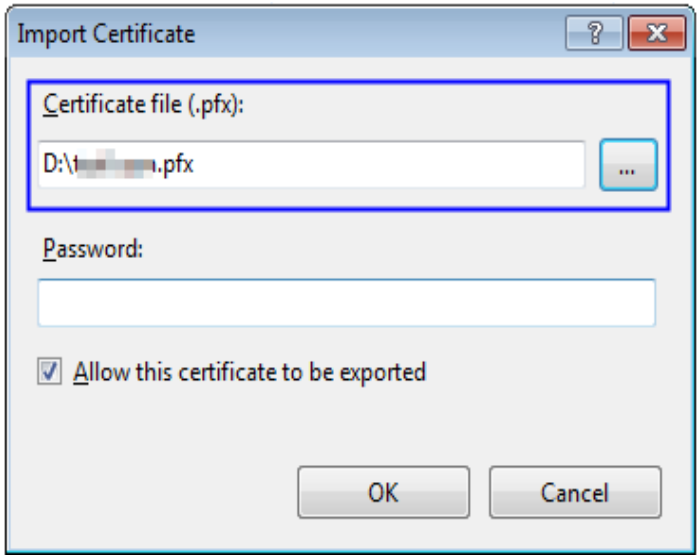


4. Import the **server.pfx** certificate file. Then click **OK**.

   NOTE

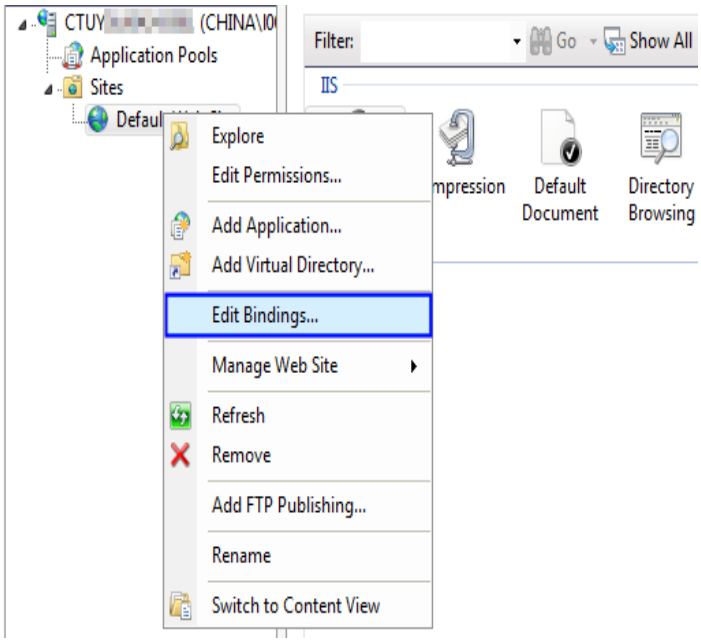
   In the **Password** box, enter the password provided in the **keystorePass.txt** file.

Figure 2-8 Importing a PFX certificate file



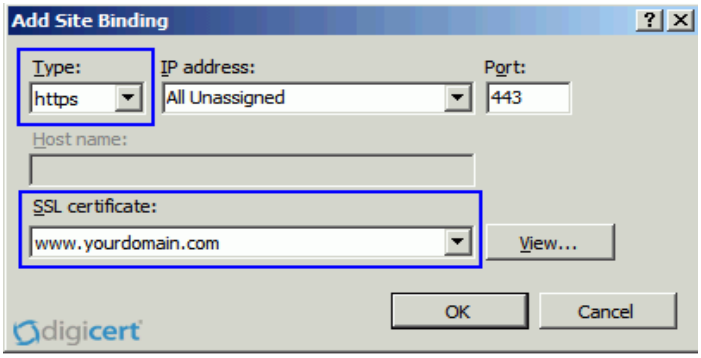
5. Right-click the target site (the default site is used as an example). Choose **Edit Bindings** from the shortcut menu.

Figure 2-9 Choosing Edit Bindings



6. In the dialog box that is displayed, click **Add**. Then enter the following information.

Figure 2-10 Binding a website



   - **Type**: Select **https**.
   - **Port**: Retain the default port **443**.
   - **SSL certificate**: Select the certificate imported in **4**.
7. Click **OK**.

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

2.1.2 Deploying an SSL Certificate to Other Cloud Products

2.1.2.1 Deploying an SSL Certificate to WAF

When an SSL certificate is issued, you can deploy it to Web Application Firewall (WAF) on cloud in just a few clicks. With SSL certificates, data access to your website protected with WAF is more secure.

Prerequisites

- You have enabled WAF, routed your website domain name to WAF, and configured an SSL certificate for the domain name in WAF.
- If you have not purchased WAF or the domain name you want to use the certificate for has not been added to WAF, deploying the certificate to WAF may fail.
- You have uploaded the SSL certificate issued by another platform to CCM and the certificate is in **Hosted** status.

Constraints

- Currently, you can use SCM to quickly deploy an SSL certificate to WAF in the **default** enterprise project only. For other enterprise projects, download the certificates first, upload them to WAF, and then deploy them in WAF.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 In the row containing the target certificate, click **Deploy** in the **Operation** column.

Step 5 On the displayed page, select **WAF** in the **Deployment Details** area.

Step 6 Click on the right of the **Region** drop-down list and select the region where you want to deploy the certificate.

Step 7 Select the domain name you want to deploy the certificate for and click **Deploy** in the **Operation** column.

To deploy the certificate for multiple domain names, select all the domain names you want and click **Deploy** above the domain name list.

Step 8 In the displayed confirmation dialog box, click **Confirm**.

When the certificate is deployed, the **Deployment** column for the domain name reads **Deployed**.

----End

2.1.2.2 Deploying an SSL Certificate to ELB

When an SSL certificate is issued, you can deploy it to Elastic Load Balance (ELB) in just a few clicks. With SSL certificates, data access to your website that uses ELB is more secure.

Prerequisites

- You have enabled Elastic Load Balance (ELB) as required below, added your website domain name to ELB, and configured an SSL certificate for the website in ELB.

  If you have not purchased ELB or the domain name you want to use the certificate for has not been added to ELB, deploying the certificate to ELB may fail.
- You have uploaded the SSL certificate issued by another platform to CCM and the certificate is in **Hosted** status.

Constraints

- You have configured the original certificate in ELB. This means the certificate that is being used for ELB and you want to update in SCM must have been configured in ELB at the very beginning. Then, you can quickly update it in SCM. For details, see "Managing Certificates" in *Elastic Load Balance User Guide*.
- You can use SCM to update the certificate deployed on listeners in ELB. If you update an SSL certificate in SCM, the certificate content and private keys are updated in ELB accordingly. ELB then updates the certificate content and private keys on all listeners where the certificate is deployed for.
- To update a certificate used for ELB in SCM, domain names must be associated with the certificate in ELB.
- If an ELB certificate is used for multiple domain names, ensure that the new certificate you want to update in SCM for ELB must match with those domain names. If they do not match, the domain names in the new certificate will overwrite the ones in the original certificate after the update.

  For example, the primary domain name and additional domain name of the new certificate are example01.com and example02.com, respectively, and the domain names associated with the original certificate in ELB are example01.com and example03.com. When you update the certificate in SCM, the domain names associated with the certificate in ELB are updated to example01.com and example02.com.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 In the row containing the target certificate, click **Deploy** in the **Operation** column.

Step 5 On the displayed page, select **ELB** in the **Deployment Details** area.

Step 6 Click on the right of the **Region** drop-down list and select the region where you want to deploy the certificate.

Step 7 Select the domain name you want to update the certificate for and click **Update Certificate** in the **Operation** column.

To update the certificates for multiple domain names, select all the target domain names and click **Batch Update** above the domain name list.

Step 8 In the displayed confirmation dialog box, click **Confirm**.

If a message indicating that the certificate is updated successfully is displayed, the SSL certificate is updated for ELB.

----End

2.2 Managing SSL Certificates

2.2.1 Uploading an External Certificate to SCM

You can upload your SSL certificates (SSL certificates that have been purchased and issued on other platforms) to the CCM service for centralized management.

This topic describes how to upload a local (external) SSL certificate onto CCM.

Prerequisites

You have prepared the following files to be uploaded:

- Certificate file in PEM encoding format (the file name extension is PEM or CRT).
- Certificate private key in PEM encoding format (the file name extension is KEY).

NOTE

- Currently, only certificates in PEM format can be uploaded to CCM. Certificates in other formats can be uploaded only after they are converted to certificates in the PEM format.
- The private key you want to upload cannot be protected by a password.
- For uploaded certificates, SCM reminds you of certificate expiration 30 days before the certificates expire.

Constraints

- Expired certificates cannot be uploaded.
- A certificate whose certificate chain length is 1 cannot be uploaded. That is, the certificate to be uploaded must contain a certificate chain and cannot be a single certificate.
- The CN of the certificate to be uploaded must be in DNS or IP address format.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 Click **Upload Certificate**.

Step 5 In the **Hosted Certificates** dialog box, enter the certificate information.

Table 2-4 Parameters for uploading an international standard certificate

Parameter	Description
Certificate Name	A certificate name you specify.
Enterprise Project	Add the uploaded SSL certificate to the corresponding enterprise project.
Certificate File	Open the PEM file in the certificate to be uploaded as a text file and copy the certificate content in the file to this text box. Note that you need to upload a combined certificate file that contains both the server certificate content and certificate chain content into this field. The content of the certificate chain should be pasted right below the content of the server certificate.
Private Key	Use a text editor to open the KEY file in the certificate you want to upload and copy the private key content to this text box.

NOTE

- The uploaded certificate and key must correspond to each other.
- Ensure that the private key is not protected by a password.

Step 6 Click **Submit** to upload the certificate.

When the certificate is uploaded successfully, a certificate in the **Hosted** state is added to the certificate list.

----End

2.2.2 Pushing an SSL Certificate to Other Cloud Services

After an SSL certificate is issued, you can push it to other services, such as Web Application Firewall (WAF) and Elastic Load Balance (ELB) in just few clicks. In this manner, data access through the cloud services is more secure.

Prerequisites

The certificate is in the **Hosted** status.

Constraints

- If you have not purchased a given cloud service or the service is not available for the domain name associated with your certificate, do not push the certificate to it because the process may fail.
- A certificate can only be pushed to a product once in CCM. If you push a certificate that has been pushed or uploaded to a cloud product, a push failure will occur.

Procedure

Step 1 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 2 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 3 On the SCM page, locate the row that contains the target certificate, and click **MorePush** in the **Operation** column. The certificate push details page is displayed on the right.

Step 4 Select the cloud service you wish to push the certificate to.

Step 5 Set the target region.

Click on the right of the target project and select the target region. You can select up to 10 regions.

Step 6 Click **Push Certificate** at the lower right corner of the page.

If a message indicating that the certificate is successfully pushed is displayed, the SSL certificate is successfully pushed to the target service.

You need to further configure the certificate on the console of the service to enable HTTPS for it.

Step 7 Check whether you need to immediately access the console of the target service to configure the certificate.

- If yes, click **Configure Now**. The management page of the target service is displayed. Configure the certificate:
- If no, click **Continue Pushing** or  in the upper right corner of the page. The certificate push page or SSL certificate management page is displayed.

  You can access the console of the target service for certificate management.

You can view the latest 10 push records on the certificate push page.

----End

Follow-up Operations

You can manage pushed certificates on the console of the corresponding service.

If you have any questions during the configuration, refer to the corresponding service documentation or consult the corresponding service personnel.

- ELB: If HTTPS data transmission encryption is required, you need to associate a certificate when creating an HTTPS listener. If you choose to push the certificate to ELB in one click, you can select the pushed certificate in ELB. Otherwise, you need to manually upload the certificate.

  Generally, only server certificates need to be configured to authenticate servers for HTTPS-based business. For some key businesses, such as bank payment, two-way authentication is required for enhanced business security.
- WAF: You need to configure a certificate when adding a domain to WAF if HTTPS is used for communications between the client and WAF. If you choose to push the certificate to WAF in one click, you can select the pushed certificate in WAF. Otherwise, you need to manually upload the certificate.

  If a certificate has been configured in WAF, you only need to update the certificate.

2.2.3 Adding an SSL Certificate to an Enterprise Project

You can use enterprise projects to more efficiently manage cloud resources and project members. For more details, see *Enterprise Management User Guide*.

This topic describes how to add an SSL certificate to an enterprise project.

Prerequisites

- An enterprise project has been created.
- The account used to purchase the certificate has the **EPSFullAccess** permission.

  NOTE

  EPSFullAccess: All EPS permissions.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 On the SCM page, locate the row that contains the target certificate and choose **More** > **Add to Project** in the **Operation** column.

Step 5 Select an enterprise project.

Step 6 Click **Submit**.

----End

2.3 Managing Tags

2.3.1 Overview

Scenario

Tags can be used to identify SSL certificates. You can use tags to group certificates by usage, owner, or environment and manage them centrally.

You can add a tag when purchasing a certificate or add a tag on the certificate details page after the purchase.

Tag Naming Rules

- Each tag consists of a key-value pair.
- A maximum of 20 tags can be added for an SSL certificate.
- For each certificate, a tag key must be unique and can have only one tag value.
- A tag consists of a tag key and a tag value. The naming rules are listed in **Table 2-5**.

Table 2-5 Tag parameters

Parameter	Rule	Example
Tag key	• This parameter is mandatory. • An SSL certificate can have only one tag key. • The value can contain a maximum of 128 characters. • The value cannot start or end with a space. • The value cannot start with _**sys**_. • The following character types are allowed: – Chinese – English – Digit – Space – Special characters: _.:=+-@	cost
Tag value	• This tag value can be left blank. • The value can contain a maximum of 255 characters. • The value cannot start or end with a space. • The following character types are allowed: – Chinese – English – Digit – Space – Special characters: _.:=+-@	100

2.3.2 Creating a Tag

This topic describes how to add a tag to an SSL certificate.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 Click the name of the target SSL certificate to go to its details page.

Step 5 Click the **Tags** tab to go to the tag management page.

Step 6 Click **Edit Tag**. In the displayed **Edit Tag** page, click **Add Tag**. In the text box, specify **Tag key** and **Tag value**.

Step 7 Click to complete.

----End

2.3.3 Searching for SSL Certificates by Tag

This section describes how to search for an SSL certificate by tag in a project on the SCM console.

Prerequisites

A tag has been added.

Constraints

At most 20 tags can be added for one search. If multiple tags are added, SSL certificates that meet all search criteria will be displayed.

Procedure

Step 1 Log in to the management console.

Step 2 Click in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 Click the search box and enter the tag key and tag value to search for the resource. SSL certificates that meet the search criteria are displayed

NOTE

- At most 20 tags can be added for one search. If multiple tags are added, SSL certificates that meet all search criteria will be displayed.
- If you want to delete an added tag from the search criteria, click ✕ next to the tag.

----End

2.3.4 Editing a Tag Value

This section describes how to edit an SSL certificate tag.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 Click the name of the target SSL certificate to go to its details page.

Step 5 Click **Tags** to go to the tag management page.

Step 6 Click **Edit Tag**. The **Edit Tag** page is displayed on the right. Edit the tag value and click **OK**.

----End

2.3.5 Deleting a Tag

This section describes how to delete an SSL certificate tag.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **SSL Certificate Manager** > **SSL Certificates**.

Step 4 Click the name of the target SSL certificate to go to its details page.

Step 5 Click **Tags** to go to the tag management page.

Step 6 Click **Edit Tag**. In the displayed dialog box, locate the row that contains the target tag, click **Delete**, and then click **OK**.

----End

3 Private Certificate Authority (PCA) User Guide

3.1 Overview of Private Certificate Application

Cloud Certificate Manager (CCM) is a private CA and certificate management platform. You can use CCM to set up a complete CA hierarchy and use it to issue and manage private certificates for your organization. It is used to authenticate application identities and encrypt and decrypt data within your organization.

Certificates issued by a private CA are trusted only within your organization, but not the Internet.

For details, see **Figure 3-1** and **Table 3-1**.

Figure 3-1 Private certificate application procedure

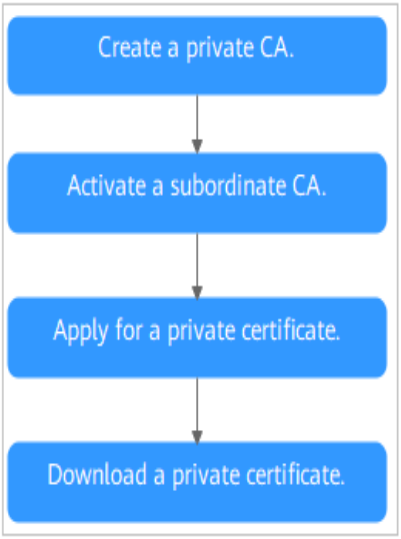


Table 3-1 Application procedure

Step	Operation	Description
1	**Creating a Private CA**	Create a private CA as required. If this is your first time creating a private CA, you must create a root CA. You can create multiple subordinate CAs under the existing root CA.
2	**Activating a Private CA**	A private root CA can be used to issue private certificates once it is created. A private subordinate CA must be activated before it is used to issue certificates.
3	**Applying for a Private Certificate**	Apply for a private certificate with the activated private CA.
4	**Downloading a Private Certificate**	After the application is approved, you can download the private certificate and install it on the server.

3.2 Private CA Management

3.2.1 Creating a Private CA

CCM helps you set up an internal CA for your organization with low costs and use it to issue certificates with ease.

This topic describes how to create a private root CA and subordinate CA.

Overview

- Private CAs are classified into root CAs and subordinate CAs (intermediate CAs). A subordinate CA belongs to a root CA. A root CA can have multiple subordinate CAs.
- If this is your first time creating a private CA, you must create a root CA.
- A maximum of 100 CAs can be created for each user. Private CAs in the pending deletion state are also counted in the private CA quota until the private CAs are deleted.

Prerequisites

The account for creating a private CA has the **PCA FullAccess** permission.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 In the upper right corner of the private CA list, click **Create CA** to switch to the **Create CA** page.

Step 5 Configure the CA information.

You need to specify the basic information, distinguished name, **enterprise project**, and certificate revocation configuration.

1. Configure the basic information. **Table 3-2** describes the parameters.

Table 3-2 Basic information parameters

Parameter	Description	Example Value
CA Type	Indicates the type of the CA to be created. The values can be: – **Root CA**: Select this option if you want to create a CA hierarchy. **NOTE** If you create a private CA for the first time, you must create a root CA. – **Subordinate CA**: Select this option if you want to add a layer to the existing CA hierarchy.	Root CA
Key Algorithm	Indicates the key algorithm. The values can be: – **RSA2048** – RSA3072 – **RSA4096** – **EC256** – **EC384**	**RSA2048**

Parameter	Description	Example Value
Signature Algorithm	This parameter is displayed when **CA Type** is set to **Root CA**. You can select any of the following hash algorithms: – **SHA256** – **SHA384** – **SHA512** – SHA256_PSS – SHA384_PSS – SHA512_PSS	**SHA256**
Validity Period	This parameter is displayed when **CA Type** is set to **Root CA**. Indicates the validity period of a private certificate issuer. The longest period is 30 years.	3 years

2. Configure the certificated distinguished name. **Table 3-3** describes the parameters.

Table 3-3 Parameters

Parameter	Description	Example Value
Common Name	Indicates the CA name.	N/A
Country/Region	Indicates the country or region where your organization belongs. Enter the two-letter code of the country or region.	MA
State/Province	Indicates the name of the province or state where your organization is located.	Kuala Lumpur
Locality	Indicates the name of the city where your organization is located.	Kuala Lumpur
Organization	The legal name of your company.	N/A
Organizational Unit	Indicates the department name.	Cloud Dept

3. Select an enterprise project from the **Enterprise Project** drop-down list.

   This option is only available if you have logged in using an enterprise account, or if you have enabled enterprise projects.

To use this function, see section "Enabling the Enterprise Center" in the Enterprise Management User Guide. You can use an enterprise project to centrally manage your cloud resources and members by project.

NOTE

Value **default** indicates the default enterprise project. Resources that are not allocated to any enterprise projects under your account are displayed in the default enterprise project.

4. (Optional) Configure certificate revocation.

   If you want to publish the certificate revocation list (CRL) for a private CA, you can configure parameters in this pane.

   If no configuration is required, skip this step.

   Configure certificate revocation information. **Table 3-4** describes the parameters.

Table 3-4 Certificate revocation parameters

Parameter	Description
OBS Authorization	Whether to authorize CCM to access your OBS bucket and upload the CRL file. If you want to authorize, click **Authorize Now** and complete the authorization as prompted. If you want to cancel the authorization, go to the IAM console to delete the PCAAccessPrivateOBS agency from the agency list. After the permission has been granted, follow-up operations do not require the permission to be granted again.
Enable CRL publishing	Indicates whether to enable CRL publishing.
OBS Bucket	Select an existing OBS bucket or click **Create OBS Bucket** to create an OBS bucket.
CRL Update Period	Indicates the CRL update period. PCA will generate a new CRL at the specified time. You can set the period to an integer between 7 and 30. If you do not specify a value, it is set to 7 days by default.

5. (Optional) Click **Add Tag** and configure a tag for the private CA.

   Tags can be used to identify private CA. You can use tags to group private CAs by usage, owner, or environment and manage them centrally. For details, see **Overview**.

Step 6 Click **Next** to enter the confirmation page.

Step 7 After confirming the information about the private CA, click **Confirm and Create**.

If you create a root CA, the root CA is automatically activated after being created. If you create a subordinate CA, you need to manually activate it.

After you create a subordinate CA, click **Activate Now** or **Activate Later** to determine whether to activate the subordinate CA immediately.

----End

Follow-up Procedure

After a root CA is created, it can be used to issue private certificates. For details about how to apply for a private certificate, see **Applying for a Private Certificate**.

After a subordinate CA is created, you need to install a certificate and activate the CA. For details, see **Activating a Private CA**.

3.2.2 Activating a Private CA

You need to active a subordinate CA after it is created. A subordinate private CA takes effect and can be used to issue private certificates only after it is activated.

This topic describes how to activate a subordinate CA. You can use either an internal private CA or external private CA to activate the subordinate CA.

- Internal private CA: Use a private CA in CCM to activate a subordinate CA.
- External private CA: Use a private CA from a third party to activate a subordinate CA.

Prerequisites

- You have created a private subordinate CA. For details, see **Creating a Private CA**.
- The subordinate CA is in the **Pending activation** state.

Activating a Subordinate Private CA with an Internal Private CA

Step 1 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 2 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 3 Locate the row of the private CA to be activated, click **Activate** in the **Operation** column. The details page is displayed on the right. Configure the required information.

1. Configure **Issued From**.

   Select **Internal private CA**.
2. Configure the required parameters.

Table 3-5 Parameters

Parameter	Description
Common Name	Indicates the name of the CA. The CA can be a root CA or a subordinate CA. After you select the CA, the system automatically displays the type and ID of the CA.
Signature Algorithm	Indicates the signature algorithm. The values can be: – **SHA256** – **SHA384** – **SHA512** – SHA256_PSS – SHA384_PSS – SHA512_PSS
Validity Period	Indicates the validity period of a private CA. The longest period is 20 years.
Path Length	The path length of the subordinate CA. The path length controls how many layers of subordinate CAs the current subordinate CA can issue. (The last layer of the certificate chain is a private certificate). **NOTE** A certificate chain is made up of root CAs, subordinate CAs, and private certificates in a fixed sequence to validate the trust of a certificate at a lower layer.

Step 4 Confirm the configuration and click **OK**.

----End

Activating a Subordinate Private CA with an External Private CA

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Locate the row of the subordinate CA and click **Activate** in the **Operation** column. In the **Install CA Certificate and Activate CA** page, configure the required parameters.

1. Configure **Issued From**.

   Select **External private CA**.

2. Export the CSR.

   In the **CA CSR** pane, click **Export File**.

   The PEM CSR is exported to a file and is signed by a parent CA.

3. Use the external CA to issue a certificate.

   Use your private CA to issue a certificate for the subordinate private CA you want to activate.

4. Import the certificate.

   Import the certificate and certificate chain in the **Import the Certificate Issued by an External CA** pane.

   Table 3-6 Parameter descriptions

Parameter	Description
Certificate	Open the PEM file in the certificate to be uploaded as a text file with the extension **.pem** and copy the certificate content to this text box.
Certificate Chain	Open the PEM file in the certificate to be uploaded as a text file with the extension **.pem** and copy the certificate chain to this text box.

Step 5 Confirm the configuration and click **OK**.

If the status of the subordinate CA changes to **Activated**, the subordinate CA has been activated.

----End

Follow-up Procedure

After a subordinate CA is activated, it can be used to issue private certificates. For details about how to apply for a private certificate, see **Applying for a Private Certificate**.

3.2.3 Viewing Private CA Details

This topic describes how to view the private CA information, including **Common Name**, **Organizational Unit**, **Type**, and **Status**.

Prerequisites

A private CA has been created. For details, see **Creating a Private CA**.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 View private CA information in the private CA list. **Table 3-7** describes the parameters.

Table 3-7 CA parameter description

Parameter	Description
Common Name	Indicates the user-defined CA name.
Status	Indicates the private CA status. The value can be: • **Pending activation**: The private CA is to be activated. • **Activated**: The private CA is activated. • **Disabled**: The private CA is disabled. • **Pending deletion**: The private CA is to be deleted. • **Expired**: The private CA is expired. • Revoked: The private sub-CA has been revoked.
Type	Indicates the private CA type. The value can be: • **Root CA**: The private CA is a root CA and can be used to issue subordinate CAs. • **Subordinate CA**: The private CA is a subordinate CA.
Key Algorithm	Key algorithm of the private CA.
Organizational Unit	Indicates the name of the organizational unit to which the private CA belongs.
Issued By	Indicates the name of the CA that issues the private CA.
Expiration Time	Indicates the time when a private CA expires.
Enterprise Project	Enterprise project to which the private CA belongs.
Operation	You can activate, enable, or disable a CA.

Step 5 Click the name of a private CA to view its details, CA certificate, CRL configuration, and tag information.

You can click **Edit Tag** on the tags page. If you want to use a tag to identify multiple types of cloud resources, you are advised to create predefined tags in TMS.

----End

3.2.4 Configuring a CRL

If you want to publish the certificate revocation list (CRL) for a private CA, you can enable CRL configuration.

This topic walks you through how to enable or disable CRL configuration.

Prerequisites

The private CA for which you want to configure a CRL is in the **Activated** or **Disabled** state.

Enabling CRL Configuration

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 Click the name of a private CA. The private CA details page is displayed.

Step 4 On the private CA details page, click the **CRL Configuration** tab and configure certificate revocation. Configure certificate revocation parameters by referring to **Certificate revocation parameters**.

Table 3-8 Certificate revocation parameters

Parameter	Description
OBS Authorization	Whether to authorize CCM to access your OBS bucket and upload the CRL file. If you want to authorize, click **Authorize Now** and complete the authorization as prompted. If you want to cancel the authorization, go to the IAM console to delete the PCAAccessPrivateOBS agency from the agency list. After the permission has been granted, follow-up operations do not require the permission to be granted again.
Enable CRL publishing	Indicates whether to enable CRL publishing.
OBS Bucket	Select an existing OBS bucket or click **Create OBS Bucket** to create an OBS bucket.
CRL Update Period	Indicates the CRL update period. PCA will generate a new CRL at the specified time. You can set the period to an integer between 7 and 30. If you do not specify a value, it is set to 7 days by default.

Step 5 Click **Enable** to enable the CRL. If the system displays a message indicating that the CRL configuration is enabled, the CRL configuration has been enabled.

----End

Disabling CRL Configuration

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 Click the name of a private CA. The private CA details page is displayed.

Step 4 On the private CA details page, click the **CRL Configuration** tab and click **Disable**. If the system displays a message indicating that the CRL configuration is disabled, the CRL configuration has been disabled.

----End

3.2.5 Exporting a Private CA Certificate

After a private CA is created and activated, you can export the private CA certificate.

If your web services are accessible through browsers, add the root certificate to your browser trust list and install the private certificate issued by the root CA on your web server to implement HTTPS communications between the client and the server.

If your web services are accessible through a client like Java, manually install the root certificate on the client to ensure that the client can validate the encrypted information on the server.

This topic walks you through how to export a private CA certificate.

Prerequisites

The private CA for which the certificate is to be exported is in the **Activated** state.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Locate the row of the desired private CA and click **Export CA Certificate** in the **Operation** column.

Step 5 In the displayed dialog box, click **OK**.

When you click **OK**, CCM will use the download tool provided by the browser to download the private CA certificate to the specified local directory.

Now, you will obtain a private CA certificate file named ***root CA name_certificate*.pem**.

----End

3.2.6 Disabling a Private CA

If you no longer need a private CA to issue certificates, you can disable the private CA.

If a private CA is disabled, it cannot be used to issue any private certificates. If you want to use this private CA to issue certificates again, it must be enabled first. For details, see **Enabling a Private CA**.

This topic describes how to disable a private CA.

Prerequisites

The private CA to be disabled is in the **Activated** or **Expired** state.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Locate the row of the desired private CA and click **Disable** in the **Operation** column.

Step 5 In the displayed dialog box, enter **DISABLE** and click **OK**.

When "CA xxx disabled." is displayed in the upper part of the page, and the private CA status changes to **Disabled**, the private CA is disabled successfully.

----End

3.2.7 Enabling a Private CA

If you need to use a disabled private CA to issue certificates, you can restore the certificate to the activated state.

The following walks you through how to enable a private CA so that you can quickly restore a disabled private CA to the activated or expired state.

Prerequisites

The private CA to be enabled is in the **Disabled** state. For details about how to disable a private CA, see **Disabling a Private CA**.

Procedure

Step 1 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 2 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 3 Locate the row of the desired private CA and click **Enable** in the **Operation** column.

When "CA xxx enabled." is displayed in the upper part of the page, and the private CA status changes to **Disabled**, the private CA is disabled successfully.

----End

3.2.8 Deleting a Private CA

Before deleting a private CA, ensure that it is not in use and will not be used.

If deletion is scheduled for a private CA in the **Disabled** or **Expired** state, the deletion will take effect after a waiting period of 7 to 30 days. If deletion is scheduled for a private CA in the **Pending activation** state, the deletion will take effect immediately. Before the specified deletion date, you can cancel the deletion if you want to use the private CA again. If the specified deletion period expires, the private CA will be permanently deleted. Exercise caution when performing this operation.

Prerequisites

The private CA you want to delete is in the **Disabled**, **To be activated**, **Revoked**, or **Expired** state.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Locate the row of the private CA to be deleted and click **Delete** in the **Operation** column.

Step 5 The operations vary according to the private CA status.

- Private CA in the revoked state

  In the displayed dialog box, enter **DELETE** in the text box.

- Private CA in the **Pending activation** state

  In the displayed dialog box, enter **DELETE** in the text box.

- Private CA in the **Disabled** or **Expired** state

  In the dialog box that is displayed, enter **DELETE** in the text box and configure the waiting period.

Step 6 Click **OK**. If **CA xxx deleted.** is displayed in the upper part of the page, the private CA is successfully deleted.

----End

3.2.9 Canceling the Deletion of a Private CA

This topic describes how to cancel the scheduled deletion of one or more private CAs prior to the real deletion. After the cancellation, the private CA is in the **Disabled** state.

Prerequisites

The private CA for which you want to cancel the scheduled deletion is in **Pending deletion** status.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Locate the row of the desired private CA and click **Cancel CA Deletion** in the **Operation** column.

Step 5 In the displayed dialog box, click **OK**.

If message "Deletion of CA xxx cancelled successfully." is displayed in the upper right corner of the page and the private CA status changes to **Disabled**, the deletion of the private CA is cancelled successfully.

After the deletion is canceled, if you want to use the private CA to issue certificates, you need to enable the private CA. For details, see **Enabling a Private CA**.

----End

3.3 Private Certificate Management

3.3.1 Applying for a Private Certificate

After you create and activate a private CA, you can apply for private certificates from the private CA and use them for identity authentication, data encryption, and data decryption of internal applications.

This topic walks you through how to apply for a private certificate. You can apply for a maximum of 100,000 certificates.

Prerequisites

You have created and activated a private CA. For details, see **Creating a Private CA** and **Activating a Private CA**.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed. In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 3 In the upper right corner of the private certificate list, click **Apply for Certificate**.

1. Select the CSR file generation method.

   Table 3-9 Certificate signing request (CSR)

Parameter	Description
System generated CSR	The system automatically generates a certificate private key. Once the certificate is issued, you can download your certificate and private key on the certificate management page.
Upload a CSR	You can use an existing CSR. The procedure is as follows: 1. You need to manually generate a CSR file and paste the content of the CSR file into the text box. 2. Click **Parse**.

   NOTE

   - To obtain a certificate, a CSR file needs to be submitted to the CA for review. A CSR file contains a public key and a distinguished name (DN). Typically, a CSR file is generated by a web server, and a pair of public and private keys are created along with the CSR file.
   - You are advised to select **System generated CSR** to avoid approval failure caused by incorrect content.
   - A private key file will be generated when the CSR file is generated manually. Keep and back up your private key properly. If a private key is lost, the corresponding certificate becomes invalid.
   - SCM has strict requirements on the key type and length of the CSR file. The key must be RSA and it must be 2,048 bits long.

2. Configure certificate details.

   Perform this step only when you select **System generated CSR** for **CSR**.

   Common Name: You can customize the name of the private certificate.

3. Click ^ on the right of **Advanced Configuration**.

   Perform this step only when you select **System generated CSR** for **CSR**.

Table 3-10 Advanced settings

Parameter	Description	Example Value
Key Algorithm	**Key Algorithm**: Select the key algorithm and key size for the private certificate. The options are as follows: – **RSA2048** – **RSA3072** – **RSA4096** – **EC256** – **EC384** – **ED25519**	RSA2048
Signature Algorithm	Select the signature hash algorithm of the private certificate. The options are as follows: – **SHA256** – **SHA384** – **SHA512** – **SHA256_PSS** – **SHA384_PSS** – **SHA512_PSS**	SHA256

Parameter	Description	Example Value
Key Usage	Select the key usage of the certificate. You can select more than one option. – **digitalSignature**: The key is used as a digital signature. – **nonRepudiation**: The key can be used for non-repudiation. – **keyEncipherment**: The key can be used for key encryption. – **dataEncipherment**: The key can be used for data encryption. – **keyAgreement**: The key can be used as a key-agreement protocol. – **keyCertSign**: The key can be used to issue certificates. – **cRLSign**: The key can be used for signing blacklists. – **encipherOnly**: The key can be used for encryption only. – **decipherOnly**: The key can be used for decryption only.	digitalSignature
Enhanced Key Usage	Select the enhanced key usage for the certificate. You can select more than one option. – **Server identity authentication** – **Client identity authentication** – **Code signature** – **Secure email** – **Timestamp**	Server identity authentication
Customized Extension Field	Enter customized information.	None

Parameter	Description	Example Value
Configure Certificate AltName	This field is optional. If you want to use the private certificate to multiple subjects, you can add more AltName records. You can configure **IP address**, **DNS**, **Email**, or **URI** for **AltName**. When you configure **AltName**, enter the value according to the value type you select. – **IP address**: Enter an IP address. – **DNS**: Enter the domain name. – **Email**: Enter an email address. – **URI**: Enter the network address. A maximum of 20 AltName records can be configured.	None

4. Select a CA.

Table 3-11 Parameters for selecting a CA

Parameter	Description
Common Name	Select a common name of the private CA you want.
Type	The CA type is autofilled after you specify **Common Name**.
CA ID	The CA ID is autofilled after you specify **Common Name**.
Validity Period	Configure the validity period of the private certificate. **NOTE** – You can customize the validity period of a private certificate. The validity period cannot outlive the validity period of the activated private CA. – A private CA can be valid for up to 20 years.

5. Select an enterprise project from the **Enterprise Project** drop-down list.

   This option is only available if you have logged in using an enterprise account, or if you have enabled enterprise projects.

   You can use an enterprise project to centrally manage your cloud resources and members by project.

NOTE

Value **default** indicates the default enterprise project. Resources that are not allocated to any enterprise projects under your account are displayed in the default enterprise project.

Step 4 Confirm the information and click **OK**.

After you submit your application, the system will return to the private certificate list page. Message "Certificate xxx applied for successfully." is displayed in the upper right corner of the page, indicating that the private certificate application is successful.

----End

Follow-up Operations

When a private certificate is issued, you can download it and distribute it to the certificate subject for installation. For details, see **Downloading a Private Certificate**.

3.3.2 Downloading a Private Certificate

Before using a private certificate, you need to download it. Only downloaded certificate can be assigned to the corresponding certificate subject so that they can install and use the certificate.

This topic describes how to download a private certificate. Only certificates in the **Issued** state can be downloaded.

Prerequisites

Your private certificate is in the **Issued** state. For details, see **Applying for a Private Certificate**.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed. In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 3 Locate the row of the desired private certificate and click **Download** in the **Operation** column.

Step 4 Click the target tab based on your server type and click **Download Certificate**.

When you click **OK**, CCM will use the download tool provided by the browser to download the private certificate to the specified local directory.

----End

Installing a Private Certificate

After downloading the private certificate, install it on the client or server.

- To install a certificate on a client, see **Installing a Private Certificate on a Client**.
- To install a certificate on a server, see **Table 3-12**.

Table 3-12 Example for installing a private certificate

Server Type	Operation
Tomcat	**Installing a Private Certificate on a Tomcat Server**
Nginx	**Installing a Private Certificate on an Nginx Server**
Apache	**Installing a Private Certificate on an Apache Server**
IIS	**Installing a Private Certificate on an IIS Server**
WebLogic	**Installing a Private Certificate on a WebLogic Server**
Resin	**Installing a Private Certificate on a Resin Server**

Description of Downloaded Certificate Files

The downloaded certificate files vary depending on the CSR file type (**System generated CSR** or **Upload a CSR**) configured when you apply for a private certificate.

- **System generated CSR**

  Table 3-13 describes the downloaded files.

Table 3-13 Description of downloaded files (1)

Server Type	Files in the Package
Tomcat	**keystorePass.txt**: certificate password **server.jks**: certificate file
Nginx	**server.crt**: certificate files, containing the server certificate and certificate chain **server.key**: certificate private key file
Apache	**chain.crt**: certificate chain file **server.crt**: certificate file **server.key**: certificate private key file

Server Type	Files in the Package
IIS	**keystorePass.txt**: certificate password **server.pfx**: certificate file
Others	**chain.pem**: certificate chain file **server.key**: certificate private key file **server.pem**: certificate file

- **Upload a CSR**

  Table 3-14 describes the downloaded files.

  Table 3-14 Description of downloaded files (2)

Server Type	Files in the Package
Tomcat	**server.crt**: certificate file **chain.crt**: certificate chain file
Nginx	**server.crt**: certificate file
Apache	**server.crt**: certificate file **chain.crt**: certificate chain file
IIS	**server.crt**: certificate file **chain.crt**: certificate chain file
Others	**cert.pem**: certificate file **chain.pem**: certificate chain file

3.3.3 Installing a Private Certificate

3.3.3.1 Trusting a Private Root CA

Before installing a private certificate, you need to add the root CA to the trusted root certificate authorities of the client or server.

Prerequisites

You have created and exported a private root CA. For details, see **Exporting a Private CA Certificate**.

Constraints

- One-way authentication

  To win more trust from the client for your server, you need to add the root CA that issue the server certificate to the client-end trusted CA store.
- Two-way authentication

To enable two-way authentication between a server and a client, each side needs to add the root CA of the other side to their own trusted root CA store.

Procedure

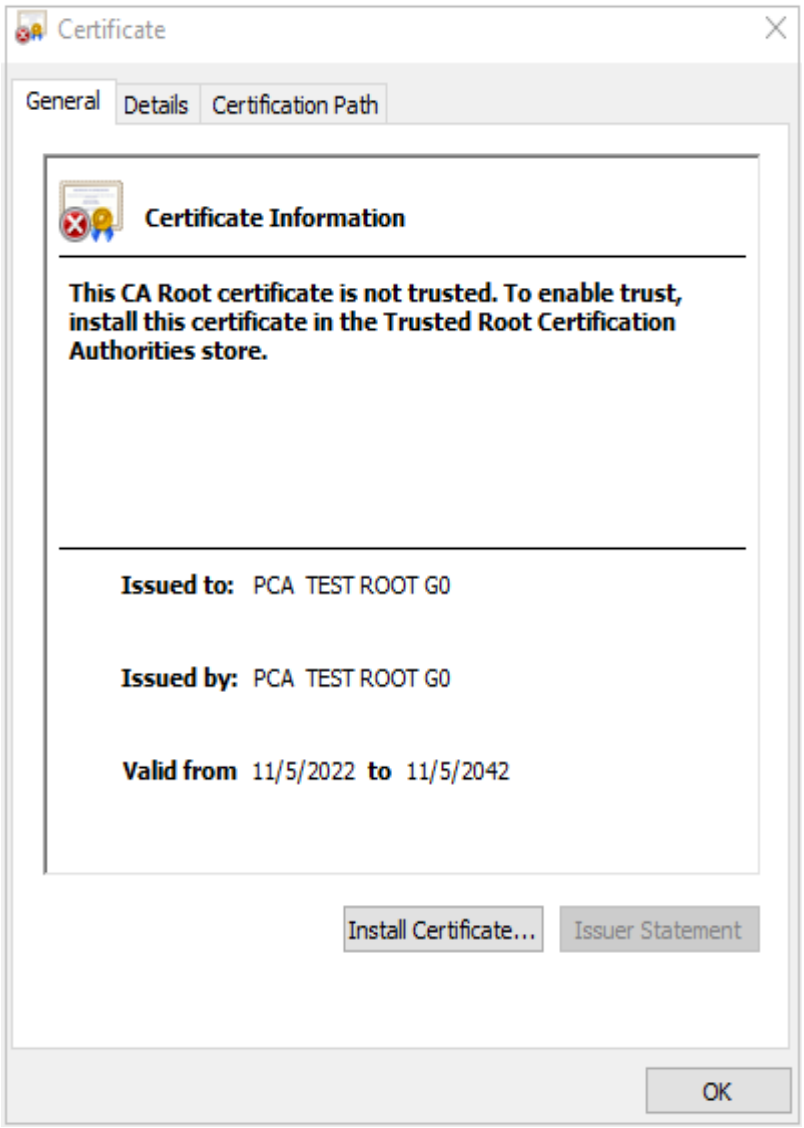
Use either of the following methods to add the root CA to trusted root certification authorities based on the operating system:

NOTE

Root CA **PCA TEST ROOT G0** is used as an example.

- **Windows**
  a. Change the file name extension of the root CA certificate from .pem to .crt. and double-click the certificate file. The root CA certificate information shows that the root certificate is untrusted.

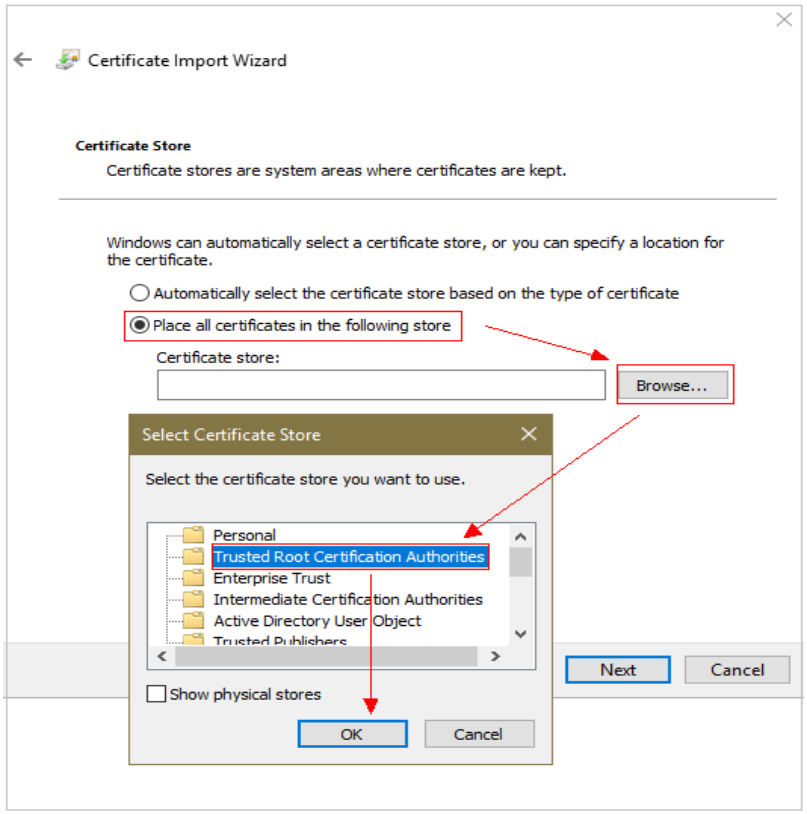
     Figure 3-2 Root CA not trusted

  b. Click **Install Certificate**, select a certificate storage location based on the certificate usage, and click **Next**.

c. As shown in **Figure 3-3**, select **Place all certificates in the following store** and click **Browse**. Then, select **Trusted Root Certification Authorities** and click **OK**.

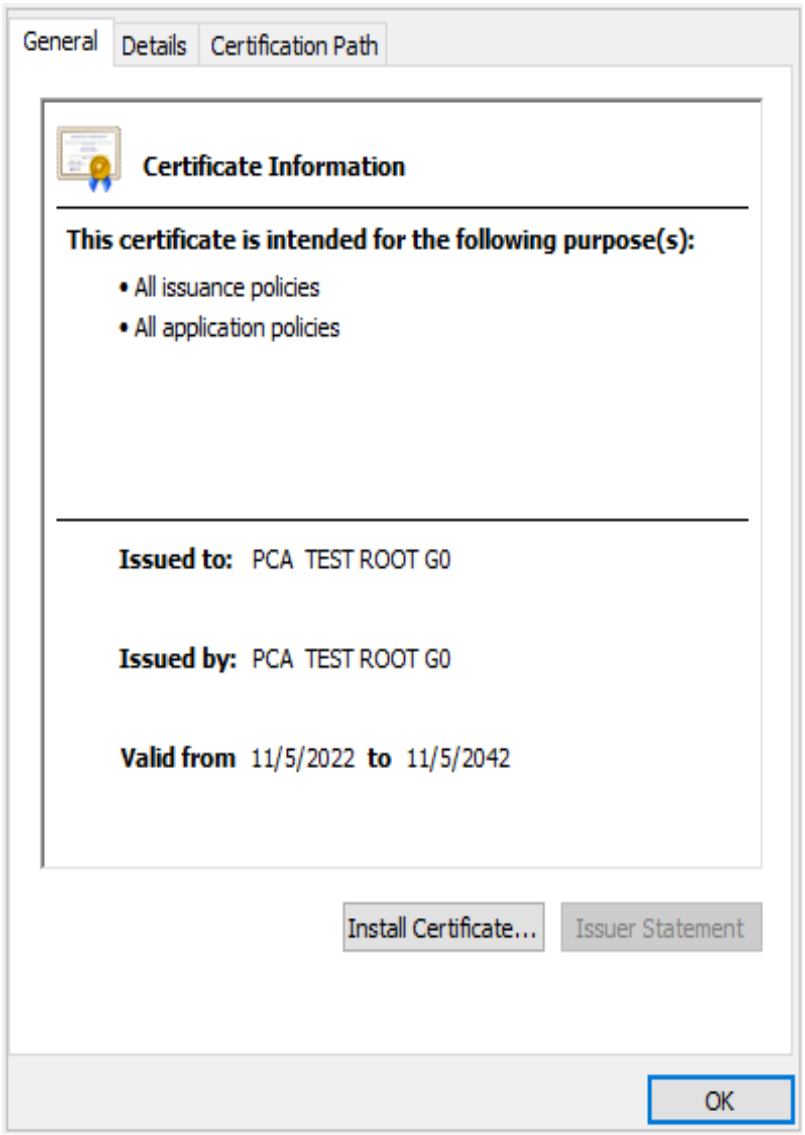
Figure 3-3 Storing a root certificate



d. Click **Next**, and then click **OK**. A dialog box is displayed, indicating that Windows will trust all certificates issued by the private root CA. Click **Yes**.

e. Double-click the root CA certificate file. If the **Certificate Information** area shows that the system trusts the root CA certificate, the root CA is added to the trusted root CAs.

Figure 3-4 Trusted root CA



- **Linux**

  The path for and method of storing root CA certificates vary depending on Linux OS versions. The following procedure use CentOS 6 as an example:

  a. Copy the root CA certificate file to the **/home/** directory.

  b. If **ca-certificates** is not installed on the server, run the following command to install **ca-certificates**:

     yum install ca-certificates

  c. Copy the root CA certificate to the **/etc/pki/ca-trust/source/anchors/** directory:

     cp /home/root.crt /etc/pki/ca-trust/source/anchors/

  d. Add the root CA certificate to the trusted root certificate file:

     update-ca-trust extract

  e. Check whether the information about the newly added root CA certificate is included in the command output:

     view /etc/pki/tls/certs/ca-bundle.crt

Figure 3-5 Root CA certificate added to the trusted CA list

NOTE

If the OpenSSL version is too old, the configuration may not take effect. You can run the **yum update openssl -y** command to update the OpenSSL version.

- **macOS**
  a. Open the macOS startup console and select **Keychain Access**.
  b. Enter the password to log in to **Keychain Access**.
  c. Drag and drop the target root CA certificate into **Keychain Access**. The root CA certificate now is untrusted by the system.
  d. Right-click the root CA certificate to load its details.
  e. Click **Trust**, select **Always Trust** for **When using this certificate**, and click **Close**.
  f. Enter the password to make the configuration of the trusted root CA certificate take effect.
  g. View the root CA certificate on the Keychain Access window. If the certificate is trusted by the system, the root CA is successfully added to the trusted root CA store.

3.3.3.2 Installing a Private Certificate on a Client

This topic describes how to install a private certificate on the client.

Prerequisites

You have downloaded an issued private certificate. For details about how to download a certificate, see **Downloading a Private Certificate**.

Constraints

If the server needs to verify the client certificate, you need to add the root CA of the client certificate to the trusted root CA store on the server. For details, see **Trusting a Private Root CA**.

Procedure

This procedure uses Windows as an example.

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed. In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 3 Locate the row containing the desired certificate. In the **Operation** column, click **Download**.

Step 4 Set Server type to **IIS** and click **Download Certificate**.

Step 5 Decompress the downloaded certificate file package **client_iis.zip** to obtain certificate file **server.pfx** and private key password file **keystorePass.txt**.

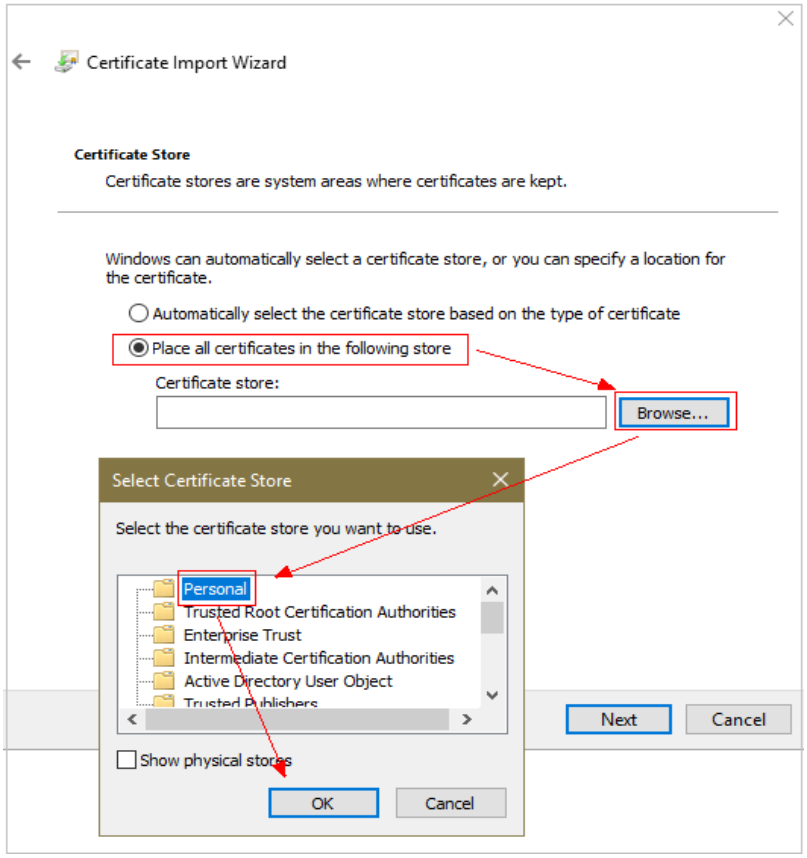
Step 6 Double-click certificate file **server.pfx**, select a certificate storage location based on its usage, and click **Next**.

Step 7 Confirm the name of the certificate file you want to import and click **Next**.

Step 8 Enter the password obtained from private key password file **keystorePass.txt** and click **Next**.

Step 9 Select **Place all certificates in the following store**, click **Browse**, select **Personal**, and click **OK**, as shown in **Figure 3-6**.

Figure 3-6 Storing a private certificate



Step 10 Click **Next** and **Finish**. The certificate is installed when a dialog box is displayed indicating that the certificate is imported successfully.

----End

3.3.3.3 Installing a Private Certificate on a Server

3.3.3.3.1 Installing a Private Certificate on a Tomcat Server

This topic describes how to install a private certificate on a Tomcat 7 server running a Linux OS.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Issued** status.
- You have downloaded the private certificate in the format that is supported by Tomcat. For details, see **Downloading a Certificate**.
- You have used a system-generated CSR to apply for the certificate.

Constraints

- Before installing the certificate, enable port 443 on the server where the private certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- A root CA must be added to the trusted client CA list so that all server certificates issued by the root CA can be trusted by the client. For details, see **Trusting a Private Root CA**.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- A private certificate can only be installed on the server that maps to the domain name associated with the certificate. Otherwise, the web browser will display a message indicating that the domain name is insecure.

Procedure

The installation process is as follows (for Tomcat 7 servers):

Step 1: Obtaining Files → **Step 2: Creating a Directory** → **Step 3: Modifying Configuration Files** → **Step 4: Restarting the Tomcat** → **Verifying the Result**

Step 1: Obtaining Files

Decompress the downloaded Tomcat certificate file to obtain the certificate file **server.jks** and password file **keystorePass.txt**.

Step 2: Creating a Directory

Create a **cert** directory in the Tomcat installation directory, and copy the **server.jks** and **keystorePass.txt** files to the **cert** directory.

Step 3: Modifying Configuration Files

NOTICE

Before modifying the configuration file, back up the configuration file. You are advised to deploy the configuration file in the test environment and then configure it on the production environment to avoid service interruptions caused by incorrect configurations.

The installation process is as follows (for Tomcat 7 servers):

1. Find the following parameters in the **server.xml** file in the Tomcat installation directory **conf**:

```
<!--
   <Connector port="8443" protocol="org.apache.coyote.http11.Http11Protocol"
              maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
              clientAuth="false" sslProtocol="TLS" />
   -->
```

2. Find the preceding parameters and delete the comment characters **<!-** and **->**.
3. Add the following parameters. Change the values of the parameters according to **Table 3-15**.


```
keystoreFile="cert/server.jks"
keystorePass="Certificate key"
```

The complete example configuration is as follows. Modify other parameters based on your needs.

```
<Connector port="443" protocol="org.apache.coyote.http11.Http11Protocol"
               maxThreads="150" SSLEnabled="true" scheme="https" secure="true"
      keystoreFile="cert/server.jks"
    keystorePass="Certificate key"
      clientAuth="false" sslProtocol="TLS" />
```

NOTICE

Do not directly copy all configuration. Only parameters **keystoreFile** and **keystorePass** need to be added. Set other parameters based on site requirements.

Table 3-15 Parameter description (1)

Parameter	Description
port	Port number to be used on the server. You are advised to set the value to **443**.
protocol	HTTP protocol. Retain the default value.
keystoreFile	Path for storing the **server.jks** file. The value can be an absolute path or a relative path. Example: **cert/server.jks**
keystorePass	Password of **server.jks**. Set this parameter to the password provided in the **keystorePass.txt** file. **NOTICE** If the password contains **&**, replace it with **&** to avoid configuration failure. An example command is provided as follows: If the password is **keystorePass="Ix6&APWgcHf72DMu"**, change it to **keystorePass="Ix6&APWgcHf72DMu"**.
clientAuth	Whether to require all customers to show the security certificate and authenticate their identity. Retain the default value.

4. Find the following parameters in the **server.xml** file in the Tomcat installation directory **conf**:

```
<Host name="localhost"  appBase="webapps"
    unpackWARs="true" autoDeploy="true">
```

5. Change the value of **Host name** to the domain name bound to the certificate.

   The complete configuration is as follows (**www.domain.com** is used as an example):

```
<Host name="www.domain.com"  appBase="webapps"
    unpackWARs="true" autoDeploy="true">
```


6. Save the configuration file.

Step 4: Restarting the Tomcat

Run the **./shutdown.sh** command in the **bin** directory of Tomcat to stop the Tomcat service.

After 10 seconds, run the **./startup.sh** command to start the Tomcat service. If the process is automatically started by the daemon process, you do not need to manually start the process.

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

3.3.3.3.2 Installing a Private Certificate on an Nginx Server

This topic describes how to install a private certificate on an Nginx 1.7.8 server running CentOS 7.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Issued** status.
- You have downloaded the private certificate in the format that is supported by Nginx. For details, see **Downloading a Certificate**.
- You have used a system-generated CSR to apply for the certificate.

Constraints

- Before installing the certificate, enable port 443 on the server where the private certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- A root CA must be added to the trusted client CA list so that all server certificates issued by the root CA can be trusted by the client. For details, see **Trusting a Private Root CA**.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- A private certificate can only be installed on the server that maps to the domain name associated with the certificate. Otherwise, the web browser will display a message indicating that the domain name is insecure.

Procedure

The installation process is as follows (for Nginx 1.7.8 servers running CentOS 7):

Step 1: Obtaining Files → **Step 2: Creating a Directory** → **Step 3: Modifying Configuration Files** → **Step 4: Verifying the Configuration** → **Step 5: Restarting Nginx** → **Verifying the Result**

Step 1: Obtaining Files

Decompress the downloaded certificate file on your local PC.

You will obtain certificate file **server.crt** and private key file **server.key**.

- The **server.crt** file contains two segments of certificate code: **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**, which are the server certificate and intermediate CA certificate respectively.
- **server.key** contains one segment of private key code: **-----BEGIN RSA PRIVATE KEY-----** and **-----END RSA PRIVATE KEY-----**.

Step 2: Creating a Directory

Create a **cert** directory in the Nginx installation directory, and copy the **server.key** and **server.crt** files to the **cert** directory.

Step 3: Modifying Configuration Files

NOTICE

Before modifying the configuration file, back up the configuration file. You are advised to deploy the configuration file in the test environment and then configure it on the production environment to avoid service interruptions caused by incorrect configurations.

Configure the **nginx.conf** file in the **conf** directory of Nginx.

1. Find the following configuration:

```
#server {
#    listen       443 ssl;
#    server_name  localhost;
#    ssl_certificate      cert.pem;
#    ssl_certificate_key  cert.key;
#    ssl_session_cache    shared:SSL:1m;
#    ssl_session_timeout  5m;
#    ssl_ciphers  HIGH:!aNULL:!MD5;
#    ssl_prefer_server_ciphers  on;
#    location / {
#        root   html;
#        index  index.html index.htm;
#    }
#}
```

2. Delete comment tags (#) at the beginning of the lines.

```
server {
            listen              443 ssl;
            server_name         localhost;
            ssl_certificate     cert.pem;
            ssl_certificate_key cert.key;
            ssl_session_cache   shared:SSL:1m;
            ssl_session_timeout 5m;
            ssl_ciphers         HIGH:!aNULL:!MD5;
            ssl_prefer_server_ciphers  on;
```



```
        location / {
                root     html;
                index    index.html index.htm;
                 }
}
```

3. Modify the following parameters according to **Table 3-16**.

```
ssl_certificate          cert/server.crt;
ssl_certificate_key      cert/server.key;
```

The complete configuration is as follows. Modify other parameters based on your needs.

```
server {
    listen       443 ssl; # Set the default HTTPS port to 443. If the default HTTPS port is not configured, Nginx may fail to start.
    server_name  www.domain.com; #Replace www.domain.com with the domain name associated with your certificate.
    ssl_certificate      cert/server.crt; #Replace cert/server.crt with the path of the certificate file.
    ssl_certificate_key  cert/server.key; #Replace cert/server.key with the path of the private key.
    ssl_session_cache    shared:SSL:1m;
    ssl_session_timeout  5m;
    ssl_ciphers  HIGH:!aNULL:!MD5; #Encryption suite
    ssl_prefer_server_ciphers  on;
    location / {
        root   html; #Site directory
        index  index.html index.htm; #Add attributes.
            }
}
```

NOTICE

Do not directly copy all configuration. Only attributes starting with **ssl** are directly related to the certificate configuration. Modify other parameters based on site requirements.

Table 3-16 Parameters

Parameter	Description
listen	SSL access port number. Set the value to **443**. Set the default HTTPS port to 443. If the default HTTPS port is not configured, Nginx may fail to start.
server_name	Domain name which the certificate is used for. Example: www.domain.com
ssl_certificate	Certificate file **server.crt** Set the value to the path of the **server.crt** file. An example of the path is **cert/server.crt**.
ssl_certificate_key	Private key file **server.key** Set the value to the path of the **server.key** file. An example of the path is **cert/server.key**.

4. Save the configuration file.

Step 4: Verifying the Configuration

Go to the execution directory of Nginx and run the following command:

sbin/nginx -t

If the following information is displayed, the configuration is correct.

```
nginx.conf syntax is ok
nginx.conf test is successful
```

Step 5: Restarting Nginx

Run the following command to restart Nginx to make the configuration take effect:

cd /usr/local/nginx/sbin

./nginx -s reload

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

3.3.3.3.3 Installing a Private Certificate on an Apache Server

This topic describes how to install a private certificate on an Apache 2.4.6 server running CentOS 7.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Issued** status.
- You have downloaded the private certificate in the format that is supported by Apache. For details, see **Downloading a Certificate**.
- You have used a system-generated CSR to apply for the certificate.

Constraints

- Before installing the certificate, enable port 443 on the server where the private certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- A root CA must be added to the trusted client CA list so that all server certificates issued by the root CA can be trusted by the client. For details, see **Trusting a Private Root CA**.
- If a domain name maps to multiple servers, deploy the certificate on each server.

- A private certificate can only be installed on the server that maps to the domain name associated with the certificate. Otherwise, the web browser will display a message indicating that the domain name is insecure.

Procedure

The installation process is as follows (for Apache 2.4.6 servers running CentOS 7):

Step 1: Obtaining Files → **Step 2: Creating a Directory** → **Step 3: Modifying Configuration Files** → **Step 4: Restarting Apache** → **Step 5: Verifying the Result**

Step 1: Obtaining Files

Decompress the downloaded certificate file on your local PC.

You will obtain certificate files **ca.crt** and **server.crt** and private key file **server.key**.

- **ca.crt** contains one segment of intermediate CA certificate code: **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**.
- **server.crt** contains one segment of server certificate code: **-----BEGIN CERTIFICATE-----** and **-----END CERTIFICATE-----**.
- **server.key** contains one segment of private key code: **-----BEGIN RSA PRIVATE KEY-----** and **-----END RSA PRIVATE KEY-----**.

Step 2: Creating a Directory

Create a **cert** directory in the Apache installation directory, and copy the **server.key**, **server.crt**, and **ca.crt** files to the **cert** directory.

Step 3: Modifying Configuration Files

NOTICE

Before modifying the configuration file, back up the configuration file. You are advised to deploy the configuration file in the test environment and then configure it on the production environment to avoid service interruptions caused by incorrect configurations.

1. Open the **conf.d/ssl.conf** file in the Apache root directory.
2. Configure the domain name associated with the certificate.

   Find and modify the following parameter:

   ```
   ServerName www.example.com:443
   ```

   The complete configuration is as follows (**www.domain.com** is used as an example):

   ServerName *www.domain.com*:443 #Replace *www.domain.com* with the domain name of your server.
3. Configure the public key for the certificate.

   Find and modify the following parameter:

   ```
   SSLCertificateFile "${SRVROOT}/conf/server.crt"
   ```


Set the value to the path of the **server.crt** file. An example of the path is **cert/server.crt**.

The complete configuration is as follows:

```
SSLCertificateFile "cert/server.crt"
```

4. Configure the private key for the certificate.

   Find and modify the following parameter:

   ```
   SSLCertificateKeyFile "${SRVROOT}/conf/server.key"
   ```

   Set the value to the path of the **server.key** file. An example of the path is **cert/server.key**.

   The complete configuration is as follows:

   ```
   SSLCertificateKeyFile "cert/server.key"
   ```

5. Configure the certificate chain.

   Find and modify the following parameter:

   ```
   #SSLCertificateChainFile "${SRVROOT}/conf/server-ca.crt"
   ```

   Delete the comment tag **#** at the beginning of the line. Set this parameter to the path of the **ca.crt** file. An example of the path is **cert/ca.crt**.

   The complete configuration is as follows:

   ```
   SSLCertificateChainFile "cert/ca.crt"
   ```

6. Save the **ssl.conf** file and exit.

Step 4: Restarting Apache

Restart the Apache service for the configuration to take effect:

1. Run the **service named stop** command to stop the Apache server.
2. Run the **service httpd start** command to start the Apache server.

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://***Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

3.3.3.3.4 Installing a Private Certificate on an IIS Server

This topic describes how to install a private certificate on an IIS server.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Issued** status.
- You have downloaded the private certificate in the format that is supported by IIS. For details, see **Downloading a Certificate**.

- You have used a system-generated CSR to apply for the certificate.

Constraints

- Before installing the certificate, enable port 443 on the server where the private certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- A root CA must be added to the trusted client CA list so that all server certificates issued by the root CA can be trusted by the client. For details, see **Trusting a Private Root CA**.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- A private certificate can only be installed on the server that maps to the domain name associated with the certificate. Otherwise, the web browser will display a message indicating that the domain name is insecure.

Procedure

To install a private certificate on an IIS server, perform the following steps:

Step 1: Obtaining Files → **Step 2: Configuring IIS** → **Step 3: Verifying the Result**

Step 1: Obtaining Files

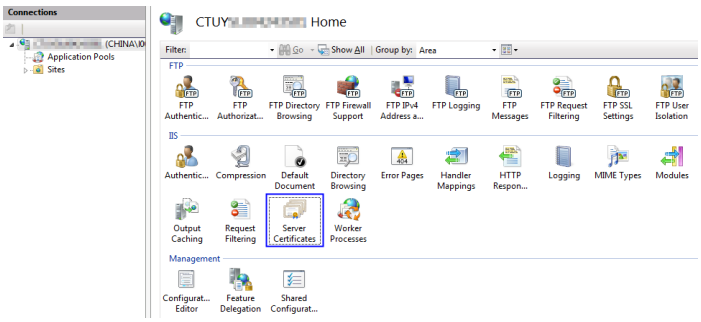
Decompress the downloaded certificate file on your local PC.

You will obtain certificate file **server.pfx** and password file **keystorePass.txt**.

Step 2: Configuring IIS

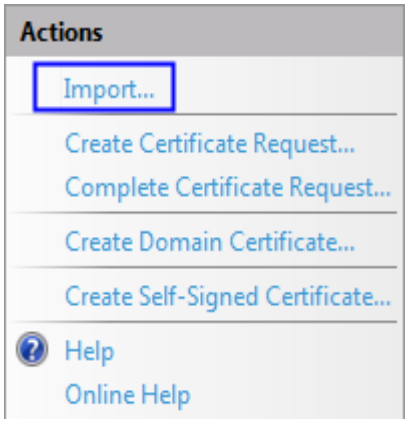
1. Install IIS as instructed by IIS guides.
2. Open the IIS management console, double-click **Server Certificates**.

Figure 3-7 Double-clicking Server Certificates



3. In the displayed dialog box, click **Import**.

Figure 3-8 Import

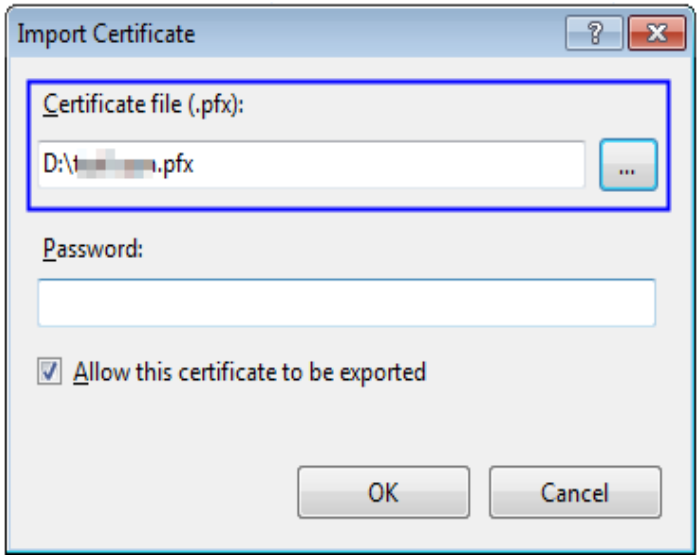


4. Import the **server.pfx** certificate file. Then click **OK**.

   NOTE

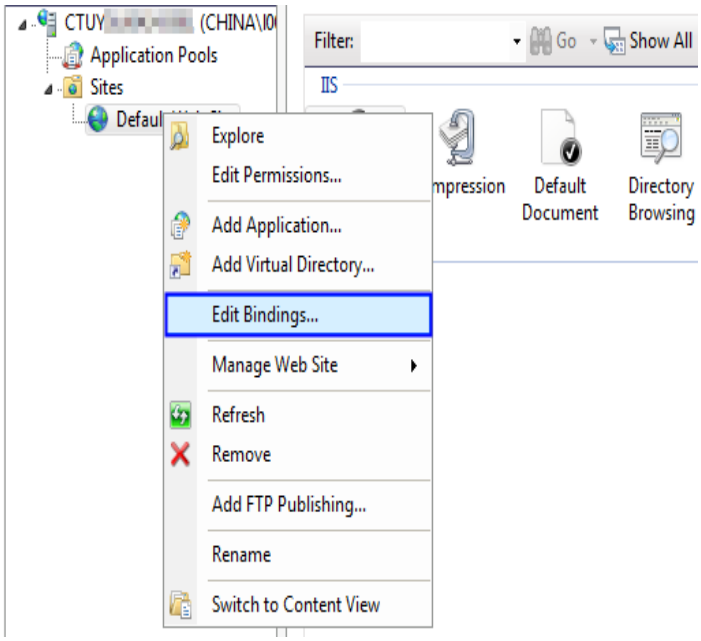
   In the **Password** box, enter the password provided in the **keystorePass.txt** file.

Figure 3-9 Importing a PFX certificate file



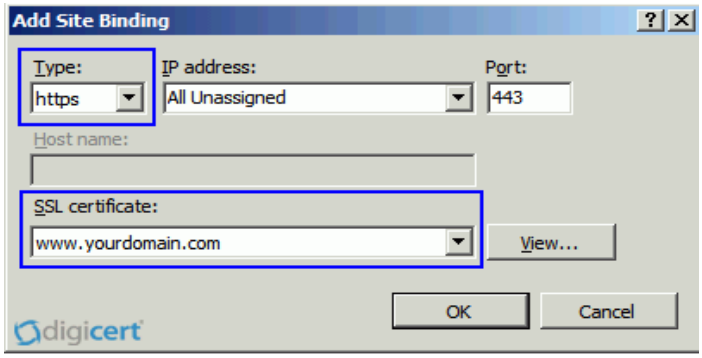
5. Right-click the target site (the default site is used as an example). Choose **Edit Bindings** from the shortcut menu.

Figure 3-10 Choosing Edit Bindings



6. In the dialog box that is displayed, click **Add**. Then enter the following information.

Figure 3-11 Binding a website



   - **Type**: Select **https**.
   - **Port**: Retain the default port **443**.
   - **SSL certificate**: Select the certificate imported in **4**.
7. Click **OK**.

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

3.3.3.3.5 Installing a Private Certificate on a WebLogic Server

WebLogic is a Java EE application server, used to develop, integrate, deploy, and manage large-scale distributed Web apps, network apps, and database apps. It

applies dynamic functions of Java and security of the Java Enterprise standard to the development, integration, deployment, and management of large-scale network applications.

Currently, WebLogic 10.3.1 and later versions support SSL certificates of all mainstream brands. Versions earlier than WebLogic 10.3.1 do not support SSL certificates of brands.

This topic describes how to install a private certificate on a Weblogic server.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Issued** status.
- You have downloaded the private certificate in the format that is supported by Tomcat. For details, see **Downloading a Certificate**.
- You have used a system-generated CSR to apply for the certificate.
- The JDK has been installed.

  The JDK has been installed after WebLogic installation is complete. If the JDK has not been installed, install the **Java SE Development Kit (JDK)**.

Constraints

- Before installing the certificate, enable port 443 on the server where the private certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- A root CA must be added to the trusted client CA list so that all server certificates issued by the root CA can be trusted by the client. For details, see **Trusting a Private Root CA**.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- A private certificate can only be installed on the server that maps to the domain name associated with the certificate. Otherwise, the web browser will display a message indicating that the domain name is insecure.

Procedure

To install a private certificate on a WebLogic server, perform the following steps:

Step 1: Obtaining Files → **Step 2: Configuring WebLogic** → **Verifying the Result**

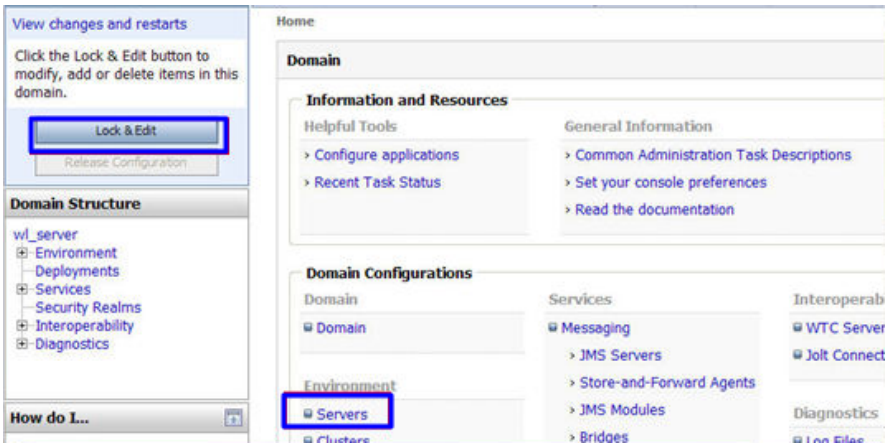
Step 1: Obtaining Files

Decompress the downloaded Tomcat certificate file to obtain the certificate file **server.jks** and password file **keystorePass.txt**.

Step 2: Configuring WebLogic

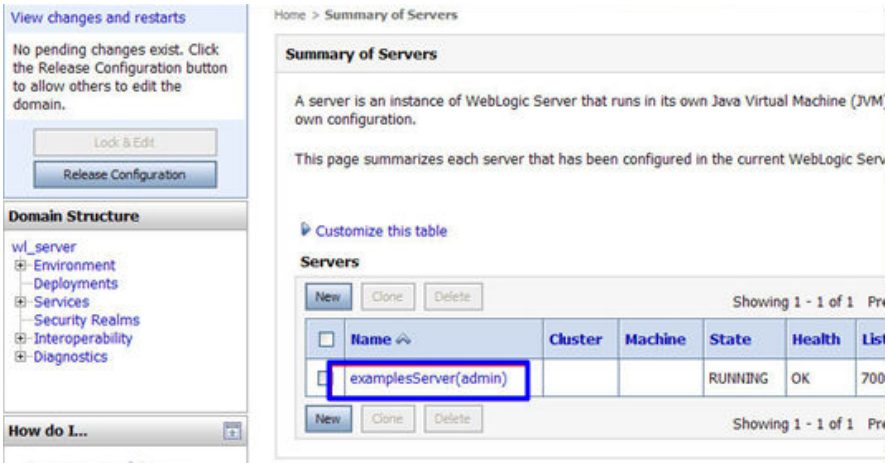
1. Log in to the management console of the WebLogic server.
2. Click **Lock & Edit** in the upper left corner of the page to unlock the configuration.
3. Click **Servers** in **Domain Configurations**.

Figure 3-12 Server



4. In the server list, select the server for which you want to configure the server certificate. The server configuration page is displayed.

Figure 3-13 Target server

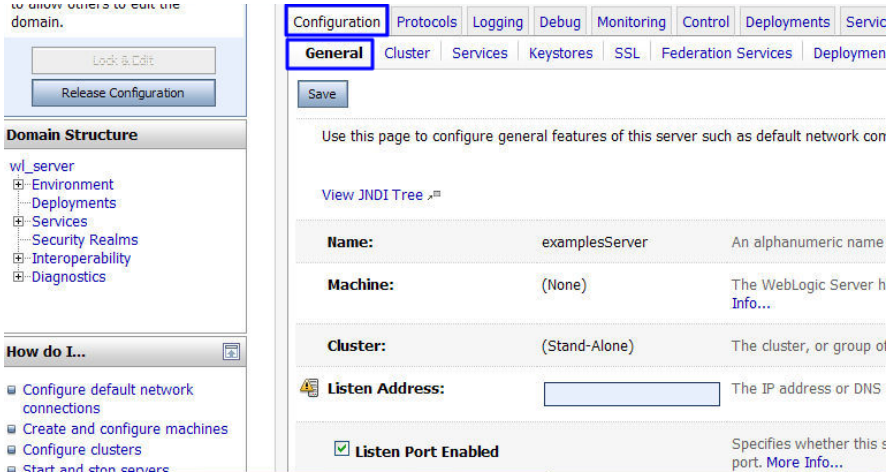


5. Modify the HTTPS port.

   On the server configuration page, click the **General** tab and configure whether to enable HTTP and HTTPS and the access port number.

   Select **Listen SSL Port Enabled** and change the port number to **443**.

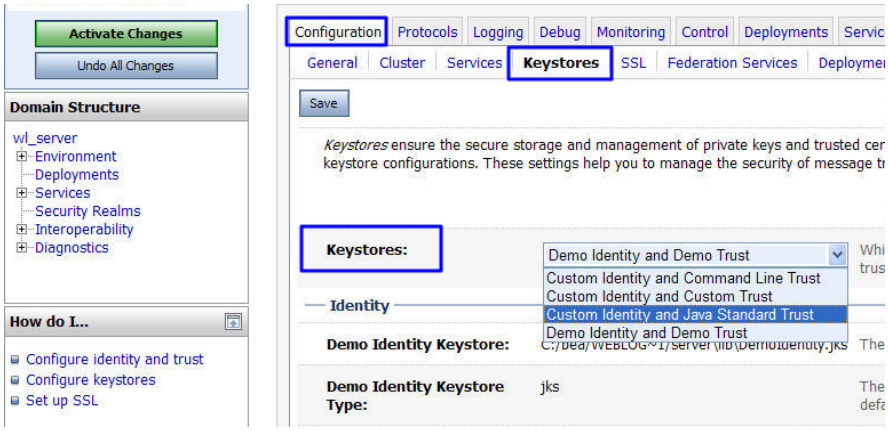
Figure 3-14 port



6. Configure an authentication mode and a key.

   a. On the server configuration page, click the **Keystores** tab and configure an authentication mode.

      Figure 3-15 Authentication mode

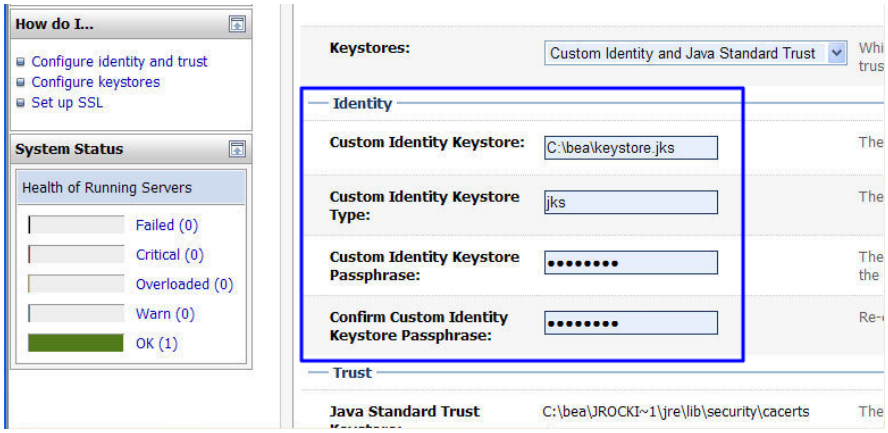


      - Select **Custom Identity and Java Standard Trust** for server authentication.
      - Select **Custom Identity and Custom Trust** for bidirectional authentication.

   b. Configure a key in the **Identity** area.

      Configure the path for storing the keystore file **server.jks** on the server and enter the password of the keystore file.

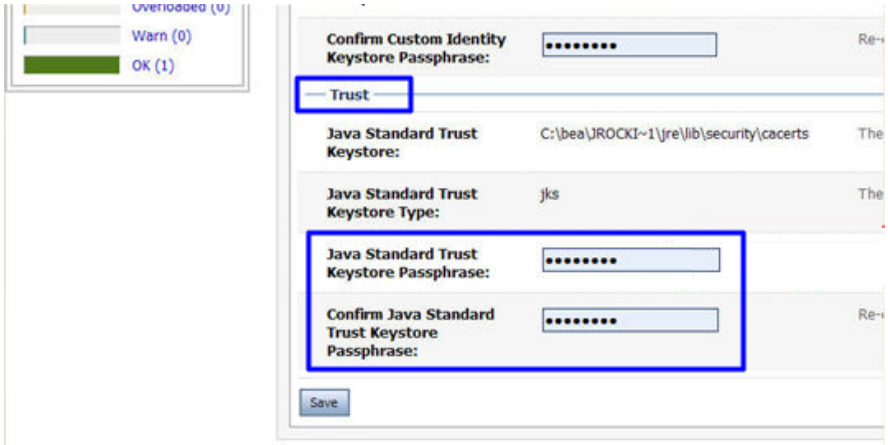
Figure 3-16 Key



- **Custom Identity Keystore**: Enter the path for storing the .jks file. Example: C:\bea\server.jks
- **Custom Identity Keystore Type**: Set the file format to **jks**.
- **Custom Identity Keystore Passphrase**: Enter the certificate password, that is, the password in **keystorePass.txt**.
- **Confirm Custom Identity Keystore Passphrase**: Re-enter the certificate password.

c. In unidirectional authentication, configure the default trust store file **cacerts** of the JRE.

Figure 3-17 Trust store file

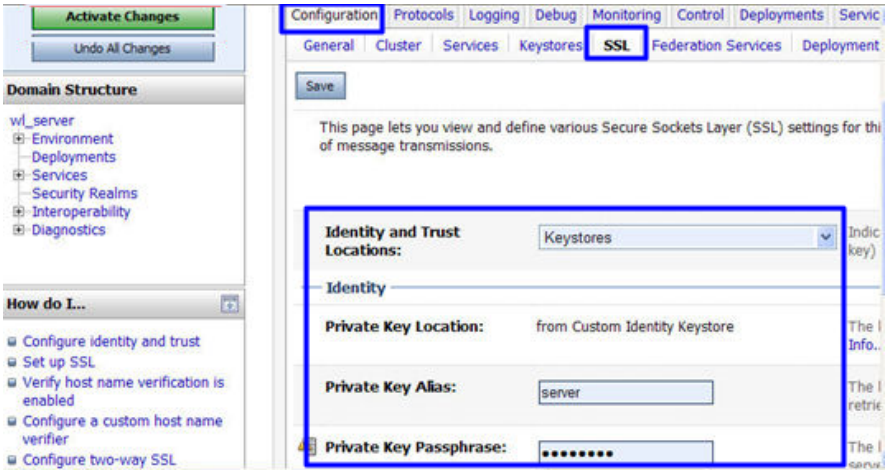


- **Java Standard Trust Keystore Passphrase**: Enter the password.
- **Confirm Java Standard Trust Keystore Passphrase**: Re-enter the password.

7. Configure the private key alias of the server certificate.

   On the server configuration page, click the **SSL** tab and set the following parameters:

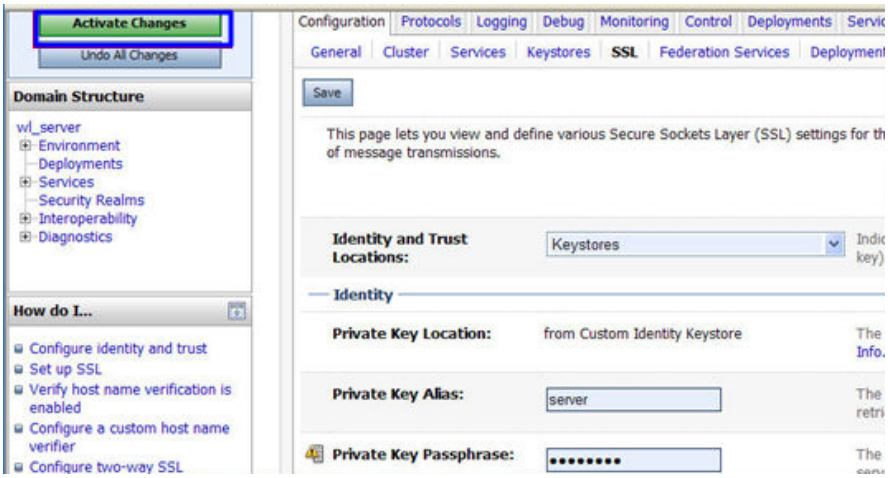
Figure 3-18 Private key



- **Identity and Trust Locations**: Select **Keystores**.
- **Private Key Alias**: Configure a private key alias in the private key library. You can run the **keystool -list** command to view the private key alias.
- **Private Key Passphrase**: Enter the private key protection password. Generally, the private key protection password is the same as the keystore file protection password.
- **Confirm Private Key Passphrase**: Enter the private key protection password again.

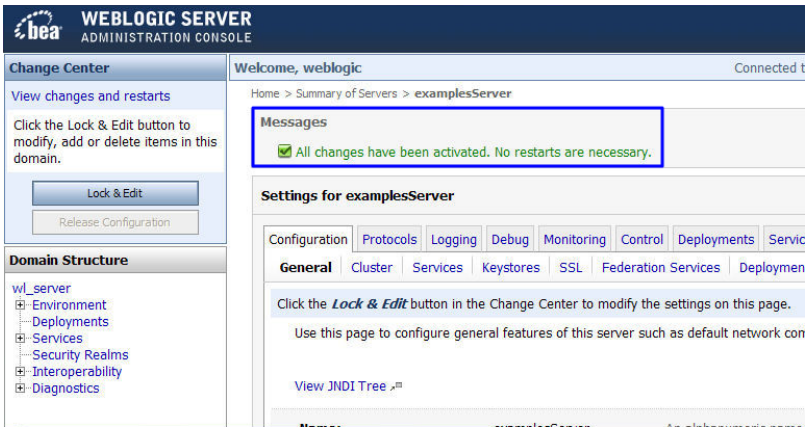
8. Click **Active Changes** to save the settings.

Figure 3-19 Saving the settings



9. (Optional) If the system prompts you to restart the WebLogic server, restart the WebLogic server for the settings to take effect. As shown in **Figure 3-20**, you do not need to restart the WebLogic server.

Figure 3-20 Message displayed



Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

3.3.3.3.6 Installing a Private Certificate on a Resin Server

This topic describes how to install a private certificate on a Resin server.

NOTE

The installation procedure in this topic is for your reference only as the commands executed and configuration file modified during the installation may vary depending on OS types and server configurations.

Prerequisites

- The certificate is in the **Issued** status.
- You have downloaded the private certificate in the format that is supported by Tomcat. For details, see **Downloading a Certificate**.
- You have used a system-generated CSR to apply for the certificate.

Constraints

- Before installing the certificate, enable port 443 on the server where the private certificate is installed and add port 443 to the security group. Otherwise, HTTPS cannot be enabled after the installation.
- A root CA must be added to the trusted client CA list so that all server certificates issued by the root CA can be trusted by the client. For details, see **Trusting a Private Root CA**.
- If a domain name maps to multiple servers, deploy the certificate on each server.
- A private certificate can only be installed on the server that maps to the domain name associated with the certificate. Otherwise, the web browser will display a message indicating that the domain name is insecure.

Procedure

To install a private certificate on a Resin server, perform the following steps:

Step 1: Obtaining Files → **Step 2: Configuring Resin** → **Verifying the Result**

Step 1: Obtaining Files

Decompress the downloaded Tomcat certificate file to obtain the certificate file **server.jks** and password file **keystorePass.txt**.

Step 2: Configuring Resin

NOTICE

Before modifying the configuration file, back up the configuration file. You are advised to deploy the configuration file in the test environment and then configure it on the production environment to avoid service interruptions caused by incorrect configurations.

1. (Optional) Install Resin.

   If you have installed Resin, skip this step.

   a. Log in to the **Resin** official website and download the appropriate application packages for your operating system.

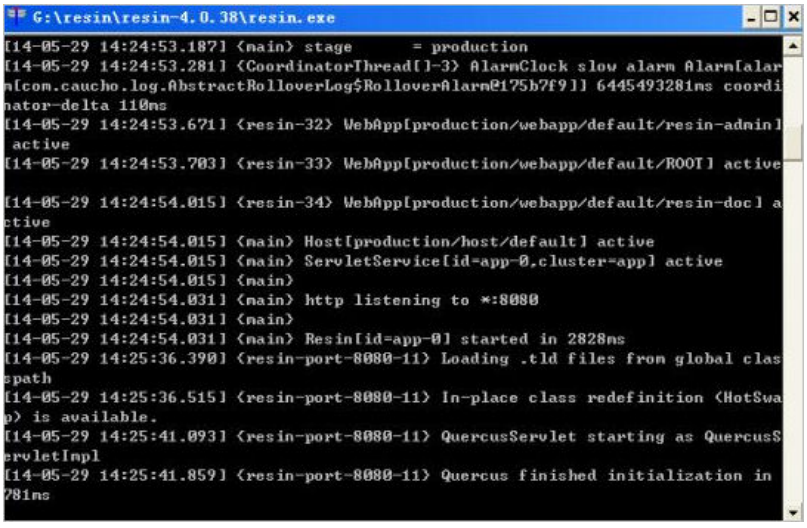
      The following uses **Resin-4.0.38** for Windows as an example.

   b. Decompress the downloaded Resin software package.

   c. Access the root directory of Resin-4.0.38 and find the **resin.exe** file.

   d. Run the **resin.exe** file. During the execution, the command prompt window **Figure 3-21** will display.

      Figure 3-21 Information dialog box

   e. After the resin.exe file is executed. Start the Explorer, enter the default address **http://127.0.0.1:8080** of Resin in the address bar, and then press **Enter**.

If the information similar to **Figure 3-22** is displayed, Resin is installed successfully.

Figure 3-22 Logging In to Resin

2. Modify the configuration file.
   a. Find the following parameters in the **Resin.properties** configuration file in the Resin installation directory (the configuration file may be **resin.xml** for different Resin versions):

```
# specifies the --server in the config file
# home_server : app-0

# Set HTTP and HTTPS bind address
# http_address : *

# Set HTTP and HTTPS ports.
# Use overrides for individual server control, for example: app-0.http : 8081
app.http         : 8080
# app.https         : 8443

web.http         : 8080
# web.https         : 8443
```

   b. Delete the comment symbol (#) before **app.https** and **web.https**. Then modify port **8443** to **443**. After the modification:

      app.https and **web.https**: Port to be used on the server. You are advised to set the value to **443**.

```
# specifies the --server in the config file
# home_server : app-0

# Set HTTP and HTTPS bind address
# http_address : *

# Set HTTP and HTTPS ports.
# Use overrides for individual server control, for example: app-0.http : 8081
app.http         : 8080
app.https         : 443

web.http         : 8080
web.https         : 443
```

   c. Find the following parameters and delete the comment symbol (#) before **jsse_keystore_type**, **jsse_keystore_file**, and **jsse_keystore_password**.

```
# JSSE certificate configuration
# Keys are typically stored in the resin configuration directory.
jsse_keystore_tye : jks
jsse_keystore_file : cert/server.jks
jsse_keystore_password: certificate password
```


d. Modify certificate-related parameters. For details, see **Table 3-17**.

```
# JSSE certificate configuration
# Keys are typically stored in the resin configuration directory.
jsse_keystore_tye : jks
jsse_keystore_file : cert/server.jks
jsse_keystore_password: certificate password
```

Table 3-17 Description

Parameter	Description
jsse_keystore_tye	Type of the Keystore file. Generally, this parameter is set to **jks**.
jsse_keystore_file	Path for storing the **server.jks** file. The value can be an absolute path or a relative path. Example: **cert/server.jks**
jsse_keystore_password	Password of **server.jks**. Set this parameter to the password provided in the **keystorePass.txt** file. **NOTICE** If the password contains **&**, replace it with **&** to avoid configuration failure. An example command is provided as follows: If the password is **keystorePass="lx6&APWgcHf72DMu"**, change it to **keystorePass="lx6&APWgcHf72DMu"**.

e. Save the configuration file.

3. Restart Resin.

Verifying the Result

After the deployment succeeds, in the address bar of the browser, enter **https://** *Domain name* and press **Enter**.

If a security padlock is displayed in the address bar of the browser, the certificate has been installed successfully.

3.3.4 Revoking a Private Certificate

If a private certificate is no longer needed or its private key is lost before it expires, you can revoke it on the console. If a private certificate is revoked, it is no longer trusted within the organization.

If a private certificate is revoked, the billing stops.

The following describes how to revoke a private certificate.

Prerequisites

The private certificate is in the **Issued** state.

Constraints

- After you apply for revoking a private certificate, your application cannot be withdrawn. Exercise caution when performing this operation.
- All its records will be cleared and cannot be recovered, including private CA records. Therefore, exercise caution when performing this operation.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed. In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 3 Locate the row of the desired private certificate and click **Revoke** in the **Operation** column.

Step 4 In the displayed dialog box, enter **REVOKE** and select the revocation reason to confirm the revocation. The default revocation reason is in the **UNSPECIFIED** field. Table 3-18 describes the revocation reasons you can select.

Table 3-18 Revocation reasons and meaning

Reason for Revocation	Reason Code in RFC 5280	Description
UNSPECIFIED	0	Default value. No reason is specified for revocation.
KEY_COMPROMISE	1	The certificate key material has been leaked.
CERTIFICATE_AUTHORITY_COMPROMISE	2	Key materials of the CA have been leaked in the certificate chain.
AFFILIATION_CHANGED	3	The subject or other information in the certificate has been changed.
SUPERSEDED	4	The certificate has been replaced.
CESSATION_OF_OPERATION	5	The entity in the certificate or certificate chain has ceased to operate.

Reason for Revocation	Reason Code in RFC 5280	Description
CERTIFICATE_HOLD	6	The certificate should not be considered valid currently and may take effect in the future.
PRIVILEGE_WITHDRAWN	9	The certificate no longer has the right to declare its listed attributes.
ATTRIBUTE_AUTHORITY_COMPROMISE	10	The authority that warrants the attributes of the certificate may have been compromised.

Step 5 Click **OK**.

When "Certificate xxx revoked." is displayed in the upper right corner of the page, and the private certificate status changes to **Revoked**, the private certificate is revoked successfully.

----End

3.3.5 Viewing Details of a Private Certificate

This topic describes how to view details of a private certificate, including the common name, expiration time, and status.

Prerequisites

You have applied for a private certificate. For details, see **Applying for a Private Certificate**.

Procedure

Step 1 Log in to the management console.

Step 2 Click  in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed. In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 3 View the private certificate information. **Table 3-19** describes the private certificate parameters.

Table 3-19 Private certificate parameters

Parameter	Description
Common Name	Indicates the name of the private certificate configured during certificate application.
Issued By	Indicates the name of the private CA that issues the private certificate.
Creation Time	Indicates the time when a private certificate is created.
Expiration Time	Indicates the time when a certificate expires.
Status	Indicates the certificate status. The value can be: • **Issued** The private certificate is issued. • **Expired** The private certificate is expired. • **Revoked** The private certificate is revoked.
Enterprise Project	Enterprise project to which the private certificate belongs.
Operation	You can download, revoke, or delete the certificate.

Step 4 Click the common name of a private certificate to view its details.

You can click **Add Tag** on the tab page of the private certificate details page to identify the private certificate. TMS's predefined tag function is recommended for adding the same tag to different cloud resources.

----End

3.3.6 Deleting a Private Certificate

This topic describes how to delete a private certificate. A deleted private certificate remains valid and trusted.

You can delete a certificate that is no longer needed.

Prerequisites

The private certificate is in the **Issued**, **Expired**, or **Revoked** state.

Constraints

- A deleted certificate cannot be restored. Exercise caution with the deletion.
- After you submit a certificate deletion application, you cannot cancel it. Exercise caution when performing this operation.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed. In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 3 Locate the row of the private certificate to be deleted and click **Delete** in the **Operation** column.

Step 4 In the displayed dialog box, enter **DELETE** to confirm the deletion.

Step 5 Click **OK**. If message "Certificate xxx deleted successfully." is displayed in the upper right corner of the page, the private certificate is deleted successfully.

----End

3.4 Managing Tags

3.4.1 Overview

Scenario

Tags can be used to identify private CAs and private certificates. You can use tags to group and centrally manage private CAs and private certificates by usage, owner, or environment.

You can add tags when purchasing a CA or private certificate, or add tags on the details page of the CA or private certificate after the purchase.

Tag Naming Rules

- Each tag consists of a key-value pair.
- A maximum of 20 tags can be added to each private CA or private certificate.
- For each resource, a tag key must be unique and can have only one tag value.
- A tag consists of a tag key and a tag value. The naming rules are listed in **Table 3-20**.

  NOTE

  If your organization has configured a tag policy for the CCM service, you need to add tags to private CA or private CAs based on the tag policy. Otherwise, the tagging operation might fail. For more information about the tag policy, contact your organization administrator.

Table 3-20 Tag parameters

Parameter	Rule	Example
Tag key	• This parameter is mandatory. • For a private CA or private certificate, the tag key must be unique. • The value can contain a maximum of 128 characters. • The value cannot start or end with a space. • The value cannot start with _**sys**_. • The following character types are allowed: – Chinese – English – Digit – Space – Special characters: _.:=+-@	cost
Tag value	• This parameter can be left empty. • The value can contain a maximum of 255 characters. • The value cannot start or end with a space. • The following character types are allowed: – Chinese – English – Digit – Space – Special characters: _.:=+-@	100

3.4.2 Creating a Tag

This topic describes how to add tags to private CAs and private certificates.

Creating a Tag for a Private CA

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Click the name of the target private CA. The private CA details page is displayed.

Step 5 Click the **Tags** tab to go to the tag management page.

Step 6 Click **Edit Tag**. On the displayed **Edit Tag** page displayed on the right, click **Add Tag**. In the text box, specify **Tag key** and **Tag value**.

NOTE

To delete a tag, click **Delete** next to it.

Step 7 Click **OK** to complete.

----End

Creating a Tag for a Private Certificate

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security** > **Cloud Certificate Management Service**.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 4 Click the name of the target private certificate to go to its details page.

Step 5 Click the **Tags** tab to go to the tag management page.

Step 6 Click **Edit Tag**. In the displayed **Edit Tag** page, click **Add Tag**. In the text box, specify **Tag key** and **Tag value**.

NOTE

To delete a tag, click **Delete** next to it.

Step 7 Click **OK** to complete.

----End

3.4.3 Searching for Private CAs or Certificates by Tag

This topic describes how to search for private CAs or certificates that meet the search criteria by tag in the current project.

Prerequisites

A tag has been added. For details, see **Creating a Tag**.

Constraints

At most 20 tags can be added for one search. If multiple tags are added, private CAs or certificates that meet all search criteria will be displayed.

Searching for Private CAs by Tag

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Click the search box and enter the tag key and tag value to search for the resource. Private CAs that meet the search criteria are displayed.

NOTE

- At most 20 tags can be added for one search. If multiple tags are added, private CAs that meet all search criteria will be displayed.
- If you want to delete an added tag from the search criteria, click × next to the tag.

----End

Searching for Private Certificates by Tag

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 4 Click the search box and enter the tag key and tag value to search for the resource. Private certificates that meet the search criteria are displayed.

NOTE

- At most 20 tags can be added for one search. If multiple tags are added, private certificates that meet all search criteria will be displayed.
- If you want to delete an added tag from the search criteria, click × next to the tag.

----End

3.4.4 Modifying a Tag Value

This section describes how to modify a private CA or private certificate tag.

Modifying the Private CA Tag Value

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Click the name of the target private CA. The private CA details page is displayed.

Step 5 Click **Tags** tab to go to the tag management page.

Step 6 Click **Edit Tag**. In the displayed dialog box, change the tag value and click **OK**. The tag value is changed.

----End

Changing the Tag Value of a Private Certificate

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 4 Click the name of the target private certificate. The details page is displayed.

Step 5 Click **Tags** tab to go to the tag management page.

Step 6 Click **Edit Tag**. In the displayed dialog box, change the tag value and click **OK**. The tag value is changed.

----End

3.4.5 Deleting a Tag

This section describes how to delete a private CA tag or private certificate tag.

Deleting a Private CA Tag

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs**.

Step 4 Click the name of the target private CA. The private CA details page is displayed.

Step 5 Click **Tags** tab to go to the tag management page.

Step 6 Click **Edit Tag**. In the displayed dialog box, locate the row that contains the target tag, click **Delete**, and then click **OK**.

----End

Deleting a Private Certificate Tag

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private Certificates**.

Step 4 Click the name of the target private certificate. The details page is displayed.

Step 5 Click **Tags** tab to go to the tag management page.

Step 6 Click **Edit Tag**. In the displayed dialog box, locate the row that contains the target tag, click **Delete**, and then click **OK**.

----End

3.5 Assigning a CA or Private Certificate to an Enterprise Project

You can use enterprise projects to more efficiently manage cloud resources and project members. For more details, see *Enterprise Management User Guide*.

This topic describes how to add a CA or private certificate to an enterprise project.

Prerequisites

An enterprise project has been created. To use this function, enable it by referring to "Enabling Enterprise Management" in *Enterprise Management User Guide*.

Procedure

Step 1 Log in to the management console.

Step 2 Click ☰ in the upper left corner of the page and choose **Security & Compliance** > **Cloud Certificate Management Service**. The service console is displayed.

Step 3 In the navigation pane on the left, choose **Private Certificate Management** > **Private CAs** or choose **Private Certificate Management** > **Private Certificate**.

Step 4 In the row containing the target private CA or private certificate, click **Add to Project** in the **Operation** column.

Step 5 Select an enterprise project.

Step 6 Click **OK**.

----End

3.6 Permissions Management

3.6.1 Creating a User and Granting CCM Permissions to the User

This topic describes how to use IAM to implement fine-grained permissions control for your CCM resources. With IAM, you can:

- Create IAM users for employees based on the organizational structure of your enterprise. Each IAM user has their own security credentials, providing access to CCM resources.
- Grant only the permissions required for users to perform a task.
- Entrust an account or cloud service to perform professional and efficient O&M to your CCM resources.

If your account does not require individual IAM users, skip this chapter.

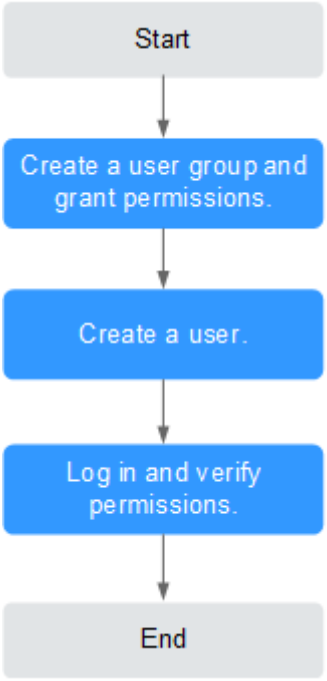
This section provides some methods for you to assign permissions to a user. **Figure 3-23** shows the process.

Prerequisites

Before authorizing permissions to a user group, you need to know which CCM permissions can be added to the user group.

Process Flow

Figure 3-23 Process for granting CCM permissions



1. Create a user group and assign permissions.

   Create a user group on the IAM console and grant the user group the **PCA FullAccess**.

2. Create a user and add it to a user group.

   Create a user on the IAM console and add the user to the group created in **1**.

3. Log in and verify the permissions.

   Log in to the CCM console by using the created user, and verify that the user only has read permissions for CCM.

   Choose **Cloud Certificate Management Service** under **Security** in the **Service List**. If no message appears indicating that you have no permissions to access the service, the policy **PCA FullAccess** has already taken effect.

3.6.2 CCM Custom Policies

Custom policies can be created to supplement the system-defined policies of CCM.

You can create custom policies in either of the following two ways:

- Visual editor: Select cloud services, actions, resources, and request conditions. This does not require knowledge of policy syntax.
- JSON: Edit JSON policies from scratch or based on an existing policy.

The following section contains examples of common CCM custom policies.

Example CCM Custom Policies

- Example 1: authorizing users to create a CA

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Action": [
                "pca:ca:create
            ],
            "Effect": "Allow"
        }
    ]
}
```

- Example 2: denying certificate deletion

  A policy with only "Deny" permissions must be used in conjunction with other policies to take effect. If the permissions assigned to a user contain both "Allow" and "Deny", the "Deny" permissions take precedence over the "Allow" permissions.

  If you need to assign permissions of the **PCA FullAccess** policy to a user but you want to prevent the user from deleting certificates, you can create a custom policy for denying certificate deletion, and attach both policies to the group that the user belongs to. Then, the user can perform all operations on certificates except deleting certificates. The following is an example of a deny policy:

```
{
    "Version": "1.1",
    "Statement": [
        {
            "Action": [
                "pca:ca:delete"
```



```
                ],
                "Effect": "Deny"
            }
        ]
}
```


4 FAQs

4.1 What Is a Public Key and a Private Key?

A pair of public and private keys are used in the encryption method commonly known as the asymmetric encryption method. The key pair, consisting of a public key and a private key, is generated based on an algorithm. The public key is open while the private key is not. The public key is usually used to encrypt session keys, verify digital signatures, or encrypt data that can be decrypted using the corresponding private key.

The public and private key pair is unique across the whole world. If one key is used to encrypt a piece of data, the other key must be used to decrypt the data.

The standard application of the public key and private key is as follows:

- Encryption and decryption scenarios: The public key is used for encryption and the private key is used for decryption.
- Signing and signature verification scenarios: The private key is used for signing and the public key is used for signature verification.

NOTE

Due to the privacy of a private key, you are advised to generate and keep it properly by yourself. Loss of the private key may cause website information leakage. If the private key is lost, revoke the certificate immediately and apply for a new SSL certificate for the domain name.

Working Principles of a Digital Certificate

A digital certificate uses the public key system which consists of a pair of matched keys to encrypt and decrypt data. Each user sets a specific private key that is known only to himself or herself and uses it for decryption and signature. At the same time, the user sets a public key and shares it with a group of other users for encryption and signature verification.

Because only the owner has the key, the owner can use it to generate a digital signature that no other users can generate.

A digital certificate is a file digitally signed by a CA and contains information about the owner of a public key and the public key. The simplest certificate

contains a public key, name, and digital signature of the CA. Another important feature of a digital certificate is that it is valid only within a specific period of time.

Creating a Private Key

CCM has the following requirements on the cryptographic algorithm and length of your private key:

- RSA
- At least 2048 bits

  NOTE

  The 2048-bit SHA256 digest algorithm is recommended.

You can use either of the following methods to create your private key:

- Using OpenSSL

  OpenSSL is a powerful and widely used security library tool. You can download the latest OpenSSL installation package from **http://www.openssl.org/source/**.

  NOTE

  The OpenSSL version must be 1.0.1g or later.

  After installing OpenSSL, run the **openssl genrsa -out** *myprivate.pem* **2048** command in the command-line interface (CLI).

  - *myprivate.pem* indicates your private key.
  - **2048** indicates the encryption length.
- Using Keytool

  Keytool is a key management tool coming with JDK. You can use it to create a KEYSTORE (JKS) certificate file. Obtain Keytool by downloading a JDK package from **http://www.oracle.com/technetwork/java/javase/downloads/index.html**.

  By default, the public key and private key created using Keytool cannot be exported. You need to export the private key from the created KEYSTORE file.

  In the exported file, the following part is the private key:

  ```
  -----BEGIN RSA PRIVATE KEY-----
  ......
  -----END RSA PRIVATE KEY-----
  ```

  or

  ```
  -----BEGIN PRIVATE KEY-----
  ......
  -----END PRIVATE KEY-----
  ```

NOTICE

No matter which method you use to generate a private key, you need to keep it properly because once it is lost or damaged the corresponding public key and digital certificate will be unusable.

4.2 Why Is a Non-Password-Protected Private Key Required?

When using your certificate, other services will require its private key from you. If the key is password-protected, the services will fail to use the certificate, which will cause certificate decryption failure and HTTPS failure. Therefore, you need to provide a private key that is not password protected.

When you generate a private key, remove its password protection before uploading the certificate.

How Do I Remove Password Protection for a Private Key?

You can run the following command using OpenSSL to remove password protection for a protected private key:

openssl rsa -in encryedprivate.key -out unencryed.key

encryedprivate.key indicates the private key with password protection. **unencryed.key** indicates the private key with password protection removed. The extension name can be **.key** or **.pem**.

How Do I Determine Whether a Private Key Is Password Protected?

Use the text editor to open a private key file. If the private key file is in the following format, then it is password protected:

- Password-protected private keys in PKCS#8 format
  ```
  -----BEGIN ENCRYPTED PRIVATE KEY-----
  .....BASE64 Private key content.....
  -----END ENCRYPTED PRIVATE KEY-----
  ```
- Password-protected private keys in OpenSSL ASN format
  ```
  -----BEGIN RSA PRIVATE KEY-----
  Proc-Type: 4,ENCRYPTED
  DEK-Info:DES-EDE3-CBC,4D5D1AF13367D726
  .....BASE64 Private key content.....
  -----END RSA PRIVATE KEY-----
  ```

NOTE

All keys generated using Keytool are protected by passwords. You can convert them into key files that are not password protected. For details, see **What Are Mainstream Formats of Digital Certificates?**

4.3 What Are Mainstream Formats of Digital Certificates?

Mainstream web service software uses a basic password library provided by OpenSSL or Java.

- Tomcat, WebLogic, and JBoss use the password library provided by Java. Java Keystore (JKS) certificate files are generated with the Keytool tool in the Java Development Kit (JDK) tool package.

- Apache and Nginx use the password library provided by OpenSSL to generate PEM, KEY, or CRT certificate files.
- IBM web service products, such as WebSphere and IBM HTTP Server (IHS), use the built-in iKeyman tool to generate KDB certificate files.
- The Internet Information Services (IIS) service of uses the built-in certificate library to generate PFX certificate files.

Checking the Format of a Certificate File

- You can determine whether a certificate file is text or binary based on its name extension:
  - A DER or CER file is binary and contains only the certificate information.
  - A CRT file can be either binary or text. Most CRT files are text and have the same function as DER or CER files.
  - A PEM file is text typically and contains a certificate or private key or both. If a PEM file contains only a private key, it is usually replaced by a KEY file.
  - A PFX or P12 file is binary. Containing both a certificate and a private key, it is password protected typically.
- You can also use Notepad to open the certificate file. If strings of digits and letters are displayed in the file, the certificate file is in text format.

  Examples:

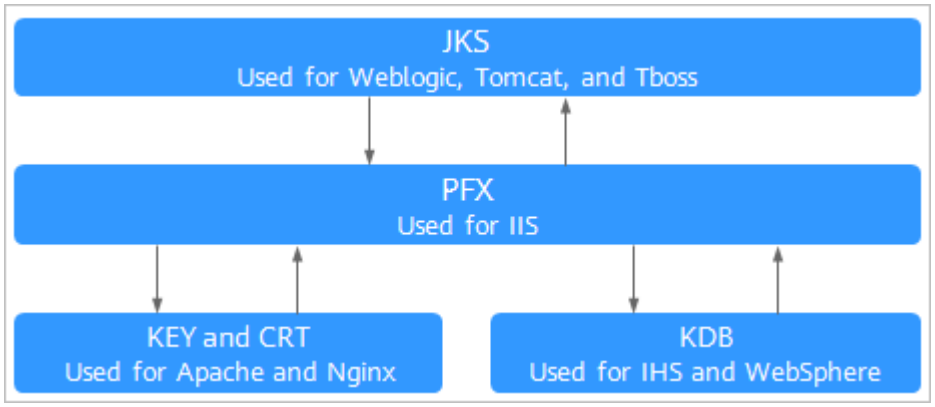
  ```
  —–BEGIN CERTIFICATE—–
  MIIE5zCCA8+gAwIBAgIQN+whYc2BgzAogau0dc3PtzANBgkqh.....
  —–END CERTIFICATE—–
  ```

  - If **--BEGIN CERTIFICATE--** is displayed, the file contains a certificate.
  - If **--BEGIN RSA PRIVATE KEY--** is displayed, the file contains a private key.

Certificate Format Conversion

Certificate formats as listed in **Figure 4-1** can be converted mutually.

Figure 4-1 Certificate Format Conversion



You can use the following methods to convert certificate formats:

- Converting from JKS into PFX

You can use the built-in Keytool of JDK to convert a JKS certificate file into PFX.

For example, you can run the following command to convert **server.jks** into **server.pfx**:

keytool -importkeystore -srckeystore D:\server.jks -destkeystore D:\server.pfx -srcstoretype JKS -deststoretype PKCS12

- Converting from PFX into JKS

  You can use the built-in Keytool of JDK to convert a PFX certificate file into JKS.

  For example, you can run the following command to convert **server.pfx** into **server.jks**:

  keytool -importkeystore -srckeystore D:\server.pfx -destkeystore D:\server.jks -srcstoretype PKCS12 -deststoretype JKS

- Converting from PEM/KEY/CRT into PFX

  You can use the **OpenSSL** tool to convert a KEY key file and CRT public key file into a PFX certificate file.

  For example, copy the **server.key** key file and **server.crt** public key file to the OpenSSL tool installation directory and run the following command to convert the certificate into the **server.pfx** certificate file:

  openssl pkcs12 -export -out server.pfx -inkey server.key -in server.crt

- Converting from PFX into PEM/KEY/CRT

  You can use the **OpenSSL** tool to convert a PFX certificate file into a PEM certificate file, KEY key file, and CRT public key file.

  For example, copy your PFX certificate file to the OpenSSL tool installation directory, and use the OpenSSL tool to run the following command to convert it into the **server.pem** certificate file, **server.key** key file, and **server.crt** public key file:

  openssl pkcs12 -in server.pfx -nodes -out server.pem

  openssl rsa -in server.pem -out server.key

  openssl x509 -in server.pem -out server.crt

NOTICE

This conversion method is used only for scenarios where OpenSSL is used to generate private keys and CSRs for applying for certificate files. Using this method, you can separate the private keys when you have obtained PEM public keys. When deploying a digital certificate, use the private key separated with this method to match the public key certificate issued to you.

4.4 How Do I Make a CSR File?

Before applying for a digital certificate, you must generate a private key and a certificate signing request (CSR). The CSR file is the source file for your public key certificate. It contains your server and company details and needs to be submitted to the CA for review.

NOTE

The **System generated CSR** option is recommended because manually generated CSRs often include errors.

A private key file will be generated when the CSR file is generated manually. Keep your private key stored safely.

The following describes how to generate a CSR file. You can select whichever method you prefer.

- **Generating a CSR File Using OpenSSL**

  If you need to enter Chinese characters, use Keytool to generate a CSR file.
- **Generating a CSR File Using Keytool**

NOTE

SCM has strict requirements on the key type and length of the CSR file. The key must be RSA and it must be 2,048 bits long.

Generating a CSR File Using OpenSSL

Step 1 Install the **OpenSSL** tool.

Step 2 Run the following command to generate a CSR file:

openssl req -new -nodes -sha256 -newkey rsa:2048 -keyout *myprivate.key* -out *mydomain.csr*

- **-new** specifies that a new CSR is generated.
- **-nodes** specifies that the private key file is not encrypted.
- **-sha256** specifies the digest algorithm.
- **-newkey rsa:2048** specifies the type and length of the private key.
- **-keyout** specifies that a private key file is generated. The file name can be customized.
- **-out** specifies that the name of the CSR file is generated. The name can be customized.

Step 3 Generate a CSR file named **mydomain.csr**.

Figure 4-2 Generating a CSR file

```
Generating a 2048 bit RSA private key
....+++
.....................................................................+++
writing new private key to 'myprivate.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [CN]:CN
State or Province Name (full name) []:ZheJiang
Locality Name (eg, city) [Default City]:HangZhou
Organization Name (eg, company) [Default Company Ltd]:HangZhou xxx Technologies,Inc.
Organizational Unit Name (eg, section) []:IT Dept.
Common Name (eg, your name or your server's hostname) []:www.example.com
Email Address []:

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

The information to be entered is as follows:

Field	Description	Example Value
Country Name	Two-letter code of the country where your company is located. For example, enter **CN** for China.	CN
State or Province Name	The name of the province or state where your company is located.	ZheJiang
Locality Name	The name of the city where your company is located.	HangZhou
Organization Name	The legal name of your company.	HangZhou xxx Technologies, Inc.
Organizational Unit Name	The department of your company that the applicant belongs to	IT Dept.
Common Name	The website domain name you are applying for a certificate for. **NOTE** • For a certificate with multiple domain names, enter the primary domain name to be associated with the certificate. • For a wildcard-domain certificate, enter the wildcard domain name. Example: *.example.com	www.example.com

Field	Description	Example Value
Email Address	Email of an applicant. The CSR file password does not need to be entered. Just press **Enter**.	-
A challenge password	CSR file password. The CSR file password does not need to be entered. Just press **Enter**.	-

NOTE

- Make sure that UTF8 encoding format is used for a Chinese character–based certificate with OpenSSL. In addition, enable the UTF8 support during OpenSSL compilation.
- SCM has strict requirements on the key type and length of the CSR file. The key must be RSA and it must be 2,048 bits long.

After you enter information as prompted, the **myprivate.key** (private key file) and **mydomain.csr** (CSR) files are generated in the current directory.

----End

Generating a CSR File Using Keytool

Step 1 Install Keytool, which is typically included in the Java Development Kit (JDK) tool package.

Step 2 Use Keytool to generate a Keystore certificate file.

NOTE

The Keystore file contains a key. For details about how to export the key, see **What Are Mainstream Formats of Digital Certificates?**

1. Run the following command to generate the **keystore** certificate file:

   keytool -genkey -alias mycert -keyalg RSA -keysize 2048 -keystore ./mydomain.jks

   - **-keyalg** specifies the key type, which must be **RSA**.
   - **-keysize** specifies the key length, which must be 2,048.
   - **-alias** specifies the certificate alias, which can be customized.
   - **-keystore** specifies the path for saving the certificate file. The certificate file name can be customized.

Figure 4-3 Generating the **keystore** certificate file

```
Enter keystore password:
Re-enter new password:
What is your first and last name?
  [Unknown]:  www.example.com
What is the name of your organizational unit?
  [Unknown]:  IT Dept.
What is the name of your organization?
  [Unknown]:  HangZhou xxx Technologies,Inc.
What is the name of your City or Locality?
  [Unknown]:  HangZhou
What is the name of your State or Province?
  [Unknown]:  ZheJiang
What is the two-letter country code for this unit?
  [Unknown]:  CN
Is CN=www.example.com, OU=IT Dept., O="HangZhou xxx Technologies,Inc.", L=HangZhou, ST=Zhe
Jiang, C=CN correct?
  [no]:  Y

Enter key password for <mycert>
        (RETURN if same as keystore password):
```

2. Enter the certificate password and enter information described in the following table:

Question	Description	Example Value
What is your first and last name?	Domain name for which you are applying for a certificate. **NOTE** – For a certificate with multiple domain names, enter the primary domain name to be associated with the certificate. – For a wildcard-domain certificate, enter the wildcard domain name. Example: *.example.com	www.example.com
What is the name of your organizational unit?	Name of the department that the applicant belongs to.	IT Dept
What is the name of your organization?	The name of the company to which the applicant belongs.	HangZhou xxx Technologies,Ltd
What is the name of your City or Locality?	The city where an applicant is located.	HangZhou
What is the name of your State or Province?	The state or province where an applicant is located.	ZheJiang

Question	Description	Example Value
What is the two-letter country code for this unit?	The country where the applicant belongs. Use a two-character ISO country code.	CN

After you enter the information, review the entered content for errors. If there are no errors, press **Y**.

3. Enter the key password as prompted. The password can be the same as the certificate password. If they are the same, press **Enter**.

Step 3 Use the certificate file to generate a CSR.

1. Run the following command to generate a CSR file:

   keytool -certreq -sigalg SHA256withRSA -alias mycert -keystore ./mydomain.jks -file ./mydomain.csr

   - **-sigalg** specifies the digest algorithm, which is **SHA256withRSA**.
   - **alias** specifies the alias, which must be the same as the certificate alias in the keystore file in **-alias**.
   - **-keystore** specifies the certificate file.
   - **-file** specify the CSR file. The file name can be customized.

2. Enter the certificate password as prompted to generate the **mydomain.csr** file.

----End

4.5 How Do I Apply an SSL Certificate to Other Services?

After an SSL certificate is issued or uploaded, it can be used in other services, such as WAF and ELB.

In SCM, you can deploy an SSL certificate to WAF, ELB in just a few clicks. If a certificate needs to be pushed to another cloud service, you need to download the certificate, upload the certificate to the corresponding service console, and deploy the certificate.

Constraints

- Before updating an SSL certificate for ELB, ensure that the following conditions are met:
  - You have configured the original certificate in ELB. This means the certificate that is being used for ELB and you want to update in SCM must have been configured in ELB at the very beginning. Then, you can quickly update it in SCM. For details, see "Managing Certificates" in *Elastic Load Balance User Guide*.
  - You can use SCM to update the certificate deployed on listeners in ELB. If you update an SSL certificate in SCM, the certificate content and private

keys are updated in ELB accordingly. ELB then updates the certificate content and private keys on all listeners where the certificate is deployed for.

- To update a certificate used for ELB in SCM, domain names must be associated with the certificate in ELB.
- If an ELB certificate is used for multiple domain names, ensure that the new certificate you want to update in SCM for ELB must match with those domain names. If they do not match, the domain names in the new certificate will overwrite the ones in the original certificate after the update.

  For example, the primary domain name and additional domain name of the new certificate are example01.com and example02.com, respectively, and the domain names associated with the original certificate in ELB are example01.com and example03.com. When you update the certificate in SCM, the domain names associated with the certificate in ELB are updated to example01.com and example02.com.

- Currently, you can use SCM to quickly deploy an SSL certificate to WAF in the **default** enterprise project only. For other enterprise projects, download the certificates first, upload them to WAF, and then deploy them in WAF.

Applying Certificates in SCM to WAF and ELB.

In SCM, you can deploy an SSL certificate to WAF, ELB in just a few clicks. With SSL certificates, data access through the cloud products is more secure.

For details, see .

Applying Certificates in Other Cloud Products

Alternatively, you can download the certificate to your local PC and then upload it to the management console of the specific cloud product and complete deployment.

4.6 Why Is a Message Displayed Indicating that the Certificate Chain Is Incomplete When I Configure HTTPS?

Perform the following operations to locate and rectify the fault:

Check whether the certificate chain is complete, whether the certificate is added in the format as required, whether all certificates are typed, and whether the certificate sequence is correct.

Ensure that the content of the certificate chain is pasted right below the content of the server certificate.

If the certificate chain is incomplete, complete the certificate chain by referring to **How Do I Fix an Incomplete SSL Certificate Chain?**

4.7 Issues Related to SSL Certificate Uploading

If you encounter problems related to certificate uploading, use a specific solution based on your situation.

Which Format Is Required of a Certificate to Be Uploaded to SCM?

Currently, only certificates in the PEM format can be uploaded to SCM.

Certificates in other formats can be uploaded only after being converted into those in the PEM format. For details, see **How Do I Convert a Certificate into the PEM Format?**

Is the Use of Certificate on the Original Platform Affected After Uploading?

No. Uploading a certificate does not affect the use of it on the original platform.

Certificate uploading can be regarded as copying a local certificate to our platform. The copy operation does not affect the use of the certificate.

Why Is a Message Indicating That the Website Is Insecure After the Certificate Is Uploaded to SCM?

After a certificate is uploaded, you need to deploy the certificate on the corresponding cloud product and complete required configuration.

In SCM, you can deploy an SSL certificate to WAF or ELB in just a few clicks. With SSL certificates, data access through the cloud products is more secure.

What Is a Public Key and a Private Key?

You can upload a certificate and private key. Ensure that the private key matches the certificate. For details about public and private keys, see **What Is a Public Key and a Private Key?**.

Why Is a Non-Password-Protected Private Key Required?

To use an SSL certificate for a cloud service, ensure that the private key is not password-protected. For details about why a non-password-protected private key is required, see **Why Is a Non-Password-Protected Private Key Required?**

4.8 Validity Periods of Private Certificates

How Long Is the Validity Period of a Private Certificate?

- Setting the Validity Period

  The validity period of a private certificate is set when it is applied for.

NOTE

A private certificate is issued by an activated private CA. Therefore, the validity period of a private certificate must be shorter than or equal to that of the private CA that issued it.

- Viewing the expiration time

  After the private certificate is obtained, you can log in to the management console and view the certificate expiration time on the private certificate list page.

How Do I Prevent Service Interruptions When My Private Certificate Is About to Expire?

To prevent service interruption caused by certificate expiration, perform the following steps:

Step 1 Apply for a certificate.

A private certificate cannot be renewed after it expires. You are advised to apply for a new certificate before it expires.

Step 2 Replace the expired certificate.

Before the old certificate expires, replace it with the newly issued certificate.

----End

4.9 How Is PCA in CCM Billed?

How Do I Stop the Billing for a Private CA or Certificate?

Private certificates support pay-per-use billing. To stop billing for a private certificate, revoke it.

CAUTION

- Disabled private CAs will also be billed.
- If you delete a private CA, it takes a few days for the deletion to take effect. It takes at least 7 days for a scheduled deletion to take effect (depending on the delay time you configured). During the scheduled deletion period, you will be billed in accordance with the following rules:
  - If you have not canceled the scheduled deletion and the private CA is deleted, the private CA is not billed for this period.
  - If you cancel the scheduled deletion but the private CA is not deleted during this period, the private CA is still billed for this period.

  For example, if you delete a private CA at 00:00 on January 1, 2022 and the private CA is deleted seven days later as scheduled, you will not be billed for the seven days. If you cancel the scheduled deletion at 00:00 on January 4, 2022 and the private CA is not deleted, you will still be billed for the CA for the period from 00:00 on January 1, 2022 to 00:00 on January 4, 2022.

4.10 Can I Discontinue a Private CA After It Issues A Private Certificate?

You can use either of the following methods to disable some functions of a private CA or discontinue a private CA:

- If you do not need to use a private CA to issue certificates but need to use it to revoke certificates or sign CRLs, you can disable the private CA. After a private CA is disabled, using of all certificates subordinated to the CA is not affected.
- If you no longer need a private CA, delete it. When a private CA is deleted, the billing stops. The exported certificates (not revoked) can still be used. However, all certificates subordinated to the private CA cannot be revoked, and the CRL cannot be updated. All private certificates issued by the private CA or its subordinate CAs cannot be exported.

4.11 How Do I Convert a Certificate into the PEM Format?

Certificate formats can be converted mutually.

It is recommended that **OpenSSL** be used to convert certificates in other formats into the **PEM** format. The following examples illustrate some popular conversion methods.

Converting the Certificate Format to PEM

Table 4-1 Certificate format conversion commands

Format	Conversion Method (Using OpenSSL)
CER/CRT	Rename the **cert.crt** certificate file to **cert.pem**.
PFX	• Obtain a private key. As an example, run the following command to convert **cert.pfx** into **key.pem**: **openssl pkcs12 -in cert.pfx -nocerts -out key.pem** • Obtain a certificate. As an example, run the following command to convert **cert.pfx** into **cert.pem**: **openssl pkcs12 -in cert.pfx -nokeys -out cert.pem**
P7B	1. Convert a certificate. As an example, run the following command to convert **cert.p7b** into **cert.cer**: **openssl pkcs7 -print_certs -in cert.p7b -out cert.cer** 2. Rename obtained certificate file **cert.cer** to **cert.pem**.

Format	Conversion Method (Using OpenSSL)
DER	• Obtain a private key. As an example, run the following command to convert **privatekey.der** into **privatekey.pem**: **openssl rsa -inform DER -outform PEM -in privatekey.der -out privatekey.pem** • Obtain a certificate. As an example, run the following command to convert **cert.cer** into **cert.pem**: **openssl x509 -inform der -in cert.cer -out cert.pem**

PKCS8 Certificate Encoding Format

As WAF and ELB do not support the PKCS8 format, an error will occur if you upload a certificate in PKCS8 format to SCM and then deploy it on WAF and ELB.

NOTE

- If the private key file of a certificate starts with **-----BEGIN PRIVATE KEY-----**, the certificate is in PKCS8 format.
- If the private key file of a certificate starts with **-----BEGIN RSA PRIVATE KEY-----**, the certificate is in PKCS1 format.

If your public or private key is in PKCS8 format, perform the following operations to use the PKCS8 certificate to WAF and ELB services:

Step 1 Check whether the certificate is in PEM format.

- If yes, go to **Step 2**.
- If no, convert the certificate format to PEM by referring to **Converting the Certificate Format to PEM** and then go to **2**.

Step 2 Run the following commands to convert format from PKCS8 to PKCS1:

- Converting the private key format from PKCS8 to PKCS1:

  openssl rsa -in pkcs8.pem -out pkcs1.pem
- Converting the public key format from PKCS8 into PKCS1:

  openssl rsa -pubin -in public.pem -RSAPublicKey_out

Step 3 Upload the converted certificate to SCM. For more details, see .

Step 4 Deploy the uploaded certificate on other product. For more details, see .

----End

4.12 How Do I Fix an Incomplete SSL Certificate Chain?

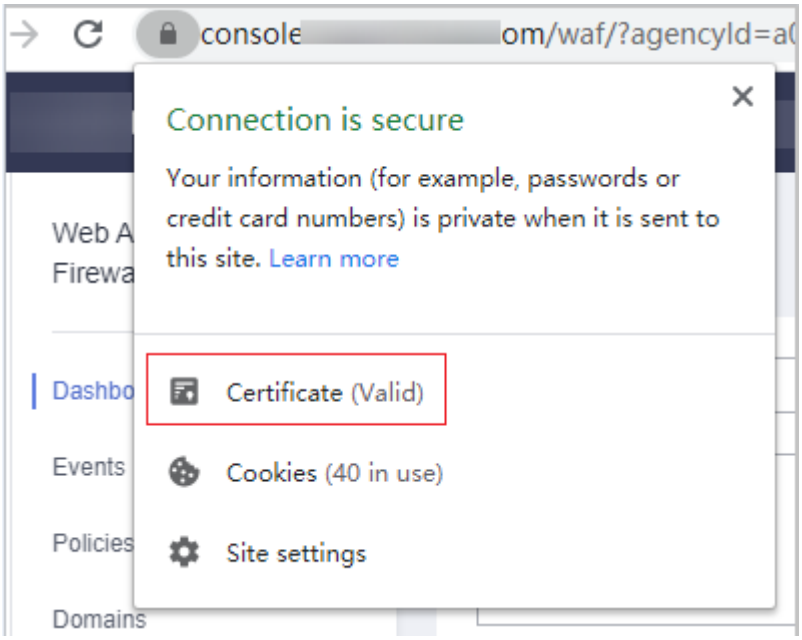
If the certificate provided by the certificate authority is not found in the built-in trust store on your platform and the certificate chain does not have a certificate authority, the certificate is incomplete. If you use the incomplete certificate to access the website corresponding to the protected domain name, the access will fail.

You can manually create a complete certificate chain to solve this problem. The latest Google Chrome version supports automatic verification of the trust chain. The following describes how to manually create a complete certificate chain:

Step 1 View the certificate.

Click the padlock in the address bar to view the certificate status, as shown in **Figure 4-4**.

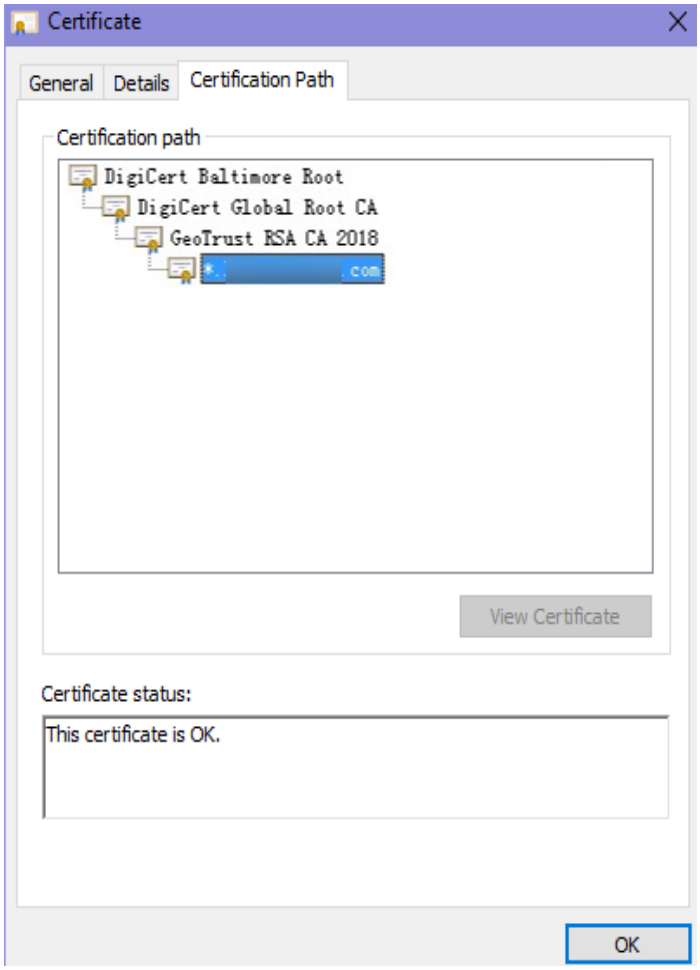
Figure 4-4 Viewing the certificate status



Step 2 Check the certificate chain.

Click **Certificate**. Select the **Certificate Path** tab and then click the certificate name to view the certificate status, as shown in **Figure 4-5**.

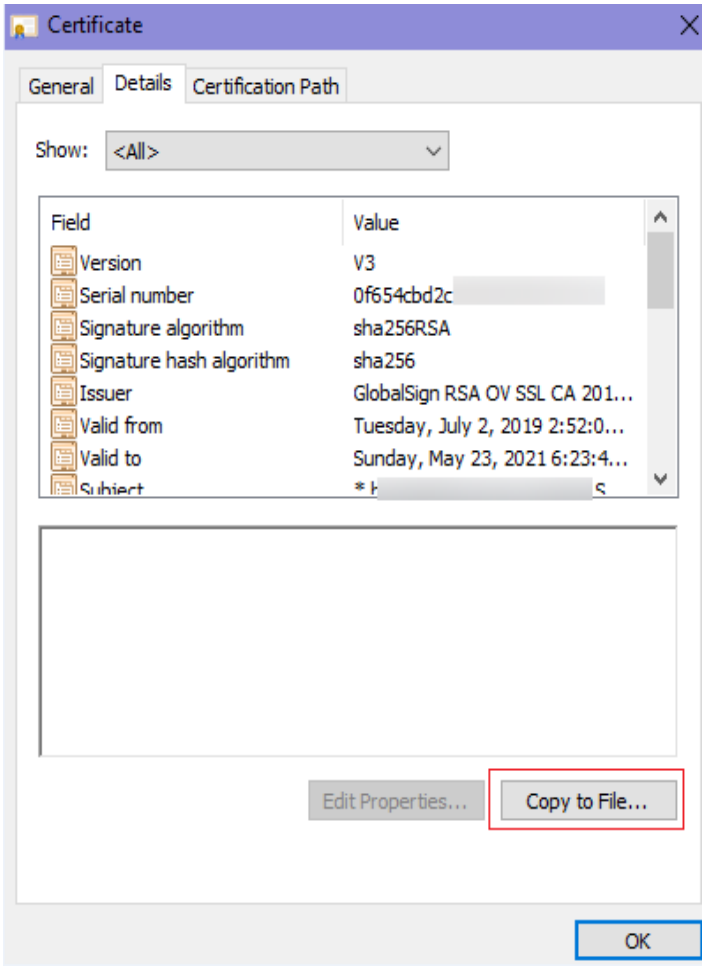
Figure 4-5 Viewing the certificate chain



Step 3 Save the certificates to the local PC one by one.

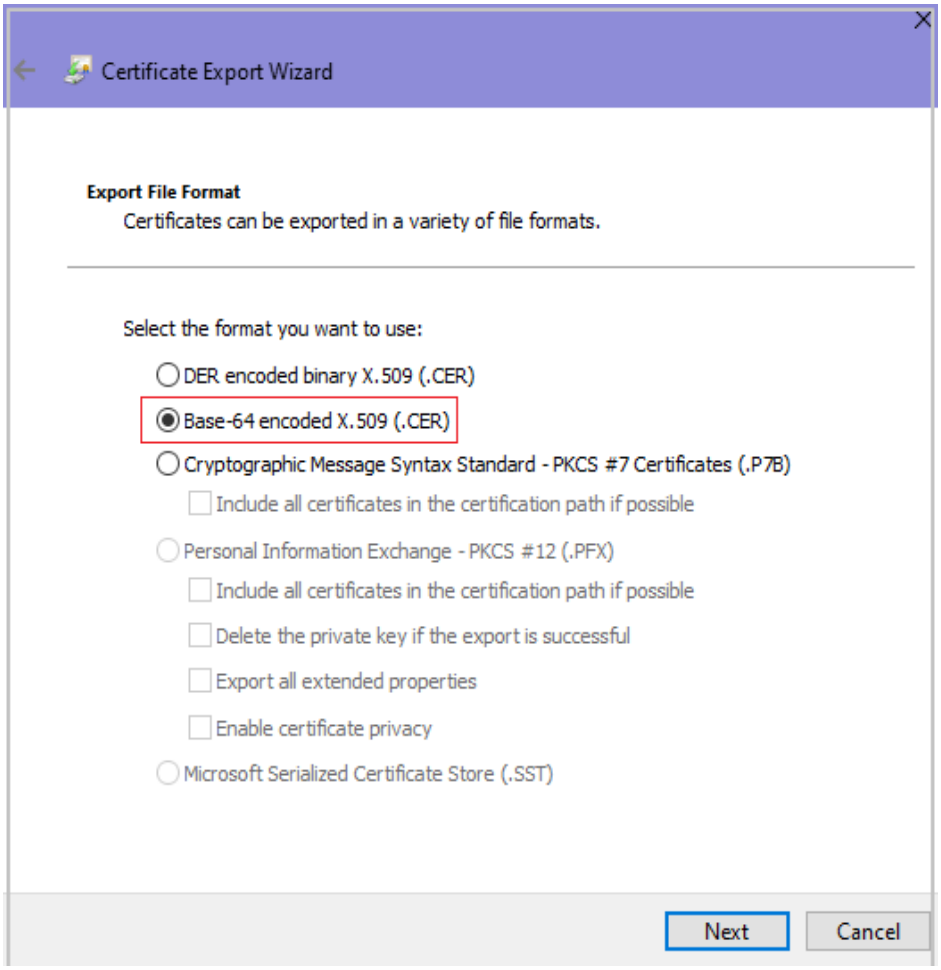
1. Select the certificate name and click the **Details** tab, as shown in .

Figure 4-6 Details



2. Click **Copy to File**, and then click **Next** as prompted.
3. Select **Base-64 encoded X.509 (.CER)** and click **Next**. **Figure 4-7** shows an example.

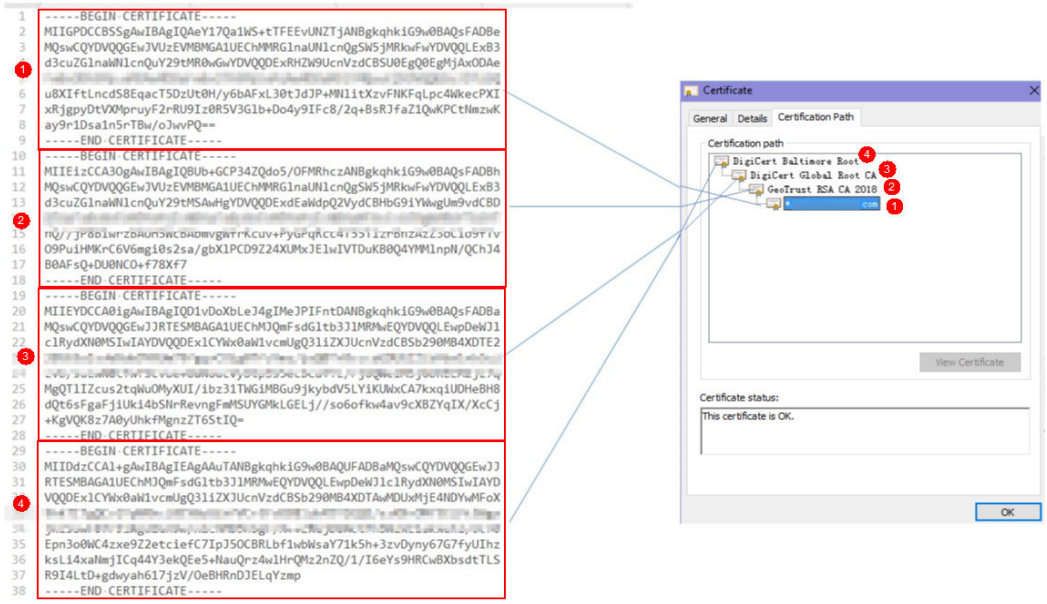
Figure 4-7 Certificate export wizard



Step 4 Rebuild the certificate.

After all certificates are exported to the local PC, open the certificate file in Notepad and rebuild the certificate according to the sequence shown in **Figure 4-8**.

Figure 4-8 Certificate rebuilding



Step 5 Upload the certificate again.

----End

A Change History

Release Date	Description
2025-10-30	This issue is the first official release.

