

# Request and Install a Global Certificate on the CSS

Document ID: 47782

## Contents

### Introduction

#### Prerequisites

- Requirements

- Components Used

- Conventions

#### Configure

- Configurations

#### Verify

#### Troubleshoot

#### Related Information

## Introduction

If you do not have pre-existing keys and certificates for the Content Services Switch (CSS), you can generate them on the CSS. The CSS includes a series of certificate and private key management utilities to simplify the process of generating private keys, Certificate Signing Requests (CSR), and self-signed temporary certificates. This document describes the process for obtaining a new certificate from a certificate authority (CA) and installing it to the CSS.

## Prerequisites

### Requirements

There are no specific requirements for this document.

### Components Used

This document is not restricted to specific software and hardware versions.

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

### Conventions

For more information on document conventions, see the Cisco Technical Tips Conventions.

## Configure

In this section, you are presented with the information to configure the features described in this document.

**Note:** In order to find additional information on the commands used in this document, use the Command Lookup Tool (registered customers only).

# Configurations

This document uses these configurations:

- Generate Rivest, Shamir, and Adelman (RSA) Key Pair
- Associate the RSA Key Pair File
- Generate CSR
- Obtain the Verisign Intermediate Certificate
- Import Chained Certificate File
- Associate the Certificate File
- Configure the SSL Proxy List
- Configure Secure Socket Layer (SSL) Service and Content Rules

## Generate Rivest, Shamir, and Adelman (RSA) Key Pair

Issue the **ssl genrsa** command to generate an RSA private/public key pair for asymmetric encryption. The CSS stores the generated RSA key pair as a file on the CSS. For example, to generate the RSA key pair `myrsakey.pem`, type the following:

```
CSS11500(config) # ssl genrsa myrsakey.pem 1024 passwd123
```

```
Please be patient this could take a few minutes
```

## Associating the RSA Key Pair File

Issue the **ssl associate rsakey** command to associate the RSA key pair name to the generated RSA key pair. For example, to associate the RSA key name `myrsakey1` to the generated RSA key pair file `myrsakey.pem`, type the following:

```
CSS11500(config) # ssl associate rsakey myrsakey1 myrsakey.pem
```

## Generate CSR

Issue the **ssl gencsr rsakey** command to generate a CSR file for an associated RSA key pair file. This CSR will be sent to the CA for signing. For example, to generate a CSR based on the RSA key pair `myrsakey1`, type the following:

```
CSS11503(config)# ssl gencsr myrsakey1
```

```
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.
For some fields there will be a default value,
If you enter '.', the field will be left blank.
Country Name (2 letter code) [US] US
State or Province (full name) [SomeState] CA
Locality Name (city) [SomeCity] San Jose
Organization Name (company name) [Acme Inc]Cisco Systems, Inc.
Organizational Unit Name (section) [Web Administration] Web Admin
Common Name (your domain name) [www.acme.com] www.cisco.com
Email address [webadmin@acme.com] webadmin@cisco.com
```

The **ssl gencsr** command generates the CSR and outputs it to the screen. Most major CAs have Web-based applications that require you to cut and paste the certificate request to the screen.

```
-----BEGIN CERTIFICATE REQUEST-----
MIIBWCCAQICAQAwwZwxCzAJBgNVBAYTAlVTMQswCQYDVQQLIEwJNQOTETMBEGA1UE
```



```
BgNVBAsTKlZlcm1TaWduIEludGVybmF0aW9uYWwgU2VydmVyIENBIC0gQ2xhc3Mg
MzFJMEcGA1UECmNAD3d3LnZlcm1zaWduLmNvbS9DUFMgSW5jb3JwLmJ5IFJlZi4g
TElBQk1MSVRZIEExURC4oYyk5NyBwZXJpU2lnbjCBnzANBGRkqhkig9w0BAQEFAAOB
jQAwYkCgYEA2IKA6NYZAn0fhRg5JaJlK+G/1AXTvOY2O6rwTGxhtueqPHNFVbLx
veqXQu2aNAoV1Klc9UA13dkHwTKydWzEyruj/1YncUOqY/UwPpMo5frxCTvzt010
OfdcSVq4wR3Tsr+cDCVQsv+K1GLWjw6+SJPkLICp1OcTzTnqwSye28CAwEAAaOB
4zCB4DAPBgNVHRMECDAGAQH/AgEAMEQGA1UdIAQ9MDSwOQYLYIZIAYb4RQEHAQEW
KjAoBggrBgEFBQcCARYCaHR0cHM6Ly93d3cudmVyaXNpZ24uY29tL0NQUSAOBgNV
HSUELTArBggrBgEFBQcDAQYIKwYBBQUHAWIGCWGSAGG+EIEAQYKYZIAYb4RQEI
ATALBgNVHQ8EBAMCAQYwEYQYJYZIAYb4QgEBBAQDAgEGMDEGA1UdHwQqMCgwJqAk
oCKGIGh0dHA6Ly9jcmwudmVyaXNpZ24uY29tL3BjYTMuY3JsMA0GCSqGSIb3DQEB
BQUAA4GBAAgB7ORo1ANC8XPxI6I63unx2sZUXCM+hurPaJozq+qcBBQHNgYL+Yhv
1RPuKSvD5HKNRO3RrCAJLeH24RkFOLA9D59/+J4C3IYChmFOJl9en5IeDCSk9dBw
E88mw0M9SR2egi5SX7w+xmYpAY5Okiy8RnUDGqzxz6dl+C2fvVF1a
-----END CERTIFICATE-----
```

## Import Chained Certificate File

Once the CSR has been signed by a CA, it is now called a Certificate. The Certificate file must be imported to the CSS. Issue the **copy ssl** command to facilitate the import or export of certificates and private keys from or to the CSS. The CSS stores all imported files in a secure location on the CSS. This command is available only in SuperUser mode. For example, to import the mychainedrsacert.pem certificate from a remote server to the CSS, type the following:

```
CSS11500# copy ssl sftp ssl_record import mychainedrsacert.pem PEM passwd123

Connecting
Completed successfully
```

## Associate the Certificate File

Issue the **ssl associate cert** command to associate a certificate name to the imported certificate. For example, to associate the certificate name mychainedrsacert1 to the imported certificate file mychainedrsacert.pem, type the following:

```
CSS11500(config)# ssl associate cert mychainedrsacert1 mychainedrsacert.pem
```

## Configure the SSL Proxy List

Issue the **ssl-proxy-list** command to create an SSL proxy list. An SSL proxy list is a group of related virtual or backend SSL servers that are associated with an SSL service. The SSL proxy list contains all the configuration information for each virtual SSL Server. This includes the SSL Server creation, certificates and corresponding SSL key pair, Virtual IP (VIP) address and port, SSL ciphers supported, and other SSL options. For example, to create the ssl-proxy-list ssl\_list1, type the following:

```
CSS11500(config)# ssl-proxy-list ssl_list1
Create ssl-list <ssl_list1>, [y/n]: y
```

Once you create an SSL proxy list, the CLI enters you into the ssl-proxy-list configuration mode. Configure your SSL server as shown below.

```
CSS11500(ssl-proxy-list[ssl_list1])# ssl-server 20
CSS11500(ssl-proxy-list[ssl_list1])# ssl-server 20 vip address 192.168.3.6
CSS11500(ssl-proxy-list[ssl_list1])# ssl-server 20 rsacert mychainedrsacert1
CSS11500(ssl-proxy-list[ssl_list1])# ssl-server 20 rsakey myrsakey1
CSS11500(ssl-proxy-list[ssl_list1])# ssl-server 20 cipher rsa-export-with-rc4-40-md5 192.1
CSS11500(ssl-proxy-list[ssl_list1])# active
```

## Configure Secure Socket Layer (SSL) Service and Content Rules

Once the SSL proxy list is activated, a service and content rule need to be configured to allow the CSS to send SSL traffic to the SSL module. This table provides an overview of the steps required to create an SSL service for a virtual SSL server, including adding the SSL proxy list to the service and creating an SSL content rule.

### Create an SSL service

```
CSS11500(config)# service ssl_serv1Create service <ssl_serv1>,
  [y/n]: y
CSS11500(config-service[ssl_serv1])# type ssl-accel
CSS11500(config-service[ssl_serv1])# slot 2
CSS11500(config-service[ssl_serv1])# keepalive type none
CSS11500(config-service[ssl_serv1])# add ssl-proxy-list ssl_list1
CSS11500(config-service[ssl_serv1])# active
```

### Create an SSL content rule

```
CSS11500(config)# owner ssl_owner
Create owner <ssl_owner>, [y/n]: y
CSS11500(config-owner[ssl_owner])# content ssl_rule1
Create content <ssl_rule1>, [y/n]: y
CSS11500(config-owner-content[ssl_rule1])# vip address 192.168.3.6
CSS11500(config-owner-content[ssl_rule1])# port 443
CSS11500(config-owner-content[ssl_rule1])# add service ssl_serv1
CSS11500(config-owner-content[ssl_rule1])# active
```

### Create a clear text content rule

```
CSS11500(config-owner[ssl_owner])# content decrypted_www
Create content <decrypted_www>, [y/n]: y
CSS11500(config-owner-content[decrypted_www])# vip address 192.168.11.2
CSS11500(config-owner-content[decrypted_www])# port 80
CSS11500(config-owner-content[decrypted_www])# add service linux_http
CSS11500(config-owner-content[decrypted_www])# add service win2k_http
CSS11500(config-owner-content[decrypted_www])# active
```

At this point, client HTTPS traffic can be sent to the CSS at 192.168.3.6:443. The CSS decrypts the HTTPS traffic, converting it to HTTP. The CSS then chooses a service and sends the HTTP traffic to a HTTP Web server. The following is a working CSS configuration using the examples above:

```
CSS11501# show run
configure

!***** GLOBAL *****
ssl associate rsakey myrsakey1 myrsakey.pem
ssl associate cert mychainedrsacert1 mychainedrsacert.pem

ip route 0.0.0.0 0.0.0.0 192.168.3.1 1

ftp-record conf 192.168.11.101 admin des-password 4f2bxansrcehjgka /tftpboot

!***** INTERFACE *****
interface 1/1
bridge vlan 10
description "Client Side"

interface 1/2
bridge vlan 20
description "Server Side"

!***** CIRCUIT *****
```

```

circuit VLAN10
description "Client Segment"

ip address 192.168.3.254 255.255.255.0

circuit VLAN20
description "Server Segment"

ip address 192.168.11.1 255.255.255.0

!***** SSL PROXY LIST *****
ssl-proxy-list ssl_list1
ssl-server 20
ssl-server 20 vip address 192.168.3.6
ssl-server 20 rsakey myrsakey1
ssl-server 20 rsacert mycertcert1
ssl-server 20 cipher rsa-with-rc4-128-md5 192.168.11.2 80
active

!***** SERVICE *****
service linux-http
ip address 192.168.11.101
port 80
active

service win2k-http
ip address 192.168.11.102
port 80
active

service ssl_serv1
type ssl-accel
slot 2
keepalive type none
add ssl-proxy-list ssl_list1
active

!***** OWNER *****
owner ssl_owner

content ssl_rule1
vip address 192.168.3.6
protocol tcp
port 443
add service ssl_serv1
active

content decrypted_www
vip address 192.168.11.2
add service linux-http
add service win2k-http
protocol tcp
port 80
active

```

## Verify

Use this section to confirm that your configuration works properly.

Use the **show ssl file** and **show ssl associate** commands to verify the configuration.

Verify that all files have a size larger than 0.

You can remove any certificate or key by using the **clear ssl file** command.

## Troubleshoot

Use this section to troubleshoot your configuration.

If SSL negotiation fails, use the **show ssl statistics** command to view useful information about the failed SSL negotiation.

For example, check these fields:

```
0 Unknown issuer certificates
0 Failed signatures decryptions
0 Invalid issuer keys
0 Not yet valid certificates
0 Expired Client certificates
0 Revoked certificates
0 CRLs not obtained from host
0 CRLs with bad HTTP return codes
0 CRLs not loaded because of low memory
0 CRLs obtained but failed to load
0 CRLs with invalid signatures
0 CRLs successfully loaded
0 Successful server authentications
0 Server authentications failed
0 Expired Server certificates
```

## Related Information

- [CSS 11500 Series Content Services Switches Hardware Support](#)
- [CSS 11000 Series Content Services Switches Hardware Support](#)
- [Cisco WebNS CSS11500 Software Download \( registered customers only\)](#)
- [Cisco WebNS CSS11000 Software Download \( registered customers only\)](#)
- [Technical Support & Documentation – Cisco Systems](#)

---

[Contacts & Feedback](#) | [Help](#) | [Site Map](#)

© 2010 – 2011 Cisco Systems, Inc. All rights reserved. [Terms & Conditions](#) | [Privacy Statement](#) | [Cookie Policy](#) | [Trademarks of Cisco Systems, Inc.](#)

---

Updated: Dec 21, 2005

Document ID: 47782

---